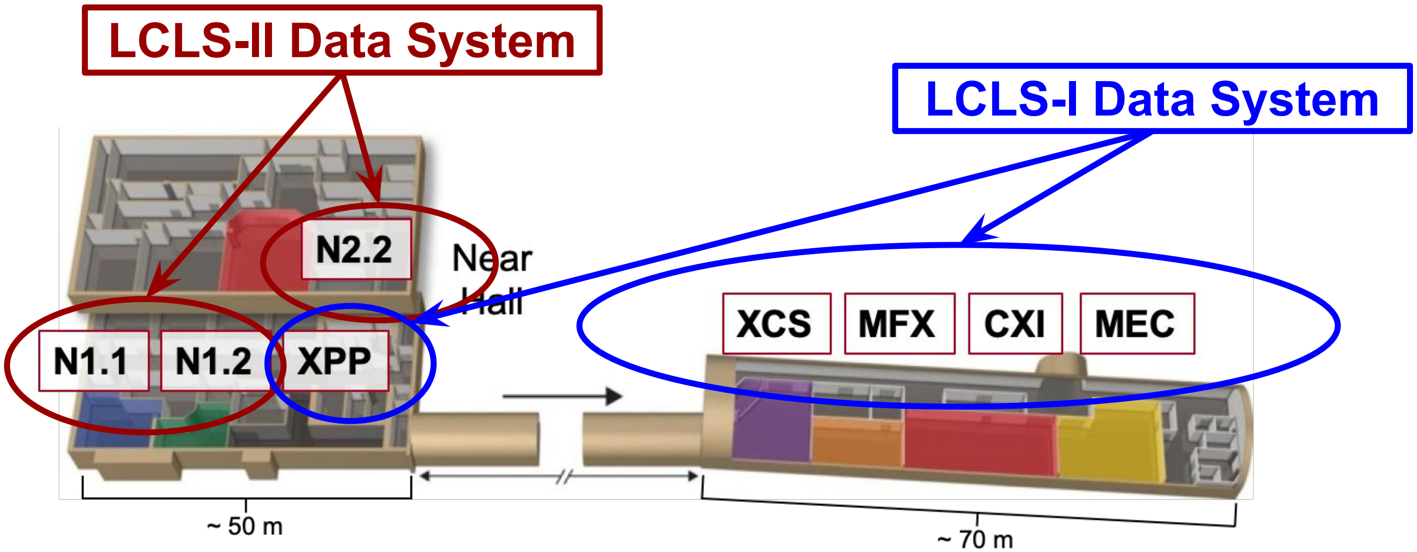
LCLS2 Data Analysis: Discussion with UEC

Christopher O’Grady

SLAC National Accelerator Laboratory

Aug. 25, 2023

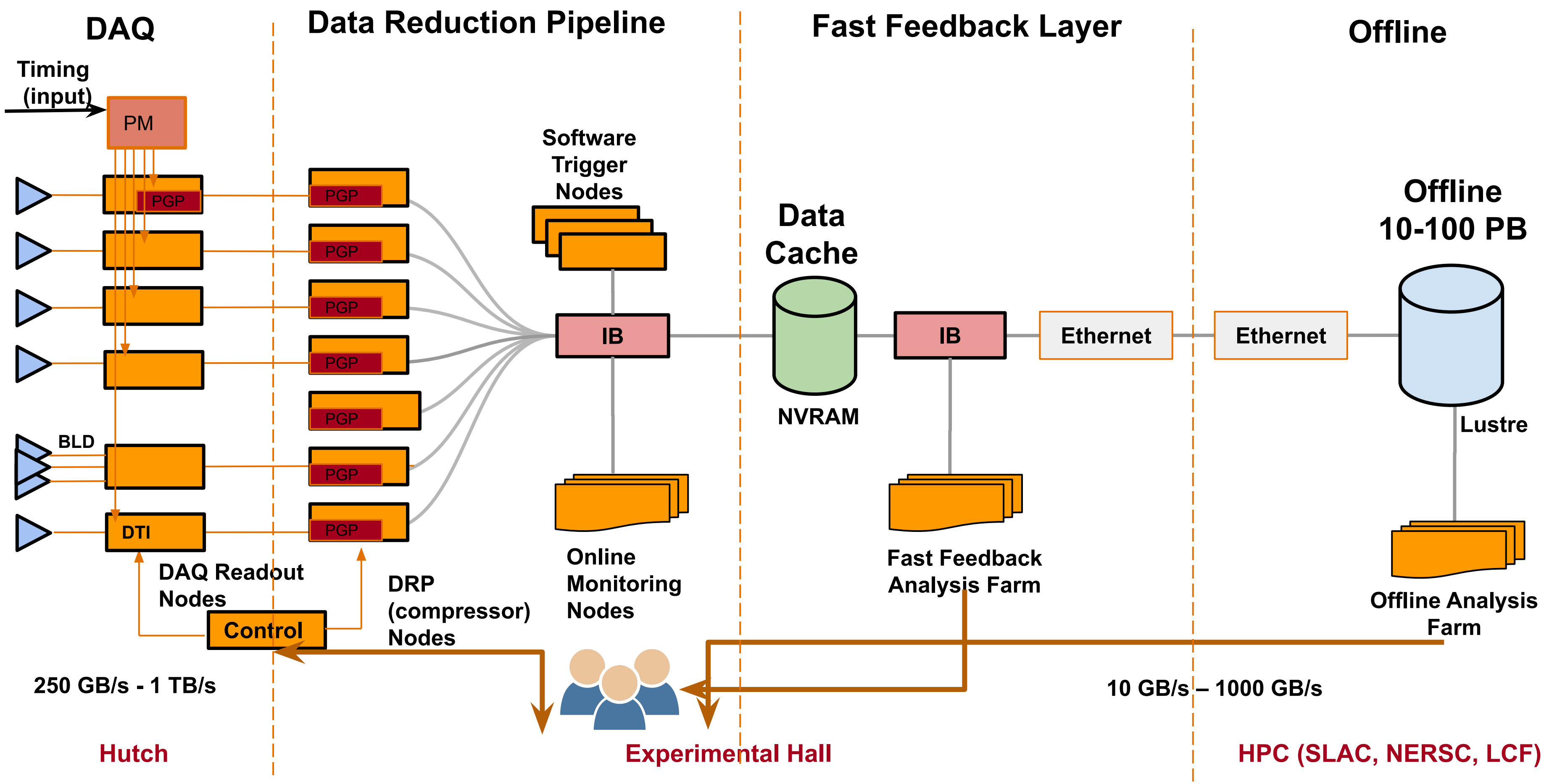
LCLS-II plans



The LCLS-II Data System is the focus.

SXU	NEH 1.1:	Atomic, Molecular and Optical	2 Soft X-ray
SXU	NEH 2.1:	Resonant Inelastic X-ray Scattering	2 Soft X-ray
SXU	NEH 2.2:	Soft X-ray Research	2 Soft X-ray
SXU / HXU	NEH 1.2:	Tender X-ray Instrument	1.5 “Tender” X-ray
HXU	XPP:	X-ray Pump Probe	4.5 Hard X-ray
HXU	XCS:	X-ray Correlation Spectroscopy	4.5 Hard X-ray
HXU	MFX:	Macromolecular Femtosecond Crystallography	4.5 Hard X-ray
HXU	CXI:	Coherent X-ray Imaging	4.5 Hard X-ray
HXU	MEC:	Matter in Extreme Conditions	4.5 Hard X-ray

Data System Overview



Analysis/DAQ Information

Analysis/DAQ user documentation:

https://confluence.slac.stanford.edu/display/LCLSIIData/LCLS-II+Data+Acquisition+and+Analysis

New Shared Computing Facility ("S3DF"):

https://confluence.slac.stanford.edu/display/PCDS/Running+at+S3DF

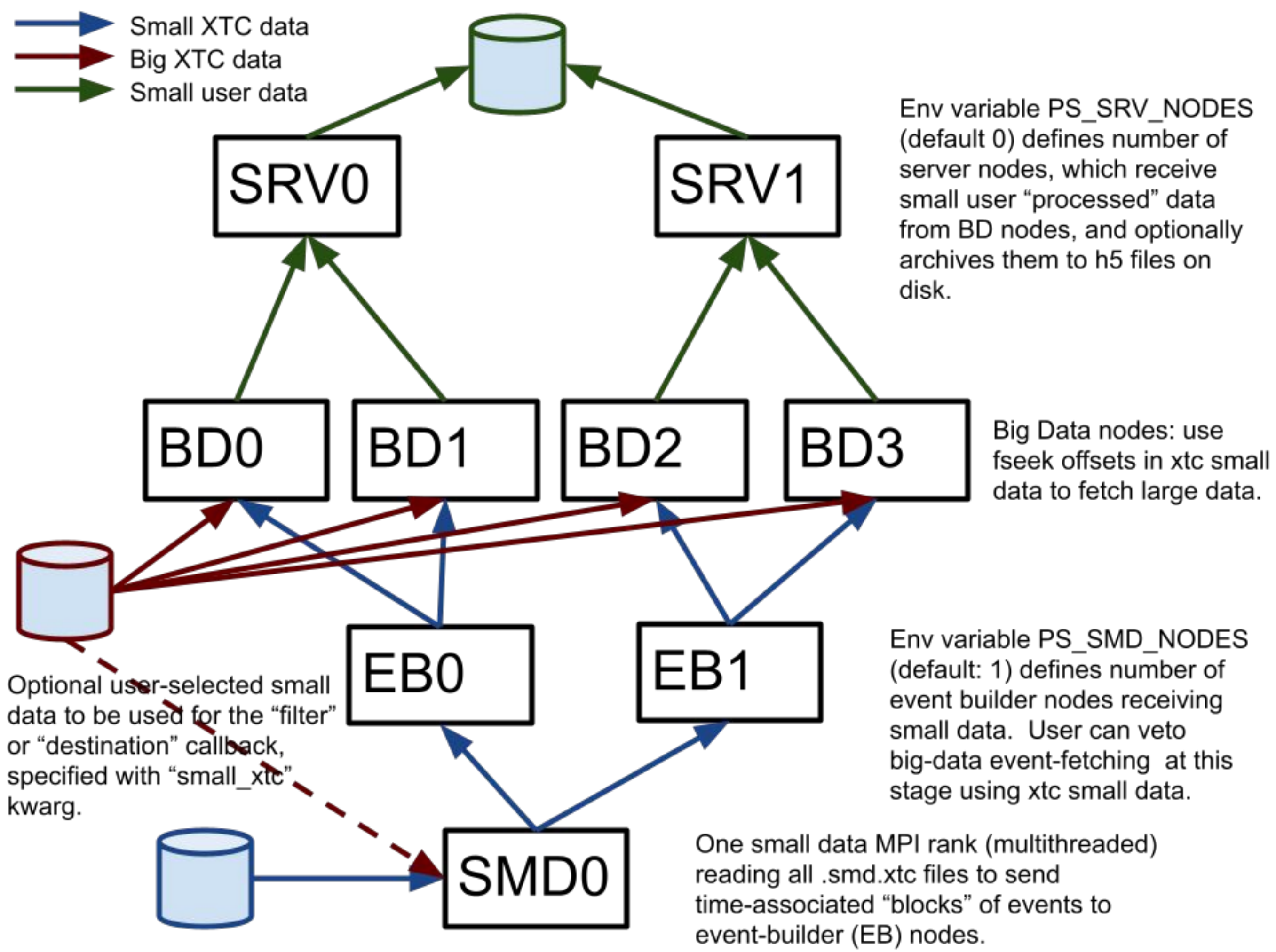
Analysis (“psana”) Big Ideas

- “**Perfectly Parallel**”: different events given to different cores using world-standard MPI parallelization. Scalable to 1MHz.
- Detector **calibrations stored in MongoDB/GridFS** for read-only access at HPC centers around the world
- **Same scripts works everywhere**: real-time and offline analysis (and embedded in DAQ!)
- **Hide complexity** that users don’t need to see: MPI parallelization, HDF5 production, detector corrections…
- Raw data (messy) in XTC2 format, Calibrated data (neat) in **much smaller** HDF5
- **psana is the fundamental LCLS analysis tool**. other tools exist on top: e.g. crystfel/cctbx, OM, smalldata_tools… (https://github.com/slac-lcls/smalldata_tools)

Example psana Script

```
from psana import DataSource
import numpy as np
import os
def my_smalldata(data_dict):
    print(data_dict)
os.environ['PS_SRV_NODES']='1'
ds = DataSource(exp='tmoc00118', run=222)
smd = ds.smalldata(filename='mysmallh5.h5', callbacks=[my_smalldata])
for run in ds.runs():
    opal = run.Detector('tmo_opal1')
    ebeam = run.Detector('ebeam')
    runsum  = 0
    for evt in run.events():
        img = opal.raw.image(evt)
        photonEnergy = ebeam.raw.ebeamPhotonEnergy(evt)
        if img is None or photonEnergy is None: continue
        evtsum = np.sum(img)
        # pass either dictionary or kwargs
        smd.event(evt, evtsum=evtsum, photonEnergy=photonEnergy)
        runsum += evtsum # beware of datatypes when summing: can overflow
    if smd.summary:
        tot_runsum = smd.sum(runsum) # sum (or max/min) across all mpi cores
        # pass either dictionary or kwargs
        smd.save_summary({'sum_over_run' : tot_runsum}, summary_int=1)
    smd.done()
```

psana MPI Task Structure

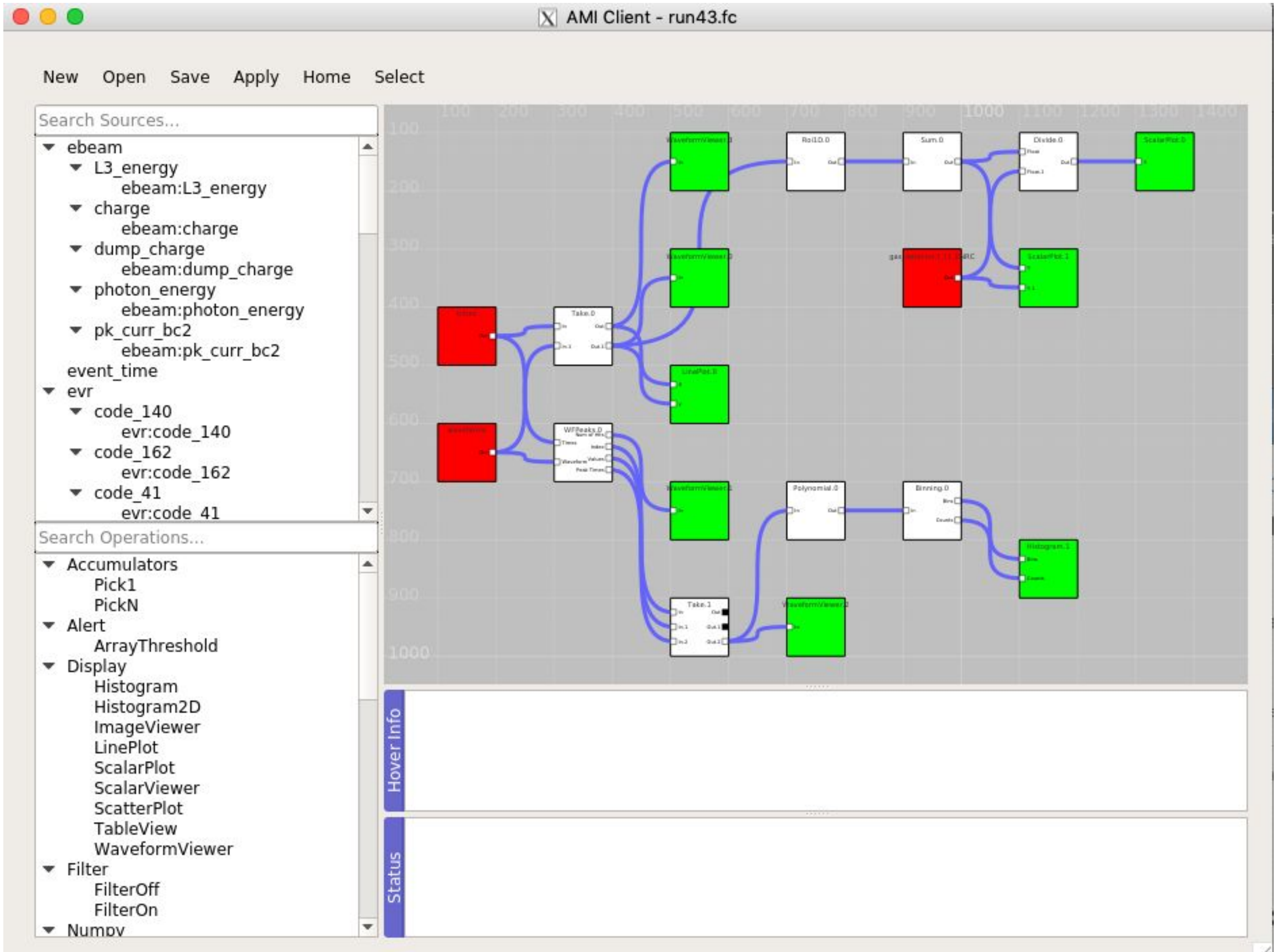


AMI Big Ideas

- Graphical **LabView-style** analysis
- **Scalable**: multi-node/multi-core
- **Based on psana-Python**
  - As soon as detector is supported in psana, it works immediately in AMI
- Easily **extendable by users** with Python
- Can be **run online/offline** (with MPI parallelization)
- **8-minute video tutorial** and many offline examples that anybody can run here:

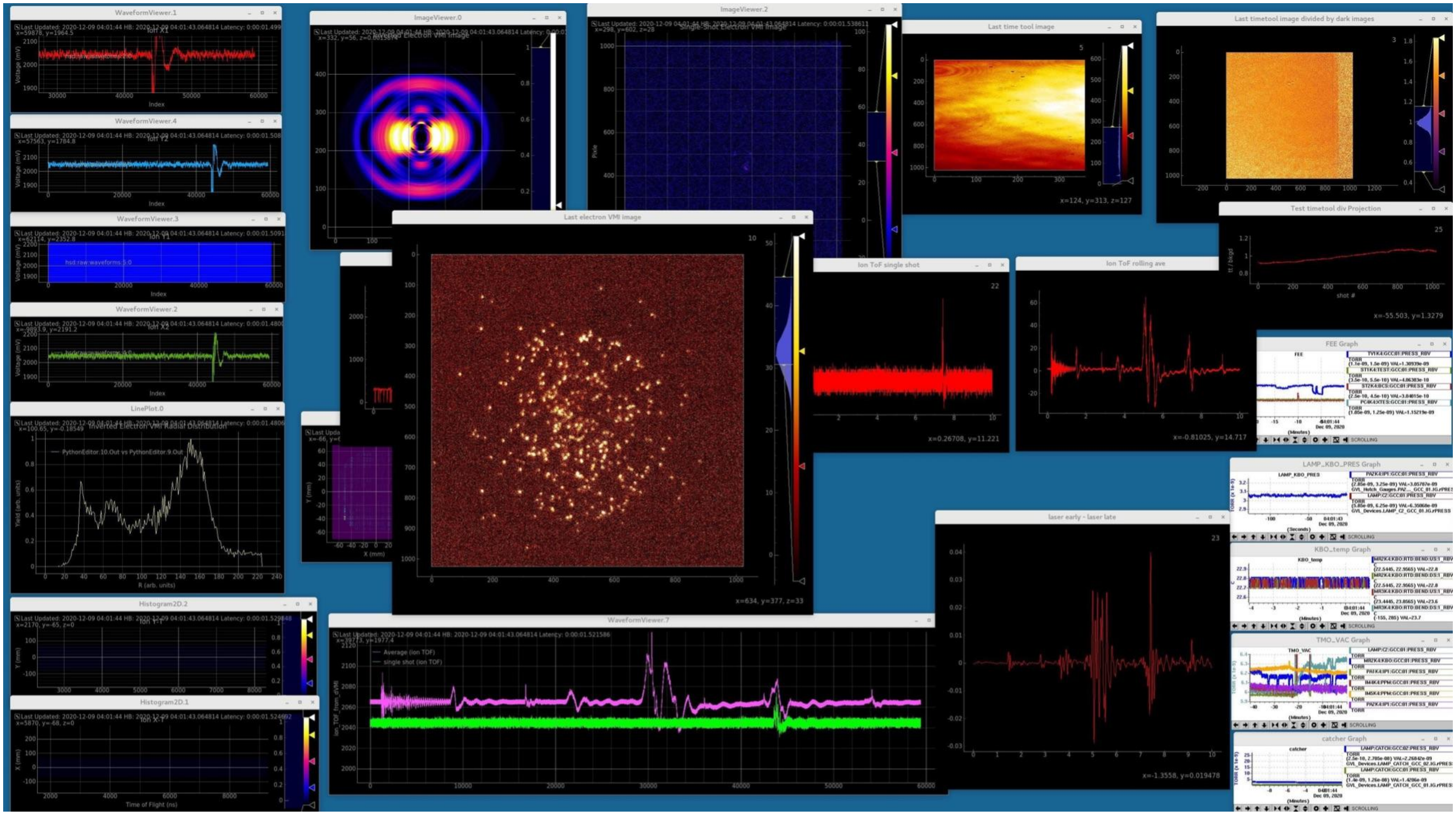
https://confluence.slac.stanford.edu/display/LCLSIIData/ami#ami-HowToLearnAMI

Real-time Analysis with AMI



Real-time (~1s latency) TMO Analysis With AMI

SLAC



Data Selection and Retention

- Data selection:
  - DAQ: aiming for 200GB/s real-time reduction to 20GB/s
    - start simple: thresholding/triggering
    - studying lossy compression for a variety of experiments (https://confluence.slac.stanford.edu/display/PSDMInternal/SZ+Compression)
    - save a fraction of events with un-reduced data
  - second reduction step: users select which data they want in hdf5
    - per-event hdf5 data may not work as well at high rates?
    - dask is an option for larger-scale hdf5 analysis (https://confluence.slac.stanford.edu/display/LCLSIIData/DASK)
- Data retention:
  - https://confluence.slac.stanford.edu/display/PCDS/Data+Retention+Policy

Undulator	Instrument	Endstation	Technique	Detector	Detector Size	Detector Rate (Hz)	Data Rate (aggregate) (GB/s)	Ultilization Factor (0-1)	Data Reduction Type (1st Cut)	DR Factor (1st cut)
SXU	NEH 1.1	DREAM	COLTRIMS	Digitizer	800000	100000	160.0	0.75	Zero suppression	0.020
SXU	NEH 1.1	DREAM	Time of Flight	Digitizer	1000000	100000	200.0	0.75	Zero suppression	0.020
SXU	NEH 1.1	LAMP	Time of Flight	Digitizer	1000000	100000	200.0	0.75	Zero suppression	0.020
SXU	NEH 1.1	LAMP	Imaging	SXR Imag. + Digi.	4000000	10000	82.0	0.45	Veto	0.100
SXU	NEH 2.2	LJE	XAS / XES	TES	1000	100000	20.0	0.60	Zero suppression	0.100
SXU	NEH 2.2	LJE	XAS / XES	TES	10000	100000	200.0	0.60	Zero suppression	0.100
SXU	NEH 2.2	LJE	XAS / XES	RIXS-ccd	4096	1000	0.0	0.60	N.A.	1.000
SXU	NEH 2.2	RIXS	IXS / RIXS	RIXS-ccd	4096	1000	0.0	0.60	N.A.	1.000
SXU	NEH 2.2	RIXS	XRD / RXRD	SXR Imaging	1000000	10000	20.0	0.60	ROI	0.100
SXU	NEH 2.2	RIXS	XPCS	SXR Imaging	1000000	10000	20.0	0.60	Compression	0.500
SXU	NEH 1.2	---	X-ray/X-ray	SXR Imaging	1000000	10000	20.0	0.30	ROI	0.100
SXU	NEH 1.2	---	Imaging	epix100-HR + Digi.	4000000	5000	42.0	0.45	Veto	0.100
SXU	NEH 1.2	---	XAS / XES	RIXS-ccd	4096	1000	0.0	0.60	N.A.	1.000
HXU	NEH 1.2	---	X-ray/X-ray	SXR Imaging	1000000	10000	20.0	0.30	ROI	0.100
HXU	NEH 1.2	---	Imaging	epix100-HR + Digi.	4000000	5000	42.0	0.45	Veto	0.100
HXU	NEH 1.2	---	XAS / XES	RIXS-ccd	4096	1000	0.0	0.60	N.A.	1.000
HXU	NEH 1.2	---	Imaging	ePixUHR	4000000	40000	336.0	0.45	Veto	0.100