Table of Contents

Table of Contents 1

Preface: Copyrights, About the Manual, User support, How to reference HKL, History, Acknowledgments 3

Introduction: Advantages of using *Denzo* 5

Data collection process: Experimental Procedure, Detector 7

The Denzo Manual 11

Denzo for Dummys 13

A Denzo Session 16

Displaying the Image Using Xdisplay F: Visualisation of the diffraction space, Starting Denzo 17

Getting that First Orientation: Indexing: Autoindexing, Peak picking, When Autoindexing Fails 25

Refinement: Using the Fit Command: What to fit first, If things get worse, finishing 34
- Mosaicity and Spot Shape 37
  - Spot Shape and Size 37
  - Profile Fitting Radius 38
- How it works: Integration of the Diffraction Maxima 39
- Ice Rings 41
- Refinement Statistics 42
- Finishing Up 43
- Understanding the DENZO Log File 45
- Understanding the DENZO .x file 46

Input Formats and Conventions 48

Denzo Orientation Conventions 49
- Detector Conventions 49
- Camera Conventions 50
  - Gravity/beam system 50
  - Spindle/beam system 50
- Determining the values of FILM ROTATION and Y SCALE 51

Alphabetical Denzo Keyword Descriptions 52

Denzo Appendixes 77
- Keywords by Category 77
- Default Parameters 78
- Recommended Hardware 79
- Manual Indexing of Diffraction Images 80
  - Determining the spindle axis and vertical axis 80
  - Determining crystal orientation angles: CRYSTAL rotz roty rotx 81
- Film Cassette and Off-line Image Plate Users 83
- Detector Specifications 82

The SCALEPACK Manual 87

Program Description 88
- Statistics and Scalepack 89

Implementation: Limits on variable dimensions, Platforms **93**

Scenarios **94**
List of Scenarios 94
Scenario 1: Scale Data from One Crystal Processed with Denzo 95
Scenario 2: Scale Data from One Crystal not Processed with Denzo 97
Scenario 3: Heavy Atom Search: Scale native + a few frames of a potential derivative 98
Scenario 4: Scale Multiple Native data sets together. Data from Denzo 100
Scenario 5: Reindexing 102
Switch *a* and *c* in $P2_1$ or P2 102
Change order of *a, b, c* to *b, c, a* in orthorhombic 103
Convert *a, b, c* to *c, a, b* in orthorhombic 104
Reindex P23, switch *a, b* 105
P321 or P312. Reindex data and combine with another data set 106
p6, p4 or h3: Reindex data and combine with another data set 107
Reindex from H32 to R32 108
Reindex two H3 data sets, each with inconsistent index, and combine 109
Reindex four $P3_1$ data sets, each with inconsistent index, and combine 110
Scenario 6: Comparing two data sets from Scalepack 112
Scenario 7: Compare I+ and I− : detection of anomalous signal 113
Scenario 8: Fitting polarization 115
Scenario 9: High resolution + low resolution data sets from one crystal 116
Scenario 10: Thin or few frames (SCALE RESTRAIN and B RESTRAIN) 117
Scenario 11: Space Group Identification 118

Alphabetical Scalepack Keyword Descriptions **123**

HKL Q & A **143**
2Θ 144
Absorption Correction Factors 144
Auto editing 144
Autoindexing 145
Box definition 147
CCP4 Scaling 147
Computer problems and questions 148
Consequences of Profile fitting errors in Denzo 150
DENZO output 151
Display 153
Example of Bad Detector Pathology 153
Ignore 155
Mosaicity 156
Negative Intensivities 157
Other 157
Overload 158
Refinement 159
Rhombohedral 160
Twinned Crystals 161

Index **163**

Preface

The **HKL** suite is a package of programs intended for the analysis of X-ray diffraction data collected from single crystals, and consists of three programs: ***XdisplayF*** for visualization of the diffraction pattern, ***Denzo*** for data reduction and integration, and ***Scalepack*** for the merging and scaling of intensities obtained by ***Denzo*** or other programs.

Copyrights

Programs *Denzo* and *Scalepack* are © 1985-1994 by Zbyszek Otwinowski

The first version of *Denzo* and the first prototype of *Scalepack* were written by Zbyszek Otwinowski at the University of Chicago in 1983.

The program *XdisplayF* was developed by Wladek Minor, and is © 1991-1995 by the *Purdue Research Foundation*.

The HKL Manual, Edition 4, also known as **The *Denzo* Manual**, and **The *MacDenzo* Manual**, was written by Daniel Gewirth, with the cooperation of Zbyszek Otwinowski, and Wladek Minor and is © 1993-95 by Daniel Gewirth.

Edition 5 was revised and updated by Zbyszek Otwinowski and Wladek Majewski. The new typographic design was created by Grzegorz Jablonski.

About the Manual (from Edition 4)

"This manual began life as a simple compilation of some of the more useful things I (D.G.) needed to remember when indexing X-ray diffraction images with *Denzo*. This was especially important with earlier versions of the program, which lacked both the autoindexing option and a sophisticated graphical display package. For various reasons, Zbyszek convinced me to expand these initial efforts into a comprehensive manual, the fruits of which you see here. The emphasis of this manual is on how to run *Denzo* and *Scalepack* for typical applications. There is a keyword documentation section later in the manual, for those who wish to learn more about the features of the program.

Every effort has been made to make this manual as up-to-date and as accurate as possible. Since the program continues to undergo constant revision and development, however, it cannot be promised that the version of the *HKL* package you are using is exactly the same one described here. Fortunately, most of the new features of the *HKL* package do not preclude the older commands, so it is usually possible to continue to use the program without taking advantage of the new stuff.

As I originally conceived it, this manual was never meant to be a polished, boring, professional product, and occasionally you may notice that I use slang and make various insinuations which could, but really should not, be interpreted as insulting. Rather than taking offense, please read these few sections in the playful spirit in which they were written, and remember that the primary audience for whom this manual was written, and at whom the barbs are directed, is the author himself."

HKL user support

This edition incorporates much material in response to questions addressed to Zbyszek Otwinowski and Wladek Minor through the server hkl@tomis.bio.purdue.edu.

Questions about the programs described here should be directed to Zbyszek Otwinowski or Wladek Minor at the addresses below. You may also use `hkl@hkl-xray.com` or `news://bionet.xtallography` as a medium for *HKL* user's ideas exchange):

HKL mailing list:	`majordomo@hkl.hkl-xray.com`
Information and user support:	`hkl@hkl-xray.com`
HKL WWW homepage:	`http://www.hkl-xray.com`
Zbyszek Otwinowski:	`zbyszek@mix.swmed.edu`
Wladek Minor:	`wladek@iwonka.med.virginia.edu`

How to reference *Denzo* and *HKL*

If you want **to reference** *Denzo* for publication, use the following citation:

Z. Otwinowski and W. Minor, " Processing of X-ray Diffraction Data Collected in Oscillation Mode ", Methods in Enzymology, Volume **276**: Macromolecular Crystallography, part A, p.307-326, 1997,C.W. Carter, Jr. & R.M. Sweet, Eds., Academic Press

☛ The authors encourage referring to the use of the *HKL* programs when describing structural work. This is particularly significant when statistics from data reduction are being quoted, for example, resolution limit, signal-to-noise ratio, or R_{merge}. The *HKL* package, when properly used, gives a realistic estimate of measurement uncertainties. By default, the R_{merge} statistic in *Scalepack* do not reject observations. In contrast, most other programs define the R_{merge} statistic with a large number of observations deleted from R_{merge}. This difference in the definition of R_{merge} (a more detailed discussion is in the *Scalepack* program description) between programs frequently is very large. Quoting R_{merge} statistics without referring to the program / definition used does not properly describe the crystallographic experiment.

History

Until 1988 *Denzo* and *Scalepack* were developed at the University of Chicago, from 1989 through 1994 at Yale University, and currently at the University of Texas, Southwestern Medical Center at Dallas. *XdisplayF* was developed at Purdue University, and currently, at the University of Virginia.

Acknowledgments

We would like to thank Janet Smith, Anne Bloomer, Alan Friedman, John Tesmer, Iwona Minor, Bob Sweet for comments and contribution to programs and documentation, Michael Sintchak for the first UNIX port of programs Denzo and Scalepack, Paul Sigler for his encouragement in this project, and Michael Rossmann for providing the initial stimulus and ideas for the first implementation of the programs.

Introduction

The theory behind data reduction methods is complex enough that a series of EEC (European Economic Community) Cooperative Programming Workshops on Position-Sensitive Detector Software at LURE, Paris were dedicated to this task alone (phase I and II: May 16 - June 7, '86; phase III: November 12-19, '86). Proceedings from these workshops contain the best, but voluminous, presentation of the theory. The *HKL* suite implement most of ideas discussed there.

X-ray data can be collected with 0, 1 and 2-dimensional detectors, 0-d (single counter) being the simplest and 2-d the most efficient in terms of measuring diffracted X-rays in all directions. Two dimensional detectors have been used from the very beginning of X-ray diffraction studies, the year 1912. Initially the 2-d detector was made of X-ray sensitive film. At present, electronic and IP (**phospholuminescent**, best known by the trade name **Image Plate**) detectors dominate. To analyze single-crystal diffraction data collected with these detectors several computer programs have been developed. The 2-d detectors and related software are now used predominantly to measure and integrate diffraction from single crystals of biological macromolecules. However, the usefulness of these systems in small-molecule, high-resolution crystallography is just being recognized and much of the rest of this discussion is applicable to that field also.

The four most important recent developments in the data analysis of macromolecular diffraction measurements are autoindexing, profile fitting, transformation of data to a reciprocal-space coordinate system, and the demonstration that a single oscillation image contains all of the information necessary to derive the diffraction intensities from that image. The analysis and reduction of single crystal diffraction data consists of seven major steps. These are:

Denzo & XdisplayF

1. Visualization and preliminary analysis of the original, unprocessed, diffraction pattern.
2. Indexing of the diffraction pattern.
3. Refinement of the crystal and detector parameters.
4. Integration of the diffraction maxima.

Scalepack

5. Finding the relative scale factors between measurements.
6. Precise refinement of crystal parameters using the whole data set.
7. Merging and statistical analysis of the measurements related by space group symmetry.

The programs *Denzo, XdisplayF* and *Scalepack* together form the *HKL* package.

Every January in Daresbury Laboratory the CCP4 Study Weekend is held. Proceedings, distributed by The Librarian, Daresbury Laboratory, Daresbury, Warrington WA4 4AD, UK, contain comprehensive coverage of the X-ray diffraction data analysis, especially the volume "Data collection and Processing" (January 29-30, 1993) compiled by L. Sawyer, N. Isaacs, and S. Bailey.

The HKL package can estimate Bragg intensities from single-crystal diffraction data that are recorded on a position-sensitive X-ray (also potentially neutron-diffraction or electron-diffraction) detector, for example film, IP scanners, or charge-coupled device (CCD) area detectors. The programs allow for data collection by oscillation, Weissenberg, and precession methods. The detector can be either flat or cylindrical. The detector readout can be either a rectilinear or spiral, although spiral coordinates must be converted to rectilinear before processing. The package allows for random changes in the position and the sensitivity of the detector between consecutive exposures. The package features profile fitting, weighted refinement, eigenvalue filtering, and a universal definition of detector geometry.

Denzo is a program which takes raw X-ray diffraction data, for example from a scanned piece of film or an imaging plate, and reduces it to a file containing the h, k, l indexes and background and L-P corrected intensity of the spots on the image, along with an estimate of the error. In this way the useful information is extracted from the image. Depending on the number of spots on the image, a typical 7 to 18 Mb imaging plate raw data file (or, in case of the new RAXIS-IV scanner and 50 micron scan, the size of the image is 72 Mb!!) will yield a reduced file of approximately 0.1 to 2.0 Mb.

Advantages of using *Denzo*

to process raw X-ray diffraction images:

1. The use of a **flexible, weighted, profile fitting** algorithm for measuring the intensity of the spots provides high accuracy.
2. Crystal and detector parameters can be refined in any order and combination desired.
3. The initial **indexing procedure is automatic** and uses only a single still or oscillation image.
4. **Partially recorded reflections are used in the positional refinement** of the crystal orientations, leading to high accuracy and computational stability. In addition, *Denzo*, when combined with the companion program *Scalepack*, accurately tracks and sums up partially recorded reflections over an essentially unlimited number of frames. This provides the user with the option of collecting thin oscillation frames or extracting usable data from crystals of high mosaicity.

To "*Denzo*" a data set involves two separate steps. The first is to determine the precise crystal and detector orientation parameters for a single image in the collected series. The intrinsic crystal parameters include lattice type and unit cell dimensions. The crystal orientation parameters describe the orientation of the reciprocal lattice with respect to the spindle, beam, and vertical axes of the camera. The detector and X-ray parameters include the wavelength, the crystal to the detector distance, the precise coordinates of the direct beam, the detector missetting angles, and the internal scanner alignment parameters, like the non-orthogonality of the scanner head motions or the scale of the y pixels to the x pixels of the scanned image.

The second step in *Denzo*-ing a data set is to process all of the images in the series using the parameters determined from indexing one frame in the series. The *Denzo* user has full control over which parameters are refined in every cycle. The strategy in refinement, therefore, is to start at low resolution, adjusting just a few parameters, and then to extend the refinement to higher resolution, where more parameters can be fitted.

Once a data set is processed with *Denzo*, the output is ready to be scaled using the companion program, *Scalepack*. Together these programs make up a useful package for the reduction and merging of X-ray diffraction data.

Data collection process

The recent review of basic problems of practical crystallography is given in Jan Drenth "Principles of X-ray Protein Crystallography" Springer-Verlag 1994 (ISBN 0-387-94091-X)

Every element of the data collection process must function close to its optimum in order for one to solve a macromolecular structure. The sheer amount of the data collected makes computer programs an inevitable intermediary between the researcher and the experiment. The *HKL* package provides several levels of insight into the data at each stage of the measurement and data analysis process:

- *Scalepack*, which provides statistics for the full data set;
- *Denzo*, which provides numerical analysis of one oscillation image; and
- *XdisplayF*, which presents data visually, up to the single pixel level.

Different problems manifest themselves most clearly at different levels of analysis. The traditional method of judging the success of the experiment by the final statistics (e.g. from *Scalepack*) is not sufficient, since it does not show if the experiment was done optimally. The biggest problem with final statistics is that they do not differentiate well among the sources of the problems, and often come too late to fix them. Therefore, the experimenter must be aware of how the detector, the X-ray beam, the crystal, and the procedure all contribute to the final data quality and how each of them can make the experiment a failure.

Experimental Procedure

In the traditional approach, one collects data first and then starts analyzing the result. This strategy has a risk that there may be a gross inefficiency in the setup of the experiment, for example: the data set may be incomplete, the reflections may overlap, the zones may overlap, the large percentage of the reflections may be overloaded, etc. At that stage the only solution is to repeat the experiment, which may be difficult with unique crystals or with experiments that require a synchrotron source.

Data collection is best performed as a highly interactive process. Immediate data processing, which the authors encourage, provides fast feedback during data collection. Most macromolecular crystallographic projects go through iterative stages of improving crystal quality and data collection strategy. Typically, most of the data collection time and effort is spent before the optimal point is reached. Then, if data collection is going well, there is pressure to use the expensive detector and X-ray beam resource efficiently.

The three basic questions are:

1. whether to collect,
2. what to collect,
3. how to collect?

1. Are the data worth collecting?

Quick scaling of a partial data set collected in the first minutes may eliminate the need to collect full set of non-derivative data. Observing many diffraction spots in the image encourages one to collect a full data set; however a high number of spots may be due to high mosaicity, making such a data set unprocessable. One image is enough to index it, estimate mosaicity, and notice how severe is the problem with overlaps between the reflections. If the Bragg peaks are not resolved, there is no point in collecting such data, however many spots one may see in the image.

2. What range of data to collect?

Typically, one wants to collect up to the resolution limit, defined by the ratio of average intensity to σ (noise) being about 2. The safest way to establish this, is by processing a test image, rather than by guessing. One has to note that some space groups have inherent ambiguities in indexing, which only scaling of the initial image to the previously collected data can resolve. Otherwise, one risks recollecting already measured indexes rather than filling in the missing data.

3. How to collect data?

The detector should be placed as far away as possible to cover the desired resolution limit. Long unit cells, large mosaicity, or large oscillation range all affect spot separation and potential overlaps. Some overlaps are immediately visible - the ones arising from a long unit cell axis in the plane of the detector. At high resolution, due to the weakness of the spots, the overlaps may be less obvious. The simulation of a diffraction pattern, based on indexing of the first image and proposed data collection protocol, is the right tool to define an adequate oscillation range and the correct detector placement. There is no particular need to collect fully recorded reflections, so the optimal oscillation range is typically narrow, or even equal to a fraction of crystal mosaicity.

Detector

Problems are best diagnosed by collecting data with benchmark, high quality crystals, e.g. tetragonal lysozyme. **There is no particular advantage of lysozyme** crystals, with the possible exception of how easy is to grow them, and a larger unit cell crystal would be preferable, e.g. **tetragonal chymotrypsin**.

One should expect very high quality data from test crystals. The resulting anomalous-difference Fourier map should identify all the sulfurs in the protein. The detector parameters refinement should produce a **very small spread** (tens of microns, hundredths of a degree) from one image to another. Such a test may require the mounting of a test crystal in a way that avoids slippage and minimizes absorption. R_{merge} statistics in the range 2-3%, based on high redundancy (4 fold or higher) and high resolution (2Å or better) should be expected. Only very few (less than 0.1%) outliers should be found during merging.

Results worse than the above, indicate a problem with the test crystal or with the experimental setup. Preferably, the test crystal should be kept at 100K to minimize radiation damage. Problems with the test crystal may mask detector problems. For instance, slippage of the test crystal makes it very difficult to notice a spindle motor backlash or malfunctioning of the X-ray shutter.

Benchmarks and Calibration

Many macromolecular crystallography labs have not developed stringent benchmarks of acceptable performance. The value of such a test depends on how it is analyzed. Frequently, lack of rigor leads to the acceptance of many outliers in the test data. The **outliers rejection** in the merging of symmetry-related data is a valid statistical procedure, but it should be applied with great caution. The definition of an outlier is "a large, but sporadic fluctuation in the data", for example, due to a cosmic radiation hit. A small number - less than 0.01% - of outliers is something to be expected, even in a well functioning system. However, the practice of many labs has been to accept a much larger number of outliers, even as high as 10%. Many serious problems may be masked by such a liberal outlier rejection. It should be emphasized that outlier rejection always improves the consistency (including consistency indexes, e.g. R_{merge}), but not necessarily the correctness of the merged data.

Even if the detector marginally passes a standardized test, it may fail during data collection. For example, if the test data are collected in the large oscillation angle mode, a shutter opening delay or spindle motor backlash may affect fewer partials than if the data are collected in the narrow frame mode. **It is dangerous to accept results** from the test **with significant number of reflections flagged as outliers**, even if the R_{merge} statistics seem to be good. This is almost a sure sign of a **problem**, and unless the problem is well understood, it may not be sporadic when one collects data to solve a crystal structure. One way to attempt to understand the nature of outliers is to locate them in the detector space in order to identify the problem. The clustering of outliers in one area of the detector may indicate a damaged surface; if most outliers are partials, it may indicate a problem with spindle backlash or shutter control. The zoom mode may be used to display the area around the outliers in order to identify a source of problem, for example the existence of a satellite crystal or single pixel spikes due to electronic failure. Sometimes a histogram of the pixel intensities may suddenly stop below the maximum valid pixel value, indicating a saturation of the data acquisition hardware/software.

Detector geometry

A correct understanding of the detector geometry is essential for accurate positional refinement. Unfortunately**, most detectors deviate from perfect flat or cylindrical geometry**. These deviations are detector specific. The primary sources of error include:

- **misalignment** of the detector position sensors (Mar, R-axis),
- **non-planarity** of the film or IP during exposure or scanning,
- **inaccuracy of the wire placement** and distortions of the position readout in multi-wire proportional counters (MWPC),
- **optical distortion** (which can also be due to a magnetic field acting upon the image intensifier) in the TV or CCD based detectors.

☛ **If the detector distortion can be parametrized,** then these **parameters should be added to the refinement**. For example, in the case of the spiral scanners there are two parameters describing the end position of the scanning head. In the perfectly adjusted scanner these parameters would be zero. In practice, however, they may deviate from zero by as much as 1 mm. Such misalignment parameters can correlate very strongly with other detector and crystal parameters. If the program does not have the ability to describe detector distortions, then the other parameters such as the unit cell and crystal-to-detector distance will be systematically wrong.

With films and IPs handled manually in cassettes, as at many synchrotrons, the biggest problem lies in keeping the detector flat during exposure and subsequent scanning. In the manual systems, it is much harder to model the possible departures from ideal flat or cylindrical geometry, and *Denzo* makes limited attempts to correct such distortions. **Non-ideal film or IP geometry is one of the main factors behind the variable quality of data collected with the manual systems.**

X-ray beam

See Arndt U.W. "X-ray Collimation and Generation" in "Data Collection and Processing", Daresbury 93

The main properties of the X-ray beam that need to be checked are: stability, focus, angular spread, and wavelength in the case of MAD experiments. Large **fluctuations in beam intensity** show in a variable background intensity and variable scale factors during scaling. The quality of the **beam focus** is immediately visible in the spot profile of low resolution reflections. The **angular spread of the beam** contributes to the reflection width, and it may introduce overlaps between reflections for crystals with very long unit cells. The beam properties (except stability) are best analyzed by the inspection of images. The beam parameters are less significant for crystals with large mosaicity.

Summary

Macromolecular crystallography is an iterative process. Rarely do the first crystals provide all the necessary data to solve the biological problem being studied. Each step benefits from the experience learned in previous steps. To monitor progress, the *HKL* package provides two tools:

a) **Statistics** - both weighted (χ^2) and unweighted (R_{merge}) are provided. The Bayesian reasoning and multi component error model facilitates obtaining proper error estimates.

b) **Visualization** of the process plays two roles: it helps the operator to confirm that the process of data reduction, including the resulting statistics, is correct, and it allows one to evaluate problems for which there are no good statistical criteria. Visualization also provides confidence that the point of diminishing returns in data collection and reduction has been reached. At that point the effort should be directed to solving the structure.

Temple B., Harrison S. C., private communication.

The methods presented here have been applied to solve a large variety of problems, from inorganic molecules with 5Å unit cells to a rotavirus crystallized in the form with cell dimensions: 700x1000x1400Å. The overall quality of the method has been tested by many researchers by successful application of the program to MAD structure determinations.

THE *Denzo* MANUAL

A Description of the Program *Denzo* (version 1.5.x)

An Oscillation Data Processing Program for Macromolecular Crystallography

Denzo for Dummys

Yo. If you have a good crystal and just want to index your frames without knowing anything, follow this strategy. This is not the only way to process images with *Denzo*, nor it is necessarily the best, fastest, or most efficient, but it covers the basics. Sorry, but if your problem is more complicated or you have data from a non-automatic scanner, you may have to read the rest of this manual. Basically, you will be using as many defaults as possible. Hopefully someone has installed the programs on your computer and set up the proper aliases.

Step 1 see *Getting that First Orientation: Indexing* section	Load your frames onto a local disk (for fast access). Put all your files in the same directory. Make sure you have the four processing command files called `site.dat`, `myexperiment.dat`, `auto.dat`, and `refine.dat`. `Site.dat` should have been set up for your X-ray and detector system.		
Step 2	Sit down at your fast computer workstation (e.g. Alpha, Indigo, etc.). Open two windows (I call them ***ImageWindow*** and ***DenzoWindow***) and in each window set your directory to the directory where you are going to store the results from your processing.		
Step 3	Edit file `myexperiment.dat` to update the experimental parameters: distance, oscillation range and start, space group or lattice, sector range, mosaicity, and file names. If you don't know your space group or lattice, enter `P1` there. Edit file `refine.dat` to include the refinement resolution ranges.		
Step 4 see appendix: *Detector Specifications*	***ImageWindow:***	`rx xtal001.osc`	launches the display program for R-Axis frames (rx is an alias to the command 'xdisp raxis2' in versions higher than 1.6)
	DenzoWindow:	`denzo` "prompt."	launches Denzo; get a blank line
Step 5	***ImageWindow:***	Middle mouse button, hit Peak Sear. See tiny circles.	

	DenzoWindow:	`@site.dat`
		`@myexperiment.dat`
		`@auto.dat`
			begins autoindexing

Step 6 see *Getting that First Orientation: Indexing* section	***ImageWindow:***	Make sure green circles (corresponding to whole reflections) cover real reflections. Partial reflections are described by yellow circles. If circles don't cover real spots, sorry, gotta read more manual.	
	DenzoWindow:	Examine distortion table. Note highest symmetry lattice with low distortion. Hopefully this corresponds to the lattice type you entered in the file `myexperiment.dat`. If not, exit both windows, edit	

`myexperiment.dat` appropriately, and repeat from *Step 4* in the ***DenzoWindow***.

Step 7

ImageWindow: Hit [Zoom wind]. Select a region of your pattern to examine. Hit the **Int. Box** button. Now you are going to make sure your integration box parameters are correctly set. This exploits the advantage of *Denzo*: interactive analysis.
Examine the diffraction pattern to determine the resolution limits. The resolution at the position of the cursor is displayed in the green box (on the right side of the window). Let's say it's 2.2 Å.

DenzoWindow:

```
RESOLUTION LIMITS 100 2.2
FIT ALL
GO GO GO GO GO
```

ImageWindow: Examine the diffraction pattern. Are all the real reflections covered by predicted reflections ('preds')? If not, you will have to increase the mosaicity. If there are too many preds, you will have to decrease the mosaicity. Let's say you started with a mosaicity of 0.4 degrees and you need to decrease it a bit. Here's how to change it:

see *Mosaicity and Spot Shape* in *Refinement: Using the Fit Command* section

DenzoWindow:

```
MOSAICITY 0.3 GO
```

ImageWindow: Check again. Let's assume you've got it right now. Note it. If you have to go to unreasonable values of mosaicity (>> 1 degree), see *Mosaicity and Spot Shape* later in this manual.

Look at the integration boxes in the zoom window. Zoom in enough to see the spots. Are the preds big enough to cover the spot? Are they too big? Are the reflections so close together that the boxes overlap adjacent spots (lots of red circles)? If you have overlaps that a small adjustment of the spot or box size is not going to fix, then you will have to adjust your data collection, perhaps by changing the oscillation range or the crystal-to-film distance.

Let's say that your reflections are smaller than the inner white jagged outline, which is the *Denzo* spot area, and that a slight adjustment in the box size will prevent most of your overlaps. In the `site.dat` file the spot size was defined as a 0.75 x 0.75 mm ellipse and the box was 3.5 mm on a side. Change that now:

DenzoWindow:

```
SPOT ELLIPTICAL 0.5 0.5 0.0
BOX 2.5 2.5 GO GO GO
```

ImageWindow: Hit [Prof fit R]. The display will show the profile fitting radius as a white circle and put preds only on those spots with I/σ greater than the weak level (default value 5σ). If you don't have enough spots for profile fitting (say > 10), you may want to increase the profile fitting radius. Let's say you need to increase it slightly:

DenzoWindow:

```
PROFILE FITTING RADIUS 30 GO
```

Hopefully you get the idea of how to adjust your parameters.

Step 8

see *Mosaicity and Spot Shape* and *Refinement Statistics* in

DenzoWindow:

```
@refine.dat
```

begins sequential refinement of frames specified in the sector n to m section of `myexperiment.dat`. Depending on how fast your

Refinement: Using the Fit Command section see *If things get worse* in *Refinement: Using the Fit Command* section		computer is and how fast you can read, examine the screen output for the correct histogram shape (meaning you chose the mosaicity correctly, and for good χ^2 values and convergence). If your `site.dat` file was set up correctly, things should work.
	ImageWindow:	Watch refinement proceed. Little circles should line up on reflections. Cool, dude! If not, gotta read more manual.
Step 9	***DenzoWindow:***	After last frame, type STOP. This ends the program.
	ImageWindow:	Hit close button to kill image. You're done! Result is a bunch of `xtal###.x` files.
Step 10 see *Understanding the Denzo Log file* in *Refinement: Using the Fit Command* section	*Analysis:*	Scroll backwards through the ***DenzoWindow*** to examine the histogram and the final χ^2 values. If they are normal, you can proceed to *Scalepack*. If not, see the discussion in the following chapter. If this scenario didn't work and you entered the correct values in the `myexperiment.dat` file, it is most likely due to an error in the `site.dat` file. Candidates include X BEAM and Y BEAM.

☛ Notice that the session was run interactively, not as a batch job. This is so that you can watch the progress of the refinement in the ***ImageWindow***. Also, it is to encourage you to process your images as they are collected, so that you can spot problems before you have wasted lots of time. If the refinement is not working, you will be able to see very quickly that the predicted reflections (preds) do not line up with the real reflections and stop the job. If the data collection has problems, you can stop the experiment and fix your crystal, oscillation range, etc.. If the computer is reasonably fast, the data are on a local disk, and the number of reflections is average, you can probably process 90 frames in less than half an hour, which is not too bad. If you still want a log file to examine later once you catch your breath, try typing something like:

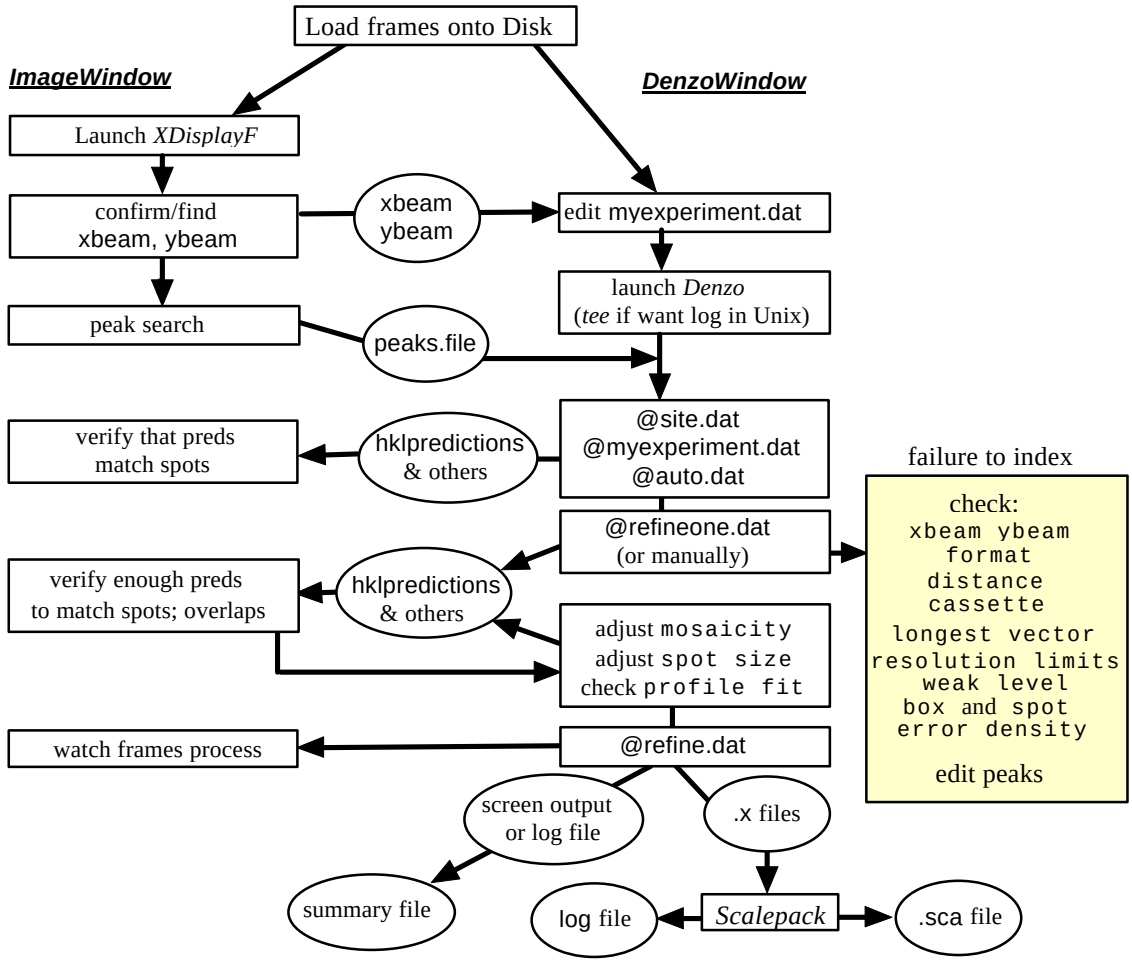
```
> denzo | tee denzo.log
```

(in UNIX only) when starting *Denzo*.

A Denzo Session

This chapter describes data processing in more detail than in "*Denzo* for Dummies".

Denzo session outline



1. Display the image and start *Denzo*.
2. Autoindex to deduce an orientation for the crystal, and refine the parameters for the image you have displayed.
3. The entire series of frames is processed, either in batch or interactively.
4. The log file of the batch job is then examined to see the results of the processing, or the .x files are displayed over the diffraction image, or the data are processed in *Scalepack*.

What follows is a description of each of these operations.

Displaying the Image Using XdisplayF

In the process which leads to *h*, *k*, *l*, F^2 and σ, the final statistics are affected by many factors. The program outputs a substantial amount of information to help identify the potentially sub-optimal steps in the data collection and reduction. Interactions between factors like detector geometry, crystal symmetry, radiation damage, detector failures, non-ideal crystals, etc. are far too complex to be formalized in a method that could be implemented strictly in a computer algorithm.

XdisplayF allows the crystallographer to visualize the data in their original form as well as to observe the progress of data reduction. Displaying the raw data takes advantage of the human ability to grasp the significance of complex patterns that would otherwise be hard to analyze numerically.

Visualization of the data collection and reduction process allows efficient recognition of what is correct or wrong by taking advantage of the brain's ability to recognize complex visual patterns. Problems like detector vibration may produce very striking patterns on the diffraction image, yet the deleterious effect on the final R_{merge} statistics may not be so obvious.

The optimal experiment has to take into account the interaction between all of the following: the experimental point spread function, the unit cell length, the detector position, the angular range of data collection, the mosaicity, and the definition of the integration box. The ability to visualize the influence of different data acquisition parameters allows the experimenter to understand his particular experimental limitations and how to overcome them.

Using XdisplayF

The display package for *Denzo* (developed by Wladek Minor) is based on the X-window interface. The *XdisplayF* runs concurrently with a *Denzo* session either in a separate window or on a separate graphics terminal. Due to the fact that both *Denzo* and *XdisplayF* store the entire image in memory, the host computer must have approximately (depending on the exact size of the image) 15 to 25 Mb of RAM in excess of that needed for the operating system. It also needs a substantial amount (e.g. 300 Mb) of swap space. Note that if you are indexing 72 Mb Raxis-IV images you will need a whopping 144+ Mb of memory.

In order to run *XdisplayF*, open a window on a graphics terminal ***(ImageWindow)*** and launch the program. Launching the program involves the following:

- (XdisplayF, version 1.5) calling the program itself,
- telling the program the name of the image file to be displayed

Here's an example:

```
dip2000 xtal001.osc [hklpredictions]          (VMS)
dip2000 xtal001.osc [hklpredictions]          (UNIX)
```

In this example, `dip2000` is an alias to INST_DIP2000 (UNIX) or a logical (VMS) for a statement like `$dip2000:==$disk:[wladek.dist.vms]dip2000.exe` that calls the version of *XdisplayF* that displays DIP2000 frames. (The reason you cannot simply "run" the executable for *XdisplayF* in VMS is because the run statement does not accept arguments

like file names, etc.) The name of the frame to be displayed is `xtal001.osc`. `hklpredictions` is a name of the file which *Denzo* will write which will contain the predicted reflections. It is now a default, so technically you can omit this, but I put it there to show you where you would put any other file you wanted *XdisplayF* to display.

☛ Example:

```
INST_DIP2000 xtal001.osc xtal001.x
```

`xtal001.x` represents the name of the file which contains list of reflections in *Denzo* output format. You need to click the **Update Pred** option in the ***ImageWindow*** for predictions to appear on the screen.

☛ The display program can display frames from most IP or CCD detectors. The executable files are named `INST_`*`Scanner-type`* `(UNIX) or SCANNER-TYPE.EXE (VMS)`.

The communication between *Denzo* and *XdisplayF* is using files `hklpredictions` and `.test`. This explains why the programs must be started in the same directory. The program version number is stored in these files, so you should delete them when switching to a new version of the programs.

- (XdisplayF, version 1.6) calling the program itself
- calling the format variable (the same as the modifier of the keyword **`FORMAT`** in *Denzo*)
- telling the program the name of the image file to be displayed

Here's an example:

```
mar18 xtal001.osc [hklpredictions]                (VMS)
xdisp dip 2000 xtal001.osc [hklpredictions]       (UNIX)
```

(For VMS version you need to put into your login.com file line:
$ mar18 :== "$disk2:[user.programs.hkl]xdisp.exe mar 18cm")

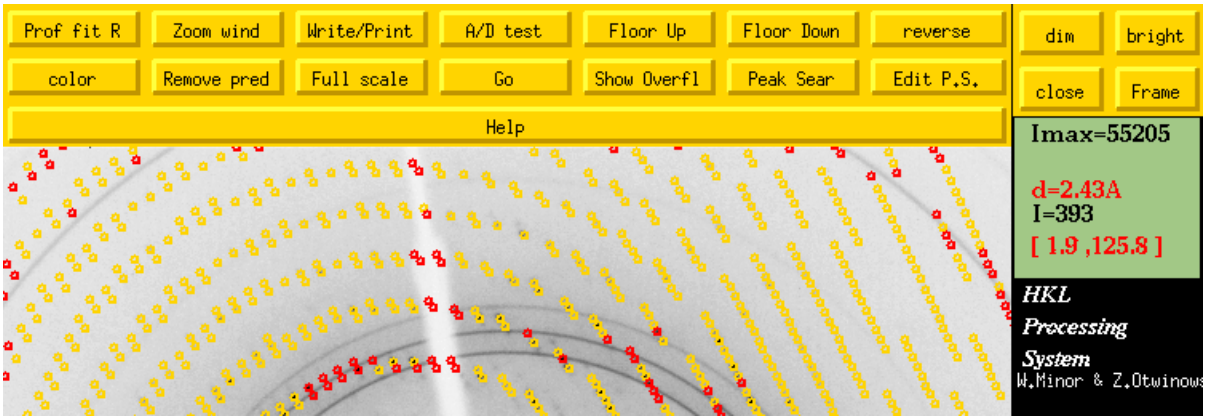
☛ You may get a list of all supported formats by typing “xdisp’. Formats are also listed in appendix *Detector Specifications*.

Lets start the XdisplayF.

The frame should now be displayed on the screen.

For the most part, the features of the display program are self-explanatory. The display is changed by clicking middle mouse button (or both buttons for two-button device) once on any of the buttons above the image. It is possible to zoom in and out, to change the color and intensity of the image, and to determine the resolution and x-y coordinates (in pixels and mm) of any point on the image. To exit from the display program click once on close. What follows is a more detailed description of the display.

Main Display Window



Toggles on the profile fitting radius. You must have indexed the image already for this to be of any use, because the preds shown in the display are those which are stronger than the weak level, and this is determined by running *Denzo*. The profile fitting radius is changed by entering it in the ***DenzoWindow***. See PROFILE FITTING RADIUS in the keyword documentation.

Toggles between black & white and color. Mono is often combined with reverse imaging to get black spots on a white background.

Launches the zoom window. The window will prompt you to use the middle mouse button to select the region you want to magnify. Any time you click the middle mouse button in either the regular or the zoom window, that point becomes the new center of the zoom display.

Toggles between updating and removing predictions. Once update predictions is turned on, new predictions from each *Denzo* cycle will be updated automatically in the display.

This tells the program to write out a file which contains the current image in the .xwd format. You may use a local utility to convert this file into a postscript file.

Toggles this function on and off. To use, click on

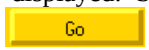
, then use the right mouse button to bracket a region of the diffraction pattern. A pretty black and red window will appear with a plot of the number of times a pixel has a certain intensity, as a function of intensity (x axis). The plot (but not the window) can be expanded or contracted by clicking on the arrow buttons. This function is useful for diagnosing hardware problems. Detectors with problems will have funny dips in what would otherwise be a smoothly varying curve. To exit from the A/D test, click on the

again.

Changes the right side scale bar so you can get an idea of where your image is on the full dynamic range of the detector.

Changes the lower level for the display range. Basically causes the image pixels with intensities below the "floor" to be displayed at the lowest color/darkness level, so in effect it narrows the displayed dynamic range, but increases the contrast.

Used in conjunction with the

. Starts a "movie" by sequentially incrementing or decrementing the last digit of the file name being displayed. So if your file is called `xtal001.osc`, and **frame up** is selected, hitting 

will tell the display to sequentially display `xtal002.osc`, then `xtal003.osc`, etc. until you stop it or run out of frames. This is a neat way of screening through your frames and spotting any that have

obvious problems. Of course, it works better on a fast computer. You can do the same thing manually with the Frame (below).

Floor Down — Opposite of Floor Up. Increases the dynamic range being displayed by lowering the floor.

Show Overfl — Places little red dots on saturated image pixels.

reverse — Inverts the dark/light display. Changes white to black, and black to white, and all others in between. Especially useful for black and white displays, so that the spots appear black and the background appears white.

Peak Sear — Starts the peak search routine and results in the creation of a file called `peaks.file` The number of spots Peak Sear picks can be controlled with options of the menu, which will appear after peak search finishes. The picked peaks are displayed as little black circles over the image. This set of peaks is the starting point for the autoindexing routine in *Denzo*.

dim — Increases the display dynamic range, which has the effect of emphasizing the stronger reflections with a concomitant loss of the weaker ones. Could also be called **Top up** to make it analogous with the Floor Down button.

bright — Opposite of dim. Decreases the display dynamic range, which has the effect of emphasizing the weaker pixels. The dark pixels will just stay dark. This is analogous to the Floor Up, but at the other end of the spectrum.

close — Quit and exit the display program. This will not interfere with any *Denzo* jobs running in a separate window as long as you don't log off the computer.

Frame — Tells the display which direction to increment the last digit of the image file when displaying sequential images. See the description above for Go. The right mouse button causes the file number to increment, and the left mouse button cause the file number to decrement. The middle mouse button is analogous to Go except that one click results in only a one frame advance (or decrease).

Help

Help — Has some help menus. Not very helpful yet. To close click Help again.

Buttons in the Zoom Window

Right mouse button. This displays a set of cross sections through the zoom window so that you can see the 2-D profiles of the pixels.

Mid mouse button. Re-centers the zoom display on the selected point. Works in either the zoom window or the full image window.

Left mouse button. Click and hold displays the information about a particular point in the zoom window in the green box in the lower and upper left corners of the zoom window. If the frame has been indexed already by *Denzo* then when you click on a spot the h, k, l indexes of that spot will be displayed. Otherwise, the display only shows the position of the spot and its intensity.

Zoom in — Increases the magnification of the zoom window. When you get to really high magnification, the intensity of each pixel is written in that pixel. However, this is not always accurate, and will be discussed under the Green Box window section below.

Zoom out — Decreases the magnification of the zoom window.

Int. box — This is a toggle through three different displays. The first (default) display shows a circle centered on the predicted reflection. This circle is always 6 pixels in diameter and has nothing to do with the size of the spot. This is the display you see if you don't hit the Int. box. The first time you hit the Int. box, you will see the integration box (the BOX of *Denzo*) plus the "circles" which define the spot and the guard regions, both in white. Note that you have to have indexed the frame in *Denzo* for this display to be useful, since the integration box and the predicted reflections are determined in *Denzo*, not *XdisplayF*. Even though you entered the spot, guard, and box sizes in mm in *Denzo*, their actual sizes must be an integral number of pixels, so you are not going to see perfect circles. Also, the spot must have numerical parity with the box, i.e. the background will always be symmetrical around the spot, so if the box has an odd number of pixels in a direction, then the spot will as well, even if it is supposed to be circular. The next time you hit the Int. box, the box is removed and the spot outline is colored red. This makes it easy to see things when spots are close together. When you hit the Int. box once more, the original display is restored.

Diff Vec — When this button is clicked, a second, black circle is displayed which is centered on the actual observed reflection. The green or yellow circles (representing whole or partial reflections) are centered on the predicted positions of the reflections. If the indexing and refinement have gone well, the two circles should be very close to concentric. Notice that most of the time when they are not, they differ by only a little bit. When they differ by a lot, you will see a little line drawn which connects the centers of the observed and predicted reflection positions. These lines are easy to spot in a low zoom window, so you can easily examine any problem reflections.

Zoom close — Closes the ***ZoomWindow***.

Imax=55205

d=2.43A
I=393
[1.9 ,125.8]

Green Box on the right side of the window

Imax=55205 — The maximum pixel intensity in an image.

d=2.43A — Resolution of the cursor position, in Å. Note that this is only accurate once *Denzo* has been launched and has read in the distance and wavelength and communicated these to the display program. This means that at least one GO statement has been given in *Denzo*. Otherwise, it is wrong. Note also that it is only accurate for flat detectors which are perpendicular to the beam.

I=393 — Intensity at the cursor position.

Historical note ☛ In a previous version, for some detectors, the value of I displayed was accurate for intensities up to 32,768 only. Intensities above 2^{15} are written by some scanners in a special manner: the

R-Axis divides strong pixels by 8 before writing them to the image file, Mar scanners create an overflow table, which must be consulted to find the correct value. Since the listing of the intensity of strongly exposed pixels was not the highest priority for the display program, these values were left uncorrected. But rest assured, **all *Denzo* versions deal accurately with strong pixel values** and saturation levels.

[1.9 ,125.8] Red numbers giving coordinates of the cursor in mm, using the *Denzo* coordinate frame. The first number is the x coordinate, the second is the y coordinate. These values can be entered into *Denzo* command files directly when needed, for example, to give the `X BEAM` and `Y BEAM` position.

Visualisation of the diffraction space

A diffraction data set forms an **image of three-dimensional reciprocal space**. 3-d image consists of a series of two-dimensional diffraction images, each of them representing a different, curved, slice of reciprocal space. In order to integrate the diffraction maxima accurately, they must appear as separated (non-overlapping) spots in the individual 2-d images. Unless the data are collected by the precession method, the diffracted image contains a **distorted view** of reciprocal space. This distortion of the image is a function of the data collection method, the diffraction geometry, and the characteristics of the detector. For the data reduction to be successful, the distortion of reciprocal space as viewed by the detector has to be accounted for correctly by the program. The distortion of the image of reciprocal space can vary even between images collected on the same detector. This is because the position of the detector, the X-ray wavelength, the oscillation range, pixel size, scanner gain, and the exposure level all affect the detector representation of diffraction space.

One should start data collection and reduction with a careful inspection of the data in their raw (original) form. The zoom option of the program *XdisplayF* allows one to examine reflections in pixel-by-pixel detail to check that the diffraction maxima are resolved. Because the program displays the resolution (in Ångstroms) corresponding to the position of the mouse-driven cursor, the diffraction limit of the crystals can be estimated even without data reduction. The display in high zoom mode provides digital pixel values, so one can check, among other things, that the exposure level is appropriate.

If problems exist with the detector or other components of the data-collection system, the display option helps to discover these before all the data are recorded. The examination of the image may reveal if there are extraneous sources of X-ray background. There are other statistics that can be provided instantly by *XdisplayF* which may indicate for example A/D converter malfunction. If there are many diffraction maxima in the image that form a characteristic pattern of diffraction from a single crystal, then the next step is deducing a crystal lattice that accounts for such a pattern. This step is called **indexing.**

Starting Denzo

Denzo is run concurrently with the display program from another window (let's call it ***DenzoWindow***) on a multi-window workstation. To run *Denzo*, type something like this:

```
$DENZO        (VMS ALIAS FOR $DENZO:==$DISK:[WLADEK.DISTR.VMS]DENZO.EXE)

>/usr/users/wladek/HKL/denzo                          (UNIX)
```

Make sure you are in the same directory from which you launched *XdisplayF*, because the two programs share information through the `hklpredictions` and `peaks.file` files (and other X-window data structures called 'atoms'). The program has no prompt, so when you get a blank line, the program is ready to accept your input, be it commands or a macro file. Since most interactive *Denzo* sessions start out with trying to index a particular diffraction pattern, that's what I'll discuss next.

Getting that First Orientation: Indexing

In order to assign an index properly to each spot on a diffraction image, the orientation of the crystal in space must be known. Historically this chore was performed by adjusting the arcs of the goniometer on which the crystal was mounted until test, or 'setting' photographs showed that the crystal was aligned. Usually this meant that one of the unit cell axes was perfectly aligned with the spindle axis. It was then a relatively straightforward matter to index the spots on an oscillation photograph, knowing the unit cell parameters, the oscillation range, and crystal alignment. Modern data collection strategies, however, typically involve the collection of oscillation photographs from crystals where the orientation of the reciprocal lattice relative to the X-ray beam and the film is not known ahead of time. The chore of indexing has thus become one of deducing the orientation from the collected oscillation photographs, rather than adjusting the position of the crystal before data collection begins.

There are **five parameters** that specify the orientation of the crystal relative to the X-ray beam:

- `VERTICAL AXIS`
- `SPINDLE AXIS`, and
- `CRYSTAL` `rotz`, `rotx`, and `roty` values.

These are what we wish to deduce in our *Denzo* session. Note that our primary goal is to deduce the orientation of the crystal relative to the X-ray beam, even though the orientation of the detector relative to the beam is of equal importance. The reason we focus our attention on the crystal orientation is simply because the detector, or "site" parameters, like `DISTANCE`, 2θ, and the position of the direct beam at the detector are (and should be!) reasonably well known before processing begins. Thus they can be refined automatically by the program.

Indexing

The *HKL* package offers two indexing methods: automatic and interactive. The **automatic method**, applicable in most cases, is fast and simple. The first step in the automatic method is the peak search, which chooses the spots to be used by the autoindexing subroutine. Ideally, the peaks should come from a diffraction by a single crystal. The *Denzo* program accepts peaks for autoindexing only from a single oscillation image. It is important that the oscillation range be small enough (it even can be zero, i.e. a still) so that the **lunes** (rings of spots from one reciprocal plane) are resolved.

Lune separation ☛ The requirement of lune separation is distinct from the requirement of spot separation. If the lunes overlap, spots may have more than one index consistent with a particular position on the detector. On the other hand, the oscillation range should be large enough to have a sufficient number of spots for the program to be able to establish the periodicity of the diffraction pattern. This may require at least 0.5 degree oscillation for a small unit cell protein crystal and 2-3 degree oscillation in the case of organic small molecule crystals.

The second step in the autoindexing is the **mapping of the diffraction maxima** identified by the peak search onto reciprocal space. Because the precise angles at which reflections diffract are *a priori* unknown for oscillation data, the center of the oscillation range is used as the best estimate of the angle at which the diffraction occurs.

The **autoindexing** in *Denzo* is based on a novel algorithm: a complete search of all possible indices of all reflections, that are found by a peak search or are selected manually. When the program finds values (integer numbers) of one index (for example, **h**) for all reflections, this is equivalent to having found one real-space direction of the crystal axis (in this case, **a**). For this reason such indexing is called **real-space indexing**. Finding one real-space vector is logically equivalent to finding the periodicity of the reciprocal lattice in the direction of this vector. The search for real-space vectors is performed by a Fast Fourier Transform (FFT) and takes advantage of the fact that finding all values of one index (e.g. **h**) for all reflections is independent of finding all values of another index (e.g. **k**). The *Denzo* implementation of this method is not dependent on prior knowledge of the crystal unit cell, however for efficiency reasons, the search is restricted to a reasonable range of unit cell lengths, obtained by default from the requirement of spot separation.

After the search for real-space vectors is completed, the program finds the three linearly independent vectors, with minimal determinant (unit cell volume), that would index all (or, more precisely, almost all) of the observed peaks. These three vectors form a basis, but are unlikely to form a standard basis for a description of the unit cell. The process of converting a basis into a standard basis is called **cell reduction**. The program follows the definitions in the International Tables for Crystallography, and finds the best cells for all of the fourteen Bravais lattices.

The transformation of the primitive cell to a higher symmetry cell may require some **distortion** of the best triclinic lattice that fits the peak search list. Due to experimental errors the fit is never perfect for the correct crystal lattice. Sometimes the observed reflections can fit a higher symmetry lattice than one defined by space group symmetry. Such a condition is called lattice (or metric tensor) **pseudo symmetry**. If this happens, the lattice determination and assignment of lattice symmetry may get complicated. The procedure in such case is to index the data in lowest symmetry lattice that does not introduce wrong lattice symmetry (triclinic lattice is always a safe choice), and look for the symmetry of the intensity pattern during the scaling of symmetry related reflections. *Denzo* calculates the distortion index for all fourteen of the Bravais lattices. It is up to the user to define the lattice and space group symmetry, since the program at this stage of the calculation cannot distinguish lattice symmetry from pseudo symmetry.

Autoindexing

The universal autoindexing procedure is now part of *Denzo*. This procedure is able to deduce the lattice type, crystal unit cell parameters, and crystal orientation parameters from a single oscillation image. It is also able to index more complicated lattices routinely, for example body-centered or rhombohedral lattices. It really makes life a lot simpler!

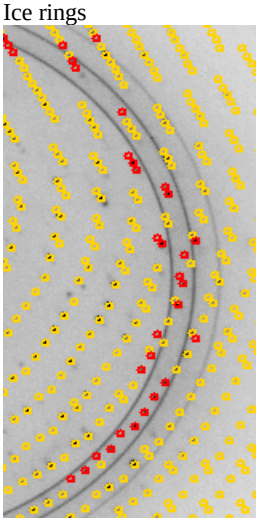
Autoindexing in *Denzo* always finds a **standard lattice**, however the crystallographer may prefer a non-standard choice, for example similar to one in a different space group. Reindexing in *Scalepack* or manual indexing in *Denzo* accommodates such needs.

Peak picking

To start the autoindexing, the first step is to select a set of spots from the displayed oscillation (or still) image. A subset of these spots will be used by the autoindexing routine. To select the spots, go to the ***ImageWindow*** which contains the displayed image and click Peak Sear (**Peak Search**). The number of spots the peak search routine picks can be controlled with options of the menu, which will appear after peak search finishes. If the number of spots is too few or too many, click the **More Peaks** or **Fewer Peaks** respectively, as many times as necessary, until you have about the right number of spots. If your spots are unusually large or small for your detector, choosing **Size Peak Up** or **Size Peak Down**, will change peak search criteria (peak size and overlap) towards larger or smaller spots. You can also edit results of the peak search procedure by using **Pick (Add)** and **Pick (Remove)** options. For example,

you can remove a series of peaks by clicking first **Pick (Remove)**, than clicking in either zoom or main window on peaks you want to remove.

Ice rings



There is no best number of spots to pick for autoindexing, because it is the **quality** of the picked peaks, not the quantity, which is critical. The peaks used in autoindexing should come mostly (say > 90%) from a single lattice and should line up on real reflections in the image. If most do not, then there is something wrong and you may have to change some parameters as discussed below under *When Autoindexing Fails.*

The two most common problems which interfere with the peak search are **satellite crystals** and **ice rings**. With the noisy image, the typical thing one sees is that the peak search program picks a large number of spots which are clearly not part of the lattice. By clicking on **Fewer Peaks** the number of peaks selected can be reduced. If your diffraction pattern has some strong peaks, these will usually get picked before the noise.

Ice rings pose a different problem because the peak search program will pick tons of "spots" within the very intense ice ring. However, it is easy to overcome this by simply specifying resolution limits (say 20 to 4.1 Å) in the autoindexing macro file (`auto.dat`) which exclude the ice ring (more on this later in *Ice Rings* section).

The picked peaks are written to a file called `peaks.file`, in whatever directory you happened to launch the display program in. This file is now read by *Denzo* to autoindex the image.

Running autoindexing

Back to the ***DenzoWindow*** now, you are ready to input the initial parameters and start the autoindexing. For this purpose you will need the following files: the oscillation image file, the `peaks.file` file, and three macro files called `site.dat, myexperiment.dat,` and `auto.dat`. These three macros list the detector and crystal parameters and tell the program to begin the autoindexing. Then, in your ***DenzoWindow***, you would simply type `@site.dat, @myexperiment.dat,` and `@auto.dat`, and the program begins. Here's an example of the macro files:

`site.dat`

FORMAT `dip2000`	*detector type. Very important!*
WAVELENGTH `1.5418`	
FILM ROTATION `270.00`	*careful, only for certain goniostats*
CASSETTE `roty 0.465 rotx -0.102`	*probably determined by previous user*
[**ERROR** `density 0.60`]	*error parameters; change only if it has been determined accurately to be different from default*
BOX `2.4 2.4`	
SPOT ELLIPTICAL `0.6 0.7`	*spot shape descriptions, etc.*
BACKGROUND `elliptical 0.7 0.8`	
OVERLAP `spot`	*flags overlapped spots*
X BEAM `99.4`	*Very important detector parameters!!!*
Y BEAM `99.6`	

In this example and throughout this manual, each major keyword is in **BOLD CAPITALS**, `modifiers` and *`values`* are printed in lower case letters. Comments are *italicized* and do not actually appear in the command file. For example, the modifiers `rotz, roty,` and `rotx` modify the keyword **CRYSTAL** as well as **CASSETTE**. A precise description of each of these keywords and parameters can be found later in this manual.

myexperiment.dat

```
OSCILLATION range 1.0
OSCILLATION start 0.0
DISTANCE 150.0
SPACE GROUP P212121                          use P1 unless you know correct
MOSAICITY 1.0                                adjust to match # spots
TITLE 'my crystal'
SECTOR 1 to 90
RAW DATA FILE 'data:[mydir]xtal0##.osc'      data file ; note single quotes due
FILM OUTPUT FILE 'data:[mydir]xtal0##.x'     to special characters like [ ]
X BEAM 99.4
Y BEAM 99.6                                  [These are included only if they
BOX 2.4 2.4                                  differ from those of the site default.
SPOT ELLIPTICAL 0.6 0.7                      Otherwise, you can omit them]
BACKGROUND elliptical 0.7 0.8
OVERLAP spot
ERROR positional 0.050
PROFILE FITTING RADIUS 30
```

auto.dat

```
PEAK SEARCH FILE peaks.file                  created by display program
FIT x beam y beam cell crystal rotx roty rotz refine crystal and beam
[RESOLUTION LIMITS 100 4.1]                  use only if you want to exclude
                                             data from autoindexing.
                                             Default is all data
WRITE PREDICTIONS                            makes file hklpredictions
GO
```

When *Denzo* indexes the image, we have also told it to write a predictions file (called `hklpredictions`) which contains the positions of the predicted reflections ('preds'). This file is read by the display program and the preds are superimposed on the diffraction pattern. To do this, go to the ***ImageWindow*** and click on Remove pred to toggle between the **update** and **remove predictions** modes. The display program will read the latest `hklpredictions` file and display the predicted reflections. If the predicted reflections line up with the spots, then that's one way you know that things are working.

☛ Note that if you already used Peak Sear in the present session, the program will automatically switch itself into the **update predictions** mode.

After you run the little autoindexing routine, you get a lot of screen output. The critical part is the lattice and unit cell distortion table, and the crystal orientation parameters. An example is shown below:

```
Vector lengths in autoindexing from  5.7 to  152 Angstroms
primitive cubic        60.08%    92.79  92.79  92.79 90.00  90.00  90.00
                                 77.48 124.78  65.26 89.82  32.01  89.54
I centred cubic        31.25%   130.81 130.81 130.81 90.00  90.00  90.00
                                130.97 136.43 124.78 90.00 161.72  84.76
F centred cubic        37.67%   146.44 146.44 146.44 90.00  90.00  90.00
                                146.35 146.67 146.31 63.19 127.02 147.44
primitive rhombohedral  9.82%   132.55 132.55 132.55 27.55  27.55  27.55
                                131.62 140.63 124.78 27.65  18.16  32.85
                                 58.37  58.37 360.56 90.00  90.00 120.00
primitive hexagonal     9.14%    71.24  71.24 124.78 90.00  90.00 120.00
                                 76.75  65.26 124.78 90.18  90.15 147.65
primitive tetragonal   11.96%    54.53  54.53 124.78 90.00  90.00  90.00
                                 41.07  65.26 124.78 90.18  90.58  89.40
I centred tetragonal    6.51%    71.24  71.24 252.50 90.00  90.00  90.00
                                 76.75  65.26 252.50 90.08  85.24 147.65
primitive orthorhombic  0.30%    41.07  65.26 124.78 90.00  90.00  90.00
                                 41.07  65.26 124.78 90.18  90.58  89.40
C centred orthorhombic  2.27%    41.07 252.50  65.26 90.00  90.00  90.00
                                 41.07 252.50  65.26 89.92  90.60  81.22
I centred orthorhombic  4.72%    41.07  77.48 269.93 90.00  90.00  90.00
                                 41.07  77.48 269.93 92.77  72.99  57.38
F centred orthorhombic  3.33%    41.07 136.43 252.50 90.00  90.00  90.00
                                 41.07 136.43 252.50 92.71  98.78  73.08
primitive monoclinic    0.18%    41.07  65.26 124.78 90.00  90.58  90.00
                                 41.07  65.26 124.78 90.18  90.58  89.40
C centred monoclinic    2.27%    41.07 252.50  65.26 90.00  90.60  90.00
                                 41.07 252.50  65.26 89.92  90.60  81.22
primitive triclinic     0.00%    41.07  65.26 124.78 89.82  89.42  89.40
autoindex unit cell              41.07  65.26 124.77 90.00  90.00  90.00
➡ crystal rotx, roty, rotz -131.740 -39.042 -4.743
Volume of the primitive cell  334451.
Autoindex Xbeam, Ybeam  96.42  95.58
crystal rotx   48.260 roty  39.042 rotz -175.257
crystal rotx   48.260 roty  39.042 rotz    4.743
crystal rotx -131.740 roty -39.042 rotz  - 4.743
crystal rotx -131.740 roty -39.042 rotz  - 4.743
```

Blue lines represent the 14 possible Bravais lattices in decreasing order of symmetry, with **primitive cubic** being the highest symmetry and **primitive triclinic** the lowest. After the lattice name is a percent value which represents the amount of distortion the unit cell parameters would suffer in order to fit the lattice. Next to this percentage are the 'distorted-to-fit' unit cell parameters, and below are the undistorted unit cell parameters, for comparison (in case of **primitive rhombohedral** lattice there are actually two lines - second with parameters in hexagonal indexing).

As you can see, for some lattices (e.g. primitive tetragonal, all the cubics, etc.) a substantial amount of distortion is required. Similarly, in all cases there will be zero distortion required to fit the data to a **primitive triclinic** cell, since all three unit cell lengths and angles are unrestricted. **The goal here is to find the highest symmetry lattice which fits the data with minimal distortion**. In the example above, there are two candidates, aside from the

trivial case of P1. These are **`primitive monoclinic`**, with a 0.18% distortion index, and **`primitive orthorhombic`**, with a 0.30% distortion index. Since 0.30% is, in this case, an acceptable amount of distortion, especially given that the unit cell parameters were refined from a single frame, a reasonable choice for the lattice would be **`primitive orthorhombic.`**

Note that this choice of lattice must be confirmed, since it may well be a monoclinic lattice that has a beta angle which is very close to 90 degrees. To do this, process the data with *Denzo*, and then postrefine the unit cell parameters and scale the intensities in *Scalepack*. There will be more discussion of space group determination in the *Scalepack* manual.

First **`crystal`** line (pointed by red ➡) is very important, as it gives the crystal orientation parameters.

Last group of **`crystal`** lines give alternative choices for these values, based on crystal symmetry. Thus, for example, if you preferred to work with small numbers, you would choose the values from the second of these lines for your crystal orientation parameters. But it doesn't really matter, it's just what you prefer.

`Autoindex` Line gives the `x BEAM` and `y BEAM` positions. However, since the success of autoindexing is critically dependent on knowing this accurately in advance, these values should be close to the initial values in the file `site.dat`.

When Autoindexing Fails

Occasionally the autoindexing routine does not give clear-cut results.
There are four classes of data reduction problems:

1. Utter failure
2. Misprediction
3. Kind-of-failed
4. The predicted positions agree perfectly with the peaks, but data scale poorly

Utter failure means that there is no reasonable predicted lattice, the crystal unit cell parameters are unreasonable, and the predicted diffraction pattern does not correspond to the observed image. In most cases failure can be ascribed to problems in three areas:

- experimental conditions,
- the parameters of autoindexing,
- the peak search (rarely).

experimental conditions Problems with defining experimental conditions account for the vast majority of the problems with autoindexing single crystals. If any of the following parameters is widely off, your autoindexing is screwed up (listed in order of importance):

- the `x BEAM` and `y BEAM` positions. They can be measured from the diffraction image using the cursor and reading off the position from the little green box in *XdisplayF*. A direct beam shot or a good educated guess (say based on the intersection of Laue streaks, the symmetry of the diffraction pattern, etc.) is often necessary. At synchrotrons, when the data is collected on IPs in removable cassettes and the beam wanders all over the place on the image, it is **very** helpful to have a **beam stop** which has been **thinned** sufficiently at its center so that a tiny amount of the direct beam passes through and makes a spot on each frame whose position can be measured.
- specification of the **detector format** (believe it, it's happened),
- the crystal-to-film `DISTANCE`. The distance should be known accurately (within about 1%) and can be measured with a tape measure as a last resort
- the `CASSETTE` `rotx` and `roty` values. Misset `CASSETTE` `rotx` and `roty` values may be due to 2θ angles other than zero.

Really, knowing the detector parameters only makes sense. **If you want to deduce some unknowns** (crystal lattice and cell, and orientation), **you need some knowns too**.

oscillation range Very occasionally autoindexing will fail because too large or too small oscillation range was collected. In the case of **large** oscillations, the lunes may have severe overlap and the autoindexing program will have a problem with the non-uniqueness of the mapping from detector to reciprocal space. In the case of **small** oscillations or stills collected on non-mosaic crystals with a very parallel beam, too few reflections may be present to describe a lune. As a rule of thumb the following types of oscillation ranges should be used:

- **viruses:** 0.25^o oscillations,
- **proteins:** 1.0^o oscillations,
- **small molecules:** 2.5^o oscillations,

assuming well diffracting samples. There is a fairly wide margin on either side, at least a factor of 2.

autoindexing parameters Before autoindexing, *Denzo* edits the peak list produced by the peak search algorithm of the XdisplayF program. Problems with the *Denzo* autoindexing parameters account for a few of the remaining problems with autoindexing. These may have to be adjusted in the case of non-single crystals, very large or small unit cells, lots of noise peaks, or ice rings. These parameters are controlled by the keywords: `LONGEST VECTOR`, `RESOLUTION LIMITS`, `WEAK LEVEL`, `BOX`, and `SPOT`.

`LONGEST VECTOR` may need to be adjusted if the peaks are sparse due to a small unit cell or low mosaicity. In this case, it should be set to equal the longest vector of the reduced primitive real space unit cell. If you did not understand that last sentence, simply use the longest unit cell axis (which is the same in primitive space groups, anyway). Better to err on the side of larger rather than smaller values.

`RESOLUTION LIMITS` can be adjusted, if you have:

- an **ice ring** which interferes with the indexing, you can set the resolution limits to, say, 100 - 4.1 Å to eliminate these reflections.
- a **small satellite crystal** which contributes some reflections at low resolution but not at high resolution, you can set the resolution limits to, say, 8 - 2 Å, thereby excluding the strongest contributors from the contaminating lattice.
- a **cracked crystal** where the two lattices are only 2-3 degrees apart, you can often adjust the resolution limits as in the case of an ice ring so that you index at low resolution where the two lattices are approximately identical, and then use manual refinement to zero in on both lattices. The idea here is to index each lattice separately and to combine the data later in *Scalepack*.
- a **twinned crystal** where the two lattices have a non-random orientation relative to one another, the program may fail spectacularly by finding a **superlattice** which indexes both lattices simultaneously but cannot be used to integrate the data because the superlattice does not usually fit both lattices exactly. Then it's time to call an expert.

`WEAK LEVEL` If you have a **number of spurious peaks** in the image distributed randomly over the image, you can adjust the value of the `WEAK LEVEL` which appears before the first `GO` statement in the `auto.dat` macro. This will affect the number of peaks accepted by the autoindexing routine. By increasing the `WEAK LEVEL`, you can exclude weak spurious peaks, then run autoindexing, and examine the superimposition of the autoindexing preds (green circles) on the image. If most of them cover real reflections and not noise, then you are succeeding.

`BOX` and `SPOT` sizes may have to be adjusted if too many spots are being removed by the overlap criteria. **Autoindexing does not use** spots which *Denzo* determines are **overlaps**.

peak search Finally, you can try changing the number of peaks you select in the peak picking routine. However, since *Denzo* edits the list of peaks anyway, this is unlikely to yield much

improvement if the suggestions above do not work. In that case, do you have a better crystal?

Misprediction results in location of reflection masks not corresponding to the positions of the Bragg peaks. It is visibly obvious and is disastrous. It may be due to:

- forcing wrong space group symmetry,
- misindexing or
- serious detector malfunction.

misindexing Autoindexing is based on the assumption that the diffraction spots are correctly mapped from detector coordinates to diffraction (reciprocal) space. The origin of the diffraction space is defined by the position of the direct beam on the detector. A substantial error in the beam position can shift the indexing of the diffraction pattern by an integer vector. Such misindexing can be totally self-consistent until the stage when symmetry related reflections are compared. For any assumed (starting) value of the beam position, the origin of the diffraction space during indexing will be shifted to the nearest grid point of the best primitive lattice. An initial error in the direct beam position by 0.48 times the distance between reflections will lead to correct indexing, while an error of 0.52 times the same distance will cause a misindexing of the diffraction pattern by one index. **Misindexing by one is never corrected by subsequent refinement** of the crystal and detector parameters. Misindexing often produces poor agreement between the predicted and the observed positions of the reflections, but for some crystal orientations, the agreement between the predicted and the observed positions can be equally good for both correctly indexed and misindexed cases. This property of the diffraction geometry creates a potential trap for the unwary crystallographer.

Sometimes the **crystal asymmetric unit** may have molecules related by an approximate translation by a fraction (typically **one half**) of a unit cell edge or diagonal. The resulting diffraction pattern will have odd index reflections much weaker than even index reflections. Autoindexing may find one of the two possible solutions, the choice depending on whether odd reflections are weak enough to be assumed (within experimental error) systematically absent or not. This depends on what is the fraction of odd reflections in the peak search. If there are only a few odd reflections, then most of the peak search result may be explained with a **smaller real-space unit cell**. To prevent autoindexing from finding such a smaller cell, one should enhance the fraction of odd reflections used in autoindexing by changing the peak-search criteria or by using only high-resolution reflections in autoindexing. If one still cannot index odd reflections, then one should consider ignoring them all together. In such a case, one can solve the structure in a smaller unit cell and the resulting structural error will not be very significant if odd reflections are much weaker than even ones.

Kind-of-failed means that a lattice other than P1 was predicted, but it had a high distortion index, say 1-1.5%. This may not be a "failure" but simply an indication that further refinement of parameters is needed because the initial detector parameters were very far off. Unfortunately, misindexing may give similar results. A quick check of the **superposition** of the preds on the reflections should tell you if the refinement is likely to succeed. Note that refinement may converge to low χ^2 values even with misindexed crystals but eventually the problem will surface and you will have to index correctly.

Predicted positions **agree perfectly with the peaks** and no detector or diffraction artifacts are visible, but **data scale poorly** and produce many outliers. This problem may be:

- a simple mistake in data processing, like using **wrong file format**, or
- a **non-uniform exposure** during crystal oscillation.

The **non-uniform exposure** may be caused by spindle motor backlash, shutter malfunction (opening too early or too late), ionization chamber electronics failure (if used), decay or

variation of the X-ray beam intensity (if ionization chamber is not used), variable speed of the spindle motor etc. Non-uniform exposure is best diagnosed by exclusion of other problems that may affect data quality. Graphical feedback provides confidence that the problem cannot be at the indexing/integration stage.

Absorption

Large variations in **absorption** of X-rays by the crystal will make data scale poorly and will produce visible variation of the background, however it will not affect positional agreement. The variation in the absorption can be avoided easily by a proper mounting of the crystal. The correction for absorption is a whole field in itself.

Errors in detector orientation will produce distorted mapping from detector to reciprocal space. A **wrong crystal-to-detector distance** will enlarge (or decrease) the apparent reciprocal space separation between Bragg peaks. This error will not be uniform in three directions; in the first approximation, along the beam direction, the error will be proportional to square root of the error in the distance; in the other two directions the error will have linear dependence. In most cases a significantly wrong crystal-to-detector distance (say an error of 10%) will not make the autoindexing step fail immediately; however, **the calculated unit cell will be quite wrong**. The length of unit cell along the beam direction will be 5% shorter, in the perpendicular directions 10% shorter. If the crystal is diagonally oriented (no principal axis along the beam direction) than the apparent angles between axes will violate the lattice symmetry. Incorrectly defined **angles between detector and the beam** will result in wrong angles between crystal axes. Therefore, the interpretation of the lattice symmetry is dependent, to a some extent on how precisely the detector parameters are known *a priori*.

Manual indexing

See also *Manual indexing* appendix.

When autoindexing fails, you can try to index oscillation images manually, but if you tried the suggestions made above, you are unlikely to do any better. Manual indexing used to be the only way to run *Denzo*, but autoindexing has superseeded this difficult task.

Autoindexing succeeded !

If the autoindexing looks good, then it's often nice to finish up the frame you have been working on to make sure you have the correct values for `MOSAICITY` and `SPOT` shape. This can be done interactively in the ***DenzoWindow***, by calling a refinement macro, examining the superposition of the preds on the diffraction image and then testing the `MOSAICITY` and `SPOT` parameters to make sure all the reflections are indexed.

Multiple crystals, simulation and interactive indexing

Autoindexing can fail when more than one crystal contributes to the diffraction image. Sometimes, editing of weaker reflections and resolution cuts can make one crystal dominate the peak search list enough for the autoindexing method to succeed. If crystals have a similar orientation, sometimes using only very low resolution data can be the right method. In case of twinned crystals, autoindexing sometimes finds a **superlattice** that finds integer indexes for both crystals. In such case *Denzo* solves the problem of finding the best 3-d lattice that goes through all of the observed peaks. Unfortunately, for a **twinned crystal** this is a mathematically correct solution to a wrong problem. Since there is no general algorithm to index a diffraction image from **multiple crystals**, one has to rely on the human ability to sort-out complex patterns in such cases. The approximate orientation can be determined by an iterative trial-and-error process where the predicted pattern is being adjusted, keeping the diffraction image constant.

Crystal orientation can be defined relative to any principal or higher order zone perpendicular to the X-ray beam. This flexibility helps the interactive indexing when only a higher order zone is visible in the diffraction pattern. This is particularly useful in centered space groups where it is easier to orient a diagonal zone, rather than a major one.

Manipulation of the predicted diffraction patterns also can be used to simulate diffraction experiments. The simulation can help set a proper data-collection strategy in order to avoid later problems in data reduction. Using the program for **simulation of diffraction patterns also can be a tool for teaching crystallography**.

Refinement: Using the Fit Command

If autoindexing has given you a pretty good match between the observed diffraction pattern and your preds, you can use *Denzo* to refine the parameters to several significant figures.

The crystal and detector orientation parameters require refinement for each processed image. The refinement can be simple, for a series of images collected with an on-line detector, or more complex, if the detector orientation is only crudely known and varies from image to image, as it is in the case of off-line scanners. The refinement is controlled by the user and can consist of several steps. In each step the user defines the resolution limits and the order and number of parameters to be fitted. Both detector and crystal parameters can be fitted simultaneously by the fast-converging least squares method. The refinement is done separately for each image to allow for the processing of data even when the crystal (or the detector) slips considerably during data collection.

What to fit first

Although *Denzo* command FIT ALL can successfully refine all the parameters - crystal rotations, film rotations, camera distance, unit cell dimensions, etc. - simultaneously if their initial values are close enough, in practice it is a good idea to start by refining those values you are **least sure** of, and which are **most critical**, first. These are usually the crystal rotations, unit cell parameters, and the beam positions. In addition, the resolution limits must be set, and generally you start refining at medium resolution and progress to higher resolution in one or two steps. So first round of refinement would start with two commands:

```
RESOLUTION LIMITS 100 2.8
FIT crystal rotx roty rotz cell x beam y beam GO GO GO.
```

This tells the program to refine the crystal rotation angles, the unit cell lengths and angles, and the direct beam position for three cycles (= 3 GO statements), using reflections between 100 and 2.8 Å, and applying the results of the first cycle to the second, and so on. In the ***ImageWindow***, you will see the preds move after each cycle to their new positions. If things look like they are getting closer, you can do more cycles simply by typing GO more times.

FIT remains on! ☛

Once you specify the FIT operation, as in FIT `crystal rotx roty rotz`, these parameters will be refined every time you type GO, no matter what you have specified on the same line as GO. For example, let's say that after the 3 cycles of CRYSTAL `rotx roty rotz` refinement, you decide to refine the X BEAM and Y BEAM positions using the command FIT `x beam y beam` GO GO GO. Now ***not only*** will the beam position be refined, but simultaneously so will the CRYSTAL `rotx roty rotz` rotations. Once FIT is on for a parameter, it is always on unless you turn it off. Those of you with sharp eyes will have noticed that the `auto.dat` macro already included the FIT `crystal rotx roty rotz cell x beam y beam` statement, but it doesn't hurt to say them again.

FIT & autoindexing ☛

The program uses one of two target functions specified by the keyword FIT. In the first cycle following autoindexing, the target function uses the reflections found by the peak search. In all subsequent cycles, the target function includes all reflections which overlap predicted reflections. The first target function which follows autoindexing has a much larger radius of convergence, at a tradeoff of less precision. On all subsequent FIT cycles, a more

precise fit algorithm is used which completely ignores the list of reflections from the initial peak search.

Turning FIT off

Generally, **once the refinement is progressing you don't need to turn it off**.

☛ **Don't** set up multiple steps of fitting and fixing in an attempt to **refine a single parameter** at a time. That is neither necessary nor smart, since *Denzo* can fit multiple parameters simultaneously, and because single parameter refinement may not allow correlated parameters to be refined sufficiently.

When you really don't want to refine a parameter any more, for example to simulate an experiment, use the command FIX, as in FIX `crystal rotx roty rotz` GO. The crystal rotation parameters will now be fixed at their last refined position (unless you re-enter them manually) and will not refine on subsequent GO commands unless you specify FIT again.

If things get worse

If the autoindexing succeeded, subsequent **refinement never should make things worse**. If it does, it is best to make sure that the autoindexing really did work after all. Go back and re-read the section on failures of autoindexing and try the suggestions given there. This is probably easier than trying to index your crystal manually.

Finishing the Refinement

As a general guide, for automatic scanners the parameters are refined in this order:

CRYSTAL `rotx roty rotz`	
X BEAM Y BEAM	
UNIT CELL	
CROSSFIRE `x y xy`	
CASSETTE `rotx roty`	
RADIAL OFFSET and ANGULAR OFFSET	*for radial scanners*
Y SCALE and SKEW	*for orthogonal scanners*
DISTANCE	

Not all parameters are refined for every crystal because some are highly correlated. In particular, the crystal-to-film distance, and the detector parameters Y SCALE and SKEW (or RADIAL OFFSET) are correlated with the unit cell dimensions. A particularly dangerous combination to be aware of is the simultaneous refinement of DISTANCE, CELL, and RADIAL OFFSET. These are so highly correlated that they can be refined only simultaneously for high quality, high resolution data. Generally, the greater the curvature of the Ewald sphere and the higher the symmetry of the crystal lattice, the better conditioned the problem. That is, correlations are weaker and parameters are more independent. Some simple rules for when you can safely refine all parameters together are given below:

Refine all parameters when resolution/λ	≤ 1.5 for triclinic lattices
	≤ 2.0 for trigonal / tetragonal lattices

Otherwise don't refine Y SCALE, SKEW and DISTANCE.

If things are going well, you should see an **excellent** agreement between the diffraction spots and the preds. You can now extend your refinement to higher resolution. Here is a macro which finishes the refinement of one frame after autoindexing:

refineone.dat

RESOLUTION LIMITS `100 2.8`	*don't use highest res. yet*
FIT `crystal rotx roty rotz`	*refine crystal orientations*
GO GO GO	
RESOLUTION LIMITS `100 2.2`	*highest resolution*
FIT `cell x beam y beam crossfire x y xy`	*fit other parameters*
FIT `cassette rotx roty`	

`FIT distance y scale skew`	*only refine if res Λ ≤ 1.5 or 2*
`GO GO GO GO GO`	

A summary of the current values for all the parameters you have refined can be obtained by typing the command `LIST`. Now it's time to examine the refined predictions and adjust the mosaicity and spot shape.

More about refinement - how it works

The integration of reflections requires knowledge of their indices and position. The weak reflections can be found only by prediction based on the information obtained from strong reflections. The autoindexing provides only approximate orientation of the crystal, and the result may be imprecise if the initial values of the detector parameters are poorly known. The least squares refinement process is used to improve the prediction. The parameters describing the measurement process either have to be known *a priori* or have to be estimated from diffraction data by the refinement procedure. Depending on the particulars of the experiment, the same parameters (e.g. crystal to detector distance) are more precisely known *a priori*, or are better estimated from the data. *Denzo* allows for the choice of fixing or refining each of the parameters separately. However, using it well requires considerable knowledge of diffraction experiments. Fortunately, **the `FIT ALL` option and detector specific default values seem to be reliable under most conditions**.

Eigenvalue filtering
Occasionally the refinement can be unstable because of a high correlation among some parameters. High correlation makes it possible for the errors in one parameter to compensate partially for the errors in other parameters. If the compensation is 100%, the parameter would be undefined, but the error compensation by other parameters would make the predicted pattern correct. In such cases **eigenvalue filtering** (described in *Numerical Recipes* as Singular Value Decomposition) is employed to remove the most correlated components from the refinement and make it numerically stable. Eigenvalue filtering works reliably when starting parameters are close to correct values but may fail to correct large errors in the input parameters. Once the whole data set is integrated, the global refinement (sometimes called **postrefinement**) can refine crystal parameters more precisely. The unit cell used in further calculations should come from the global refinement (in *Scalepack*) and not from *Denzo* refinement.

Partiality refinement
The refinement of the fraction partiality works by minimizing the function:

$$\sum_{partials} \frac{1}{\sigma^2_{(-)}} \left(I_{partial} - F \cdot I_{fully} \right)^2$$

where

$$\sigma^2_{(-)} = \sigma^2_{I_p} + \left(F \cdot I_{fully} \right)^2 + \left(error\ positional \cdot I_{fully} \right)^2$$

$\sigma^2_{I_p}$	= the sigma of the measurement of the reflection
$I_{partial}$	= the intensity of the partially recorded reflection
F	= the fraction partiality
I_{fully}	= the average intensity of the fully recorded reflections in this particular resolution shell
error positional	= the error in the positional measurement (see **`ERROR`** `positional` keyword)

The quantity F, as a function of the crystal parameters, is the variable in the equation. The other parts are determined by the observations and by the error model.

The detector and crystal parameters are refined by a **least squares method** that minimizes the deviation of the reflection centroids from their predicted positions. Such refinement by itself is seriously deficient when applied to a single oscillation image, since one crystal rotation parameter (rotation around spindle axis) is undefined and the others are highly correlated and/or poorly defined. To overcome this problem, another term (**partiality refinement**) is added within *Denzo*, in which the intensity of the partially recorded reflections is compared to the predicted partiality multiplied by an average intensity in the same resolution range.

The formula for the residual (difference between expected and predicted value) is the same as in the postrefinement; however, at this stage, the error of the predicted fully-recorded intensity is much larger, equal to the expected intensity. Nonetheless, the **concomitant positional and partiality refinement** used in *Denzo* is both stable and very accurate. The power of this method is in proper weighting (by estimated error) of two very different terms - one describing positional differences and the other describing intensity differences. The method leads to a reduced correlation between the detector and crystal parameters. An additional benefit is the uniform treatment of both detector and crystal variables in the whole refinement process.

Mosaicity and Spot Shape

Denzo defines **mosaicity** as the smallest angle through which the crystal can rotate, about any axis or combination of axes, while a reflection is still observed. Typical values range from less than 0.1 degree to several degrees. Generally, the higher the mosaicity, the less useful the crystal. ***Denzo* does not refine mosaicity**, so you must input a reasonable value. The easiest way to do this is to simply increase the mosaicity (entering it as `MOSAICITY 0.5 GO,` for example) until all of the observed reflections are matched with preds. Thus, if you have:

- more observed reflections than preds ➔ **increase** the mosaicity.
- more preds than observed reflections ➔ **decrease** the mosaicity.

Useful increments are 0.1° for typical protein crystals and 0.02° for virus crystals.

Depending on how *dark* you have set your display, you can get fooled into thinking that your mosaicity is higher than it really is, because you are trying to get preds to cover very weak reflections which result from diffuse or satellite scattering. To get around this, dim your display so that only the reflections representative of your Bragg scattering are displayed. Then adjust the mosaicity as above until a good match is seen.

A third, and more precise way of setting the mosaicity is to examine the histogram that *Denzo* produces in the log file (or outputs to the screen, if working interactively) after the `CALCULATE GO` statement is given (you can do this now, but it is the last operation you will be able to execute on this image unless you restart *Denzo* since the `SECTOR` argument is incremented with the `CALCULATE` command, and thus a new frame is read; so hold off on this until you've done the best you can with matching up preds to reflections, and after you've adjusted the spot shape and size). The use of the histogram is discussed in the section *Understanding the Denzo Log File*, later in this chapter. To summarize briefly, you want to adjust the mosaicity so that the observed average-partiality histogram declines almost to zero in the last mosaicity zones.

☛ **It is better to err on the side of a slightly high mosaicity** because this way you can be sure of catching all the spots. This is because *Scalepack* can re-label measured (partial) reflections as non-diffracting, but cannot do the opposite, i.e. make up a reflection which *Denzo* did not find. Postrefinement with *Scalepack* will calculate an accurate measure of mosaicity, and if the value used in *Denzo* turns out to be low or way too high, the frames can be reprocessed with *Denzo* using this corrected value for the mosaicity.

Spot Shape and Size

When *Denzo* actually fits a profile to each reflection and measures its intensity, it does so only to the portion of the reflection that is within the borders of the predicted spot (the **pred**). If you choose a pred which is:

- too **small** ➔ intensities will be too low because part of the reflection was not measured.
- too **big** ➔ you run into the risk of:
  a) rejecting reflections due to overlaps (overlaps are defined by whether or not the **predicted** spots, guard, or background regions are overlapped, not whether the actual spots are),

b) including portions of a neighboring reflection in the measured intensity, and
c) including portions of the background in your intensity measurement. Therefore, it is wise to choose a pred size that matches those of your reflections as closely as possible.

Often the diffracted spots are slightly elliptical. They also vary in shape with the size of the crystal, shape of the X-ray beam, etc. *Denzo* allows you to change the size and shape of the pred spot to match the actual reflection better. The way to judge the correct spot size is to zoom in on an area of the image and compare the magnified shape of the actual spot with the magnified pred shape.

☛ You can't use different spot shapes for different regions of the image. If your spot shape varies over the image, you'll have to choose the best compromise. This is not so bad, though, because you pay only a small penalty for including the background in the spot when using profile-fitted intensities.

The keyword **SPOT** is modified by either `radius` or `elliptical`, followed by numbers. It is convenient to get used to using **SPOT** `elliptical`, because a circular spot can be described by an ellipse whose major and minor axes are equal and you will undoubtedly run into elliptical spots one day. A typical command for an elliptical spot would be **SPOT** `elliptical 0.6 0.5 0.0` **GO**, where the first number is the length in mm of the major semi-axis, the second number is the length of the minor semi-axis, and the third number is the clockwise angle between a horizontal line and the major semi-axis. Once you have enlarged a region of your image, it is quite easy to find the appropriate spot shape. An idealized example is shown below:

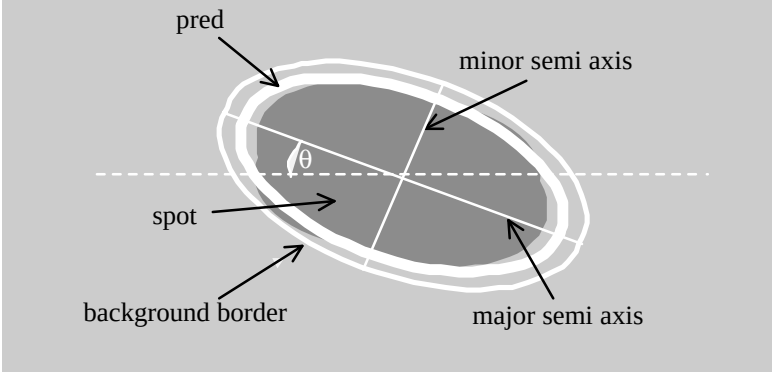


Figure 4: *DENZO spot terminology. Note that most of the time* q *is zero.*

Make sure you adjust the **BACKGROUND** to match the size of the spot. In our example above, a reasonable value for the background is 0.1 mm larger than the spot in all dimensions, i.e. **BACKGROUND** `0.7 0.6 0.0` **GO**.

Profile Fitting Radius

In addition to adjusting the spot shape and size, it is also a good idea to check and see if the **PROFILE FITTING RADIUS** is set properly. This is easy to do with the ***ImageWindow***. In the ***ImageWindow***, simply click on the Prof fit R, and when you move the mouse you will see a white circle which corresponds to the **PROFILE FITTING RADIUS** set in *Denzo*. The pred display changes now too. Instead of seeing all of the predicted reflections, you see only the preds that correspond to those reflections that are stronger than the weak level. Only this subset is shown because these are the only reflections used in the calculation of the average spot profile within the radius. Your radius must be large enough to include enough strong reflections for proper profile fitting, and not too large to average out significant profile

variations. As a rule of thumb, you want the radius to include between 10 and 50 *non-weak* reflections. Useful increments of the profile fitting radius are 5% of the detector size.

How it works: Integration of the Diffraction Maxima

To calculate the diffraction intensity, the detector **background** has to be estimated and then subtracted from the reflection profile. The standard method used to estimate the background value was to calculate an average detector signal in the neighborhood of a specific reflection. In *Denzo* it is assumed that the background is a **linear** function of the detector coordinates. Robust statistics (as discussed in the *Numerical Recipes*) are applied to remove the contribution of pixels that deviate by more than 3σ from the best fit to the background function. If too many background pixels are flagged as outliers the whole reflection is removed from integration.

Denzo ignores pixels in three other cases, when:

- they have been flagged as having no measurement by an auxiliary program,
- they have a special value (e.g. zero in the case of R-axis or MAR) or
- they are in the spot area (based on the predicted, rather than the measured position) of an adjacent reflection.

Profile Fitting

The accurate prediction of spot positions is necessary to achieve a precise integration of Bragg peaks. The most important need for accurate prediction of the spot positions arises from the application of profile fitting.

Profile fitting is a two step process:

- First, the profile is predicted **based on the profiles of the other reflections** within a chosen radius. The predicted profile in *Denzo* is an average of profiles shifted by the predicted separation between the spots, so that they are put on top of each other. If the predicted positions are in error, then the average profile will be broadened and/or displaced from the actual profile of the reflection.
- In the second step the **information from the actual and the predicted profile is combined** by the following process:

The observed profile M_i is a sum of the Bragg peak and background. P_i (the estimate of M_i), is expressed by the formula:

$$P_i = B_i + \text{Const} * p_i \qquad (1)$$

where B_i is the predicted value of the background and p_i is predicted profile.

Profile fitting **minimizes** function:

$$\sum_{i} \frac{\left(M_i - P_i\right)^2}{V_i} \qquad (2)$$

where V_i is a **variance** (σ^2) of M_i . V_i is a function of the expected signal in a pixel, which in the case of a counting detector is P_i . The index i represents all pixels in a two-dimensional profile; the same formulation of profile fitting applies to one and three-dimensional profiles. The predicted profile can be normalized arbitrarily; the most natural definition of normalization is that the sum of p_i is equal 1. Such a choice makes the constant in expression (1) the fitted intensity I, i.e. I = Const

The solution to the profile fitting can be expressed by an alternative, but mathematically equivalent, approach, presented below.

Each pixel provides an estimate of the spot intensity I equal $(M_i-B_i)/p_i$ with variance V_i/p_i^2. A profile-fitted intensity is then simply a weighted average of all observations:

$$I = \frac{\sum \frac{p_i^2}{V_i}\frac{(M_i - B_i)}{p_i}}{\sum \frac{p_i^2}{V_i}} = \frac{\sum \frac{p_i(M_i - B_i)}{V_i}}{\sum \frac{p_i^2}{V_i}} \qquad (3)$$

This approach (without an explicit solution presented in Eq.1) was first published by Diamond in 1969 for the one-dimensional case. However, in 1974 Ford proposed a simplified formula where V_i is constant. This was based on the mistaken idea that the variance of the optical density value of the exposed film is independent of the degree of X-ray exposure. Equation 3 thus became simpler:

$$I = \frac{\sum p_i(M_i - B_i)}{\sum p_i^2} \qquad (4)$$

Many of the subsequent programs followed the formulation of Ford rather than that of Diamond, even when applied to data collected with proportional counters or IPs. The **unweighted** formula proposed by Ford works quite well where the peak spot intensity is not much higher than the background intensity. This situation arises more often with data collected on film, which has a high intrinsic background and low saturation, or when the crystals have low scattering power due to a very large unit cell, high solvent content, or a disorder. The unweighted profile fitting improves the accuracy of the weak reflections compared to a straight summation, but at the cost of reducing the accuracy of the strong ones. This observation did lead in the past to a partial solution based on taking a weighted average between profile-fitted and summed intensities, where the weight is a function of the reflection intensity. The **weighted** formula (Eq.3) used in *Denzo* does not deteriorate the accuracy of strong, low resolution reflections. Thus, the observed problem with the unweighted formula is in the lack of weighting.

Errors of the profile fitting

The profile fitting increases the precision (decreases the statistical error) of the measurement, but it may introduce an error due to lack of accuracy of the predicted profiles. *Denzo* applies the averaging of profiles in detector coordinates and, unlike other programs that use the profile-fitting method, **averages profiles separately for each spot**. This approach has two main advantages:

1. first, **only nearby spots are chosen for averaging**, ones which should have the most similar profiles.
2. second, ***Denzo* avoids interpolation** in the profile-prediction step; instead, it shifts the contributing profiles by vectors that make the smallest possible pixel-truncation error. These translation vectors precisely center the predicted profile on the reflection to be fitted and the error introduced by these shifts is smaller than that due to interpolation used in some other programs.

Diamond, R. 1974 *"Profile analysis in Single Crystal Diffractometry"* Acta Crystallographica A25:43-55

Prediction of the profile shape is never exact, due to errors in the positional refinement, due to averaging of different shapes, due to truncation of pixel shifts or interpolation, etc. The resulting error of the fitted intensity was analyzed by Diamond in the case of one-dimensional Gaussian profiles and an unweighted profile-fitting formula. The important parameters are w - the root mean square (rms) width of the actual profile, f - the rms width of the predicted profile, and d - the displacement of the predicted profile from the actual profile. Let's define the relative change in the square of the reflection width:

$$\Delta^2 = \frac{f^2 - w^2}{w^2} \tag{5}$$

Diamond calculated that the fitted intensity will be wrong by a factor of

$$\sqrt{1 + \frac{\Delta^2}{2 + \Delta^2}} \tag{6}$$

The averaging of profiles adds a value of $r^{2/3}$, where r is the detector pixel size, to the value of f^2. Averaging will increase the profile-fitted intensity of most reflections by a constant multiplicative factor, which has little effect on crystallographic procedures. The interpolation broadens the profile by a factor dependent upon the position of the predicted reflection relative to the pixel boundaries. The interpolation will also increase f^2 by a number between zero and $r^{2/3}$. The interpolation method will increase the profile fitted intensities on average by the same factor, but will also add random noise to the reduced data.

Other aspects of spot integration

There are other, often subtle, ways in which errors in spot positions can lead to serious integration errors. In many experiments the detector is placed as close as possible to the crystal while keeping the diffraction spots separated. In such cases the **reflections are barely separated** and even small errors in the spot prediction would make integration and background measurement areas of a reflection intruding upon the adjacent peaks, and thus lead to an inaccurate estimation of the peaks' intensities.

Errors in the prediction of spot positions also affect the statistical error (**precision**) of the summed intensities. If the predictions do not match the peak position exactly, one has to enlarge the expected spot area in order to sum the intensity of the whole spot. This enlargement of the predicted spot area increases the total background to be subtracted. A larger background has a larger variance, and this adds to the measurement variance**. Auto-centering** of the spot area can compensate for errors in the prediction, but this works well only for strong spots. It would seriously bias the calculated intensity if applied individually to every spot. Some programs do auto-centering by averaging the local deviations between the observed and the predicted positions. While this is not done explicitly in *Denzo*, the profile-prediction algorithm used in the program has a similar effect.

A correction for the non-linear response function of the detector to the photon flux is applied internally in *Denzo* so that it can read the original data without the need for any prior transformations (with the exception of the data from spiral scanners). Pixel values can represent two special cases: **no measurement** or **detector overload**. Overloaded pixels are assumed to be close to the center of gravity of the diffraction spots and as such they are used in determining the spot centroids. Pixels that are either overloaded or had no measurement are ignored in calculating the spot intensity by the profile-fitting, but the existence of such pixels in the spot area is flagged by a negative sign applied to the σ estimate. Profile-fitted intensities seem to be reliable independent of the existence of such pixels in the spot area.

Ice Rings

Ice rings are sometimes seen in diffraction images from frozen crystals, but they need not ruin a data set. There are two places in the indexing process where ice rings can cause problems. The first is during autoindexing, and the second is during the refinement and integration steps after getting the initial orientations.

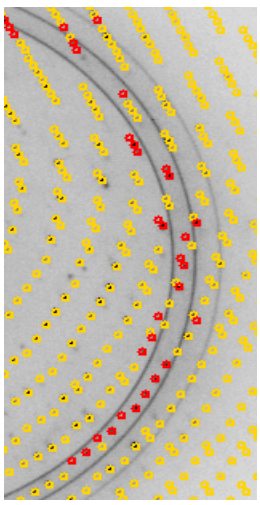
Ice rings in autoindexing

The first thing to do if you have ice rings is to simply ignore them and run autoindexing as usual. *Denzo* edits the peak list produced by the Peak Search routine of *XdisplayF* and may succeed in autoindexing the image regardless of whether the display program picked a lot of ice (non-protein lattice) peaks. You can tell if this worked by inspecting the image after autoindexing and then again after running the `@refine` macro. If the preds line up on real spots, then autoindexing worked despite the ice ring.

If the ice ring does interfere with autoindexing, here is a simple remedy**: adjust your resolution limits** to exclude the ice ring. This means you would edit the `auto.dat` macro to include the resolution limits. In the example provided above, the resolution limits were commented out. You could exclude the ice ring from autoindexing by setting the high resolution limit to about 4.1 Å. Once the initial orientation is found, all of the data can be included in the subsequent refinement.

Ice rings in your data

There are two ways in *Denzo* to reject reflections corrupted by ice rings.



The elegant way exploits the fact that the **distinguishing feature of spots corrupted by ice rings is a very high and irregular background**. *Denzo* assesses the quality of the background of each spot against criteria for slope and uniformity, and if a certain fraction of the pixels in the background fail this test, then the whole spot is rejected. This is controlled by the keyword **`REJECT`** `fraction`. The default value for this rejection is 0.75 (i.e. 75% of the background pixels must be acceptable). If you have an ice ring, you can increase **`REJECT`** `fraction` until most of the spots in the ice ring are no longer picked up. Useful increments are 1-2% at a time. Do this while you are looking at the display, just as you did for spot shape and mosaicity. As you increase the reject fraction, the number of good preds in the ice ring should decrease, while leaving most of the others alone.

The second way to avoid ice rings in your data is to simply **process your frames twice**, once at a resolution cutoff below the ice ring, and then again for the data beyond the ice ring. The two data sets are then merged in *Scalepack* along the lines of Scenario 10 in The *Scalepack* Manual. This is more of a brute-force approach and is likely to result in the loss of more good data than the approach outlined above.

Refinement Statistics

After each **`GO`** command, *Denzo* not only updates the pred display, it also prints out the a numerical summary of the refinement cycle:

1. The first part lists the **refined parameters**, their final values, and how much they changed in the last refinement cycle.
2. The last part lists the **number of spots** used in the refinement, the χ^2 values for the x and y positions of the preds, and the predicted decrease in the χ^2 values from the previous refinement cycle. If things refined well, the number of spots used in the refinement should be a significant fraction of the total number of spots on the image. This number reflects the number of preds that line up with observed spots. The χ^2 values represent the average ratio, squared, of the error in the fitting, divided by the expected error.

A good refinement will have χ^2 values near 1.0. In the early stages of refinement these values will be high, but towards the end they should be below 2 or so. The **predicted decrease** is a measure of the convergence of the refinement. When the predicted decrease is near zero, further **`GO`** commands will have no added effect. In other words, the current parameters have been refined as far as possible.

☛ **The magnitude of the χ^2 values is not a critical test for the success of *Denzo***, since they represent only the comparison of the spatial differences between the observed and predicted reflections to an error model that is typically quite strict in the prediction of small errors. Assuming default values of the error estimates, χ^2 values of 2 or even 3 are acceptable, because the position of the predicted reflection, and hence the intensity, is still very accurate. Thus, the χ^2 values may not reflect errors in the integration of the reflections.

It is a good idea to keep your error model roughly the same from crystal to crystal so that you can use the χ^2s to pick up any other variations in your data collection. For example, if you are getting χ^2s in the range of 2 to 5 consistently and your crystal is good, consider problems with the detector alignment.

If the χ^2 values are very high, (> 10 or so for default values of the error parameters) **something is seriously wrong** with the indexing, refinement, or the detector. This will be apparent from **visual inspection of the displayed image**, in any case.

Since the goal of *Denzo* is to produce a list of *hkl*s and unscaled intensities, it does not matter that the errors in the positions differ a bit from the standard expected error. On the other hand, χ^2 values substantially different from 1 in *Scalepack* should be investigated, because this directly affects a critical result of the scaling procedure, namely the σ value assigned to each scaled intensity. So that's where you should pay attention to finer deviations of χ^2 values from 1, not in *Denzo*.

Finishing Up

Once your first frame is indexed and you've adjusted the mosaicity, spot, and box sizes, it's time **to process the rest of your frames**. This can be done as a batch job or as a hands-off interactive session. Wladek and Zbyszek recommend that you take half an hour and watch your frames refine and integrate interactively, because this is the best way to spot any problems. For example, if you have misindexed the first frame by one (due to an incorrect beam position), you are likely to see it very quickly this way.

Since you still have your ***DenzoWindow*** and ***ImageWindow*** open, just continue on from there using the following refinement macro:

refine.dat

`FIX all`	*good to do just in case*
`START REFINEMENT`	*beginning of refinement loop*
`PRINT no shifts no profiles`	*limits output*
[insert beam search routine here if necessary (see below)*]*	
`RESOLUTION LIMITS 100 2.8`	*don't use highest res. yet*
`FIT crystal rotx roty rotz`	*refine crystal orientations*
`GO GO GO`	
`RESOLUTION LIMITS 100 2.2`	*highest resolution*
`FIT ALL`	*fit other parameters*
`GO GO GO GO GO`	
`PRINT profiles 1 1`	*print average profile in single frame*
`CALCULATE GO`	*end of refinement loop*

This macro will refine all of the subsequent frames specified by the `SECTOR` in the macro `myexperiment.dat` (p. 26). You can just sit back and watch them zip along on the screen. The result will be a series of .x files called `xtal0##.x`.

Alternatively, you can process everything in batch mode using the following command file:

denzo.com

`setenv cnverr yes`	*this line is only for IBM RS/6000*
`cd /people/mydir/DENZO`	*machine specific calls*
`/usr/local/bin/DENZO << STOPINPUT >> xtal.log`	*start the program, name log file*
`@site.dat`	
`@myexperiment1.dat`	*note modified expt. macro here*
`@refine.dat`	
`END OF PACK`	*end marker, increment sector*
`END OF JOB`	
`STOPINPUT`	*unix stop signal*

Note that the file `myexperiment1.dat` has been modified from `myexperiment.dat` to include the final refined values of everything determined in the interactive refinement of the first frame, done above. The easiest way to do this is to examine the `xtal001.x` file produced by the `CALCULATE GO` statement. At the end of this file are all the final parameters determined from refinement. Cut these out and just paste them into the end of `myexperiment.dat` and rename as `myexperiment1.dat.` Since you are not running autoindexing in the batch mode, you have to take the results and paste them in by hand.

Warning: for unusual instruments (e.g. R-Axis with non-vertical spindle, synchrotrons) make sure that you have the correct value of **FILM ROTATION.** The defaults in the formats are only for the standard configuration of each instrument.

All that remains is to submit the job and examine the log file for the results. The processed frames in this case have the name `xtal0##.x.` (The ## wild cards have been replaced by the **SECTOR** argument.) These ".x" files are ready to be scaled in *Scalepack*.

findbeam.dat

BOX `5.5 5.5`	*big box*
SPOT `elliptical 2.0 10.0 0.0`	*large spot shape; can be symmetric also*
BACKGROUND `radius 1.5`	
USE BEAM `only`	*ignores all spots except the beam spot*
X BEAM `100.0` **Y BEAM** `102.0`	*starting values*
FIT `x beam y beam`	*begin fitting the beam*
GO GO GO	
BOX `2.5 2.5`	*narrow down box now*
SPOT `radius 1.0`	*smaller spot too*
BACKGROUND `radius 1.0`	
GO GO GO	*more refinement of beam position*
SPOT `elliptical 0.6 0.7 0.0`	*normal spot size now*
BACKGROUND `elliptical 0.7 0.8 0.0`	
USE NO BEAM	*Once the beam is found, it is best not to refine.* ☛ *The reflections, which were ignored with* `USE BEAM only`*, will be refined once the* `RESOLUTION LIMITS` *are given in* `refine.dat`.

Another problem sometimes encountered with off-line scanners involves not the scanner, but rather the carousel on which the imaging plate cassettes are mounted during data collection. Sometimes these old war-horses, carrying their heavy load of IP cassettes, do not stop precisely at the zero position such that the imaging plate is perpendicular to the beam axis. In this case, care should be taken to refine the **CASSETTE** `roty` value.

Understanding the DENZO Log File

The log file can be examined to see if the data processing went well. It is very long, since it contains a record of every refinement cycle. The interesting part comes after the **CALCULATE** statement at the end of the refinement. After printing out the final refined parameters, the following output is given. Notes are in green:

1. These numbers below are the maximum **h**, **k**, **l** expected for a given resolution range, the number of pixels needed to save the spot profile, and two computer memory statements.

```
hklmax=  99  99  99
93 pixels needed to save spot profile
maximal usage of the common /sfprof/ :283044
```

2. Below is a complete list of the suspect, but not rejected, reflections. The columns contain, in order: index, x and y coordinate of the reflection centroid, the oscillation angle at which maximum diffraction occurs (within the oscillation range), the measured intensity of the reflection, the normalized intensity of the reflection, the error on I, and then some flags which describe the problem with the reflection. There are only two types of problems flagged: negative intensity, and positional error. This list is useful because you can go back to the image and see what the problem is for each reflection.

```
  h   k  l      x        y  oscang intens scalI errI   xdel  errx   ydel  erry
 -27 -84 -5 p  60.25  31.81  0.02 -797.9 -0.29 129.6  0.24  1  negative
  -4 -28  2 p  96.13  91.63  0.50 1696.6  0.03 114.8  0.115 0.033 -0.274 0.036 0.0 1 position
  15   2 -1 p 109.66 126.93  0.47 5601.3  0.03 122.1 -0.019 0.031 -0.129 0.031 0.3 1 position
```

3. Below is the mosaicity histogram for the reflections of the frame. The reflections are sorted into 20 zones, which range from minus the input mosaicity/2 to plus the input mosaicity/2. Each zone represents the shortest angular distance of the center of the reflection from the surface of the Ewald sphere at the end of the oscillation range. Only reflections that are single partials are included in the analysis. A reflection in a negative zones means that the center of the reflection has already passed through the Bragg condition. The fourth column lists the average observed partiality of the reflections in each zone, and the histogram is just a plot of the average observed partiality. In all cases, no matter what mosaicity you have input to Denzo, the histogram should pass through 50% in the zero zone. If the mosaicity you guessed/input was too small, then the histogram will not descend to zero at +mos/2. If the mosaicity you guessed/input was too large, then you will see a histogram which looks like a step function, falling to zero just past the zero zone. In the example below, the mosaicity was chosen correctly (2 x 0.06° = 0.12°), or slightly overestimated (which is the preferred side to err on) but there is some diffuse scattering which accounts for the small tail on the histogram. Note also that this is a particularly nice example because there are many contributing reflections to each zone. If there are only a few reflections, the histogram may be quite choppy. If the histogram does not resemble any of the shapes mentioned here, it may be indicative of motor, spindle, or shutter problems.

```
# refl   zone   av. part.-10...0..10..20..30..40..50..60..70..80..90.100.110
 169 -0.060 - -0.054  0.927  .*************************************
 179 -0.054 - -0.048  1.019  .*****************************************
 155 -0.048 - -0.042  0.929  .*************************************
 161 -0.042 - -0.036  0.897  .************************************
 164 -0.036 - -0.030  1.005  .****************************************
 165 -0.030 - -0.024  1.039  .*****************************************
 176 -0.024 - -0.018  1.040  .*****************************************
 175 -0.018 - -0.012  0.898  .************************************
 176 -0.012 - -0.006  0.700  .******************************
 160 -0.006 -  0.000  0.607  .************************
 195  0.000 -  0.006  0.409  .****************
 192  0.006 -  0.012  0.217  .*********
 188  0.012 -  0.018  0.142  .******
 163  0.018 -  0.024  0.091  .****
 188  0.024 -  0.030  0.072  .***
 162  0.030 -  0.036  0.055  .**
 193  0.036 -  0.042  0.056  .**
 186  0.042 -  0.048  0.042  .**
 167  0.048 -  0.054  0.041  .**
 181  0.054 -  0.060  0.041  .**
```

4. The table below gives a summary of the types of reflections measured from the image. The first column is the type of reflection. The next three columns sort the reflections by strength: above the weak level, saturated (overflows), and below the weak level. Note that weak reflections are NOT ignored or omitted; they are just not used in the calculation of the average profile. Overlap tabulates the number of reflections that are flagged as overlapping (depending on whether you specified overlap spot, guard, or box), background overflow is the number of reflections with unmeasurable backgrounds due to saturation or excessive background editing, and others represent reflections which sit on the edge of the image and are lost but not outside the detector. Outside is a calculation of how many reflections are missing because they landed outside of the scanned image area (i.e. off the IP), for the specified resolution range. Remeasured is the same as total on the last cycle.

type	good intns	ovfl	weak	overlap	backgr ovfl	others	total	outside	remeasured
fiducials	0	0	0	0	1	0	1	0	1
partials	2004	60	1556	0	1	4	3625	850	3625
whole	2739	52	740	0	0	0	3531	273	3531
total	4743	112	2296	0	2	4	7157	1123	7157

5. These are the final statistics for the positional refinement.

```
spots refined: 4855 chi**2/#refl: x 0.70 y  0.78 pred. decrease: 0.000 * 4855 = 0.0
partiality refinement: 3620 chi**2/#refl:   0.80 pred. decrease: 0.000 * 3620 = 0.0
```

Understanding the DENZO .x file

The product of *Denzo* is a file called the .x file, named for the suffix it is usually given. Each file represents the distillation of a single image. The .x file is organized into three consecutive parts: header, listing of reflections, and refined parameters.

Here is an annotated example:

```
ajit alpha toxin                                   title of the frame
a matrix representation of orientation                u matrix orientation
-0.00265575   0.00167628  -0.00863879 -0.069715 -0.985799 -0.152772
 0.00877937  -0.00028329  -0.00281712 -0.289696  0.166551 -0.942517
-0.00067600  -0.00903865  -0.00142214  0.954577 -0.021450 -0.297193
osc. start   osc. end   dist. in pixel units   wavelength   cryst rotz   roty       rotx       mosaicity
 0.00000   1.50000 1332.82227     1.54180   72.1579 178.6024 170.2019 0.2500
these are the actual reflections now....
  1   2   3 4     5      6      7        8   9        10      11    12       13
-27   7   5 1   249.8  231.0  1.82    20.4 0.919    29.6   684.4 0.172 1529.5
-27   7   4 1    68.2   50.5  1.69    19.6 0.920    37.6   702.4 0.208 1119.9
-23   5  15 1   125.2  208.5  1.73    25.5 0.922    49.9   462.7 0.261 1412.3
-22   5  17 1   306.9  301.4  1.65    35.9 0.919    53.8   414.9 0.354 1029.9
-23   5  14 1 1413.6 1409.2  3.28    68.4 0.924    56.7   481.6 0.224 1397.3
(… some lines skipped …)
✳ 0 -12 -32 1 118141  5399  24.77 -10031 0.997 1372.8 1196.3 0.191 3094.7 .
```

1,2,3 h,k,l not reduced **4** flag 0 full, 1 partial **5 I** by profile fitting **6 I** by profile summation **7** χ^2 of profile fitting **8** σ of **I** **9** cosine of incidence angle at detector **10** x coordinate of predicted centroid, in pixel units **11** y coordinate of predicted centroid, in pixel units **12** Lorentz, polarization, obliquity factor **13** strength of averaged profile, arbitrary units

In columns **5**,**6**, and **8** the intensity of very strong reflections (i.e. overflows) will be written as an integer, rather than as a floating point number - see example on line ✳ above. Scalepack understands this *Denzo* convention.

```
999 17 17
11111111111111111
11111111111111111
11111111111111111
11111111111111111
11111100200111111
11111022222011111
11110222222201111
11110222222201111,
11112222222221111.
11110222222201111
11110222222201111
11111022222011111
11111100200111111
11111111111111111
11111111111111111
11111111111111111
11111111111111111
```

999 → flag for end of reflection list, 17, 17 → size of the integration box (left)

this is a saved spot profile for use in the display program if you want to overlay the final predictions on the original diffraction image. The "rectangle" is the box, 1's represent the background area 0's the guard area, and 2's the spot area

```
error density 0.750
error positional 0.030
sector  1
raw data file'alpha2_###.image'
profile fitting radius  40.00
overlap spot
monochromator  0.111
oscillation start  0.00 end  1.50
wavelength 1.54180
mosaicity  0.250
resolution limits 100.0  3.8
spindle axis  0  0  1 vertical axis  1  0  0
space group h32
unit cell 108.740 108.740 108.740  89.433  89.433  89.433
crystal rotx 170.202 roty 178.602 rotz  72.158
cassette rotx  -0.16 roty  0.00 rotz  0.00
distance 199.92
x beam  89.194 y beam  89.582
film rotation  0.000
radial offset -0.202 angular offset -0.144
crossfire y 0.138 x 0.132 xy 0.000
```

these are the final *Denzo* parameters as refined from this image

useful for restarting data processing..

if clipping out, may start from here

Input Formats and Conventions

The input to *Denzo* consists of:

- **the digitized diffraction image** - a disk file of optical density readings from a detector. Data are organized as unformatted files. Headers, if any, are ignored. One record corresponds to a vertical or horizontal stripe (depending on the detector) one pixel wide, with one or two bytes per pixel. Pixel values are interpreted as integers which have no sign to them. The orientation conventions used by various detectors and the *Denzo* convention are discussed below.
- **the control data** - a series of keywords followed by modifying words and/or some numbers and a few command words. These are usually written in a command file (macro file), although they can be typed from the keyboard.

All data are read in free format. Capital letters are converted to lower case unless in single quotation marks, so one should be aware when inputting case-sensitive UNIX file names. Some keywords contain a space in their name, for example `Y SCALE`. This space must be maintained and cannot break the line. Where a word or a number modifies a keyword, a space, comma, left or right parenthesis, carriage return or equal sign must separate the two. These characters are recognized as *separators* in *Denzo*. Comments can be put in square brackets `[as in this example]`. If you wish to include a *separator character* in a datum, for example in a title or in the name of a file, the whole datum must be put within single quotes - the title `'my crystal'` must be enclosed in single quotes due to the space between the words “my” and “crystal”. Similar problems crop up with file names. When in doubt or if the program says it can't read the data file, put the file names in single quotes.

A different type of parser pitfall is discussed for the keywords `END FIT` and `END FIX`. See the documentation under the keyword `FIT`.

Most of the `KEYWORDS` in *Denzo* require that a `modifier` and/or a `value` follow. For example, in `CRYSTAL` `rotx -6.012` the keyword `CRYSTAL` is followed by the modifier `rotx` which is in turn followed by the value `-6.012`. Floating point and integer data should follow the FORTRAN convention. Text strings are limited to 80 characters. When an uninterpretable input is encountered, the program produces an error message and continues. Omitting a necessary input causes the program to write a warning to the output file and although the program may continue it is likely to stop with an error message or maybe crash.

Although there are a lot of keywords in *Denzo*, the program provides defaults for most of the critical ones that are not specific to the crystal itself. These are listed in the documentation for each keyword in this Chapter and a more concise tabulation is presented in *List of default parameters* appendix, just after the keywords list. Keep in mind that **just because *Denzo* provides a default doesn't mean that it is best to use that default**. The sample command files in Chapter *Denzo Session* and the listing in *List of default parameters* appendix give a listing of the keywords you should ordinarily state in your command files.

Denzo Orientation Conventions

The standard display and processing convention for *Denzo* is to show the diffraction pattern such that as you view the image from the crystal side of the detector, the spindle axis is horizontal and coming in from the right hand side. What follows is an explanation of how this corresponds to the detector data storage convention.

Detector Conventions

A detector writes data to a file as a series of lines. *X* coordinate is defined as line number times pixel (RASTER) size and *y* coordinate being pixel position within the line times pixel size in *y* direction. The outer corner of first pixel in data file begins at $x = 0$, $y = 0$. Unfortunately, correspondence of data file convention to the laboratory frame of reference is not standardized among detectors manufacturers. Thus, there are a total of 8 possible combinations of *x* and *y* conventions. They are shown below:

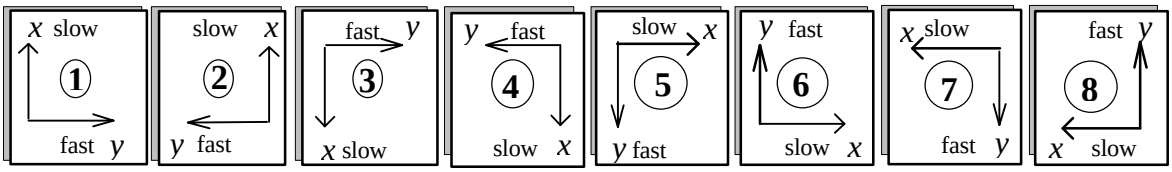


Figure 5: The eight possible data file conventions.

As one might guess, detector manufacturers have managed to use 7 of these 8 conventions. Only 1 left, guys! The table below gives some examples of the conventions used by the major detectors supported by *Denzo* (for detailed list see *Detector Specifications* appendix).

Detector	Convention	y scale
RaxisIIc	1	+
upside down Kodak	2	-
Kodak, DIP100	3	-
Fuji_8	4	+
DIP 3000	5	+
Mar	6	-
DIP320	7	-

What really counts is not the scanning direction, but rather how the detector writes the data stream into the data file, which thus makes the computer record of the exposed image. Closer inspection of Figure 5 above should convince you that **there are two possible "hands"** to the conventions. These are grouped as {1, 4, 5, and 8} and {2, 3, 6, and 7}. To use an X-ray

film analogy, this is equivalent to looking at the film either from the back side, i.e. towards the X-ray source, or from the crystal side. *Denzo* handles this aberration by changing the sign of the `Y SCALE` keyword from positive to negative. If this is not done, then the assignment of the **Friedel mates (Bijvoet differences)** will be inverted, and thus the hand of the structure will be similarly inverted.

Spiral scanners Obviously, spiral scanners like the Mar scanner or some of the Mac Science scanners do not scan the IP in an orthogonal pattern. However, the controllers for these detectors convert the spiral data into Cartesian coordinates by using an interpolation algorithm. The conventions listed in the table above for these detectors are based on how the data is written in the file.

Camera Conventions

It is inconvenient and non-intuitive to work with images in the data conventions, because we are used to thinking about diffraction geometry in terms of the x-ray beam, the spindle axis, and the laboratory frame of reference (horizontal, vertical). There are two possible conventions within this "intuitive" set:

Gravity/beam system

In this system, the X-ray beam is defined as being perpendicular to the direction of the force vector due to gravity ("down"). The beam is the *z* axis, gravity is parallel to the *y* axis, and the *x* axis is perpendicular to both of these. In terms of crystal orientation rotations

- `rotz` would denote rotations or the crystal around the **beam** axis,
- `roty` would denote rotations of the crystal about the **vertical** axis and
- `rotx` would denote rotations of the crystal about the **horizontal** axis.

This convention has the advantage that it does not change with different camera spindle geometry. In addition, it corresponds directly to the common orthogonal detector directions, in that the detector directions x and y on the film or IP are perpendicular to the x-ray beam. A disadvantage of this system is that the x axis may or may not correspond to the spindle axis, depending upon the χ (chi) and Ω (omega) setting angles of the 3-axis goniostat. For the purposes of aligning your crystal, this makes life more complicated, as we are used to thinking about aligning crystals by moving the arcs on a goniometer head. For χ equal 90 or 270 degrees and Ω equal to zero, however, the spindle axis is parallel to the x axis and thus `rotx` corresponds to rotations about the spindle axis.

Although this system will not (formally) work in Outer Space (where the gravity vector is essentially zero), nor in the event that the x-ray beam is vertical, the fact remains that (so far) all macromolecular crystallography is done with horizontal x-ray beams on the surface of good old planet Earth, and thus it is a viable option. While *Denzo* does not use this convention, *XdisplayF* does. Hey, life is messy...

Spindle/beam system

The spindle/beam system is the *Denzo* crystal and cassette orientation convention.
In this system,

⇒ the **z** axis is again parallel to the **beam**,
⇒ the **x** axis is parallel to the **spindle** axis and
⇒ the **y** axis, also termed the **"vertical"** axis (for want of a better word), is perpendicular to the spindle and beam axes.

In terms of crystal orientation rotations:

- `rotz` would again denote rotations of the crystal around the **beam** axis,
- `rotx` would denote rotations of the crystal about the **spindle** axis and
- `roty` would denote rotations about the axis perpendicular to the beam and the spindle.

The advantage of this system is that it is very intuitive to the crystallographer. In addition, with Eulerian χ values of 0, 90, 180 or 270 degrees the spindle and vertical axes are parallel to the x or y axes of the detector. The main disadvantage of this system is that it is dependent on knowing the geometry of the camera.

☛ While the crystal and cassette orientations follow the spindle/beam convention for *Denzo*, the beam, box, box printout, spot, margin, film width and length in the *Denzo* log file follow the data convention.

Determining the values of FILM ROTATION and Y SCALE

As discussed above, the sign of `Y SCALE` depends upon the data storage convention of the instrument. As was shown in the table under the Figure 5 in Detector Conventions analysis, to date about 50% of the detectors have a negative values for `Y SCALE`. The default values in *Denzo* are good for all automatic detectors (MacScience DIP2000 series, R-axis, Mar) in their standard configurations.

`FILM rotation` depends on the spindle orientation. The standard *Denzo* convention is to display the image such that looking along the x-ray beam, the spindle axis is horizontal and comes in from the right hand side (Eulerian χ=90). In such situation `FILM rotation` = 0. If the spindle is at any other χ value `FILM rotation` = χ - 90

☛ On the Raxis, the data is *pre-rotated* by 90 degrees when written to the data file. This is a convention specific to this machine. Thus, for Raxis images, `FILM rotation` = χ - 180 *(Raxis only)*

☛ If the film or IP is rotated about the beam axis prior to scanning, then the value of film rotation will have to be determined by noting the angle of rotation.

Alphabetical Denzo Keyword Descriptions

This is a listing of all of the *Denzo* keywords and their modifiers, if any:

AIR ABSORPTION LENGTH
ANGULAR OFFSET
BACKGROUND
 radius
 elliptical
BAD VALUE
BOX
CALCULATE
CASSETTE
 flat
 radius
 rotx, roty, rotz
CELL
COUPLING
CROSSFIRE
 x, y, xy
CRYSTAL
 rotx, roty, rotz
DISTANCE
EIGENVALUE FILTER
END FIT
END OF JOB
END OF PACK
ERROR
 density
 overload
 partiality
 positional
 systematic
 y/x factor
FIDUCIALS
FILM
 absorption
 length
 output file
 rotation
 width
FIT/FIX
 a* b* c*
 alpha*, beta*, gamma*
 crystal rotx roty rotz
 cassette rotx roty rotz
 distance
 x beam
 y beam
 y scale
 skew
 film rotation
 radius
 crossfire x y xy
 all
FORMAT
see *Detector Specific Info*
GO
IBOX
IGNORE
IMAGING PLATE
INPUT / @
INTENSITY AVERAGING WINDOW
LAMBDA
LATTICE
LEFT MARGIN
LIST
LONGEST VECTOR
MONOCHROMATOR
 filter
 graphite
MOSAICITY
NO BEAM
NUMBER OF RECORDS
OMIT
 radius
 elliptical
ORIENTATION AXIS 1
ORIENTATION AXIS 2
OSCILLATION
 start
 end
 range
 step
OVERLAP
 spot
 guard
 none
OVERLOAD VALUE
PLATE ROTATION
PEAK SEARCH FILE peaks.file
POLARIZATION
PRINT
 chi square; no chi square
 fitted
 hkl
 memory; no memory
 profiles; no profiles
 shifts; no shifts
 solver; no solver
 statistics; no statistics
PROFILE FITTING RADIUS
QUIT
RADIAL OFFSET
RASTER
RAW DATA FILE
RECORD LENGTH
REFINE PARTIALITY
REFINE NO PARTIALITY
REJECT
 fraction
 cutoff
 slope
 systematic
RESOLUTION LIMITS
SECTOR
SKEW
SPACE GROUP
SPINDLE AXIS
SPOT
 radius
 elliptical
START REFINEMENT
STOP
SWAP BYTES (obsolete)
TITLE
TOP MARGIN
UNIT CELL
USE PARTIALS POSITION
USE BEAM
USE FIDUCIALS
USE NO FIDUCIALS
USE PARTIALS POSITION
USE PARTIALS NO POSITION
VERTICAL AXIS
WAVELENGTH
WEAK LEVEL
WRITE PREDICTIONS
WRITE NO PREDICTIONS
X BEAM
Y BEAM
Y SCALE
YORK OUTPUT FILE

AIR ABSORPTION LENGTH

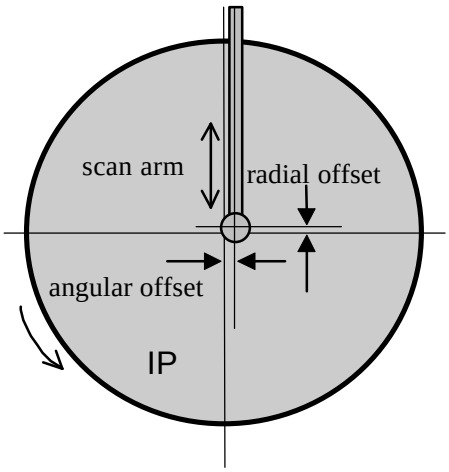
Length of air or helium, in mm, over which the scattered X-rays are attenuated by a factor of e (2.718)

format	AIR ABSORPTION LENGTH *value_(mm)*	
default	AIR ABSORPTION LENGTH 860	
example	AIR ABSORPTION LENGTH 200	[for 2.5 Å X-rays]

ANGULAR OFFSET

This is one part of the alignment correction needed for the spiral MacScience and MAR scanners. It is the distance, in mm, that the reading head is offset in the non-radial scanning direction from the center of the imaging plate. See also RADIAL OFFSET.

format	ANGULAR OFFSET *value_(mm)*
default	ANGULAR OFFSET 0.0
example	ANGULAR OFFSET 1.2



BACKGROUND

Defines the shape and size of the background area where the background measurement takes place. Usually defined to be outside of the spot area. However, when no background is specified, the default area is the area just outside of the spot boarder.

modifiers	radius	defines the background to be circular with a radius value, in mm. The center of the background area and of the box coincide.
	elliptical	defines the background to be ellipsoidal with characteristic major and minor semi-axis lengths (in mm), along with the clockwise angle between a horizontal line and the major semi-axis. Refer to the diagram under the keyword SPOT.

formats	BACKGROUND radius *value_(mm)*
	BACKGROUND elliptical *value-major_(mm) value-minor_(mm) value-angle_(degrees)*
default	a ring just outside of the spot border

examples	BACKGROUND `radius 0.45`
	BACKGROUND `elliptical 0.80 0.70 0.0`

BAD VALUE

The value in the image file which does not represent a measurement. Typically 0. When set to 40000 it disables 0. Used only on DIP series scanners to process small molecule data.

BOX

Defines the size of the box where both the spot and the background are measured, given as two lengths, in mm. If the BOX size is larger than the average separation between spots (i.e. the boxes overlap), then the program will be slowed down by the spot overlap calculation. The current limit on the box size is 90 pixels in each direction, for a total of 8100 pixels.

☛ The box size can alternatively be specified by the IBOX, where the size is given in pixels instead of mm.

modifier	`print`	tells the program to print the spot definition at the beginning of every refinement cycle (i.e. for every GO statement)
formats	BOX *value1_(mm)* [slow scan direction] *value2_(mm)* [fast scan direction]	
	BOX `print` *value1 value2*	
default	none	
example	BOX `2.4 2.4`	

CALCULATE

This command tells the program that it is the last cycle, when the spot intensities are calculated and written to the film output file. It is followed by the command GO. After the cycle both the RAW DATA FILE and the FILM OUTPUT FILE are closed. Signals the SECTOR argument to be incremented by 1.

CASSETTE

Defines the orientation and the shape of the detector, historically, a film cassette.

modifier	`rotz, roty, rotx`	the three angular deviations from zero
	`radius`	radius of the film in a curved cassette
	☛	Radius of curvature does not have to be equal to the film-to-crystal distance, and may be different than the radius of the cassette due to the thickness of the film or imaging plate.
	`flat`	turns off radius
formats	CASSETTE `rotx` *value1* `roty` *value2* `rotz` *value3_(degrees)*	
	CASSETTE `radius` *value1_(mm)*	
default	all angles = `0.0`	
example	CASSETTE `rotx 2.0 roty -1.0`	

☛ You rarely have to enter these values since the program can refine them once the crystal parameters are established. However, some computer time can be saved if previously determined values are used. Usually they change significantly only after the camera is realigned. There is no limit on values.

☛ **The cassette rotations are the primary means by which a non-zero 2θ angle is specified** (along with DISTANCE, X BEAM and Y BEAM). For example, a 2θ setting of, say, +20 degrees (R-Axis convention), would be described by CASSETTE `rotx -20.0` (note change in sign). The DISTANCE will have to be changed as well, since it is the distance from the crystal to the point where it intersects the IP surface. This can be calculated by the formula: Distance (*Denzo*) = distance (R-Axis) / cos(2θ). X BEAM and Y BEAM are the position of the direct beam on the IP during exposure, not at 2θ = 0. For large values of 2θ the Y BEAM may be off the detector. The correct value may be negative or more than 200 in such cases.

☛ CRYSTAL `rotz` and CASSETTE `rotz` are strongly correlated and **cannot be refined simultaneously**.

CELL

Alias for the keyword UNIT CELL

COUPLING

This value reflects how fast the detector moves during exposure for Weissenberg images. Given as mm per degree. This is the **inverse** of the Photon Factory parameter degree/mm, and is only useful for Weissenberg images.

format	COUPLING *value_(mm/degree)*
default	0.0
example	COUPLING 0.057

CROSSFIRE

Crossfire is a measure of the X-ray beam divergence and focusing as it leaves the collimator and illuminates the crystal. Crossfire, being a symmetric tensor, has x, y, and xy components. It affects the prediction of partial reflections and their positions, not their angular width. It is expressed as angular divergence of the beam. The default value is zero crossfire, i.e. a perfectly parallel beam or a beam focused on the detector.

modifiers	x	angular spread of the beam in the x direction
	y	angular spread of the beam in the y direction
	xy	correlated xy component of the beam spread. This tends to be zero within error.

format	CROSSFIRE x *value1_(arbitrary_units)*	
	CROSSFIRE y *value2_(arbitrary_units)*	
	CROSSFIRE xy *value3_(arbitrary_units)*	
defaults	0 0 0	
example	CROSSFIRE x 0.023	*value taken from previous refinement*
	CROSSFIRE y 0.345	
	CROSSFIRE xy 0.0004	

	CROSSFIRE x 0.023 y 0.345 xy 0.0004

CRYSTAL

Defines the crystal orientation for the calculation of predicted spots.

Modifiers	rotz, roty, rotx	the three angular deviations from the reference orientation specified by SPINDLE AXIS and VERTICAL AXIS. See the diagram and further description in *Denzo orientation conventions*.
format	CRYSTAL rotz *value1* roty *value2* rotx *value3_(degrees)*	
defaults	0 0 0	
example	CRYSTAL rotx 2.34 roty -4.566 rotz 34.560	

DISTANCE

The distance from the crystal to the beam spot on the film, in mm, along the X-ray beam path, regardless of the 2Θ angle.

format	DISTANCE	*value_(mm)*
default	DISTANCE	75.0
example	DISTANCE	155.453

EIGENVALUE FILTER (USE IT WITH GREAT CAUTION)

When attempting to simultaneously refine two or more highly correlated parameters with *Denzo*, an eigenvalue filter is triggered and an error message appears in the log file stating that the parameters being fit were too highly correlated to be fit simultaneously. The eigenvalue filter value provides a way of defeating this check, by lowering the value. Unfortunately, just because you change the value doesn't mean that the correlated parameters are going to become uncorrelated, so changing the value is unlikely to help you and may lead to some horrible result.

format	EIGENVALUE FILTER *value*	
default	0.001	
example	EIGENVALUE FILTER 0.0005	

END OF JOB

Signals the end of the *Denzo* job. Placed at the end of the command file or used to exit from *Denzo* during manual fitting and indexing.

END OF PACK (REDUNDANT)

Signals the SECTOR argument to be incremented by 1. The CALCULATE GO statement does now the previous work of END OF PACK.

ERROR

The keyword ERROR and its modifiers defines the model for the error in the measurement of the spot positions. **The default values should be adequate in most cases** and you should not

worry too much about them. The error model does, however, directly affect the χ^2 values printed out at the end of each cycle of refinement. Since this is usually the only check you can reasonably make as to the success of your data reduction in *Denzo*, it is wise to know something about how these are calculated. The names of the modifiers contributing to the error definition should not necessarily be taken literally, since they are short hand notation for complicated effects on the accuracy of the measurement. The model has been developed and black boxed (i.e. given the Z.O. Good Housekeeping seal of approval) to give a realistic estimate of the weight each spot should contribute to the refinement of the crystal and camera coordinates. This model gives χ^2 values close to one for good (non-mosaic) crystals. For mosaic crystals either error positional needs to be increased or higher values than 1 of χ^2 should be expected.

modifiers

`positional` The **minimum** error, in mm, in the positional measurement. It is usually a fraction of the pixel size, typically one half or one quarter of a pixel. Its squared value is added to the other components of the error estimate. Enlarging this value will lower your χ^2 values, but will not necessarily improve your data.
☛ if you have to increase this value much for a well-tested instrument like MacScience or R-axis, check the instrument alignment.

`density` Models the error measurement of the optical density of the film. The error equals the square root of the corrected reading times the density value. This is used only to estimate the error in spot positions and in the definition of strong reflections.
☛ the error density is affected by the voltage on the photomultiplier. The higher the voltage, the higher the error density, with a very steep dependence.

`overload` Refers to the increase in the modeled positional error due to the existence of pixels with intensity overflow. For every overflowed pixel the modeled error squared is increased by the overload value squared.

`y/x factor` This reflects the increase in the positional error of the spot due to its being close to the oscillation axis. The error estimate is by 1 + the ratio of y/x factor to the absolute value of the tangent of the angle between the oscillation axis and the line connecting the spot with the beam spot. This prevents the spot distortion which occurs close to the oscillation axis from affecting the refinement very much.

`partiality` the expected error of the partiality determination The value for the error is expressed as a decimal, e.g. ±10% would be written 0.1.

`systematic` the weight given the positional refinement compared to the partiality refinement when refining crystal orientation parameters. The higher the value, the more weight is given to the positional refinement. Typical values range from 1 to 20. A value of 5 seems to work almost all the time.

format

`ERROR modifier1 value modifier2 value ...`

defaults

```
overload 0.005
density                          detector and wavelength dependent
positional 0.030
y/x factor 0.1
partiality 0.10
```

	`systematic 5.0`
example	`ERROR systematic 5.00 partiality 0.100 positional 0.050`

☛ **Avoid modifying the error parameters other than positional and density**. Moreover, error density should be constant for a detector (site) and changed only with caution.

FIDUCIALS

Defines the position of the fiducial marks on the film relative to the beam spot, before `FILM rotation` and `Y SCALE` are applied. Given as a series of *x*,*y* coordinates, in mm. These days this is only used with Photon Factory Weissenberg images.

format	`FIDUCIALS` *`xvalue_(mm) yvalue_(mm) xvalue_(mm) yvalue_(mm) ...`*
default	none
example	`FIDUCIALS -90.0 -90.0 90.0 -90.0`

FILM

This keyword tells the program about the characteristics of the data collection medium. Identical to the keyword `IMAGING PLATE`.

modifiers	`absorption`	Total absorption of the film. Default is 100 for image plates
	`length`	Distance from the top of the detector to the last useful pixel in the **fast** direction. Depending on the detector, on the computer display it could be the apparent width of the detector.
	`width`	Width of the detector in the **slow** direction. Depending on the detector, on the computer display it could be the apparent height of the detector.
	`rotation`	Describes the orientation of the **spindle** relative to the detector. The value of the `FILM rotation` is function of which file storage convention (see figure 5 in *Detector Conventions analysis*) is being used. For manual scanner, adding 180° to film rotation can compensate for putting plate upside-down into scanner. Small changes (refined by *Denzo*) to film rotation can compensate for small rotation slop that is inherent when manually handling plates.
	`output file`	This gives the name of the processed output file

formats	`FILM modifier1` *`value ...`*	
	`output file` format is a string of characters	
defaults	`absorption 100`	
	`length`	*`detector dependent`*
	`rotation`	*`detector dependent`*

	`output file` (none)
examples	FILM `rotation 270.0`
	FILM `absorption 2`

FIT, FIX, END FIT, END FIX

These command words tell the program to FIT (refine) or FIX (not refine) a parameter specified by a *Denzo* keyword. The FIT command is followed by the command GO, which calls one cycle of refinement. One GO command calls one cycle of refinement, 5 GO commands call for 5 cycles of refinement, etc. If a keyword parameter is fixed, it will retain its current value through the refinement**. The program assumes that all** of the keyword **parameters are fixed at the beginning of every new frame**.

The effect of these commands is cumulative and carries over through subsequent cycles of refinement. For example, if the command FIT CRYSTAL `rotx roty rotz` GO is given, then on every subsequent refinement cycle (called by the GO statement), the CRYSTAL `rotx roty` and `rotz` parameters will be refined. Typically a refinement starts by fitting just a few parameters, like CRYSTAL `rotx roty rotz`, allowing the refinement to proceed towards convergence for a few cycles, and then fitting more parameters a few at a time. The keyworded parameters that can be FIT are given below.

☛ All of the keyworded parameters can be refined simultaneously, so care should be taken not to fit parameters which are too strongly correlated at low resolution. For example, the crystal to film distance and the unit cell lengths are highly correlated, so one should be allowed to refine before refining the other (usually the unit cell lengths are refined first).

The END FIT and END FIX command words mark the end of a list of fitted parameters. This is especially useful for inputing more parameters after you have called a FIT command.

☛ Consider the following case: you want to refine the crystal rotation values, so you type FIT CRYSTAL `rotx roty rotz`. Now let's say that you also want to enter a new value for the distance before giving a GO command. You might think to try FIT CRYSTAL `rotx roty rotz` DISTANCE `200` GO, but this won't work! Program will not recognize the `200`, and it will give you an error message, plus it will try to refine the distance value as well. The solution is to use the END FIT command, as follows: FIT CRYSTAL `rotx roty rotz` END FIT DISTANCE `200` GO. The same goes for the END FIX command.

The option FIT `cell`, when used in conjunction with the SPACE GROUP or LATTICE command, tells the program to fit the unit cell lengths and angles. The program is smart enough to know which lengths and angles are to be fit, and which are to be fixed. For example, it knows enough to keep all angles at 90 degrees in orthorhombic space group. Likewise, it also knows enough to keep $a = b$ in tetragonal space groups, etc. The LATTICE or SPACE GROUP must be declared before FIT CELL can be used.

In practice **most often option used is** FIT `all`. This will fit all parameters that can be sensibly fit given the circumstances (space group, detector). Fitting all parameters is now stable even at low resolution.

FIT- and FIX- able keywords	`all, a*, b*, c*, alpha*, beta*, gamma*, cell,` CRYSTAL `rotz rotx roty,` CASSETTE `rotz rotx roty, radius,` DISTANCE, X BEAM, Y BEAM, Y SCALE, SKEW, FILM `rotation,` CROSSFIRE `x, y, xy`
format	FIT *KEYWORD* `modifier`

default	all keyword parameters fixed (i.e. type GO and nothing will be refined)
examples	FIT CRYSTAL `rotx roty rotz` FIT X BEAM Y BEAM Y SCALE SKEW DISTANCE GO GO GO

☛ **Do not run alternative cycles of fit and fix commands!** This is the old way and does not take advantage of the fact that the *Denzo* refinement is very stable and can refine many parameters simultaneously. Simultaneous refinement is the best way to avoid getting into a false minima.

FORMAT

A critical keyword. The modifier following this keyword tells the program about a series of defaults to use for the detector parameters. This keyword can thus substitute for many of the parameters called by, for example, the FILM (or IMAGING PLATE) keyword.

A detailed table of all the keywords specified by the format keyword can be found in the *Defaults Values* and *Detector Specific Information* appendixes.

modifiers	see *Detector Specific Information* appendix
format	FORMAT `modifier`
default	this is a required input!
example	FORMAT `dip2000`

GO

This command tells the program to execute one iteration cycle of refinement. After the cycle the fitted parameters will have new values. You can specify more than one GO in one line. This will tell *Denzo* to print results of the fit only for the last cycle from the series of GOS. See also FIT/FIX.

IBOX (REDUNDANT)

IBOX is exactly the same as the keyword BOX, except that instead of giving the dimensions of the box in mm, the dimensions of the box are given in pixels. The I in IBOX stands for *integer*. The maximum dimensions of the box are 90 pixels in each direction.

format	IBOX *`integer integer_(pixels)`*
default	none
example	IBOX `25 25`

IGNORE

Tells the program to ignore a specific region of the X-ray image. For example, this is useful for excluding the shadow due to the beam stop. Don't try to use this for masking out ice rings. That is accomplished using the REJECT `fraction` keyword. It is useful to think of IGNORE as describing a masked-out area of the film. The mask is provided by a series of

geometrical descriptions, followed by their coordinates. For example, the beam stop shadow could be masked out by a combination of a quadrilateral and a circle (i.e. a lollipop shape).

modifiers	`quadrilateral`	A quadrilateral as defined by four *x*,*y* coordinates, in mm, picked in a **counterclockwise** (if `Y SCALE` > 0) or **clockwise** (if `Y SCALE` < 0) direction. The best way to do this is to use the cursor to pick the points off of the display window. The *x*, *y* coordinates in mm are listed in the bottom row of the green box of the display.
	`circle`	defined by the *x*,*y* coordinates of the center, and the radius of the circle, all in mm.
	`triangle`	defined by three *x*,*y* coordinates. Use the display program to pick these.
format	`IGNORE`	`quadrilateral` *`x1 y1 x2 y2 x3 y3 x4 y4`* (all values in mm) `circle` *`x5_(mm) y5_(mm) r_(mm)`* `triangle` *`x6 y6 x7 y7 x8 y8`* (all values in mm)
default	nothing ignored	
example	`IGNORE quadrilateral 50 0 52 0 52 100 50 100`	

IMAGING PLATE (REDUNDANT)

This substitutes 1 for 1 with the keyword `FILM`. *See* `FILM` for all the details.

INPUT / @

This keyword is the same as the `@` keyword. Signals to the program to change reading of the control data to an auxiliary file. At the beginning of the program the auxiliary file is closed, so the keyword `INPUT` has to be followed by the filename of the auxiliary file. The auxiliary file is read until a semicolon is reached (or an end of file mark is detected). The control is then returned to the original input stream. In the case of reaching a semicolon, the auxiliary file remains open, and the next time the keyword `INPUT` appears it must not be followed by a file name or else the program will become confused, since it still wants to read from the open auxiliary file. On the other hand, if control is returned due to an end of file mark, then the next time input is encountered it must have an auxiliary file name next to it. If you give it the same auxiliary file name as before, then it will begin reading the auxiliary file from the beginning. Nesting of auxiliary files is not allowed. Auxiliary files are useful to keep the data common to many films in a separate file.

format	`INPUT` *`filename`*
default	none
examples	`INPUT subbatch1.dat` `@ refine`

INTENSITY AVERAGING WINDOW

Sets the number of reflections in a resolution shell used to define that resolution shell for the purpose of refining partiality. Default is 100.

LAMBDA (REDUNDANT)

X-ray wavelength. Same as keyword **WAVELENGTH.**

format	wavelength in Ångstroms	
default	`1.5418`	[Cu K_α]
example	**LAMBDA** `1.5418`	

LATTICE (REDUNDANT)

Lattice type. Note that this command is superseded by the **SPACE GROUP** command for most routine use.

format:	single letter or single letter with name. Can also specify lattice type. Possible choices are: `P, C, A, B, F, I` and `R`
default:	`P triclinic`
example:	**LATTICE** `P monoclinic`

LEFT MARGIN

The left margin, in mm from the left edge, of useful data. It is defined in the raw data convention, not in the display program convention.

format	**LEFT MARGIN** *value_(mm)*
default	`0.0`
example	**LEFT MARGIN** `2.0`

LIST

This command is normally used in the manual mode. It tells the program to provide a list of the current parameters. This is convenient if you forget where you were in the refinement, or you want to copy down the latest parameters. To use, just type **LIST.**

LONGEST VECTOR

In the autoindexing subroutine of *Denzo*, **LONGEST VECTOR** denotes the longest real-space vector, in Å, the program expects to find in the reduced, primitive, unit cell. If the autoindexing fails, and you are confident of your **X BEAM** , **Y BEAM** and **DISTANCE** values, then this is the next parameter to try changing. Typically you would either input a value close to or somewhat larger (say ≈ 50%) than what you really suspect is the longest vector in the unit cell.

format	**LONGEST VECTOR** *value_(Å)*
default	**LONGEST VECTOR** $\approx \dfrac{\lambda \cdot \text{distance}}{2 \cdot \text{spot radius}}$
example	**LONGEST VECTOR** `300`

MONOCHROMATOR

Defines the polarization of the X-ray beam. Same as the keyword **POLARIZATION.** This correction is not a big deal except for synchrotron radiation, where it is a minor deal unless you are collecting high angle data. The *Denzo* definition of polarization is as follows:

$$\frac{\left(I_s - I_p\right)}{\left(I_p + I_s\right)}$$

where I_S is the intensity of the electric field squared of the X-ray beam along the spindle axis, and I_p is the intensity perpendicular to the spindle axis. Values range from +1.0 to -1.0.

Generally speaking, this is not a number you can measure very easily, and is often calculated theoretically. It is really the responsibility of beam line operators to know this value for their beam lines. Note that the **sign of the correction depends on whether the spindle is horizontal or vertical**. As a rule of thumb, if the polarization correction is > 0, then the intensity of the X-ray scattering is stronger perpendicular to the axis of the spindle than it is to the axis parallel to the spindle. You can check the background of your image to confirm this. Typical values range from ± 0.11 for graphite to + 0.9 for synchrotron beams (the spindle axis is typically oriented at synchrotrons to make the polarization positive). For a **laboratory source**, use modifiers: `graphite` (if graphite monochromator is installed) or `filter` (if X-ray beam is filtered or mirrors are installed).

modifiers	`filter`	No polarization, value = 0
	`graphite`	Reflecting plane perpendicular to the plane formed by the spindle and the beam, CuKa and MoKa radiation. Value of ±0.11 for CuKα or ±0.03 for MoKα.
format	**MONOCHROMATOR** `modifier`	
default	`0.0`	
example	**MONOCHROMATOR** `graphite`	
	POLARIZATION `-0.11`	

☛ The old keyworded values for the synchrotrons are probably wrong now. **Always ask the synchrotron staff what the correct number is.** A good guess for a bending magnet source is polarization 0.9. For a wiggler source it can vary a lot from station to station, in the range of 0.0 to 0.9.

MOSAICITY

Defined in *Denzo* as the rocking angle, in degrees, in both the vertical and the horizontal directions which would generate all the spots seen on a still diffraction photograph. It includes contributions due to X-ray bandwidth, beam crossfire, etc.

format	**MOSAICITY** *`value_(degrees)`*
default	this is a required input!
example	**MOSAICITY** `0.5`

NO BEAM

Tells the program not to use the beam spot in the refinement. If the option USE BEAM has been invoked earlier in the refinement, it should be revoked with the NO BEAM keyword in the later stages.

format	NO BEAM
default	this is the default option. see USE BEAM

NUMBER OF RECORDS

Specifies the size of the data file. Equal to the number of vertical (fast scanning direction) lines. The default value is now 'hard wired' into the FORMAT modifier. In other words, once you specify the detector, the NUMBER OF RECORDS value is input from a look-up table.

format	NUMBER OF RECORDS *integer*
default	comes along with whatever FORMAT is specified

OMIT

Defines the area **outside** a specified circle or ellipse, but **within** the box, to be omitted from any use. By default there is no omitted area.

modifiers	`radius` or `rad`	Defines the omit boundary to be circular with a radius value, in mm. The center of the omit boundary and that of the box coincide
	`elliptical` or `ell`	Defines the omit boundary to be ellipsoidal with characteristic major and minor semi-axis lengths (in mm), along with the clockwise angle between a horizontal line and the major semi-axis. Refer to the diagram under the keyword SPOT.
formats	OMIT `radius` *value*	
	OMIT `elliptical` *valuemajor valueminor valueangle*	
default	no area omitted	
examples	OMIT `rad 0.35`	
	OMIT `ell 0.70 0.60 0.0`	

OSCILLATION

This keyword tells the program that the oscillation method is being used.

modifiers	`start`	Starting oscillation angle
	`end`	Ending oscillation angle
	`step`	Angle between 1st oscillation start and 2nd oscillation start
	`range`	The actual oscillation angle
formats	OSCILLATION `modifier1` *value1_(degrees)* `modifier2` *value2_(degrees)* `...`	

default end = start + range

step = range

examples `OSCILLATION range 1.0 start 30.0`

`OSCILLATION start 30.0 end 31.0`

`OSCILLATION range 1.0 step 0.8 start 30.0`

☛ the first two examples describe the same thing. The third describes a series of 1 degree oscillations with ranges which overlap by 0.2 degrees.

OVERLAP

Overlap defines which pixels of any spot are checked for overlap with the pixels of a neighboring reflection box. The choices are the `spot` pixels, the `guard` pixels (found between the spot circle and the background circle) or `none`. For example, if you specify `OVERLAP spot` and if the spot pixels overlap with the spot pixels of another reflection, then the two reflections are flagged as overlaps.

modifiers `spot`
`guard`
`none`

format `OVERLAP modifier`

default `none`

example `OVERLAP spot`

OVERLOAD VALUE

The lowest optical density reading considered unreliable due to saturation of the film or image plate. The default value corresponds to the full dynamic range being useful with only the highest value being considered overflow. A value that is higher than the maximum possible in the data tells the program to ignore the intensity overflow. The range of the values themselves are determined by the analog to digital converter used to write the data. For 8-bit ADC's (found on film scanners) the highest value is 2^8-1 =255. For the 16-bit ADC's found on most on-line detectors the range is from 1 to 2^{16}-1 = 65535. The particular `OVERLOAD VALUE` that is considered unreliable has been determined for several detectors and is input as the default when you specify the detector using the `FORMAT` keyword. *Denzo* applies the overload table in Mar scanners so that the overload value should represent the highest reliable entry in the Mar overload table.

format `OVERLOAD VALUE` *`integer`* (units are function of detector type)

default separate values for different IP formats, contained in `FORMAT` defaults

example `OVERLOAD VALUE 255` (film)

- ☛ One-of-a-kind detectors as well as the very old Mar scanners may have the wrong default value.
- ☛ The overload value is a data representation value before applying the data format dependent correction for strong intensities.
- ☛ Default values assume the full dynamic range of the instruments (Mac Science, Mar, R-Axis).

PEAK SEARCH FILE peaks.file

Tells the program to perform **autoindexing**. Get slightly confused?

PLATE ROTATION (REDUNDANT)

This is exactly the same as the keyword FILM `rotation.` I recommend you stick with FILM `rotation,` to avoid getting confused with too many aliases.

POLARIZATION (REDUNDANT)

Same as MONOCHROMATOR.

PRINT/ END PRINT

The PRINT command tells the program to print the output specified by subsequent keywords to the log file. It does not affect the results of the refinement. Default values of the print sub options can be of several types. Some options are printed by default at every cycle, some are printed on the last cycle of every film of the refinement, and some are never used unless specified. See specifically the various print descriptions below.

modifiers	`chi square`	print the χ^2 statistics, including the number of spots fitted, the χ^2 of x and y (a measure of how well the spots were fitted based on what sort of error you expected - see also ERROR), and the predicted decrease in the χ^2s after the current cycle of refinement. For those of you who just want numbers, **χ^2's should be close to 1.0** (between 0.5 and 2.0), and the predicted decrease should be near zero, showing that the refinement has converged.
	`fitted`	Print to the log file the observed and fitted profiles of those spots having the specified *hkl* value. No density readings in the background will be printed. By default, no fitted spots printed.
	`hkl`	Print to the log file the density readings and other statistics of those spots having the specified *hkl* value. By default, no *hkl*s printed.
	`memory`	Print statistics of the usage of variables.
	`profiles`	The `profiles` modifier tells the program to print the average spot profile in a specified number of regions to the log file. It is best to refrain from printing profiles until the very last cycle of refinement or else your log file will become truly gargantuan. You can

also specify PRINT `no profiles` at the beginning of your refinement, just to make sure it is **off** until you want to turn it **on**.

`shifts` PRINT `shifts` tells the program to print the value, the change from the previous value, and the estimated error of all the keyworded parameters fitted on a current cycle of refinement. PRINT `no shifts` turns this off.

`solver` This little-used command prints out correlation matrices calculated in the refinement.

`statistics` Print the *hkl* statistics, including the number of whole and partially recorded reflections matched with preds.

`zones` Print out a list of equivalent zones, given as alternative values for CRYSTAL `rotx roty` and `rotz`.
One advantage to invoking this option is to use `rotx`, `roty` and `rotz` values that are more convenient to remember or type. For example, instead of using crystal rot values of, say, -213.6, 137.4, -13.27, you could use the equivalent values of, say, (I'm just making these up) 18.2, -15.4, 63.9. If you wanted to orient your crystal for a particular data collection scheme, these alternative zones could make life easier. PRINT `zones` lists all of these equivalent values.

---

format:

PRINT `chi square` / `no chi square`

PRINT `fitted` ***integer integer integer***

PRINT `hkl` ***integer integer integer***

PRINT `memory` / `no memory`

PRINT `profiles` ***integer integer*** (the first integer specifies the number of equal sized horizontal regions to divide the frame into, the second specifies the number of equal sized vertical regions. The total number of regions is the product of the first and second integers) / PRINT `no profiles`

PRINT `shifts` / `no shifts`

PRINT `solver` / `no solver`

PRINT `statistics` / `no statistics`

---

defaults:

`chi squares` printed after every cycle of refinement

no `fitted` spots printed

no `hkl`s printed

print `memory` statistics on the last cycle

`no profiles` printed except on the last cycle, when a 3 x 3 grid is printed

all `shifts` printed

`no solver`

`statistics` turned on on the last cycle only

examples

PRINT `fitted 12 8 0`

PRINT `profiles 3 3` (tells the program to divide the frame into 9 equal regions and print out 9 average spot profiles).

☛ the more profiles you specify, the larger the log file will be

PRINT `hkl 12 8 0`

☛ You can easily generate gigabytes of log file if you use some of the print options, so be careful!

PROFILE FITTING RADIUS

The radius of the area around each particular spot, in mm, containing neighboring spots, used to calculate the average spot profile. The spot in question is fitted to the average profile of all the spots within the specified radius. Generally, **the radius is set so that spots on roughly 3-5% of the area of the detector are included in the averaging**. Only on the last cycle of the refinement is the profile fitting applied, via the keyword CALCULATE.

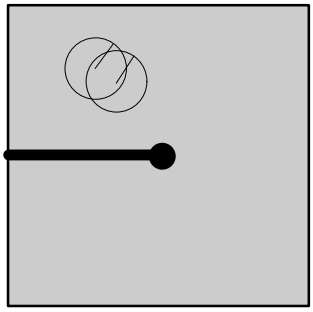
The particular size of the area chosen depends on how much the profiles vary across the film and the density of the spots on the film. For example, if the spot profiles vary a lot across the film, then you would choose a smaller radius. If you have a small lattice and the spots are widely separated, you might want to choose a larger radius.

☛ The calculation of the average profile is a time consuming task, proportional to the number of spots in the profile fitting radius circle, but too small a radius will not capture enough spots and lead to a noisy average profile. Too large a radius will average out significant profile variations.

To see the PROFILE FITTING RADIUS in ***ImageWindow*** of *XdisplayF* click Prof fit R. A thin white circle corresponding to the profile fitting radius will appear and will move around with the cursor; only those spots whose intensity is above the weak level will have predictions displayed in this mode and only these reflections are used in the calculation of the average spot profile (to return to the normal mode, click Prof fit R again).

A good rule of thumb is that the PROFILE FITTING RADIUS should enclose about 10 to 50 spots *whose intensities are above the weak level*. Useful increments of the profile fitting radius are 2.5% of the detector size (e.g. 5 mm for a 200 mm wide IP).

format	PROFILE FITTING RADIUS *value_(mm)*
default	20.0
example	PROFILE FITTING RADIUS 30.0



The circles in the upper left qadrant are two 20 mm profile fitting radii centered on adjacent spots. Each of the hundreds of spots on the film wolud have its own radius on a real image.

QUIT

Exit from program *Denzo*. Same as STOP or a semicolon (;).

RADIAL OFFSET

This is one part of the alignment corrections needed for the spiral MacScience and MAR scanners. It is the radial departure, in mm, of the scanning heads from perfect concentricity with the imaging plate. A perfectly aligned scanner would have a RADIAL OFFSET of 0.0. The radial offset is perpendicular to the angular offset. A diagram of these offsets is shown under the ANGULAR OFFSET keyword.

format	RADIAL OFFSET *value_(mm)*
default	0.0
example	RADIAL OFFSET -0.43

RASTER

Pixel size in the horizontal (slow scanning) dimension

format	real number in mm
default	defined by the values in the FORMAT statement
example	RASTER 0.1

RAW DATA FILE

This defines the detector data file name template. If the file name contains a run of # characters, then they will be substituted with the numerical value of the SECTOR argument. This allows for easy processing of a large series of sequential images.

format	RAW DATA FILE *file_name*
default	none
examples	RAW DATA FILE 'bigdisk:[dir.frames]xtal001.osc'
	RAW DATA FILE /vms/data/dir/xtal###.osc
☛	in the first example the single quotes are necessary, or else the directory names in square brackets will be ignored (commented out).

☛ You must not repeat this command before the next GO statement.

RECORD LENGTH

Defines the number of pixels in the data record. It is equal to the number of horizontal (slow scanning direction) pixels.

format	RECORD LENGTH *value_(pixels)*
default	FILM length / (RASTER / Y SCALE) (this is defined by the FORMAT statement)
example	RECORD LENGTH 2000

REFINE PARTIALITY / REFINE NO PARTIALITY (DO NOT USE!)

Tells the program to determine the fraction partiality of the reflections on the image. It is critical that this be on during the refinement of the crystal rotation parameters, since CRYSTAL rotx and roty cannot be determined without knowing the partiality of the reflections on the image.

format	REFINE PARTIALITY
default	REFINE PARTIALITY

☛ **Do not use** `REFINE NO PARTIALITY` ! (see theoretical explanation in *Refinement: Using the Fit Command* chapter).

REJECT

Describes a condition whereby a reflection is rejected by *Denzo*. This is especially useful for eliminating spurious measurements due to ice rings.

modifiers	`fraction`	The fraction of **acceptable** pixels in the background which are needed for the reflection to be accepted. Default = 0.75 (i.e. 75%). **Very sensitive to small changes**, so increment by 1-2% if you are messing with this. To get rid of spots in an ice ring, increase this value by 1-2% until the preds in the ice ring are colored red. After all, the defining feature of an ice ring is a horrible background. The acceptability of a background pixel is determined by the next three modifier words.
	`cutoff`	The **σ deviation** from the best fit to the background plane at which a pixel has a 50% chance of being incorrect and thus rejected. Default is 3σ. The background of a spot can be described by a plane and should be uniform and flat. Sometimes it is not; for example, it may have a slope to it, which can be calculated. If a pixel is still out of the best-fit background plane by (default) 3σ, then it is flagged as a reject. It is best not to mess with this unless you really know what you are doing.
	`slope`	The slope of the background plane above which the entire background and spot are considered unacceptable and rejected. Default is 50 units. Flat = 0, vertical = infinite. It is sometimes useful to increase this to 100 for very low background images.
	`Systematic`	This is a way of changing the background rejection criteria when you have a major problem with **diffuse scattering**. Default is 0.01. The range is from 0 to 1, with 0 meaning rely totally on the cutoff criteria, and 1 meaning don't reject anything. As you can see from the default, you normally use the 3σ cutoff criterion.
format	`REJECT fraction value`	
defaults	`REJECT fraction 0.75`	
	`REJECT cutoff 3.0`	
	`REJECT slope 50`	
	`REJECT systematic 0.01`	
example	`REJECT fraction 0.80`	

RESOLUTION LIMITS / RES

Defines the resolution range used in the program, in Ångstroms.

format	`RESOLUTION LIMITS` *`valuelower valueupper_(Å)`*
default	none
example	`RESOLUTION LIMITS 20.0 2.5`

```
RES 20.0 2.5
```

SECTOR

This is the **counter and range delimiter** for a series of frames when processing in batch mode. The number in the sector argument is substituted for the ### (or #, or ##, ...) variable in the RAW DATA FILE and FILM `output file` commands. Make sure there are enough # marks to accommodate all of the digits in the sector argument, e.g. if you have sector values in the 100's then there must be at least three # marks in the file names specified in RAW DATA FILE and FILM `output file`. The sector value can start at any non-negative integer, and is incremented by 1 every time an END OF PACK statement is reached. The first number in the sector argument is the first frame and the last number is the last frame. See also START REFINEMENT.

format	SECTOR *integer* to *integer*
default	none
example	SECTOR 1 to 50

SKEW

SKEW refers to the **non-orthogonality** of the vertical and horizontal scanning directions, measured in radians. For a perfectly orthogonal system SKEW is defined to be zero. Generally SKEW values will be very close to this number. It can be fitted in the later rounds of parameter refinement.

format	SKEW *value_(dimensionless)*
default	0.0000
example	SKEW 0.00169 (value taken from the erlier refinement)

☛ Spiral scanners and CCD detectors should use the default value of zero, which should not be refined.

SPACE GROUP

The space group symbol or number as given in the **International Tables**. Note that for rhombohedral space groups autoindexing only works for the primitive choice, i.e. H3 or H32, not R3 or R32. The SPACE GROUP must be declared before the FIT `cell` command can work.

☛ *Denzo* does not actually care about the SPACE GROUP per se. It indexes films by Bravais lattice and does not take into account systematic absences arising from two-folds, screws, etc. These are accounted for in Scalepack, when the data are reduced. So for the purposes of *Denzo*, the resulting .x file will be the same whether you declare the space group to be P222 or P212121, or, P3 and P6122.

format	SPACE GROUP modifier
default	P1
example	SPACE GROUP P212121

SPINDLE AXIS / ORIENTATION AXIS 1

The reciprocal lattice vector parallel to the spindle axis. Same as keyword **ORIENTATION AXIS 1**

format	3 integers representing the reciprocal vector
default	`0 0 1` (c* axis)
example	**SPINDLE AXIS** `1 0 0` (a* axis)
	SPINDLE AXIS `1 0 1` (a non-major axis)

SPOT

Defines the shape and area of the region in the box where the spot is to be found.

modifiers	`radius`	Defines the spot to be circular with a radius value, in mm. The center of the spot area and of the box coincide
	`elliptical`	Defines the spot to be ellipsoidal with characteristic major and minor semi-axis lengths (in mm), along with the clockwise angle between a horizontal line and the major semi-axis.
formats	**SPOT** `radius` *`value_(mm)`*	
	SPOT `elliptical` *`valuemajor valueminor_(mm) valueangle_(degrees)`*	
defaults	none	
examples	**SPOT** `radius 0.35`	
	SPOT `elliptical 0.70 0.60 0.0`	
	☛ See diagram (fig. 4) in *Mosaicity and Spot Shape* section	

START REFINEMENT

START REFINEMENT denotes the start of the series of commands given to index the diffraction image. The end of the **START REFINEMENT** loop is denoted by the **CALCULATE GO** statement. The **START REFINEMENT** statement must come after the **SECTOR** *`x`* `to` *`y`* statement, and the loop is run for every sector value.

format	**START REFINEMENT**
default	this is not a default
example	☛ see the example command file in *Finishing Up* section

STOP (REDUNDANT)

This command stops the *Denzo* job. Same as `QUIT` or a semicolon (`;`).

SWAP BYTES (OBSOLETE)

All *HKL* programs now automatically recognizes swapped data.

TITLE

Data set title

format:	string. If it contains any spaces, commas, equal signs, or brackets (*DENZO*-recognized separators), it must be enclosed in single quotes.
default:	none
example:	`TITLE 'my crystal'`

TOP MARGIN

The top margin, in mm from the top edge, of useful data, i.e. the beginning of every record. Do not set the top margin to be larger than the film length. Note that commands like top margin are working in the data coordinate system. It is very important to check the results of these commands in the display window.

Format	`TOP MARGIN` *`value_(mm)`*
default	`0.0`
example	`TOP MARGIN 2.0`

UNIT CELL

Unit cell parameters in either real or reciprocal space.

format	`UNIT CELL` a* b* c* α* β* γ* or `UNIT CELL` a b c α β γ. Lengths in Ångstroms or reciprocal Å, angles in degrees.
default	none
example	`UNIT CELL 120.92 120.92 131.65 90 90 90`

☛ If the space group has been declared, the program is smart enough to know which cell lengths should be equivalent, and which angles are variable, and it will hold those lengths equal or fixed throughout the refinement.

USE BEAM

Tells the program to use the beam spot in the refinement. This keyword has the opposite effect of the keyword **NO BEAM**. This is useful in the initial steps of the refinement, when not too many spots are matched by preds.

modifier	`only`	Tells the program to use only the beam spot in the refinement. Has the effect of turning off the **RESOLUTION LIMITS** keyword.
format	**USE BEAM** **USE BEAM** `only`	
default	not the default option. See **NO BEAM.**	

USE FIDUCIALS / USE NO FIDUCIALS

Tells the program to use the **fiducials** in the refinement. Turned off with **USE NO FIDUCIALS.** Fiducial positions are entered with the keyword **FIDUCIALS.**

modifier	`only`	Tells the program to use only the fiducial positions in the refinement.
format	**USE FIDUCIALS** **USE FIDUCIALS** `only`	
default	not the default. Default is **USE NO FIDUCIALS.**	

USE PARTIALS POSITION / USE PARTIALS NO POSITION

Normally, data reduction programs use only fully recorded reflections in the refinement of the crystal and detector positional parameters. The keyword **USE PARTIALS POSITION** tells the program to use a reflection that is predicted to be only partially recorded in the positional refinement. A spot is predicted to be partially recorded based on the values entered for the oscillation range, as well as the crystal mosaicity and beam crossfire. This is a very useful feature of *Denzo*, because it allows one to refine the orientation of the crystal even if none of the reflections are fully recorded. This occurs if one collects either very thin oscillation sectors (in an attempt to minimize background) or has a crystal of relatively high mosaicity, or some combination of both. **USE PARTIALS NO POSITION** is mostly of historical interest.

default	**USE PARTIALS POSITION**
examples	**USE PARTIALS POSITION** **USE PARTIALS NO POSITION**

VERTICAL AXIS / ORIENTATION AXIS 2

The reciprocal lattice vector perpendicular to both the spindle axis and the X-ray beam. Equivalent to **ORIENTATION AXIS 2**.

format	3 integers representing the reciprocal vector
default	`1 0 0` (a* axis)
example	**VERTICAL AXIS** `0 1 0` (b* axis)

WAVELENGTH

X-ray wavelength. Same as **LAMBDA**.

format	**WAVELENGTH** *value_(Ångstroms)*
default	`1.5418` (CuK$_\alpha$)
example	**WAVELENGTH** `1.5418`

WEAK LEVEL

This is the intensity rejection level, expressed as an I/σ ratio. Only strong spots with an integrated intensity (i.e. I) above the **WEAK LEVEL** times the intensity error measurement (i.e. σ) are used in the positional refinement and as reference profiles in profile fitting.

☛ The intensities of all reflections determined to be *weak* are still measured and included in the .x file. **They're just not used in the profile fitting**. You can use Prof fit R in the *XdisplayF* window to see which reflections are above the weak level.

format	**WEAK LEVEL** *value_(dimensionless)*
default	`5.0`
example	**WEAK LEVEL** `5.00`

WRITE PREDICTIONS / WRITE NO PREDICTIONS

This tells *Denzo* to write a file called `hklpredictions`. This file contains the positions of the predicted reflections, and is read by *XdisplayF* to update the predictions in the display. Once the flag is turned on, the file is updated every time a new cycle of refinement is executed. This way you can follow the course of the refinement in the display program. Turned off with **WRITE NO PREDICTIONS**.

format	**WRITE PREDICTIONS**
default	**WRITE NO PREDICTIONS**
example	**WRITE PREDICTIONS**

X BEAM

Defines the distance from the edge of the data to the beam spot, in mm. You can use coordinates of the cursor in the green box of the *XdisplayF* window to find the beam position.

Format	X BEAM *value_(mm)*
default	FILM width/2
example	X BEAM 98.961

Y BEAM

Defines the distance from the edge of the data to the beam spot, in mm. Again, use coordinates of the cursor in the green box of the *XdisplayF* window to find the beam position.

format	Y BEAM *value_(mm)*
default	FILM length/2
example	Y BEAM 100.146

Y SCALE

Y SCALE in *Denzo* serves two purposes. First, it is a dimensionless correction factor for anisotropy in the pixel dimensions. It is applied to the fast scanning, or *y*, dimension of the pixel. If the pixel is larger in *y* than in *x*, then Y SCALE is smaller than 1.

☛ It is possible to create a data file which has the inverted direction of scanning along one of the axis - similarly as in case of film scanned from back side. *Denzo* handles this aberration by changing the sign of the Y SCALE keyword to negative. If this is not done, then the assignment of the Friedel mates (Bijvoet differences) will be inverted, and thus the hand of the structure will be similarly inverted - see *Determining the values of* FILM ROTATION *and* Y SCALE section in *Denzo documentation* chapter.

☛ The sign of Y SCALE is a detector property, while the absolute value of Y SCALE seems stable for all detectors. Once the correct sign has been determined, the calibration of Y SCALE is done by refining with a high symmetry crystal (trigonal, tetragonal, hexagonal, or cubic).

☛ for CCD and spiral scanners the value of Y SCALE should be exactly ± 1 and not refined !

format	Y SCALE *value_(dimensionless)*
default	detector dependent, defined by the FORMAT statement
example	Y SCALE 1.03152

YORK OUTPUT FILE

Substitutes for the OUTPUT FILE command, and tells the program to write the *'.x'* file in the **Eleanor Dodson** York file format. Note that Scalepack reads the *Denzo* YORK OUTPUT FILE as format denzo_york1.

Denzo Appendixes

Keywords by Category

1. Film Pack Parameters: Defined at the beginning of each job

TITLE
UNIT CELL
LATTICE
OSCILLATION
MOSAICITY
WAVELENGTH/LAMBDA
SPINDLE AXIS
VERTICAL AXIS
MONOCHROMATOR
RASTER
FILM...
CRYSTAL
CASSETTE
RADIUS
DISTANCE
Y SCALE
AIR ABSORPTION LENGTH
FIDUCIALS
FORMAT

2. Film Parameters: Defined at the beginning of every frame

FORMAT or
FILM...
RECORD LENGTH
NUMBER OF RECORDS
LEFT MARGIN, TOP MARGIN
RAW DATA FILE
SECTOR
X BEAM
Y BEAM

3.Refinement Parameters: Defined at the beginning of every iteration step

BOX
SPOT
BACKGROUND
OMIT
RESOLUTION LIMITS
USE BEAM/NO BEAM
OVERLOAD VALUE
PROFILE FITTING RADIUS
FIT/FIX
CALCULATE
PRINT
ERROR
WEAK LEVEL
GO
END OF PACK
END OF JOB
INPUT/@

Default Parameters

Format - independent keywords with predefined default values:

AIR ABSORPTION LENGTH	860
ANGULAR OFFSET	0.0
BACKGROUND	lowest reading in spot area
CASSETTE rotx roty	0.0
COUPLING	0.0
CROSSFIRE x y	0.0 0.0
CRYSTAL rotz roty rotx	0.0 0.0 0.0
DISTANCE	75.0
EIGENVALUE FILTER	0.001
ERROR	
overload	0.005
positional	0.012
y/x factor	0.1
partiality	0.1
systematic	5.0
FILM rotation	0.0
FIX	all
I/O BUFFER	full file size
LAMBDA	1.5418
LATTICE	P
MONOCHROMATOR	graphite
NO BEAM	
OVERLAP	none
PLATE ROTATION	0.0
PRINT chi square	
PRINT no profiles	
PRINT no solver	
PRINT shifts	
PRINT statistics	
PROFILE FITTING RADIUS	20.0
RADIAL OFFSET	0.0
RADIUS	flat
SPINDLE AXIS	0 0 1
VERTICAL AXIS	1 0 0
WAVELENGTH	1.5418
WEAK LEVEL	5.0
X BEAM	FILM width/2
Y BEAM	FILM length/2

Format - Specific Keywords. Theirs default values can be found in *Detector Specific Information* section:

ERROR density	FILM width	RASTER
FILM absorption	NUMBER OF RECORDS	RECORD LENGHT
FILM length	OVERLOAD VALUE	Y SCALE

Keywords with no defaults:

without which **the program absolutely will not run:**

BOX	OSCILLATION or PRECESSION	RESOLUTION LIMITS
FORMAT	OVERLAP	SPOT
MOSAICITY	RAW DATA FILE	UNIT CELL

required for **interactive processing:** FIT ... GO

required for **batch processing:**

FIT ... GO	END OF PACK	SECTOR
CALCULATE	FILM OUTPUT FILE	TITLE
END OF JOB	INPUT/@	

which may or may not be required, depending on the circumstances:

FIDUCIALS	PRINT hkl	READ UMAT
IGNORE	PRINT profiles	UNIX I/O
OMIT	PRINT solver	LONGEST VECTOR
PRINT fitted	READ AMATRIX	

Recommended Hardware

The operative philosophy behind the programs *Denzo*, *XdisplayF*, and *Scalepack* is that computer hardware is cheap compared to all of the other expenses of crystallography. Modern platforms are supported; old ones are gradually rendered obsolete by their speed, discontinuation, or by other hardware requirements, like memory. Here's a list of current hardware recommendations, but note that this is not meant to be a complete list of computers on which the program runs. It's more to give you an idea of the direction your crystallographic computing should be heading.

CPU	DEC Alpha, HP (fast but not familiar to program authors), Sun, SGI, Pentium Pro
Operating System	OSF, UNIX, VMS
Memory	64 Mb or more (for ≤ 20 cm IP images)
Display	17-21”, 1280x1024 or more, 8 bit, workstation, not X-terminal
Disk	➢ much more that one data set, local to workstation ➢ preferably more than 2-3 data sets
Swap Space	>300 Mb (very important!)

Rock Bottom

Memory	48 Mb
Display	1024 x 768 or X-terminal
Disk	one data set, over a network
Swap Space	>300 Mb (very important!)

☛ You should never economize on swap space because otherwise the programs may get killed randomly by the operating system.

Manual Indexing of Diffraction Images

Manually indexing and orienting diffraction patterns is a tricky business, more so with small unit cells, centered space groups, or lattices containing pseudosymmetry, and less so with high resolution data and simple lattices. Suffice it to say that the more you know about your crystal and detector parameters the easier the job will be. For the purposes of this section, I will assume that you know the following:

- detector type (duh!)
- wavelength
- crystal to film distance within 1% or so
- x beam and y beam within a pixel or two each (take a quickie; direct beam shot if unsure!)
- detector 2θ angle within 1 degree
- film rotation exactly (depends on detector and χ value of spindle)
- crystal unit cell lengths within a couple of Ångstroms, cell angles within a few degrees
- space group/lattice type
- oscillation range
- effective mosaic spread of the reflections. If in doubt, enter in a higher number. A good choice is 0.1 to 0.3 degrees for capillary mounts, and 0.5 to 1.0 degrees for frozen crystals.
- Furthermore, I will assume that the non-2θ dependent cassette rotation parameters are near zero.

All of the non-trivial parameters listed above can be refined if they are close enough and the crystal orientation is figured out to a good first approximation. You will know you have successfully indexed your crystal when the predicted reflections (“preds”) line up on the observed diffraction spots.

☛ It is indeed possible to manually index crystals whose lattice type and unit cell parameters are not known. However, this significantly increases the number of variables you will have to worry about.

The indexing proceeds in two steps. The first is to determine the spindle and vertical axes. Then the crystal rotation angles are determined.

Determining the spindle axis and vertical axis

The **SPINDLE AXIS** is the unit cell vector which is parallel to the axis of rotation of the camera spindle. It is given in *hkl* notation, so that non-major axes can also be specified. For example, the axis 0 0 1 means that the c* axis is parallel to the spindle axis. The **VERTICAL AXIS** is the unit cell vector which is perpendicular to both the **SPINDLE AXIS** and the beam axis. In practical terms, this is simply the axis running in the vertical direction on the screen.

The way to determine each axis is to look at your diffraction pattern for two lines of spots running perpendicular to each other. Call the one more parallel to the spindle axis the **SPINDLE AXIS**, and measure the spacing between the spots using the mouse pointer. If you are lucky, you will have measured spacing corresponding to a major axis, for example the c* axis, and you can enter the command **SPINDLE AXIS** `0 0 1`. The same holds true for the **VERTICAL AXIS**.

But let’s say you are unlucky, or you have a centered space group and you can’t see a major zone. Here’s what you can do:

1. Page through your frames until you find one that does have a recognizable major zone. Depending on your oscillation range, this could mean looking at every 5th or 10th frame, which is not such a bad investment of time.
2. If no major zone is apparent, you will have to figure out which pseudo zone you are looking at. For example, let's say you have a tetragonal, body centered unit cell with a=b=120Å, c=130Å. You measure a spacing of about 90Å. This is clearly the 1 0 1 or 0 1 1 axis. There is another axis 90 degrees away from this one, with spacing 62Å. This must be the 0 2 0 or 2 0 0 axis, and can be entered for the vertical axis. Once the axes have been determined, enter them using the commands **SPINDLE AXIS** 1 0 1 **VERTICAL AXIS** 0 2 0 **GO**.

Determining crystal orientation angles: CRYSTAL rotz roty rotx

The convention used in *Denzo* is as shown in the pictures on the next two pages.

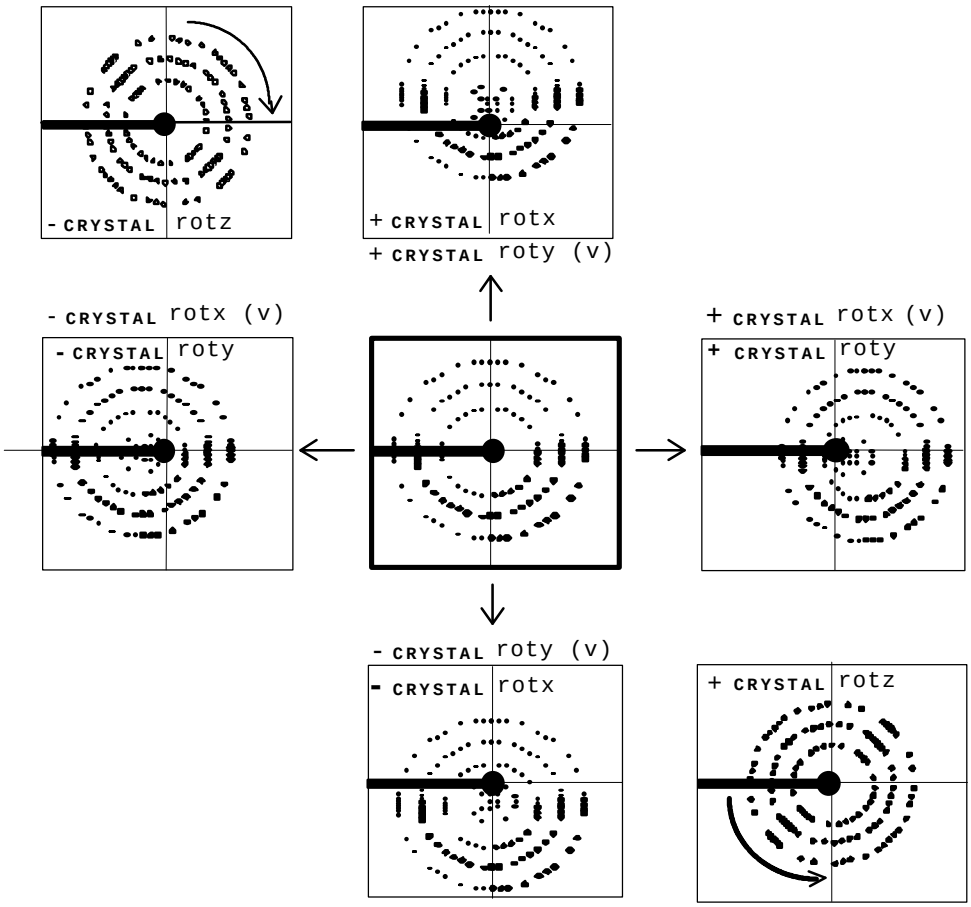


Figure 2: *Denzo rotation conventions. A simulated array of predicted reflections is shown in the center box, along with the operations that shift the position of the preds to those seen in the other boxes. The operation used to move the preds away from the pattern given in the center display (heavy lines) is given above each box. The spindle axis is assumed here to be horizontal, 180 degrees away from the beam stop axis (which is indicated by its shadow). In the case of vertical spindles, the subsequent effects of crystal rotations are given in parentheses, and denoted with a 'V'.*

The rotations CRYSTAL `rotz`, CRYSTAL `roty`, and CRYSTAL `rotx` are best thought of as rotations, expressed in degrees, applied to the predicted reflections (preds) from the initial conditions of CRYSTAL `rotz 0.0 roty 0.0 rotx` 0.0. Values are entered as absolute rotations from zero. For example, if you have CRYSTAL `rotz 23.7`, and then you subsequently enter CRYSTAL `rotz -15.2`, the applied value of CRYSTAL `rotz` is -15.2, ***not*** (23.7 - 15.2 = 8.5). Another way to picture these rotations is as the rotations of the arcs on a goniometer head mounted on the spindle of a camera:

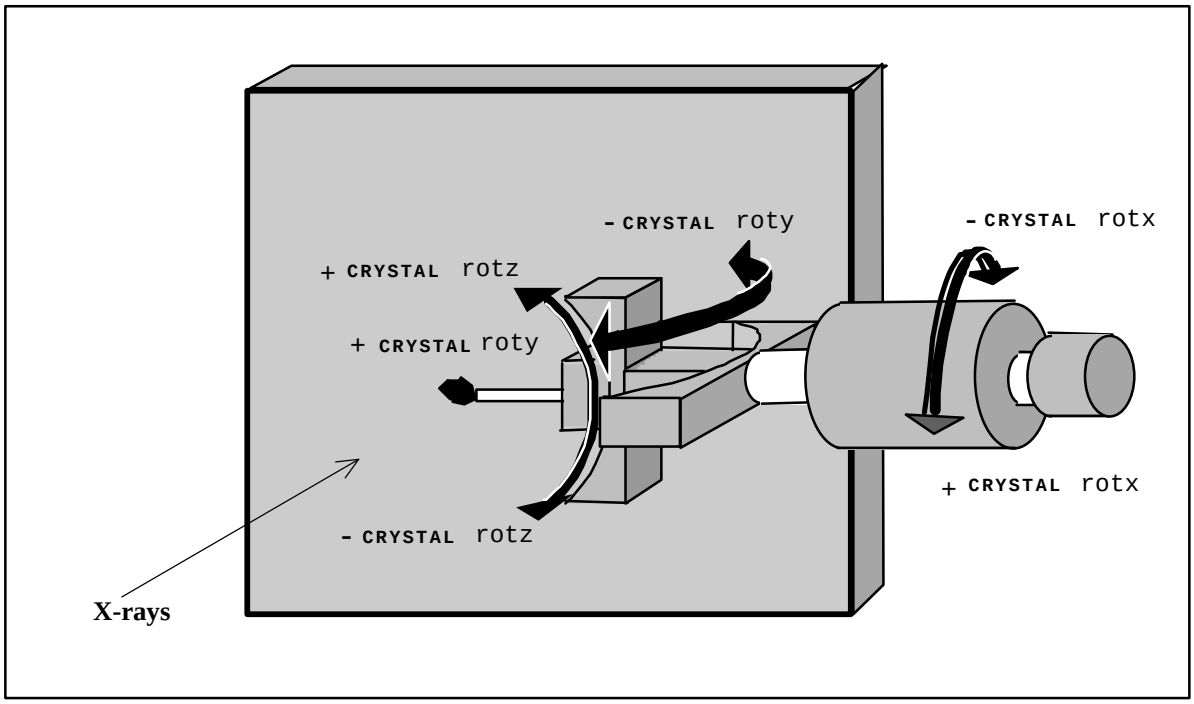


Figure 3: *Another way of thinking about Denzo rotation conventions: arcs on a goniometer Tip: to interpret this diagram for non-horizontal spindles, simply align the axis of your spindle with that of the spindle in the diagram. All the rotations line up then as well.*

It is easiest to change the crystal rotations one at a time in the following order: CRYSTAL `rotx` (spindle), CRYSTAL `roty` and then CRYSTAL `rotz`. To enter a new value, type it in the command window as: CRYSTAL `rotx -22.0` GO. You can change more than one value at a time, for example: CRYSTAL `rotx -20 rotz 5.0 roty 5.5` GO. In the manual part of the indexing, it is best to change the rotations in increments of no **less** than 0.5 degrees, since finer adjustments can be done automatically by the program. *Denzo* can recapitulate the current parameters with the command LIST.

The best advice is to think clearly, keep track of your movements and their results, and be patient. If you do manage to get a reasonable agreement with the diffraction pattern, then proceed with refinement.

Film Cassette and Off-line Image Plate Users

The success of indexing with *Denzo* (or any program, for that matter) depends critically on knowing or finding the position of the direct X-ray beam for each frame. With automatic image plate scanners like the MacScience, R-axis, or Mar, this is generally not a problem because the detector and the beam do not move much during data collection, and the scanning mechanism is accurate enough to scan the IP in a reproducible fashion. Thus, once the position of the direct beam is established for one frame, it is pretty much the same for all the frames of the series. Cassette-based systems like the Fuji scanners found at several synchrotrons (CHESS F1, Photon Factory, Brookhaven X25, X4, etc.) do not have this reproducibility, however. This is due to the slight slop inherent in the positioning of the IP in the cassette, the attachment of the cassette onto the camera carousel, and the positioning of the IP in the scanner magazine. Thus, the **X BEAM** and **Y BEAM** positions on each frame can vary beyond the normal radius of convergence for refinement in *Denzo*. One way to overcome this is simply to examine each frame one by one and note down the position of the direct beam in each case, and enter this value into the command file for batch processing .

Another way is to index the first frame and refine each subsequent frame interactively, one by one after determining the direct beam position, again, by inspection of each image. With a fast computer this is a viable, albeit tedious, option. Note however that you should ***not*** keep re-running the autoindexing routine. This is because the autoindexing routine will not use the same values for crystal rotx, roty, and rotz each time it is run, but instead will randomly choose between equivalent values. This does not matter the first time these values are chosen, but if each image is indexed with different values of **CRYSTAL** `rotx, roty, rotz`, then the original index of all the reflections will be inconsistent. This means that you will not be able to add partially recorded reflections correctly, or even scale the data, which is a major drag.

A third way is to incorporate a small "beam search" routine into the `refine.dat` macro file above. An example of such a search routine is shown below. Essentially, what this routine does is create a large zone within which the direct beam may be found, and then exclusively search for this single "reflection." Note that you must actually have a spot on the frame to find; if the beam stop does not have a thinned center through which the attenuated direct beam can pass, then this search routine is pointless and will not find what is not there to be found. Also, it helps if the starting values of **X BEAM** and **Y BEAM** are reasonably good average values. If this search is going to fail, it is usually because the starting point for the direct beam search was too far away from the actual point. It has also been known to fail if there are other small intense noise spots within the beam spot search zone. Once the beam is found, the box and spot sizes are reset to normal values, and the refinement proceeds normally.

Detector Specifications

Detector	XdisplayF version 1.5 executable	XdisplayF version 1.6 (xdisp executable) modifier	Denzo FORMAT modifier
CCD 1k 55mm [original] detector at CHESS	**INST_CCD**	**ccd sol**	**ccd sol**
CCD 2k detector 2048x2048 image - CHESS	**INST_CCD2x2**	**ccd 2048**	**ccd 2048**
CCD 2k detector in binned mode - CHESS	**INST_CCD2048b**	**ccd 2048 binned**	**ccd 2048 binned**
CCD (ADSC) detector - CHESS	**INST_ADSC**	**ccd adsc**	**ccd adsc**
CCD (ESRF) detector - beam line 19 at ESRF - format similar to MAR	**INST_BL19**	**ccd bl19**	**ccd bl19**
CCD (ESRF exprimental) detector	**INST_ESRF**	**ccd esrf**	**ccd esrf**
MAC Science DIP-100 scanner	**INST_DIP100**	**dip 100**	**dip 100**
MAC Science DIP-2000 scanner 12-bit A/D converter	**INST_DIP2000**	**dip 2000**	**dip 2000**
MAC Science DIP-2000 scanner 16-bit A/D converter	**INST_DIP2000b**	**dip 2000b**	**dip 2000b**
MAC Science DIP-2030 scanner 12-bit A/D converter	**INST_DIP2030**	**dip 2030**	**dip 2030**
MAC Science DIP-2030 scanner 16-bit A/D converter	**INST_DIP2030b**	**dip 2030b**	**dip 2030b**
MAC Science DIP-3000 cylindrical Weissenberg scanner	**INST_DIP3000**	**dip 3000**	**dip 3000**
MAC Science DIP-320 Weissenberg scanner	**INST_DIP320**	**dip 320**	**dip 320**
MAC Science DIP-420 off-line scanner	**INST_DIP420**	**dip 420**	**dip 420**
MAR scanner in small plate mode	**INST_HAMBURGN**	**mar 18cm**	**mar 18cm**
MAR scanner in large plate mode	**INST_HAMBURGD**	**mar 30cm**	**mar 30cm**
red EMBL scanner in Hamburg	**INST_HAMBURG**	**mar_corr**	**mar_corr**
MAR scanner at LBL	**INST_**	**mar lbl**	**mar lbl**
MAR scanner in MRC-LMB compressed 'pck' mode	**INST_MARPCK**	**mar compressed**	**mar compressed**
RAXIS-IIc Rigaku scanner 100 micron scan	**INST_RX**	**raxis**	**raxis**
RAXIS-IIc Rigaku scanner 200 micron scan	**INST_RXS**	**raxis small**	**raxis small**

RAXIS-IV Rigaku scanner 50 micron raster scan	**INST_ RXIV50**	**raxis4 50**	**raxis4 50**
RAXIS-IV Rigaku scanner 100 micron raster scan	**INST_RXIV100**	**raxis4 100**	**raxis4 100**
RAXIS-IV Rigaku scanner 200 micron raster scan	**INST_RXIV200**	**raxis4 200**	**raxis4 200**
RAXIS-IV Rigaku scanner (prototype) 100 micron raster scan		**raxis4 old**	**raxis4 old**

Note:

There is only one display program xdisp in **version 1.6** (and higher) which supports many detector systems. You are running that program by typing:

```
xdisp  format  <FileWithImage>
```

Format is the same as modifier of **`FORMAT`** keyword in *Denzo.*

You may get a list of all supported formats by typing xdisp.

examples:	xdisp dip 2000 lysoz0001.ipf	- DIP2000 12 bit A/D
	xdisp dip 2000b lysoz0001.ipf	- DIP2000 16 bit A/D
	xdisp raxis lysoz0001.img	- Raxis-II
	xdisp raxis4 50 lysoz0001.img	- RAXIS-IV 50 micron scan
	xdisp mar 30cm lysoz0001.image	- MAR 30cm scan
	xdisp mar 18cm lysoz0001.image	- MAR 18cm scan
	xdisp ccd sol lysoz0001.tif	- CCD 1k detector at CHESS

There are following detectors families:

dip	- DIP family of scanner
raxis2	- RAXIS-II
raxis4	- RAXIS-IV
mar	- MAR Research detectors
pf	- Photon Factory data
stoe	- STOE scanner
adsc	- ADSC detector (current model, at CHESS)
synch	- synchrotron data (fuji off-line, ccd's at Cornell, ESRF bl19 beam line)

THE *SCALEPACK* MANUAL

About the Scalepack manual (from Edition 4)

The manual is organized somewhat differently than the *Denzo* manual. Compared to film or IP processing, scaling is complex. There's a lot more to scaling data than just plugging in a bunch of *Denzo* .x files and getting a merged data set out the other side. In particular, scaling is where you make explicit estimates of the errors associated with your experiment, and these error estimates are crucial in determining the statistics by which you judge the quality of your data. In addition, *Scalepack* is also used for many other manipulations of the diffraction data. As a result, we decided to make the heart of the manual a series of scenarios, each accompanied by a sample command file, which correspond to many of the typical applications of *Scalepack*. If you do not find your particular problem addressed by these scenarios, we hope that they, along with the keyword documentation, will provide enough guidance so that you can construct your own.

Program Description

The scaling algorithm used in *Scalepack* is described in: Fox G.C., Holmes K.C., 1966 "An Alternative Method of Solving the Layer Scaling Equation of Hamilton, Rollet & Sparks" Acta Cryst. 20:886-891

The scaling and merging of different data sets as well as the global refinement of crystal parameters (postrefinement) is performed by *Scalepack*. Unlike in other procedures, in *Scalepack* the **estimated error** of the measurement is enlarged by a fraction of the **expected**, rather than the observed, intensity. **This algorithm reduces the bias towards reflections with an integrated intensity below the average**.

Due to the correlation between crystal and detector parameters the values of the unit cell parameters refined from a single image may be quite imprecise. This lack of precision is of little significance to the process of integration, as long as the predicted positions are on target. There is no contradiction here, because at some crystal/detector orientation the positions of the reflections may only weakly depend on the value of a particular crystal parameter. At the end of the data reduction process one would wish to get precise unit cell values. This is done in a procedure referred to as a global refinement or **postrefinement**. The implementation of this method in *Scalepack* allows for separate refinement of the orientation of each image, but with the same unit cell value for the whole data set. In each batch of data (a batch is typically one image) different unit cell parameters may be poorly determined. However, in a typical data set there are enough orientations to determine precisely all unit cell lengths and angles. Global refinement is also more precise than the processing of a single image in the determination of crystal mosaicity and the orientation of each image.

Scalepack allows you to analyze the **quality of the diffraction** from your crystals. The quality of diffraction data is judged by many standards, ultimately including the quality of the electron density map which results, but a widely used objective standard is based on the statistics which result from scaling and merging the raw diffraction data. This program calculates **single, isotropic scale** and **B factors** for each of the "films" or "batches" of processed data that are input. Scaled, merged data are output.

A **multiplicative correction factor**

$$S = \frac{e^{2B\left(\frac{\sin q}{l}\right)^2}}{scale}$$

is applied to I and σ(I). Merging of redundant measurements, which occurs throughout the refinement process, is done as follows. The multiplicative factor for each measurement is calculated from the scale and B of its film and applied to I to give Icorr. $\langle I_{corr} \rangle$ is a solution to the following equations:

$$W = \frac{1}{\left(s \cdot S \cdot E_1\right)^2 + \left(\langle I_{corr} \rangle \cdot E_2\right)^2}$$

where E_1 and E_2 are the input variables **error scale factor** and **estimated error**, respectively, and

$$\left\langle I_{corr} \right\rangle = \frac{\sum I_{corr} \cdot W}{\sum W}$$

The output σ(I) is

$$\frac{1}{\sqrt{\sum W}}$$

During **parameter refinement** the scale (and B, if requested) for all refinable films are shifted simultaneously to minimize the difference between the $\langle I_{corr} \rangle$'s and I_{corr}'s for individual measurements, summed over all reflections on all films. $\langle I_{corr} \rangle$'s are redetermined in each cycle as described above. One (or more) of the "batches" that is input is designated as the reference batch, and its scale and B factors are not refined. There is room for many reference batches; this program will also work with no reference batch in which case **eigenvalue filtering** stabilizes the shifts in scale and B factors.

Initial default scale and B factors are applied to each film, this can be replaced by explicit input of initial values. Initial scale factors should be around 1.0 (or larger for very strong data), so the output values are not too large, otherwise they can overflow fixed formats.

The user controls whether **anomalous pairs** are treated as equivalents or not and whether they are merged. The program rejects measurements that fail an I/σ(I) test when they are read in, and these do not contribute to the output. Partially recorded reflections can be added or not, according to the control input. Individual measurements can be rejected explicitly in the control input.

Control input is like that for *Denzo*; each record has a title and the order matters only to some degree. The title tells what input is expected to follow. All **file names** must be 80 characters or less and it is usually safer to enclose all file names in single quotes (`'myxtal##.x'`). Usually input variables follow in the same record as the title, except where the input is a table or a list. Anything between [and] in the command line is considered a comment by the program. A semicolon (;) terminates input. Unlike *Denzo*, *Scalepack* is usually run as a batch job because there is no point in watching your frames scale and because the *Scalepack* log file is where the statistics associated with scaling and merging are presented.

Statistics and Scalepack

The quality of X-ray data is initially assessed by statistics. In **small molecule** crystallography there is almost always a large excess of strong data, so this allows the crystallographer to discard a substantial amount of suspect data and still accurately determine a structure. Compared to small molecules, however, **proteins** diffract poorly. Moreover, important phase information comes from weak differences and we must be sure these differences do not arise from the noise caused by the limitations of the X-ray exposure and detection apparatus. As a result, we cannot simply throw away or statistically down-weight marginal data without first making a sophisticated judgment about which data is good and which is bad.

To accurately describe the structure of a protein molecule, we ordinarily need higher resolution data than the crystal provides. That is life. One of the main judgment the crystallographer makes in assessing the quality of his data is thus the **resolution** to which his crystal diffracts. In making this judgment, we wish to use the statistical criteria which are most discriminatory and which are the least subjective. In practice, there are two ways of assessing the high resolution limit of diffraction. The first is **the ratio of the intensity to the**

error of the intensity, i.e. I/σ. The second way, which is traditional but inferior, is the agreement between symmetry related reflections, i.e. R_{merge}.

From a statistical point of view**, I/σ is a superior criterion**, for two reasons. First, it defines a resolution "limit" since by definition **I/σ is the signal to noise** of your measurements. In contrast, R_{merge} is not directly related to signal to noise.

Second, the σ assigned to each intensity derives its validity from the χ^2's, which represent the **weighted** ratio of the difference between the observed and average value of I, ⟨I⟩, squared, divided by the square of the error model, the whole thing times a factor correcting for the correlation between I and ⟨I⟩. Since it depends on an explicit declaration of the expected error in the measurement, the user of the program is part of the Bayesian reasoning process behind the error estimation.

The essence of **Bayesian reasoning** in *Scalepack* is that you bring χ^2 (or technically speaking, the goodness-of-fit, which is related to the total χ^2 by a constant) close to 1.0 by manipulating the parameters of the error model. R_{merge}, on the other hand, is an **unweighted** statistic which is independent of the error model. It is sensitive to both intentional and unintentional manipulation of the data used to calculate it, and may not correlate with the quality of the data. An example of this is seen when collecting more and more data from the same crystal. As the redundancy goes up, the final averaged data quality definitely improves, yet the R_{merge} also goes up. As a result, R_{merge} is only really useful when comparing data which has been accumulated and treated the same. This will be discussed again later.

In short, **I/σ is the preferred way** of assessing the quality of diffraction data because it derives its validity from the χ^2 (likelihood) analysis. Unless all of the explicit and implicit assumptions which have been made in the calculation of an R_{merge} are known, this criterion is less meaningful. This is particularly true when searching for a "number" which can be used by others to critically evaluate your work.

There are two modes of analysis of data using χ^2s. The first mode keeps the χ^2 (or more precisely, the goodness-of-fit) constant and compares the error models. Basically, this means that you are adjusting your estimates of the errors associated with the measurement until the deviations within observations agree with the expectation based on the error model.

The second mode keeps the error model constant and compares χ^2s. This mode is computationally much faster and is used in refinement procedures. Of the two modes, the first is more informative, because it forces you to consider changes in the error model. Which mode you use generally depends on what you are comparing. When assessing the general quality of your detector, the first mode is used. When comparing a **derivative** to a **native**, the second mode is used due to an incomplete error model which does not take into account important factors like non-**isomorphism**. Thus, the χ^2 of scaling between a native and a derivative provides a measure of non-isomorphism, assuming the detector error is accurately modeled for both samples.

R_{merge} was historically used as an estimate of the non-isomorphism of data collected on film using several different crystals, and for this purpose it still has some validity because we do not account for non-isomorphism in our error model. It is not so important now that complete x-ray data sets are collected from single, frozen crystals.

One of the drawbacks of using $\mathbf{R_{merge}}$ as a measure of the quality of a data set is that it **can be intentionally and unintentionally manipulated**. Unintentional factors which can artificially lower R_{merge} generally have the effect of reducing the redundancy of the data or eliminating weaker observations. In crystallography, the greater the redundancy of the data, the worse the R_{merge}, because of the correlation between I and ⟨I⟩ which reduces the R_{merge}. The greater

the redundancy, the lower the correlation. For two measurements with the same σ, the correlation is 50%, so R_{merge} is underestimated by $\sqrt{2}$ compared to the case of no correlation. Known unintentional factors which lower R_{merge} include the following:

1. Data collected so that lower resolution shells, where the data is strong, have a higher redundancy than the higher resolution shells, where the data is generally weaker. This can be accomplished by collecting data on detectors where $2\theta \neq 0$, or including data from the corners of rectangular or square image plates. There is nothing wrong with using this data; it will just artificially lower the R_{merge}.
2. Inclusion of single measurements in the calculation of R_{merge} in one widely used program, which is why a table using this erroneous calculation used to be presented in the *Scalepack* output. Although the bug in the widely used program was unintentional, it nonetheless reduced the R_{merge} and this may have accounted for its longevity. A second, more subtle bug that reduced R_{merge} prompted the introduction of the keyword **BACKGROUND CORRECTION.** Fortunately, both bugs have now been fixed, but the point is that errors of this type can persist.
3. Omission of negative or weak reflections from the calculation of R_{merge}. This is often undocumented behavior of crystallographic data scaling/merging software. Examples include:
   a) elimination of a reflection from both the data set and the calculation of R_{merge} if $I < 0$
   b) conversion of $I < 0$ to $I = 0$ before the calculation of R_{merge} and inclusion of this reflection in the data set (the statistics of such a type are included in the Scalepack output for reasons of comparison. This is the first R_{merge} table in the log file, not the final one).
   c) omitting reflections with $\langle I \rangle < 0$ from the calculation of R_{merge} but inclusion of these reflections in the output data set.
4. Default sigma cutoffs set to unreasonable values, like 1. This is in fact the default of the R-Axis software.
5. Use of unreasonable default/recommended rejection criteria in other programs. These eliminate individual I's which should contribute to R_{merge} and yet are still statistically sound measurements.
6. Use of the eigenvalue filter to determine the overall B factor of a data set collected on a non-frozen, decaying crystal. In this case, the eigenvalue filter will calculate an overall B factor which is appropriate for the middle of the data set, yet apply this to all data. As a result, the high resolution data will be down weighted compared to data processed with the first, least decayed frame as the reference. The high resolution data is generally weaker than the low resolution data, and as a result is more likely to result in higher R_{merge}. By down-weighting the high resolution data, the R_{merge} is artificially lowered. Any program which does not allow the option of setting the reference frame will have this problem. Of course, there is no problem with non-decaying crystals.

There are also intentional ways of lowering your R_{merge}. Like those ways listed above, they generally result from the statistically invalid elimination of weak reflections, reduction of the redundancy of the data, or de-emphasis of weak data. The difference between these methods and those listed above is that they are generally under the control of the user.

1. Use of an unreasonable **SIGMA CUTOFF** (e.g.≥ 0). The rejection of weak data will always improve R_{merge}. There is a further discussion of **SIGMA CUTOFF** in the *Scalepack keywords documentation* section.

2. Use of a resolution limit cutoff. Again, the omission of weak data will improve R_{merge}. A reasonable resolution cutoff is the zone where $I/\sigma < 2$.

3. Combining into a single zone for the purposes of calculations those resolution shells where R_{merge} is rapidly changing. In this case, the shell will be dominated by the strong data at the low resolution end of the zone and give the impression that the high resolution limit of the zone has better statistics than it really does. For example, if you combined all your data into a single zone, the R_{merge} in the *final* shell would be pretty good (=R_{merge} overall), when in fact it was substantially worse. It is more sensible to divide your zones into equal volumes and have enough of them so that you can accurately monitor the decay with resolution.

4. Omitting partially recorded reflections. This has the effect of a) reducing the redundancy, and b) eliminating poorer reflections. Partially recorded reflections will always have a higher σ associated with them because they have a higher total background, due to the inclusion of background from more than one frame in the reflection.

5. Scaling I+ and I- reflections separately in the absence of a legitimate anomalous signal (**SCALE ANOMALOUS**) . This has the effect of reducing the redundancy.

6. Ignoring overloaded reflections using the **IGNORE OVERLOADS** in Scalepack. The intensity of overloaded or saturated, reflections cannot be directly measured, because obviously some of the pixels are saturated. Profile fitting only measures these reflections indirectly, by fitting a calculated profile to the spot using the information contained in the *wings* or *tail* of the spot. Ignoring the inaccuracies inherent in this method by ignoring overloads may have a dramatic effect on R_{merge}.

☛ **IGNORE OVERLOADS** is often a useful tool, however. For example, when calculating anomalous differences you do not want to use overloaded reflections because you are looking for very very small differences and want to use only the most accurate data. Another time you might ignore overloads is when you collect multipass data. In this case, a crystal is exposed twice, once for a short time, the other for a longer time. The longer exposure is to sufficiently darken the high resolution reflections, but will result in saturated low resolution reflections. Since the low resolution reflections can be obtained from the short exposures, the overloaded ones can be ignored in the long exposures.

See: Press, William H., Teukolsky Saul A., Vettering William T., Flannery Brian P. “Numerical Recipes in C, The Art of Scientific Computing”, Second Edition, Cambridge University Press, 1992

Implementation

Limits on variable dimensions

The current standard program dimensions allow 2,000,000 total observations, 600,000 unique reflections, and 800 films or batches. The program will refine up to 1400 parameters (e.g. scale factor, B factor) simultaneously. This should be enough for most uses of the program. If you have more than a two million observations or 800 batches or 1400 parameters, the trick is to subdivide the problem into smaller batches, produce an output file and use this to scale the remainder of the data.

The only absolute limitation at the moment is on the number of unique reflections. This is usually only a problem in virus crystallography, and for this purpose three other versions of *Scalepack* have been made: *ScalepackBig* (limit 10^6 unique reflections), *ScalepackVeryBig* (2 x 10^6 unique reflections) and *ScalepackHuge* (5 x 10^6 unique reflections). Below the maximum values of several parameters are shown:

Scalepack:	Small	(Normal)	Big	VeryBig	Huge
observations	1.2×10^6	2×10^6	3×10^6	5×10^6	8×10^6
possible hk pairs	30,000	40,000	40,000	60,000	60,000
unique reflections	0.4×10^6	0.6×10^6	1×10^6	2×10^6	5×10^6
fitted parameters (scale & B factor pairs)	2×600	2×700	2×800	2×800	2×800
batches	800	800	1000	1000	1000
rejected spots	30,000	50,000	80,000	100.000	100.000

Warning: the bigger version the more swap memory *Scalepack* needs!

Authors would like to receive e-mail about inadequacy of Scalepack versions.

Platforms

See the Appendix *Recommended Hardware* of the *Denzo* Manual.

Scenarios

Scalepack allows you to analyze the quality of the diffraction from your crystals providing the statistics which result from scaling and merging the raw diffraction data. Scenarios presented in this chapter are designed to show you how to get accurate data and statistics in situations typically encountered in protein crystallography. These scenarios consist of input *Scalepack* command files, followed by some comments about the statistics which result.

Scalepack is run in batch mode, unlike *Denzo* which has a graphical display component and is best run interactively. All of the scenarios are thus given as command files. For clarity the first two or three lines of each command file which set the directory and call the program have been omitted but are given here:

```
setenv cnverr yes                    only for IBM RS/6000 (obligatory)
cd /frames/mydir/Denzo               set directory to file which contains .x files
Scalepack <<STOPINPUT>> scale.log    start the program, name the log file
...
STOPINPUT
```

It is necessary to run the program this way, rather than making each of the scenarios an input file called by the @ command, e.g. `@scalesimple.inp`, because the Scalepack parser does not recognize nested @ commands. This is important for Scalepack because the list of rejected reflections is usually called with the `@reject` command. Just one of those things.

List of Scenarios

1. Scale data from one crystal processed with *Denzo*
2. Scale data from one crystal not processed with *Denzo*
3. Heavy atom search: Scale native + a few frames of a potential derivative
4. Scale multiple native data sets together. Data from *Denzo*
5. Reindexing
   a. Switch ***a*** and ***c*** in $P2_1$ or P2
   b. Change the order of ***a***, ***b***, ***c*** to ***b***, ***c***, ***a*** in orthorhombic
   c. Convert ***a***, ***b***, ***c*** to ***c***, ***a***, ***b*** in orthorhombic
   d. Reindex P23, switching ***a***, ***b***
   e. P321 or P312: reindex data and combine with another data set
   f. P6, P4, or H3: reindex data and combine with another data set
   g. Reindex from H32 to R32
   h. Reindex two H3 data sets, each with inconsistent index, and combine
   i. Reindex four $P3_1$ data sets, each with inconsistent index, and combine
6. Comparing two data sets from Scalepack
7. Compare I+ and I– : detection of an anomalous signal
8. Fitting polarization
9. High resolution + low resolution data sets from one crystal
10. Thin frames or few frames (`SCALE RESTRAIN` and `B RESTRAIN`)
11. Space group determination

Scenario 1: *Scale Data from One Crystal Processed with Denzo*

scalesimple.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	estimated error for each resolution shell
ERROR SCALE FACTOR 1.3	Multiplicative factor applied to input σ
REJECTION PROBABILITY 0.0001	expected fraction of outliers in the data.
[@reject]	[read in rejected reflections]
WRITE REJECTION FILE 0.9	observations with greater than 90% chance of being outliers are written to the reject file
SPACE GROUP P21	
OUTPUT FILE mydata.sca	output file containing ***h, k, l***, I, σ_I
REFERENCE BATCH 1	reference batch in scaling procedure. Can also leave out and let eigenvalue filter set reference scale and B
ADD PARTIALS 1 to 90	partials in batches 1 through 90 summed
POSTREFINE 10	number of cycles of postrefinement
FIT crystal a* 1 to 90 FIT crystal b* 1 to 90	refine unit cell parameters based on all data from batches 1 through 90.
FIT crystal c* 1 to 90 FIT crystal beta* 1 to 90	☛ space group dependent! e.g. don't refine alpha for monoclinic space groups.
FIT batch roty 1 to 90 FIT batch roty 1 to 90	crystal orientation of spindle axis angle will be refined for each batch (frame) separately. This allows you to spot slippage & other problems.
FIT crystal mosaicity 1 to 90	one mosaicity value will be fit
FORMAT denzo_ip	format of the input intensity data
SECTOR 1 to 90	these values will be substituted in for the ###
FILE 1 '/frames/denzo/myxtal###.x'	batch numbers will start at 1 for file *myxtal001.x* and increase to 90 for file *myxtal090.x*.

☛ Proper scaling requires multiple rounds of *Scalepack*, as the statistics may improve as outliers are rejected. On the second and all subsequent rounds of Scalepack the brackets would be removed from the @REJECT command on line 9 and outliers rejected in previous rounds would be removed from the data prior to refinement of scale and B factors. It usually takes about 1 to 3 iterations before no more reflections are rejected and the process converges.

The log file should be examined after each iteration. In particular, the errors, χ^2 and both R factors should be checked. You can adjust the error estimates in the command file to improve the accuracy of the statistics. Here are some helpful guidelines:

1. Examine the difference between the total error and the statistical error in the final table of statistics. The difference between these numbers tells you what contribution statistical error makes to the total error (σ). If the difference is small, then reasonable changes in the values following the ESTIMATED ERROR keyword will not help your χ^2 much. This is because the estimated errors represent your guess / knowledge of the contribution of systematic error to the total error and a small difference indicates that estimated error is not contributing much. If the difference between total and statistical error is significant, and the χ^2 are far from 1, then consider adjusting the ESTIMATED ERROR values in the affected resolution shells.

2. Examine the difference between the linear and square R_{merge}s. For this comparison to be valid, you cannot have strong overloads in your data, so the command **IGNORE OVERLOADS** must be given for this analysis. If R_2 is larger than R, then the distribution of the deviations (I - ⟨I⟩) is skewed. This implies that you have too many outliers in your data, so you should reject more reflections. The way to increase rejections is to choose a higher value of **REJECTION PROBABILITY** or write rejections to the reject file which have a lower probability of being outliers. Useful increments are factors of 10 in **REJECTION PROBABILITY** and replacing default value of **WRITE REJECTION FILE** (0.9) with 0.5. If after one round this does not bring R_2 close to R (e.g. R_2 is more than 50% greater than R), then you should consider whether you have detector problems. Likely culprits include the shutter and the spindle. Note that bad crystals (split, twinned) and ice rings can also produce this phenomenon, as can the inclusion of a few frames of data from a badly decayed crystal.

3. If $R \approx R_2$ and your χ^2 are high, then your crystal may have suffered radiation damage which cannot be corrected by scale and B factors. This type of damage may not even increase B factors or decrease scale factors. This is frequently seen with non-frozen crystals.

4. The χ^2 is your best indication of whether your error model accurately reflects the actual errors in your data. Since the errors in your data are expressed in the σ value you attach to every I, it is important that your error model be correct. χ^2 values of close to 1 indicate that your model is pretty good. If your χ^2 values deviate much from 1, you can fix this by adjusting the value of the error scale factor. Note that you don't have to hit 1.0, since χ^2 goes as the square of your error parameters, so 20% deviations from 1.0 represent only a 10% adjustment to σ, which is OK for most practical purposes. Even a $\chi^2 = 2$ is unlikely to seriously affect your subsequent calculations.

Scenario 2: *Scale Data from One Crystal not Processed with Denzo*

scalexengen.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.0	Multiplicative factor applied to input σs
REJECTION PROBABILITY 0.0001	expected fraction of outliers in the data. Here expect 1/10,000 will be outliers
SPACE GROUP P21	
UNIT CELL 39 94 67 90 96.1 90	UNIT CELL needed because not in header
OUTPUT FILE mydata.sca	OUTPUT FILE containing ***h***, ***k***, ***l***, **I**, σ_I
REFERENCE BATCH 101	REFERENCE BATCH in scaling procedure. Can also leave out and let eigenvalue filter set reference scale and B factor
[POSTREFINE 10]	can't POSTREFINE because not Denzo data
@reject	file with list of rejected reflections from previous round of *Scalepack*
WRITE REJECTION FILE 0.9	observations with greater than 90% chance of being outliers are written to the reject file
FORMAT xengen urf	FORMAT of the input intensity data
SECTOR WIDTH 5.0	must specify these two because not in header
FRAME WIDTH 0.2	
PROFILES SUMMED	since didn't run *Denzo*, this may be better than PROFILES FITTED
FILE 1 ctbn1.urf	urf files are an intermediate file format created by *xengen*
FILE 101 ctbn2.urf	
FILE 201 ctbn3.urf	this would be data from 4 crystals or perhaps 4 sweeps of the same crystal. A sweep can be a full data set.
FILE 301 ctbn4 urf	

☛ Because the file format is not from *Denzo*, you will not be able to POSTREFINE your data to get better values of the cell constants. Partials are not summed in these cases because the *xengen* (or *XDS*, *MADNES*, or *Scalepack*) software has already done these tasks. You would perform the iterative refinement with this data just as described above for Scenario 1.

☛ Some of the programs produce binary files that *Scalepack* may not be able to read-in if *Scalepack* runs on a different computer architecture than data reduction program that produced binary file.

Scenario 3: *Heavy Atom Search: Scale native + a few frames of a potential derivative*

scalepotderiv.com

[The Native Dataset***]***	
NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0	ESTIMATED ERROR for each resolution shell. Set to zero because native has already been scaled
[ERROR SCALE FACTOR 1.0]	don't need here because data already scaled
SPACE GROUP P3121	SPACE GROUP of native & derivative duh!
FORMAT Scalepack	native data has been processed already and is in *Scalepack*
FILE 101 native.sca	format.
REFERENCE BATCH 101	REFERENCE BATCH in scaling procedure. We will compare derivative to native, so native is reference
[Potential Derivative***]***	
RESOLUTION 3.5	better to search at medium to low resolution
ESTIMATED ERROR 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05	derivative is new crystal so need new ESTIMATED ERROR
ERROR SCALE FACTOR 1.6	we need this here because new data
REJECTION PROBABILITY 0.0001	expected fraction of outliers in the data. This is critical because the presence of be indicative of a potential derivative!
IGNORE OVERLOADS	this removes overloaded reflections from statistics. Overloads may cause large differences but are not indicative of a derivative
ADD PARTIALS 1 to 3	partials in derivative will be summed
FORMAT denzo_ip	data format of potential derivative
[HKL MATRIX -1 0 0 0 -1 0 0 0 1]	only reindex if native and derivative have different indices. See scenario #5
SECTOR 1 to 3 FILE 1 'NewHg###.x'	3 frames of a potential Hg derivative e.g. 3 x 1 degree oscillations.
OUTPUT FILE 'junk.sca'	not interested in .sca file, only the log

☛ The goal here is to scale a few frames of a potential derivative with a native data set. Here you only do a single round of refinement, not the iterative cycles done in Scenario 1 because you are not doing any cycles of rejection. In this case three frames of the derivative, say as much as 3 degrees, will be compared with the native to a resolution of 3.5 Å.

☛ This scenario is only valid for a small wedge because otherwise you will be looking at R_{merge}s which result from reflections combined with the derivative data set as well as with those of the native and so the statistics will be less indicative.

We are only interested in the last page of the Scalepack log file, where we will examine the χ^2 values and R_{merge}. Here are some general rules for assessing your potential derivative.

☛ you should pay most attention to the low resolution shells, where the measurements are most accurate.

1. If $\chi^2 \approx 1$ then your crystal scales well with the native so it is not a useful derivative. Remember, derivatives are different!
2. If χ^2 = very large (say 50) then you have either severe non-isomorphism (sorry), your have very heavy substitution (but it is rare to have such a strong isomorphous difference signal), or your derivative is misindexed relative to the native. Reindexing is only a potential problem in the case of all trigonal, P6, R3, P4, I4, I23, and P23 Laue groups, and in cases where there are accidental metric tensor symmetries (e.g. monoclinic $b \approx 90^o$, or orthorhombic a = b). To reindex, apply an HKL MATRIX to test the other possibilities (*discussed in Scenario 5*) and see which one gives the lowest χ^2 and R_{merge}. That is then the correct index.
3. If $\chi^2 \approx 10$ and the detector is not a problem (as indicated by $R_2 \approx R$), then this indicates a potential derivative because you know that the differences are much larger than the expected measurement error.
4. If $\chi^2 \approx 2$ you either have a poorly substituted derivative or a slight detector problem. Either way this is not going to be your best derivative. The resolution shell where the χ^2 drops to about 2 represents the useful limit of your derivative measurements.

☛ Instead of three 1 degree photographs, in many cases you can get away with a single 3 degree oscillation.

Scenario 4: *Scale Multiple Native data sets together. Data from Denzo*

There are two ways to scale multiple native data sets together. The first and preferred way is to scale all of the .x files together. If you do it this way, you get to do postrefinement over all of the raw data, and the scaling is applied at the level of the individual frames, rather than over the whole data set. This is probably a little more accurate. Also it saves you from having to run *Scalepack* twice: once for the second native data set and then to scale the two .sca files together.

scaletwo.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.3	Multiplicative factor applied to input σs
REJECTION PROBABILITY 0.0001	expected fraction of outliers in the data.
WRITE REJECTION FILE 0.9	reflections with ≥ 90% probability as outlier are written to the rejection file
SPACE GROUP P21	
REFERENCE BATCH 1	may use the first frame as reference for scaling or can omit this and use eigenvalue filter
[@reject]	[reject outliers from previous rounds]
POSTREFINE 10	10 cycles of postrefinement
FIT crystal a* 1 to 327 FIT crystal b* 1 to 327 FIT crystal c* 1 to 327 FIT crystal beta* 1 to 327 FIT batch rotx 1 to 327 FIT batch roty 1 to 327 FIT batch mosaicity 1 to 327	there are missing batch numbers created by the file statement below, but we do not need to mention this for postrefinement ☛ note which parameters are fit over the two data sets and which are for each frame
ADD PARTIALS 0 to 103 201 to 327	the consecutive sequences must be declared here, otherwise program will try to ADD PARTIALS from non-consecutive frames
FORMAT denzo_ip	
*[****Data Set #1*******]*	
SECTOR 1 to 103 FILE 1 'my_crystal1###.x'	first data set has 103 frames
*[****Data Set #2*******]*	
SECTOR 1 to 127 FILE 201 'my_crystal2###.x'	second data set sectors in batches starting with 201
OUTPUT FILE 'my_crystal12.sca'	

Alternatively, if you have already processed the two data sets and have two .sca files you want to scale and merge them, you can use the following command file.

☛ Note that **you cannot POSTREFINE or ADD PARTIALS between the two data sets**. You are basically scaling and merging one set of processed and outlier-rejected native data to another.

scaletwosca.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0	ESTIMATED ERROR for each resolution shell. Set to zero because native has already been scaled
ERROR SCALE FACTOR 1.0	Multiplicative factor applied to input σ's expected fraction of outliers in the data.
FORMAT Scalepack	
*[****Data Set #1*******]*	
FILE 1 'my_crystal1.sca'	
*[****Data Set #2*******]*	
FILE 2 'my_crystal2.sca'	
OUTPUT FILE 'my_crystal12.sca'	

Scenario 5: *Reindexing*

Reindexing involves reassigning indices from one unit cell axis to another. This becomes an important issue when comparing two or more data set which were collected and processed independently. This is because *Denzo*, when confronted with a choice of more than one possible indexing convention, makes a random choice. This is no problem, except that if it makes a different choice for a second data set, the two will not be comparable without reindexing one or the other. One cannot distinguish non-equivalent alternatives without scaling the data, which is why this is not done in *Denzo*. You can tell if you need to reindex a data set if the χ^2 values upon merging the two are very high (e.g. 50). This makes sense when you consider that scaling two or more data sets involves comparing reflections with the same *hkl* or index. If the two indexing schemes are equivalent but not identical, chaos will result.

☛ No reindexing, no new autoindexing, and **nothing** except changing the sign of Y SCALE in *Denzo* can change the sign of the anomalous signal.

Below are nine reindexing scenarios which should cover all of the possibilities encountered in protein crystallography.

Switch ***a*** and ***c*** in $P2_1$ or P2

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input **σ**
REJECTION PROBABILITY 0.0001	
WRITE REJECTION FILE 0.9	
@reject	
RESOLUTION 40 3.3	
SPACE GROUP p21	
REFERENCE BATCH batch 1	
ADD PARTIALS partials 1 to 180	
POSTREFINE 10	POSTREFINE will affect partials
FIT crystal a* 1 to 18	
FIT crystal b* 1 to 180	
FIT crystal c* 1 to 180	
FIT crystal beta* 1 to 180	
FIT batch rotx 1 to 180	
FIT batch roty 1 to 180	
FIT batch mosaicity 1 to 180	
FORMAT denzo_ip	
HKL MATRIX 0 0 1 0 -1 0 1 0 0	reindexing matrix: h l this is -1 to keep determinant = 1 l h
SECTOR 1 to 180	
FILE 1 '/frames/myxtal###.x'	
OUTPUT FILE 'myxtal.sca'	OUTPUT FILE contains reindexed reflections

Change order of ***a***, ***b***, ***c*** to ***b***, ***c***, ***a*** in orthorhombic

`NUMBER OF ZONES 10`	number of resolution shells for statistics
`ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04`	`ESTIMATED ERROR` for each resolution shell
`ERROR SCALE FACTOR 1.4`	multiplicative factor applied to input **σ**
`REJECTION PROBABILITY 0.0001`	
`WRITE REJECTION FILE 0.9`	
`@reject`	
`RESOLUTION 40 3.3`	
`SPACE GROUP p21212`	can also use for P2221
`REFERENCE BATCH 1`	
`ADD PARTIALS 1 to 180`	
`POSTREFINE 10`	good idea to postrefine independent of reindexing

`FIT crystal a* 1 to 180`
`FIT crystal b* 1 to 180`
`FIT crystal c* 1 to 180`	

`FIT batch rotx 1 to 180`
`FIT batch roty 1 to 180`
`FIT batch mosaicity 1 to 180`	

`FORMAT denzo_ip`	

`HKL MATRIX 0 1 0`
`0 0 1`
`1 0 0`
	reindexing matrix: h k
	k l
	l h

`SECTOR 1 to 180`	
`FILE 1 '/frames/myxtal###.x'`	
`OUTPUT FILE 'myxtal.sca'`	`OUTPUT FILE` contains reindexed reflections

Convert ***a***, ***b***, ***c*** to ***c***, ***a***, ***b*** in orthorhombic

`NUMBER OF ZONES 10`	number of resolution shells for statistics
`ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04`	`ESTIMATED ERROR` for each resolution shell
`ERROR SCALE FACTOR 1.4`	Multiplicative factor applied to input σ
`REJECTION PROBABILITY 0.0001`	
`WRITE REJECTION FILE 0.9`	
`@reject`	
`RESOLUTION 40 3.3`	
`SPACE GROUP p21212`	
`REFERENCE BATCH 1`	
`ADD PARTIALS 1 to 180`	
`POSTREFINE 10`	

`FIT crystal a* 1 to 180`
`FIT crystal b* 1 to 180`
`FIT crystal c* 1 to 180`	

`FIT batch rotx 1 to 180`
`FIT batch roty 1 to 180`
`FIT batch mosaicity 1 to 180`	

`FORMAT denzo_ip`	

`HKL MATRIX 0 0 1`
`1 0 0`
`0 1 0`
	reindexing matrix: $h \rightarrow l$
	$k \rightarrow h$
	$l \rightarrow k$

`SECTOR 1 to 180`	
`FILE 1'/frames/Denzo/myxtal###.x'`	
`OUTPUT FILE 'myxtal.sca'`	`OUTPUT FILE` contains reindexed reflections

Reindex P23, switch ***a***, ***b***

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input **σ**
[REJECTION PROBABILITY 1.0e-4] [WRITE REJECTION FILE 0.9] [@reject]	commented out to spare log file in case this reindexing is incorrect
RESOLUTION 40 3.3	
SPACE GROUP p23	
REFERENCE BATCH 1001	
ADD PARTIALS 1 to 180 1001 to 1180	
POSTREFINE 10	
FIT crystal a* 1 to 180	
FIT batch rotx 1 to 180 FIT batch roty 1 to 180 FIT batch mosaicity 1 to 180	
FORMAT denzo_ip	
HKL MATRIX 0 1 0 -1 0 0 0 0 1	reindexing matrix: h k k -h also works for I23 and F23
SECTOR 1 to 180	
FILE 1 '/frames/myxtal2###.x'	this data will be reindexed by the matrix above
[*** **Unit Matrix** ***]	
HKL MATRIX 1 0 0 0 1 0 0 0 1	data set #1 to be scaled with reindexed data
SECTOR 1 to 180	
FILE 1001 '/frames/myxtal1###.x'	this data will not be reindexed
OUTPUT FILE 'myxtal.sca'	output combines 2 data sets with consistent indexes

P321 or P312. Reindex data and combine with another data set

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input σ
[REJECTION PROBABILITY 0.0001]	commented out to spare log file in case
[WRITE REJECTION FILE 0.9]	reindexing is incorrect.
[@reject]	
RESOLUTION 40 3.3	
SPACE GROUP p321	
REFERENCE BATCH 101	
ADD PARTIALS 1 to 60 101 to 160	
POSTREFINE 10	good idea to POSTREFINE
FIT crystal a* 1 to 160 FIT crystal c* 1 to 160	
FIT batch rotx 1 to 160 FIT batch roty 1 to 160 FIT batch mosaicity 1 to 160	
FORMAT denzo_ip	
HKL MATRIX -1 0 0 0 -1 0 0 0 1	reindexing matrix: h - h k - k also works for P312, P3121, P3221, P3112 and P3212
SECTOR 1 to 60	
FILE 1 '/frames/myxtal2###.x'	this data will be reindexed by matrix above
*[***Unit Matrix ***]*	
HKL MATRIX 1 0 0 0 1 0 0 0 1	data set #1 to be scaled with reindexed data
SECTOR 1 to 60	
FILE 101 '/frames/myxtal1###.x'	this data will not be reindexed
OUTPUT FILE 'myxtals12.sca'	output combines 2 data sets with consistent indexes

p6, p4 or h3: Reindex data and combine with another data set

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input σ
[REJECTION PROBABILITY 1.0e-4]	
[WRITE REJECTION FILE 0.9]	
[@reject]	
RESOLUTION 40 3.3	
SPACE GROUP p6	
REFERENCE BATCH 1	
ADD PARTIALS 1 to 60 101 to 160	
POSTREFINE 10	good idea to POSTREFINE because this will affect partials
FIT crystal a* 1 to 160 FIT crystal c* 1 to 160	
FIT batch rotx 1 to 160 FIT batch roty 1 to 160 FIT batch mosaicity 1 to 160	
FORMAT denzo_ip	
HKL MATRIX 0 1 0 1 0 0 0 0 -1	reindexing matrix: h k k h l - l, also works for p4 and h3
SECTOR 1 to 60	
FILE 1 '/frames/myxtal2###.x'	this data will be reindexed by matrix above
*[*** **Unit Matrix** ***]*	
HKL MATRIX 1 0 0 0 1 0 0 0 1	data set #1 to be scaled with reindexed data
SECTOR 1 to 60	
FILE 101 '/frames/myxtal1###.x'	this data will not be reindexed
OUTPUT FILE 'myxtals12.sca'	output combines 2 data sets with consistent indexes

Reindex from H32 to R32

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input σ
REJECTION PROBABILITY 1.0e-4	
WRITE REJECTION FILE 0.9	
@reject	
RESOLUTION 40 3.3	
SPACE GROUP r32	after reindexing this will be the space group
REFERENCE BATCH 1	
ADD PARTIALS 1 to 90	
POSTREFINE 10	good idea to POSTREFINE because this will affect partials
FIT crystal a* 1 to 90 FIT crystal c* 1 to 90	
FIT batch rotx 1 to 90 FIT batch roty 1 to 90 FIT batch mosaicity 1 to 90	
FORMAT *denzo_ip*	
HKL MATRIX -1 0 1 1 -1 0 1 1 1	
SECTOR 1 to 90	
FILE 1 '/frames/myxtal###.x'	this data will be reindexed by matrix above
OUTPUT FILE 'myxtals12.sca'	output combines 2 data sets w/ same index

Reindex two H3 data sets, each with inconsistent index, and combine

This scenario shows how to reindex two H3 data sets into R3 and combine them. This is slightly more complicated than the scenarios above, since the second data set must also be reindexed into R3 from H3. It is entirely possible that both of the two data sets would use the same reindexing matrix to arrive at a consistent R3 index, but the example below assumes that you have already determined or are just testing that the two H3 data sets have indices which are inconsistent with each other.

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input σ
[REJECTION PROBABILITY 1.0e-4]	don't reject until you know reindexing makes the two data set
[WRITE REJECTION FILE 0.9]	consistent
[@reject]	
RESOLUTION 40 3.3	
SPACE GROUP r3	H3 in Denzo, converted to R3 in Scalepack by reindexing matrix
REFERENCE BATCH 1	
ADD PARTIALS 1 to 90 101 to 190	
POSTREFINE 10	good idea to POSTREFINE because this will affect partials
FIT crystal a* 1 to 190 FIT crystal c* 1 to 190	
FIT batch rotx 1 to 190 FIT batch roty 1 to 190 FIT batch mosaicity 1 to 190	
FORMAT denzo_ip	
HKL MATRIX -1 0 1 1 -1 0 1 1 1	reindexing first matrix
SECTOR 1 to 90	
FILE 1 '/frames/myxtal1###.x'	this will be reindexed by matrix 1 above
HKL MATRIX 1 -1 0 -1 0 1 -1 -1 -1	data set #2 to be scaled with reindexed data
SECTOR 1 to 90	
FILE 101 '/frames/myxtal2###.x'	this data will be reindexed by matrix 2 above
OUTPUT FILE 'myxtals12.sca'	output combines 2 data sets with consistent indexing

Reindex four $P3_1$ data sets, each with inconsistent index, and combine

This scenario shows how to reindex four $P3_1$ data sets and combine them. This is slightly more complicated than the scenarios above. Again, the procedure is as follows: choose one of your 4 data sets as the *reference* set. This reference is only for the purposes of indexing, not for finding scale factors. Then what you do is to take each of the other 3 data sets, one by one, and scale them with the reference data set. Each time, you will have to try a different **HKL MATRIX** for the second data set until you find the one which is consistent with the reference data set. Make a note of this **HKL MATRIX**, and move on to the next data set. When you have figured out the proper **HKL MATRIX** for each of the three other data sets, then you can run this command file to reindex them and scale them all together. This command file assumes that you were unlucky enough to have all four data sets in four inconsistent indices.

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	**ESTIMATED ERROR** for each resolution shell
ERROR SCALE FACTOR 1.4	multiplicative factor applied to input σ
[**REJECTION PROBABILITY** 1.0e-4]	don't reject until you know all data sets are in a consistent index
[**WRITE REJECTION FILE** 0.9]	
[@reject]	
RESOLUTION 40 3.3	
SPACE GROUP p31	
REFERENCE BATCH 201	
ADD PARTIALS 1 to 60 101 to 160 201 to 260 301 to 360	
POSTREFINE 10	good idea to **POSTREFINE** because this will affect partials
FIT crystal a* 1 to 360	
FIT crystal c* 1 to 360	
FIT batch rotx 1 to 360	
FIT batch roty 1 to 360	
FIT batch mosaicity 1 to 360	
FORMAT denzo_ip	
HKL MATRIX -1 0 0 0 -1 0 0 0 1	reindexing first matrix: h - h k - k
SECTOR 1 to 60	
FILE 1 '/frames/myxtal1###.x'	this will be reindexed by matrix 1 above
HKL MATRIX 0 -1 0 -1 0 0 0 0 -1	data set #2 to be scaled with reindexed data
SECTOR 1 to 60	
FILE 101 '/frames/myxtal2###.x'	this data will be reindexed by matrix 2 above
HKL MATRIX 0 1 0 1 0 0 0 0 -1	data set #3 to be scaled with reindexed data
SECTOR 1 to 60	
FILE 201 '/frames/myxtal3###.x'	this data will be reindexed by matrix 3 above
HKL MATRIX 1 0 0 0 1 0 0 0 1	original unit matrix
SECTOR 1 to 60	
FILE 301 '/frames/myxtal4###.x'	this is the reference index data set

OUTPUT FILE 'myxtals12.sca'	*output combines 2 data sets with same index*

Scenario 6: *Comparing two data sets from Scalepack*

scalenatder.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0	Set to zero because native & derivative have already been assigned proper errors.
[ERROR SCALE FACTOR 1.0]	don't need here because data already scaled
SPACE GROUP I422	SPACE GROUP of native & derivative (optional)
FORMAT Scalepack	data has been processed already and is in Scalepack FORMAT
FILE 1 native.sca	
REFERENCE BATCH 1	REFERENCE BATCH in scaling procedure. We will compare derivative to native, so native is reference
[***Derivative***]	
RESOLUTION 2.8	use highest resolution from the worse data set
[HKL MATRIX 0 0 1 0 -1 0 1 0 0]	only reindex if native and derivative have different indices
FILE 2 'NewHg.sca'	derivative data set
OUTPUT FILE 'NL:junk.sca'	not interested in .sca file, only the log
print total chi2 1000 print single chi2 1000	these are set super high so that basically no outliers are printed to log file

You would use this scenario if you wanted to see how well a native and derivative data set scaled together. It is different from Scenario 3 because both data sets are complete and are being read in as .sca files. This is a more accurate way of identifying your derivatives but it requires enough data from the derivative so that it can be internally scaled.

Scenario 7: *Compare I+ and I- : detection of anomalous signal*

This involves a 2-step scaling procedure. In the first step the data are scaled and postrefined along the lines of *scenario 1* with one exception, noted below, namely that the **ANOMALOUS** flag has been set. This tells the program to output a .sca file where the I+ and I– reflections are separate. Then, in the second step, a clever trick is used: the .sca file is read in, but the **ANOMALOUS** flag is turned off. In this case Scalepack will treat the I+ data and the I– data as two separate measurements within a data set and the statistics that result from merging the two will reflect the differences between the I+ reflections and the I– reflections. Cool. Notice that there is no need to go through a lot of jiffies which separate the I+ and I– data and reformat them, etc.

Step 1 `scaleano.com`

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	**ESTIMATED ERROR** for each resolution shell
ERROR SCALE FACTOR 1.3	multiplicative factor applied to input σ
SPACE GROUP P212121	
REFERENCE BATCH 1	
ADD PARTIALS 1 to 180	
POSTREFINE 10	
FIT crystal a* 1 to 180 **FIT** crystal b* 1 to 180 **FIT** crystal c* 1 to 180	
FIT batch rotx 1 to 180 **FIT** batch roty 1 to 180 **FIT** crystal mosaicity 1 to 180	
REJECTION PROBABILITY 0.0001 **WRITE REJECTION FILE** 0.9	expected fraction of outliers in the data; if your anomalous signal is strong, consider using I+ & I- considered equivalent when calculating scale & B and for statistics, but output keeps them separate and they are not merged
ANOMALOUS	
FORMAT denzo_ip	
SECTOR 1 to 180	
FILE 1 '/frames/Denzo/myxtal###.x'	
OUTPUT FILE 'mydataano.sca'	output file contains h, k, l, I+, σ_{I+}, I-, σ_{I-}

Step 2 `scalecompareano.com`

```
NUMBER OF ZONES 10
ESTIMATED ERROR 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0   these are zero because we have already scaled the data
ERROR SCALE FACTOR 1.0
NUMBER OF ITERATIONS 0                                    no scaling done, we aren't interested in .sca, only stats in log
OUTPUT FILE 'NL:junk.sca'                                 we will be reading in a .sca file from step 1 above
FORMAT Scalepack
FILE 1 'mydatano.sca'
```

Step 2 reads in the .sca file, which contains both I+ and I–. Obviously, for centric reflections there is no I–, so the merging statistics will only reflect the non-centric reflections. You can tell what percentage of your data is being used to calculate the merging statistics by examining the redundancy table near the end of the log file. Under the column of redundancy > 2 you will find out what percentage of the data is being compared. Since you only have I+ and I– you will never have a redundancy of more than 2.

The presence of an anomalous signal is detected by examining the χ^2 values in the log file from Step 2. Assuming that the errors in the Step 1 cycles were reasonable, a $\chi^2 \approx 1$ in the log file of Step 2 means that there is no useful anomalous signal in your data. On the other hand, if $\chi^2 > 1$ in Step 2, this may indicate the presence of an anomalous signal. Of course, it may also indicate absorption problems, detector problems, etc. But at the very least you can examine the χ^2 as a function of resolution and decide where to cut off your resolution for the purposes of calculating an anomalous difference Patterson (use the shells where $\chi^2 \geq 2$). This whole analysis assumes that the error model is reasonable and gives you a χ^2 close to 1 when the `SCALE ANOMALOUS` flag is used.

☛ You can only use `SCALE ANOMALOUS` when you have enough redundancy to treat the F+ and F– completely independently.

Scenario 8: *Fitting polarization*

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.3	multiplicative factor applied to input σ
REJECTION PROBABILITY 0.0001	
SPACE GROUP P422	
OUTPUT FILE mydata.sca	
REFERENCE BATCH 1	
ADD PARTIALS 1 to 90	
@reject	
[WRITE REJECTION FILE 0.9]	commented out here because we do not want to increase the reject list
FORMAT denzo_ip	
POLARIZATION denzo 0.9 corrected 0.8	POLARIZATION correction is made here.
SECTOR 1 to 90	
FILE 1 '/frames/Denzo/myxtal###.x'	

Polarization correction testing is done in two stages. First, you scale the data as in Scenario 1, writing a rejection file. Then, second, you run the command file above, which changes the POLARIZATION. The log file is then examined to see if the χ^2 crystalhave decreased, compared to the previous round when no polarization correction was done. If the χ^2 have improved (a significant improvement may be as low as 0.01), then you know that this is a better value for the polarization. If there is no change in the χ^2, then you cannot establish whether the polarization was right or wrong. This requires a tetragonal crystal, trigonal crystal, or cubic crystal, exposed in the proper orientation. It is mostly for synchrotron beamline staff, not ordinary users.

Scenario 9: *High resolution + low resolution data sets from one crystal*

This scenario is useful if you have made multiple passes of data collection from the same crystal, where one of the passes had short exposures to keep the low resolution reflections from saturating, and the other pass was a longer exposure to get the high resolution data. In the high resolution, long exposure pass, the low resolution reflections are mostly saturated. Note the more complicated batch numbering scheme. This is often typical of data collected at synchrotrons where there are gaps in the file numbering for various reasons. This is particularly applicable to CCD detectors at CHESS.

scalehilo.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each res. shell. Set to zero because native has already been scaled
ERROR SCALE FACTOR 1.5	good for CCD at CHESS
REJECTION PROBABILITY 1. e-4	expected fraction of outliers in the data
WRITE REJECTION FILE 0.5	
SPACE GROUP P21	
UNIT CELL 50 93 95 90 89.9 90	
RESOLUTION 1.4	one res. range for both high & low res.
REFERENCE BATCH 1	use the first frame as reference for scaling
@reject	reject outliers from previous rounds
POSTREFINE 20	20 cycles of postrefinement
FIT crystal a* 0 to 1039	
FIT crystal b* 0 to 1039	note that there are missing batch numbers created by the file statement below, but
FIT crystal c* 0 to 1039	that we do not need to mention this for postrefinement
FIT crystal beta* 0 to 1039	
FIT batch rotx 0 to 1039	
FIT batch roty 0 to 1039	
FIT batch mosaicity 0 to 1039	
ADD PARTIALS 0 to 180 184 to 216	note that the consecutive sequences must be declared here,
217 to 238 1000 to 1039	otherwise program will try to ADD PARTIALS from non-consecutive frames
FORMAT denzo_ip	
	*****High Resolution Data*****
IGNORE OVERLOADS	for the high res. data (long exposure)
SECTOR 0 to 35	
FILE 0 'my_crystal_hi###.x'	the .x files not numbered consecutively. Note renumbering of sector 39.x to batch 36
SECTOR 39 to 183	frame 39.x is next SECTOR after 35.x because 36, 37, 38 were calibration frames
FILE 36 'my_crystal_hi###.x'	
SECTOR 184 to 238	there was change in crystal orientation at 184.x.
FILE 184 'my_crystal_hi###.x'	Note that now we're even with batch numbering
	*****Low Resolution Data*****
INCLUDE OVERLOADS	now we go to low exposure low resolution data
SECTOR 244 to 283	we put low resolution sectors in batches starting
FILE 1000 'my_crystal_lo###.x'	with 1000, just to keep it straight
OUTPUT FILE 'my_crystalhilo.sca'	

Scenario 10: *Thin or few frames (SCALE RESTRAIN and B RESTRAIN)*

Occasionally you may try to scale data which was collected in very thin sectors or in sectors whose width is less than the mosaicity or from data which consists of only a few frames. In these cases, there may not be enough intersections between frames to get accurate scale and B factors. Indeed, what you may see is both scale and B factors ranging all over the place. If things get really bad, the program may crash due to floating point arithmetic exceptions when taking the exponent of unreasonable B factors. This does not mean that the data is unusable, it simply means that the scale and B factors must be restrained. The keywords SCALE RESTRAIN and B RESTRAIN are used for this purpose, and an example of there use is shown below. Note that the restraints only apply to the frames over which you are adding up partially recorded reflections, so you must have the ADD PARTIALS keyword in the command file for this to make any sense.

scalethins.com

NUMBER OF ZONES 10	number of resolution shells for statistics
ESTIMATED ERROR 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04	ESTIMATED ERROR for each resolution shell
ERROR SCALE FACTOR 1.3	multiplicative factor applied to input σ
REJECTION PROBABILITY 0.0001	expected fraction of outliers in the data
[@reject]	[read in rejected reflections]
WRITE REJECTION FILE 0.9	obs. with greater than 90% chance of being outliers are written to the 'reject' file
SPACE GROUP P21	
OUTPUT FILE mydata.sca	OUTPUT FILE containing h, k, l, I, σ_I
REFERENCE BATCH 1	REFERENCE BATCH in scaling procedure.
SCALE RESTRAIN 0.05 B RESTRAIN 0.1	scale and b factors between adjacent frames cannot vary by more than these values
ADD PARTIALS 1 to 90	must have this for restraints to work number of cycles
POSTREFINE 10	of postrefinement
FIT crystal a* 1 to 90	
FIT crystal b* 1 to 90	refine UNIT CELL parameters based on all data from batches 1
FIT crystal c* 1 to 90	through 90
FIT crystal beta* 1 to 90	☛ Space group dependent!
FIT batch rotx 1 to 90	crystal orientation of spindle axis angle will be refined for each
FIT batch roty 1 to 90	batch (frame) separately.
FIT crystal mosaicity 1 to 90	one mosaicity value will be refined
FORMAT denzo_ip	FORMAT of the input intensity data
SECTOR 1 to 90	these values will be substituted in for the ###. Batch numbers
FILE 1 '/frames/Denzo/myxtal###.x'	will start at 1 for file myxtal001.x and increase to 90 for file myxtal090.x

☛ It is not correct to try to "bin" the individual frames into larger batches to try to overcome the problem of few intersections between frames. This is because you then lose the ability to ADD PARTIALS between the new "bins." You can, however, overcome the problem in another way, by including instead the statement NUMBER OF ITERATIONS 0. In this case, the data will not be scaled, but simply merged. Obviously, this has its drawbacks.

Scenario 11: *Space Group Identification*

Scalepack can be used to determine the space group of your crystal. What follows is a description of how you would continue from the lattice type given by *Denzo* to determine your space group. This whole analysis, of course, only applies to enantiomorphic compounds, like proteins. It does not apply to small molecules, necessarily, which may crystallize in centric space groups. If you expect a centric space group, you use any space group which is a subgroup of the Laue category to which your crystal belongs. You also need enough data for this analysis to work so that you can see systematic absences.

To determine your space group, follow these steps:

1. Determine the lattice type in *Denzo*.
2. Scale by the primary space group in Scalepack. The primary space groups are the first Laue groups in each Bravais lattice class in the table which follows. In the absence of lattice pseudosymmetries (e.g. monoclinic with $\beta \approx 90^o$) the primary space group will not incorrectly relate symmetry related reflections. Note the χ^2 statistics.. Now try a higher symmetry Laue group (next down the list) and repeat the scaling, keeping everything else the same. If the χ^2 is about the same, then you know that this is OK, and you can continue. If the χ^2 are much worse, then you know that this the wrong Laue type, and the previous choice was your Laue class. The exception is primitive hexagonal, where you should try $P6_1$ after failing $P3_121$ and $P3_112$.
3. Examine the bottom of the log file for the list of candidate systematic absences. If this was the correct space group, all of these reflections should be absent and their values very small. Compare this list with the listing of reflection conditions by each of the candidate space groups. The set of absences seen in your data which corresponds to the absences characteristic of the listed space groups identifies your space group or pair of space groups. Note that you cannot do any better than this (i.e. get the handedness of screw axes) without phase information.
4. If it turns out that your space group is orthorhombic and contains one or two screw axes, you may need to reindex to align the screw axes with the standard definition. If you have one screw axis, your space group is $P222_1$, with the screw axis along ***c***. If you have two screw axes, then your space group is $P2_12_12$, with the screw axes along ***a*** and ***b***. If the *Denzo* indexing is not the same as these, then you should reindex using one of the two transformations described above in Scenario 5 so that you are following the standard conventions.
5. So far, this is the way to index according to the conventions of the International Tables. If you prefer to use a private convention, you may have to work out own transformations. One such transformation has been provide in the case of space groups P2 and $P2_1$.

Bravais Lattice	Space Groups used first to id Laue group	Candidates		Reflection Conditions along screw axes
Primitive Cubic	P213	195	P23	
		198	P213	(2n,0,0)
	P4132	207	P432	
		208	P4232	(2n,0,0)
		212	P4332	(4n,0,0)*
		213	P4132	(4n,0,0)*
I Centered Cubic	I213	197	I23	*
		199	I213	*
	I4132	211	I432	
		214	I4132	(4n,0,0)
F Centered Cubic	F23	196	F23	
	F4132	209	F432	
		210	F4132	(2n,0,0)
Primitive Rhombohedral	R3	146	R3	
	R32	155	R32	
Primitive Hexagonal	P31	143	P3	
		144	P31	(0,0,3n)*
		145	P32	(0,0,3n)*
	P3112	149	P312	
		151	P3112	(0,0,3n)*
		153	P3212	(0,0,3n)*
	P3121	150	P321	
		152	P3121	(0,0,3n)*
		154	P3221	(0,0,3n)*
	P61	168	P6	
		169	P61	(0,0,6n)*
		170	P65	(0,0,6n)*
		171	P62	(0,0,3n)**
		172	P64	(0,0,3n)**
		173	P63	(0,0,2n)
	P6122	177	P622	
		178	P6122	(0,0,6n)*
		179	P6522	(0,0,6n)*
		180	P6222	(0,0,3n)**
		181	P6422	(0,0,3n)**
		182	P6322	(0,0,2n)

Primitive Tetragonal	**P41**	**75**	**P4**	
		76	**P41**	**(0,0,4n)***
		77	**P42**	**(0,0,2n)**
		78	**P43**	**(0,0,4n)***
	P41212	**89**	**P422**	
		90	**P4212**	**(0,2n,0)**
		91	**P4122**	**(0,0,4n)***
		95	**P4322**	**(0,0,4n)***
		93	**P4222**	**(0,0,2n)**
		94	**P42212**	**(0,0,2n),(0,2n,0)**
		92	**P41212**	**(0,0,4n),(0,2n,0)****
		96	**P43212**	**(0,0,4n),(0,2n,0)****
I Centered Tetragonal	**I41**	**79**	**I4**	
		80	**I41**	**(0,0,4n)**
	I4122	**97**	**I422**	
		98	**I4122**	**(0,0,4n)**
Primitive Orthorhombic	**P212121**	**16**	**P222**	
		17	**P2221**	**(0,0,2n)**
		18	**P21212**	**(2n,0,0),(0,2n,0)**
		19	**P212121**	**(2n,0,0),(0,2n,0), (0,0,2n)**
C Centered Orthorhombic	**C2221**	**20**	**C2221**	**(0,0,2n)**
		21	**C222**	
I Centered Orthorhombic	**I212121**	**23**	**I222**	*****
		24	**I212121**	*****
F Centered Orthorhombic	**F222**	**22**	**F222**	
Primitive Monoclinic	**P21**	**3**	**P2**	
		4	**P21**	**(0,0,2n)**
C Centered Monoclinic	**C2**	**5**	**C2**	
Primitive Triclinic	**P1**	**1**	**P1**	

☛ Note that for the pairs of similar candidate space groups followed by the * (or **) symbol, scaling and merging of diffraction intensities cannot resolve which member of the possible pair of space groups your crystal form belongs to.

Alphabetical Scalepack Keyword Descriptions

ADD

Increments batch numbers by a constant to every batch from this point on until another add command is read. Useful to make unique batch numbers from two or more files which have the same batch numbers inside. For example, ***Denzo*_york1** format embeds the batch number in the .x file.

format	ADD `value`	
default		nothing added
example	ADD `1000`	This will add 1000 to each batch number.

ADD PARTIALS

Tells the program to add partially recorded reflections among consecutive batches, even if the batches do not have consecutive numbering. **Essentially obligatory.**

modifier	`all`	partials are added over all consecutive batches
format	ADD PARTIALS *`start1`* `to` *`end1 start2`* `to` *`end2 ...`* etc	
default	do not ADD PARTIALS	
example	ADD PARTIALS `0 to 49 51 to 99 100 to 149`	

☛ Be sure that ranges of numbers do not overlap.

ANOMALOUS

Flag for keeping Bijovets (I+ and I–) separate in output file. If the ANOMALOUS flag is on, anomalous pairs are considered equivalent when calculating scale and B factors and when computing statistics, but are merged separately and output as I+ and I– for each reflection.

format	ANOMALOUS
default	not turned on (I+ and I– are combined)
example	ANOMALOUS

B RESTRAIN

Can be used to restrain B factor differences from consecutive films or batches. The value which follows the flag represents the amount in $Å^2$ you will allow the B factors to differ from consecutive frames or batches. See also the keyword **SCALE RESTRAIN**.

Format	**B RESTRAIN** *value_(Å²)*
default	not turned on
example	**B RESTRAIN** 0.5

BACKGROUND CORRECTION

The **BACKGROUND CORRECTION** command specifies amount per frame to increase background. Corrects for errors in SDMS (Hamlin) data integration in the software distributed in the late 1980's and early 1990's. The value given after the keyword increases the background by #counts/frame. Valid **for archive file format only**.

format	**BACKGROUND CORRECTION** *value_(#counts/frame)*
default	not turned on
example	**BACKGROUND CORRECTION** 15

DEFAULT B FACTOR

Overall B used only in the absence of an **INITIAL B FACTOR**. You can apply a higher value to lower your R_{merge}. Does not affect the quality of the data.

☛ Applies only to data read-in after this command.

format	**DEFAULT B FACTOR** value
default	0
example	**DEFAULT B FACTOR** 5

DEFAULT SCALE

Overall scale factor used in the absence of an initial scale factor. This is useful if the data are too strong, which is sometimes the case with small molecules. It will reduce the output intensities by the factor entered.

format	**DEFAULT SCALE** value
default	1
example	**DEFAULT SCALE** 10

☛ Useful to reduce overall scale of the data set. If the numbers in the output file are too large, **DEFAULT SCALE** 10 will reduce them 10-fold.

DO NOT REJECT OUTLIERS

Turns off the reject outliers flag.

format	`DO NOT REJECT OUTLIERS`	
default	`DO NOT REJECT OUTLIERS`	*outliers not rejected automatically, but see* `REJECT OUTLIERS` *for more discussion.*

END FIT

Provides a logical end for the fit command. At the moment this has **no effect**, but may be implemented in future versions of the program.

ERROR SCALE FACTOR

This is a single multiplicative factor which is applied to the input σ_I. This should be adjusted so the normal χ^2 (goodness of fit) value that is printed in the final table of the output comes close to 1. By default the input errors are used (`ERROR SCALE FACTOR` = 1). It applies to the data which are read after this keyword, so you can apply different error scale factor to subsequent batches by repeating this input with different values.

format	`ERROR SCALE FACTOR` *`value`*	
default	`ERROR SCALE FACTOR 1`	
example	`ERROR SCALE FACTOR 1.3`	*good starting value for format denzo_ip*

ESTIMATED ERROR

Estimate of the systematic error for each of the resolution shells. **There must be exactly the same number of error estimates here as there are** `NUMBER OF ZONES`. So if you have 10 zones, you need 10 numbers following the keyword estimated error - one for each zone.

☛ The error estimates to do not all have to be the same. The estimated error applies to the data which are read **after** this keyword, so you can apply different error scale factor to subsequent batches by repeating this input with different values. This is an important point if you enter data from a previous Scalepack output that does not need its σ to be increased.

The error estimates should be approximately equal to the R-factor in the table at the end of the output for resolution shells **where statistical errors are small**, namely the earlier resolution shells where the data is strong. This is a crude estimate of the systematic error, to be multiplied by I, and is usually invariant with resolution. Default = 0.06 (i.e. 6%) for all zones.

format	`ESTIMATED ERROR` *`value1 value2 value3 value4`* ...etc
default	each value set to `0.06`
example	`ESTIMATED ERROR 0.02 0.03 0.03 0.03 0.04 0.04` (6 zones)

EXTEND PARTIALS / DO NOT EXTEND PARTIALS

Some partially recorded reflections may be predicted by *Denzo* or *Scalepack* to start or end their Bragg condition in between consecutive frames due to small variations in crystal orientation from frame to frame. For these reflections only, there are two choices of defining where the reflection started (or ended): including the extra frame, or not. **EXTEND PARTIALS** tells *Scalepack* to include this extra frame. It only affects a very small fraction of the reflections. Opposite of **DO NOT EXTEND PARTIALS.**

format	**EXTEND PARTIALS**
default	this is the default

FILE

This specifies the files read by *Scalepack*. The input has two components. The first is a number. The second is a file name, which usually contains wildcard characters (###) that are incremented by the **SECTOR** command. The number which follows **FILE** specifies the **starting batch number**. A batch, previously called a **film**, can be as small as a single .x file (or the equivalent). It can be a group of .x files, even an entire data set. The most frequent content of a batch is a single .x file.

This conversion of files into batches is particularly useful if you want to scale more than one data set together. For example, let's say you want to scale 10 oscillation frames (numbered 1 through 10) from first data set with 37 oscillation frames (numbered 1 through 37) from other one. The **FILE** statement will take each of the individual .x files and assign them a batch number. Thus, you would enter something like this: **FILE** `1 'setone###.x'` and **FILE** `101 'settwo###.x'` (see Scenario 5d). Thus, batch numbers 1 - 10 will correspond to files `setone001.x, setone002.x`, etc. (assuming you used the **SECTOR** `1 to 10` command above **FILE** so that the wildcards would be substituted with numbers). The batch numbers 101 - 137 will correspond to files `settwo001.x, settwo002.x`, etc.

☛ the ### is replaced by the sector argument, not by the batch number.

format	**FILE** *`value 'filename'`* (see also note below)
defaults	none
example	**FILE** `101 '/frames/scale/lysoz###.x'`

☛ The **FILE** must come after **FORMAT** because the syntax depends on which input format is being read. The **FILE** must not be followed by number in the case of `archive, denzo_york, and denzo_york1` formats, where the number after the word **FILE** is not given because the batch numbers are already stored in the file. If you want to change the batch numbers in these file formats, see the **ADD** command described above.

FIT

Tells the program what parameters to fit in **postrefinement** and specifies the group of files over which the fitting is to be performed. Postrefinement fit can be applied to an entire set of batches (one batch being the entire set of frames, for example) using the modifier `crystal`, or to each individual input file using the modifier `film` or `batch`.

modifiers	`crystal`	specifies that the fit operation is over the **entire set** of frames specified by a range and restrains the fit parameter to have exactly the same value over this range.
	`batch`	specifies that the fit operation is performed on **each member of the set** of frames specified by a range.
	`film`	alias for `batch`
parameters	`a*, b*, c*`	unit cell lengths. Values returned are real space
	`alpha*`	unit cell angles. Values returned are real space.
	`beta*`	
	`gamma*`	
	`rotx`	crystal orientation parameters deduced in *Denzo*
	`roty`	
	`rotz`	
	`wavelength`	incident wavelength. `Untested`.
	`mosaicity`	mosaicity as defined in *Denzo*, in degrees
format	`fit modifier parameter1 filmnumber to filmnumber`	
default	the default is that nothing is fit unless specified	
examples	`FIT crystal roty 1 to 137`	
	`FIT crystal mosaicity 1 to 5 7 to 10 102 to 104 to 137`	
	`FIT batch rotx 1 to 137`	

Most mistakes in *Scalepack* can be attributed to errors in `FIT` commands because the program is very sensitive to mistakes in the batch numbers. If you input non-existent batch numbers or define overlapping ranges (e.g. `1 to 10 5 to 20`), the program is likely to fail in a strange way. If you specify a range of numbers, the program will only use the batch numbers that exist within the range. For example, if your batch numbers go from 1 to 40 and 70 to 90, you can get away with saying, say, `FIT batch (`*`parameter`*`) 1 to 90`, which is the same as `FIT batch (`*`parameter`*`) 1 to 40 70 to 90`. For `FIT crystal`, these two inputs are not equivalent. In the case of `FIT crystal (`*`parameter`*`) 1 to 90`, one value will be fit for all batches. In the case of `FIT crystal (`*`parameter`*`) 1 to 40 70 to 90`, two values will be fit, one for each range. Note that different parameters may be fit over different ranges and either over `batch` or `crystal`. You can also mix `batch` and `crystal` for the same parameters.

☛ **Do not fit unit cell parameters restrained by space group symmetry**. For example, if you have space group P3, you must not fit `b*`.

☛ **Do not** `FIT batch rotz` because this parameter is very poorly defined by the intensities of observed partial reflections. This is another very common mistake.

☛ Unless you know what you are doing, **do not** `FIT crystal rotx roty rotz`, because if the spindle is even slightly misaligned, the assumption that there is only one crystal orientation parameter for a large sweep of data will force incorrect restraints on the crystal orientation refinement.

☛ About **fitting rotations:** Changes in rotations, like crystal rotx roty and rotz, are expressed as small rotations, call them Δ, about the laboratory frame of reference. These Δ: Δ_x, Δ_y, and Δ_z, are used because to a first approximation they commute with one another (commute means that the order in which they are applied is irrelevant). This in turn is because these Δ are small, typically less than one degree. The crystal or cassette rotations, R_x, R_y, and R_z, on the other hand, do not commute with one another because their values tend to be large (much greater than one degree). So when you ask *Scalepack* postrefinement or *Denzo* to fit these rotations, what is actually happening is that the **Δ are being refined**. After each refinement cycle, the Δ are converted into changes in R_x, R_y, and R_z by a (complicated) algebraic relation. Those of you with sharp eyes will have noticed in *Denzo* that the shifts reported by the program when fitting the crystal rotations do not correspond to the changes in the rot values. This is because the shifts reported are the Δ, not the changes in rotx, roty, and rotz. The other reason for fitting Δ rotations the way they are defined in *Denzo* is to make them have a more intuitive correlation with the other parameters. Otherwise, changes in **crystal rotz** would not correlate with **cassette rotz**. This is of more importance in *Scalepack*, where only rotx and roty, and not rotz, are refined.

☛ About **fitting unit cell** parameters: In both *Denzo* and Scalepack the unit cell is fitted in **reciprocal space**, not real space. This means that for a non-orthogonal space group, refining the value of **a*** may end up changing the values of **b** and **c**, even though **b*** and **c*** remain the same. The same is true for the angles: fitting α* may end up changing β and γ, even though β* and γ* remain the same. So what you may notice sometime if you are not careful is that when you ask the program to fit crystal **a***, but not **b***, **c***, and the angles, then **a** will not be a constant. **a*** will be constant, but when converted back to **a** when the other unit cell parameters have not been changes, **a** will not. The moral**: when fitting unit cells, fit all the relevant parameters.**

FIT B

This flag tells the program to refine B factors of every batch from the very first cycles of refinement. This is in contrast to the default procedure, where the B factors are fit only after the convergence of the scaling. In the default procedure, if scaling does not converge in 20 (default) cycles of refinement, B factors will be not be fitted. The `FIT B` command can override this. Not to be confused with the postrefine `FIT B*` command described above. **You cannot postrefine B (temperature) factors.**

format	`FIT B`
default	this flag is turned **off**
example	`FIT B`

FIX B

This flag tells the program not to fit B factors at all. Usually it is combined with the input of the B factors you want to apply but do not wish to refine anymore or it is used for frozen crystals where you do not expect significant decay.

format	`FIX B`
default	turned **off**; B factors are fit after the scale factor refinement converges.
Example	`FIX B`

FIXED WINDOW

For **Hamlin** archive files a fixed window of 3, 5, 7, or 9 frames or the original Hamlin-determined window of frames may be used for summing a reflection. **Valid for archive file format only.**

format	`FIXED WINDOW value`
default	uses the **Hamlin** definition of the window
example	`FIXED WINDOW 7`

FORMAT

This keyword specifies the format of the input *hkl* and intensity data. Input data can come from any of nine types of files. This program requires this keyword to properly read the files.

modifiers	`denzo_ip`	*from frames processed with Denzo*
	`denzo_york1`	*output created with Denzo option york*
	`scalepack`	*from Scalepack output file*
	`rigaku raxis`	*binary **R-Axis** software output*
	`xds profit.hkl`	*binary output from XDS output file*
	`madnes procor intout`	*binary madness output*
	`madnes procor ascii`	*ascii madness output*
	`xengen urf`	*binary output from xengen. Also have to supply program with info about frame width, in degrees, using frame width keyword.*
	`Archive`	*binary output from **Hamlin** software*
format	`FORMAT modifier`	
default	`denzo_ip`	
example	`FORMAT denzo_ip`	☛ *if you use one of the binary formats that the data must be scaled on the same type of computer that created the binary files due to incompatibilities in number representations between computers.*

FRAME WIDTH

Only for URF files created by *Xengen*. Other formats do not need this specification because *Scalepack* can read this information off the file header. The oscillation range for each frame.

format	`FRAME WIDTH value`
default	there is no default, this input is required for this format
example	`FRAME WIDTH 0.2`

HKL MATRIX

Matrix for re-indexing. Default = unit matrix. This matrix is applied to the *hkl*'s as they are read in (applies to data read in after this command):

```
h' = (1)*h + (2)*k + (3)*l
k' = (4)*h + (5)*k + (6)*l
l' = (7)*h + (8)*k + (9)*l
```

Input is in the order (1....9). HKL MATRIX works with postrefinement.

☛ program will not accept a matrix which has a negative determinant

format	HKL MATRIX `value11 value12 value13 value21 value22 value23` ...etc
default	the default is the unit matrix
example	HKL MATRIX `0 1 0 1 0 0 0 0 -1` transforms *hkl* into *k h -l*, has the effect of switching *a* and *b*. -1 is to keep the determinant positive.

HKL SCALE

Divides each *h*, *k*, and *l* by the input values. Useful for reducing the unit cell volume, particularly after *hkl* matrix transformation. Rarely needed. Example, index data originally in C222 and want you data in, say, P3, you would apply HKL SCALE of 2 2 1, and HKL MATRIX of 1 1 0 1 -1 0 0 0 -1 so that the new indices would have values of `h +k/2`, `h-k/2`, and `-l`.

format	HKL SCALE `value1 value2 value3`
default	`1 1 1`
example	HKL SCALE `2 2 1`

HKL SHIFT

Used to make a quick test of **misindexing**. Adds the specified integer vector to each original *h*, *k*, and *l*. If you are successful in using this, congratulations, you now have to reprocess your data in *Denzo* with the correct indexing.

format	HKL SHIFT `integer integer integer`
default	`0 0 0`
example	HKL SHIFT `0 1 0` *we'll assume you weren't too far off*

IGNORE OVERLOADS

Opposite of INCLUDE OVERLOADS. This is **not the default**. Useful if you collect data at low and high exposures and is useful to ignore the saturated reflections at the high exposures. Applies to data read in after this command.

INCLUDE OVERLOADS

INCLUDE OVERLOADS is flag for whether fitted profiles with some pixels missing (typically due to overload) should be included in the scaling. Affects only *Denzo* image plate output files

(formats denzo_ip, denzo_york1). Note that for summed profiles this does not apply because only profile fitting can estimate the value of the overloaded pixels.

format	**INCLUDE OVERLOADS**
default	**INCLUDE OVERLOADS**

INITIAL B FACTOR

Table of starting **B factors**, one per batch, beginning on the record after the title. After running the program once, the output table of B factors can be cut out and pasted in here, for example, when you have set the **FIX B** flag and are not refining B values anymore.

format	**INITIAL B FACTOR** `batch no. b value batch no. b value` ...
default	value of the default B factor, which is defaulted to 0
example	**INITIAL B FACTOR** `1 0.0 2 0.1 3 0.1 4 0.1 5 -0.2` ...etc

INITIAL SCALE

Table of **starting scale factors**, one per batch, beginning on the record after the title. After one run of the program, the output table of scale factors can be cut out and pasted in here. This table is not required. If it is not included in the control input, then the **DEFAULT SCALE** is used (this is 1 unless otherwise specified).

☛ If the initial scale factor is set to zero, that frame is ignored in the scaling and refinement.

format	`batch no. scale value batch no. scale value` ...
default	value of the default scale factor, which is defaulted to 1
example	**INITIAL SCALE FACTOR** `1 1.0 2 0.9 3 0.0 4 0.85 5 1.1` ...etc.

INPUT / @

Redirection. Tells the program to read the file which follows the keyword. Same as in *Denzo*.

format	`@filename`
default	no redirection
example	`@reject`

MERGE

Flag that tells the program to merge (combine, average) reflections with the same unique index. This is the *Scalepack* default.

format	**MERGE**
default	this is the default and need not be specified

MOSAICITY

Allows you to input the value of the mosaicity of the data set. This is normally read from the header of the `.x` file.

format	MOSAICITY `value`
default	read from the header of the .x file
example	MOSAICITY `0.5`

NO ANOMALOUS

Opposite of the keyword ANOMALOUS. Causes I+ and I- to be merged. This is the default.

format	NO ANOMALOUS
default	this is the default

NO MERGE

Flag for the output of **unmerged** (reflections with the same unique *hkl* are not combined) data. Opposite of MERGE. This is a very handy for specialized work. This flag has subsidiary modifiers `include partials, original index` or as default `include no partials`

modifiers	`include partials`	all observations, both fully and partially recorded, are included in the output. The output will consist of the unique *hkl*, batch number, asymmetric unit, I, σ, and fractionality of the reflection. There is no information about I+ and I-, although it may be possible to get this in subsequent versions of the program.
	`include no partials`	only fully recorded reflections and those fully recorded reflections created by the summation of partials are included in the output. Partials which cannot be summed to a fully recorded reflection are lost. The output will consist of the unique *hkl*, batch number, asymmetric unit, I, and σ. This is the **default** for NO MERGE. There is no information about I+ and I-.
	`original index`	the output will also contain the original (not unique) *hkl* for each reflection. This is **designed for MAD/local scaling** work. The `original index` modifier only works with the default INCLUDE NO PARTIALS. The output will consist of the original *hkl*, unique *hkl*, batch number, a flag (0 = centric, 1 = I+, 2 = I-), another flag (0 = *hkl* reflecting above the spindle, 1 = *hkl* reflecting below the spindle), the asymmetric unit of the reflection, I (scaled, Lorentz and Polarization corrected), and the σ of I. The format is (6i4, i6, 2i2, i3, 2f8.1).
format	NO MERGE `modifier`	
example	NO MERGE `original index`	

☛ This command is also useful if you just want to combine all of the information contained in multiple .x files into a single file. Simply read in all of the .x files, don't do any scale or B factor refinement, and output NO MERGE `include partials`.

NO PROFILE TEST

This is a flag which operates only on **ARCHIVE** format files. Tells *Scalepack* that reflections with weird profiles should not be rejected. Opposite of **PROFILE TEST**.

format	**NO PROFILE TEST**
default	this is not the default

NUMBER OF ITERATIONS

Number of **cycles for refinement** of scale and B-factors. Default value is 20 cycles. If it is set to 0, the program computes statistics, merges reflections with the same unique *hkl*, and writes the output file based on the initial scale and B factors. Normally you do not have to specify this unless you want to avoid scaling.

format	**NUMBER OF ITERATIONS** `integer`
default	20 iterations
example	**NUMBER OF ITERATIONS** `0`

NUMBER OF ZONES

Number of **resolution shells** the data is divided into for the basis of calculating statistics. This input is required and must match the number of zones specified under the **ESTIMATED ERROR** keyword. Handy tip: it's nice to set up the **NUMBER OF ZONES** to equal the number of zones used by *X-plor* for the output of refinement statistics by shell, for when you get around to publishing your data and refinement statistics together in a paper.

format	**NUMBER OF ZONES** `integer`
default	this input is required
example	**NUMBER OF ZONES** `10`

ORIENTATION AXIS 1

Three integer vector which describes the orientation of the vertical axis of the crystal. Equivalent to *Denzo* and *Scalepack* keywords **VERTICAL AXIS**. This information is usually not input by the user but subsequent versions will read it from the header of the .x files. This command does not affect scaling or postrefinement, however it does affect the values of the misorientation angles reported in the *Scalepack* log file.

format	**ORIENTATION AXIS 1** `integer integer integer`
default	`1 0 0`
example	**ORIENTATION AXIS 1** `0 1 0`

ORIENTATION AXIS 2

Same as **ORIENTATION AXIS 1** except describes the spindle axis of the crystal. Equivalent to *Denzo* and *Scalepack* keywords **SPINDLE AXIS**. Otherwise the same as above.

format	ORIENTATION AXIS **2** `integer integer integer`
default	0 0 1
example	ORIENTATION AXIS **2** `1 0 0`

ORIGINAL WINDOW

Refers to the window size of ARCHIVE format files. Opposite of FIXED WINDOW.

OUTPUT ANOMALOUS

alias for the keyword ANOMALOUS.

OUTPUT FILE

Name of file for output of scaled measurements. A new file will be created if none exists, but a pre-existing file will be overwritten. Maximum of 80 characters allowed in the file name. Depending on whether ANOMALOUS flag is set, there are one or two sets of I and σ(I) per reflection. Output is *h*, *k*, *l*, I, σ(I) (NO ANOMALOUS flag) or *h*, *k*, *l*, I+, σ(I+), I-, σ(I-) (ANOMALOUS flag) in format (3I4, 4F8.1). If the NO MERGE flag has been set, unmerged data are output as *h*, *k*, *l*, asymmetric unit #, I, σ(I) in format (4I4,2F8.1) and the ANOMALOUS flag has no effect. For NO MERGE `original index` the format is: original *hkl*, unique *hkl*, batch (film) number, a flag (0 = centric, 1 = I+, 2 = I-), another flag (0 = *hkl* reflecting above the spindle, 1 = *hkl* reflecting below the spindle), the asymmetric unit of the reflection, I (scaled, Lorentz and Polarization corrected), and the σ(I). The format is (6i4,i6,2i2,i3,2f8.1).

format	OUTPUT FILE `name` use single quotes for safety
default	you must specify the output file name
example	OUTPUT FILE `'/people/myname/LYSOZ.sca'`

POLARIZATION

This command allows you to correct a mistaken polarization/monochromator value entered when you ran *Denzo*. It saves you from the chore of reprocessing all of your original images once you have learned of a mistake. This is mainly for synchrotron users who collect very high resolution data.

format	POLARIZATION *Denzo value* `corrected` *value*
default	the polarization is read off the header of the .x file
example	POLARIZATION *Denzo* `0.0 corrected 0.9`

POSTREFINE

Number of cycles of postrefinement do be done. Works in conjunction with the FIT commands described above. If you set the number of postrefinement cycles to zero, then postrefinement will be skipped even if you have FIT commands in your control file.

format	POSTREFINE `integer`
default	postrefinement is not done unless you specify it
example	POSTREFINE 10

PRINT

This flag tells the program **to print the results** of the specified calculation **after every refinement cycle**. Useful in the case where one scales non-isomorphous data and wants to make the outlier list short by using the statement PRINT `total chi2 200` PRINT `single chi2 100`.

modifiers	`total chi2`	cutoff for printing poor reflections. Total χ^2 for all measurements of a reflection. Default = 5^2
	`single chi2`	cutoff for printing poor reflections. χ^2 for any single measurement of a reflection. Default = 3.5^2.
	`correlation / no correlation`	prints correlation matrix. Default: `no correlation`
	`solution / no solution`	prints new scale and B factors. Default: `no solution`
	`shifts / no shifts`	prints changes to scale and B factors.
format	PRINT `modifier`	
defaults	PRINT `total chi2`	
	PRINT `single chi2`	
	PRINT `no correlation`	
	PRINT `no solution`	
	PRINT `no shifts`	

New scale and B factors will always be printed if the number of fitted parameters exceeds 300.

PROFILE TEST

PROFILE TEST is a flag that can be turned on and off as desired. This is the default. Bad profiles can be used to reject a DCREDUCE summed *hkl*. Valid for ARCHIVE file format only. Opposite of NO PROFILE TEST.

PROFILES FITTED

This flag tells *Scalepack* how the profiles were treated by the indexing program (e.g. *Denzo*). The two choices for profiles are fitted and summed. Valid for *denzo_ip*, *Denzo*_york1, and *xengen* **urf** formats. Most of the time people fit profiles, so this is the default and need not be specified in the command file.

format	PROFILES FITTED
default	this is the default

PROFILES SUMMED

Opposition of PROFILES FITTED. This is not the default and must be specified if true.

RECSQ

Metric tensor description of the reciprocal space unit cell. Why would you ever use this when you can specify the real space constants?

format	RECSQ `value1 value2 ... value6`
default	same as UNIT CELL
example	RECSQ `0.0001 0 0 0.0004 0 0.0009` [to describe unit cell 100 50 33 90 90 90.]

REFERENCE BATCH

Specifies which batch or film or set of batches or films will be the reference for the scaling and B refinement. The scale and B factor for these are not refined. More than one film or batch may be used as the reference. This is important only for crystals which decay during data collection. If the crystal is frozen and does not decay, then the default may be used, which is to let the eigenvalue filter define the overall scale and B factor. With a large number of batches, reliance on the eigenvalue filter is a little bit dangerous, so you should consider using a reference batch number in those cases.

This keyword is entirely equivalent to the keywords REFERENCE BATCHES, REFERENCE FILM, AND REFERENCE FILMS. The others exist because some people have a gramatical hangup about using the singular to describe more than one object.

format	REFERENCE BATCH `integer integer integer` ... etc.
default	no reference batch. **Eigenvalue filter** defines overall scale and B
example	REFERENCE BATCH `1 3 4 5`

REFERENCE BATCHES, FILM, FILMS

Same as reference batch

REJECT HKL

Flag which tells the program to reject a list of individual *h, k, l*'s which follow. This is useful for iterative rejection cycles, since the file containing the rejected reflections can be reread. Each record after the title contains one reflection with the variables h, k, l (original index) and film number in free format. This information is most easily edited from the printed output of a previous run, can contain the whole line. One can read it from a reject file with command `@reject`.

format **REJECT HKL**

default this is not the default

REJECT OUTLIERS

Automatic rejection of outliers. Not yet reliably implemented. This is supposed to replace multiple rounds of running *Scalepack*. The idea is that when this keyword is set, *Scalepack* runs once, makes a `reject.dat` file, then runs again, reads the `reject.dat` file and scales the data based on the reduced set of observations.

REJECTION PROBABILITY

Applies **Bayesian rejection of outliers**. Rejected observations are written to the `reject.dat` file (see **WRITE REJECTION FILE**). The whole *hkl* (all original *hkl*s with the same unique *hkl*) with at least one observation with a probability of being an outlier greater than or equal to the value specified after the **WRITE REJECTION FILE** keyword (default 0.9) are written to the log file. This is an estimation on your part of how frequently you expect any observation to be an outlier. In principle, the rejection probability should be about equal to the number of outliers divided by the number of observations. A good value to use is 1/10,000, i.e. 0.0001, for normal good data. If you have a non-random signal in your background (e.g. satellite crystals, malfunctioning detector, ice rings) then you will probably have to increase the rejection probability. If you do not want to generate a reject list in the log file then omit the keyword. A comparison of $R_{\text{-linear}}$ and $R_{\text{-squared}}$ is often helpful for deciding whether to increase the rejection probability. See the discussion following *Scenario 1* for more on this.

The rejection algorithm used in *Scalepack* is the most sensible and statistically sound outlier rejection algorithm. Unlike some other programs, the *Scalepack* outlier rejection is based on comparing differences to σ. So it will typically reject, say 4 or 5 σ outliers. Some reflections with a large discrepancy from the average value of I may simply represent a lack of adequate statistics in measurement and not a "mistake" (non-random error) in measurement. If a reflection has a large discrepancy from the average and a large σ, its contribution to the average will be very small anyway.

Here's an example of how the outlier rejection algorithm works:

Input data (after scaling)		
Obs #	I	σ
1	10.0	0.1
2	10.1	0.1
3	9.9	0.1

4	11.0	0.1
5	20.0	10.0

Although you may think that {5} is the outlier, in fact the consistent set of observations will be {1, 2, 3, 5}, and {4} will be the outlier. Why is this? A consistent average will be 10.0 (or more exactly 10.000003), because observation 5 has a 1/1000000 lower statistical weight than the other observations, due to its large σ. Observation 4, on the other hand, which was measured more accurately, will be 8.7 σ deviations from the average of the remaining observations because the expected error of the difference 4 - <{1,2,3,5}> is 0.115, and 8.7 = (11.0 - 10.000003) / 0.115. So the observation with the largest deviation from the average is not necessarily a statistical outlier.

☛ R_{-merge} (after outlier rejection) will be very bad, 25%, even if the average is measured extremely well, with a sigma of 0.057%! This has to do with R_{-merge} being an unweighted statistic, and I/σ being a weighted statistic. If some other program rejected observation 5 (for no good reason) and observation 4 (for good reason), R_{-merge} would be 0.067%. Practice shows that most programs would reject observation 5 because many users want a good R_{-merge}. And sometimes these program would not even flag the true outlier - observation 4!

format **REJECTION PROBABILITY** `value`

default `0.0001`

example **REJECTION PROBABILITY** `0.0001`

RESOLUTION

Minimum **d-spacing** for this run. Default is the maximum resolution found in the input data. One can supply two numbers in any order and they will be minimum and maximum d-spacings.

format **RESOLUTION** `value [value]`

default highest resolution detected in .x file

examples **RESOLUTION** `10 2.2`

RESOLUTION `2.2`

RESOLUTION STEP

Changes the number of reflections per shell for the purposes of printing out statistics. Formally, it represents the exponent of the zone volume calculation. Normally, this is 3, because the volume of a sphere goes as radius cubed, so all the statistics shells will have the same number of reflections. Changing the resolution step could be useful to prepare a table of statistics to compare with other programs which may print out the statistics differently.

format **RESOLUTION STEP** `value`

default **RESOLUTION STEP** `3`

example **RESOLUTION STEP** `2.5`

SCALE ANOMALOUS

This is the flag for keeping Bijvoets (I+ and I-) separate both in scaling and in the output file. If the SCALE ANOMALOUS flag is on, anomalous pairs are considered non-equivalent when calculating scale and B factors and when computing statistics, and are merged separately and output as I+ and I- for each reflection.

☛ This is a dangerous option because scaling may be unstable due to the reduced number of intersections between images. The danger is much larger in low symmetry space groups. SCALE ANOMALOUS will always reduce R_{merge}, even in the absence of an anomalous signal, because of the reduced redundancy. However, χ^2's will not be affected in the absence of an anomalous signal.

format	SCALE ANOMALOUS
default	definitely not a default

SCALE RESTRAIN

Can be used **to restrain scale factor differences** from consecutive films or batches. The value which follows the flag represents the amount you will allow the scale factors to differ from consecutive films or batches. It adds a factor of $(\text{scale1} - \text{scale2})^2/(\text{scale restrain})^2$ to the target function minimized in scaling. This only applies to batches between which you add partials. For very thin frames, this is almost obligatory. The value should roughly represent the expected relative change in scale factors between adjacent frames.

format	SCALE RESTRAIN `value`	
default	not turned on	
example	SCALE RESTRAIN `0.01`	(expect 1% change between adjacent frames)

SECTOR

Substitutes for the ### wildcard to specify a group of files to be read. See FILE keyword.

format	SECTOR `integer to integer`
default	no default
example	SECTOR `1 to 40`

SECTOR WIDTH

Sector width can be specified in degrees. A pseudofilm is a sector width's worth of data from one detector. Valid for area detector data only, default value 5 degrees.

format	SECTOR WIDTH `value`
default	5 degrees
example	SECTOR WIDTH `3.0`

SIGMA CUTOFF

Cutoff for **rejecting measurements** on input. Default = -3.0. Be careful if you increase this!

☛ What is the rationale for using σ cutoff -3.0 in *Scalepack*? Wouldn't you want to reject all negative intensities? Why shouldn't you use a σ cutoff 1.0 or zero? The answer is as follows: The **best estimate of I may be negative**, due to **background subtraction** and **background fluctuation**. Negative measurements typically represent random fluctuations in the detector's response to an X-ray signal. If a measurement is highly negative (≤ -3σ) than it may be more likely the result of a mistake, rather than just random fluctuation. If one eliminates negative fluctuations, but not the positive ones before averaging, the result will be highly biased. In *Scalepack*, `SIGMA CUTOFF` is applied before averaging. If one rejects all negative intensities before averaging a number of things would happen:

1) The averaged intensity would always be positive;
2) For totally random data with redundancy 8, in a shell where there was no signal, , there would be on average 4 positive measurements, with average intensity one σ. This is because the negative measurements had been thrown out. So the average of the four remaining measurements would be about 2 σ! This would look like a resolution shell with a meaningful signal!
3) R_{-merge} would be always less than the R_{-merge} with negative measurements included
4) A `SIGMA CUTOFF` of 1 would improve R_{-merge} even more, by excluding even more valid measurements!

Why should this worry you? Exclusion of valid measurements will deteriorate the final data set. One may notice an **inverse relationship between R_{-merge} and data quality** as a function of `SIGMA CUTOFF` So much for using R_{-merge} as any criterion of success.

Even the best (averaged) estimate of intensity may be negative. How to use negative I estimates in subsequent phasing and refinement steps? The author of *Scalepack* suggests the following:

1) You should never convert I into F
2) You should square F_{calc} and compare it to I. Most, but not all of the crystallography programs do not do this. That is life. In the absence of the proper treatment one can do approximations. One of them is provided by French and also by French and Wilson and implemented in the CCP4 program TRUNCATE. A very simplified and somewhat imprecise implementation of TRUNCATE is this:

if I > σ(I), F=sqrt(I)

if I < σ(I), F=sqrt(σ(I))

format `SIGMA CUTOFF value`

default `-3`

example `SIGMA CUTOFF -2.5`

SPACE GROUP

Space group symbol from the list below. This input is required! The space group may be entered as a name (e.g. P212121) or as a number (e.g. 19, for the same space group). Most of the numbers correspond to those of the International Tables. The numbers above 230 are non-standard definitions of space groups.

1	P1	89	P422	154	P3221	207	P432
3	P2	90	P4212	155	R32	208	P4232
4	P21	91	P4122	168	P6	209	F432
5	C2	92	P41212	169	P61	210	F4132

16	P222	93	P4222	170	P65	211	I432
17	P2221	94	P42212	171	P62	212	P4332
18	P21212	95	P4322	172	P64	213	P4132
19	P212121	96	P43212	173	P63	214	I4132
20	C2221	97	I422	177	P622	303	P2C
21	C222	98	I4122	178	P6122	305	B2
22	F222	143	P3	179	P6522	318	P21221
23	I222	144	P31	180	P6222	401	C1
24	I212121	145	P32	181	P6422	403	P21C
75	P4	146	R3	182	P6322	446	H3
76	P41	149	P312	195	P23	455	H32
77	P42	150	P321	196	F23	501	I1
78	P43	151	P3112	197	I23	503	I2
79	I4	152	P3121	198	P213	505	C21
80	I41	153	P3212	199	I213		

Notes to particular space groups:

146	R3	R3 in hexagonal setting
446	H3	R3 in primitive setting
155	R32	R32 in hexagonal setting
455	H32	R32 in primitive setting
401	C1	Non-standard, but useful to make angles close to 90.
501	I1	Non-standard, but useful to make angles close to 90.
303	P2C	P2, C axis unique
403	P21C	P21, C axis unique
305	B2	like C2, B face centered, c axis unique
503	I2	Non-standard, but useful to make beta angle close to 90.

SPINDLE AXIS

Alias for **ORIENTATION AXIS 2**.

UNIT CELL

Real cell specified as a, b, c, alpha, beta, gamma. **UNIT CELL** is included in the header of *Denzo*_IP, *Denzo*_york1, ARCHIVE and *Scalepack* files, otherwise it must be included.

format	UNIT CELL `value value value angle angle angle`
default	the value from the first header encountered
example	UNIT CELL `50 62 100.3 90 90 90`

☛ the postrefined value of the unit cell constants is not used nor is it output in the *Scalepack* output file. You must get this information from the log file if you are interested in it.

VERTICAL AXIS

Same as **ORIENTATION AXIS 1**

WRITE BADDIES

Writes `*.xrej` files so that reflections from the reject file may be displayed by *XdisplayF*. Not fully implemented. Will terminate the program and prevent scaling and postrefinement. Note that the command `@reject` must precede the input of the .x files.

WRITE REJECTION FILE

Creates a file with *hkl*'s to be rejected. UNIX file name: `reject`, VMS file name: `REJECT.DAT`. Reject file is created if it does not exist. If the file exists it is overwritten.

You can specify the threshold for what you consider to be a **rejectable probablity**. The default is 0.9, which is fairly safe, but you may want to decrease this to, say, 0.5 on later rounds of rejection.

format	**WRITE REJECTION FILE** `value`
default	does not write the rejection file, but if it does, the default value is 0.9
example	**WRITE REJECTION FILE** `0.5`

HKL Q & A

This chapter presents a selection of the questions sent to the *hkl* server over the past year or two. They were selected to illustrate points which may not be covered in the main portion of the *Denzo* and *Scalepack* manuals. Answers are by Zbyszek Otwinowski and Wladek Minor. Spelling and grammar have been cleaned up only to a limited extent. The general topic headings are listed here:

2θ

Absorption correction factors

Auto Editing

Autoindexing

Box Definition

CCP4 Scaling

Computer problems and questions

Confusion & other

Consequences of Profile fitting errors in Denzo

Corrections

Denzo Output

Denzo Problems

Display

Example of Bad Detector Pathology

Ignore

Mosaicity

Negative Intensities

Overload

Refinement

Rhombohedral Lattices

Twinned Crystals

*2*Q

Q: *In HKL manual we found a notation that 2*q *could go into* CASSETTE rotx*. However, this doesn't seem to work. According to the R-axis, we have set 2*q *to -10 degrees. Does the 2*q *option work correctly?*

The 2θ swing is described by CASSETTE rotx in *Denzo*, with the opposite sign to the R-axis definition for 2θ. DISTANCE is defined in *Denzo* as the distance between crystal and a point there beam intersects detector; it is = R-axis distance / cos(2θ). X BEAM and Y BEAM are the actual position of the beam on the detector during exposure at the current 2θ (not at 2θ =0). X BEAM and Y BEAM change with 2θ, mostly Y BEAM. If the detector is aligned perfectly X BEAM does not change. The Y BEAM, in extreme cases, may be outside the detector. The correct value may be negative or more than 200 in such case. The definition in *Denzo* is that the cassette (detector) center of rotation is the point where the beam intersects detector surface. R-axis (2θ) rotation definition makes the crystal at the center of detector rotations. All options in *Denzo*, including autoindexing, work with any detector orientation. Only the resolution display in *XdisplayF* is incorrect for detector with 2θ non-zero.

Absorption Correction Factors

Q: *Do you have any information about the use of the air absorption card in Denzo. The manual says that reflections are attenuated by e (2.78) but it says nothing about the angular dependence, or the wavelength dependence, both critical factors in determining the true effect of air absorption on reflection intensity.*

The air absorbs equally in all directions! The program calculates air path and corrects for the absorption in the air. It is a very small correction.

C: *I think I've answered my own question about* **AIR ABSORPTION LENGTH** *appropriate for 0.9 A synchrotron radiation. It works out to 4500 mm using appendix 3 in Helliwell's book and the CRC Handbook of Chemistry & Physics.*

C: *With regard to our discussion of air absorption, I looked up what is available in the International Tables and derived the equation log(mu) = 3.014*log(lambda) - 0.498 from some of the tabulated values. This gives the following values for input to Denzo:*

lambda	*air absorption length*
1.7425	590 mm
1.5418	854 mm (*Denzo* default is 860)
1.5000	928 mm
1.0000	3145 mm

Auto editing

Q: *The program seems to reject most of the bad reflections that have interference from ice. But still I see big peaks in the R-factor distribution printed in Scalepack where I see the ice rings to be. What's the best way to take care of ice rings?*

It is done automatically, however one can adjust REJECT fraction parameter to improve the accuracy of the rejections. Default value is 0.75, increasing it to 0.8 - 0.85 may help. Look at the display to see how many reflections turn red.

Q: *We got really bad icing on a couple of frames. How do we leave them out?*

You can omit whole frames from scaling, or use **RESOLUTION LIMITS** to avoid ice. In the extreme case you may reprocess the same image twice with resolution ranges below and above the ice rings.

Autoindexing

***Q**: How does the definition of* **BOX** *and* **SPOT** *size in Denzo affect autoindexing?*

The size of the integration spot affects autoindexing procedure in two ways:

1) the size affect the default value of **LONGEST VECTOR** if the **LONGEST VECTOR** command is absent from `auto.dat` file .
2) The spot size also affects the signal-to-noise ratio. Only spots above **WEAK LEVEL** (default value 5) are used in autoindexing. You can see these spots in *XdisplayF* window.

If the spot size is very wrong you may not get enough spots for autoindexing.

***Q**: After processing the first image and getting refined values for the various parameters, I then entered the refined parameters, in particular the values for* **CRYSTAL** *rotx, roty and rotz, into another auto.dat file and processed a different image. However, the values for* **CRYSTAL** *rotx, roty and rotz that come out of the indexing run are quite different though clearly related. Is this normal and why does it happen? I have to think a bit about whether it changes the indexing - in particular, whether the assignment of F_+ and F_- is switched. Certainly, if, after running the auto.dat in Denzo , I then redefine again the crystal angles to be consistent with those refined from the first image then the predictions look fine and the refinement proceeds nicely - though in one case CrysX is -366.26 instead of -6.26 but that should not matter. Any advice on this would be welcome.*

It is normal for autoindexing with different images to result in indexings related by a lattice symmetry operation. Such a rotation **never** changes the assignment of F(+) and F(-). No rotation can. Because you can get different solution related by lattice symmetry, you must only do autoindexing **once** per crystal.

C: I would like to remind you that in order to find indexing you need to collect one oscillation image (i.e. without Weissenberg motion) and only later move to Weissenberg mode.

***Q:** On a second data set from another protein, I found that XDISPLAYF and Denzo were unable to index from the first oscillation, which was very close to a zone, i.e. it could not get the third axis. Depending on the way I picked peaks, Denzo either failed entirely (with zeros in the matrix) or it found a cell axis of ~1.25 Å (as compared to ~164 Å). If I chose a frame midway through the data collection, then Denzo indexed quite quickly.*

The problem was really quite different. The oscillation range was too large for the combination of resolution and unit cell. Lunes were overlapping and autoindexing failed, by the way this applies to all autoindexing algorithms. When you rotated the crystal you were no longer collecting data down 164Å cell and lune overlap was less severe. The right solution is to collect small oscillation sectors down long unit cell axes. If you already have collected large oscillation sectors down the long unit cell restricting autoindexing to using only low resolution data may help. In this case you may try **RESOLUTION LIMITS** `40 6`.

***Q:** Is it possible to create a file of spots for Denzo that comes from more than one image? We have a rather weak diffractor that we would like to get some cell constants on and, in particular, determine whether it is monoclinic or orthorhombic. One image does not seem to be enough to help Denzo make a clean distinction.*

Sorry, it is impossible. It may be possible in the future, but it will require substantial change to the program code. You can make clean distinction of monoclinic from orthorhombic during scaling.

***Q**:* *I am trying to process data collected on an R-axis using Denzo and my problem is the following: Although I have what seems to be a pretty decent orientation using the R-axis software, when I import an oscillation frame into Denzo , the overlap between predicted and observed spots is far from decent. Some one suggested that I try to obtain a conversion between the angles* j_x, j_y, j_z *as defined in R-axis and those used by Denzo .*

There is conversion for the angles for some space groups, however it is much simpler to do autoindexing in *Denzo*. If you still have a problem it is most likely due to misindexing. *Denzo* reverses the definition of x and y axis relative to R-axis software. If you enter `X BEAM` and `Y BEAM` swapped, *Denzo* may misindex.

***Q**:* *Before we indexed this crystal on an orthorhombic lattice with cell constants of 133.7, 67.6, 35.7. Now the unit cell is primitive monoclinic 61.89 103.76 70.37 90 109.27 90, and distortion index for orthorhombic is 7.36%.*

This is why *Denzo* **always** outputs 14 best fitting Bravais lattices when autoindexing. Often protein crystal are polymorphic and unit cell is unexpected one.

Q**:* *It seems to me that if one has indexed the diffraction pattern from one image, for example Frame #1. Then after inputing all the parameters determined from this image (such as* ***CRYSTAL *rotx roty rotz etc.) one should be able to predict other frames, for example Frame #40 with rotx adjusted accordingly. However, the program seems always reindex the new image and somehow ignore the input parameters. I know that because the crystal symmetry would allow many different but equivalent indexings, however, I think consistency in indexing for successive image has its advantages in case that one wants to apply some anisotropic absorption correction. The confusion comes from the misunderstanding that the rotx value should be adjusted for every frame with different starting* j *-scan value.*

You should autoindex only one image per crystal, to have consistent description of crystal orientation, and you should omit the `peaks search file` *`peaks.file`* command on subsequent images.

Q: *After Denzo autoindex, should you have a program to output* ***CRYSTAL*** *rotx roty rotz if one wants to switch spindle axis and vertical axis. In the case of [unnamed co-conspirator], because odd ratio of cell axes, and same frames resulted in 3 or 4 equivalent autoindex results, but we could not find the location of major zone on the basis of any index results. I have a P21212 diffracted to 2.2Å but failed to find major zone, and collected data cross a mirror plane (60% completion after 90 degrees data).*

You can use option `PRINT` `zones` in *Denzo*. Or, you can redo the autoindexing with the desired values of `SPINDLE AXIS` and `VERTICAL AXIS`. Also *Scalepack* can do the requested calculation.

Q: *Also, is there some way to set a resolution cutoff for the peak search? At 2Å I have a dark nickel diffraction ring that peak search labels, understandably, with many peaks. This hasn't yet skewed the auto-indexing, as there I can set* ***RESOLUTION LIMITS****, but it would be nice to be able to avoid those areas in the data at the outset.*

Specify `RESOLUTION LIMITS` 20 2.1 before autoindexing. This will remove all spots found by peak search outside 20 - 2.1 Å range.

Q: *If I have good cell parameters to start with, can the indexing routine use them as given rather than recalculating them from the peak positions? For one image that I have the Denzo autoindexing gives somewhat different cell parameters from what I think are the correct ones, and the refinement then doesn't work as well as it should. I'd like to be able to try fixing the cell parameters and see if it helps.*

type...`peak search file` *`peaks.file`* `GO FIX` `cell` [It you did `FIT` `cell` in autoindexing] `UNIT CELL` 98 99 115 90 90 90 and the program will use the `UNIT CELL` value you supplied. This

will work only if the autoindexing unit cell is close enough (say within 1-2%) of the unit cell specified by your command.

Q: *Denzo bug: I am very dissatisfied with the inability of Denzo to correctly predict and integrate low resolution reflections. If one has a .3 mosaic crystal and you have to give it a 1.5 degree mosaicity in order to get a mere prediction of a low resolution reflection, this is ridiculous.*

Whoa, fella. The problem is with the wrong value of `X BEAM`, `Y BEAM` and resulting misindexing of the image. If all reflections have wrong index, at most random crystal orientations, low resolution reflections will not match predicted pattern. With right beam position mosaicity 0.3 did generate all low and high resolution reflections. Program is OK.

Q: *Actually, I did manage to index the 9-deg rotations.*

Autoindexing of 9-deg rotations is tricky and often fails. You were lucky.

Box definition

Q: *Is the reason that you recommend using at least twice as many pixels for background as for spot definition simply statistical?*

Yes.

Q: *Is it possible to create spots masks of differing ellipticity for the same image plate?*

No.

Q: *When I redefine the box parameters, the shape of the spot ellipse changes also. Why?*

If you change box parameters you may change parity (even of odd number) of pixels in the box. Spot ellipse has the same parity as box, so there has to be some adjustment to its shape.

Q: *Is the definition of the x,y directions in the averaged profile boxes printed by Denzo the same as in* **INST_HAMBURGD***? I have the impression that the axes are interchanged.*

Yes you are correct. The printout definition of axes are arbitrary and are different from display direction definition.

Q: *The question I have concerns the choosing of box size and background size. I have just completed processing data (taken from a CCD detector) where the box size was .75 .75, the spot elliptical was .3 .3, and the spot background was .35 .35. Data from 2 crystals was processed using these parameters and yielded R_{merge} values that were good. However, I have recently read in documentation of Denzo that the background area should be twice that of the spot area. Why does this ratio of background area to spot area have to be so high ? Is it possible to get by with a lower ratio of background to spot area? Should I go back and reprocess my data with a larger background area even though it has processed well? For imaging plates does the background area still have to be twice that of the spot area?*

Your background area was about 5 times the spot area. You had a large safety margin. You may get by with lower ratio at the cost of data quality.

C: *XdisplayF: You should click* ***Integr box*** *in zoom window. Now, if you will type (in Denzo) command:* ***SPOT*** *elliptical 0.4 0.4 20 or command:* ***BACKGROUND*** *elliptical 0.5 0.5 20 you will see integration area on the screen. The spot region should include whole Bragg peak - in the case of kalpha1 kalpha2 split, box should include both peaks. The spot shape is defined by statement such as* ***SPOT*** *elliptical 0.6 0.5 2.0* ***GO***

CCP4 Scaling

Q: *If one writes york format files with Denzo, what is the appropriate keyword to use with Scalepack? I thought Denzo_york worked, but it seems to fail. (The error is "ERROR:*

Unable to open 1 for reading", but if the format is changed to Denzo _ip, the file is opened correctly, but, of course, the data format is incorrect.)

Use *Denzo*_york1 instead. *Scalepack* will read also concatenated *Denzo*_york1 format (but not *Denzo* _ip). *Denzo* _york is another creation by Eleanor Dodson that hopefully died by now. The error is "ERROR: Unable to open 1 for reading" comes from the fact that *Denzo* _york and *Denzo* _york1 **do not require** staring batch number, as it is coded in the header of *Denzo* _york formats. The "1" is interpreted as file name with hkl's instead and *Scalepack* cannot open it if (unless you called output from *Denzo* "1").

C: *To go to CCP4 format you should use different option in Denzo:* **YORK OUTPUT FILE**. *That will produce somewhat different *.x file that can go into rotaprep (part of ccp4 package). Lack of satisfactory result from Scalepack maybe either from misunderstanding how the program works or from problems in the data collection. Scalepack algorithms are fairly similar to Rotavata/Agrovata and should not give qualitatively different results.*

Q: *I am still unable to go to SCALA since the ROT information does not carry through rotaprep. Any advice would be appreciated. So far though Scalepack does the same as rotavata and agrovata. Not surprising.*

Ask Eleanor Dodson, author of Rotaprep or Phil Evans, author of SCALA.

Computer problems and questions

Q: *Unfortunately I have problems with Denzo package. Here, we have the Open VMS AXP v 1.0. Running any *.exe I get this message:*

```
$ run ccd
%DCL-W-ACTIMAGE, error activating image LIBOTS
-CLI-E-IMGNAME, image file DKA300:[SYS0.SYSCOMMON.][SYSLIB]LIBOTS.EXE
-SYSTEM-F-SHRIDMISMAT, ident mismatch with shareable image
```

Thinking that this error was due to fact that I have an old version I got the LIBOTS.* from the v 1.5. I still have the same error.

You need Open(vms)AXP v 6.1 to run *Denzo*. Upgrade your operating system.

Q: *Is the new Scalepack slower than the old one?*

Scalepack speed is about the same. Postrefinement may take a lot of extra time, and very old versions had no postrefinement, however postrefinement is optional now so it can be as fast as the old one. Subjective variation in the speed of the program may be strongly affected by other programs on the same computer, in particular if the operating system starts to swap programs.

Q: *While I was trying to index the data from a twinned crystal, the Denzo always quit on me with a simple error message "Killed". Then, I realized that it occured to me once when I was processing the other data set from a "good" crystal. At that instance I logged off and rebooted the system. Afterwards, everything was OK. However, this time the system is corrupted seriously with error message of disk I/O error on rebooting. Besides, the window management files must have been corrupted because the PC icon-styled login window disappeared instead it gives me a dumb-terminal styled login window and every step runs very slowly. A check on /usr/adm/SYSLOG found the error message as,*

<DENZO > KILLED DUE TO INSUFFICIENT MEMORY/SWAP

By the way, my machine has 32 Mb main memory. It should be sufficient according to your manual. My best (uneducated) guess is that the program's memory management might have some serious flaws.

Message killed is from operating system running out of swap (not memory!) space. Swap space is shared by all programs, so it is dependent on how many processes are active, you can run out of swap space in unexpected moments. This has nothing to do which data set you process. Rebooting the computer killed some other programs and increased the amount of

swap space for *Denzo*, however rebooting computers can make life difficult if your system is not well managed. Find a better system manager.

Q: *We would like to use Denzo on our SUNs running Solaris 2.2 if that's not too much trouble for you. They are 50 MHz 2-processor machines with plenty of memory, so I'd be surprised if they were inadequately slow. For the time being the only other machine we could use for Denzo is our DECstation 5000, but that's slower.*

Both SUN's and DECstation 5000 are OK, but you run the risk of problems with the programs specific to these machines that program authors may not be aware of.

Q: *I told you that we have a SGI Power Series with VGX Graphics. This will handle your program, correct? Any special hints?*

No hints. You will not take advantage of VGX graphics over standard entry level graphics.

Q: *What have you done to Scalepack such that it now exceeds any page file quota which I have and which runs the older version quite happily?*

Increased allocation of memory for storage of reflections.

Q: *Also, are you still supporting your IPVIEW/Denzo package for SGI machines? We have noticed some binary executable incompatibilities between our platform and older ones which are meant to be compatible.*

SGI creates hopeless number of almost compatible releases of operating system. SGI are generally supported, however older versions (4.0.5 or so) may produce problems.

Q: *I'm not sure what the byte-swap of the image is.*

Denzo and *Scalepack* are now educated enough about little-endian and big-endian computers, so both programs can cope with byte-swapping problems automatically.

C: *On UNIX (OSF) computers you may install all programs in /usr/local/bin directory (remember that only root privileges allow you to do that). Check if in your path statement /usr/local/bin is included. Another strategy is to put all files in arbitrary directory, for example /usr/users/wladek/HKL and add to your .cshrc file a bunch of aliases like:*

```
alias denzo '/usr/users/wladek/HKL/denzo '
alias ccd   '/usr/users/wladek/HKL/INST_CCD' etc, etc.
```

On VMS system, you can put all files in any directory. You should set-up logicals by including into your login.com file a bunch of commands like:

```
$ KODAK :== $ZBYSZEK:[WLADEK.DISTR.VMS]CORNELL.EXE
$ FUJI :== $ZBYSZEK:[WLADEK.DISTR.VMS]FUJI.EXE
$ RX :== $ZBYSZEK:[WLADEK.DISTR.VMS]RX.EXE
$ RXS :== $ZBYSZEK:[WLADEK.DISTR.VMS]RXS.EXE
$ MAR18 :== $ZBYSZEK:[WLADEK.DISTR.VMS]HAMBURGN.EXE
$ MAR30 :== $ZBYSZEK:[WLADEK.DISTR.VMS]HAMBURGD.EXE
$ EMBL :== $ZBYSZEK:[WLADEK.DISTR.VMS]HAMBURG.EXE
$ DENZO :== $ZBYSZEK:[WLADEK.DISTR.VMS]DENZO.EXE
```

Q: *I am very happy with Scalepack and am trying to use it to look at data from a Xuong-Hamlin detector. The .ar files were on a MicroVAX where the data was collected, and I moved them over to an IRIS Indigo to run Scalepack. I have been using the following input file, but I get the following error:*

```
ENDFILE: END OF FILE -1
APPARENT STATE: UNIT 5 NAMED
LAST FORMAT: (132A1)
LATELY READING SEQUENTIAL FORMATTED INTERNAL IO
```

This occurs even if I specify a non-existent file after the keyword **FILE**. *Do I need to include another keyword, such as file size?*

Hamlin Archive files are binary and have to be processed with *Scalepack* on the same type of computer. If you did reduce Hamlin data on VAX you have to run *Scalepack* on VAX or Alpha-VMS. SGI version of *Scalepack* does not understand VAX Hamlin binary files. Sorry.

C: *Avoid byte swapping of raw data with command dd conv = swab. Better to do it in Denzo. Program dd may incorrectly change header information that may be useful.*

C: *On SGI one must load the files using /dev/tapens wnich tells it* ***not*** *to swap the bytes (/dev/tape, the default, does a byte swap to conform to ANSI standards…). To wit, if you try to load using /dev/tapens tar will complain about checksums and formats and then abort. Load using /dev/tapens then use dd conv=swab.*

Q: *We would like to process our data on a VAX/VMS (old-style) machine, although we can also use HP and SGI workstation. Therefore we would be mostly interested in VMS version of Denzo (for R-axis and also synchrotron data).*

Why use VAX if you have SGI and HP?

Q: *I ran the programs on a SGI running the 5.1.1.2 operating system, and I see they were compiled on 4.0.5. UN:csh: ERROR: cannot execute binary file UN:csh: ERROR: Scalepack - Exec format error*

Q: *The new version of Scalepack doesn't seem to run on our SGI machines. I get an error message ("Scalepack: Exec format error. Wrong Architecture."). I've tried IRIX versions 4.0.5 and 5.1.1.2. Any ideas?*

You probably try to run *Scalepack* compiled on OSF or other brand of computer.

Q: *Do you have a Alpha/VMS version of the latest stuff?*

Yes, however future versions (2.0 and higher) will not run on Alpha/VMS.

Q: *Error message from my computer:*

```
MARD TGABND_001.IMAGE

%LICENSE-F-NOAUTH, DEC DW-MOTIF USE IS NOT AUTHORIZED ON THIS NODE
-LICENSE-F-NOLICENSE, NO LICENSE IS ACTIVE FOR THIS SOFTWARE PRODUCT
-LICENSE-I-SYSMGR, PLEASE SEE YOUR SYSTEM MANAGER
%TRACE-F-TRACEBACK, SYMBOLIC STACK DUMP FOLLOWS
```

Install properly your window manager.

Consequences of Profile fitting errors in Denzo

Q: *I reduced some mar 30cm data with Denzo, and things seemed to go OK. I am now having trouble with Scalepack. It seems to reject many reflections:*

```
1 - count of observations deleted manually
2 - count of observations deleted due to zero sigma or profile test
3 - count of non-complete profiles (e.g. overloaded) observations
4 - count of observations deleted due to sigma cutoff
5 - count of observations deleted below low resolution limit,
6 - count of observations deleted above high resolution limit,
7 - count of partial observations
8 - count of fully recorded observations used in scaling
this line seems truncated:
                        \/
IP fitted, no o   2 1.0000  0.00  0 3520  0  0  0  0 886 1559
IP fitted, no o   3 0.9419 -0.27  0 3581  0  0  0  0 924 1503
IP fitted, no o   4 0.9741  0.30  0 3632  0  0  0  0 881 1551
IP fitted, no o   5 0.9717  0.67  0 3626  0  0  0  0 841 1500
IP fitted, no o   6 0.9535  0.79  0 3736  0  0  0  0 817 1450
IP fitted, no o   7 0.9241  0.78  0 3916  0  0  0  0 821 1364
...stuff deleted...
```

```
but yet gives a fairly good r-symms:
         Shell Lower Upper Average   Average     Norm.  Linear Square
         limit  Angstrom     I     error  stat.  Chi**2 R-fac  R-fac
               99.00  2.77 42346.7 1428.2  286.0 12.222 0.081  0.105
                2.77  2.20 15502.0  625.4  312.4 12.300 0.113  0.143
                2.20  1.92  9276.0  495.0  345.2 10.242 0.137  0.176
                1.92  1.75  5322.0  418.9  357.7  5.829 0.142  0.189
                1.75  1.62  4095.8  430.3  392.5  3.866 0.138  0.189
                1.62  1.53  3472.9  483.1  416.1  2.413 0.138  0.191
                1.53  1.45  3163.7  514.0  456.7  2.185 0.134  0.190
                1.45  1.39  2780.1  500.9  453.3  1.522 0.126  0.207
                1.39  1.33  2624.7  523.3  480.5  1.848 0.158  0.215
                1.33  1.29  6305.7  882.4  733.4  6.906 0.241  0.363
         All reflections   17538.7  756.0  346.3 10.403 0.096  0.113
But there are no reflections with I/σ(I) less than 5:
                Shell      I/Sigma in resolution shells:
         Lower Upper   No. of reflections with I / Sigma less than
         limit limit   0    1    2    3    5   10    20  >20  total
         99.00 2.77    0    0    0    0    5  143  1372  4655  6027
          2.77 2.20    0    0    0    0    9  466  2152  3283  5435
          2.20 1.92    0    0    0    0   36  804  2870  1756  4626
          1.92 1.75    0    0    0    0   93 1201  2770   494  3264
          1.75 1.62    0    0    0    0  106 1053  1817    80  1897
          1.62 1.53    0    0    0    0  160  859  1079     1  1080
          1.53 1.45    0    0    0    0  118  481   515     0   515
          1.45 1.39    0    0    0    0   73  204   211     0   211
          1.39 1.33    0    0    0    0   61  111   112     0   112
          1.33 1.29    0    0    0    0   89  602   665     0   665
          All hkl      0    0    0    0  750 5924 13563 10269 23832
```

*In the Scalepack output I get in the table for I/*s *in resolution shells 0 reflections in the first five columns (I/*s *0-5) i.e. I only get reflections with I/*s*>= 5. I nowhere input a 5*s *cutoff and looking through the manual the only explanation I found was the keyword* **WEAK LEVEL** *which has a default value of 5. On the other hand as I understand it, this only means that reflections <= 5*s *are not used in refinement and as reference profiles. Is* **WEAK LEVEL** *the explanation (and should I rerun Denzo with an explicit value for* **WEAK LEVEL***) or would you know any other reason for my missing data below 5*s*?*

Your problem comes from **PROFILE FITTING RADIUS** not being set. You have to rerun *Denzo* . **WEAK LEVEL** in *Denzo* has nothing to do with *Scalepack* keyword **SIGMA CUTOFF**

DENZO output

Q: *We would like to have some information about the output files *.x of Denzo .*

Regular (film output file, *Denzo* _ip) imaging plate output format (i.e. the format you would you to read in in FORTRAN) is:

format (3i4,i2,2f8.0,f7.0,f6.0,f6.0,2f7.0,f6.0,f8.0)
h,k,l
Flag 0 - full 1 - partial
Intensity (F**2) by profile fitting
σ of intensity (F**2)
χ^2 of profile fitting
Intensity (F**2) by profile summation
Cosine of incidence angle at detector
Predicted pixel position of spot centroid (slow, fast) directions
Lorentz, polarization, obliquity combined factor
Strength of averaged profile in arbitrary units

Alternative, (york output file, *Denzo* _york1) format is:

format (3i4,i2,2f8.0,f7.0,f6.0,f6.0,2f7.0,f6.0,f8.0)
h,k,l

Flag 0 - full 1 - partial
Intensity (F**2) by profile fitting
σ of intensity (F**2)
χ^2 of profile fitting
Intensity (F**2) by profile summation
Cosine of incidence angle at detector
Predicted pixel position of spot centroid (slow, fast) directions
Lorentz, polarization, obliguity combined factor
Strength of averaged profile in arbitrary units
Partiality (fraction of the intensity of the reflection extected in this image)
Diffraction angle (degrees)

The last may be very useful when preparing input to program SCALA

Q: *Is the Denzo output file format dependent on the type of detector?*
It is now the same.

Q: *total number of reflections output from Denzo = 174715*
unique reflections = 83873
reflections in reject.dat file = 1398
output from Scalepack = 73161
From this I think I've only rejected about 2% of data.
No, only 1398/174715, about 0.6%

Q: *Below is a bit of a *.x file from Denzo. Notice the middle reflection. The profile fitted intensity and the* `s` *are not fn.1 but seem to be integer data. Is this an intended part of the program to deal with overflows, or have I run into an operating system bug? Do you think this will only happen on reflections flagged with a negative* `s` *?*

```
    2 -16 -28 1  134.8  191.2  1.37  25.3 0.995 1371.5 1284.9 0.191 3252.4
    0 -12 -32 1 118141   5399 24.77-10031 0.997 1372.8 1196.3 0.191 3094.7
   -8  6  -50 1 9095.4 9536.0 13.85 393.1 0.996 1372.2  805.2 0.192 5856.9
```

Yes, this an intended part of the program to deal with overflows.

Q: *The problem is that the Denzo .x file does not have information about the* `j` *angle of the individual reflection, I know it has information about the starting* `j` *, ending* `j` *of the batch but NOT the* `j` *of each and every reflection which is the ROT column used in the CCP4 package. This is the problem, although rotavata and agrovata do not need this info, scala does.*
`york` output file has this information, see above.

Q: *Is it possible to suppress this function of putting information at the end of the .x file?*
No, it is not possible. Current rotaprepa can skip this information.

Q: *On integration the output often has many reflections with negative* `s`*s - what have I done wrong?*
Nothing or almost nothing, e.g. resolution limit too high for MAR scanner.

Q: *Could you tell me if the Denzo format has changed greatly? Rotaprep use to convert Denzo images to mtz images but it no longer works. I want to compare CCP4 processing to Scalepack processing. Just using the "lcf batch" or york output file does not do the trick.*
Use current version of rotaprep (from ccp4) with `york output file`

Q: *I am still unable to go to Scala since the ROT information does not carry through rotaprep. So far though Scalepack does the same as rotavata and agrovata. Not surprising.*
Check with Eleanor Dodson or Phil Evans.

Q: *Does Scalepack use the top 6 line of a *.x Denzo output file when it reads them in?*
Yes.

Q: *The last column in a *.x file, is it used by Scalepack at all?*
No.

Q: *It appears that the intensity has already had the Lorentz polarization factor applied to it. Is this true?*
Yes.

Display

Q: *Why doesn't the center of the integration box fall on top of the center of the predictions (yellow, cyan circles)? I found them to be offset towards the upper and lower edges of the Zoom box, but more or less centered in the middle.*
Boxes are centered on pixel boundaries, circles are centered on precise values.

Q: *the integration circle/ellipse and background "square" are in white; if one clicks the left (I think) mouse button. a second time, one gets red circles/ellipses. Are these red circle/ellipses the integration box or the background box?*
Both of them.

Example of Bad Detector Pathology

Q: *I didn't get satisfactory result from Scalepack.*
I would check following:

a) Beam line
b) Shutter
c) Detector
d) Crystal size, radiation sensitivity, mosaicity etc.
e) Consistency of indexing if data from more than one crystals are used in some space groups
f) Crystal isomorphism

C: *It is now clear that we do have shutter problems. We have only recently discovered that ours apparently either opens late, or closes intermittently.*

Q: *I recently scaled a data set that had good R_{merge} values but high normalized c^2 values in Scalepack. Is there something wrong or should I ignore the c^2 values? I know that I can boost the* `s` *values in Scalepack with the* **ERROR SCALE FACTOR**. *I was just wondering whether it could be due to any other parameter in Denzo. For example incorrect background definition or box size or anything like that. Our spots are very elongated but data is strong and c^2 values from Denzo appear to be good. I know that other data sets give values in the range 1-2 but this one is in the range 3-4. Does it get harder when the* **SPOT** *elliptical is more asymmetric?*
Reprocessing with *Denzo* rarely improves χ^2 dramatically , unless you correct a major mistake in the processing.

Q: *I am trying to scale some data which processed very nicely from 30cm mar at 9.6. I am trying to see why it doesn't scale well. If I change the value of* **ERROR SCALE FACTOR** *should I try to get the final c^2 column at the bottom of the output of Scalepack to be 1.## or should they be equal to the value of the* **ERROR SCALE FACTOR** *I give?*

At the moment it looks like:

```
          Shell  Lower  Upper  Average  Average       Norm.    Linear
Square
          limit  Angstrom        I  error     stat. Chi**2 R-fac R-
fac
             100.00    4.31  205113.5  17423.5  13959.9  4.838  0.087
0.155
              4.31    3.42  140289.3   9859.9     7050.6  5.088  0.077
0.127
              3.42    2.99   59864.0   4943.1     3835.6  5.593  0.092
0.174
              2.99    2.71   31249.4   3211.6     2690.9  4.465  0.101
0.180
              2.71    2.52   21540.4   2658.0     2326.9  3.501  0.113
0.172
              2.52    2.37   16115.9   2358.6     2132.3  2.615  0.120
0.156
              2.37    2.25   13138.6   2214.7     2044.7  2.435  0.132
0.182
              2.25    2.15   10403.3   2107.3     1985.4  2.597  0.163
0.228
              2.15    2.07    7984.0   2084.2     2001.9  1.883  0.191
0.241
              2.07    2.00    6503.1   2127.2     2059.1  1.790  0.226
0.290
        All reflections     52388.8   4959.8     4047.5  3.657  0.092
0.149
```

with **ERROR SCALE FACTOR** *3.0* and **ESTIMATED ERROR** *0.06*.

Things are pretty bad.

I'd try with *Scalepack* with e.g. **ESTIMATED ERROR** 0.10 (usually I use 0.06) and **REJECTION PROBABILITY** 0.01. A value for **ERROR SCALE FACTOR**? Usually about 1.5 seems quite good, but I guess in this case I should experiment a bit - but if I need to have **ERROR SCALE FACTOR** greater than say 2.0 then I know that it still isn't right?

I tried scaling again with **ERROR SCALE FACTOR** *1.5* **ESTIMATED ERROR** *0.10* **REJECTION PROBABILITY** *0.1*. This time the scaling statistics look like...

Shell limit	*Lower Angstrom*	*Upper*	*Average I*	*Average error*	*Norm. stat.*	*Chi**2*	*Linear R-fac*	*Square R-fac*
100.00	*4.31*		*207375.1*	*17638.1*	*6884.6*	*4.242*	*0.088*	*0.136*
4.31	*3.42*		*141622.7*	*11636.3*	*3527.6*	*4.565*	*0.080*	*0.115*
3.42	*2.99*		*60834.7*	*5307.6*	*1921.0*	*4.949*	*0.091*	*0.145*
2.99	*2.71*		*31748.7*	*3013.8*	*1348.5*	*4.555*	*0.099*	*0.146*
2.71	*2.52*		*21800.0*	*2247.5*	*1161.2*	*4.118*	*0.109*	*0.146*
2.52	*2.37*		*16230.9*	*1822.3*	*1063.3*	*3.710*	*0.117*	*0.141*
2.37	*2.25*		*13217.2*	*1602.3*	*1020.6*	*3.642*	*0.128*	*0.160*
2.25	*2.15*		*10430.2*	*1422.9*	*988.6*	*4.094*	*0.157*	*0.206*
2.15	*2.07*		*8010.3*	*1310.4*	*1000.5*	*3.451*	*0.184*	*0.220*
2.07	*2.00*		*6584.4*	*1296.2*	*1033.9*	*3.515*	*0.220*	*0.268*
All reflections			*52826.6*	*4803.3*	*2009.7*	*4.142*	*0.093*	*0.131*

numbers Any suggestions as to how to make this a bit better? At least now all the χ^2 are similar - I know that I can compensate for these by increasing **ERROR SCALE FACTOR**, but is there a better way? What exactly is the scanner doing to give such funny data? Elspeth thought that there was a problem with the nonlinearity of the detector above a certain value, which is why I tried using **OVERLOAD VALUE** in Denzo to try and get rid of rubbish data!

I've fiddled around with all my data, and it doesn't scale well no matter what I do. It's a shame, because my data is 96% complete to 2.1Å. I think though because it doesn't scale I will not use the data, and instead collect it all again. It seems like a waste, but I think it is safer to start off with good data with good scaling etc. before making maps and drawing conclusions from their interpretation. Do you think I should give up with the data?

No other choice. It seems you got screwed up.

Ignore

Q: *I would like to be able to blank out the region of the imaging plate blocked by the beam stop. I have tried* **IGNORE** *circle centre_x centre_y radius and that works to get no predictions in a circular area. Is there an equivalent ignore rectangle command?*

Use: **IGNORE** `circle 96. 98. 8` (ignore circle of radius 8mm centered at 96. 98).

```
IGNORE quadrilateral    199.4 81.2
                        108.8 98.4
                        109.5 92.9
                        199.4 74.8
```

These are the four corners of quadrilateral to be ignored. Note that order of the 4 corners matter, they have to be input in counterclockwise order.

C: *We've discovered the utility of the* **IGNORE** *commands in Denzo to really define the beam stop shadow. The data is weak enough that reflections behind the beam stop don't always get rejected in Scalepack. The ignore circle command works fine. Very handy.*

Q: *The* **IGNORE** *bar also works but gives either a vertical bar (R-axis) or something that looks like a very thin line at an angle. The thin line is only detectable because the pattern of overlapped reflections is altered. Can this bar be made horizontal?*

Do not use **IGNORE** `bar`. Use **IGNORE** `quadrilateral` instead.

Mosaicity

Q: *We're sure the mosaic spread is increasing. How do we refine a slowly varying mosaic spread in Scalepack?*

FIT `batch mosaicity 1 to 60` mosaicity will be refined separately in each batch of data. or **FIT** `crystal mosaicity 1 to 10 11 to 20 21 to 30 31 to 40 41 to 50 51 to 60` mosaicity will be refined separately for 6 groups of batches, with separate values for first, second, third, etc.. groups of 10 batches. These groupings may be different for mosaicity and for other unit cell parameters.

Q: *How do we handle a slowly varying mosaic spread in Denzo?*

If you have 60 frames and you want to process first 10 with mosaicity 0.2 degree next 10 with mosaicity of 0.25 degree, and so on, up to the last 10 with mosaicity of 0.45, you may do that in the following way: In the command file type:

```
MOSAICITY 0.2 ... SECTOR 1 to 10 ... GO      [to execute autoindexing]
@refine.dat   [refine first 10 images] MOSAICITY 0.25 SECTOR 11 to 20
@refine.dat   [refine next 20 images] MOSAICITY 0.3  SECTOR 21 to 30
@refine.dat   [refine next 20 images] MOSAICITY 0.35 SECTOR 31 to 40
@refine.dat   [refine next 20 images] MOSAICITY 0.4  SECTOR 41 to 50
@refine.dat   [refine next 20 images] MOSAICITY 0.45 SECTOR 51 to 60
@refine.dat   [refine next 20 images] END OF JOB
```

Q: *Looking at the raw data, I found that Denzo had some reflections spread over 3 or 4 frames. Given the frame size and mosaic spread, I'm puzzled by this. They were near the spindle, however, so is this a Lorentz effect, i.e., what is your criterion for excluding reflections with large Lorentz correction?*

All reflections that completely cross Ewald sphere are included.

Q: *Do you only do the postrefine business on the final run after rejecting the rejects or do you do it right away? Is there an isotropic mosaicity I can refine, should I be refining in all directions, or should I really be refining just mosxx as was in the sample file somebody handed me? Does it matter as long as I'm not going to re-Denzo the frames? (I set it to .4 in Denzo but it seems to refine to around .25 or so so it seems pretty safe to me).*

You can only refine isotropic mosaicity. It is better to have larger mosaicity in *Denzo* and refined by *Scalepack*, as *Scalepack* can reliable observed (partial) reflections as non-diffracting, but not the opposite. If mosaicity is decreased some partial reflections will be relabeled as fully recorded.

Q: *In Scalepack I can postrefine all 6 mosaicity parameters..., but in Denzo I can only use one mosaicity parameter. So how can I tell Denzo that the crystal was more mosaic in the y direction than the x? Do I change the value of* **CROSSFIRE***?*

CROSSFIRE has to do with focusing of X-rays, not reflection width in spindle angle. Both *Scalepack* and *Denzo* use isotropic mosaicity only. Anisotropic option is not yet implemented.

Negative Intensivities

Q: *What is the output from Scalepack? Does it contain I & I_{sig} or F & F_{sig}? Why are there some negative values for I (or F) in the output file?*

Output from *Scalepack* is I, sig(I), where I refers to Lorentz, polarization etc. corrected diffraction intensity. Sometimes this I is referred to as F**2. Even the best (averaged) estimate of intensity may be negative - see discussion of **SIGMA CUTOFF** keyword in *Scalepack keyword Description*.

Other

Q: *I cannot to input several .ar files from different data collections into Scalepack. Is this possible?*

One should be able to put many different `.ar` files. I tried up to 25. Be sure that they have different batch numbers! If not use the option

```
ADD 1000
FILE XXXX1.AR
ADD 2000
FILE XXXX2.AR
```

to give different batch numbers to different `.ar` files.

Q: *I am using an Scalepack on a VMS machine with multiwire .ar files. How or can I refine the mosaicity and the B factors using Scalepack?*

You cannot refine mosaicity in `*.ar` files. It has not been implemented. However you can refine B factors, it is done by default, so you do not need to do anything special.

Q: *We run our data collection on a VMS machine, and when the data is transferred to a UNIX machine the file is corrupted (using ftp)*

Some versions of ftp will create wrong file format on VMS. Change to UCX (native VMS implementation) in such a case. Also remember to use binary ftp not ASCII ftp (sometimes a default option in ftp!)

Q: *I sometime get a message like insufficient space in common/spots losing reflection. What does that mean?*

Denzo is running out of memory and ignores some reflections. This may happen if the **BOX** is unreasonably large. Reduce the **BOX** size or use the *DenzoBig* version of the program.

Q: *Would you make a larger version of Scalepack as I have some 2.8A data on an I222 form of a virus crystal (a=328.2, b=341.8, c=363.75) and Scalepack can't quite cope with the number of reflections?*

Use **ScalepackBig** or **ScalepackVeryBig**

Q: *We collect our data on an Raxis (VAX workstation) and transfer it using FTP to the SGI. We have no trouble getting the Denzo to run on the oscillations but the display program (the one we got is INST_CORNELL) gives a message that the data file may be corrupted. The sample data that was packed in the tar file runs just fine. Is this the right version of the display program?*

Use INST_RX instead. INST_CORNELL is design to read in data scanned with KODAK scanner at Cornell (now decommissioned).

Q: *I believe that my copy of the Scalepack executable file is somehow corrupted, because every time I run it, it always return with strange error message:*

```
CRYSTAL6% SCALEPACK SCALEPACK: SYNTAX ERROR AT LINE 1: ^A_^H,>Q^CE^FMH^P^B^A^K^A ^
```

Yes, you are correct. This is not the right executable.

Q: *Scalepack runs, it reads the frames but then just sits there. I used the example as a guideline and I also tried with deleting the postrefinement stuff but it still just sits there.*

Try Crtl D (UNIX) or Crtl Z (VMS), to terminate input into the program.

Q: *I tried to run Scalepack. I get an error message:* Exec format error. Wrong Architecture. *Any ideas?*

You try to run program compiled for different operating system or OS version. Investigate.

Q: *I got back from SSRL collecting data on the MAR. I want to process data with Denzo as well on our Indigo and I was wondering if I need to swap bytes for INST?*

INST and *Denzo* programs recognize swap bytes automatically.

Q: *How do I get the radial and angular offsets for SSRL scanner? Do you have those numbers?*

Sorry, we do not know. Ask SSRL staff.

Q: *** TERMINATING /usr/perutz1/wladek/DENZO /DENZO *** Received signal 10 (SIGBUS) Bus error

An error from old SUN version. Problem specific to SUN. If you get such an error contact authors of the program.

Q: *I have an older version of Scalepack, which I assume is fine to use. Or is it?*

It should work but it is not as good as the new one.

Q: *Can you produce a version of Scalepack that will take up to 1,000,000 reflections*

Yes, now it is a standard version.

Q: *Should we have* **Y SCALE** *-1.0 for this MAR plate?*

Yes, however it is a default value.

Q: *We are looking at some Daresbury MAR data. The rotation axis is horizontal and I thought that we always processed this with* **FILM** *rotation -90.0 (or 90.0) but now we are all using 0.0. Is this because the convention has changed ? I am worried that the polarization corrections may be all wrong....This is 0.9A data so the polarization correction will be quite important ...*

I hope that you did not process Daresbury MAR with film rotation -90.0 (or 90.0). The data would be disastrously bad. Maybe you confused MAR with R-axis at Daresbury? R-axis with horizontal spindle (3 such machines exist) requires film rotation -90.0 or 90.0 Polarization number should be positive for horizontal spindle at Daresbury regardless of value of film rotation. If you specified **POLARISATION** `daresbury` you are safe both with MAR and R-axis.

Q: *When I tried to process one of virus images there was a message about sfprof too small*

The program is running out of memory and ignores some reflections. This may happen if the profile fitting radius is unreasonably large. Reduce the **PROFILE FITTING RADIUS** size or use the *DenzoBig* version of the program.

Overload

Q: *If we have not overload cutoff value in the command file what does it decide as the overload value when printing the statistics after each image?*

Default value is OK for R-axis and some MAR scanners. It represents the full dynamic range of possible numbers in the file. Now *Denzo* reads overload table from MAR scanners. On some old MAR scanners spiral to Cartesian conversion program incorrectly handled saturation. Newer version seem to be OK. Some sites may have locally modified software

(EMBL-Hamburg, MRC Cambridge), check with whoever made the modification. With MAR scanners user has to know instrument he or she is using.

Q**:* *When looking at the Denzo output I noticed that some reflections are flagged with backgr ovfl. This seems to be independent from the parameters of* ***BACKGROUND *elliptical or* ***SPOT*** *elliptical or* ***OVERLOAD VALUE****. Are there other parameters controlling this backgr ovfl ?*

The `backgr ovfl` flag may be set a an overloaded pixel, but also by a high percentage of pixels rejected from background calculation. On MAR scanners spots outside active area (corners of the image) will never have valid background and will have `backgr ovfl` condition set.

`backgr ovfl` column is really sum of all reflections with any kind of problem with background, one of the possibilities being having one or more overflowed pixels in the background areas. Other problems are background varying too much, or too many pixels being obscured by other reflections, or too many pixels not having a valid measurements. The last affects mostly spots outside active area of the detector. Adjusting **RESOLUTION LIMITS** can prevent prediction of spots there. **OVERLOAD VALUE** has only effect on one of the factors involved.

Q**:* *When you say outlier rejection - what exactly do you mean? I tried changing* ***OVERLOAD VALUE *to 40000, but it doesn't make much difference. Should* ***ERROR*** *systematic be increased then? I integrated with it at 5.0 - should I try with 10.0 say?*

Neither **OVERLOAD VALUE** nor **ERROR** `systematic` have any direct impact on outlier rejection. **OVERLOAD VALUE** defines detector saturation. Systematic error in *Denzo* is a number that describes expected correlation between positional errors on the detector. Outlier rejection happens when symmetry related reflections do not agree in measured intensity.

***Q**:* *What exactly is the scanner doing to give such funny data?*

Another problem with MAR scanners is sometimes a wrong scale factor applied to strong fluorescence signal compared to weak signal. Setting **OVERLOAD VALUE** to below transition point between strong and weak range of data may help, but this would eliminate a lot of reflections. Transition point is roughly at about 10000 MAR units.

Q**:* ***SHOW OVERLOADS *does not work with most file formats.*

It works now with all formats.

Refinement

Q**:* *I have set up a batch job for a set of images taken from one crystal at CHESS. I indexed the first one with autoindex and wrote a shell script. I understand that with the Fuji scanner you can give* ***SECTOR *1 to n instead of giving the line* ***@****refine n times. But then some images did not refine because the beam position shifts a lot. I picked the beam position in the display and wrote the following:* ***SECTOR*** *1...* ***@****refine* ***X BEAM*** *...* ***Y BEAM****...* ***@****refine....So I still have one* ***@****refine per image but can alter the beam position before the next image. Is this the best way to do it (it works) or is there a better way?*

This will work.

Q: *Autoindexing gives almost correct cell constants, but the predictions are somewhat off, so the refinement does not kick in. I have 60 frames, but the data is weak and only to 4Å.*

There are some improvements to refinement algorithm that make it more stable, however the problem is likely be somewhere else. One likely explanation that would agree perfectly with your description is misindexing of lattice by one index. The solution in such a case is to input correct (or close to correct) values of **X BEAM** and **Y BEAM.**

***Q**:* *Is it the* **SECTOR** *n to m command that sets up the do-loop for data integration? Therefore, if one puts this command at the beginning of the auto.dat section, one will get independent indexing of each image; if it goes at the beginning of refine.dat then one indexing is used and each image is simply made to fit as best it can. Right?*

No. It does not matter where you put the `SECTOR n to m` command as long as it is before the first `GO` statement. The do-loop starts at `START REFINEMENT` and ends at `CALCULATE GO`

***Q**:* *When I ran a batch job to process 46 frames, I noticed that the* **CRYSTAL** *rotx was not updated with the oscillation step during the time of each job. For example, it I start with* **CRYSTAL** *rotx of 32.167, after 5 frames are processed, the value of the* **CRYSTAL** *rotx is about the same, 31.469. However, the* **OSCILLATION** *start and end are updated each time. How can we understand this?*

That you collected data starting each image at a different `OSCILLATION start`, the way everybody collects his/her data.

***Q**:* *Last time you said that* **CRYSTAL** *rotx is defined relative to spindle angle for which oscillation angle is 0 and it should not change when advancing to the next frame (within a batch job). However, when I processed the data frame by frame manually, the* **CRYSTAL** *rotx values changed each time at the end of the refinement no matter what value was input before the process. Should the* **CRYSTAL** *rotx values be different for the different frames (only different by an oscillation step, saying 0.5) or should they be the same? What I have seen is that the* **CRYSTAL** *rotx values are the same for each frame when the job is submitted as a batch one, while they are quite different (not within an oscillation step) when the data is processed manually frame by frame. If the* **CRYSTAL** *rotx values are different for every frame, how can one include partial reflections during the process of scaling?*

What counts is only sum of the `OSCILLATION start + CRYSTAL rotx`. It seems that you are inconsistent in how you describe crystal orientation using different procedures.

Q: *In the sample input of Scalepack (scale.in) you sent me a while ago, there are statements for postrefinement. After the postrefinement do you suggest a reintegration?*

It depends. If mosaicity from postrefinement is higher than that set in *Denzo*, then definitely yes. If you want to precisely characterize the detector, you may wish to rerefine and reintegrate the data with unit cell parameters fixed to postrefinement results. Detector parameters refined in *Denzo* will then be much more precise. Otherwise the gain from reprocessing is very small.

***Q**:* *What does* **START REFINEMENT** *do?*

Start a loop that finishes at `CALCULATE GO`. Loop range is defined by `SECTOR 1 to 50.` This will execute the loop 50 times.

C: *Zero* `S` *often comes from too-small (or absent)* **PROFILE FITTING RADIUS**. *Increase the value and judge it on the display (click top left button) .. For small molecule crystal spot density is smaller and larger* **PROFILE FITTING RADIUS** *may be optimal.*

Rhombohedral

***Q**:* *R32 causes problems.*

You can reindex to hexagonal setting in *Scalepack*. *See* Scenario 5 of the *Scalepack* manual.

You can use both primitive and hexagonal settings in *Scalepack*, you can also reindex primitive setting (from *Denzo*) to hexagonal in *Scalepack*.

C: *I have now processed my data as C2 monoclinic and the processing looks good .... but after discussing it I realized that C monoclinic lattice is a subset of rhombohedral lattice.*

Twinned Crystals

***Q**:* *The problem is that the data is from a twinned crystal. Both orientation angles have been figured out manually for two lattices. Due to the overlapping, most of frames have relative high average c^2 value (up to 6.0 sometimes). The data set reduced turn out to be good enough to yield right molecular replacement answer without any further modification of the data (reduced by Scalepack). But for high resolution structure, the data is bad as regards to R value (28% after simulated annealing). So I do want to get a better data set. If one could modify the source code a little bit, it will do the work. The idea is:*

1) *Do a prediction of spots from one lattice*
2) *Pull out a box around each predicted spot, set the background to 0 (or averaged background value of the frame)*
3) *Do everything else the same as original Denzo do for the reflections from the other lattice.*

I tried to do it myself. But it is really difficult for me to decode the Denzo. Would you like to tell me if that is possible for you to do it for me or is there any help that I can get from you?

The algorithm needs to be somewhat more complicated. One has to include the possibilities of overlap between lattices. Maybe in the future.

***Q**:* *All I need is a way of being able to process data from a twinned crystal and I'd be so happy.*

Put detector as far back as possible to separate the two lattices. Process one at the time. Hope for best. Do not expect outstanding results.

INDEX

;

; (semicolon) *Denzo* keyword, 73

@

@ (input) *Denzo* keyword, 61
@ (input) *Scalepack* keyword, 131

2

2θ angle (beam to detector axis), 25, 30, 55, 56, 144
2-D profiles of the spots - displaying, 20

A

α*, β* and γ*, reciprocal unit cell angles, 128
a*, **b***, **c***, reciprocal unit cell vectors, 128
A/D test *XdisplayF* button, 19
absorption, 8, 33, 114, 146
  AIR ABSORPTION LENGTH *Denzo* keyword, 53
  correction factors, 144
  of the film, 58
ADD PARTIALS *Scalepack* keyword, 123
ADD *Scalepack* keyword, 123
advantages of using *Denzo*, 6
AIR ABSORPTION LENGTH *Denzo* keyword, 53
Alpha (DEC workstation), 149, 150
ANGULAR OFFSET *Denzo* keyword, 35, 53, 68
angular offset in *Denzo* output file, 47
anomalous signal, 8, 89, 92
  ANOMALOUS *Scalepack* keyword (flag), 113, 123
  change of anomalous signal sign, 102
  NO ANOMALOUS *Scalepack* keyword, 132
  NO MERGE original index *Scalepack* keyword, 132
  OUTPUT ANOMALOUS *Scalepack* keyword, 134
  Patterson difference, 114
  SCALE ANOMALOUS *Scalepack* flag, 92, 114, 139
  Scalepack scenario, 113
archive file format, 126, 129, 141, 149
  BACKGROUND CORRECTION *Scalepack* keyword, 124
  FIXED WINDOW *Scalepack* keyword, 129
  NO PROFILE TEST *Scalepack* keyword, 133
  ORIGINAL WINDOW *Scalepack* keyword, 134
  PROFILE TEST *Scalepack* keyword, 135
  UNIT CELL is included in the header, 141
auto editing Q&A, 144
auto.dat command file, 13, 31, 160
  example, 28
autoindexing, 5, 20, 25, 26, 27, 102, 144, 156
  autoindexing Q&A, 145
  BOX and **SPOT** *Denzo* commands, 31
  Denzo outputs 14 best fitting lattices, 146
  Denzo parameters, 31
  don't keep re-running autoindexing!, 83
  failure, 30
  FIT command, 34
  general description, 26
  ice rings, 41
  LONGEST VECTOR *Denzo* keyword, 62
  misindexing, 32, 159
  oscillation range, 31
  PEAK SEARCH FILE peaks.file *Denzo* command, 66
  peaks used in, 27
  rhombohedral space groups autoindexing, 71
  twinned crystal, 33

B

B factor, 88, 89, 91, 93, 95, 96, 97, 117, 123, 132, 133, 135, 136, 139, 157
  B RESTRAIN *Scalepack* keyword, 124
  DEFAULT B FACTOR *Scalepack* keyword, 124
  FIT B *Scalepack* keyword, 128
  FIX B *Scalepack* keyword, 128
  INITIAL B FACTOR *Scalepack* keyword, 131
background, 6, 9, 21, 22, 33, 37, 39, 41, 42, 65, 74, 161
  BACKGROUND CORRECTION *Scalepack* keyword, 91, 124
  BACKGROUND *Denzo* keyword, 27, 38, 44, 53
  background overflow, 46, 159
  background size, 147
  background subtraction and fluctuation, 140
  BOX *Denzo* kayword, 54
  high intrinsic background, 40
  non-random signal in background, 137
  REJECT fraction *Denzo* command, 70
BAD VALUE *Denzo* keyword, 54
batch (data set) in *Scalepack*, 126
batch processing, 15, 16, 43, 83, 159
  batchDenzo.com command file, 43
  film pack parameters, 77
  keywords required for…, 78
  Scalepack, 89, 94
  SECTOR *Denzo* keyword, 71
batchDenzo.com command file, 44
Bayesian reasoning, 10, 90
  rejection of outliers, 137
beam, 9
  angular spread, 9, 31, 38, 55
  beam search routine, 83
  beam stop shadow, 60
  CROSSFIRE *Denzo* keyword, 55
  direct beam, 25, 30, 32, 55, 83
  efficient use, 7
  findbeam.dat search routine, 44

focus, 9, 55
intensity fluctuations, 9, 33
LAMBDA *Denzo* keyword, 75
NO BEAM *Denzo* keyword, 64, 74
synchrotrons, 63
thinned beam stop, 30, 83
USE BEAM *Denzo* keyword, 44, 64, 74
USE NO BEAM *Denzo* keyword, 44
WAVELENGTH *Denzo* keyword, 27, 75
wavelength modifier to **FIT** *Scalepack* keyword, 127
x beam & y beam parameters, 80
X BEAM and **Y BEAM** positions, 30
X BEAM *Denzo* keyword, 27, 30, 55, 76, 83
x beam *Denzo* log statement, 47
Y BEAM *Denzo* keyword, 27, 30, 55, 76, 83
y beam *Denzo* log statement, 47
Bijvoet differences, 50, 76, 139
box (integration box), 21, 47, 51, 53, 65, 147, 153
BOX *Denzo* keyword, 14, 27, 31, 44, 54, 60, 145, 157
Bravais lattice, 26, 29, 71, 118, 119, 146
centered monoclinic , 120
centered orthorhombic , 120
centered tetragonal , 120
cubic, 29, 76, 115, 119
hexagonal, 29, 76, 119
monoclinic, 29
orthorhombic, 29
primitive monoclinic , 120
primitive orthorhombic , 120
primitive tetragonal , 120
primitive triclinic, 120
rhombohedral , 29, 119
tetragonal, 29, 76, 115
triclinic, 29
trigonal, 76, 115

C

χ^2, weighted error measure, 10, 15, 32, 42, 57, 66, 95, 114, 115, 118, 139, 151, 153, 155, 161
chi square - modifier to keyword **PRINT**, 66
CALCULATE *Denzo* keyword, 43, 54, 56, 68, 72, 160
calculate *Denzo* log statement, 45
CASSETTE *Denzo* keyword, 27, 30, 35, 44, 54
CCD detectors, 6, 9, 71, 76, 116, 147
CCP4 file format, 148
CELL *Denzo* keyword, 55
cell reduction, 26
centered space group, 80
chymotrypsin, 8
circle, 61
Close *XdisplayF* button, 20
Color *XdisplayF* button, 19
command file, 13, 48
auto.dat, 28
batchDenzo.com, 44
experiment.dat
example, 28
findbeam.dat, 44
refine.dat, 43
scaletwosca.com, 101
site.dat
example, 27
Computer problems and questions, 148
computer recommendations, 79
Control input in *Scalepack*, 89
conventions
detector orientation, 49
Gravity/beam orientation system, 50
input to *Denzo*, 48
International Tables, 118
rotation in *Denzo*, 81
Spindle/beam orientation system, 50
COUPLING *Denzo* keyword, 55
CROSSFIRE, 35, 55, 156
CROSSFIRE *Denzo* keyword, 55
CRYSTAL *Denzo* keyword, 56, 81
CRYSTAL rotz, roty and rotx *Denzo* keyword
rotation conventions, 82
crystal symmetry
Bravais lattice, 26, 29, 71, 118, 119, 146
pseudo symmetry, 26
crystallography as an iterative process, 10
crystals
cracked crystal, 31
frozen, 41
intrinsic parameters, 6
orientation, 25, 28
parameters, 34
satellite crystals, 9, 27, 31, 137
size, 38
slippage, 8
twinned crystals, 31, 33, 148, 161
cutoff, 70

D

DEC Alpha, 149, 150
DEFAULT B FACTOR *Scalepack* keyword, 124
DEFAULT SCALE *Scalepack* keyword, 124, 131
density of the spots, 68
*Denzo*_ip format, 125, 129, 131, 136, 141
*Denzo*_york1 format, 123, 126, 129, 131, 136, 141, 148
detectors, 8, 9
alignment, 9, 57
calibration, 8
CCD, 6, 9, 71, 76, 116, 147
data file template
RAW DATA FILE *Denzo* keyword, 69
DIP (MacScience), 49, 51, 54, 57, 66, 83
distortion, 9, 22
dynamic range, 19
Fuji, 49
Hamlin (SDMS), 124, 149
archive file format, 124, 126, 129, 149
FIXED WINDOW *Scalepack* keyword, 129
Image Plate, 5, 9
Kodak, 49
manual IP, 9, 30, 44, 83
Mar, 9, 39, 49, 51, 66, 83
multi-wire proportional counters, 9
non-planarity of film or IP, 9
orientation, 25
orientation conventions, 49
parameters, 6, 27, 30, 36, 80
FORMAT *Denzo* keyword, 60
precession method, 6
problems, 8, 9, 19, 96, 114, 137, 153
quality, 90
R-axis, 9, 13, 22, 39, 44, 49, 51, 55, 57, 66, 83, 91, 129, 144, 146, 150, 156, 158
saturation, 9, 41, 159
OVERLOAD VALUE *Denzo* keyword, 65

shutter, 8, 96, 153
single pixel spikes, 9
spindle, 96
spindle motor backslash, 8
spiral, 6, 9, 71
unusual instruments, 44
vibration, 17
Weissenberg method, 6, 55, 58
Diff Vec, 21
diffuse or satellite scattering, 37
diffuse scattering, 70
Dim *XdisplayF* button, 20
DIP (MacScience) ⇒ detectors, 57
Display, 153
DISTANCE *Denzo* keyword, 30, 35, 55, 56
distortion, 26, 28, 29, 30, 32, 146
DO NOT EXTEND PARTIALS *Scalepack* keyword, 126
DO NOT REJECT OUTLIERS *Scalepack* keyword, 125

E

eigenvalue filter, 36, 89, 91, 95, 97, 136
EIGENVALUE FILTER *Denzo* keyword, 56
END FIT *Denzo* keyword, 59
END FIT *Scalepack* keyword, 125
END FIX *Denzo* keyword, 59
END OF JOB *Denzo* keyword, 44, 56, 156
END OF PACK *Denzo* keyword, 56
error
χ^2, weighted measure, 57
ERROR *Denzo* keyword, 56
error model, 10, 36, 43, 57, 90, 96, 114
error scale factor, 88, 96, 125
ERROR SCALE FACTOR *Scalepack* keyword, 95, 97, 125
estimated error, 37, 67, 88, 95
ESTIMATED ERROR *Scalepack* keyword, 95, 97, 98, 100, 125, 133, 154
expected error, 42, 57, 90, 138
in the fitting, 42
overload, 57
partiality, 57
positional, 57
profile fitting, 150
systematic error, 57, 95, 125
y/x factor, 57
ERROR SCALE FACTOR *Scalepack keyword*, 153
estimate of I , 140
Ewald sphere, 35, 156
experiment.dat command file
example, 28
experimental parameters, 13
experimental procedure, 7
EXTEND PARTIALS *Scalepack* keyword, 126

F

F**2, 151
F_+ and F_-, 145
factors which lower R_{merge}, 91
Fast Fourier Transform, 26
Fcalc, 140
FIDUCIALS *Denzo* keyword, 58
FILE *Scalepack* keyword, 126
film
optical density, 57
FILM *Denzo* keyword, 58, 60, 76
FILM ROTATION *Denzo* keyword, 27
filter, 63
final statistics, 7
findbeam.dat command file, 44
fit algorithm, 35
FIT B *Scalepack* keyword, 128
fit cell, 71
fit command
first round, 34
FIT *Denzo* keyword, 59
FIT cell modifier description, 59
FIT *Scalepack* keyword, 126
fitted, 66
FIX B *Scalepack* keyword, 128
FIX *Denzo* keyword, 59
FIXED WINDOW *Scalepack* keyword, 129
Floor Down *XdisplayF* button, 20
Floor Up *XdisplayF* button, 19
format, 129
FORMAT *Denzo* keyword, 60, 64, 65
FORMAT *Scalepack* keyword, 129
Fourier map, 8
frame width, 129
Frame *XdisplayF* button, 20
Fuji scanners, 83
Full Scale *XdisplayF* button, 19

G

GO *Denzo* keyword, 60
GO statement, 34
Go *XdisplayF* button, 19
goniometer, 25, 82
graphite, 63
Gravity/beam system, 50
Green Box, 21
guard, 65

H

Hamlin (SDMS) ⇒ detectors, 129
hand of the structure, 50
Heavy Atom Search , 98
Help *XdisplayF* button, 20
histogram in *Denzo* log file, 37
history of *Denzo*, 4
hkl, 66
hkl matrix, 130
hkl scale, 130
hkl shift, 130
hklpredictions, 18, 22, 75
reference *Denzo*, 4
HP, 79

I

I+ and I- , 113
IBOX *Denzo* keyword, 60
ice, 144
ice rings, 27, 31, 41, 60, 137

icing, 144
Ignore, 155
IGNORE *Denzo* keyword, 60
ignore overloads, 130
imaging plate, 61
IMAGING PLATE *Denzo* keyword, 58, 60
include no partials, 132
include overloads, 130
include partials, 132
indexing, 8
  manual indexing, 80
infinite loop, 158
INITIAL B FACTOR *Scalepack* keyword, 124, 131
INITIAL SCALE *Scalepack* keyword, 131
INPUT *Denzo* keyword, 61
input file format, 48
INPUT *Scalepack* keyword, 131
Int. Box *XdisplayF* button in *ZoomWindow*, 21
intensity, 21
  displayed in *XdisplayF*, 21
  maximum pixel intensity in *XdisplayF*, 21
INTENSITY AVERAGING WINDOW *Denzo* keyword, 61
interactive processing by *Denzo*
  keywords required for…, 78
interactive session, 43
International Tables, 118

K

keyword
  format conventions, 27
keywords
  defaults, 48
Keywords with no Defaults, 78

L

λ, wavelength, 35
LAMBDA *Denzo* keyword, 62, 75
lattice
  superlattice, 33
LATTICE *Denzo* keyword, 62
Launching the program, 17
LEFT MARGIN *Denzo* keyword, 62
length, 58
linear and square R_{merge}, 96
Linux operating system, 79
LIST *Denzo* keyword, 36, 62
local scaling, 132
log file, 15, 67
LONGEST VECTOR *Denzo* keyword, 31, 62, 145
Lorentz polarization factor, 6
lune, 31
lysozyme, 8

M

MacScience, 53
MacScience (DIP) Þ detectors, 57
MAD, 132
MAD structure determination method, 10
madnes, 129
magnification, 21
major axis, 80
major zone, 80
MAR, 39
memory, 17, 66, 79, 148, 149, 157, 158
MERGE *Scalepack* keyword, 131
Minor Wladek, wladek@iwonka.med.virginia.edu, 4
MIPS R8000, 79
misindexing, 32, 146, 159
MONOCHROMATOR *Denzo* keyword, 63
mosaicity, 8, 14, 156
  mosaicity modifier to **FIT** *Scalepack* keyword, 127
  very low, 31
MOSAICITY *Denzo* keyword, 63
mosaicity histogram, 45
MOSAICITY *Scalepack* keyword, 132
mouse buttons, 20, 26
  left, 20, 153, 160
  middle, 13, 18, 20
  right, 19, 20
Multiple Native data sets , 100
multiple passes of data collection, 116
myexperiment.dat command file, 13

N

negative intensities, 140
Negative I's, 157
NO ANOMALOUS *Scalepack* keyword, 132
NO BEAM *Denzo* keyword, 64, 74
NO MERGE *Scalepack* keyword, 132
NO PROFILE TEST *Scalepack* keyword, 133
none, 65
non-isomorphism, 90, 99
non-vertical spindle, 44
NUMBER OF ITERATIONS *Scalepack* keyword, 133
NUMBER OF RECORDS *Denzo* keyword, 64
NUMBER OF ZONES *Scalepack* keyword, 133
numerical parity, 21

O

Off-line Image Plate, 83
OMIT *Denzo* keyword, 64
OPERATING SYSTEMS
  Linux, 79
  OSF, 79
  Unix, 79
  VMS, 22, 79, 142, 149, 150, 157
operative philosophy, 79, 84
ORIENTATION AXIS1 *Denzo* keyword, 72
ORIENTATION AXIS1 *Scalepack* keyword, 133
ORIENTATION AXIS2 *Denzo* keyword, 75
ORIENTATION AXIS2 *Scalepack* keyword, 133
original index, 132
ORIGINAL WINDOW *Scalepack* keyword, 134
OSCILLATION *Denzo* keyword, 64
oscillation image, 27
oscillation range, 8, 30, 31, 145
  low range & still images, 31
OSF, 79
Otwinowski Zbyszek zbyszek@mix.swmed.edu, 4
outliers, 8, 137
outliers rejection, 8, 137

OUTPUT ANOMALOUS *Scalepack* keyword, 134
output file, 58
OUTPUT FILE *Denzo* keyword, 76
output file format
Denzo _ip, 151
Denzo _york1, 151
OUTPUT FILE *Scalepack* keyword, 134
Outside, 46
overflows, 152
OVERLAP *Denzo* keyword, 65
overlaps, 8, 31
overload, 57, 158
OVERLOAD VALUE *Denzo* keyword, 65
overloaded reflections, 92, 96

P

page file quota, 149
parser pitfall, 48
partiality, 57
partiality refinement, 36
partially recorded reflections, 6, 83, 92
partials, 74
Patterson, 114
Peak picking, 26
PEAK SEARCH FILE peaks.file *Denzo* command, 66
Peak search *XdisplayF* button, 20
peaks.file, 20, 22, 27
phase information, 89
Photon Factory, 55
Photon Factory - *see* synchrotrons, 58
plate rotation, 66
polarization, 115, 158
POLARIZATION *Denzo* keyword, 63
POLARIZATION *Scalepack* keyword, 134
polymorphic crystals, 146
POSTREFINE *Scalepack* keyword, 97
POSTREFINE *Scalepack* keyword, 135
postrefinement, 30, 37, 126
postscript file., 19
potential derivative, 98
precession method, 6
predicted decrease , 42
predictions, 14
PRINT *Denzo* keyword
chi square modifier, 66
PRINT *Scalepack* keyword, 135
print zones, 146
print/end print, 66
Profile Fit R *XdisplayF* button, 19
profile fitting, 14, 150, 151
profile fitting radius, 19
profile fitting radius, 14, 68, 151
PROFILE TEST *Scalepack* keyword, 135
profiles, 66
PROFILES FITTED *Scalepack* keyword, 136
PROFILES SUMMED *Scalepack* keyword, 136
prompt,, 22
pseudo symmetry, 26
pseudo zone, 81

Q

quadrilateral, 61, 155
QUIT *Denzo* keyword, 68, 73

R

RADIAL OFFSET *Denzo* keyword, 35, 53, 68
radiation damage, 96
radius of convergence, 34
RAM, 17
RAW DATA FILE *Denzo* keyword, 69
R-axis ⇒ detectors, 39, 57
reciprocal space, 22, 31, 33, 128
recommended hardware, 79, 84
RECORD LENGTH *Denzo* keyword, 69
RECSQ *Scalepack* keyword, 136
redundancy, 90
reference batch, 89
REFERENCE BATCH *Scalepack* keyword, 95, 97, 98, 100, 103, 136
REFINE NO PARTIALITY *Denzo* keyword, 69
REFINE PARTIALITY *Denzo* keyword, 69
refine.dat command file, 13
refine.dat command file, 43
refinement, 9, 14, 15
refinement Q&A, 159
strategy, 6
refineone.dat, 35
reflection conditions, 118
Reindexing, 102
reject, 70
REJECT *Denzo* keyword, 42, 70
REJECT HKL *Scalepack* keyword, 137
REJECT OUTLIERS *Scalepack* keyword, 137
reject.dat, 137
rejection criteria, 70, 91
REJECTION PROBABILITY *Scalepack* keyword, 137
rejects, 89
remeasured, 46
reprocess, 37
RES *Denzo* keyword, 70
resolution
displayed in *XdisplayF*, 21
resolution cutoff, 146
resolution limit, 8, 92
RESOLUTION LIMITS *Denzo* keyword, 31, 70, 74, 145
RESOLUTION *Scalepack* keyword, 138
resolution shells, 92
RESOLUTION STEP *Scalepack* keyword, 138
Reverse *XdisplayF* button, 20
Rhombohedral, 160
R-linear, 137
R_{merge}, unweighted error measure, 8, 10, 90
rotaprep, 148
rotation, 58
R-squared, 137

S

σ, standard variation of intensity, 14, 39, 88, 90, 104, 105, 106, 138
F**2, 151
output file with σ_{I+}, σ_{I-}, 113
SIGMA CUTOFF *Scalepack* keyword, 140
saturated image pixels, 20
SCALA, 148
SCALE ANOMALOUS *Scalepack* keyword, 92, 139
SCALE RESTRAIN *Scalepack* keyword, 139

SCALEPACK, 44
scaletwosca.com command file, 101
scattering, 70
Scenarios, 94
screen output., 28
screening through your frames, 19
SDMS (Hamlin) Þ *detectors*, 129
sector, 159
SECTOR *Denzo* keyword, 71, 72
SECTOR *Scalepack* keyword, 139
SECTOR WIDTH *Scalepack* keyword, 139
separators, 48
SGI, 149
shifts, 67
Show overflow *XdisplayF* button, 20
shutter, 96, 153
shutter malfunction, 8
sigma cutoff, 91, 139
SIGMA CUTOFF *Scalepack* keyword, 91, 139
simulation as a teaching tool, 33
simulation of diffraction pattern, 8
single measurements, 91
site.dat command file, 13
  example, 27
SKEW *Denzo* keyword, 35, 71
slope, 70
small molecules, 5, 10, 31, 54, 89
software
  agrovata, 152
  rotavata, 152
  Scala, 152
solver, 67
SPACE GROUP *Denzo* keyword, 71
  as replacement to LATTICE keyword, 62
Space Group Identification, 118
SPACE GROUP *Scalepack* keyword, 140
space groups, 8
spindle axis, 80
SPINDLE AXIS *Denzo* keyword, 56, 72, 80
SPINDLE AXIS *Scalepack* keyword, 133, 141
spindle motor backlash, 8
Spindle/beam orinetation system, 50
SPOT, 38, 65
  distortion, 57
  elliptical, 38
  radius, 38
  spot radius, 62
SPOT *Denzo* keyword, 31, 72, 145
spot profile, 9
spot separation, 8
spot shape, 37, 39
spots
  ellipsoidal, 38
spread, 8
spurious peaks, 31
SSRL, 158
start refinement, 160
START REFINEMENT *Denzo* keyword, 72
starting point, 20
statistic error measures
  problems with no good statistical criteria, 10
statistic error measures
  χ^2, weighted, 10
  R_{merge}, unweighted, 10
statistical error, 95
statistics, 67
still images, 31
STOP *Denzo* keyword, 73
structure solving, 10
successfully indexed, 80
sulfur, 8
summary, 36
Sun, 79
superlattice, 31, 33
SWAP BYTES Denzo keyword, 73
swap space, 17, 148
synchrotrons, 9, 30, 44
  Brookhaven, 83
  CHESS, 83, 116
  Daresbury, 158
  Hamburg, 159
  Photon Factory, 55, 58, 83

T

Text strings, 48
thin frames, 139
Thin or few frames , 117
TITLE *Denzo* keyword, 73
TOP MARGIN *Denzo* keyword, 73
total error, 95
triangle, 61
TRUNCATE, 140

U

unit cell, 8
  cell reduction, 26
UNIT CELL *Denzo* keyword, 55, 73
UNIT CELL *Scalepack* keyword, 141
Unix, 79
unsufficient space Denzo message, 157
Update Pred. *XdisplayF* button, 19
USE BEAM *Denzo* keyword, 44, 64, 74
USE FIDUCIALS *Denzo* keyword, 74
USE NO FIDUCIALS *Denzo* keyword, 74
USE NO BEAM *Denzo* keyword, 44
USE PARTIALS NO POSITION *Denzo keyword*, 74
USE PARTIALS POSITION *Denzo* keyword, 74

V

varying mosaic spread, 156
vertical axis, 80
VERTICAL AXIS *Denzo* keyword, 56, 75, 80
VERTICAL AXIS *Scalepack* keyword, 133, 142
visualisation of the data, 10
VMS OPERATING SYSTEM, 22, 79, 142, 149, 150, 157

W

wavelength, 35
  Cu K_α (1.5418), 62
  LAMBDA *Denzo* keyword, 62
WAVELENGTH *Denzo* keyword, 75
WEAK LEVEL *Denzo* keyword, 31, 75
weak reflections, 91
weak reflections, 46

weaker pixels, 20
Weissenberg method, 6, 55, 145
width, 58
workstations
  Alpha, 149
  DEC Alpha, 13, 149, 150
  HP, 79
  SGI, 79, 149
  SGI Indigo, 13
  SUN, 79, 149
WRITE BADDIES *Scalepack* keyword, 142
WRITE NO PREDICTIONS *Denzo* keyword, 75
WRITE PREDICTIONS *Denzo keyword*, 75
WRITE REJECTION FILE *Scalepack* keyword, 137, 142
Write/Print *XdisplayF* button, 19

X

X BEAM *Denzo* keyword, 27, 30, 55, 76, 83, 144
x beam *Denzo* log statement, 47
XDisplayF
  general description, 17
xengen urf file format, 129
X-plor, 133
xtal###.x files, 15
X-terminal, 79

Y

Y BEAM *Denzo* keyword, 27, 30, 55, 76, 83
y beam *Denzo* log statement, 47
Y SCALE *Denzo* keyword, 35, 50, 51, 76
york output file, 76, 148
YORK OUTPUT FILE *Denzo* keyword, 76

Z

Zero sigma, 160
zones, 67
Zoom close, 21
Zoom In *XdisplayF* button in *ZoomWindow*, 21
Zoom Out *XdisplayF* button in *ZoomWindow*, 21
Zoom Window *XdisplayF* button, 19