An Environment for Matlab Analysis of DAQ Data from the 9ma Experiments

Ned Arnold

October 2009

Interactive analysis of DAQ Data using Matlab

- Learning Curve Questions …
  - How do you know which DAQ data directory to use?
  - How do you know what valid timeframes of data are in a particular directory?
  - How do you know the channel names archived in the DAQ?
  - What do you do if you get an error from daq_fetch?
  - Understanding the structure of the data returned for a channel from daq_fetch
    - DAQ_data{n}(1:numPulses, firstSample:lastSample, data_index*) (*1-sample #s, 2-subchannel 1, 3-subchannel 2, etc)
  - How do you know the sub-channels of a channel …
    - How many and what do they represent (I, Q, Amplitude, Phase, X, Y, Klystron parameters, etc)
  - Since the number of samples returned varies from channel to channel, how to you align them?
    - Answer: correlate the sample #s … (easily said!)
  - How do you know which sample #'s are supposed to have beam (i.e. valid diagnostic data)
  - How do you decipher the TIME array returned by daq_fetch
  - "I'm sure someone has had to implement this algorithm, where can I find an example?"

Interactive analysis of DAQ Data using Matlab

- Existing Tools…
  - daq_fetch.m (Michael Davidsaver)
    - Input arguments: list of channel_names, Tstart, Tend
      - DAQ directory hard-coded in daq_fetch.m
    - Output arguments: list of channel names (+ time), daq data [dlbl, daq]
  - daq_fetch.m (enhanced by Ned Arnold)
    - Input arguments: list of channel_names, Tstart, Tend, DAQ directory, retry flag
    - Output arguments: list of channel names (+ time), daq data [dlbl, daq], missing channels
  - DESY’s DAQ GUI
    - Helps find channel names which exist in DAQ data
  - DOI Version 1
    - Matlab GUI to retrieve DAQ data and …
      - Store in a local file (for interactive analysis)
      - Pass on to an analysis program
    - On daq_fetch error, searches for channel that is missing data
    - Allows channel names to be stored in “groups” for easier entry
    - Provides introspection of analysis scripts to designate channels to be fetched (i.e. selecting an analysis script puts those channels on the list to be fetched)

Ideas to simplify the learning curve …

(or "Wouldn't it be nice if …")

- A consistent naming convention to use in Matlab analysis programs (including sub-channels) so code is easily readable
  - Current:
    - LLRF/C7.ACC2.PROBE [sub-channels I, Q]
    - LLRF.ML/ACC23_AMPL LLRF.ML/ACC23_PHASE [Vector Sum]
  - Proposed:
    - rf_acc2c7_probe.phase   rf_acc2c7_probe.ampl
    - rf_acc23_vs.phase rf_acc23_vs.ampl
- A simplified data structure of returned data
  - Use nicknames and structures:
    - Current: DAQ_data{7}(1:numPulses, 1:numSamples, 3)
    - Proposed: bpm10TCOL.Y(1:numPulses, 1:numSamples)
  - Fill in under-sampled waveforms so all waveforms are 1 -2048 (when needed)
- Frequently needed algorithms written once and shared …
  - Conversion from I/Q to phase & amplitude
  - Fill in arrays (see above)
  - Find sample #'s where beam is present
  - Find sample #'s where RF is at the flattop

Possible Results …

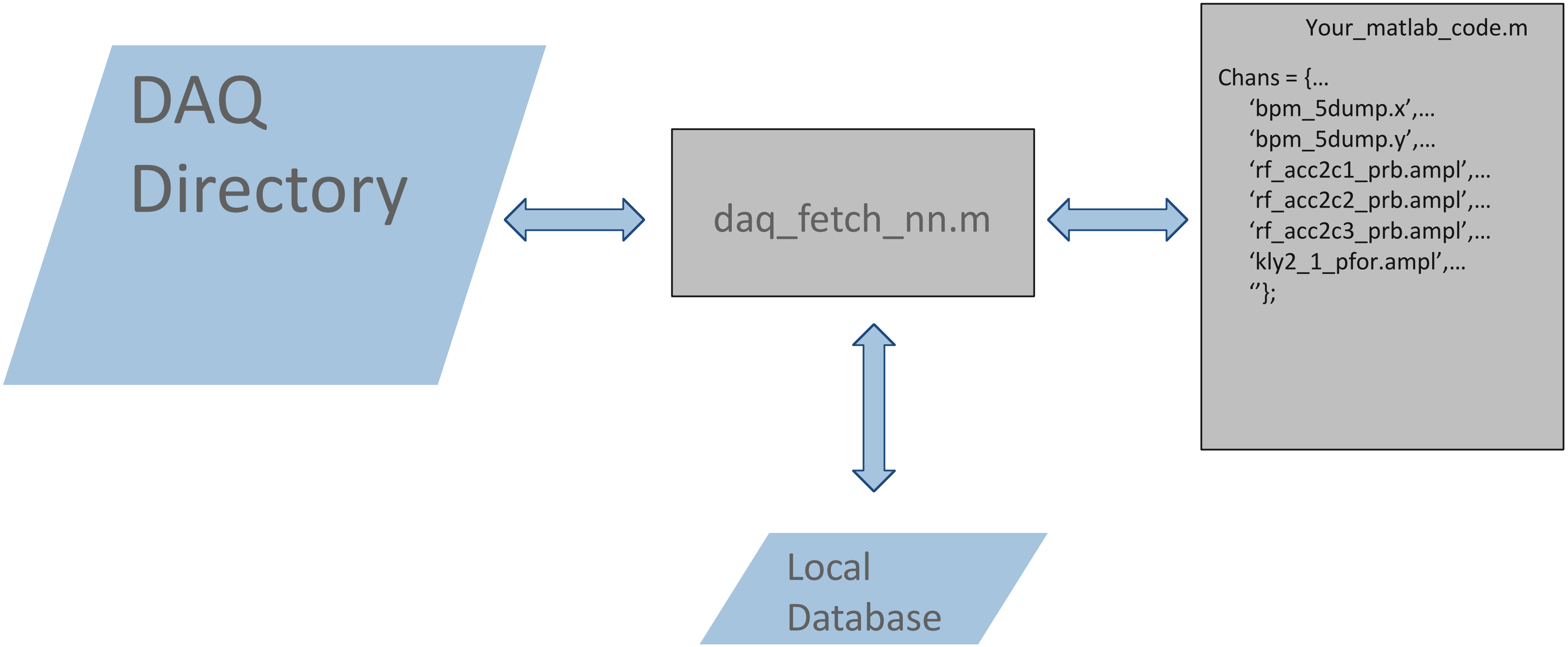
- Typical sequence for interactive Matlab analysis …
  - **Fetch and store data** using DOI Version 2
    - Select channel names interactively, via groups, or by selecting an analysis program
  - In matlab …

```
>> load <myfile>
>> assignNicknames;
>> preprocessSubchannels;
>> findBeamSamples;
>> plot(energyDump(:, samplesWithBeam), bpm10TCOL.X(:, samplesWithBeam));
```

What’s required?

- A local relational database (RDB) to relate …
  - Channel names to sub-channels
  - Sub-channels to nicknames
  - Specify “special processing” for certain sub-channels
  - Keep track of sample # offsets for each channel
- An enhanced daq_fetch.m that …
  - Allows scripts to use real channel names or nicknames
  - Interacts with the database to determine nicknames/sub-channels/channels
  - If data is not returned for a given channel, gives the user an option to fetch all other channel data (eliminates bad channels from the list)
- An enhanced DOI
  - Allows users to view and retrieve channel names, sub-channel names and nicknames from RDB
- Commonly used analysis scripts in a centralized location

Nicknames to DAQ names



```
Chans = {…
    ‘bpm_5dump.x’,…
    ‘bpm_5dump.y’,…
    ‘rf_acc2c1_prb.ampl’,…
    ‘rf_acc2c2_prb.ampl’,…
    ‘rf_acc2c3_prb.ampl’,…
    ‘kly2_1_pfor.ampl’,…
    ‘’};
```

Useful Information in a Database

DAQ Channel Name	Description	Sys	Dimensions (returned by daq_fetch)	Sub-channel Nicknames (structures)	Samples Return’d	Processing Req’ts
BLM/11BC3	BLM @ 11BC3	BLM	1 – Sample #’s 2 – Voltage	blm_11bc3	0:1599	None
BPM/5DUMP	BPM @ 5DUMP	BPM	1 – Sample #’s 2 – X 3 – Y	bpm_5dump.x bpm_5dump.y	690:1699	None
LLRF/C3.ACC2.PROBE	ACC2 C3 Probe	ACC2	1 – Sample #’s 2 – I 3 – Q 4 – phase 5 - amplitude	rf_acc2c3_prb.ampl rf_acc2c3_prb.ph	0:2:2046	Type 1 – I/Q
LLRF/C3.ACC2.PFOR	ACC2 C3 Fwd Pwr	ACC2		rf_acc2c3_pfor.ampl rf_acc2c3_pfor.ph		Type 1 – I/Q
LLRF/C3.ACC2.PREFL	ACC2 C3 Refl Pwr	ACC2		rf_acc2c3_prefl.ampl rf_acc2c3_prefl.ph		Type 1 – I/Q
KLY.ADC/kly2.1	Klystron 2 Parameters	KLY	1 – Sample #’s 2 – PFOR 3 – PREFL 4 – PFOR Circulator 5 – PREFL Circulator 6 – Klystron Voltage 7 – Klystron Current 8 – PFOR Preamp Out 9 – Power Meter Trigger	kly2_1_pfor.ampl kly2_1_prefl.ampl kly2_1_circ_pfor.ampl kly2_1_circ_prefl.ampl kly2_1.voltage kly2_1.current kly2_1_preamp_pfor.ampl	0:1:2047	None

Comments on the use of a database

- Advantages
  - With ‘sqlite’ (a “serverless” RDB), the ‘database’ is a single file that can be easily backed up and shared
  - Provides a central place to store answers to the question “What’s in the DAQ?” (for 9ma experiments)
  - Provides a consistent naming convention … at least for DAQ retrieval to matlab
  - Easier to read others’ code
  - Eases the learning curve!
- Disadvantages / Concerns
  - Yet another name space (DOOCS, DAQ, 9ma-DAQ, …)
    - Stay with channel names and just add an extension for sub-channels?
  - Can we keep the database accurate?
    - What happens if the sub-channels, # of samples, etc change between ‘runs’? Do we need entries for each run? (it could get ugly! … but better to do it one-time-for-all)
  - How long will it take to have a useful environment?
    - Do you want “Quick and Ugly” or “Well designed for long term support”