GlueX Data Analysis

Paul Mattione (JSA)

Outline

* Offline data processing:
  * Calibration, monitoring, & reconstruction
* GlueX analysis software
* Coordinating collaboration analysis efforts

Offline Data Processing

Offline data processing team:
Paul Mattione (JSA), Sean Dobbs (NWU),
Alex Austregesilo (JSA), Thomas Britton (JSA)

Previous members:
Justin Stevens (W&M), Kei Moriya

Calibration Automation

- Batch farm calibration train (SWIF)
  - Run plugins & scripts to automatically calibrate data
  - Timing offsets, drift time-to-distance, tagger time-walk, SC, etc.
  - Multiple passes: Dependencies
  - Calibration constants published once human-verified

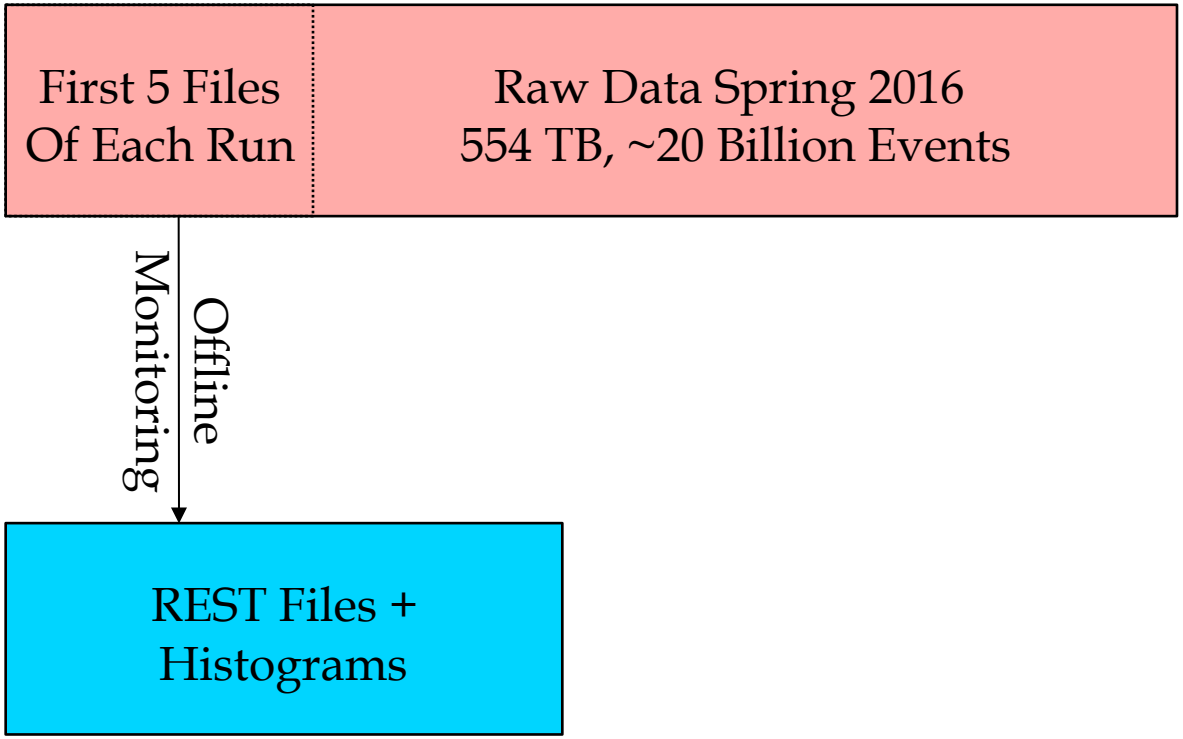
Calibration Automation

- Batch farm calibration train (SWIF)
  - Run plugins & scripts to automatically calibrate data
  - Timing offsets, drift time-to-distance, tagger time-walk, SC, etc.
  - Multiple passes: Dependencies
  - Calibration constants published once human-verified
- Some (complex) procedures not finalized/automated yet
  - E.g. TOF, π^0, Tagger/PS calibrations
  - Skims created to speed up calibrations

Calibration Automation

- Batch farm calibration train (SWIF)
  - Run plugins & scripts to automatically calibrate data
  - Timing offsets, drift time-to-distance, tagger time-walk, SC, etc.
  - Multiple passes: Dependencies
  - Calibration constants published once human-verified
- Some (complex) procedures not finalized/automated yet
  - E.g. TOF, π^0, Tagger/PS calibrations
  - Skims created to speed up calibrations
- Prompt calibrations:
  - Spring 2016: ~Weekly calibration trains, 1st recon. ~3 weeks after start
  - Fall 2016: Calibrate as data hits the tape
  - Future: Run calibrations online

Production Overview (SWIF)



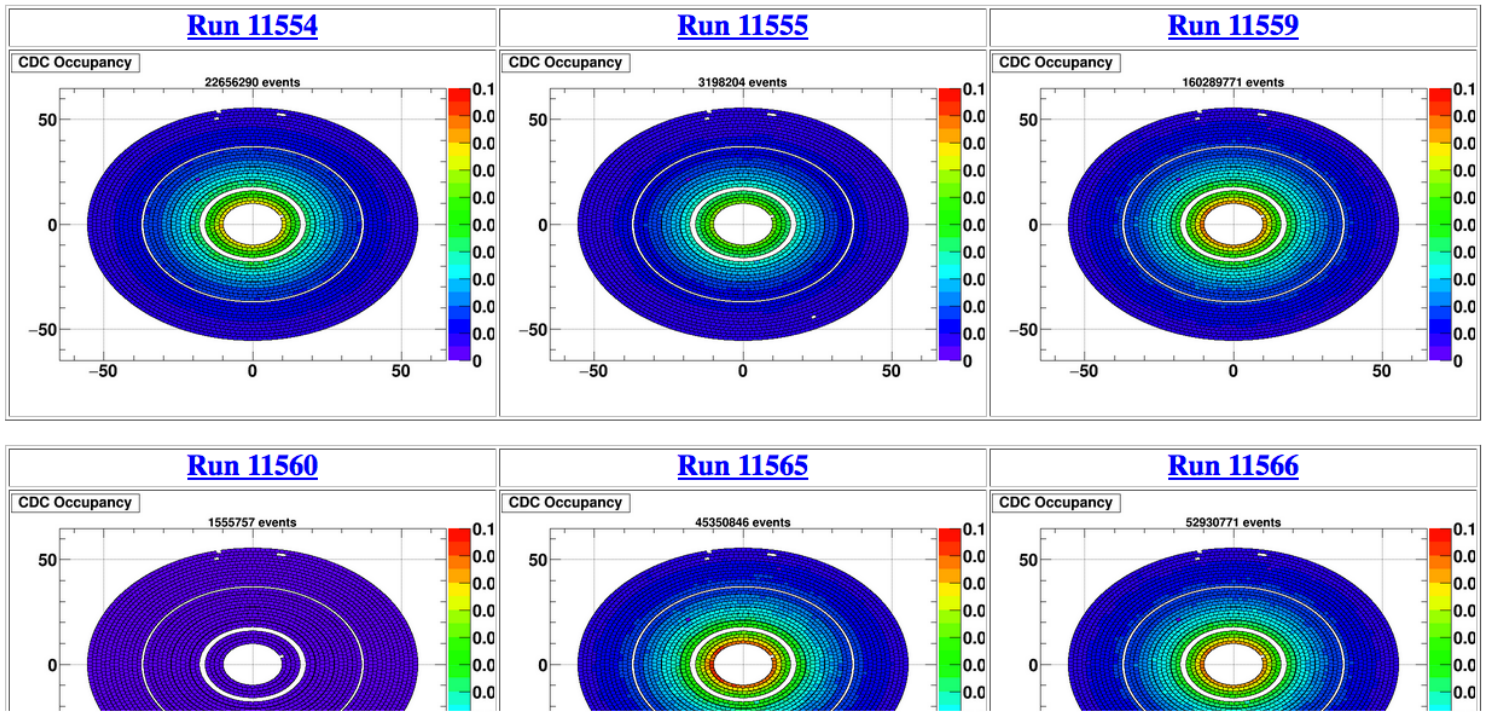
REST: Reconstructed data (tracks, showers, etc.)

Offline Monitoring (SWIF)

* Run 40 JANA plugins: Occupancies, calibrations, reconstruction
  * Incoming data (cron), + every ~2 weeks (as changes come in)

Offline Data Monitoring: Plot Browser

Select Run Period: RunPeriod-2016-02 and Recon. Version: ver00 RootSpy
Select plot to display: CDC Occupancy and run number range to query: 11554 11568 Display
Add additional MYSQL query requirements as string: eg. and beam_current>20 and solenoid_current>1190
Add additional RCDB query requirements as string: @is_production eg. @is_production
Note: Click on figure to open larger image in new tab, or click on Run # to open runBrowser page for that Run.



Offline Monitoring (SWIF)

* Web browse plots, can browse (& download) ROOT files

RunPeriod-2015-03 | ver06 | Display

Link displays plots or ROOT opens file

- **Mar 11** (Run 2607-2616)
- **Mar 12** (Run 2617-2630)
- **Mar 13** (Run 2631-2641)
- **Mar 14** (Run 2642-2642)
- **Mar 15** (Run 2643-2643)
- **Mar 16** (Run 2644-2648)
- **Mar 17** (Run 2649-2649)
- **Mar 18** (Run 2650-2652)
- **Mar 20** (Run 2653-2661)
- **Mar 21** (Run 2662-2669)
- **Mar 22** (Run 2670-2678)
- **Mar 23** (Run 2679-2682)
- **Mar 24** (Run 2683-2684)
- **Mar 25** (Run 2685-2685)
- **Mar 27** (Run 2689-2692)
- **Apr 01** (Run 2698-2701)
- **Apr 02** (Run 2702-2710)
- **Apr 03** (Run 2711-2711)
- **Apr 05** (Run 2712-2712)
- **Apr 06** (Run 2713-2717)
- **Apr 07** (Run 2718-2737)
- **Apr 09 (Run 2739-2760)**
  - 2739 (380066 events) ROOT
  - 2740 (1148 events) ROOT
  - 2741 (274 events) ROOT
  - 2742 (1523347 events) ROOT
  - 2743 (10000 events) ROOT
  - 2746 (10000 events) ROOT
  - 2747 (10000 events) ROOT
  - 2749 (1 events) ROOT
  - 2750 (10000 events) ROOT
  - 2752 (10000 events) ROOT
  - 2753 (10000 events) ROOT
  - 2754 (10000 events) ROOT
  - 2755 (10000 events) ROOT
  - 2756 (10000 events) ROOT

BCAL: DigiSummary | DigiTime | DigiOccupancy | Cluster | Shower | Effic | BCALInvMass | B/FCALInvMass | Trigger

FCAL: DigiPulseInt | DigiOccupancy | DigiTime | HitSummary | HitTime | ClusterEnergyTime | ClusterSpace

TOF: Energy | Time | OccupancyPlane1 | OccupancyPlane2

SC/ST: DigiPulseInt | DigiTime | DigiOccupancy

TAGM: DigiPulseInt | DigiMultiplicity | HitOccupancy | HitTime

TAGH: DigiRawInt | DigiTDCTime | DigiPedVsSlot | HitSummary | HitSummary2

PS: PSC1 | PSC2 | PSC3 | PS1 | PS2 | PS_E | PairCoinc | PairEff | PairTagEnergy

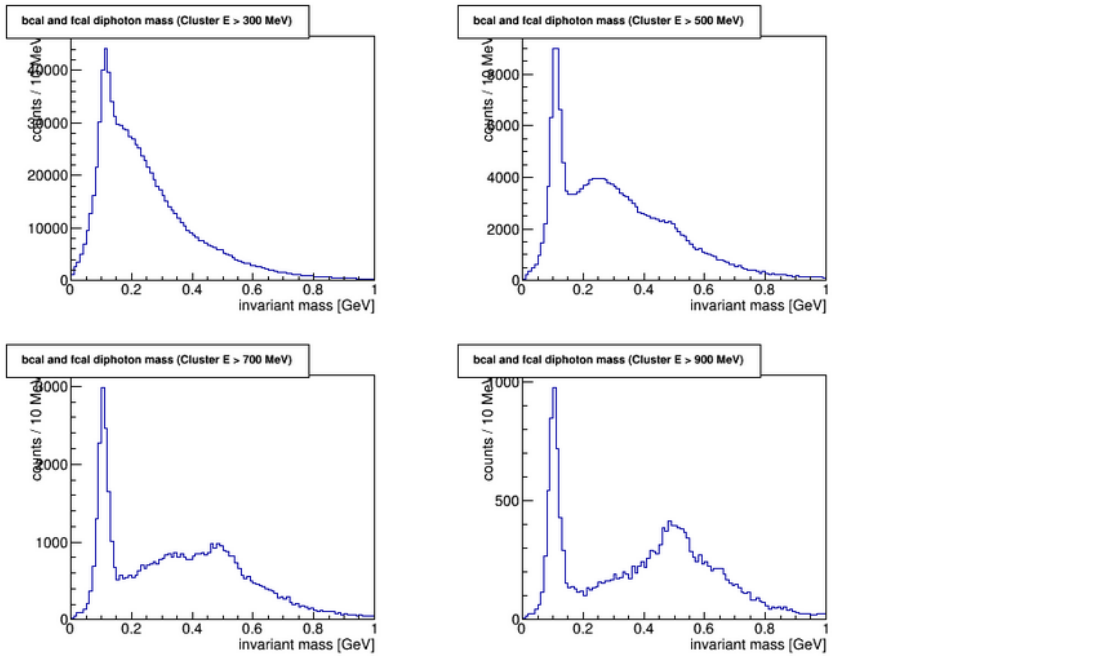
RF: RF1 | RF2 | RF3

HLDetectorTiming: Tagger Timing | Tagger-RF | Tagger-SC | FCAL/BCAL | SC/TOF | Track Matched Timing

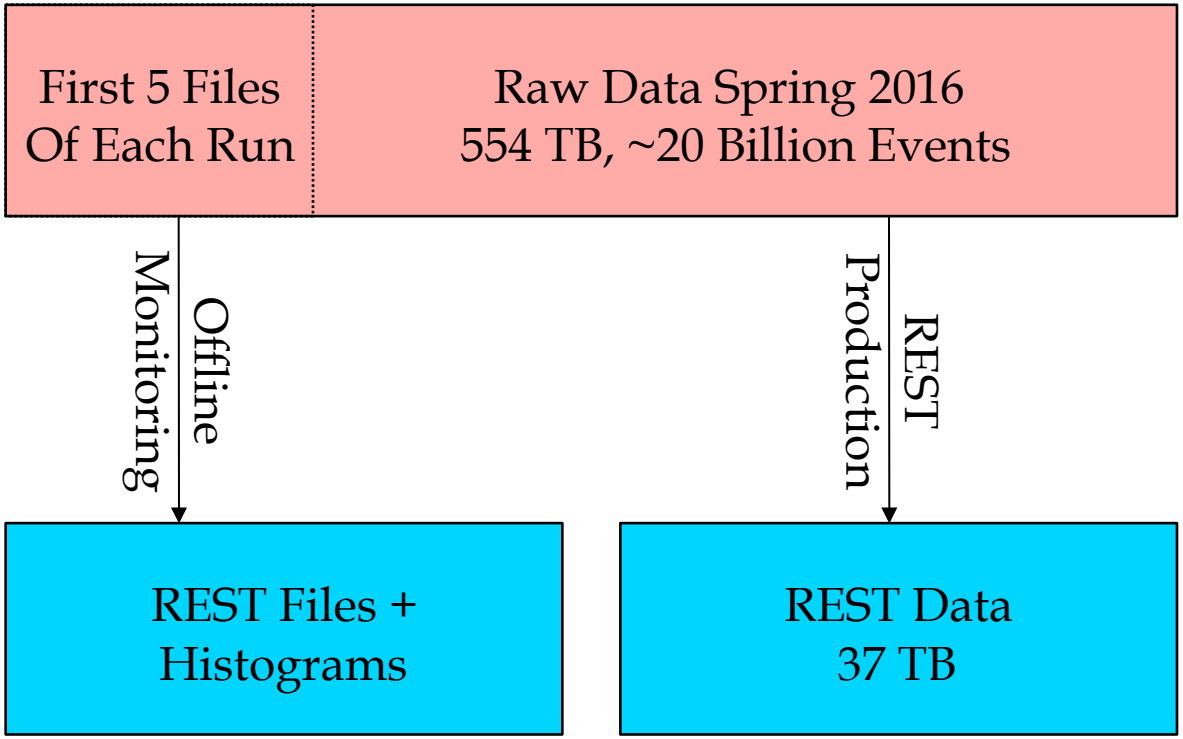
RECO: EventInfo | LLObjects1 | LLObjects2 | HLObjects | TrackMult | Tracking1 | Tracking2 | Tracking3 | MatchBCAL | MatchFCAL | MatchSC/ST | MatchTOF | FCAL1 | FCAL2 | FCAL3 | BCAL1 | BCAL2 | BCAL3 | SC/ST1 | SC/ST2 | SC/ST3 | TOF1 | TOF2 | Kinematics1 | Kinematics2

ANA: $\pi^+\pi^-1$ | $\pi^+\pi^-2$ | $\pi^+\pi^-\pi^0$(2FCAL) | $\pi^+\pi^-\pi^0$(F/BCAL)

Run 2931: Beam current = 70.3852 nA, Radiator = None, Solenoid current = 799.846 A, Trigger = current.conf



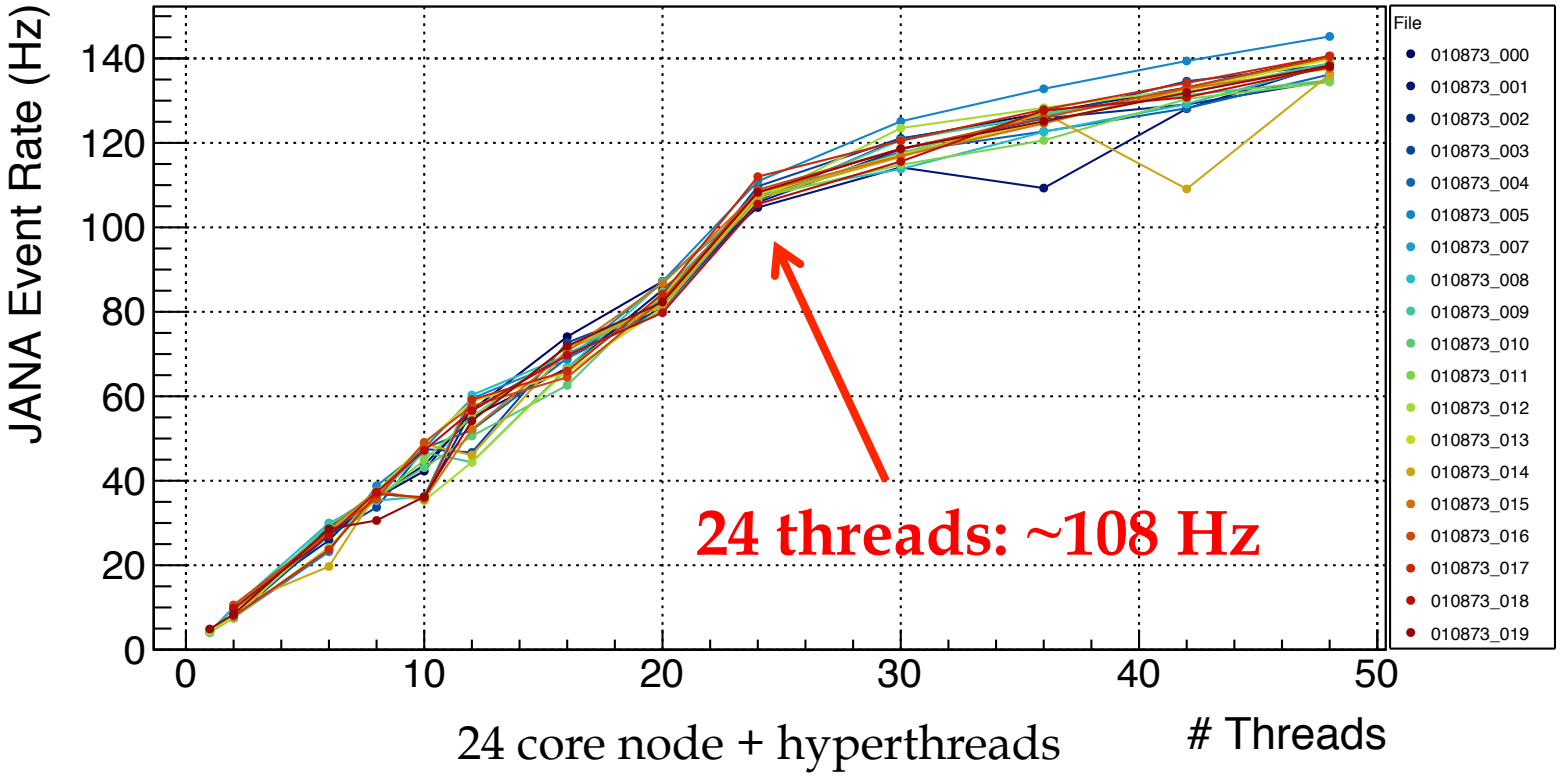
Production Overview (SWIF)



REST: Reconstructed data (tracks, showers, etc.)

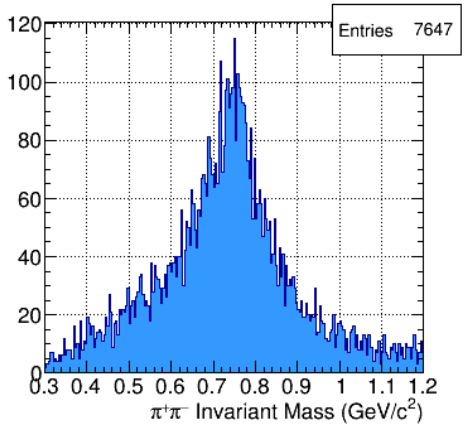
Reconstruction Rate

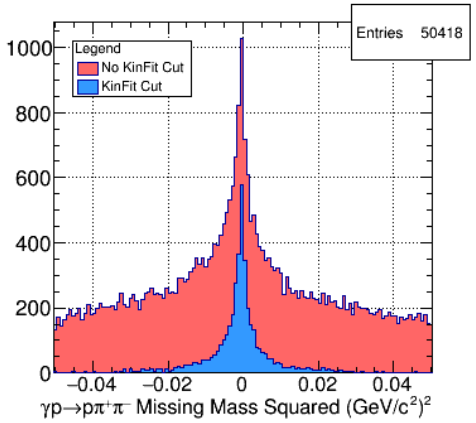
* Issues with multi-threaded scaling: 24 threads: 5x scaling 24 Hz
* Fixed how locking was handled: 24 threads: 108 Hz, 23x scaling



REST Production (SWIF)

* Full reconstruction (tracks, showers, etc.)

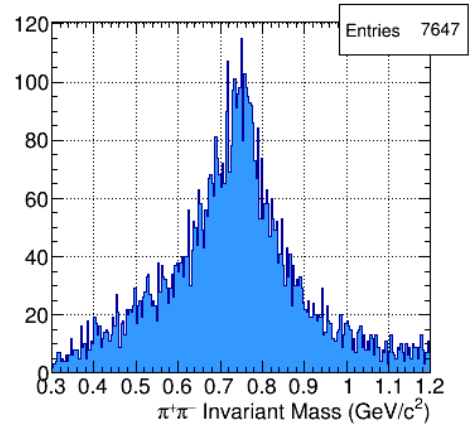


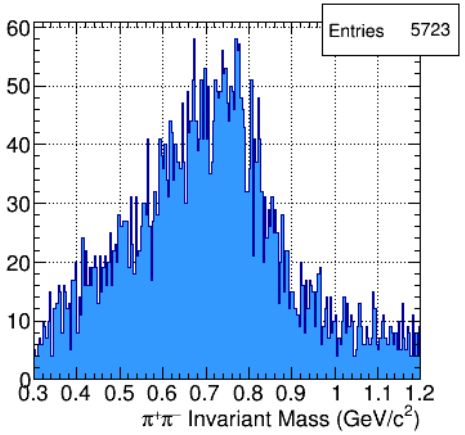


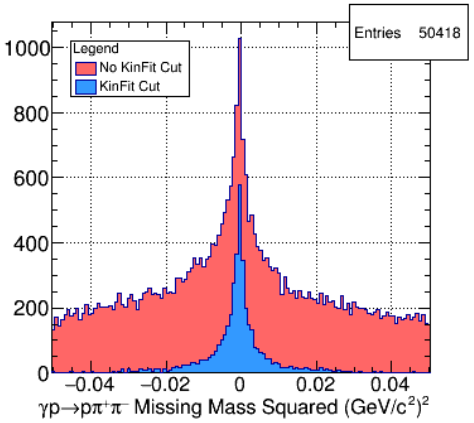
Reconstruction ver01

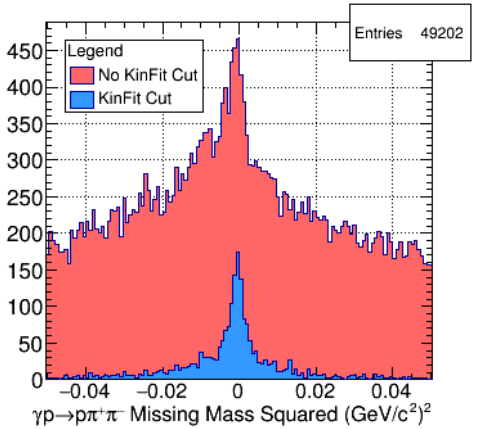
REST Production (SWIF)

* Full reconstruction (tracks, showers, etc.)









Reconstruction ver01

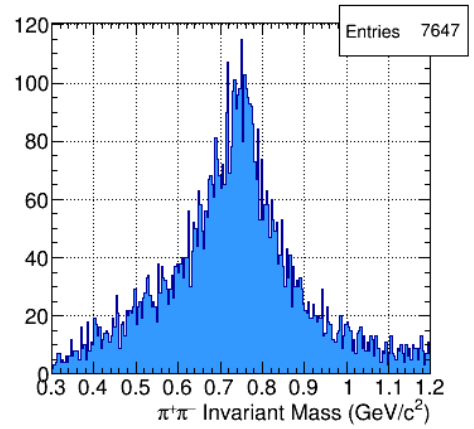
Reconstruction ver02

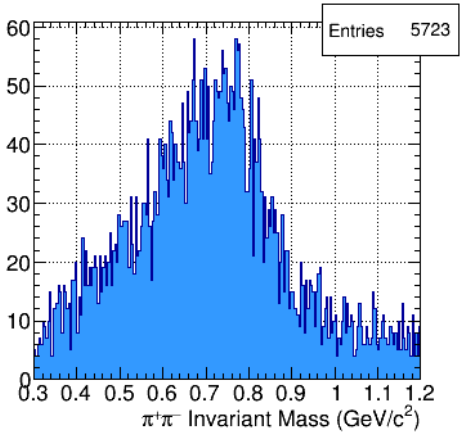
Ver02 problem:
- Track timing overhauled
- Lingering issues

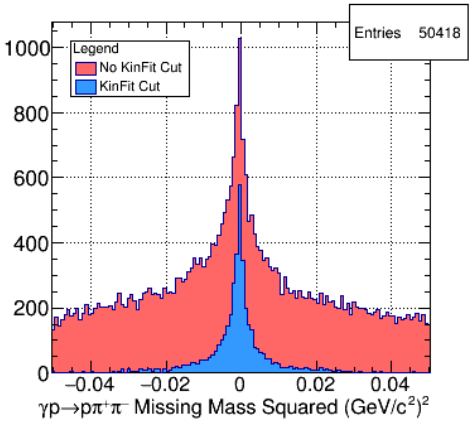
Didn't notice before launch
- Not in existing monitoring
- Noticed after ~ 1 week

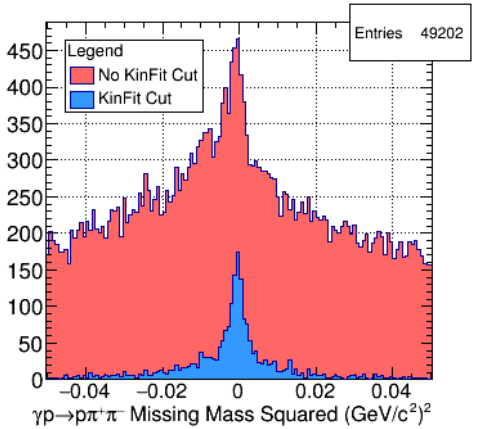
REST Production (SWIF)

* Full reconstruction (tracks, showers, etc.)









Reconstruction ver01

Reconstruction ver02

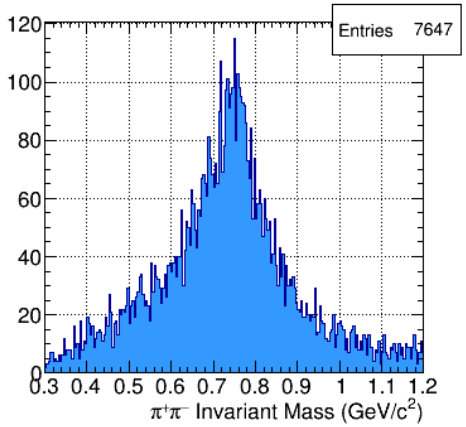
Ver02 problem:
- Track timing overhauled
- Lingering issues

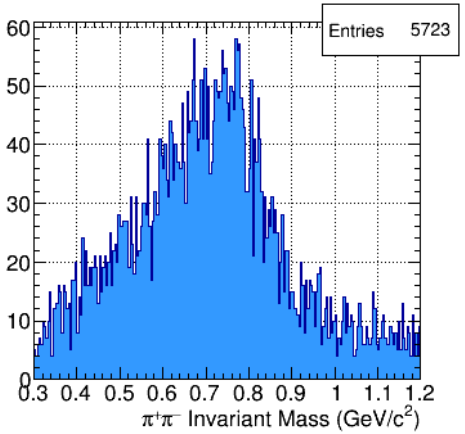
Didn't notice before launch
- Not in existing monitoring
- Noticed after ~ 1 week

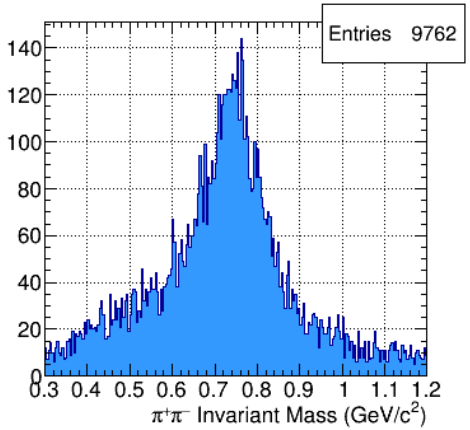
Remedy:
- New ρ, ω monitoring
- New reconstruction tests:
  - Cron every 3 days
  - 1–to–1 comparison

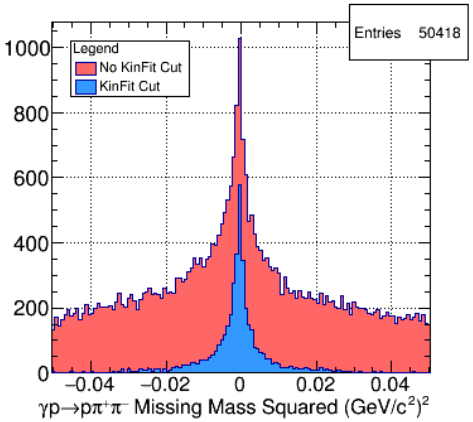
REST Production (SWIF)

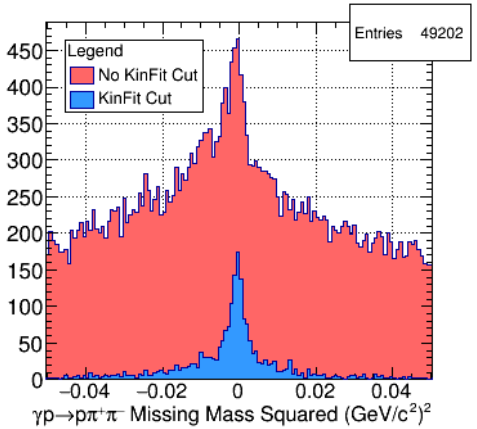
* Full reconstruction (tracks, showers, etc.)

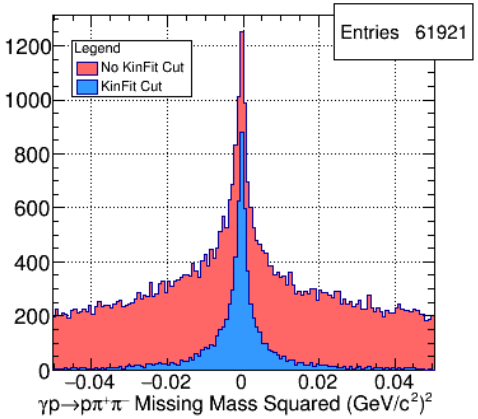












Reconstruction ver01 | Reconstruction ver02 | Current

Reconstruction now better than ever!

GlueX Analysis Software

JANA Analysis Library (C++)

* Library overview (30+ active users):
  * Provide: Best-practices, efficient, validated, user-friendly code
  * GlueX program: > 100 channels to study: Must be easy, scalable
  * Built on JANA: Multi-threaded, factory-based, EVIO or REST

JANA Analysis Library (C++)

* Library overview (30+ active users):
  * Provide: Best-practices, efficient, validated, user-friendly code
  * GlueX program: > 100 channels to study: Must be easy, scalable
  * Built on JANA: Multi-threaded, factory-based, EVIO or REST
* User plugin: 15 – 30 minutes:
  * Run perl script: Generates user plugin with example code
  * In plugin, user specifies their reaction, sets control settings
  * Optionally apply built-in/custom cuts, histogram

JANA Analysis Library (C++)

* Library overview (30+ active users):
  * Provide: Best-practices, efficient, validated, user-friendly code
  * GlueX program: > 100 channels to study: Must be easy, scalable
  * Built on JANA: Multi-threaded, factory-based, EVIO or REST
* User plugin: 15 – 30 minutes:
  * Run perl script: Generates user plugin with example code
  * In plugin, user specifies their reaction, sets control settings
  * Optionally apply built-in/custom cuts, histogram
* Run with plugin: Automatically:
  * Find all combos of reconstructed particles match the reaction
  * Kinematic fit the reaction: Hypothesis test
  * Runs user-selected cuts, histograms
  * Save analysis data to ROOT trees for further analysis

$\gamma p \rightarrow \omega p$: Setup Reaction

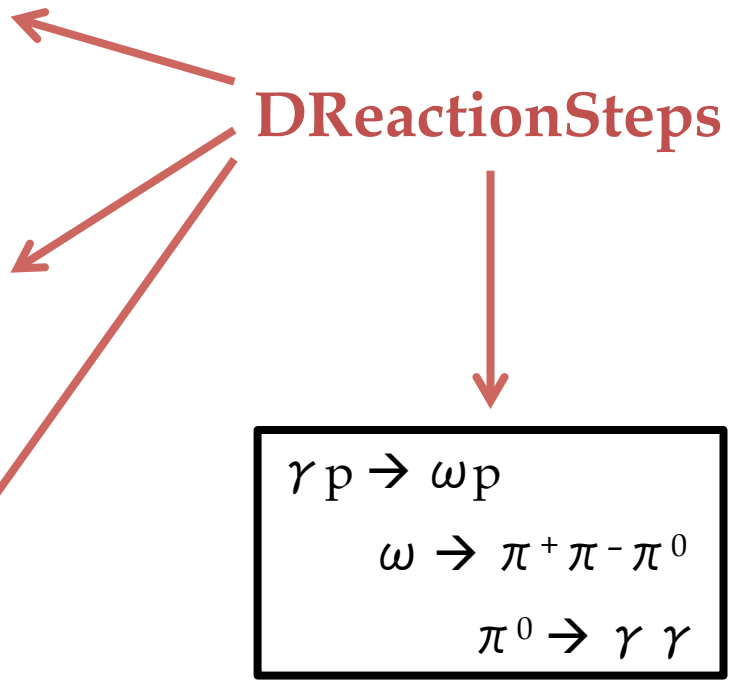
* **DReaction**: Collection of **DReactionSteps**
  * Example code is auto-generated: Uncomment, modify

```
DReaction* locReaction = new DReaction("omega");

//g, p -> omega, p
DReactionStep* locReactionStep = new DReactionStep();
locReactionStep->Set_InitialParticleID(Gamma);
locReactionStep->Set_TargetParticleID(Proton);
locReactionStep->Add_FinalParticleID(omega);
locReactionStep->Add_FinalParticleID(Proton);
locReaction->Add_ReactionStep(locReactionStep);

//omega -> pi+, pi-, pi0
locReactionStep = new DReactionStep();
locReactionStep->Set_InitialParticleID(omega);
locReactionStep->Add_FinalParticleID(PiPlus);
locReactionStep->Add_FinalParticleID(PiMinus);
locReactionStep->Add_FinalParticleID(Pi0);
locReaction->Add_ReactionStep(locReactionStep);

//pi0 -> g, g
locReactionStep = new DReactionStep();
locReactionStep->Set_InitialParticleID(Pi0);
locReactionStep->Add_FinalParticleID(Gamma);
locReactionStep->Add_FinalParticleID(Gamma);
locReaction->Add_ReactionStep(locReactionStep);
```



Particle Combinations

$\gamma p \rightarrow \omega p$

$\omega \rightarrow \pi^+ \pi^- \pi^0$

$\pi^0 \rightarrow \gamma \gamma$

* Want to isolate a channel:
  * GlueX detects: Beam- γ , final-state: $p \pi^+ \pi^- \gamma \gamma$
* Build combinations of detected particles that match our channel

Particle Combinations

$\gamma p \rightarrow \omega p$

$\omega \rightarrow \pi^+ \pi^- \pi^0$

$\pi^0 \rightarrow \gamma \gamma$

* Want to isolate a channel:
  * GlueX detects: Beam- γ , final-state: $p \pi^+ \pi^- \gamma \gamma$
* Build combinations of detected particles that match our channel

For example:
Need: **2 q^+, 1 q^-, 2 q^0**
Measure: **2 q+, 1 q^-, 4 q^0**
Beam: **3** in-time γ's

Particle Combinations

$\gamma p \rightarrow \omega p$

$\omega \rightarrow \pi^+ \pi^- \pi^0$

$\pi^0 \rightarrow \gamma \gamma$

* Want to isolate a channel:
  * GlueX detects: Beam- γ , final-state: $p \pi^+ \pi^- \gamma \gamma$
* Build combinations of detected particles that match our channel

For example:
Need: **2 q^+, 1 q^-, 2 q^0**
Measure: **2 q+, 1 q^-, 4 q^0**
Beam: **3** in-time γ's

Test each q^+ as p (**2x**), π^+ (**1x**)
Test each q^- as π^-: **1x**
Test each neutral as γ : **6x**
Beam: **3x**
Total: **36**

Particle Combinations

$\gamma p \rightarrow \omega p$

$\omega \rightarrow \pi^+ \pi^- \pi^0$

$\pi^0 \rightarrow \gamma \gamma$

* Want to isolate a channel:
  * GlueX detects: Beam- γ, final-state: $p \pi^+ \pi^- \gamma \gamma$
* Build combinations of detected particles that match our channel

<u>For example:</u>
Need: **2 q^+, 1 q^-, 2 q^0**
Measure: **2 q+, 1 q^-, 4 q^0**
Beam: **3** in-time γ's

Test each q^+ as p (**2x**), π^+ (**1x**)
Test each q^- as π^-: **1x**
Test each neutral as γ: **6x**
Beam: **3x**
Total: **36**

* Cuts reduce #-combos: Particle ID, missing mass, kinematic fit, etc.

Histogram, Cut Actions

* Analysis actions: Particle ID, invariant mass histograms, etc.
  * Share common base class
* Are executed in order on particle combos
  * If a combo fails a cut, it will stop executing actions on it

Histogram, Cut Actions

* Analysis actions: Particle ID, invariant mass histograms, etc.
  * Share common base class
* Are executed in order on particle combos
  * If a combo fails a cut, it will stop executing actions on it
* Below: PID section performed before kinematic fit
  * Fit not performed until needed (when results are requested)
  * Can reject background before fitting

```
/****** In plugin DReaction_factory ******/

//PID
locReaction->Add_AnalysisAction(new DHistogramAction_PID(locReaction));
locReaction->Add_AnalysisAction(new DCutAction_AllPIDFOM(locReaction, 0.01)); //1%

//Kinematic Fit Results and Confidence Level Cut
   //0.05 -> 5% confidence level cut on pull histograms only
locReaction->Add_AnalysisAction(new DHistogramAction_KinFitResults(locReaction, 0.05));
locReaction->Add_AnalysisAction(new DCutAction_KinFitFOM(locReaction, 0.01)); //1%
```

Run the Analyses

* Tell JANA to run the analyses:

```
jerror_t DEventProcessor_p3pi_hists::evnt(jana::JEventLoop* locEventLoop, int locEventNumber)
{
   //Get the analysis results (drives the analysis)
   vector<const DAnalysisResults*> locAnalysisResultsVector;
   locEventLoop->Get(locAnalysisResultsVector);

   return NOERROR;
}
```

* Code is pre-generated for you: Just uncomment

OR: Run, Save to ROOT

* Or: Tell JANA to run the analyses, AND save to ROOT TTree:

DReaction:

```
// Highly Recommended: Enable ROOT TTree output for this DReaction
locReaction->Enable_TTreeOutput("tree_p3pi.root");
```

DEventProcessor:

```
jerror_t DEventProcessor_p3pi::evnt(JEventLoop* locEventLoop, uint64_t locEventNumber)
{
   const DEventWriterROOT* locEventWriterROOT = NULL;
   locEventLoop->GetSingle(locEventWriterROOT);
   locEventWriterROOT->Fill_DataTrees(locEventLoop, "p3pi");

   return NOERROR;
}
```

Code is auto-generated: Just uncomment

OR: Run, Save to ROOT

* Or: Tell JANA to run the analyses, AND save to ROOT TTree:

DReaction:

```
// Highly Recommended: Enable ROOT TTree output for this DReaction
locReaction->Enable_TTreeOutput("tree_p3pi.root");
```

DEventProcessor:

```
jerror_t DEventProcessor_p3pi::evnt(JEventLoop* locEventLoop, uint64_t locEventNumber)
{
   const DEventWriterROOT* locEventWriterROOT = NULL;
   locEventLoop->GetSingle(locEventWriterROOT);
   locEventWriterROOT->Fill_DataTrees(locEventLoop, "p3pi");

   return NOERROR;
}
```

* TTree contents (PART format):
  * Event info (Run #, event #, etc.) & metadata (your channel)
  * Particles (beam, charged, neutral, MC thrown)
  * Surviving combos for your channel

Code is auto-generated: Just uncomment

DSelector

* ROOT TSelector class: Helps you work with TTrees
  * Can generate code (TSelector) to read TTree, analyze data
  * Knows nothing about GlueX data format

DSelector

* ROOT TSelector class: Helps you work with TTrees
  * Can generate code (TSelector) to read TTree, analyze data
  * Knows nothing about GlueX data format
* DSelector (GlueX):
  * Inherits from TSelector: Can use in same way
  * Provides C++ interface classes to TTree data (for particles, combos)

DSelector

* ROOT TSelector class: Helps you work with TTrees
  * Can generate code (TSelector) to read TTree, analyze data
  * Knows nothing about GlueX data format
* DSelector (GlueX):
  * Inherits from TSelector: Can use in same way
  * Provides C++ interface classes to TTree data (for particles, combos)
* DSelector has knowledge of your analysis:
  * Generates starting, example code for analyzing your channel
  * Analysis actions: Similar to JANA
    * Cut PID, histogram masses, cut kinematic fit, etc.

DSelector Usage

* Make a custom DSelector with:

```
MakeDSelector tree_file.root tree_name my_selector
```

* Run with:

```
root -l -b tree_file.root
root [1] .x $ROOT_ANALYSIS_HOME/scripts/Load_DSelector.C
root [2] tree_name->Process("DSelector_my_selector.C+");
```

DSelector Usage

* Make a custom DSelector with:

```
MakeDSelector tree_file.root tree_name my_selector
```

* Run with:

```
root -l -b tree_file.root
root [1] .x $ROOT_ANALYSIS_HOME/scripts/Load_DSelector.C
root [2] tree_name->Process("DSelector_my_selector.C+");
```

* PROOF-Lite: Run multi-threaded over TChain on a node
  * No change to DSelector code needed
  * Already setup for users
  * Run with:

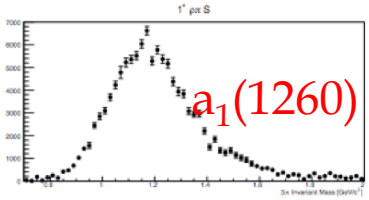
```
root -l -b tree_file.root
root [1] .x $ROOT_ANALYSIS_HOME/scripts/Load_DSelector.C
root [2] DPROOFLiteManager::Process_Tree("tree_file.root",
   "tree_name", "my_selector.C+", "outfile.root", 4); //4 = #threads
```

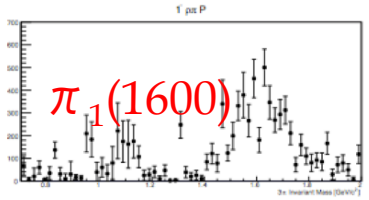
Coordinating Collaboration Efforts

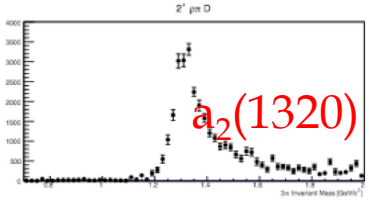
Analysis Coordinators:
Paul Mattione (JSA), Justin Stevens (W&M)

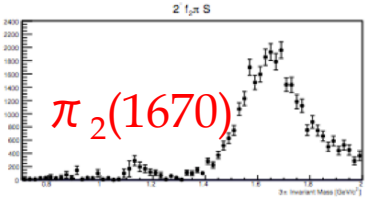
Physics & Analysis Workshops

- 2013: 35 registered participants, ~25 on-site
  - Talks & exercises: Extracting $\pi_1(1600)$ hybrid in $\gamma p \rightarrow \pi^+\pi^+\pi^-(n)$
- 2016: 57 registered participants, ~45 on-site
  - Talks & exercises: Measuring $\gamma p \rightarrow \omega p$ polarization observables



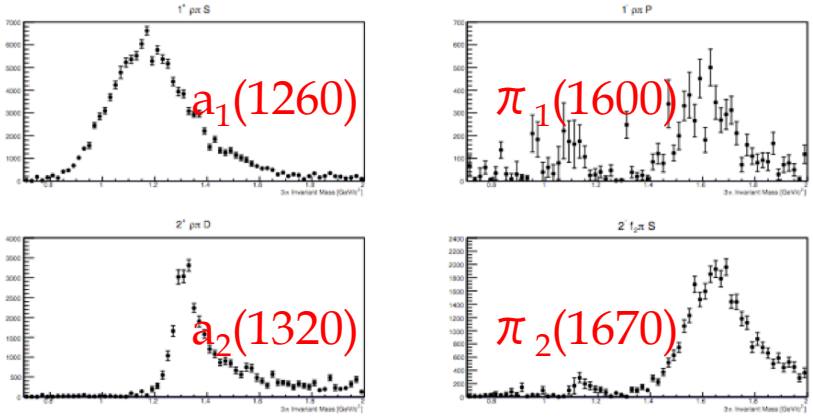




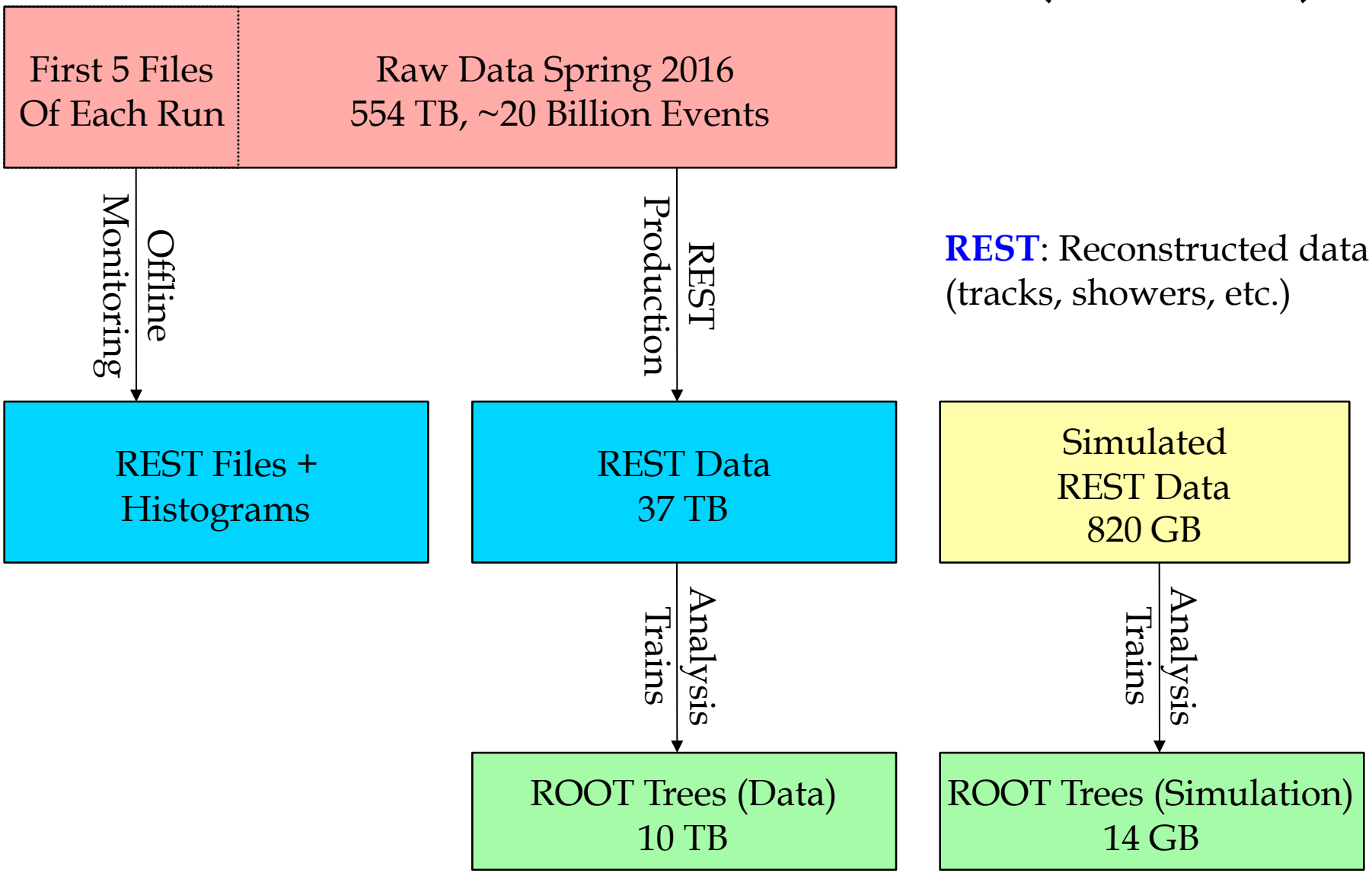


Physics & Analysis Workshops

* 2013: 35 registered participants, ~25 on-site
  * Talks & exercises: Extracting $\pi_1(1600)$ hybrid in $\gamma p \rightarrow \pi^+\pi^+\pi^-(n)$
* 2016: 57 registered participants, ~45 on-site
  * Talks & exercises: Measuring $\gamma p \rightarrow \omega p$ polarization observables
* Some software topics covered:
  * Simulation, analysis library, ROOT analysis, batch farm, etc.
* All sessions recorded (audio + screen): New user startup



Production Overview (SWIF)



Analysis Trains (SWIF)

* Analysis train: Run user JANA analysis plugins on REST data
  * Produce ROOT trees for further analysis
* Run every ~month on reconstructed data
* Large collaboration participation: ~15 Users, ~50 Plugins

Analysis Trains (SWIF)

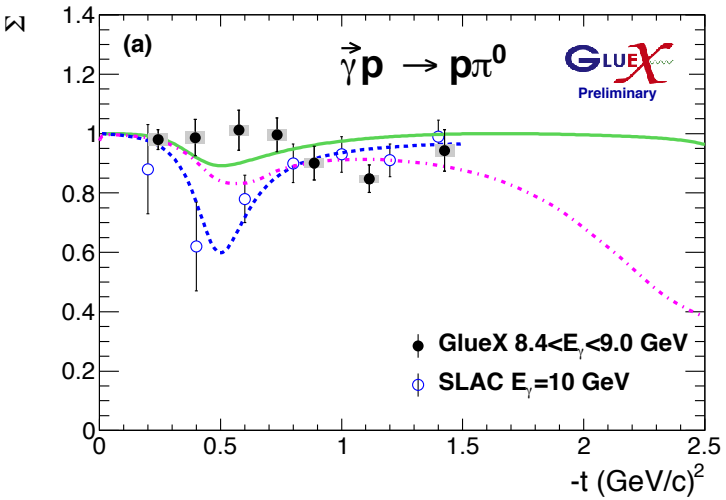
- Analysis train: Run user JANA analysis plugins on REST data
  - Produce ROOT trees for further analysis
- Run every ~month on reconstructed data
- Large collaboration participation: ~15 Users, ~50 Plugins
- Wide variety of channels:
  - Single meson: π^0, π^+, η, ρ, ω, η', ϕ
  - Multi-meson: 2π, 3π, 4π, 2η, $\eta\,\eta'$, $\pi\,\omega$, $\phi\,\eta$, KK, KK$\pi\,\pi$
  - Strangeness studies: K*'s, Λ, Σ's, Σ*'s, Λ*'s, Ξ^-
  - Charm physics: J/ψ, $D^0\Lambda_c$
  - Other: Antiproton, B-boson, multi-γ

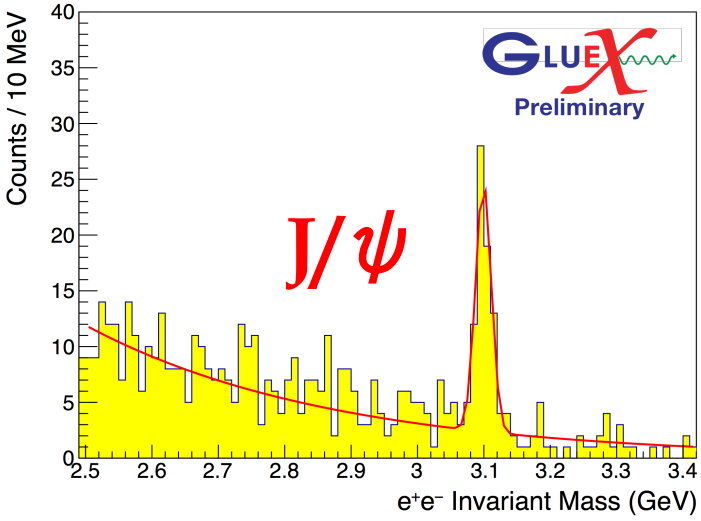
Analysis Trains (SWIF)

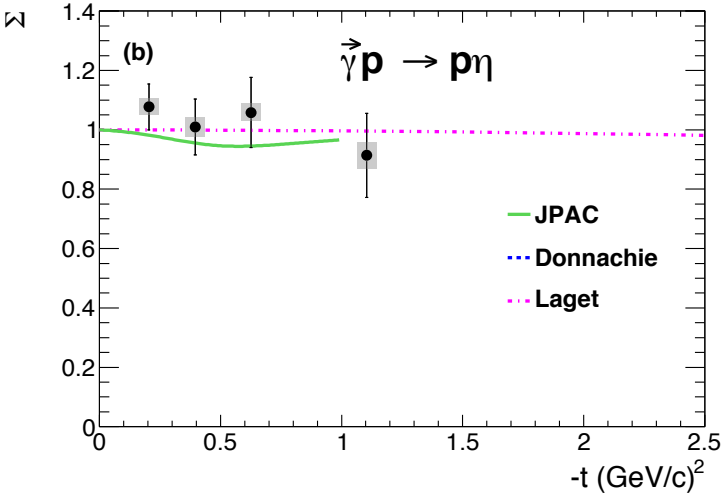
- Analysis train: Run user JANA analysis plugins on REST data
  - Produce ROOT trees for further analysis
- Run every ~month on reconstructed data
- Large collaboration participation: ~15 Users, ~50 Plugins
- Wide variety of channels:
  - Single meson: π^0, π^+, η, ρ, ω, η', ϕ
  - Multi-meson: 2π, 3π, 4π, 2η, $\eta\,\eta'$, $\pi\,\omega$, $\phi\,\eta$, KK, KK$\pi\,\pi$
  - Strangeness studies: K*'s, Λ, Σ's, Σ*'s, Λ*'s, Ξ^-
  - Charm physics: J/ψ, $D^0\Lambda_c$
  - Other: Antiproton, B-boson, multi-γ
- ROOT trees saved to cache/tape: Available for everyone's use

Early GlueX Physics: DNP

- DNP Physics:
  - Asymmetries: π^0, η, ρ, ω, η'
  - Peaks: $a_0(980)$, $b_1(1235)$, J/ψ,…
  - 4 months after end of run
- Analysis software: Success!
  - Many users, channels studied







Blue: Preparing for publication

Documentation

* Extensive documentation:
  * Monitoring: https://halldweb.jlab.org/wiki/index.php/Data_Monitoring_Procedures
  * Analysis: https://halldweb.jlab.org/wiki/index.php/GlueX_Analysis_Software
  * How-To's: https://halldweb.jlab.org/wiki/index.php/Offline_HOWTO_List
  * Etc. etc.
* Tracking collaboration analysis activities:
  * https://halldweb.jlab.org/wiki-private/index.php/GlueX_Physics_Analyses
* Workshops:
  * 2016: https://halldweb.jlab.org/wiki/index.php/GlueX_Physics_Workshop_2016
  * 2013: https://halldweb.jlab.org/wiki/index.php/GlueX_Analysis_Workshop_2013
* YouTube channel (2016 Workshop): "Jefferson Lab Hall-D"
  * https://www.youtube.com/channel/UCjI87hRy7U60CdkGpMSk2Fw

Summary

Summary

- Offline data processing
  - Many calibrations automated (SWIF, still improving)
  - Monitoring, reconstruction, & analysis: SWIF
  - Software tests: Simulation, experiment, nightly build, etc.

Summary

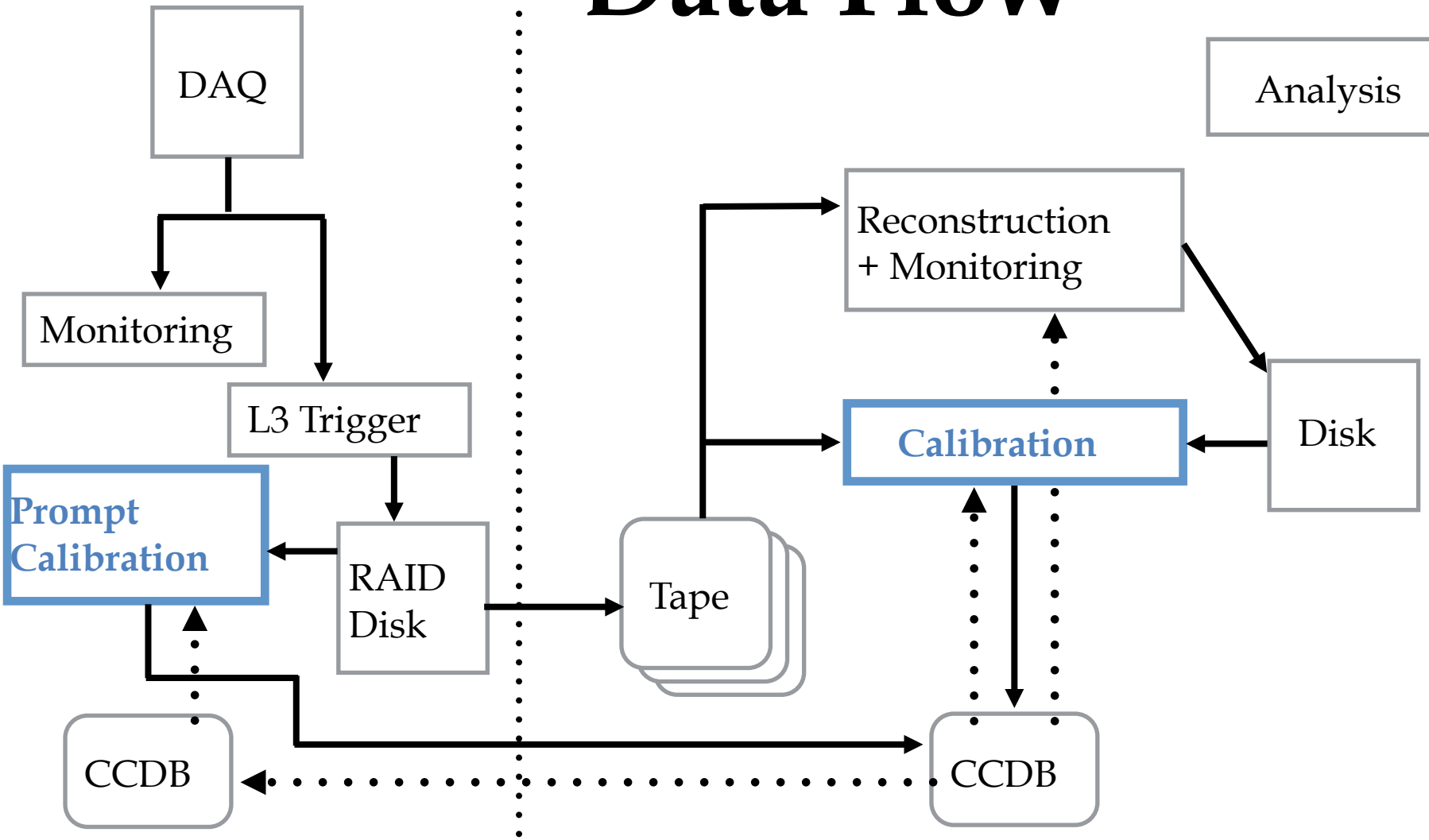
* Offline data processing
  * Many calibrations automated (SWIF, still improving)
  * Monitoring, reconstruction, & analysis: SWIF
  * Software tests: Simulation, experiment, nightly build, etc.
* Analysis software
  * Easy to use, best-practices analysis framework
  * Built-in actions for common tasks: No re-inventing the wheel
  * Mature: Library since 2012, 30+ active users

Summary

* Offline data processing
  * Many calibrations automated (SWIF, still improving)
  * Monitoring, reconstruction, & analysis: SWIF
  * Software tests: Simulation, experiment, nightly build, etc.
* Analysis software
  * Easy to use, best-practices analysis framework
  * Built-in actions for common tasks: No re-inventing the wheel
  * Mature: Library since 2012, 30+ active users
* Collaboration
  * 2013, 2016 Workshops: Software, physics, & analysis
  * Many early results shown at DNP
  * First publication under collaboration review

Reference Slides

Data Flow

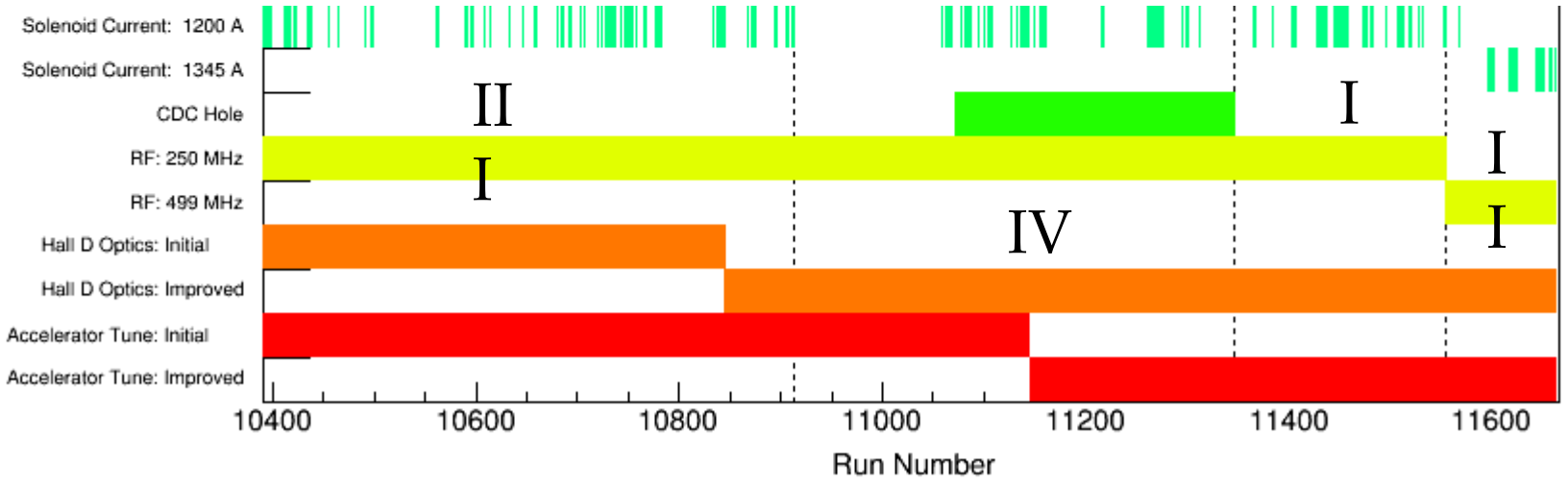


Jefferson Lab

GlueX

REST Production (ver01)

- Planned for May 31th, started June 8th
- Included number of plugins for performance studies (detector efficiency, track reconstruction)
- Split the Data into 4 priority periods

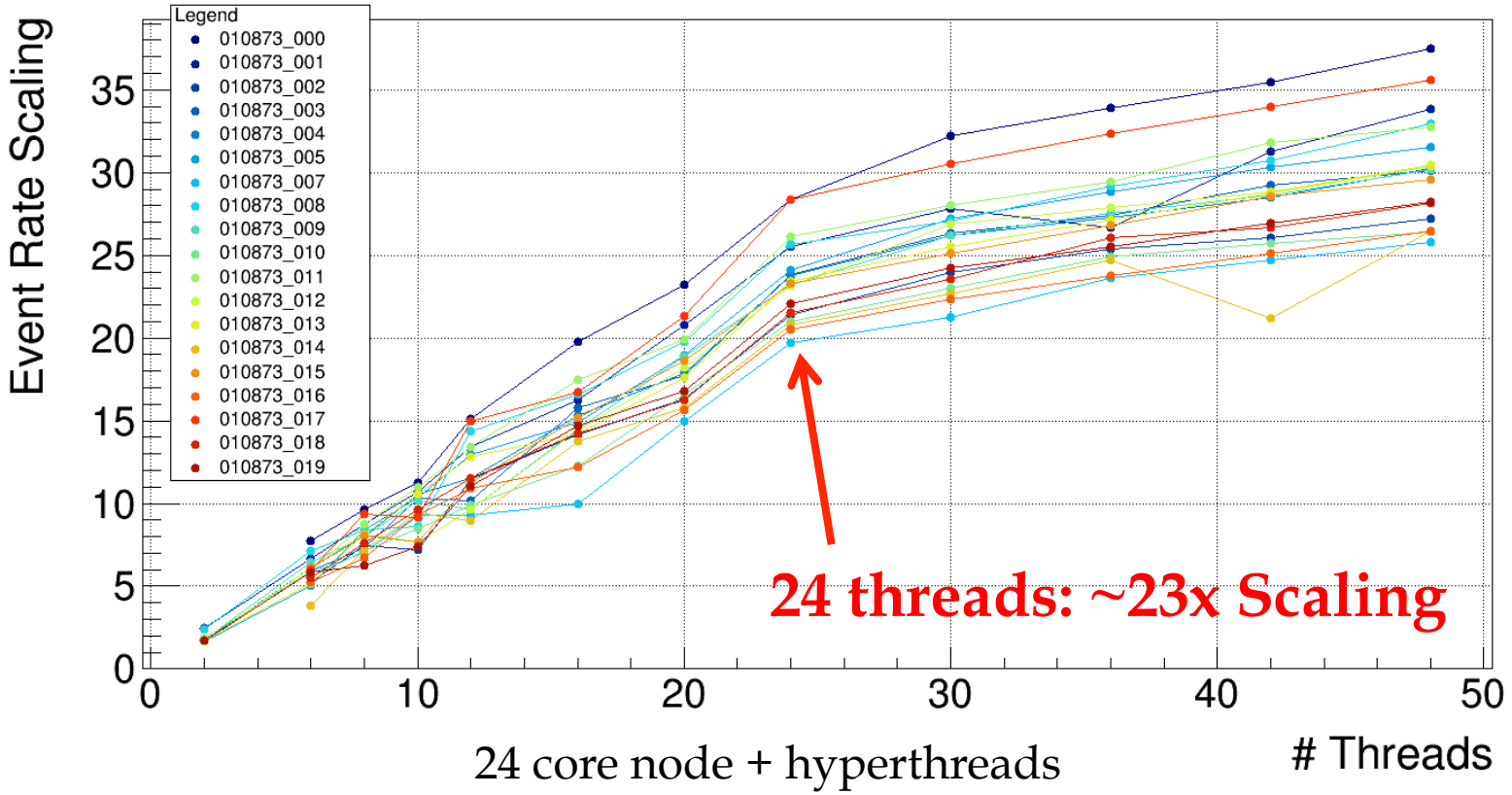


- Intermittent with periods waiting for detector calibration
- Successfully completed July 10th, 21d net processing time
- Failure rate after resubmissions: ~0.1%

Slide courtesy
Alex Austregesilo

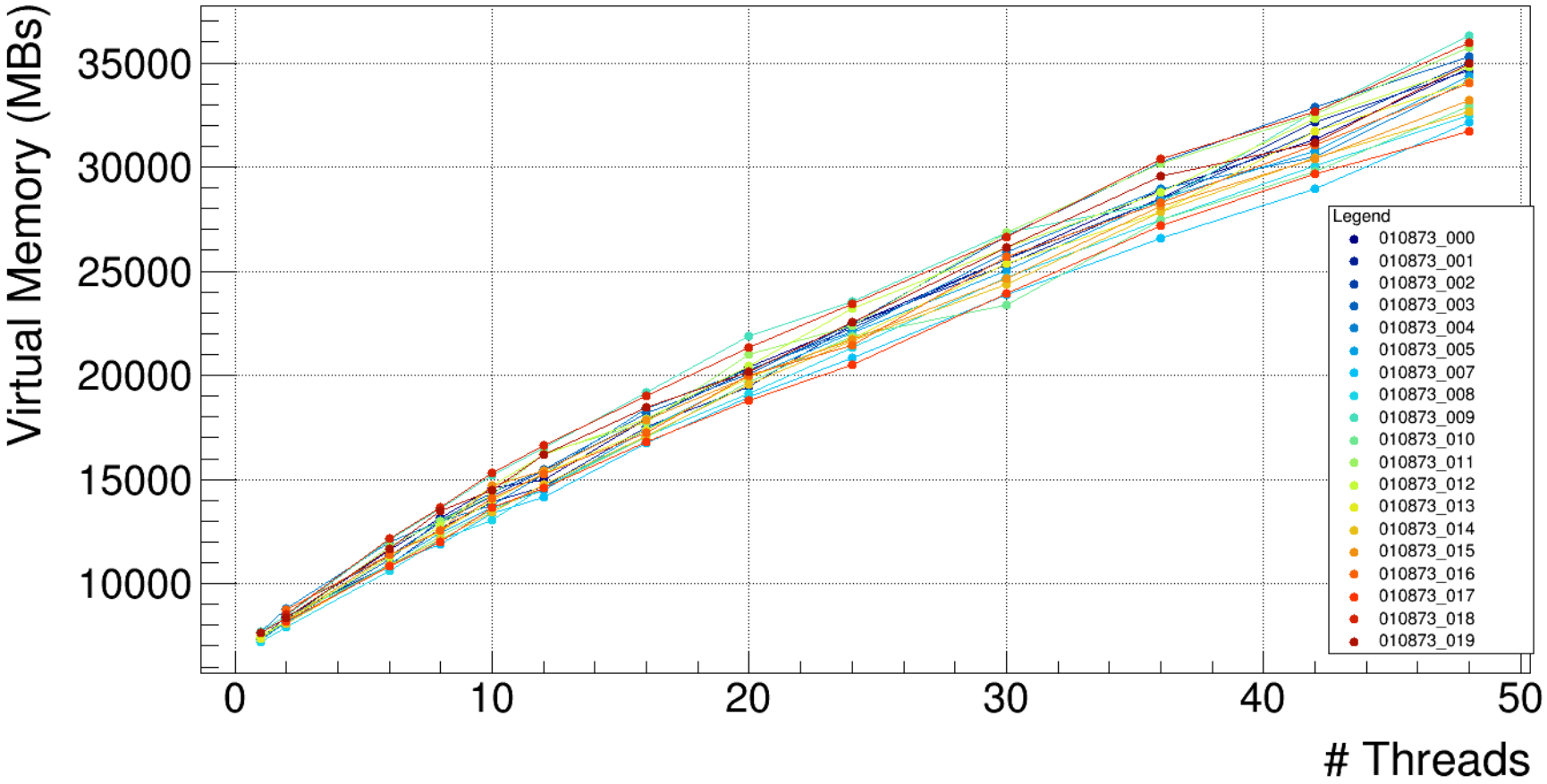
Multi-threaded Scaling: April

* Compartmentalized histogram locks
* At 24 threads, **~23x** scaling: 450% improvement, within 5% of max



Virtual Memory

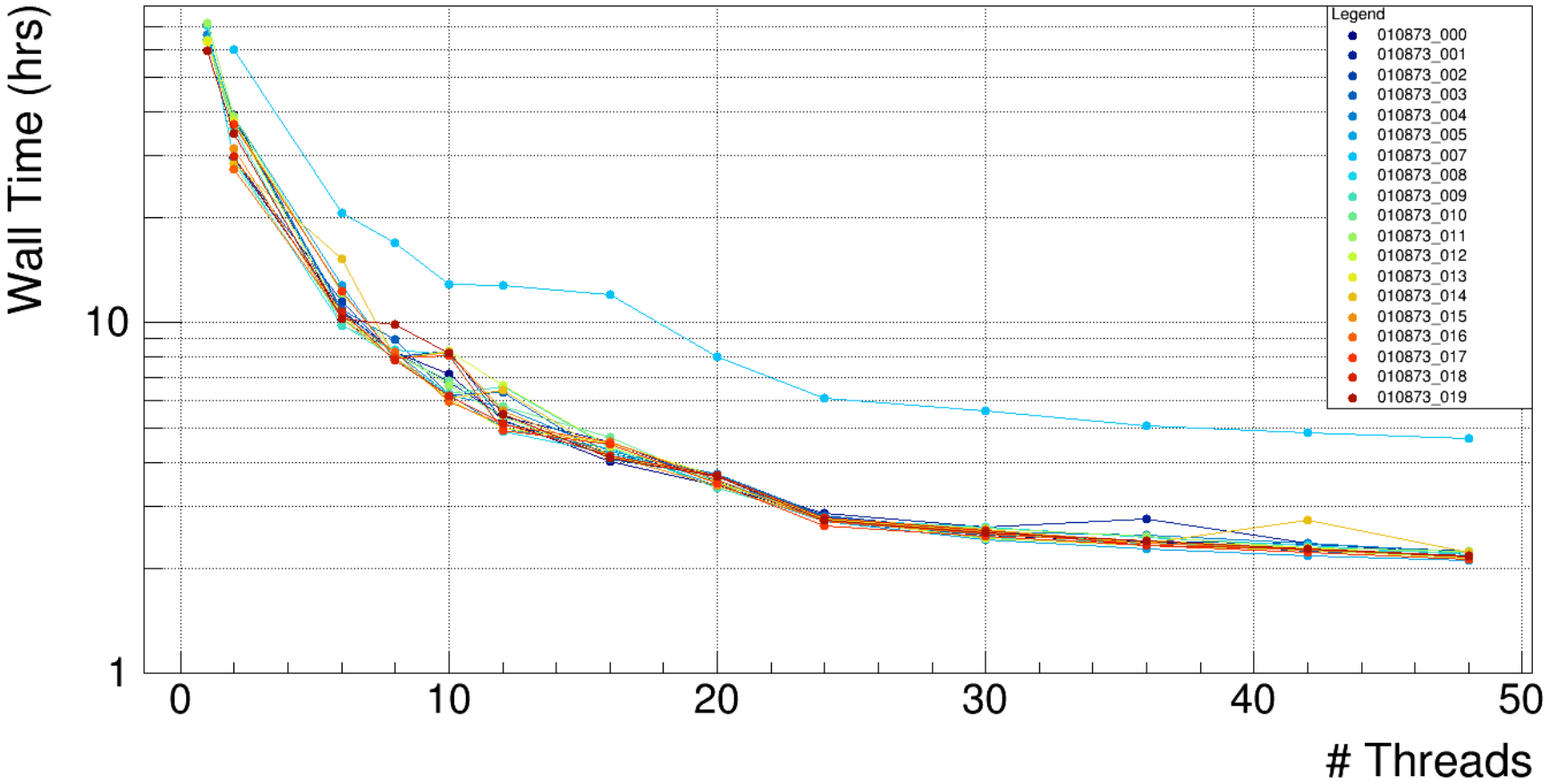
* Virtual memory: Max allowed is node-RAM / 0.7



Scaling: ~612 MB / thread

Wall Time

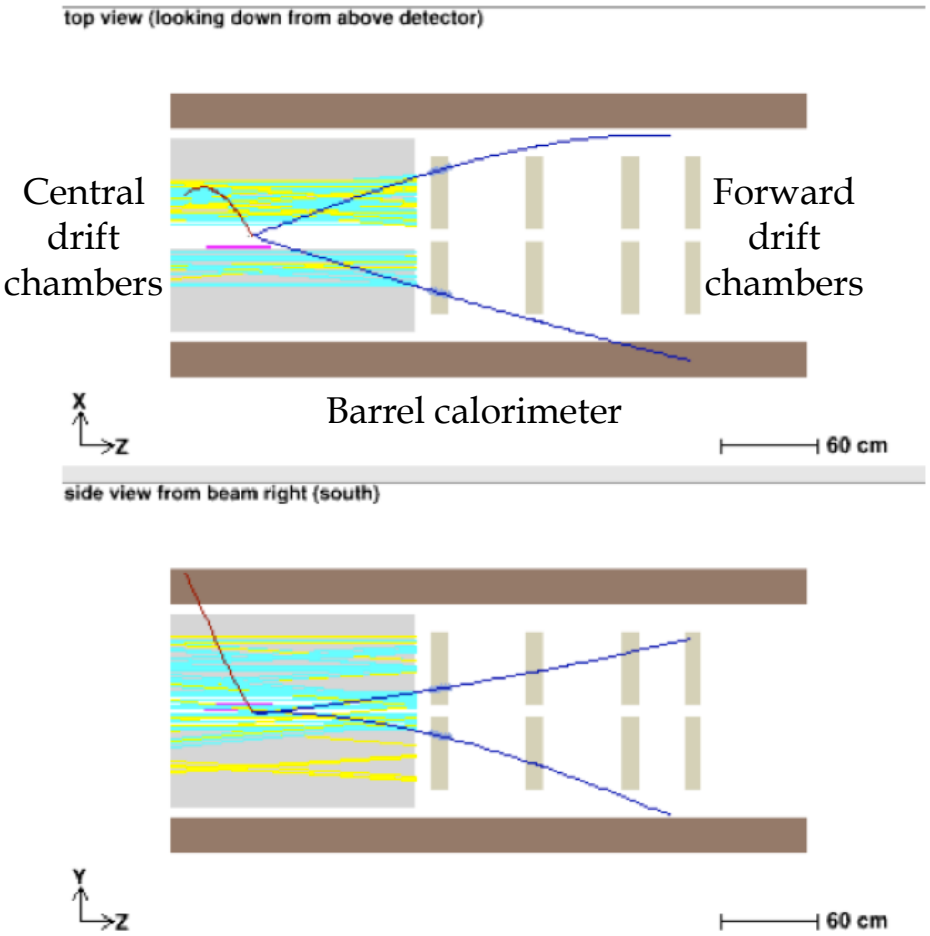
* At 24 threads, takes < 3 hrs

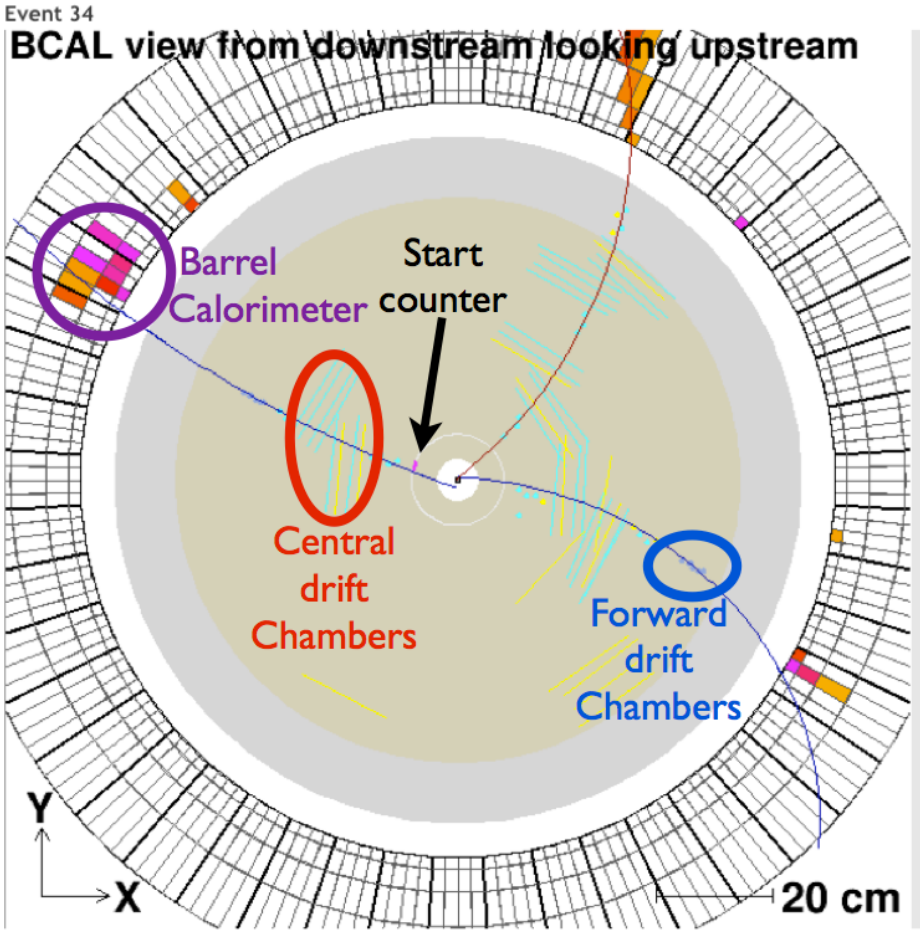


At 1 thread, many jobs timeout!!

Example Reconstructed Event

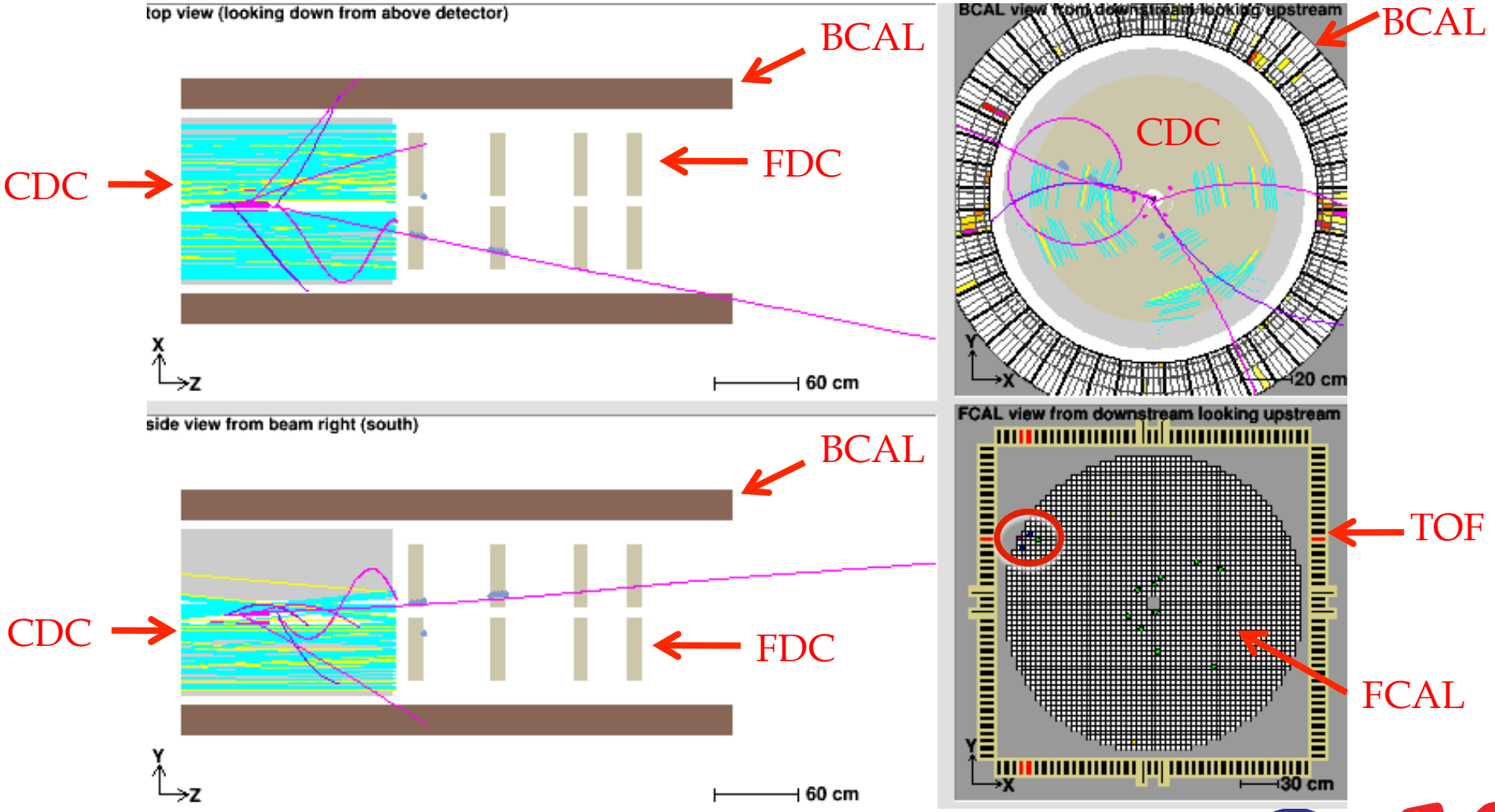
* Tracks, calorimeter showers reconstructed, correlated





Reconstructed Events

- Detector correlation: Tracks, calorimeter showers reconstructed
  - From online reconstruction, first few days of beam



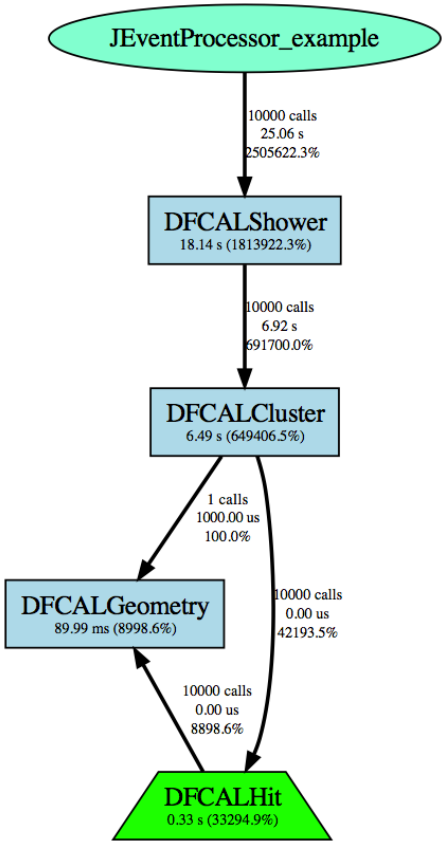
JANA

* JANA: Multithreaded, factory-based, plugin-driven
  * Factory: Dedicated code for creating objects of that type
* User writes plugin to drive reconstruction/analysis
  * Perl script generates template code

E.g. Plugin for FCAL reconstruction (called every event)
- Factory calls on right (**DFCALHit** from file)

```
#include <FCAL/DFCALShower.h>
jerror_t JEventProcessor_example::evnt(JEventLoop* loop, int EventNum)
{
   vector<const DFCALShower*> locFCALShowers;
   loop->Get(locFCALShowers);

   return NOERROR;
}
```



Analysis

- Plugin DReaction: Reaction, analysis
- DANA: Create particle combos
- DANA: Kinematic fit, make combos
- DANA: Execute analysis actions
- Plugin processor: Save results

Kinematic Fitting

DReaction

Particle Combinations

Multiple combinations per event
- Event fails cuts only if all combos fail
- 3 Versions: "Thrown," "PreKinFit," Default (KinFit)

Analysis Actions/Results

Analysis actions: Make cuts, fill histograms
- Encapsulate into DAnalysisAction: easily insert & execute

Many common actions provided by library, insert and run:
- e.g.: Kinematics, PID, KinFit results, misses

Thrown Particles

Three DMCThrown factories:
- Default (all), tag="Decaying," tag="FinalState"

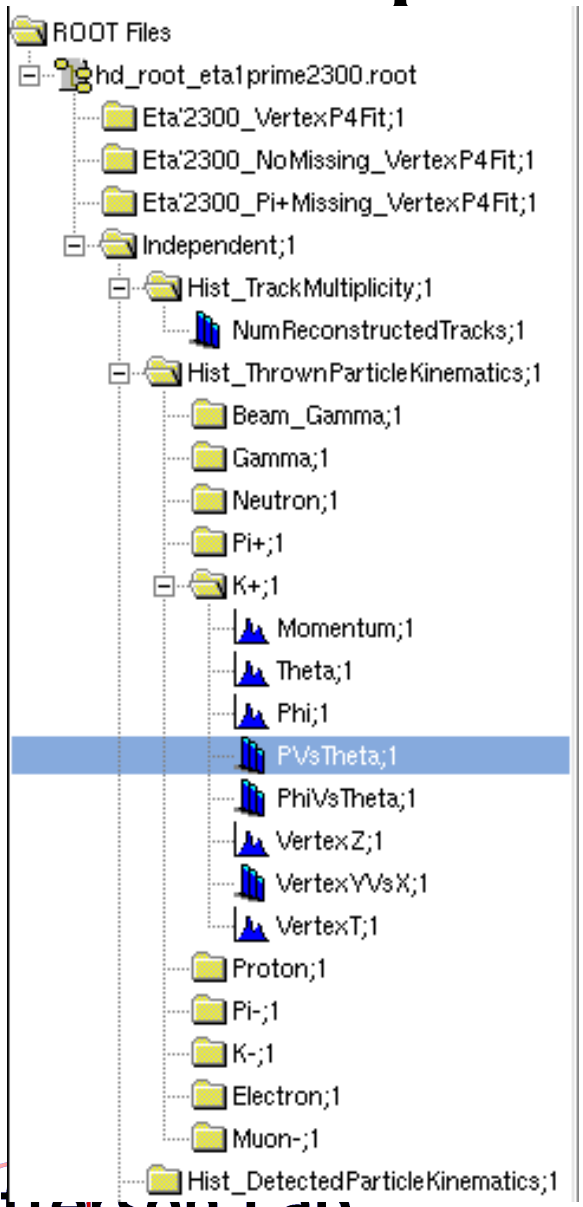
DMCThrown "parentid" = parent "myid"

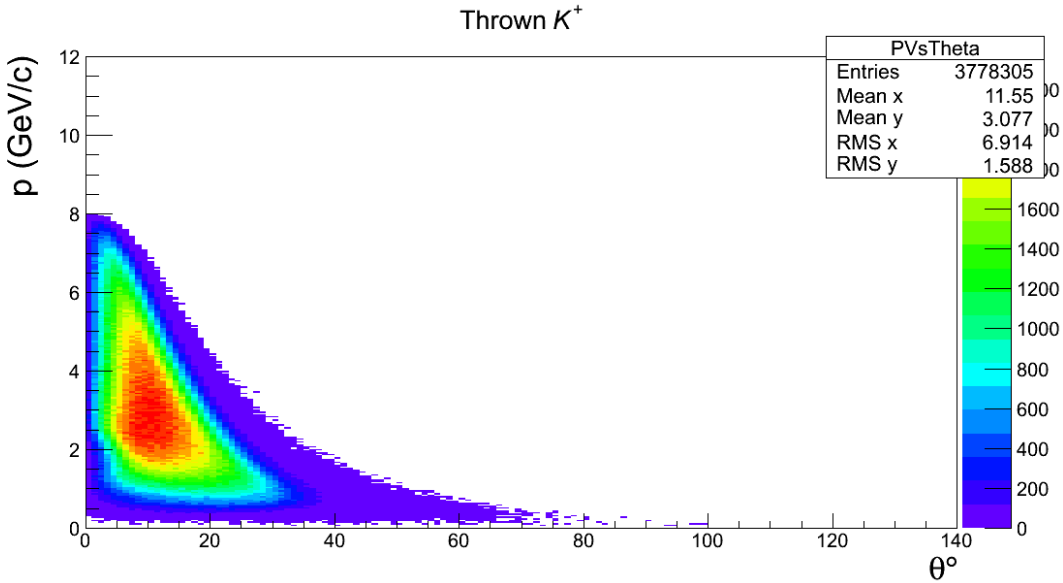
DMCThrownMatching:
- Thrown ⟺ Reconstructed
- Charged & neutral hypotheses: by χ^2
- FCAL & TOF: by χ^2, BCAL: by angle

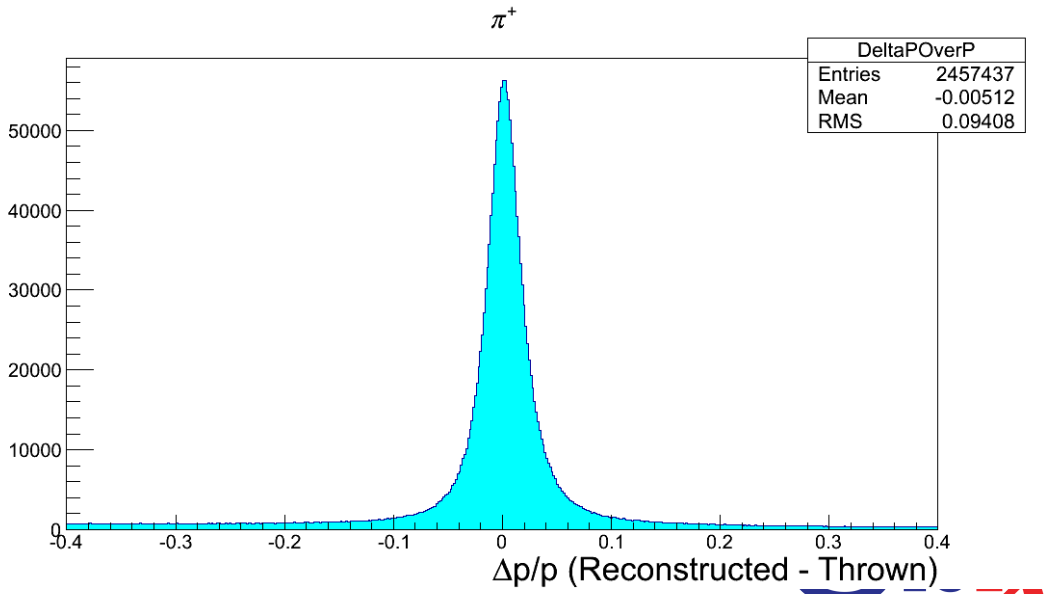
Other factories:
- DReaction tag="Thrown"
- DParticleCombo tag="Thrown"

Output REST, ROOT

Example Histogram Actions

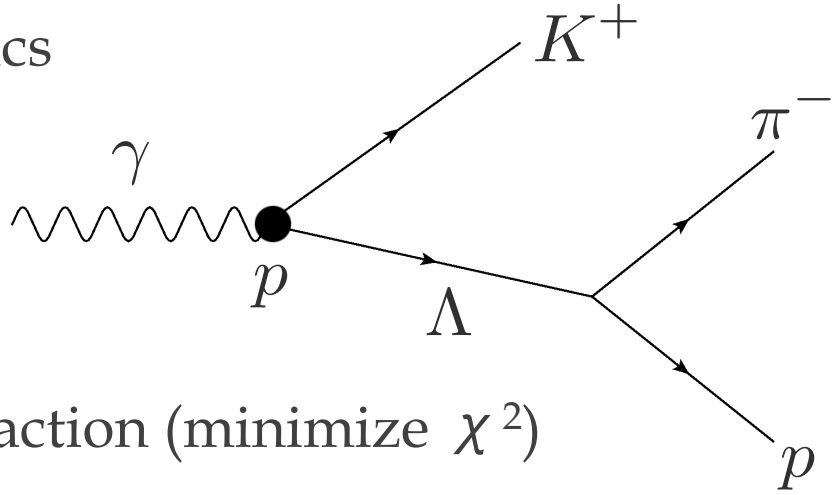


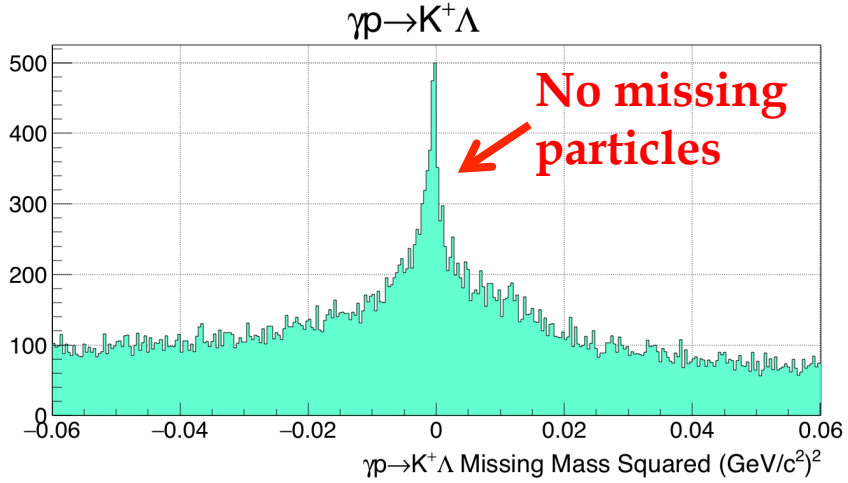


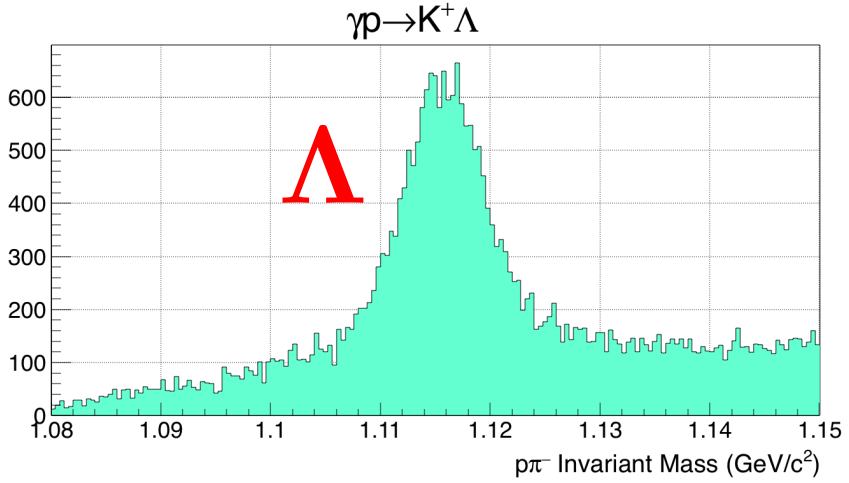


Kinematic Fitting (C++)

* Want to do strange-quark (Λ) physics
  * Backgrounds, e.g. $\gamma p \rightarrow p \pi^+ \pi^-$
* Hypothesis test: Fit the data
  * Was this event the reaction I want?
* Constrain the data to match your reaction (minimize χ^2)
* Powerful: Apply many constraints simultaneously

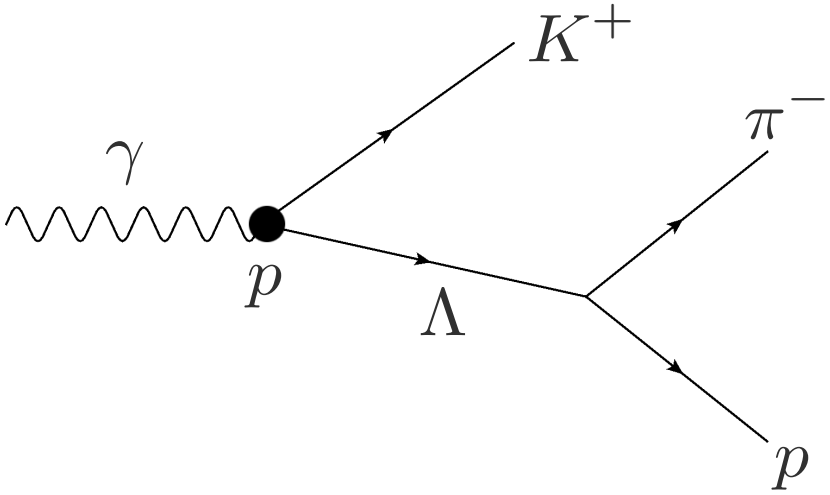


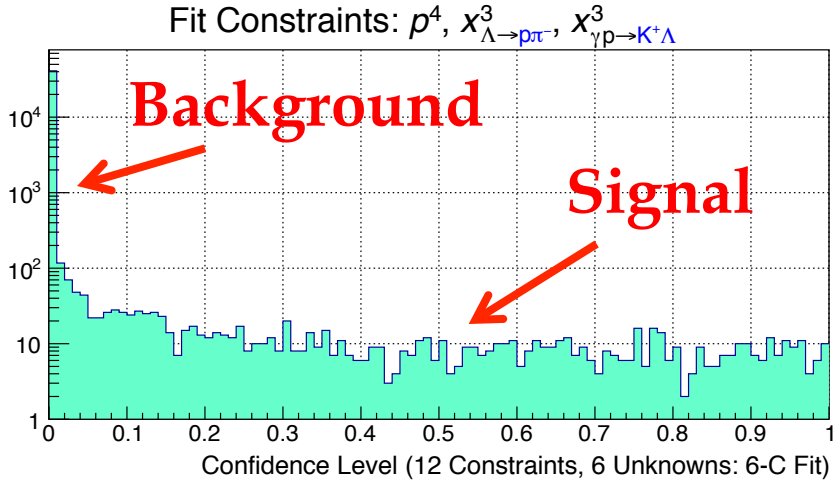




Λ Reconstruction

- Constraints:
  - E & p conservation
  - Production vertex, decay vertex
  - Over-constrained: 6 DF
- Confidence level: Cut near zero
- Clean Λ peak (mass not constrained)

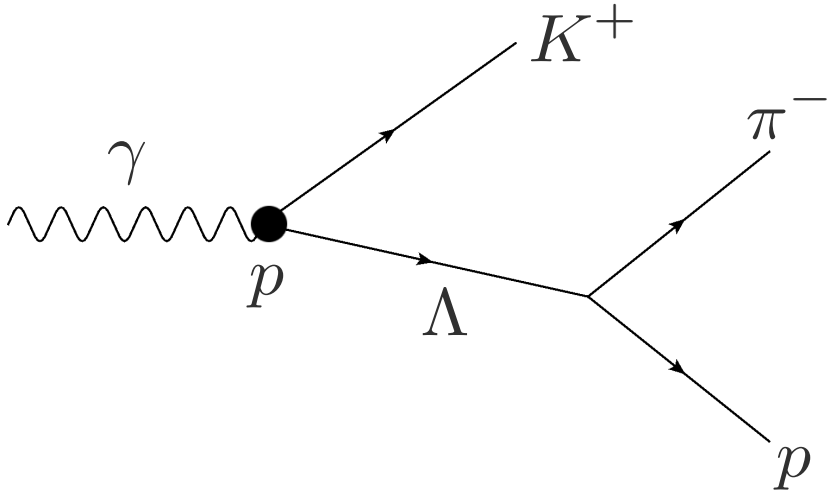


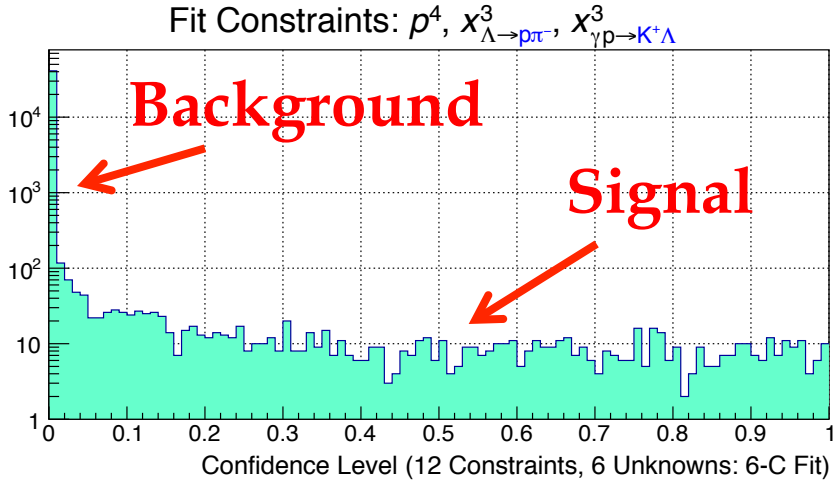


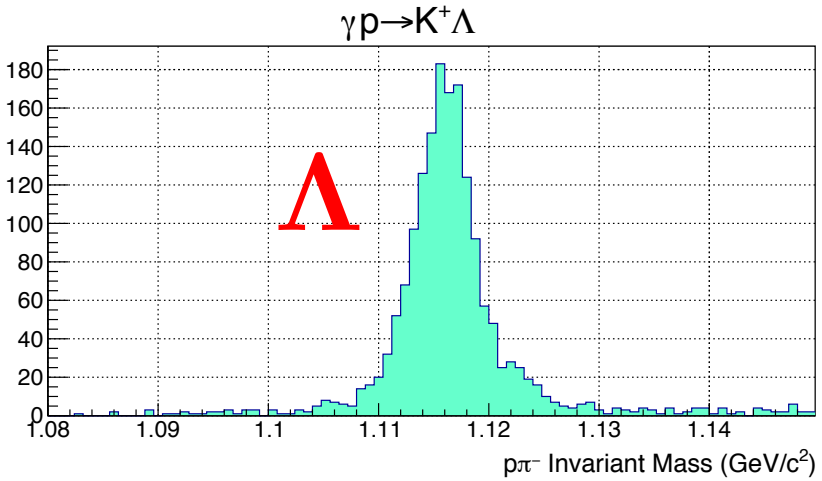
Very low background: Clean Λ selection!!

Λ Reconstruction

* Constraints:
  * E & p conservation
  * Production vertex, decay vertex
  * Over-constrained: 6 DF
* Confidence level: Cut near zero
* Clean Λ peak (mass not constrained)







Very low background: Clean Λ selection!!

Analysis Tracking

* Coordinate collaboration efforts for understanding data

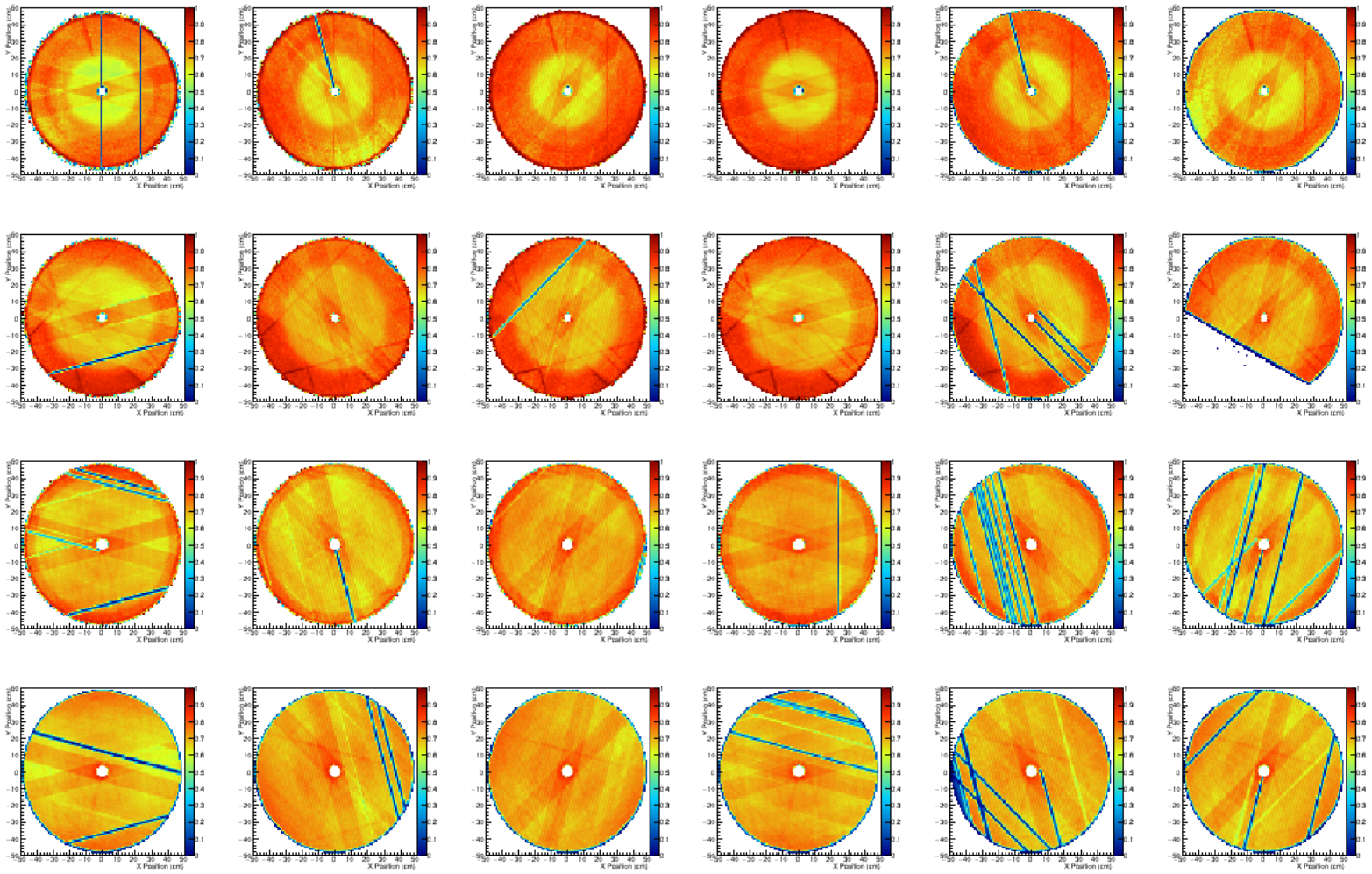
Non-Strange Meson Channels [edit]

Channel	Topology(ies)	Measurement(s)	Analyzer(s)	Status	Analysis/Presentations/Documents
π^0	$\gamma p \to \pi^0 p$, $\pi^0 \to \gamma\gamma$	Σ Asym., $d\sigma/d\theta$, Effic. Study	Sebastian Cole, George Vasileiadis, Justin Stevens, Igor Strakovsky, David Mack, Zhenyu Zhang		Example plugin and Event Generator
η	$\gamma p \to \eta p$, $\eta \to \gamma\gamma$	Efficiency Study	Will McGinley, Sebastian Cole, Regina, David Mack, Zhenyu Zhang		Example plugin
	$\gamma p \to \eta p$, $\eta \to \pi^+\pi^-\pi^0$	Efficiency Study, Dalitz Analysis	Will McGinley, Simon Taylor, Regina		Example plugin
	$\gamma p \to \eta^{(\prime)} p$, $\eta^{(\prime)} \to e^+e^-\gamma$	Efficiency Study, TFF	Cristiano Fanelli, MIT		
η_c	$\gamma p \to \eta_c p$, $\eta_c \to K^+K^-\pi^0$	Effic. Studies	Maria Patsyuk		
ω	$\gamma p \to \omega p$, $\omega \to \pi^+\pi\pi^0$	Σ Asym., $d\sigma/dt$, SDME, Effic. Study, Dalitz Analysis	Alex Barnes, Mike Staib, Alex Somov, Alyssa Henderson, Sebastian Cole, Paul Mattione		Example plugin, Tutorial
	$\gamma p \to \omega p$, $\omega \to \pi^0\gamma$	Σ Asym., $d\sigma/dt$, SDME, Effic. Study	Mike Staib, Alex Somov		Example plugin
	$\gamma p \to \omega p$, $\omega \to \pi^0\gamma$, $\omega \to \pi^+\pi\pi^0$	Calorimeter Effic. Study	Jon Zarling		Example plugin
η'	$\gamma p \to \eta' p$, $\eta' \to \pi^+\pi^-\eta$	Bump Hunt	Regina, FIU		Example plugin

FDC Hit Efficiencies (Alex A.)

Pseudo hit = wire position + clusters in both cathodes (position along wire)

- Requires matching position perpendicular to wire and timing

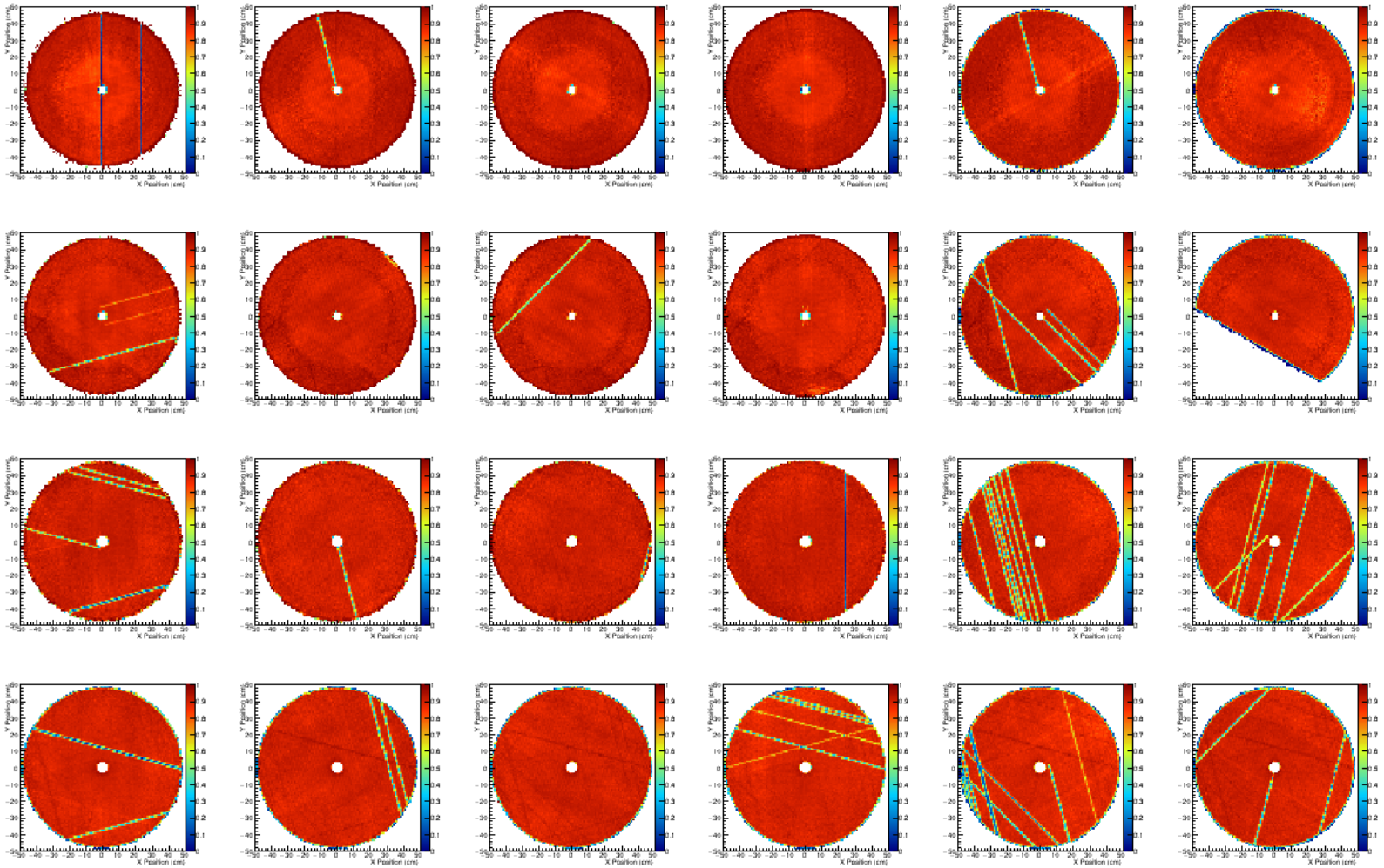


Efficiency ≈70%

FDC Hit Efficiencies (Alex A.)

Pseudo hit = wire position + clusters in both cathodes (position along wire)

- Requires matching position perpendicular to wire and timing



Efficiency ≈90%
+10% reconstructed tracks

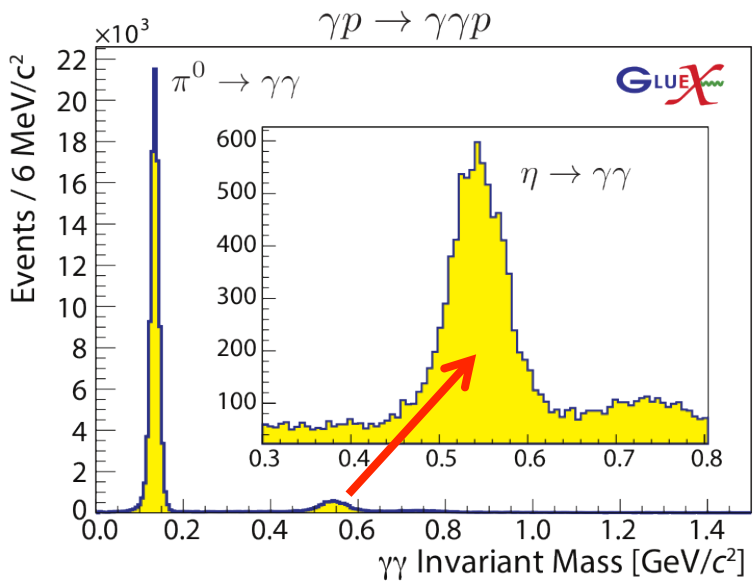
Detector/Reconstruction Studies

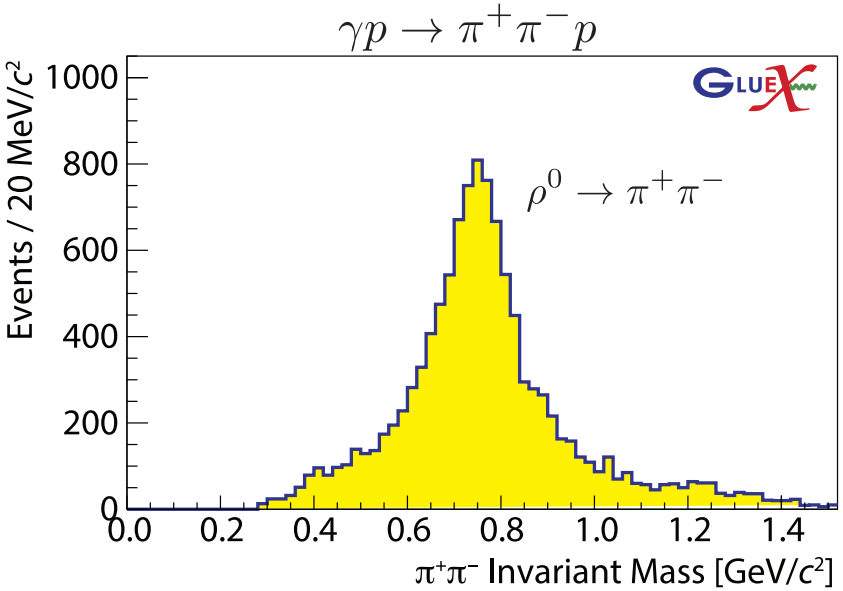
* Beam:
  * Beam Polarization (Talks by Justin & Mike D.)
  * Beam Energy (Talk by Alex D.)
  * Beam Flux (Talk by Justin)
* Tracking
  * CDC Hit Efficiency (Mike S.)
  * FDC Hit Efficiency (Alex A.)
  * Track Reconstruction (Talk by Simon)
  * Track Resolution & Efficiency (Paul M.)
* TOF/SC:
  * TOF Efficiency (Beni)
  * SC Efficiency (Mahmoud)

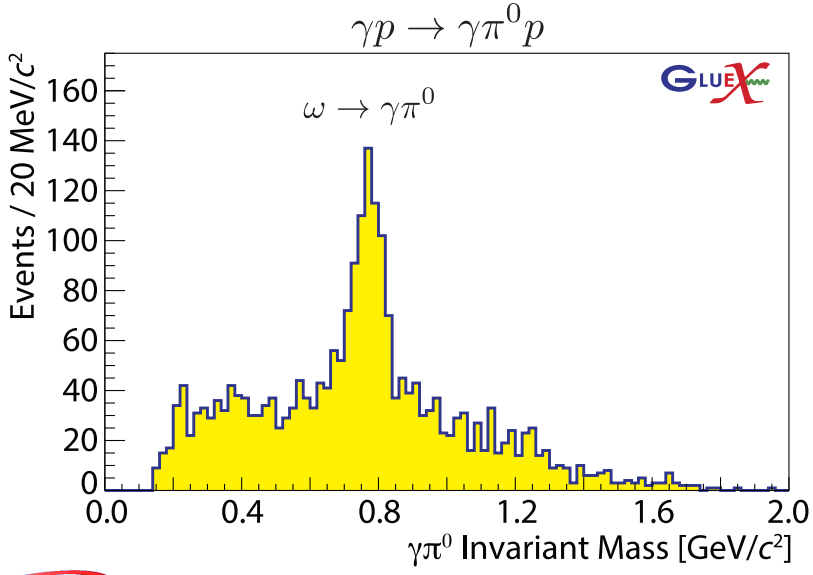
Detector/Reconstruction Studies

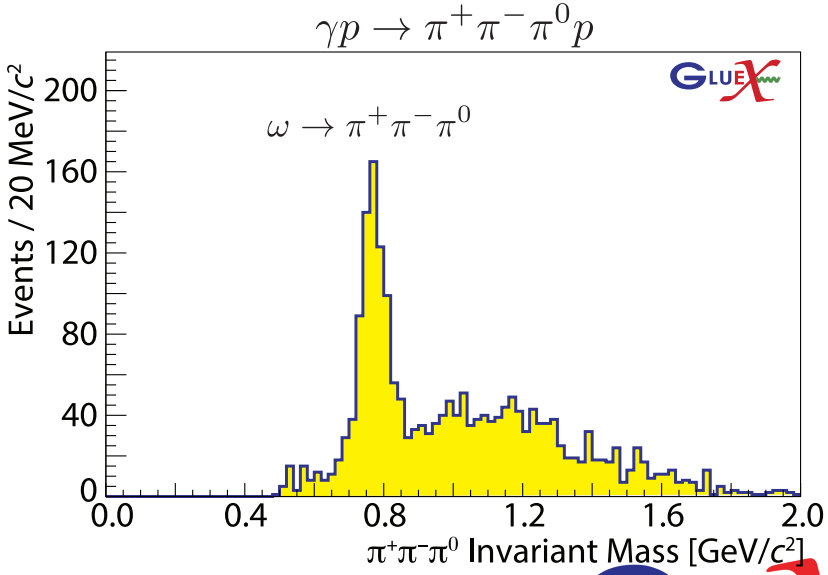
- BCAL:
  - Neutral Energy & Efficiency
  - Hadronic Energy & Efficiency (Elton)
  - Covariance Matrix (Mark D., testing soon)
- FCAL:
  - Neutral Energy & Efficiency (Jon Z.)
  - Hadronic Energy & Efficiency
  - Covariance Matrix (Mark D., testing soon)
- Channel reconstruction, triggering, & acceptance:
  - Triggering (Talk by Alex S.)
  - Magnetic field comparison
    - ρ (Alex A.), 4π (Alex A.), ω (Mike S.), ϕ (Alex B., see his talk)

Reconstructed Meson Peaks









$\gamma p \rightarrow pK^+K^-$

* Use KinFit cut to ~remove ρ