Path

Z:\ARDB Analysis Matlab\RHS_private
Z:\ARDB Analysis Matlab\RHS_private\Projects
Z:\ARDB Analysis Matlab\Work_area
Z:\ARDB Analysis Matlab\Library
Z:\ARDB Analysis Matlab\Pbgfa
Z:\ARDB Analysis Matlab\mpgread
Z:\ARDB Analysis Matlab\mpgread\src
Z:\ARDB Analysis Matlab\mpgwrite
Z:\ARDB Analysis Matlab\mpgwrite\mpgwrite
Z:\ARDB Analysis Matlab\mpgwrite\src

The order is important since programs load in the order given in the path.

1) You should replace the RHS_private path with a directory of your own. This gives you a place to put routines you are writing and to give them priority over ones in the work_area, library, etc. Remember programs in the directory in which you are working override anything in the path, so it is a good idea not to put programs in shared disk areas. Your colleagues as well as you will use them. The private areas are good for that
2) Most of the analysis programs are in the library. Those that have been or are being modified are in the work_area. Programs should be loaded from the work_area first since bug repairs, etc will be there until the library is updated.
3) Pbgfa contains programs related to CUDOS analysis. This is not needed in your path for e163, E167, etc
4) The 6 mpgread and mpgwrite path elements are used to generate and view mpg movies. You may or may not need them. It is easiest to just include them in the path.

Programs

dataload_control – determines the format of the data being analyzed. Must be run before any other programs
option_control – determines the fits that are performed in spring2
spring2 – image fitting program
correlate – program to plot and histogram variables

Calcs & Hooks in correlate

An example of calculating high and low energy sigmas from an asymmetric Gaussian fit

```
[nmatch,sig_mean]=cell_compare('AsymFit xRMS LP',CPname);
  if nmatch > 0
      [nmatch,asymm]=cell_compare('AsymFit Asym x LP',CPname);
      if nmatch > 0
          sigma_left=sig_mean.*(1+asymm);
          sigma_right=sig_mean.*(1-asymm);
          nCALC=nCALC+1;
```

```
            nCPtotal=nCPtotal+1;
            CPname(nCPtotal)=cellstr('High Energy RMS');
            CPdata(:,nCPtotal)=sigma_left;
            nCALC=nCALC+1;
            nCPtotal=nCPtotal+1;
            CPname(nCPtotal)=cellstr('Low Energy RMS');
            CPdata(:,nCPtotal)=sigma_right;
        end
    end
```

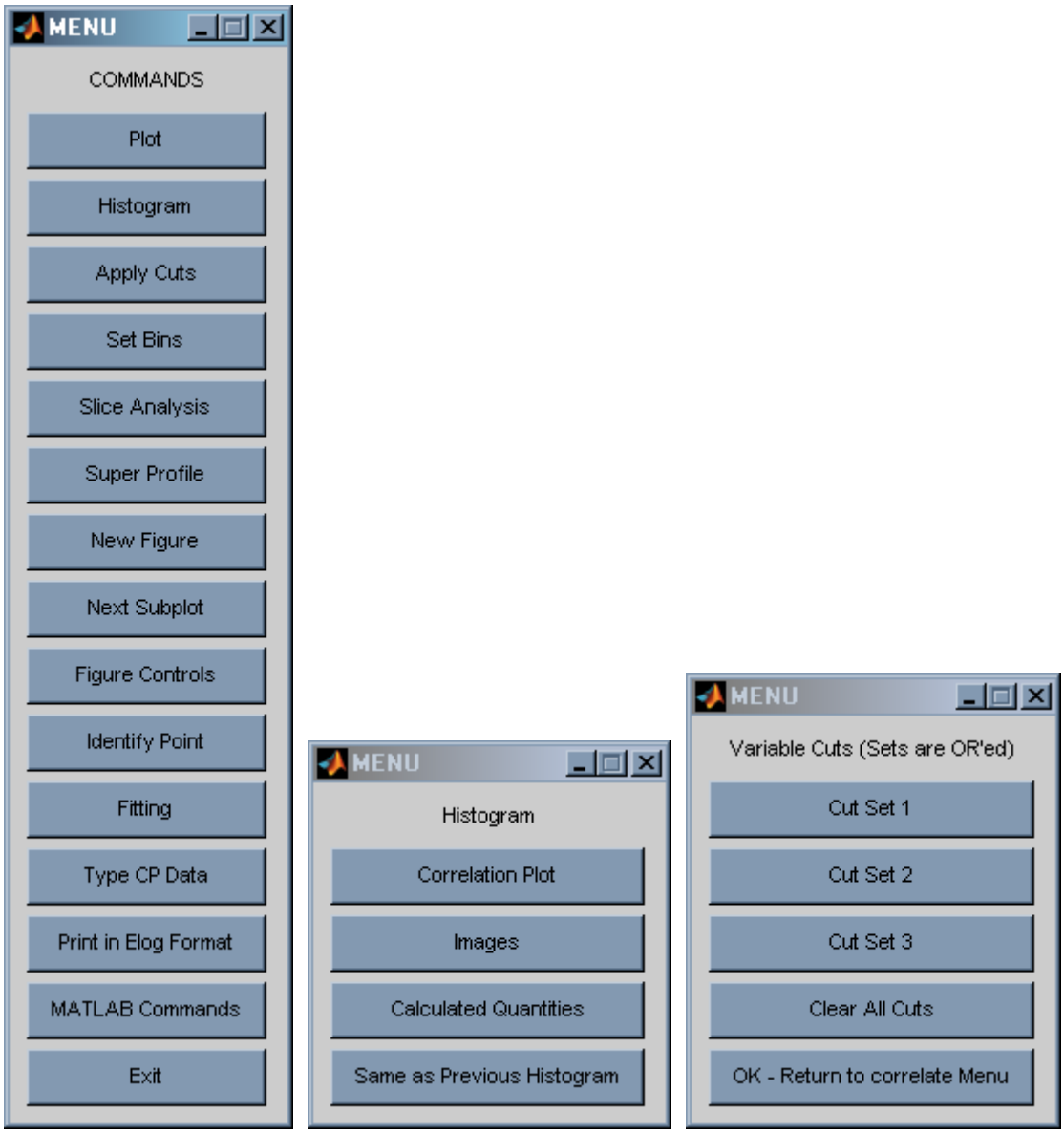
1) Variable names are stored in CPname and variable data are stored in CPdata
2) cell_compare extracts data
3) nCALC and nCPtotal must be incremented. (There is one more calculated variable and one more total number of variables.)
4) The new variable must be given a name CPname(nCPtotal), and the data must be placed in the data array CPdata(:,nCPtotal)=

Each user has a "hook", for example hook_siemann, that is called during the process of loading data. This is the place to put calculations you are interested in.

Basic data structure

Data are stored in an array CPdata(nmeas,nCPtotal) where nmeas = number of events in run, and nCPtotal = number of variables. The location of a particular quantity (for example the mean value of a Gaussian image fit) is not fixed; it depends on other fits, etc. Therefore, there has to be a pointer to the data. That pointer is the variable name, CPname. CPname is also the label given to buttons in the menus.

Menus in correlate



Slice Analysis = E157 streak camera analysis
MATLAB Commands = allows you to use MATLAB commands without exiting correlate
Super Profile = waterfall plot with choice of sorting variable

User fitting functions

spring2 and correlate have the ability to incorporate "user" fitting functions. Three functions need to be written to use this feature.

The names of these functions are fixed and you need to use them. The functions are (X = 1, 2, …, 5 depending on which user function you are writing)
csf_userX = the fitting function that is called by spring2.
extract_userX = function used by correlate to extract data from files written by spring2
userX_names = function to give names to the fit variables. These are the associated CPname's which are menu button labels and pointers to the data.

Here is an example of the three functions:

```
function f=csf_user1
%   9/1/2007 - example of a user fit in spring2.
%   In this example
%       1) background is subtracted
%       2) x value of the point where y is a maximum is determined
%       3) x values of the half maximum points are determined.
%   The function returns the maximum point and the
%   the differences of the half maximum points from the maximum point.
%       f=[max_x,upper_x,lower_x]

% Input data arrays
global xdata
global ydata

%   Data processing and determination of fit parameters
nlen=length(ydata);
back=sum(ydata(1:5))+sum(ydata(nlen-4:nlen));
back=back/10;
ybf=ydata-back;
[max_y,imax]=max(ybf);
max_x=xdata(imax);
n=imax;
while 1
    n=n+1;
    if ydata(n) < max_y/2,break,end
end
upper_x=xdata(n)-max_x;

n=imax;
while 1
    n=n-1;
    if ydata(n) < max_y/2,break,end
end
lower_x=max_x-xdata(n);

%   The fit parameters are returned
f=[max_x,upper_x,lower_x];

return
```

```
function f=extract_user1(dim,A)
%   9/1/2007 - This is a function that is used in correlate to extract
the parameters
%       image fits.  In this example the fit had three parameters.
%       dim = h or H for the horizontal dimension and v or V for the
%       vertical dimension
%       A is an array which is filled by correlate
%   Note on pointers.  SPRING2_H and SPRING2_V are pointers to the
%   locations of data.  These need to be included to extract data
%       SPRING2_H(21) or SPRING2_V(21) = user 1 fit function
%       SPRING2_H(22) or SPRING2_V(22) = user 2 fit function
%       SPRING2_H(23) or SPRING2_V(23) = user 3 fit function
%       SPRING2_H(24) or SPRING2_V(24) = user 4 fit function
%       SPRING2_H(25) or SPRING2_V(25) = user 5 fit function

E157_globals     %  Must be included - brings in shared variables

f=[NaN,NaN,NaN];    %   NaN's are returned if valid data not returned

if dim == 'h'|dim =='H'
    n=SPRING2_H(21);    %   Pointer to data location - see comment
above - must be included
    f=A{n};
end
%
if dim == 'v'|dim =='V'
    n=SPRING2_V(21);    %   Pointer to data location - see comment
above - must be included
    f=A{n};
end

return


function s=user1_names(dim,ident)
%   9/1/2007 - example of a function that gives names to fit parameters

E157_globals     %  Must be included - brings in shared variables



%   Braces {}'s should be used rather than parentheses ()'s to index
%       the array
s{1}=cellstr(['max ',dim, char(dia_acr(diagnostic))]);
s{2}=cellstr(['upper sigma',dim, char(dia_acr(diagnostic))]);
s{3}=cellstr(['lower sigma',dim, char(dia_acr(diagnostic))]);

return
```