Introduction to πScope

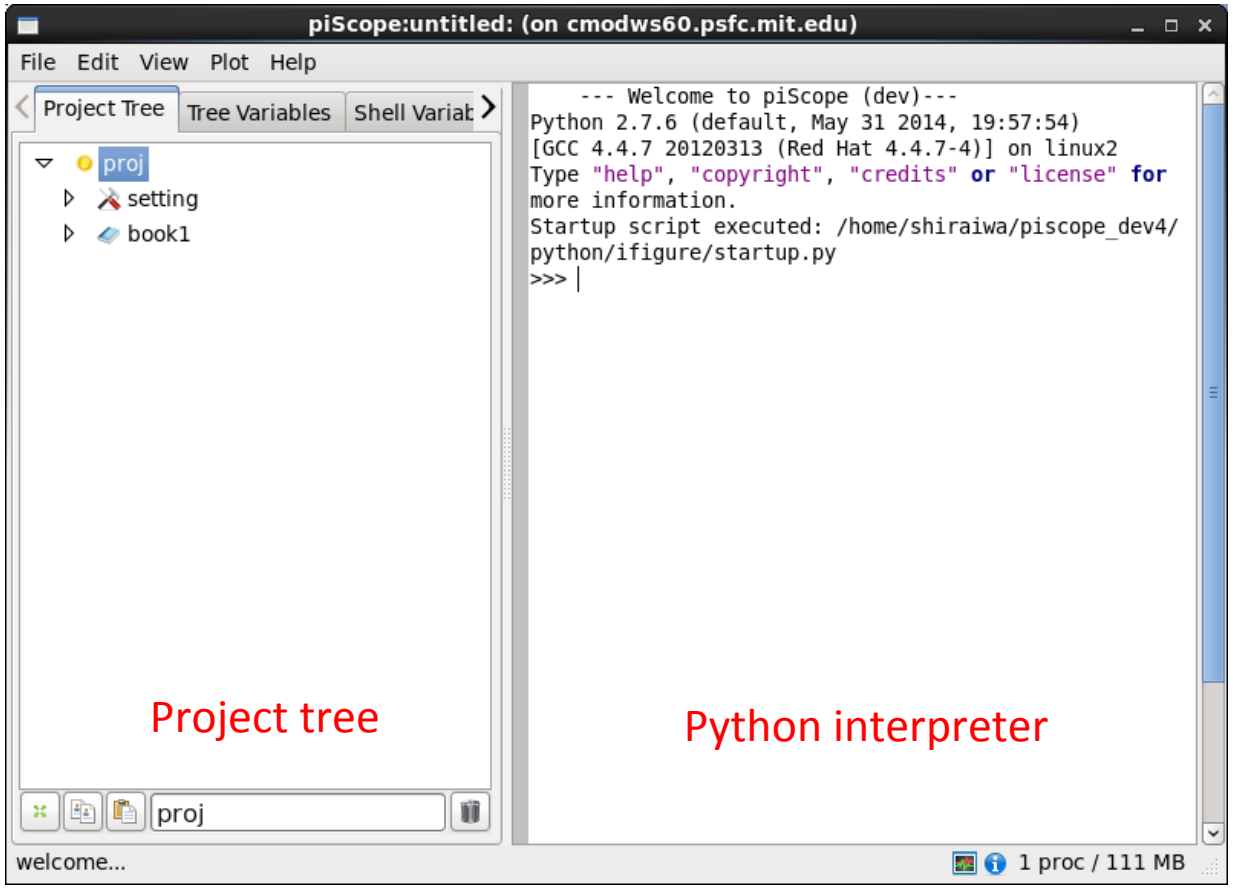
S. Shiraiwa

Introduction : What is πScope?

- A major upgrade of iScope using python (from i to 3.14)
- Why Python?
  - Interpreter
  - Dynamic execution (like execte() in IDL)
  - Object-oriented
  - Large user population, and many modules for non-scientific tasks (socket, SSH, threading, MP etc)
  - Better graphics based on matplotlib
  - Modern graphical interface using wxPython
  - No need to write ";" in every line (:D)
  - No license fee
- Goal
  - Develop versatile front-end for…
    - Brows experimental data (part 1)
    - Run simulation codes (part 2)
    - Prepare publication quality graphics
  - Implement missing-features in iScope
    - Undo/redo
    - Object-oriented design

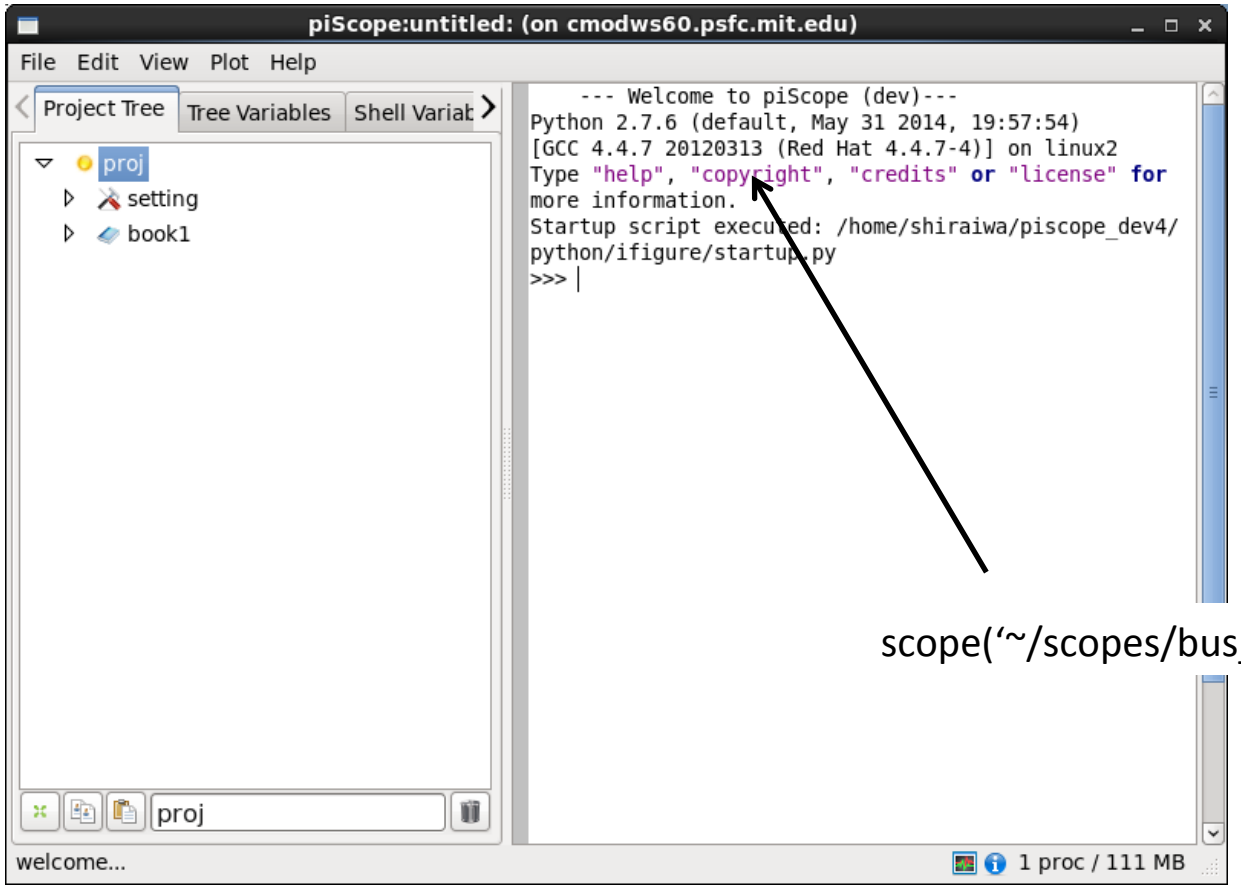
Part 1: πScope as a MDS+ data browser (What is new from dwscope and iScope)

Main screen of πScope



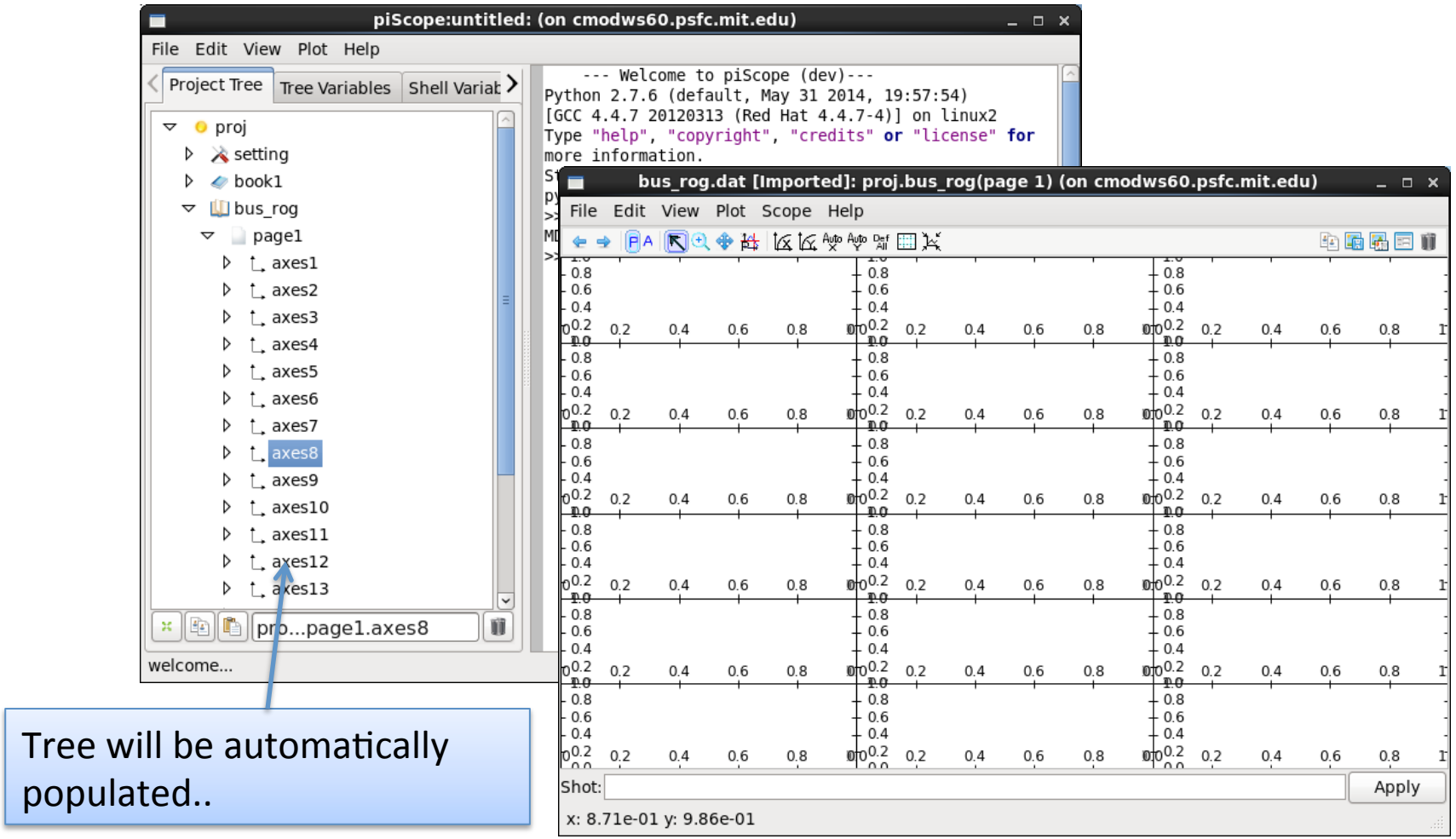
- πScope is designed as a work environment, rather than single window application.
  - Multiple scopes can be opened and saved as a "project"
  - "project" can hold various data, python scripts, plots, and so on, allowing for recording process of analysis

Use scope() to start MDS+ data browser



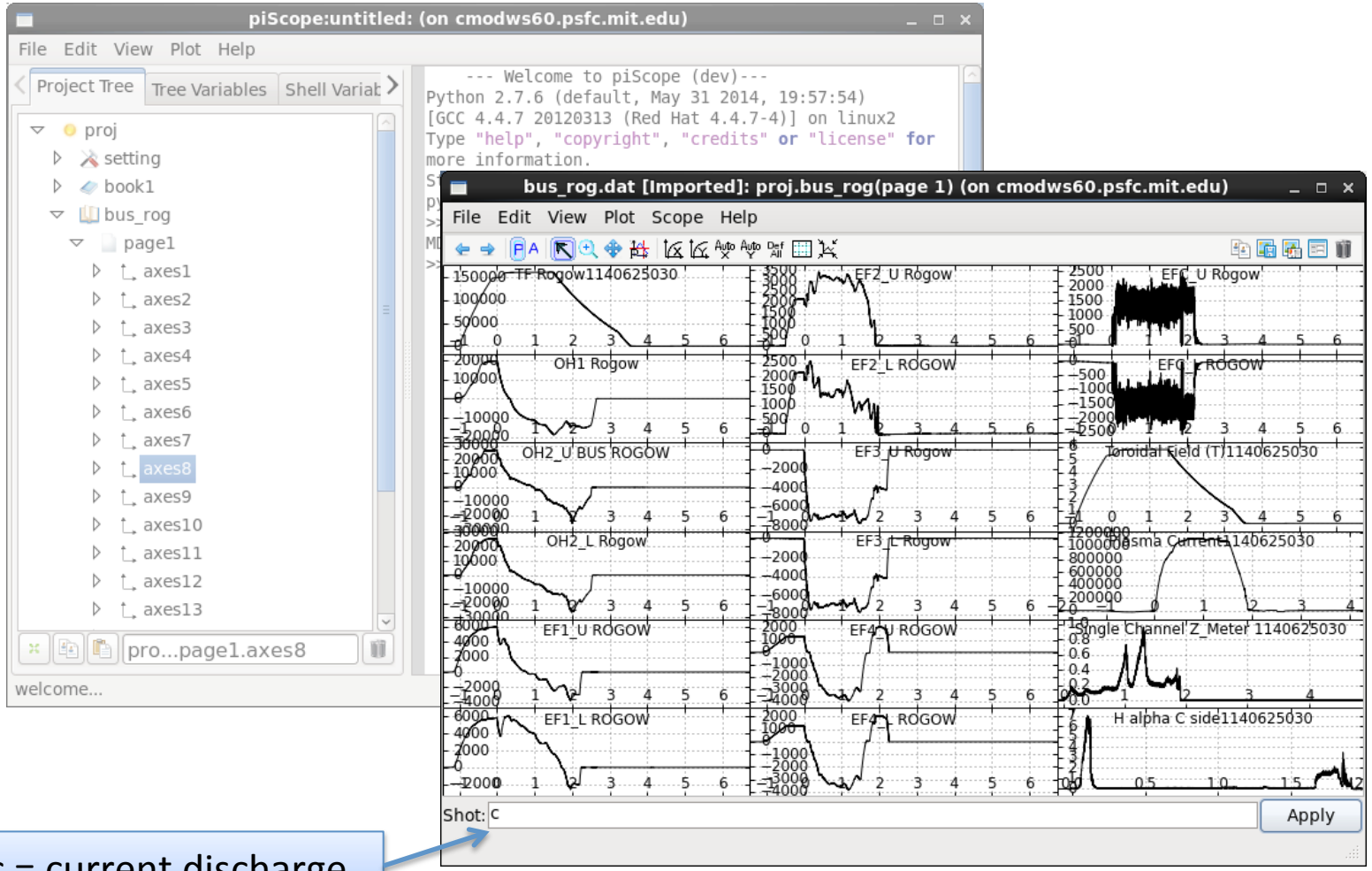
- scope() will open an empty scope window
- scope('~/scopes/bus_rog.dat') can be used to import a dwscope file

Use scope() to start MDS+ data browser



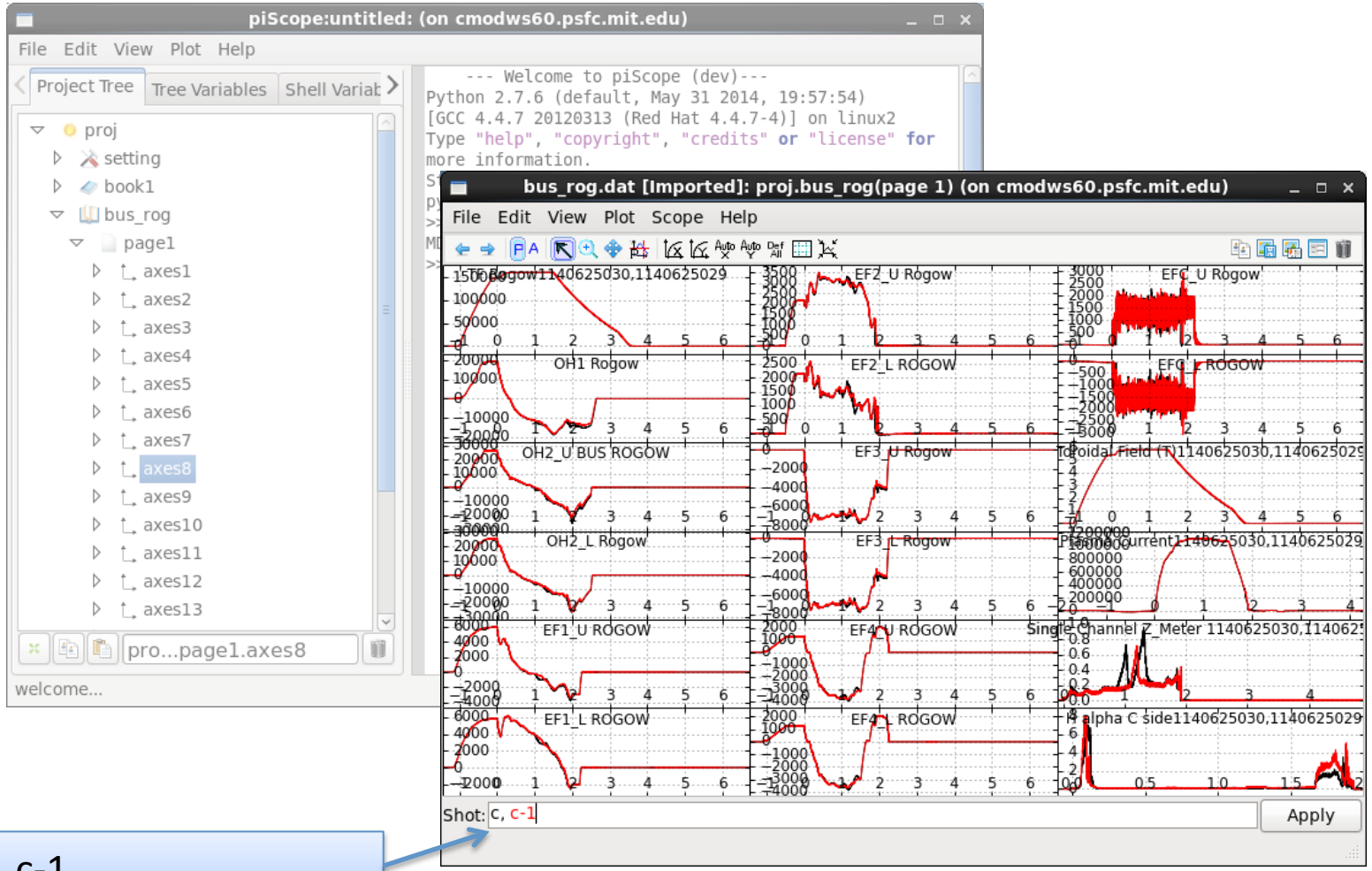
Tree will be automatically populated..

Enter a shot number…



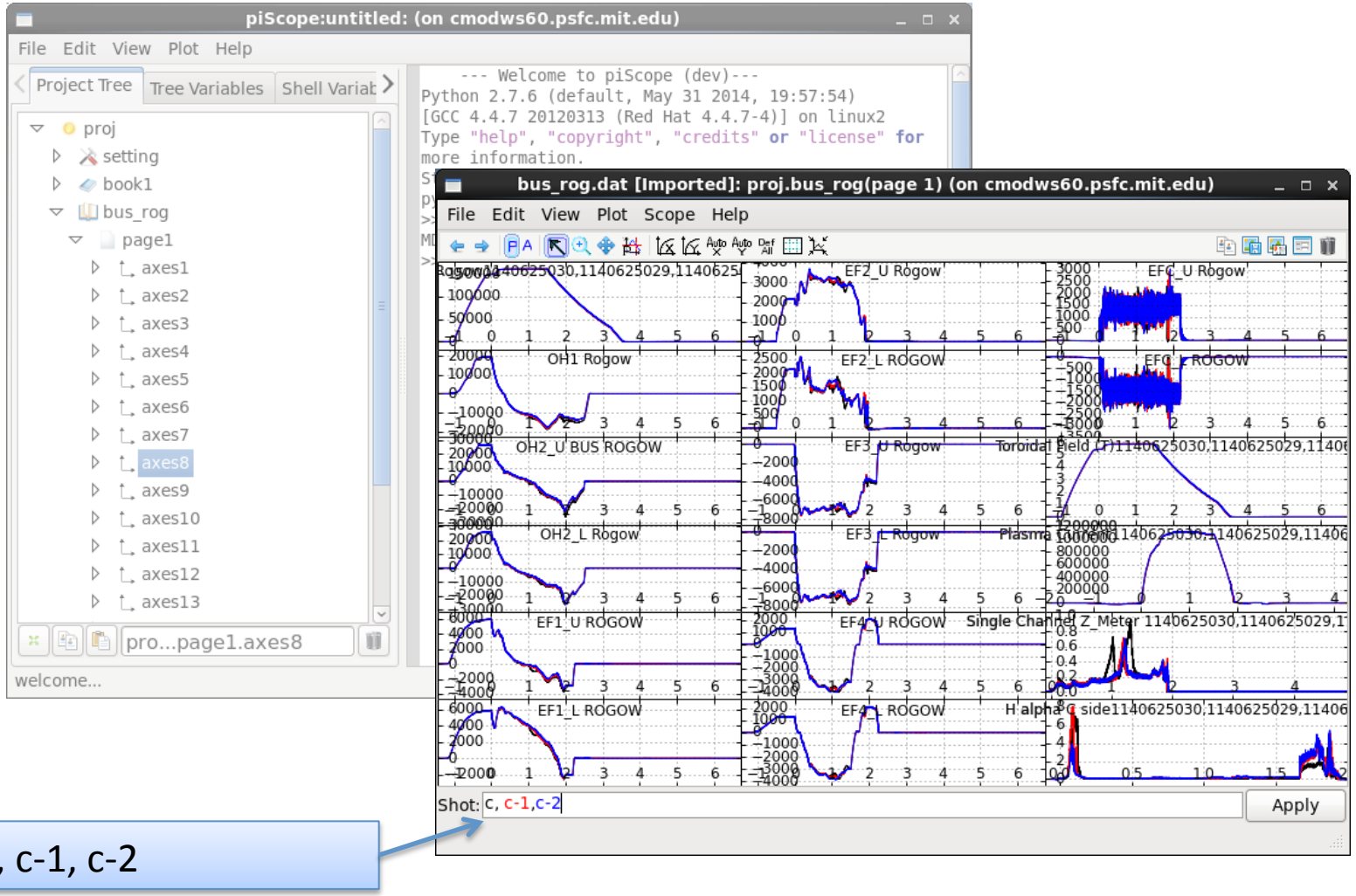
c = current discharge

… overlay 2nd discharge



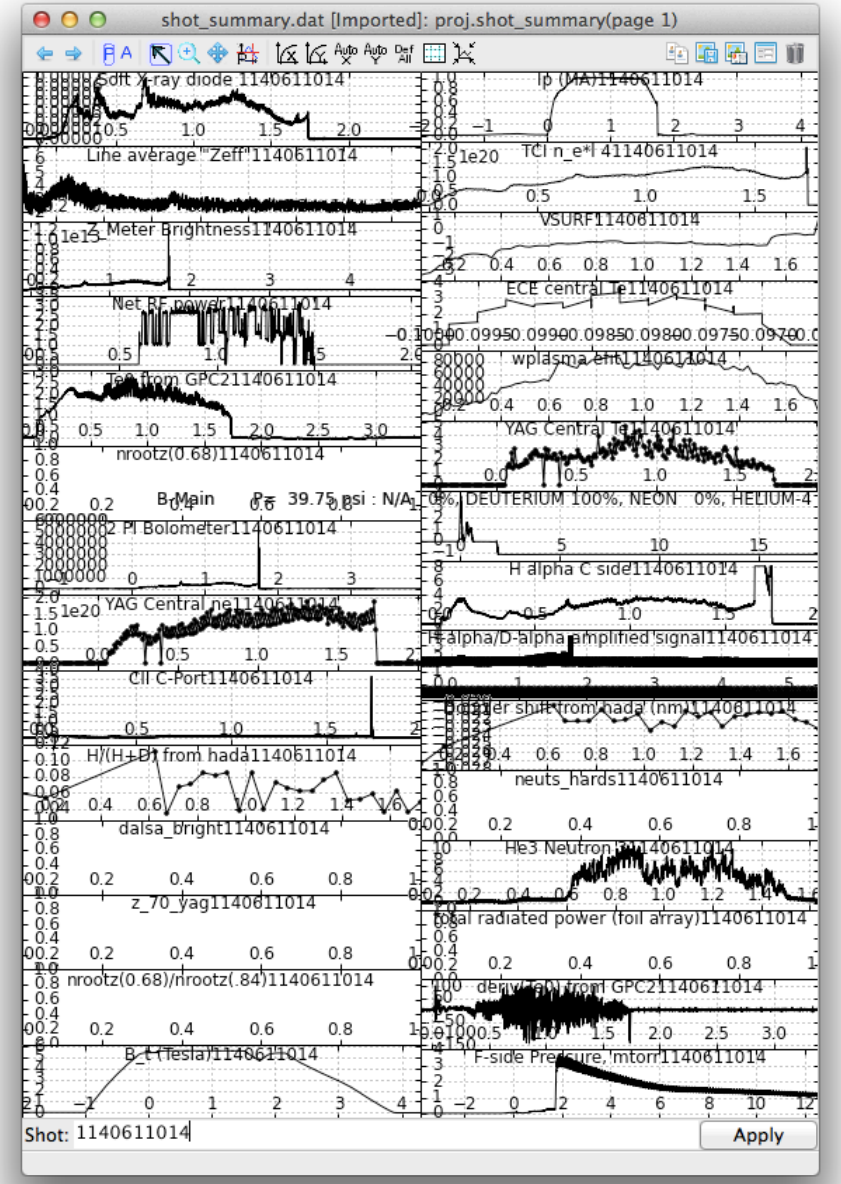
c, c-1

… and 3rd one



c, c-1, c-2

πScope imports dwscope files…

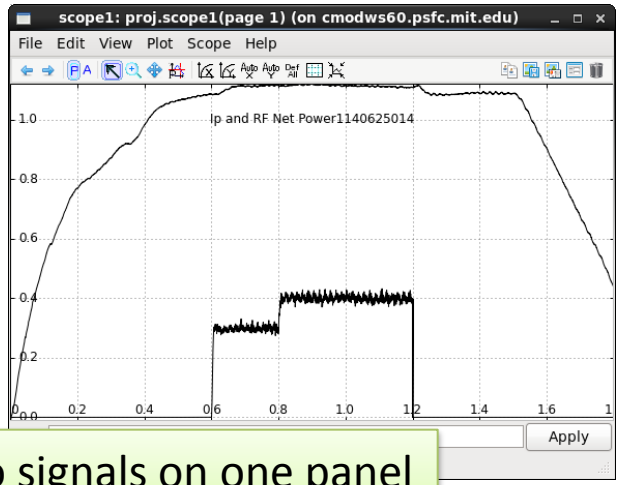


- Just to choose “import dwscope” from menu
- Imports
  - Panel layout
  - Signal TDI expressions
  - Symbol
  - Global default settings
- Mimics the algorithm used in dwscope
  - Sharing variables among panels
  - Position variables (_index, _col, _row)

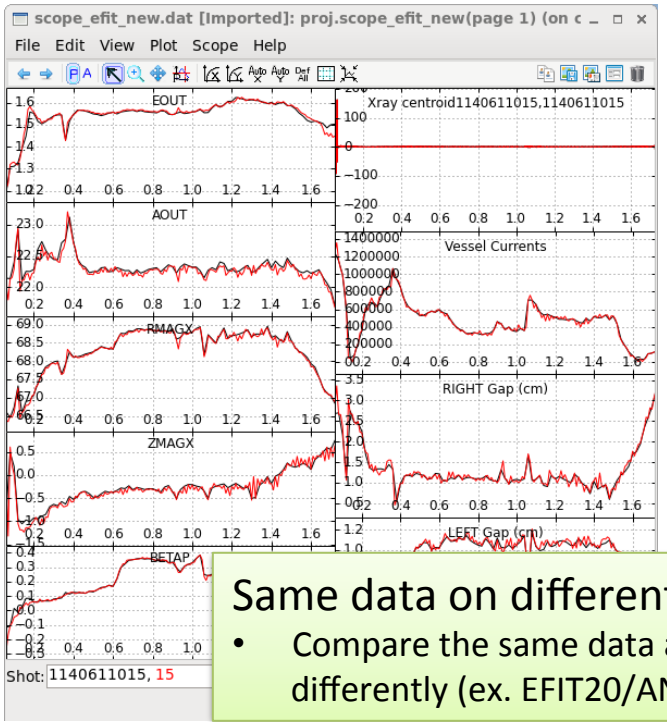
Shot number field

- Comma-separated numbers
- Remember history

Multiple ways of overploting

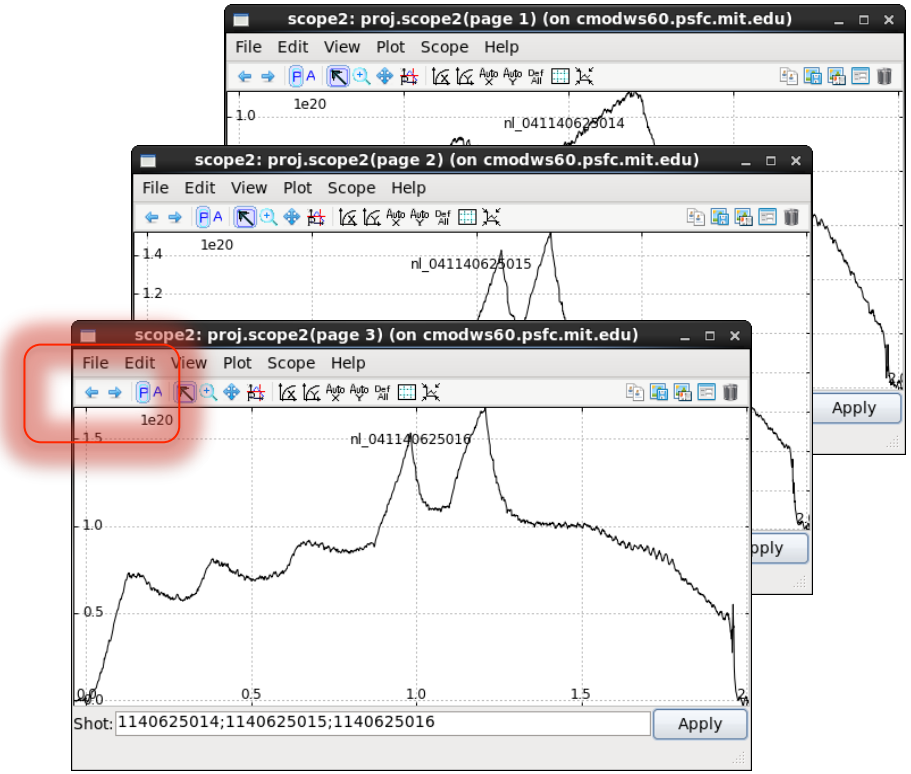


Two signals on one panel



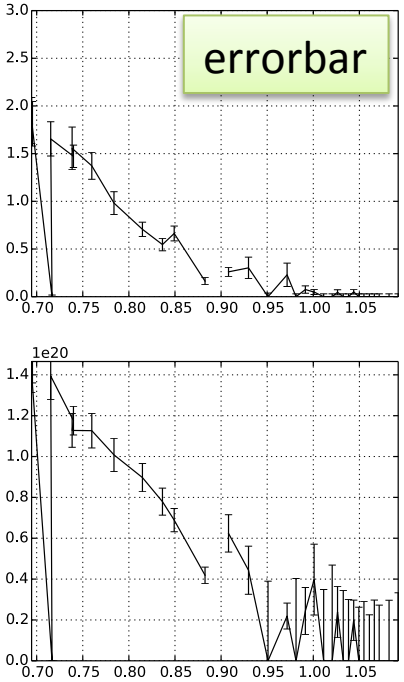
Same data on different trees

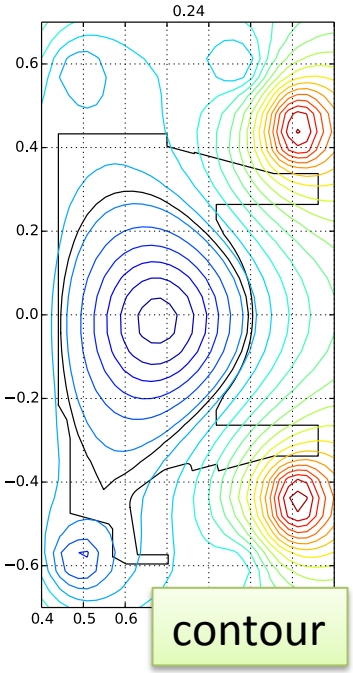
- Compare the same data analyzed differently (ex. EFIT20/ANALYSIS)

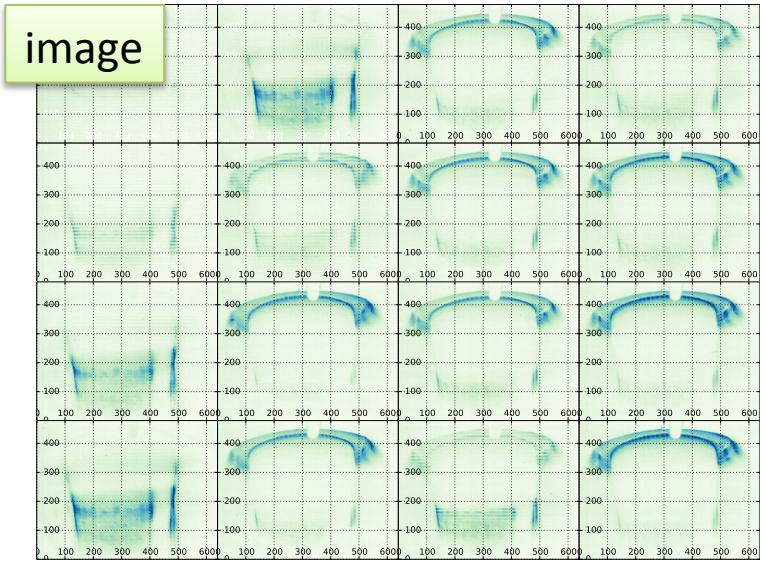


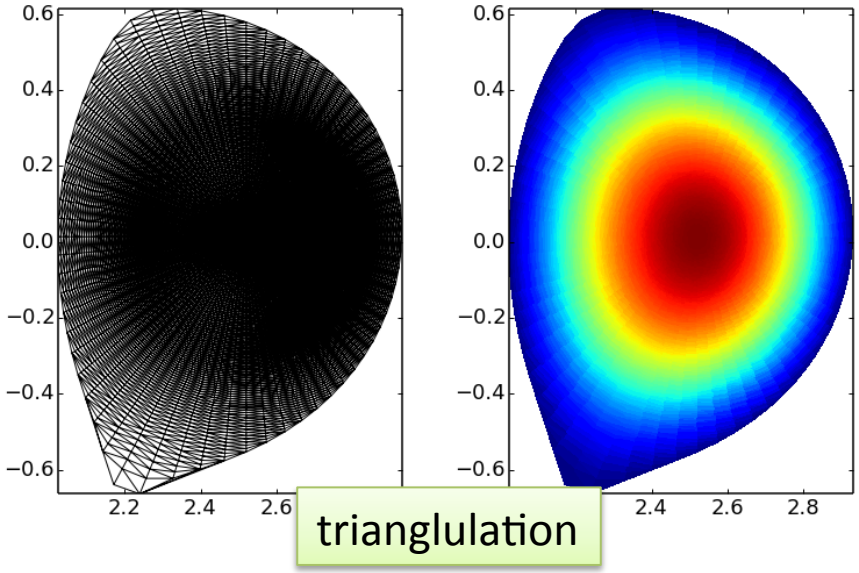
Multipage plots

- ";" separated shot numbers automatically generate multiple pages
- Can move between pages either by Prev/Next buttons or mousewheel button







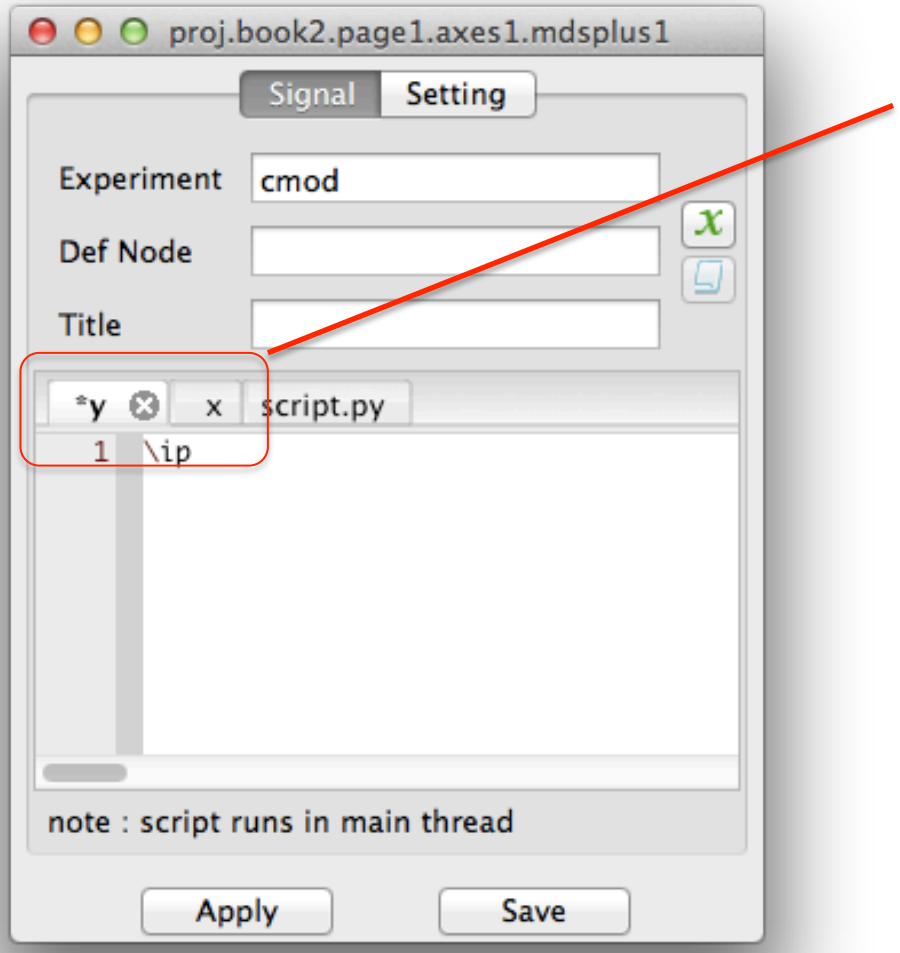


Uses matplotlib, a popular plotting library for python

- various plotting commands
- anti-aliased graphics

In-panel data analysis using python

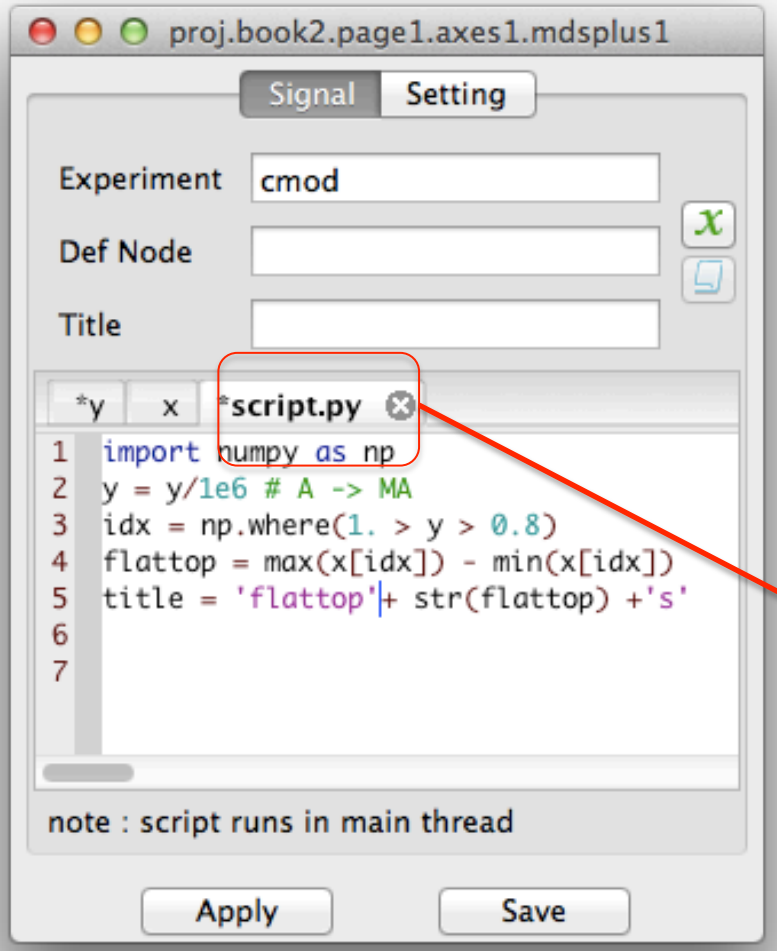
Data panel in πScope



- Variables to be read from MDS+ server
  - Read as many variables as you want
  - This is the same as dwscope. User can process data with TDI

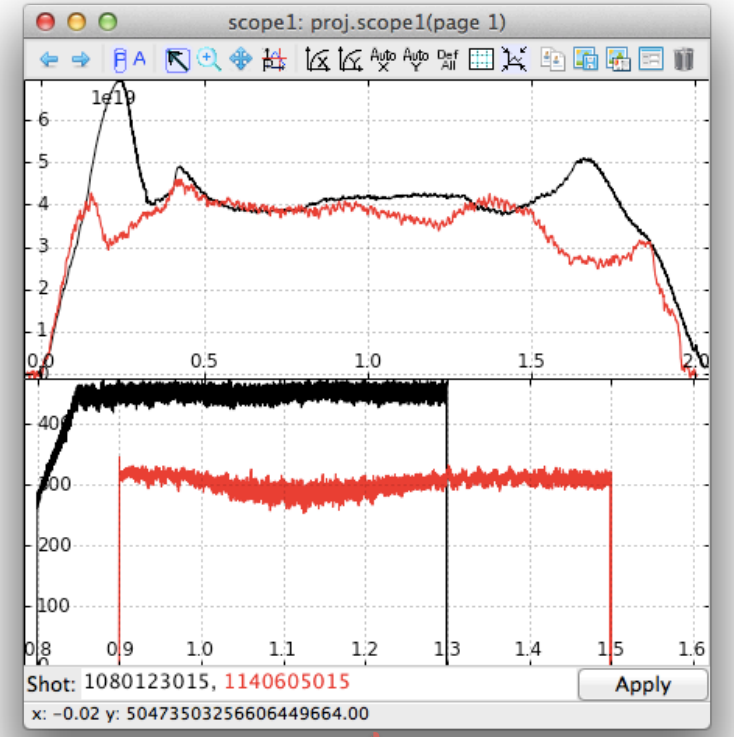
In-panel data analysis using python

Data panel in πScope

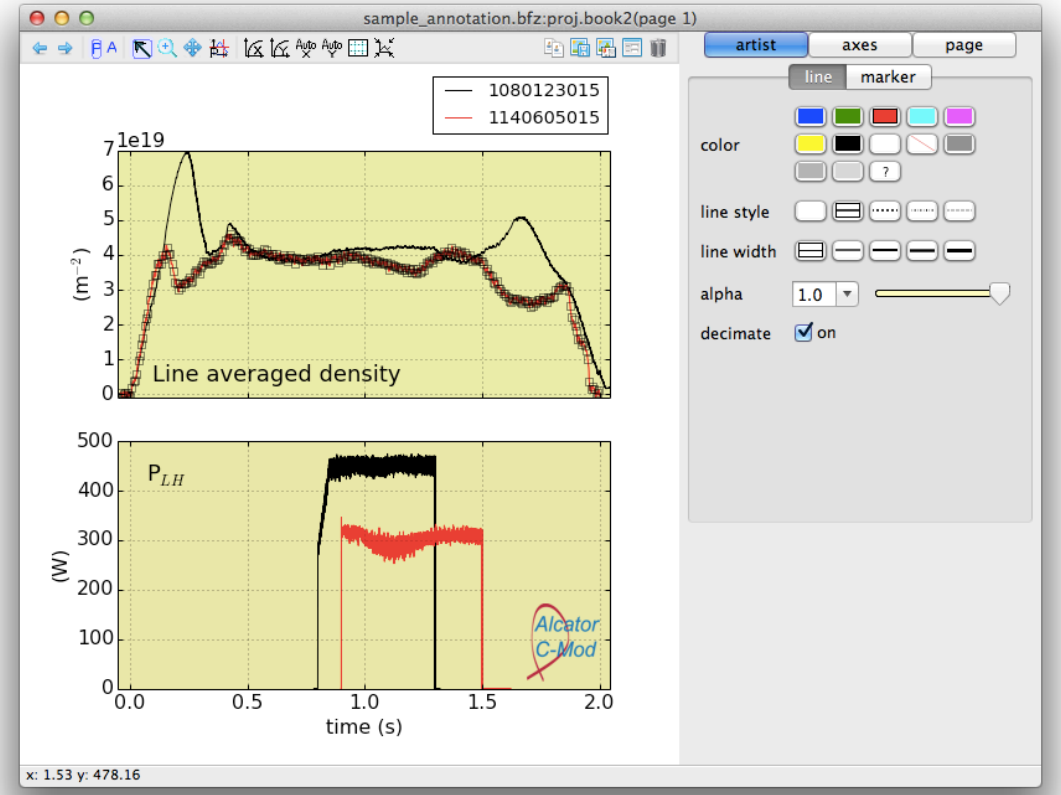


- Variables to be read from MDS+ server
  - Read as many variables as you want
  - This is the same as dwscope. User can process data with TDI
- Data can be processed by python script also
  - This example will write flattop length to title

Publication quality graphics directly from scope screen



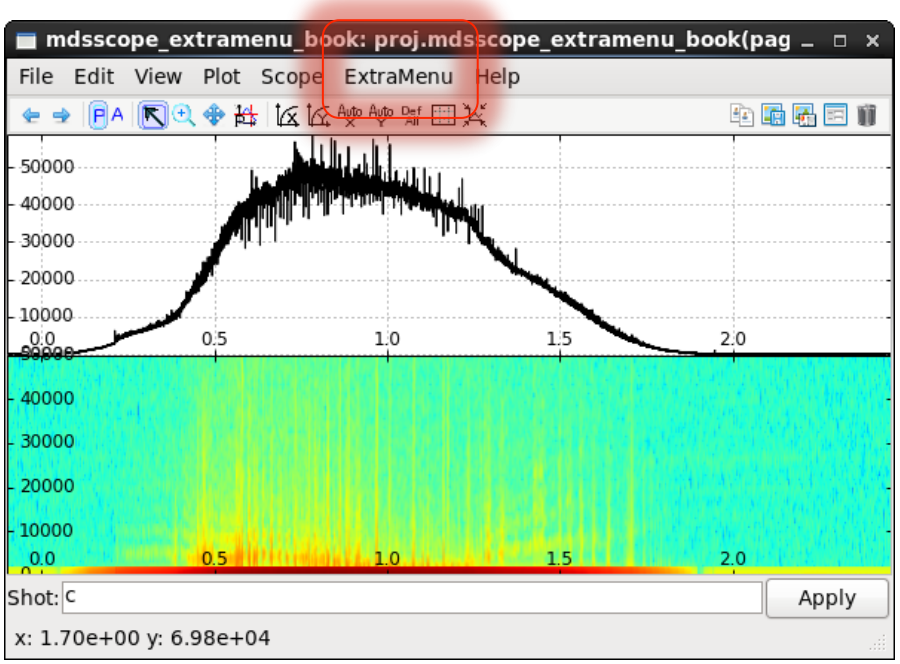
- Edit graphics properties
  - Line colors, Line style
  - Label font size
- Add annotations
  - Legends
  - embed EPS picture



Editing feature supports…

- copy/pastes between panels
- undo/redo (first ever in all *scope series for MDS+ ???)

Object-oriented design : easy to build a custom app.

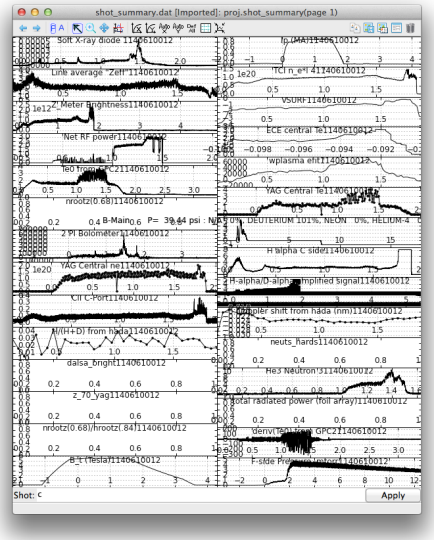


- Scope is a class object
  - User can expand it by inheriting it and only need to write what he wants to add
- Example (only 40 lines)
  - Add an extra menu in the menu bar
  - Show a dynamic spectra using the signal in #1 panel data

gui.py | mdsscope_extramenu.py

```
1  '''
2      this example shows how to add extra menu to mdsscope
3      and add call-back routines when the menu item is
4      selected
5  '''
6
7  from ifigure.interactive import figure
8  from ifigure.mdsplus.mdsscope import MDSScope
9  import wx
10
11 exp = 'xtomo'
12 node = '.BRIGHTNESSES.ARRAY_3:CHORD_23'
13
14 class MDSScopeExtraMenuGUI(MDSScope):
15     def __init__(self, *args, **kargs):
16         MDSScope.__init__(self, *args, **kargs)
17         extra_menu = wx.Menu()
18         self.menuBar.Insert(self.menuBar.GetMenuCount()-1,
19                             extra_menu,"ExtraMenu")
20         self.add_menu(extra_menu, wx.ID_ANY,
21                       "Plot Spectra...",
22                       "plot spectra of panel #1",
23                       self.onPlotSpectra)
24         self.nsec(2)
25         self.cls()
26
27         self.data = self.AddSignalScript(experiment = exp,
28                                          signals = {'y': node},
29                                          kind = 'plot')
30
31     def onPlotSpectra(self, evt):
32         try:
33             p = self.data.get_child(0)
34             x = p.getvar('x')
35             y = p.getvar('y')
36             self.isec(1)
37             self.spec(x, y)
38         except:
39             import traceback
40             traceback.print_exc()
41         evt.Skip()
42
43 if not obj.get_parent().has_child('mdsscope_extramenu_book'):
44     v = figure()
```

Multi-thread/multi-processing



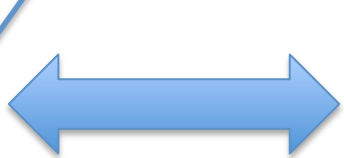
Scope in πScope is parallelized

- Normally 5-10 threads are running in parallel CPUs.
- Better overall performance when loading a large data.
- Hide data transfer latency.

MainThread
- Request MDS+job
- Run postprocessing script
- Draw Screen

Thread-1 (MDS+ Session runner)
- Send MDS+ job to subprocesses
- Monitor PP done
- Send screen update events

Thread-2 (Post-processsing)
- Receive data from workers
- Request python scripts



MDS+

Worker threads or processes

πScope is ready for "control room use" on C-Mod

- Tested few times as PhysOp (by GW and SS)
- SB and SB are using it in daily basis as scope and also for running GENRAY/ CQL3D

- Command to execute beta release version is in public cmod bin directory
  - Most people already setup a path
  - Type "piscope" to launch it.
  - The executable is actually shell script to take care of adding piscope script directory to PYTHONPATH and use python2.7 to start piscope. (No need to change your user environment, such as .bashrc)

- Resources
  - Wiki page http://piscope.psfc.mit.edu (under construction)
  - Mail list piscope-users@mit.edu

- Development
  - Official HG repository is on alc-software, and will be released under GPL, if test on C-Mod goes well.
  - Latest development version is shiraiwa's own area
  - Support/suggestiosn/comments are welcome