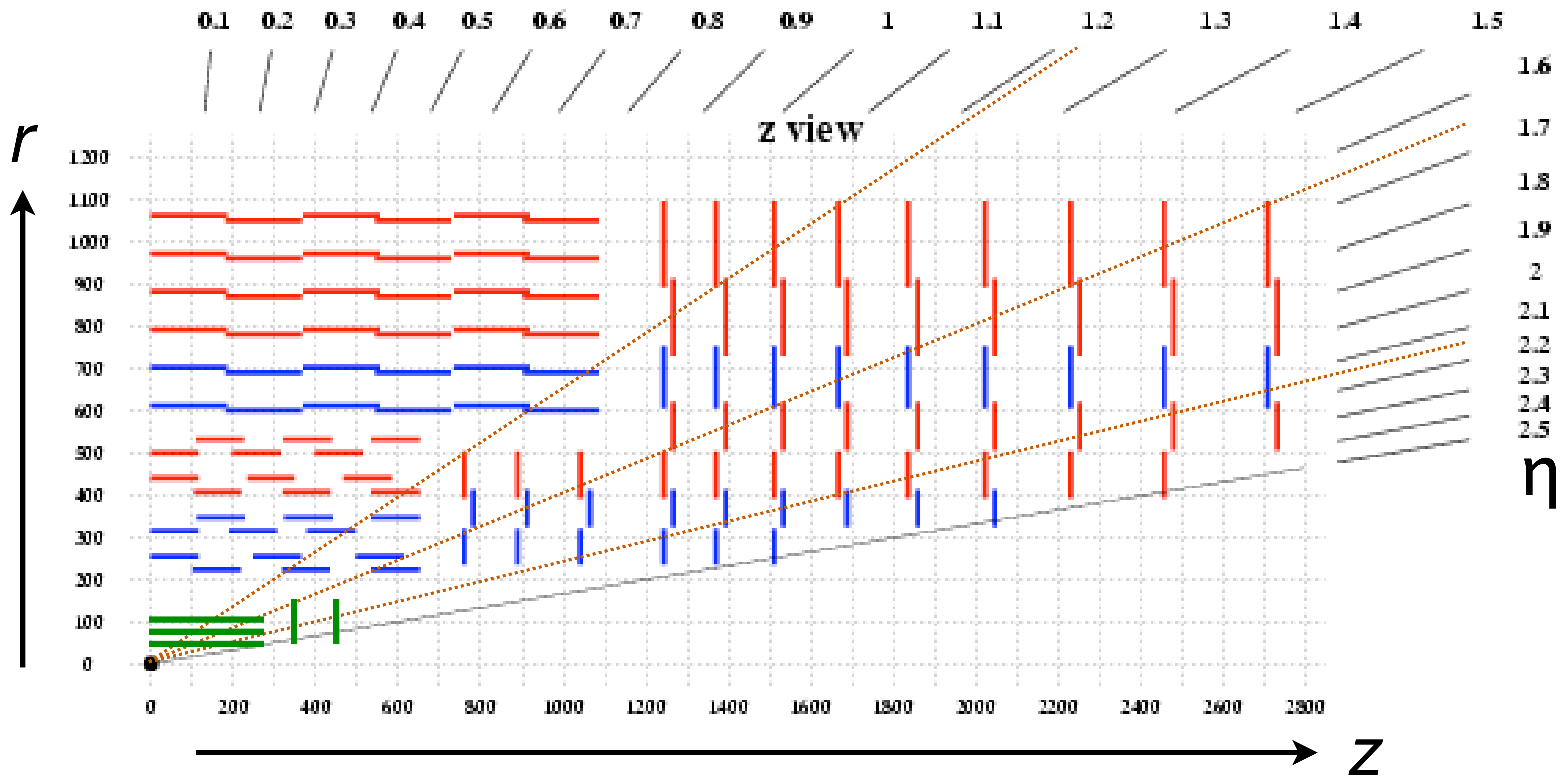
Details of Tracking at CMS

LHC Physics Workshop, TIFR, Mumbai
October 27, 2009

Kevin Burkett (Fermilab)
on behalf of the CMS Collaboration

Summary of Tracking System

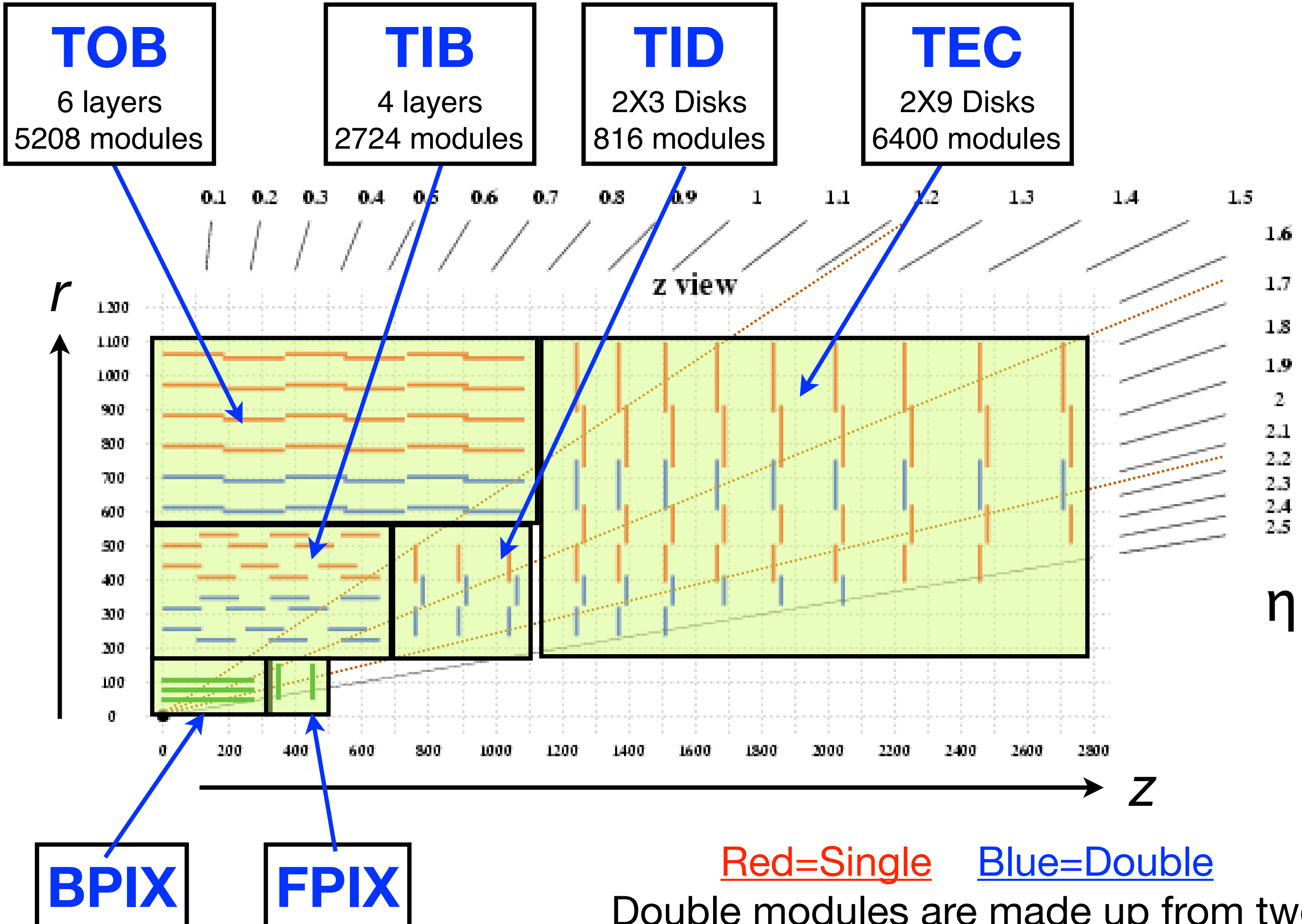
1.1m lever arm in 3.8T B-field



Red=Single Blue=Double

Double modules are made up from two single sided modules glued back-to-back

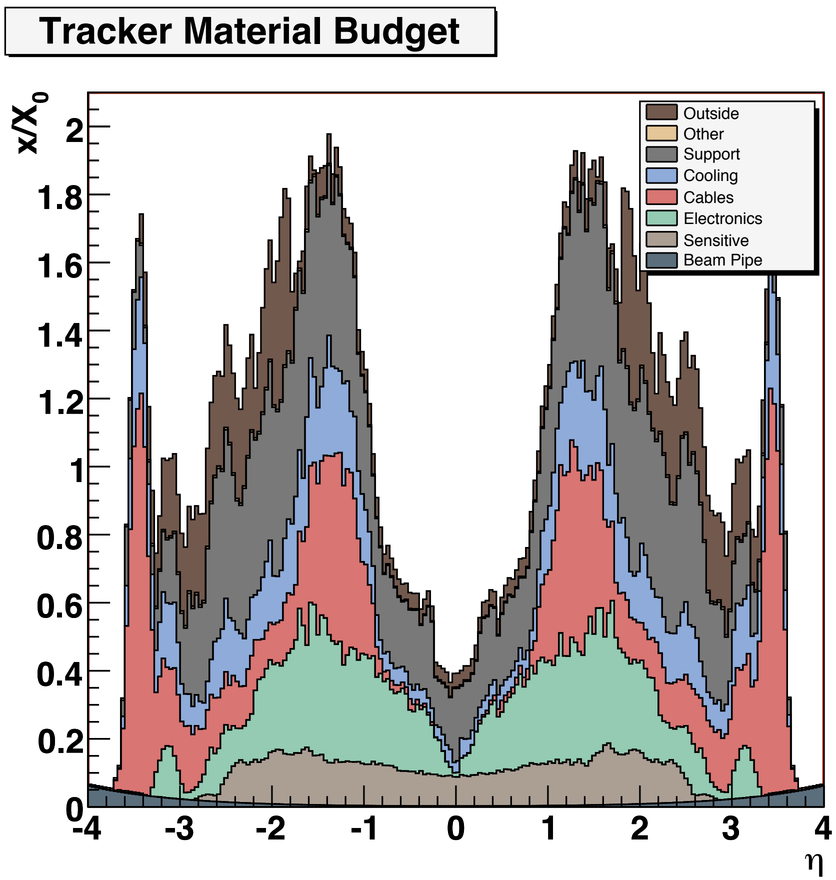
Summary of Tracking System

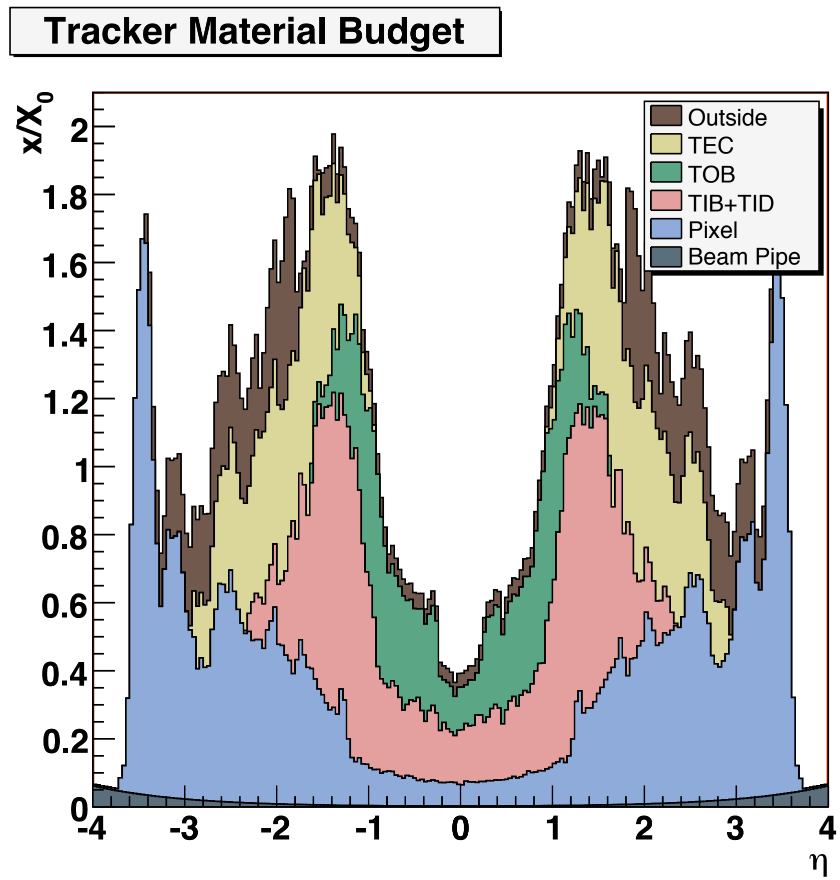


Red=Single Blue=Double

Double modules are made up from two single sided modules glued back-to-back

Material Budget





There is a significant amount of material in the Tracker

- More pronounced at higher η
- Not just from sensitive elements, but from cables, cooling, etc.
- Must not be forgotten!

Reconstruction of Tracks

Outline of Track Reconstruction

Local Reconstruction: Clustering of silicon strips and pixels to find “hit” positions and errors

↓

Track Seeding: Initial estimate of track parameters using a minimal number of hits

↓

Trajectory Building: Collection of the remaining hits associated to the particle trajectory

↓

Track Fitting: Final estimate of the track parameters using the full set of associated hits

↓

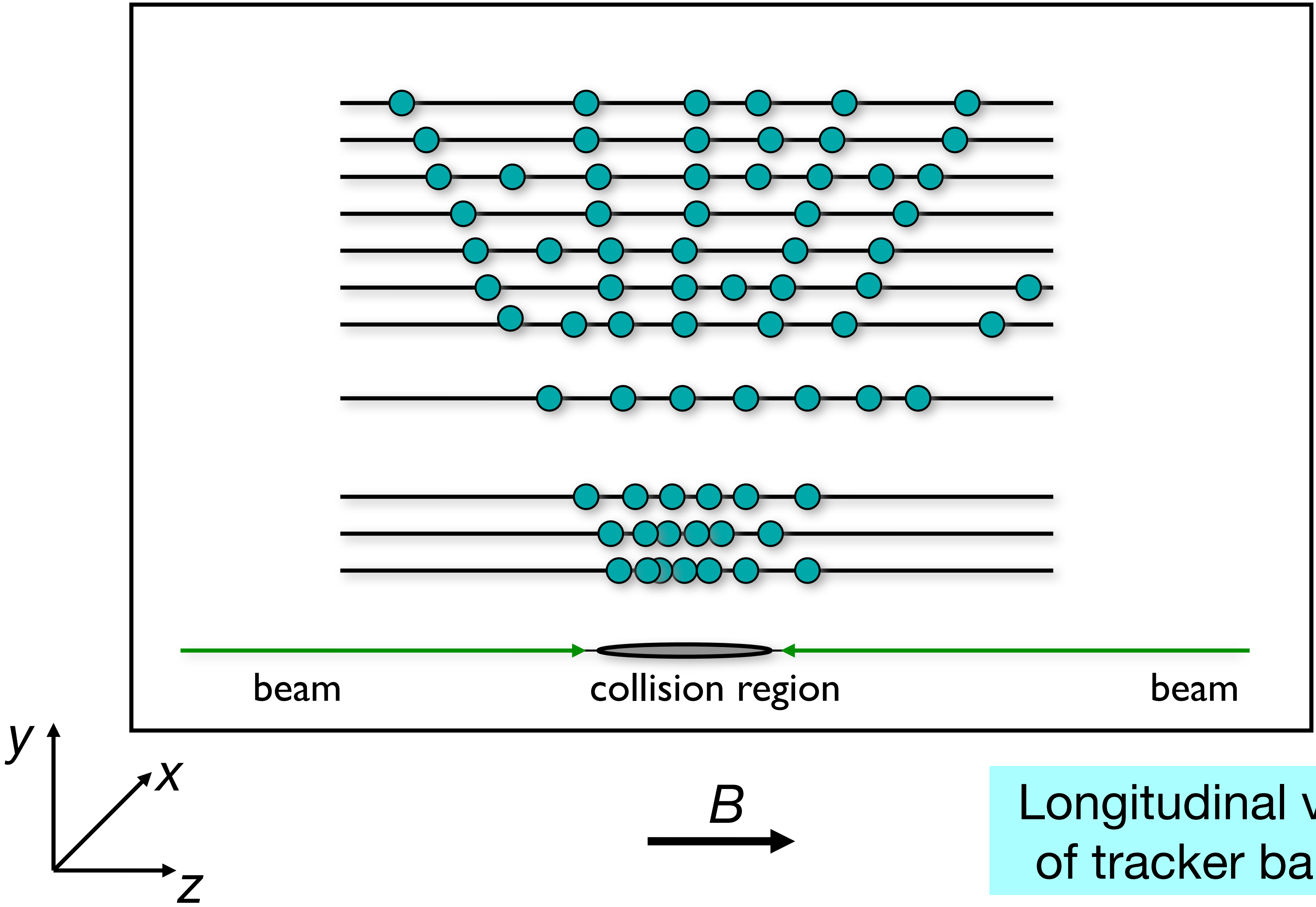
Track Filtering: Removal of tracks likely to be fakes

Steps of Track Reconstruction

<u>*Track Seeding*</u>

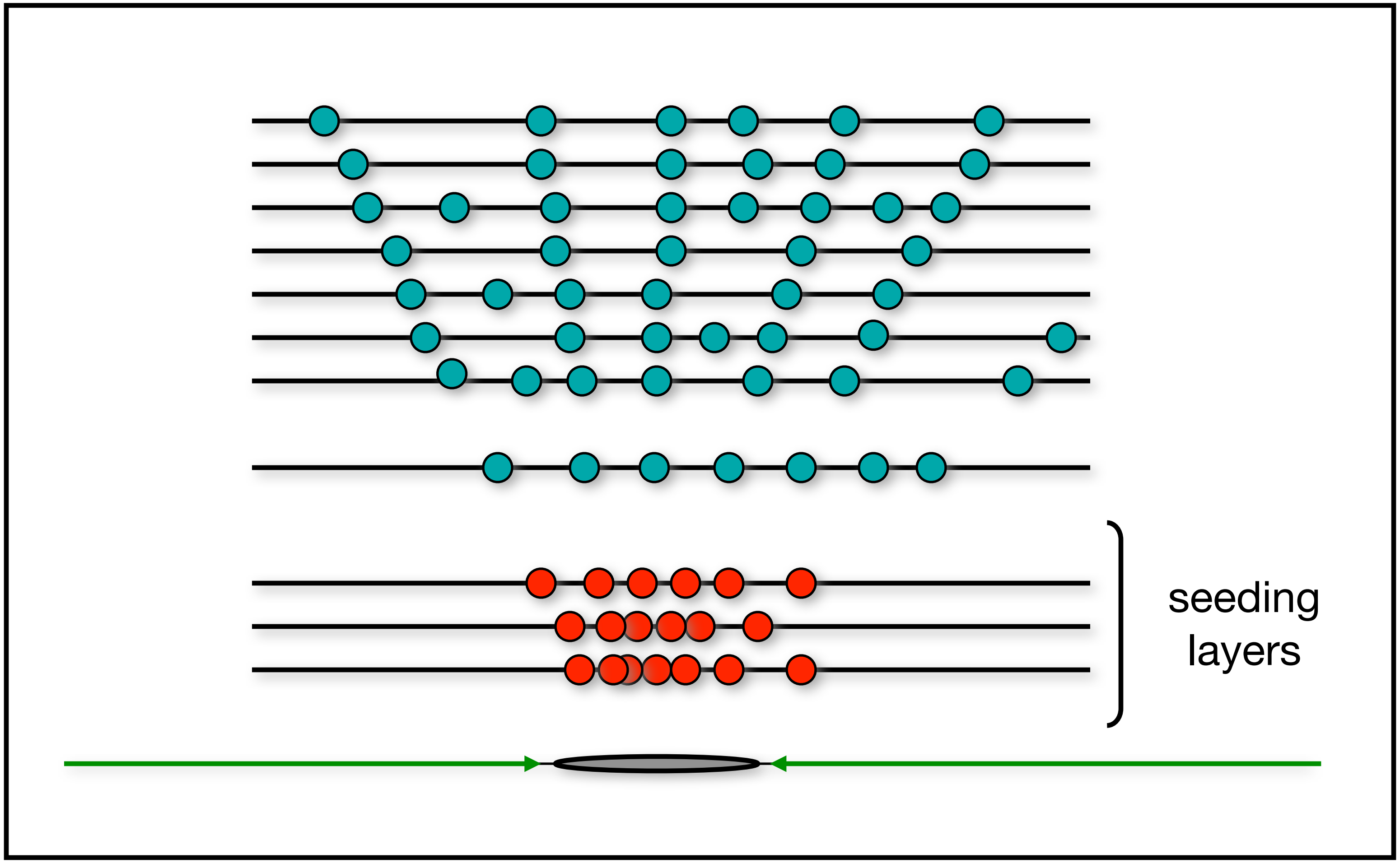
* *Default seeding is done in pixel layers*
* *Tracks are built from inside to out because many particles will interact before crossing all layers of the tracker*
* *We need a minimum of three points to seed the track*
    - Gives a fully defined helix, with uncertainties
    - We can use either three hits in the tracker (either pixel or strip) OR a pair of hits plus a vertex or the beamline
    - In fact, we will make use of all of these possibilities

After Local Reconstruction



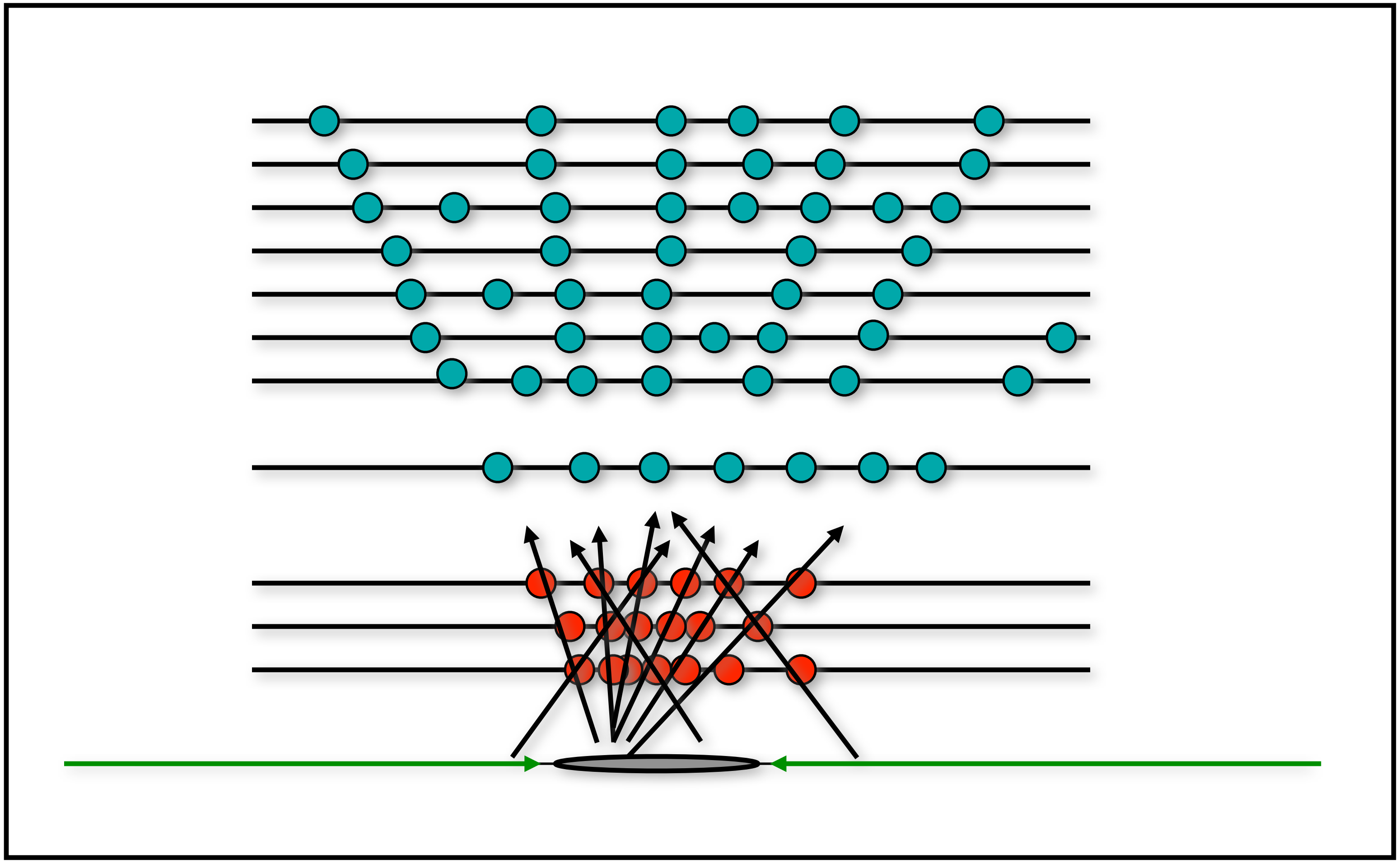
Longitudinal view of tracker barrel

Seed Finding



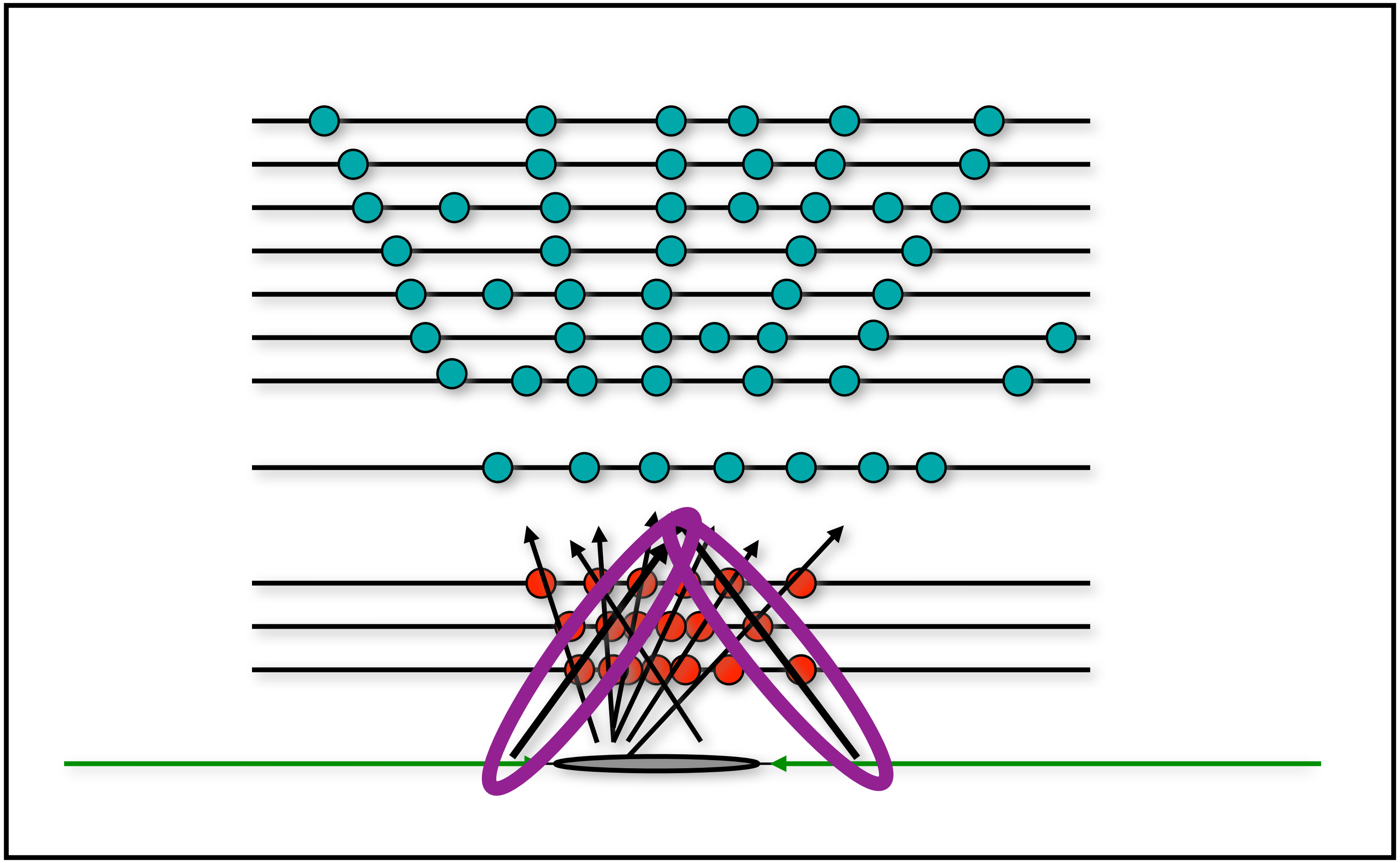
Hits in a subset of tracker layers are used to find trajectory seeds. Default seeds are made from pixels, the innermost layers of the tracker.

Seed Finding



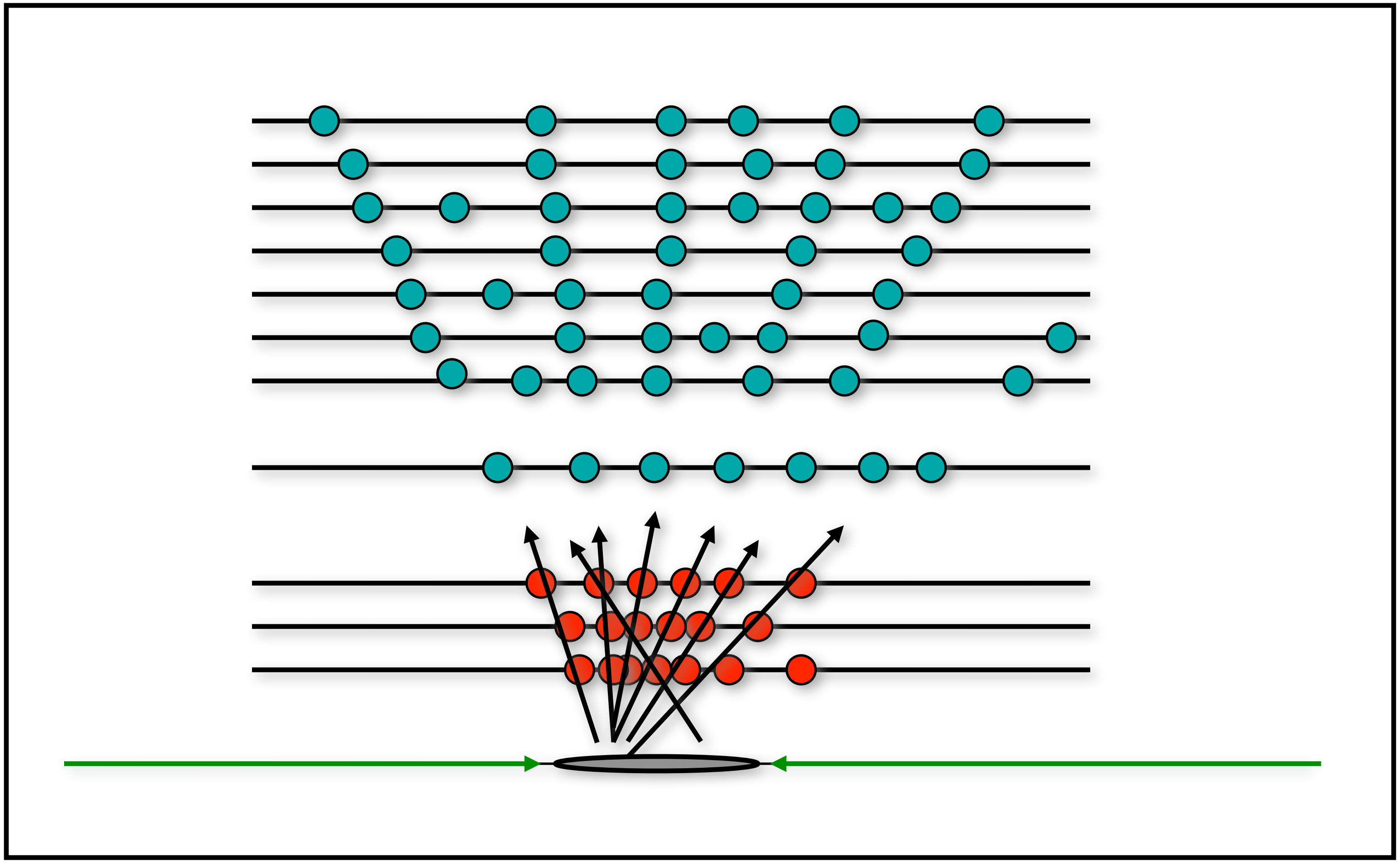
Triplets of hits (or pairs plus the beamspot) are combined to produce trajectory seeds whose directions are **compatible with the beam collision region.**

Seed Finding



Seeds that point well outside the collision region
are discarded

Trajectory Building



For the remaining seeds, each one is then propagated outward to collect more hits to find the full trajectory of the charged particle.

Steps of Track Reconstruction

<u>Trajectory Building:</u>

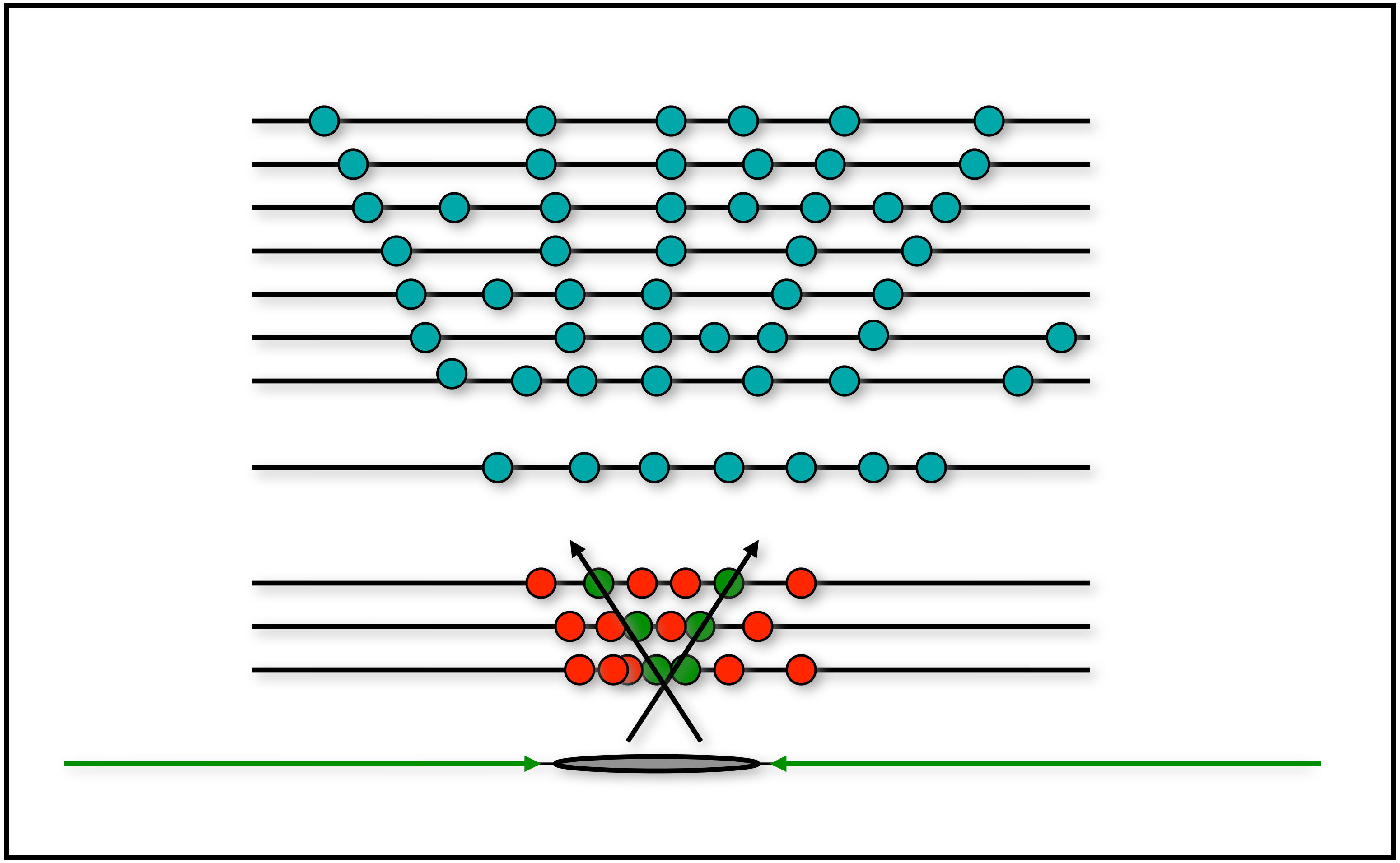
- *The seed has a fully defined helix, so we can extrapolate this outward to the next layer, along with the uncertainties, to look for compatible hits*
- *The window for "compatible" hits depends on the uncertainty on the trajectory, and also on the hit uncertainty*
- *If a hit is found, the trajectory is updated, and you extrapolate to the next layer*
- *If NO hit is found, an "invalid hit" is placed at the point that the trajectory intersects the layer, and you extrapolate the trajectory to the next layer, continuing to extrapolate the uncertainties from the previous layer*
  - You are essentially allowed one invalid hit per track
  - However, the reconstruction knows about dead modules, so the track is not "penalized" for them
  - Invalid hits allow us to still account for the material in that layer

Steps of Track Reconstruction

The track finder is called the ***Combinatorial Track Finder*** *(****CTF****)*

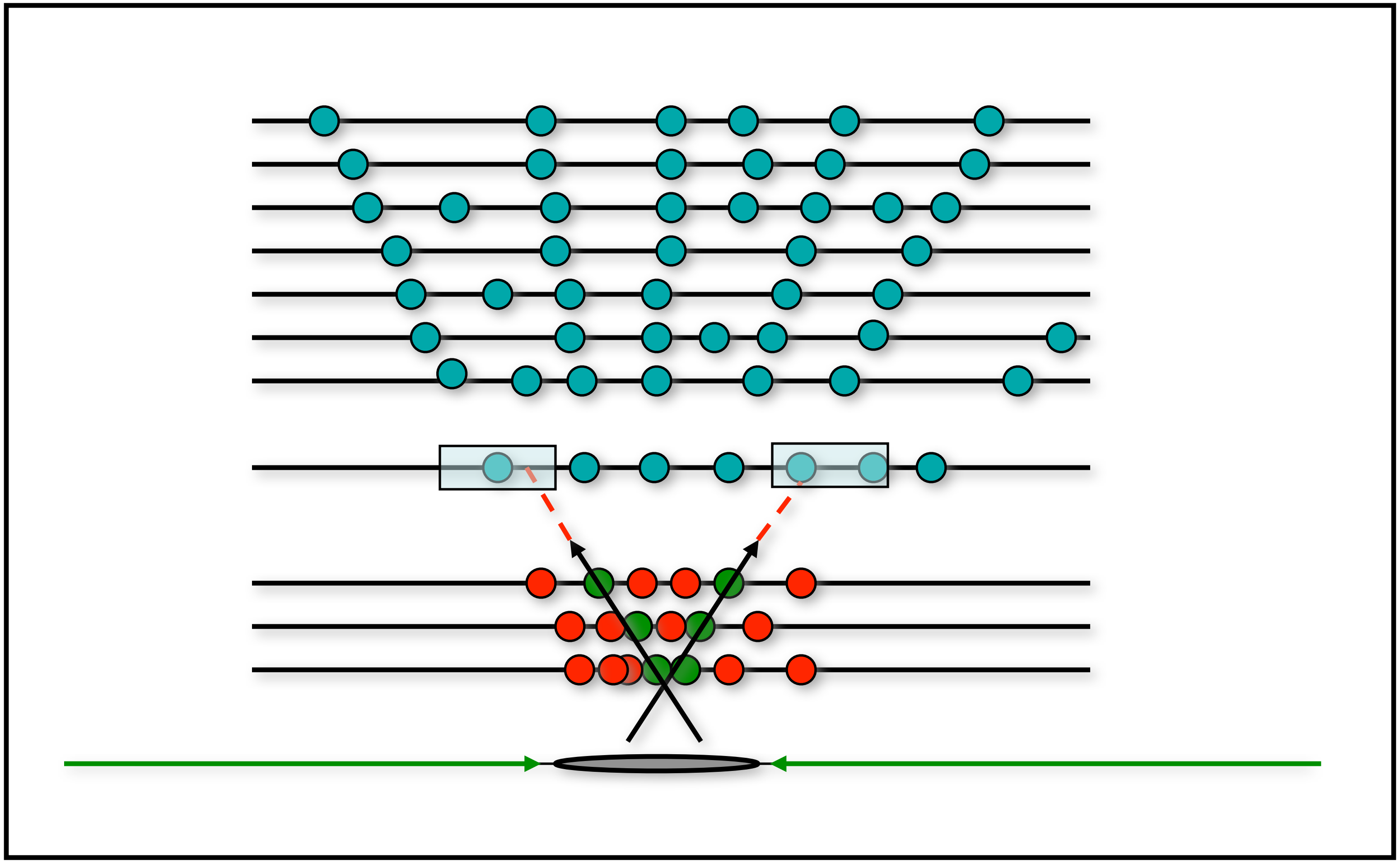
- *If more than one compatible hit is found on a layer, then two trajectories are made, using each of the two hits*
- *Then each of those trajectories is extrapolated outward, looking for compatible hits in the next layer*
- *There is a configurable parameter that limits the maximum number of combinations that will be retained*
    - By default set to 5
- *At the end of trajectory building, there is a cleaning stage where trajectories are compared for duplicates*
    - If two trajectories share a majority of their hits, then you retain the trajectory with the most hits and lowest chi2

Trajectory Building



Consider these two specific seeds

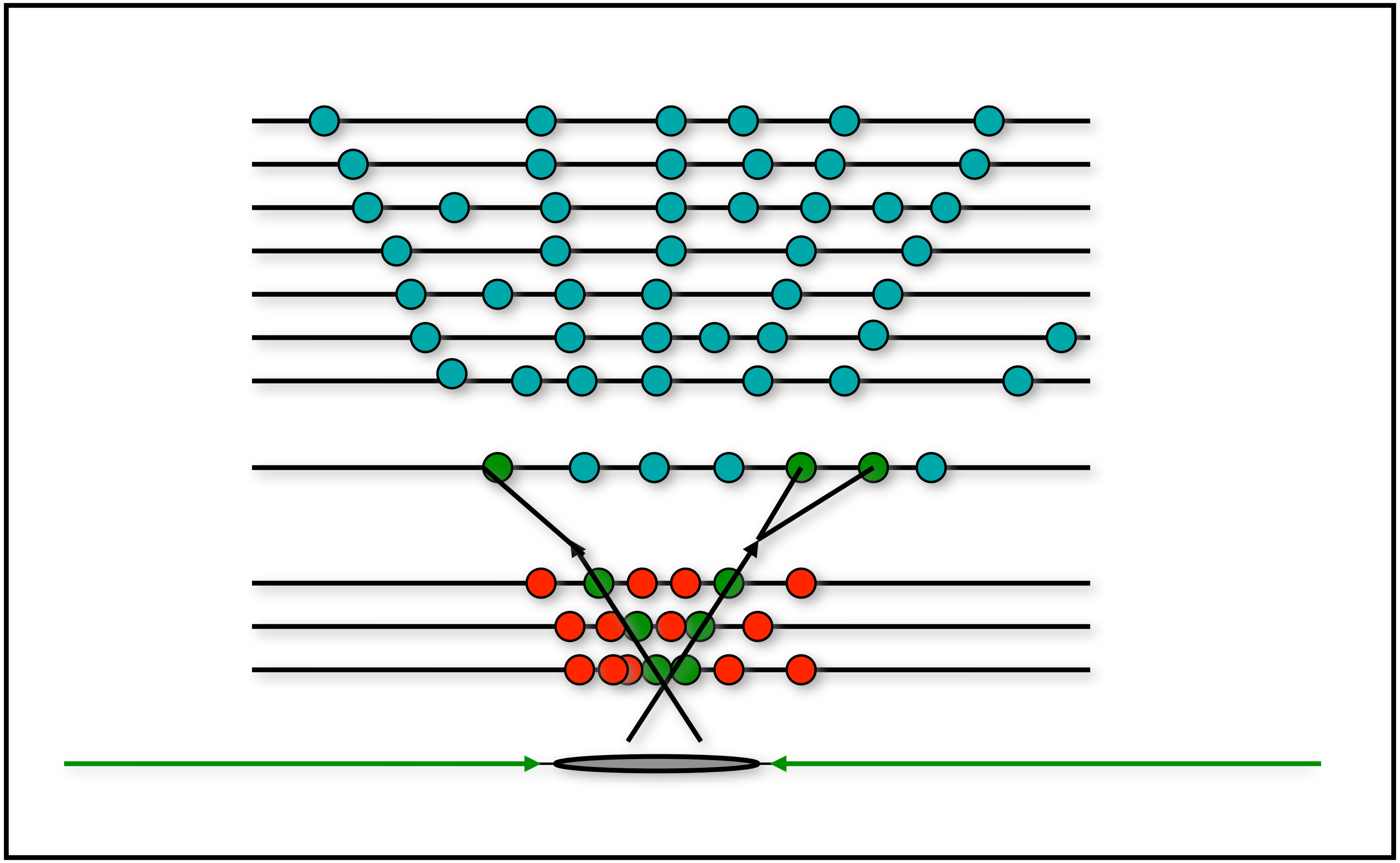
Trajectory Building



Seeds are propagated to the next layer to find all compatible hits.

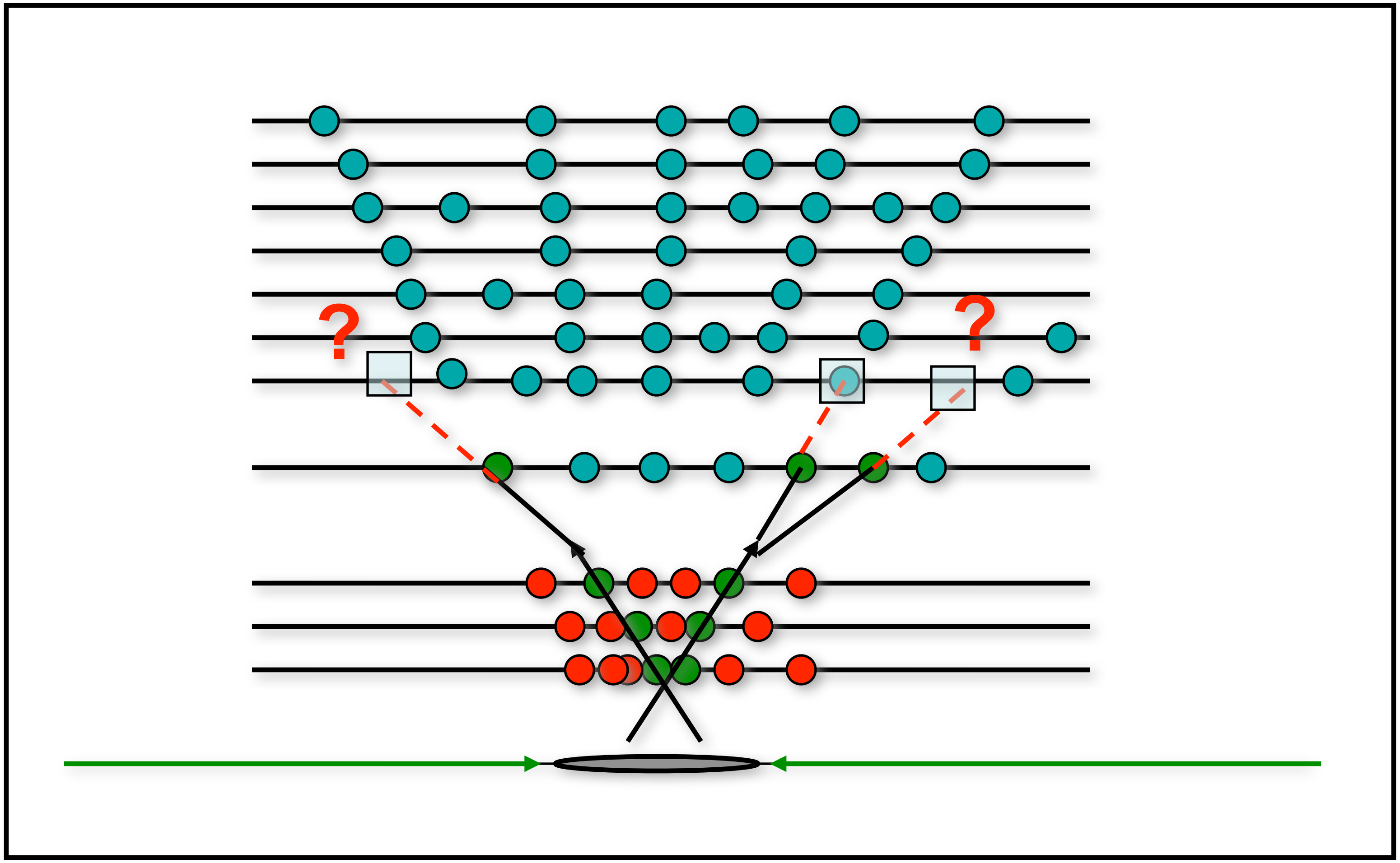
The compatibility considers both the uncertainty on the trajectory and the uncertainty on the hit position

Trajectory Building



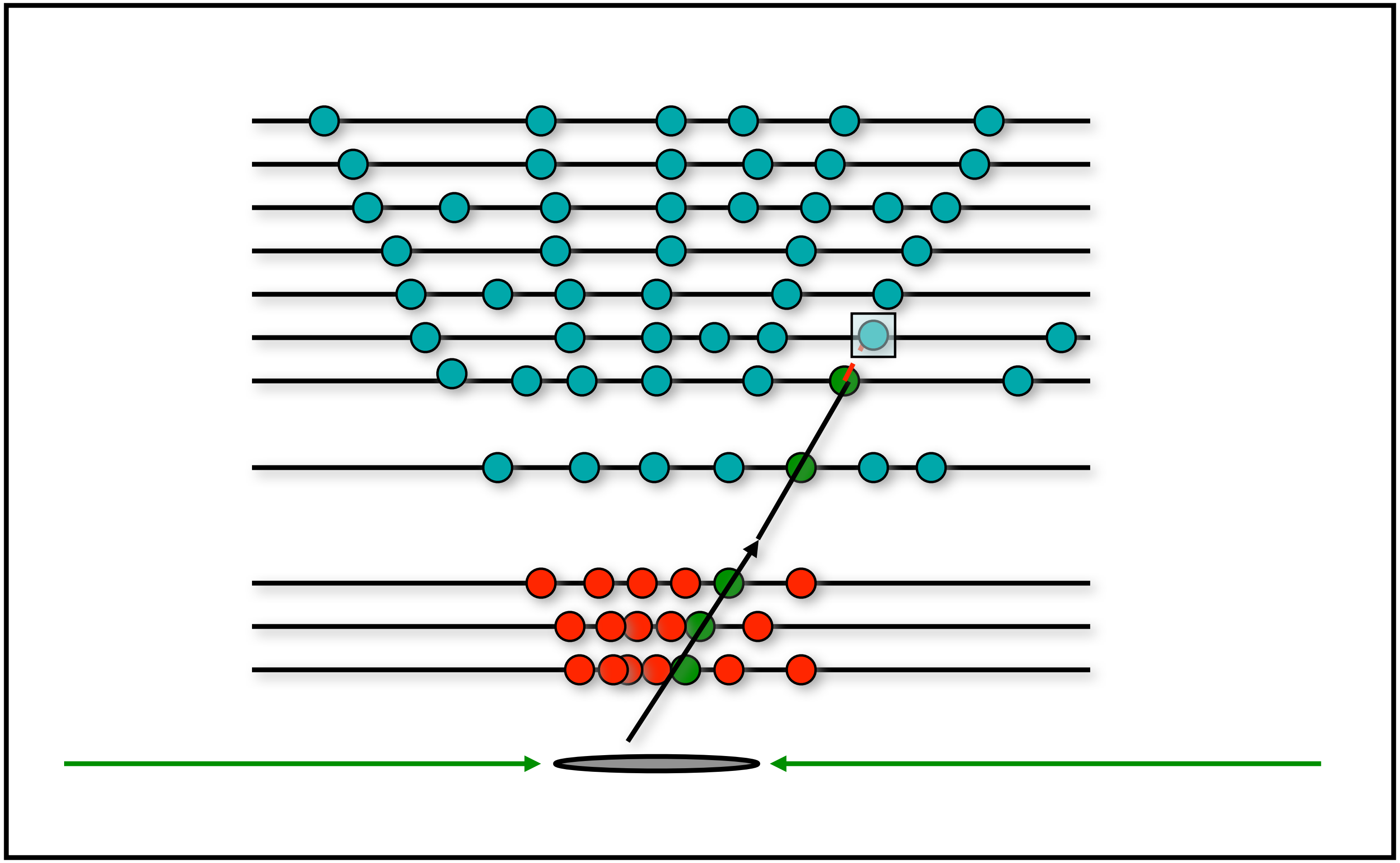
If more than one hit is compatible with the propagated trajectory, the track is split into 2 or more candidates, which are then built in parallel

Trajectory Building



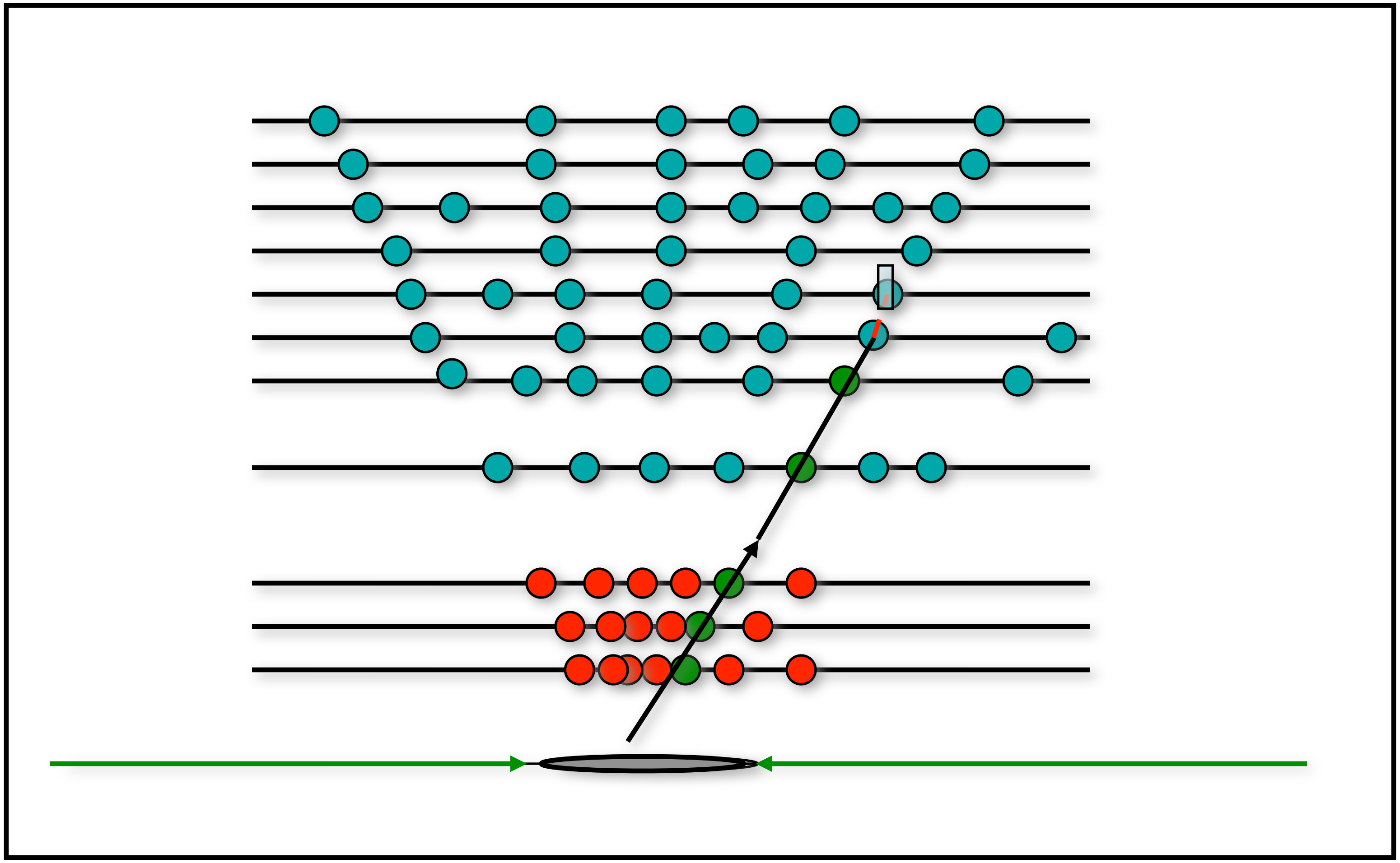
If the propagated trajectory intersects a layer where there are no compatible hits, the trajectory is likely a fake and can be rejected.

Trajectory Building



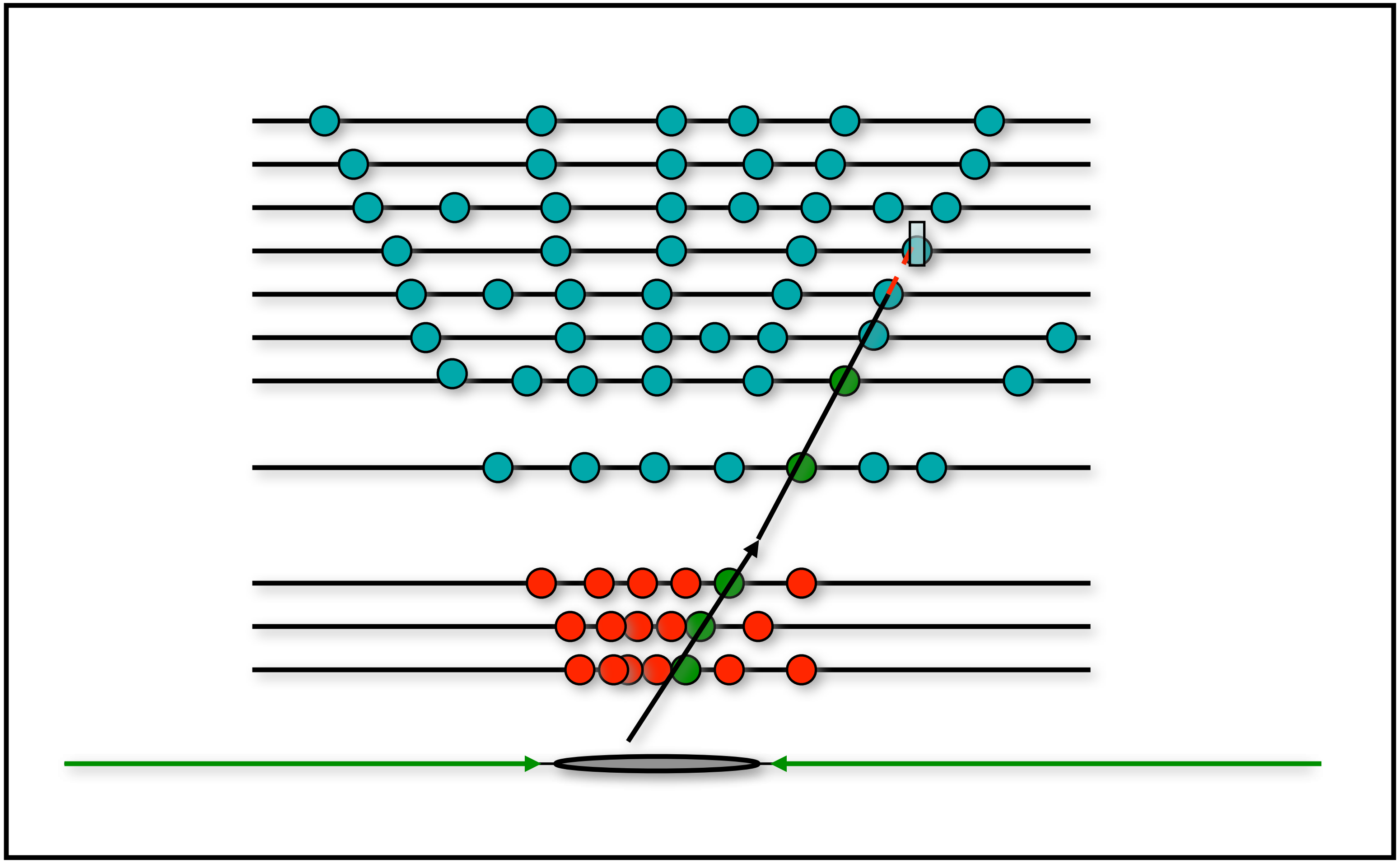
As more hits are added, the parameters become well-constrained and the remaining hits are found easily

Trajectory Building



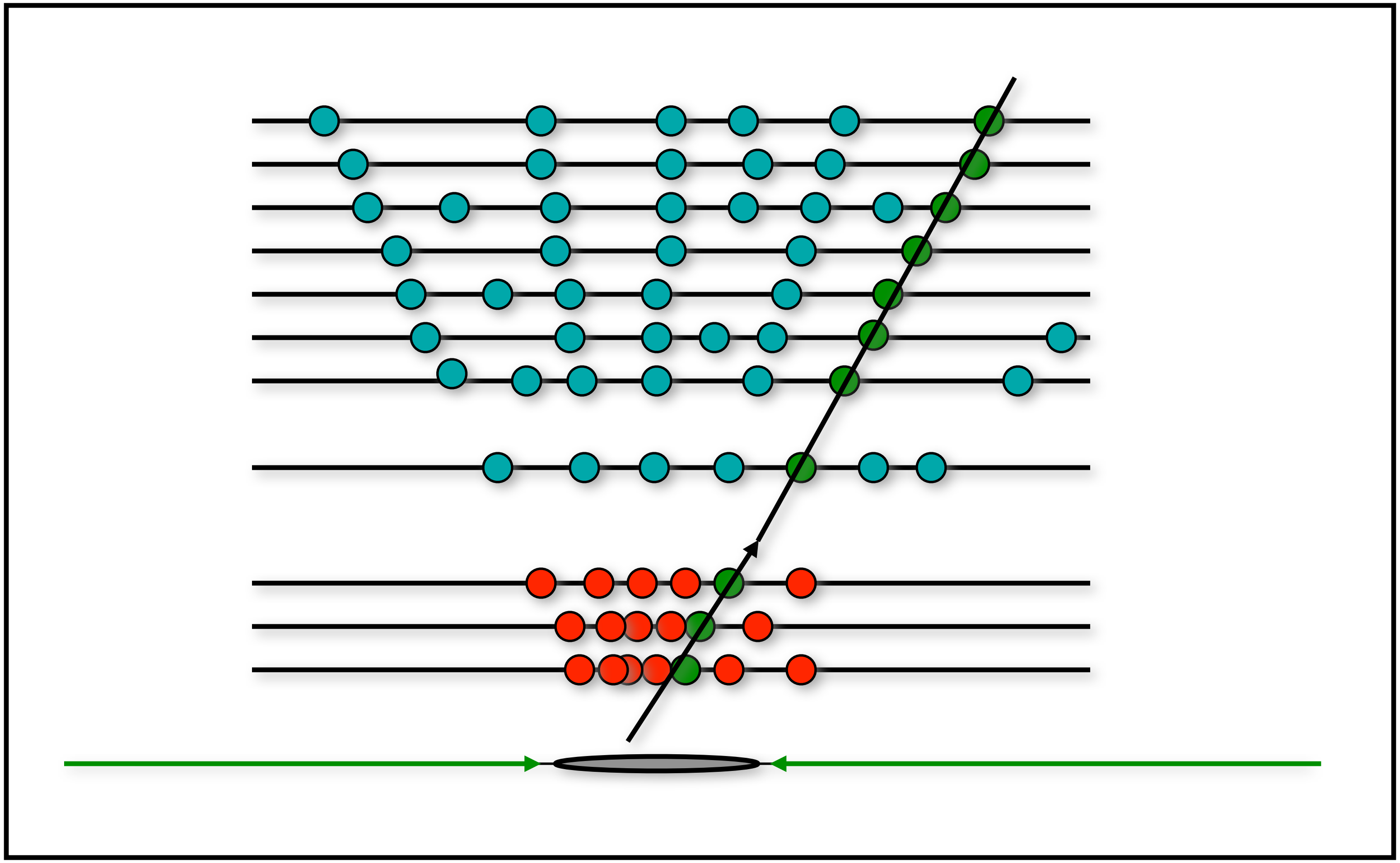
As more hits are added, the parameters become well-constrained and the remaining hits are found easily

Trajectory Building



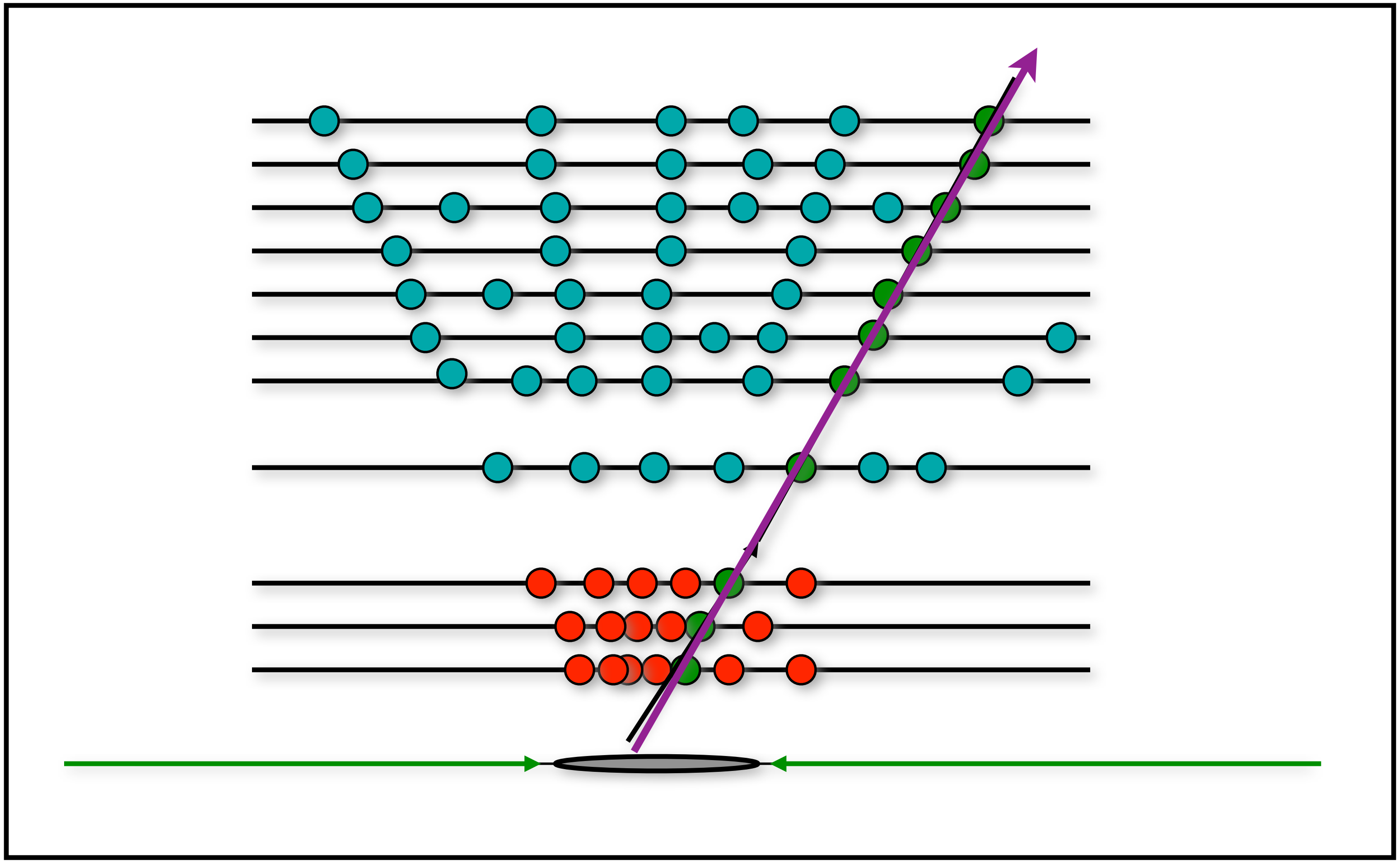
As more hits are added, the parameters become well-constrained and the remaining hits are found easily

Trajectory Building



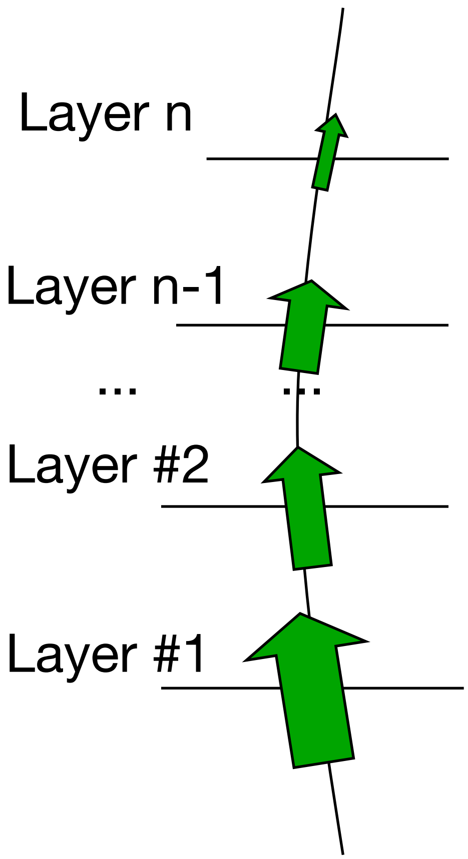
Trajectory building continues until no more compatible measurements are found, or the trajectory reaches the end of the tracker

Track Fitting



Final fit uses all hits to obtain the best measurement of the track parameters at the point of production

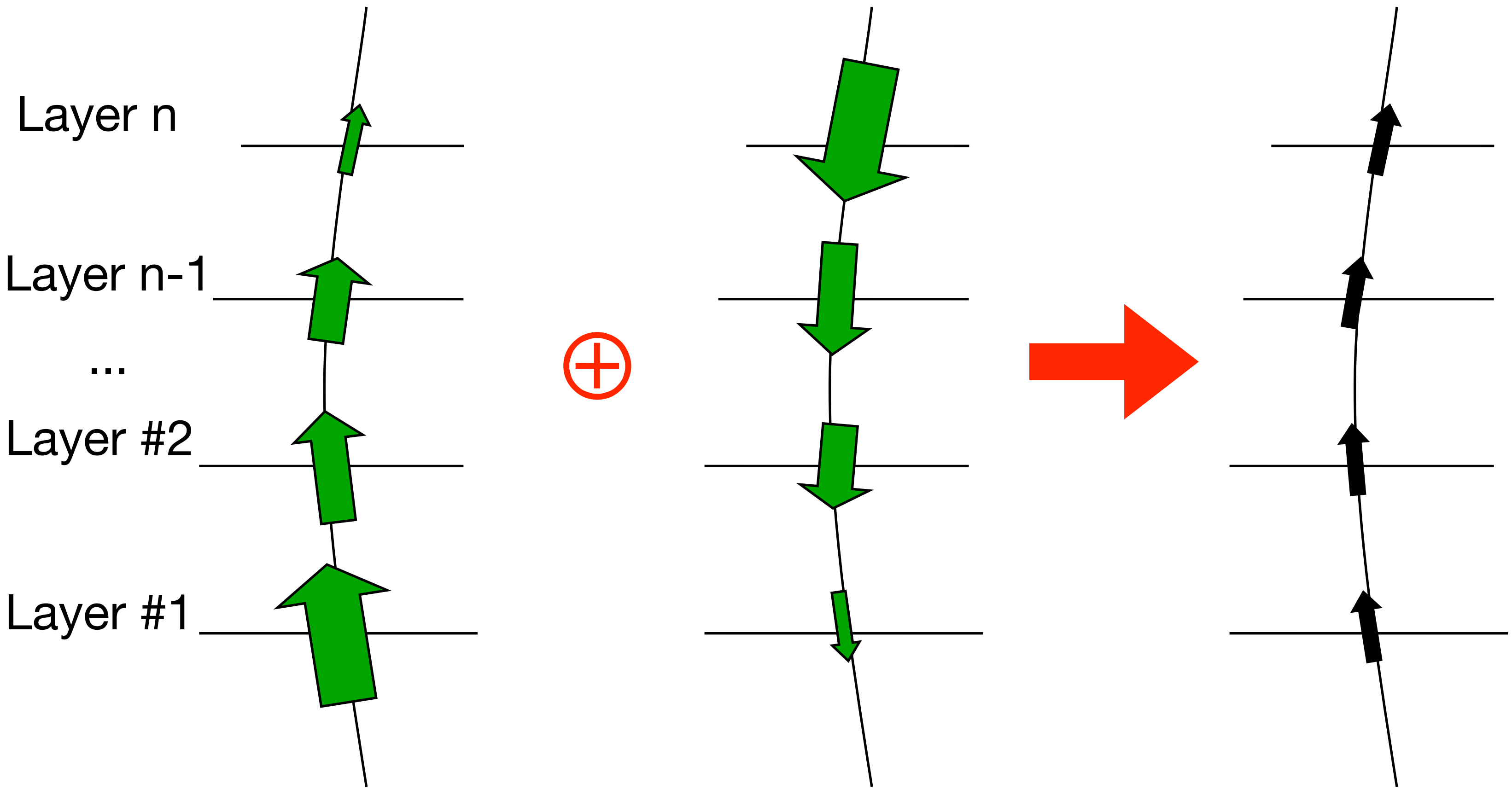
Track Fitting



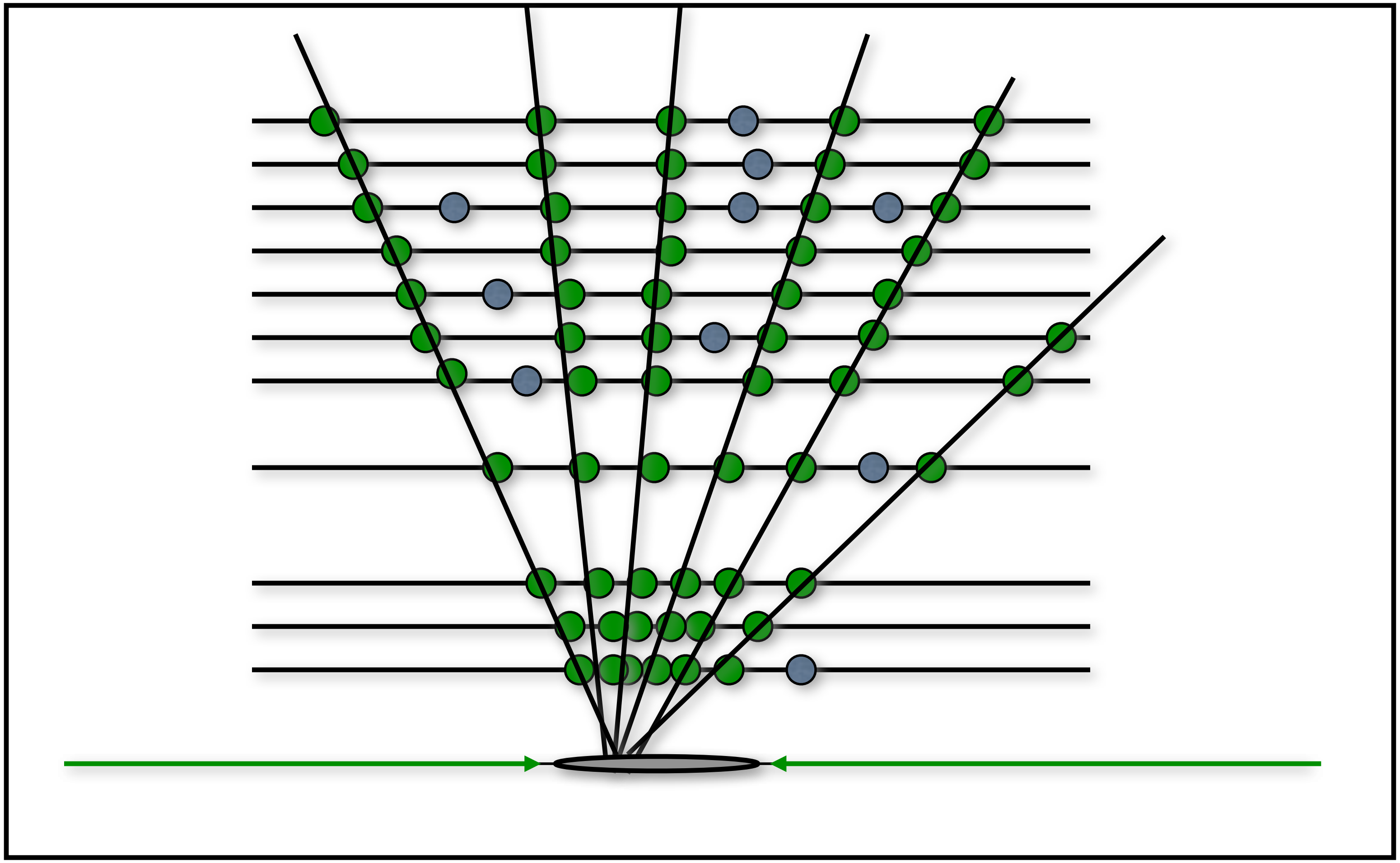
- *As each hit is added, the accuracy of the trajectory measurement increases*
- *The best accuracy during trajectory building will be for the outermost state, which is using the information of all the hits*
- *In general, we want to know the track parameters at the point of production, or the point of closest approach to the primary interaction*
- *But the uncertainty on the trajectory at the innermost layer has not been updated*

Track Fitting

- *To obtain the final track parameters, a "smoothing" step is run, where the track is extrapolated in-out and then back out-in, so that the optimal accuracy is obtained on all the layers*



Reconstructed Tracks in the Event



Each seed is tested in turn to reconstruct the other tracks in the event

Steps of Track Reconstruction

Track Filtering:

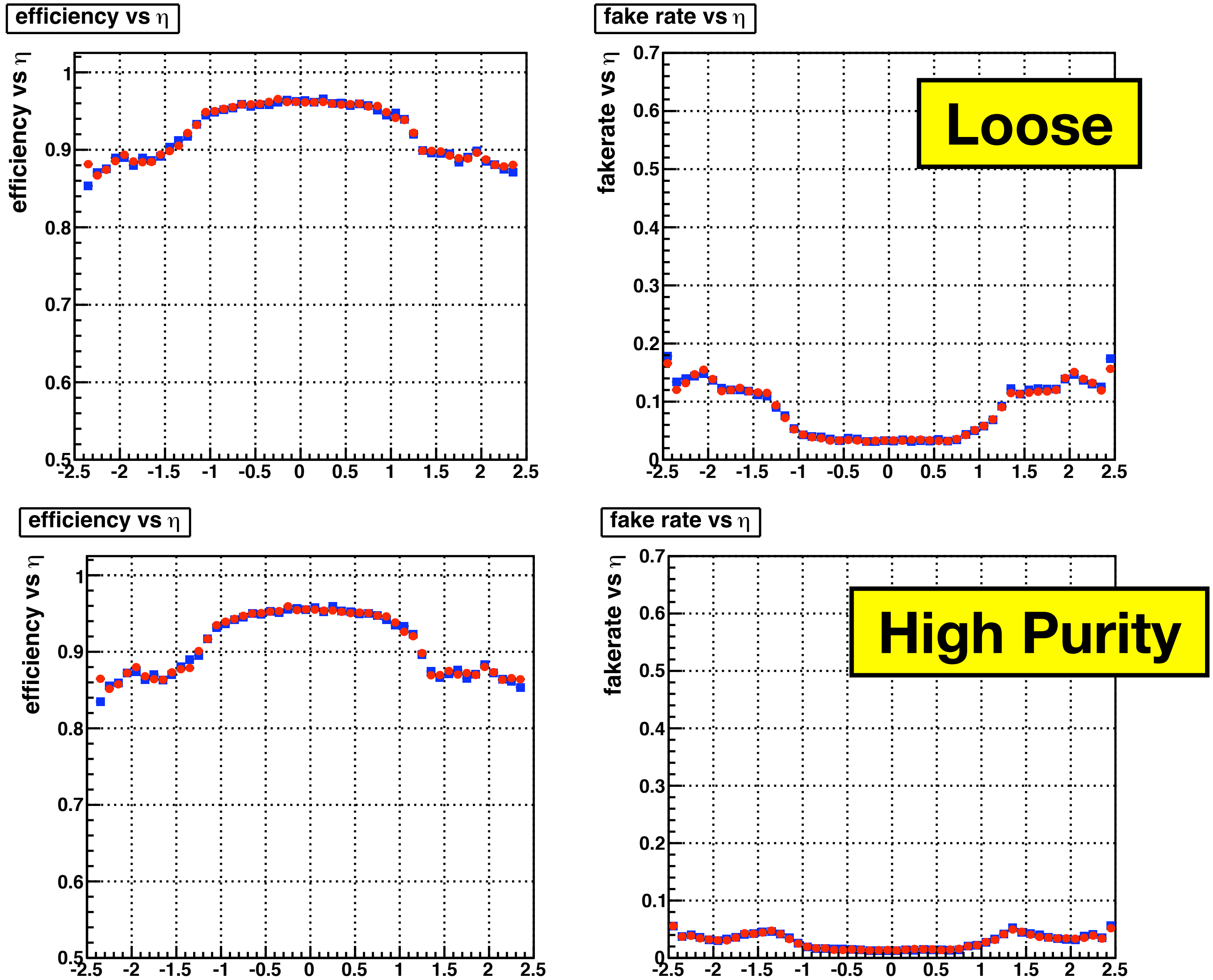
- *After all tracks have been reconstructed, we apply filters to remove tracks that have a high probability of being fake*
- *The general philosophy is that if a track has more hits, you can apply looser cuts*
- *Cuts are applied to d_{xy}, $d_{xy}/\sigma(d_{xy})$, d_z, $d_z/\sigma(d_z)$, χ^2*
- *There are actually three sets of cuts applied:*
  - They are labeled “loose”, “tight”, and “high purity”
  - Tracks that fail loose cuts are dropped
  - Tracks that pass tight or high purity cuts have that recorded in a “track quality” variable

Steps of Track Reconstruction

Track Filtering: Why three sets of cuts?

* *The general idea is to let the user decide the appropriate level of cuts for their analysis*
* *If tracks don't even pass loose cuts, you don't want to use them*
* *Then, if your analysis has little background, but depends on a very pure sample, you would ask for highPurity tracks*
* *If your analysis is using a very pure sample, but looking for something rare, then you could ask for loose tracks*

Steps of Track Reconstruction

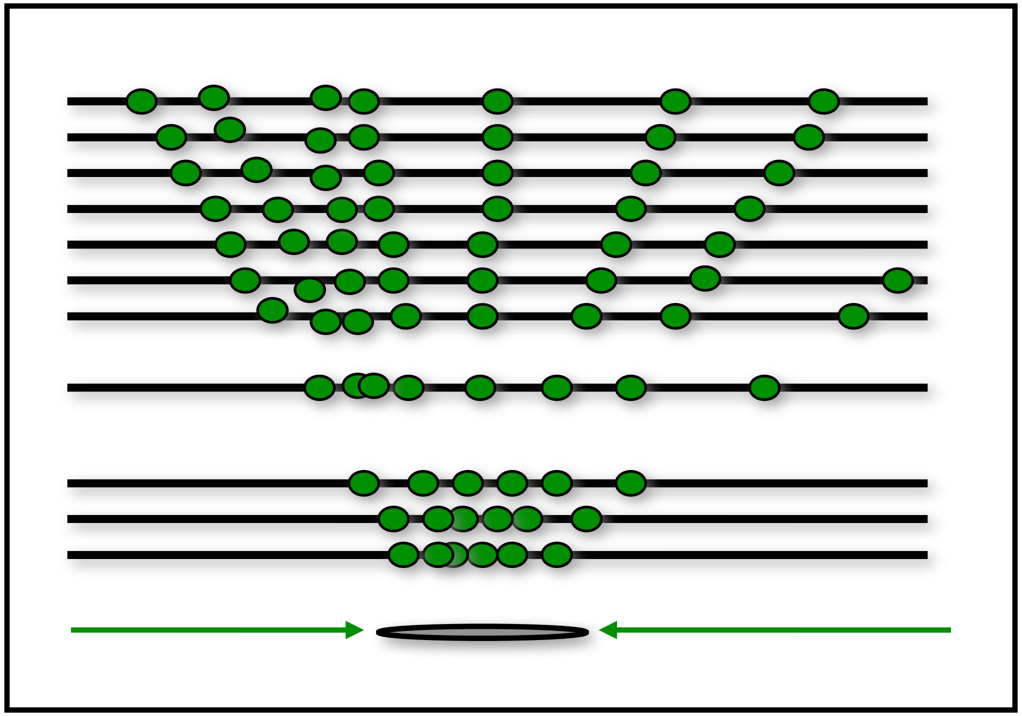


Iterative Tracking

A recent improvement in track reconstruction is the use of multiple iterations of track finding

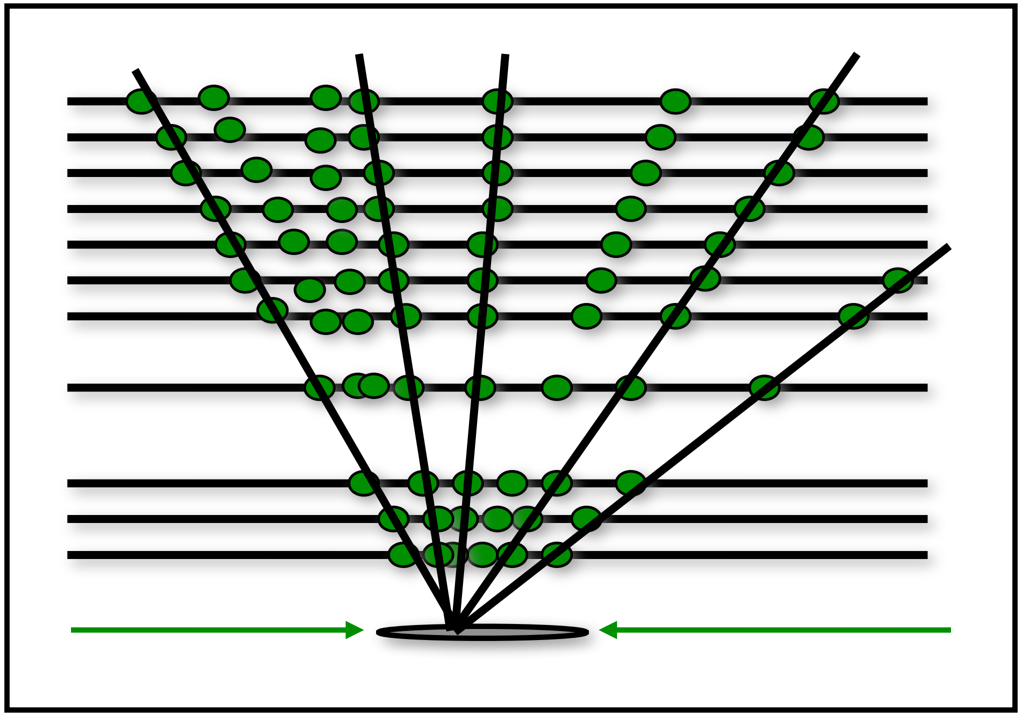
* *Start with the full collection of clusters*
* *Apply tight requirements in finding higher P_T primary tracks*
* *Remove the clusters associated with the hits on the found tracks to create a new collection of hits (clusters)*
* *Repeat the pattern recognition, this time with looser cuts to find lower P_T tracks or tracks not from the primary interaction*
* *Each pass of pattern recognition is just like the one already described (seed finding, trajectory building....)*

Iterative Tracking



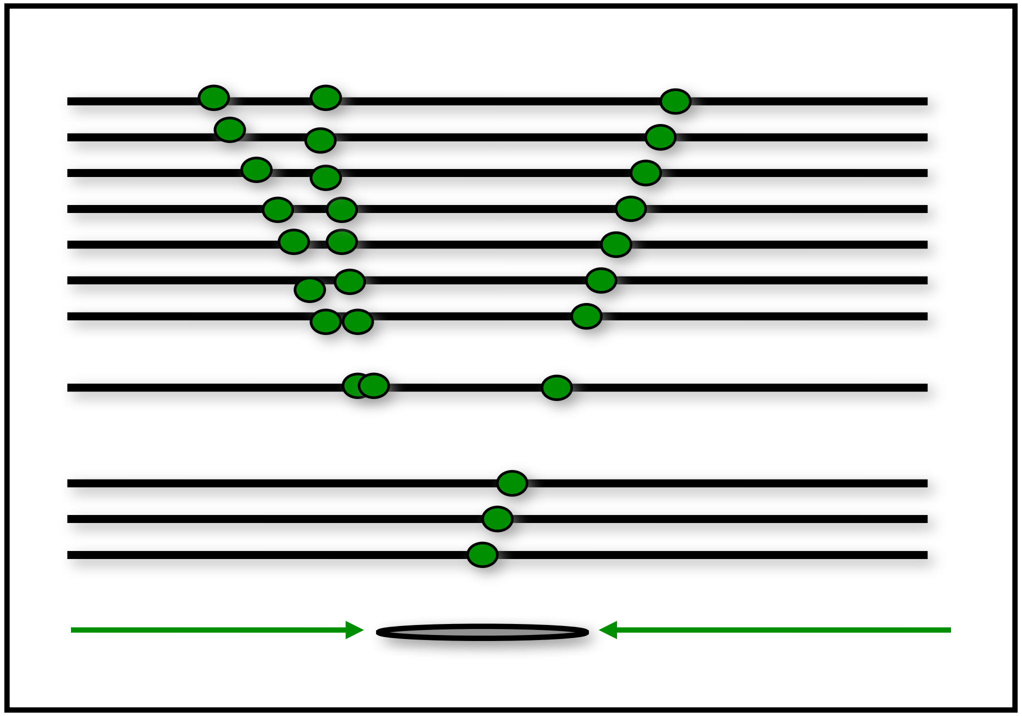
Start with the initial collection of hits in the event

Iterative Tracking - First Pass



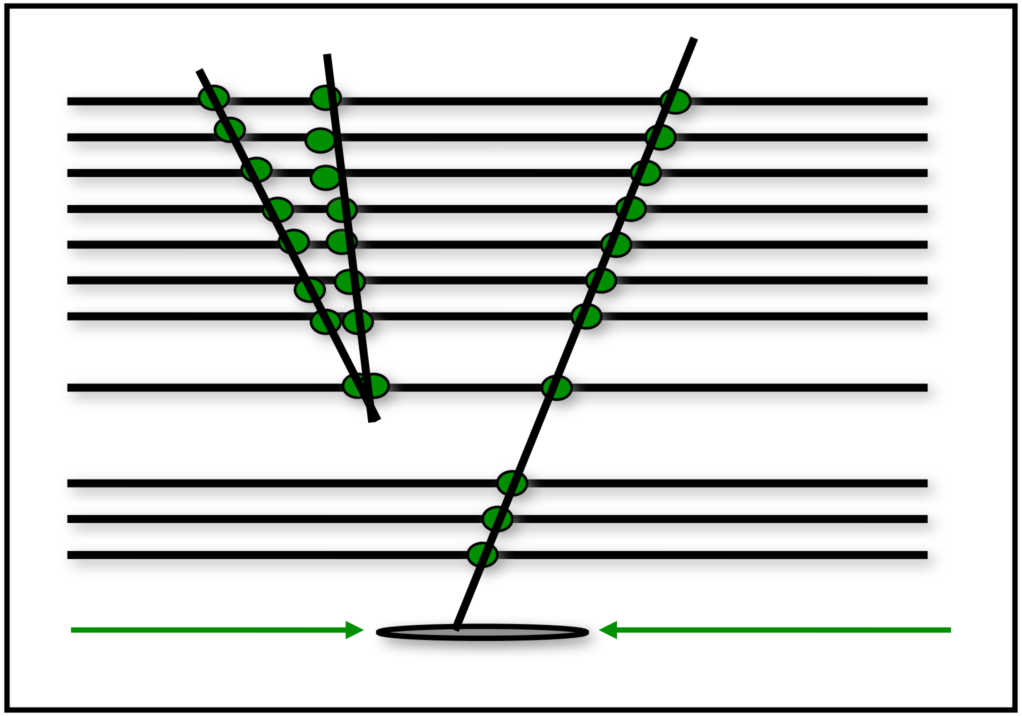
Find the first set of tracks - high P_T primary tracks

Iterative Tracking - Removal of Used Hits



Remove the hits on the first set of tracks to create a new hit collection

Iterative Tracking - Second Pass



Find the next set of tracks - lower P_T or not primary

Can be repeated for several iterations and final collection will include tracks from all iterations

Iterative tracking

Cuts in seeding and track building parameters for the iterative tracking steps.

Iter-ation	Seeds	P_T cut (GeV)	d_{xy} cut (cm)	d_z cut (cm)	Min hits	Max lost hits
0	pixel triplets	0.5	0.2	15.9	3	1
1	pixel pairs	0.9	0.2	0.2	3	1
2	pixel triplets	0.075	0.2	17.5	3	1
3	pixel pairs	0.35	1.2	7.0	4	0
4	TIB, TID, TEC	0.5	2.0	10.0	7	0
5	TOB, TEC	0.8	5.0	10.0	7	0

Iterative tracking

Iterations 0,1: Primary tracks of medium P_T

Iter-ation	Seeds	pT cut (GeV)	d0 cut (cm)	dz cut (cm)	Min hits	Max lost hits
0	pixel triplets	0.5	0.2	15.9	3	1
1	pixel pairs	0.9	0.2	0.2	3	1
2	pixel triplets	0.075	0.2	17.5	3	1
3	pixel pairs	0.35	1.2	7.0	4	0
4	TIB, TID, TEC	0.5	2.0	10.0	7	0
5	TOB, TEC	0.8	5.0	10.0	7	0

Iterative tracking

Iteration 2: Low P_T primary tracks

Iter-ation	Seeds	pT cut (GeV)	d0 cut (cm)	dz cut (cm)	Min hits	Max lost hits
0	pixel triplets	0.5	0.2	15.9	3	1
1	pixel pairs	0.9	0.2	0.2	3	1
2	pixel triplets	0.075	0.2	17.5	3	1
3	pixel pairs	0.35	1.2	7.0	4	0
4	TIB, TID, TEC	0.5	2.0	10.0	7	0
5	TOB, TEC	0.8	5.0	10.0	7	0

Iterative tracking

Iteration 3: Non-prompt tracks (b, tau, etc.)

Iter-ation	Seeds	pT cut (GeV)	d0 cut (cm)	dz cut (cm)	Min hits	Max lost hits
0	pixel triplets	0.5	0.2	15.9	3	1
1	pixel pairs	0.9	0.2	0.2	3	1
2	pixel triplets	0.075	0.2	17.5	3	1
3	pixel pairs	0.35	1.2	7.0	4	0
4	TIB, TID, TEC	0.5	2.0	10.0	7	0
5	TOB, TEC	0.8	5.0	10.0	7	0

Iterative tracking

Iterations 4,5: Detached tracks (V^0s, conversions)

Iter-ation	Seeds	pT cut (GeV)	d0 cut (cm)	dz cut (cm)	Min hits	Max lost hits
0	pixel triplets	0.5	0.2	15.9	3	1
1	pixel pairs	0.9	0.2	0.2	3	1
2	pixel triplets	0.075	0.2	17.5	3	1
3	pixel pairs	0.35	1.2	7.0	4	0
4	TIB, TID, TEC	0.5	2.0	10.0	7	0
5	TOB, TEC	0.8	5.0	10.0	7	0

Iterative Tracking

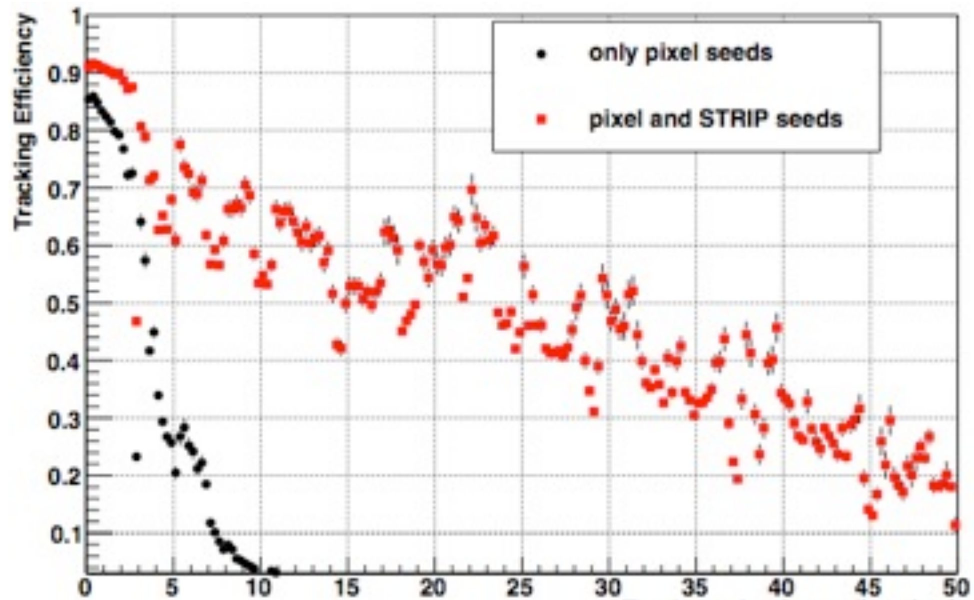
<u>*Track Filtering (again)*</u>

* *Track filtering is an important part of iterative tracking*
* *We filter tracks at the end of each step*
* *We don't want to remove hits on fake tracks from the collection, otherwise they are not there to be found in a later iteration by the track that they truly belong to*
* *So we only remove the hits on high purity tracks*
* *This implies that sometimes a "loose" track is found later*
* *At the end of each step, when we merge the new tracks with the ones from previous steps, being careful to remove duplicates*

The final set of tracks (the ones you should be using for analysis) is called ***generalTracks***

Performance with Iterative Tracking

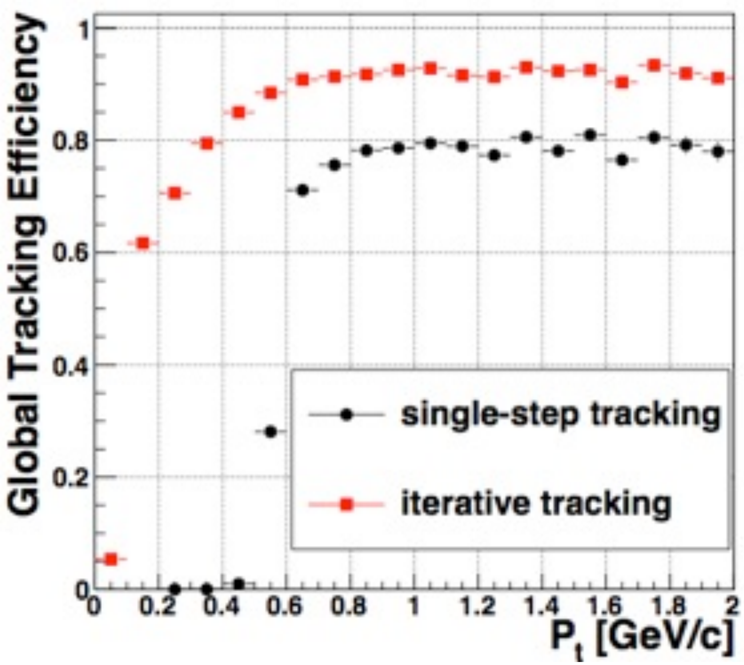
The use of multiple iterations in tracking has been important for improving efficiency at low P_T and large impact parameter

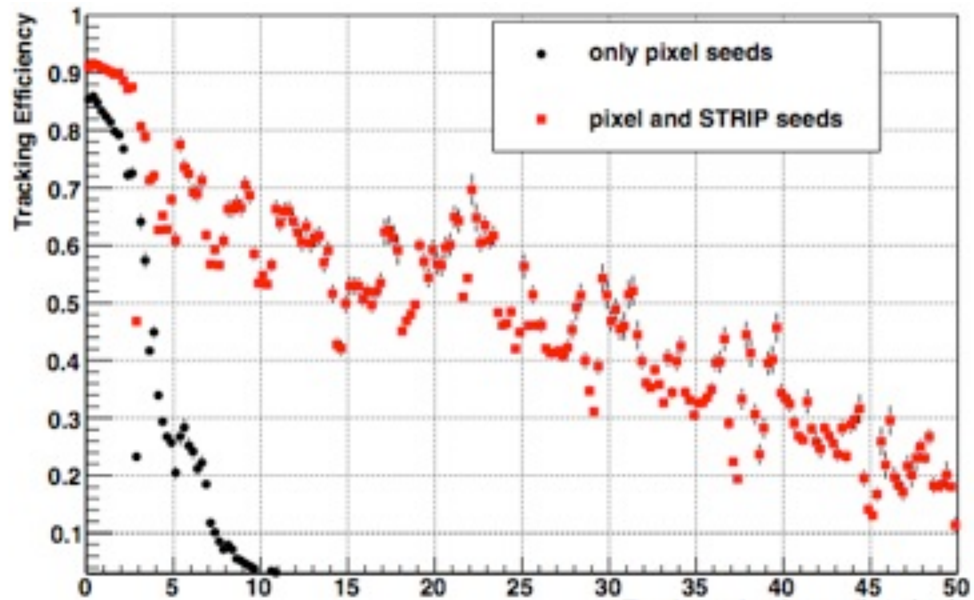


Distance R from z-axis [cm]

Performance with Iterative Tracking

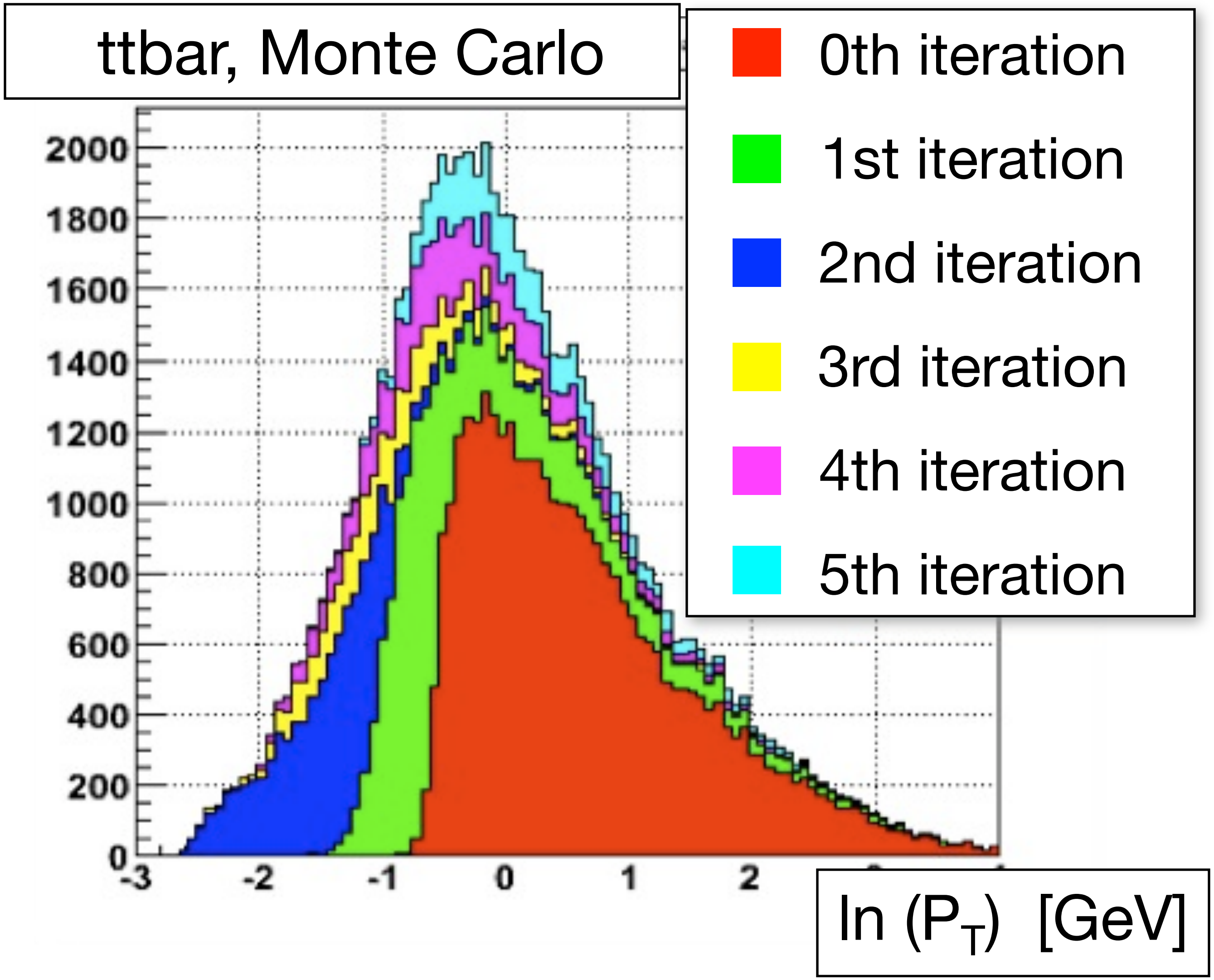
The use of multiple iterations in tracking has been important for improving efficiency at low P_T and large impact parameter



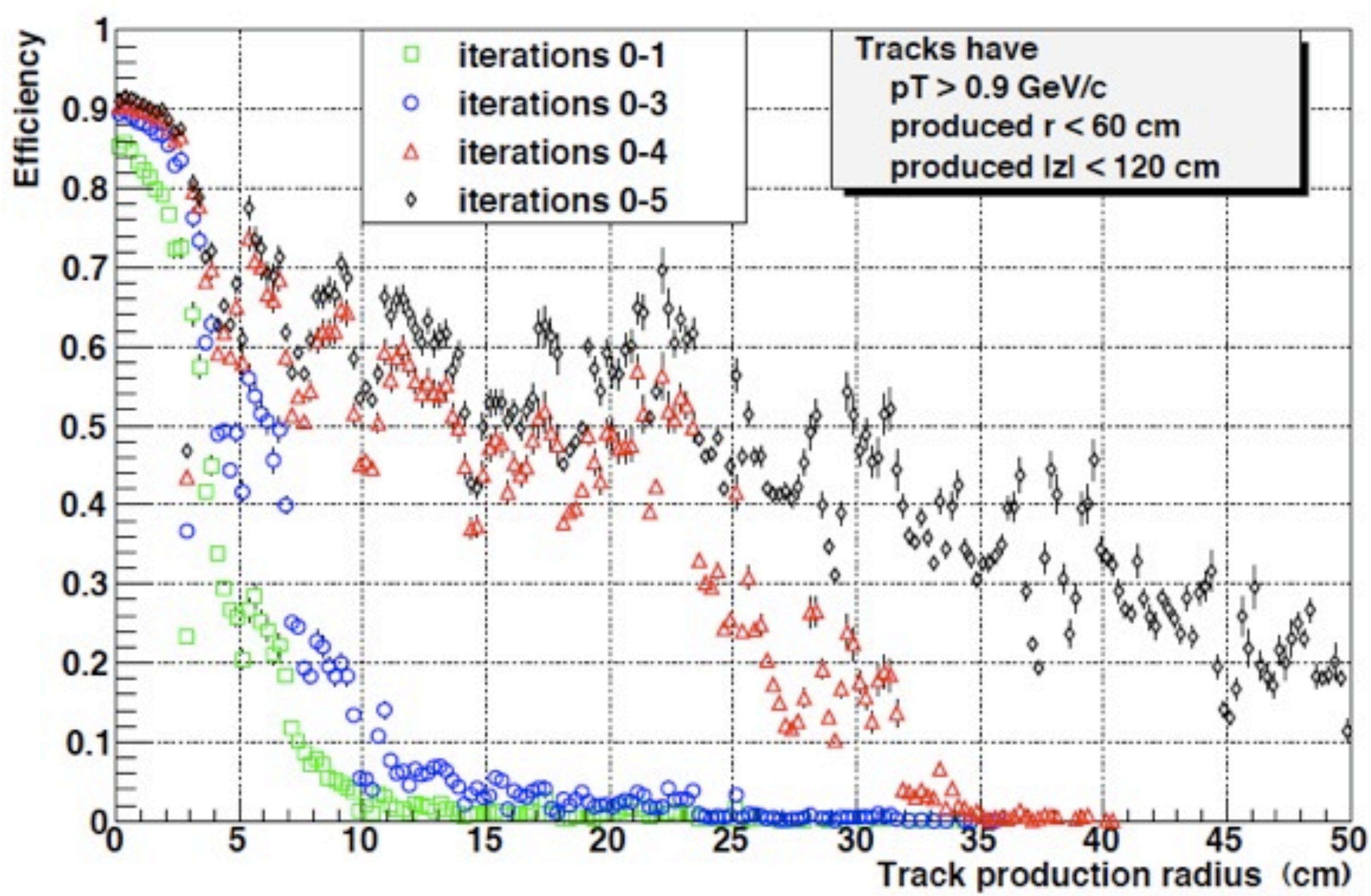


Distance R from z-axis [cm]

Tracks Found by Each Iteration



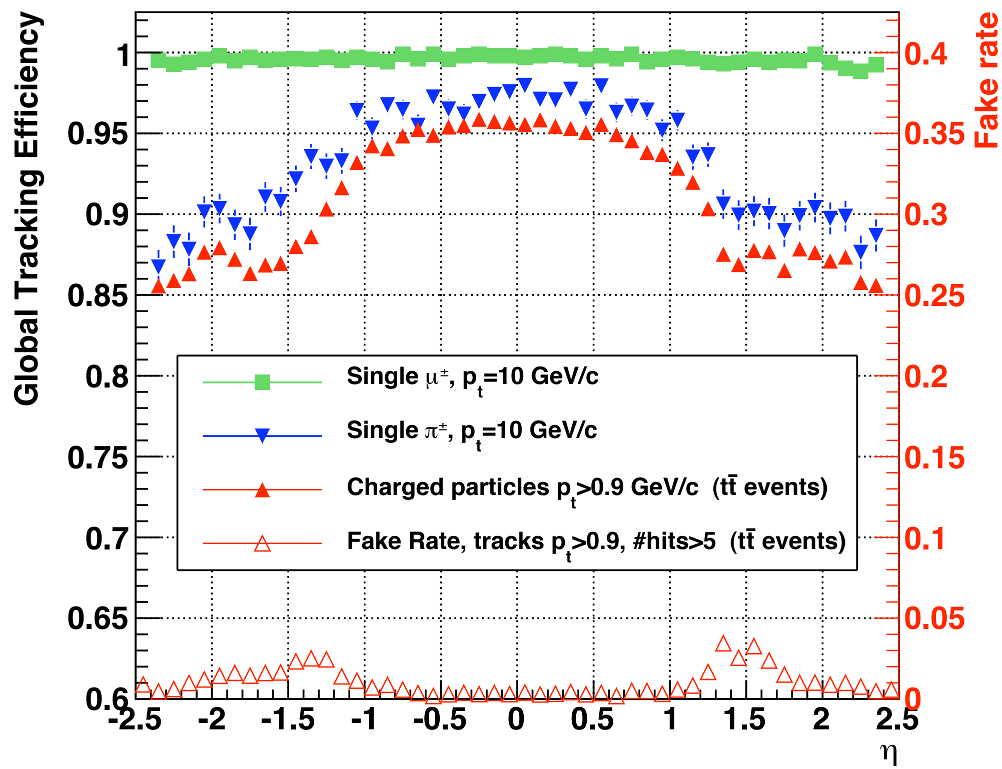
Reconstruction of Displaced Tracks



Tracking Efficiency Summary

- Efficiency for muons is nearly 100%
- Efficiency for hadrons in the central region is >95%, with some decrease in the forward
- Most of inefficiency for hadrons is due to particles that have nuclear interactions before crossing three layers

✓ Meets the requirements described earlier that are necessary to do the physics we want to do



Measuring Tracking Performance in Data

Measuring Tracking Performance in Data

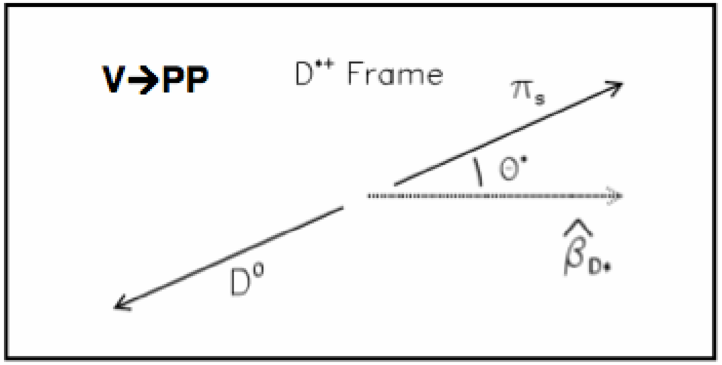
- *Once tracking has been validated in the collision data, we will have to turn our attention to measuring tracking performance*
    - Tracking efficiency, for both muons and hadrons
    - Momentum scale, momentum resolution
    - Impact parameter resolution
    - Vertex resolution
    - Tracker material budget
- *These are quantities that are necessary for some of the first physics papers*

Tracking Efficiency

- *Ideally the tracking efficiency could be measured with as we do with cosmics, using standalone muons. However, the relatively poor standalone muon resolution makes this difficult. This method may also not go low enough in P_T*
- *In addition, tracking efficiency should be measured in several ways to cover multiple operating points, from high P_T isolated muons down to lower P_T non-isolated hadrons*
- *In addition to using standalone muons, there are other methods proposed for measuring tracking efficiency, including for low P_T, non-isolated hadrons*

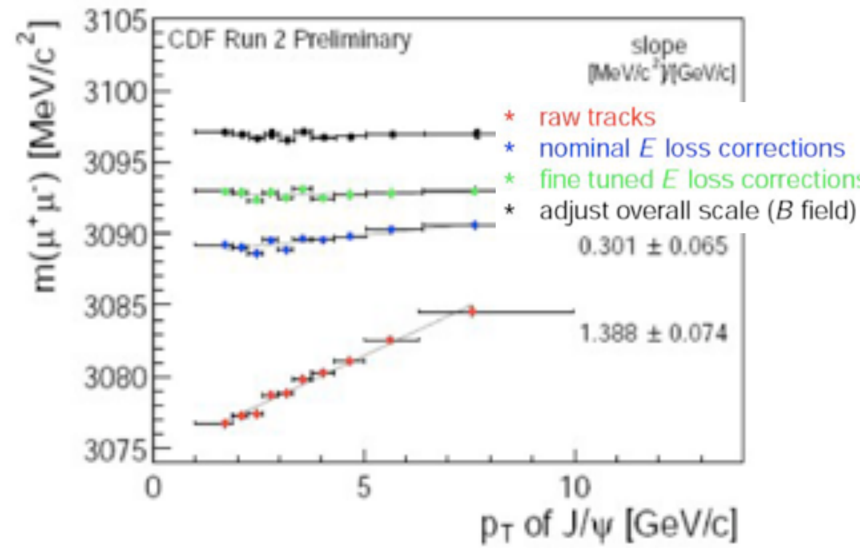
Tracking Efficiency

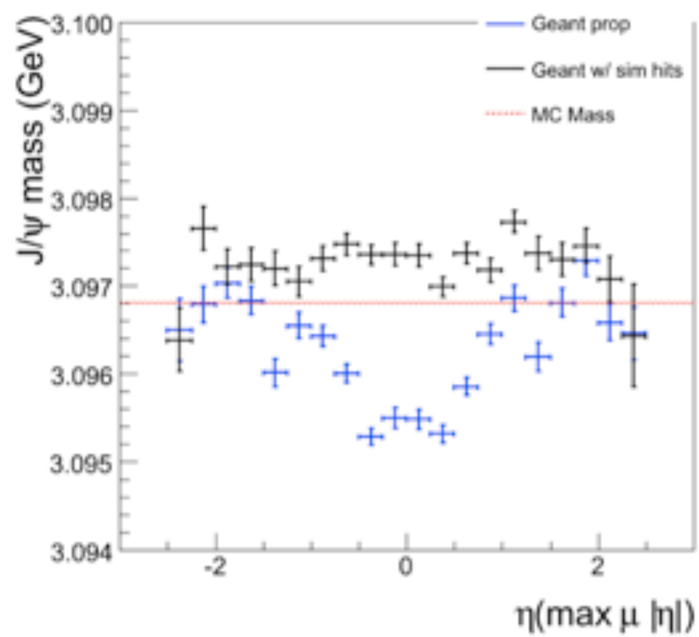
- *Track Embedding:*
  - Take a reconstructed track from one event, embed the hits in another event (e.g., QCD dijets), and see if you can reconstruct the original track
- *Tracking efficiency starting from pixel triplets*
  - Given a pixel triplet, do you find a track?
  - Need a pure sample of pixel triplets, plus the efficiency for finding the original pixel triplet
- *Measure efficiency from slow pion reconstruction using known helicity angle distribution in the decay $B^0 \rightarrow D^* l \nu$, $D^* \rightarrow D^0 \pi$*



Momentum Scale

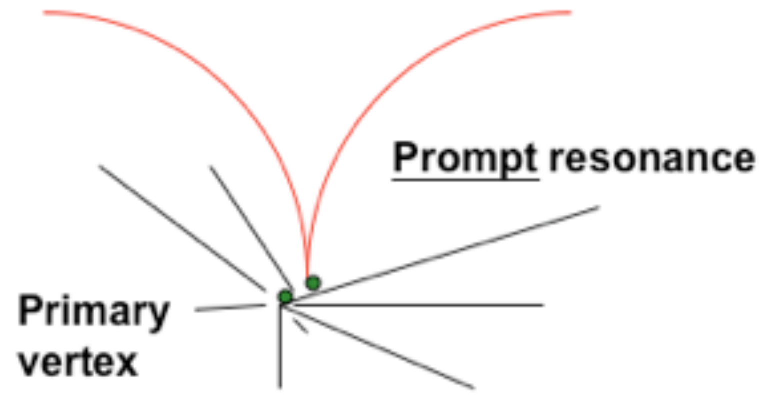
- *Study reconstructed resonance masses (K^0_s, J/ψ, ϒ, Z) vs P_T or other kinematic variables*
- *Try to correct for any shape with improved energy loss corrections*
- *Final step sets momentum scale to obtain correct mass*

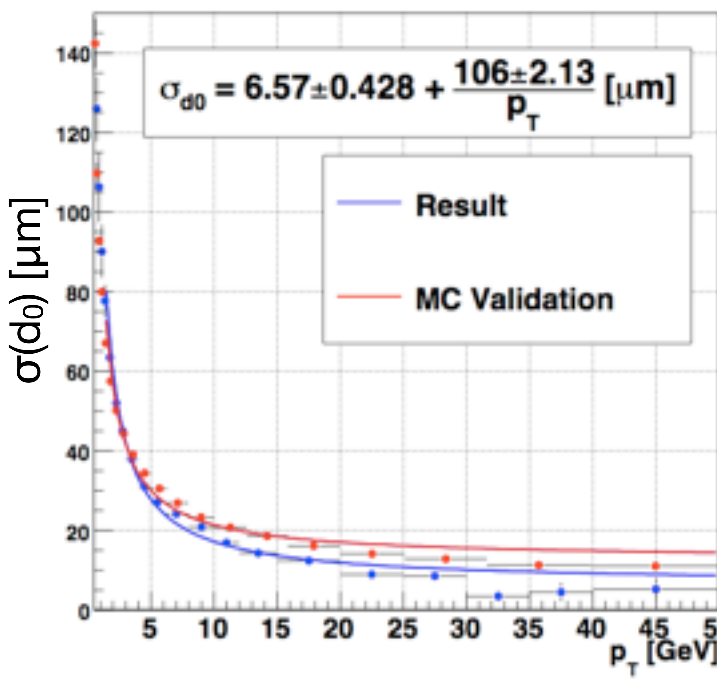




Impact Parameter Resolution

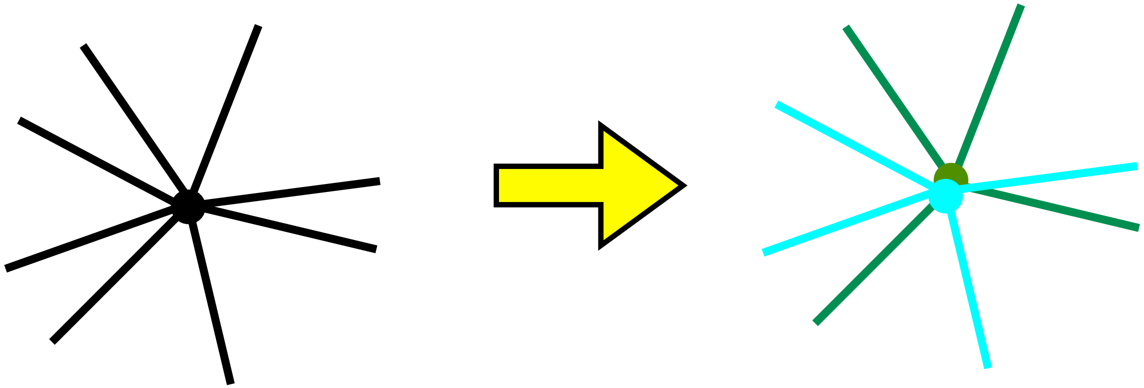
- *Measure resolution using prompt resonances*
  - Reconstruct ϒ→μμ, Z→μμ
  - Study impact parameters of other tracks in the event relative to the dimuon vertex
  - Study muon track impact parameter vs. primary vertex
- *Now testing if this can be extended to use all tracks with respect to the primary vertex*
  - Requires much less luminosity
  - Tracks under study should be excluded from the primary vertex fit



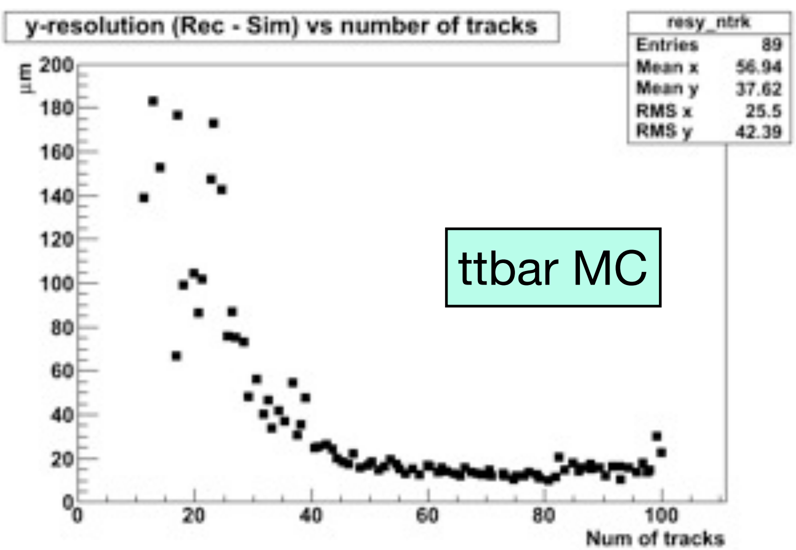


Primary Vertex Resolution

* *To measure primary vertex resolution, split tracks into two sets in the same event and fit two primary vertices*

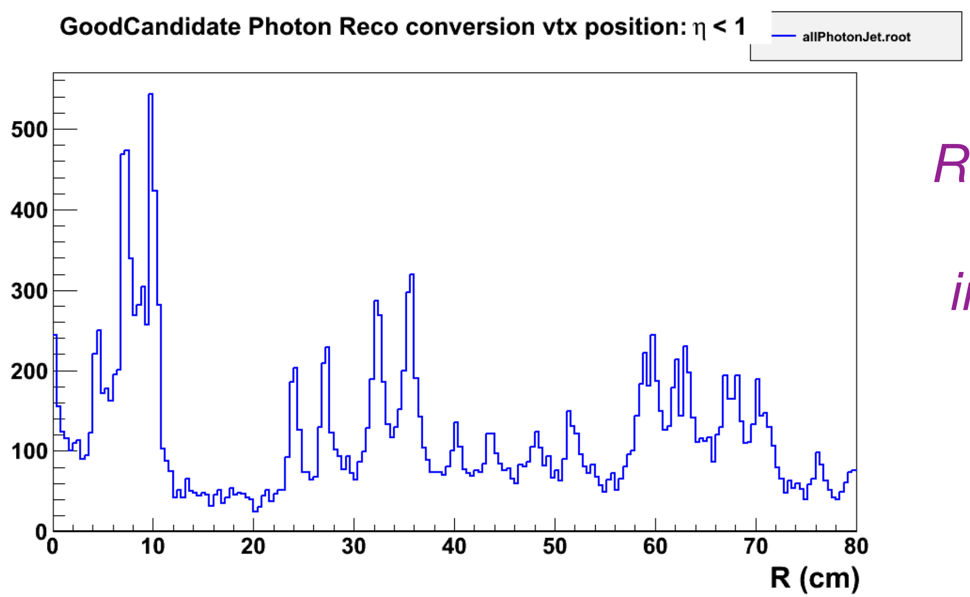


* *Difference in vertex positions yields resolution (vs. # tracks)*
    - Resolution depends strongly on # tracks in vertex
    - Same technique can be used to study PV efficiency



Tracker Material Budget

* *Measuring the material budget is an important issue, given the significant material in the tracker. Two general approaches are under development:*
  - Use of reconstructed conversions to determine location and amount of material



Resolution on vertex position can be improved with new fitters

  - Layer-by-layer multiple scattering to determine material

Summary

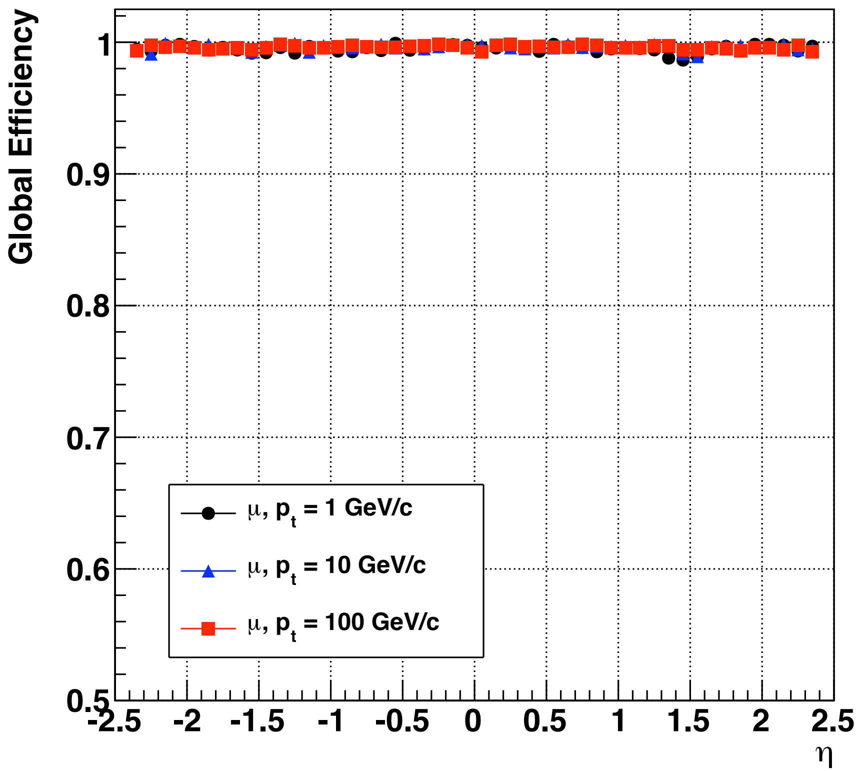
- *Tracking is a powerful tool for doing physics at CMS*
- *The current tracking performs quite well on Monte Carlo, reconstructing tracks down to low P_T and large impact parameter with good efficiency*
- *But there is still a lot of work to do to commission tracking with data, and we are excited to get started*

- *If you have trouble, you can send questions to the hypernews forum: hn-cms-tracking@cern.ch*
- *Or, if you want to learn more, come to the Tracking Meetings every other Monday at 1630 GVA*

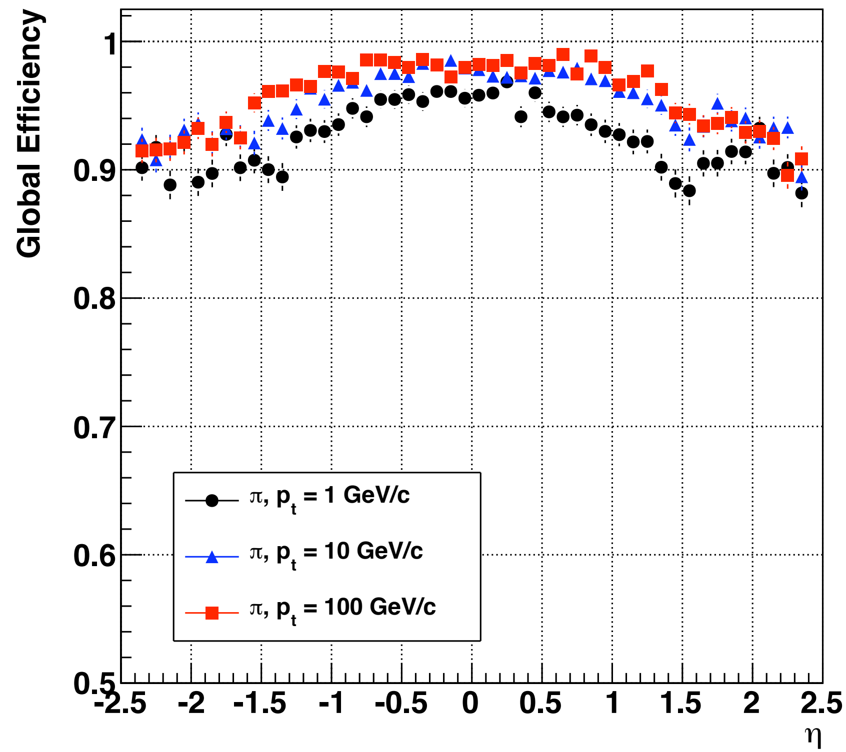
BACKUP

Tracking Efficiency in Monte Carlo

Single μ

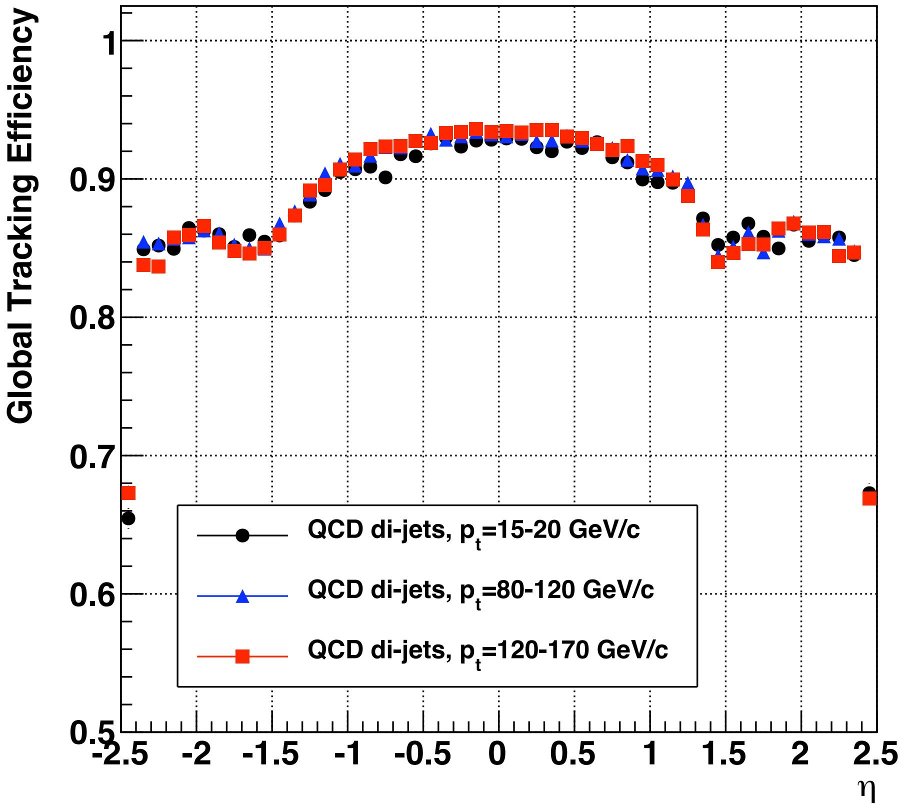


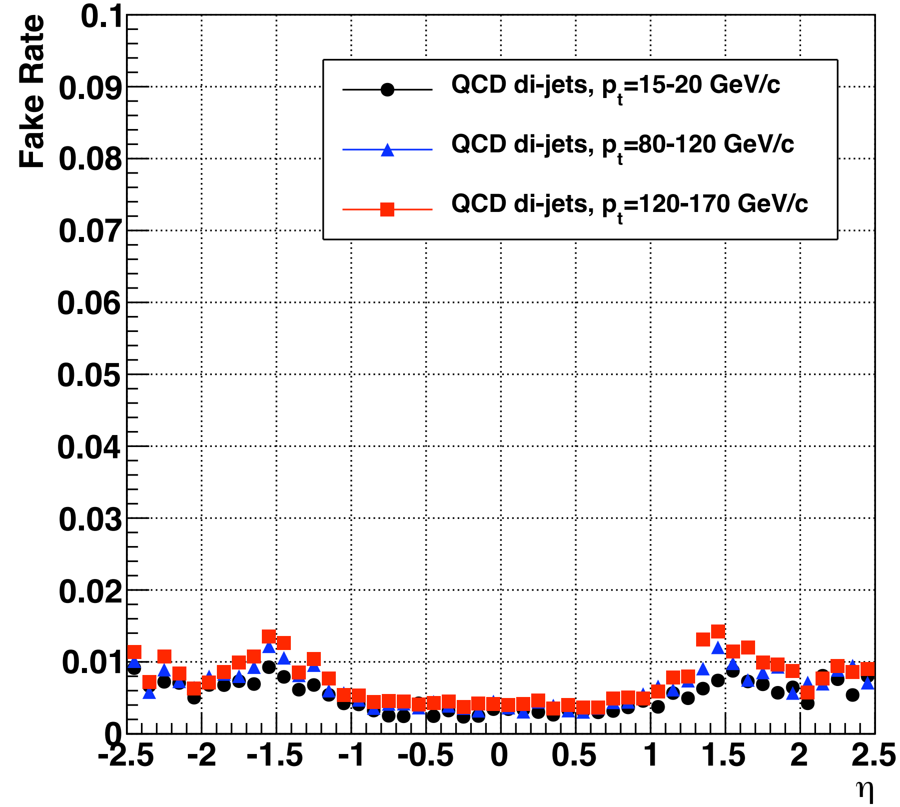
Single π



- *Single muon efficiency in essentially 100%*
- *For pions, there is a loss of efficiency due to particles that interact in the tracker material before crossing enough layers to be reconstructed*

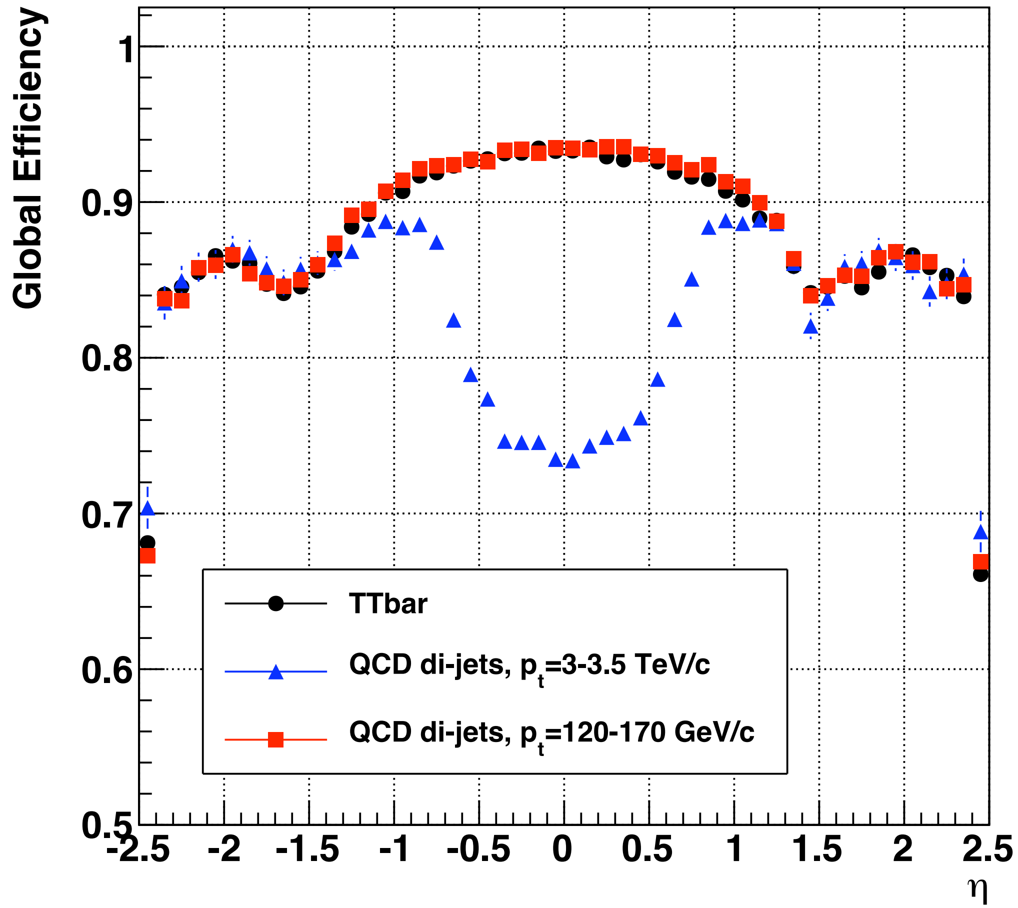
Tracking Efficiency in Monte Carlo





- *In QCD events, tracking efficiency follows the single π efficiency*
  - Efficiency is above 90% in the central region, with a drop in the forward region where there is more material in the tracker
- *Fake rate < 0.5% in barrel with small increase in forward*

Tracking Efficiency in Monte Carlo



For very high E_T (here 3-3.5 TeV) jets, we see a drop in efficiency in the core of the jets due to overlapping tracks and merged clusters in the tracker