# Parallelisation of (Track) Fits

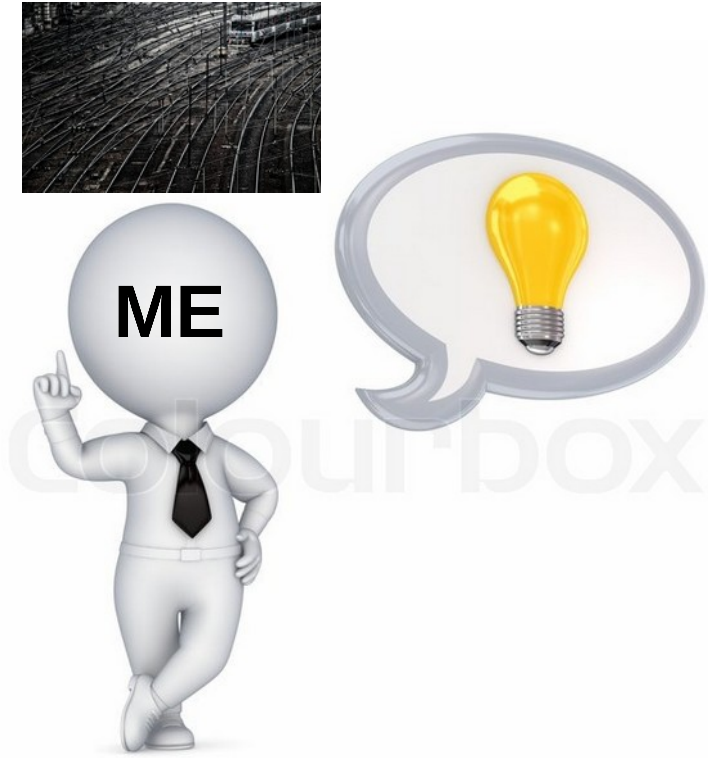
**HighRR Lecture Week**
27. Sep. - 1. Oct .2022

DFG Deutsche Forschungsgemeinschaft

Andre Schöning
(Heidelberg PI)

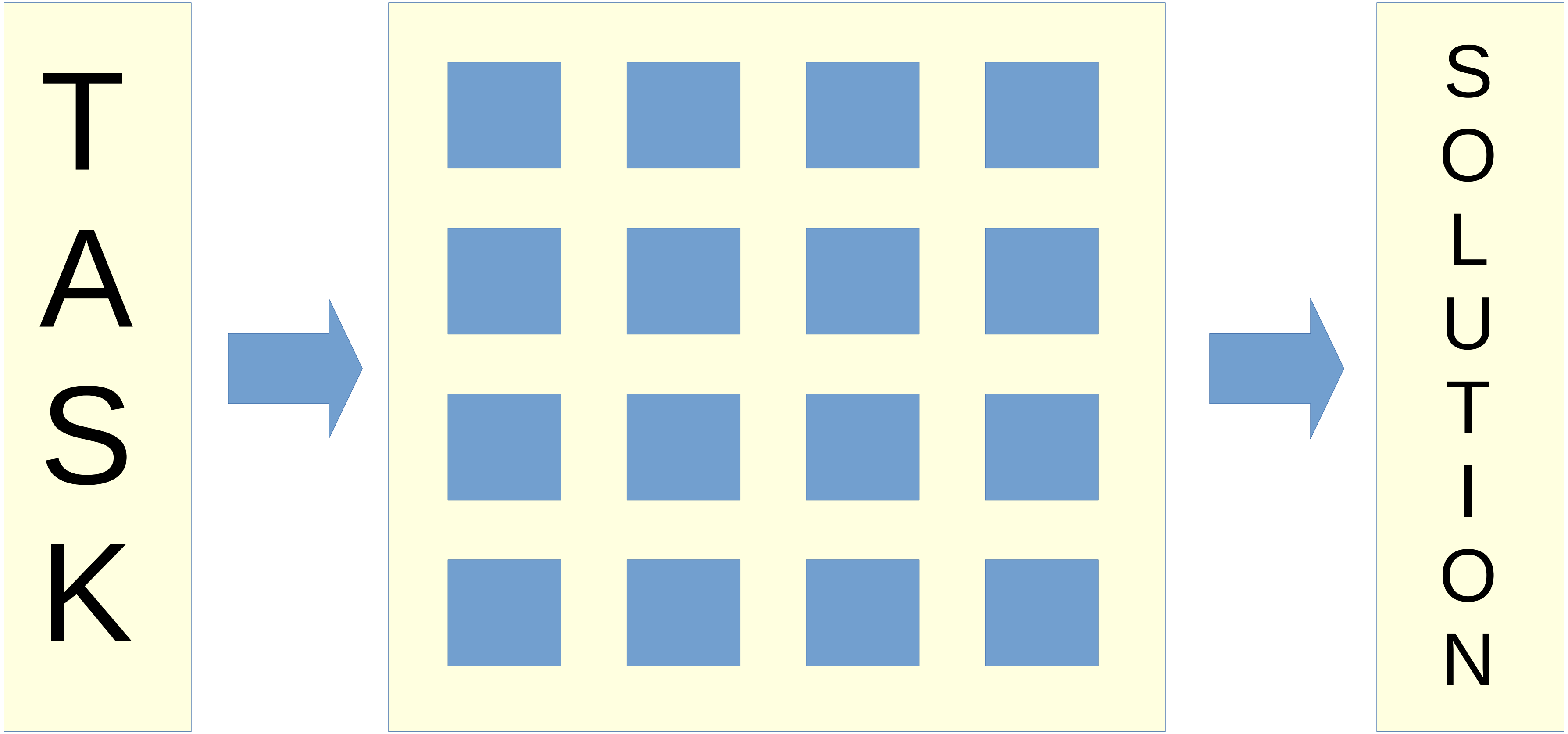
HighRR

→ **with a clear focus on algorithms**



YOU

# Parallelisability



TASK

SOLUTION

# Overview

- Motivation
- Intro to Track Fits (Overview)
- Fitting Tracks with Hit Uncertainties
- Linearisation
- **A New Hit Uncertainty Track Fit**
  - Triplet Representation (parallelisable)
  - Cholesky Decomposition (non-parallelisable)
- Summary

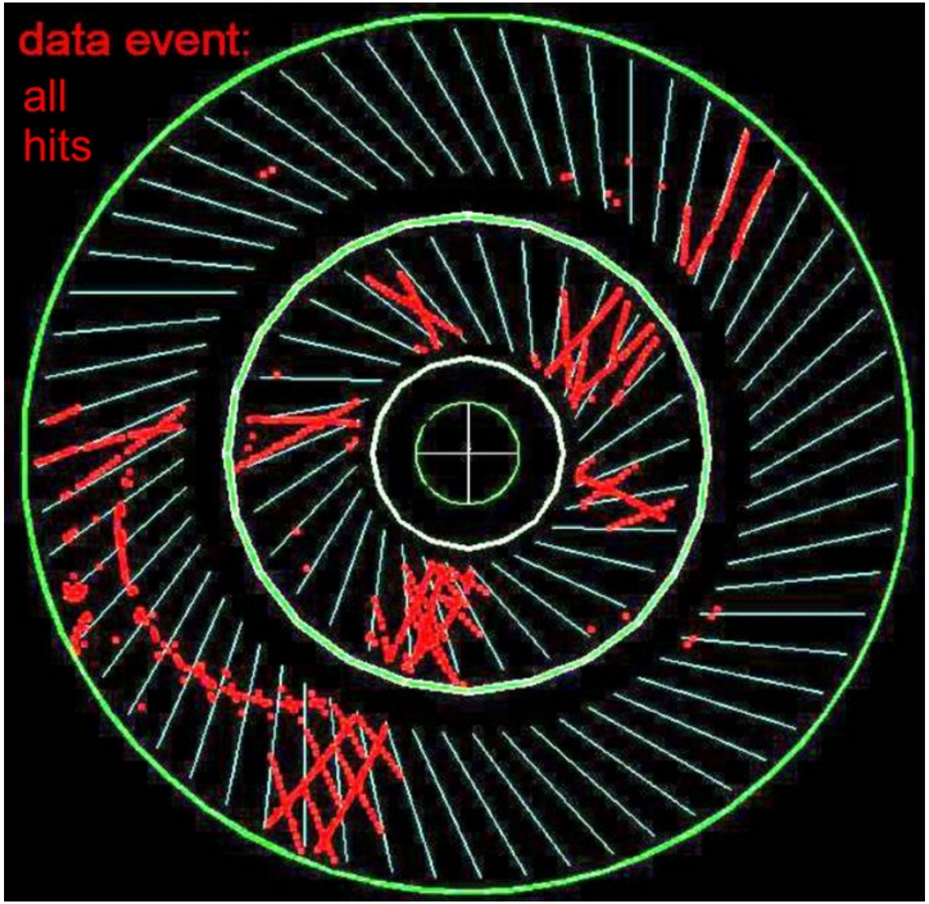
DIDACTICAL

I will try to avoid formulas whenever possible

# What do you see?

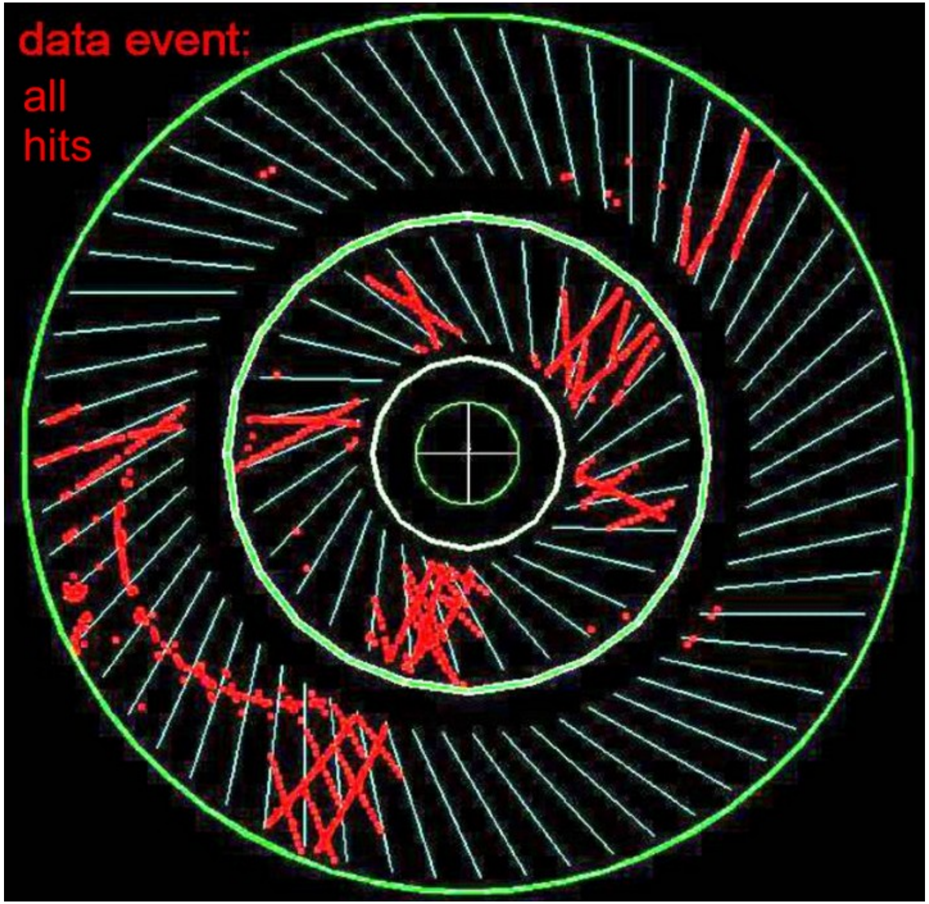
Drift chamber hits of the H1 experiment (1991-2007)

- low track multiplicity
- many measurement points per track
- despite some ambiguities track finding is very easy



# What do you see?

Drift chamber hits of the H1 experiment (1991-2007)



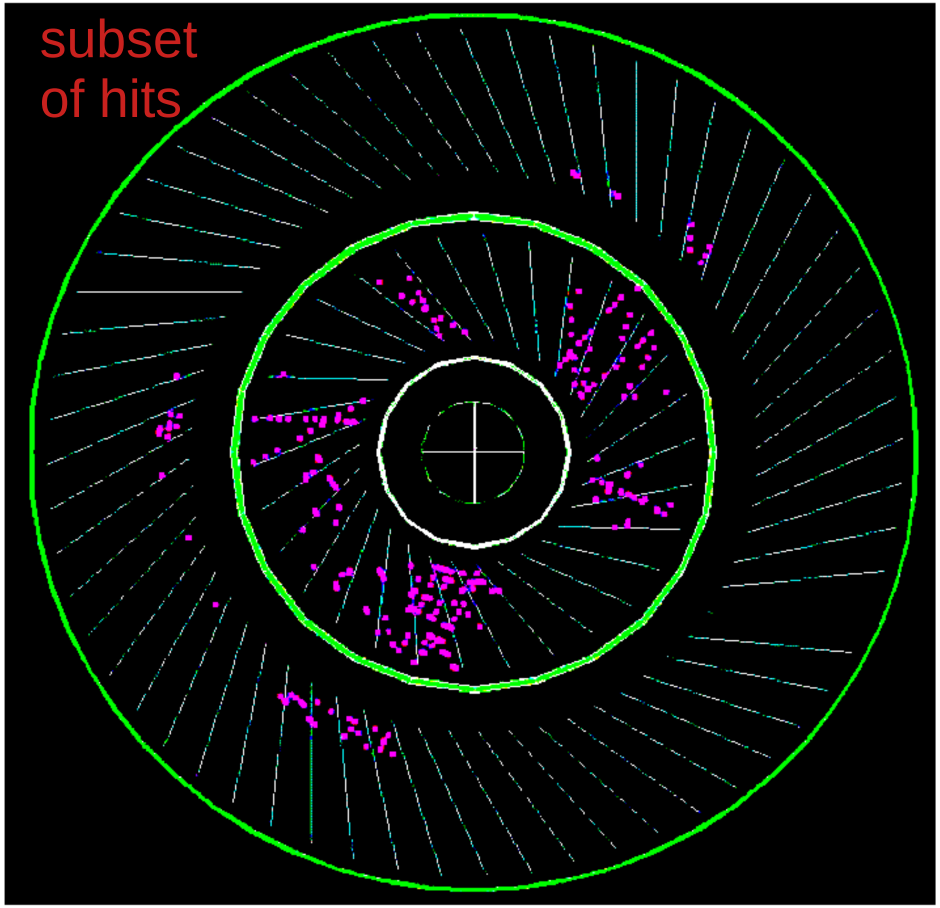
- low track multiplicity
- many measurement points per track
- despite some ambiguities track finding is very easy

- no physics model is needed
- any amateur can find tracks!

# What do you see?

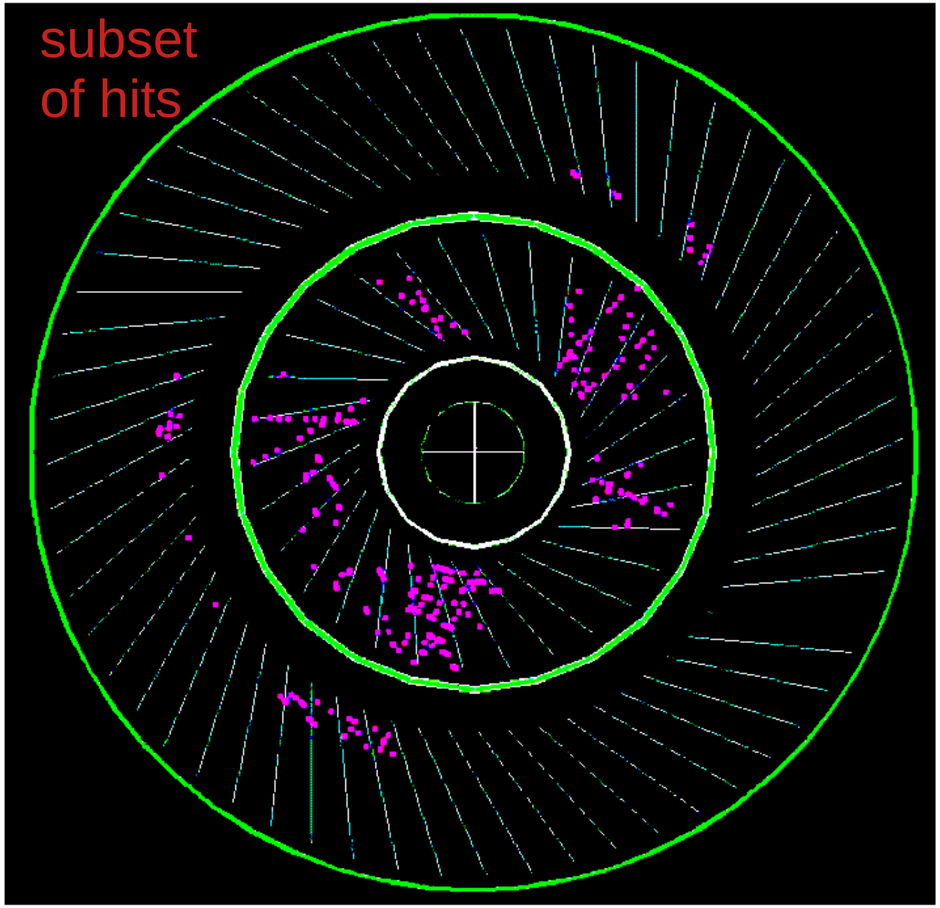
- low track multiplicity
- many measurement points per track
- despite some ambiguities track finding is very easy

Drift chamber hits of the H1 experiment (1991-2007)



# What do you see?

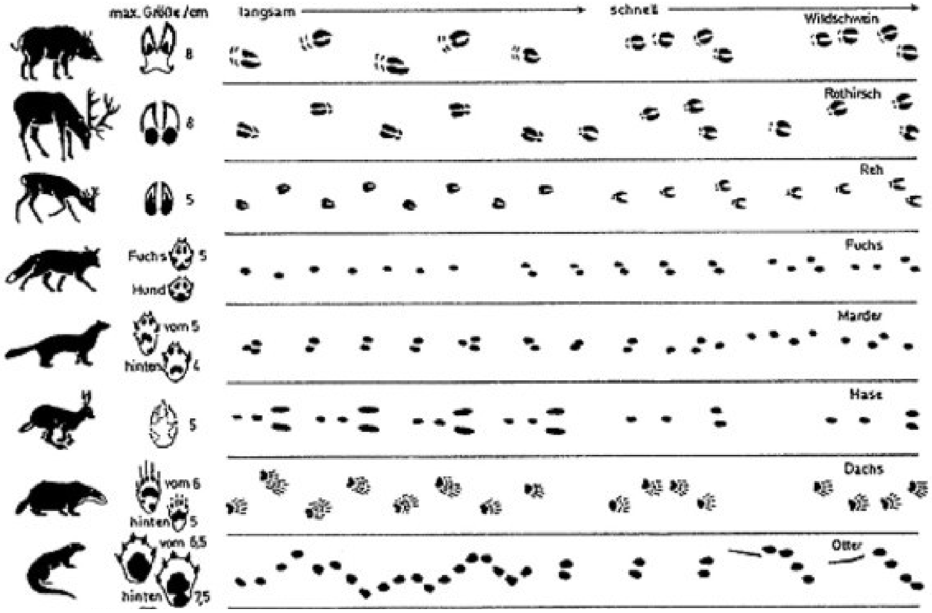
Drift chamber hits of the H1 experiment
(1991-2007)

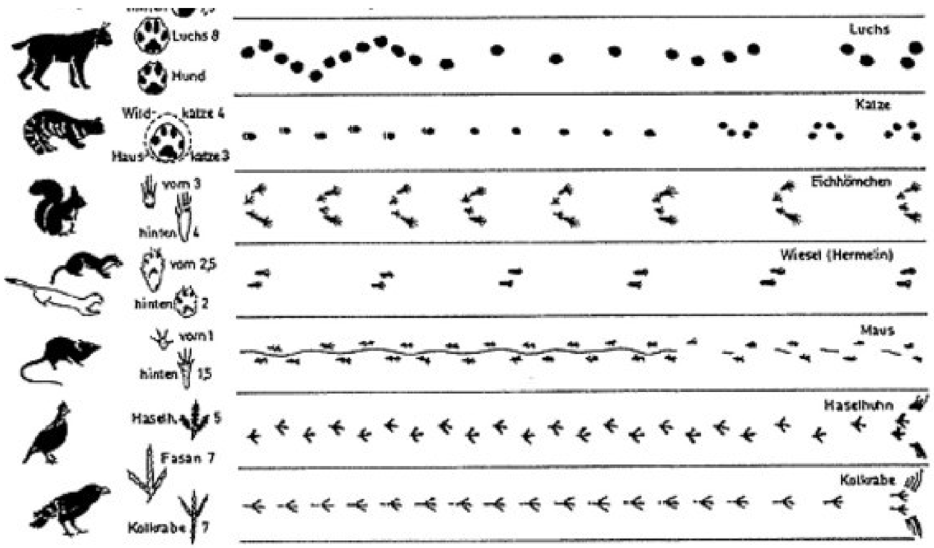


- low track multiplicity
- many measurement points per track
- despite some ambiguities track finding is very easy

- but in the situation of sparse information the task becomes much more difficult!
- physics model (B-field, lorentz force, momentum conservation) is required to reconstruct the tracks

# Sparse Detector (Hit) Information





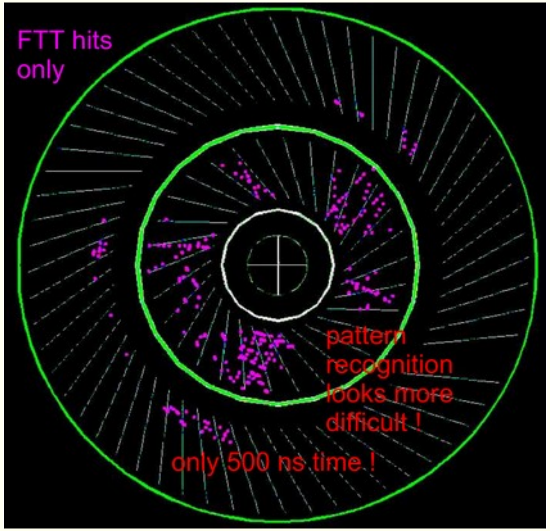
Knowledge helps to find (identify) tracks → should make use of it!

# Examples for Sparse Hit Information

## Trigger:

→ limitations from bandwidth & processing power

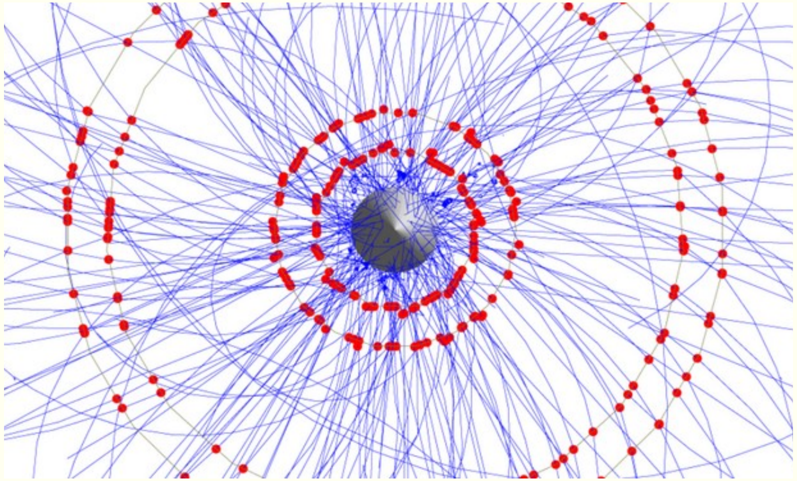
FTT = Fast Track Trigger



## Semiconductor Trackers:

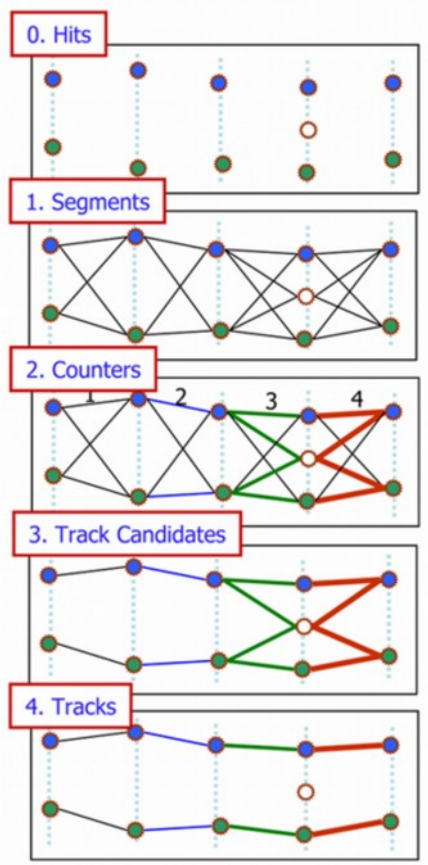
→ limitations from multiple scattering (resolution) & powering & costs

Mu3e: only four (pixel) tracking layers



# Track Reconstruction Example I

## Cellular Automaton (e.g. CBM experiment, ALICE):



- **local** method based on segments
- uses mostly **topological** information
- parallelisable
- implemented e.g. on GPUs

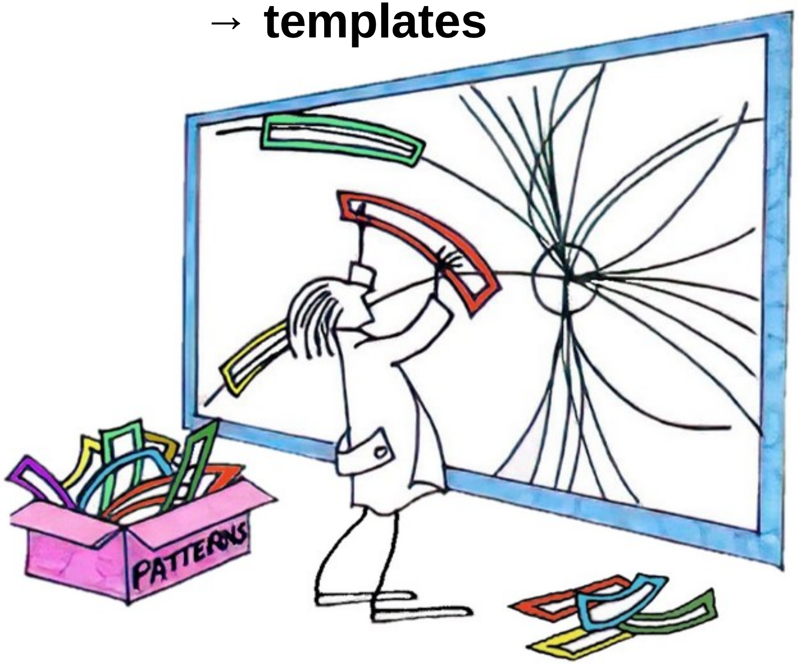
works only if hit density is dense enough

sketch from I.Kisel

# Track Reconstruction Example II

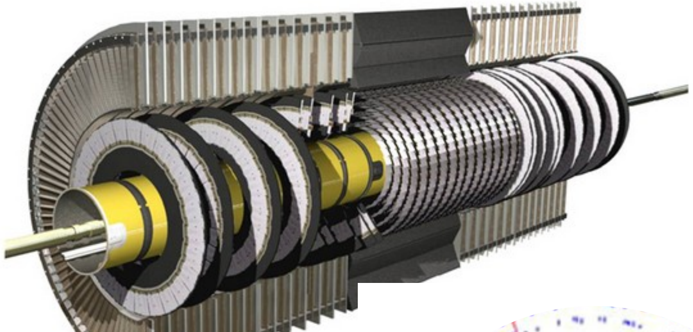
## ATLAS Fast TracKer Project (2019†)

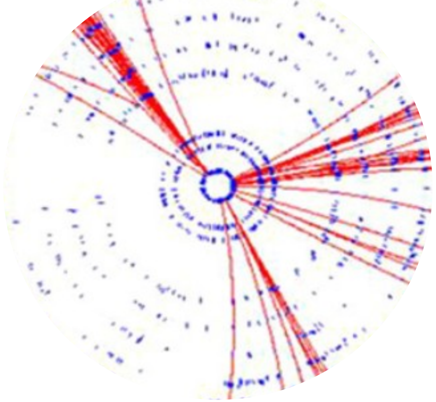
→ **templates**



→ trigger

## ATLAS SCT





- pattern lookup technique (approximation)
- pre-calculated roads (from simulation)
- parallelisable and fast
- implemented in ASICs (AMchip)

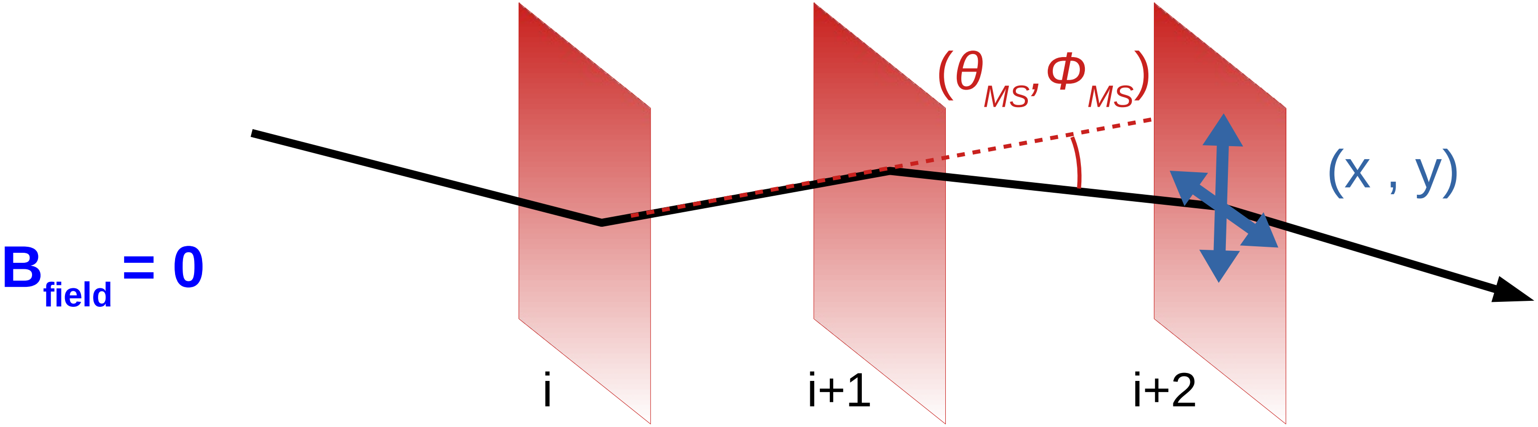
# Track Reconstruction Example III

**Full track fit:**

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$

includes **multiple scattering** and **hit uncertainties**

not best option to fit all track hypothesis



# Track Reconstruction Example III

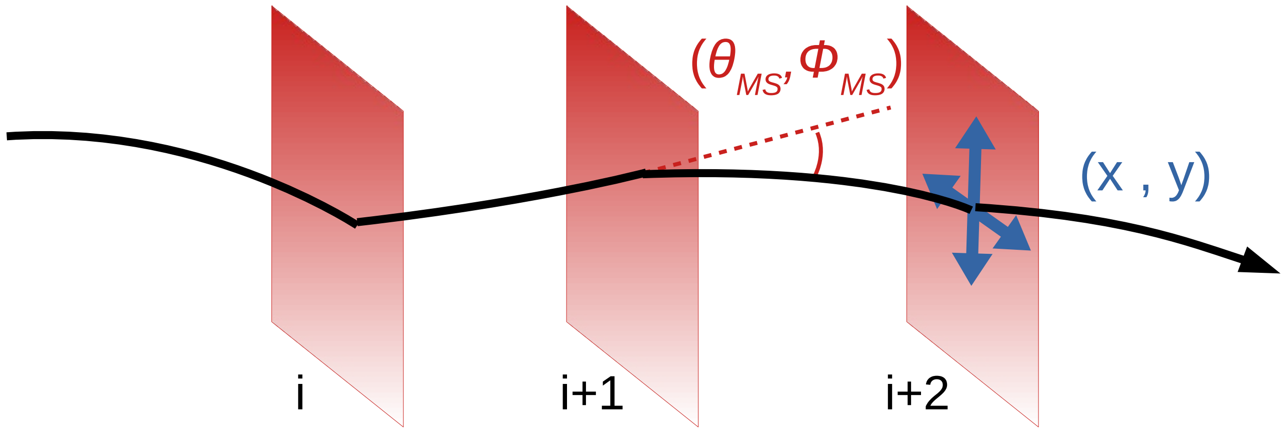
**Full track fit:**

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$

includes **multiple scattering** and **hit uncertainties**

not best option to fit all track hypothesis

**$B_{\text{field}} \neq 0$**

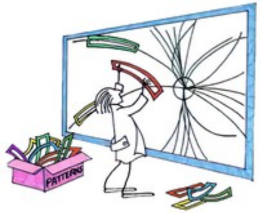


**Full glory fit (magnetic field) is computationally intensive!**

# High Track Multiplicities (ATLAS)

**Motivation for fast (& full) tracking:**

- Identify special or rate track based signatures (e.g. long lived particles)
- track-assisted object reconstruction for tracker (e.g. high energy particles)

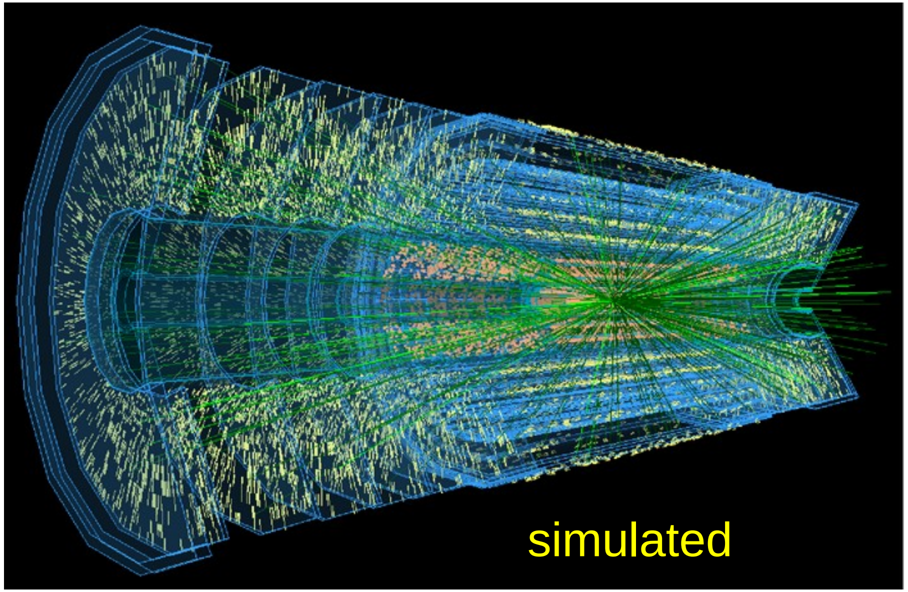


## ATLAS Approaches

- **FTK** Project (2019†)
- **Phase II** (High Lumi-LHC) Hardware Track Trigger Project (**HTT**, 2021†)
- **New**: fast tracking on **Event Filter** (EF)
  - option A: **CPU** only?
  - option B: **GPU-** or **FPGA-** accelerated?

→ provide highly parallel computing architectures!

**Possible to reconstruct all tracks?**



ATLAS High Luminosity Inner TracKer (ITK) with **200 pileup** events at **40 MHz** collisions

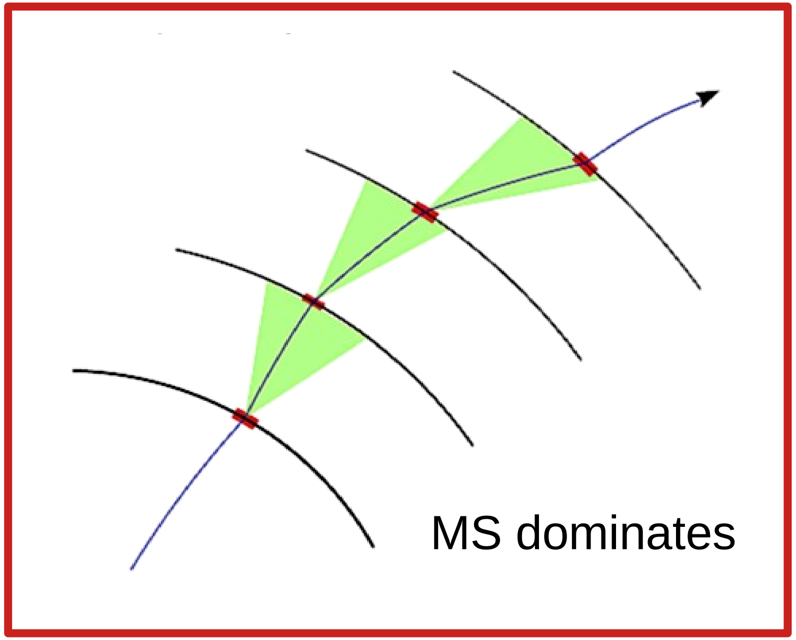
# Question to students:

What is your favorite tracking concept or algorithm?
And why?

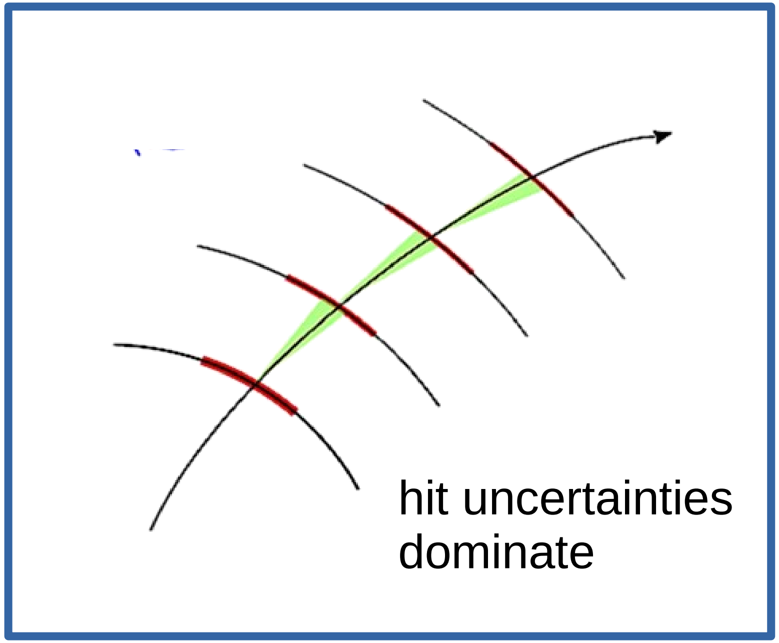
# Chapter 2
# Introduction to Track Fits

# The Master Equation

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$



MS dominates



hit uncertainties dominate

# The Master Equation

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$

Problems and difficulties:

- hit error matrix $V^{-1}$ depends on the trajectory (result)
- scattering uncertainties and scattering angles depend on trajectory (result)
- fitted hit positions are all correlated → **no local** processing possible
- single outliers can spoil the fit → **iterative** outlier rejection

**non-locality and iterations are the enemy of parallelisation**

# Kálmán Filter (KF)

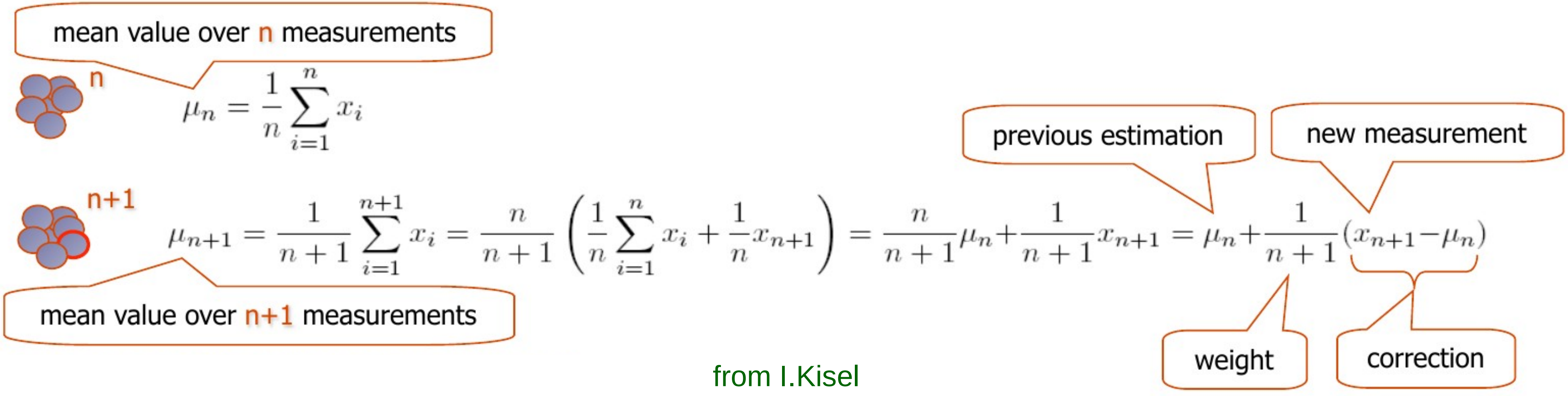
The Kálmán Filter is an algorithm method which combines

- **track finding** (aka hit linking) and
- trajectory determination (**track parameter fitting**)

Note, the KF as such does not implement any physics!

## Simple example:

Calculation of an average


mean value over n measurements
n
n+1
mean value over n+1 measurements
previous estimation
new measurement
weight
correction

$$\mu_n = \frac{1}{n}\sum_{i=1}^{n} x_i$$

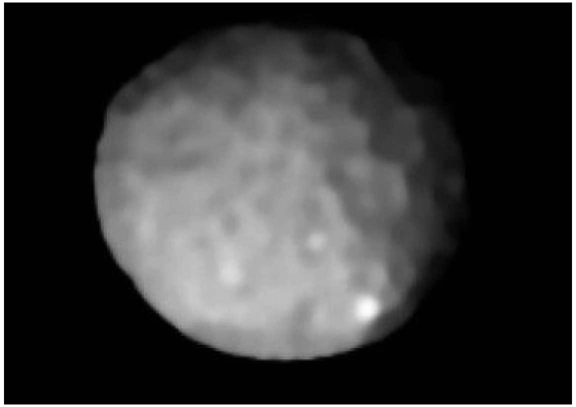
$$\mu_{n+1} = \frac{1}{n+1}\sum_{i=1}^{n+1} x_i = \frac{n}{n+1}\left(\frac{1}{n}\sum_{i=1}^{n} x_i + \frac{1}{n}x_{n+1}\right) = \frac{n}{n+1}\mu_n + \frac{1}{n+1}x_{n+1} = \mu_n + \frac{1}{n+1}\underbrace{(x_{n+1}-\mu_n)}_{\text{correction}}$$

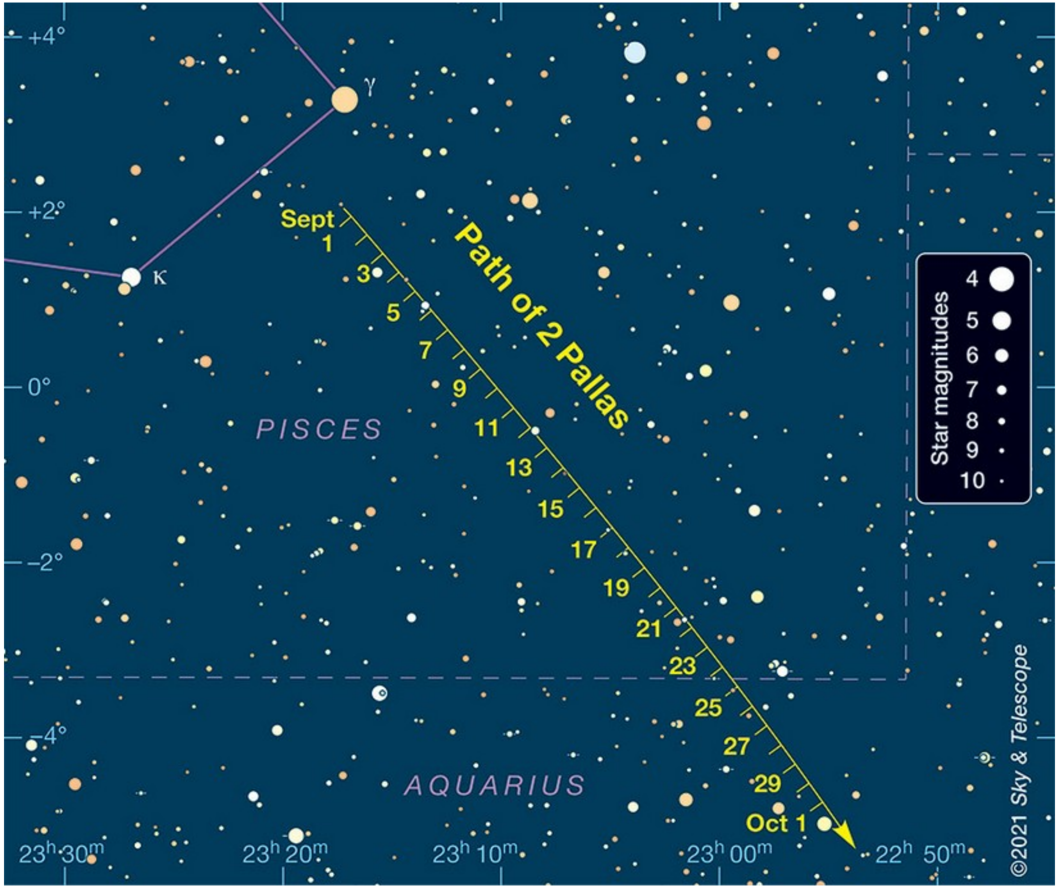
from I.Kisel

# Example: Asteroids

Asteroid Pallas

(first found be German astronomer Heinrich Olbers in 1807)





# Tracking of Asteroids

Suppose we discover an asteroid!

**State vector** described by

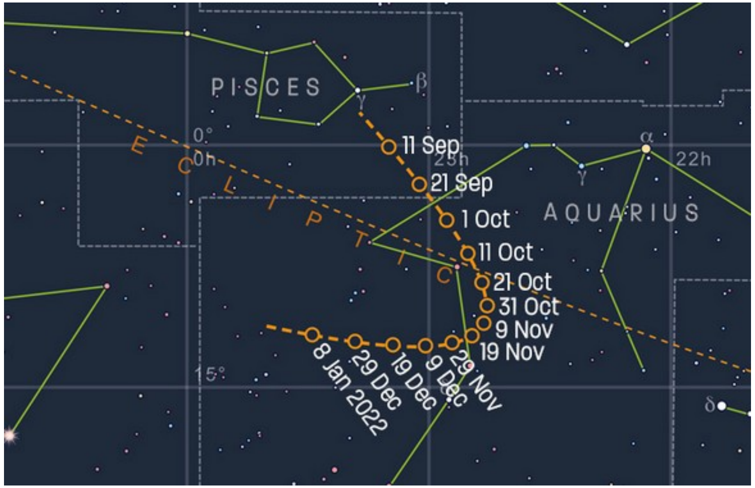
$$\vec{r} = (x, y, z, v_x, v_y, v_z)$$

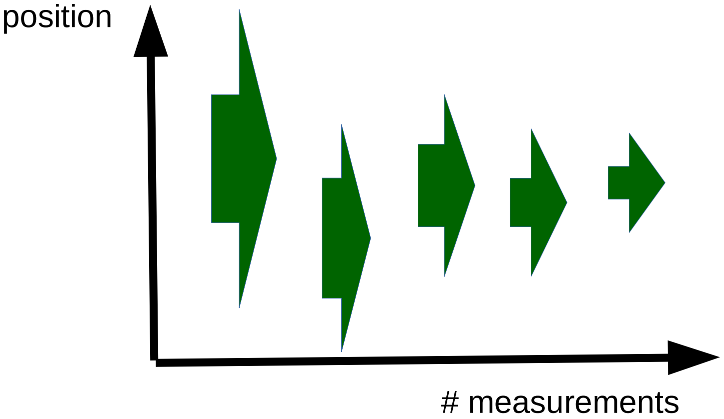
1st measurement (11.9): → **2d** information
2nd measurement (21.9): → 2d+2d=**4d** information
3rd measurement (1.10): → 4d+2d=**6d** information

→ able to reconstruct state vector ***r***
and first guess of error matrix

4th measurement (11.10): → **update** state and error matrix
5th measurement (21.10): → **update** state and error matrix
...

→ **the more measurements, the more precise!**

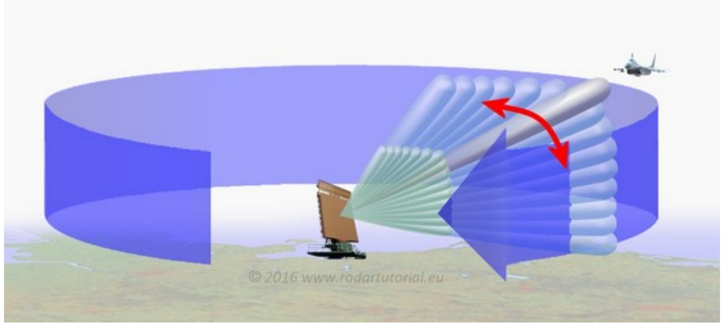




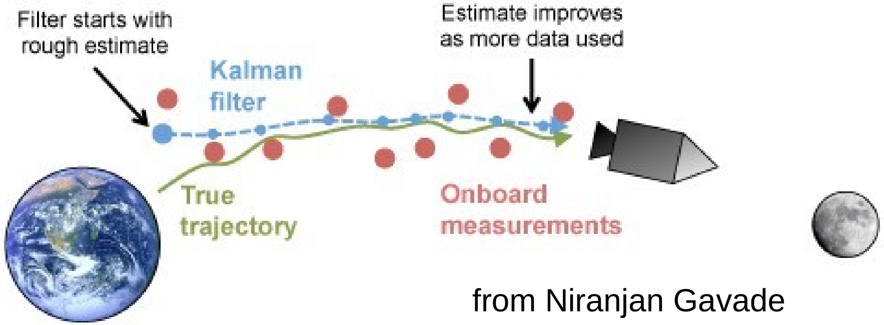
# Rudolf Kálmán (1930-2016)

Kálmán (emeritus ETH professor ) receiving the National Medal of Science from US president Obama in Oktober 2009.

Tracking of air-planes



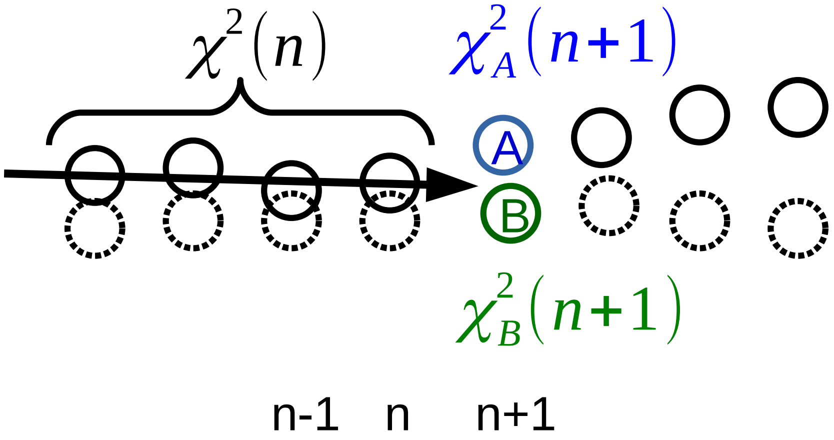
Tracking of space crafts (NASA Apollo mission)


Filter starts with rough estimate
Kalman filter
Estimate improves as more data used
True trajectory
Onboard measurements

from Niranjan Gavade

# Kálmán Filter Applied to Tracking

Hit linking example:



Properties:

- **flexible**
- relies on track **extrapolation** (works also in inhomogeneous magnetic fields)
- **iterative** algorithm (**not parallelisable**)
- results depend on the **order** and **direction** (e.g. **inside-out** versus **outside-in** tracking)

The Kálmán fitter is the **gold standard** in particle physics, nowadays

# Is there any parallelisable track fit?

# The Master Equation

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$

Multiple scattering uncertainties dominate for:

- **low momentum** tracks (→ low energy physics)
- **high precision** trackers (→ instrumentation technology)

Highland formula (PDG):

(RMS of scattering angle)

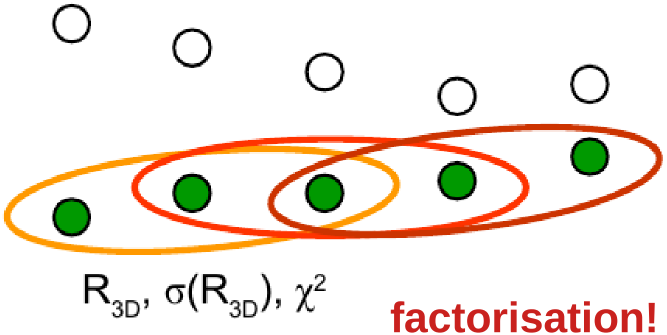
$$\theta_0 = \theta^{\mathrm{rms}}_{\mathrm{plane}} = \frac{13.6\ \mathrm{MeV}}{\beta c p}\, z \sqrt{\frac{x}{X_0}} \left[1 + 0.038 \ln\left(\frac{x\, z^2}{X_0 \beta^2}\right)\right]$$

→ A multiple scattering fit can be **parallelised**!

# The Multiple Scattering Fit

Literature:

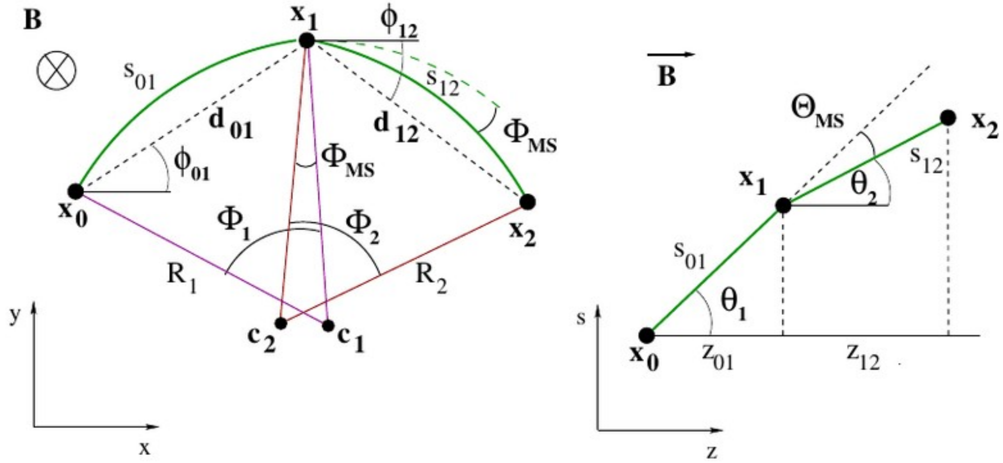
- **A New Track Reconstruction Algorithm suitable for Parallel Processing based on Hit Triplets and Broken Lines**
  [AS], EPJ Web Conf. 127 (2016) 00015



factorisation!

- **A New Three-Dimensional Track Fit with Multiple Scattering,**
  N.Berger, M.Kiehn, A.Kozlinskyi, [AS],
  NIMA 844C, 135 (2017)

**fit of hit triplet:**

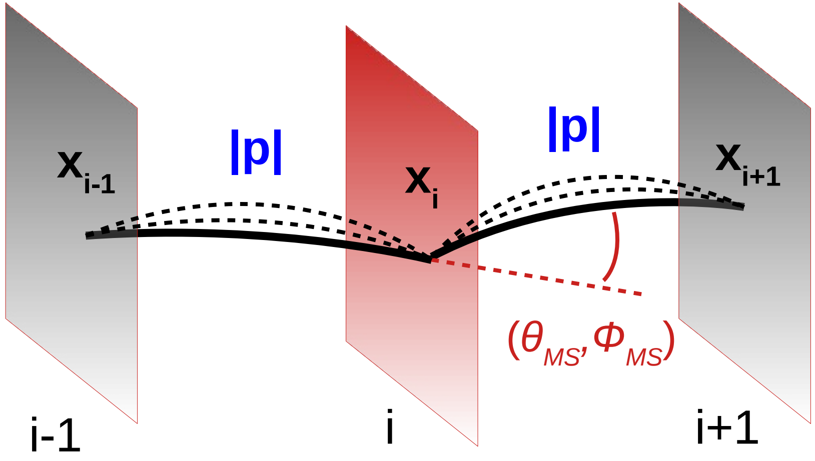


→ used by **Mu3e** and **Belle2** experiments

# The Multiple Scattering Triplet Fit

Assumptions:

- All points $\mathbf{x}_{i-1}$, $\mathbf{x}_i$, $\mathbf{x}_{i+1}$ (instrumentation layers) are given
- the modulus of the **momentum p** of the particle is **conserved**
- the magnetic **field B** is constant
- the **material** in **layer i** is known



(convenient to use cylinder coordinates)

- The **momentum |p|** is the only free (unknown) parameter of the particle
- The MS angles $(\theta_{MS}, \Phi_{MS}) \rightarrow$ minimised depend on **|p|**

$$\chi^2 = \frac{\Theta_{MS}^2}{\sigma_\theta^2} + \frac{\Phi_{MS}^2}{\sigma_\phi^2}$$

- All other parameters (e.g. direction) can be derived from triplet geometry if **|p|** is known!

# B-Field: The Helix

Parameterisation (Cartesian coordinates):

$$\boldsymbol{r}(t) = R\,[\cos(2\pi t)\,\boldsymbol{e}_x + \sin(2\pi t)\,\boldsymbol{e}_y] + h\,t\,\boldsymbol{e}_z$$

In transverse plane (2D) of a magnetic field

$$R = \frac{p_\perp}{qB}$$

Define:

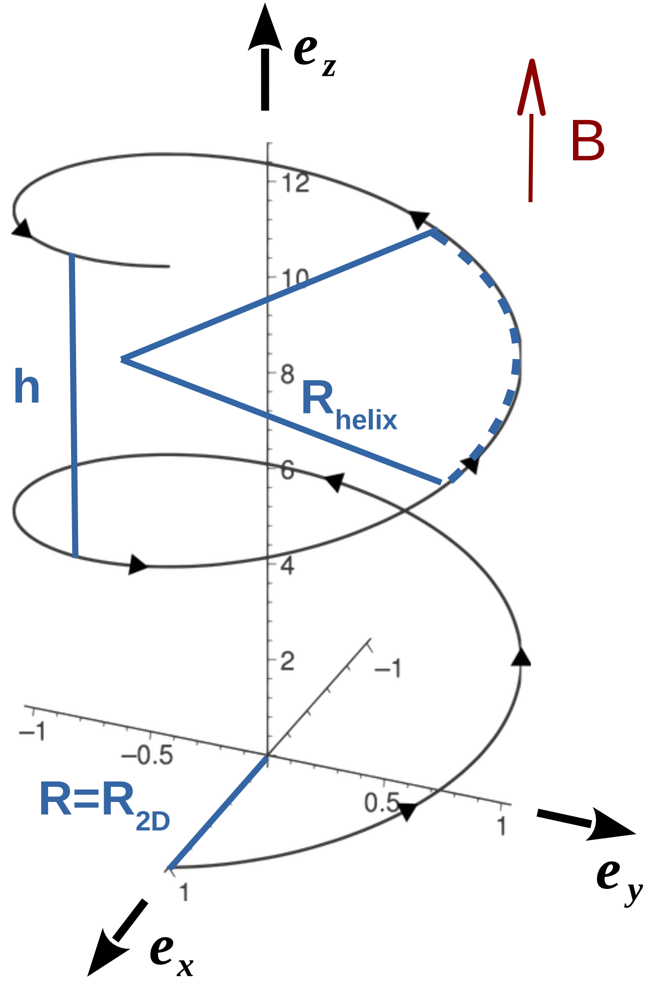
$$R_{3\mathrm{D}} = \frac{R}{\sin(\theta)} = \frac{p}{qB}$$

**invariant for MS!**

Relation:

$$R_{3D}^2 = R_{2D}\,R_{\mathrm{helix}}$$

(geometric average)



# Solution of MS Triplet Fit

Calculate:

$$\Phi_{\mathrm{MS}} = \Phi_{\mathrm{MS}}(R_{\mathrm{3D}})$$

$$\Theta_{\mathrm{MS}} = \Theta_{\mathrm{MS}}(R_{\mathrm{3D}})$$

Solution given by

$$\sin^2 \frac{\Phi_1}{2} = \frac{d_{01}^2}{4R_{\mathrm{3D}}^2} + \frac{z_{01}^2}{R_{\mathrm{3D}}^2} \frac{\sin^2(\Phi_1/2)}{\Phi_1^2}$$

$$\sin^2 \frac{\Phi_2}{2} = \frac{d_{12}^2}{4R_{\mathrm{3D}}^2} + \frac{z_{12}^2}{R_{\mathrm{3D}}^2} \frac{\sin^2(\Phi_2/2)}{\Phi_2^2}$$

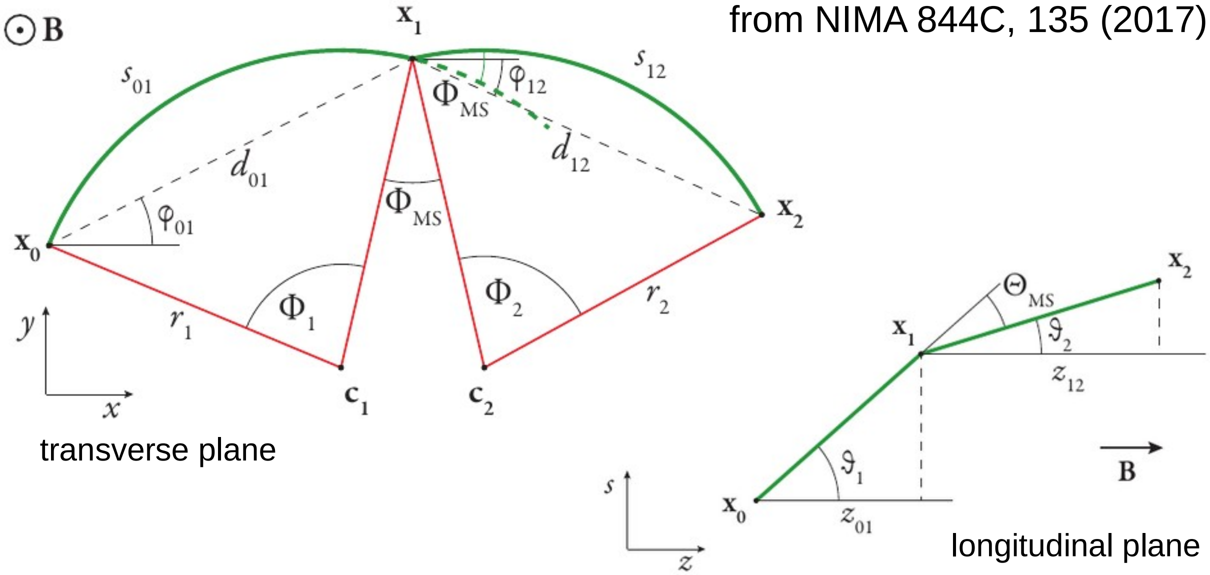
and

$$\sin \theta_1 = \frac{d_{01}}{2R_{\mathrm{3D}}} \operatorname{cosec}\left(\frac{z_{01}}{2R_{\mathrm{3D}} \cos \theta_1}\right)$$

$$\sin \theta_2 = \frac{d_{12}}{2R_{\mathrm{3D}}} \operatorname{cosec}\left(\frac{z_{12}}{2R_{\mathrm{3D}} \cos \theta_2}\right)$$

→ transcendent equations

from NIMA 844C, 135 (2017)



transverse plane

longitudinal plane

Important geometric relations for solution:

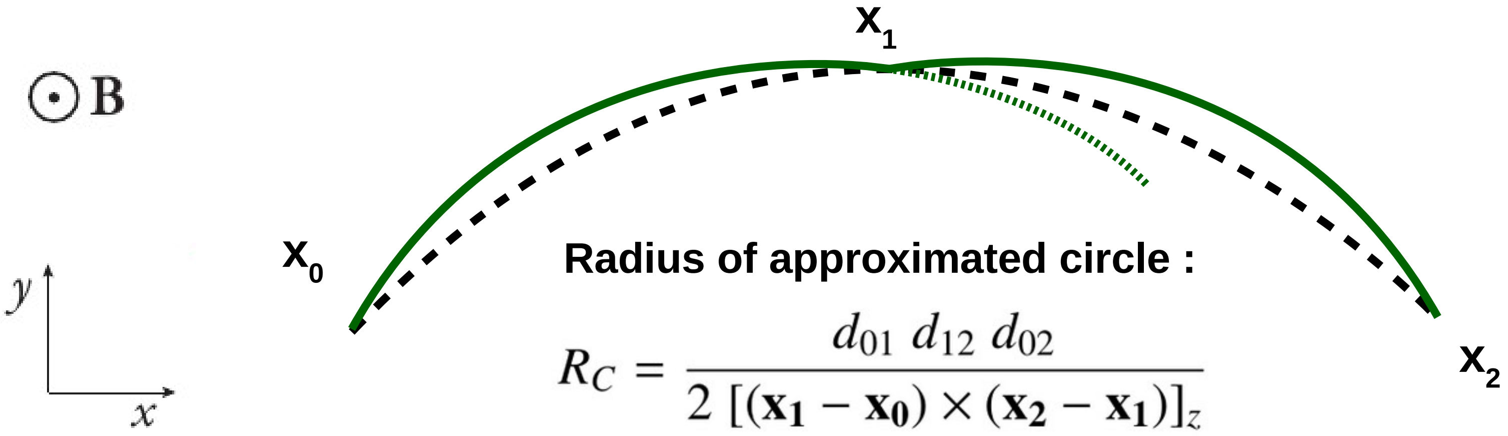
$$R_{\mathrm{3D}}^2 = R_1^2 + \frac{z_{01}^2}{\Phi_1^2} = R_2^2 + \frac{z_{12}^2}{\Phi_2^2}$$

$$\Phi_{\mathrm{MS}}(R_{\mathrm{3D}}) = (\phi_{12} - \phi_{01}) - \frac{\Phi_1(R_{\mathrm{3D}}) + \Phi_2(R_{\mathrm{3D}})}{2}$$

# Linearisation of MS Triplet Fit

- Minimisation of the $\chi^2$-function requires the **derivative** of **transcendent** equations (→ **OK**)
- But the derivatives are again transcendent equations; **no algebraic solution** (→ **NOK**)
- However, the functions are **analytical** → **linearisation ansatz**

**Trick:** assume that the scattering **angles are small!** (→ good assumption)



**Radius of approximated circle :**

$$R_C = \frac{d_{01}\, d_{12}\, d_{02}}{2\,[(\mathbf{x}_1 - \mathbf{x}_0) \times (\mathbf{x}_2 - \mathbf{x}_1)]_z}$$

transverse plane

→ **treat multiple scattering as small perturbation!**

# Single MS Triplet Fit

**3D Radius** (momentum):

$$R_{3D}^{min} = -\frac{\eta\,\tilde{\Phi}\,\sin^2\vartheta + \beta\,\tilde{\Theta}}{\eta^2\sin^2\vartheta + \beta^2}$$

independent
of MS uncertainty!

**Fit quality**:

$$\chi^2_{min} = \frac{1}{\boxed{\sigma^2_{MS}}}\,\frac{(\beta\,\tilde{\Phi} - \eta\,\tilde{\Theta})^2}{\eta^2 + \beta^2/\sin^2\vartheta}$$

3D Radius **uncertainty**:

$$\sigma(R_{3D}) = \boxed{\sigma_{MS}}\sqrt{\frac{1}{\eta^2\sin^2\vartheta + \beta^2}}$$

Note that $\sigma_{MS}$ is calculated from MS-formula
using above momentum result

Geometry parameters are based on circle solution:

$$\tilde{\Phi} = -\frac{1}{2}(\Phi_{1C}\alpha_1 + \Phi_{2C}\alpha_2)\,,$$

$$\eta = \frac{d\Phi_{MS}}{dR_{3D}} = \frac{\Phi_{1C}\,\alpha_1}{2R_{3D,1C}} + \frac{\Phi_{2C}\,\alpha_2}{2R_{3D,2C}}$$

$$\tilde{\Theta} = \vartheta_{2C} - \vartheta_{1C} - \Big((1-\alpha_2)\cot\vartheta_{2C} - (1-\alpha_1)\cot\vartheta_{1C}\Big)$$

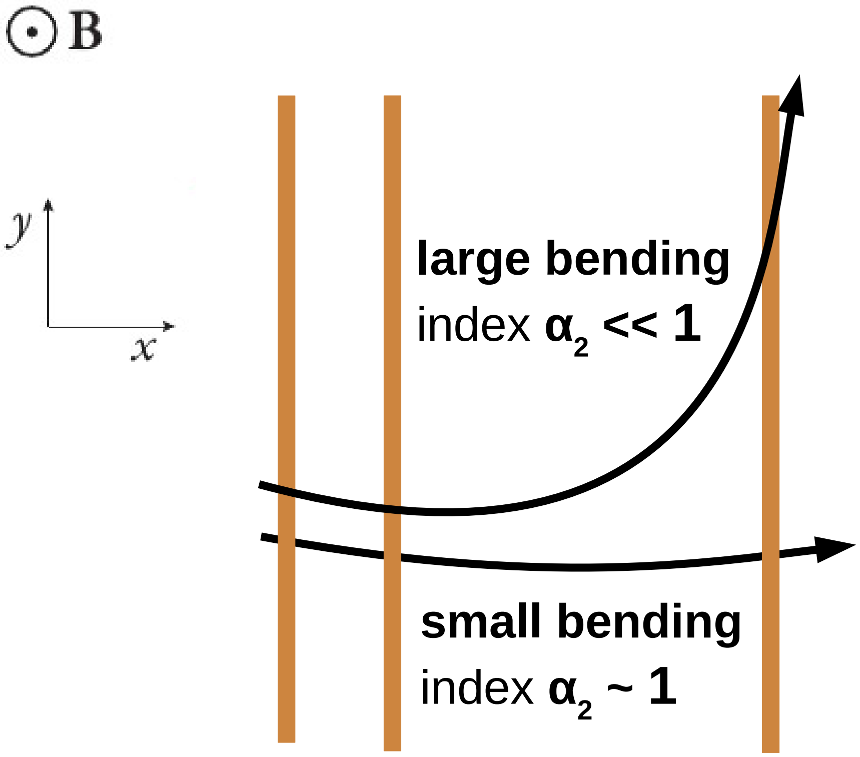
$$\beta = \frac{d\Theta_{MS}}{dR_{3D}} = \frac{(1-\alpha_2)\cot\vartheta_{2C}}{R_{3D,2C}} - \frac{(1-\alpha_1)\cot\vartheta_{1C}}{R_{3D,1C}}\,.$$

with index
parameters:

$$\alpha_1 = \frac{R_C^2\Phi_{1C}^2 + z_{01}^2}{\frac{1}{2}R_C^2\Phi_{1C}^3\cot\frac{\Phi_{1C}}{2} + z_{01}^2}$$

$$\alpha_2 = \frac{R_C^2\Phi_{2C}^2 + z_{12}^2}{\frac{1}{2}R_C^2\Phi_{2C}^3\cot\frac{\Phi_{2C}}{2} + z_{12}^2}$$

# Example Spectrometer

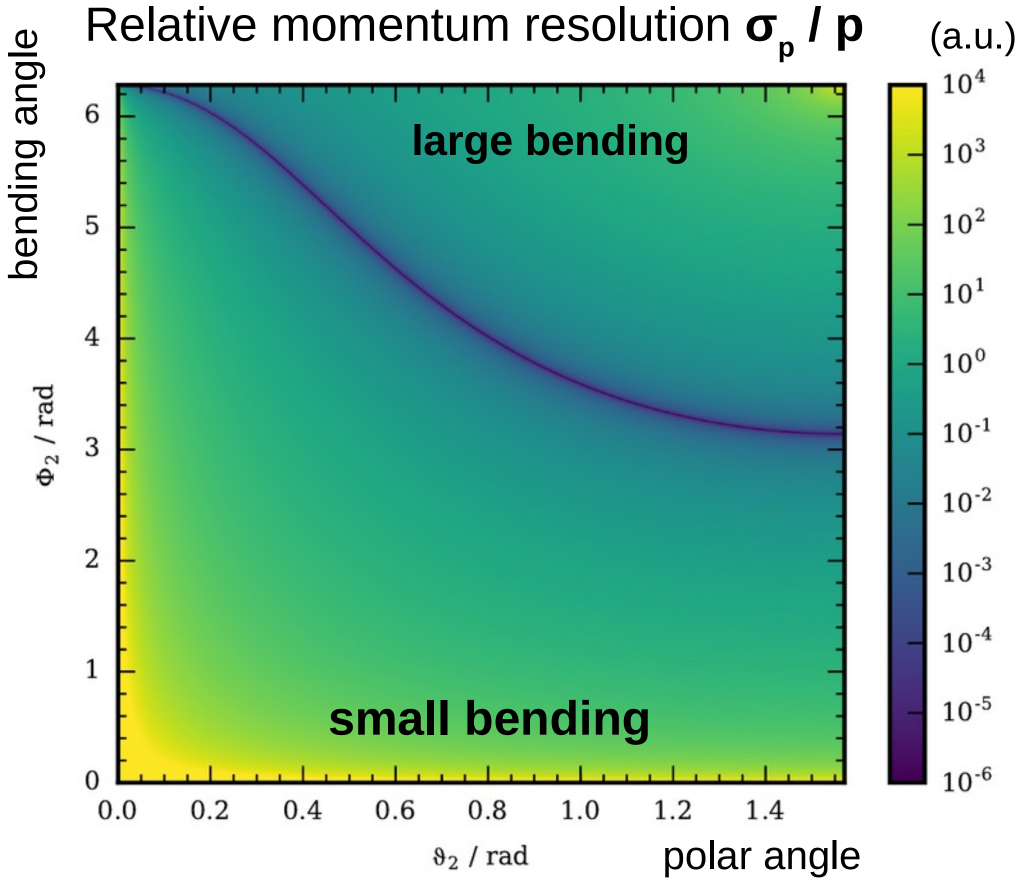


**large bending**

index $\boldsymbol{\alpha_2} \mathbf{<< 1}$

**small bending**

index $\boldsymbol{\alpha_2} \mathbf{\sim 1}$


Relative momentum resolution $\boldsymbol{\sigma_p / p}$ (a.u.)

# Combination of Triplets

Each triplet fit provides:

$$R_{3\mathrm{D}i}\ ,\ \sigma(R_{3\mathrm{D}})_i\ ,\ \chi^2_i$$

averaging:

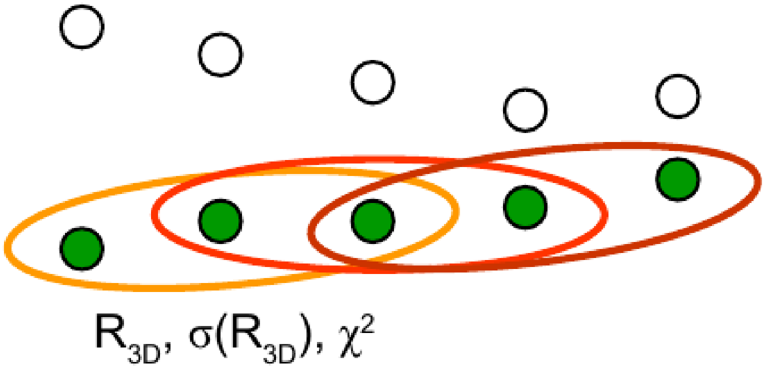
$$\overline{R_{3\mathrm{D}}} = \sum_i^{n_{hit}-2} \frac{R_{3\mathrm{D,i}}}{\sigma_i(R_{3D})^2} \Big/ \sum_i^{n_{hit}-2} \frac{1}{\sigma_i(R_{3D})^2}$$

combination:

$$\chi^2_{comb} = \sum_{i=\mathrm{triplet}} \chi^2_i + \frac{(R_{3\mathrm{D,i}} - \overline{R_{3\mathrm{D}}})^2}{\sigma_i(R_{3D})^2}$$

Comments:

- every triplet is independent (hit positions are given)
- thus, all momentum measurements are independent!
- errors are uncorrelated



number of hits: $\mathbf{N_{hit}}$

number of triplets: $\mathbf{N_{triplet} = N_{hit}-2}$

Remark: track building is simple:

- connecting triplets share two hits
- connecting triplets should have compatible momenta

→ **graph theory**

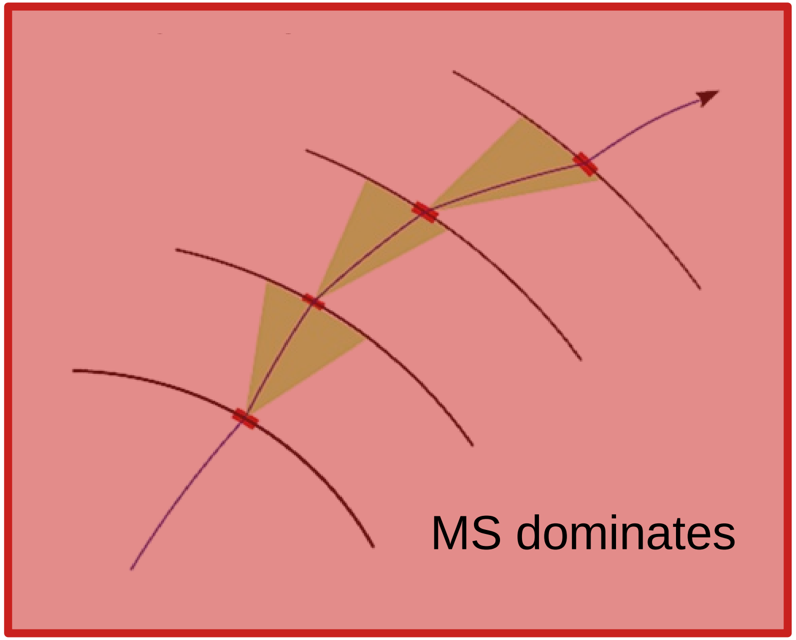
# Question to students:

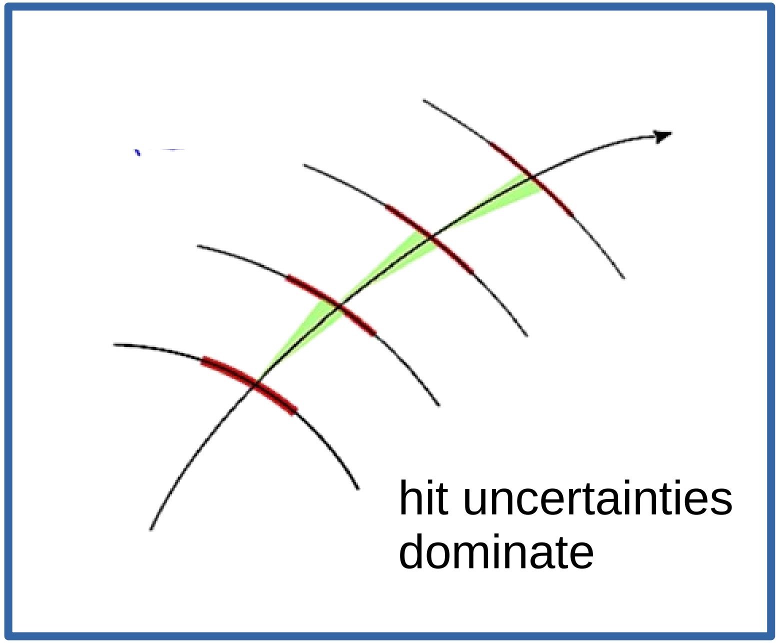
What are the advantages and achievements of the MS Triplet fit?

# Chapter 3
# Fitting Tracks with Hit Uncertainties

# The Master Equation

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$



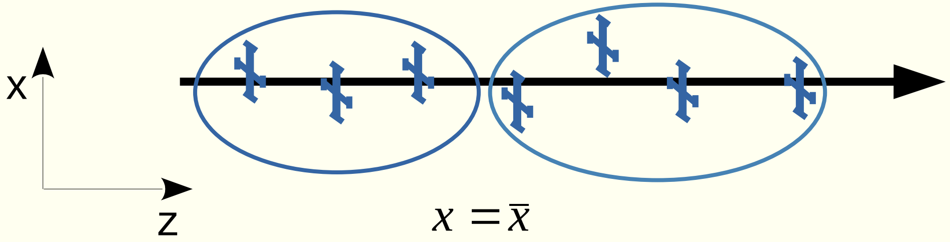


# Fitting Tracks with Hit Uncertainties

(no multiple scattering)

**Case A**

- no B-field
- no slope
  - ➔ **averaging**



$x = \bar{x}$

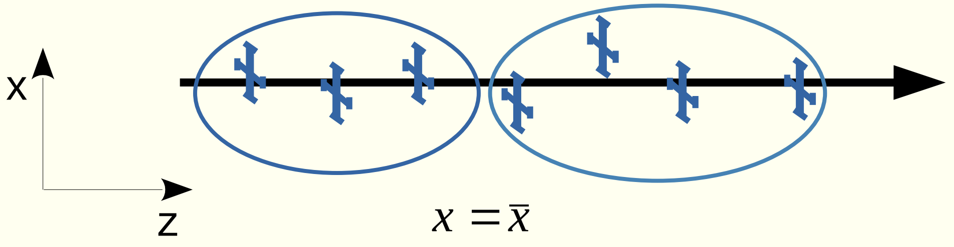
$\bar{x} = \sum_{\text{hits } i} x_i \, w_i$

# Fitting Tracks with Hit Uncertainties

(no multiple scattering)

**Case A**

- no B-field
- no slope
  - ➔ **averaging**

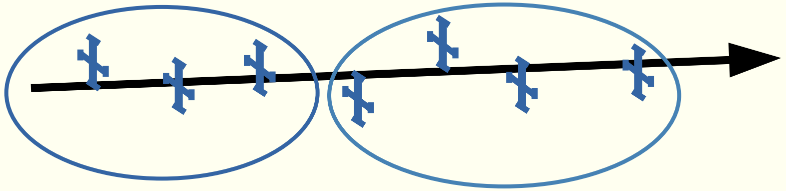


$x = \bar{x}$

$$\bar{x} = \sum_{\text{hits } i} x_i \, w_i$$

**Case B**

- no B-field
- slope unknown
  - ➔ **straight line fit**



$x = \bar{x} + \beta(z - \bar{z})$

$$\bar{x} = \sum_{\text{hits } i} x_i \, w_i$$

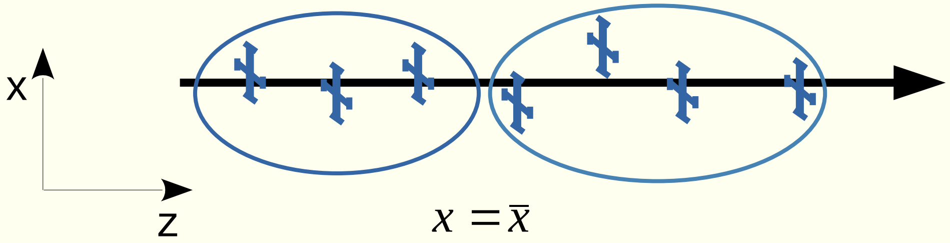
$$\beta = \frac{Cov(Z,X)}{Var(Z)}$$

# Fitting Tracks with Hit Uncertainties

(no multiple scattering)

**Case A**
- no B-field
- no slope

  → **averaging**

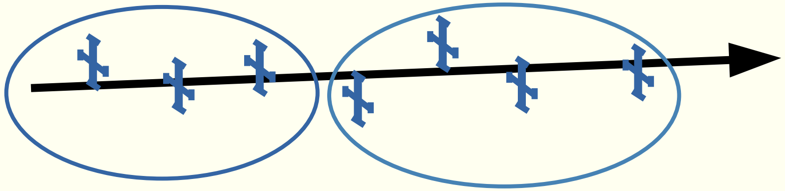


$x = \bar{x}$

$\bar{x} = \sum_{\text{hits } i} x_i \, w_i$

**Case B**
- no B-field
- slope unknown

  → **straight line fit**



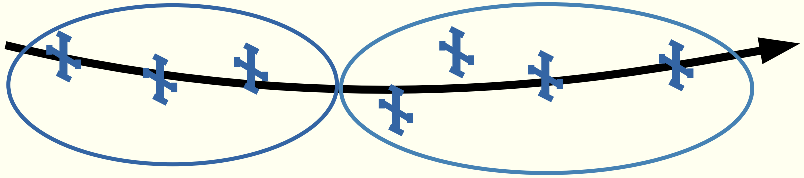
$x = \bar{x} + \beta(z - \bar{z})$

$\bar{x} = \sum_{\text{hits } i} x_i \, w_i$

$$\beta = \frac{Cov(Z, X)}{Var(Z)}$$

**Case C**
- B-field > 0
- slope unknown
- momentum unknown

  → **helix fit**



$x = x(z; \bar{x}, \beta, R)$

$\bar{x} = \sum_{\text{hits } i} x_i \, w_i$

$\beta = \ldots$

$R = \ldots$

# Question to students:

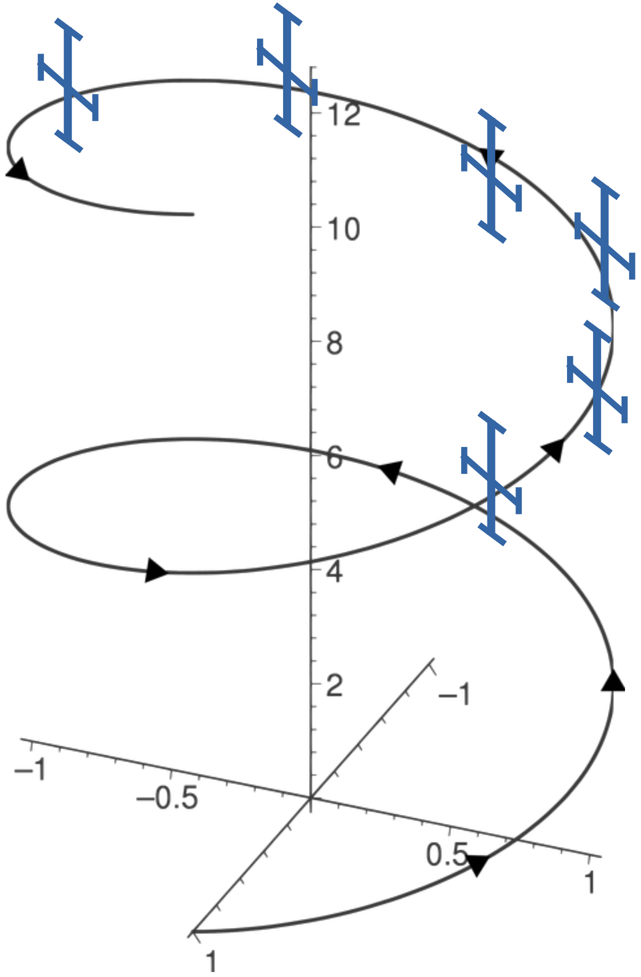
Which of the three cases are parallelisable?

And if yes, how?

# Fitting a Helix to Hits with Errors

Described by a non-linear equation:

$$\boldsymbol{r}(t) = R\,[\cos(2\pi t)\,\boldsymbol{e}_x + \sin(2\pi t)\,\boldsymbol{e}_y] + h\,t\,\boldsymbol{e}_z$$



In general, difficult to solve:

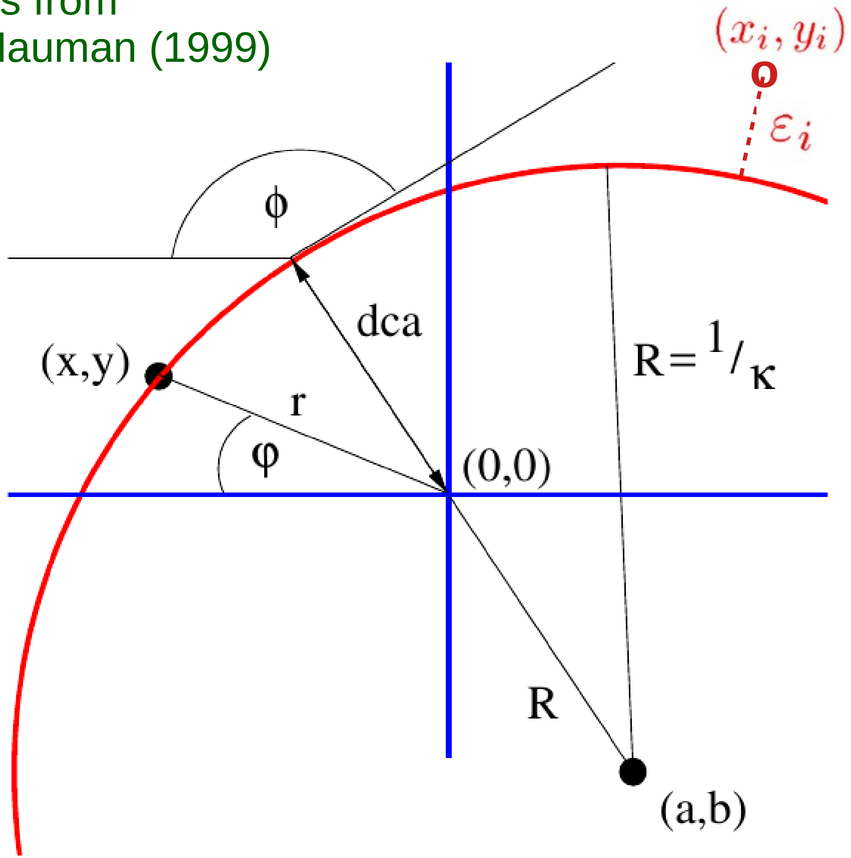
- hit errors need to be **projected** on trajectory
- minimisation problem is **non-linear**

However, for the case of simple hit weights an **algebraic solution** exists for **circle** fit:

- **circle fit from Karimaki** (1991)

# Karimaki Circle Fit

plots from
J. Nauman (1999)



New parameters defined:

dca = distance of closest approach to origin (aka $d_0$)

Φ = initial angle at dca

κ = 1/R = curvature of radius

Closest distance between hit and circle:

$$\varepsilon_i = \pm \left[ \sqrt{(x_i - a)^2 + (y_i - b)^2} - R \right]$$

Task: minimise $\chi^2$ with respect to → κ, dca, Φ:

$$\chi^2 = \sum_i w_i \epsilon_i^2$$

($w_i$ are weights)

# Parameter Transformations

If hits are positioned close to circle:

$$\varepsilon_i \overset{|\varepsilon_i| \ll R}{\approx} \pm R^{-1}\left[(x_i - a)^2 + (y_i - b)^2 - R^2\right]$$

$$\boxed{\chi^2 = \sum_i w_i \epsilon_i^2}$$

Problems:

- parameters *a, b, R* can become very large (high momentum tracks) → numerical unstable
- uncertainties (for example on *R*) are not Gaussian distributed!

1. Switch to polar coordinates and use new parameters:

$$\varepsilon_i = \frac{1}{2}\kappa r_i^2 - (1 + \kappa d_{ca})\, r_i \sin(\phi - \varphi_i) + \frac{1}{2}\kappa d_{ca}^2 + d_{ca}$$

2. simplify expression further by transforming $\chi^2$ function:

$$\chi^2 = (1 + \kappa d_{ca})\,\hat{\chi}^2$$

$$\varepsilon_i = (1 + \kappa d_{ca})\,\eta_i$$

and minimisation of

$$\boxed{\hat{\chi}^2 = \sum_i w_i \eta_i^2}$$

result depends now weakly on position of origin!

# Results

Parameters:

$$\begin{aligned}\kappa &= \frac{2\beta}{\sqrt{1-4\delta\beta}} \\ d_{ca} &= \frac{2\delta}{1+\sqrt{1-4\delta\beta}} \\ \phi &= \frac{1}{2}\mathrm{arctan}\left(\frac{2q_1}{q_2}\right)\end{aligned}$$

Geometry parameters:

$$q_1 = C_{r^2r^2}C_{xy} - C_{xr^2}C_{yr^2}$$
$$q_2 = C_{r^2r^2}(C_{xx} - C_{yy}) - C^2_{xr^2} + C^2_{yr^2}$$

$$\phi = 1/2\mathrm{arctan}(2q_1/q_2)$$
$$\beta = (\sin\phi C_{xr^2} - \cos\phi C_{yr^2})/C_{r^2r^2}$$

$$\delta = -\beta\langle r^2\rangle + \sin\phi\langle x\rangle - \cos\phi\,\langle y\rangle$$

$C_{pq}$ are the covariance of samples *p* and *q*

Fit quality (only approximate):

$$\chi^2 = S_w\,(1+\kappa d_{ca})^2\,(\sin^2\phi\, C_{xx} - 2\sin\phi\,\cos\phi\, C_{xy} + \cos^2\phi\, C_{yy} - \kappa^2 C_{r^2r^2})$$

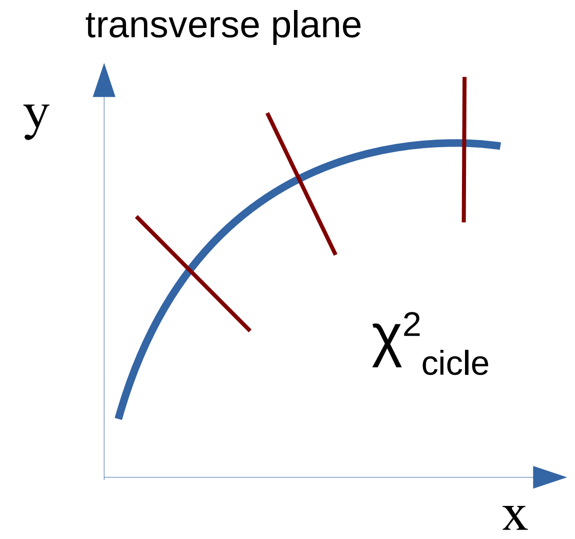
- non-iterative track fit
- provides error matrix (not shown)
- complexity of calculation a bit higher than for MS fit (but different regime)

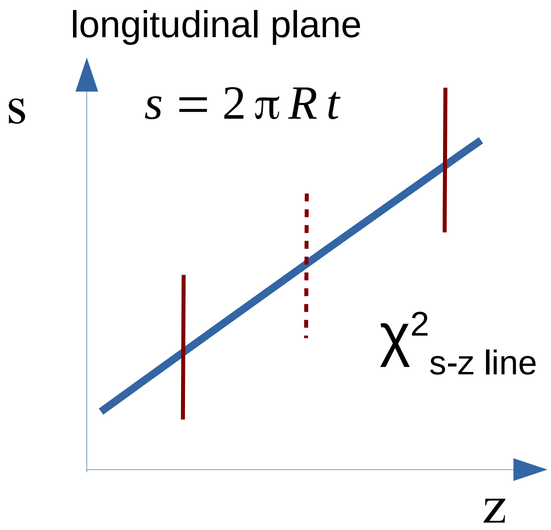
# Question for students:

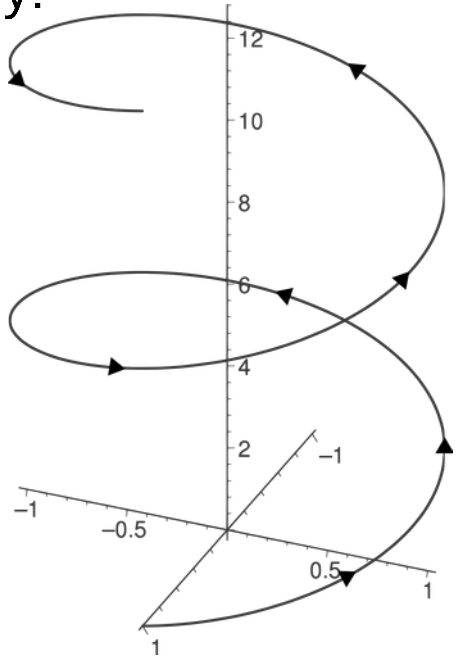
What are the advantages
of the circle fit?
What are the difficulties for
parallelisation?

# Helix Fit with Karimaki

“**2.5D tracking**”: fit transverse and longitudinal plane separately:







x-y fit provides:

- R = radius
- Φ = azimuth at dca
- dca = distance of closest approch

s-z fit provides:

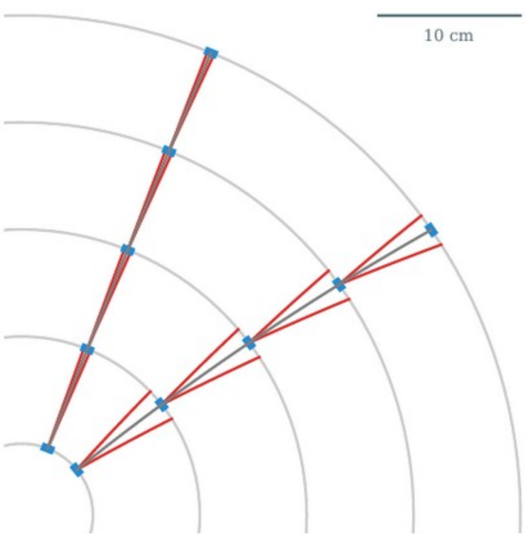
- abscissa $z_0$
- $\tan\Theta = \Delta s / \Delta z$

$$\boldsymbol{r}(t) = R\,[\cos(2\pi t)\boldsymbol{e}_x + \sin(2\pi t)\boldsymbol{e}_y] + h\,t\,\boldsymbol{e}_z$$

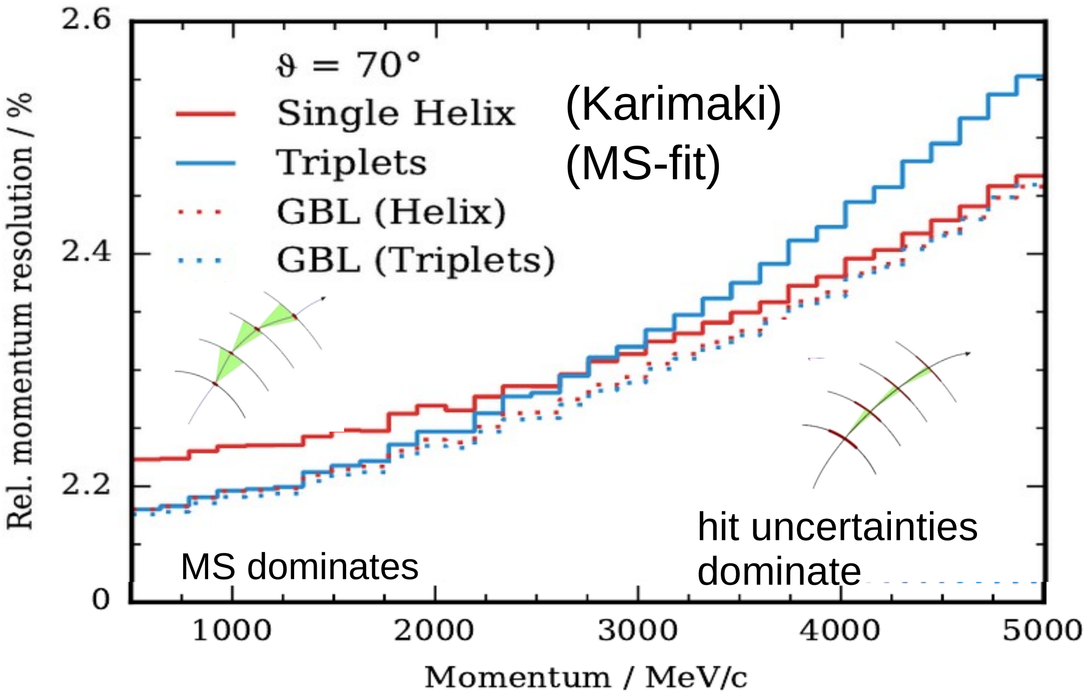
→ hit correlations between transverse and longitudinal plane are not considered!

# Comparing Results

plot from NIMA 844C, 135 (2017)
(aka MS-fit)



$x/X_0$ = 2% per layer
B-field=2Tesla



(Karimaki)
(MS-fit)

MS dominates

hit uncertainties dominate

GBL=General Broken Line

V. Blobel, NIMA, 566 (2006) 14.

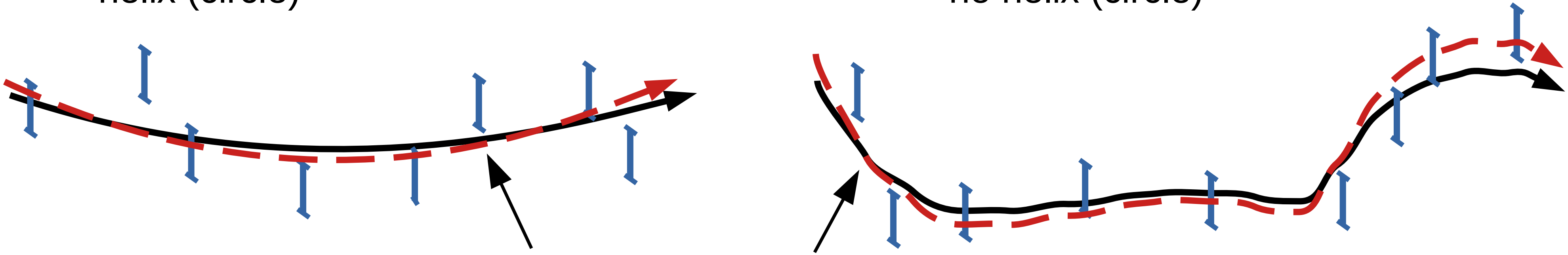
- MS-fit is 2-5 times faster than Karimäki
- GBL is about O(100) slower than the others

# Chapter 4
# Linearisation

# Linearisation & Linear Fit

A) homogeneous magnetic field
→ helix (circle)

B) non-homogeneous magnetic field
→ no helix (circle)



- **reference trajectories**
- **fitted trajectories**

If a reference trajectory close to the final resolution is given, the problem can be **linearised** by treating the hit displacements as **small corrections**

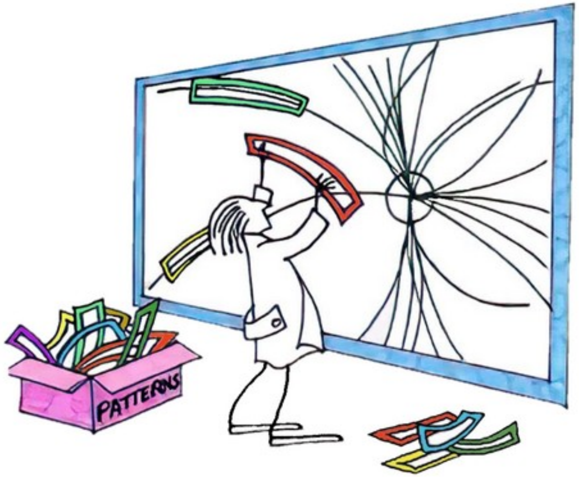
1. calculate hit positions and pulls with respect to reference trajectory
2. update **position $\bar{x}$** , **slope β** , **curvature radius R** (momentum)
3. can be **repeated** (iterated) for high **precision**

# Applications

**Linearised track fit is a good approach if**

- the **track parameters** are **roughly known** by a previous reconstruction step
  (e.g. pattern match, other track finding techniques)
- Tracks are known to be roughly **straight lines** (no B-field, high momentum tracks)

Example: **ATLAS FTK & HTT** track trigger projects (similar project in CMS)



- The different **roads** describe/contain bundles of **similar trajectories**
- The roads provide an **initial guess** of the **track parameters**

# Linearisation of a Circle/Helix Fit

Given a list of *p* track parameters (e.g. *R, Φ, dca, θ, z₀*):

$$p_{\mathrm{i}}^{\mathrm{true}}, i = 1, ..., p$$

and *N* hit displacements with respect to reference orbit:

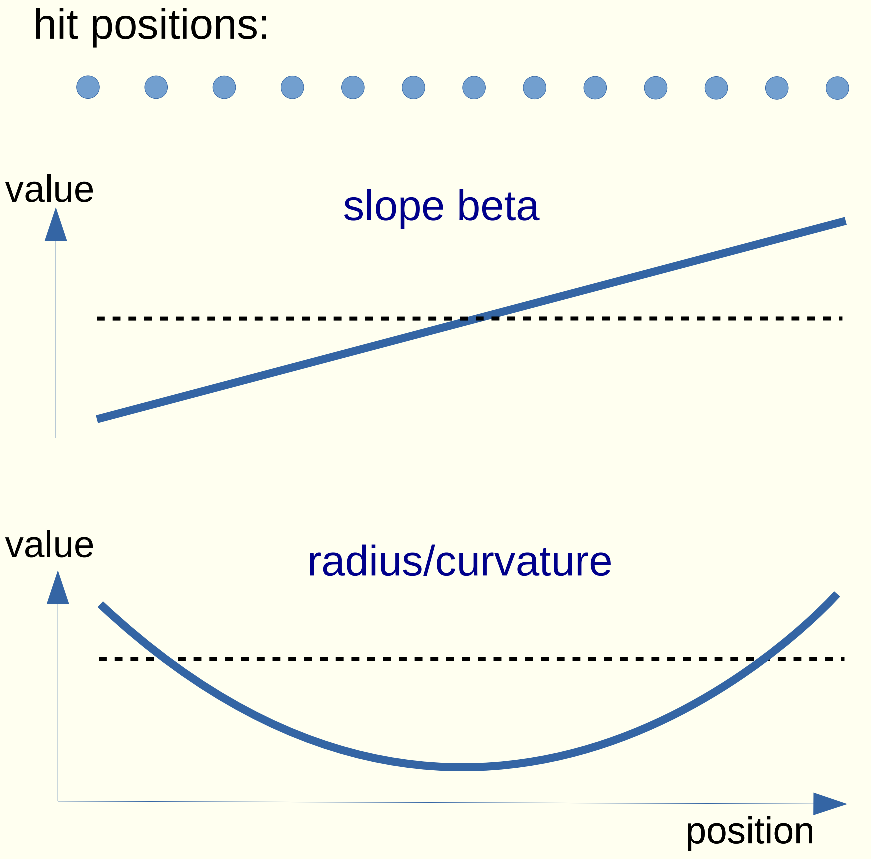
$$\delta x_{\mathrm{j}} = x_{\mathrm{j}} - \overline{x}_{\mathrm{j}}$$

Then the track is linearised using:

$$p_{\mathrm{i}} = \Sigma_{\mathrm{j}=1}^{N} A_{\mathrm{ij}} \delta x_{\mathrm{j}} + \overline{p}_{\mathrm{i}}$$

with coefficients $\boldsymbol{A_{ij}}$ (matrix of N x p coefficients)

Example for coefficients (weights):



# Linearisation of Fit Quality

**For track finding (good/bad) or a track trigger the fit quality is crucial!**

The calculation of the chi2 function can also be linearised using a principal component analysis:

$$\chi_i = \Sigma_{\mathrm{j}=1}^{N} B_{\mathrm{ij}} \delta x_{\mathrm{j}}$$

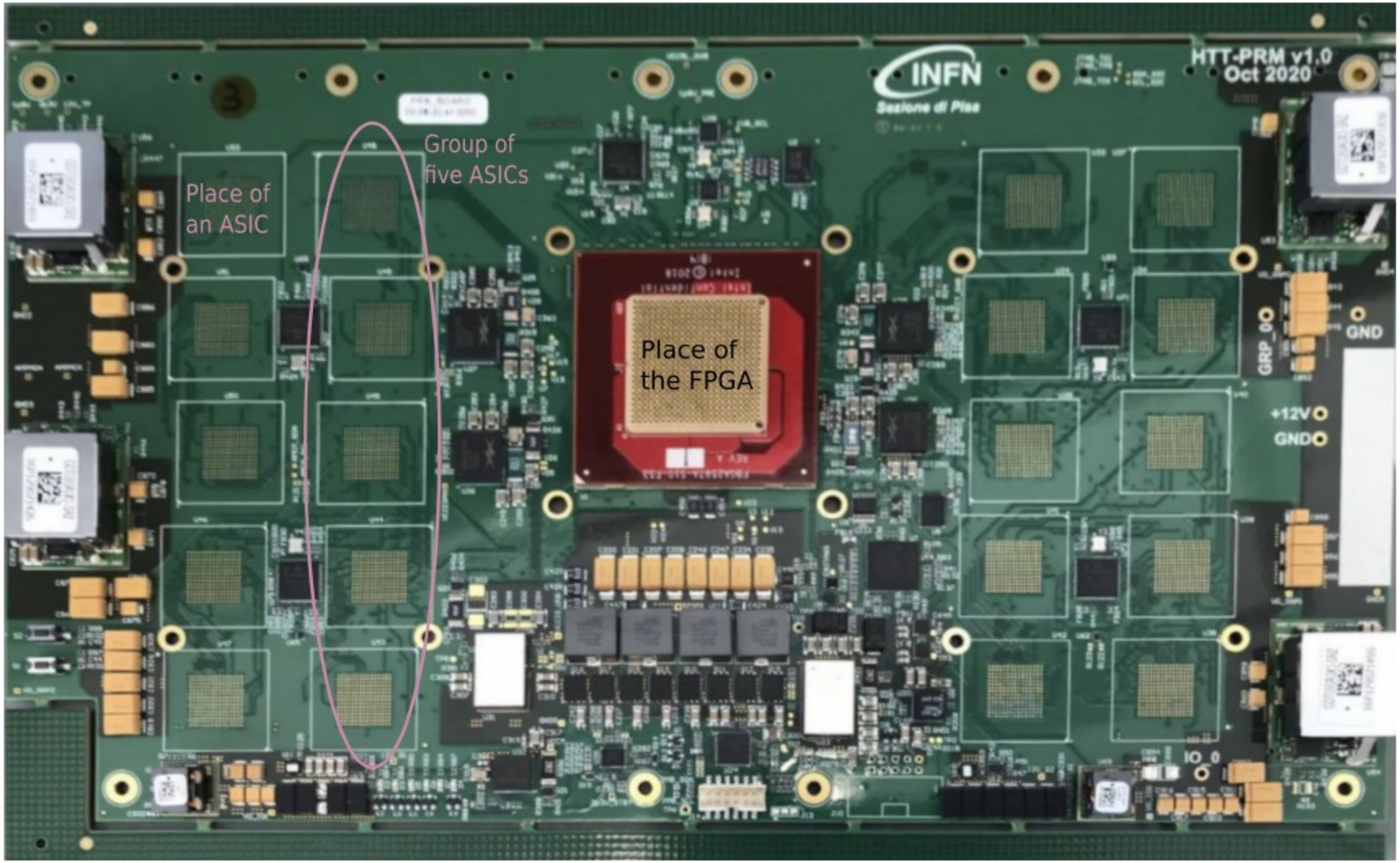
$$\chi^2 = \sum_{i=1}^{N-p} \chi_i^2$$

The coefficients $B_{ij}$ can be represented by a **N x (N-p)** matrix.

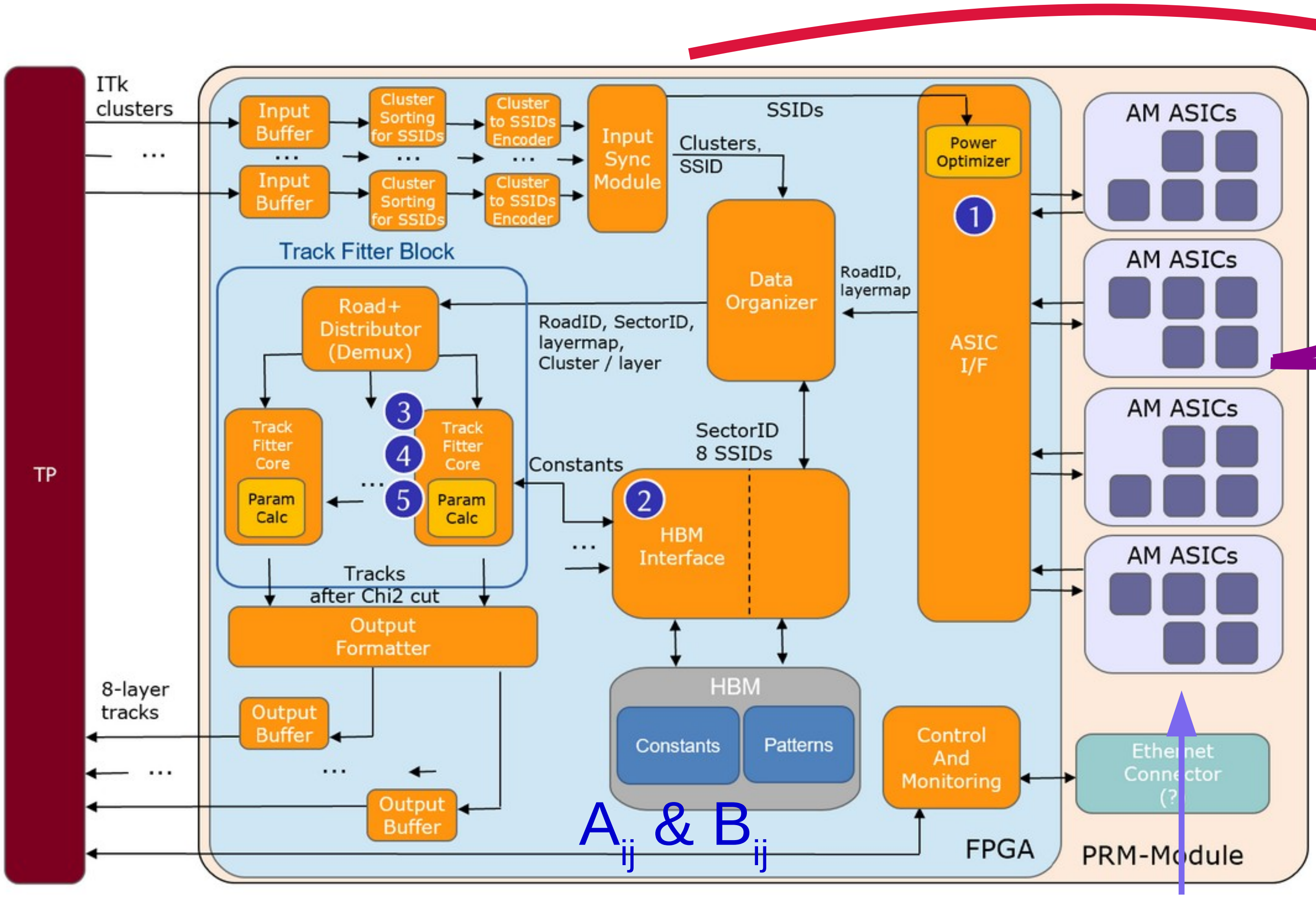
- Example: p=5 parameters, N=12 hits → 84 parameters
- not all coefficients are significant
- clever choice of parameters can reduce the complexity (→ extra slide)

# Example: ATLAS HTT

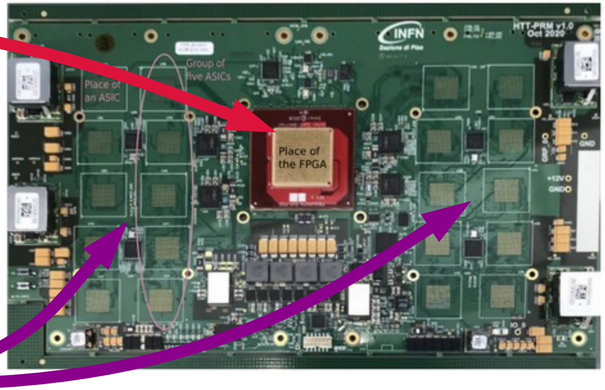
## Pattern Recognition Mezzanine



# Example: ATLAS HTT



$A_{ij}$ & $B_{ij}$



1. pattern match → ref. track
2. request constants
3. linearised $\chi^2$ calculation
4. apply $\chi^2$ cut
5. linearised parameter fit

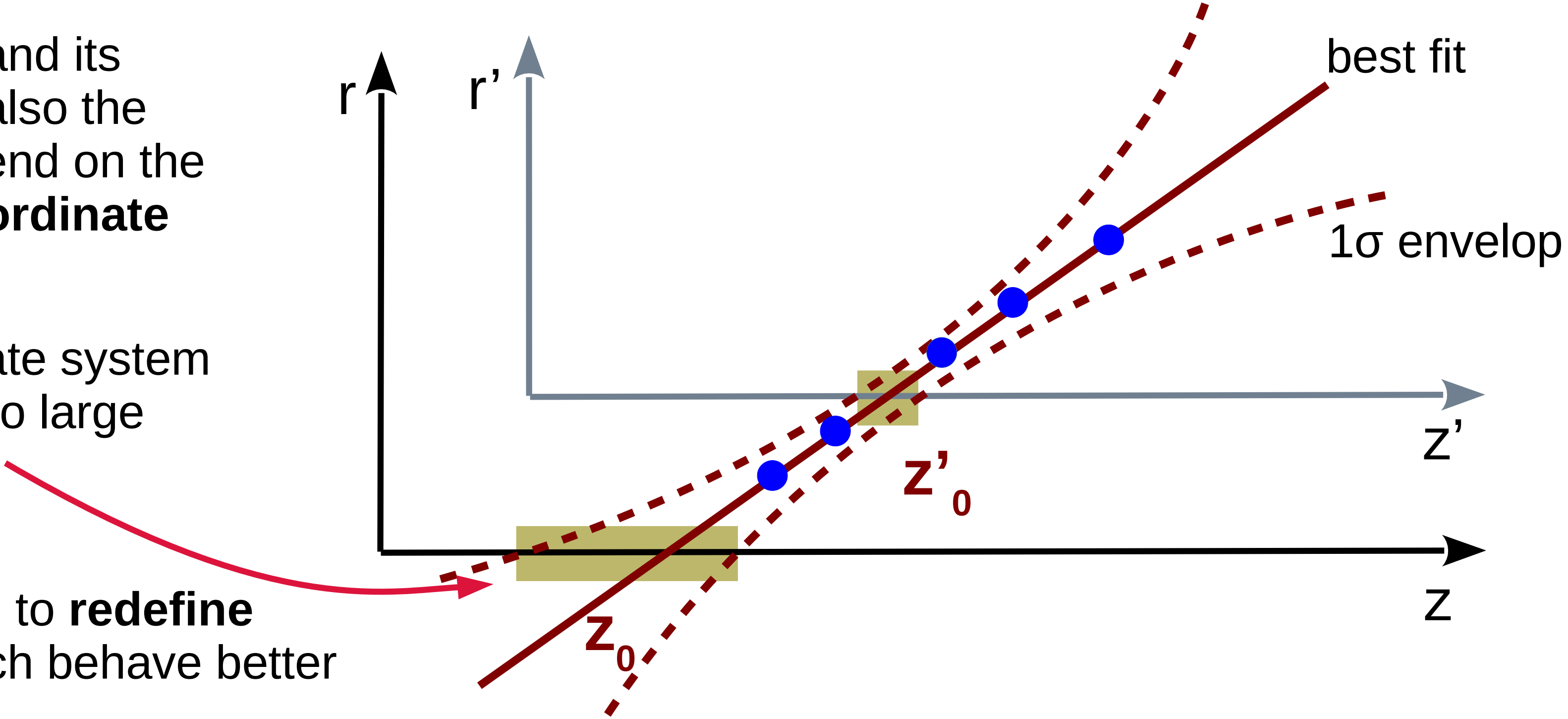
highly parallel

# Question to students:

What do you consider is most challenging and technologically ambitious in this design?

# Best Fitting Parameters?

- The fitted value and its uncertainty and also the correlations depend on the choice of the **coordinate system**!
- a wrong coordinate system choice can lead to large **non-linearities**
- it is also possible to **redefine parameters** which behave better in the fit



e.g. $$z' = z - \cot(\theta)(R - R') - \frac{\cot(\theta)R^3}{6(2\rho)^2}$$

from [arXiv:1809.01467]

# Chapter 5
# A New Hit Uncertainty Track Fit

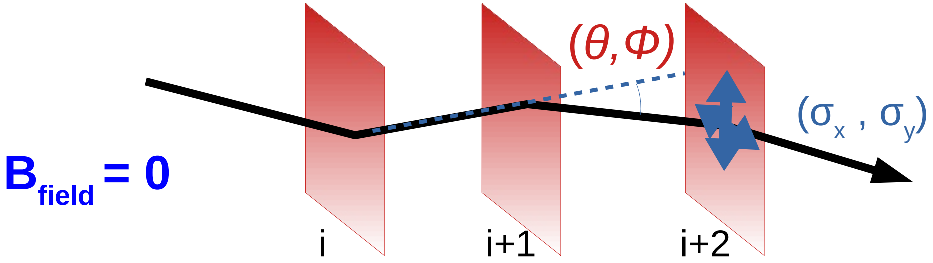
# Recap

Full track fit:

$$\chi^2 = \boxed{\sum_{\text{layer } i} \frac{\Theta^2_{\mathrm{MS},i}}{\sigma^2_{\theta,i}} + \frac{\Phi^2_{\mathrm{MS},i}}{\sigma^2_{\phi,i}}} + \boxed{\sum_{\text{hits } jk} (x_j - \xi_j) V^{-1}_{jk} (x_k - \xi_k)}$$

includes **multiple scattering** and **hit uncertainties**

- **MS fit** alone can be **parallelised**
- This parallelisation is based in **hit triplets**



- Hit uncertainty fit can be **linearised**

  → good for parallisation
- But linearisation needs a **reference** trajectory

## Question to students:

What is the next logical step?

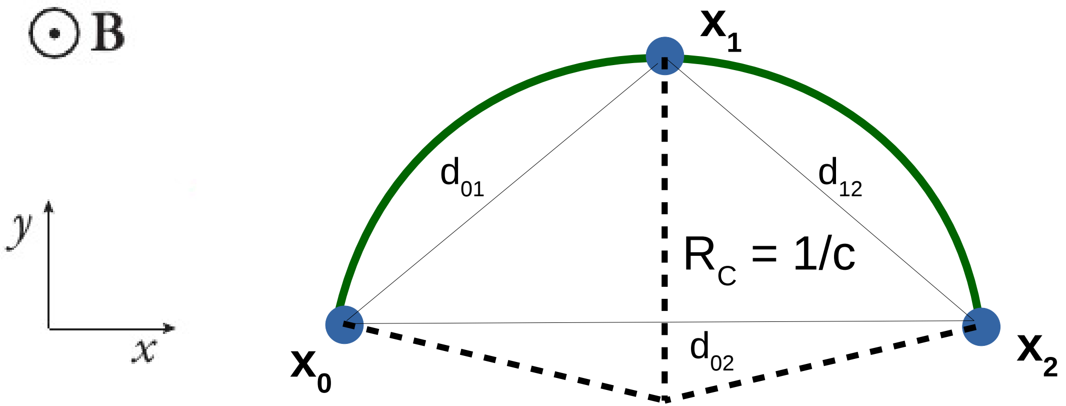
# Hit Triplets

- A hit **triplet** is the smallest tracking **element** which contains all track parameter information
- The **precision** of the triplet track parameters depend on the **lever arm** (size)
- For track finding, triplets are often used as **seeds** (combinatorics is small):

**seed finding = triplet finding**!

**Easy reconstruction in homogeneous magnetic field**

- **three points** can always be connected by a **circle**
- all track parameters can be calculates → **reference track**



Radius of circle:

$$R_C = \frac{d_{01}\; d_{12}\; d_{02}}{2\,[(\mathbf{x_1}-\mathbf{x_0})\times(\mathbf{x_2}-\mathbf{x_1})]_z}$$

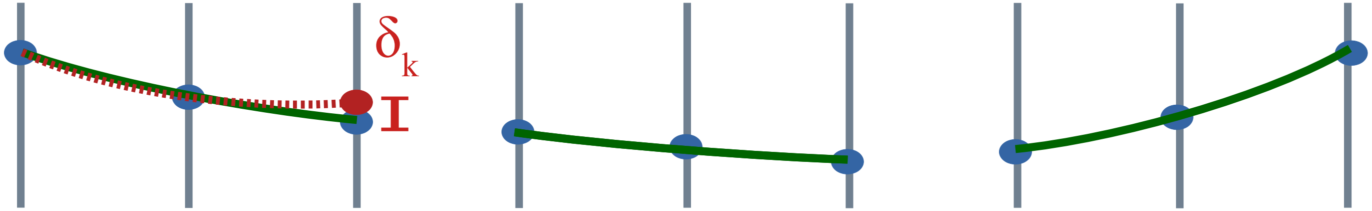
$$c = \frac{2\sin(\phi_{12}-\phi_{01})}{d_{02}}$$

(and permutations)

# Track Fit with N Hits

uncertainties are not shown

Possible **configuration** where all hits lie on their own reference trajectory



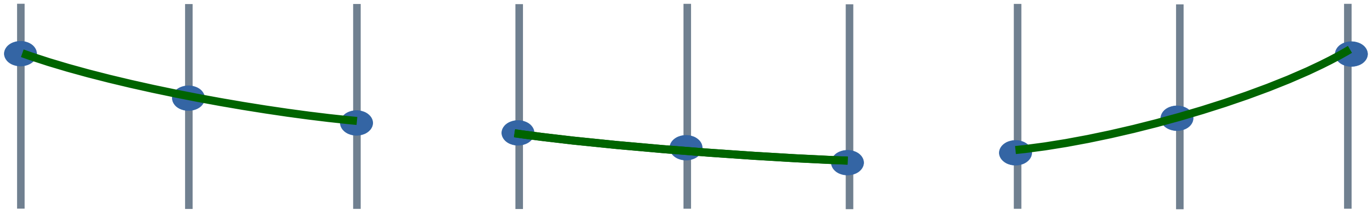
We can calculate how the track parameters change if we displace one point

$\bar{x}_j , \beta_j , R_j \rightarrow \bar{x}(\delta_k)_j , \beta(\delta_k)_j , R(\delta_k)_j$ (j=1,…, $N_{triplet}$)

Now we can also calculate a **weight** or fit quality for small displacements $\delta_k$

$$\chi_j^2 = \sum_{k=0}^{2} \frac{\delta_k^2}{\sigma_k^2} \longleftarrow \text{hit pull}$$

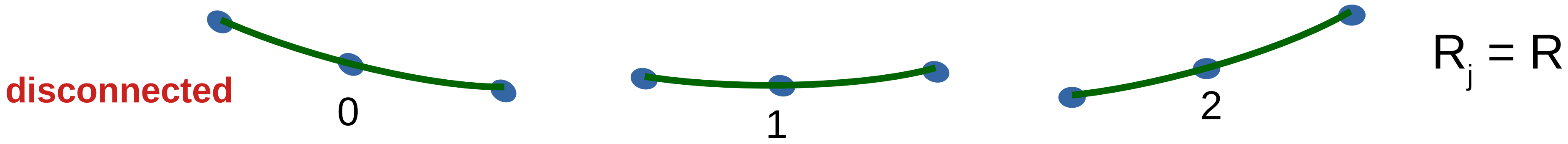
Idea of common fit → combine all hit triplets with the constraint: $R_j = R$



Question to students:

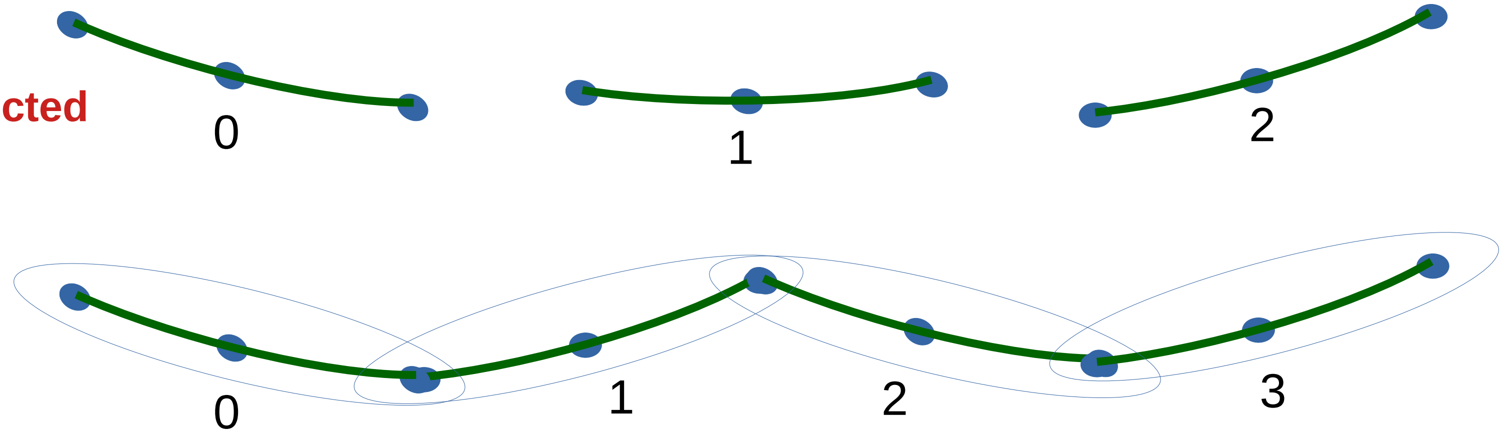
Will such a fit work?

# Correlations and triplet topologies



# Correlations and triplet topologies

**disconnected**

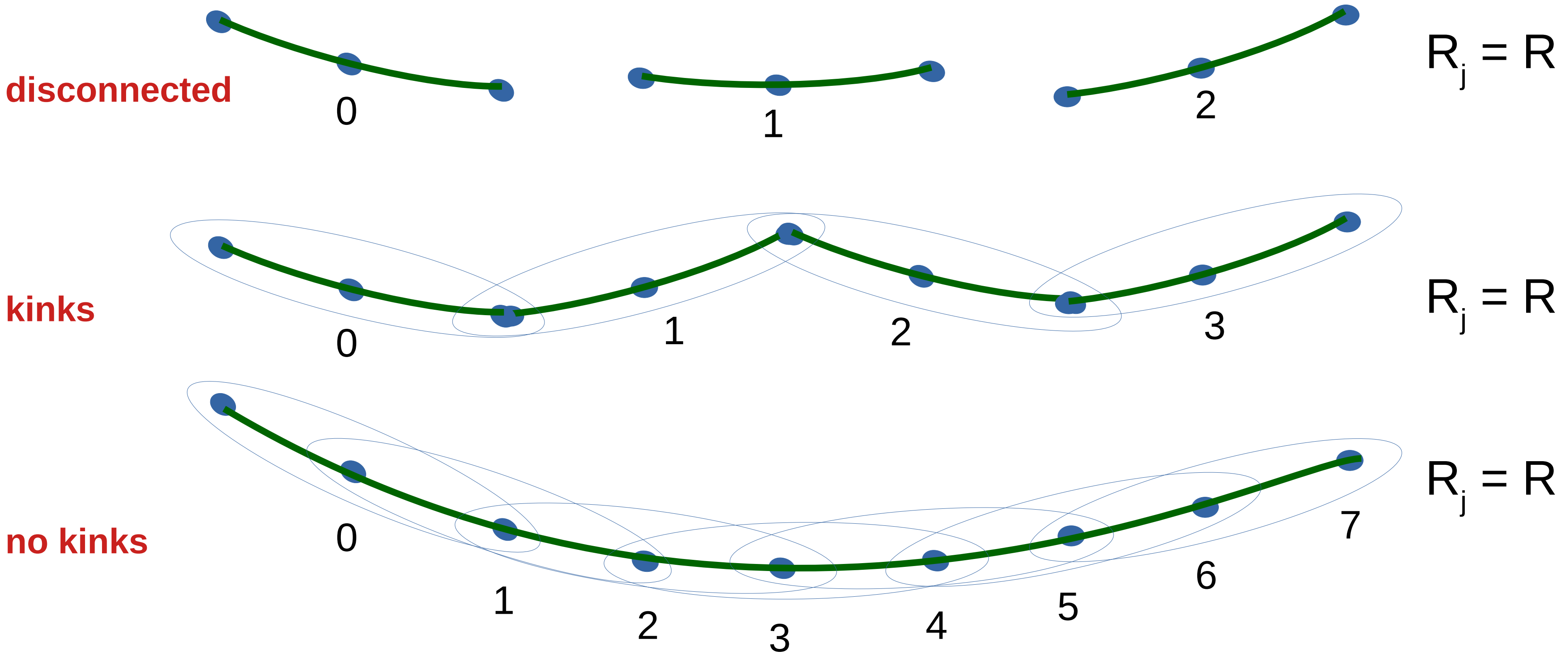


$R_j = R$

**kinks**

$R_j = R$

# Correlations and triplet topologies



correlations are automatically taken into account if **consecutive** triplets are considered

Question to students:

How to make a constraint fit?

# Method of Lagrange Multipliers

Lagrange function

$$\mathcal{L}(x,\lambda) = f(x) - \lambda g(x)$$

to be minimised　　constraint g(x)→0

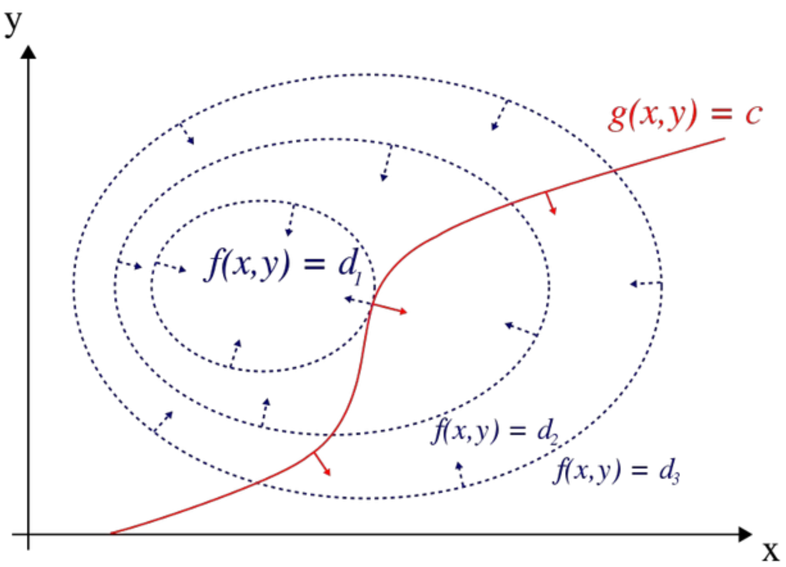
Minimisation of this Lagrangian results in:

$$Df(x^*) = \lambda^{*T} Dg(x^*)$$

with the partial derivatives: $D := \frac{\partial}{\partial x_k}$

The minimisation results in a **system of equations** yielding the new fitted hit positions x*

**Since the displacements $x^*$ are small, the system can be linearised and solved!**



The Lagrange parameters $\lambda_k$ is the rate of change of the quantity being optimized as a function of the constraint parameter (from Wikipedia)

**In other words, $\lambda_k$ described how well the radius/curvature is measured!**

# Lagrangian for Hit Uncertainty Fit

here use c=1/R (curvature),
which is numerically more stable

$$\mathcal{L}(c;\delta_k,\lambda_j) = \sum_{\text{hit } k}^{n_{\text{hit}}} \frac{\delta_k^2}{\sigma_k^2} + \sum_{\text{triplet } j}^{n_{\text{hit}}-2} \lambda_j \; [c_j(\delta_j,\delta_{j+1},\delta_{j+2}) - c]$$

note c=1/R

L.M.

hit pulls

reference trajectory

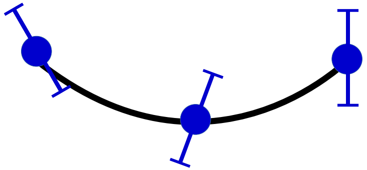
to be fitted curvature

Linearisation:

$$\begin{aligned} c_j &= \tilde{c}_j + \sum_{k=0}^{3} \frac{\mathrm{d}c_j}{\mathrm{d}\delta_{j+k}} \delta_{j+k} \\ &= \tilde{c}_j + \Xi_j \left( \frac{(\tau_0)_j}{(d_{01})_j}\delta_j + \frac{(\tau_3)_j}{(d_{12})_j}\delta_{j+2} - \left( \frac{(\tau_1)_j}{(d_{01})_j} + \frac{(\tau_2)_j}{(d_{12})_j} \right) \delta_{j+1} \right) \end{aligned}$$

projection parameters

with: $$\Xi_j = \sqrt{\frac{4}{(d_{02})_j^2} - \tilde{c}_j^2}$$



# Lagrangian for Hit Uncertainty Fit

Minimisation $\frac{\partial \mathcal{L}}{\partial \delta_k} = 0$ (hits) and $\frac{\partial \mathcal{L}}{\partial \lambda_j} = 0$ (constraints) yields:

$$\begin{pmatrix} 2/\sigma_0^2 & 0 & 0 & 0 & 0 & (\xi_0)_0 & 0 & 0 \\ 0 & 2/\sigma_1^2 & 0 & 0 & 0 & -(\xi_{12})_0 & (\xi_0)_1 & 0 \\ 0 & 0 & 2/\sigma_2^2 & 0 & 0 & (\xi_3)_0 & -(\xi_{12})_1 & (\xi_0)_2 \\ 0 & 0 & 0 & 2/\sigma_3^2 & 0 & 0 & (\xi_3)_1 & -(\xi_{12})_2 \\ 0 & 0 & 0 & 0 & 2/\sigma_4^2 & 0 & 0 & (\xi_3)_2 \\ (\xi_0)_0 & -(\xi_{12})_0 & (\xi_3)_0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (\xi_0)_1 & -(\xi_{12})_1 & (\xi_3)_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & (\xi_0)_2 & -(\xi_{12})_2 & (\xi_3)_2 & 0 & 0 & 0 \end{pmatrix} \underbrace{\begin{pmatrix} \delta_0 \\ \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \lambda_0 \\ \lambda_1 \\ \lambda_2 \end{pmatrix}}_{\text{hits and Lagrange multipliers}} = \underbrace{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ c - \tilde{c}_0 \\ c - \tilde{c}_1 \\ c - \tilde{c}_2 \end{pmatrix}}_{\text{fitted curvature}} \begin{array}{l} \left.\vphantom{\begin{matrix}0\\0\\0\\0\\0\end{matrix}}\right\} N_{\text{hits}} \\ \left.\vphantom{\begin{matrix}0\\0\\0\end{matrix}}\right\} N_{\text{triplets}} \end{array}$$

with

$$(\xi_0)_j = \frac{\Xi_j \, (\tau_0)_j}{(d_{01})_j}$$

$$(\xi_{12})_j = \frac{\Xi_j \, (\tau_1)_j}{(d_{01})_j} + \frac{\Xi_j \, (\tau_2)_j}{(d_{12})_j}$$

$$(\xi_3)_j = \frac{\Xi_j \, (\tau_3)_j}{(d_{12})_j}$$

**Note the symmetry of the matrix!**

**How to solve the system of equations?**

# Lagrangian for Hit Uncertainty Fit

Minimisation $\frac{\partial \mathcal{L}}{\partial \delta_k} = 0$ (hits) and $\frac{\partial \mathcal{L}}{\partial \lambda_j} = 0$ (constraints) yields:

$$\left(\begin{array}{ccccc|ccc}
2/\sigma_0^2 & 0 & 0 & 0 & 0 & (\xi_0)_0 & 0 & 0 \\
0 & 2/\sigma_1^2 & 0 & 0 & 0 & -(\xi_{12})_0 & (\xi_0)_1 & 0 \\
0 & 0 & 2/\sigma_2^2 & 0 & 0 & (\xi_3)_0 & -(\xi_{12})_1 & (\xi_0)_2 \\
0 & 0 & 0 & 2/\sigma_3^2 & 0 & 0 & (\xi_3)_1 & -(\xi_{12})_2 \\
0 & 0 & 0 & 0 & 2/\sigma_4^2 & 0 & 0 & (\xi_3)_2 \\
\hline
(\xi_0)_0 & -(\xi_{12})_0 & (\xi_3)_0 & 0 & 0 & 0 & 0 & 0 \\
0 & (\xi_0)_1 & -(\xi_{12})_1 & (\xi_3)_1 & 0 & 0 & 0 & 0 \\
0 & 0 & (\xi_0)_2 & -(\xi_{12})_2 & (\xi_3)_2 & 0 & 0 & 0
\end{array}\right)
\begin{pmatrix} \delta_0 \\ \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \lambda_0 \\ \lambda_1 \\ \lambda_2 \end{pmatrix}
=
\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ c - \tilde{c}_0 \\ c - \tilde{c}_1 \\ c - \tilde{c}_2 \end{pmatrix}
\begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \\ \end{array}\right\} N_{\text{hits}} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} N_{\text{triplets}} \end{array}$$

hits and Lagrange multipliers

fitted curvature

with

$$\begin{aligned}
(\xi_0)_j &= \frac{\Xi_j\,(\tau_0)_j}{(d_{01})_j} \\
(\xi_{12})_j &= \frac{\Xi_j\,(\tau_1)_j}{(d_{01})_j} + \frac{\Xi_j\,(\tau_2)_j}{(d_{12})_j} \\
(\xi_3)_j &= \frac{\Xi_j\,(\tau_3)_j}{(d_{12})_j}
\end{aligned}$$

**Note the symmetry of the matrix!**

**How to solve the system of equations?**

# Structure of Equations

system of equations

$$\mathbf{M} \cdot \begin{pmatrix} \vec{\delta} \\ \vec{\lambda} \end{pmatrix} = \begin{pmatrix} \vec{0} \\ c\vec{1} - \vec{\hat{c}} \end{pmatrix}$$

$$\begin{aligned} \vec{\xi'_j} &= (\xi_0,\ \xi_{12},\ \xi_3)^T_j \\ \vec{\delta} &= (\delta_0, \delta_1, \ldots, \delta_{n_{\mathrm{hit}}-1})^T \\ \vec{\lambda} &= (\lambda_0, \lambda_1, \ldots, \lambda_{n_{\mathrm{triplet}}-1})^T \\ \vec{\hat{c}} &= (\hat{c}_0, \hat{c}_1, \ldots, \hat{c}_{n_{\mathrm{triplet}}-1})^T \end{aligned}$$

with

$$\mathbf{M} = \begin{pmatrix} \mathbf{D} & \mathbf{E} \\ \mathbf{E}^{\mathbf{T}} & \mathbf{0} \end{pmatrix}$$

$$\mathbf{D} = \mathrm{diag}(\frac{2}{\delta_0^2}, \frac{2}{\delta_1^2}, \ldots, \frac{2}{\delta_{k-1}^2})$$

$$\mathbf{E} = \left. \underbrace{\begin{pmatrix} \vec{\xi'_0} & 0 & \ldots & 0 \\ 0 & \vec{\xi'_1} & \ldots & 0 \\ \ldots & \ldots & \ldots & 0 \\ 0 & 0 & \ldots & \vec{\xi'}_{n_{\mathrm{triplet}}-1} \end{pmatrix}}_{\mathrm{N}_{\mathrm{triplet}}} \right\} n_{\mathrm{hit}}$$

matrix size is $\mathbf{N}_{\mathbf{triplet}}$ **x** $\mathbf{N}_{\mathbf{hit}}$

# Solution

Ansatz for inverted matrix

$$\mathbf{M}^{-1} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^{\mathbf{T}} & \mathbf{C} \end{pmatrix}$$

with

$$\begin{aligned} \mathbf{C} &= -\left(\mathbf{E}^T\,\mathbf{D}^{-1}\,\mathbf{E}\right)^{-1} = \mathbf{C}^T \\ \mathbf{B} &= -\mathbf{D}^{-1}\,\mathbf{E}\,\mathbf{C} \end{aligned}$$

Task is the inversion of the matrix: $\mathbf{P} = \mathbf{E}^T\,\mathbf{D}^{-1}\,\mathbf{E}$ (rank $N_{\text{triplet}}$)

$$\mathbf{P} = \begin{pmatrix} p_{00} & p_{01} & p_{02} & 0 & 0 & \dots & 0 \\ p_{10} & p_{11} & p_{12} & p_{13} & 0 & \dots & 0 \\ p_{20} & p_{21} & p_{22} & p_{23} & p_{24} & \dots & 0 \\ 0 & p_{31} & p_{32} & p_{33} & p_{34} & \dots & 0 \\ 0 & 0 & p_{42} & p_{43} & p_{44} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots & p_{n-1,n-1} \end{pmatrix}$$

P is a symmetric and sparse matrix. It can be inverted be the method of **Cholesky Decomposition**

usually a small matrix!

similar problem appears in the (related) broken line fit

# Result

The **fitted curvature** is given by a weighted sum of the triplet curvatures:

$$c_0 = \frac{\sum_j w_j \hat{c}_j}{\sum_j w_j} \qquad \text{with} \qquad w_j = -\frac{1}{2}\sum_i \mathbf{C}_{ij}$$

$$\sigma_c^{-2} = \frac{1}{2}\left(\frac{\mathrm{d}^2\chi^2(c)}{\mathrm{d}c^2}\right)_{(c=c_0)} = \frac{1}{2}\sum_{i,j}(\mathbf{B}^T\mathbf{D}\mathbf{B})_{ij} = -\frac{1}{2}\sum_{i,j}\mathbf{C}_{ij}$$

The **hit positions** are obtained from:

$$\delta_k = \sum_j B_{kj}(c - \hat{c}_j)$$

$$\sigma_{\mathrm{hit},k} = \sigma_c \sum_j B_{kj}$$

The **correlation** between hits can be calculated as well:

$$\mathbf{cov}_{\mathrm{hit},kl} = \sigma_c^2 \left(\sum_j B_{kj}\right)\left(\sum_i B_{li}\right)$$

→ **the fit provides all information!**

# Cholesky Decomposition

Want to solve: $A\ x = b$

Matrix A is symmetric, so we can write:

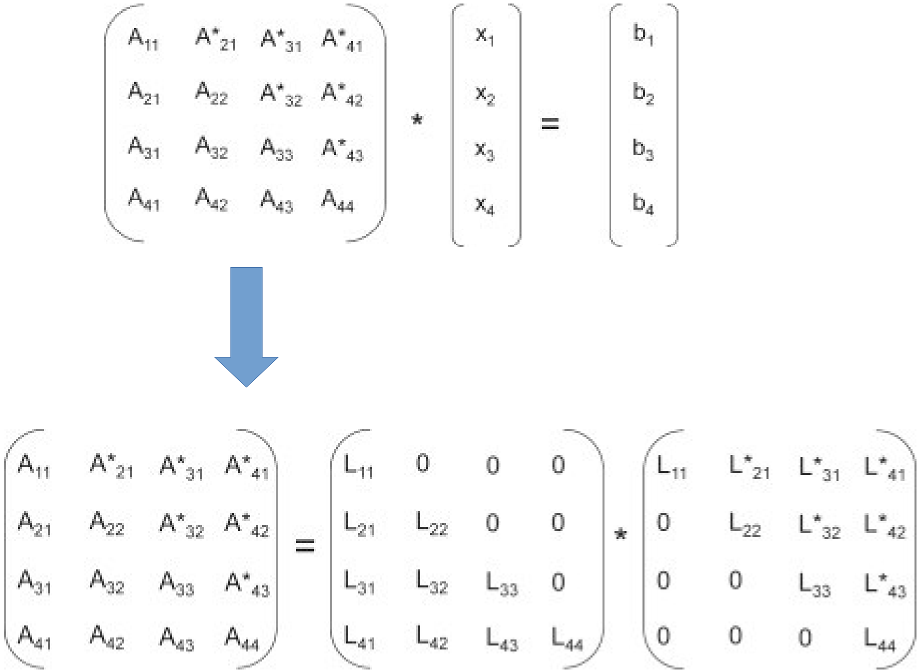
$$A = L L^T$$

with L being a left-sided matrix.

(alternatively one can also use: $A = L D L^T$

If L is known (→ next page)
the matrix inversion is done by a recursive
**A) forward** and
**B) backward substitution:**

**A)** $L\ y = b$ **B)** $L^T\ x = y$



$$\begin{pmatrix} A_{11} & A^*_{21} & A^*_{31} & A^*_{41} \\ A_{21} & A_{22} & A^*_{32} & A^*_{42} \\ A_{31} & A_{32} & A_{33} & A^*_{43} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} * \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}$$

$$\begin{pmatrix} A_{11} & A^*_{21} & A^*_{31} & A^*_{41} \\ A_{21} & A_{22} & A^*_{32} & A^*_{42} \\ A_{31} & A_{32} & A_{33} & A^*_{43} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{pmatrix} * \begin{pmatrix} L_{11} & L^*_{21} & L^*_{31} & L^*_{41} \\ 0 & L_{22} & L^*_{32} & L^*_{42} \\ 0 & 0 & L_{33} & L^*_{43} \\ 0 & 0 & 0 & L_{44} \end{pmatrix}$$

Note, this a sequential algorithm – not parallelisable!

# Calculation of Matrix L

from Wikipedia

Cholesky–Banachiewicz algorithm

$$L_{j,j} = (\pm)\sqrt{A_{j,j} - \sum_{k=1}^{j-1} L_{j,k}^2},$$

$$L_{i,j} = \frac{1}{L_{j,j}}\left(A_{i,j} - \sum_{k=1}^{j-1} L_{i,k}L_{j,k}\right) \quad \text{for } i > j.$$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];
        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

- Note, that the algorithm needs to be executed in a predefined **sequential** order

- In general **NxN steps** are required, costs scale as $\mathbf{N^3}$ (sums)

- However, for a **pentadiagonal matrix** the algorithm scales as $\mathbf{N^2}$

- (Other algorithms for a tri-diagonal matrix scale as **N logN** )

# Example: Calculation of L and y and x

Cholesky–Banachiewicz algorithm

For 3x3 matrix:

**1**

$$\mathbf{L} = \begin{pmatrix} \sqrt{A_{11}} & 0 & 0 \\ A_{21}/L_{11} & \sqrt{A_{22} - L_{21}^2} & 0 \\ A_{31}/L_{11} & \left(A_{32} - L_{31}L_{21}\right)/L_{22} & \sqrt{A_{33} - L_{31}^2 - L_{32}^2} \end{pmatrix}$$

Example:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 5 \\ 1 & 5 & 21 \end{pmatrix} \Rightarrow L = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 4 & 2 \end{pmatrix}$$

$$L^T = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 4 \\ 0 & 0 & 2 \end{pmatrix}$$

**2** $L\ y = b$ :

$$\downarrow \begin{pmatrix} 1\,y_0 & 0 & 0 \\ 1\,y_0 & 1\,y_1 & 0 \\ 1\,y_0 & 4\,y_1 & 2\,y_2 \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix}$$

$$\Rightarrow y = \begin{pmatrix} b_0 \\ b_0 - b_1 \\ 1/2\,b_2 + 3/2\,b_1 - 2\,b_0 \end{pmatrix}$$

**3** $L^T\ x = y$ :

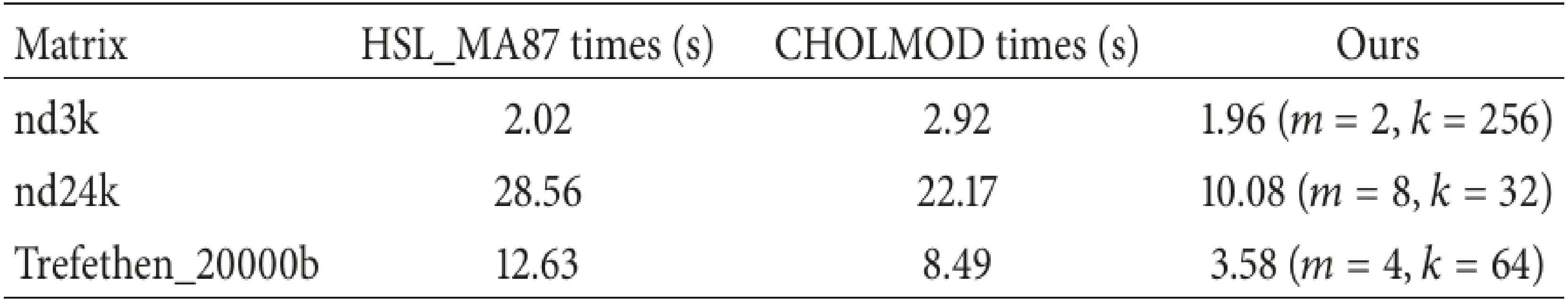
$$\begin{pmatrix} 1\,x_0 & 1\,x_1 & 1\,x_2 \\ 0 & 1\,x_1 & 4\,x_2 \\ 0 & 0 & 2\,x_2 \end{pmatrix} \uparrow = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix}$$

$$\Rightarrow x = \begin{pmatrix} y_0 - y_1 + 3\,y_2 \\ y_1 - 4\,y_2 \\ y_2 \end{pmatrix}$$

# Peformance of Choleski Decomposition

**Sparse Cholesky Factorization on FPGA Using Parameterized Model**
(Yichun Sun et al., https://doi.org/10.1155/2017/3021591)

different **big** matrices



| | **CPU** | **CPU-GPU** | **FPGA (200MHz)** |
|---|---|---|---|
| Matrix | HSL_MA87 times (s) | CHOLMOD times (s) | Ours |
| nd3k | 2.02 | 2.92 | 1.96 ($m = 2, k = 256$) |
| nd24k | 28.56 | 22.17 | 10.08 ($m = 8, k = 32$) |
| Trefethen_20000b | 12.63 | 8.49 | 3.58 ($m = 4, k = 64$) |

Conclusions

- FPGA implementation is about 2 times faster for large matrices
- The FPGA runs at O(10) times lower speed and consumes 50% of the power

**For hit finding with O(10) hits, the gain using FPGAs is probably small**

# Summary

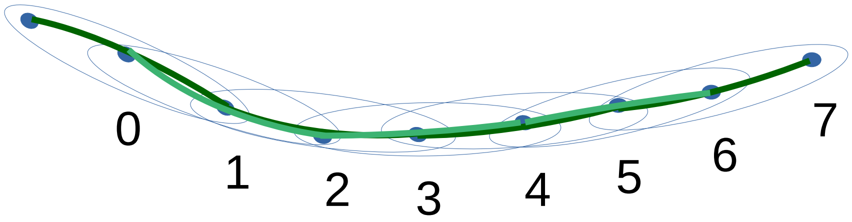
- **Linearisation** and **parallelisation** are very powerful methods to accelerate computations
- A new **parallelisable track fit** with hit uncertainties is presented (for 2D)
  - An extension to 3D is straight forward (not shown)
  - An extension to include multiple scattering can also easily be done (not shown)
    → **General Triplet Based Track Fit** (paper in preparation)
  - **Inverting** the sparse **matrix A** (e.g. Cholesky decomposition) is not parallelisable; but FPGA and GPU could maybe used for acceleration
- The proposed track fit is based on hit-triplets which are **ideal seeds for track finding** (→ graph theory)

# Backup

# Comparison: Triplet Fit versus GBL

## Triplet Model:

fit MS angles and hit positions with respect to **reference triplets**



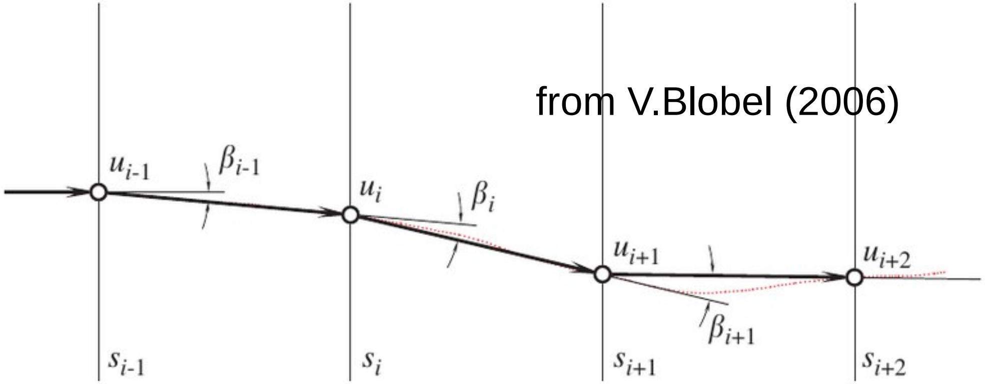
- **Single triplets** can be fitted including **MS and hit uncertainty algebraically** (not shown)
- triplets can be all fitted in **parallel** including **data preparation**

→ looks like an **ideal** algorithm for track finding?

- But speed of algo not measured yet

## GBL Model:

fit MS angles and hit positions with respect to a given **reference track**



from V.Blobel (2006)

Fig. 3. Particle trajectory with fitted residuals $u_i$ and kink angles $\beta_i$.

- **GBL** is faster than **Kalman** fitter
- GBL requires **reference trajectory** as input

→ therefore not suitable for track finding