Hexanode detector test on data

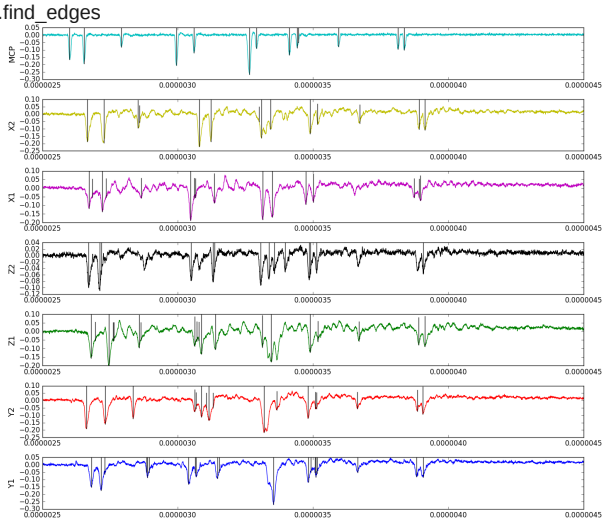
Content

LCLS data of hexanode detector

- exp=xpptut15:run=390 - amod3814, runs 85, see Publicly Available Practice Data

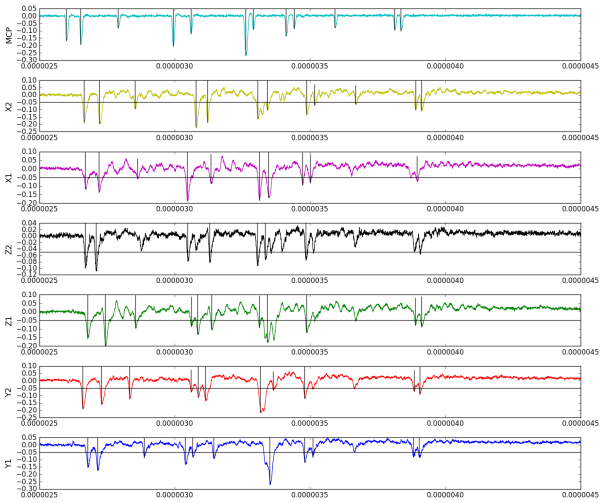
Waveform processing with CFD method

- algorithm psalg.find_edges



    - as is:
    - problems
        - definition of rising:
          bool rising = threshold_value > baseline_value; // does not allow to use real baseline...
        - finds many signals in crossing of threshold forth and back due to noise - add constrain on signal width as: width>deadtime
        - bug in sign ➕ for fraction*(peak-baseline_value)

- fixed:



Examples

- `expmon/src/HexDataIO.py` - has implementation of methods to access psana data like it is done `in hexanode/src/LMF_IO.cpp` for LMF file format.
- `hexanode/examples/ex-05-sort.py` - example of LMF data processing in python equivalent to firmware example `sort`
- `hexanode/examples/ex-06-sort-graph.py` - example of LMF data processing with matplotlib graphicas
- `hexanode/examples/ex-07-sort-graph-data.py` - example of processing LCLS data with graphics - slow 30Hz, but 200Hz for cached data...
- `hexanode/examples/ex-08-proc-data-save-h5.py` - acqiris waveform processing and saving in hdf5 file

Data processing

exp=xpptut15:run=390 - amod3814, runs 85, see Publicly Available Practice Data

`hexanode/examples/ex-07-sort-graph-data.py`

Calibration

- sorter_data_cfg.txt - configuration file
  - 2 = calibrate fu,fv,fw, w_offset - process data in this mode, adjust scale factors (using values printed at the end of data processing) and set parameters to get time sum peak at 0:

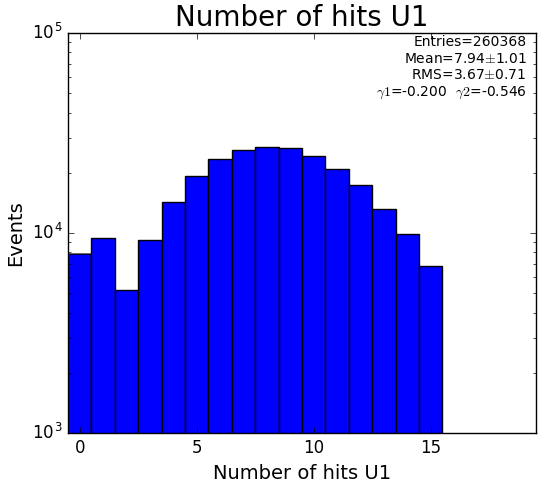
time sum offsets

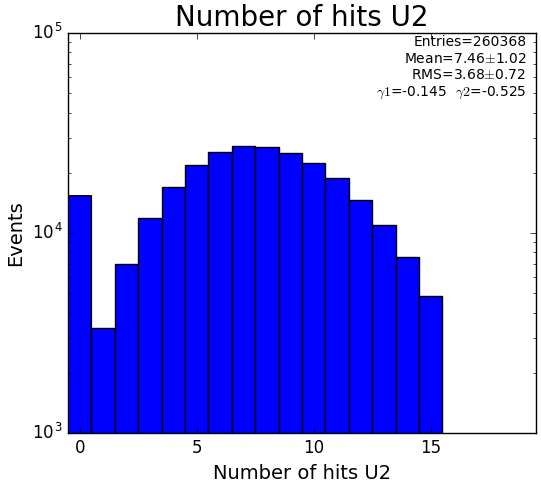
```
-138.5  // offset to shift timesum layer U to zero (in nanoseconds)
-148.3  // offset to shift timesum layer V to zero (in nanoseconds)
-135.5  // HEX ONLY: offset to shift timesum layer W to zero (in nanoseconds)

0.680   // scalefactor for layer U
0.6795  // scalefactor for layer V
0.702   // HEX ONLY: scalefactor for layer W
```

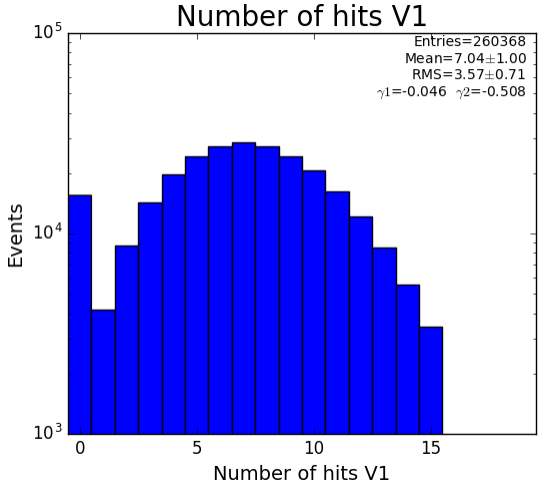
  - 3 = generate correction tables and write them to disk - processing in this mode generates correction tables saved in the file calibration_table_data.txt
  - 1 = sort - normal mode to get calibrated data

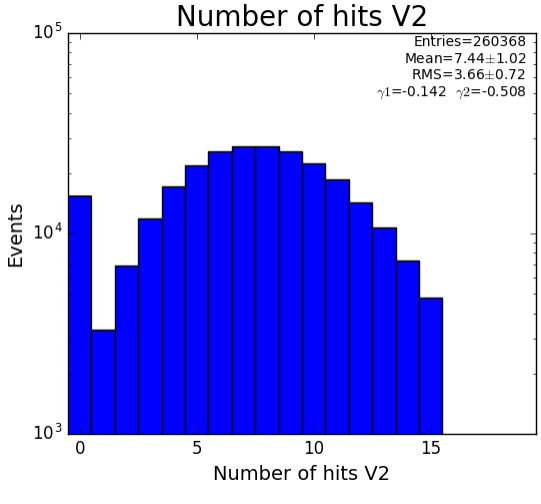
Graphics

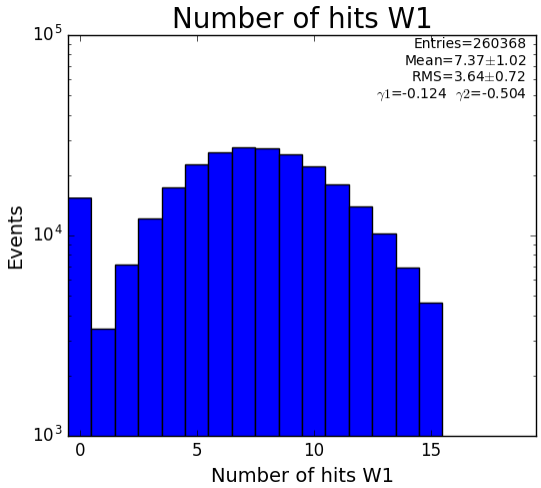
Number of hits per channel

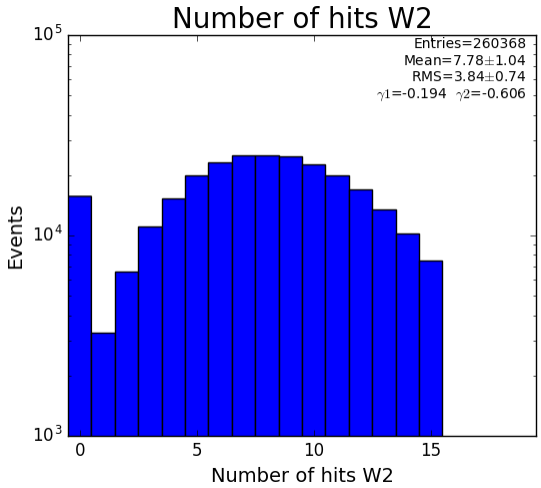

Number of hits U1
Entries=260368
Mean=7.94±1.01
RMS=3.67±0.71
γ1=-0.200 γ2=-0.546
Events
Number of hits U1

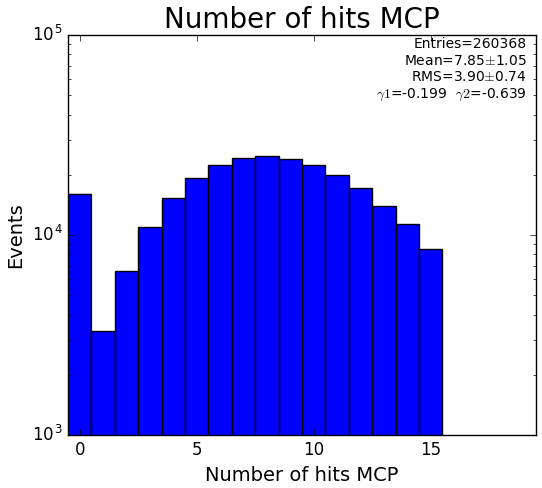

Number of hits U2
Entries=260368
Mean=7.46±1.02
RMS=3.68±0.72
γ1=-0.145 γ2=-0.525
Events
Number of hits U2


Number of hits V1
Entries=260368
Mean=7.04±1.00
RMS=3.57±0.71
γ1=-0.046 γ2=-0.508
Events
Number of hits V1

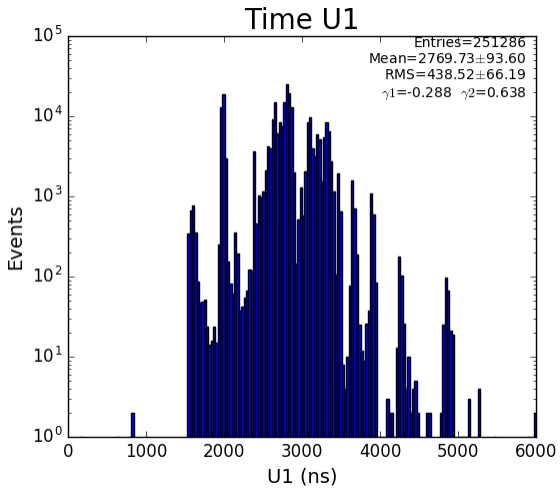

Number of hits V2
Entries=260368
Mean=7.44±1.02
RMS=3.66±0.72
γ1=-0.142 γ2=-0.508
Events
Number of hits V2

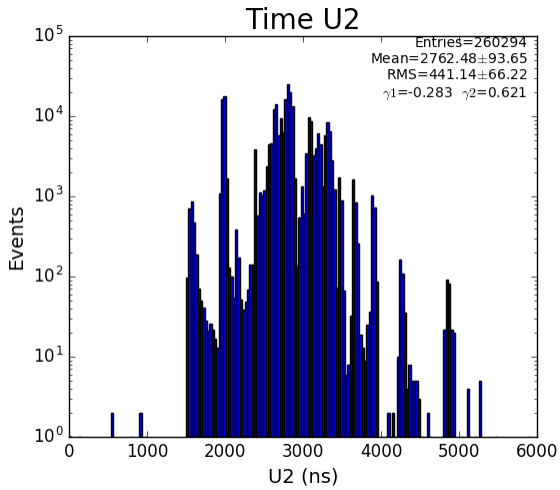

Number of hits W1
Entries=260368
Mean=7.37±1.02
RMS=3.64±0.72
γ1=-0.124 γ2=-0.504
Events
Number of hits W1

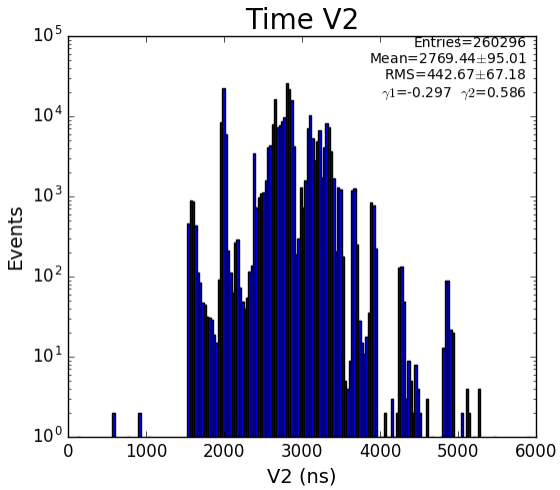

Number of hits W2
Entries=260368
Mean=7.78±1.04
RMS=3.84±0.74
γ1=-0.194 γ2=-0.606
Events
Number of hits W2

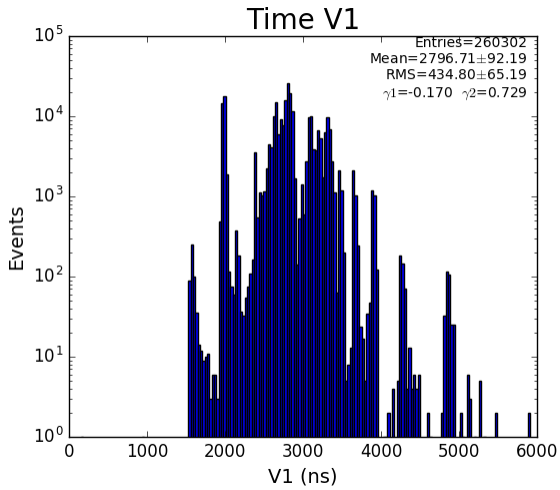


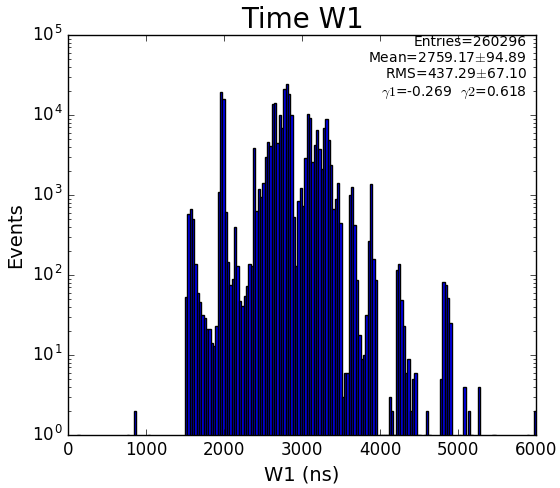
Spectra of time per channel

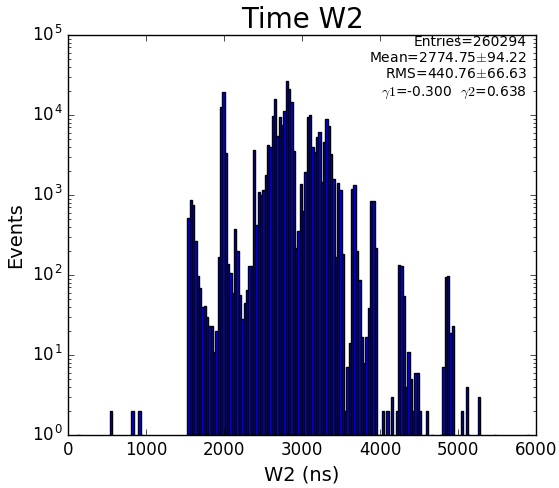


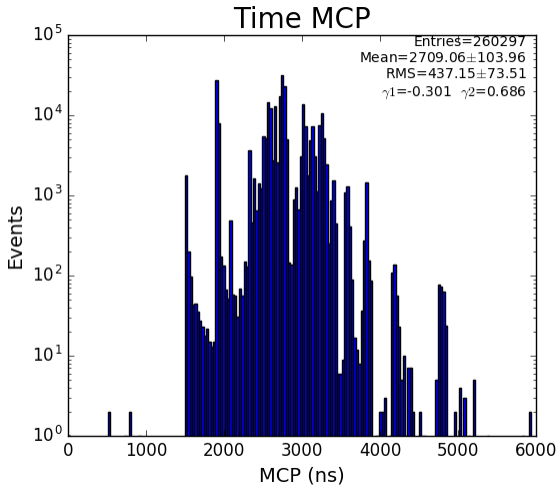




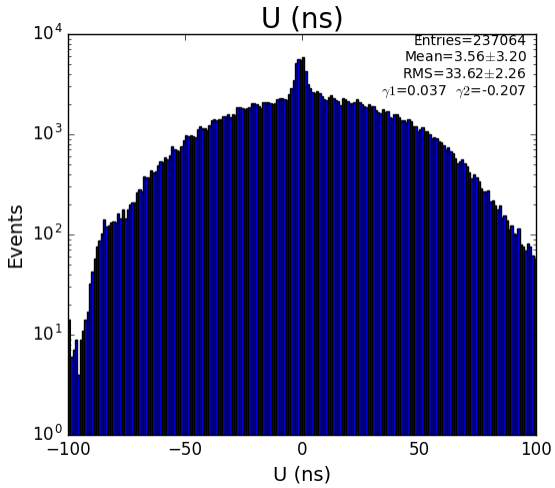


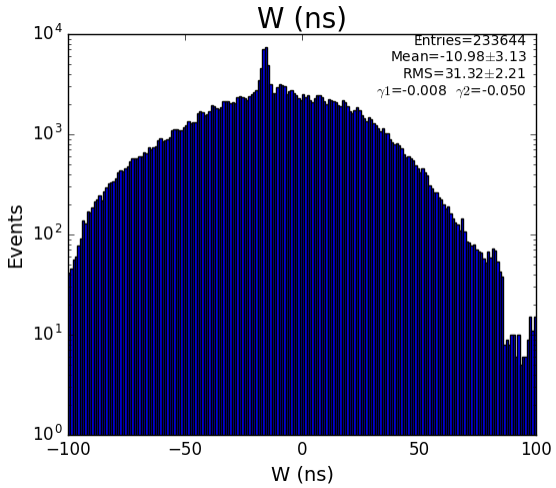


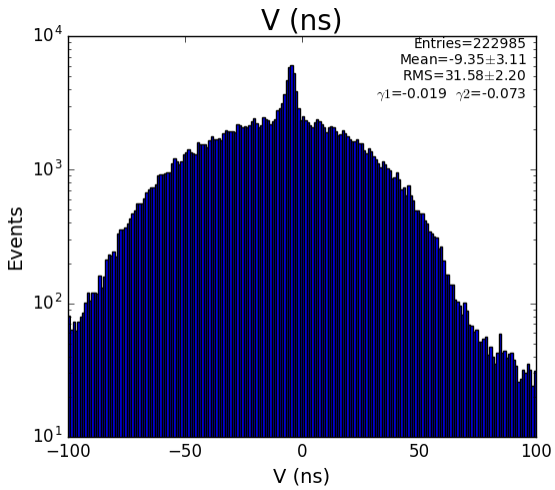




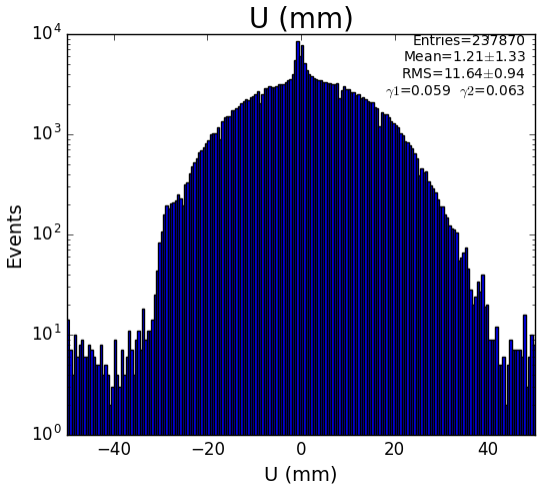
Spectra of U, V, W (ns)

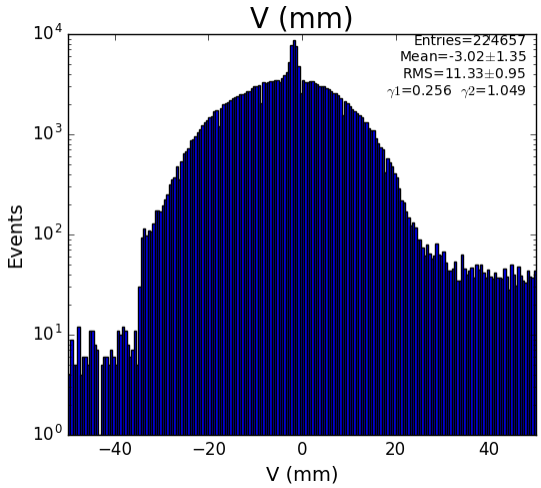


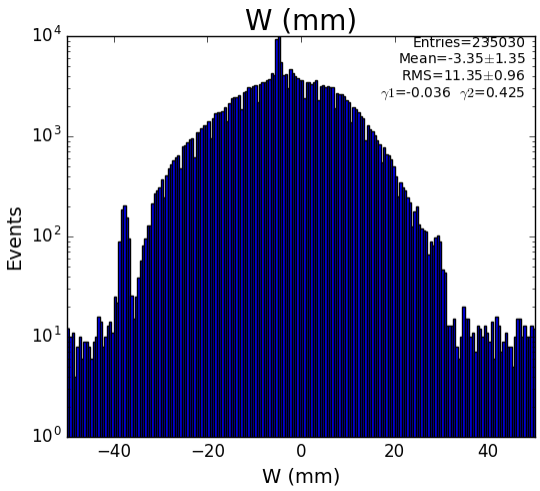




Spectra of U, V, W (mm)







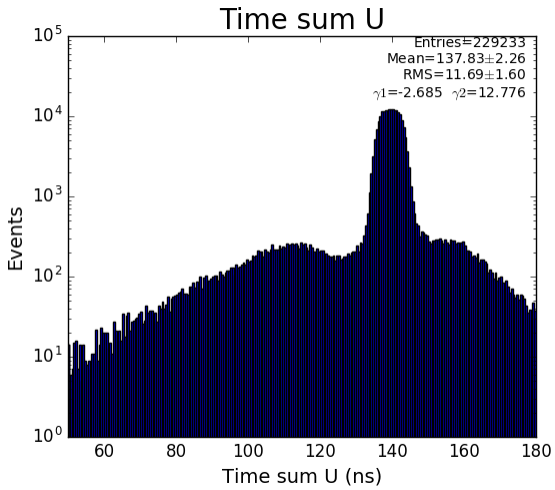
Spectra of Xuv, Xuw, Xvw (mm)

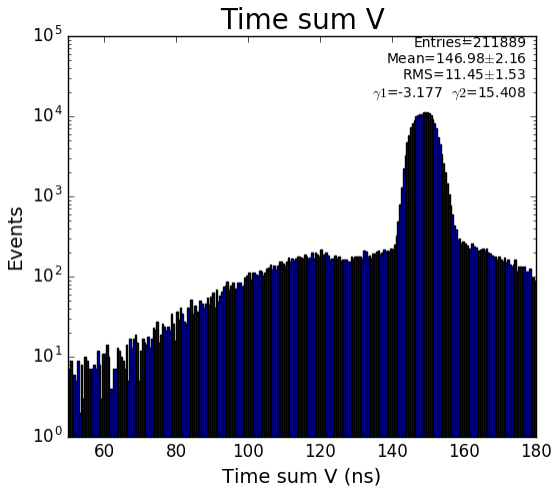
- about the same as above spectra

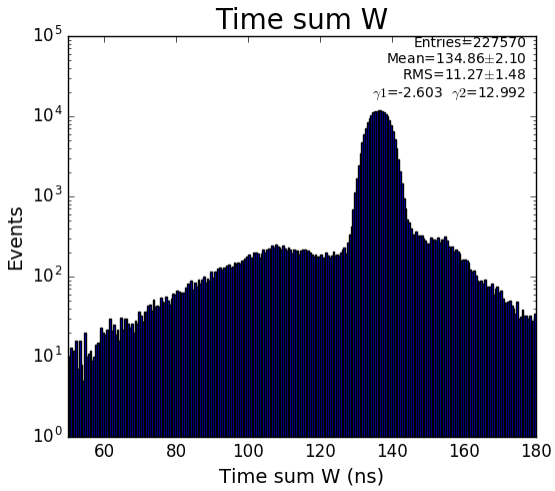
Spectra of Yuv, Yuw, Yvw (mm)

- about the same as above spectra

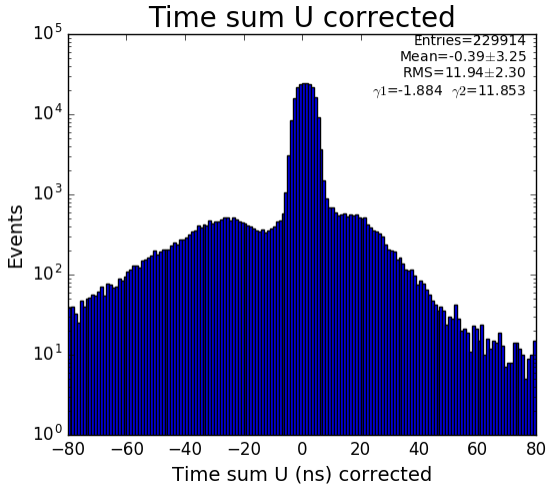
Time sum (ns) for U, V, W

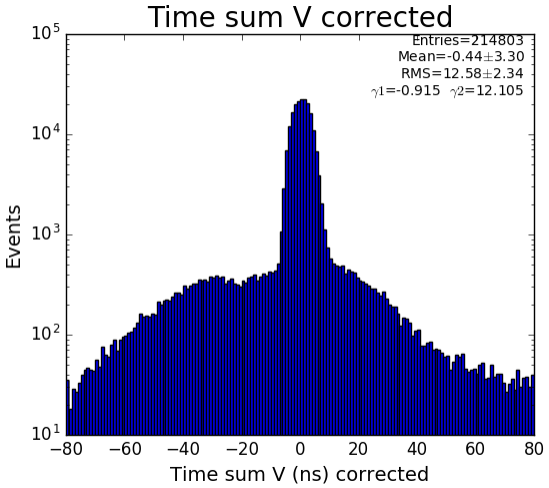


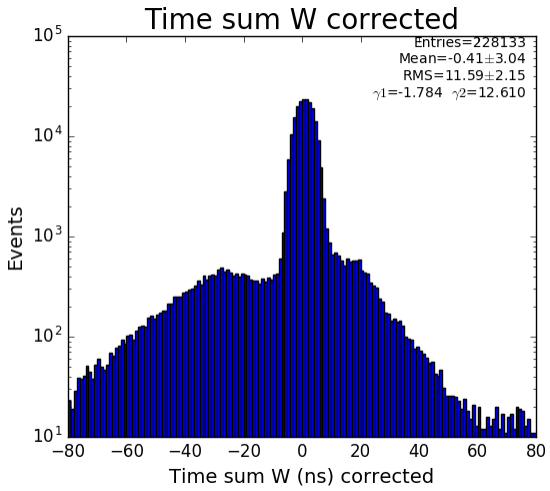




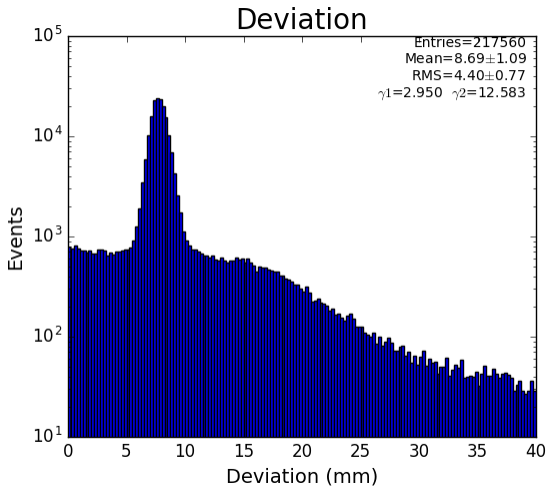
Time sum (ns) corrected for U, V, W

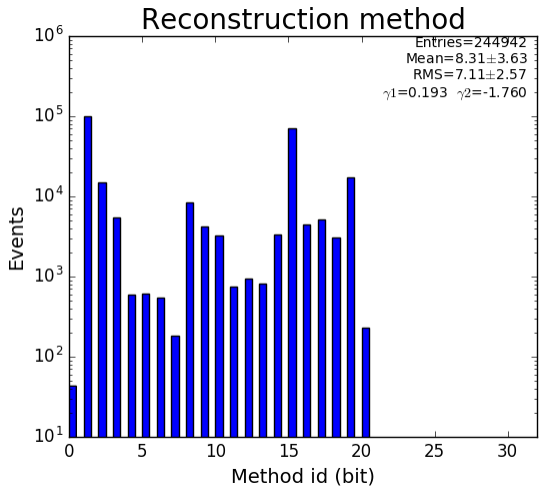


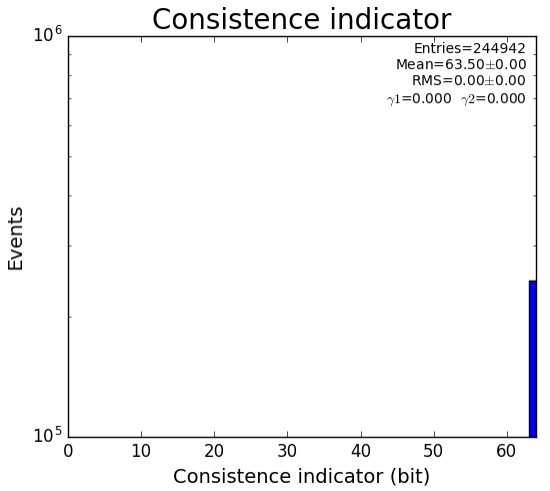




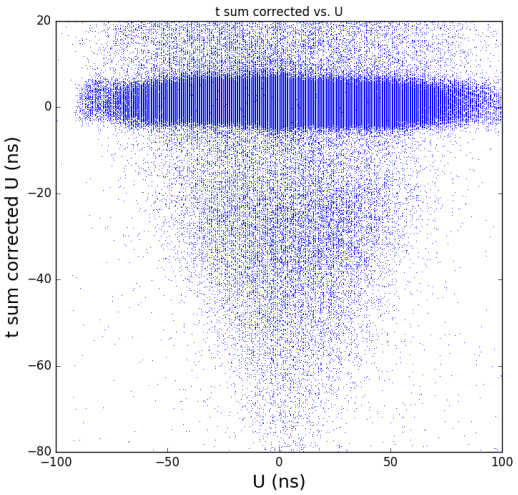
Deviation, Consistency Indicator, Reconstruction method

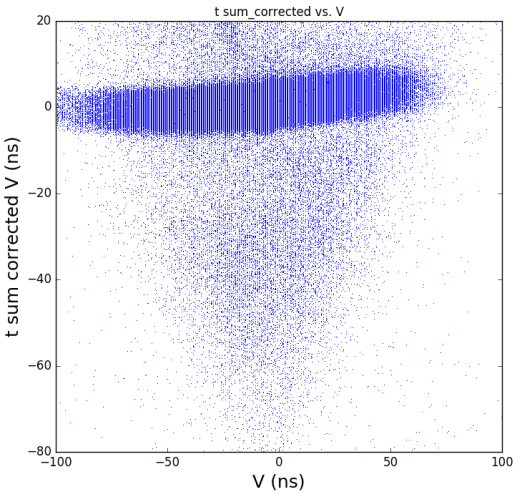


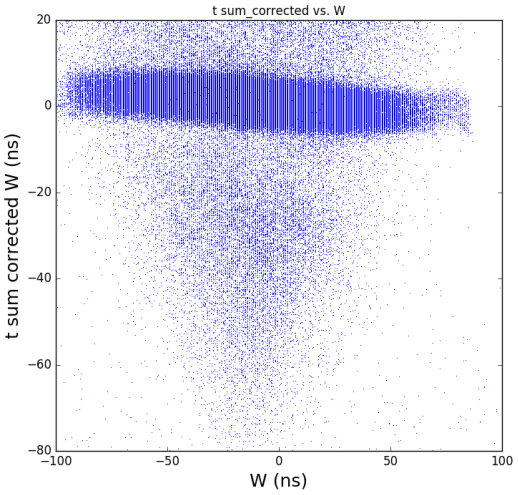




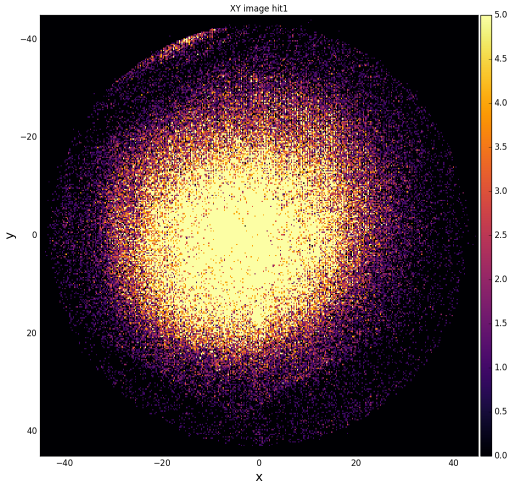
Time sum vs. variable U, V, W

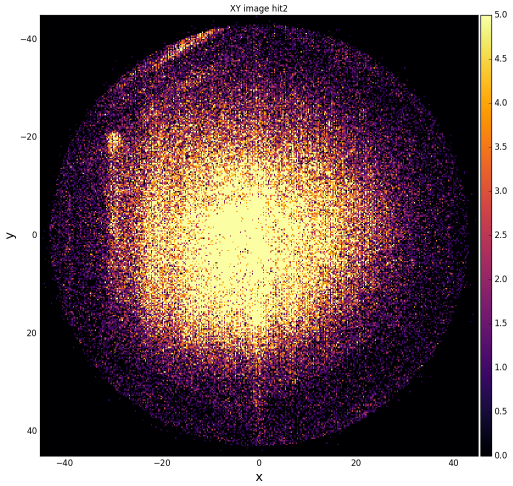




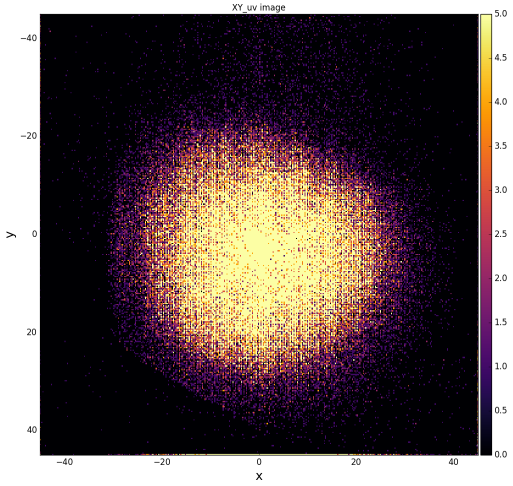


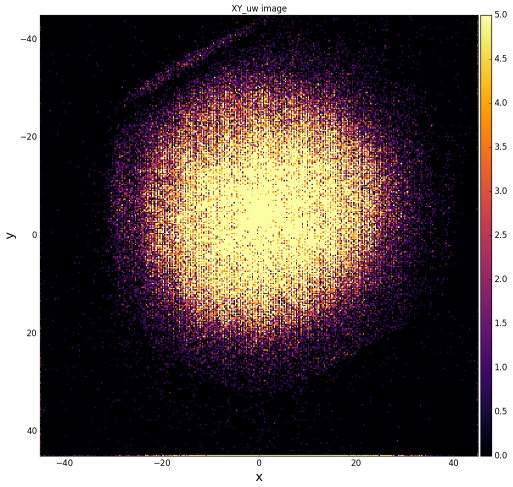
xy image for hit1 and 2

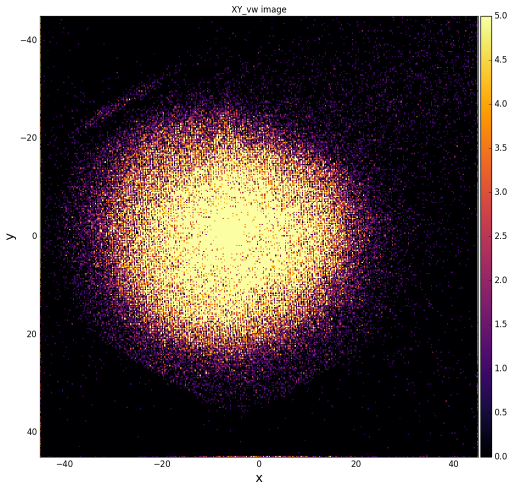




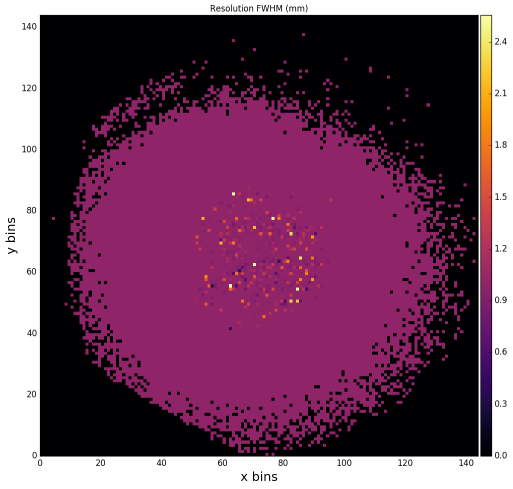
XY image for uv, uw, and vw components



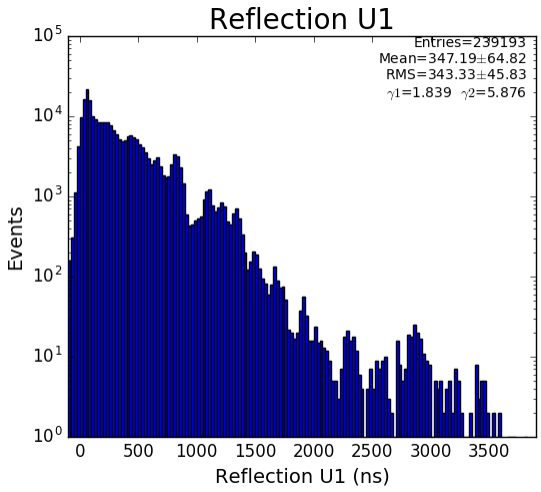


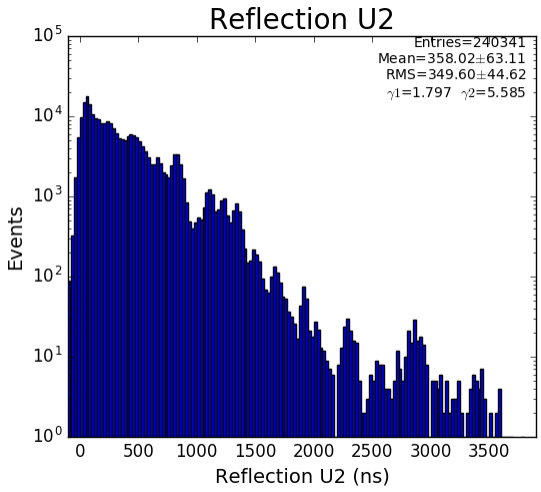


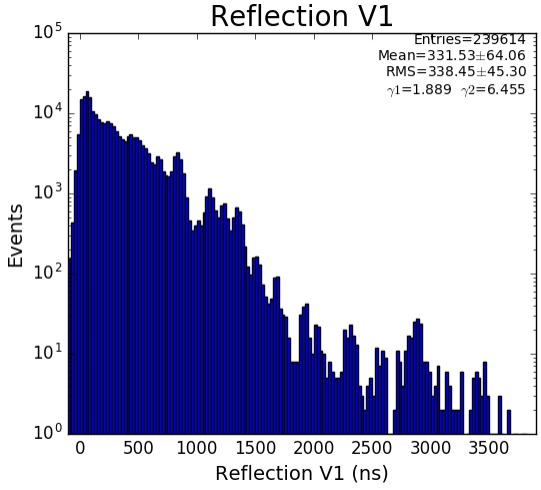
Resolution map

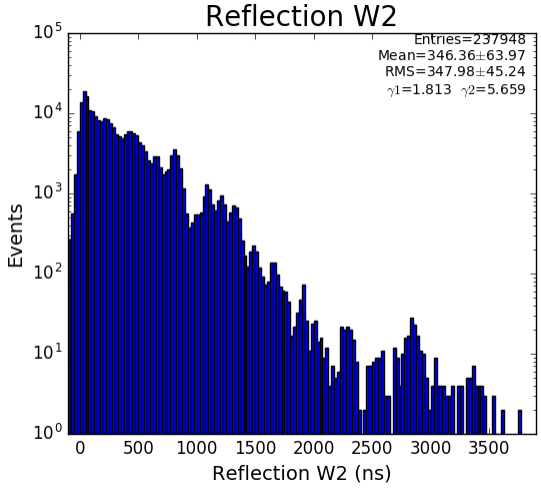


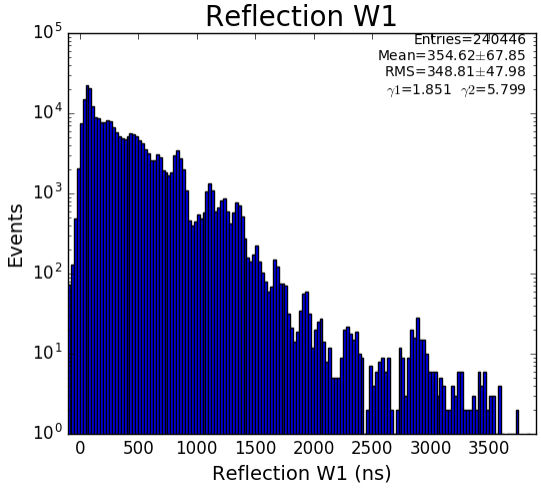
Reflection for all channels

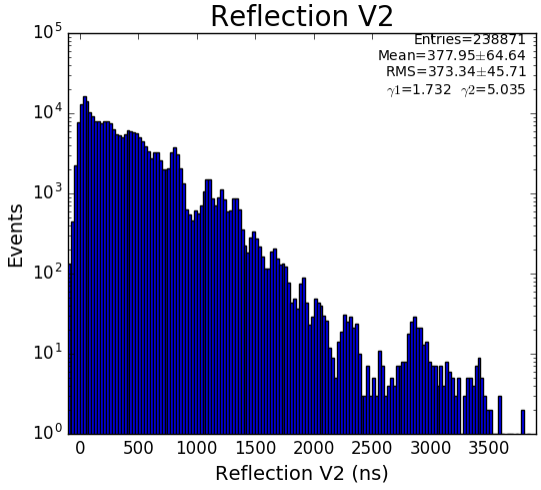




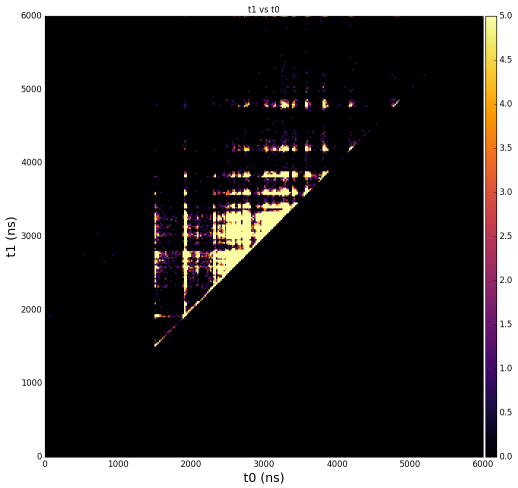


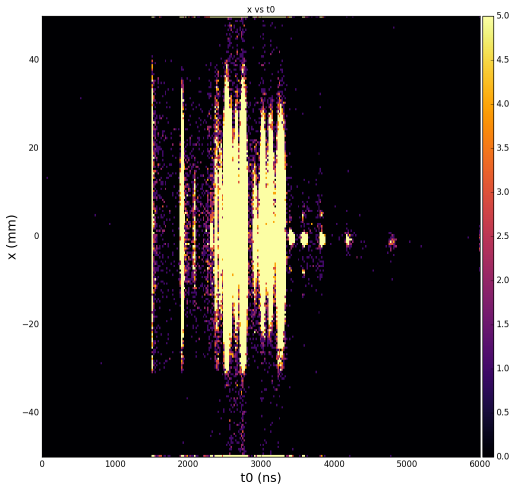


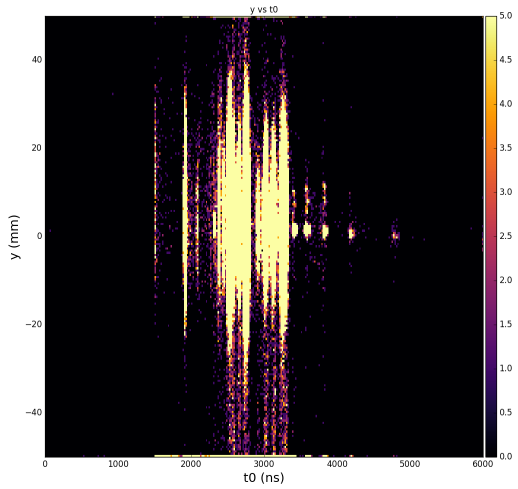




Physics plots t1,x,y vs t0







Problem

- slow data processing 30Hz, but 200Hz for cached data (1-st 1000 events at repeated processing)
- Presumably it is due to slow access to psana data

Solution

Interactive job performance

- Script hexanode/examples/ex-11-MPIDS-save-h5.py - generates hdf5 file using MPIDataSource and smalldata classes
- Speed of processing test on psanaphi110 for entire sample of 260386 events:

Number of core (-n)	Processing time (sec)	Processing frequency (Hz)
2	2018	129
4	3613	72
8	708	367
16	597	430
16	1985	131
32	531	490
64	3001	131

- Script hexanode/examples/ex-07-sort-graph-data.py -processes hdf5 file and generates a bunch of plots

MPI job in batch

Command to process 260368 events on psnehq (one on psanaq), sending one job in queue in order to not compete for the same data:

```
bsub -o log-mpi-n16-%J.log -q psnehq -n 16 mpirun python hexanode/examples/ex-08-proc-MPIDS-save-h5.py
```

Number of cores (-n)	Processing time (sec)	Processing frequency (Hz)
1	6977	37
2	1931	135
4	1163	224
8	714	365
16 on psnehq	493	528
on psanaq	658	357
32	1046	249
	732	356
64	288	904
	329	791

References

- 2016-11-30-HexAnodeSoftware.pdf
- 2016-12-07-email-achim-czasch.txt
- Publicly Available Practice Data