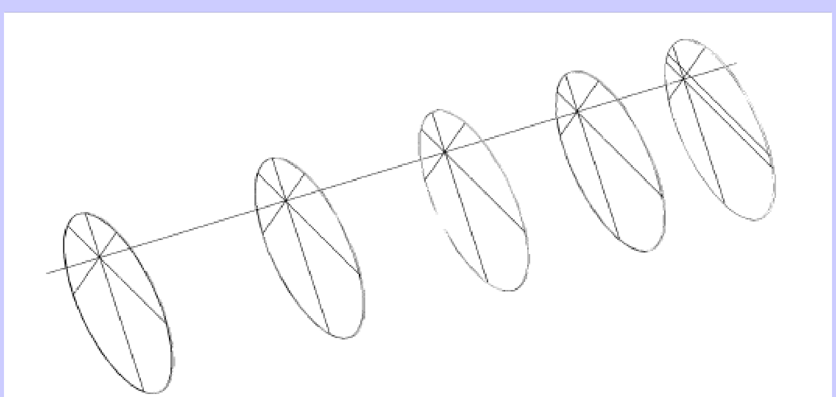
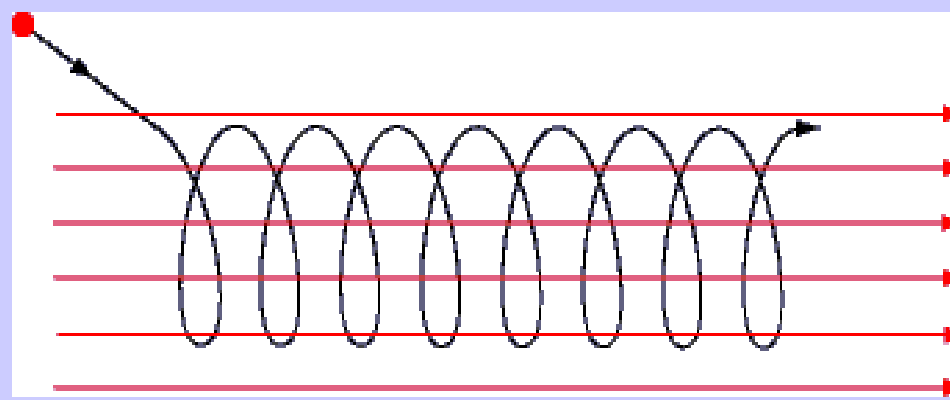
Pattern Recognition

Space Points

Initial track parameters for fitting

- We consider either:
  - Straight trajectories, in the absence of an external magnetic field; or
  - Helical trajectories, when the trackers are placed within superconducting solenoid magnets



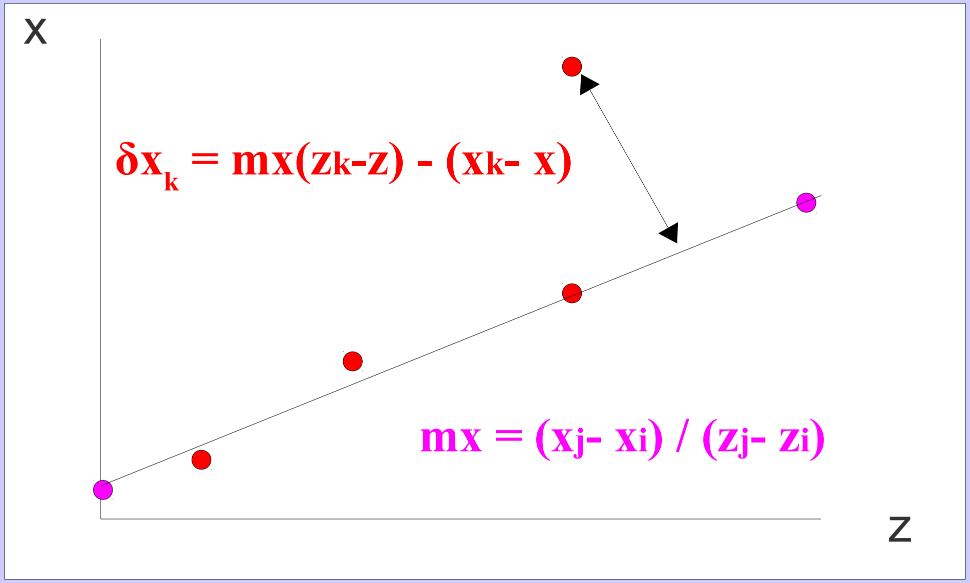


Straight Tracks

- Use to analyze the cosmic ray data
  - No external B → particles will follow straight trajectories
- Straight track PR is a ***three step process***

1) Form initial line between first and last space points

2) Calculate the deviation of intermediate station space points from this line



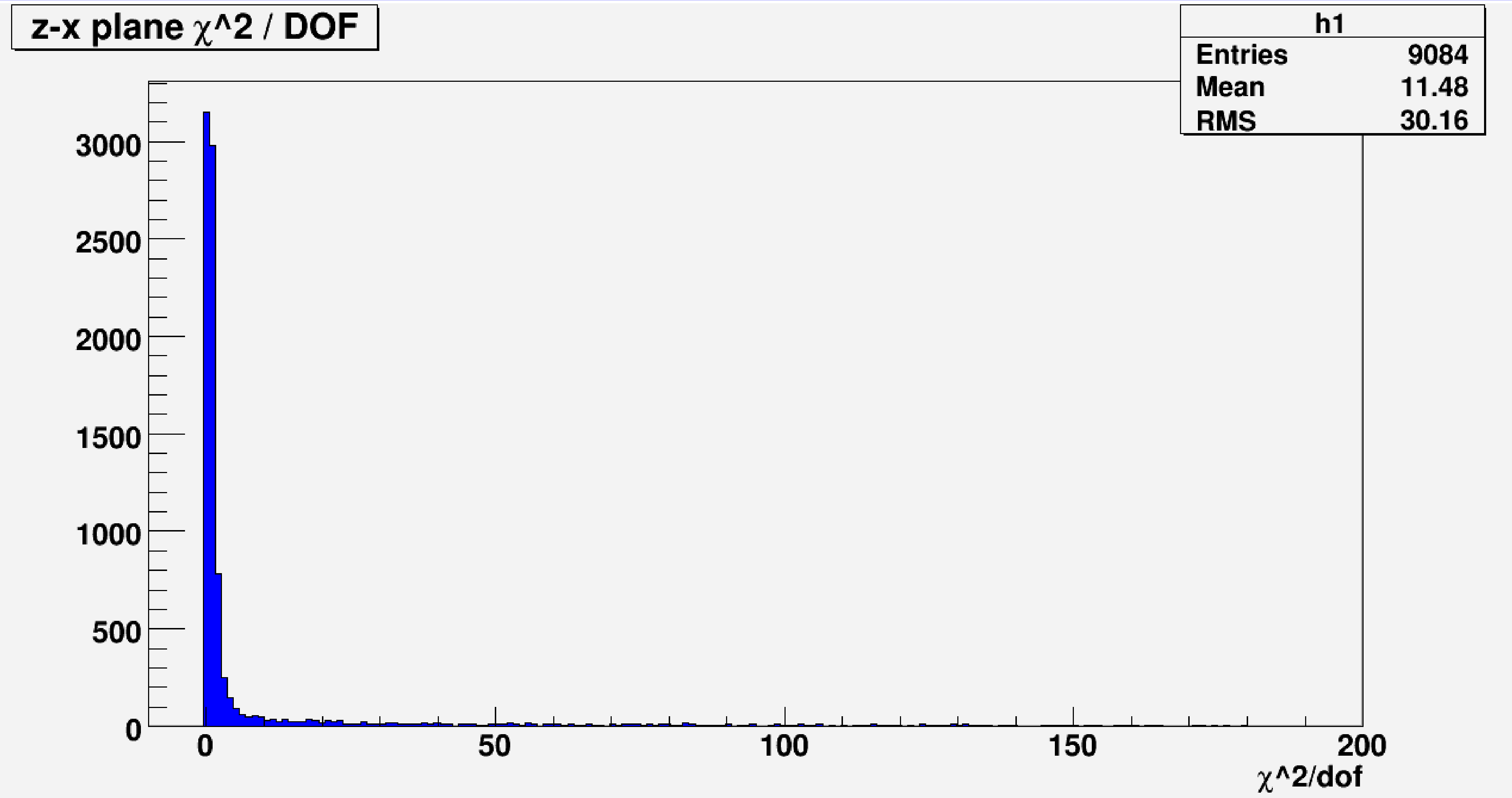
3) If (δx_k && δy_k < cut) for ALL intermediate space points
→ Do a linear least squares fit, append points and parameters to track object

- Cut = 10mm
- Same is done in z-y projection

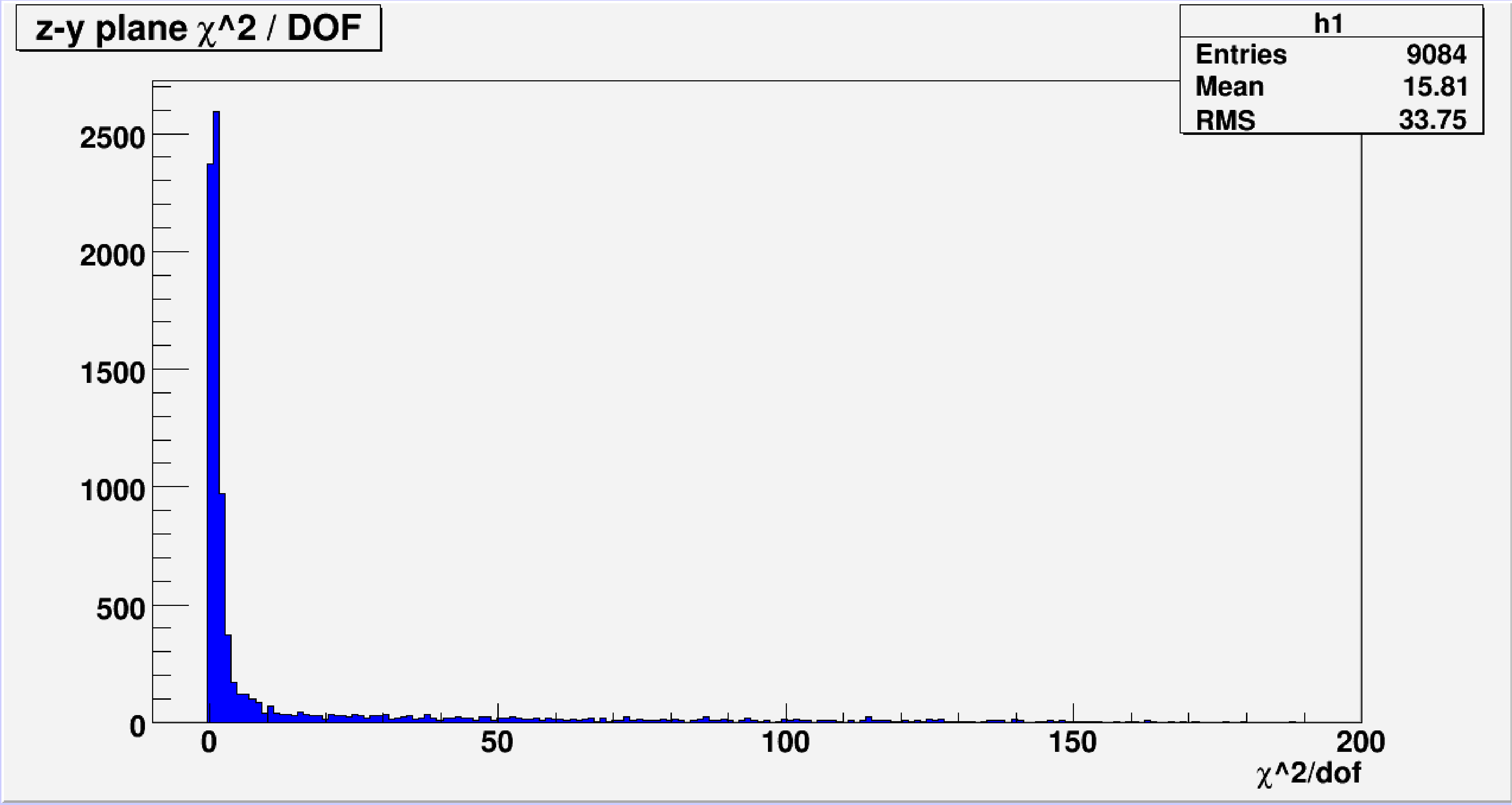
Testing on Real Data

- Out of 7619 events (64MB)...
  - 5 point tracks: 1816
  - 4 points tracks: 1351
  - 3 point tracks: 1375
- CPU time: 1 min 11.69 sec
- I think it works well, but can't be sure without MC to test on.
- We could still make another cut on the $\chi 2$/dof value after the fit, but this still needs to be determined

$\chi 2$ Distribution for z-x fit



$\chi2$ Distribution for z-y fit



$\chi 2$ Distributions

- Large $\chi 2$ values common to all #-point tracks, but most come from 3-point tracks
- Should we lower the track cut limit? (currently a box of 10x10mm)
- Need to decide on a $\chi 2$ cut eventually...

Helical Pattern Recognition

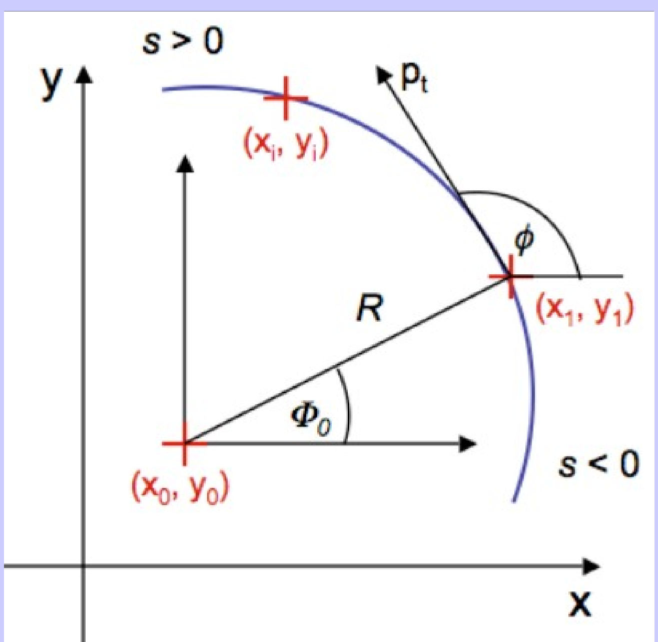
Helix Parameterization

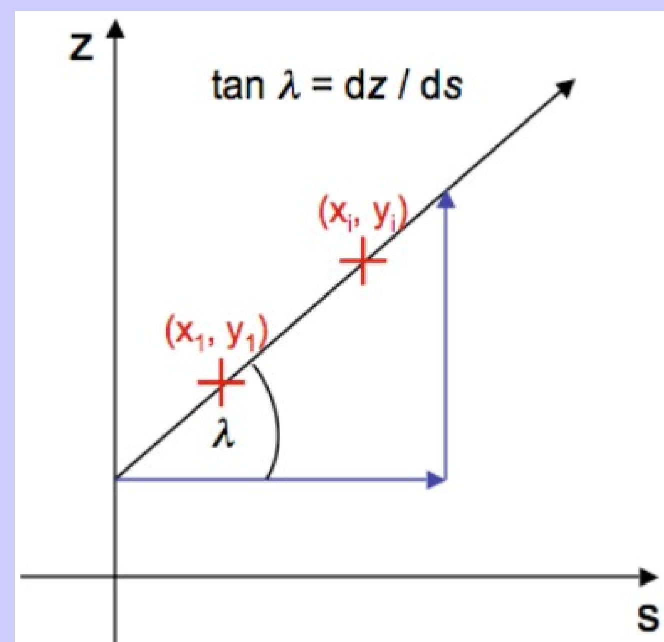
- **x1, y1, z1** are starting point of track
- **R**
- **tanλ**
- **h** is ±1, depending on charge
- **Φ0** is the azimuthal angle of the starting point w.r.t. helix axis
- **Ψ0** is the track direction given by $\Psi_0 = \Phi + h\pi/2$

$$x(s) = x_1 + R\left[\cos\left(\Phi_0 + \frac{hs\cos\lambda}{R}\right) - \cos\Phi_0\right]$$

$$y(s) = y_1 + R\left[\sin\left(\Phi_0 + \frac{hs\cos\lambda}{R}\right) - \sin\Phi_0\right]$$

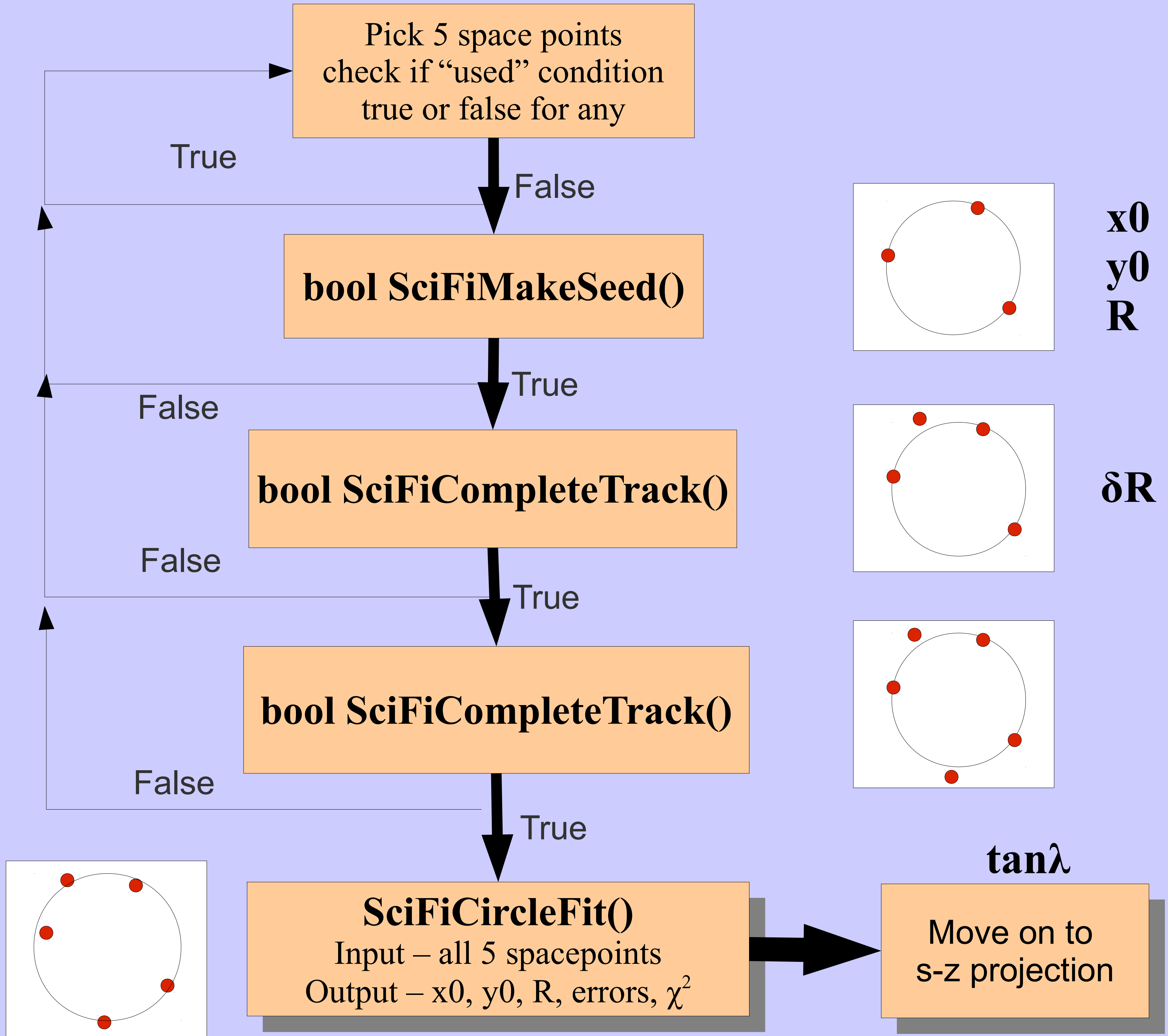
$$z(s) = z_1 + s\ tan\lambda$$





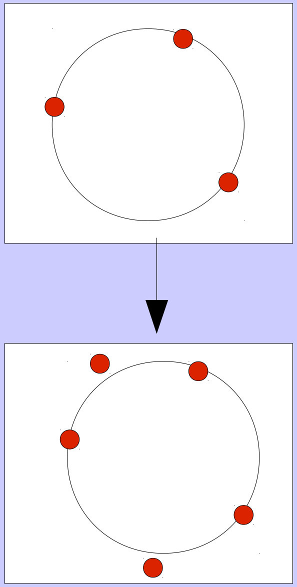
Helical Track PR

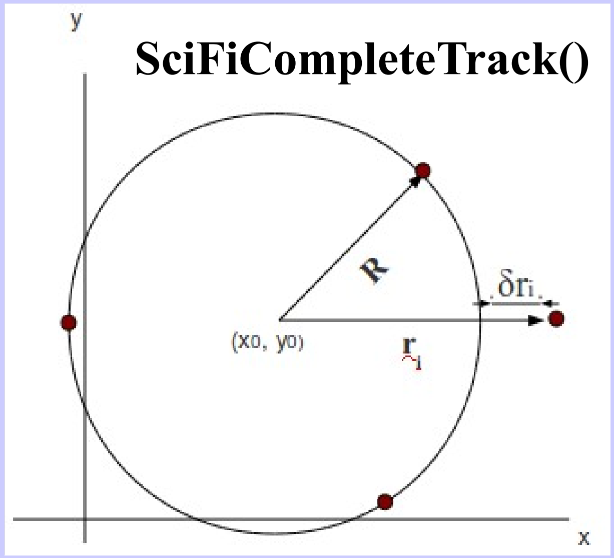
XY Logic



SciFiMakeSeed() and SciFiCompleteTrack()

- SciFiMakeSeed()
  - Input 3: space points
  - Output: x0, y0, R
- SciFiCompleteTrack()
  - Input: intermediation station space points
  - Output: true or false
  - Calculates the radial deviation of the added space point from the circle created from the seed
  - If this δR is < radial_cut , returns true.
    - Currently radial_cut == 25 mm
    - This is based on my own choice; needs to be optimized!





SciFiCircleFit()

- I worked with this parameterization: $$A(x^2+y^2)+Bx+Cy=1$$

Where

$$x_0=\frac{-B}{2A} \qquad y_0=\frac{-C}{2A}$$

$$R=\frac{\sqrt{4A+B^2+C^2}}{2A}$$

- 3 parameters (x_0, y_0, and R), three equations from taking derivatives of χ^2 w.r.t. each parameter

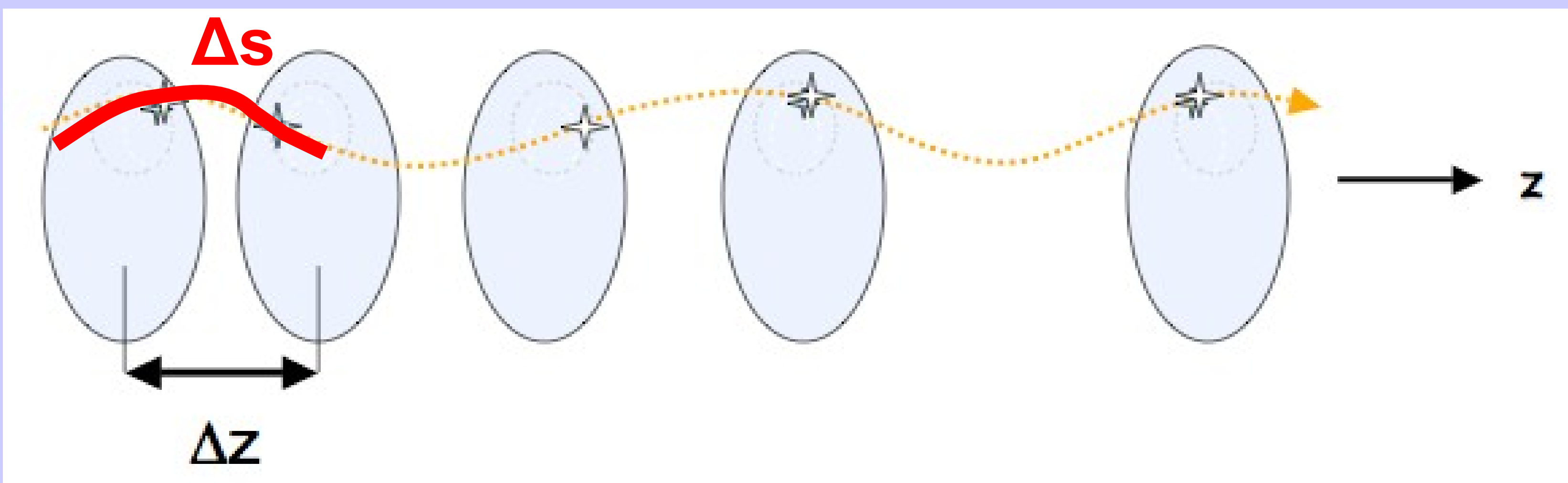
$$\chi^2=\sum_i^n \left(A(x_i^2+y_i^2)+Bx_i+Cy_i-1\right)^2$$

- However, I have found that doing this extra fit **doesn't** change x0, y0 or R much so we might want to scrap it....

S-Z Projection

SciFiCircleFit() → SciFiDetermineDip()

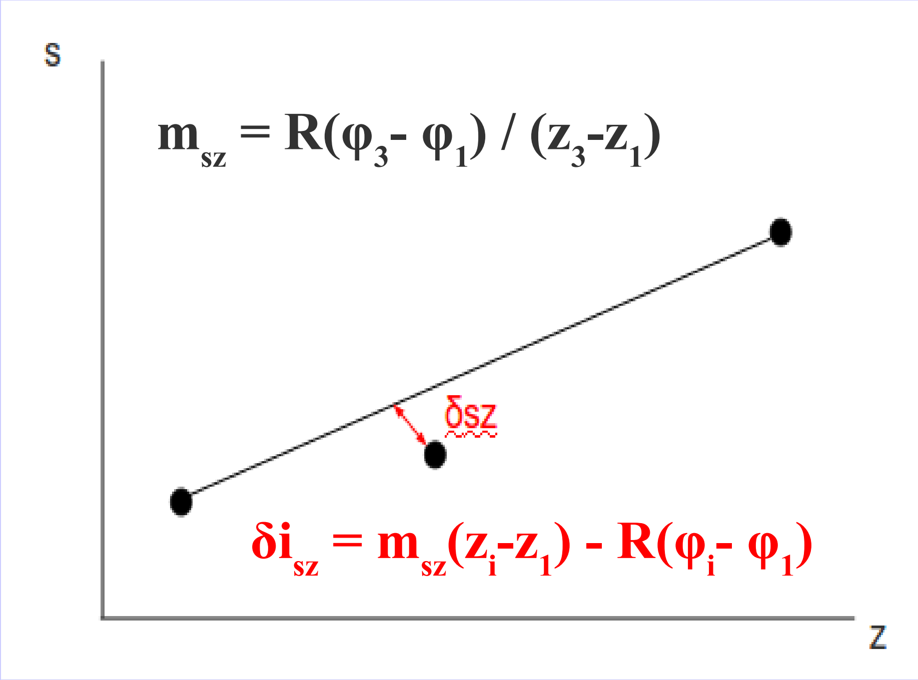
- ***s*** is defined as, **$s = R\Delta\Phi$** , or simply the ***integrated path length*** along the trajectory.



- Since momentum is conserved as the muon travels through the length of the detector, **we expect there to be a linear relationship between s and z.**

SciFiDetermineDip()

- Input: all space points that belong to a track
- Output: **dz/ds** (or **tanλ**)
- Note though that we must be careful when calculating Δsji
  - must take account of the fact that the particle may have gone around 2nπ between stations i and j, meaning Δφji is really **Δφji + 2nπ**



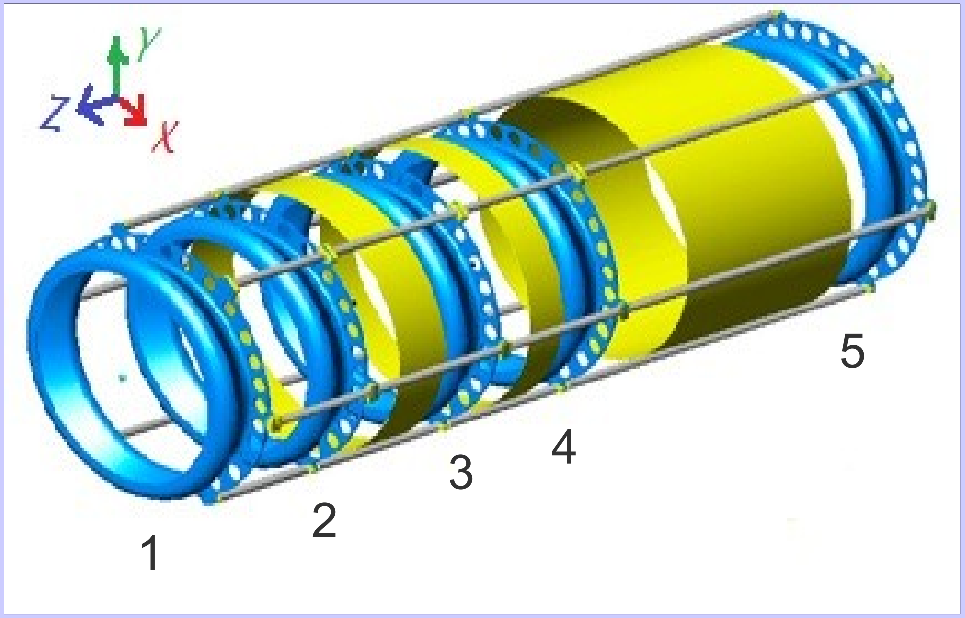
tanλ = dz/ds

Pz = Pt * tanλ

Turns Between Stations

- The 2π rotations can be determined because the trackers were built with different spacings between all of the stations
- For this reason, we can assume that

Δφji can not be the same between any two stations. And Δφji should always increase as you increment j and i (since the distance from station 1 to 2 is smaller than from 2 to 3, etc)



- Recall the line in s-z is

$$cot\lambda = \frac{R(\Delta\varphi_{ji} + 2n\pi)}{\Delta z_{ji}}$$

Therefore....

$$\frac{\Delta\varphi_{ji} + 2n\pi}{\Delta z_{ji}} = \frac{\Delta\varphi_{kj} + 2m\pi}{\Delta z_{ki}} \quad etc...$$

- So let's define

$$A = \frac{\Delta\varphi_{ji} + 2n\pi}{\Delta\varphi_{ki} + 2m\pi} \qquad B = \frac{\Delta z_{ji}}{\Delta z_{ki}}$$

-
- It should therefore be true that for the right ***n*** and ***m*** we would have

|A – B| < cut

- Once all of the $\Delta\varphi_{ji}$'s have the right number of 2π's added, we can then fit a line to all of the points using a least-squares fit **(exactly the same way as with the comic tracks)**

Turns Between Stations

- Due to problems with simulations, I haven't had a good input file to test the code with. **So, I created my own helices.**
- Tested the code, and returns accurate values for the dip angle (as well as other parameters)
- SciFiDetermineDip() can technically be called as early as when we calculate the seed, but do we want to call it there or after we've found all 5 space points that we think belong to a track?

Pattern Recognition Complete

- Once we calculate the dip angle, **we now have all of the seed parameters** to feed into the final fitting program
- x_0, y_0, Φ_0, R, $\tan\lambda$, *Ψ_0
- All of these values are *reported at the reference station*
  - station 5 for tracker 1
  - station 1 for tracker 2
    - Right now this part is hard coded, but I don't know a way around it since we treat both trackers the same in PR

*** Ψ_0 (track direction) can not be determined without knowing a charge and a *z* direction. These values will be fetched either at the end of pattern recognition, or at the beginning of the final fitting program.**

Summary and Outlook

- Straight track PR is **complete** – **trying to figure out launchpad so that it can begin code review**
- Helical track PR is also **complete**, although I need to clean it up before sending off for code review
  - **Should we keep the circle re-fit?**
  - **Need to know where to put s-z fit**
- Have yet to test the entirety of the code with MC simulated values. Working with my own creations...
- You have me until Jan 7, after which point I'll be in Morocco and won't have time to work on MICE again until April.

Extra Slides

Least Squares Fit

- If we define our lines in z-x and z-y projections as such, and **minimize χ^2 with respect to *a* and *b*** we get this matrix of equations
- Just invert middle matrix to solve for parameters
- This is showing an unweighted fit but it is straightforward to generalize to include errors, and this is done in the code

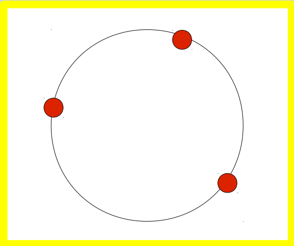
$$x = a + b z$$

$$\chi^2 = \sum_i^n \frac{(x_i - (a + b z_i))^2}{\sigma_{xi}^2}$$

$$\begin{pmatrix} \sum x_i \\ \sum x_i z_i \end{pmatrix} = \begin{pmatrix} n & \sum z_i \\ \sum z_i & \sum z_i^2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

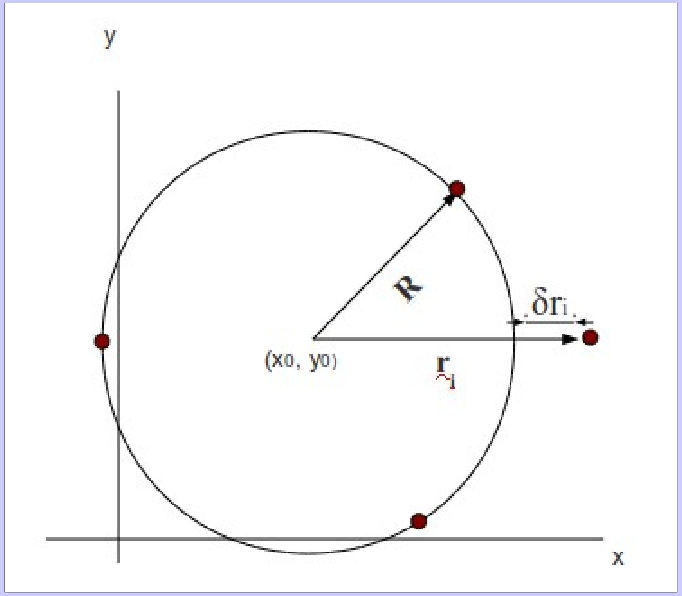
*Same procedure in z-y projection

SciFiMakeSeed()

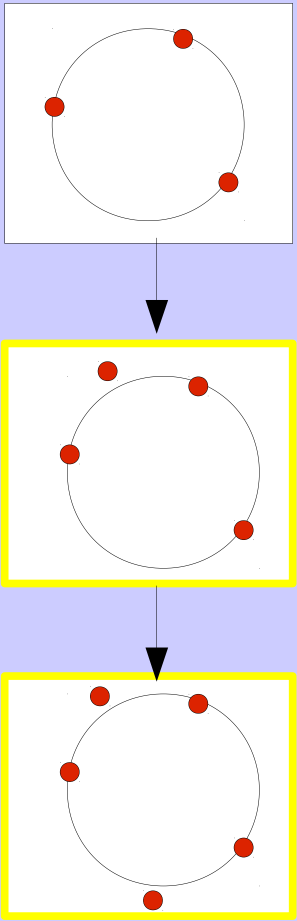


- Boolean function – returns true if the seed can be made
- Input – 3 space points
  - 1 from **each** boundary station
  - 1 from intermediate station
- Finds the unique triangle fitting to the 3 points, and then the circle that circumscribes this triangle
- Output – preliminary values of x_0, y_0, and R

SciFiCompleteTrack()



- boolean, returning true if the added space point fits to the track
- Input - 4th or 5th space point (from intermediate stations)
- Calculates the radial deviation of the added space point from the circle created from the seed



- If this δR is < radial_cut , returns true.
  - Currently radial_cut == 25 mm
  - This is based on my own choice; needs to be optimized!
- Output – just true or false