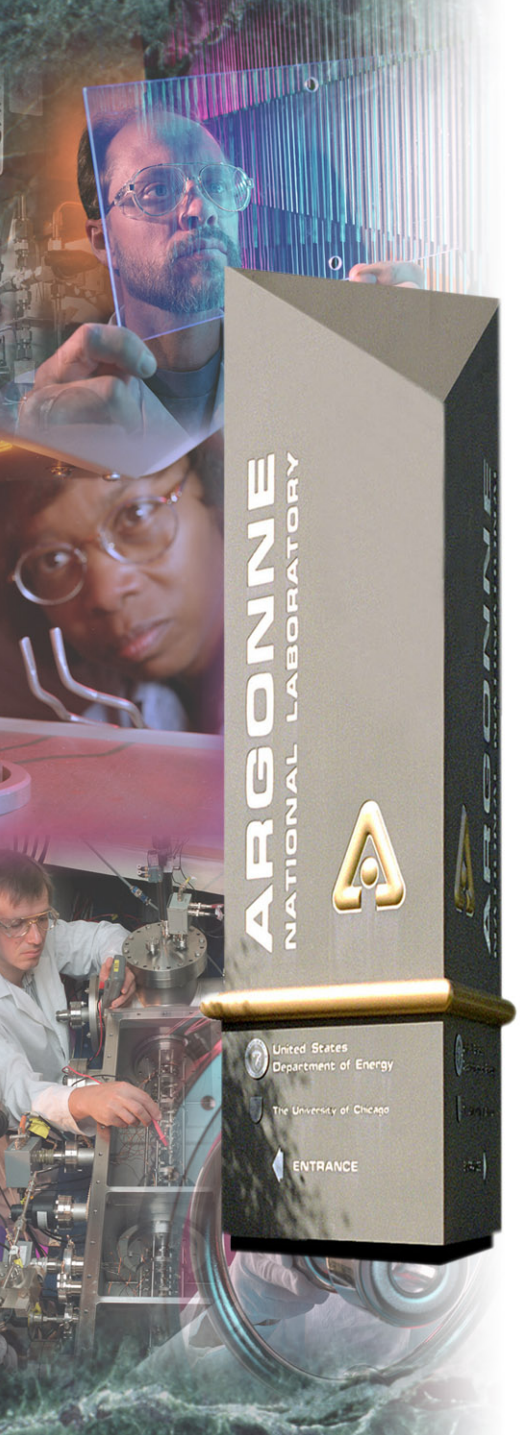


Python Applications for Beamline Control

Mark Rivers

Argonne National Laboratory

Office of Science
U.S. Department of Energy

A U.S. Department of Energy
Office of Science Laboratory
Operated by The University of Chicago

Motivation

- **Replace IDL applications (e.g. MCA GUI) with Python so that other beamlines don't need to buy IDL**
- **Send users home with data and display/analysis programs that are free.**
  - They don't want to buy IDL.

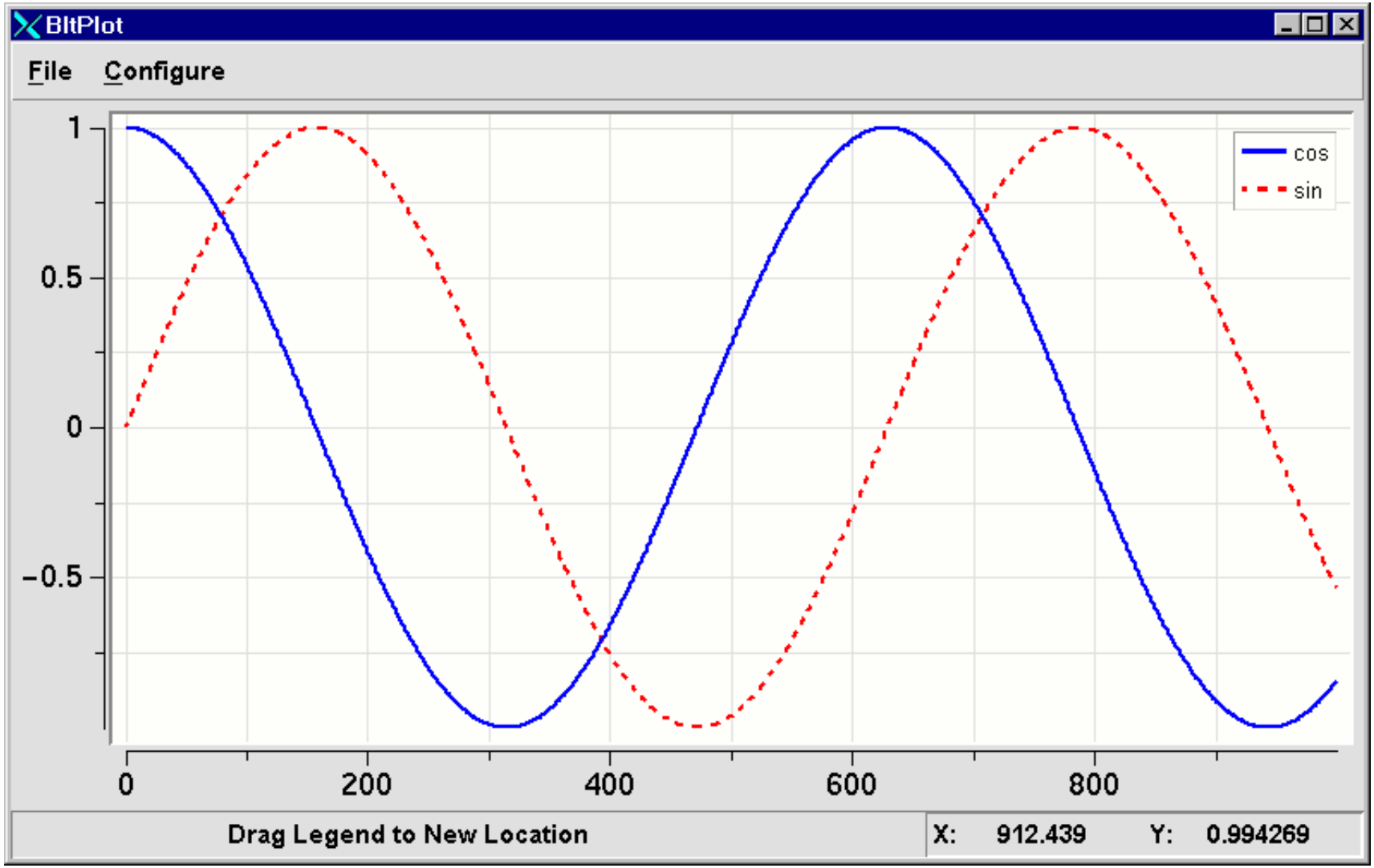
Building Blocks

- **CaChannel from Geoff Savage for EPICS interface**
- **Tkinter and Pmw for GUIs**
- **Pmw.Blt for plots**
- **Numeric for arrays**

BltPlot: Enhancements to the Pmw.Blt.Graph widget

- **A standalone plotting widget, BltPlot.BltPlot. This widget has menus to:**
  - Configure all of the plot characteristics
  - Save and restore the plot settings and data
  - Print the plot to a file or printer
- **Methods (BltPlot.plot and BltPlot.oplot) to create a new plot, to overplot more data, etc.**
- **Designed to provide a rough emulation of the command line plotting capabilities of IDL.**

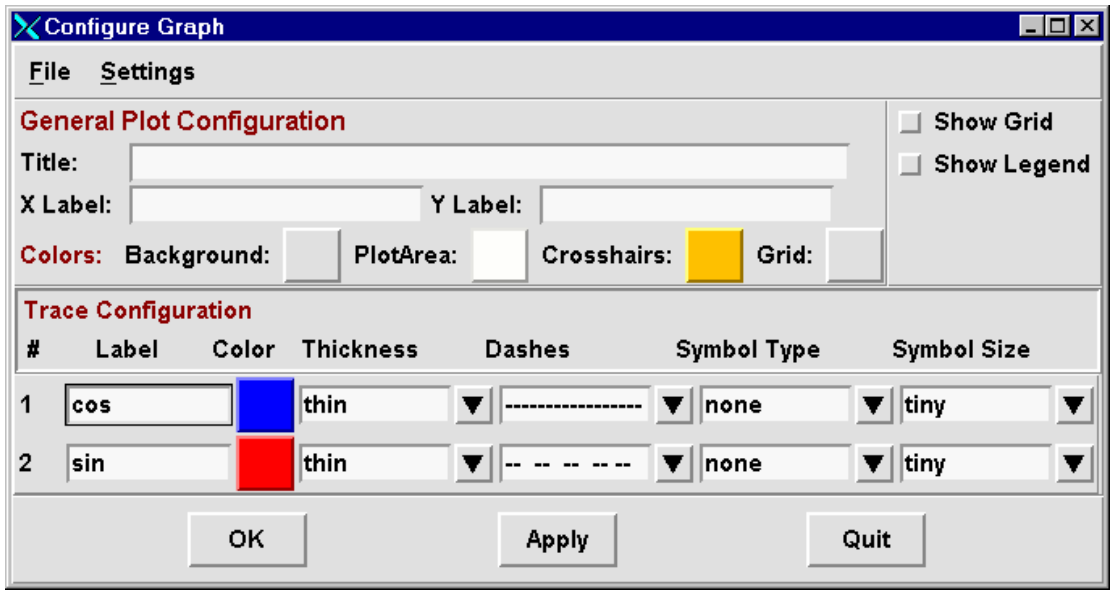
BltPlot

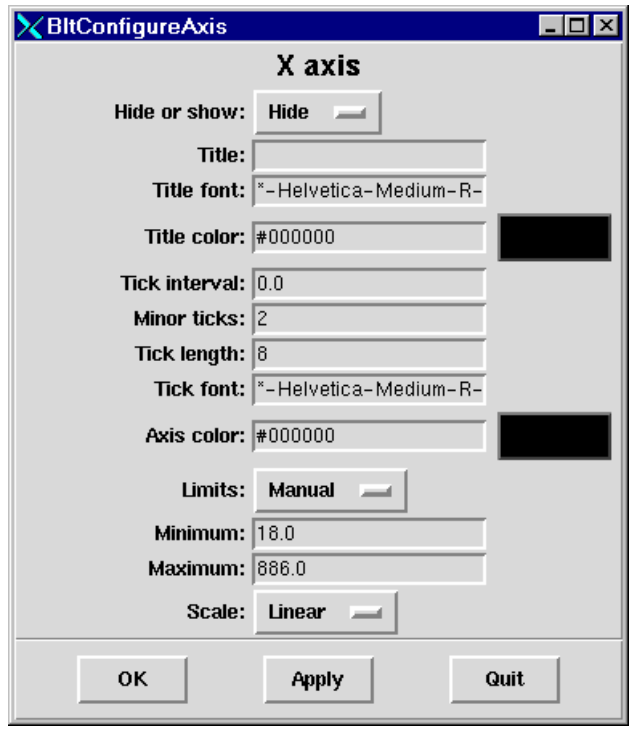

BltPlot
File
Configure
1
0.5
0
−0.5
0
200
400
600
800
cos
sin
Drag Legend to New Location
X: 912.439
Y: 0.994269

BltPlot: Enhancements to the Pmw.Blt.Graph widget

- **GUI routines to configure all of the plot characteristics, such as axes, markers, legends, etc.**
  - These routines work with any Pmw.Blt.Graph instance so they can be used from the standalone plotting widget in this package (BltPlot.BltPlot) or from any application that uses the Pmw.Blt.Graph widget
- **Routines to save and restore plot settings and data.**
- **Used in the mcaDisplay described later.**

BltPlot Dialogs





epicsPV: Subclasses Geoff Savage's CaChannel class

- **If a PV name is given then the class constructor will do a searchw() by default.**
- **setMonitor() sets a generic callback routine for value change events. Subsequent getw(), getValue() or array_get() calls will return the value from the most recent callback, and hence do not result in any network activity or latency. This can greatly improve performance.**
- **checkMonitor() returns a flag to indicate if a callback has occurred since the last call to checkMonitor(), getw(), getValue() or array_get(). It can be used to increase efficiency in polling applications.**

epicsPV

- **getControl() reads the "control“ and other information from an EPICS PV without having to use callbacks. In addition to the PV value, this will return the graphic, control and alarm limits, etc.**
- **putWait() calls array_put_callback() and waits for the callback to occur before it returns. This allows programs to use array_put_callback() synchronously and without user-written callbacks.**

epicsMotor

- **Class library for EPICS motor record**
- **Methods:**
  - move(), stop(), wait(), get_position(), set_position()
- **Virtual attributes:**
  - slew_speed, base_speed, high_limit, low_limit, done_moving, backlash, resolution, etc.
- **Example use:**

```
from epicsMotor import *
m = epicsMotor('13LAB:m5')
m.move(10.)
m.wait()
m.get_position(dial=1, readback=1)
9.9609375
```

epicsScaler

- **Class library for EPICS scaler record**
- **Methods:**
  - start(), stop(), read(), wait(), get_label(), set_label()
- **Example use:**

```
>>> from epicsScaler import *
>>> s = epicsScaler('13LAB:scaler1')
>>> s.get_counts()
>>> s.read()
[0, 0, 0, 0, 0, 0, 0, 0]
>>> s.start(1.)
>>> s.wait()
>>> s.read()
[10000000, 0, 0, 0, 0, 0, 0, 0]
```

epicsLogger

GUI appplication for logging EPICS PVs to the screen and to a disk file

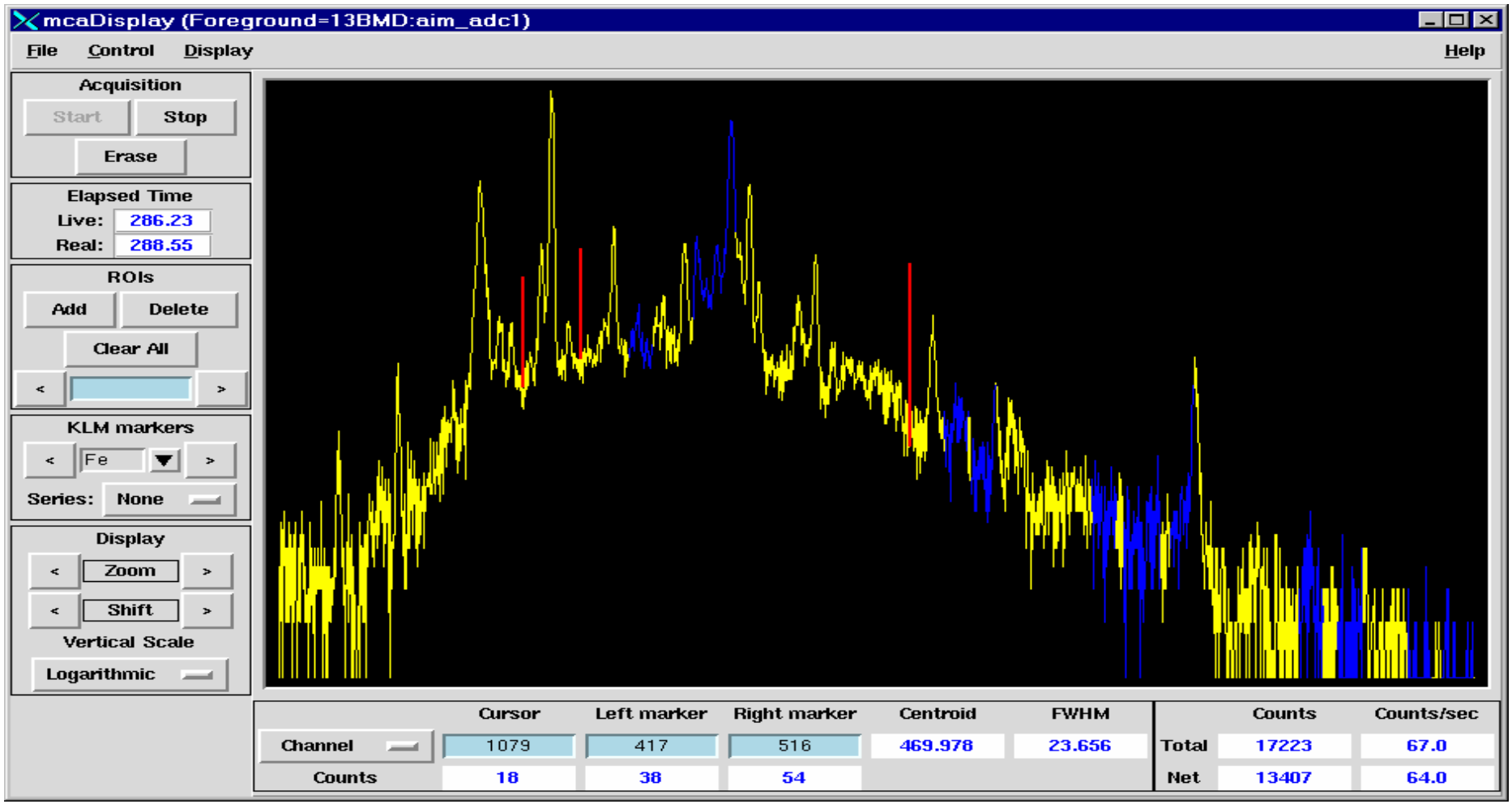
epicsLogger

File Help

Start Stop Update time (seconds) 20 Output file: None

Date and time	13BMD:DMM1Ch2_calc.VAL Load, Tons	13BMD:DMM1Ch4_calc.VAL Ram Ht, mm	room temperature, C	13BMD:DMM1Ch1_calc.VAL Sample Tc, mV	13BMD:DMM1Ch7_calc.VAL T (C)	13BMD:D Volts
26-Mar-2003 19:30:27	0.43917	-25.16517	34.87248	-81.71421	-4939.15607	0.03511
26-Mar-2003 19:30:37	0.33953	-25.16598	34.87110	-1.42667	-87.51303	0.03511
26-Mar-2003 19:30:47	0.44808	-25.16619	34.86937	-31.10073	-2608.25114	0.03513
26-Mar-2003 19:30:57	0.29490	-25.16464	34.86736	-54.78670	-4882.51811	0.03514
26-Mar-2003 19:31:07	0.44804	-25.16483	34.86229	-61.13838	-5287.09796	0.03514
26-Mar-2003 19:31:17	0.29018	-25.16452	34.86673	-56.22646	-4987.64225	0.03513
26-Mar-2003 19:31:20	0.29018	-25.16452	34.86673	-56.22646	-4987.64225	0.03513
26-Mar-2003 19:31:22	0.29018	-25.16452	34.86673	-56.22646	-4987.64225	0.03513
26-Mar-2003 19:31:24	0.47972	-25.16535	34.86702	-53.39937	-4774.91081	0.03513
26-Mar-2003 19:31:26	0.47972	-25.16535	34.86702	-53.39937	-4774.91081	0.03513
26-Mar-2003 19:31:28	0.47972	-25.16535	34.86702	-53.39937	-4774.91081	0.03514
26-Mar-2003 19:31:30	0.47972	-25.16535	34.86702	-53.39937	-4774.91081	0.03514
26-Mar-2003 19:31:32	0.47972	-25.16535	34.86702	-53.39937	-4774.91081	0.03514
26-Mar-2003 19:31:34	0.28039	-25.16466	34.86693	-36.80112	-3194.25474	0.03514

mcaDisplay


mcaDisplay (Foreground=13BMD:aim_adc1)
File
Control
Display
Help
Acquisition
Start
Stop
Erase
Elapsed Time
Live: 286.23
Real: 288.55
ROIs
Add
Delete
Clear All
KLM markers
Fe
Series: None
Display
Zoom
Shift
Vertical Scale
Logarithmic
Cursor
Left marker
Right marker
Centroid
FWHM
Channel 1079 417 516 469.978 23.656
Counts 18 38 54
Counts
Counts/sec
Total 17223 67.0
Net 13407 64.0

mcaDisplay()

- **Replacement for my IDL MCA display program**
- **Much nicer in many respects, since the Blt plot widget has many more east-to-use features than IDL's direct graphics**
- **Python object with callable methods, so it can be remotely controlled**
- **Device independent. It reads files and controls the "hardware_mca" class. "hardware_mca" can be subclassed for any hardware. Presently the EPICS MCA record is supported**

Mca: Device-independent MCA class

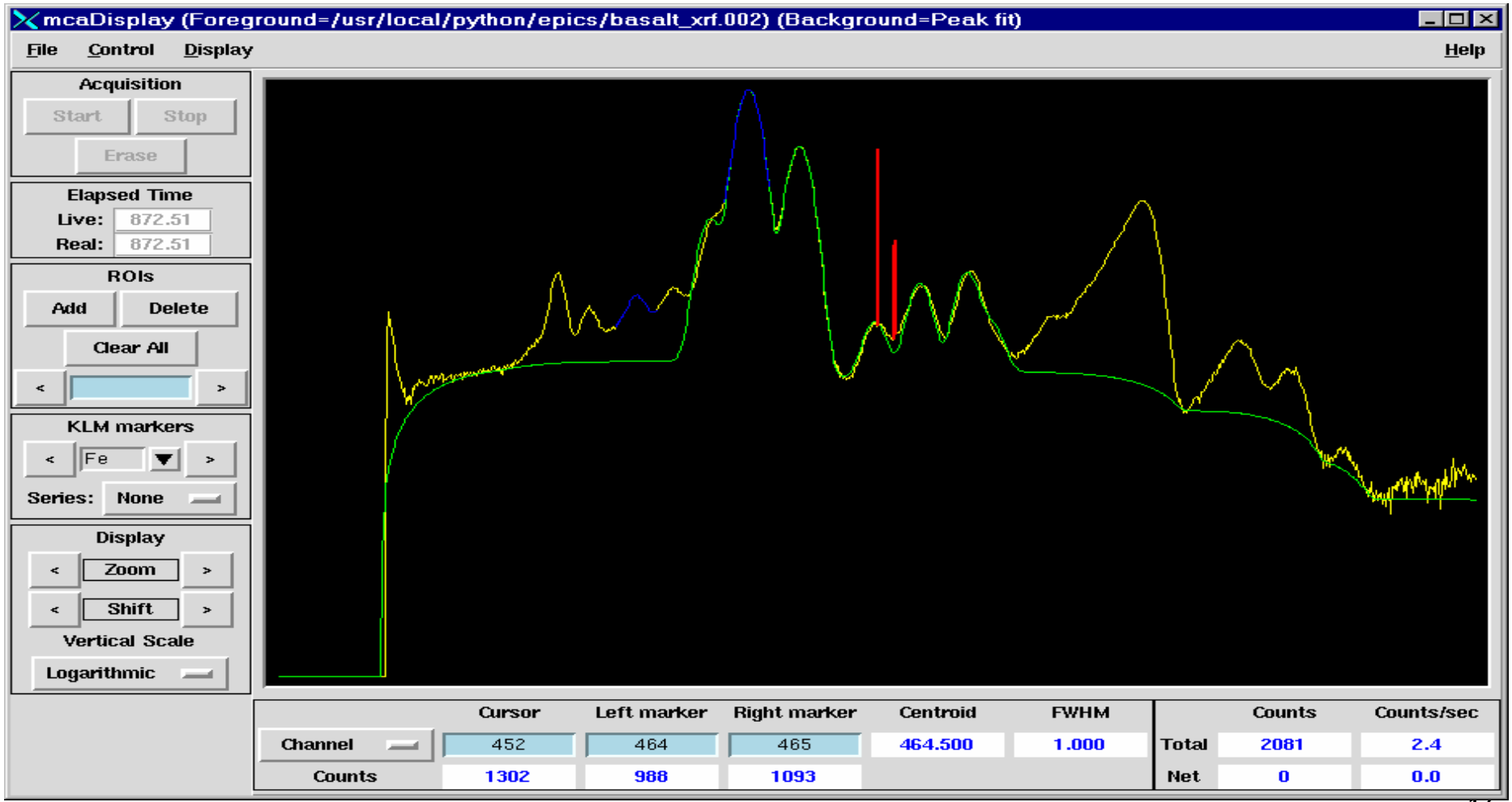
- **Support classes: mcaROI, mcaCalibration, mcaElapsed, mcaPresets, mcaPeak, etc.**
- **Many methods: add_roi(), fit_background(), fit_peaks(), get_calibration(), set_calibration(), write_file(), read_file(), etc.**
- **Used as base class of epicsMca.**

epicsMca:
Subclass of hardwareMca, which is subclass of Mca

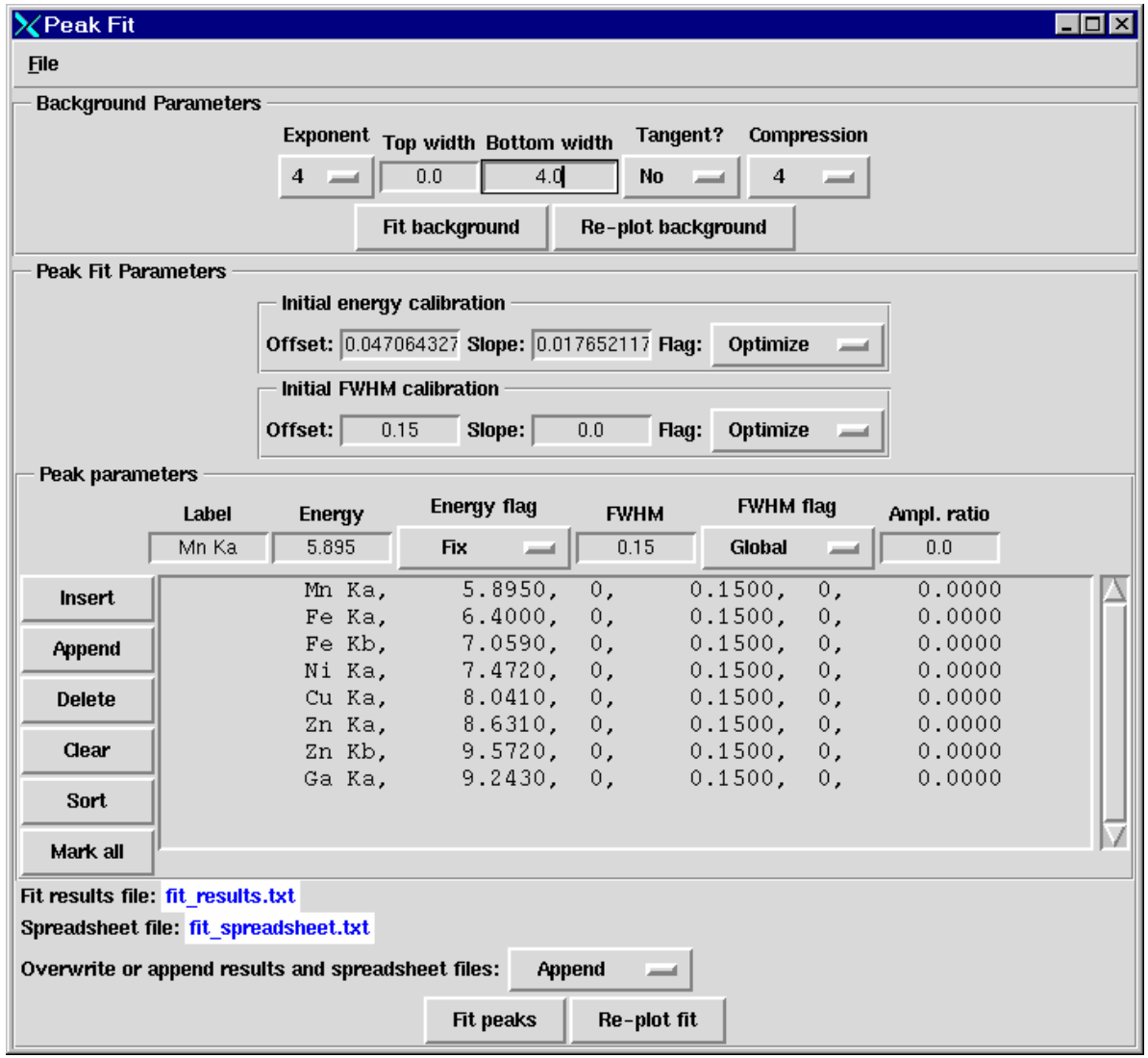
- **All methods of Mca, plus start(), stop(), erase(), wait(), etc.**
- **Re-implements base class routines for set_calibration(), set_rois(), etc. to communicate with fields in the EPICS MCA record**
- **Example use:**

```
from epicsMca import *
mca = epicsMca('13BMD:aim_adc1')
mca.erase()
mca.start()
mca.wait()
mca.write_file('test.001')
```

mcaPeakFit


mcaDisplay (Foreground=/usr/local/python/epics/basalt_xrf.002) (Background=Peak fit)
File
Control
Display
Help
Acquisition
Start
Stop
Erase
Elapsed Time
Live: 872.51
Real: 872.51
ROIs
Add
Delete
Clear All
KLM markers
Fe
Series: None
Display
Zoom
Shift
Vertical Scale
Logarithmic
Cursor
Left marker
Right marker
Centroid
FWHM
Channel
452
464
465
464.500
1.000
Counts
1302
988
1093
Counts
Counts/sec
Total
2081
2.4
Net
0
0.0

mcaPeakFit Parameters



mcaPeakFit Output

mcaPeakFit Results

```
*******************************************************
    Fit of /usr/local/python/epics/basalt_xrf.002:3

Real time (seconds):                    872.51
Live time (seconds):                    872.51
Initial FWHM offset, slope:           0.150000  0.000000
Optimized FWHM offset, slope:         0.305848  0.002562
Initial energy offset, slope:         0.047064  0.017652
Optimized energy offset, slope:      -0.046042  0.017947
# Iterations, function evals:                8        93
Chi squared:                        5.92047e+09
Status code:                        2
Time to fit:                     3.246
```

Peak	Energy	FWHM	Area	Background	Area/MDL	Area/Bkg
Mn Ka	5.895	0.3121	204198.9	19513.0	487.3	10.5
Fe Ka	6.400	0.3123	3079580.9	19196.0	7409.1	160.4
Fe Kb	7.059	0.3127	949900.2	17498.0	2393.7	54.3
Ni Ka	7.472	0.3129	0.0	15107.0	0.0	0.0
Cu Ka	8.041	0.3131	17554.6	15052.0	47.7	1.2
Zn Ka	8.631	0.3134	49345.0	15625.0	131.6	3.2
Zn Kb	9.572	0.3138	14772.8	16192.0	38.7	0.9
Ga Ka	9.243	0.3136	63076.2	15686.0	167.9	4.0

OK

Mpfit

- **Generalized non-linear least squares data fitting**
- **Based on LMFIT from Minpack**
- **Originally translated to IDL by Craig Markwardt, I translated to Python**
- **Much faster and more accurate than the version provided in the Scientific Python package in Scientific.Functions.LeastSquares.**
- **Contraints, fixed parameters, analytic or numerical derivatives, etc.**
- **Used in mcaPeakFit**

Med: Device-independent multi-element detector class

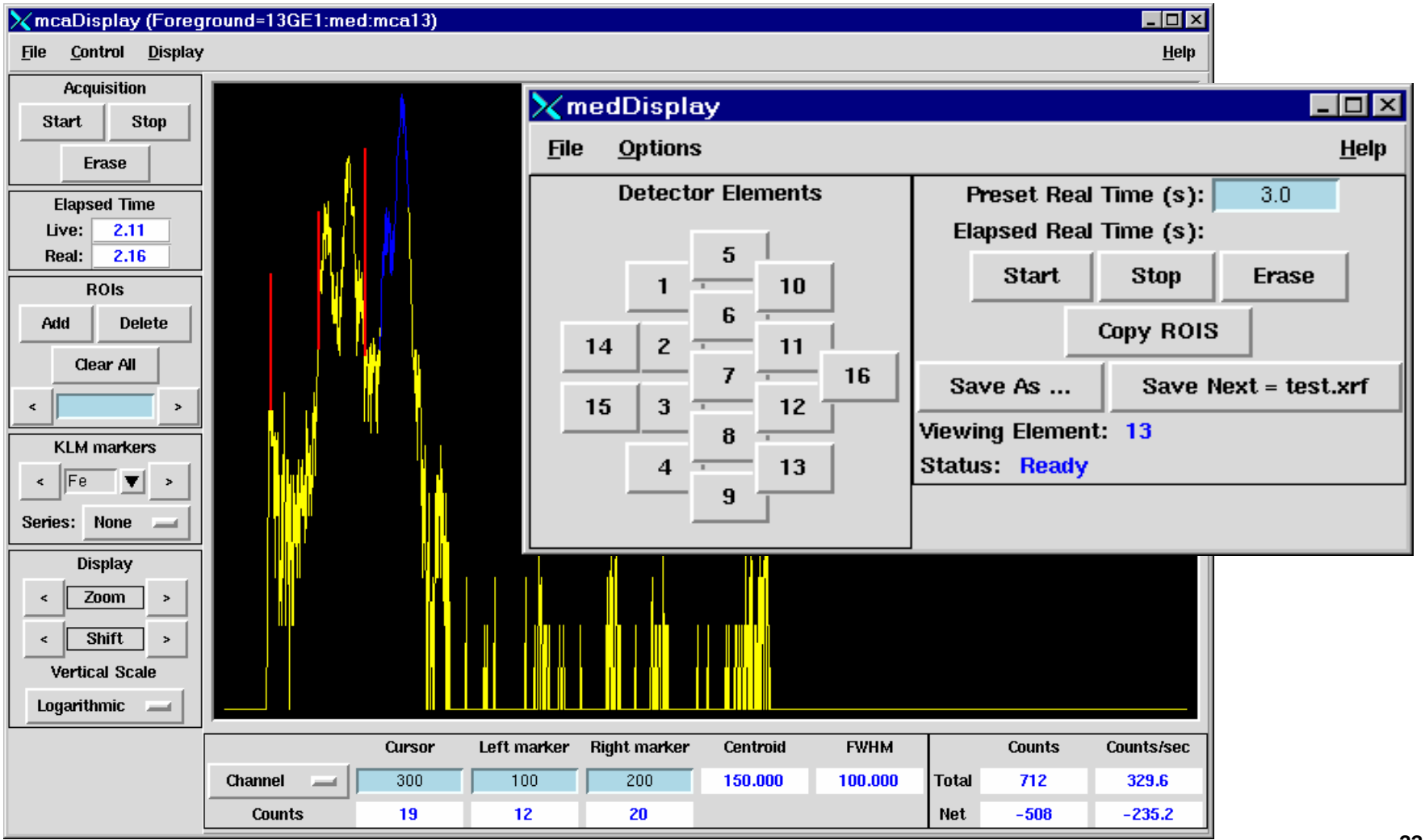
- **Collection of Mca objects. Methods operate on all contained Mca objects. Example:**
  - add_roi(), set_presets(), get_calibration(), etc.
- **Used as base class of epicsMed.**

epicsMed

- **Subclass of Mca and Med**
- **All methods of Mca and Med, plus start(), stop(), erase(), wait(), etc.**
- **Re-implements base class routines for set_calibration(), set_rois(), etc. to communicate with fields in the EPICS MCA record**
- **Example use:**

```
from epicsMed import *
med = epicsMed('13GE1:med:', 16)
med.erase()
med.start()
med.wait()
med.write_file('test.001')
```

medDisplay


mcaDisplay (Foreground=13GE1:med:mca13)
File
Control
Display
Help
Acquisition
Start
Stop
Erase
Elapsed Time
Live: 2.11
Real: 2.16
ROIs
Add
Delete
Clear All
KLM markers
Fe
Series: None
Display
Zoom
Shift
Vertical Scale
Logarithmic
medDisplay
File
Options
Help
Detector Elements
5
1
10
6
14
2
11
7
16
15
3
12
8
4
13
9
Preset Real Time (s): 3.0
Elapsed Real Time (s):
Start
Stop
Erase
Copy ROIS
Save As ...
Save Next = test.xrf
Viewing Element: 13
Status: Ready
Cursor
Left marker
Right marker
Centroid
FWHM
Channel
300
100
200
150.000
100.000
Counts
19
12
20
Counts
Counts/sec
Total
712
329.6
Net
-508
-235.2