How to add a new file type in dcs_mslice

Yiming Qiu(yiming.qiu@nist.gov) 3/2007

This document describes the procedures to add a new file format generated by a time-of-flight neutron spectrometer in the IDL dcs_mslice program. To obtain the IDL source code of dcs_mslice program, go to http://www.ncnr.nist.gov/dave/download.html, and download the latest development version. If you want to include your file type in DAVE distribution, please contact the author.

The affected files are listed below. They are in the dave\programs\DCS\dcs_mslice directory.

dm_calc_projection.pro	: calculate projection
dm_det_spurion.pro	: calculate detector caused spurion
dm_filemenu.pro	: initialize and set up file menus
dm_filetools.pro	: handling file related events

The following steps describe in detail how to modify the files.

1. File menu initialization (dm_filemenu.pro)

In the following two lines,

self.ftypename = ptr_new(['DCS','SPE','INX'])
self.ftypeext = ptr_new(['DCS','SPE','INX'])

add the name of the file type in the first array, and the corresponding file extension(without dot) in the second array. The type index will be the location of the type in the array, e.g. 0 for DCS and 1 for SPE. If you want dcs_mslice to start with your file type as the default, change *self.ftype = 0* in dcs_mslice::Init function in *dcs_mslice.pro*.

Then in the "*case self.ftype of* " statement, add tool buttons specific to the file type to the "File Tool" menu. If no tools are added, you can either follow the example of SPE file type, or do nothing. If you do have tools, make sure you assign a unique uname to the button, because the events will be sorted by uname. Don't use a uname that starts with "ftype_". Follow the example of DCS file type to add the tool buttons. In that example, **(self.rebinsize)* is an array of integers that are factors of the time channel number. This array should start with the integer 1. *self.detPosPtr* is a pointer to a structure {two_theta:tth, dtwo_theta:dtth, psi:psi, dpsi:dpsi}. The description of the components is in the appendix. You can add a keyword to *dm_load_detpos.pro* and add the detector information of the instrument.

2. File event handling (dm_filetools.pro)

If you have added specific file tools to the menu, in the "*case strlowcase(eventname) of* " statement, add the proper event handling codes after DCS tools. Eventname is the uname you assigned to the tool menu buttons. If the tools are stand-alone widget programs, don't forget to have event.top as the group_leader of the tlb.

Besides the tools that you added, there are seven file handling events that you need to take care of. They are *loadbut*, *addbut*, *emptbut*, *bgflbut*, *bgtcbut*, *bgdkbut*, and *vandbut*. Add your own file handling codes in the "*case self.ftype of* " statements. The required and optional parameters returned from the events are listed in Table 1. Please refer to *dcs_mslice.pdf* for details about the seven file events. You are recommended to follow the example of SPE file(*dm_load_spe.pro*). Note that since SPE files usually come with their own detector information PHX file, *dm_load_phx.pro* is used to load the detector file and store the information in *self.detPosPtr*.

eventname	event function	Input	Required output parameter	optional output parameter
loadbut	This event is generated when “Load Data” button is pressed. Data is loaded and stored in self.dataStrPtr.	open_file, open_path, self.bin_avgsum, self.e_bin, self.e_range, self.eint_yn, self.samp_typ	qty[ne,ndet], or qty[ndet,nfiles] or qty[ne,ndet,nfiles], dqty, ei, emean, error, and ang[nfiles] for diffuse scattering type.	monrate, kfactor, info, weight, ewid, comment, and legend. Temperature is recommended in info.
addbut	This event is generated when “Add Data” button is pressed. Data is loaded and added or appended to the data stored in self.dataStrPtr.	All of the above and (*self.dataStrPtr).info	qty[ne,ndet] or qty[ndet,nfiles] or qty[ne,ndet,nfiles], dqty, ang[nfiles], error	weight, comment
emptbut	This event is generated when “Background->Load Empty Can File(s)” button is pressed. Data is loaded and stored in self.bgdata.	Same as loadbut.	qty[ne,ndet] or qty[ndet,nfiles] or qty[ne,ndet,nfiles], dqty, ang, error	comment
bgflbut	This event is generated when “Background->Load Detector Background File(s)” button is pressed. Data is loaded and stored in self.bgdetsPtr.	open_file, open_path, self.bgeran	qty[ndet], dqty, error	comment
bgtcbut	This event is generated when “Background->Load Time Channel Background File(s)” button is pressed. Data is loaded and stored in self.bgtchnPtr.	open_file, open_path, self.bgeran	qty[ndet], dqty, error	comment
bgdkbut	This event is generated when “Background->Load Dark Count File(s)” button is pressed. Data is loaded and stored in self.bgratePtr.	open_file, open_path	bgrate[ndet], dbgrate, error	comment
vandbut	This event is generated when “Background->Load Vanadium File(s)” button is pressed. Data is loaded and the calculated detector efficiency is stored in self.eff.	open_file, open_path	qty[ne,ndet], ei, error	comment

Table 1: List of input and output parameters for the seven file events.

3. Projection calculation (dm_calc_projection.pro)

The information needed for different instrument or file type to properly calculate projection is coded at the starting "*case self.ftype of"* statement. You can follow the example of DCS or SPE type.

is_spe variable is a flag to show whether the detector definition is a SPE type, which is described at the appendix, or a DCS type. In a DCS type definition, two theta can be both positive and negative. The angle is positive if the detector is rotated clockwise to the incident neutron beam. Psi is defined as the angle between the diffracted neutron beam detected by the detector and the horizontal scattering plane, and the angle is positive above the horizontal plane.

det_thick variable is the ^{3}He detector thickness and *det_p* is the pressure of the ^{3}He gas. These two variables are used to calculate the energy dependent detector efficiency(not to be confused with the angular dependent detector efficiency).

Since the projection calculation requires the detector information, a line of code to load the detector information should also be included if the detector pointer *self.detPosPtr* is not assigned values.

4. Detector spurion calculation (dm_det_spurion.pro)

It has been found in DCS that the detector itself can cause spurions. These spurions are scatterings from the detector casing material, stainless steel for DCS. dcs_mslice can calculate and plot the occurrence of these spurions.

The variables required for the procedure are *psi_0_only*, *d_sd*, *a_det*, *q_det*, and *i_det*. They should be provided at the starting "*case self.ftype of"* statement at the very beginning of the code. The explanation of the variables is listed in the code.

Appendix: SPE file format

SPE files are ASCII files generated from HOMER conversion program in ISIS time of flight instruments such as MARI and HET. However, all 2D data can be saved as an ASCII spe file. For example, if you have a 2D data Z[nx, ny], and the corresponding error dZ[nx,ny], with the xaxis data X[nx], and the yaxis data Y[ny], the spe file will look like:

```
nx   ny          ;First line, each number should only occupy 5 character space. In IDL, use
                 ;"(2(i5))" formatting code.
### Phi Grid     ;Second line, and followed by nx+1 X data. For time-of-flight data, this is
                 ;usually the two theta values of the detectors. If X is an nx-element array, 0 is
                 ;patched to the last element. X can be an nx+1-element array, which represents
                 ;a histogram data. Each line should contain 8 numbers, and each number should
                 ;occupy 10 character space. In IDL, use "(8(g10.5))" formatting code. Use
                 ;"(8(g10.3))" if "*******" shows up in the saved SPE file.
### Energy Grid  ;After this line, ny+1 Y data follow. For time-of-flight data, this is usually the
                 ;energy transfer for the time channels. If Y is an ny-element array, 0 is patched
                 ;to the last element. Y can be an ny+1-element array, which represents a
                 ;histogram data. The data format is the same as in phi grid, i.e. "(8(g10.5))".
                 ;After the energy grid are nx units of Z and dZ data. The first unit is shown
                 ;here.
### S(Phi,w)     ;ny Z[0,*] data follow this line. The data format is the same as in phi grid, i.e.
                 ;"(8(g10.5))".
### Errors       ;ny dZ[0,*] data follow this line, The data format is the same as in phi grid, i.e.
                 ;"(8(g10.5))".
```

```
                    ;Note that quite often there are no spaces between data, and there shouldn't be
                    ;any empty lines in the file.
```

SPE files are usually accompanied by a PHX detector file for neutron time-of-flight data. The format of a PHX file is as follows:

```
ndet                    ;number of detectors
10 0 tth psi dtth dpsi 0    ;Then ndet lines of data similar to the left. The first, second, and
                            ;seventh numbers are not important.
                            ;The numbers should each occupy 10 character space and
                            ;be separated by an empty space.
```

tth is the two theta angle of the detector, which is always greater than zero; **psi** is the azimuthal angle, defined such that psi=0, 180 at the horizontal scattering plane, and 90 and 270 perpendicular to the horizontal scattering plane. **dtth** and **dpsi** are the uncertainties of tth and psi respectively.

You can use *dm_load_spe.pro, dm_write_spe.pro, dm_load_phx.pro, and dm_write_phx* to read and write a spe and phx file.