GeoPIXE Overview

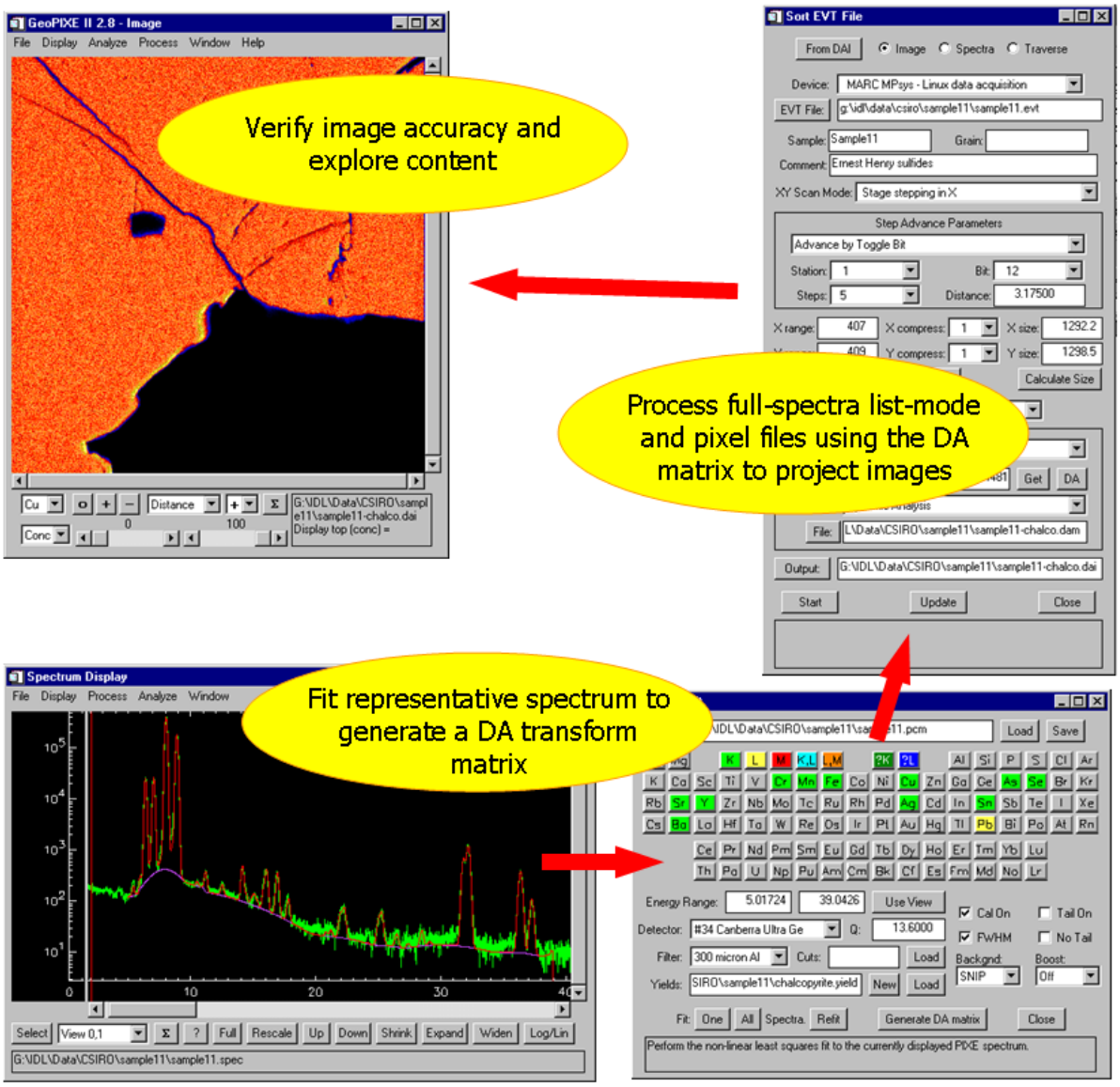
Contents

GeoPIXE workflow

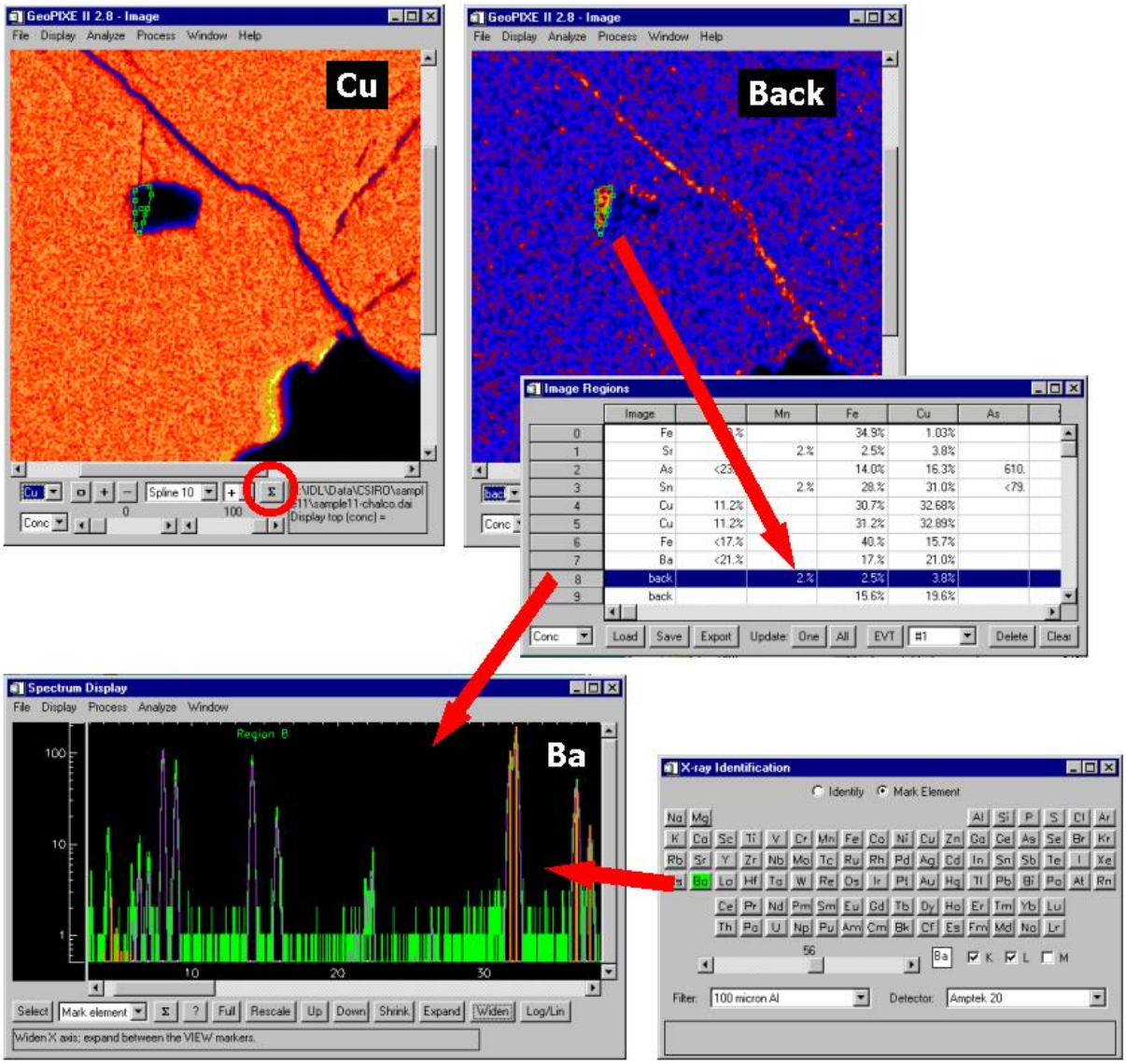
Interactive GeoPIXE

The data flow in GeoPIXE has two main phases: (i) transforming raw data into spectra or element images, and (ii) correction and exploration of these images, including extracting spectra from spatial (or concentration correlated) features seen in images (or element data). The first phase includes *Dynamic Analysis* (DA), or the construction of the matrix transform used in processing raw event data, which entails modelling the physics of the interaction of the beam particles (X-rays, protons) with the sample and how they are then detected in a detector (or array of detectors). The second phase includes a host of tools for exploration and viewing data in various ways as well as a number of tools for the correction of various artefacts caused by a range of instrumental issues.

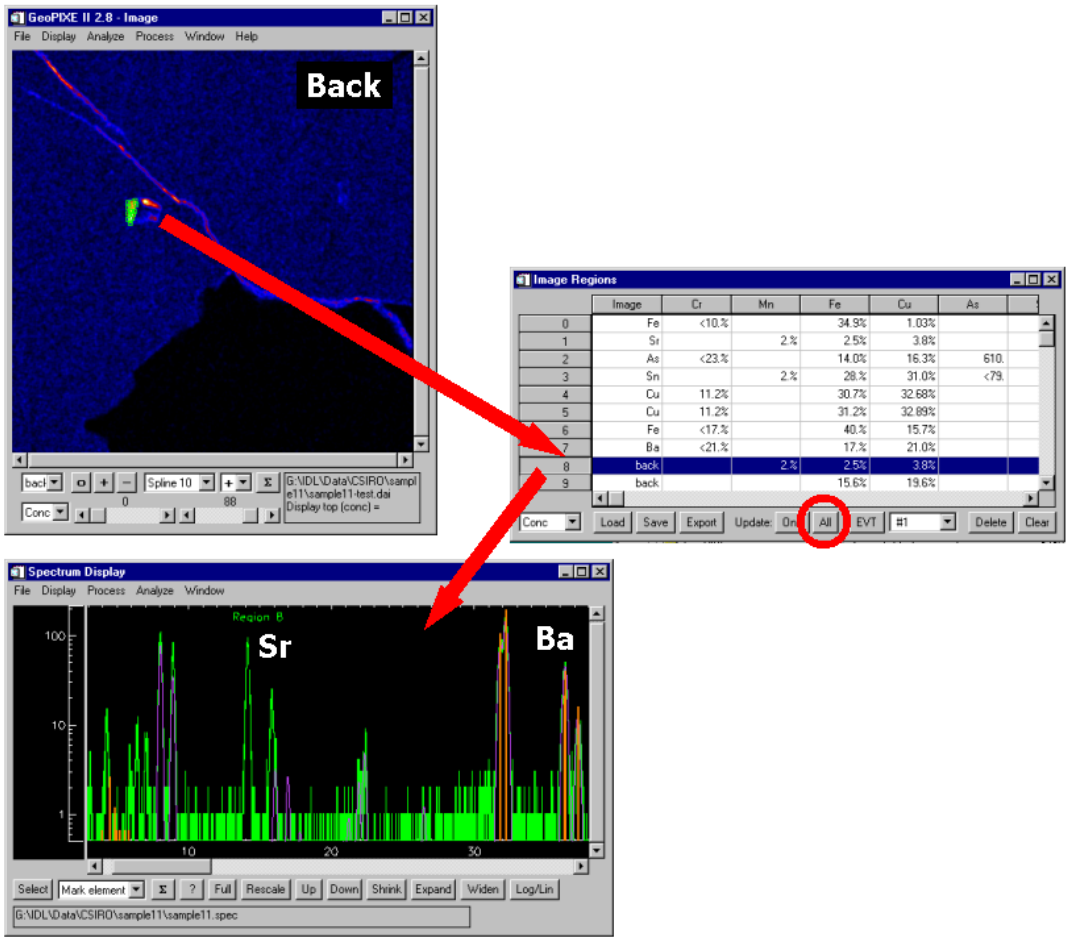
The workflow in GeoPIXE typically starts with fitting spectra (displayed in the *Spectrum Display* window and fitted using the *Xray Spectrum fit* window) to generate the *Dynamic Analysis* (DA) image projection matrix, using this DA matrix to process full-spectral data to deconvolute elemental components and project separated elemental images (displayed in the main *GeoPIXE Image* window) and exploration and processing of the images to first verify their accuracy and make corrections or to explore their content. This involves these windows (*Spectrum Display*, *X-ray Spectrum Fit*, *Sort EVT*, *Image*) …



Once images are created, the focus shifts to exploring the data, which often involves selecting pixels (in the *Image Regions* window) and determining the average concentrations for these pixels and extraction of the spectra from these pixels. Pixels can be selected using a simple shape (Box, Circle, Spline curves, etc.) or by using element-element correlations in the *Associations* window.



A common thing to look out for is “missing” elements. In other words, an element that was not included in the fit to build the DA matrix, but later found in one of the regions, usually by examination of the spectrum extracted for the region’s pixels. Often “missing” elements show up as extra signal, or hot spots, in the “Back” image (e.g. Sr in this example – note that the Sr peaks are not “explained” by the model shown in violet because Sr was missing from the fit to generate the initial DA matrix).



See the *GeoPIXE Users Guide* section "GeoPIXE Analysis Scenarios, Data Flow" for more details and then try the Demo data, following the detailed follow-along notes in the "***GeoPIXE Worked examples***".

Batch GeoPIXE

This is a bit of a misnomer. The *Batch Sort* in GeoPIXE is a window, which allows you to queue up a number of raw data-sets to be processed into images, with options to also do image processing (e.g. replicate image operations done on a previous image DAI file) on each image or generate derived outputs (e.g. RGB images, HTML web page summaries, TIFF files in various units, ASCII dumps, some image corrections, metadata files, etc.). It uses the settings in *Sort EVT* as a template (after successfully processing the first data-set, for example). *Batch Sort* displays a table which shows many image parameters (e.g. X,Y size) and input and output file names and paths and has buttons to enable image processing and output options.

GeoPIXE as a callable module in another workflow

Certain repetitive data-intensive operations in GeoPIXE can be performed with GeoPIXE acting as a callable module in some other workflow, such as triggered by data acquisition. The operations supported in this way are:

(i) Sorting raw data into images (i.e. as in *Sort EVT* window, evt.pro),
(ii) Extraction of spectra from image pixels selected in *Image Regions*, and
(iii) Export of image data (from DAI) to ZARR format for AWS data exploration and statistics.
(iv) Exporting image (DAI) metadata.
(v) Model results comparisons for testing (as a form of unit test).

GeoPIXE can now be called from the command-line of the O/S with additional parameters, which specify a "GeoPIXE Command File" and optionally input and output file-names to redirect the instructions in the

command file to process the 'input' file(s) and produce the 'output' file. This then can be used in a script, for example, to call GeoPIXE to process a given (set of) input file(s) in some automation scheme, perhaps triggered by the data acquisition system following the finishing of a run.

If the first argument is not a GCF file (no ".gcf" extension found), then the argument will be interpreted as a GeoPIXE procedure and executed, passing any supplied parameters.

The "geopixe" Linux script (without arguments launches GeoPIXE) accepts these new parameters ...

```
geopixe command-file input output
```

Arguments	contents
command-file	Full path to a **GeoPIXE Command File** (extension ".gcf"), which contains the name of a GeoPIXE command and all its parameters for execution, including "cluster=1" to trigger cluster-mode execution across multiple cores, if the data device supports it. Typically, enclose the GCF file-name (including full-path) in double quotes "", which enables paths with spaces to be used. The parameters in the GCF file are command dependent. The GCF files are simple ASCII text files, typically generated by GeoPIXE (e.g. "C*" buttons in the windows *Sort EVT*, *Image Regions, Image History* or the menu "*Create GeoPIXE Command File to Export as ZARR (C*)*" in the main *GeoPIXE Image* window. ***NOTE: If the first argument is not a GCF file (no ".gcf" extension found), then the argument will be interpreted as a GeoPIXE procedure and executed, passing any supplied parameters.***
input	An optional input data file-name (or list of file-names in "stringify" format) to replace the "file=" entry in the GCF file. There are three options here: 1. A simple data file-name (in "stringify" format, perhaps with a wildcard, enclosed in double quotes: "file1.*") 2. A file-name (preceded by a "@" character) of a text file (e.g. "@file"), which contains a list of all input files, one file per line. 3. A list of data file-names (in "stringify" format, i.e. as a string array or list with files enclosed in double quotes: ["file1","file2","file3", ...])
output	An optional output file-name to replace the "output=" entry in the GCF file. Generally, the 'input' and 'output' args would be set together. For 'input' file lists, the 'output' file would evolve following GeoPIXE rules for derived filenames.

Commands

imaging

The following commands generate 2D or 3D image files (extension .dai, .xan, respectively): 'da_evt' (2D image using a DA matrix), 'cut_evt' (2D image using CUTs), 'da_stack_evt' (3D image using a DA matrix and perhaps an energy table) and 'da_tomo_evt' (3D image, e.g. tomographic image). The GeoPIXE Command File can be generated using the "C*" button on the *Sort EVT* window, typically after successful processing of an initial data-set. This GCF file then provides a "template" for processing of further input data files, with the same process settings, specifying an output file for each.

region spectra extraction

The following commands generate a SPEC file containing an array of spectra: 'spec_evt' (spectra, one for each region in a REGION file). The GeoPIXE Command File can be generated using the "C*" button on the *Image Regions* window ("Extract" tab), typically after successful specifying regions on an image, for example

using the *Associations* window and the Hot-spot separation tools. This GCF file then provides a "template" for processing the input data files offline, specifying an output file. In this case, the spatial regions are very image data-set specific and would not generally be applied to a different image data-set.

export
Some simple export options are supported. Using the "C*" button on the *Image History* window will produce a GCF file for metadata output from images. The *Image History* window outputs GCF that uses the '**print_image_metadata**' command. It has only one parameter, aside from the "files=" and "output=" arguments, called "stats=", which can be used to optionally enable the output of image pixel statistics for the selected input file ("stats=1"). The second is a ZARR export, '**export_zarr**', which just uses the "files=" and "output=" arguments to dump an image in ZARR format for applications in the AWS cloud. If 'files' is not found, including a local dir tree search, it will pop-up a file requester to select it. This is less useful in a work-flow environment, so make sure the file path is correct.

Simple commands
If the first argument is not a GCF file (no ".gcf" extension found), then the argument will be interpreted as a GeoPIXE procedure and executed, passing any supplied parameters.

unit tests
Unit tests will be implemented as such a simple IDL procedure, which accept one argument (a vector ["old","new"]), the files to be compared, and an optional second argument, the name of the output file. These arguments follow the name of the procedure, which is used in place of a GCF file. The output report goes to a file given by the second argument, which defaults to the name of the new file (second file in vector argument) with extension .txt.

Routine	Function
Compare_yields	Compare two [old, new] yield files (.yield) for consistency
Compare_fits	Compare two [old, new] fit result files (.pfr) for consistency
Compare_source	Compare two [old, new] lab source files (.source) for consistency
Compare_pink	Compare two [old, new] pink beam files (.pink) for consistency

The idea is to generate new calculated model files (e.g. .yield) after changes to the code and then use a compare routine (e.g. "compare_yields") to test the output for consistency with an original output file. This can be done directly in the Eclipse IDLDE environment (just click run and it will prompt for a "Reference" file and a "New" file to be compared with the Reference), or using this *callable GeoPIXE* approach.

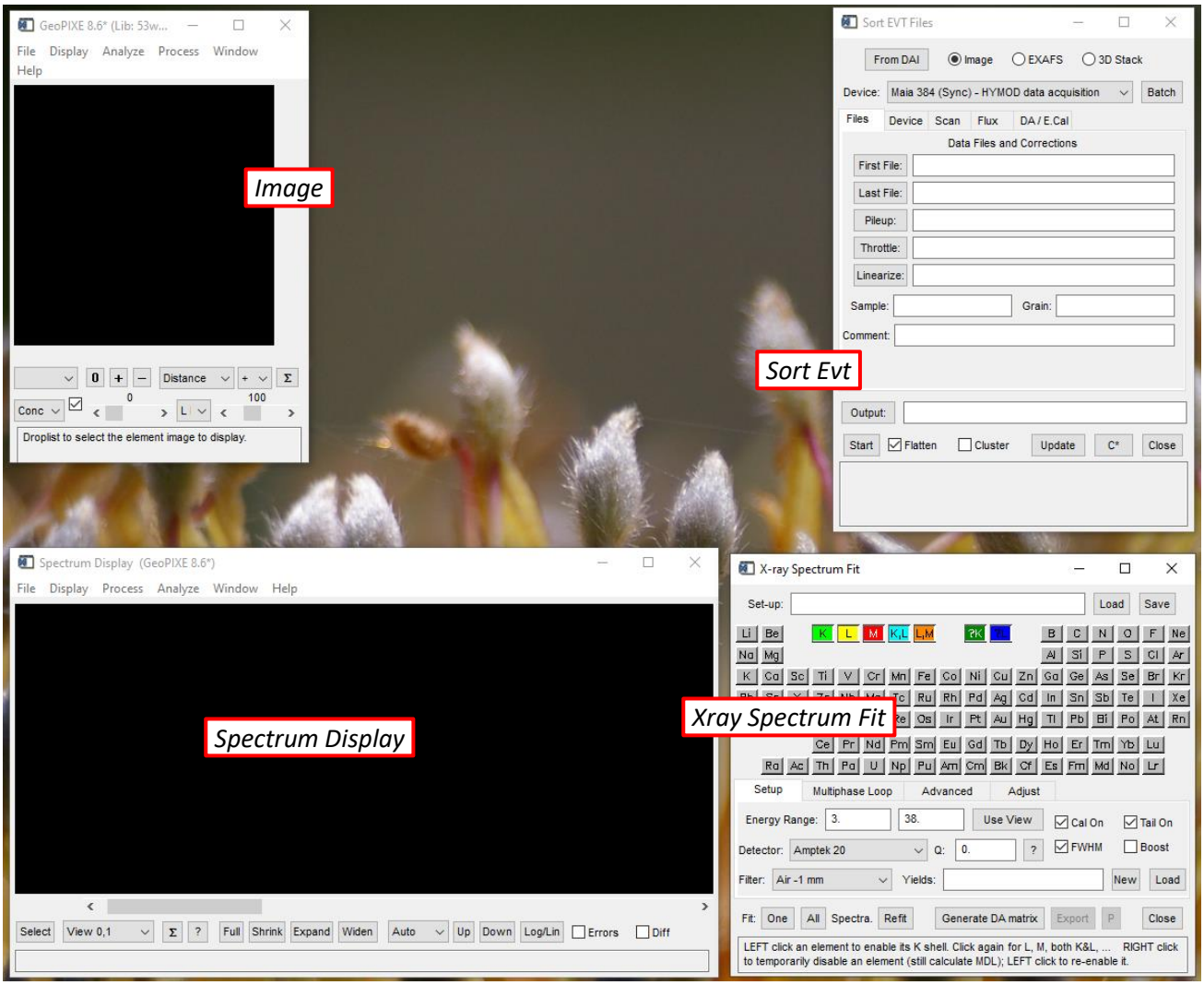
GeoPIXE Windows

Typical windows at startup

Some window can be flagged to be opened on GeoPIXE startup, using the "startup" entries in the "geopixe.conf" file.

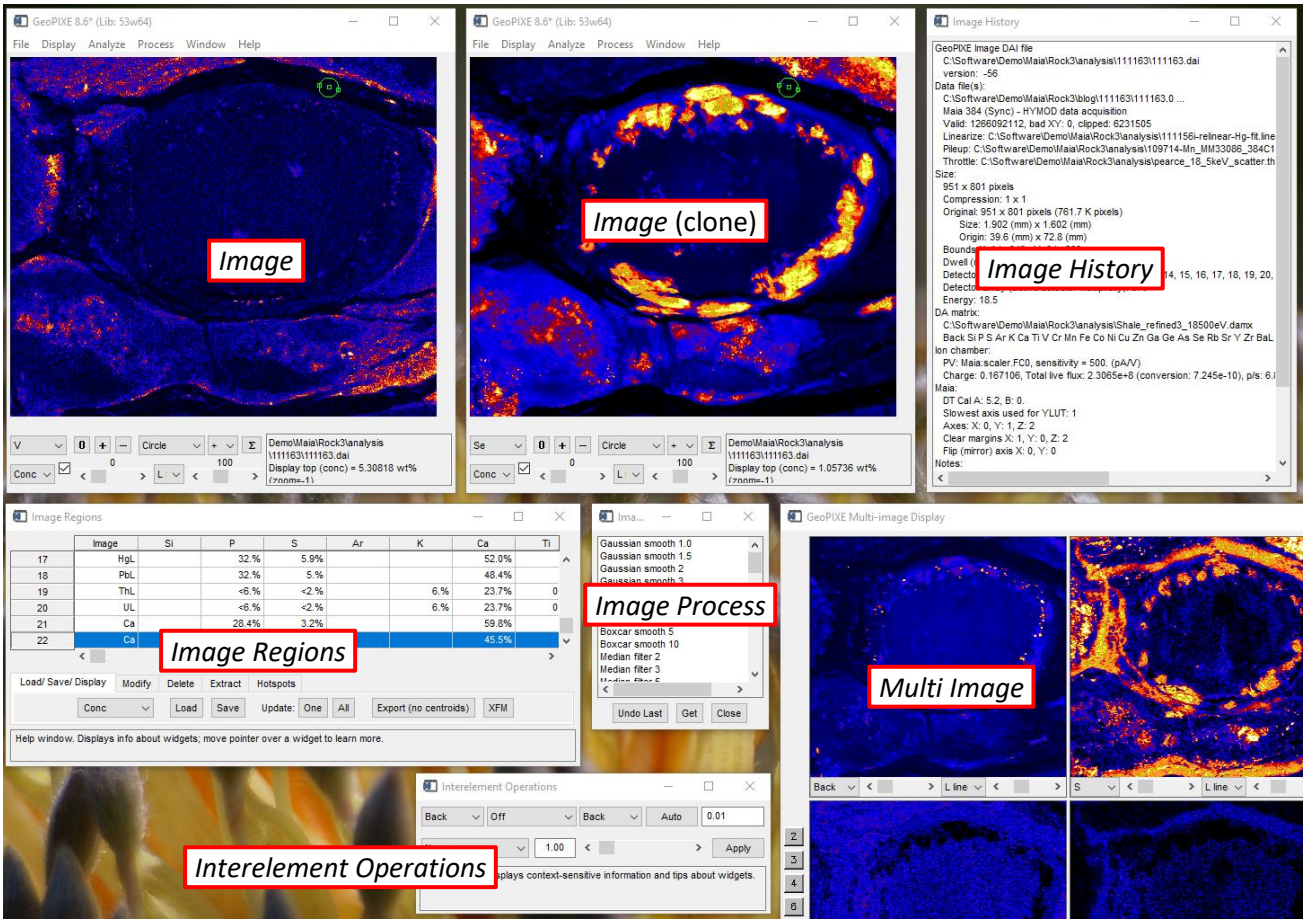
```
#
# startup
#       flags windows to open after GeoPIXE is started
#
        startup image_clone     0               # Extra clone of image window
        startup regions         0               # image regions window
        startup spectrum        1               # spectrum display
        startup identify        0               # X-ray line identification window
        startup fit             1               # Xray spectrum fit
        startup sort            1               # Sort EVT window
        startup select          1               # Spectrum select window
#
```

"*Image*" is the main GeoPIXE window. *Spectrum Display* is launched from *Image* for display of (multiple) spectra. Multiple instances of each can be opened (e.g. use "display→clone" menu in "image" to open image clones or select from the "Windows" menus in *Image* or *Spectrum Display*). *Sort EVT* is the window controlling sorting of raw data into images. "Xray spectrum fit" is for fitting spectra and generating a DA matrix for imaging. A number of other windows are launched from *Xray spectrum fit* (see below).

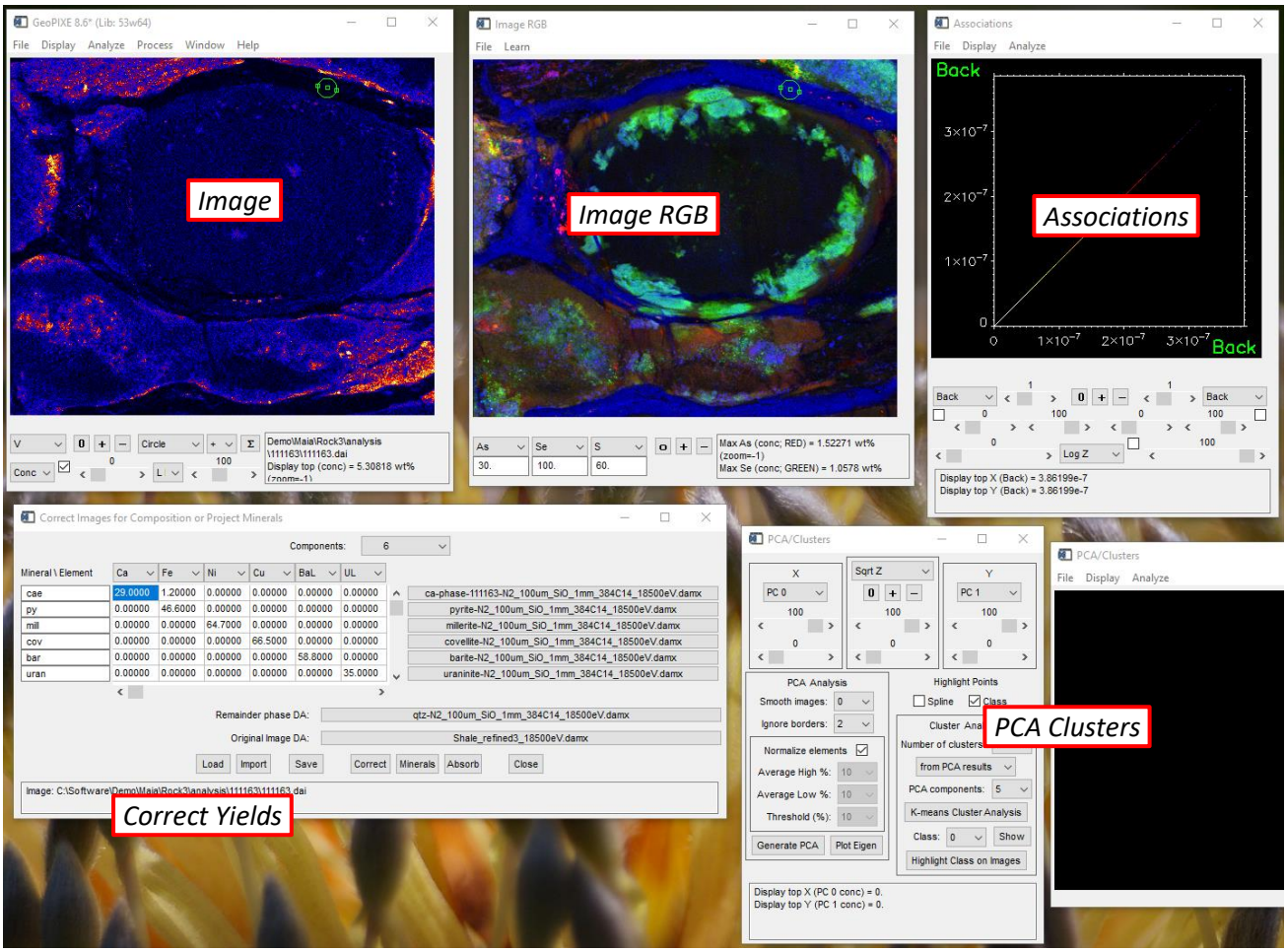


Windows launched from *Image*

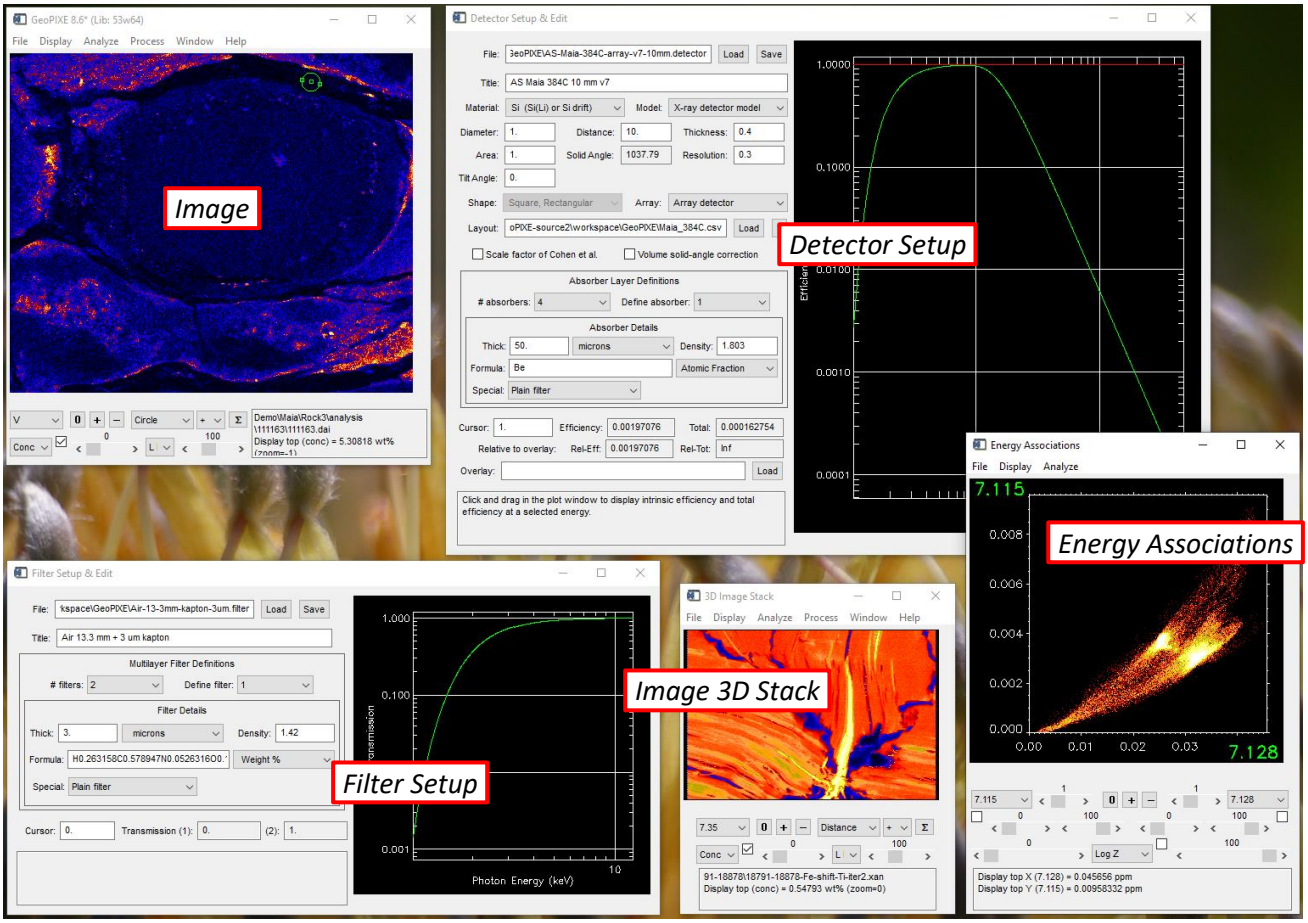
A number of windows are launched from *Image*, mostly for display of image data in various ways, to look at Image regions of interest (shapes on the image that define selected pixels, or pixels selected by element-element *Association*) or to apply processing or corrections to images. This first set shows a clone of *Image*, *Image History*, which shows details of a loaded image, *Image Regions*, which lists image regions applied to an image, *Image Process*, which provides a one-click list of operations, digital filters, etc. to apply to an image, *Multi Image*, which shows a grid of many simple image windows and *Interelement Operations*, which permits correction of artefacts caused by interaction between elements/ images.



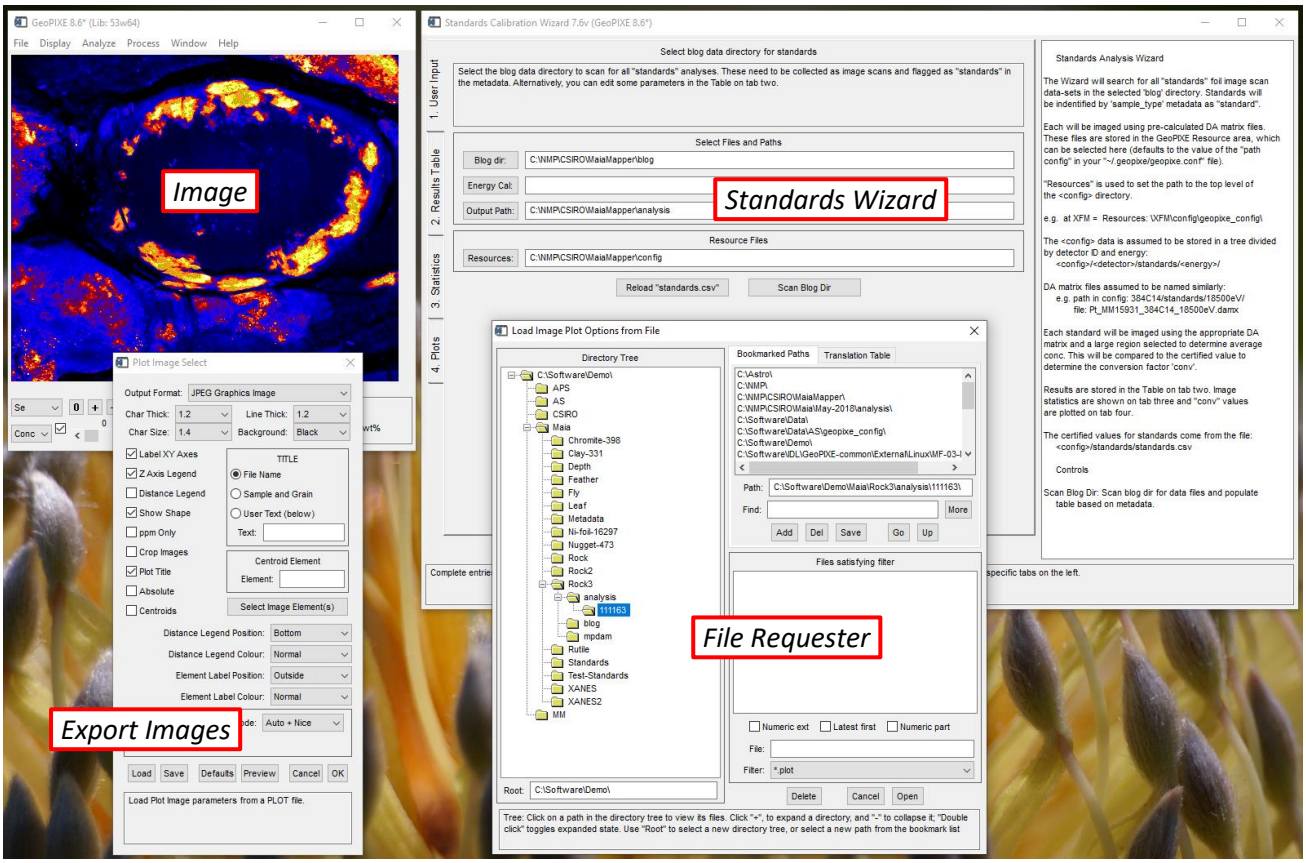
The second set of windows launched from gimage.pro shows *Image RGB*, which displays 3 element planes as Red, Green, Blue to make a 24-bit false-colour image, *Associations*, which displays the *Association* window to plot the image pixel data for 2 selected elements as a scatter plot/2D histogram (colour provides histogram intensity axis), *Correct Yields*, which provides a tool for correction of images for complex matrix effects and *PCA Clusters*, which provides a tool for simple PCA and cluster analysis (K-means).



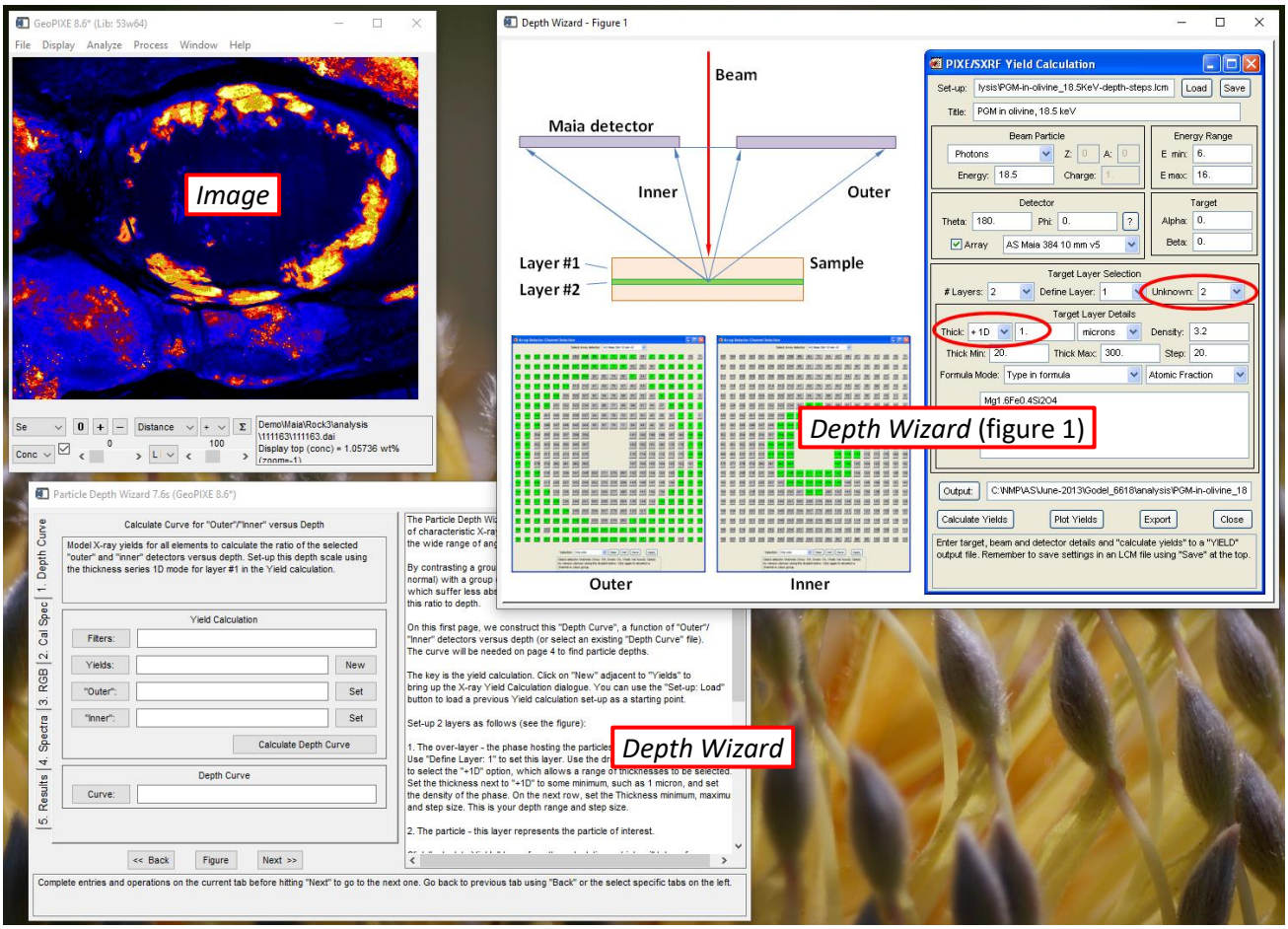
The third set of windows launched from *Image* shows *Detector Setup*, which defines the parameters for a detector specification, *Filter Setup*, which defines the parameters for a filter stack, *Image 3D Stack*, which opens a 3D stack for XANES images and an *Energy Association* window, which is to display a scatter plot/2D histogram of the association between energy planes in a XANES image stack or Line-XANES data.



The fourth set of windows launched from *Image* shows *Standards Wizard*, which is a Wizard for processing standards samples for flux calibration, *Export Images*, which is a modal/blocking popup to setup for the export of image data as a plot with axes, etc. and the “file requester”, which is also a modal/blocking widget used throughout GeoPIXE for file browsing and selection.

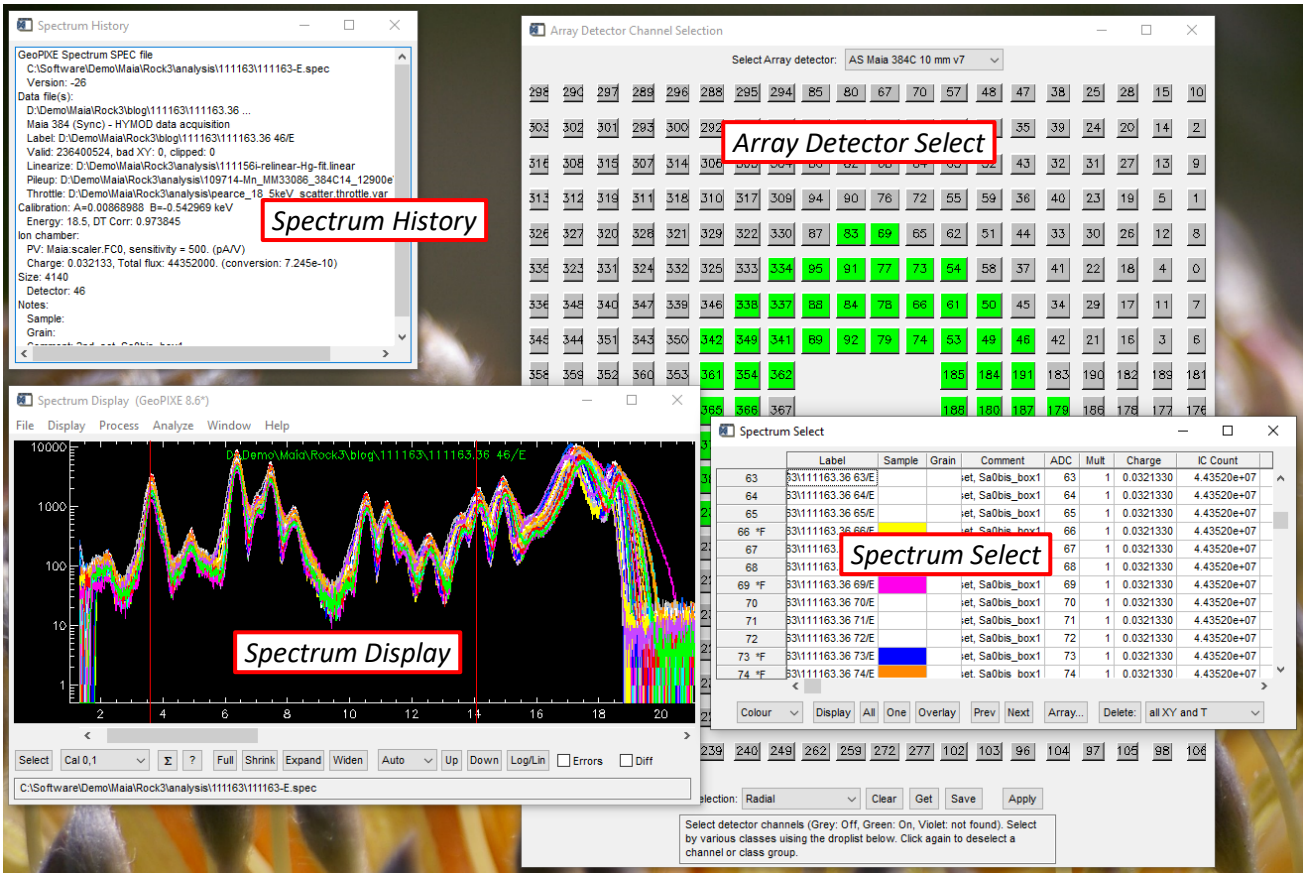


The fifth set of windows launched from *Image* shows *Depth Wizard*, which guides the user through depth analysis using the Maia detector array geometry (it also displays instructions and pops up figures to illustrate the process).

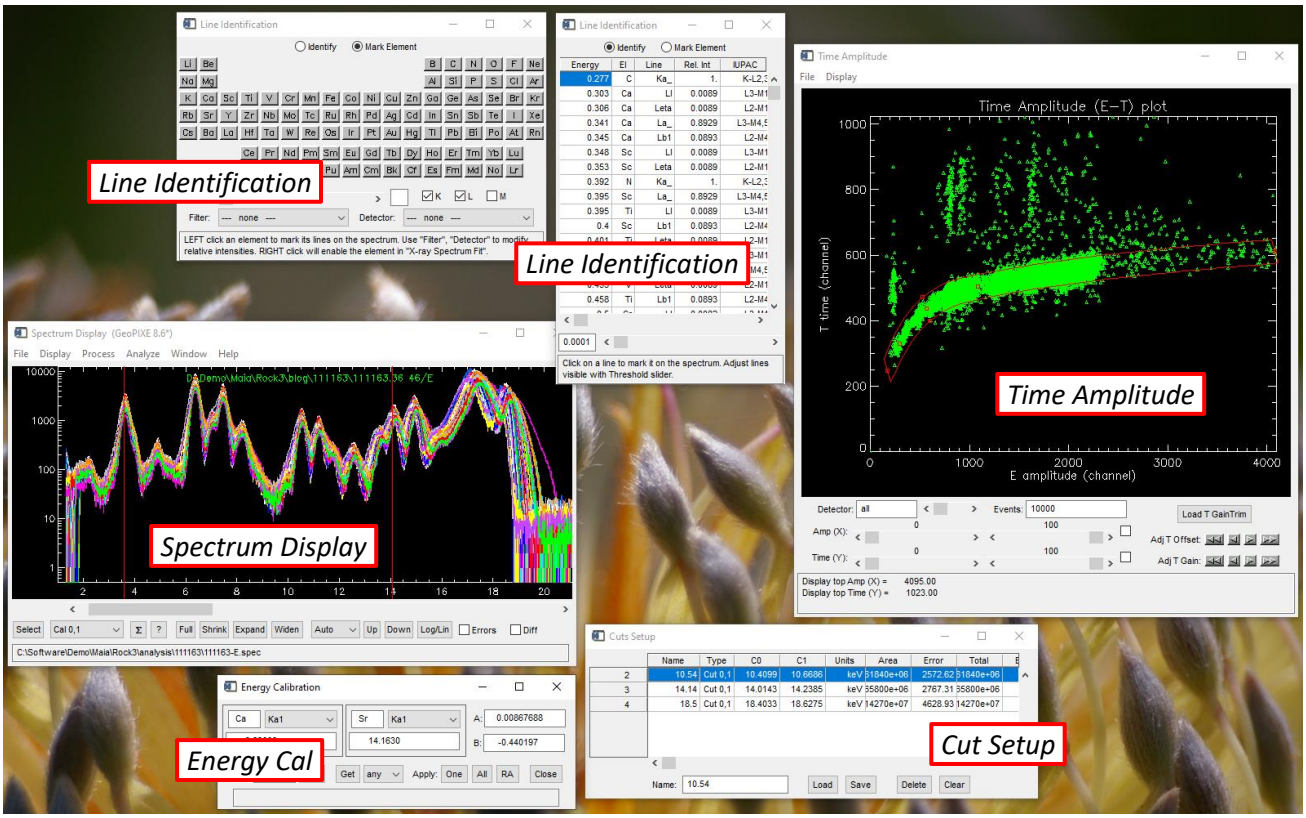


Windows launched from *Spectrum Display*

The first set of windows launched from *Spectrum Display* shows *Spectrum History*, which shows details for spectral data, *Spectrum Select*, which shows a list of spectra loaded and enables selection of selected spectra and *Detector Select*, which is launched from *Spectrum Select* using “Array …” button, to permit detailed selection of members of the detector array.

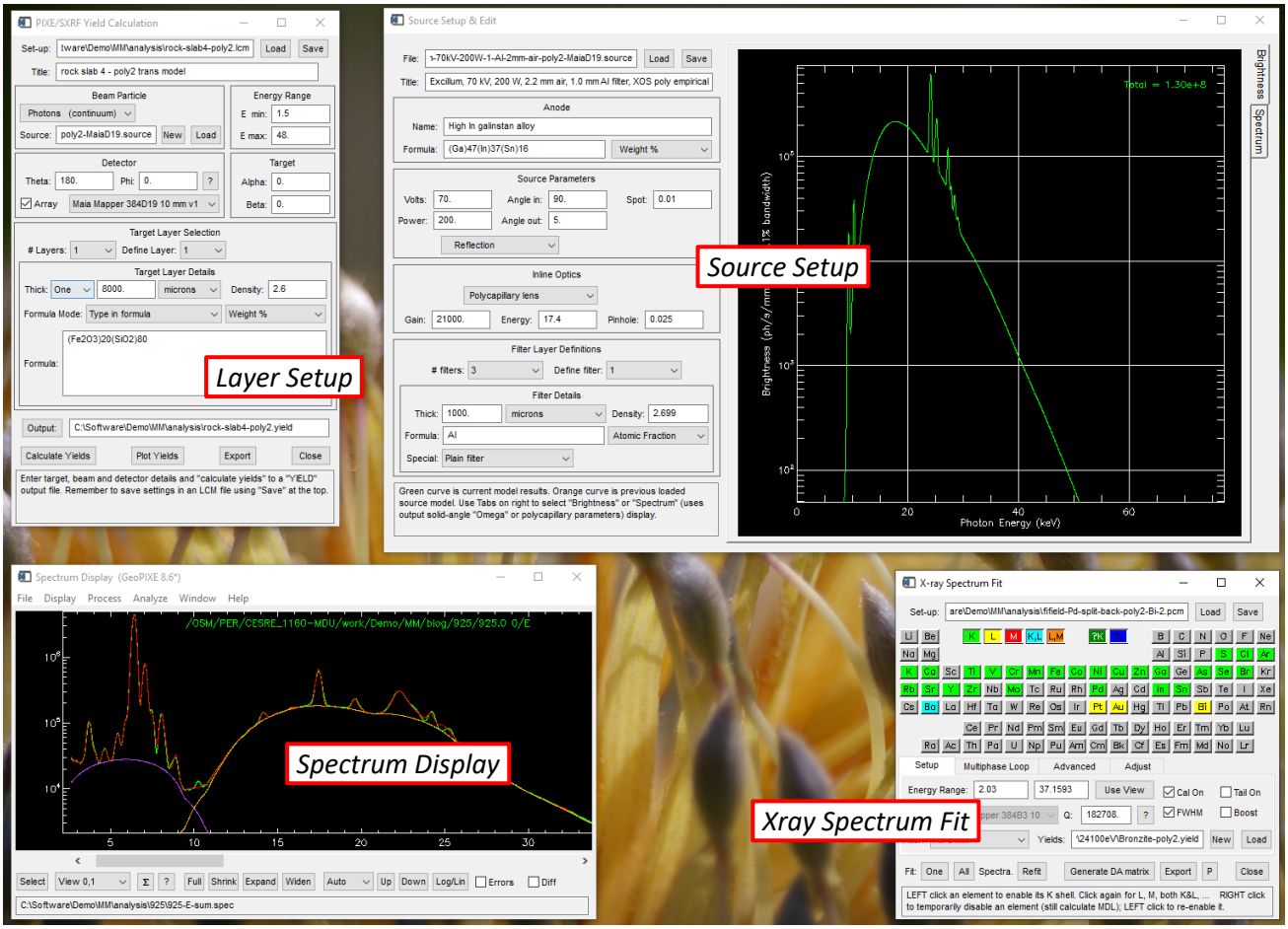


The second set of windows launched from *Spectrum Display* shows *Identify* shown in two instances as a periodic table to mark lines for an element on the spectrum and as a list to search for a match in the sorted list of X-ray lines, *Time Amplitude*, which displays raw Maia events as a scatter plot of E versus T to enable a pileup field to be defined, *Energy Cal*, which permits energy calibration of spectra and *CUT Setup*, which allows portions of spectra to be defined as CUTs using X (with background subtraction) or Cut (simple energy range) markers.



Windows launched from *Xray Spectrum Fit*

The set of windows launched from *Xray Spectrum Fit* (which is launched from *Spectrum Display)* shows *Layer Setup*, which defines a layered sample structure and calculates yields and X-ray line intensities and *Source Setup*, which defines a laboratory X-ray source model and calculates a continuum (plus characteristic anode lines) spectrum to be used in the yield calculation. *Source Setup* is launched from *Layer Setup*.



GeoPIXE from GitHub

The archive on GitHub contains all source files and essential data and documentation. It does NOT include a metadata (Workspace/.metadata directory) for the IDL DE Eclipse environment. What is missing are Eclipse project definitions and the build settings for each project. Import of the projects is easy (see below). But we must set a couple of settings regarding management of the !path in IDL. These are detailed below.

Building can be easily handled using “Builder.pro” (see below). However, you can setup Eclipse IDL build settings if you like, as described in section “Eclipse environment and organization”.

Downloading

If you are reading this, then perhaps this is done for the software… But for worked example data, you need to also download the Demo data, which is archived in the CSIRO DAP. There are two archives, a simple one (e.g. for a Windows system at DOI: https://doi.org/10.25919/ff5b-wr11) and one designed to serve multiple users in a Linux workshop environment (at DOI: https://doi.org/10.25919/3yrz-7x38). See the “Read me.txt” files for details.

NOTE: See the Readme file notes on how to get the ***GeoPIXE Demo data*** for the worked example tutorials.

Install IDL

Running GeoPIXE requires at least IDL runtime support. With IDL installed, but not licensed, you can run GeoPIXE using IDL *Virtual Machine* runtime support. To program in IDL and build GeoPIXE source, you will need an IDL license.