DESY Tracking Update

Paul Dauncey, Michele Faucci Giannelli,
Mike Green, Anne-Marie Magnan,
Fabrizio Salvatore, Hakan Yilmaz

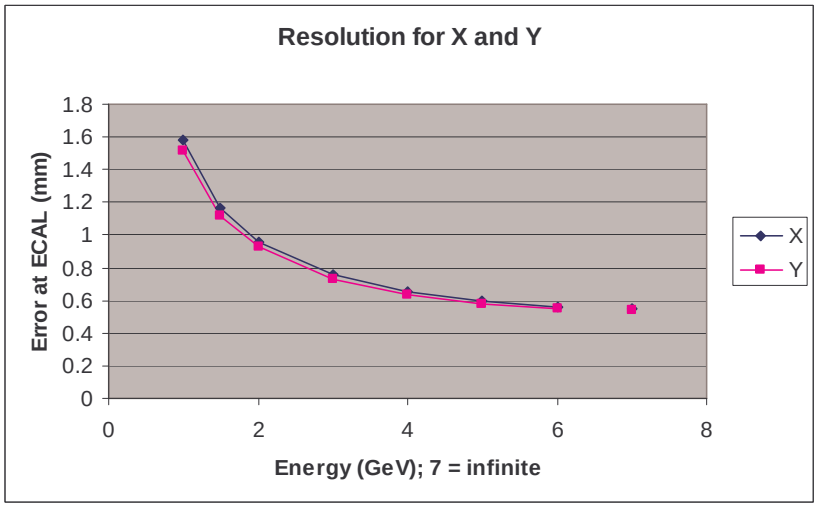
- Track resolutions
- Alignment in run 230101
- Software structure

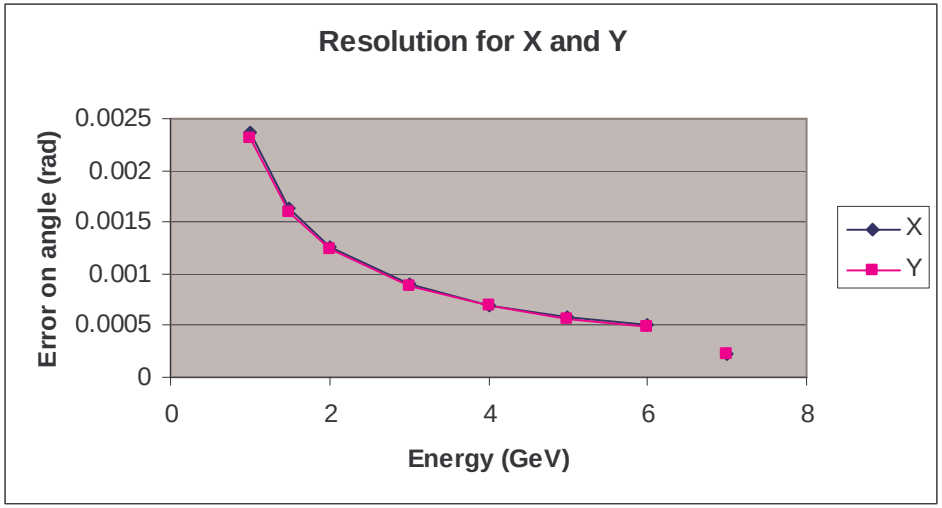
Track resolutions at DESY

- 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 0.4mm) 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

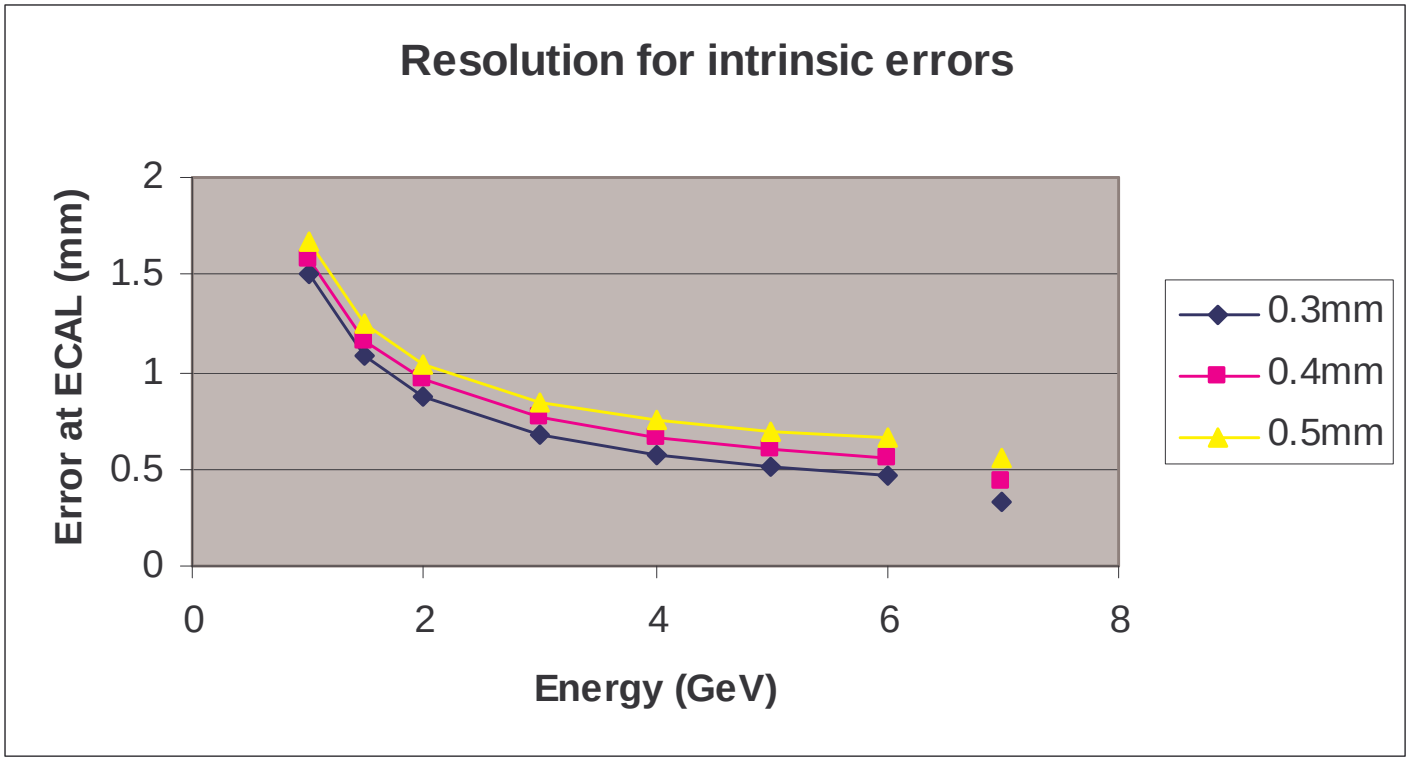
- Position and angle resolution at ECAL





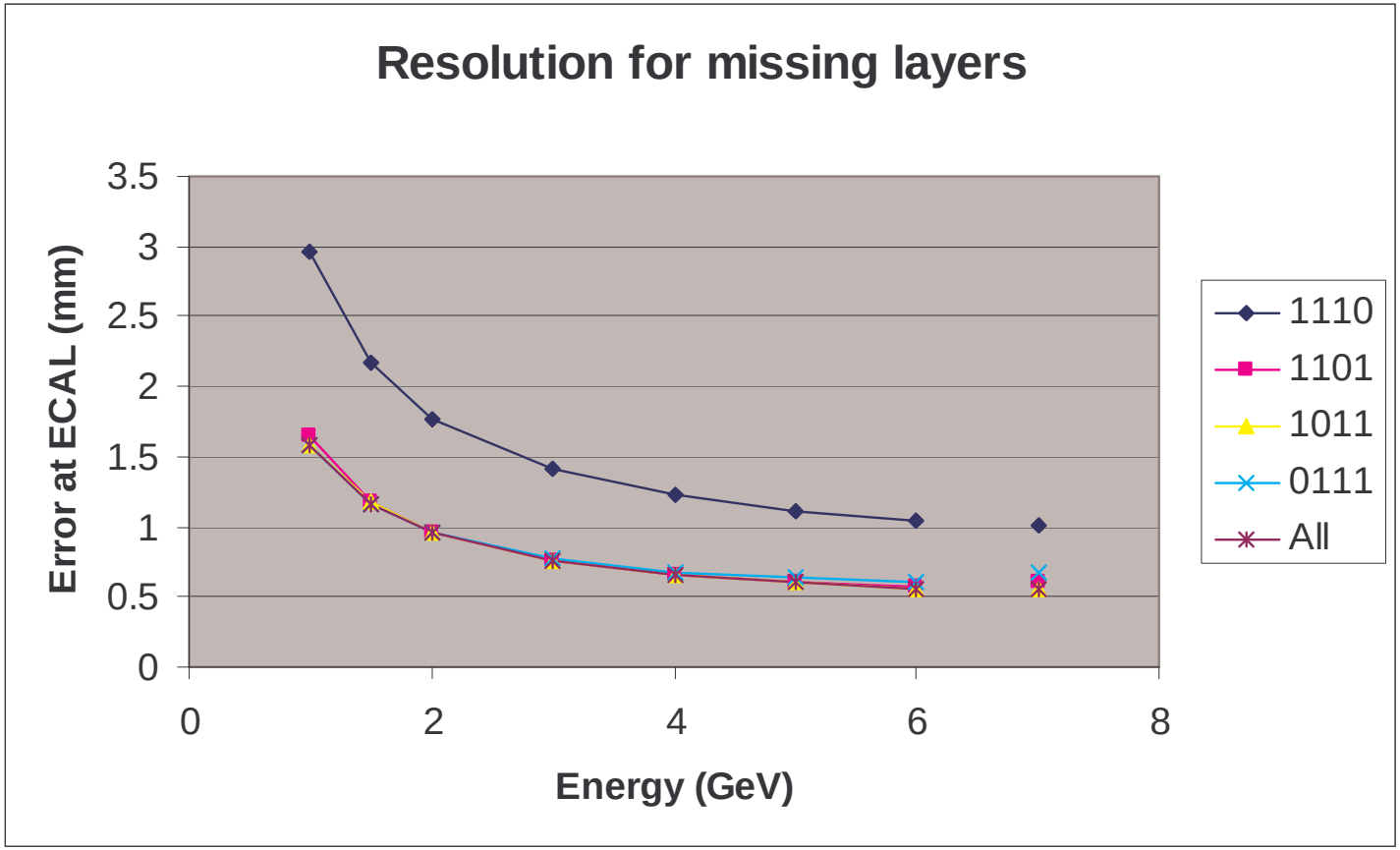
- Shift in z of tracking layers is only difference
- Position error difference ~5%
  - Negligible or use different error matrices for x and y?
- Only show x values from now on

Dependence on intrinsic resolution



- Highest energies depend most strongly on intrinsic resolution
  - Approximately equal to effect of 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?

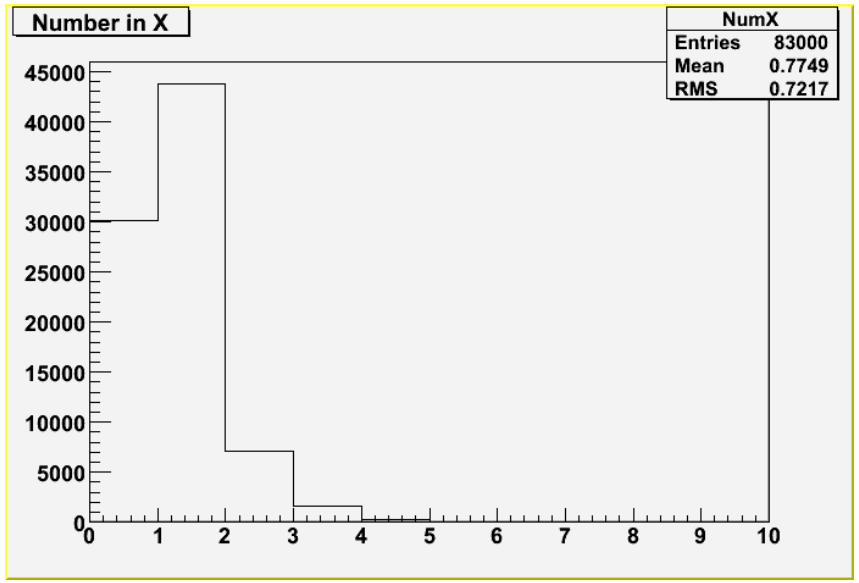
Resolution conclusions

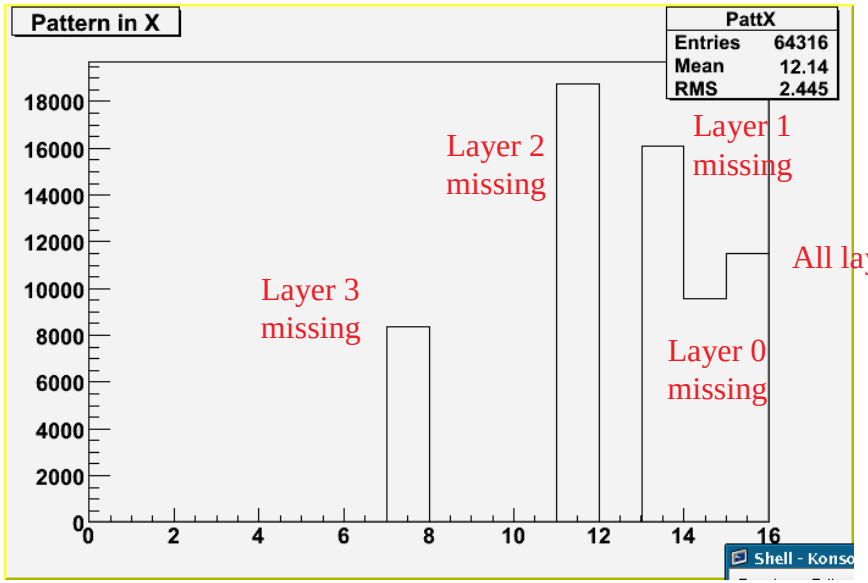
- Track resolution down to ~0.6mm at the ECAL should be achievable
  - This requires systematics to be significantly below this value
- Target: get systematics below 0.2mm (extra 10% on total error)
  - This requires alignment to be known to 0.2mm level
  - For beam spread of ~10mm, target corresponds 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 code for different scattering matrices for e and pi (at CERN) later?

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 constants of layers 1 and 2
  - Use full fit and shift offsets and drift velocities to get best probability values
  - Effectively gives relative drift velocities to average of layer 0 and 3
- Fit all four layers and extrapolate to ECAL
  - Use as ruler to get overall scale; scale all four drift velocities together to agree

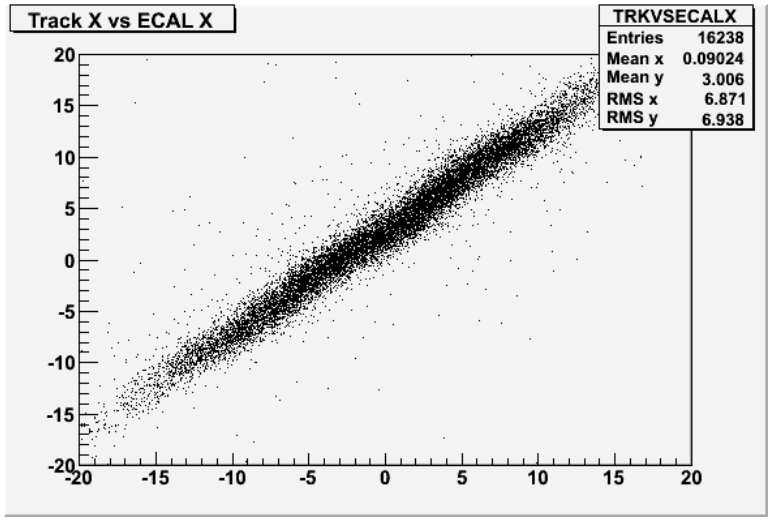
Results after alignment

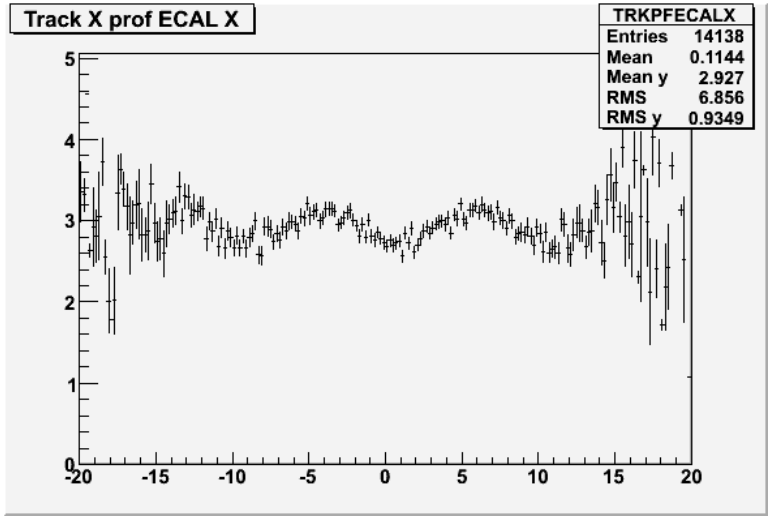


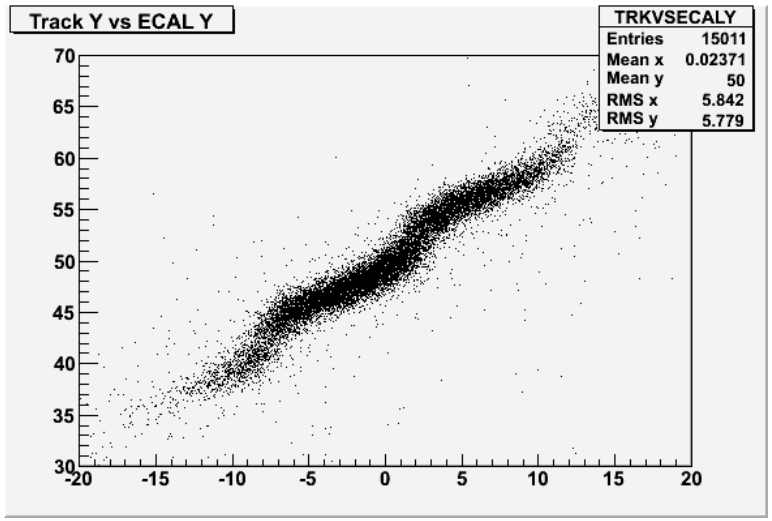


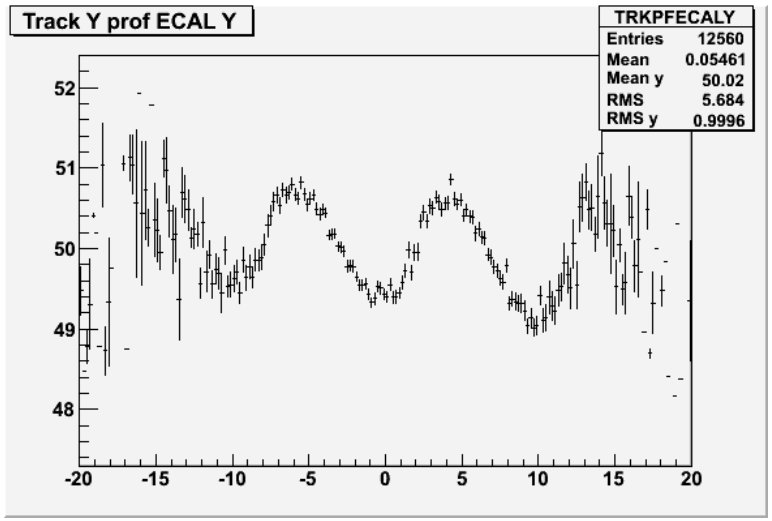
- ~40% of events have no track
  - How much is due to noisy beam conditions at DESY?
  - Need to compare with ECAL energy next
- ~20% of tracks have four hits
  - Naïvely taking ε^4 means effective layer efficiency is ~70%

Adjustment to ECAL





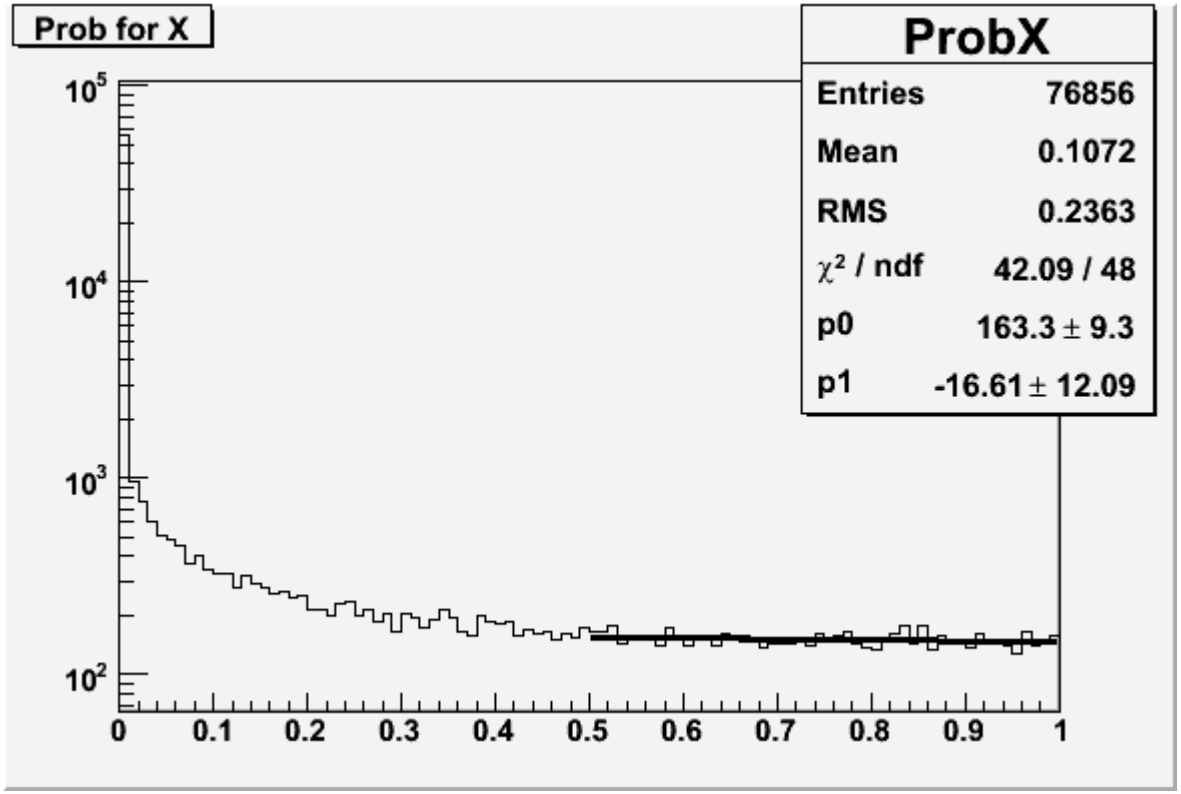




- Correct only for slope; intercept is ECAL alignment = 2.9,50.0

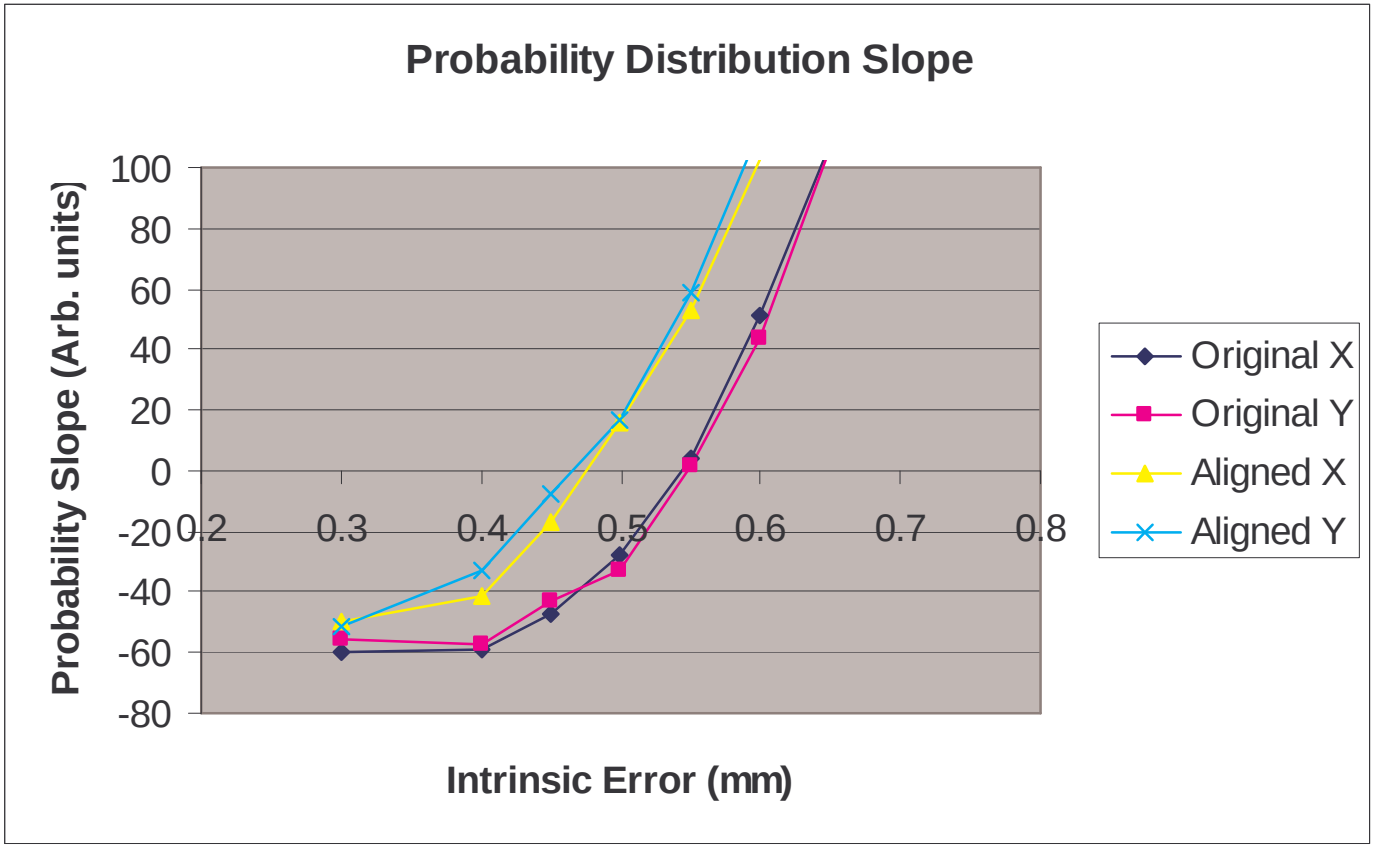
Intrinsic resolution

- 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

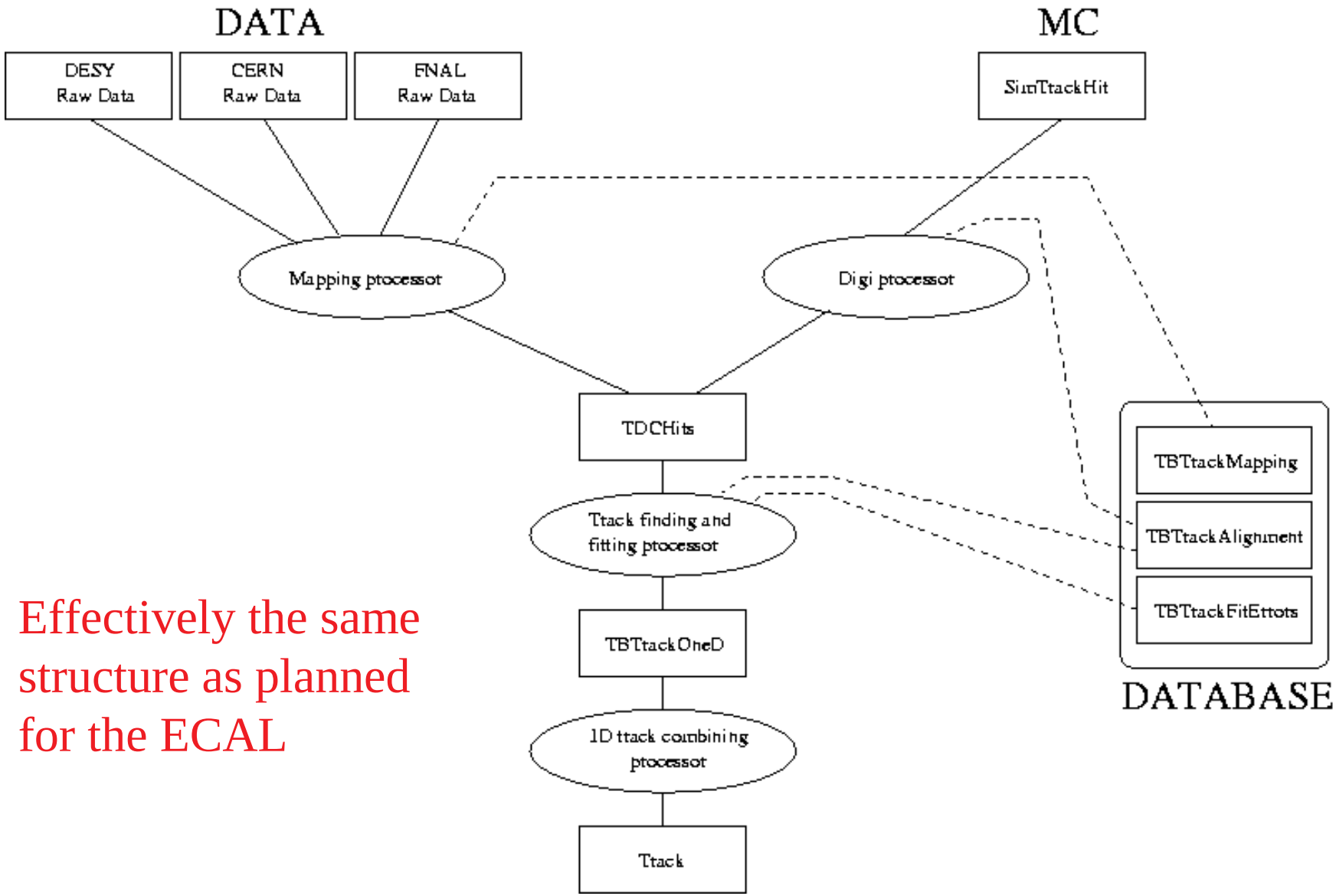


- Value for zero slope approx 0.55mm before, 0.45mm after

Alignment conclusions

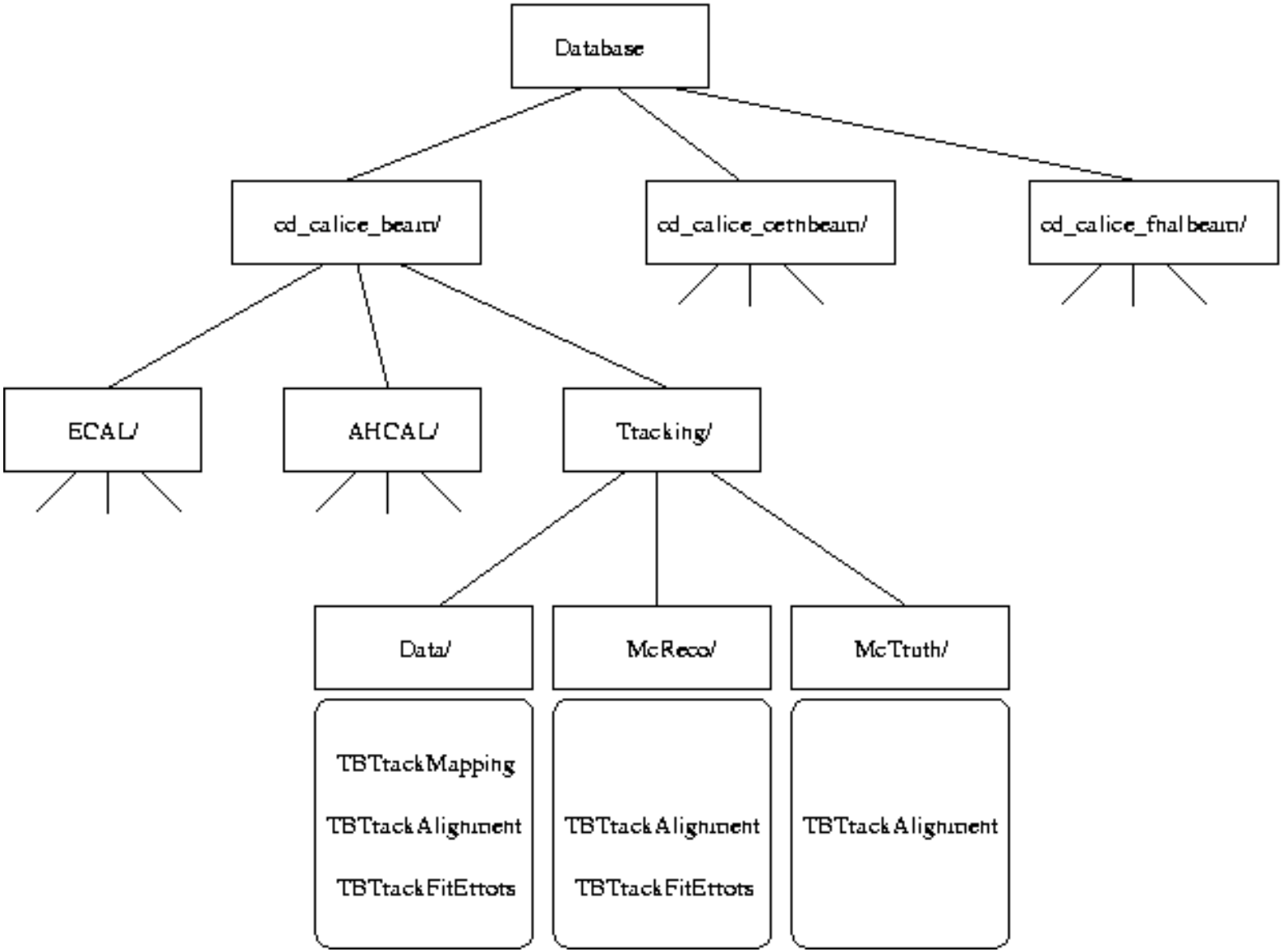
- Gains to be made from internal alignment
  - But there are clearly some remaining systematics
  - Need to cross check; e.g. fitting tracks by removing each layer in turn
- The intrinsic resolution can be got down to ~0.45mm
  - How much more it can be reduced is not clear
- How stable the alignment is is not clear either
  - Tuned on same 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 labeling; 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, intrinsic resolution and efficiency per chamber
  - Get in single TBTrackAlignment 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
  - TBTrackAlignment and TBTrackFitErrors 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
  - Could use LCIO standard Track output with normal values of parameters so event display works; unclear if helical parameterisation is useful
  - Alternative is to use another CALICE-specific class

Data storage classes

- Need five new data storage classes based on LCGeneric
- Database data:
  - TBTrackMapping
  - TBTrackAlignment
  - TBTrackFitErrors
- Event data:
  - TDCHits
  - TBTrackOneD
- Preliminary versions of TBTrackAlignment, TBTrackFitErrors, TBTrackTdcList and TBTrackOneD exist

Priorities

- Mapping: Currently hardcoded in fitting processor
  - OK until Japanese SCECAL data need processing (DESY but use different TDC mapping)
  - Need to convert to common TDCHit format
  - Common format needed but database mapping not urgent (yet)
- Digi: Previously went to separate raw data formats and alignment values passed by steering file
  - Converted to using TBTrackAlignment object
  - Filled from database or steering file
  - Currently implementing common format
  - Almost complete (but needs debugging)

Priorities (cont)

- Track finding/fitting: Currently uses separate data formats and alignment values passed by steering file
  - Need to convert to using TDCHits as input
  - Need to convert to using TBTrackOneD as output format
  - Need to convert to using TBTrackAlignment for constants
  - Need to convert to using TBTrackFitErrors for fit matrix
  - Need these constants to come from the database
  - This is all necessary for full reprocessing; pieces exist
- 2D track finding: Does not exist yet
  - No database access; could be done as part of analysis
  - Not so critical since TBTrackOneD output may be sufficient

Software conclusions

- Track finding/fitting processor and database access are the most critical parts
  - Working on these processors now
  - Must complete and test both parts within two weeks (max)
- Outstanding questions
  - Do we want to have TBTrackOneD store locations of the calorimeters for easy projections of the track?
  - Do we want to use the LCIO standard Track or a new class?