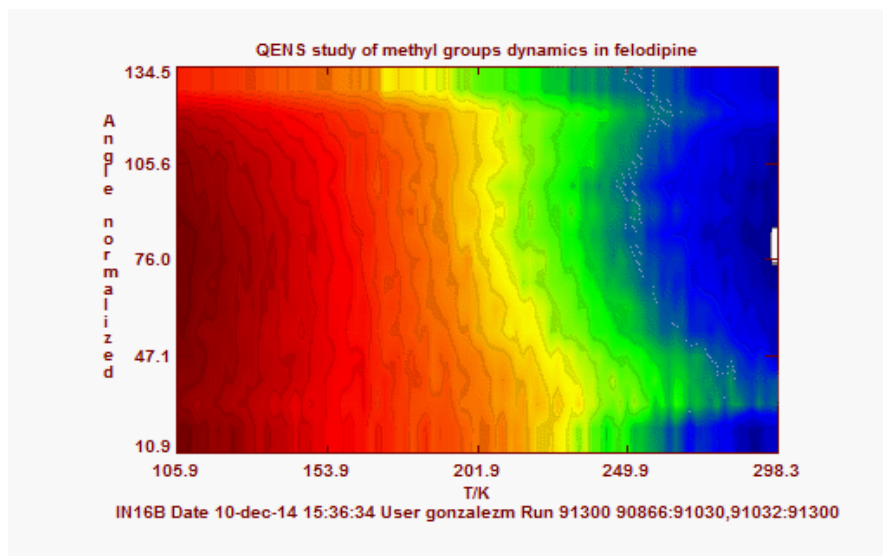
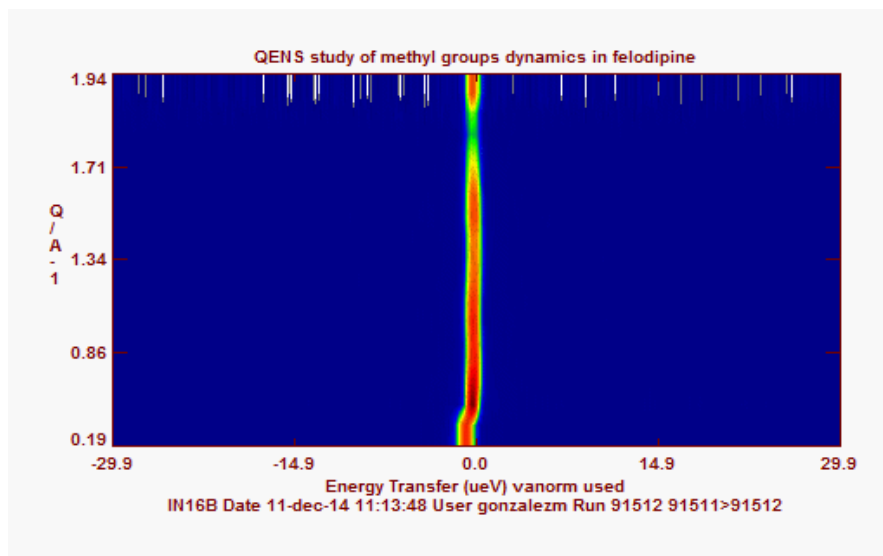


A user guide to IN16B data treatment using Lamp



Miguel A. Gonzalez

CS-DS, Institut Laue-Langevin

(May 2016)

1. Introduction

This short guide shows how to use Lamp to read and visualize the different type of data that can be acquired in IN16B and the steps needed to perform a standard data treatment. Some familiarity with Lamp is assumed. New users can get information on Lamp through the 'Help' available in the Lamp menu or consulting the Lamp book [1].

The instrument configuration of IN16B is flexible and can be modified in order to adapt to the experimental needs. Additionally, different acquisitions modes are available based on the state of the Doppler machine: stopped to measure only elastically scattered neutrons (EFWS), moving at a determined frequency and following a sinusoidal velocity profile to obtain a quasielastic neutron scattering spectrum measured up to a given maximum energy transfer $\pm\hbar\omega$ (QENS), or moving with a velocity energy profile designed to maximize the time spent at a velocity corresponding to a particular energy transfer (IFWS). In the following, the most usual instrument configuration is assumed, implying that we have a position sensitive detector (PSD) composed of 16 tubes vertically sensible (with 128 pixels) and several (usually 2) single detectors (SD). And we may want to read data corresponding to any of the three different types of scan mentioned above: quasielastic neutron scattering (QENS), elastic fixed window scan (EFWS), or inelastic fixed window scan (IFWS).

It should be mentioned also that while a numor corresponding to a QENS scan can be considered as an independent set of data and treated on its own, a single numor of an EFWS/IFWS should normally be considered part of a scan. Therefore here we assume that a series of numors is read (containing the elastic or inelastic intensity measured as a function of other parameter, typically the sample temperature) and that the operations will be performed in the full series and not in independent numors. One should also note that quite often during a (temperature) scan elastic/inelastic acquisitions can be mixed.

The user can of course read a single EFWS/IFWS numor, but in this case he should be aware that the dimensionality of the workspace is different, so some commands may need to be changed or may behave unexpectedly.

Finally, although NOMAD (the control software in IN16B) wrote both ASCII and NeXus files until 2014 included, the latter are preferred and for IN16B Lamp only reads the NeXus files. However ASCII files can still be read by selecting IN16 instead of IN16B as instrument.

2. Basic commands

These are the recommended minimum set of commands needed to read IN16B data into Lamp. they should work correctly in most cases, but as IN16B data can be collected in many different ways (QENS-FWS, elastic-inelastic, mirror sense or not), errors could happen if some options set for a given case remain active when reading a different type of data. All the possible options are explained in detail later in the manual and the user can use the command “rdset, /show” to check the active options at any time¹.

Starting Lamp from a terminal

```
> /home/cs/lamp/START_lamp
```

Elastic scans (basic manipulation using Lamp commands)

```
Rdset, /noraw, fws=1, bsnorm=0          ; To set adequate reading options for elastic scans
W1 = rdopr('1st_numor : last_numor')    ; Or read using File box
W1 = bsnorm(w1)                          ; To normalize by average monitor2
W2 = total(W1, 1)                        ; To sum together all the detectors and get Itotal vs T
```

QENS scans (basic manipulation using Lamp commands)

```
Rdset, /noraw, fws=0, bsnorm=1, unmirror=6 ; Good options for quasielastic scans
W1 = rdopr('1st_numor>last_numor')        ; Or read using File box
W1 = bsnorm(w1)                             ; Normalize (account for mirror and number of runs)3
W1 = tee(w1)                                ; X-axis → energy (microeV)
W2 = total(w1, 1)                           ; Sum over energy → I vs 2θ
W3 = total(w1, 2)                           ; Sum over detectors → I vs ω
```

Fixed Window scans (elastic + inelastic)

```
Rdset, /noraw, fws=1, bsnorm=0, unmirror=0 ; To set adequate reading options for elastic scans
W1 = rdopr('1st_numor : last_numor')        ; Or read using File box
W1 = bsnorm(w1)                             ; To normalize by average monitor
W2 = in16b_fws(w1)                          ; Separate different energies
W3 = w2[* , 0, *] & x3 = x2[* , 0]           ; Get 1st energy (typically elastic)
W4 = w2[* , 1, *] & x4 = x2[* , 1]           ; Get 2nd energy
W5 = w2[* , 2, *] & x4 = x2[* , 2]           ; Get 3rd energy, etc.
```

¹ The present options will be shown in the Journal (Lamp → Info → Journal of current session).

² Note that each temperature is already normalized by its monitor by the concatenation routine, and then all of them are multiplied by the average monitor value, which is returned as a single value in N1. In this case, the bsnorm command will simply divide all the detector intensities by this value.

³ The option bsnorm=1 normalizes each channel by its corresponding channel in the monitor before summing sides (unmirror), but then both sides and several numors can be summed, so a further normalization after rdopr is needed.

3. Raw data

After starting Lamp and selecting the instrument, the working path and the data path, by default the 'raw' tick box is selected (Figure 1). In this case, when the user reads a numor the data are loaded into the corresponding workspace without any manipulation, as shown in Fig. 1. It should be noted that only the data corresponding to the PSD are loaded and that we will have a 3D workspace of dimensions e.g. [1024, 128, 16]. Here 1024 is the number of energy channels (it could be a different number depending on the settings used in NOMAD), 128 the number of pixels of each detector tube in the vertical direction, and 16 the number of tubes in the PSD.

In the 'raw' mode, PSD and SD cannot be read or shown together due to the incompatibility in the number of dimensions. However, if you need to check the SD you can force Lamp to read them by using the command:

> RDSET, /singled or RDSET, singled=1

In this case, Lamp will read the SD data into a 2D workspace of dimensions [1024, 8], where 8 is the number of 'slots' allocated for the single detectors in the NeXuS file. However not all of them will contain data (in most cases, only 2 single detectors are connected) and the SD can be connected to any of the available entries (i.e. if 2 SD are used, they do not necessarily appear in the first two rows).

In order to go back to the default state of reading the PSD, use the following command:

> RDSET, singled=0

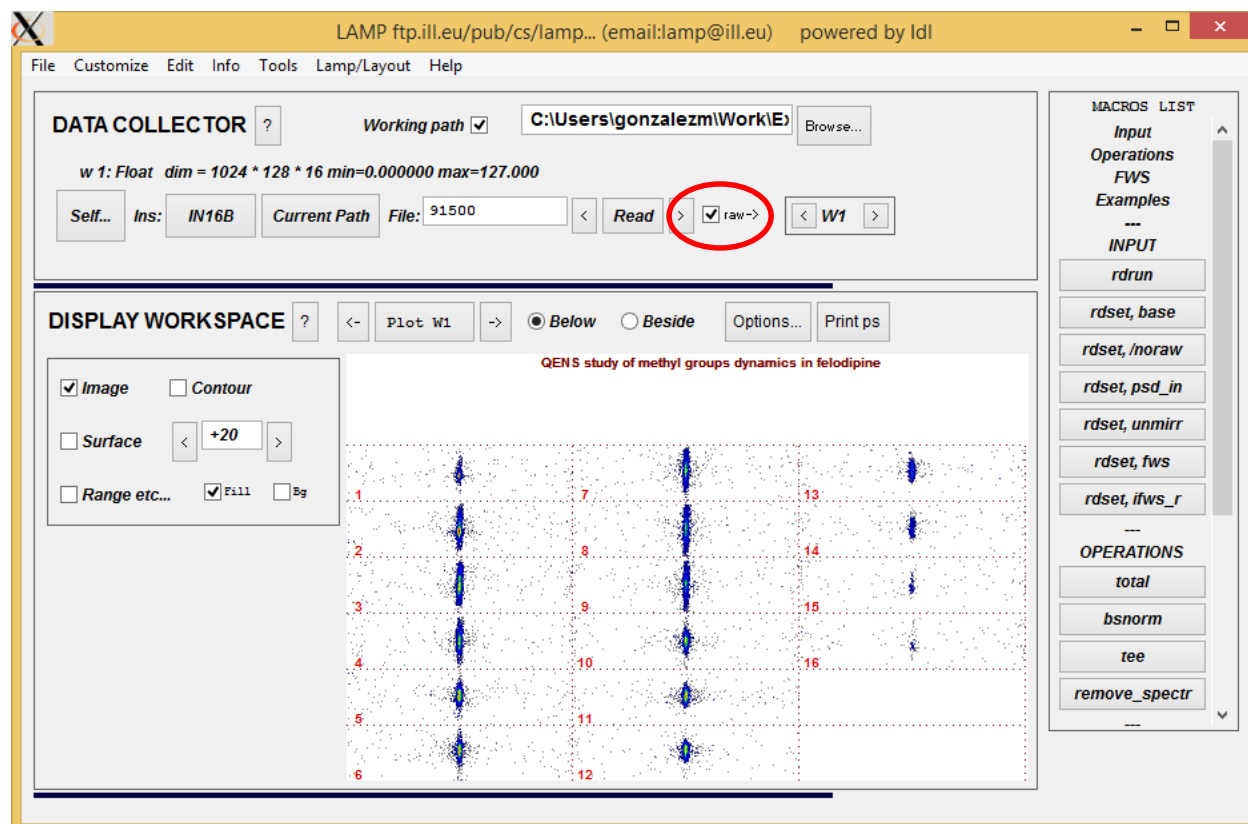


Figure 1: Initial Lamp interface, showing the 'raw' tick box selected and a raw data set read into w1.

At present (May 2016), EFWS and IFWS numors are saved in a similar way to QENS numors, so reading one of them will also result in a 3D workspace of dimensions [number of energy channels, number of pixels in a detector tube = 128, number of detector tubes = 16]. The main difference is that in the case of a EFWS numor all the detector counts will appear in a single central energy channel corresponding to $\hbar\omega = 0$, while for a IFWS all the channels may show some counts, but there will be two peaks in the first and last channels corresponding to $\hbar\omega = \pm E_{\text{fix}}$.

Normally the raw data are only needed for checking purposes, so in most occasions only a single numor is going to be read. However it is possible to read more than a single numor using the standard operators, e.g.:

1 > 10 to read and sum all the numors between 1 and 10

1 + 7 + 10 to read and sum numors 1, 7 and 10

1 + 7>10 to read and sum numors 1, 7, 8, 9 and 10

1:10 to concatenate numors 1 to 10

But it should be noted that as the 'single numor' workspaces are already 3D, the concatenation procedure will not add an extra dimension, but consecutive numors will be added in the 3rd dimension. Thus reading 10 files similar to the one shown in Fig. 1 will result in a workspace of dimensions [1024, 128, 160].

3. 'No Raw' data

In most cases the information along the vertical direction of the detector tubes is not needed, and the data can be summed up over the vertical coordinate. Furthermore, in the case of EFWS and IFWS there is not useful energy information, so an additional sum over the X (energy) axis can be done. Those sums are automatically done during the reading procedure whenever the 'raw' button is not ticked. Alternatively, the command

> RDSET, /noraw

produces the same effect (while RDSET, /raw will restore the 'raw' reading mode).

The reading routine (read_hdf.pro) will also check which SD are used (determined from the parameters P[44] to P[51], named tubes1_function, ..., tubes8_function in the NeXus file) and add those to the spectrum.

Care! It is important to note that when starting a new Lamp session the default option values are loaded. They are singled = 0, fws = 0, unmirror = 6, ifws = [0, 0, 0, 0], psd_range = [0, 0], bsnorm=1. However when any of those values is changed, it will remain active during the rest of the session, so when changing from one data mode to another (e.g. from fws=0 to fws=1) the user should take care of setting appropriate values for the important options related to the new mode.

The operations performed in the raw data can be controlled through the RDSET command. Here is a list of the available commands for IN16B:

> ***RDSET, /raw*** or ***RDSET, /noraw***

Choose between 'raw' or 'no raw' reading.

> ***RDSET, singled=1/0***

To read the SD (1) or the PSD (0) data. Only in 'raw' mode. The default value is 0.

The following commands have an effect only when the 'no raw' mode has been selected:

> ***RDSET, fws=1/0***

fws=1 indicates that the numors to read correspond to a fixed window scan (either elastic or inelastic). Fws=0 (the default) indicates that the numors to read are QENS data.

> ***RDSET, psd_int_range = [10,110]***

Defines the lower and upper detector pixel to use in the sum. This is useful to discard the noisier bottom and the top part of the detector. If not given, then the full detector tube is used.

> ***RDSET, unmirror = 0/1/2/3/4/5/6/7***

Allows to determine how to use the data when the numor file contains separately the points measured while the Doppler is accelerating and while is decelerating. Thus this command will only have an effect if P[52] = 0 or 14 (mirror_sense in the NeXus file). The option **unmirror = 0** means that nothing is done to the data, so the workspace where the data are read will have 2 elastic peaks. **Unmirror = 1** will sum both sides, while **unmirror = 2** will select the left side, and **unmirror = 3** will select the right side.

The other options involve some additional manipulation to the data. **Unmirror = 4** will fit the position of the elastic peak in both sides for each detector and use these fitted values to position the right peak in top of the left one before summing. This ensures that the elastic peak is not broadened by any mismatch in the channel positions (which is something that has been observed to happen occasionally).

Unmirror = 6 fits also both peak positions, but in this case forces both of them to be in the middle channel. By doing this, the elastic peak is forced to be at zero energy transfer. This is in a certain way equivalent to the *lineup* option for TOF instruments and is the default option for unmirror.

A possible problem with the last 2 options is that the code tries to fit a Gaussian to the elastic peak of each detector. While in most cases this works fine, it can be difficult to identify a correct peak position when the quasielastic signal is too broad or there is a defective detector. As normally the peak position should be close to the mid-channel of each mirror sense, the code will print a warning message in the Journal whenever the peak position found from the fit is shifted from more than 10 channels from this position.

When the peak positions cannot be defined with good precision due to the absence of a well defined elastic peak, it is possible to use a reference sample (e.g. vanadium or the same sample at a lower temperature) to determine the peak positions. In this case the user has to employ the command 'rdset, /vana' (see below) before reading the reference data. The peak positions found in this set of data will be then stored in memory and used to shift the spectra whenever the options unmirror = 5 or 7 are requested. **Unmirror = 5** is equivalent to unmirror = 4, while **unmirror = 7** is equivalent to unmirror = 6, but in both cases using the stored peak positions instead of fitting them to the current set of data.

***Care!** It should be noted that any of the unmirror > 3 options performs a circular shift of at least one side of the data. The monitor needs to be shifted as well, but as the number of channels shifted is not necessarily the same for all the detectors, this is done by using the median of the peak positions found in all the detectors. For the moment, the energy channels shifted to the opposite side of the elastic peak (due to the circular shift) are not set to zero, but their error bars are multiplied by a factor of 10.*

> RDSET, /vana or rdset, vana = 1/0

This option tells the reading routine that the next numor to read can be used as a reference to fit the positions of the elastic peak in a set of data measured with the mirror sense option. These positions will then be stored in memory and used to 'unmirror' the data whenever the options unmirror = 5 or 7 are requested.

In order to forget oversights, the keyword is reset to zero (its default value) after reading the first set of data. The actual positions kept in memory can be seen using the 'rdset, /show' command.

> RDSET, bsnorm = 0/1/2

This option affects only the reading of QENS data. The default value is **bsnorm = 0** and in this case nothing is done to the data and the reading routine will return a workspace W containing the neutron counts (not normalized) for each energy channel and detector and its associated monitor N containing the monitor counts registered in each energy channel. The standard Lamp command *bsnorm* (explained later) can then be used to normalize the data to the incident monitor. This will work perfectly in combination with unmirror = 0-3, but not necessarily for the other options implying some shift of the data, as different detectors may need different shifts (see above the care note about the unmirror). In this case the options bsnorm = 1 or 2 can be employed to normalize the detector counts by the monitor before doing any shift. **Bsnorm = 1** is equivalent to the standard *bsnorm* command and for each detector divides channel by channel the neutron counts by the monitor counts in the corresponding channel. **Bsnorm = 2** does the same operation, but it multiplies by the median value of the monitor in order to return a number of counts comparable to the original neutron counts. After using bsnorm = 1, all the values of the monitor vector N are set equal to 1, where with bsnorm = 2 they were set equal to M (where M is the median of the monitor). However the 'unmirror' operation will sum both sides (for the monitor as well as for the detectors), giving then N = 2 or 2M. And finally if we sum S numors together, the monitor associated to the workspace will be a vector containing a value equal to 2S (bsnorm = 1) or 2MS (bsnorm=2) in each of the energy channels. It is therefore necessary (or highly

recommended) to use again the *bsnorm* command after reading the data, even when the *bsnorm* option is used in *rdset*.

> *RDSET, ifws_range = [0,10,1010,1024]*

This command has only an effect if *fws*=1 has been set and an IFWS number is read. It allows to determine which are the energy channels that will be integrated to give the total measured inelastic intensity for each detector. The user must provide a vector with 4 numbers. The first pair gives the first and last channel to use on one side of the spectrum and the second pair the first and last channel to use on the other side. The default behavior for IFWS (when the *ifws_range* has not been declared and the four values are equal to 0) is to sum the first and the last 10 channels. This will also be the case when the last value (*ifws_range*[3]) is larger than the number of channels⁴.

> *RDSET, /show*

Shows in the journal the current values of the above variables.

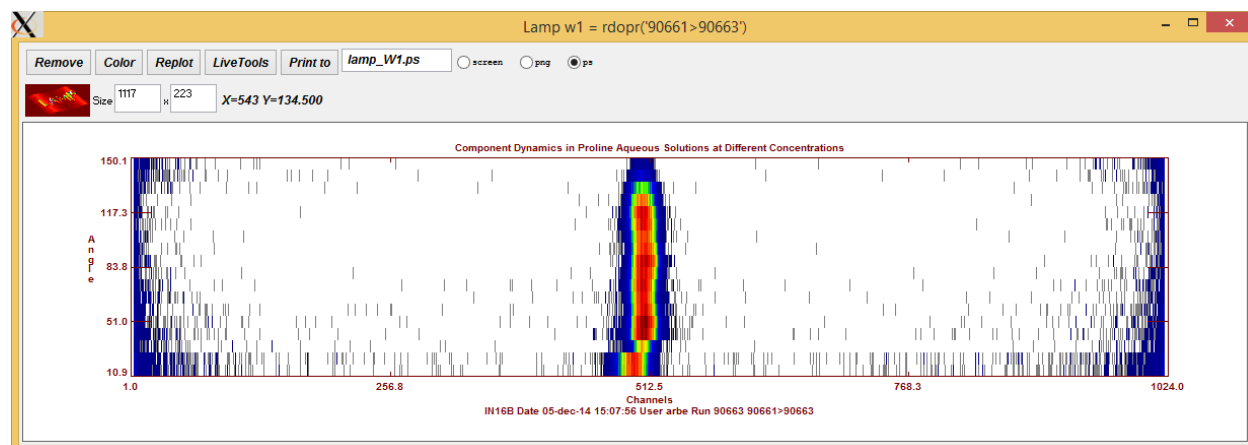
⁴ This can happen e.g. if the variable *ifws_range* has been set to read an IFWS measured using 2048 channels and later we tried to read another IFWS measured using less channels.

Here there are some examples of how to read different types of data and the output obtained in each case:

QENS

```
> RDSET, /noraw, fws=0, psd_int_range=[20,107]
```

```
> w1 = rdopr('90661>90663')
```



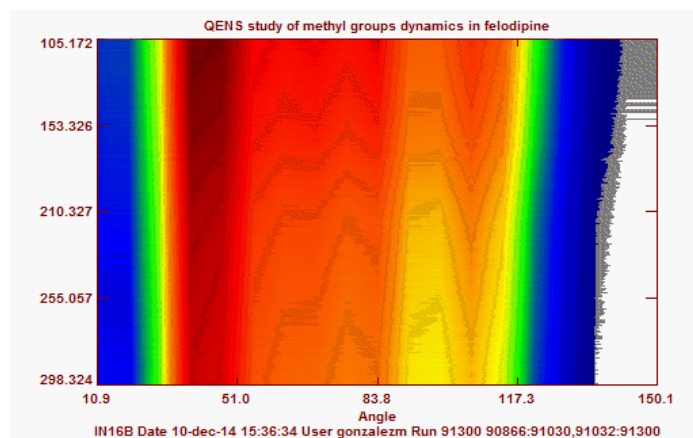
The output is a 2D workspace of dimensions [1024 x 18]. The X axis corresponds to the energy channels (1 to 1024 in this case) and the Y axis to the scattering angles corresponding to the two SD used in this measurement and the 16 PSD tubes.

Elastic scans

```
> RDSET, /noraw, fws=1, psd_int_range=[2,122]
```

```
> w2 = rdopr('90866:91030, 91032:91300')
```

The user should note that each numor read contains a large array of data ($N_{\text{TOF}} \times 128 \times 16$), so if the scan contains many numors this operation could take some time. Note also that it may take some time to write the last acquired files to the database.



The output workspace is again a 2D array, but now of dimensions [18, N] where N is the number of numors concatenated by the rdopr command.

As in most cases, the elastic intensity is measured as a function of the temperature, the reading routine will read the temperature from each file (P[9], (sample) temperature in the NeXus file) and will return the temperature values on the Y axis. If the user needs to change the values of this axis, he can replace them by any other parameter stored in the corresponding PV array (in this example, PV2 for w2). For example, typing

```
y2 = pv2[0,*]
```

will replace the temperature by the file number. The axis title can also be changed using

```
y_tit[2] = 'Numor'
```

where the index [2] refers to workspace w2.

The PV array has dimensions [N_{params}, N_{numors}] and the list of N_{params} instrument parameters can be consulted from the Info → Data parameters menu.

If it is convenient to present the temperature in the abscise axis, the transpose command can be used:

```
> w2 = transpose(w2)
```

This is for example needed to be able to work with all the existing 'elascan' commands, originally written to treat elastic fixed window scans measured on IN10, IN13 and IN16 and that required the temperature to be on the X axis.

Inelastic fixed window scans

IFWS are often measured as part of a series where acquisitions with the Doppler stopped and at a different velocities (corresponding to $|\hbar\omega| = 0, E_1, E_2, E_3$, etc) are regularly repeated.

For example, the run number 83072 to 83340 have been measured during heating alternating one elastic acquisition and two inelastic acquisitions (using 256 energy channels) at $|\hbar\omega| = 1.5$ and $3.0 \mu\text{eV}$, respectively. It is possible to read the three data sets into three different workspaces using the following commands:

```
> RDSET, /noraw, fws=1, psd_int_range=[2,122], ifws_range = [0,5,1020,1023]
```

```
> w3 = rdopr('83072:::83340') ;elastic scan
```

```
> w4 = rdopr('83073:::83340') ;inelastic 1.5 microeV
```

```
> w5 = rdopr('83074:::83340') ;inelastic 3.0 microeV
```

Repeating the : sign implies that we will read one out of two numors (::), one out of three (:::), etc. An easy way to check that different types of data have been mixed is to print the minimum and maximum values of the parameter P[14] (named maximum_delta_energy in the NeXus file). In this example, we have

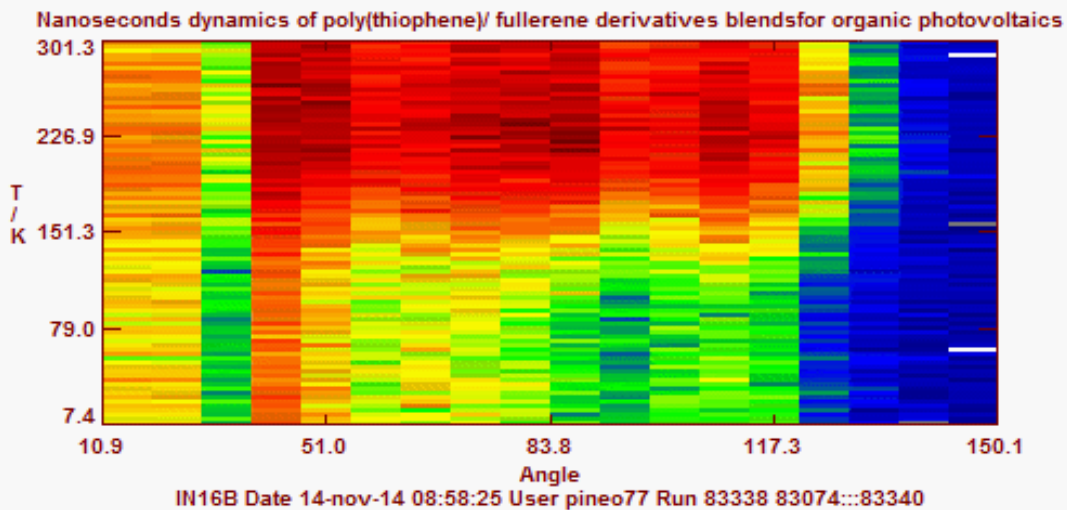
```
> show, min(pv3[14,*]) = 0.0
```

```

> max(pv3[14,*]) = 0.0
> show, min(pv4[14,*]) = 1.5
> show, max(pv4[14,*]) = 1.5
> show, min(pv5[14,*]) = 3.0
> show, max(pv5[14,*]) = 3.0

```

confirming that the three types of data have been correctly read into w3, w4, and w5.



As for the EFWS, each of those workspaces is a 2D array of dimensions [18,N] and the X axis contains the scattering angle of the detectors (SD and PSD) and the Y axis the temperature of the measurement. Again, the transpose command allows to show the temperature on the X axis.

EFWS+IFWS

If the full sequence of EFWS/IFWS has not been obtained in a regular manner, the previous procedure cannot be applied. In this case, in order to get the data the following commands can be applied:

```

> RDSET, /noraw, fws=1, psd_int_range=[20,107], ifws_range = [0,5,250,255]
> w6 = rdopr('83072:83340') ;mixed elastic/inelastic scans
> w7 = in16b_fws(w6)
> w8 = w7[* ,0,*] & x8 = x7[* ,0] ;elastic scan
> w9 = w7[* ,1,*] & x9 = x7[* ,1] ;inelastic scan at 1.5 microeV

```

```
> w10 = w7[* ,2,*] & x10 = x7[* ,2] ;inelastic scan at 3 microeV
```

```
> j =where(x8>0) & w11 = w8[j,*] & x11 = x8[j]
```

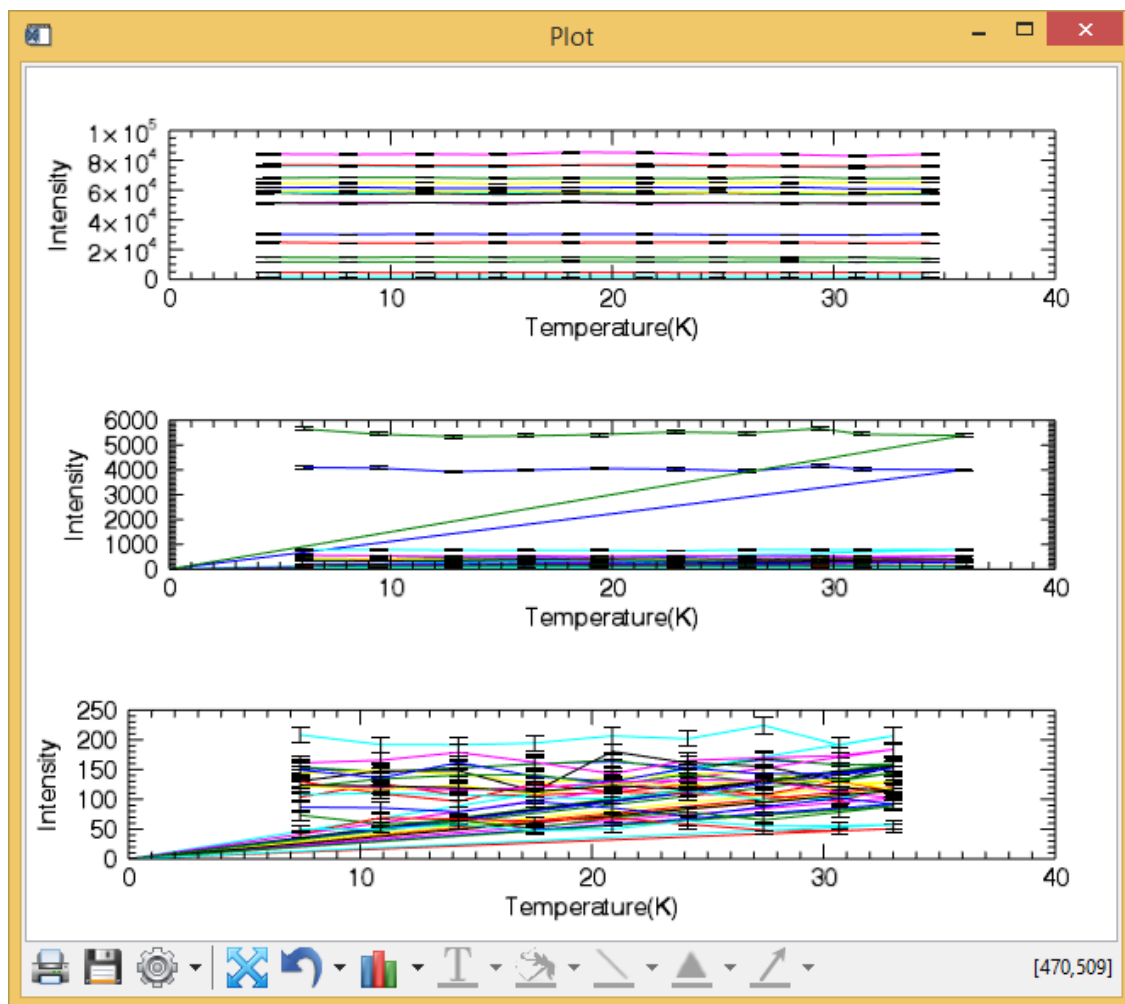
```
> j =where(x9>0) & w12 = w9[j,*] & x12 = x9[j]
```

```
> j =where(x10>0) & w13 = w10[j,*] & x13 = x10[j]
```

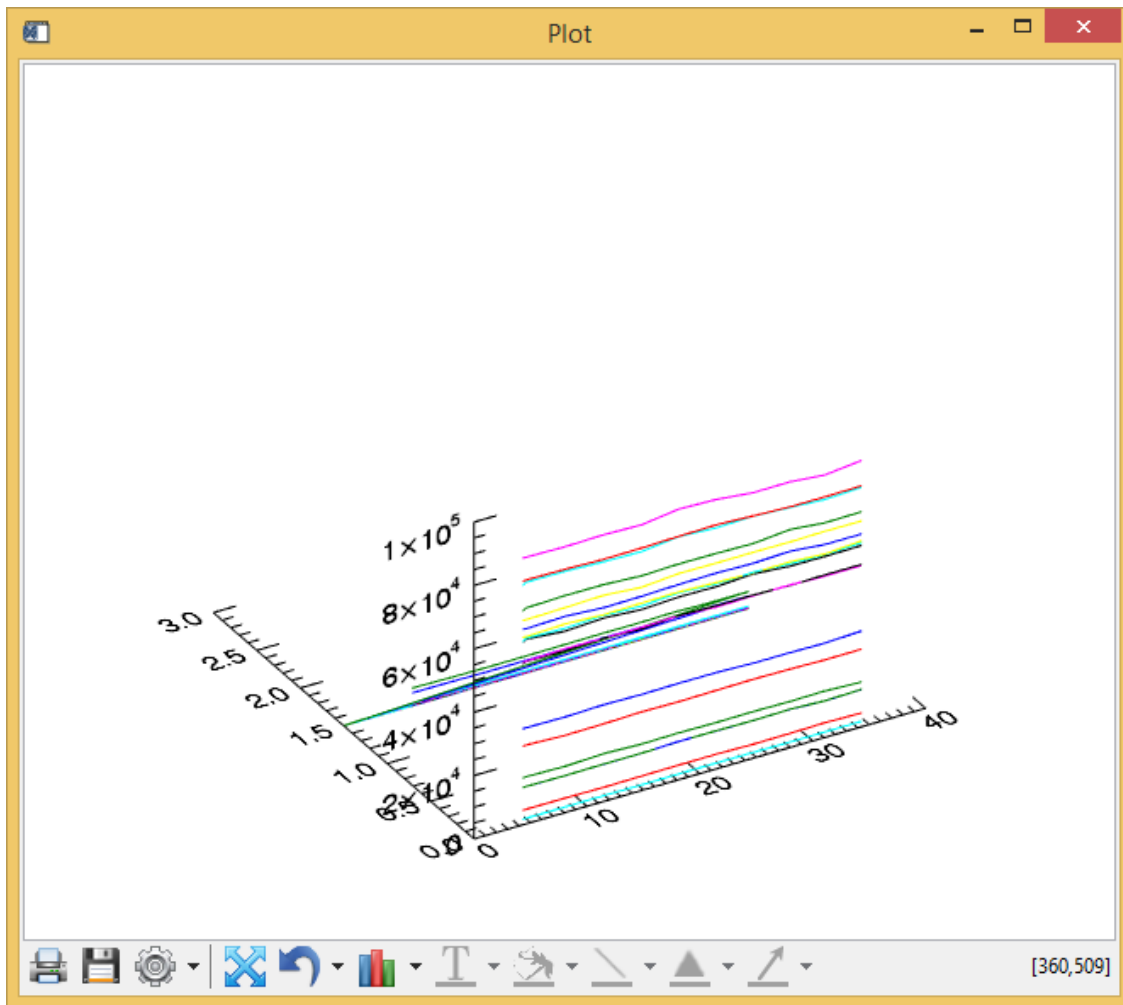
Here the `rdopr` command will read, sum (over the vertical direction of each tube and over the energy channels, using all the channels if the numor is an elastic one and the `ifws_range` if it is an inelastic one), and concatenate all the numors from 83072 to 83340.

The next command, `in16b_fws`, analyzes that workspace and creates a new workspace with an additional dimension in order to separate the numors having different energy transfers. In the present example, `w7` has dimensions [90, 3, 18] corresponding to the different number of temperatures, the three energy transfers measured (0, 1.5 and 3 μeV), and the angles of the 18 detectors. This command admits two additional keywords, `/plotcol` and `/plot3d`, that will generate automatically a plot of the data. E.g.:

```
> w7 = in16b_fws(w6, /plotcol)
```



```
> w7 = in16b_fws(w6, /plot3d)
```



Data corresponding to different energy transfers are stored along the 2nd dimension of w7, so w7[* ,0,*] selects the elastic scan, w7[* ,1,*] the first energy of the IFWS, etc. The X axis is also a 2D array, containing the temperatures of each numor in a similar way, x7[* ,0] for the elastic scan, x7[* ,1] for the first IFWS energy, and so on. In the case that the number of measurements for each energy is not the same, the first dimension corresponds to the maximum number of measurements. For energies having a fewer number of measurements, the array is filled with zeroes. The last 3 commands therefore serve to remove those zeroes, returning the final results in workspaces w11, w12, and w13.

4. Data reduction

QENS

The following script gives an example of a standard data treatment for a QENS spectrum:

```
; IN16B QENS measurement: visualization and data reduction
; Raw data are saved in w11 (sample), w12 (empty can), w13 (vanadium) and w14 ;(empty
can for vanadium)
; The corrected QENS measurement is saved in w30

; The macro:
; - reads and sums quasielastic measurements
; - normalises to the monitor and converts channels into energy scale
; - calculates self-shielding corrections (flat sample)
; - subtracts empty can applying calculated attenuation factors
; - normalizes to vanadium (but can also be adapted for low temp normalization)
; - converts angles to q

; The macro needs:
; - the correct path to read the data must have been set in the interface
; - the run numbers for sample, empty cell, vanadium, and empty cell for vana
; - sample thickness and scattering and absorption cross sections
; - thickness of cell walls and scattering and absorption cross sections
; - angle at which the slab has been oriented
; - vanadium thickness and scattering and absorption cross sections
; - fraction of neutrons passing twice through the sample/vanadium
; - number of channels covered by the elastic peak
; - incident neutron wavelength

; fill in the following lines, save the macro in your own directory
; (giveaname.prox), and launch it from the lamp interface: @giveaname

a = '91511>91512'      ; sample numors
b = '90658>90660'      ; empty numors
c = '90661>90663'      ; vanadium numors
d = '90658>90660'      ; empty numor for vanadium

e = 0.02      ; sample thickness (cm)
f = 5.625     ; scattering cross section of sample (cm-1) (5.625 for H2O)
g = 0.022     ; absorption cross section of sample (cm-1) (0.022 for H2O)
h = 0.05      ; thickness of container walls (cm)
i = 0.091     ; scattering cross section of container (cm-1) (0.091 for Al)
j = 0.014     ; absorption cross section of container (cm-1) (0.014 for HA12O)

k = 135.0     ; sample orientation (degrees)

l = 0.1        ; vanadium thickness
m = 0.359      ; scattering cross section of vanadium (cm-1)
n = 0.358      ; absorption cross section of vanadium (cm-1)

; assuming same type of empty cell for vanadium and sample

o = 0.5        ; fraction of neutrons passing a second time trough the sample
p = 0.5        ; fraction of neutrons passing a second time trough the vanadium

q = 120        ; number of channels to use around elastic peak to normalise to vanadium
```

```

t = 6.27 ; incident neutron wavelength (A)
u = 81.807/t^2 ; incident neutron energy (meV)

;
;

; remark : for data treatments with normalisation to the the lowest
; temperature run, simply change the vanadium numors by the sample
; lowest T numors.

;*****
; Settings (can also be specified directly in the lamp interface)
;*****

RDSET, /noraw, fws=0, psd_int_range=[20,107]

;*****
; Read data
;*****

; Sample

w1 = rdopr(a)           ; read single run or sums numors
w2 = bsnorm(w1)         ; normalisation to the monitor
w3 = tee(w2, /dead)     ; channels to energy scale
w11 = w3 & see, w11

; Background (sample)
w1 = rdopr(b)           ; read single run or sums numors
w2 = bsnorm(w1)         ; normalisation to the monitor
w3 = tee(w2, /dead)     ; channels to energy scale
w12 = w3 & see, w12

; Vanadium
w1 = rdopr(c)           ; read single run or sums numors
w2 = bsnorm(w1)         ; normalisation to the monitor
w3 = tee(w2, /dead)     ; channels to energy scale
w13 = w3 & see, w13

; Background (vanadium)
w1 = rdopr(d)           ; read single run or sums numors
w2 = bsnorm(w1)         ; normalisation to the monitor
w3 = tee(w2, /dead)     ; channels to energy scale
w14 = w3 & see, w14

;*****
; Compute correction factors
;*****

;sample
w19 = calc_abscof_flat(y11, k, e, f, g, cellWallThickness = h, cellSigmaS = i, cellSigmaA
= j, E0 = u)

;vanadium
w20 = calc_abscof_flat(y13, k, l, m, n, cellWallThickness = h, cellSigmaS = i, cellSigmaA
= j, E0 = u)

;*****

```

```

; Subtract background (applying self-shielding corrections)
;*****

w21 = background_abscor_correct(w11, w12, w_corr = 19, fBS = o) ;sample

w22 = background_abscor_correct(w13, w14, w_corr = 20, fBS = p) ;vanadium

;Or subtract background without applying absorption corrections
;w21 = w11 - 0.9*w12
;w22 = w13 - 0.9*w14

;*****
; Normalisation to vanadium
;*****

r = n_elements(x21)/2 - q/2
s = n_elements(x21)/2 + q/2
w30 = vanorm(w21, w22, r, s) & see, w30

;*****
; Transform Y axis to Q
;*****

y30 = (4*!pi/t)*sin(y30*!dtor/2.0) & y_tit[30] = 'Q/A-1'

```

The script uses the following commands:

Rdopr: Interprets the string given as a parameter and reads the corresponding numors. It is equivalent to write the numor files in the main Lamp window. The data are stored in the workspace $w\#$ and the error is set in the array $e\#$ as $\text{SQRT}(w\#)$.

Bsnorm: Normalizes the data in the workspace $w[\text{channel}, \text{angle}]$ by the corresponding monitor vector $n[\text{channel}]$:

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{in}[\hbar\omega, 2\theta]}{n_{in}[\hbar\omega]}; \quad n_{out}[\hbar\omega] = 1;$$

$$e_{out}[\hbar\omega, 2\theta] = \sqrt{\left(\frac{e_{in}[\hbar\omega,]}{n_{in}[\hbar\omega]}\right)^2 + \left(\frac{w_{in}[\hbar\omega, 2\theta]\sqrt{n_{in}[\hbar\omega]}}{n_{in}[\hbar\omega]^2}\right)^2}$$

Tee: Transforms the X axis from energy channels to energy transfers (given in microeV). The energy step is constant and calculated as $\Delta E = \frac{2E_{max}}{N_{channels}-1}$, where E_{max} is the maximum energy transfer read from the NeXus file, so the energy axis will contain $N_{channels}$ energy values regularly spaced and going from $-E_{max}$ to $+E_{max}$. If the /dead keyword is employed, then $\Delta E = \frac{2E_{max}}{N_{last}-N_{first}-1}$, where N_{first} and N_{last} are the first and last good channels. The later are determined during the reading procedure (but only in the 'no raw' mode) as the first and last channels having a number of monitor counts larger than 10% of the average monitor counts per channel and stored in the workspace parameters P[54] and P[54]. In this case, the energy axis

will still contain N_{channels} energy values regularly spaced but the limits will be $> E_{\text{max}}$, as in this case the energy is set using $E(N_{\text{first}}) = -E_{\text{max}}$ and $E(N_{\text{last}}) = +E_{\text{max}}$.

This routine also works with 'unmirrored' data.

Calc_absor_flat: Computes the attenuation factors for an infinite slab. Details on the input required by this routine and the operations performed are given in next section.

Background_absor_correct: Takes two workspaces and subtracts the second one (background) from the first one (sample) using the correction factors previously stored in another workspace, whose number is passed to the routine by the keyword `w_corr`.

A typical call to this routine would be:

```
> w4 = background_absor_correct(w1, w2, w_corr=3),
```

where `w1` is the workspace containing the sample data, `w2` contains the empty cell, and `w3` is a workspace containing the self-shielding factor (SSF) and the self-attenuation factor (SAF) (see next section for more details). The result with the corrected data will be returned in workspace `w4` as:

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{sample}[\hbar\omega, 2\theta] - SAF(2\theta) \times w_{bgr}[\hbar\omega, 2\theta]}{SSF(2\theta)}$$

Errors are propagated as:

$$e_{out}[\hbar\omega, 2\theta] = \frac{\sqrt{(e_{sample}[\hbar\omega, 2\theta])^2 + (SAF(2\theta) \times w_{bgr}[\hbar\omega, 2\theta])^2}}{SSF(2\theta)}$$

It is also possible to use the keyword `fBS` to pass to the routine the value corresponding to the fraction of neutrons that go through the sample a second time in their flight to the detectors after having being backscattered by the analyzers. E.g.:

```
> w4 = background_absor_correct(w1, w2, w_corr=3, fBS = 0.5)
```

In this case, the intensities returned in `w4` are corrected by the extra attenuation due to this second passage as:

$$w_{out}[\hbar\omega, 2\theta] = \frac{\{w_{sample}[\hbar\omega, 2\theta] - SAF(2\theta) \times w_{bgr}[\hbar\omega, 2\theta]\}/SSF(2\theta)}{1 - f_{BS} + f_{BS} \times T}$$

where T is the sample transmission. The latter can be passed as a parameter using the keyword `transm`, e.g. `> w4 = background_absor_correct(w1, w2, w_corr=3, fBS = 0.5, transm=0.85)`, or otherwise it will be approximated as $T \approx SSF(2\theta_{min})$, i.e. the value of the self-shielding factor for the angle closest to zero. The user should be aware that this and the fact that only the transmission $T_{\perp}(2\theta=0^\circ)$ is used (so different paths in the second passage are not taken into account) represent clearly a crude approximation.

If the background is missing only the SSF and/or second passage attenuation corrections are applied. For example:

```
> w4 = background_absor_correct(w1, w_corr=3)
```

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{sample}[\hbar\omega, 2\theta]}{SSF(2\theta)}$$

> w4 = background_absor_correct(w1, w_corr=3, fBS = 0.5, transm=0.9)

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{sample}[\hbar\omega, 2\theta]/SSF(2\theta)}{1 - 0.5 + 0.5 * 0.9} = \frac{w_{sample}[\hbar\omega, 2\theta]}{0.95 \times SSF(2\theta)}$$

The routine can also be employed without giving a correction file, although in this case the same operation can be done with a simple command. For example:

> w4 = background_absor_correct(w1, fBS = 0.5, transm=0.9)

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{sample}[\hbar\omega, 2\theta]}{1 - 0.5 + 0.5 \times T}$$

(here the value of T must be given explicitly, as otherwise will be set to 1, because no SSF is given)

This is equivalent to w4 = w1/0.95 & e4 = e1/0.95.

> w4 = background_absor_correct(w1, w2, transm=0.9)

$$w_{out}[\hbar\omega, 2\theta] = w_{sample}[\hbar\omega, 2\theta] - T \times w_{bgr}[\hbar\omega, 2\theta]$$

This is equivalent to w4 = w1 - 0.9*w2 & e4 = sqrt(e1^2+(0.9*e2)^2)

> w4 = background_absor_correct(w1, w2, fBS = 0.5, transm=0.9)

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{sample}[\hbar\omega, 2\theta] - T \times w_{bgr}[\hbar\omega, 2\theta]}{1 - 0.5 + 0.5 \times T}$$

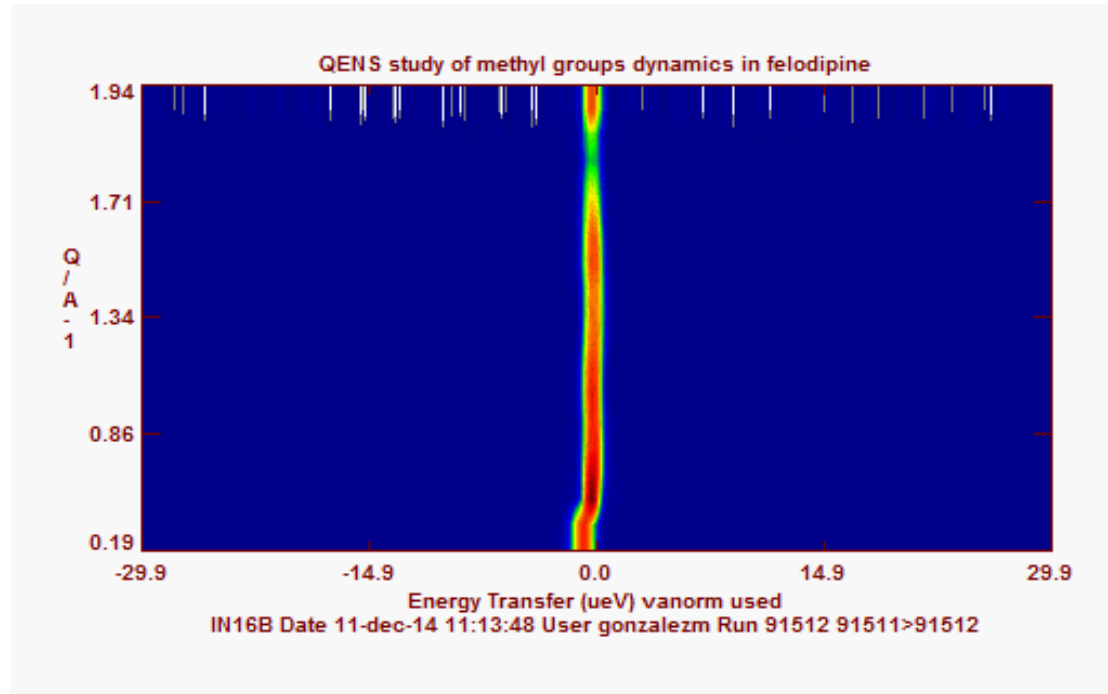
This is equivalent to w4 = (w1-0.9*w2)/0.95 & e4 = sqrt(e1^2+(0.9*e2)^2)/0.95

vanorm: Takes two workspaces and normalizes the first one (sample) to the second one (vanadium or sample at the lowest temperature) by dividing for each scattering angle by the sum over all the channels (or between the channels given in the call) of the normalization workspace:

$$w_{out}[\hbar\omega, 2\theta] = \frac{w_{in}[\hbar\omega, 2\theta]}{V[2\theta]} \times \max(V[2\theta]), \quad \text{with } V[2\theta] = \sum_{i=\text{channel } 1}^{\text{channel } 2} w_{vana}[i, 2\theta]$$

The same operation is done with the errors, but any additional contribution coming from the errors of the vanadium measurement is neglected (no error propagation here).

Finally, the last line of the script simply transforms the ordinate axis from scattering angle to wavevector using the elastic approximation (which is good enough for the small energy transfers of a backscattering spectrometer). The corrected data are then stored in w30, as $I[\hbar\omega, Q]$:



5. Calculating attenuation corrections

Usually the sample will be measured inside a container, such that the measured intensity is [2]:

$$I_{S+C}^{meas} = I_{S+C}^S + I_{S+C}^C$$

where the first term represents the scattering from the sample in the presence of the can, and the second the scattering from the sample holder in the presence of the sample. The latter can be written in terms of the real scattering intensity of the sample and container, respectively, as:

$$I_{S+C}^S = A_{S+C}^S I^S ; \quad I_{S+C}^C = A_{S+C}^C I^C$$

where A_{S+C}^S and A_{S+C}^C are attenuation coefficients to be determined to be determined for each scattering angle (and each final energy) from the sample/container geometry and cross sections.

The background is measured using the empty cell alone, giving:

$$I_C^{meas} = A_C^C I^C$$

And combining the previous equations we can extract the desired theoretical scattering intensity for the sample as:

$$I^S = \frac{1}{A_{S+C}^S} \left(I_{S+C}^C - \frac{A_{S+C}^C}{A_C^C} I_C^{meas} \right)$$

Or:

$$I^S = \frac{I_{S+C}^{meas} - SAF \times I_C^{meas}}{SSF}$$

with the "self-shielding factor" $SSF = A_{S+C}^S$ and the "self-attenuation factor" $SAF = \frac{A_{S+C}^C}{A_C^C}$.

There are two routines, *calc_abscor_flat* and *calc_abscor_cyl*, that will calculate those attenuation factors for the two standard geometries usually employed in IN16B: a flat container (infinite slab approximation) and a cylindrical sample (either massive or hollow cylinder).

It is useful to remind here that the calculated factors correct for the attenuation of the incident and output neutron beam due to scattering and absorption, but **not** for multiple scattering. A nice explanation on the dangers of over trusting self-absorption corrections is given in [3]. Finally, only two real scattering intensities are considered here, namely I^S and I^C . This implies that the present routines deal only with homogeneous samples and containers. For example, the case of a TiZr pressure cell containing an inner Al spacer to reduce the sample space cannot be rigorously treated (but the cases where the pressure cell is in Al or the spacer in TiZr can).

Both routines will return the attenuation factors into a workspace. However the dimensions of the output workspace depend on the input given to the routine:

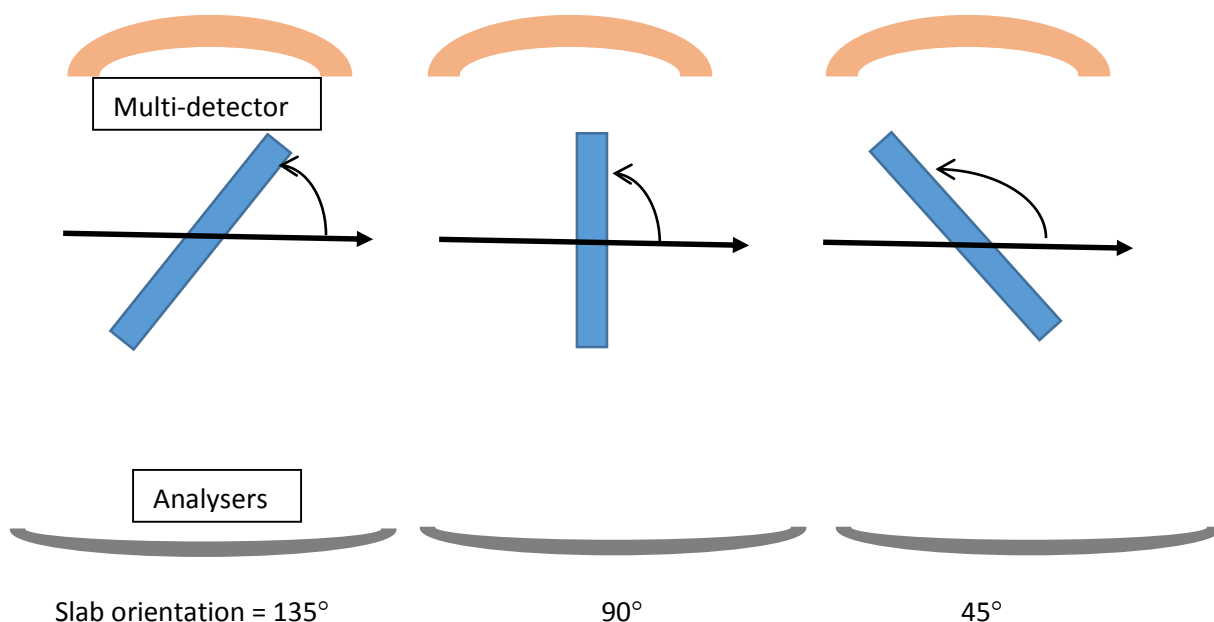
1. If there is no information about the container, then only SSF can be calculated, so the routine returns a vector (1D workspace) containing SSF as a function of the scattering angles given in the call.
2. The standard output is a 2D workspace of dimensions $[N_{\text{angles}}, 5]$, where N_{angles} is the number of scattering angles passed in the call to the routine. For each of those angles, the routine will return the values of SSF, SAF, A_{S+C}^S , A_{S+C}^C , and A_C^C .
3. Finally, if an energy vector containing the energy transfers is passed in the call to the routine, then the output will be a 3D workspace of dimensions $[N_{\text{angles}}, N_{\text{energies}}, 5]$. This is not really useful for IN16B (as the final energy is always close to the incident energy) or if the absorption cross section is negligible compared to the scattering cross section (as the latter is independent of energy), but it can be needed when analyzing TOF data. In this case, the returned workspace contains the values of $SSF(2\theta, \hbar\omega)$, $SAF(2\theta, \hbar\omega)$, $A_{S+C}^S(2\theta, \hbar\omega)$, $A_{S+C}^C(2\theta, \hbar\omega)$, and $A_C^C(2\theta, \hbar\omega)$.
4. Finally, if an energy vector is passed but no information about the container, the routine will still return a 3D workspace of dimensions $[N_{\text{angles}}, N_{\text{energies}}, 5]$. However only $w[N_{\text{angles}}, N_{\text{energies}}, 0]$ will contain useful information corresponding to $SSF(2\theta, \hbar\omega)$, while $w[N_{\text{angles}}, N_{\text{energies}}, 1:4]$ are set to zero.

Calc_abscor_flat:

This routine computes the attenuation coefficients for an infinite plane specimen of thickness d and enclosed between two plane walls of a container of thickness c using the equations given in [2] (chapter 4, eqns. 4.A.6 to 4.A.16).

It requires 5 compulsory arguments that must be given in the following order:

- A vector containing the scattering angles (in degrees) for which the attenuation factors will be calculated. They can be passed using a variable, e.g. `y1` if a QENS spectrum was read in the workspace `w1` or giving directly the desired vector as e.g. `[0, 10, 20, 30, ..., 180]`.
- The angle (in degrees) at which the sample is oriented with respect to the incident neutron beam (typically 45, 90, or 135°):



- The thickness of the sample. Typical units will be cm, but any unit could be used as long as the cross sections are given in the corresponding inverse units.
- The macroscopic scattering cross section Σ_s (typically in cm^{-1}), i.e.: $\Sigma_s = \frac{\rho N_a \sigma_s}{M}$, where ρ is the mass density (in g/cm^3), N_a is Avogadro's number, M the molecular weight (in g/mol) and σ_s is the microscopic scattering cross section (in cm^2).
- The macroscopic absorption cross section Σ_a (typically in cm^{-1}).

For example, the attenuation factor for a vanadium slab of 1 mm thickness placed in front of the beam could be calculated using the following command:

```
> a = findgen(37) * 5.0 ; generate angles between 0 and 180 degrees
```

```
> w11 = calc_abscor_flat(a, 90.0, 0.1, 0.359, 0.358)
```

Other parameters are optional and can be given in any order using the following keywords accepted by the routine:

cellWallThickness: Thickness of each of the walls of the sample container (in the same units employed for the sample, typically cm).

cellSigmaS: Macroscopic scattering cross section Σ_s of the sample holder (in the same units employed for the sample, typically cm^{-1}).

cellSigmaA: Macroscopic absorption cross section Σ_a of the sample holder (in the same units employed for the sample, typically cm^{-1}).

E0: Energy of the incident neutron beam (in meV). If given, the absorption cross sections (for sample and container) are scaled by $\sqrt{25.305/E_0}$.

w: A vector containing the energy transfers (in meV) for which to calculate the attenuation factors. These energies are used to calculate the corresponding energies in the output beam and scale the absorption cross sections accordingly. If w is given, but not E0, the routine assumes that the incident energy is 25.305 meV.

Calc absco cyl:

This routine computes the attenuation factors for a cylindrical sample (either massive or hollow) that can be confined between the internal or external walls of a container.

For this routine there are 4 compulsory arguments that again must be given in the right order:

- A vector containing the scattering angles (in degrees) for which the attenuation factors will be calculated.
- The external diameter of the sample. Typical units will be cm, but any unit could be used as long as the cross sections are given in the corresponding inverse units.
- The macroscopic scattering cross section Σ_s (typically in cm^{-1}).
- The macroscopic absorption cross section Σ_a (typically in cm^{-1}).

If no additional parameters are given, the routine considers that there is no sample container and that the sample is a massive cylinder having the diameter given in the call. For example, to calculate the self-shielding factor for a vanadium tube of diameter 8 mm:

```
> w11 = calc_absco_cyl(a, 0.8, 0.359, 0.358)
```

Additional parameters can be passed using the following keywords:

sampleThickness: Thickness of the sample annulus (in the same units employed for the sample external diameter, typically cm). This keyword allows the user to define a hollow geometry for the sample.

cellOutThickness: Thickness of the wall of the (external) sample container (in the same units employed for the sample external diameter, typically cm).

cellInThickness: Thickness of the wall of the internal sample container (in the same units employed for the sample external diameter, typically cm).

cellSigmaS: Macroscopic scattering cross section Σ_s of the sample holder (in the same units employed for the sample, typically cm^{-1}).

cellSigmaA: Macroscopic absorption cross section Σ_a of the sample holder (in the same units employed for the sample, typically cm^{-1}).

E0: Energy of the incident neutron beam (in meV).

w: A vector containing the energy transfers (in meV) for which to calculate the attenuation factors.

nr: Number of r points in each region (inner container, sample, outer container) employed to approximate numerically the required integrals (default is 50).

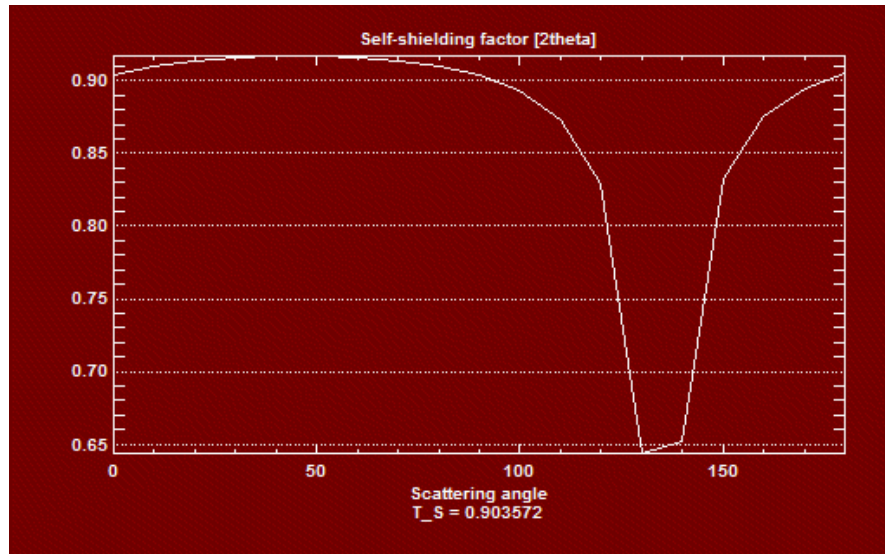
nomega: Number of ω points employed to approximate numerically the required integrals (default is 360).

Examples

In all the examples, $a = [0, 10, 20, \dots, 180]^\circ$

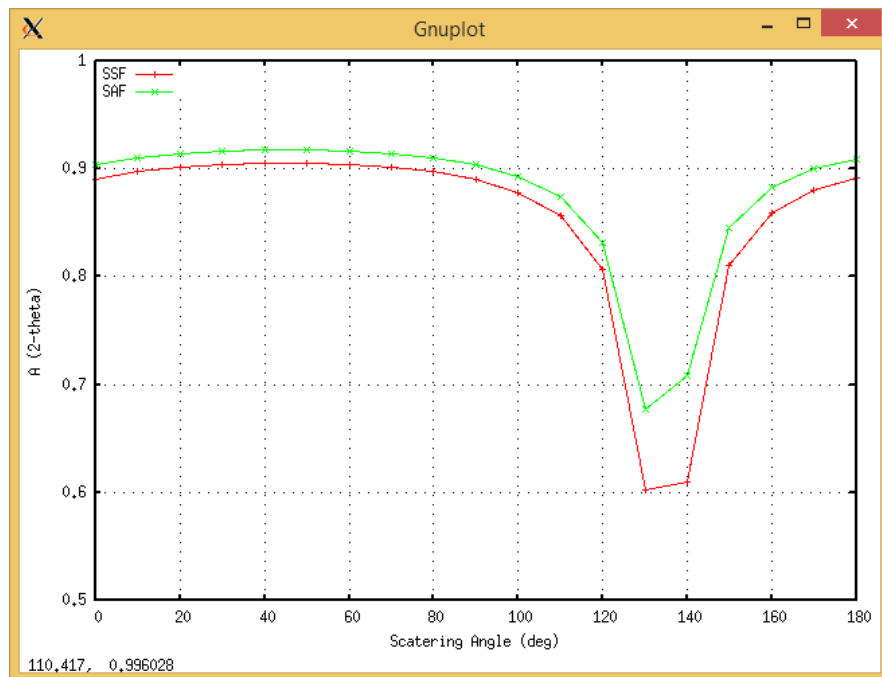
1. Vanadium slab, 1 mm thickness, no container, oriented at 135° :

```
> w11 = calc_absor_flat(a, 135.0, 0.1, 0.359, 0.358)
```



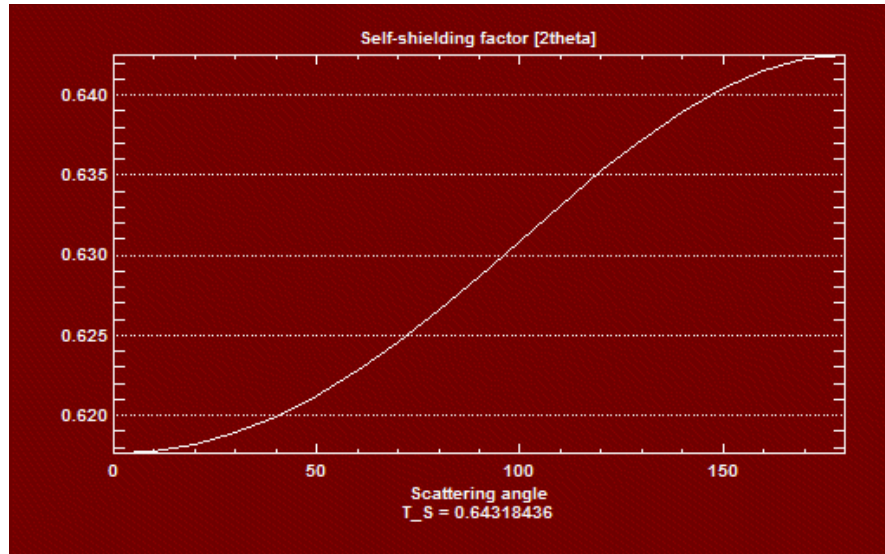
2. Vanadium slab, 1 mm thickness, in Al container with walls of thickness 0.5 mm, oriented at 135° :

```
> w11 = calc_absor_flat(a, 135.0, 0.1, 0.359, 0.358, cellWallThickness = 0.05, cellSigmaS = 0.091, cellSigmaA = 0.014)
```



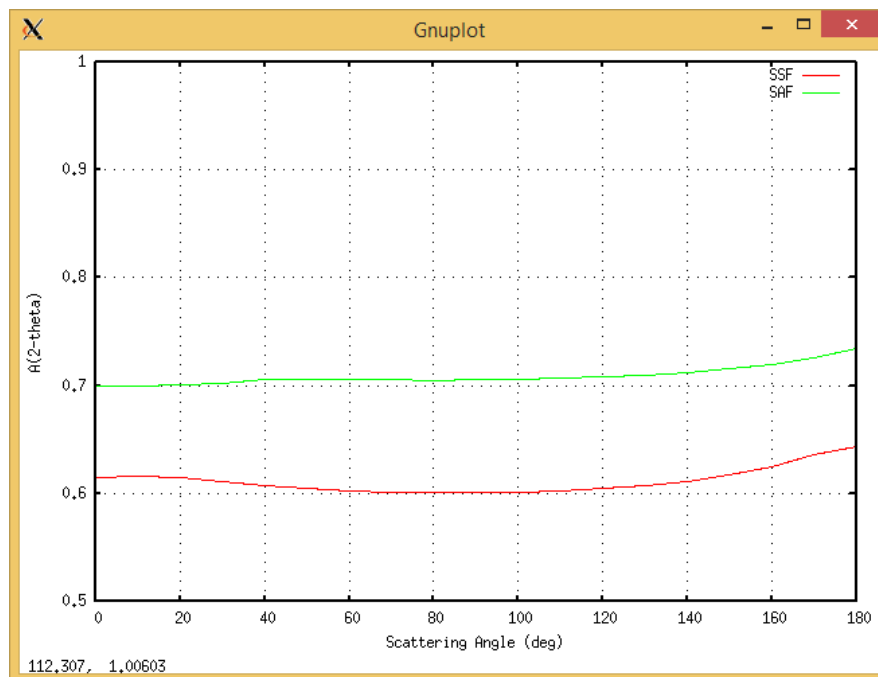
3. Massive vanadium rod, 8 mm diameter, no container:

```
> w11 = calc_absor_cyl(a, 0.8, 0.359, 0.358)
```



4. Water sample of thickness 0.2 mm enclosed between two Al cylinders of wall thickness 0.5 mm. The external diameter of the water sample (internal diameter of the outer container) is 2 cm:

```
> w11 = calc_absor_cyl(a, 2.0, 5.625, 0.022, sampleThickness=0.02, cellInThickness=0.05, cellOutThickness=0.05, cellSigmaS=0.091, cellSigmaA=0.014 )
```



References

- [1] <http://www.ill.eu/instruments-support/computing-for-science/cs-software/all-software/lamp/the-lamp-book/>
- [2] “Quasielastic Neutron Scattering”, Marc Bée, Adam Hilger (1988)
- [3] “Absorption-correction factors for scattering from flat or tubular samples: Open-source implementation libabsco, and why it should be used with caution”, White paper by Joachim Wuttke (<http://apps.jcns.fz-juelich.de/doku/sc/absco>).