DESY Tracking Update

Paul Dauncey

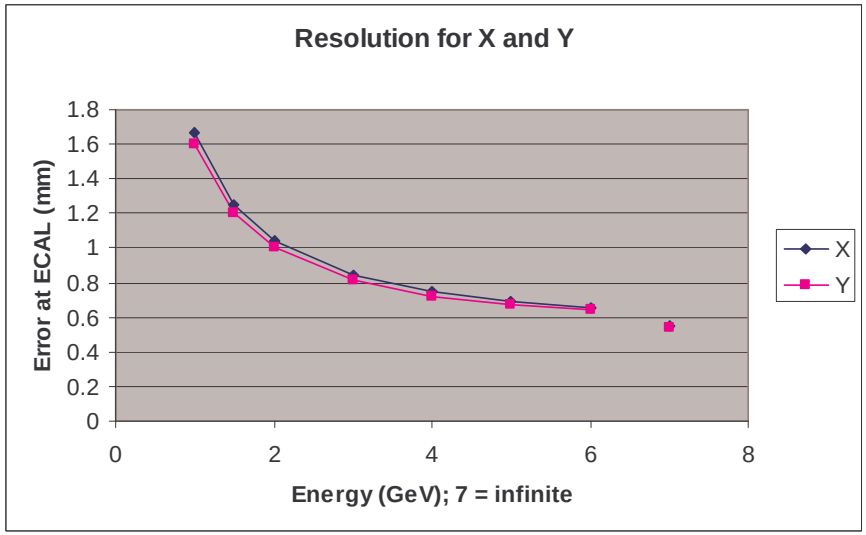
- Track resolutions
- Alignment in run 230101
- Software structure

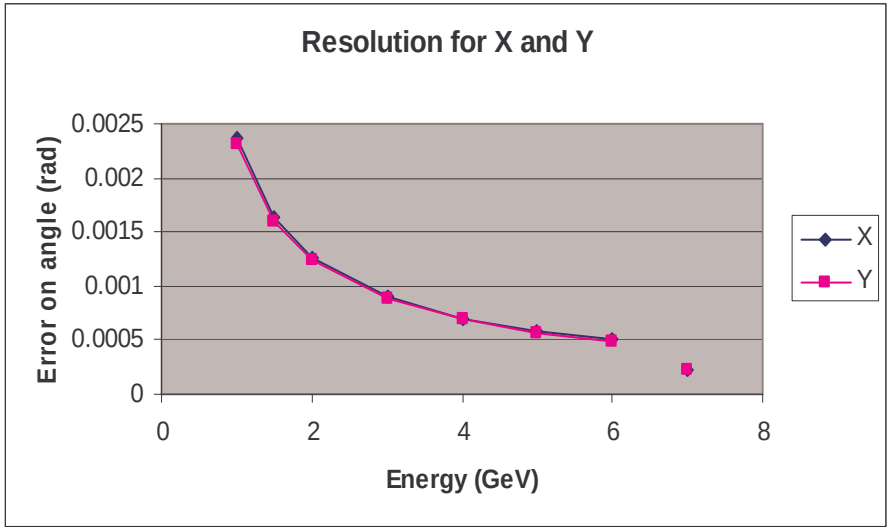
Track resolutions at DESY

- Want to see how the track resolution depends on various parameters
- Use MC to determine errors from scattering for all seven DESY energies
- Add in intrinsic resolution (assumed to be 0.5mm) to get total error matrix
- Propagate errors to get track parameter error matrix and extrapolate matrix to ECAL front face
- N.B. No actual fits; this is purely propagation of errors

Errors in x and y

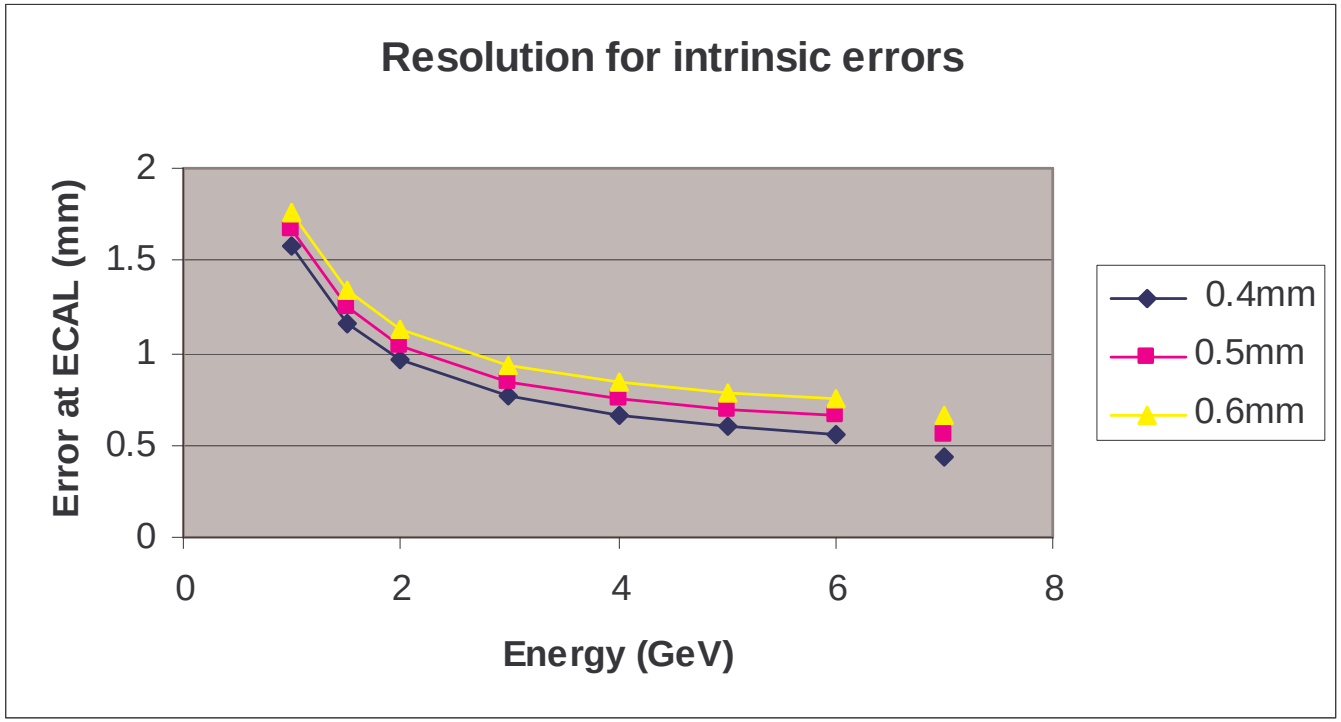
- Position and angle resolution at ECAL





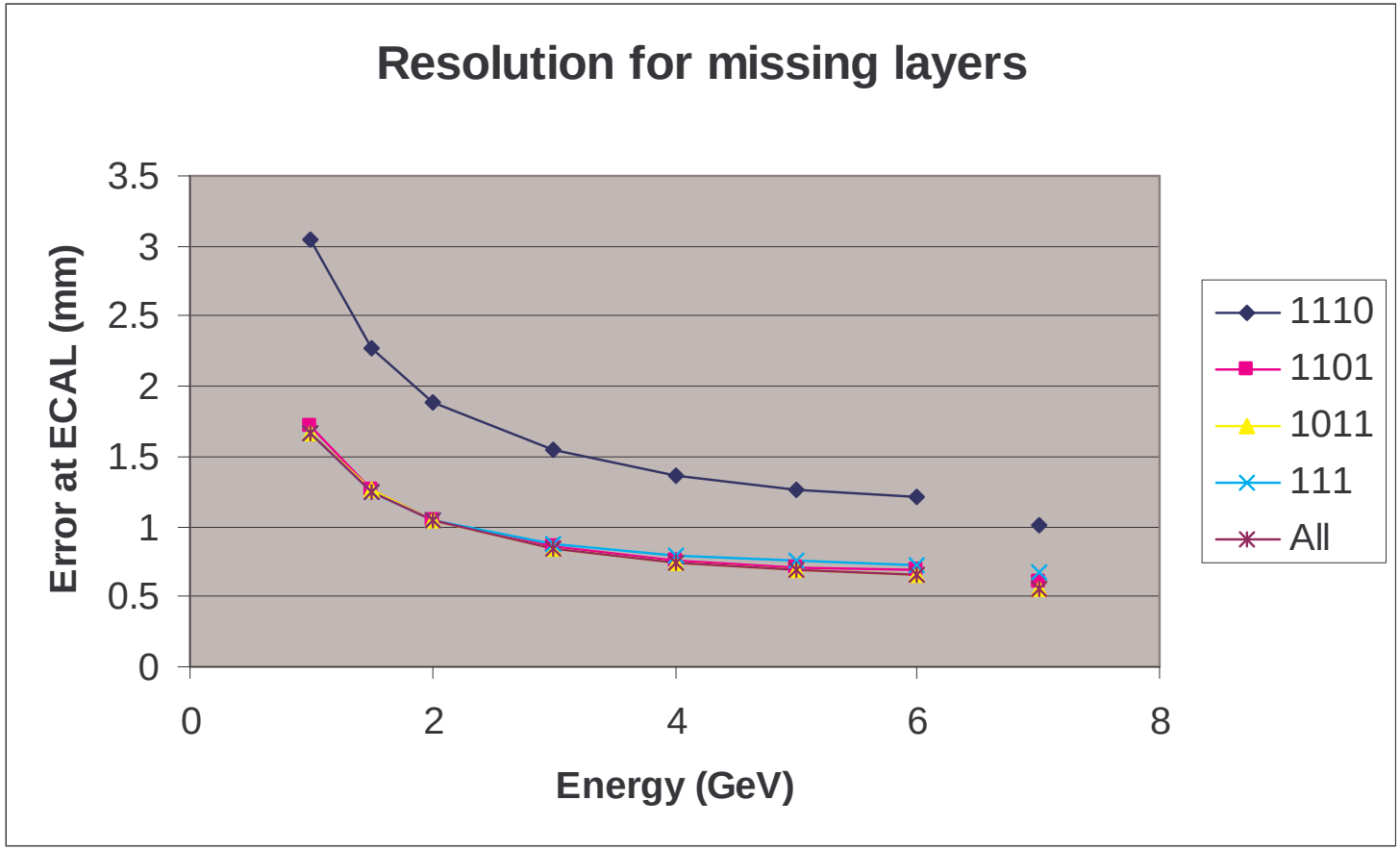
- Shift in z of tracking layers is only difference
- Position error difference ~5%
  - Negligible or use different error matrices for x and y?
- Only show x values from now on

Dependence on intrinsic resolution



- Highest energies depend strongly on intrinsic resolution
  - Dominates over scattering at these energies
- We need to determine the intrinsic resolution well to understand the track errors

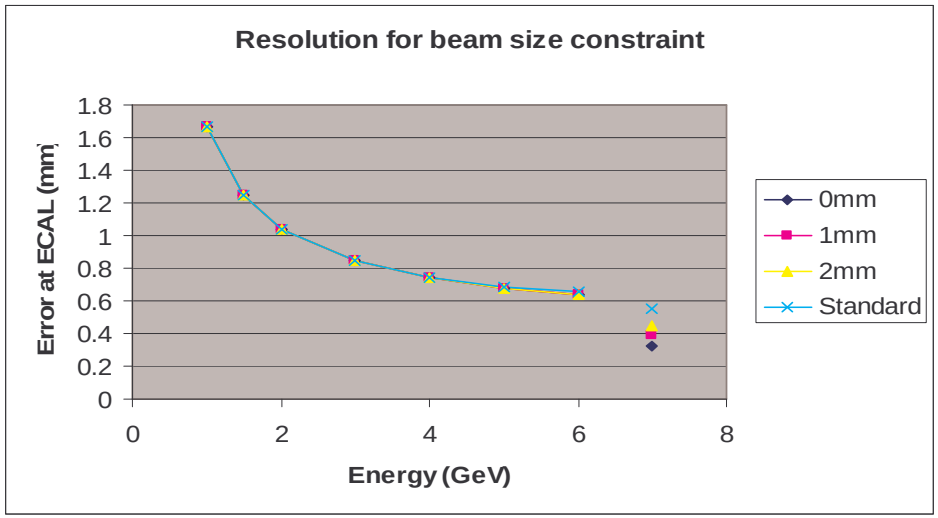
Dependence on missing layers

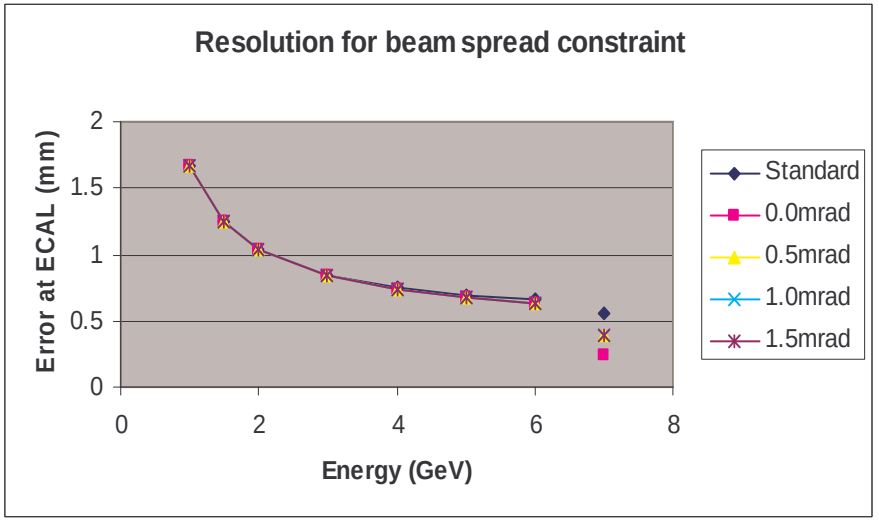


- Only small differences unless layer 0 hit is missing
  - Only allow three-hit tracks with layer 0 included?

Beam constrained fits

- Beam size and angular spread give two more constraints
  - Impact on fit depends on the width of these distributions





- Very little improvement with hits in all four layers
  - Should (?) give significant constraint if doing two-hit tracks
- At high energies it makes a significant improvement
  - Should the fit/software allow for this (for CERN)?

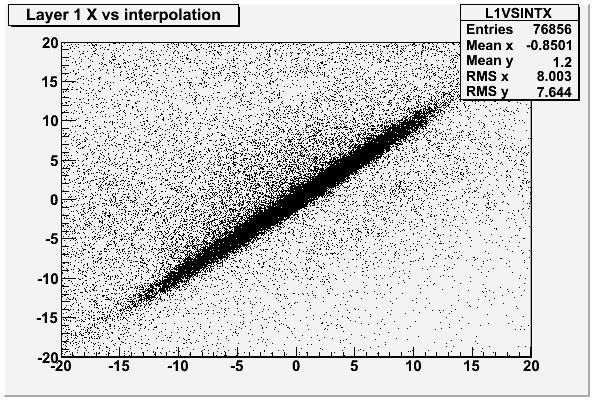
Resolution conclusions

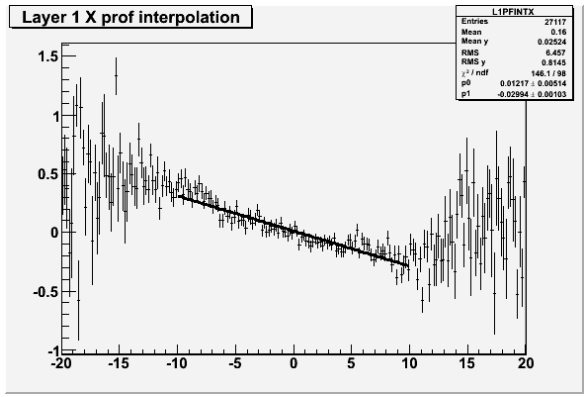
- Track resolution down to ~0.6mm at the ECAL should be achievable
  - This requires systematics to be significantly below this value
- Target: get systematics below 0.2mm (extra 10% on total error)
  - This requires alignment to be known to 0.2mm level
  - For beam spread of ~10mm, target corresponds to 2%
  - Drift velocity must be known to 2% level
- We must also know the error accurately
  - Need to measure intrinsic error well
- Outstanding questions
  - Do we use different scattering matrices for x and y?
  - Do we code for possibility of beam constraint later?
  - Do we need different scattering matrices for e and pi (at CERN)?

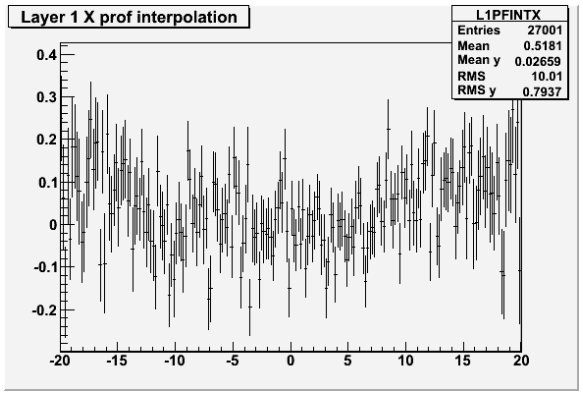
Alignment for run 230101

- Don’t have enough external “rulers” to fix drift velocity
  - Would need two external positions, or position and direction
  - The ECAL only measures position and not direction (accurately enough)
  - The beam spot is only measurable by tracking (so circular argument)
- Try assuming layer 0 and 3 have the same drift velocity
  - This is not correct exactly but gives overall distortion to system which we are not sensitive to
  - Start assuming drift velocity of 0.03mm/ns in each
- Interpolate inwards to determine drift velocity of layers 1 and 2
  - Compare to very simple straight line between the hits in layers 0 and 3
  - Effectively gives relative drift velocities
- Fit all four layers and extrapolate to ECAL
  - Use as ruler to get overall scale; adjust all four drift velocities together to agree

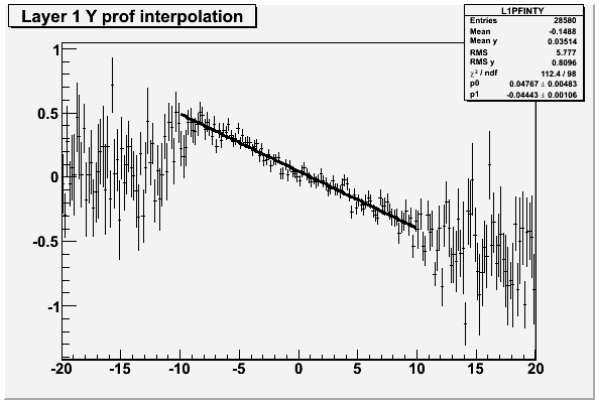
Layer 1 and 2 alignment

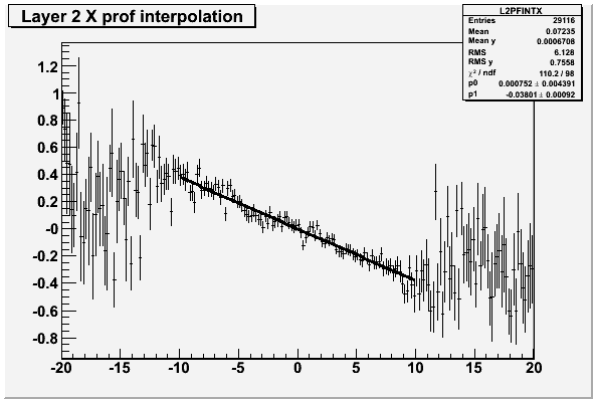


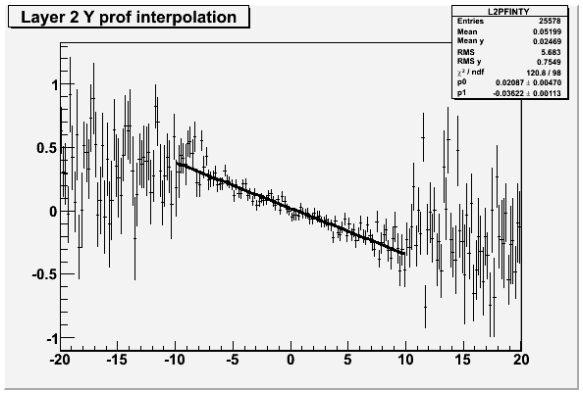




- Correct for slope (and intercept but small); still not perfect, but…

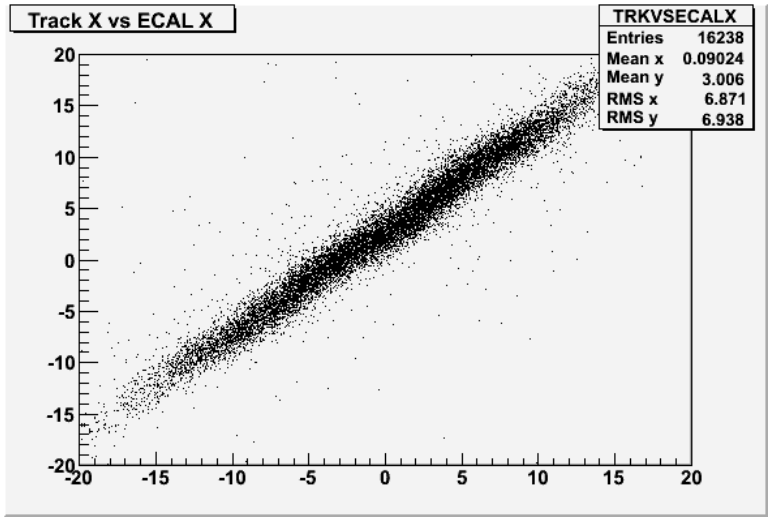


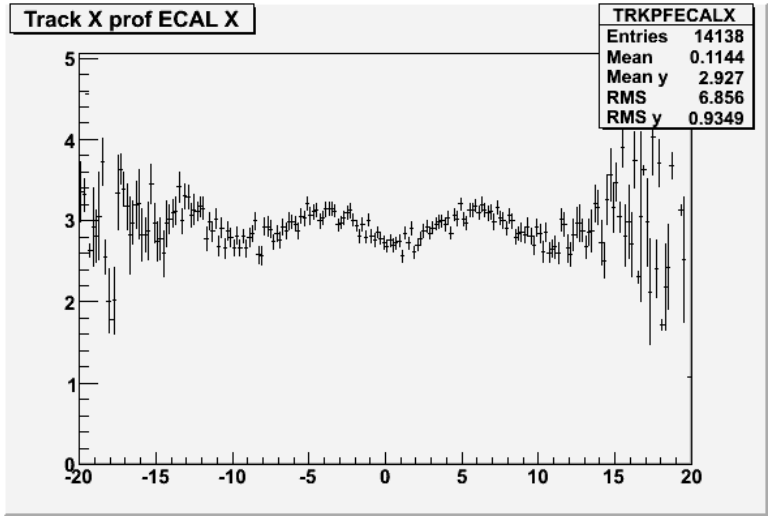


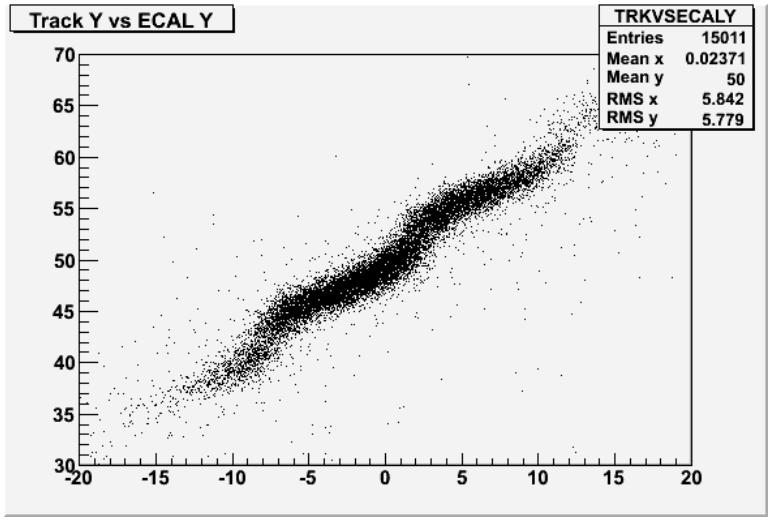


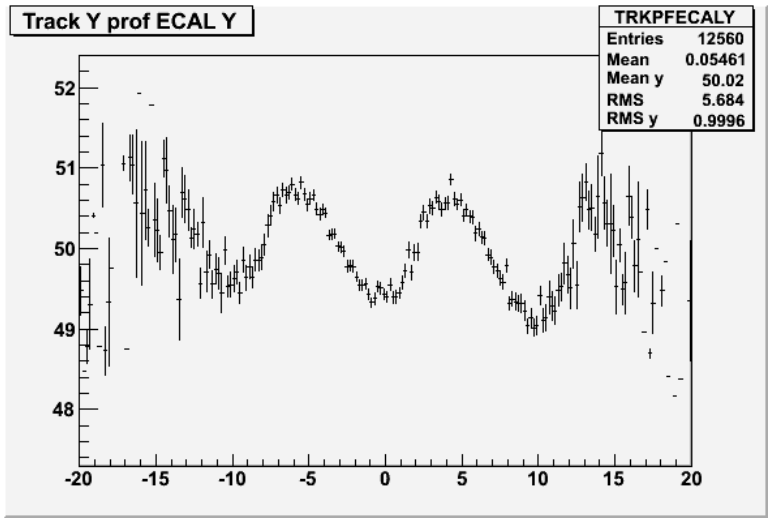
- All four cases show same (negative) direction of slope; bias?

Adjustment to ECAL





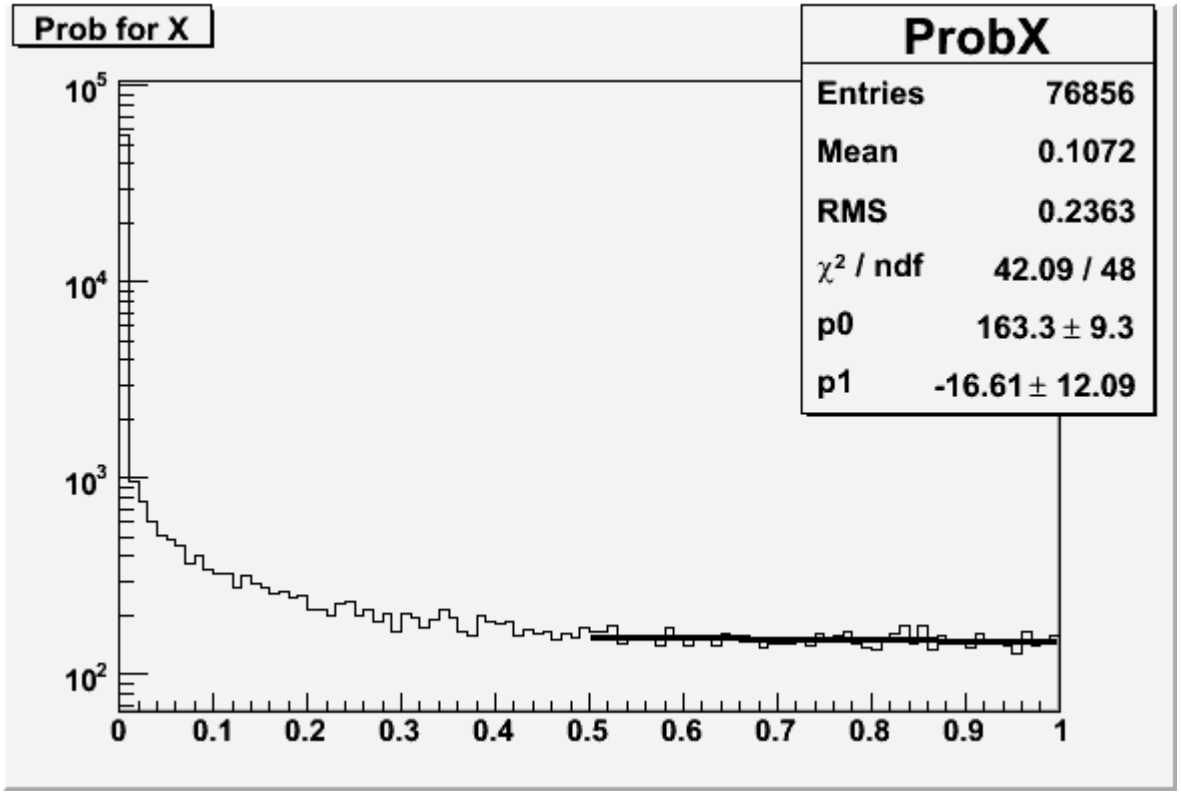




- Correct only for slope; intercept is ECAL alignment = 2.9,50.0

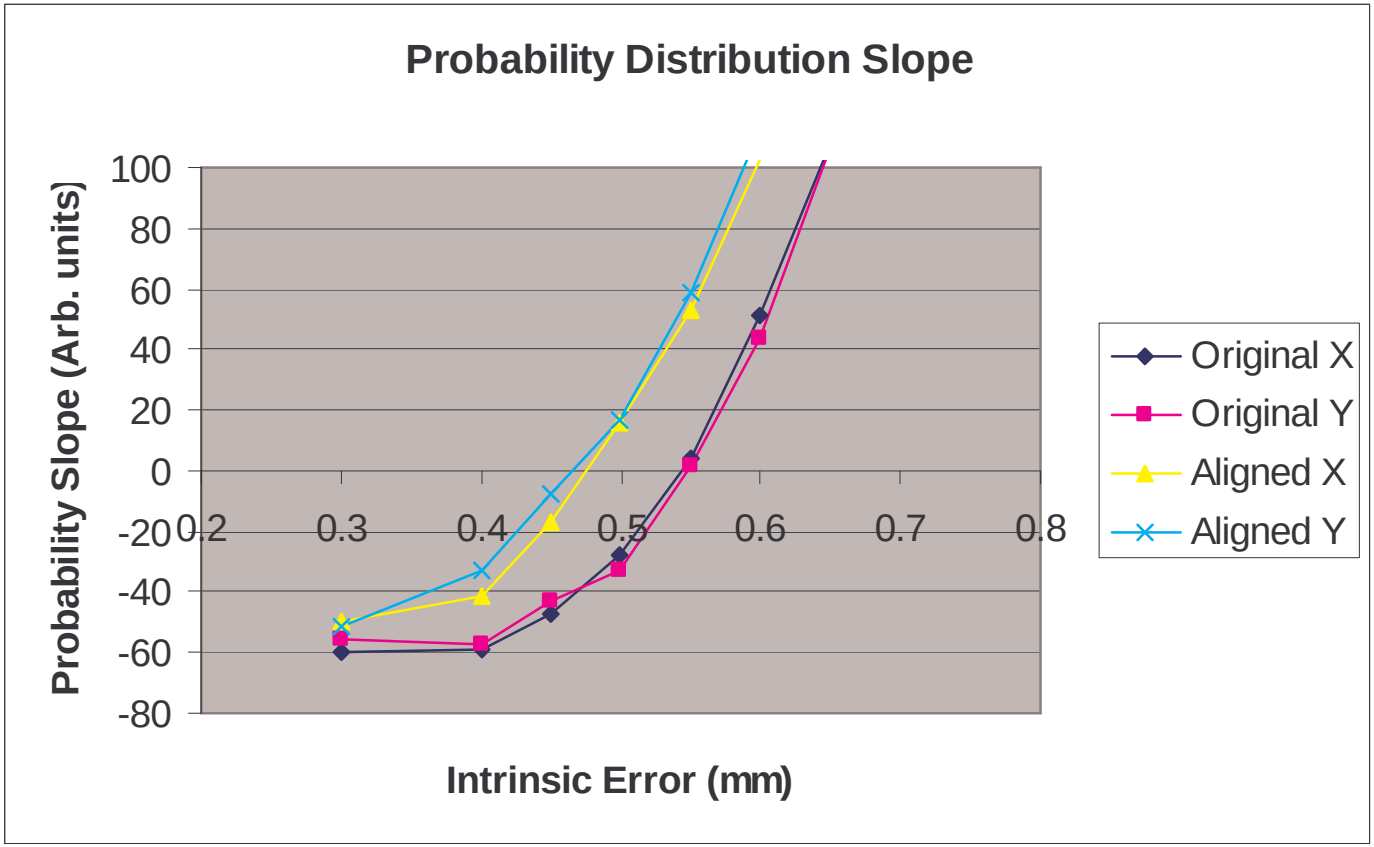
Intrinsic resolution

- Need to know intrinsic resolution
  - Try determining from track fit χ^2 probability being flat
  - Fit probability distribution in 0.5-1.0 range to avoid bad hit combinations
  - Look for slope being zero



Intrinsic resolution (cont)

- Try different intrinsic resolutions in fit and compare slopes
  - Both before and after realignment

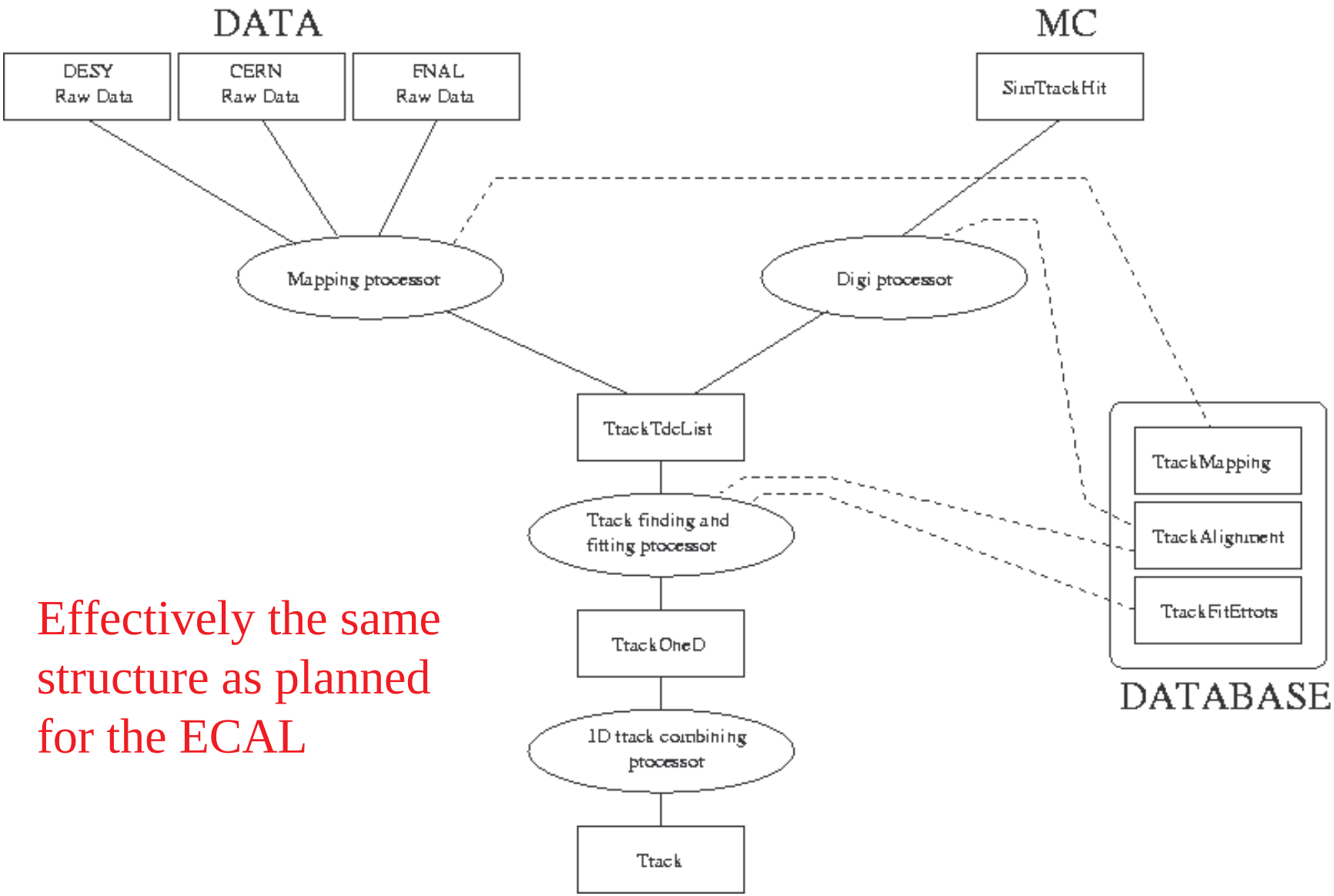


- Value for zero slope approx 0.55mm before, 0.45mm after

Alignment conclusions

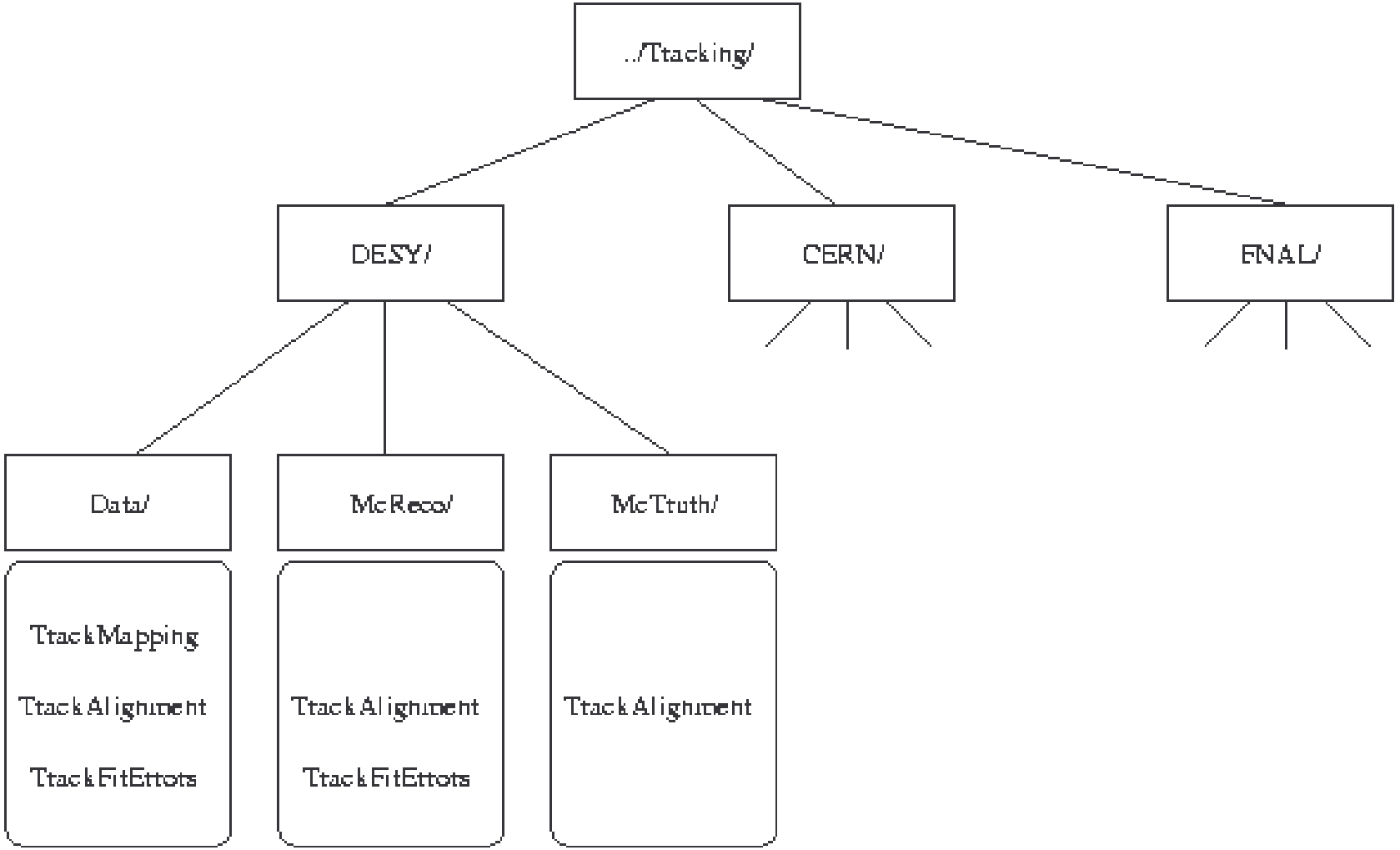
- Gains to be made from internal alignment
  - But there are clearly some remaining systematics
  - Need to cross check; e.g. fitting tracks by removing each layer in turn
- The intrinsic resolution can be got down to ~0.45mm
  - How much more it can be reduced is not clear
- How stable the alignment is is not clear either
  - Tuned on this run so not surprising the error improved
  - Could vary run-to-run; doing each run by hand is long-winded
  - Can this be automated?

Proposed software structure



Effectively the same structure as planned for the ECAL

Proposed database structure



Processors

- Mapping: converts data structures to common format
  - Run for real data only (no mapping in MC)
  - Output in layer/dimension labelling; needs TDC-to-physical channel map from database
  - Would also strip out obvious out-of-time hits
  - Combine CERN +/– values to give single int per dimension
- Digi: converts truth hit in MC to TDC hit
  - Run for MC only
  - Needs alignment, drift velocity, z positions, intrinsic resolution and efficiency per chamber
  - Get in single TrackAlignment object from database; McTruth version used here

Processors (cont)

- Track finding/fitting: forms 1D tracks from TDC hits
  - Run for real data and MC
  - Needs alignment and correct error matrices for beam energy
  - TrackAlignment and TrackFitErrors from database; use Data or McReco versions (these may be equal for the error case)
- 2D Track finding: combines 1D tracks to 2D tracks
  - Run for real data and MC
  - No database access needed
  - Must use LCIO standard Track output with normal values of parameters so event display works
- Most processors need to retrieve values from database
  - Not trivial; Anne-Marie meeting Roman today to iron this out
  - Hopefully working soon; common solution to all cases

Data storage classes

- Need five new data storage classes based on LCGeneric
- Database data:
  - TrackMapping
  - TrackAlignment
  - TrackFitErrors
- Event data:
  - TrackTdcList
  - TrackOneD
- Preliminary versions of TrackAlignment, TrackFitErrors and TrackOneD exist

Priorities

- Mapping: Currently hardcoded in fitting processor
  - OK until Japanese SCECAL data need processing (DESY but use different TDC mapping)
  - Not urgent
- Digi: Currently goes to separate raw data formats and alignment values passed by steering file
  - Can convert to using local TrackAlignment object now
  - Database interface then added later
  - How big a job to change output to common format?
  - Not essential yet but should be done soon

Priorities (cont)

- Track finding/fitting: Currently uses separate data formats and alignment values passed by steering file
  - Critical to convert to using TrackAlignment for constants
  - Critical to convert to using TrackFitErrors for fit matrix
  - Critical to use TrackOneD as output format
  - How big a job to change input to common format?
  - This is all necessary for full reprocessing
- 2D track finding: Does not exist yet
  - No database access
  - Input (effectively) and output (definitely) defined
  - Could be done completely now
  - Not so critical since TrackOneD output may be sufficient

Software conclusions

- Track finding/fitting processor and database access are the most critical parts
  - Must partition the processor work now
  - Must complete and test both parts within two weeks (max)
- Other parts can be done later
  - Changing to using common format couples processors
  - Needs to be coordinated; when to do this?
  - Before or after LCWS deadline is main question for today