Offline Software & Processing

Matt Graham, SLAC
HPS DAQ & Offline Readiness Review
June 18, 2014

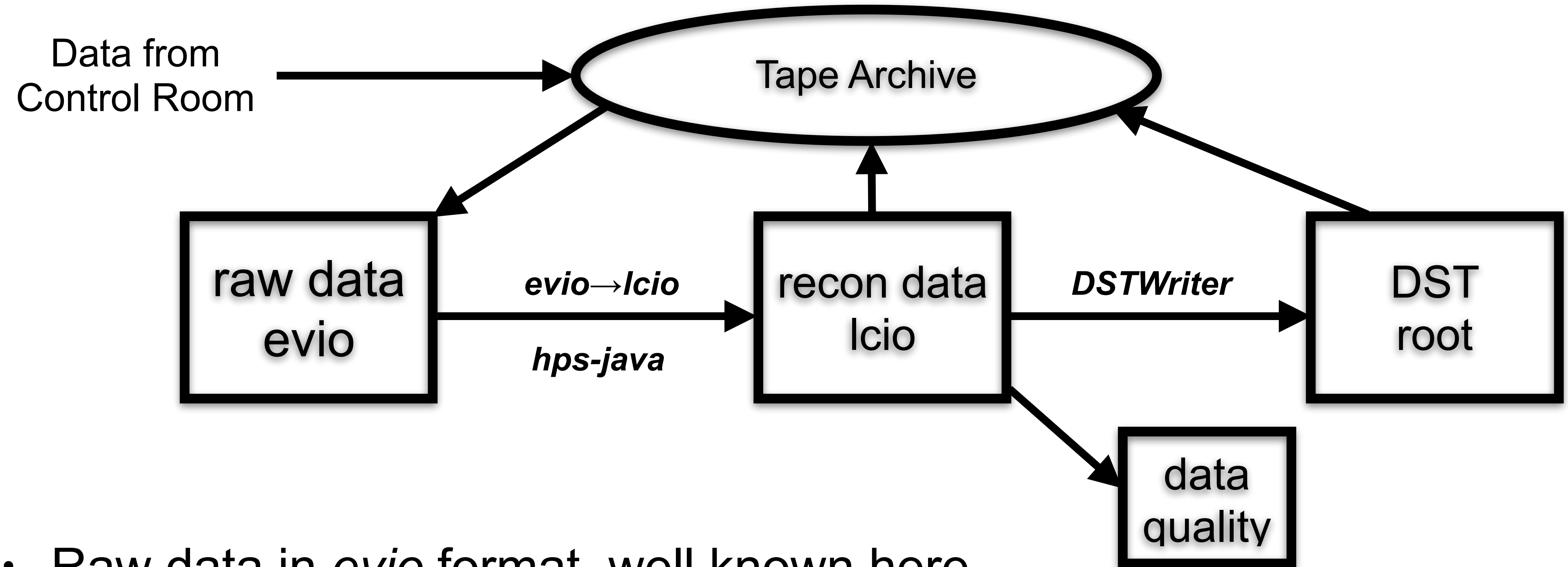
U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Things I will touch on…

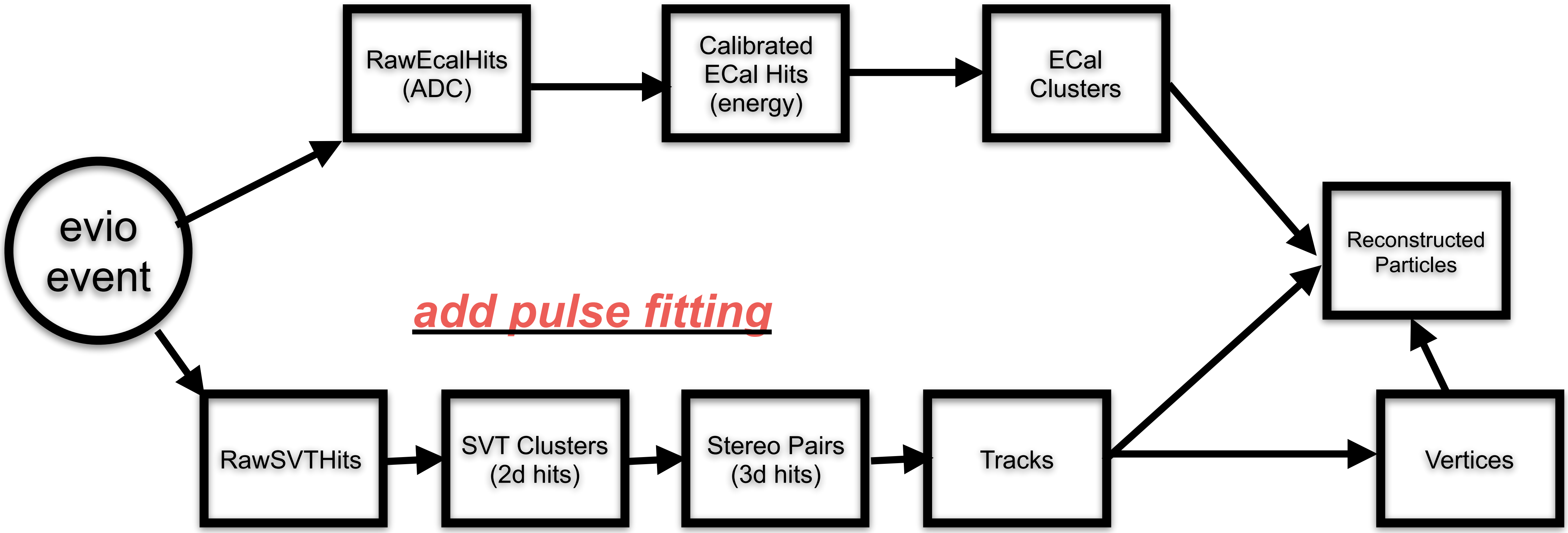
- offline reconstruction
- throughput
- resource requirements
- offline data quality
- data handling and transport
- rapid data analysis

Offline software overview



- Raw data in *evio* format, well known here
- at reconstruction time, converted to *lcio* ("linear collider I/O")
  - common event model for LC community
  - c++ & ***java*** implementations, with python bindings and ROOT dictionaries available
- *hps-java*—the HPS reconstruction package
  - depends heavily on *lcsim,* the SiD simulation & recon code
- *DSTBuilder*—slimmed down ROOT TTree for low(-ish) level analysis

Offline software path: evio to reconstructed lcio



- All of this exists; improvements planned but we are good-to-go for data taking
- There will likely be numerous reconstruction passes over the data as we improve calibrations, alignments, reconstruction code
  - all official recon passes will be based on tagged releases;
  - ***data will be reconstructed promptly after it's collected*** in order to check it's quality; *hopefully within a day*

Track finding and fitting

V. Karimaki, Computer Physics Communications 69 (1992) 133—141

Track finding inherited from linear collider simulation (lcsim “seed tracker”)

- Seed-confirm-extend philosophy: fast
- Tightly coupled to track fitting

Fit track in two independent views (const. magnetic field)

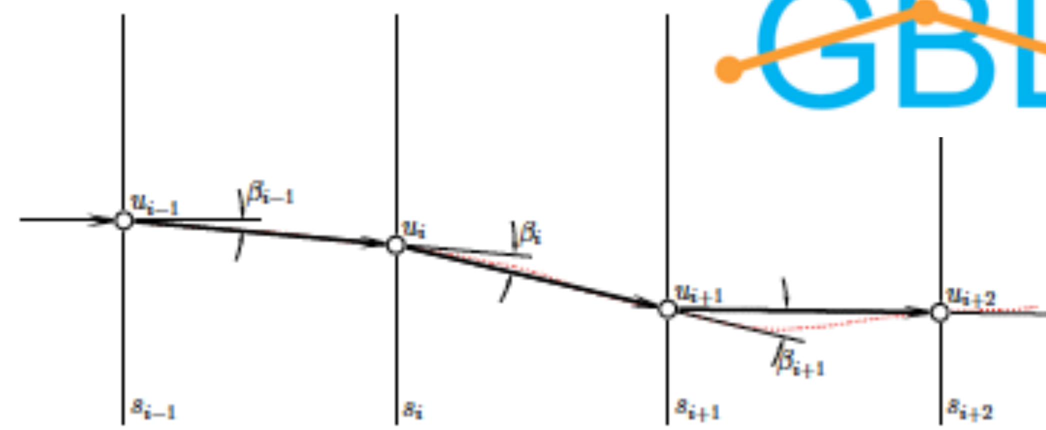
- Circle fit in the “bend plane”
- Straight line fit in non-bend plane
- Both are fast non-iterative fit algorithms
- Simplified handling of multiple scattering

Generalized Broken Lines (GBL)

- Track refit with improved handling of multiple scattering

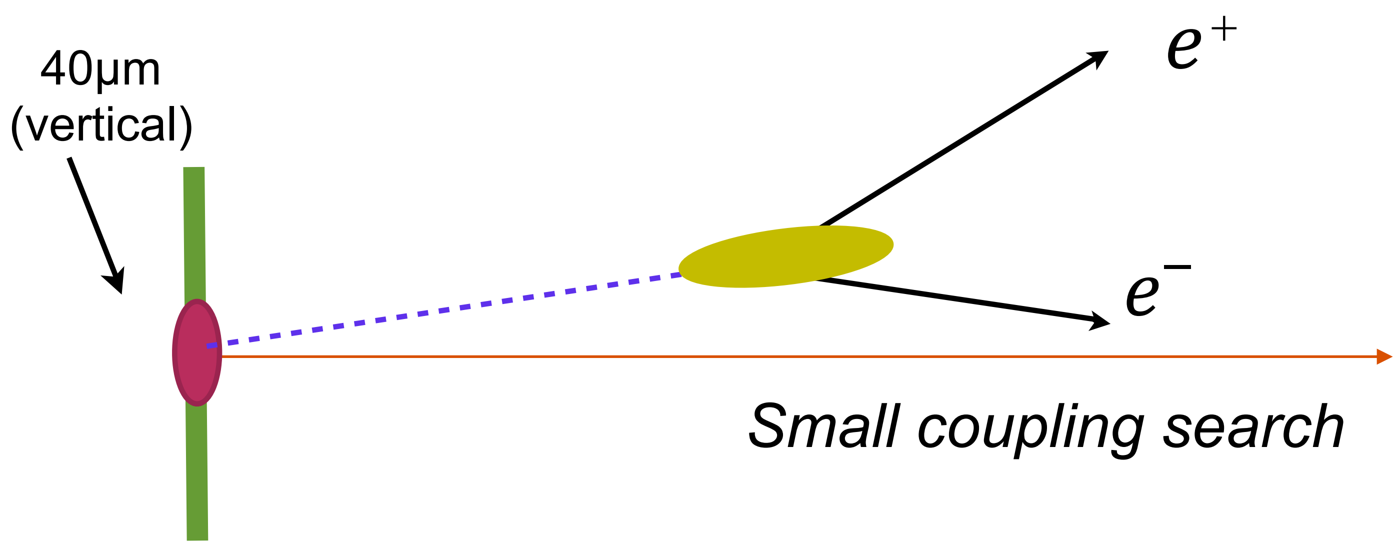
Future major developments

- Straight line track fit (for B=0 runs)
- Include hit time in track fit
- Move GBL to Java framework

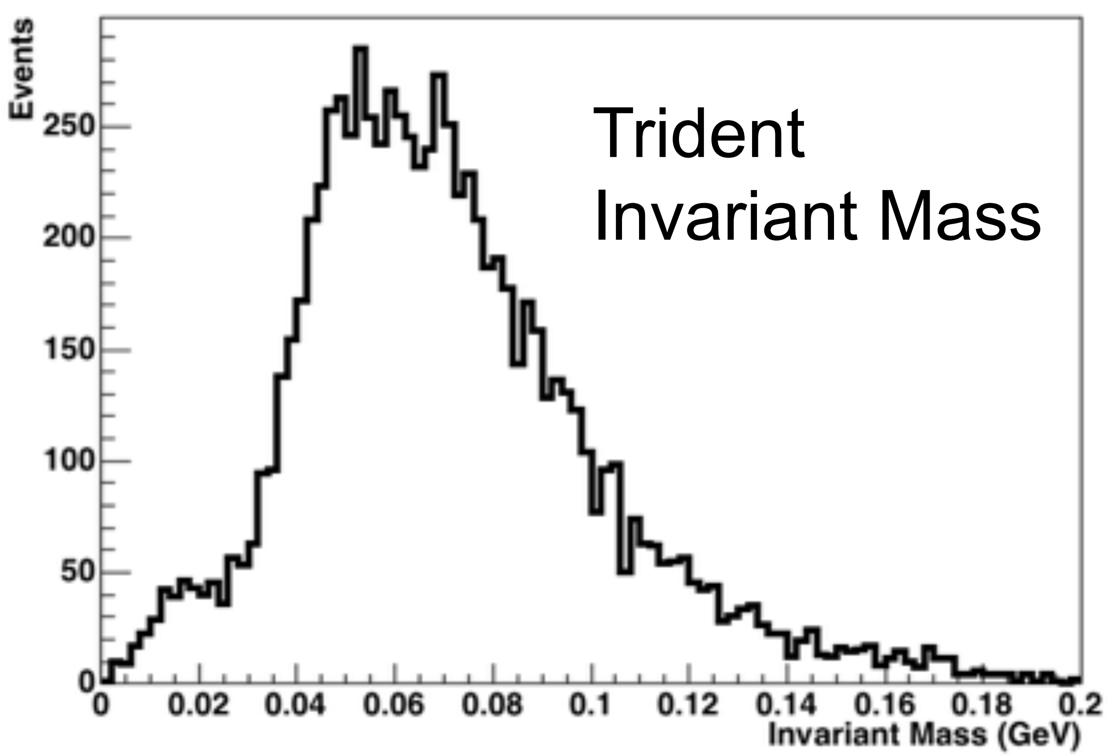


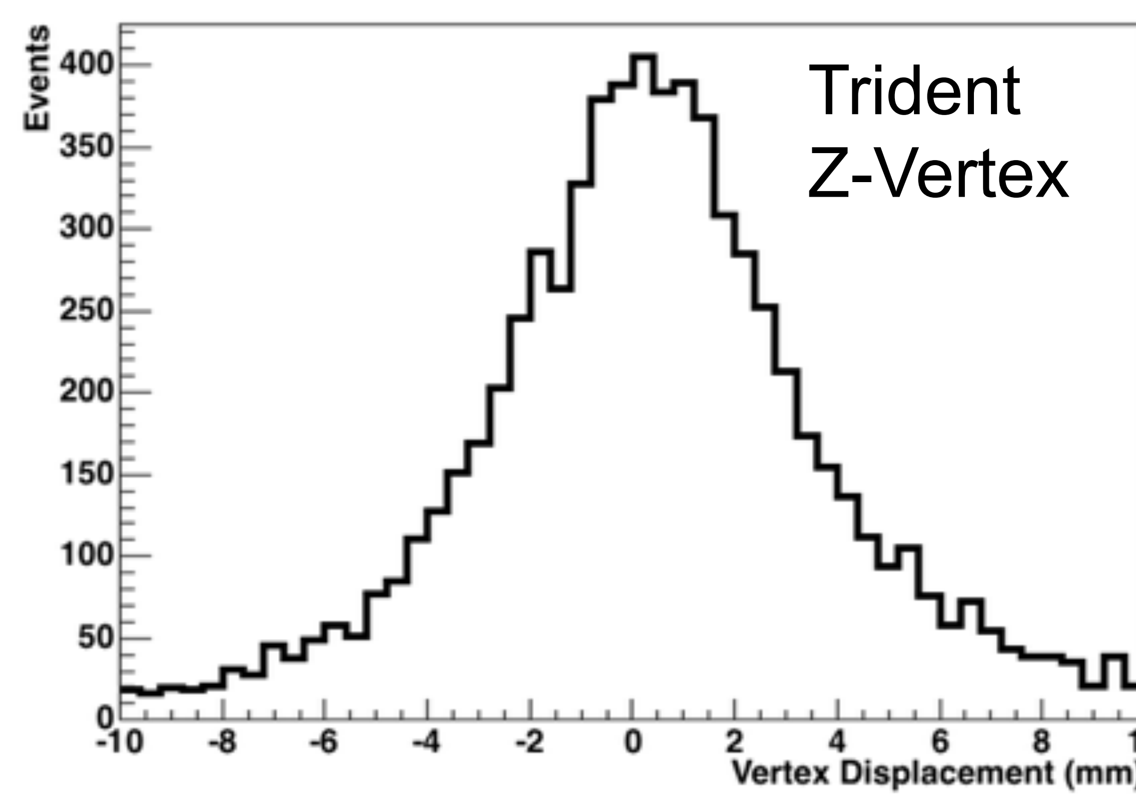
$$S(u) = \sum_{i=1}^{n} \frac{(y_i - u_i)^2}{\sigma_i^2} + \sum_{i=2}^{n-1} \frac{\beta_i^2}{\sigma_{\beta,i}^2}$$

Vertexing



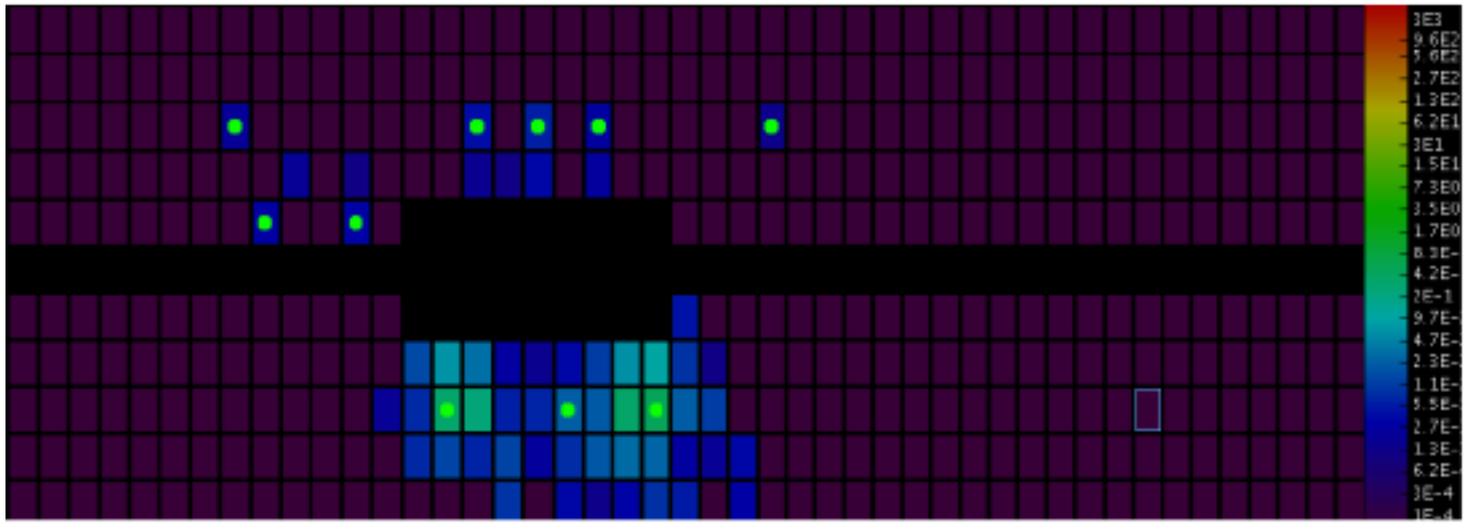
- currently, use fast vertexing described by Billoir & Qian
  - works quite well…good enough for proposal
- future developments
  - use GBL tracks/errors
  - incorporate recoil tracks (3-track vertexing)





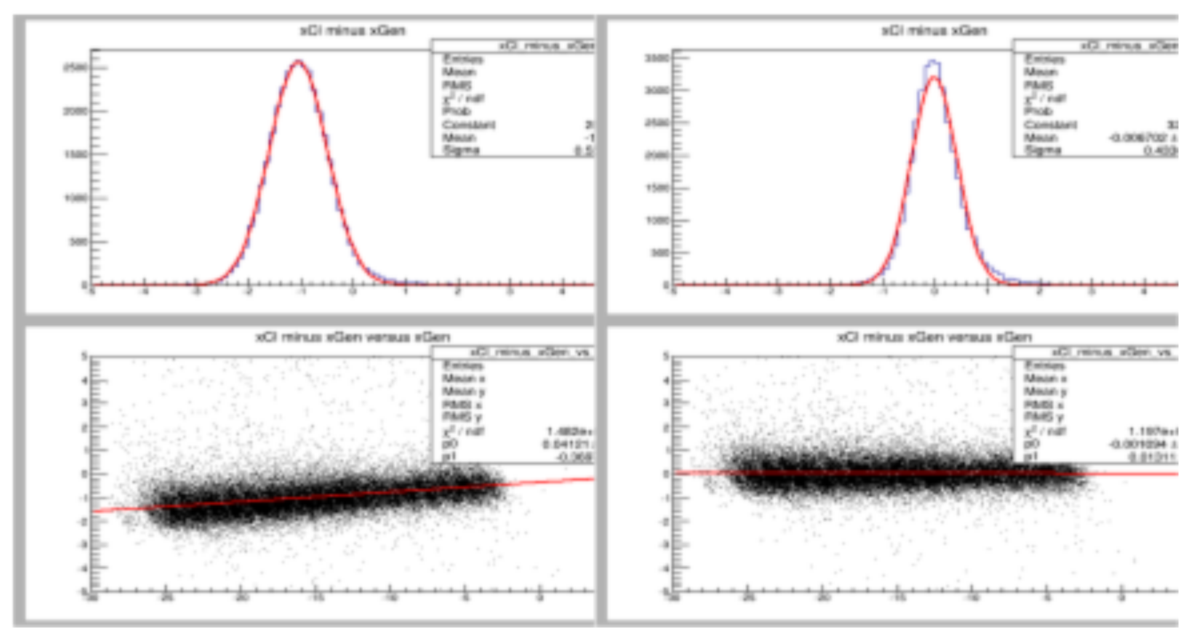
ECal reconstruction

S. Fagan

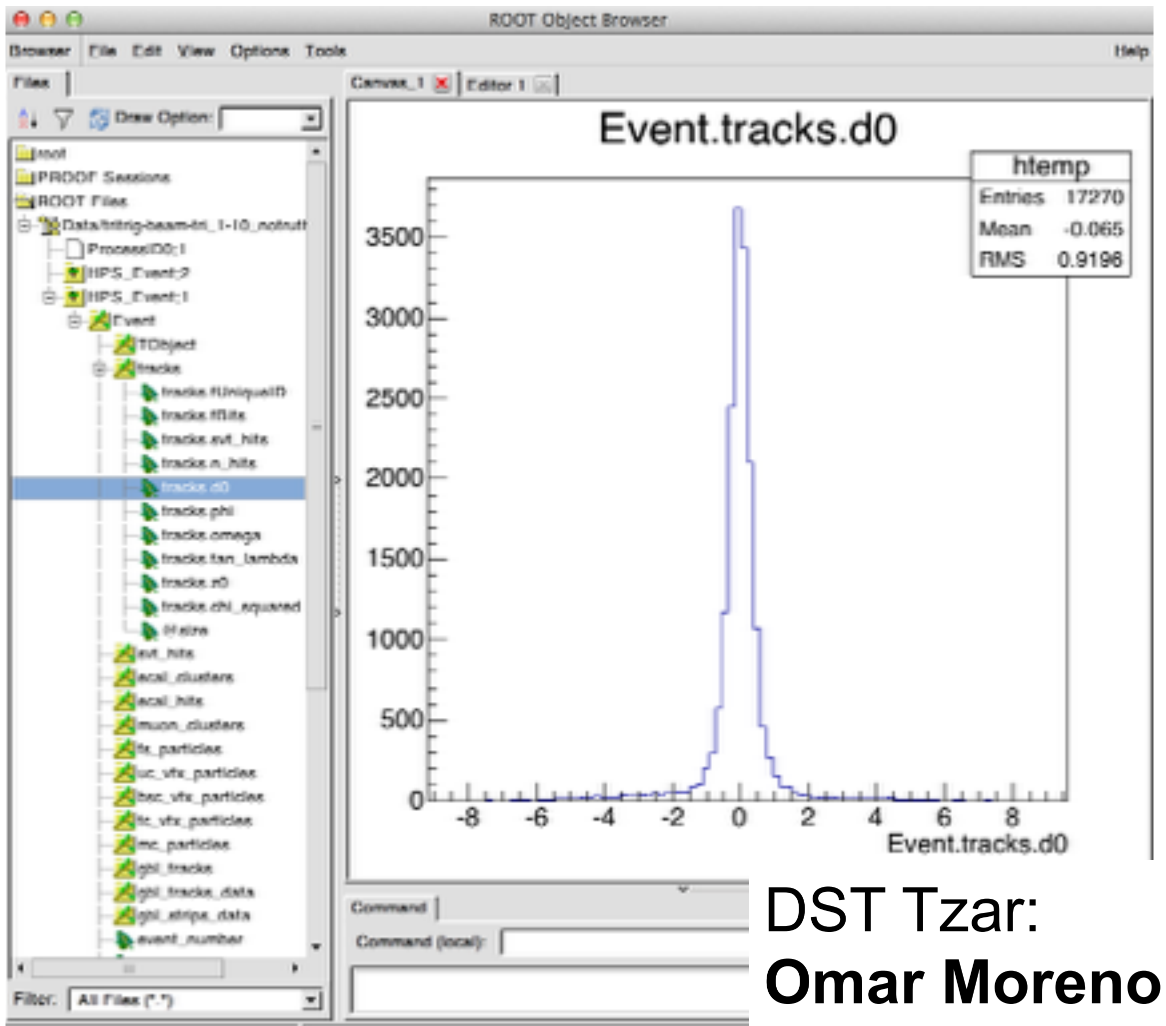


- Implementation of clustering algorithms in hps java: Done
- Testing of cluster algorithms: Done
- Cluster positioning corrections: in progress

- Existing reconstruction modules being brought into line with conditions system
- Looking good to have the reconstruction fully in place on schedule



HPS DSTs: recon lcio to DST

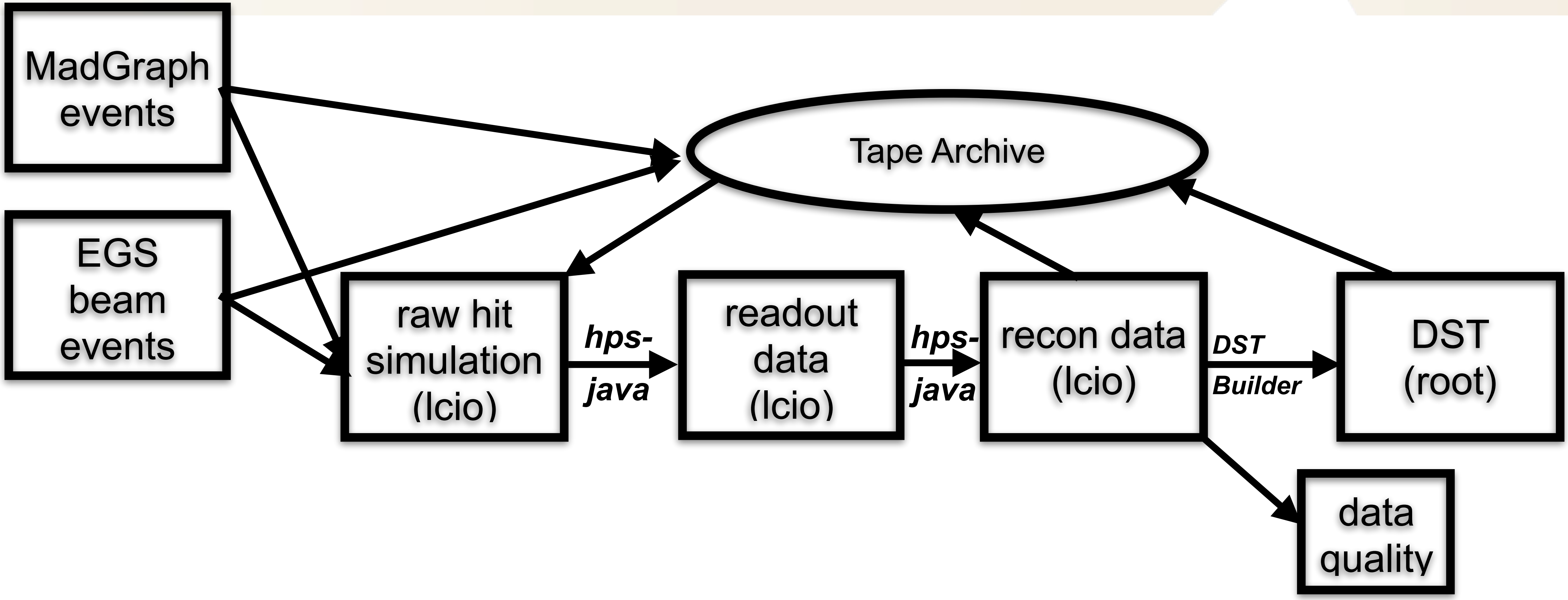


DST Tzar:
Omar Moreno

- DSTs are ROOT TTrees…simple enough to quickly make plots with “Draw” (this figure proves it), but complete enough to do some sophisticated analysis
- Why did we decide to create this layer?
  - ROOT is nice; everyone knows it, lots of tools exist (RooFit, TMVA, etc)
- not a complete rehash of recon lcio file; not all of the information is in DST
  - makes them slim and quick to make

- DSTs are made for each run; each event is written out; roughly same size as raw data (evio)
- Whereas everything (raw, reco lcio, DSTs) will be available @ JLAB, only the DSTs will be copied to SLAC
- the DSTWriter is a good example on how to use the LCIO C++ API with ROOT; this will likely be used extensively in the future (user skims?)

Simulation production…a few more steps



- Two generators: MadGraph (tridents, A's etc) and EGS (beam-target interactions)
- events run through SLIC == GEANT4 front end; gives lcio file with energy deposits & position on active elements
- hps-java based readout and trigger simulation gives "almost-just-like-data" events (except in lcio format)
- path from there is as in data

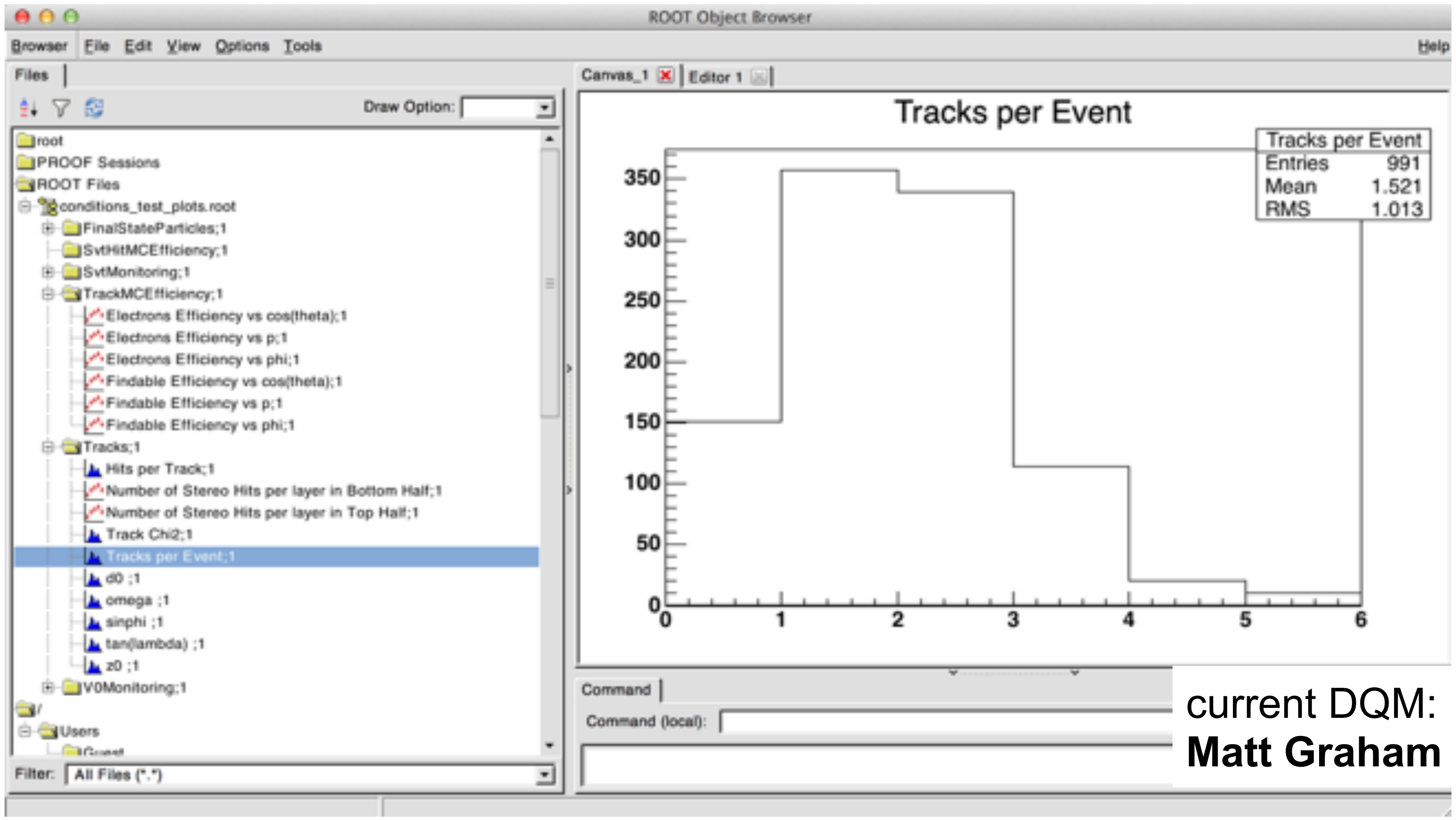
Data & simulation production management

- data production: evio(raw hits)→lcio (clusters,tracks,vertices..etc)
  - automated scripts for submitting jobs to batch & (some) bookkeeping exist; exercised for test run
- simulation production: multi-step process
  - event generation (MadGraph), beam overlay (EGS), detector simulation (slic/GEANT4), readout simulation (hps-java), reconstruction (hps-java…just like data)
  - all of the above steps are well established; automation & monitoring scripts have been written (but still being refined as well)
- dstMaker & data quality are run as a part of this process
- all of the data & sim production will take place at JLAB; use “clashps” account
- all official production will be based on tagged releases
- Data Production Manager: overseer of data and simulation production
  - currently: **Sho Uemura**

Offline data quality checks & procedures

- Maurik talked about online monitoring; we also want to monitor run-by-run offline as well..for a few reasons:
  - document on the quality of the data
  - facilitate observation of long-term trends in the data
  - keep a record of data attributes for posterity and reference
- Here are some requirements for “data quality monitoring”:
  - automatic, for every run (MC sample) ***and*** reconstruction pass
    - run during official reconstruction production; needs to be in hps-java
  - comprehensive
    - all systems & reco: SVT, ECAL, Trigger, Tracking, Vertexing
    - distributions (plots) & quantities (numbers)
  - easy to keep track of & access
    - for plots: root files with appropriate naming conventions; include in data catalog?
    - for numbers: dqm database indexed by run and reconstruction version
    - ***official*** dqm plots and numbers must be based on tagged reconstruction code only

DQM, continued



current DQM:
Matt Graham

- the hps-java DQM framework exists for both making plots and writing to database
- can be run either at reconstruction time or after (on the reco lcio file)
- currently writing to a local database; will migrate to JLAB in the next ~ month
- need to write some scripts to standardize ways of looking at plots & quantities
- as mentioned, this will be run for ***every data run***; we will have ***dedicated sub-system expert*** who will look at the information for every run and report at the ***daily run meetings***

Offline computing requirements: Data storage

# events/week	5.2 E 9
raw event size	3.5 kB
raw event storage	***16 TB***
recon event size	15 kB
recon event storage	***69 TB***
DST event size	2.6 kB
DST event storage	***12 TB***
Total storage/week	***97 TB***

Standard assumptions: 1 week, 200 nA @ 2.2 GeV;
trigger rate = 8.6 kHz

Offline computing requirements: Data processing

# events/week	5.2 E 9	
reco time/event	55 ms	expected, with 8 ns cut
total cpu time/week	***3.3k cpu-days***	
recon evt/job	~570k	2.2 GB files
# of batch jobs	9.1k	
cpu time/job	8.7 hours	
total wall time	***~7 days***	***assume 500 batch slots***

Standard assumptions: 1 week, 200 nA @ 2.2 GeV;
trigger rate = 8.6 kHz
DST & data quality are very fast…add ε to the total

Offline computing requirements: Simulation production & processing

- We have various sets of MC that are useful for us (A' signal events, pure beam-on-target events, etc) but the set that takes the bulk of computing are ***generic, fully-triggered trident events*** ***(GFTTE)***
- The rate for this (2.2GeV, 200nA) is ~850 Hz; expect 514M of these in a week

	cpu-time	storage
trident generation (MadGraph)/10k triggered	19 hours	4.5 MB
beam electron generation (egs)/10k triggered tridents	1100 s	51 MB
detector simulation (slic) /10k triggered tridents	17 hours	not archived
Total for 1 week beam time equivalent for 500 slots	***154 days***	***2.7 TB***

Data transport

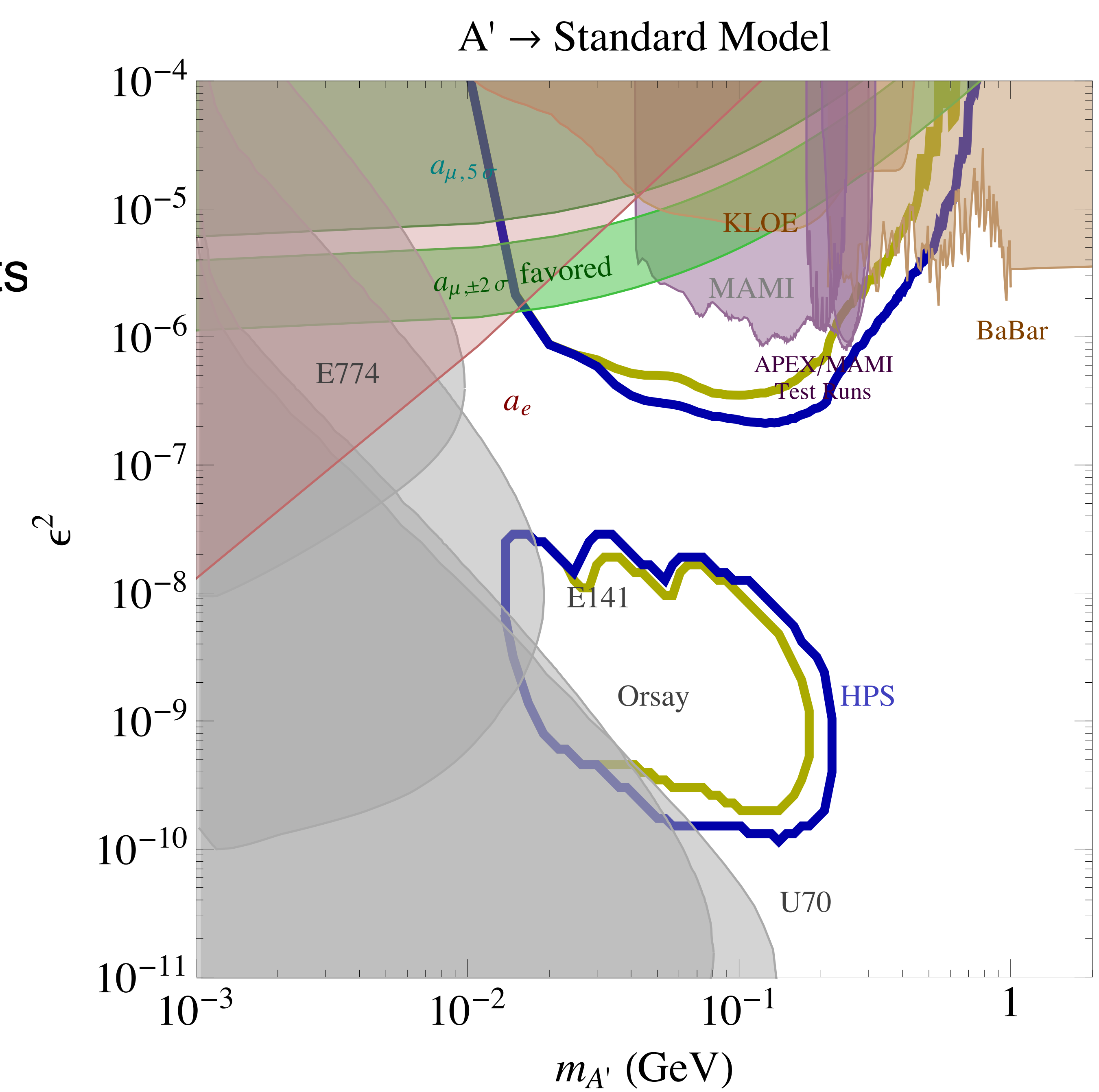
- A few places where data moves around:
  - raw data from Hall B to permanent tape storage
    - mechanisms for this are well established; at full steam, expect ~ 1TB of raw data a day so throughput should not be a major issue
  - data back&forth between tape storage and cache disks for processing
    - ditto
  - DSTs from JLAB to SLAC
    - right now, use globus to transfer data from JLAB to SLAC
    - at JLAB: clashps account; scigw machines
    - at SLAC: hpdatsrv account; use bbr-xfer machines
    - DSTs at SLAC will be saved to tape (available through xrootd) and be available on disk (we have ~30 TB now, no-backup)
    - automated scripts almost exist…will exercise them soon
      - this is not time critical

Physics Analysis: Mock Data Challenge

- Getting the A’ search analysis work going before first data is a priority for us:
  - help identify potential issues we can address “on-the-floor” (e.g. special runs for calibrations, etc)
  - quick turnaround from data taking to publication
- at DOE reviewer’s suggestion, we’re having a mock data challenge
  - beginning-to-end analysis on a data-sized chunk (1 week, 2.2 GeV) of MC, with MC samples available for tuning
    - first large scale production
  - include some realistic conditions (some sample of noisy, dead SVT channels) but assume detector is aligned/calibrated
  - simulation production is ~ done; reconstruction is ongoing (files are available now)
  - expect this will get many new collaborators involved with analysis

Physics Analysis: From Proposal to Publication…

- The reach calculation in proposal was based on a primitive analysis/ calculation…
- rates from MadGraph
- resolutions from simulations with detailed (but likely still sub-optimal) cuts
- signal extracted via simple cut-and-count



Good enough for a proposal, but there is work to be done to make a publishable analysis:

- track/event selection optimization
- cross-checks
- systematics
- cross-checks
- signal extraction/limit setting procedures

Summary & Aspirations

- The offline is in good shape for data taking
  - everything that needs to be there (hit-making/clustering, tracking, vertexing) is there
  - expect for reconstruction to be continually improved through the life of the experiment
- We have plans in place for prompt offline production & data quality determination
  - phase-0 of mock data challenge
- Moving rapidly from data-on-tape to publication-in-PRL is a priority for us
  - phase-1 of the mock data challenge
  - my goal: 1 year from data to first publication
    - this will likely be a bump-hunt analysis of ~ 1 week of 2.2 GeV data