

LAMP Time-of-flight Supplement

Supplement to the LAMP BOOK for TOF spectrometers: IN4, IN5, and IN6

LAMP is a program which allows you to read data into workspaces, and then to plot and treat these workspaces in a wide variety of ways. Each workspace contains data with its parameters, descriptive text, and axes information. The main LAMP interface is equipped with buttons and windows which allow you to manipulate workspaces as shown below. The sequence is not strictly cyclic and the steps can be performed in any sequence after data read-in. Workspaces adapt to the dimensions of the data read into them. To run LAMP type lamp.

When the LAMP window appears, I suggest that you select the "LAMP/layout" menu and then "extend to classical lamp". In the descriptions that follow, I will assume the the "classical LAMP" window is being used.

For further information see [the LAMP manual](#).

[tof-LAMP Home](#)[LAMP basics](#)[TOF reduction](#)[tof-LAMP FAQs](#)

Ross Stewart, stewart@ill.fr

last updated 19/07/2002

LAMP Basics

Reading Data

Plotting

Workspaces

Reading Data

Data reading is controlled from the `DATA COLLECTOR` area at the top of the main LAMP interface. LAMP looks at the system on which it is running and tries to set the best default values for the instrument and data-base. Normally the correct instrument and data-base will already be set.

Selecting Instrument.

If the instrument button does not indicate the desired instrument then press this button. A pull-down menu of instrument-groups appears. Select the desired group then the desired instrument.

Selecting the Data-Base.

The button to the right of the instrument-button contains the name of the data base. Press this button and a pull-down menu appears which normally offers:

<i>Instrument Name</i>	Data on the instrument computer.
<i>Current Cycle</i>	Data for the current cycle on the main ILL data-base
<i>Previous Cycle</i>	Data from the previous cycle on the main ILL data-base
<i>Current Path</i>	Data will be read from the path defined in the field at the top-right of the window. This path is editable.
<i>Old cycles</i>	Data will be read from the archives on the central data server (\\serdon for PCs /usr/illdata for UNIX machines)

Selecting a Workspace.

The workspace selector is at the far-right of the `DATA COLLECTOR` area. At start-up w1 will be selected. You may change this by pressing the "<" and ">" buttons on the interface.

Selecting and reading a Run.

Enter the desired run number in the "Read" field. To read in and sum several consecutive runs, use the ">" symbol, e.g. 1234>1237. Click "Read".



Tips:

1. The run will be read into the selected workspace if you hit carriage return in the "Read" field.
2. The selected run-number can be incremented by pressing the "+1" button which is next to the run-number window.
3. The command `w1=rdrun(1234)` can be entered directly in the "MANIPULATIONS" window. This example reads run number 1234 into `w1`.
4. The command `w1=rdsum(1234,1237)` will sum together runs 1234 to 1237 into `w1`, equivalent to entering 1234>1237 in the "Read" field.
5. It is often convenient to reserve `w1`, `w2`, and `w3` for special purposes: e.g.. current run, vanadium, background runs. This allows you to use the programmable buttons to maximum advantage.
6. LAMP uses dynamic memory allocation so that empty workspaces take no space. You can empty workspaces that are no longer required by setting them equal to zero. NB LAMP may become slow when all the available memory is used.

Plotting Data

The area controlling display is in the centre of the LAMP interface.

Simple Plot

Any workspace can be plotted by selecting the workspace number in the plot button with the neighbouring "<-""->" buttons. When you press the plot button the plot will appear according to the setting of the other buttons on the "DISPLAY WORKSPACE" area. LAMP automatically selects the last workspace which was

modified as the default for plotting.

Below Beside

Plots made in the small display in the centre of the LAMP interface are rapid, but small. This is the "below" option. If the "beside" button is selected plots will appear in a new window which can be rescaled and scrolled. Each new plot makes a new window "beside" which enables plots to be compared easily. The windows can be stored as icons, but should be removed if they are not needed to avoid complication.

Image Contour Surface.

The type of plot is selected with the appropriate button or buttons. These can be combined to obtain complex plots, but remember that contour plots of noisy data can take a long time to generate. "Image" is usually very fast.

View Angle

The view angle for surface plots is given in the box next to the "surface" button. This can be modified as desired.

Ranges.

The plot range can be chosen by the desired values in the fields next to the "x-range" ...etc. buttons. If the buttons are deselected the full range will be taken. Warning: When you change a workspace from channels to energy you may get a shock if you have selected an x-range based on channels!

Options Button

This button, followed by "Titles .."brings up a "set preferences" window. This allows you to change titles, default printer and various plot options.

Colours

Colours can be changed in the "beside" plot by pressing the colours button.

Zoom

To zoom in on a region of a plot press the left mouse-button and drag out a rectangle enclosing the area of interest. When you release the mouse-button the zoomed area will be plotted. Data in the zoomed area can be saved into a workspace with the command: `trap,w4` (into `w4` in this case). Unzoom by pressing "replot" in the "beside" window or "plot" in the main LAMP interface.

Cursor.

The cursor cannot work on "surface" plots. Press the left mouse-button on with the cursor arrow in the plot and the values of `x,y,z` will appear. Areas can be integrated by "dragging" an area with the right mouse-button pressed.

Printing.

Normally the correct printing device will be set so that you simply press the "print" button to obtain a hard-copy. If no device is set LAMP will make a postscript file. The printing device is set in the "Options/Titles..." window.



Tips:

1. Use "below" with "image" for fast plots.
 2. Press the "w-log" button to reveal small features with "image". Do not forget to unset "w-log" afterwards!
 3. For a small number of spectra the "vectors" style (in the "Options/Titles..." window) gives a multi-plot effect.
 4. Use two "beside" windows to compare runs.
-

Working with Workspaces

What is in a workspace?

In addition to the counting intensities in, say w_1 . Additional information is stored in the following arrays:

- ✿ x1: x-axis values
- ✿ y1: y-axis values
- ✿ e1: error bars (usually the standard deviation)
- ✿ n1: beam monitors
- ✿ p1: instrument parameters. Can be seen and edited by pressing the "Data Params" button.
- ✿ captions: `x_tit(1)`, `y_tit(1)`, `z_tit(1)`, `w_tit(1)` and `other_tit(1)` with the obvious meanings. `other_tit` contains a record of all workspace manipulations performed using the [TOF functions](#).

In order to add, subtract, etc. workspaces, commands are entered in the "MANIPULATIONS" area. The main scrollable formula entry is the easiest place to enter new commands. These are executed when carriage return is pressed.

Adding, Subtracting, etc. Workspaces.

To add two runs which are in workspaces w_1 and w_2 simply type:

$$w_3 = w_1 + w_2$$

in the "MANIPULATIONS" window. Subtraction, multiplication etc. are similar. In these operations LAMP will transfer the parameters, axes values, titles (and monitors) of the first workspace after the "=" sign into the new workspace. Effectively, " $w_3 = w_1 + w_2$ " invokes:

$$\begin{aligned} p_3 &= p_1 \\ x_3 &= x_1 \\ y_3 &= y_1 \\ x_tit(3) &= x_tit(1) \ \& \ y_tit(3) = y_tit(1) \ \& \ w_tit(3) = \\ w_tit(1) \end{aligned}$$

which is what you would expect. However, it also invokes

$$\begin{aligned} n3 &= n1 \\ e3 &= e1 \end{aligned}$$

which is not at all what you expect.

There are two very important points here, which are often a source of error:

1) If you are adding together raw (un-normalised) data, you should always explicitly add the monitors, i.e.

$$n3 = n1 + n2$$

in the above example

2) You should always explicitly perform the error bar calculation when manipulating workspaces, i.e.

$$e3 = \text{sqrt}(e1^2 + e2^2)$$

in the above example. Note that you do not need to do this when the manipulation is done by an existing TOF function, e.g.

$$w2 = \text{normalise}(w1)$$

automatically gets the error bars right.

When multiplying or dividing two workspaces by a number, always remember to perform the error bar calculation explicitly, e.g.

$$w2 = w1/1.87 \quad \& \quad e2 = e1/1.87$$

When dividing two workspaces by each other, a `divide` function has been written to automatically work out the appropriate error bars, in order to save tediously working out the expression every time, i.e.

$$w3 = \text{divide}(w1, w2)$$

does not require an explicit error bar calculation.

Extracting sub-spectra.

To extract spectra 20 to 30 from `w1` into `w2`:

$$w2 = w1(*, 19:29)$$

In this example the "*" implies all channels. If you were only interested in channels 300 to 400 for spectra 20 to 30:

```
w2=w1 (299:399, 19:29)
```

LAMP ensures that parameters, axes etc. are transferred correctly but it does not update the values for the number of spectra and number of channels in the new parameters. These can be edited with the "data params" button.

Summing several spectra

If the sum of spectra collected at all different scattering angles is required:

```
w2=total (w1, 2)
```

Here the "2" means sum over the second dimension i.e.. spectra. A sum of a range of spectra, say spectrum 50 to spectrum 100 can be made:

```
w2=total (w1 (*, 49:99), 2)
```

Getting s(Q)

A sum of all channels (for s(Q)):

```
w2=total (w1, 1)
```

If only elastic s(Q) is required and the elastic-peak position is known to be from say 200 to 220:

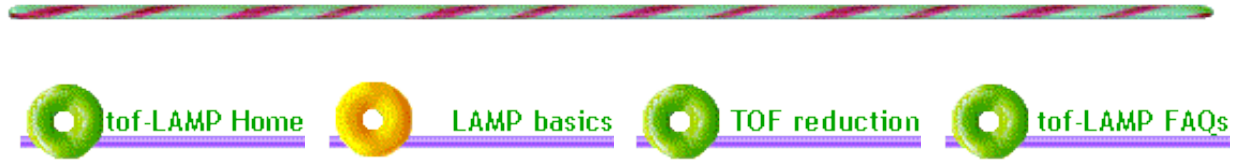
```
w2=total (w1 (199:219, *), 1)
```



Tips:

1. Use the history to cut out previous commands and paste them into "MANIPULATIONS" for reuse.

2. The programmable buttons are often easier to use than the "MANIPULATIONS"
3. Simple error messages appear between the "MANIPULATIONS" and history windows.
4. General warning and error messages appear in the winterm window from which LAMP was launched. Keep this window in view.



TOF Data Reduction

Batch Files

List of TOF Functions

Batch Files

It is recommended to write batch files to perform on-line data analysis. A batch file can be created by clicking on the "User Macros" button, or selecting "Edit/User batch files, MACROS" from the menubar. The resulting window is an editor for creating macros and command files. Enter the name of the batch file you wish to create (must end with `.prox`) in the file name field, e.g. `test.prox`. The first lines of your command file will usually consist of reading in run numbers, followed by calls to functions to process the data. A number of data processing functions have been written specifically for TOF data analysis which are described here.

Example of data reduction batch file for IN6:

a) without absolute normalisation

```
w1 = rdsum(7272,7280) ; read sample run
w1 = normalise(w1,/monitor)
w2 = rdrun(7303) ; read empty run
w2 = normalise(w2,/monitor)

w3 = w1 - w2 ; Subtract the
background
e3 = sqrt(e1^2 + e2^2) ; Propagate the
errors

w20 = rdrun(7271) ; read vanadium run
w20 = normalise(w20,/monitor)
w4 = vnorm(w3,w20,min = 100,max = 110) ; Normalise w3 to
vanadium ; spectra, integrated
between ; time channels, 100
and 110

w5 = remove\_spectra(w4,[1,87,122,123]) ; Remove bad spectra
nos.
w6 = sumbank(w5) ; Add spectra at same
Q
w7 = corr\_tof(w6,/det_eff,/frameoverlap); Correct the data
```

```

for
; energy dependent
detector
; efficiency and
frame overlap

w8 = t2e(w7) ; Convert to energy
axis
w9 = reb(w8,dE = 0.02) ; Rebin energy to
constant dE

w10 = sqw_rebin(w9,dQ = 0.05,emin = -5) ; Rebin to S(Q,w)
grid for given
; dQ and minimum
energy

output_allqs, w10, file = 'helium_T050K'; Output all constant
Qs in w10
; to x,y,e ascii
files.

w11 = qstrip(w9,Q=1.93,dQ=0.1) ; Select all data at
constant
; Q = 1.93 (NB input
is w9)
output, w11, file = 'helium_T050K.q193' ; and output this Q-
cut

w12=estrip(w9,E=0.74,dE=0.2,dQ=0.05) ; Select all data at
constant
; E = 0.74, and bin
in Q-steps
; of 0.2
output, w12, file = 'helium_T050K.e074' ; and output this E-
cut

```

b) including absolute normalisation and attenuation corrections

```

w1 = rdsum(1055,1057) ; read and sum sample runs
w1 = normalise(w1,/monitor)
w2 = rdsum(917,918) ; read and sum empty runs
w2 = normalise(w2,/monitor)
w3 = rdsum(966,967) ; read and sum cadmium runs
w3 = normalise(w3,/monitor)

w4 = w1 - w3 ; Subtract the cadmium
e4 = sqrt(e1^2 + e3^2) ; and propagate errors

w5 =
slab_tof(w4,angle=135.,inc_xs=0.37,abs_xs=12.12,thick=2.56,N=7.82)

```

```

; Correct the cadmium subtracted data (w4) for sample attenuation
; - assuming slab geometry

w6 = w5 - (w2 - w3) ; Subtract the non-
attenuated ; background (empty -
cadmium) ; from the corrected data
; and propagate errors
e6 = sqrt(e5^2 + e2^2 + e3^2)

w20=rdsun(919,922) ; read and sum vanadium
runs ;
w20=normalise(w20,/monitor)

w19 = w20 - w3 ; Subtract the cadmium
e19 = sqrt(e20^2 + e3^2) ; and propagate errors
w18 =
slab_tof(w19,angle=135.,inc_xs=5.08,abs_xs=5.08,thick=1.,N=7.19)
; Correct the cadmium subtracted vanadium data (w19) for vanadium
; attenuation - assuming slab geometry

w17 = w18 - (w2 - w3) ; Subtract the non-
attenuated ; background (empty -
cadmium) ; and propagate errors
e17 = sqrt(e18^2 + e2^2 + e3^2)

w7 = vnorm(w6,w17,min=305,max=327,tv=1.,Ns=7.82,ts=2.56)
; Normalise to the corrected vanadium data (w17) integrated between
; time channels 305 and 327, and put on absolute scale
(barns/st/meV/fu)

w8=remove_spectra(w7,[89,92,96,97,152,153,154,155,171,181,192,198])
; remove bad spectrum nos.

w9 = corr_tof(w8,/det_eff) ; Correct data for energy
; dependent detector
efficiency
w10 = t2e(w9) ; Convert to energy axis
w11 = reb(w10,dE = 0.05) ; Rebin to constant dE =
0.05

w12 = sqw_rebin(w11,dQ = .1,emin = -20)'; Rebin to S(Q,w) grid for
given ; dQ and minimum energy
output_allqs, w12, file = 'ba9300' ; and output all Q-cuts

```

Example of data reduction batch file for IN5:

```

w1 = rdsun(7272,7280) ; read and sum sample

```

```

runs (50K)
w1 = normalise(w1,/monitor)
w2 = rdrun(7303) ; read empty run
w2 = normalise(w2,/monitor)

w3 = w1 - w2
e3 = sqrt(e1^2+e2^2) ; sample signal at
T=50K (phonons)

w4=rescale_t(w3,Told=50,Tnew=10) ; rescale phonons to
T=10K ; (detailed balance)

w5=rdsun(7281,7288) ; sample at T=10K
w5=normalise(w5,/monitor)

w6 = w5 - w2 & e6 = sqrt(e5^2 + e2^2) ; sample signal at
T=10K
w7 = w6 - w4 & e7 = sqrt(e6^2 + e4^2) ; phonon subtracted
low-T data

w20 = rdrun(7271) ; read vanadium run
w20 = normalise(w20)

w8 = vnorm(w7,w20,min=150,max=170) ; Normalise w7 to
vanadium ; spectra, integrated
between ; time channels, 150
and 170
w9 = remove_spectra(w8,[116,198,199]) ; Remove bad spectrum
nos.
w10 = sumbank(w9,dQ = 0.1) ; Sum angles at
~constant Q-step

w11 = corr_tof(w10,/det_eff,/frameoverlap,/bkgd)
; Correct data for energy dependent detector efficiency,
frame overlap and
; subtract any time-independent background contribution

w12 = t2e(w11,/in5multi) ; Convert to energy
axis and ; use low-angle
multi-detector
w13 = reb(w12,dE=0.05) ; Rebin to constant
dE = 0.05
w14=total(w13(*,0:2),2) ; extract magnetic
signal at ; lowest angles
output, w14, file='magnetic.dat' ; and output data

```

All the functions automatically look up the name of the instrument associated with the LAMP session and process the data accordingly. The energy transfer convention is the usual one : neutrons losing energy at the sample (down-scattering) arrive at positive energy transfer, up-scattered neutrons at negative energy transfer. The function source code can be found under the "TOF" heading in the file list at the left hand side of the "User Macros" window. They are extensively commented and users (and instrument responsables and local contacts) are encouraged to look at the source code for details. None of the routines use separate graphical interfaces. This enables them to run directly from a batch file, which then contains a complete record of the data reduction procedure. In addition, all workspace manipulations using these functions are recorded in the workspace subtitle (other_tit), which appears in print-outs for future reference.

List of tof-LAMP reduction Functions


<u>corr_tof</u>	<u>remove_spectra</u>
<u>elastic</u>	<u>rescale_t</u>
<u>estrip</u>	<u>slab_tof</u>
<u>input</u>	<u>smooth_bkgd</u>
<u>lineup</u>	<u>sqw_interp</u>
<u>normalise</u>	<u>sqw_rebin</u>
<u>output</u>	<u>sumbank</u>
<u>output_allqs</u>	<u>t2e</u>
<u>qstrip</u>	<u>vnorm</u>
<u>reb</u>	

corr_tof

Corrects data in TOF for energy-dependence of detector efficiency, frame overlap and time-independent background. Calling procedure:



```
w6=corr_tof(w5[,/det_eff][,/frameoverlap][,/bkgd])
```

The three keywords (/det_eff, /frameoverlap and /bkgd) are optionally included to apply the corrections or not

-  `/det_eff`: For IN6, it uses the expressions given in ILL internal report 83BL21G by Y. Blanc:

$$\eta = 0.94(1 - \exp(-0.363\lambda)) \quad \text{for } \lambda < 4 \text{ \AA}$$

$$\eta = \exp(-0.0063\lambda)(1 - \exp(-0.363\lambda)) \quad \text{for } \lambda > 4 \text{ \AA}$$
 For IN5, no accurate description is available, due to the large variation in quality between the detector tubes. I have taken the expression used on D7:

$$\eta = 1 - \exp(-5.6/E_f^{1/2})$$
 The above expressions are all normalised to 1 at the elastic peak position.
-  `/frameoverlap`: This correction is appropriate if there is no frame overlap from short times over into long ones, i.e. if the beginning of the time frame is chosen such that the measurement starts at very large negative energy transfer (up-scattering) compared to the temperature scale of the sample. In this case, all up-scattered neutrons arrive at the correct position in the time frame and frame overlap occurs only for down-scattered neutrons which appear at short times in the subsequent time frame. It assumes that $S(Q, \omega)$ is constant in energy between the highest measured energy transfer $\hbar\omega$ (end of the time frame) and $\hbar\omega = E_i$. In this case the intensity as a function of time t of the frame-overlap component should have a t^{-4} dependence. The magnitude of the frame overlap component is determined by integrating the counts over the last 10 time channels, and the continuation of this, scaled with t^{-4} is subtracted from the beginning of the time frame. This correction is applied independently for each spectrum.
-  `/bkgd`: This is a useful correction if there is no empty can measurement available, to avoid the data "blowing up" at large energy transfers after transforming from TOF to $\hbar\omega$. It estimates the minimum intensity level in each spectrum by applying a moving filter 21 time channels wide and subtracts it.

elastic

Finds the elastic peak position for each spectrum and puts them into the output workspace. Calling procedure:

```
w2 = elastic(w1, min=<min>, max=<max> [, /save])
```

The two arguments (`min`, `max`) specify the time channel range in which to search for the elastic peak. Time channels are defined to run from 1 to n , where n is the number of channels. If these arguments are omitted, the routine searches in the region (± 30 time channels) of the elastic peak position specified in the input workspace parameter block. The peak position is found by least-squares fitting a Gaussian plus a flat background to each spectrum. The optional keyword `- /save` may be included to save the elastic peak positions for each detector in a file called "elastic.dat"

estrip

Rebins data to regular-grid $S(Q, \omega)$ at a single constant- $\hbar\omega$ value. Calling procedure:

```
w2=estrip(w1, E=<evalue>, dE=<dE>, dQ=<dQ>)
```

It extracts data lying on a single constant- $\hbar\omega$ strip in the Q-E plane, centred at $\hbar\omega = E$. The three arguments (E, dE, dQ) specify the Q-E range to extract as follows:

- ✿ E: The centre of the constant- $\hbar\omega$ strip to extract, in meV.
- ✿ dE: The full-width in $\hbar\omega$ of the strip to extract.
- ✿ dQ: width of Q-binning (= point spacing in Q of output data) in \AA^{-1} .

The algorithm used is the same as that used in the [qstrip](#) and [sqw_rebin](#) routines. See [qstrip](#) for more details.

input

Reads standard format ASCII data into a workspace. The file format for 1-dimensional data is three columns: x-data, y-data and error bars, followed by x-caption, y-caption and two titles. For larger-dimensional data sets, another, less portable, format is used. These are the same formats used by [output](#). Calling procedure:

```
w1=input(file='filename.dat')
```

lineup

Lines up the elastic peaks in the spectra of a workspace. Calling procedure:

```
w2=lineup(w1)
```

Because there are frequently small differences between the distance from the sample to individual detector-groups there can be slight differences in the time-of-flight channel in which the elastic peak arises. This function first smoothes each spectrum and then estimates the position of the maximum. An average of these positions is taken and then all spectra are shifted so that their elastic peaks are at the average position. Any peak which is more than 10 channels away from the elastic-peak channel given in the parameters is not shifted. The new average peak position is entered in the parameters.

Bear in mind that the counting statistics in an individual spectrum need to be adequate to enable the elastic peak to be found. Otherwise the routine does nothing.

normalise

Normalises data to monitor or counting time. This should always be the first routine called after reading in the data. Calling procedure:

```
w2=normalise(w1[,/raw][,/monitor][,/time])
```

The optional keywords `/raw`, `/monitor` and `/time` specify what to normalise to:

- ✿ `/raw`: no normalisation
- ✿ `/monitor`: normalise to 1000 monitor 1 counts (default)
- ✿ `/time`: normalise to counting time in minutes. This is dangerous, however, since the counting time is saved in the workspace parameter block. If several runs are added together before calling `normalise`, the counting time in the workspace parameter block will not be the sum of the counting times, but that of the first (or last) read run.

`normalise` finds the monitor peak using a least-squares Gaussian fit and integrates the counts over the peak region, subtracting a flat background. It also sets up the workspace error bars (square root of the counts) and finds the elastic peak position by a Gaussian fit to the sum of all the detectors and puts the value into the output workspace parameter block. For IN5 the sum is performed only over the high-angle detectors (scattering angle > 10 degrees).

output

Saves workspace data into a standard format ASCII file. The file format for 1-dimensional data is three columns: x-data, y-data and error bars, followed by x-caption, y-caption and two titles. For larger-dimensional data sets, another, less portable, format is used. These are the same formats used by `input`. Calling procedure:

```
output, w1, file='filename.dat'
```

output_allqs

Saves 2-dimensional workspace data in a series of standard format ASCII data files. The workspace should contain output data of the `sqw_rebin` routine. The data format for each file is identical to that of `output` for 1-dimensional data. Calling procedure:

```
output_allqs, w1, file='filename'
```

The workspace data are saved in a series of files with names, e.g.

```
filename.q000  
filename.q005
```

```
filename.q010
filename.q015, etc
```

where the numbers after the 'q' specify the Q-value of the data, i.e.



filename.q010 contains the data at $Q = 0.10 \text{ \AA}^{-1}$. Q-values for which no data exist in the workspace are not saved.

qstrip

Rebins data to regular-grid $S(Q, \omega)$ at a single constant-Q value. Calling procedure:

```
w2=qstrip(w1, Q=<Qvalue>, dQ=<dQ>)
```

It extracts data lying on a single constant-Q strip in the Q-E plane, centered at $Q = \langle Qvalue \rangle$. The two arguments (Q, dQ) specify the Q-E range to extract as follows:

-  Q : The centre of the constant-Q strip to extract, in units of \AA^{-1} .
-  dQ : The full-width in Q of the strip to extract.

The energy binning is unchanged from that of the input data. The algorithm used to rebin to constant Q is based on the fact that the detectors on both IN5 and IN6 lie very close together. The part of the Q-E plane covered by a detector is represented as a strip of constant scattering angle, centred at the detector scattering angle and touching the strips from the neighbouring detectors. Each strip is subdivided into a number of parallelograms, representing the data points. The constant-Q strip onto which the data are to be rebinned, is divided into a number of rectangles, and the value for $S(Q, \omega)$ assigned to each rectangle is given by

$$S(Q, \omega) = \sum_{ij} S'(Q_{ij}, \omega_{ij}) A_{ij}$$

where S' is the measured scattering function at constant scattering angle. Subscripts i and j denote detector number and channel number, respectively. A_{ij} is the overlap area between the rectangle in the chosen constant-Q strip and the parallelogram corresponding to point j of spectrum i . The same algorithm is used in the [estrip](#) and [sqw_rebin](#) routines.

reb

Rebins data to regular steps in $\hbar\omega$ with error bar propagation. Calling procedure:

```
w2=reb(w1, dE=<dE> [, /forcebin])
```

TOF data is transformed to energy transfer $\hbar\omega$ by the [t2e](#) routine. The spacing of the points once transformed to $\hbar\omega$ is no longer constant. `reb` rebins the data to constant energy bin width. The argument `dE`, and keyword `/force` specify the binning as follows:

- ✿ dE : The desired bin width (= point spacing of output data) in meV.
- ✿ `/forcebin`: If this keyword is not set, the data are only rebinned where the original point spacing is less than dE . If set, the data are rebinned everywhere, which gives rise to strong correlations between the data points where the original point spacing is greater than dE .

remove_spectra

Removes suspect spectra from a workspace. Calling procedure:

```
w2=remove_spectra(w1, [s1, s2, s3, ...])
```

Spectrum numbers are defined to run from 1 to n , where n is the number of spectra in the workspace. The spectrum numbers to remove should be given in square brackets, separated by commas.

rescale_t

Rescales data measured at one temperature to another temperature, using the Bose thermal population factor. Calling procedure:

```
w2=rescale_t(w1, Told=<Told>, Tnew=<Tnew>)
```

The input data must have energy transfer $\hbar\omega$ as x-axis, i.e. they must have been passed through `t2e`. `Told` gives the temperature that the data was measured at. `Tnew` specifies the temperature that the data should be rescaled to. The idea is that if nothing changes in the sample apart from the Bose thermal population factor, `rescale_t` can convert data taken at one temperature to what you would have measured at another temperature. There are two caveats: Firstly, the elastic peak should not rescale with the Bose factor, but this routine simply rescales everything regardless of physical origin. It is therefore incorrect for the elastic peak and also in the region where the resolution-limited tail of the elastic peak contributes to the scattering. This can be an important effect on IN6 where the resolution function has long tails in energy. Secondly, when rescaling from low to high temperature, the ratio in Bose factor becomes a very large number for high-energy up-scattering (large negative $\hbar\omega$). The rescaling can then "blow up" any, otherwise negligible, background effects in this energy range. Use with care.

slab_tof

Takes a 2-D time-of-flight workspace and corrects for sample attenuation of the scattered neutrons assuming slab sample geometry. Calling procedure:

```
w2=slab_tof(w1,angle=<ang>,inc_xs=<ixs>,abs_xs=<axs>,thick=<t>,N=<N>)
```

The 5 arguments specify the angle, thickness, number density and cross-sections of the sample as follows:

- ✿ angle: is the angle in degrees of sample plane relative to incident beam
- ✿ abs_xs: is the sample absorption cross-section per f.u. for thermal neutrons ($\lambda=1.79\text{\AA}$)
- ✿ inc_xs: is the sample spin-incoherent cross-section per f.u.
- ✿ thick: is sample thickness (mm)
- ✿ N: is the number density ($\times 10^{22} \text{ cm}^{-3}$)

smooth_bkgd

For IN5 data only. Performs a smoothing of a background measurement. Calling procedure:

```
w2=smooth_bkgd(w1,ismooth)
```

Assumes that the signal in all spectra consists of an elastic peak plus broad background scattering. *ismooth* specifies how to perform the smoothing:

- ✿ ismooth<0: replaces the elastic peak by a best-fit Gaussian and applies a moving filter of width *ismooth* on the rest. *ismooth* must be an odd number.
- ✿ ismooth=0: no smoothing (default)
- ✿ ismooth>0: applies a moving filter of width *ismooth* everywhere. *ismooth* must be an odd number.

sqw_interp

Interpolates data to regular-grid $S(Q, \omega)$ covering the entire measured Q-E region. Calling procedure:

```
w2=sqw_interp(w1,dQ=<dQ>,dE=<dE>,Emin=<emin>)
```

The three arguments (*dQ*, *dE*, *Emin*) specify the point spacing and Q-E range to extract as follows:

- ✿ dQ: Point spacing in Q (in \AA^{-1}) of the output workspace.
- ✿ dE: Point spacing in $\hbar\omega$ (in meV) of the output workspace.
- ✿ Emin: the minimum value of energy transfer for which to extract data. A typical value is ~5 meV. The *Emin* argument can be omitted to give the full energy range.

sqw_interp is a quick-and-dirty way of transforming to regular $S(Q, \omega)$ grid. It is

useful for visualising the data in 3-D plots. However, the interpolation algorithm used does not allow error bars to be extracted reliably. Do not believe the error bars! For a better, but slower, transformation to regular $S(Q, \omega)$ grid which gets the error bars right, use [sqw_rebin](#), [qstrip](#) or [estrip](#).

sqw_rebin

Rebins data to regular-grid $S(Q, \omega)$ covering the entire measured Q-E region. Calling procedure:

```
w2=sqw_rebin(w1, dQ=<dQ>, Emin=<emin>)
```

The two arguments (dQ, E_{min}) specify the point spacing and Q-E range to extract as follows:

- ✿ dQ : width of Q-binning (= point spacing in Q of output data) in \AA^{-1} .
- ✿ E_{min} : the minimum value of energy transfer for which to extract data. A typical value is ~ 5 meV. The E_{min} argument can be omitted to give the full energy range.

The energy binning is unchanged from that of the input data. `sqw_rebin` is a slow, but sound way of transforming to regular $S(Q, \omega)$ grid and gives reliable error bars. The overlap algorithm used is the same as that used in the [estrip](#) and [qstrip](#) routines. See [qstrip](#) for more details.

sumbank

Adds spectra together to improve statistics. Calling procedure:

```
w2=sumbank(w1, dQ=<dQ>)
```

The dQ argument specifies how many spectra to add together. They are summed in order to give angular spacings between spectra corresponding to a Q-spacing of approximately $dQ \text{ \AA}^{-1}$. For IN6, `sumbank` can be called without specifying dQ . It then adds together spectra lying at identical scattering angles (upper, middle and lower detector banks).

t2e

Transforms time-of-flight data to energy transfer $\hbar\omega$. Calling procedure:

```
w2=t2e(w1[, /in5multi])
```

`/in5multi` is an optional argument for IN5 only. It specifies whether to use the

data from the small-angle multidetector or the higher-angle ^3He tube detectors. If `/in5multi` is set, `t2e` uses the flight path for multidetector for energy transformation. Higher-angle data (scattering angles greater than 10 degrees) are discarded. Otherwise `t2e` uses flight path for high-angle detectors for energy transformation (default). Small-angle data (scattering angles less than 10 degrees) are discarded.

The transformation Jacobian from time-of-flight t to $\hbar\omega$ ($dt/d\omega$) is proportional to t^3 . After applying this correction and transforming the x-axis to $\hbar\omega$, one obtains the double-differential scattering cross-section $d^2\sigma/d\Omega d\omega$, which is then multiplied by a factor of k_i/k_f to obtain $S(Q, \omega)$. This factor is proportional to another factor of t . In total, the data are thus multiplied by a factor proportional to t^4 in `t2e`. This means that any background, which may appear negligible in the time-of-flight representation, is likely to "blow up" at large (positive) energy transfer. The data should therefore ideally be background-subtracted before calling `t2e`. Alternatively, a flat background can be subtracted using the [corr_tof](#) routine first.




vnorm

Normalises data to vanadium. Calling procedure:

```
w2=vnorm(w1,w20,min=<min>,max=<max>[,tv=<tv>,Ns=<Ns>,ts=<ts>])
```

In this example, `w1` is the workspace containing the data to be normalised and `w20` contains the vanadium data. The two workspaces must have the same number of spectra. The two arguments (`min`, `max`) specify the time-channel range to use for integrating over the vanadium elastic peak. Time channels are defined to run from 1 to n , where n is the number of channels.

The optional arguments (`tv`, `Ns`, `ts`) are used for absolute normalisation of the data as follows

-  `tv` : is the thickness of the vanadium (mm)
-  `Ns` : is the number density of the sample ($\times 10^{22} \text{ cm}^{-3}$)
-  `ts` : is the thickness of the sample (mm)


[tof-LAMP Home](#)

[LAMP basics](#)

[TOF reduction](#)

[tof-LAMP FAQs](#)

Frequently Asked Questions

- Please ask if there are aspects of LAMP that you cannot understand or find difficult.

1. Where are the monitor spectra?

When data are read in the monitor spectra are assumed to be the first 3 spectra in the raw data. These are stripped out and put into the array `n1` for `w1`, `n2` for `w2` etc. Each "n" contains all three monitors and its dimensions are (number-of-channels,3). You can see the monitor1 spectrum of workspace 2 by typing:
`plot,n2(*,0).`

2. How does spectrum 1 in LAMP correspond to the order of spectra in the data collection?

In order to simplify plotting runs as surfaces, images and contours, LAMP removes monitors and other "non-detector" spectra from the raw data. Each instrument has its own style in the raw data, and to make things worse the first element in IDL arrays is element number 0.

For IN5 the first LAMP spectrum is instrument-spectrum 9

For IN6 the first LAMP spectrum is instrument-spectrum 7

For IN10 the first LAMP spectrum is instrument-spectrum 1

For IN16 the first LAMP spectrum is instrument-spectrum 1

3. When I press print I only get a PS file.

The LAMP default printer is set in the "Options/Titles.." menu. Press the "Options" button (just above the plot window) followed by "Titles.." and then enter the desired printer name in the appropriate field. Instrument printers usually have their names on a label somewhere on the printer.

4. Plots are wrong or badly scaled when I use "plot" command in formula entry. (e.g. plot,n1).

The plot-range buttons only apply to workspaces. When you issue the "plot" command manually you must set `xrange`, `yrange` and `zrange` manually. To plot monitor 1 of workspace `w3` from 200 to 250 channels:

```
plot,n3(*,1),xrange=[200,250]
```

5. I see no plot!

The "color" options can be used to plot black-on-black, blue-on-blue etc. A previous user may have made this selection. Also, colour-schemes which are ideal for shaded surfaces may be poor for vectors or contours.

6. LAMP is stuck (no response)

Certain operations on large arrays can take a long time. Contour plots of noisy data are very time-consuming. Using `sqw_rebin` with small increments also requires patience. If you are convinced that LAMP has gone "dead" then:

Either close the shell window which started LAMP (if you can find it) or.....

- Select "desktop" from the desk menu at the top left of the screen. You may have to move LAMP or other windows out of the way. With desktop selected take the option "Unix Shell".
 - A new window will open. Type `ps` and a list of the processes running will appear. Note the number of the process called IDL.
 - Type `kill 12345` where in place of 12345 you put the number of the IDL process. If you got it right LAMP will go out!
 - To restart LAMP type `lamp`.
-

[tof-LAMP Home](#)[LAMP basics](#)[TOF reduction](#)[tof-LAMP FAQs](#)