Experience Using TRF

Rob Kutschke, Fermilab
ALCPG07, Simulation Session
October 26, 2007

Introduction

- This talk is about Kalman filter code for fitting charged tracks in the SiD tracking system.
  - Scope of my project is limited to use as a final fitter.
  - The code lives within the org.lcsim framework and seeks to maximize code that already exists within that framework.
    - Java based.
    - Uses the TRF toolkit.

What is TRF?

- As I use it, TRF is a toolkit for:
  - Describing material and hits along a trajectory.
  - Propagating from one surface to the next.
  - A model of multiple scattering.
  - A model of energy loss.
  - Kalman filtering tracks that are described by the above components.
- Author: Norman Graf
- Norman tells me that it can also be used for pattern recognition but I have not looked into that.

My Goal

- Create a package to:
  - Take a container of hits from an external pattern recognition routine.
  - Load them into TRF.
  - Discover additional material along the track
    - Beampipe, support structures, cables ...
  - Fit the track.
  - Return track parameters at several locations.
    - At PCA to z-axis.
    - At innermost and outermost hits.
    - Others?
    - See: Evolving the Track Class in the ILC Forum.
  - End users should not need to know about TRF internals.

How to Get There

- Internal Tests:
  - Generate tracks/hits internally.
  - Learn how to use all of the tools.
- Hits from SLIC/org.lcsim using perfect pattern recognition:
  - From charged geantinos (no Ms or Eloss).
    - Just a bookkeeping job.
  - From real particles.
    - Discover beampipe and other dead materials.
    - Tune material interaction models to match G4.
    - These two steps may be time consuming.
- At each step, understand where tails come from.
- Ready for hits from real pattern recognition codes.

TRF Basics

- TRF has its own geometry and hit classes, independent of those of org.lcsim.
- TRF has its own choices for track parameters, different from those of org.lcsim.
- Using TRF:
  - Translate inputs from org.lcsim objects
  - Run the fitter.
  - Translate outputs to org.lcsim objects.
- API browsers
  - org.lcsim, includes TRF
    - http://lcsim.org/software/lcsim/apidocs/index.html
  - GeomConverter:
    - http://lcsim.org/software/geomconverter/apidocs/index.html

TRF Surfaces

- interface Surface ....
- Classes that implement Surface:
  - SurfDCA
    - Defines a reference line parallel to the z axis and through a given (x,y) point.
  - SurfCylinder
    - Defines a cylinder, of given r, centered on z-axis
  - SurfZPlane
    - Defines a plane normal to the z-axis at a given z.
  - SurfXYPlane
    - Defines a plane parallel to the z-axis with a given radius and a given xy rotation wrt unit vector from origin.
- Also: Bounded extensions of these.

VTrack: a Track at a Surface

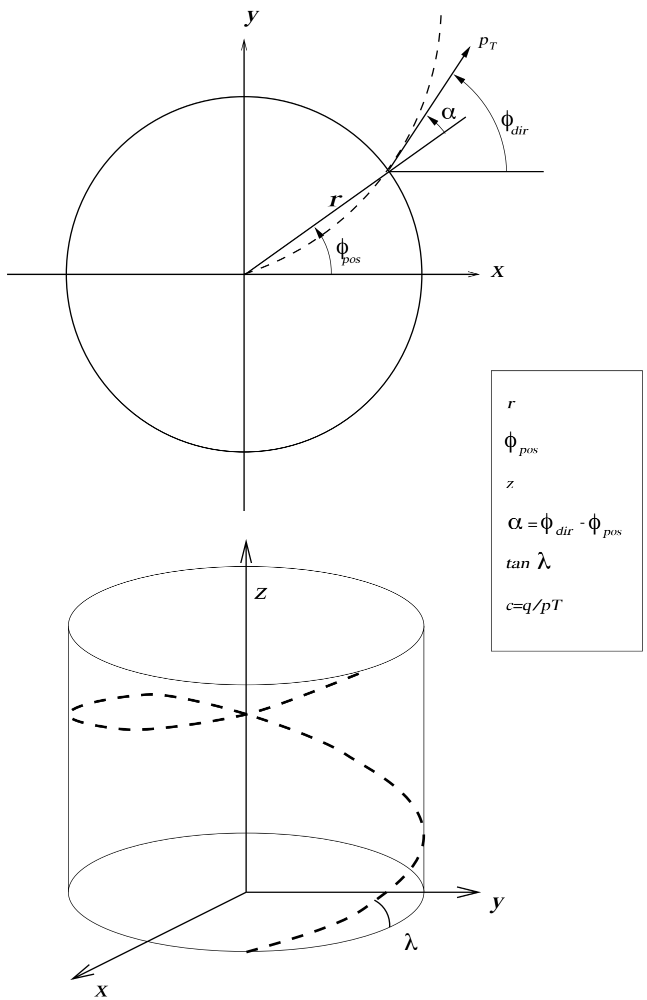
- The simplest track object is a VTrack:
  - A surface at which the track parameters are defined:
  - A 5-vector of parameters.
    - Different meaning for different surface types.
  - A direction wrt surface sense of "outward".
- The first two parameters are the quantity measured by a 2D measurement at that surface.
  - With this choice, the derivatives of the measurements wrt track parameters are simple.
  - I believe that this is a false economy and creates much more pain that it is worth.

Track Parameters wrt PCA to z-Axis

	LCIO	TRF
1	d0 (mm)	d0 (cm)
2	phi0	z0 (cm)
3	q/r (1/mm)	phi0
4	z0 (mm)	tan(lambda)
5	tan(lambda)	q/pT (1/GeV)

- r=radius of curvature;
- pT=transverse momentum.
- q=charge, opposite sign convention!
- Opposite sign convention for d0.

Track Parameters at a Cylinder



- Cylinder centered on z-axis, given r.
- Parameters:
  - phi(direction)
  - z
  - alpha
  - tan(lambda)
  - q/pt

Track Parameters at Planar Surfaces

- Z-Plane: perpendicular to the z-axis.
- XYPlane: contains a line parallel to z-axis.
  - u: unit normal;
  - z along global z;
  - v in plane; completes orthonormal basis

	ZPlane	XYPlane
1	x_global (cm)	v_local (cm)
2	y_global (cm)	z_local (cm)
3	dx/dz	dv/du
4	dy/dz	dz/du
5	q/p (1/GeV)	q/p (1/GeV)

More About Surfaces

- To any surface you can add a list of Interactors.
  - Interactors can do either energy loss or multiple scattering.
  - There are also SimInteractors that can be used for developing internal test suites.
- Fitter can invoke the Interactors to apply multiple scattering and energy loss at each surface.
- Some copy methods for the surface classes do not copy the interactors!

Propagators

- Given n types of surfaces:
- Requires n*n propagator codes:
  - PropCyl - between two cylinders.
  - PropDCA - between two DCA surfaces.
  - PropDCACyl
  - PropCylDCA
- Add a new surface type:
  - Need to write 2n+1 new propagators.
  - It takes me about one week to write and debug a propagator.
    - But I did compose PropDCAZ from PropDCACyl and PropCylZ, which took an hour or so.
  - This does not scale !!!!!!

Clusters

- Cluster depends only on a detector element and has no knowledge of tracks.
- Supports a link back to the list of MCParticles from which it came.
- Can define 1D and 2D clusters at surfaces:
  - ClusCylPhi, ClusCylZ, ClusCylPhiZ2D
  - ClusZPlane1, ClusZPlane2
  - ClusXYPlane1, ClusXYPlane2
- Code has support for stereo angles but I have not yet tested this.

Hits

- No public constructor for hits.
- A hit is created from a cluster plus a track.

```
ETrack et = .... ;                // Both must be defined at
ClusCylPhi cluster =  ...;    // same surface.
List hits = cluster.predict( et );
```

  - HitCylPhi, HitCylZ, HitCylPhiZ2D
  - HitZPlane1, HitZPlane2
  - HitXYPlane1, HitXYPlane2
- A hit can compute:
  - Measurement and its error.
  - Prediction and its error.
  - Derivatives of the measurement wrt track params.

Fitting Tracks

- ETrack
  - VTrack plus an error matrix
- HTrack
  - ETrack plus a list of hits.
- To fit a track:
  - Instantiate a FullFitKalman object.
  - Instantiate an ETrack with the starting track parameters and covariance matrix.
  - Instantiate an HTrack from and ETrack and a hit list.
  - Pass HTrack to the fitter object.
  - On return, HTrack holds the track parameters at the last hit.

A Precision Problem

- A long arc length, at the start of a fit, in which only a subset of the track parameters are measured.
  - Eg. 5 layers of axial strips from tracker barrel at start of an out-to-in fit.
  - At first z measurement, double precision is sometimes not enough to transport the covariance matrix or perform the addhit operation of the fitter.
  - Most pronounced for shallow tracks.
- The problem does not occur for forward fits!
  - Vertex detector measures all 5 parameters over a short lever arm.
- Problem is exacerbated by the choice to change parameterizations at different types of surfaces.

A Precision Problem: The Solution

- The general solution is to modify the fitter to recognize this case and exclude the unmeasured parameters from the fit.
  - Trivial to do for specific cases
  - More complicated to define a general method.
- This is in progress and Norman says that he has some tools that will help.

Things to Know (1)

- B field is in -z direction: charges are opposite!
- Distances in cm (org.lcsim in mm).
- Direction control mechanism is not robust.
  - Not all copy operations on a preserve the sense of the direction of a track.
  - Need an extra step to copy the direction.
- For internal tests:
  - Need to seed a new random number generator for every process at every surface.
  - By default, seeds are all the same!
- Missing propagators:
  - DCA surface to/from ZPlane.
  - Created by introducing an intermediate Cylinder.

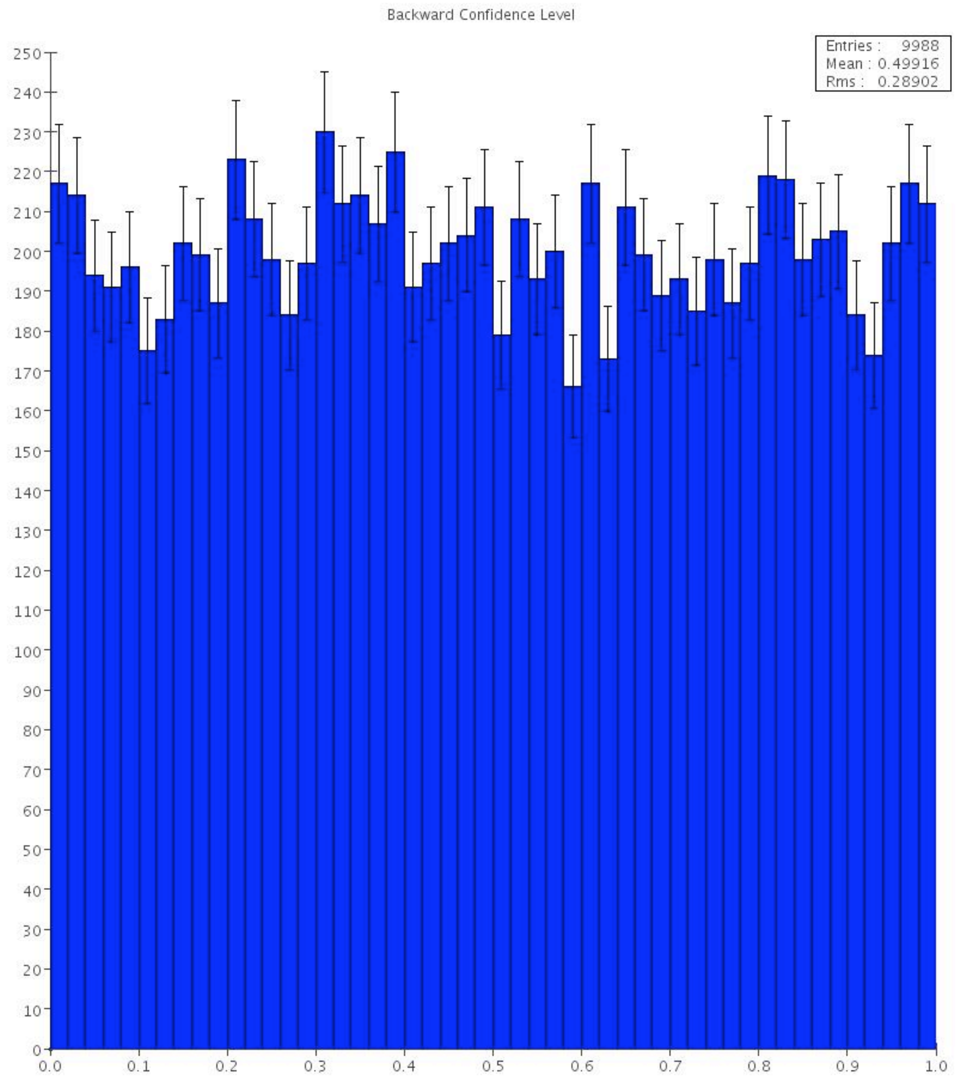
Things to Know (2)

- There are bugs in multiple scattering and eloss code:
  - ThinZPlaneMs, ThinZPlaneMsSim
    - Waiting for Norman to commit fixes.
  - CylEloss, CylElossSim.
    - One bug outstanding and I will send to Norman.

My TRF Internal Tests

- Load geometry from org.lcsim.
- Generate tracks internally.
- Propagate to surfaces, scatter, eloss
  - Measurement errors and scatterng are gaussian.
  - Energy turned off for the study shown here.
  - Resolution: 5(10) microns in Vertex(Tracker).
  - Only consider first arc of the track (for now).
- Remember true track parameters at each surface.
- Fit outwards (seeded from generated params).
- Fit inwards (seeded from outwards fit).
- Quality Control: pulls and Confidence level.
  - Parameter Pulls:
    - (measured-generated)/(measured error)

CL of Inwards Fit

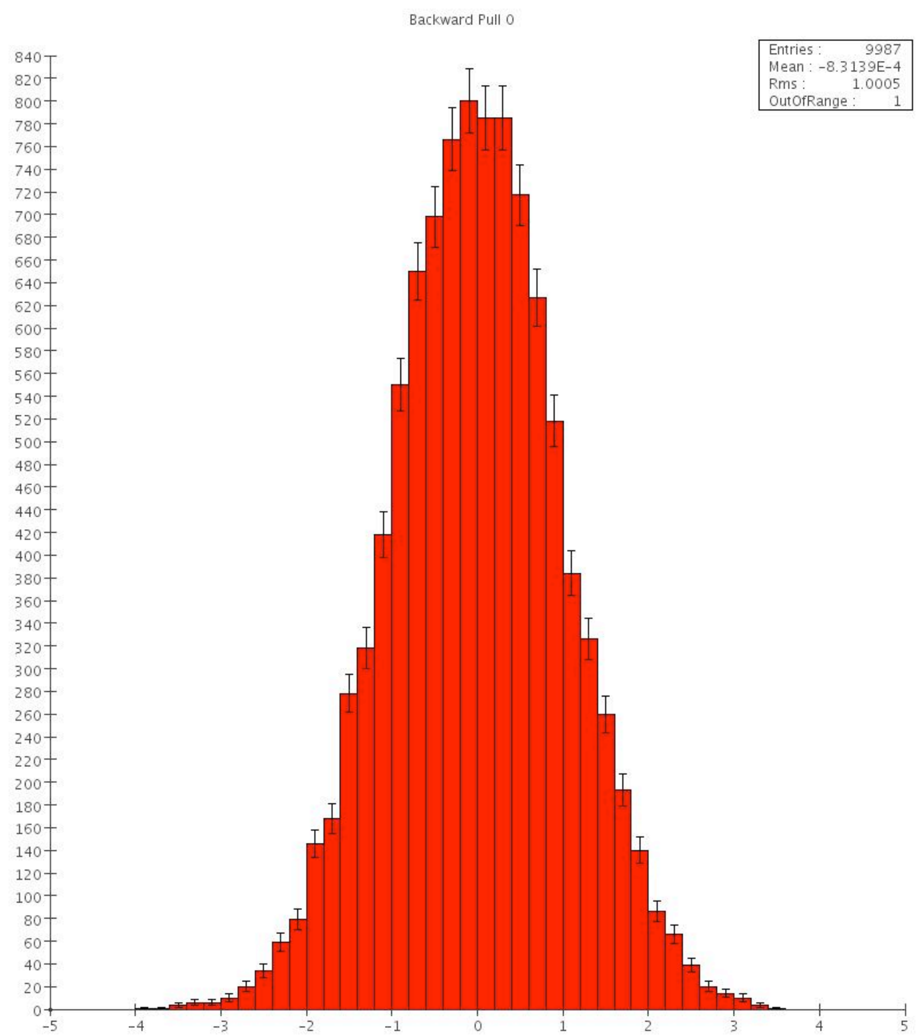


Confidence Level

- 10,000 generated tracks
- 9 tracks with too few hits.

- 3 failed fits
  - Precision bugs
  - Mostly for tracks at barrel- endcap interface.
  - Only on inwards fit!

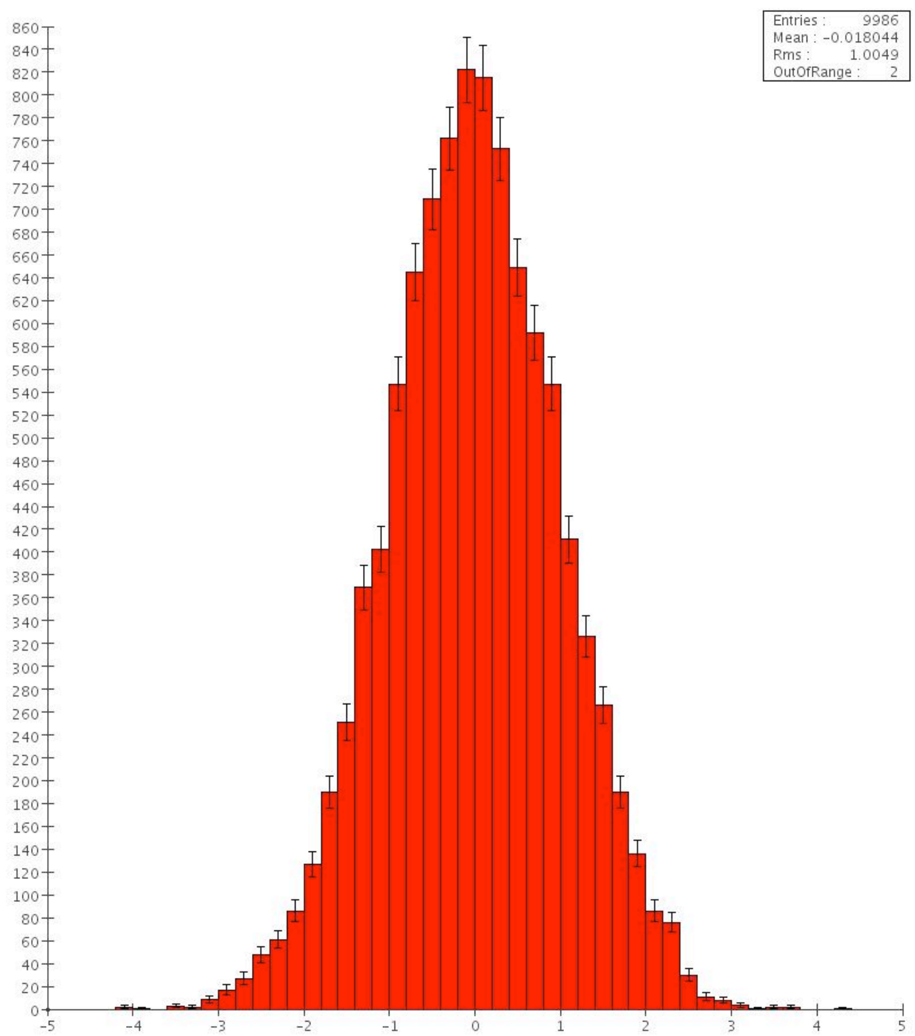
Inwards Fit: Pulls for Parameter d0



- For 10,000 entries
  - sig(mean)=0.01
  - sig(rms) =0.007
- Out of bounds:
  - Precision bugs.

- Generated Tracks
- Flat in all:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
  - 100 mrad
  - 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi

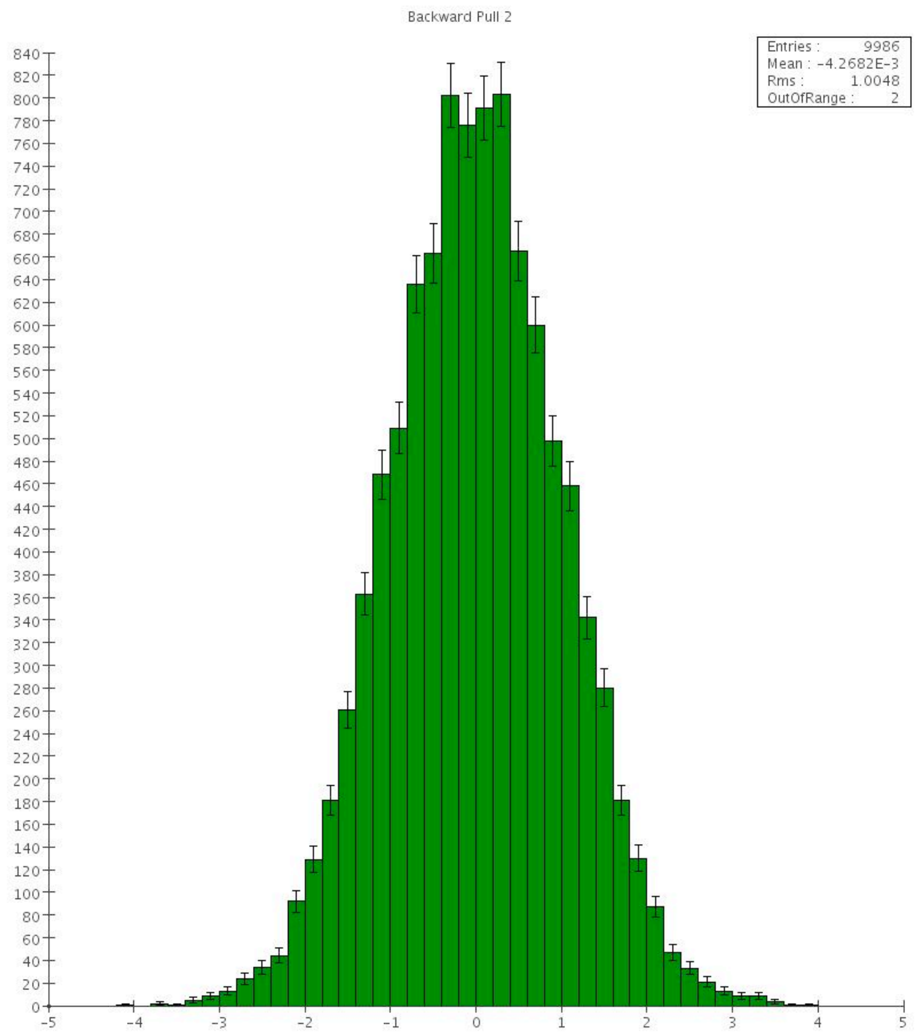
Inwards Fit: Pulls for Parameter z0



- For 10,000 entries
  - sig(mean)=0.01
  - sig(rms) =0.007
- Out of bounds:
  - Precision bugs.

- Generated Tracks
- Flat in all:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
  - 100 mrad
  - 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi

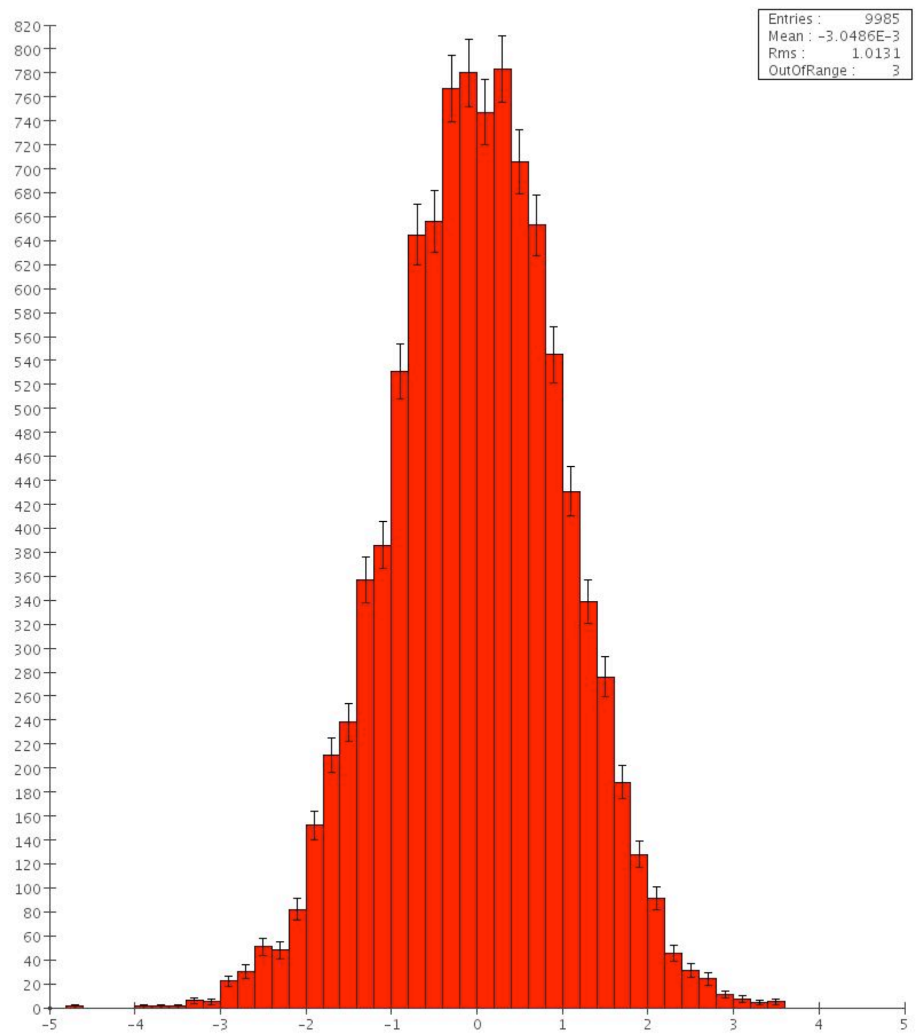
Inwards Fit: Pulls for Parameter Phi0



- For 10,000 entries
  - sig(mean)=0.01
  - sig(rms) =0.007
- Out of bounds:
  - Precision bugs.

- Generated Tracks
- Flat in all:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
  - 100 mrad
  - 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi

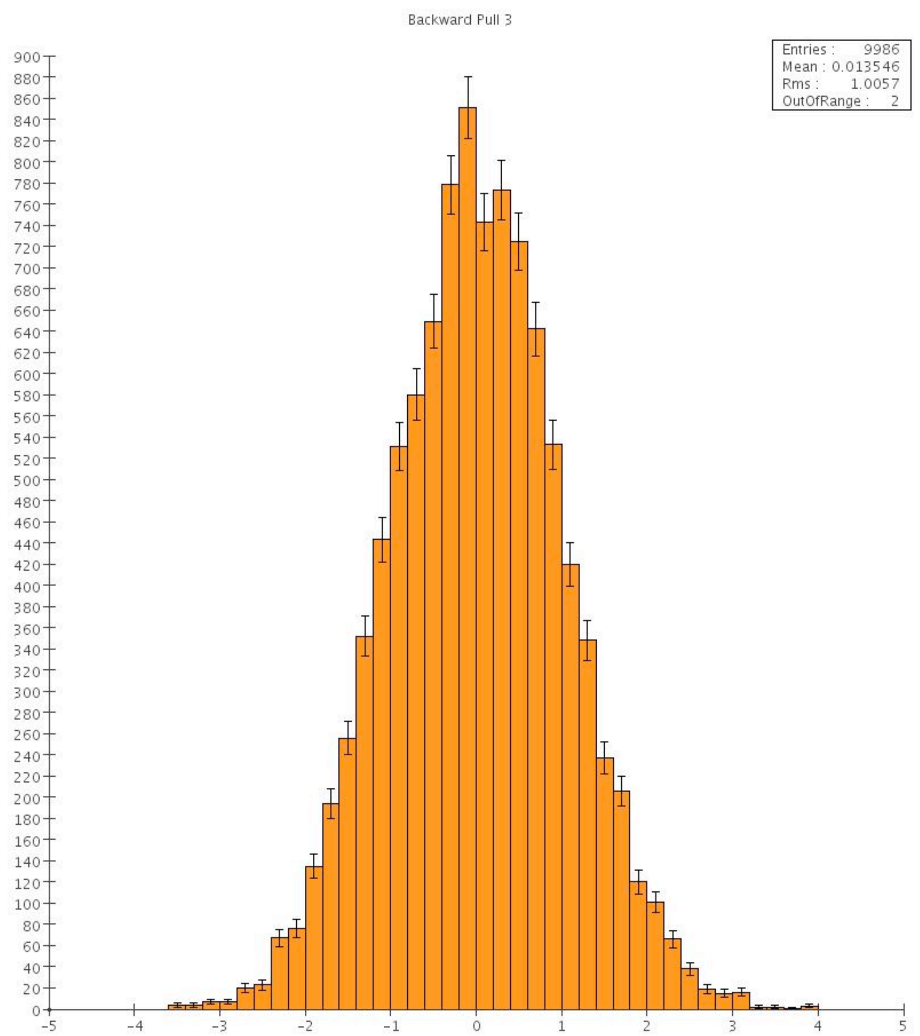
Inwards Fit: Pulls for Curvature (q/Pt)



- For 10,000 entries
  - sig(mean)=0.01
  - sig(rms) =0.007
- Out of bounds:
  - Precision bugs.

- Generated Tracks
- Flat in all:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
  - 100 mrad
  - 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi

Inwards Fit: Pulls for Parameter cot(theta)



- For 10,000 entries
  - sig(mean)=0.01
  - sig(rms) =0.007
- Out of bounds:
  - Precision bugs.

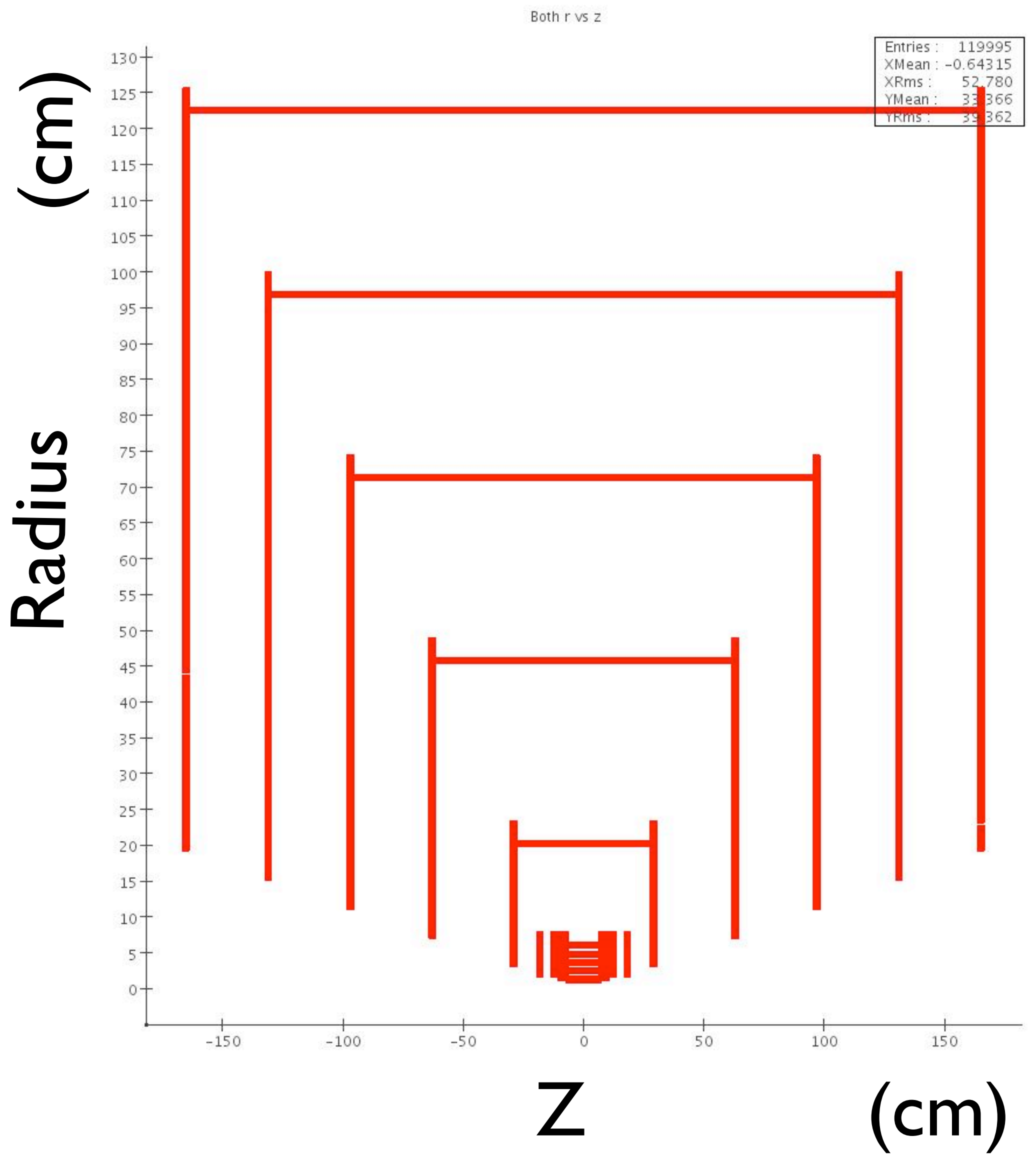
- Generated Tracks
- Flat in all:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
  - 100 mrad
  - 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi

Status and Plans

- Internal tests nearly complete:
  - One more bug in Eloss code to be resolved.
  - Bypass precision problem (hack solution for now).
- Next steps:
  - Follow the plan from page 5.
  - Track parameters at multiple locations.
    - See: Evolving the Track Class in the ILC Forum.
  - Provide examples and documentation.
- Goal: first release before Christmas 2007.
  - May require ongoing tweaks to the material discovery code and the tuning of material models to match G4.
- Work with vertexing code people to properly treat vertices outside of beam pipe.

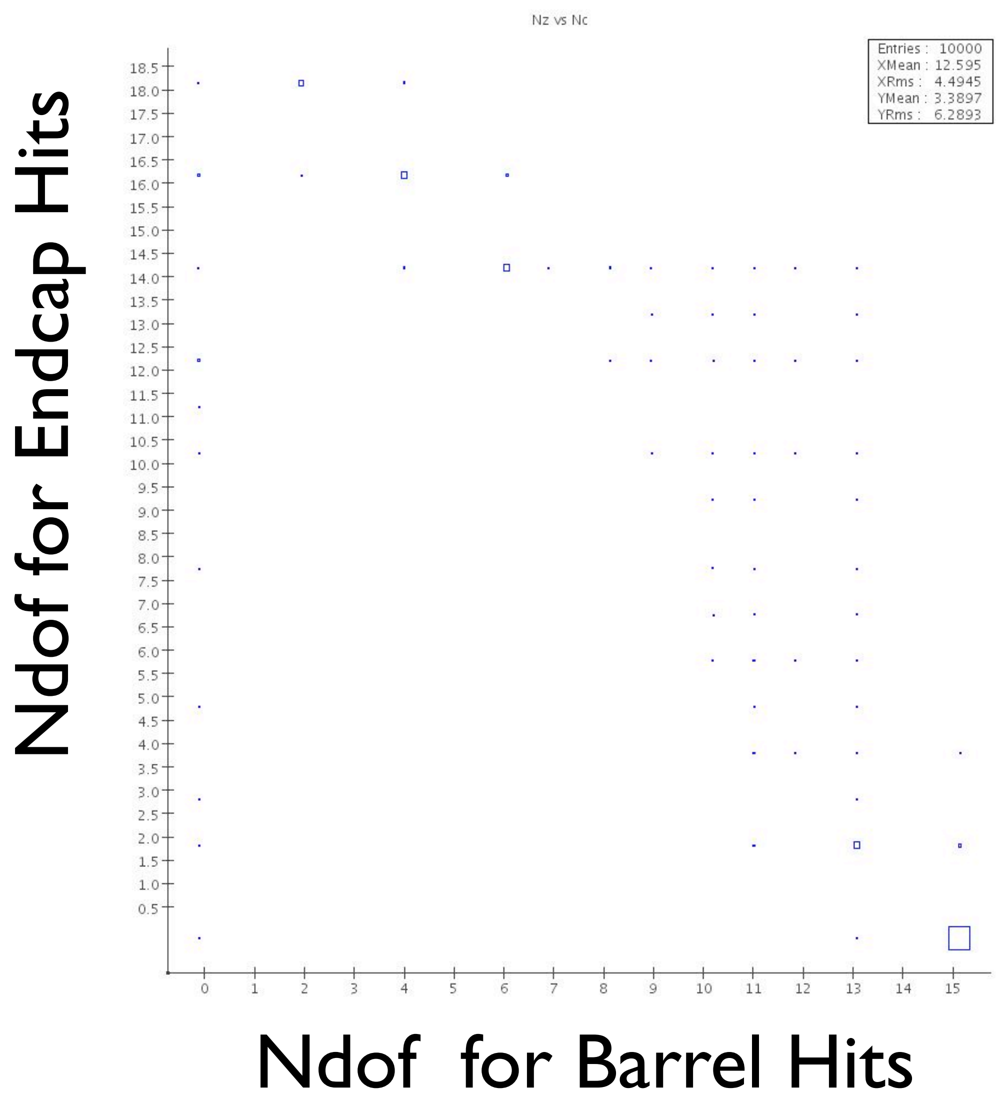
Backup Slides

Position of Generated Hits in sid01



- Generated Tracks
- Flat in:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
- 100 mrad
- 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi

Numbers of Hits



- Ndof=2 for pixels.
- Ndof=1 for strips.

- Generated Tracks
- Flat in all:
- |d0| < 4 mm
- |z0| < 10 mm
- |cos(theta)|<0.995
  - 100 mrad
  - 5.6 degrees
- 1< Pt < 10 GeV
- 0< Phi < 2pi