Contents

1 Plotting Data

1.1 Loading one or multiple files

To load and to plot Flatcone data, there is a single command: `plotmultiple`. It loads the specified files, eventually applies a detector efficiency correction, performs the conversion into the desired coordinate system, combines equivalent points and opens a figure window containing the color plot. Data is stored with the figure, and the figure's menus give access to most further options.

You can specify one or several files to plot. In the latter case, some checks for consistency of the data sets (depending on the plot type; for instance test for same energy or vertical momentum transfer) are performed and eventually a warning or error message issued in the command window. The scans that are to be combined **do not need to have identical step size, number of points, range, detector angles, monitor, etc.** They are combined in the most flexible way by maintaining the individual coordinates of each point, and the color plot is performed using a technique based on Voronoi diagrams (see section 1.3). This permits for example to use short scans with smaller step size, interleaving detector angles, longer monitor, etc. to examine a region of interest in more detail than the rest of the map. All data are normalized to the standard monitor defined in the options file.

When specifying multiple files to plot, use the usual Matlab syntax with squared brackets, for instance:

```
pathname/05502[1,2]
pathname/05502[1:8]
pathname/05502[0:2:8]
pathname/05502[1,4,7,8]
pathname/055[020,035,099,108:114]
```
etc.

The full syntax of `plotmultiple` is

```
plotmultiple(filenames, coordtype, plottype)
```

- `filenames` contains the filename(s), including path if not accessible from current Matlab directory. If multiple files are given, the data of these files are combined to a single slice. For plotting several slices in a three dimensional plot, give a cell array of strings: each element of the cell array (which may consist of several file names each) will then be a separate slice in the plot.

- `coordtype` defines in which coordinate system the data are to be treated. This defines the way of averaging, binning, etc. and in particular the calculation of the graphical representation (Voronoi cells). Currently available: **angles** (scattering and sample rotation angle, i. e. [a4, a3] for $q^{\perp}$=0), **qxy** ($\vec{q}$ in orthogonal axes, in $Å^{-1}$: [qx, qy]), **energy3d** (angles + energy: [scatt.-ang., energy, sample-ang.]) and **energyproj** (consider only $|q|$: [scatt.-ang., energy]).

- `plottype` defines the coordinate system used for the plot. Currently available: **qxy** (axes: qx($Å^{-1}$), qy($Å^{-1}$)), **angles**, **qeplane** (axes: energy vs. $|\vec{q}|$) and **qxqyen** (3D-plot: qx, qy, en).
  Note that `plottype` does not have to be the same as `coordtype` - for instance, the typical case is to treat a psi-scan in angle space and to plot it in qxy space. However, not all coordinate transformations are physically possible, and not all possible ones are implemented.

By default, plotmultiple assumes ”angles” and ”qxy”, so in the most common case of a psi-/a3-scan with constant energy transfer and $q^{\perp}$, you do not have to specify `coordtype` and `plottype`, for example

```
plotmultiple('012345');
```

or simply: `plotmultiple 012345 ;`

For energy scans, you might type

```
plotmultiple 012345 energy3d qxqyen ; or
plotmultiple 012345 energy3d qeplane ; [1]
```

1.2 The plot window

The plot window keeps the usual functionality (menus, toolbars) of a standard Matlab figure window. In addition to that, some specific commands and menus are available. The FLATCONE menu is accessed either via the menu bar or the right mouse button.[2] In particular, you can:

- Change the color scale between logarithmic and linear. The standard setting is defined in the options file.

[1] For a ”qeplane”-plot, you notice no difference if you give ”energy3d” or ”energyproj”, as the calculated 2D graphical representation is the same.

[2] Remember that some Matlab tools (like the hand, the zoom, the pointer, etc.) alter the mouse behavior and must not be activated to use the special Flatcone ones.

- Toggle between direct and interpolated representation of the data. *Direct* plots the Voronoi cell of each data point in a single color. "Interpolated" uses the "FaceColor/interp" option of the Matlab `patch` command, which is a bilinear interpolation of the color at each vertex using a Delaunay triangulation (a simple size check eliminates too large triangles). Though it produces smooth maps in many cases, it may create artifacts in other cases and is to be used with care.

- *Data grid* shows the edges of the Voronoi cells. *Measured points* shows the coordinates of the single data points.

- Orienting vectors displays the two orienting vectors ([ax,ay,az] and [bx,by,bz]) as defined in the MAD control software. HKL-grid displays dotted lines constructed using these vectors and their multiples.
  In the case of $q^{\perp} \neq 0$ and non-orthogonal crystal system, the origin used to plot these arrows and lines is the projection of a third, "vertical" vector (orthogonal to the two orienting vectors in HKL(!) space) onto the qx,qy-plane.

- You can delete single or multiple points by clicking the mouse button or dragging a frame.[3] The graphical representation (Voronoi cells and Delaunay triangulation) are reconstructed locally. This means that in general deleted cells will be replaced by enlarging their neighbors, and if the neighbors are too far, holes will appear.

- You can define different ways of extracting one-dimensional ("scan") data from the map. For more information on the different way of extracting 1D Data, see Section 1.5. New axes are created, and you define the scan either using the mouse or the menu entry in the newly appeared "Scan" menu. The "Scan" menu also permits to change the axes coordinates ("auto" takes the coordinate with the largest change as the x-axis). The points are equidistant. An "intelligent" guess for the number of points is performed. Note however, that a reasonable choice for the number of points is very important!! For instance, a too high number may, in combination with the interpolation routines, yield an unphysically smooth set of points and provide misleading information!

- The save command saves the data points with their values and errors as a linear list (using the plot coordinates) in a file.

1.3 The graphical representation of 2D-Data

Given a set of points with respective coordinates, that represent each a single measurement of one detector, it is in the general case not always trivial to obtain a filled

[3] For cells at the edges, note that you must click inside one of the Delaunay triangles.

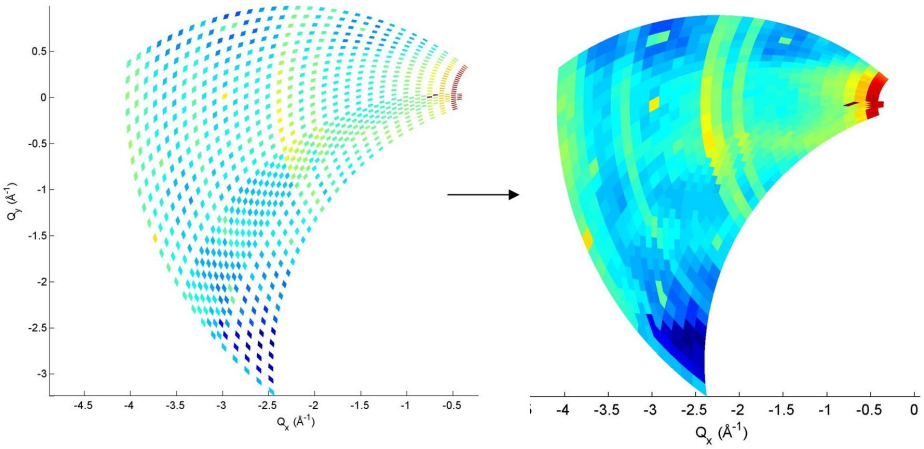

Figure 1.1: Data points from two psi scans (a long and a short one with different a4). By enlarging the cell size `stdcell`, a continuously filled area is obtained.)

colored area that is representing the information in a both physically reasonable and optically good-looking way.

Different methods may be divided into those that involve some interpolation to fill the area between the discrete points and a direct representation.

For interpolation, several possibilities and algorithms exist, and Matlab provides a number of routines the perform this task in different ways and create different kinds of plots (`patch`, `contourf`, ...), all of which have their particular advantages and disadvantages and may in some situations yield inappropriate or misleading results. The Flatcone Matlab software uses the standard Matlab routines, mostly a bilinear interpolation based on a Delaunay triangulation of the data points in 2D space.

The direct representation is more accurate in the sense that it represents each measured point directly without underlying calculation to determine the color. As the points are usually not regularly spaced (depending on the measurement strategy and the coordinate system), the different data points will of course occupy regions of different size and shape. In order to do this in an as reasonable and general way as possible, the representation is based on a Voronoi diagram, i. e. each point of the area plot is colored according to the *nearest* data point. As those regions of the map that are very far from any data should not be colored, a maximum size limit is applied to the cells surrounding each data point. This size limit is defined by the parameter `stdcell` in the options file for the different coordinate systems that can be used.

`stdcell` is defined for different coordinate systems (one number for each dimension). When the coordinate axes have different physical units (for instance meV and degrees or $Å^{-1}$), strange effects can arise because the graphics algorithms and their metrics work with dimensionless numbers. To avoid, `stdratio` (options file) is used to scale the different axes to reasonable numerical values with respect to each other.

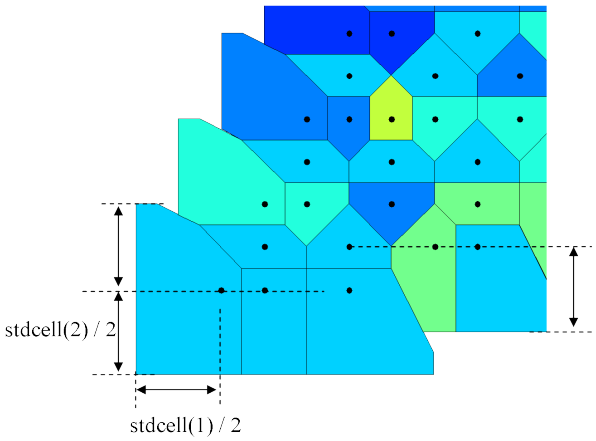

Figure 1.2: Data points (black dots) and Voronoi cells. At the edges, the cells are cut at a distance defined by the parameter `stdcell` in the options file. (This example is quite irregular on purpose in order to illustrate the principle.)

Figure 1.1 demonstrates the transition from individual cells (small value of `stdcell`) to a Voronoi diagram that fills the area without gaps. See also Figure 1.2 for an illustration of the Voronoi cells and the applied size maximum.

For a nice plot, it should be chosen such that the area is continuously filled (i. e. depending on the step sizes of the scan); then the cutting occurs only at the edges. On the other hand, by setting too large values, you risk coloring areas of the map for which actually no measured information is available.

Bear in mind that the sizes of the cells can be quite different, which gives some points a larger weight in the overall optical impression than others. Note also that the Voronoi cells and the way of cutting at the edges depend on the coordinate system used to calculate it. For instance, a simple psi-scan produces points on a regular rectangular grid in angles space but is more complicated in qx,qy-space. All calculations are performed in the coordinate system defined by the parameter `coordtype`, and the calculated Voronoi cells then undergo just the coordinate transformation, but no other change, for plotting purposes.

See the section on binning (1.4) for further remarks. Binning influences the calculated cells and is defined by the parameters in the options file (1.7).

1.4 Averaging, binning, and normalization

Normalization. All data are plotted and stored after normalization to the same monitor. This is typically either M1 or TIME. To define the monitor variable and the desired value, set `normalizeto` and `normval` in the options file.

Averaging. Points that are considered to have equal coordinates (see "binning") are averaged to obtain a single value at this point. There are two ways to obtain the resulting value. When combining scans, the original detector count rates are summed and then normalized by the sum of the (effective) monitor values, and the error bar is as usual obtained as the square root of the total true detector count rate. On the other hand, when combining data that do not correspond to original counts (for instance after subtracting background, etc.), the error bar is no longer the square root of the value. In this case, a weighted average is used,

$$\bar{x} = \frac{\sum x_i \cdot w_i}{\sum w_i} \quad \text{and} \quad \bar{\sigma} = \frac{\sqrt{\sum w_i^2 \cdot \sigma_i^2}}{\sum w_i} \tag{1.1}$$

where the w_i are the weights and the σ_i the error bars of the values x_i. By default, the weights w_i are taken as $1/\sigma_i^2$. An internal parameter `raw` in the dataset structure controls which way of averaging is performed.

Binning. In order to be able to average points, one has to determine which points are to be considered as equal (because of the positioning precision of the spectrometer normally no points are strictly equal, and you might want to average neighboring points to obtain a coarser resolution). Therefore, all points are assigned ("binned") to an underlying grid. Points binned to the same grid point are then averaged. The underlying grid is by default constructed as a regular mesh of points, whose distances along the different dimensions are controlled by the parameter `stdgrid` in the options file for the various kinds of coordinate systems. If possible, the step size of the scan, or an integer fraction of it, are a reasonable choice for this parameter in most cases.

In general, it does not necessarily have to be regular but can be any set of points, for instance you may bin one measurement to another one, etc.

The maximum acceptable distance that a point may have to the point it is assigned to, is controlled by the parameter `stdbindist` in the options file. If the distance is larger, the point is discarded. During averaging data and plotting (using `plotmultiple`), the grid points are usually constructed such that there is one near any data point, so this parameter has no effect if it is larger than `stdgrid`. It is important in other cases, for instance when subtracting data sets.

1.5 Extracting scans from two-dimensional data

In most cases, the original data points do not lie on straight lines in Q,E-space. In order to extract a one-dimensional scan from the data, different methods can be used:

- **Interpolation of nearest points.** Between start and end point of the scan, a set of equally spaced points is constructed. The value at each of these points is obtained by interpolation of the surrounding data points.

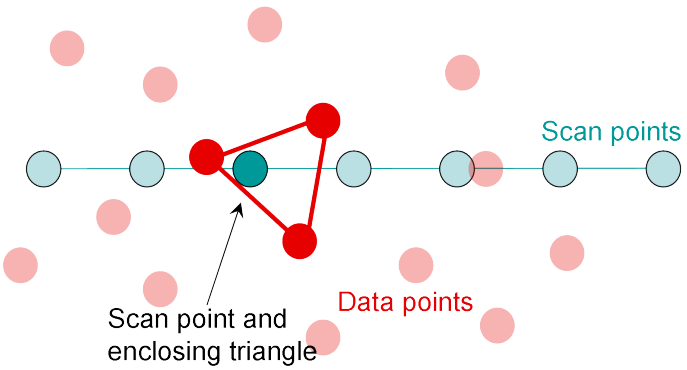

Figure 1.3: The regular scan points and the measured data. For the third scan point, the enclosing Delaunay triangle is highlighted. The three data points that form its vertices are used for the interpolation.

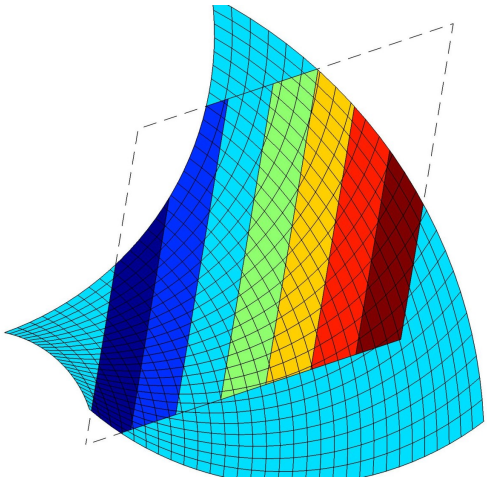

Figure 1.4: An example for the definition of integration regions. Although normally this will be a rectangle, the angle can be freely chosen, as in this example (useful for instance in non-orthogonal crystal systems).

This interpolation is based on the method that is also used by the Matlab routine `griddata`. It uses a Delaunay triangulation of the data points and determines the three vertices of the Delaunay triangle that encloses the desired point. The result is obtained via a weighted average (1.1) of the values and errors of these three data points, and the distances to these points are used as weights. See Figure 1.3 for illustration.

Triangles that are too large (controlled by parameter `stdcell`) are discarded in order to avoid influence from too distant data points. If a scan point is not enclosed by a triangle, no value is plotted.

- **Integration over a finite region.** A parallelogram is defined and divided into a number of equal slices, see Figure 1.4 for illustration. For the integration, each *cell* is considered that lies partially or fully in the slice (even if the data point is outside). The resulting value and error bar are calculated as a weighted average (1.1), in which each cell is – additionally to its error – weighted by its surface area inside the slice.

- **Projection.** Here, no interpolation, averaging etc. is performed, but the original data points are projected on the line. The region from which the data are collected are defined as above (integration), and the projection is performed parallel to the sides of the parallelogram.

1.6 Detector efficiency correction

If you choose to perform a detector efficiency correction, the count rates of each detector are normalized by this channel's efficiency. You can either provide the filename of a scan containing a Vanadium energy scan (or any other data set to be used for normalization) or the explicit correction factors. If you provide a scan file, the count rates of all channels are simply summed up and considered as the channels' efficiencies. If you prefer to use other numbers, for instance the results of more a sophisticated analysis of a Vanadium scan or numbers from other sources, give these explicitly in the options file (31 numbers).

In both cases, the detector efficiencies are normalized so that their average is 1 before applying the correction to the data.

Technical remark The efficiency correction is done by considering an "effective" monitor count rate, which is the true monitor times the efficiency (following the idea that a more efficient channel is effectively counted longer). Any normalization is then done using the effective monitor count rate. By storing the effective monitor for each point, one is also able to retrieve the real original count rate, which is important at some occasions later on to perform correct Gaussian error bar statistics.

1.7 The options file

Open the options file in the editor or type `edit options` in the command window. This file contains all important settings that define the behavior of the program, like settings for detector correction, instrument parameters, normalization, averaging and plotting behavior.

The program always looks in `options.m` when it needs some values. It must therefore be accessible in the path, and it might be useful to keep a copy with standard settings under another name.

The meaning of most entries is explained in the comments within the file itself. Note that it contains important instrument setup parameters like zeros, goniometer directions and the use of a3p, which may vary between different spectrometers and measurements and which can at present *not be inferred from the data files*. See also the sections on plotting (1.3) and averaging etc. (1.4) for explanation of some parameters.

2 Data manipulation from the command line

2.1 Data sets as workspace variables

Data loaded with `plotmultiple` are stored with the figure window and not visible in the Matlab workspace. To store the data from a window into a Matlab variable, use

```
data = getfiguredata ( figure )
```

where `figure` is the number of the figure window (use `gcf` for current figure).

You obtain a structure that contains at least the fields `coordlist` and `valuelist`, which are linear lists of the coordinates of data points (number of dimensions and meaning dependent on `coordtype`) and the normalized values (`coordlist(:,1)` are data, `coordlist(:,2)` are errors). There may further be the fields `vertexlist`, which are the vertices of the Voronoi cells, and `faces`, each line of which contains indices into `vertexlist` for one cell. The field `constants` contains the names of constants as strings, and for each entry there is an additional field of this name that contains the value of this constant.

To plot the structure `data`, use

```
fcplot( data , plottype )
```

(if plottype not given, `'qxy'` is taken by default).

2.2 Subtraction and smoothing

Subtract two data sets from each other by

```
diff = subtractdata ( data1, data2 , mode, opt )
```

This results in `diff` = `data1-data2`, where `data1` and `data2` are data sets (Matlab structure as discussed above) as obtained for instance by `getfiguredata`. There are different ways how the subtraction is performed, that are chosen by the parameter `mode`:

- 'Nearest'. From each point in data1, subtract the nearest point in data2. The parameter opt (optional) is an array that defines the maximum acceptable distance along the respective dimensions. If for a point in data1 no point in data2 is found within this range, it is neglected and does not appear in the result. If opt is not specified, the value of stdbindist from the options file is used.

- 'Interpolate'. Linear interpolation of data2 to determine the value to be subtracted. The parameter opt (optional) defines the maximum size of the Delaunay triangle used for interpolation - this permits to avoid "unphysical" interpolation by using too distant points from data2.

- 'Range'. Subtract the average of the values in data2 over a given range (defined by opt). The value to be subtracted is the weighted average of all points in data2 that lie within the given range around the respective points in data1.

The points in the result diff have the coordinates of the points in data1. Their error bar is $\sqrt{\mathrm{err1}^2 + \mathrm{err2}^2}$, if $\mathrm{err1}$ and $\mathrm{err2}$ are the error bars of the points in data1 and data2 (the latter eventually the error bar of the interpolated or averaged value).

The command

```
newdata = smoothdata (data , range )
```

smoothes the data set data by replacing the value and error of each point by the weighted average over the range defined by range along the different dimensions. For each point i, all those points are considered for which the difference in all coordinates is less or equal than the values in range.

2.3 Background, powder line and spurion subtraction

If no real background measurement (empty cryostat etc.) is available, you can, to some extent, still perform some basic background and powder line subtraction by using the above commands by using interpolation or averaging.

For instance, when you want to resolve a signal in a limited Q-range, you may create a dataset to subtract by opening the original data set in a figure window and deleting the region where the signal is expected, see Figure 2.1. Then smooth this dataset to level out the statistical variations and to fill the deleted regions. Then subtract the so obtained data set from the original one. By using the 'range' option in subtractdata, one can also directly subtract it, for example (with data2 being the data in Figure 2.1):

```
subtractdata ( data, data2, 'range', [.1,inf] )
```

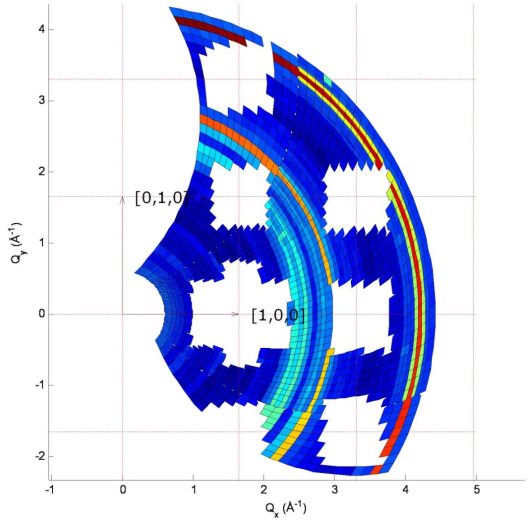

Figure 2.1: Data after manual deletion of regions where a signal is expected (integer H,K in this case). The remaining data is considered as "background". As it depends predominantly on a4 and very little on a3, the missing regions are obtained by averaging over a3.

The same procedure would work if you have a separate data set from background measurement that has only been performed at a single or few a3 – by putting `inf` for the a3 range, the same value would be subtracted independent of a3.

It is important to make use of the best suited coordinate system. For instance, if dominant powder lines are present, one should average along a3, i. e. one has to use angles as coordinates. In the above example, by giving [.1,inf] a complete average over a3 and essentially no averaging along a4 is performed. This procedure does not only reveal the signal but also removes most of the powder line contamination.

The functions `smoothdata` and `subtractdata` always work in the coordinate system of the dataset. If you want to use a different coordinate system, for instance to smooth a dataset over a region defined in qx/qy (in $Å^{-1}$), transform the dataset to these axes first:

```
newdata = coordtransform ( data, 'qxy' )
```

and then use `smoothdata` with the range given in $Å^{-1}$.

To remove sharp features in the background like spurious peaks, it is best to delete them manually in the figure window. For the plot, the deleted points are automatically colored with the colors of the neighboring cells (cf. Section 1.3). By this, also powder lines can very easily be deleted by plotting the data in angles coordinates.

2.4 Other basic operations

Scaling data. The command

```
scaledata ( data, factor )
```

returns the data set scaled by the number `factor`.

Summation. You can just use

```
subtractdata ( data1, scaledata(data2, -1), mode, opt )
```

It is important to note that due to the asymmetry of the `subtractdata` function, the result may not be the same when `data1` and `data2` are exchanged.

If several data sets are to be combined (with or without averaging), one can use `cmbavg` instead.

List combination. ...

3 Angle calculations

3.1 Coordinate system

The **laboratory system** is defined such that $\vec{k_i} \parallel y$, and z is pointing upwards. (x completes the right-handed system.)

The **sample system** is attached to the sample holder and turns with a3, gu, gl and a3p. For a3,gl,gu,a3p=0 its x axis is parallel k_i and its z axis vertical. This corresponds to the MAD convention that for zero a3,gl,gu,a3p the first orientation vector [ax,ay,az] is parallel to k_i. The sample system is related to the H,K,L coordinates via the UB matrix,

$$\begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix} = UB \cdot \begin{pmatrix} H \\ K \\ L \end{pmatrix} \tag{3.1}$$

Note that for zero angles this system is not identical to the laboratory system, as the x,y axes are rotated by 90°.

Attention: Zero a3,gl,gu,a3p do not generally correspond to zero physical angles, i. e. flat goniometers etc. This is because the zero values za3, zgl etc. are set in the way to compensate for the initial misalignment of the crystal when it was mounted, see following section. This means for instance that the z axis is generally not parallel to the a3p axis.

3.2 Zeros

The convention on the ILL triple axis spectrometers is such that for a3=0, gl=0, gu=0, a3p=0 the scattering plane defined by the two orientation vectors [ax,ay,az] and [bx,by,bz] is exactly horizontal in laboratory space and [ax,ay,az] parallel to k_i. Typically this requires that after sample alignment the a3 table is not exactly aligned with k_i and that the goniometers are not physically flat, i. e. that the zeros za3, zgl, zgu, za3p are set to values that do not correspond to their physical zeros zgl0, zgu0 and za30 (usually za30=-180, zgl0=zgu0=0).

In earlier times when only working in two dimensions within the scattering plane, zgu and zgl were often not changed, but the measurement performed at finite values of gu and gl. As with the 3D option, gu and gl are calculated and moved at every positioning, it is necessary to set all zero values properly.

The matrix $\mathbf{U_0}$. To describe the rotation that is due to the zero settings, the matrix U_0 is used. It has the form

$$\mathbf{U_0} = \mathbf{R_{a3}}\big(-(\text{za3-za30})\big) \cdot \mathbf{R_{gl}}\big(-(\text{zgl-zgl0})\big) \cdot \mathbf{R_{gu}}\big(-(\text{zgu-zgu0})\big) \cdot \mathbf{R_{gl}}\big(-(\text{za3p})\big) \quad (3.2)$$

The minus signs are due to the fact that a setting of za3 etc. corresponds to a physical rotation in the opposite direction.

The values za30, zgl0 and zgu0 are *not* considered in the instrument control software. This means that the encoder parameters of the spectrometers must be set such that these are zero (or -180 for za30 on IN14 and IN20). The Matlab routines nevertheless allow these parameters to be set via the options file.

The zero of a3p does in fact not have a physical meaning as it is the innermost rotation axis, and also the value of za3p cancels out later on.

3.3 Rotation matrices

Sample rotation

Define rotation matrices of the following form

$$\mathbf{R_{a3}}(a3) = \begin{pmatrix} \cos a3 & -\sin a3 & 0 \\ \sin a3 & \cos a3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (3.3)$$

$$\mathbf{R_{gl}}(gl) = \begin{pmatrix} \cos gl & 0 & \sin gl \\ 0 & 1 & 0 \\ -\sin gl & 0 & \cos gl \end{pmatrix} \quad (3.4)$$

$$\mathbf{R_{gu}}(gu) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos gu & -\sin gu \\ 0 & \sin gu & \cos gu \end{pmatrix} \quad (3.5)$$

$\mathbf{R_{a3p}}$ has the same form as $\mathbf{R_{a3}}$. These are the basic rotation matrices of the respective spectrometer motors, defined in the laboratory system. However, if there are offsets in the zero positions, the inner rotation axes change their orientation and therefore the rotation matrices in laboratory coordinates. For instance the true direction of the a3p-axis as the innermost axis depends on the positions of the goniometers and a3. Including this effect, one has

$$\begin{array}{lcllll}
\mathbf{M_{a3p}} & = & \mathbf{R_{a3}}(\tilde{\mathrm{za3}}) \cdot \mathbf{R_{gl}}(\tilde{\mathrm{zgl}}) \cdot \mathbf{R_{gu}}(\tilde{\mathrm{zgu}}) \cdot & \mathbf{R_{a3p}}(\mathrm{a3p}) & \cdot \mathbf{R_{gu}}(\tilde{\mathrm{zgu}})^{-1} \cdot \mathbf{R_{gl}}(\tilde{\mathrm{zgl}})^{-1} \cdot \mathbf{R_{a3}}(\tilde{\mathrm{za3}})^{-1} \\
\mathbf{M_{gu}} & = & \mathbf{R_{a3}}(\tilde{\mathrm{za3}}) \cdot \mathbf{R_{gl}}(\tilde{\mathrm{zgl}}) \cdot & \mathbf{R_{gu}}(\mathrm{gu}) & \cdot \mathbf{R_{gl}}(\tilde{\mathrm{zgl}})^{-1} \cdot \mathbf{R_{a3}}(\tilde{\mathrm{za3}})^{-1} \\
\mathbf{M_{gl}} & = & \mathbf{R_{a3}}(\tilde{\mathrm{za3}}) \cdot & \mathbf{R_{gl}}(\mathrm{gl}) & \cdot \mathbf{R_{a3}}(\tilde{\mathrm{za3}})^{-1} \\
\mathbf{M_{a3}} & = & & \mathbf{R_{a3}}(\mathrm{a3}) &
\end{array} \tag{3.6}$$

where the $\tilde{\mathrm{za3}}$ abbreviate the rotation angles due to the zeros, $-(za3 - za30)$, as in (3.2). For the overall rotation of the sample one has to consider the correct order of rotations (because of the nested axes) and obtains:

$$\begin{array}{rcl}
\mathbf{R_S} & = & \mathbf{M_{a3}} \cdot \mathbf{M_{gl}} \cdot \mathbf{M_{gu}} \cdot \mathbf{M_{a3p}} \\
& = & \mathbf{R_{a3}}(\mathrm{a3} + \tilde{\mathrm{za3}}) \cdot \mathbf{R_{gl}}(\mathrm{gl} + \tilde{\mathrm{zgl}}) \cdot \mathbf{R_{gu}}(\mathrm{gu} + \tilde{\mathrm{zgu}}) \cdot \mathbf{R_{a3p}}(\mathrm{a3p} + \tilde{\mathrm{za3}}\mathrm{p}) \cdot {\mathbf{U_0}}^{-1}
\end{array} \tag{3.7}$$

This is the rotation matrix that describes the overall rotation of the sample due to the spectrometer's a3, gu, gl and a3p.

The angle ψ

In 2D, the sample rotation is described by the angle a3, which is the angle between the first orientation vector [ax,ay,az] and k_i. With a tilted detector, this is no longer the case, and a rotation of a3 alone changes the vertical component of the momentum transfer. In order to keep it constant, i. e. to achieve a scan in a well-defined plane, analogous to an a3 scan in 2D, simultaneous movements of a3 (or a3p), gu and gl are necessary. For this reason, a new variable ψ is introduced, which can be scanned like the others. It does not correspond to a real motor of the spectrometer[1], but is a combination of the other sample rotation angles.

ψ is defined as the angle between [ax,ay,az] and $\psi_{\mathrm{ref}} = \mathbf{R}_\delta \cdot \mathbf{R}_\gamma \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ (which may be regarded as a sort of "rotated k_i"). It is thus straightforward to be calculated from the real sample and detector angles, and also vice versa using (3.16,3.17).

Note that even though for gfc=0 the rotation axes of a3 and ψ are the same, the values of these angles are in general not the same. (Further discussion in section 3.5.)

[1] If the a3p-axis is exactly vertical to the [ax,ay,az], [bx,by,bz] plane (good sample alignment), the ψ-axis may be identified with the a3p-axis. Otherwise the axes are "nearly" parallel.

Detector rotation

Define the angles

$$\begin{aligned} \gamma &= \text{gfc} \\ \delta &= \text{a4} - 52.5^\circ \\ \nu &= 2.5^\circ \cdot (n_{chan} - 1) + 15^\circ \end{aligned} \tag{3.8}$$

For $\delta = 0$ the tilt axis of the detector (gfc-axis) is parallel to the x-axis of the laboratory system, and ν is the angle between the detector channel n_{chan} (=1..31) and k_i. Channel 1 is the innermost detector, and channel 31 the detector with the largest scattering angle.

For a given channel (ν) and detector position (δ, gfc), the direction of k_f in the laboratory system is obtained via the successive rotations

$$\mathbf{R_{det}} = \mathbf{R}_\delta \cdot \mathbf{R}_\gamma \cdot \mathbf{R}_\nu = \begin{pmatrix} \cos\delta & -\sin\delta & 0 \\ \sin\delta & \cos\delta & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix} \cdot \begin{pmatrix} \cos\nu & -\sin\nu & 0 \\ \sin\nu & \cos\nu & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.9}$$

With this, the detector plane normal is

$$\vec{n_{\text{det}}} = \begin{pmatrix} \sin\delta\sin\gamma \\ -\cos\delta\sin\gamma \\ \cos\gamma \end{pmatrix} \tag{3.10}$$

The scattering vector is

$$\vec{Q} = \mathbf{R_{det}} \cdot \begin{pmatrix} 0 \\ k_f \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ k_i \\ 0 \end{pmatrix} = \begin{pmatrix} k_f\,(-\cos\delta\sin\nu - \sin\delta\cos\gamma\cos\nu) \\ k_f\,(-\sin\delta\sin\nu + \cos\delta\cos\gamma\cos\nu) - k_i \\ k_f\sin\gamma\cos\nu \end{pmatrix} \tag{3.11}$$

And, finally, the projection of $\vec{Q}$ on $\vec{n_{\text{det}}}$ is

$$Q^\perp = k_i\cos\delta\sin\gamma = k_i\cos(\text{a4 - } 52.5^\circ)\sin(\text{gfc}) \tag{3.12}$$

This is the vertical component of the momentum transfer. It is always **equal for all channels** (does not depend on ν).

During the measurement, the normal of the reciprocal lattice plane defined by the two orientation vectors [ax,ay,az] and [bx,by,bz] is always kept parallel to the detector plane normal. Therefore, all the $\vec{Q}$'s of all detectors lie in a plane parallel to the basal plane. (Although in principle not strictly necessary, this condition is always kept by the instrument control software. Nevertheless, other planes may still be obtained by changing the orientation vectors.)

3.4 Determining the rotation angles

Solve for a3, gl, gu, a3p

If the scattering vector is known in laboratory space (from the position of the detector) and with respect to the sample holder (from H,K,L via the UB matrix), the desired sample rotation is defined: it has (a) to bring the scattering vector HKL into the detector and (b) keep the z direction of the sample system normal to the detector plane. These two conditions define a rotation matrix M (expressed in the laboratory system) that has to be made equal to $\mathbf{R_S}$ by finding the correct a3, gl, gu and a3p.

The solution for these four angles is not unique. Two possibilities shall be considered: either a3 or a3p is fixed to a constant value (set a3p=0 if a3p is not installed). From equation (3.7) follows, with $\mathrm{a3sum} = \mathrm{a3} + \mathrm{z\tilde{a}3}$ etc.:

$$\begin{array}{lrcl} (1) & \mathbf{R_{a3}}(\mathrm{a3sum})^{-1} \cdot \mathbf{M} \cdot \mathbf{U_0} & = & \mathbf{R_{gl}}(\mathrm{glsum}) \cdot \mathbf{R_{gu}}(\mathrm{gusum}) \cdot \mathbf{R_{a3p}}(\mathrm{a3psum}) \\ (2) & \mathbf{M} \cdot \mathbf{U_0} \cdot \mathbf{R_{a3p}}(\mathrm{a3psum})^{-1} & = & \mathbf{R_{a3}}(\mathrm{a3sum}) \cdot \mathbf{R_{gl}}(\mathrm{glsum}) \cdot \mathbf{R_{gu}}(\mathrm{gusum}) \end{array} \tag{3.13}$$

Depending on whether a3 or a3p is given, the left side of the equation is known. The right side is solved for (a3, gl, gu) or (gl, gu, a3p) as follows. The explicit form of the right hand sides is

$\mathbf{R_{gl}} \cdot \mathbf{R_{gu}} \cdot \mathbf{R_{a3p}} =$

$$\begin{pmatrix} \cos(a3p)\cos(gl) + \sin(a3p)\sin(gl)\sin(gu) & -\sin(a3p)\cos(gl) + \cos(a3p)\sin(gl)\sin(gu) & \sin(gl)\cos(gu) \\ \sin(a3p)\cos(gu) & \cos(a3p)\cos(gu) & -\sin(gu) \\ -\cos(a3p)\sin(gl) + \sin(a3p)\cos(gl)\sin(gu) & \sin(a3p)\sin(gl) + \cos(a3p)\cos(gl)\sin(gu) & \cos(gl)\cos(gu) \end{pmatrix} \tag{3.14}$$

$\mathbf{R_{a3}} \cdot \mathbf{R_{gl}} \cdot \mathbf{R_{gu}} =$

$$\begin{pmatrix} \cos(a3)\cos(gl) & \cos(a3)\sin(gl)\sin(gu) - \sin(a3)\cos(gu) & \cos(a3)\sin(gl)\cos(gu) + \sin(a3)\sin(gu) \\ \sin(a3)\cos(gl) & \sin(a3)\sin(gl)\sin(gu) + \cos(a3)\cos(gu) & \sin(a3)\sin(gl)\cos(gu) - \cos(a3)\sin(gu) \\ -\sin(gl) & \cos(gl)\sin(gu) & \cos(gl)\cos(gu) \end{pmatrix} \tag{3.15}$$

One therefore obtains the angles as

$$\begin{array}{rcl} \mathrm{a3psum} & = & \mathrm{atan2}\big(\mathbf{R}_{2,1}, \mathbf{R}_{2,2}\big) \\ \mathrm{glsum} & = & \mathrm{atan2}\big(\mathbf{R}_{1,3}, \mathbf{R}_{3,3}\big) \\ \mathrm{gusum} & = & \mathrm{atan2}\big(-\mathbf{R}_{2,3}, \sqrt{\mathbf{R}_{2,1}^2 + \mathbf{R}_{2,2}^2}\big) \end{array} \tag{3.16}$$

in the first case (a3 fixed), and

$$\begin{array}{rcl} \mathrm{a3sum} & = & \mathrm{atan2}\big(\mathbf{R}_{2,1}, \mathbf{R}_{1,1}\big) \\ \mathrm{glsum} & = & \mathrm{atan2}\big(-\mathbf{R}_{3,1}, \sqrt{\mathbf{R}_{3,2}^2 + \mathbf{R}_{3,3}^2}\big) \\ \mathrm{gusum} & = & \mathrm{atan2}\big(\mathbf{R}_{3,2}, \mathbf{R}_{3,3}\big) \end{array} \tag{3.17}$$

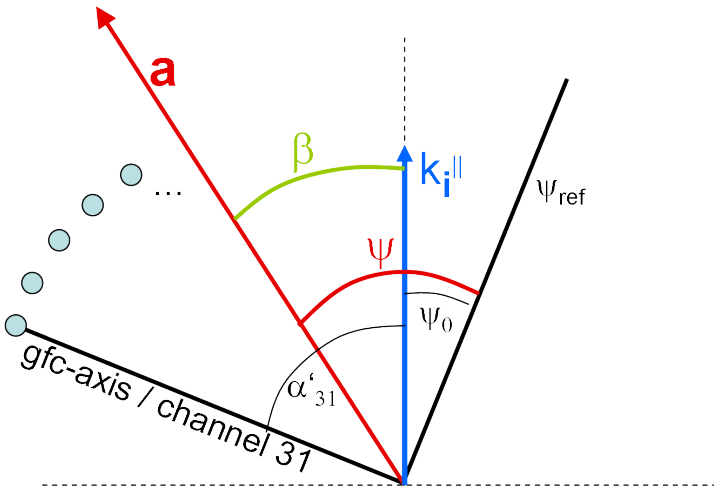

Figure 3.1: Vectors in the detector plane. $k_i^{\parallel}$ is the projection of k_i on the plane, $\vec{a}$ the orienting vector [ax,ay,az].

in the second case. Here the $\mathbf{R}_{i,j}$ are the components of the rotation matrices as obtained from the left hand sides of (3.13).

3.5 The angles plane

The "natural" coordinates of the collected data are the angles – in an a3-scan in 2D, the scattering angle is the same throughout the scan for each channel, and the sample position is described by a3. Operations like binning, averaging, calculating the Voronoi cells, etc. are therefore most simply and naturally done in an coordinate system whose axes consist of sample rotation and scattering angle.

While a3 and a4 can straightforwardly be used as axes in the case without detector tilt, the situation is slightly more complicated in the general case. The angle ψ cannot simply replace a3, as its definition depends on a4, and when the same $\vec{Q}$ is measured in different channels, different ψ are needed. Secondly, a4 has to be replaced by an "in-plane scattering angle" (note that the difference between the real scattering angle of neighbor channels is no longer 2.5° with a tilted detector).

For a visualization, see figure 3.1, which is meant to show the detector plane. For all channels, k_f is in this plane and the angle 2.5° between the k_f of neighbor channels. If the plane is tilted, k_i is not in the plane; let $k_i^{\parallel}$ be the projection of k_i onto the plane. It is thus $k_i^{\parallel} = \sqrt{k_i^2 - Q_\perp^2}$.

Let α'_i be the angle between k_f of channel i and $k_i^{\parallel}$. With α'_i, the in-plane component of the scattering vector is easily calculated, $\vec{Q}_{\text{in-plane}} = \vec{k}_f - \vec{k}_i^{\parallel}$. To obtain from the thus known $\vec{Q}_{\text{in-plane}}$ the scattering vector with respect to the sample coordinates, only the

additional sample rotation β has to be applied. It is the angle β that replaces a3, and for gfc=0, $\beta = \mathrm{a3}$. One has:

$$\beta = \psi - \psi_0 = \psi + \alpha'_{31} - 90^\circ \tag{3.18}$$

The angle α' is calculated as

$$\cos\alpha' = \frac{\cos\delta\cos\gamma\cos\nu - \sin\delta\sin\nu}{\sqrt{1 - \left(\frac{Q^\perp}{k_\mathrm{i}}\right)^2}} \tag{3.19}$$

If calculated for one channel (e. g. 31), the others are obtained by adding multiples of 2.5°.

Remark. With the help of α', one may also choose the exact a4 that is necessary to cover the gaps between the channels (for instance a step of exactly 1.25°) - note that both a4 and gfc need to be moved in order to keep $Q^\perp$ constant (3.12). If the desired step in α' is x, it follows for the new a4 ($= \tilde{\delta} + 52.5^\circ$):

$$\sin\tilde{\delta} = \sin x\cos\delta\cos\gamma + \cos x\sin\delta \tag{3.20}$$

Furthermore, this change of α' (in particular α'_{31}) changes the reference for the calculation of β. Therefore, to produce a regular grid upon combination of multiple psi-scans with interleaved detector angles, the scan range of psi has to be adjusted according the changes of α'_{31} given by equation (3.20).