Offline Analysis Tools for Detector Characterization

Peter Shawhan

LIGO/Caltech

LSC Meeting

August 16, 2000

LIGO-G000192-00-E

Planning For Offline Detector Characterization

Tools for online detector characterization are in good shape

- DataViewer
- Diagnostic Test Tools
- DMT

Also need to be able to analyze offline, *e.g.*:

- Study transients in detail
- Analyze a significant amount of data
- Analyze the same data in various ways

This talk tries to identify our needs and propose a plan

Not meant to limit options, but to focus efforts and encourage sharing of tools and expertise

To be addressed:

- Data access and viewing
- An environment for automated analysis
- A standard for “summary data objects”
- Viewing & manipulation of summary data objects

Data Access and Viewing

What we have:

- Guild
  - Retrieves & displays information from database
  - Retrieves raw data to disk using LDAS frameAPI
- Xlook
  - Displays contents of LIGO_LW files

What we need:

- Raw data display
- Trend display
- Access by programs (C, Matlab, etc.)

Proposal:

- Make DataViewer and Diagnostic Test Tools available for offline use
- Set up a trend server for off-site clients
- Create a “data flow manager”
  - Intermediary between programs and LDAS
  - “Smart NDS server” for DV and DTT
- Set up an index of information services
- Write LAL package to parse LIGO_LW (tables, etc.)

Summary Data Objects

Analysis can produce results of various types, e.g.:

- Low-rate time series
- Power spectrum
- Table (“ntuple”)
- Histogram
- Sequences of any of these

Analysis job needs to be able to store these in a standard format for later retrieval, viewing and/or further manipulation

Proposal:

- Store these in LIGO_LW format (extend if necessary)
- Store each object in a separate file in a given directory, with an index, for random access
- Use Matlab to display and manipulate, with a set of LIGO-specific tools for convenience

An Environment for Automated Analysis

Interactive tools are nice, but limited

Need to be able to automate repetitive tasks and run in "batch" mode

Essential features:

- Automated, pipelined access to raw data
- Construct sequences of component operations
- Can extend capabilities with custom code
- Store results for later inspection or post-analysis

Example 1:

- Get list of events from database
- Get raw data for a time interval around each event
- Calculate power spectra, channel correlations, ...

Example 2:

- Get a month of raw data for a few channels, in appropriate chunks
- Apply a transient-detection algorithm
- Write out a table of events found

Candidate analysis environments include plain C or C++ (with LAL and i/o libraries), Matlab, Triana, PSE / SCIRun, ...

C and C++ Programs

Advantages:

- Complete flexibility
- C, at least, is familiar to many people
- Natural to use code in LAL or DMT library (though may have to deal with different C vs. C++ paradigms)

Disadvantages:

- Want signal processing functions, etc.

Potential adaptations for LIGO:

- For C, establish standard i/o paradigm and standard main program(s)
- Link to Matlab math library?

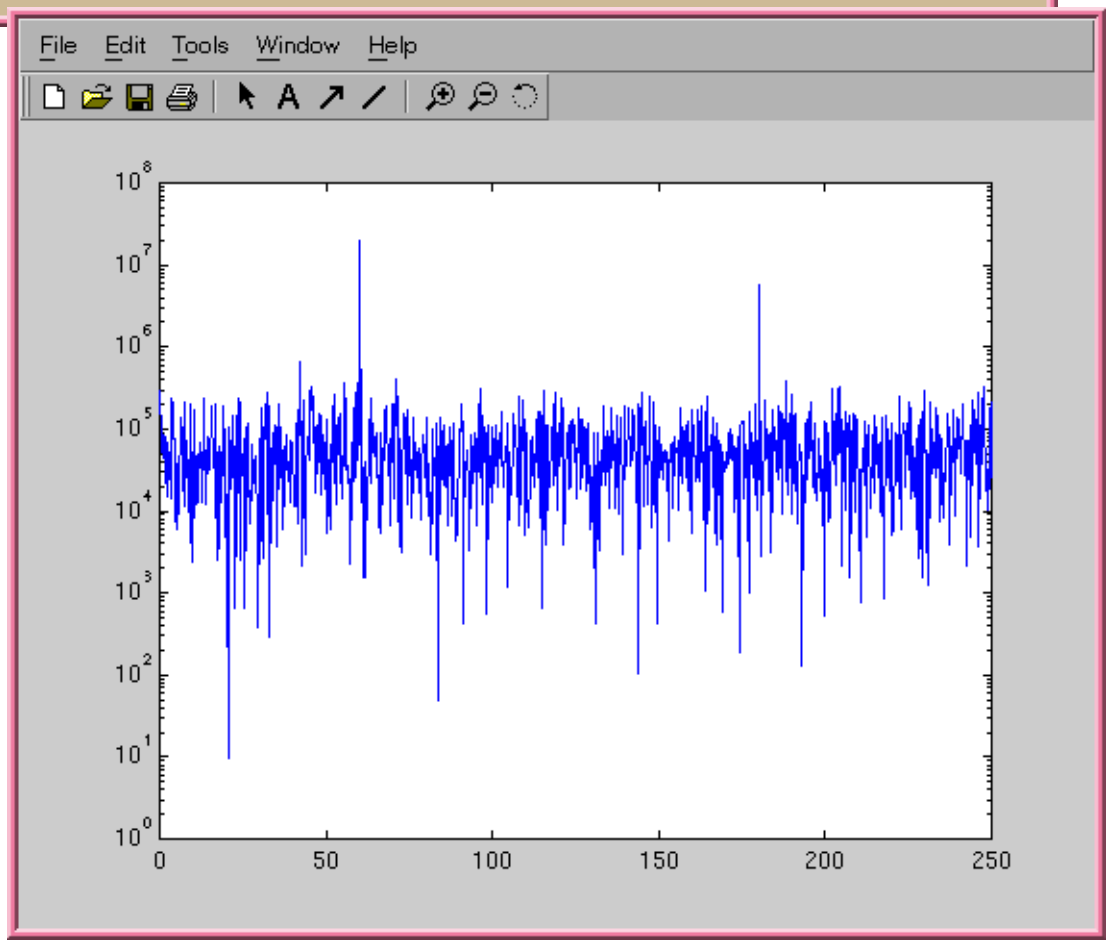
Sample Matlab Windows

```
pshawhan@capella> matlab

                                < M A T L A B >
                    Copyright 1984-1999 The MathWorks, Inc.
                          Version 5.3.0.10183 (R11)
                                  Jan 21 1999


  To get started, type one of these: helpwin, helpdesk, or demo.
  For product information, type tour or visit www.mathworks.com.

>> samprate = 16384;
>> samples = 2^16;
>> signal=0.2*sin((60.0*2*pi/samprate)*(1:1:samples));
>> nonlin = -10.0*sign(signal).*(signal.*signal);
>> noise = randn(size(signal));
>> fft1 = fft(signal+nonlin+noise);
>> psd = fft1.*conj(fft1);
>> semilogy((1:1:1000)*(samprate/samples),psd(1:1000));
>>
```



Matlab

www.mathworks.com

Advantages:

- Very mature commercial product, widely used
- Wide range of built-in functions; easy scripting
- Nice graphics
- Good documentation (on-line and books)
- Can be extended with compiled C or C++ code
- Access to primitive datatypes and operations

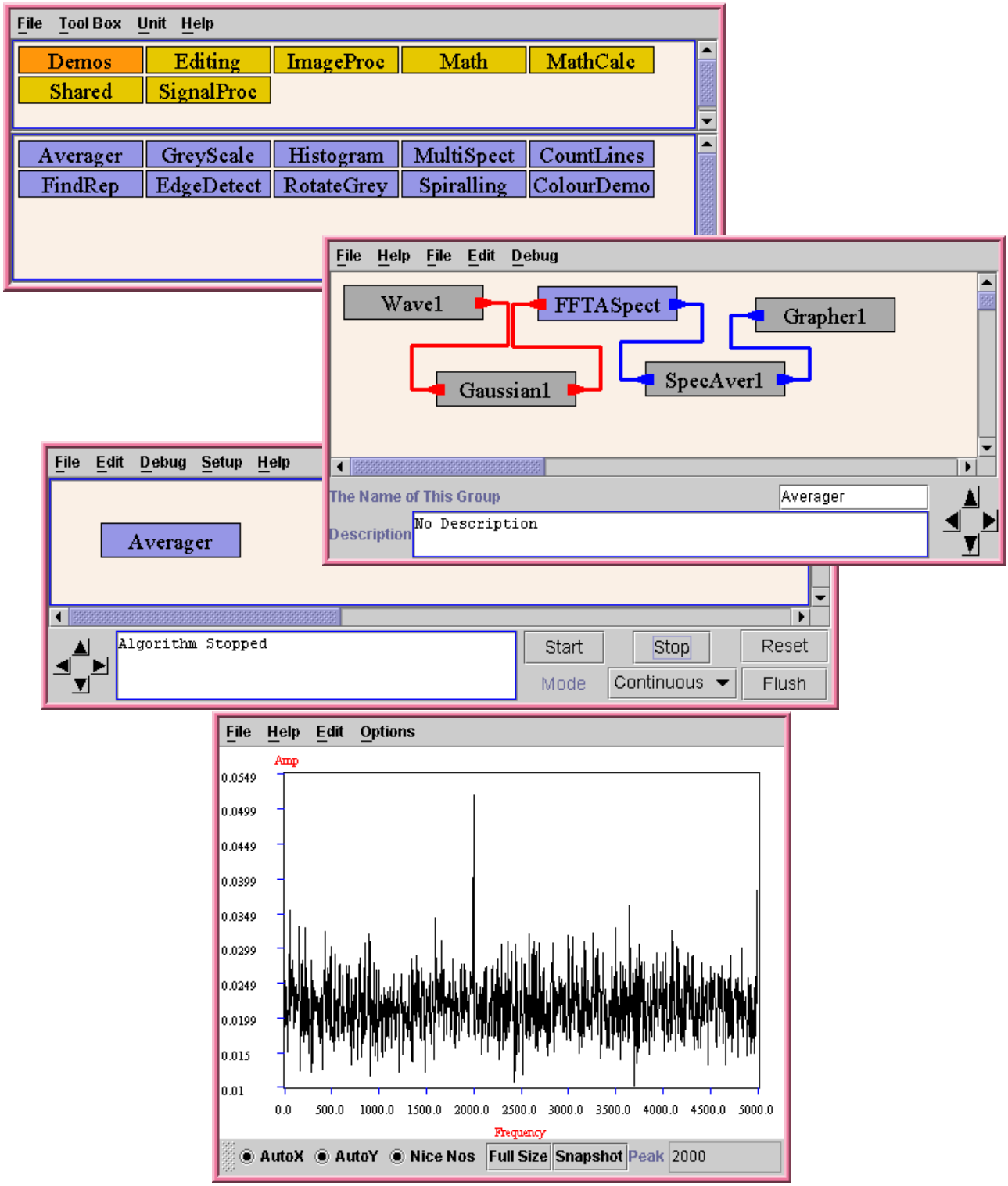
Disadvantages:

- Costs $1200+ (academic license)
- Thought to be slow (at least for interpreted scripts)
- Not parallelizable (but prototype extensions exist to use MPI, etc.)

Potential adaptations for LIGO:

- Create tools and/or example scripts for automated analysis (or adapt Simulink?)
- Use object-oriented capabilities for LIGO data objects (time series, power spectrum, event table, ...)
- Link in LAL library? DMT library?

Sample Triana Windows



Triana

www.triana.co.uk

Developed primarily for GEO600 data analysis

Written in Java; first production release will be soon

Being commercialized; cost=?

Advantages:

- Nice graphical interface
- Built-in signal processing tools are tailored to gravitational-wave analysis
- Designed for pipelined data analysis
- Parallelizable on distributed computers

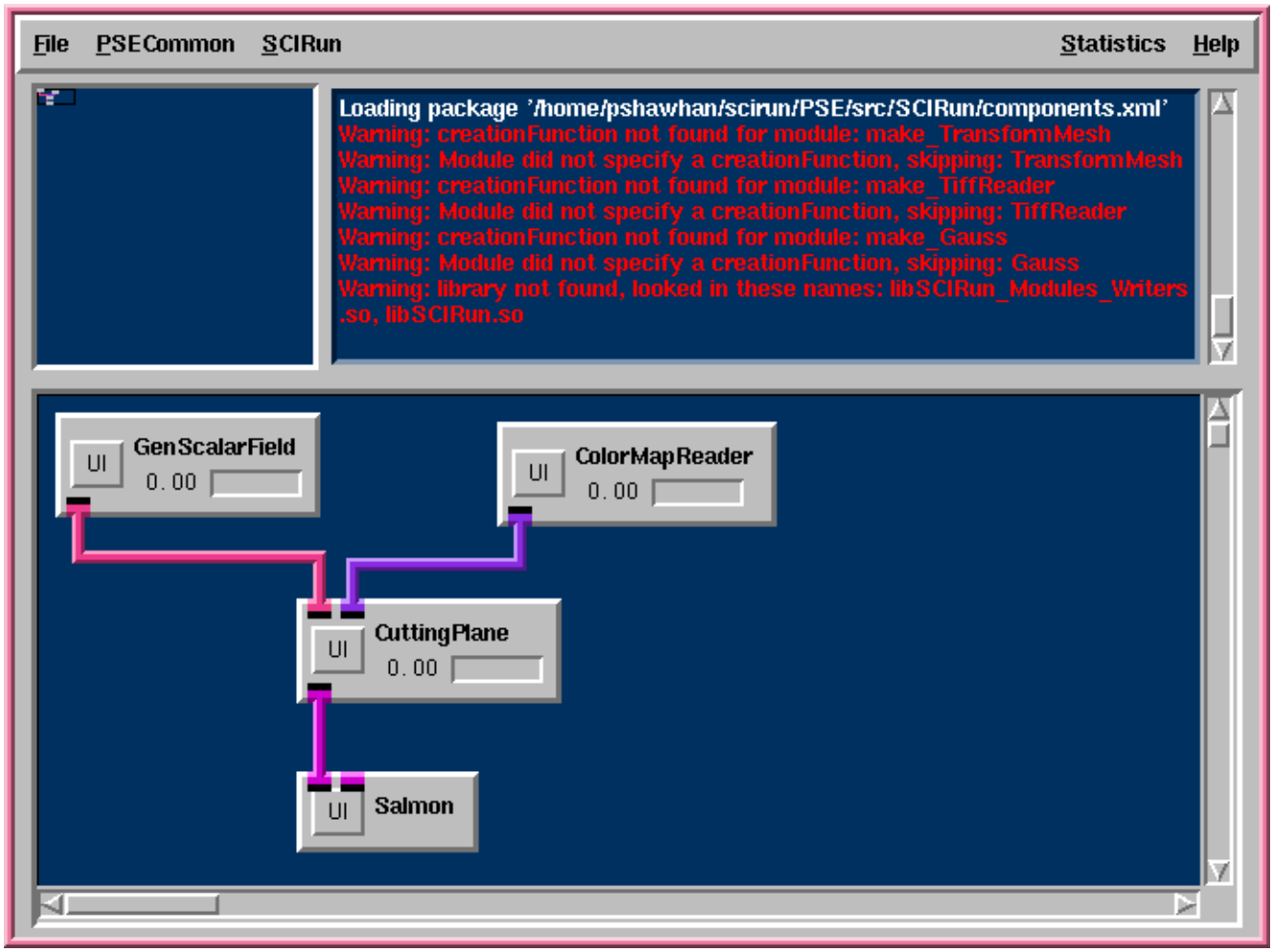
Disadvantages:

- Primitive operations may be unwieldy
- Graphing module has limited flexibility
- Java is unfamiliar to most LIGO collaborators
- Java is slow

Potential adaptations for LIGO:

- New datatype and processing modules for tables of events
- Extend with C / C++ code ??? (*e.g.* LAL)

Sample PSE / SCIRun Window



PSE / SCIRun

www.cs.utah.edu/sci/scirun

Developed at University of Utah

Currently used primarily for biophysics, for 3-D problem solving and visualization

Written in C++, with Tcl/Tk graphical interface

Advantages:

- Free!
- Multi-threaded, so can take advantage of multiprocessor machines
- Compiled C++, smart memory management $\Longrightarrow$ fast
- Extendable in languages familiar to LIGO

Disadvantages:

- No built-in signal processing modules
- No conventional 2-D visualization
- Little formal documentation

$\Longrightarrow$ would take a lot of work to make it usable

Possible adaptations for LIGO:

- Incorporate DMT datatypes and processing code

Analysis Environments: My Recommendation

Major issues:

- Learning curve
- Access to primitive operations
- Availability of signal processing functions, etc.
- Ease of scripting
- Ease of sharing enhancements among groups

$\Longrightarrow$ I recommend focussing on C++ and Matlab

Use C++ for:

- Batch processing (generate summary data objects)

Use DMT library and/or LAL (and/or Matlab math??)

Note: can basically write code in plain C, if you want

Use Matlab for:

- Scripted interactive analysis
- Viewing and manipulation of summary data objects

Final Remarks

Key point: start building up tools and expertise for offline analysis, and sharing it!

Proposals made in this talk are open for discussion

Plan will probably evolve as we gain experience

This effort will require contributions from many people

Some components may already exist, and just need to be shared

I am willing to set up a “clearinghouse” web site for offline analysis tools, documentation and usage notes

I am also happy to discuss specific tools & tasks