SVT/Tracking Software

HPS Collaboration Meeting June 18th 2014

Per Hansson Adrian

Outline

SVT DAQ related software

- SVT calibration

SVT reconstruction software

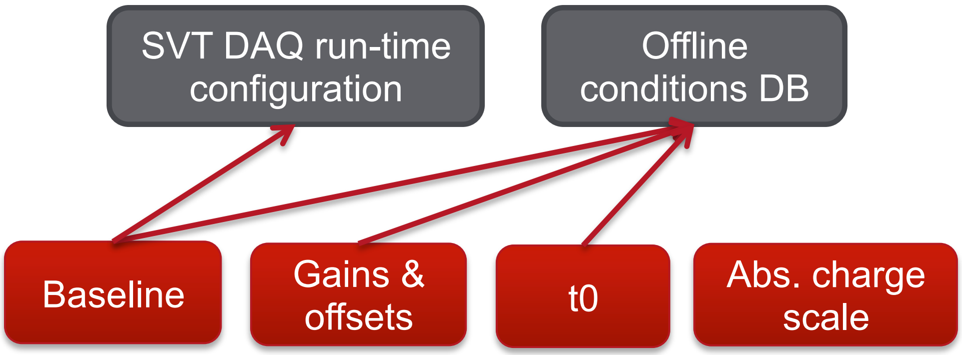
- Hit reconstruction
- Track finding
- Track fitting
- Alignment
- Magnetic field

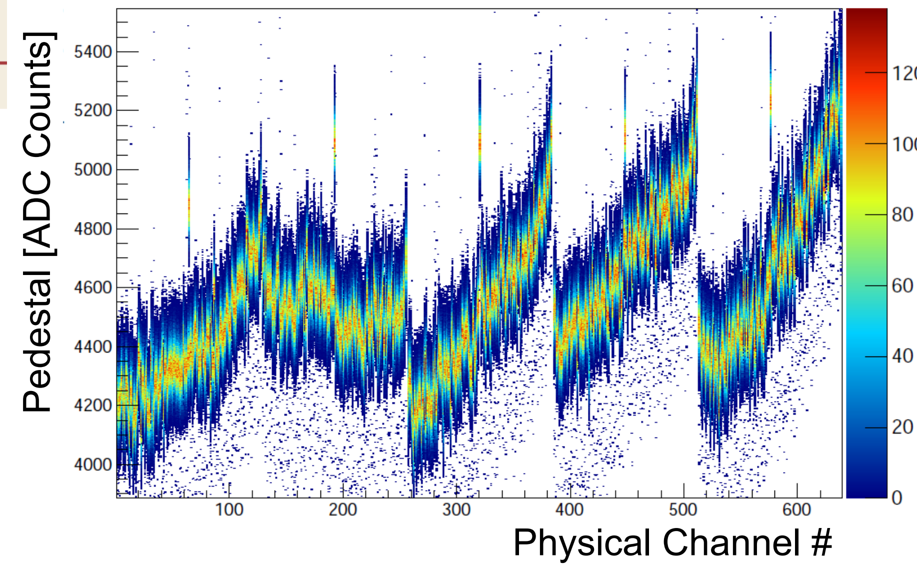
Offline calibration and performance

Monitoring

- Online Monitoring
- Offline and data quality monitoring

Calibrating the SVT





Calibration of the SVT will require

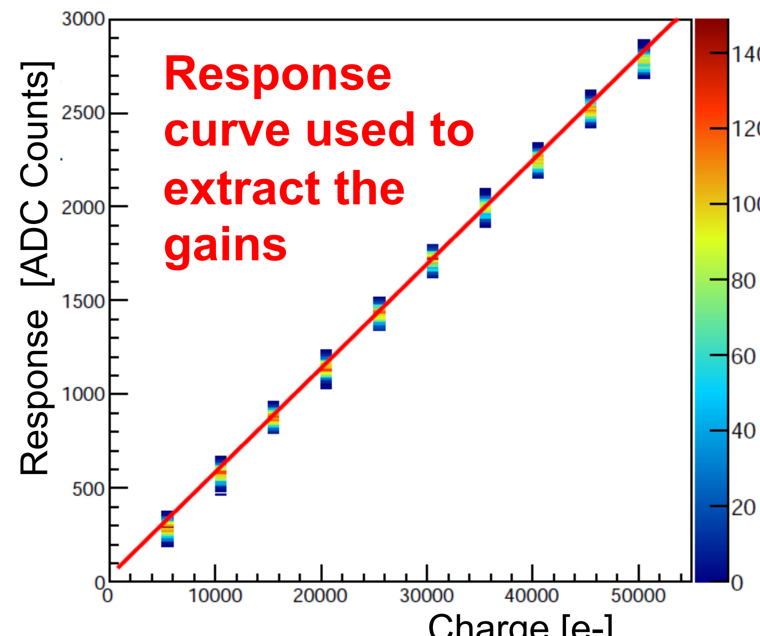
- Extraction of the pedestals and noise for each of the 23’040 channels
- Extraction of the gains and offsets
- Runs at different t0 values used to study the pulse shape

Extraction of the gains and offsets will be done using the internal calibration circuitry of the APV25’s

- The absolute charge scale will be set using Cd109 source data taken at SLAC

Baseline runs used to extract the baseline and noise will be taken before every run (if possible)

- Large variations in the baseline and/or noise can point to possible issues such as damaged chips, problems with power distribution etc.



Taking Calibration Runs

SLAC

Goal is to have calibrations runs taken by shifter

- CODA run type selects SVT special run and configuration of the SVT (new compared to Test run)
- Python analysis scripts will allow shifter to analyze calibration runs (command line and GUI if/when time permits)
- The current run will be compared to previous runs (find large variations, spot new dead/noisy channels
- Calibration constants loaded to conditions DB and new SVT configuration after shifter analysis (via text files and custom Java API)

Sergey/B. Reese

Omar/Sho

Observable	Frequency	Shifter/Expert	Estimated Time
Baseline (extract pedestal and noise)	As often as possible (every run)	Shifter	~1 minute
Calibration (Check response and calibrate fit to shaper signal)	Every six hours (?)	Shifter	~5 minutes
Gain & t0 (Check linearity and extract gain)	Once, before running begins.	Expert	~25 minutes

SVT Reconstruction Overview

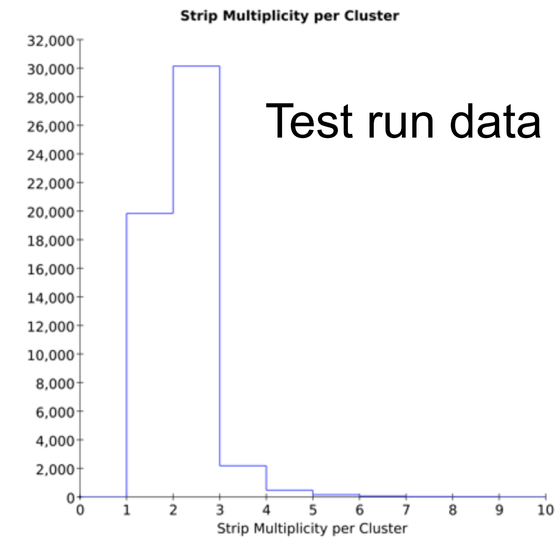
Item	Major development since Jan 21st	Ready for day 1	Future major development
Hit reconstruction	No	Yes	Yes
Track finding	No	Yes (*)	No(*)
Track fitting	No	Yes (**)	Yes
Vertexing	No	Yes	No
Track-based alignment	Yes	No	Yes
Magnetic field	No	Yes	No

(* straight through tracking)
(** hit time in track fit)

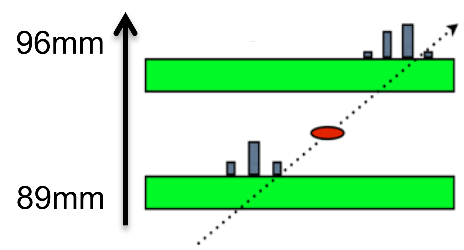
Hit Reconstruction

Two major steps

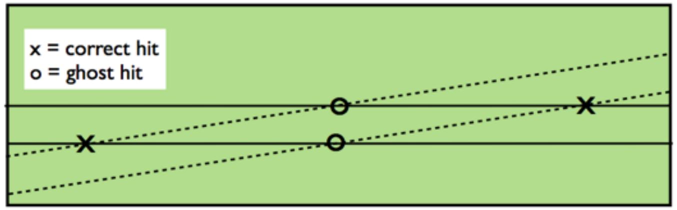
- **Strip clustering** based on nearest neighbor algorithm (1D): seed strip ($>4\times\sigma_{noise}$), neighbor ($S>3\times\sigma_{noise}$), reject clusters with $S<4\times\sigma_{noise}$
- **Stereo hit maker** based on all combinations of clusters in adjacent stereo pair sensors (starting point for tracking)



Overlapping clusters currently dealt with at track selection stage (distance to neighbor cut)

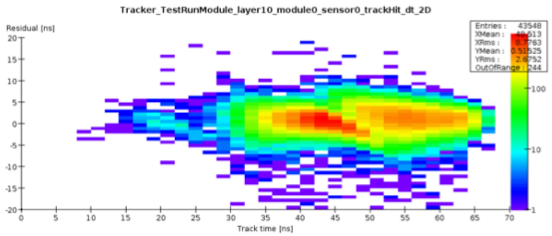


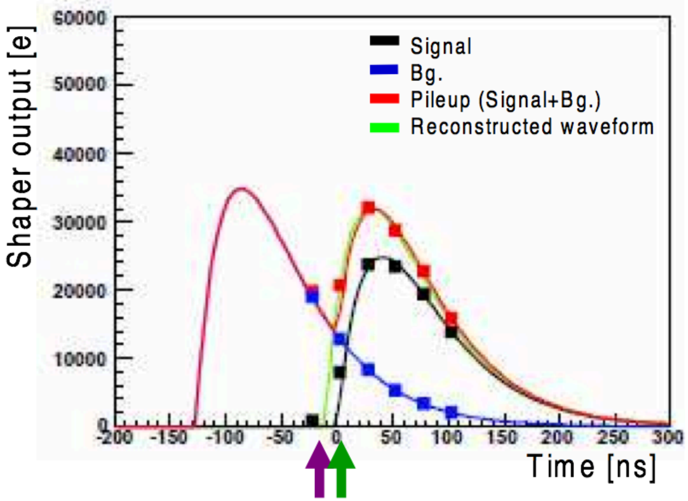
Only major development is improving the pulse shape fitting to handle pile-up better

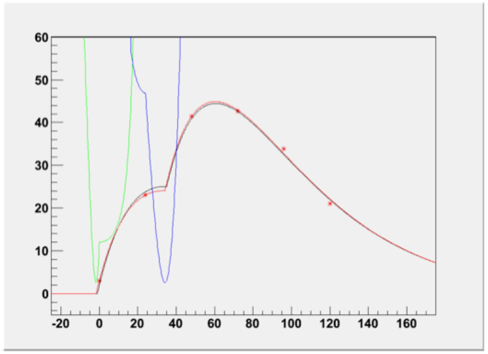


Pulse Shape Fitting

- Current recon only fits single pulses, using ideal CR-RC pulse shape
- Need to use actual pulse shape in fit, otherwise we see time-dependent pull on fitted hit time
- Identify and fit pileup — background hits before the signal hit will pull the fitted hit time to the left
  - In hottest strips, mean time between hits ≈ pulse width
  - Without this, need to weaken the time cut a lot — track finding 100x slower and track efficiency still only 80%
- Good progress made in toy MC . . . in 2011 (Sho)
  - Finish it off, port to Java — 1 month?

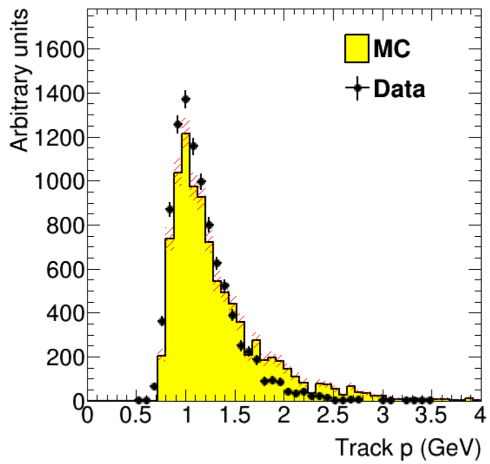




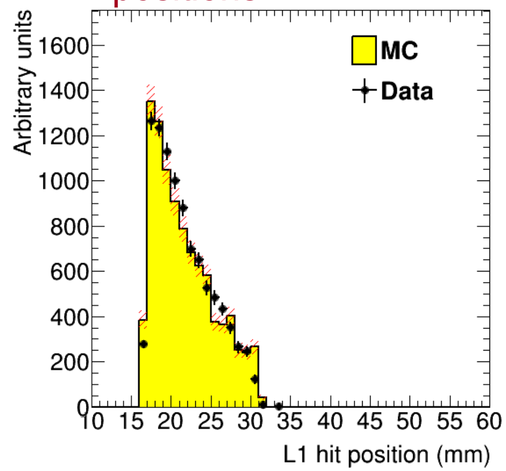


Tracking works

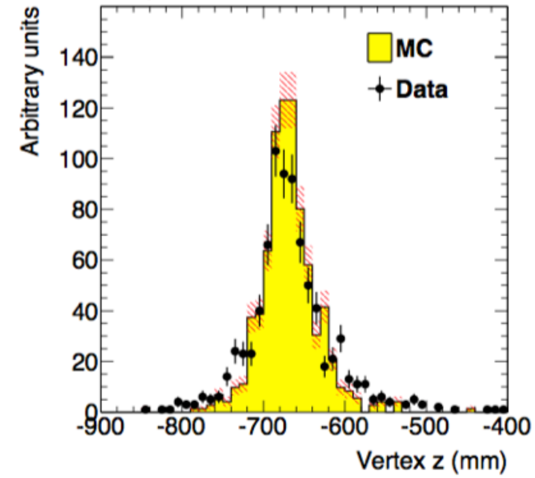
Track momentum



Vertical stereo hit positions

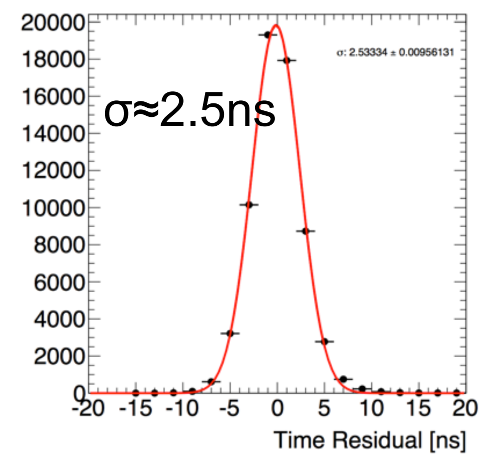


Converter (vertex) position



Tracking software already exercised in Test run

- Used in both online monitoring and offline analysis
- Good performance
- Speed exercised fully in mock data challenge



⇒ The basic software for HPS operation is already there

Track Finding and Fitting

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

SLAC

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

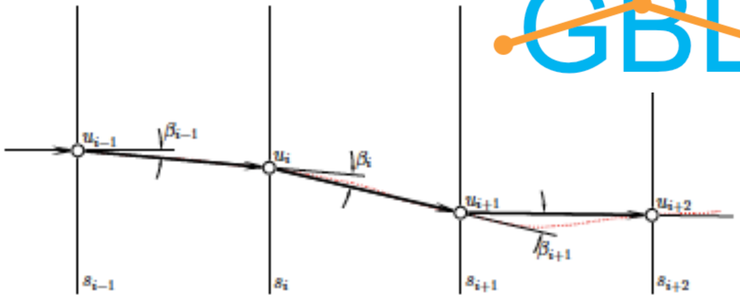
- Seed-confirm-extend philosophy: fast
- Tightly coupled to track fitting

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



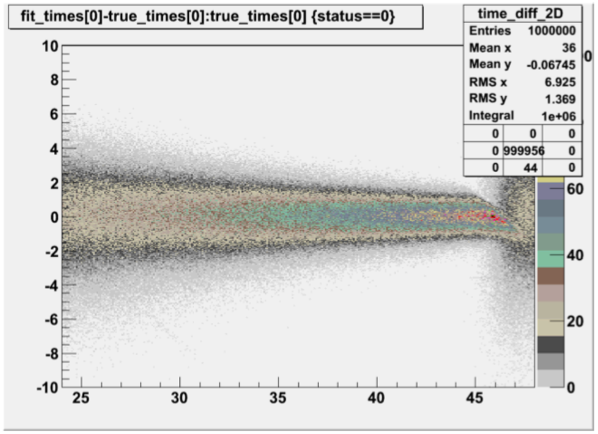
$$S(u) = \sum_{i=1}^{n} \frac{(y_i - u_i)^2}{\sigma_i^2} + \sum_{i=2}^{n-1} \frac{\beta_i^2}{\sigma_{\beta,i}^2}$$

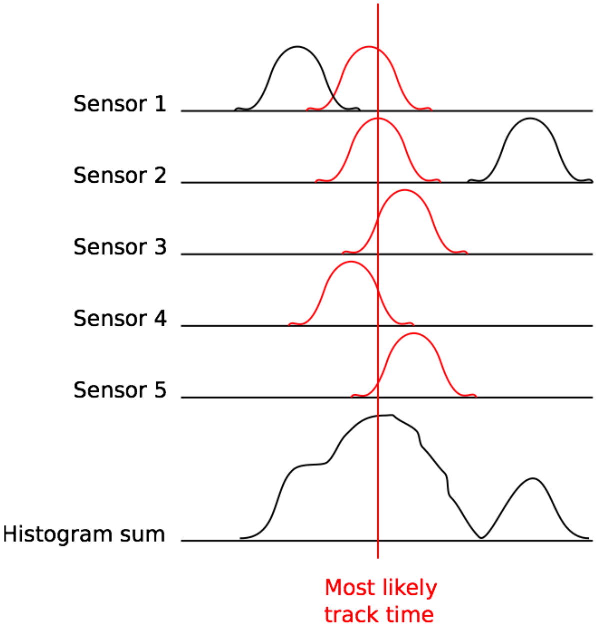
Future major developments

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

Track Finding using Hit Time

- Current recon: reject all hits outside a time window (relative to reconstructed ECal hit time), then use a time-blind track finder
- Hit time resolution varies:
  - Depends on relative timing of t0 and SVT clock
  - Pileup causes some hits to have very poor time resolution
- Simple plan: use hit times and resolutions to calculate a χ^2 or likelihood for candidate tracks, feed into track finder
  - Fun plan: "histogram tracking" finds high-likelihood sets of hits without prior knowledge of t0
- 2 weeks?



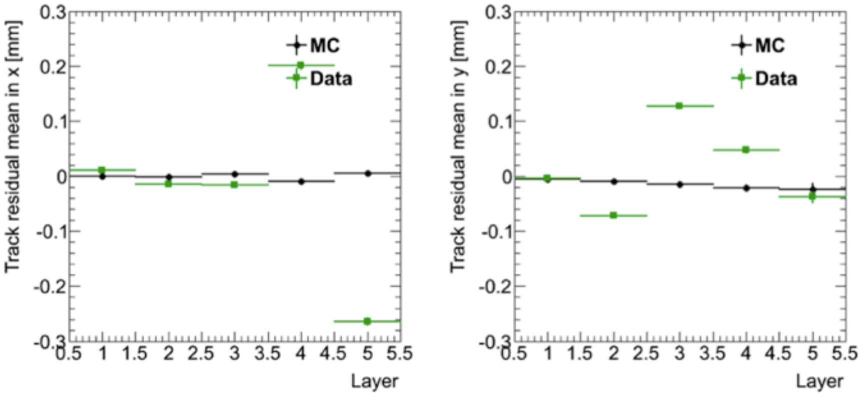


V. Blobel, C. Kleinwort, F. Meier, Computer Phys. Communications (2011)
Kleinwort, NIM A, 673 (2012), 107-110

Track-based Alignment

Survey is starting point for alignment

- Test run reached ~100um residuals w/ some manual corrections
- Expect better for 2014 (see Shawn's, Takashi and Tim's talk)
- Roughly, ~10um is sufficient for HPS



Mean of biased track residuals vs tracker layer

Multiple ways to achieve similar performance

Our approach:

- Do a least square fit of local (track) and global (alignment) parameters
- Millepede-II can do this for us and is "supported" by GBL
- Great support from C. Kleinwort (GBL/Millepede developer)

Residual for measurement *i*:

5 track parameters

Subset Ω of *n* alignment constants

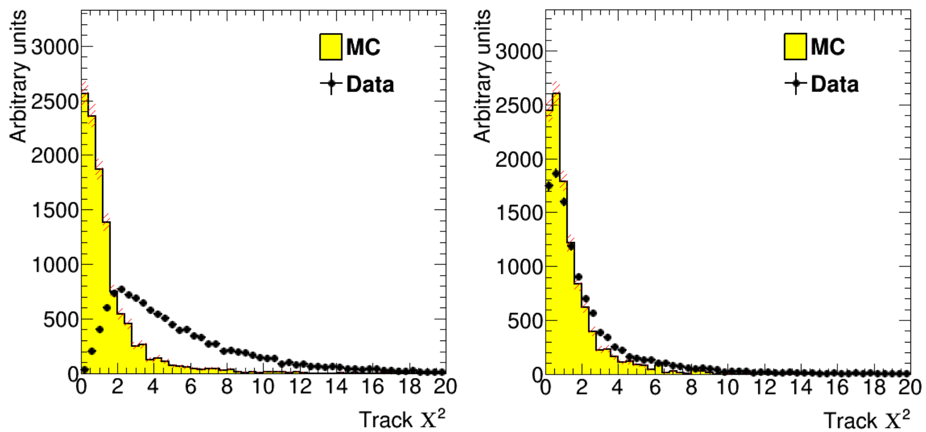
$$y_i - f(x_i, \mathbf{q}, \mathbf{p}) = \sum_{j=1}^{5} \left(\frac{\partial f}{\partial q_j} \right) \Delta q_j + \sum_{l=1}^{\Omega} \left(\frac{\partial f}{\partial p_l} \right) \Delta p_l$$

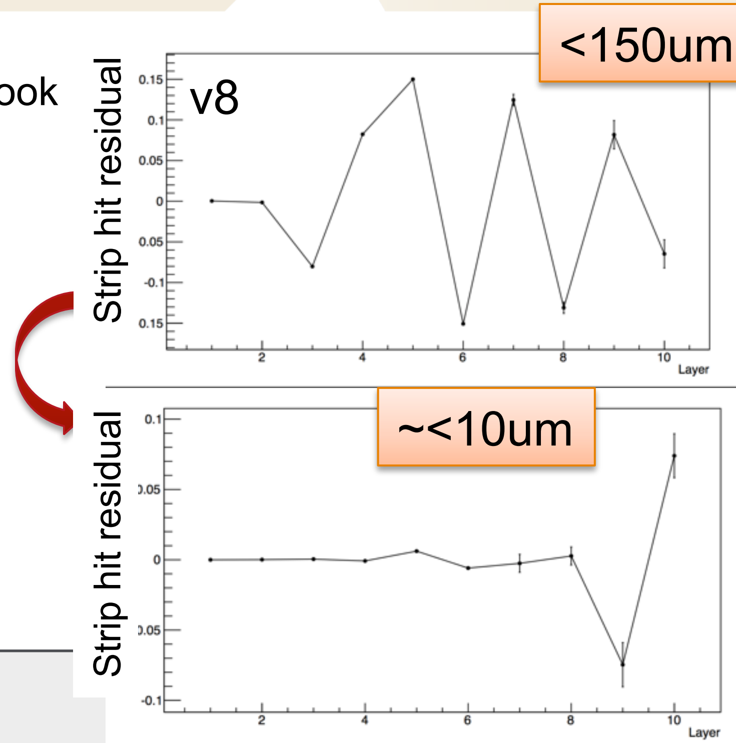
MPII

Track-based Alignment

Use Millepede-II to align the Test run detector

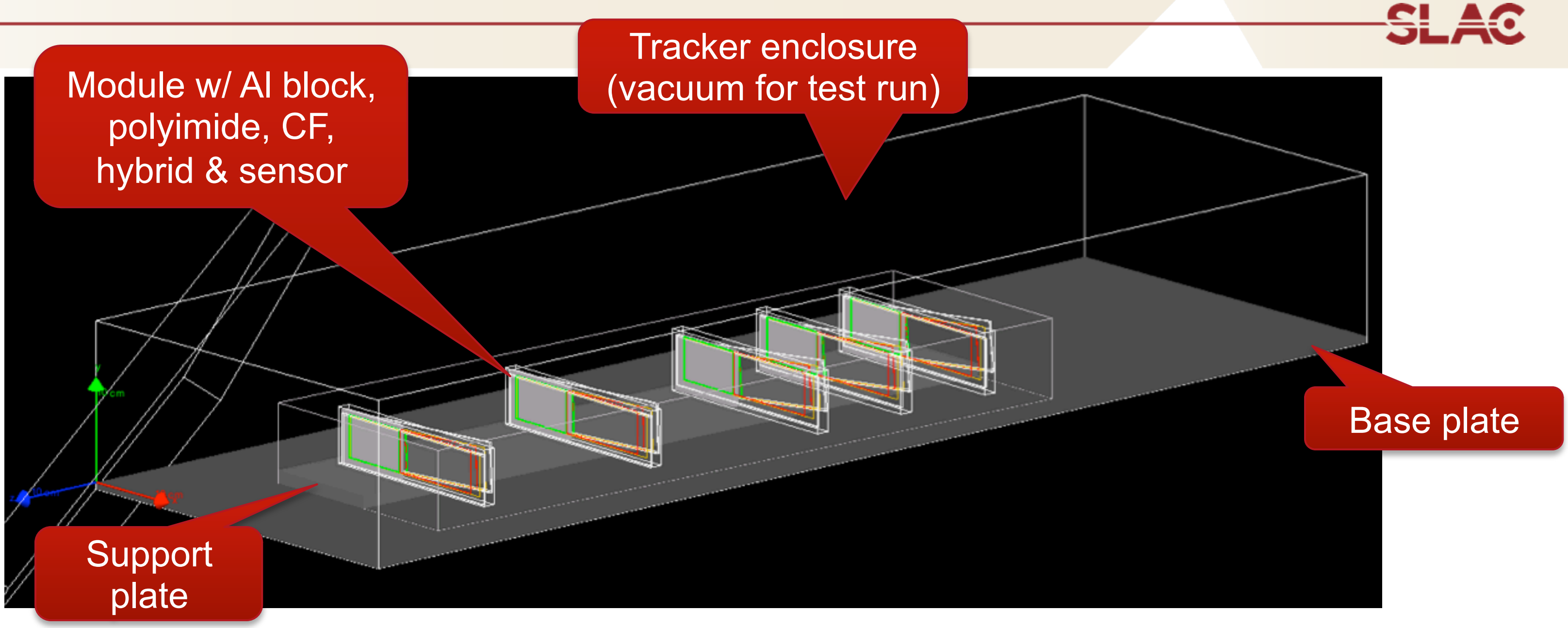
- It works! But translations only here; rotation corrections look ok but need updated geometry description





Millepede-II	•	L1-3 alignment global constants Include vertex in minimization
Geometry tools	•	Geometry implementation based on detector survey
Special run	•	Include straight line track sample and check improvement Determine trigger and sample size needed
Operational procedures	•	Streamline software Monitoring – rapid feedback during run (beam spot, chi2, track residuals) Offline and online shifter responsibilities

Track-based Alignment



New geometry based on production drawings

- Simplify bootstrapping from survey
  - Built from surveyed positions on mech. drawings
- Accommodate alignment constants better
  - Global and local translations/rotations automatically ok
- More complete dead material
- Need a couple of more weeks until finished

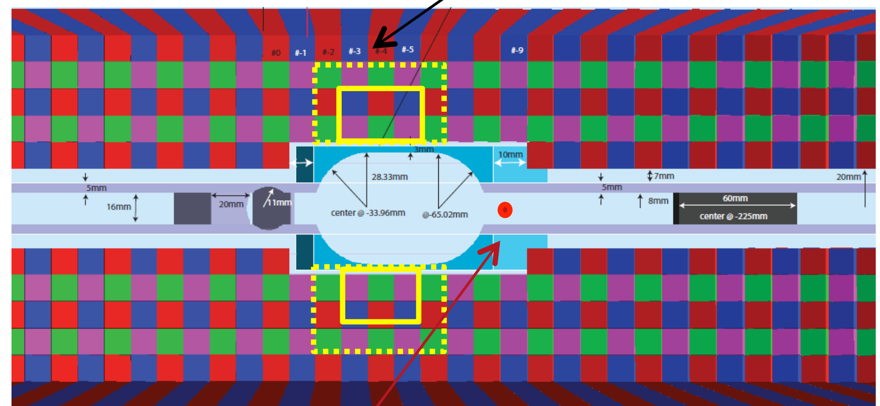
Test run used as test bed, “simple” to go to new SVT

SVT Alignment Special Run

Only these crystals needed for trigger.

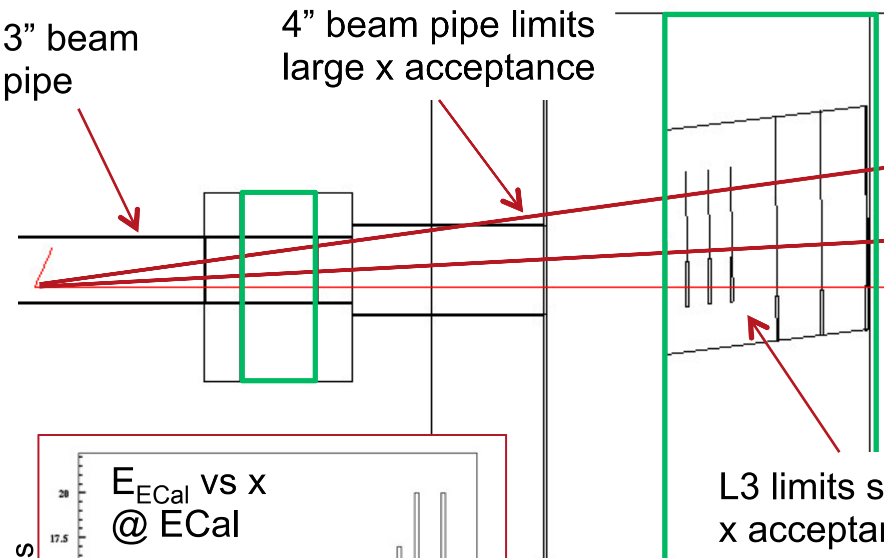
Straight-through tracks for alignment

- Chicane magnets off, particles from 10^{-4} r.l. gold foil at collimator (z=-330cm) and HARP (z=-281cm)



3” beam pipe

4” beam pipe limits large x acceptance

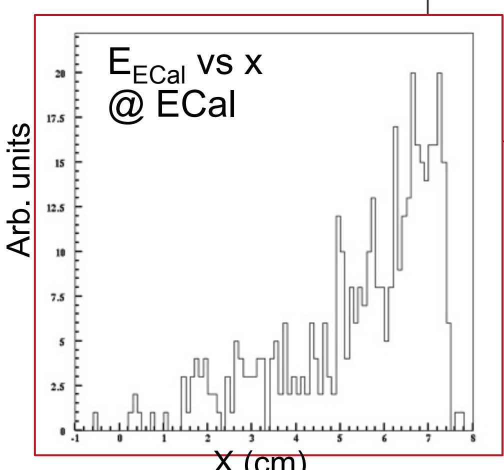


θ_x<32 mrad

θ_x>8.2 mrad

Straight-through beam

L3 limits small x acceptance.



Beam energy single cluster trigger rate:
2.6 kHz @1nA

When?
We do not need this run in the beginning; request when it is convenient and tracker/trigger is stable?

Questions:
Do we want more coverage in the Ecal for energy calibration purposes, should we extend trigger to cover more crystals?
How large sample is needed for SVT alignment?
Do we split the run between foil at collimator and HARP to get two track samples?

SVT Performance

Two main topics

- Momentum scale & resolution
- Angular resolution

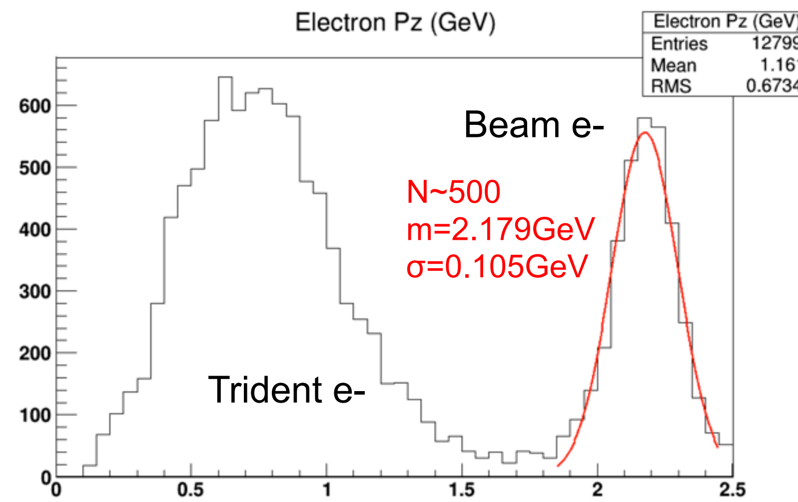
Analysis	Topic	Special trigger	Special run	Status
Beam energy electrons	Scale & resolution	Yes	No	Well understood
Moller scattering	Scale, resolution, angular resolution	Yes	No	Need trigger study and analysis
Trident kinematic fit	Scale, resolution, angular resolution	No	No	Need work

Beam Energy Electrons for Scale and Resolution

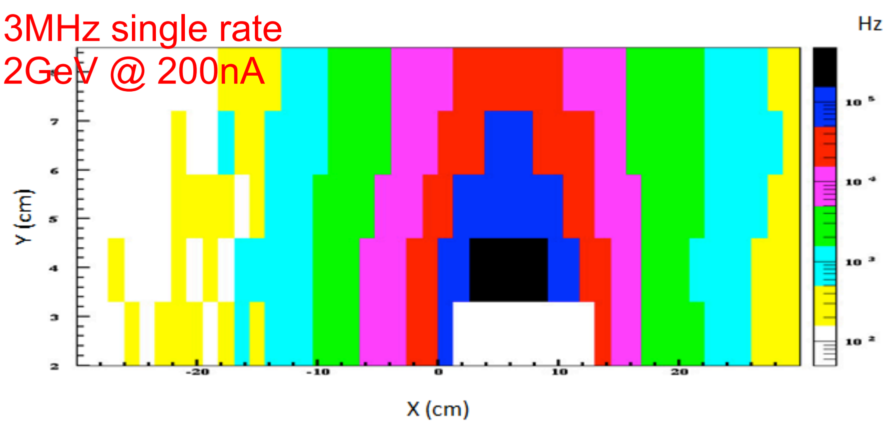
Get approximately full energy electrons from elastic e-W scattering.

- Momentum calibration point over full acceptance for SVT (and ECal)
- Plenty of them in A' signal triggered events
- Include pre-scale single cluster triggers to get more uniform coverage of ECal if needed

Mock data; ~0.65s of beam!



7.3MHz single rate
2.2GeV @ 200nA



Ecal/SVT accept.	# clusters/3h	# clusters/ crystal/3h
Total	540k	-
Black	310k	52k
Red	126k	7.4k
Pink	91k	1.6k
Green+light blue	12k (+15Hz)	150 (2k)
Yellow+white	1k (+12Hz)	15 (2k)

Estimates (John & Takashi) taking into account overlaps show 50Hz of useful clusters

- Early mock data results show this is reasonable

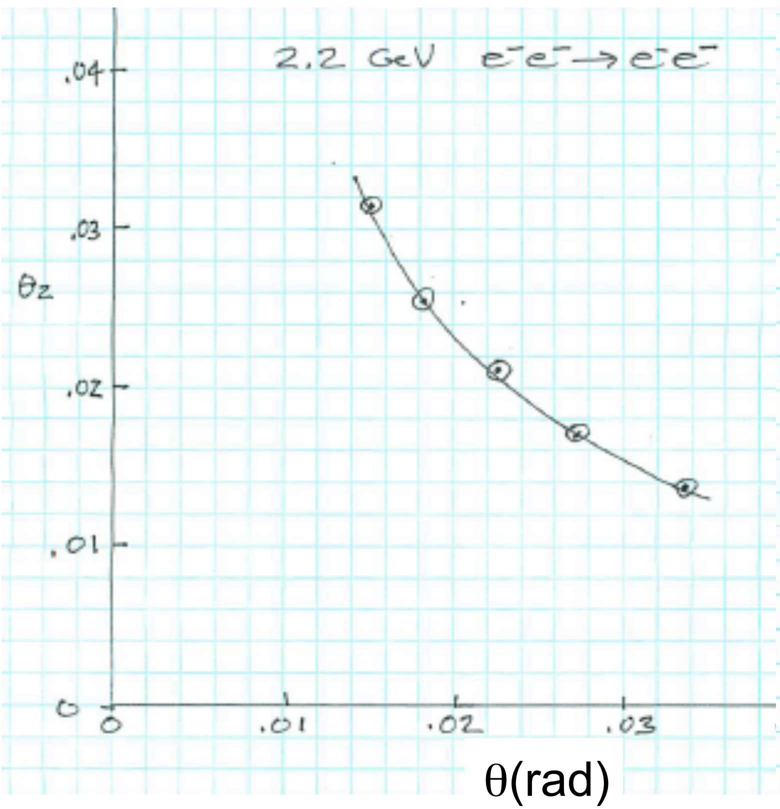
Parenthesis are menu augmented with single cluster prescaled to 10^4 and 10^3

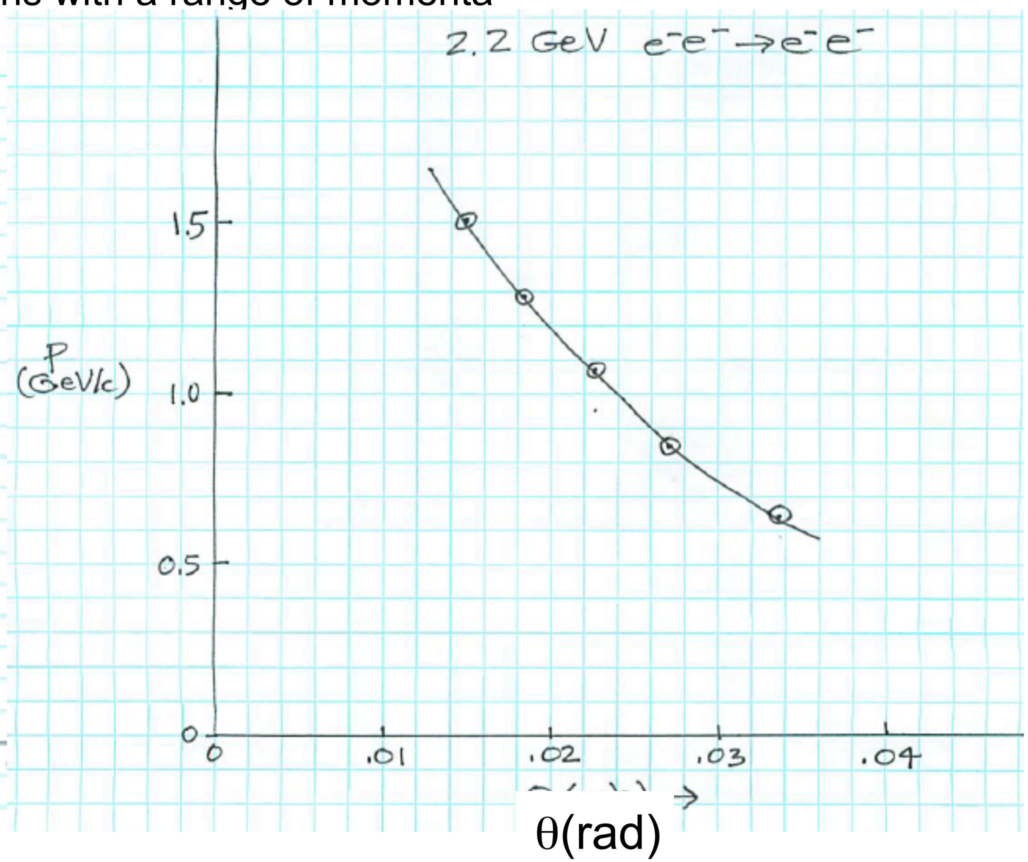
Möller Scattering

John, Takashi

Elastic e-e- scattering can put both particles into the SVT acceptance

- Angle-momentum perfectly correlated, so can measure angle and check momentum scale and resolution
- Angle-Angle perfectly correlated, so can measure angular scale and resolution.
- This would complement beam electrons with a range of momenta

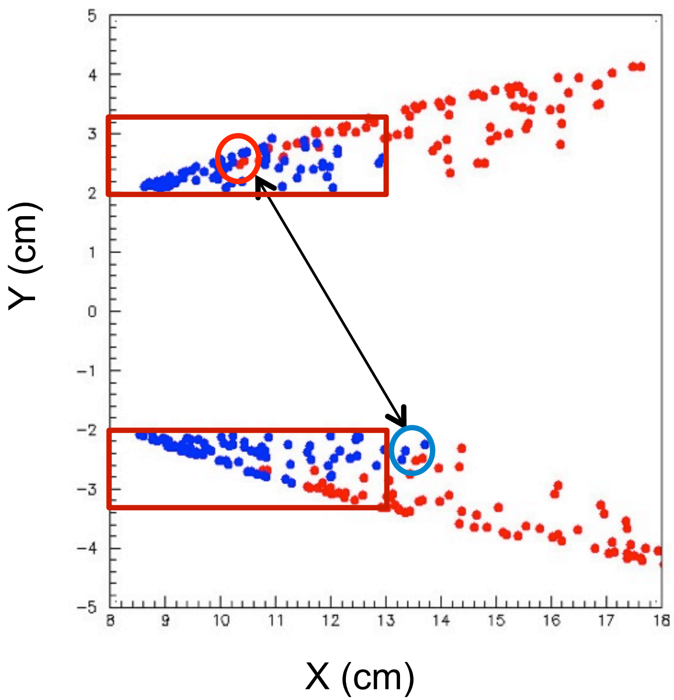




Möller Scattering

John, Takashi

SLAC

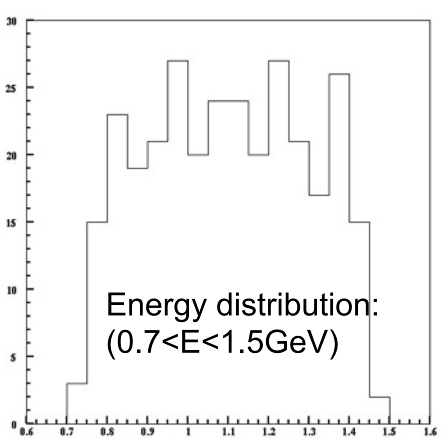


One of each pair hits region where crystals are removed

- Simple two cluster trigger won't work.

Will single cluster trigger work?

- Top right or bottom right
- Select crystals only
- Energy selection 0.7<E<1.5GeV
- Background primarily tridents: how much?



Lots of events:
0.34MHz of
Möller electrons

E (GeV)

Off-line can clean it up with SVT (?)

- top-bottom coincidence
- total energy = beam energy
- angle-momentum and angle-angle correlations

Online Monitoring

Observable	Summary	Status
Occupancy	Spot shifts in the baseline, noisy channels and misconfigured APV's	Done.
Data Rates	Cross check occupancies and DAQ	
Cluster charge	Gain information.	Minor work.
Shaper signal samples	SVT timing.	Minor work.
Hit t0 times	SVT timing.	Minor work
Hit efficiencies	Status of sensors.	Minor work.
Tracking "efficiency"	#tracks/trigger. Overall SVT system indicator.	Minor work
Track residuals	Alignment.	Minor work.
Beam spot & vertexing	Alignment and beam quality	Minor work

Slow control will monitor

- Power (FEB, hybrid), HV bias
- Temperature (FEB, hybrid, chiller)

SVT Offline Monitoring Shifter

Offline shifter for SVT

- Tracking expert shifts; in addition to data quality shifter
- Large work load initially, ~20% work load long-term
- 5 day expert shifts
- Maybe merge into single HPS offline/data quality shifter over time?

Responsibility

- Run reconstruction on previous day runs
- Detailed SVT report
  - Alignment: was the detector moved? What geometry to be used offline (and online)?
  - Feedback on calibrations taken and used during running?
  - Check conditions DB status
- Attend and give report in daily run meetings

If agreed, pool of shifters will be collected (already have five volunteers)

Summary

SVT software is in good shape overall

- Tracking for day 1 in HPS works!
- Monitoring for SVT is on track.
  - Most tasks are defining and producing the exact final plots (easy)
  - Low-level monitoring for SVT DAQ still to be understood better (system is evolving)

There are some tasks that are more-or-less critical tasks that requires focus in the coming months

  - Would be good to have alignment and new geometry in good shape (at least for bootstrapping online updates for monitoring)
  - Straight through tracking
  - …

SVT performance analysis need more work

- Can we trigger on Möllers? How and how well can we extract our resolution/scale using Möllers and tridents?

Backup

SVT Software Status in JIRA

SLAC

Key	Summary	Assignee	Status	Due Date	Critical Date	On track
HPSJAVA-87	Procedure to time in the SVT	Omar Moreno	In Progress	11-Apr-14	Day 1	
HPSJAVA-69	Track-based SVT alignment	Per Hansson	In Progress	14-May-14	Day 1	
HPSJAVA-175	Integrate SVT alignment conditions into detector model	Per Hansson	Open		Day 1	
HPSJAVA-178	Add SVT alignment constants to conditions system	Per Hansson	Open	1-Jul-14	Day 1	
HPSJAVA-52	New SVT geometry based on survey	Per Hansson	In Progress	1-Aug-14	Day 1	
HPSJAVA-59	Vertexing in varying B-field	Norman Graf	In Progress	2-Oct-13	Day 1	
HPSJAVA-88	Change the SVT readout simulation and track reconstruction to make use of the conditions database	Omar Moreno	In Progress	9-May-14	Day 1	
HPSJAVA-68	SVT monitoring plots	Omar Moreno	In Progress	6-Jun-14	Day 1	
HPSJAVA-66	Online event display for SVT	Jeremy McCormick	Open	22-Jan-14	Day 1	
HPSJAVA-64	Use hit times in track fit	Sho Uemura	Open	30-May-14	Analysis	
HPSJAVA-61	Complete SVT hit time reconstruction	Sho Uemura	Open	16-May-14	Analysis	
HPSJAVA-76	GBL track fit implementation in java	Per Hansson	In Progress	29-Nov-13	Analysis	
HPSJAVA-55	Test 3D field map in SLIC	Norman Graf	In Progress	23-Aug-13	Analysis	
HPSJAVA-110	Iterative helix and plane intercept fails for small radius tracks.	Per Hansson	Open	1-Aug-14	1-Aug-14	
HPSJAVA-95	Add proper cov matrix to gbl track in cpp implementation	Per Hansson	Open	1-Aug-14	1-Aug-14	
HPSJAVA-1	Track finding and fitting based on single Si layers	Norman Graf	Open	22-Jan-14	Improvement	
HPSJAVA-6	Tracks are displayed incorrectly in Wired	Norman Graf	Open			

Monitoring of Calibrations

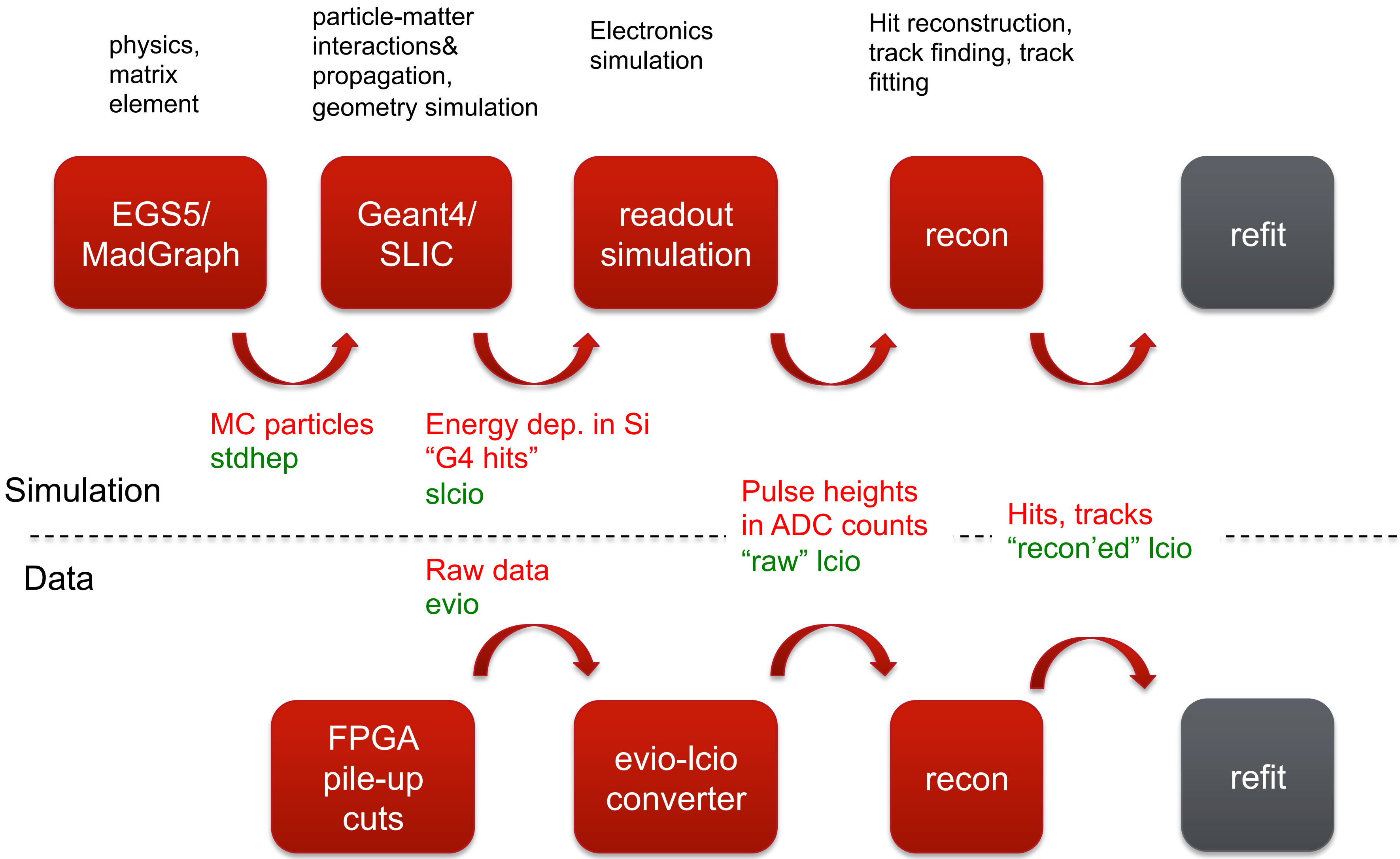
Observable	
Baseline Shifts	Large baseline shift will point to issues with power distribution or abnormal variations in temperature.
Noise shifts	Allows monitoring of dead channels. Large shifts in noise may also reveal problems with the DAQ.
Baseline and Noise Sample-to-Sample shifts	

SVT DAQ: Configuration, Calibration and Conditions

SVT is more tightly integrated with JLab DAQ (CODA)

- Configuration
  - SVT configured through link in the global configuration file
  - Relatively straightforward since SVT is configured through xml over TCP/IP anyway
  - Expect to keep SVT expert for stand-alone tests
- Calibration
  - Move calibrations from SVT GUI to CODA
  - CODA run types added for the various calibrations
  - Layer will be added in CODA to script the special settings needed (random triggers, quick reconfigurations for charge injection, etc.)
  - Calibrations from the SVT DAQ are well defined at this point
- Conditions
  - Baseline solution is to always store full HPS (including SVT) configuration in the data stream
  - Run DB may be available during running; otherwise we'll fill it after
  - Environmental variables not monitored by EPICS will be stored in data stream

HPS/SVT Reconstruction Software



Vertexing

2-track vertexing is based on the Billoir et al. method

- Billoir, Fruhwirth, Regler NIM A241, 1985
- Billoir and Qian NIM A311, 1992

Uses Kalman filter techniques and the perigee helix parameterization to calculate the vertex position and fitted track parameters

- Assumes no curvature near the vertex
- Iteration for long-lived decays?

Adding constraints is straightforward

- Implement a target/beamspot constraint for prompt ➡decays.

New features: add in functionality to fit a third (or more) track

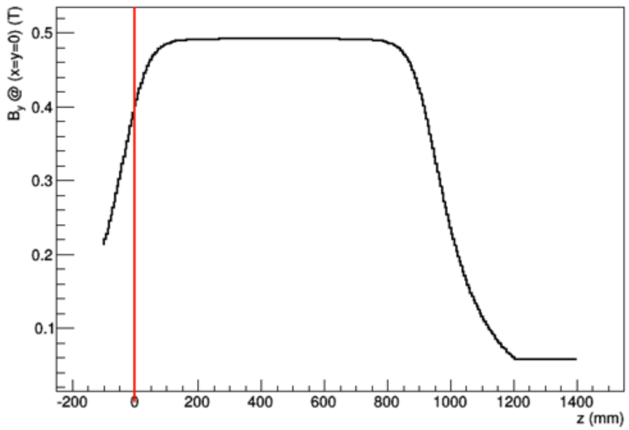
3D Magnetic Field

Primary use case is vertexing

- Target sits in fringe field
- At a minimum we need (By)@(x,z)

Already have this

Existing fringe field desc.



Existing 3D magnetic field support

- Input (Bx,By,Bz)@(x,y,z) on cartesian grid
- Linear interpolation between box of points
- Geometry code to handle field map exists
- Track propagation in inhomogeneous field with Runge-Kutta method

⇒ Need to be integrated and tested with vertexing software

⇒ Not clear we need <u>full</u> 3D map (needs testing)

Track Finding

Inherited from linear collider simulation (lcsim "seed tracker")

- Seed-confirm-extend philosophy
- Very fast: test often, reject early
- Based entirely on stereo hits

Track finding is governed using a "Strategy"

```
<Strategy name="HelicalTrackHit Strategy">
    <!--Cutoffs-->

    <MinPT>0.050</MinPT>
    <MinHits>4</MinHits>
    <MinConfirm>1</MinConfirm>

    <MaxDCA>80.0</MaxDCA>
    <MaxZ0>80.0</MaxZ0>

    <MaxChisq>25.0</MaxChisq>
    <BadHitChisq>10.0</BadHitChisq>

    <!--Layers-->
    <Layers>
        <Layer type="Seed" layer_number="1" detector_name="Tracker" be_flag="BARREL" />
        <Layer type="Seed" layer_number="3" detector_name="Tracker" be_flag="BARREL" />
        <Layer type="Seed" layer_number="5" detector_name="Tracker" be_flag="BARREL" />
        <Layer type="Confirm" layer_number="7" detector_name="Tracker" be_flag="BARREL" />
        <Layer type="Extend" layer_number="9" detector_name="Tracker" be_flag="BARREL" />
    </Layers>
</Strategy>
```

1. Fit a 3-hit track seed using stereo hits
2. Reject if failing strategy cuts
3. Add hits from confirm layers
4. Reject if failing strategy cuts
5. Add hit from extend layers, reject if worse chi2
6. Reject if failing chi2 and # hits

Remove overlapping tracks (shared hits <=1)

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

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

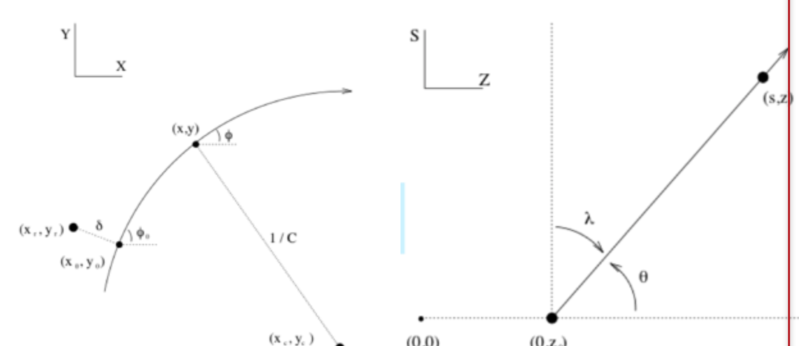
- Parameter estimations
- Covariance matrix
- (Seed)Track finding uses these algorithms at each step

⇒ Merge final fit into a “helix” track object together with the hits of the track

Parameterization and conventions inherited from lcsim

⇒ B-field in z-direction, beam in x

⇒ Rotation from natural coord. system



V. Blobel, C. Kleinwort, F. Meier, Computer Phys. Communications (2011)
Kleinwort, NIM A, 673 (2012), 107-110

Track-based Alignment

SLAC

Tracking detector alignment is a standard problem; multiple ways to achieve similar performance

Our approach:

- Do a least square fit of local (track) and global (alignment) parameters
- Millepede-II can do this for us and is "supported" by GBL
- Great support from C. Kleinwort (GBL/Millepede developer)

MPII

Residual z_i for measurement *i*:

5 track parameters

Subset Ω of *n* alignment constants

$$z_i = y_i - f(x_i, \mathbf{q}, \mathbf{p}) = \sum_{j=1}^{5} \left(\frac{\partial f}{\partial q_j} \right) \Delta q_j + \sum_{l=1}^{\Omega} \left(\frac{\partial f}{\partial p_l} \right) \Delta p_l$$

Minimize a "chi2" from entire data sample

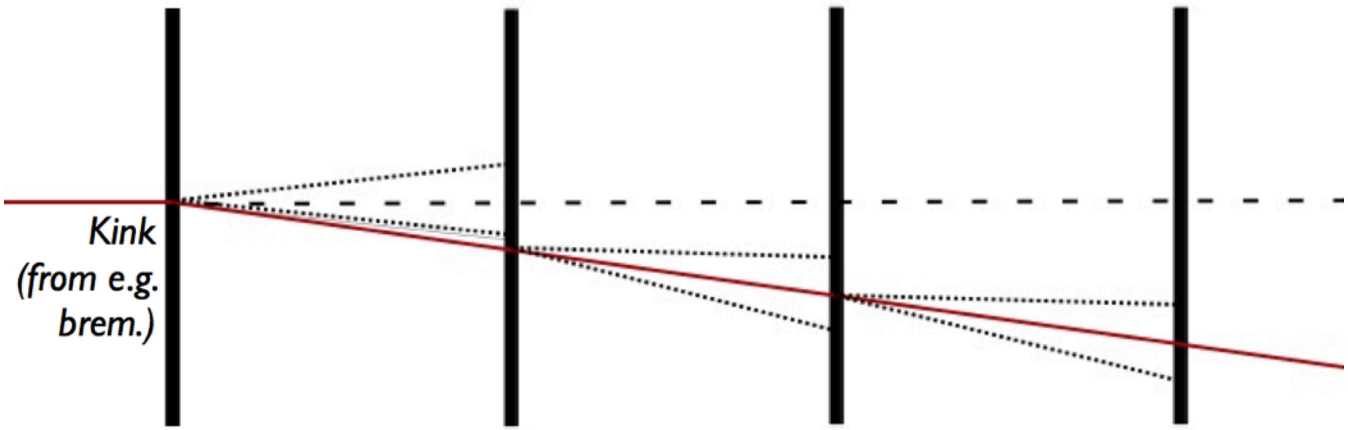
$$\chi^2 = F(\mathbf{q}, \mathbf{p}) = \sum_i \left(\frac{(y_i - f(x_i, \mathbf{q}_j, \mathbf{p}))^2}{\sigma_i^2} \right)$$

⇒ Newton minimization problem with large # parameters

⇒ single iteration for **linear** least squares

⇒ Millepede's strength is reducing the dimension of the matrix to be computed to give alignment parameters corrections only

Multiple Scattering Model



Hit uncertainty at each layer

- Multiple scattering (MS) uncertainty and spatial resolution added in quadrature
- MS uncertainty from each previous layer are added in quadrature
- No account for correlations across scattering planes or energy loss

MS uncertainty is on average correct but not an optimal fit

- Good enough for an initial fit

⇒ Different (standard) ways to deal with this problem

Generalized Broken Lines (GBL)

Generalized Broken Lines (GBL)

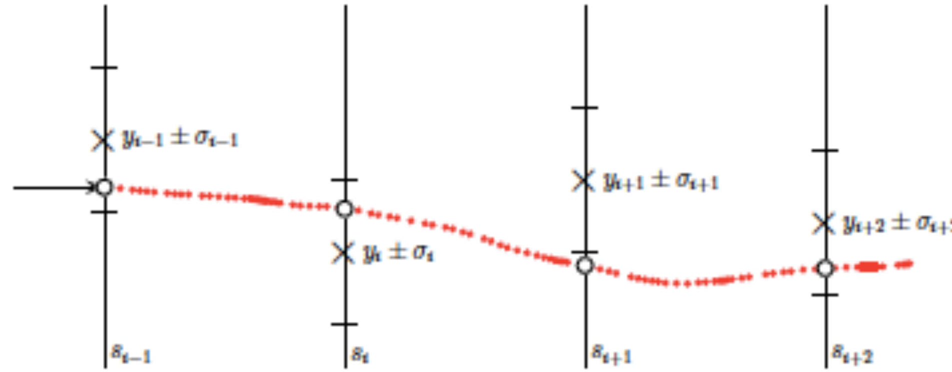
- A track fit with multiple scattering
- Widely used, e.g. CMS detector alignment

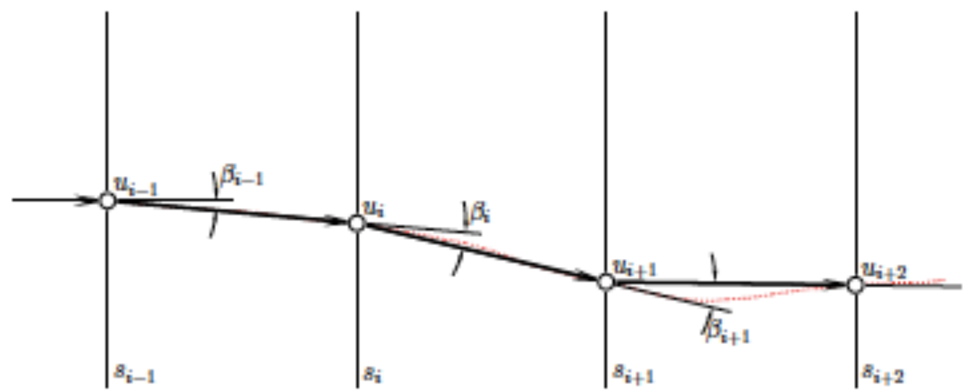
GBL is a track refit

- Initial fit to estimate residuals and momentum (using SeedTracker)
- Use residuals and estimated momentum, in a second fit that includes multiple scattering
- Covariance matrix of all track parameters are available (at each point)

Iteration needed for energy loss

⇒ Alignment software (Millepede-II) “supported”



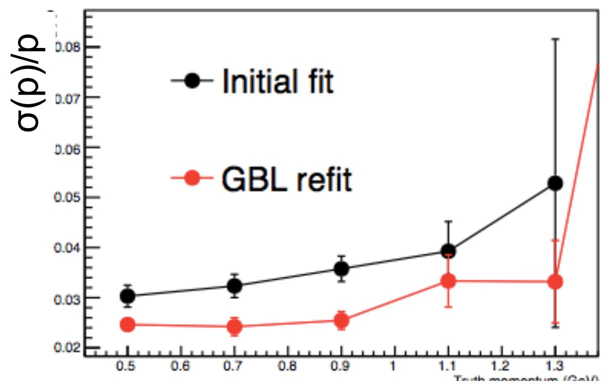


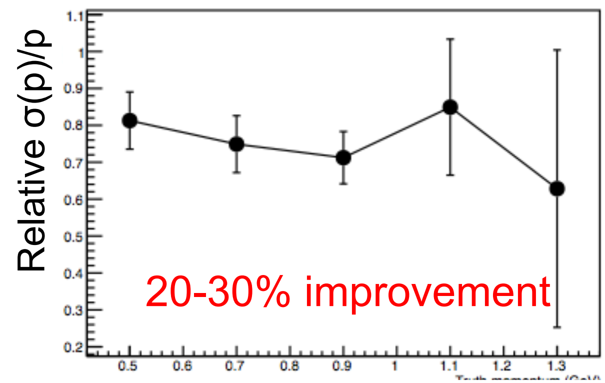
$$S(u) = \sum_{i=1}^{n} \frac{(y_i - u_i)^2}{\sigma_i^2} + \sum_{i=2}^{n-1} \frac{\beta_i^2}{\sigma_{\beta,i}^2}$$

GBL Already Implemented

Momentum resolution

A' (40MeV) events

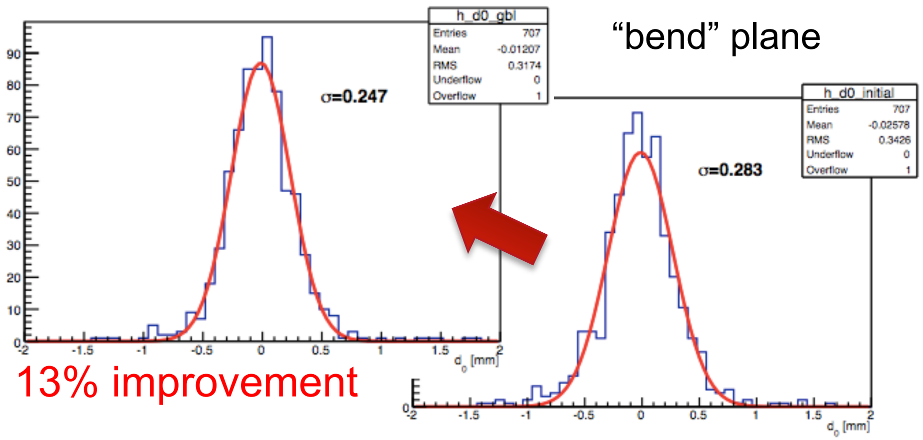




Track momentum (GeV) Track momentum (GeV)

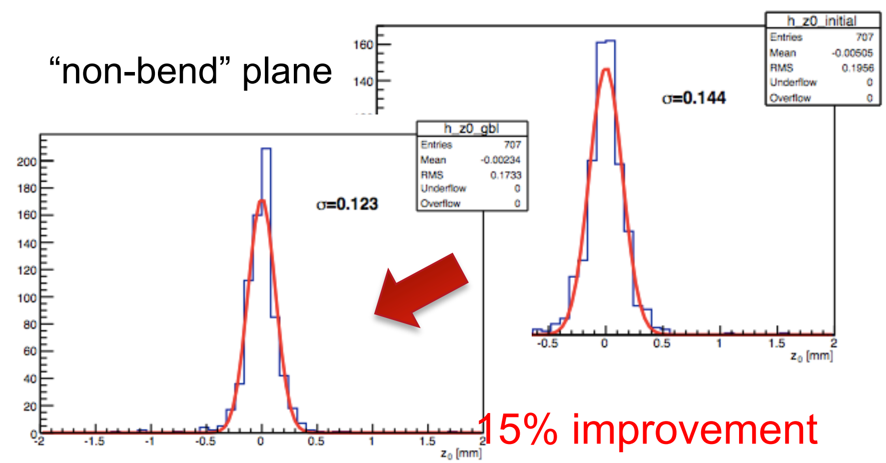
Impact parameter resolution

"bend" plane



13% improvement

"non-bend" plane



15% improvement

Currently implemented in python (used here) and C++
⇒ would like to port to Java, but not critical