Alignment of the CMS Silicon Tracker – and how to improve detectors in the future

Frank Meier
PSI Paul Scherrer Institut and ETH Zürich
on behalf of the CMS collaboration

Sep 10, 2010

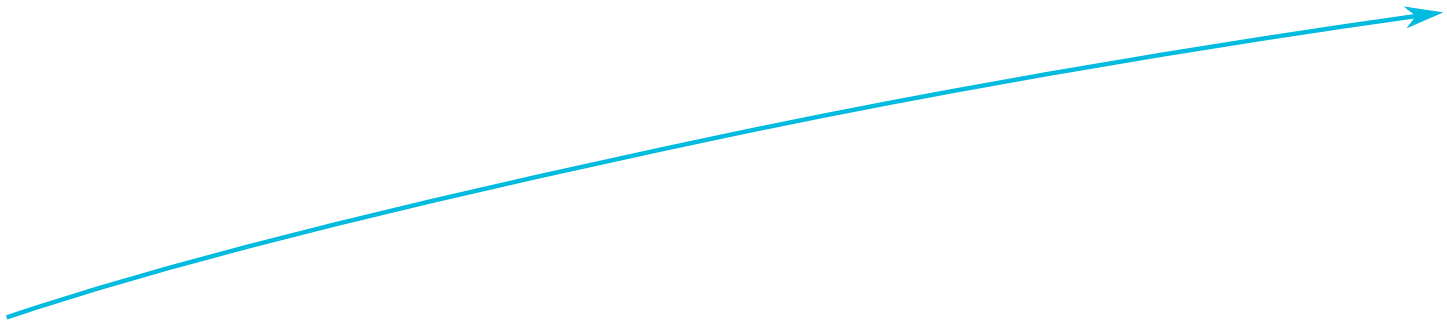
PAUL SCHERRER INSTITUT

Overview

PAUL SCHERRER INSTITUT

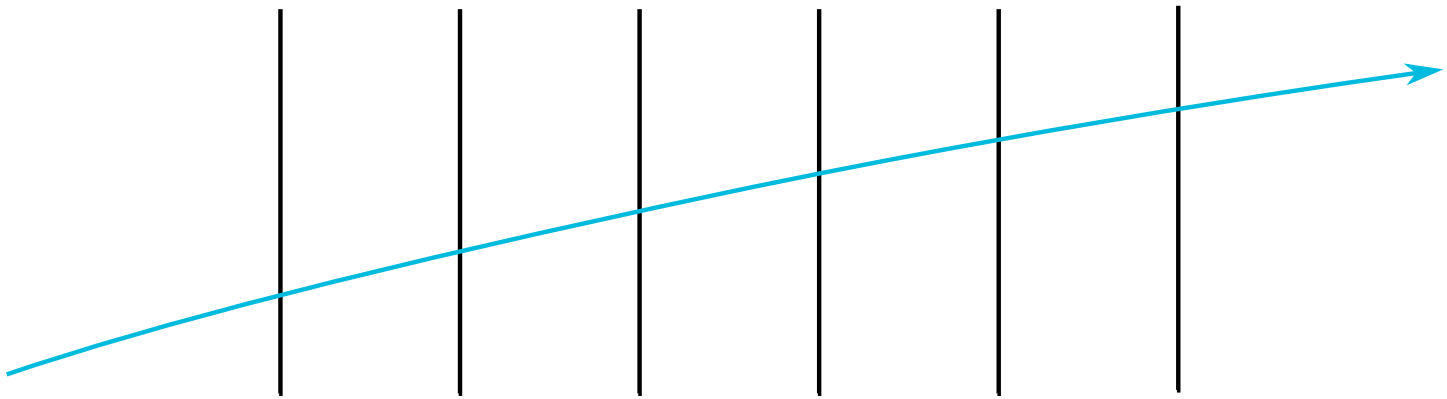
PSI

Track based alignment of a simple toy tracker



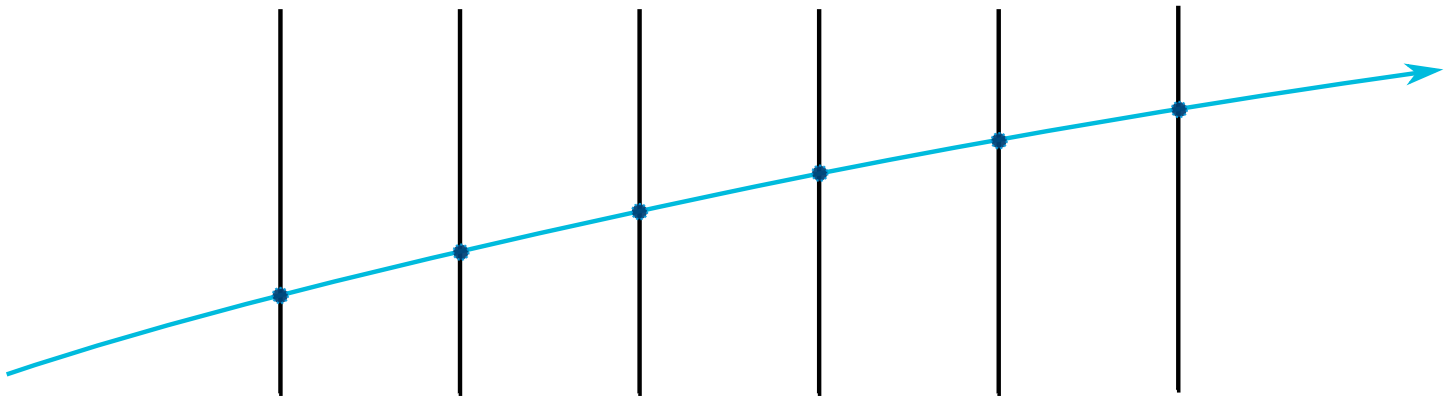
You want to track a charged particle in a magnetic field

Track based alignment of a simple toy tracker



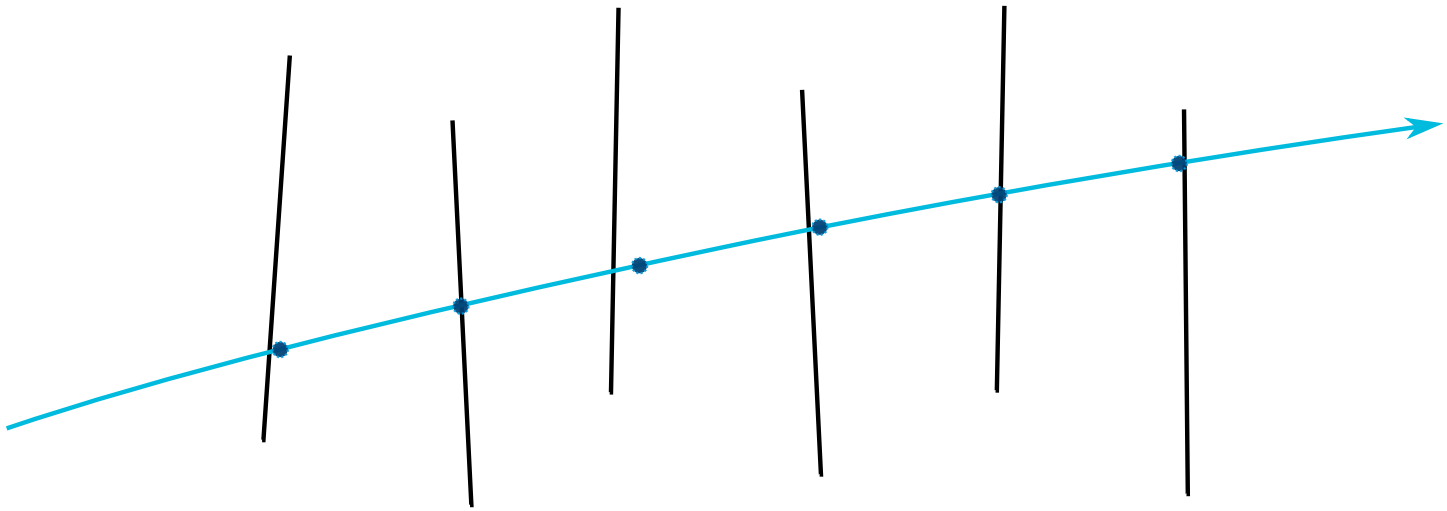
So you take six modules of your high-precision tracking detectors

Track based alignment of a simple toy tracker



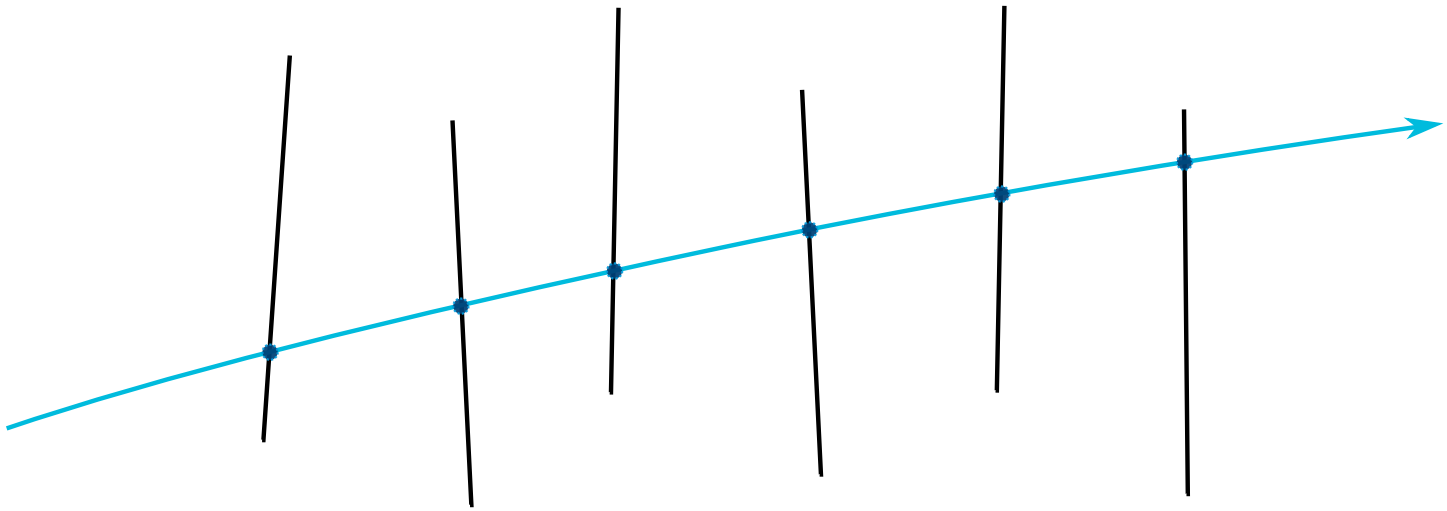
and you think you have recorded the right signal.

Track based alignment of a simple toy tracker



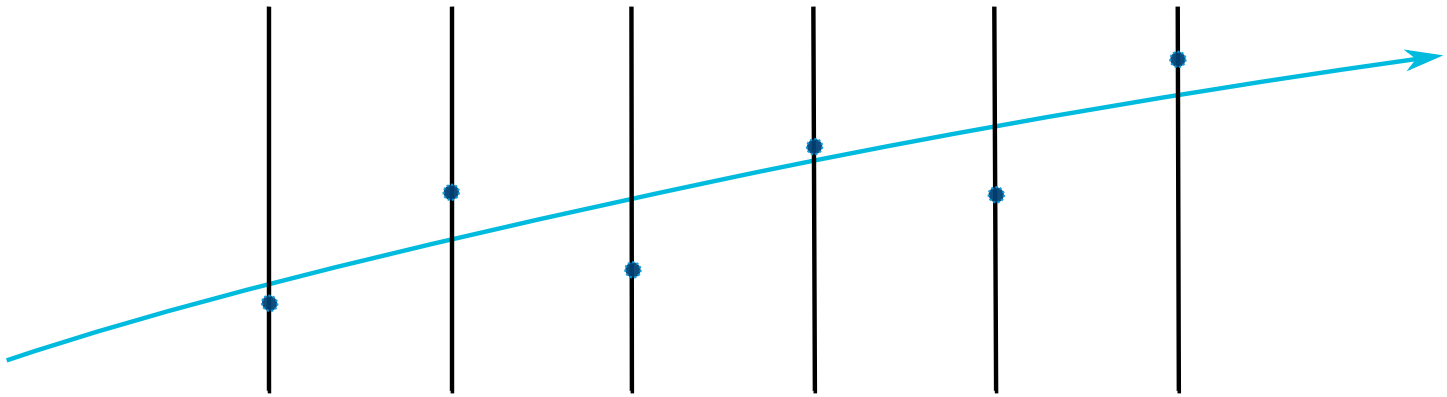
But unfortunately the modules were not mounted precisely where needed

Track based alignment of a simple toy tracker



and your hits are not where you thought

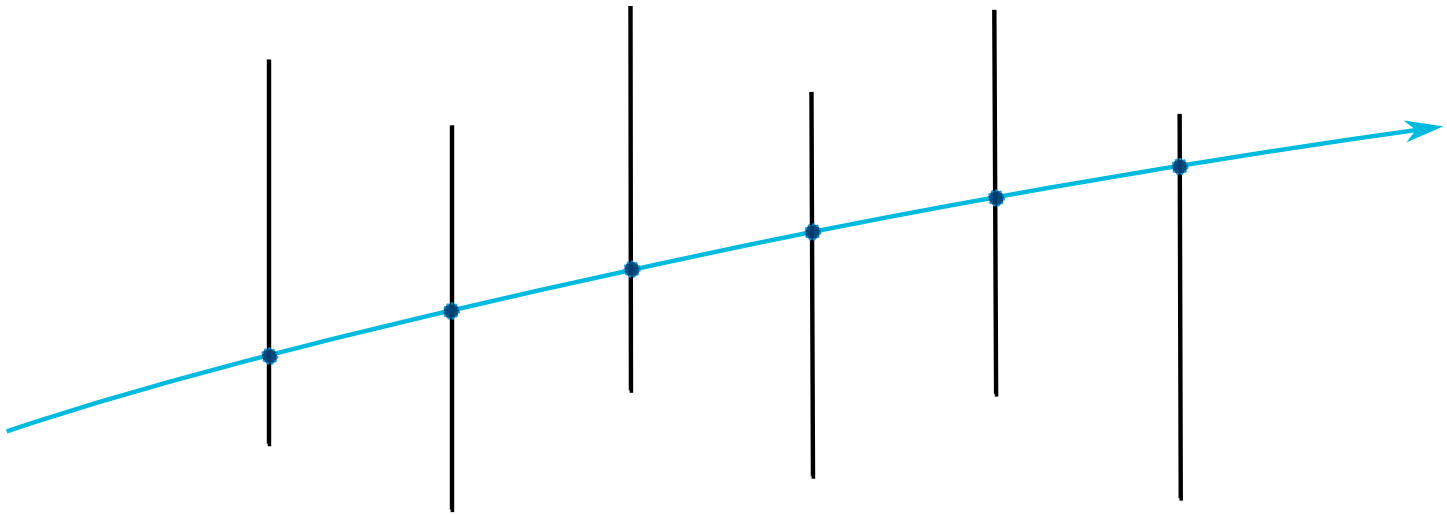
Track based alignment of a simple toy tracker



You still assume an ideal tracker and you know that the particle follows a smooth trajectory

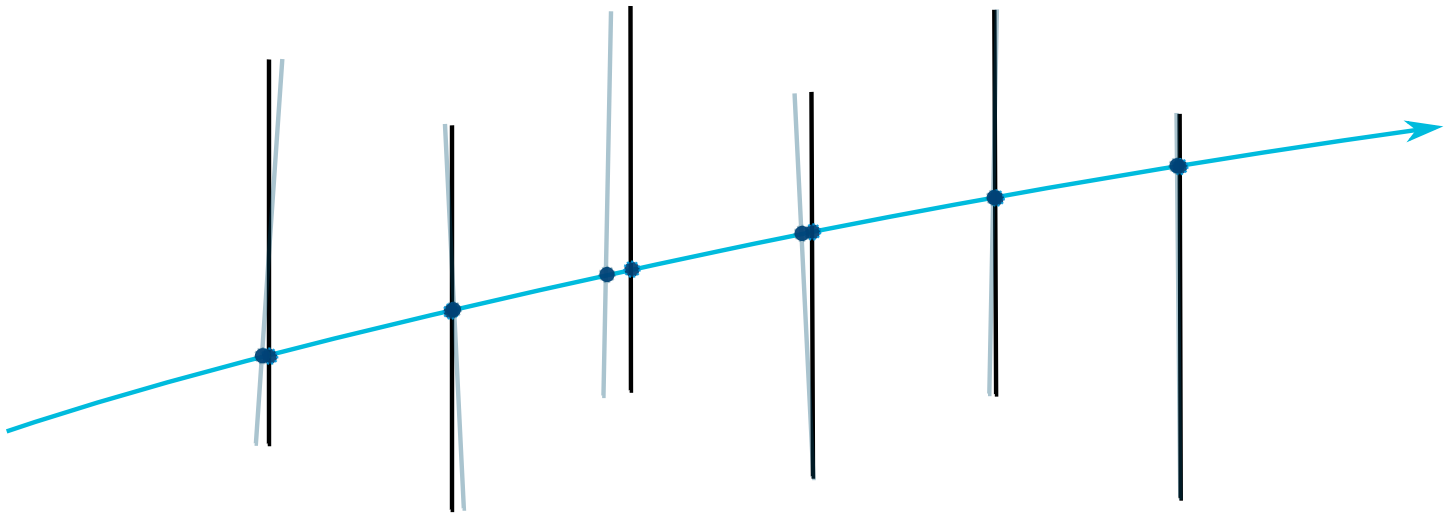
PAUL SCHERRER INSTITUT

Track based alignment of a simple toy tracker



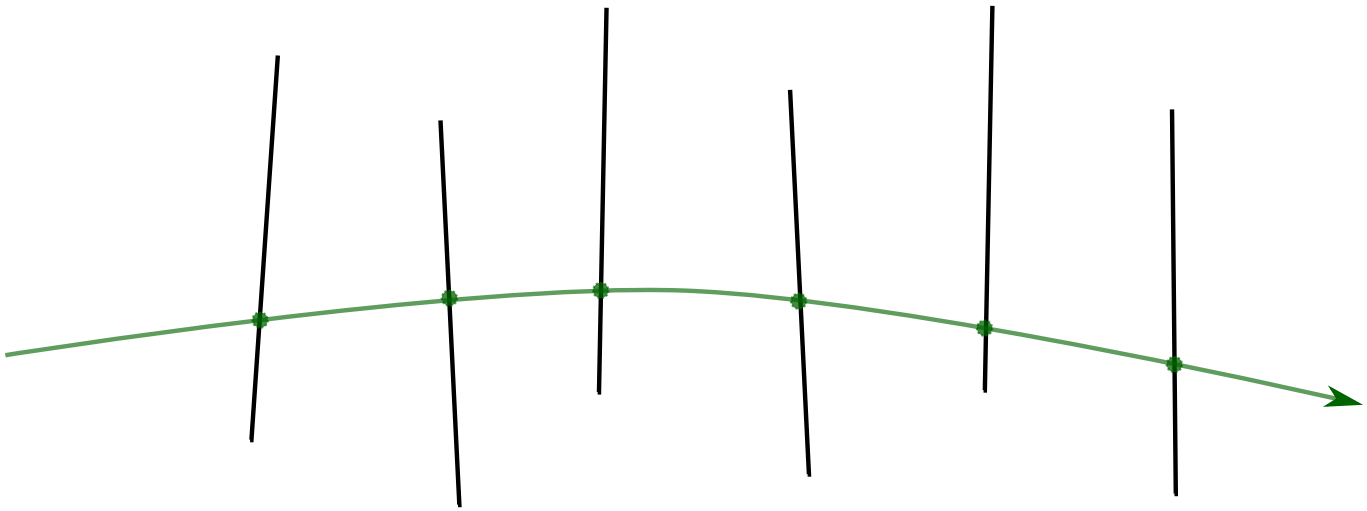
so you correct the modules' positions.

Track based alignment of a simple toy tracker



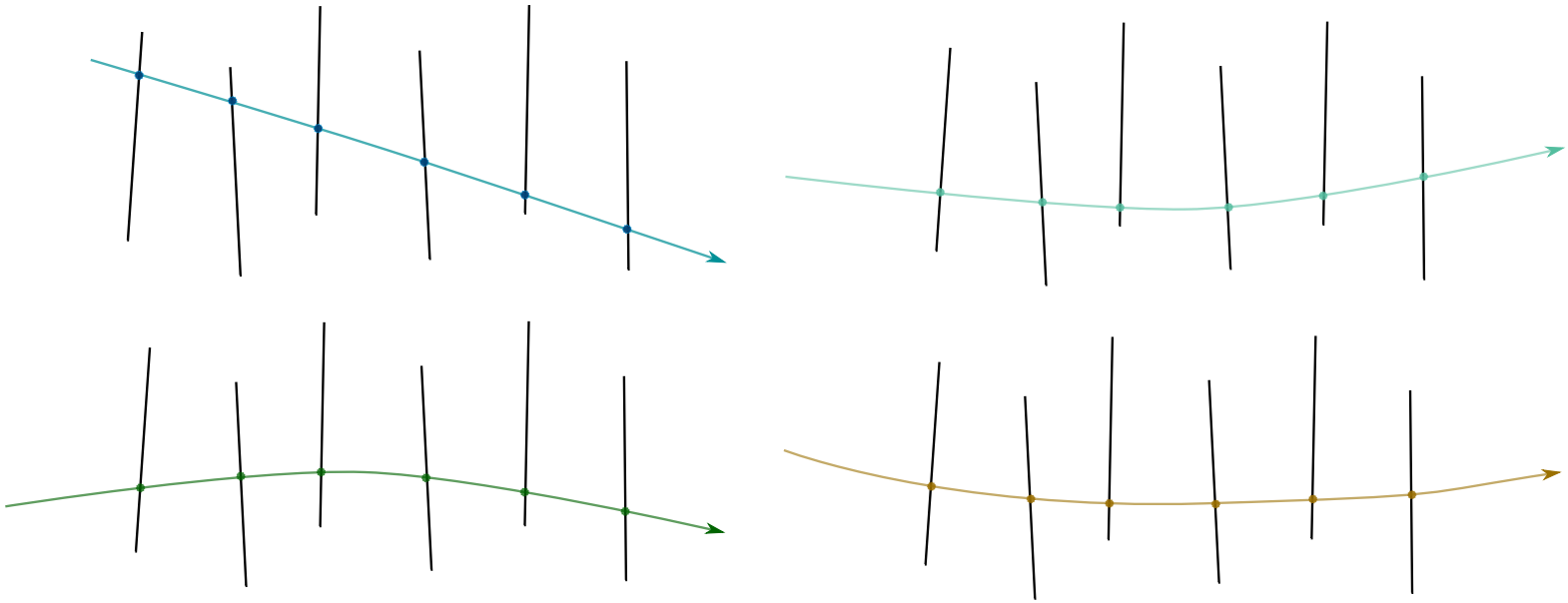
But compared to the reality, you are still off.

Track based alignment of a simple toy tracker



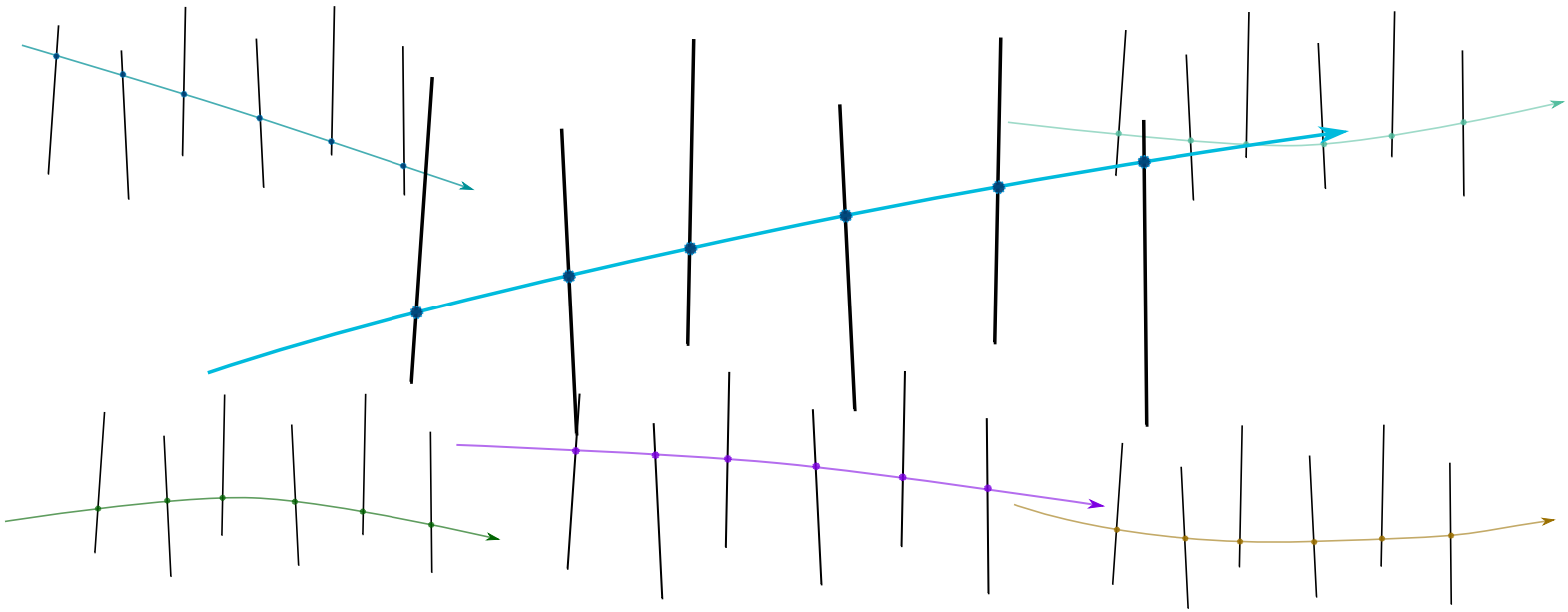
So you record more tracks

Track based alignment of a simple toy tracker



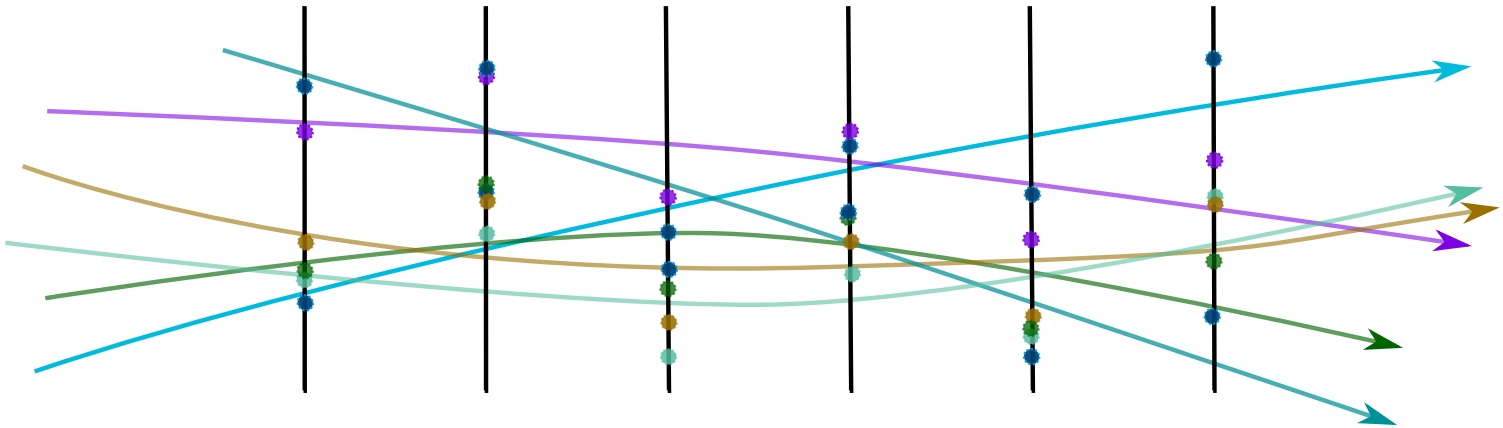
and more

Track based alignment of a simple toy tracker



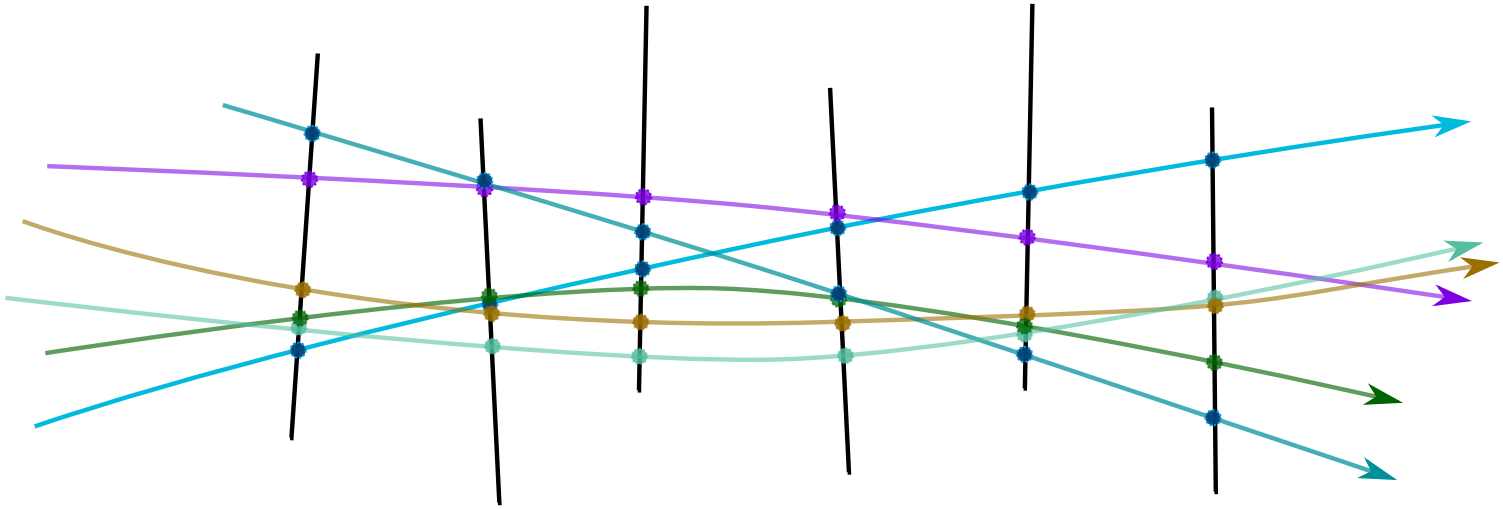
and even more

Track based alignment of a simple toy tracker



you take them all into account and deduce the true position

Track based alignment of a simple toy tracker



until you get it. This is what alignment has to do.

Algorithms

- In the context of this talk, alignment is a variant of a *linear least squares* problem

Algorithms

- In the context of this talk, alignment is a variant of a *linear least squares* problem
- The expression to be minimized is

$$\chi^2(\mathbf{p}, \mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \mathbf{r}_{ij}^{T}(\mathbf{p}, \mathbf{q}_j)\, \mathbf{V}_{ij}^{-1}\, \mathbf{r}_{ij}(\mathbf{p}, \mathbf{q}_j) \qquad (1)$$

  where
  - $\mathbf{r}_{ij}$: residuals (track-model prediction − measured hit)
  - $\mathbf{p}$: alignment parameters describing the actual geometry
  - $\mathbf{q}_j$: track parameters of the j^{th} track
  - $\mathbf{V}_{ij}^{-1}$: the inverse covariance matrix

    diagonal if measurements are uncorrelated: $\mathbf{V}_{ii}^{-1} = 1/\sigma_i^2$ with σ_i the Gaussian error of the measurement

Algorithms

- In the context of this talk, alignment is a variant of a *linear least squares* problem
- The expression to be minimized is

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \mathbf{r}_{ij}^T(\mathbf{p},\mathbf{q}_j)\,\mathbf{V}_{ij}^{-1}\,\mathbf{r}_{ij}(\mathbf{p},\mathbf{q}_j) \qquad (1)$$

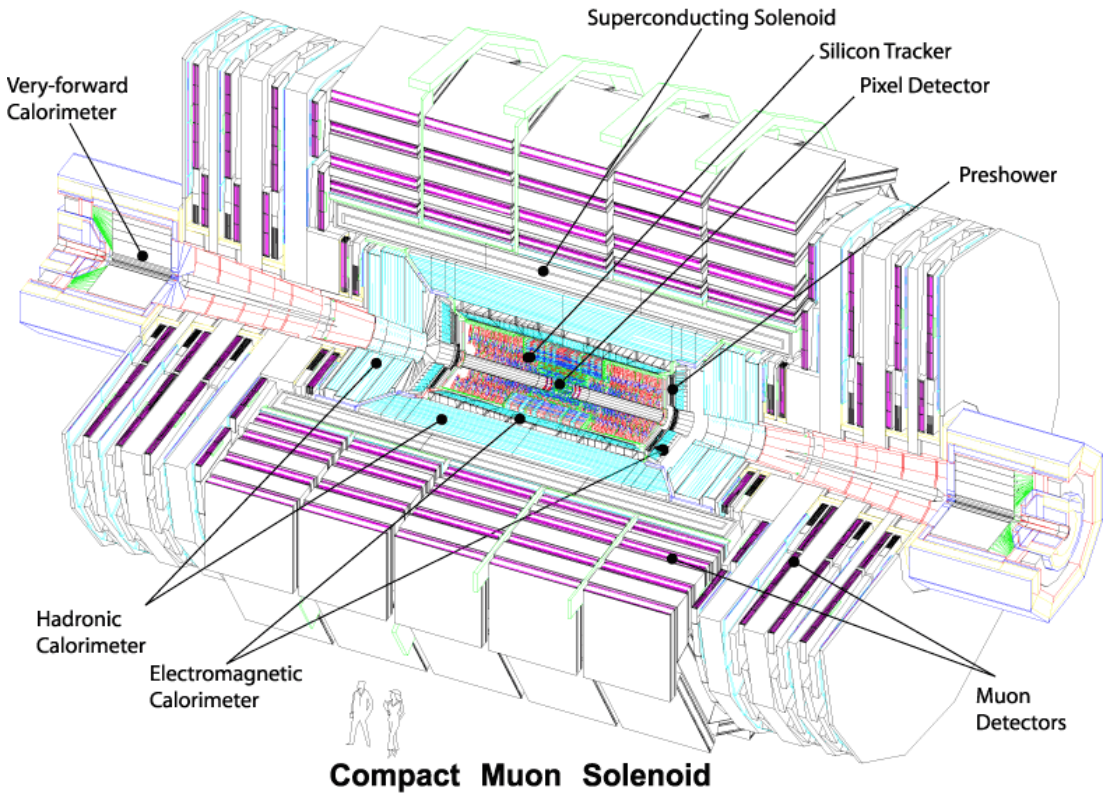
- Follows a χ^2 distribution for a given *number of degrees of freedom* (ndof), i.e.

$$\left\langle \frac{\chi^2(\mathbf{p},\mathbf{q})}{\text{ndof}} \right\rangle = 1$$

$$\left\langle \text{prob}(\chi^2, \text{ndof}) \right\rangle = 1/2$$

PAUL SCHERRER INSTITUT

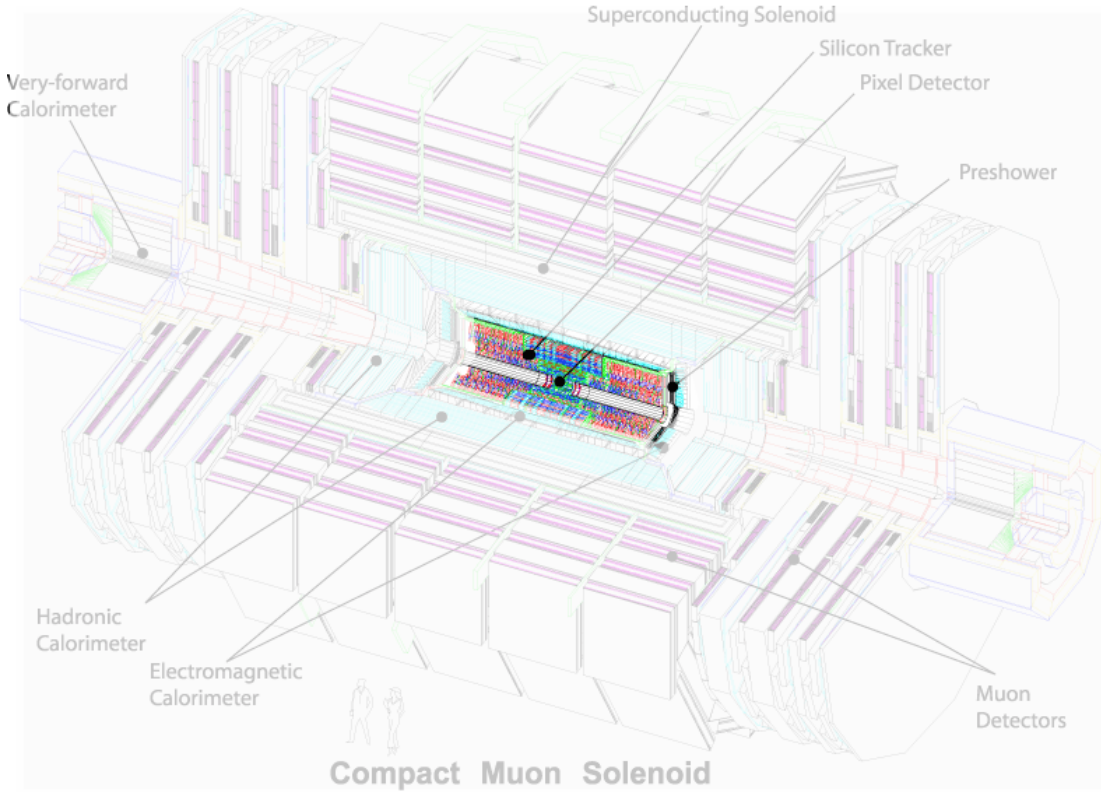
A brief detour: The CMS tracker



CMS is one of the two multi purpose detector at CERN's Large Hadron Collider (LHC)

PAUL SCHERRER INSTITUT

A brief detour: The CMS tracker



The silicon tracker is in the heart of CMS. It consists of

- 1440 silicon pixel modules
- 15 148 silicon strip modules

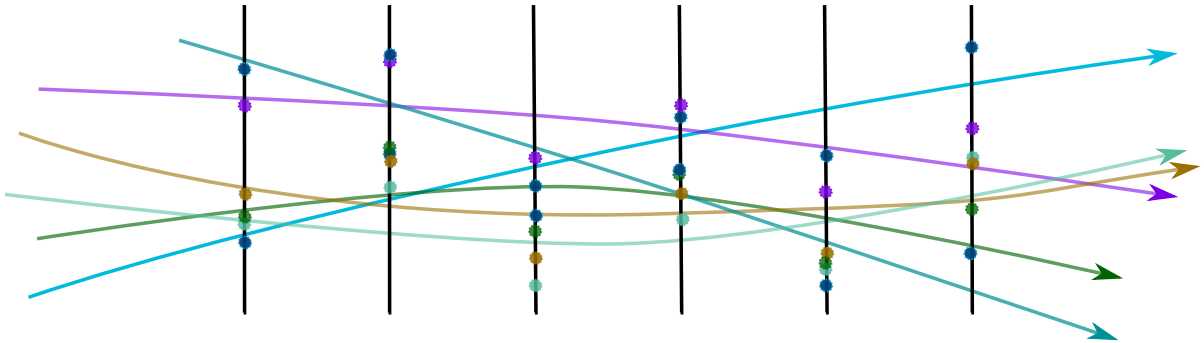
Part of the strip modules are made of two sensors and chained to one readout.

PAUL SCHERRER INSTITUT

Algorithms

In CMS we have two competing algorithms:

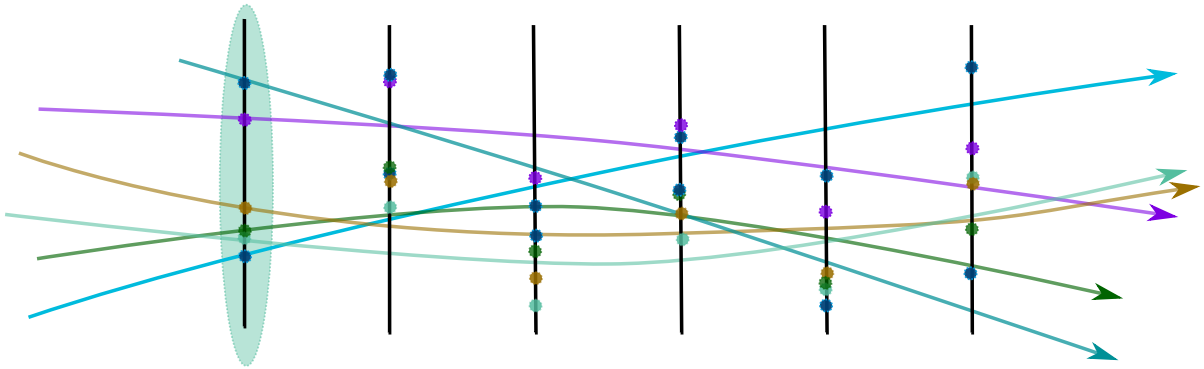
- one that works **localy** and iteratively (HIP):



Algorithms

In CMS we have two competing algorithms:

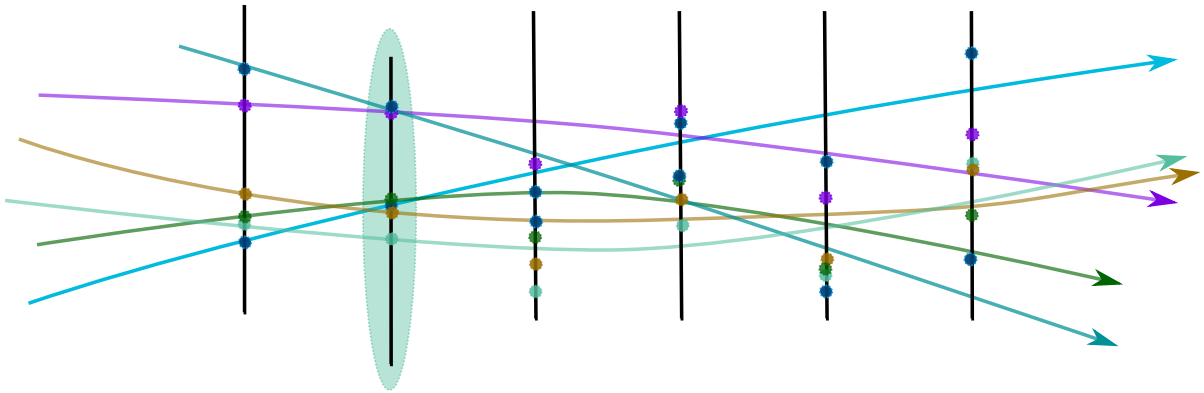
- one that works **localy** and iteratively (HIP):



Algorithms

In CMS we have two competing algorithms:

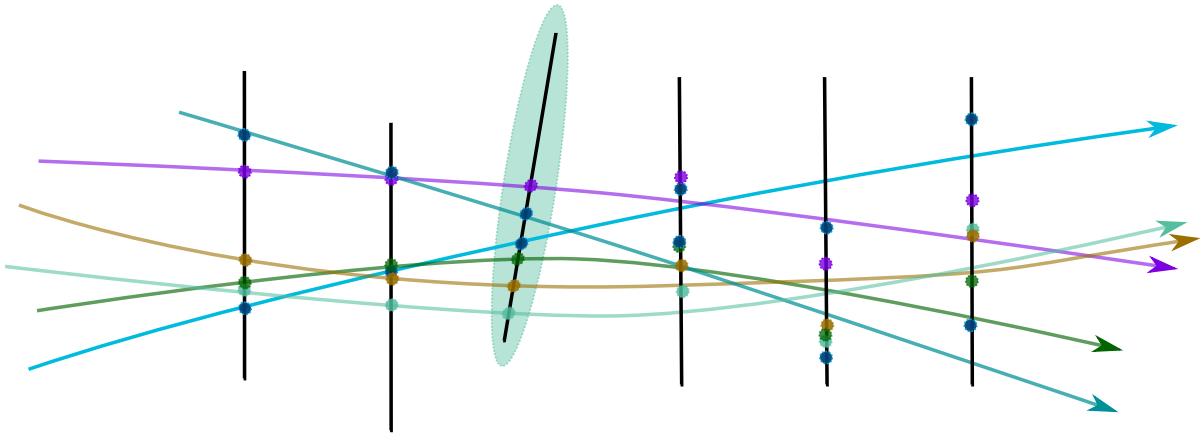
- one that works **localy** and iteratively (HIP):



Algorithms

In CMS we have two competing algorithms:

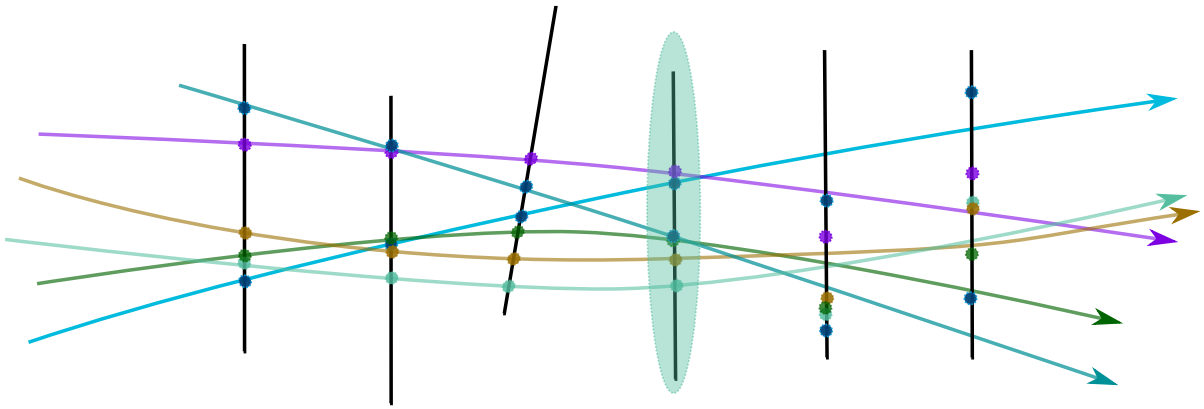
- one that works **localy** and iteratively (HIP):



Algorithms

In CMS we have two competing algorithms:

- one that works **localy** and iteratively (HIP):



i.e. the sum in the χ^2-equation (1) runs over all tracks passing through one module, optimizes the parameters **p** of this module.

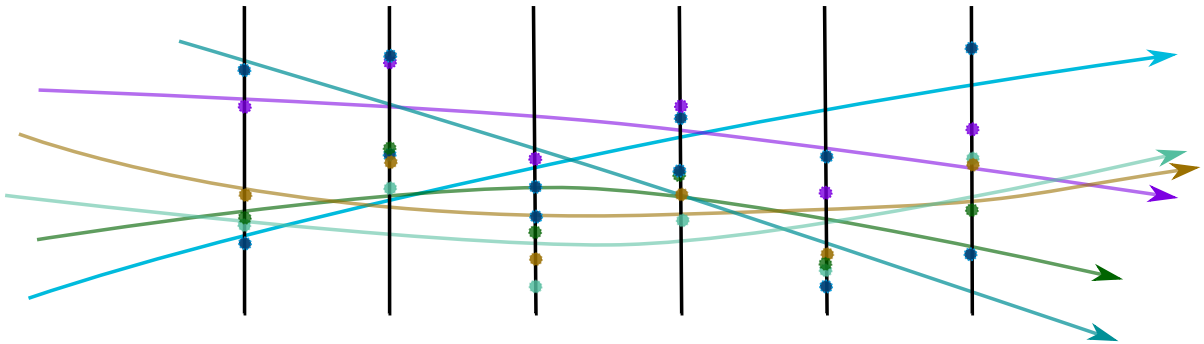
  - Benefit: small problems to solve in each step
  - Advantage: uses same track model as in tracking
  - Disadvantage: a lot of iterations needed
    (16 588 modules with 5 to 6 degrees of freedom, reiterate if needed)

PAUL SCHERRER INSTITUT

Algorithms

In CMS we have two competing algorithms:

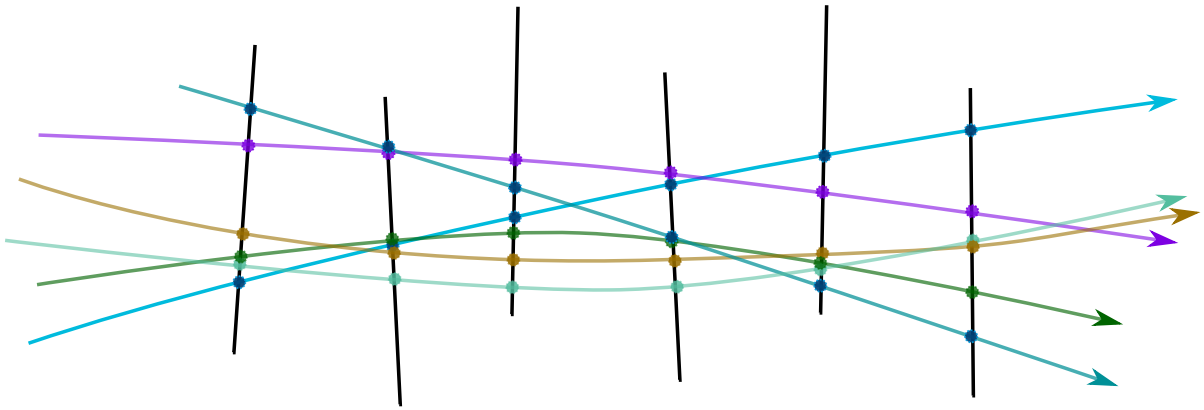
- one that works **localy** and iteratively (HIP):
- and one that works **globally** and does it in one step (Millepede-II)



Algorithms

In CMS we have two competing algorithms:

- one that works **localy** and iteratively (HIP):
- and one that works **globally** and does it in one step (Millepede-II)



i.e. the sum in the χ^2-equation (1) runs over all tracks and modules

- Benefit: everything in one go, all correlations considered
- Disadvantage: huge problem to solve
  (16 588 modules, 5 to 6 degrees of freedom
  $\Rightarrow$ 80 000 $\times$ 80 000-matrix to handle)

PAUL SCHERRER INSTITUT

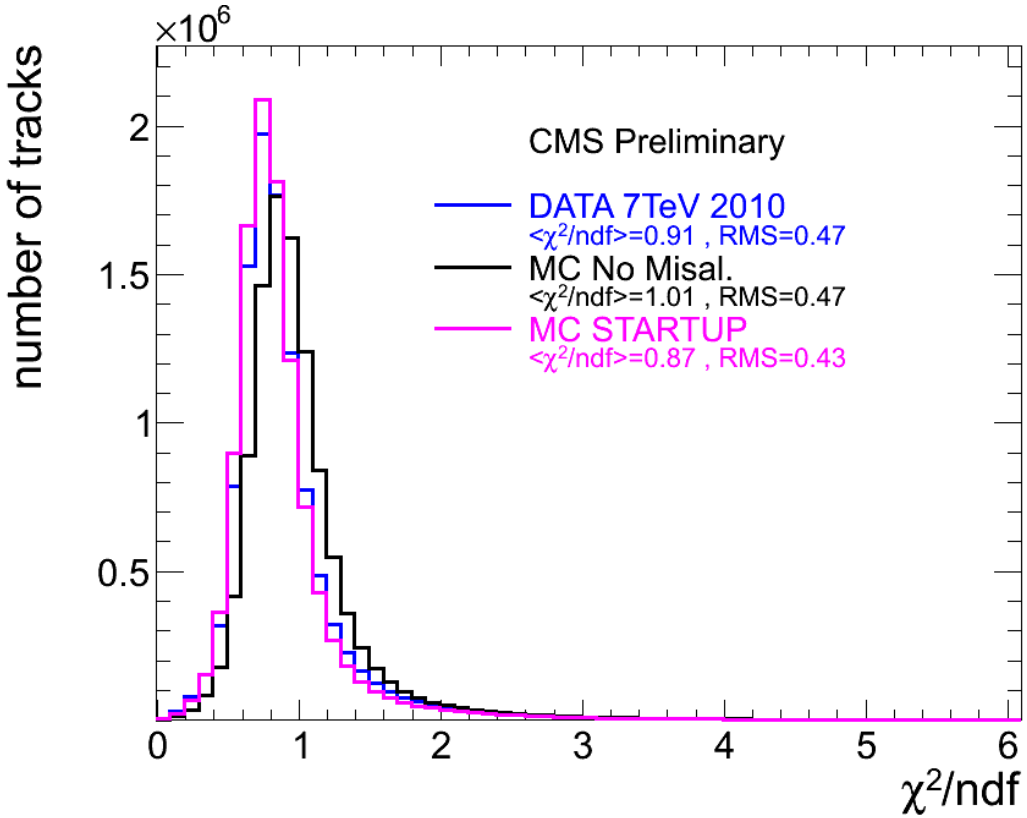
Alignment results

Where is the alignment of the CMS tracker today?

PAUL SCHERRER INSTITUT

Results – χ^2 of tracks

We have working alignment in CMS:



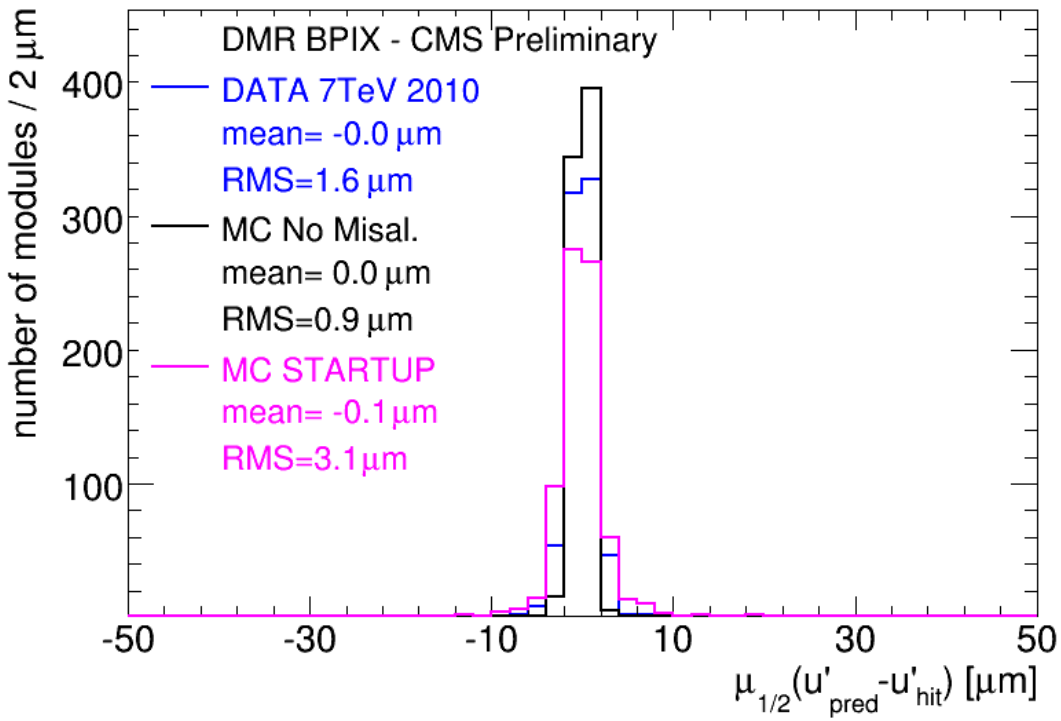
This one has been done using both algorithms run in sequence, a standard procedure since long.

Here χ^2 of tracks from track reconstruction (not used for alignment) are shown, compared to expectations from full detector simulations (MC: "Monte-Carlo").

PAUL SCHERRER INSTITUT

Results – Distribution of median of residuals

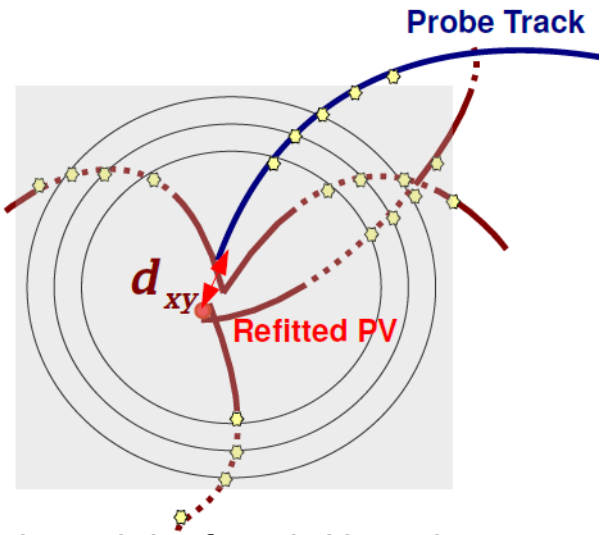
Or as an example the alignment of the pixel barrel detector:



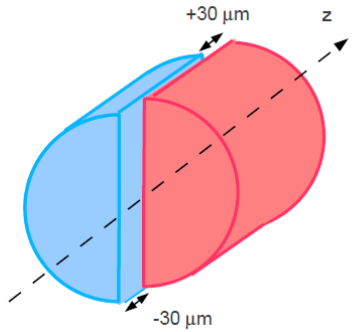
Shown here are the median values from residual distributions for each detector module. This suppresses random effects in the track-to-hit residuals.

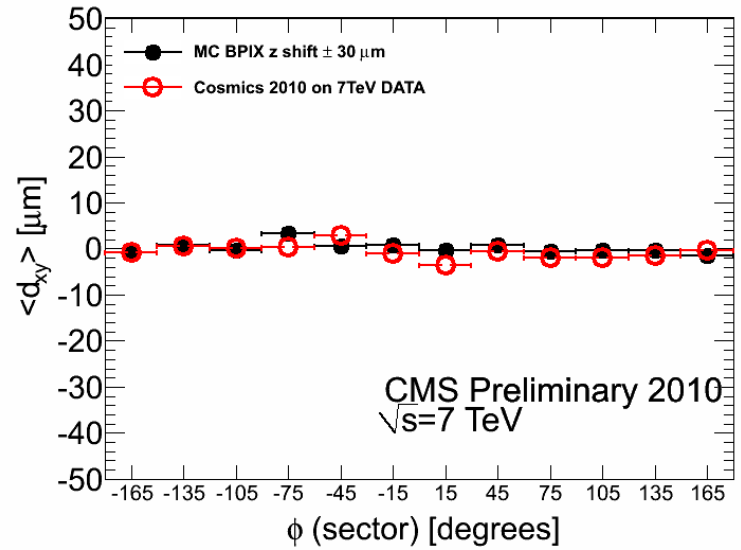
PAUL SCHERRER INSTITUT

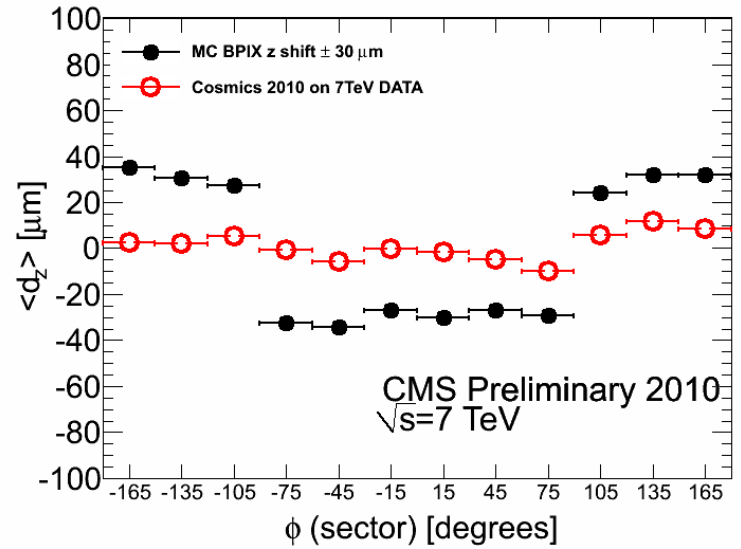
Results – vertex validation



d_{xy} and d_z of track *N* wrt the PV computed with the remaining *N-1* tracks as a function of ϕ-sector of the probe track



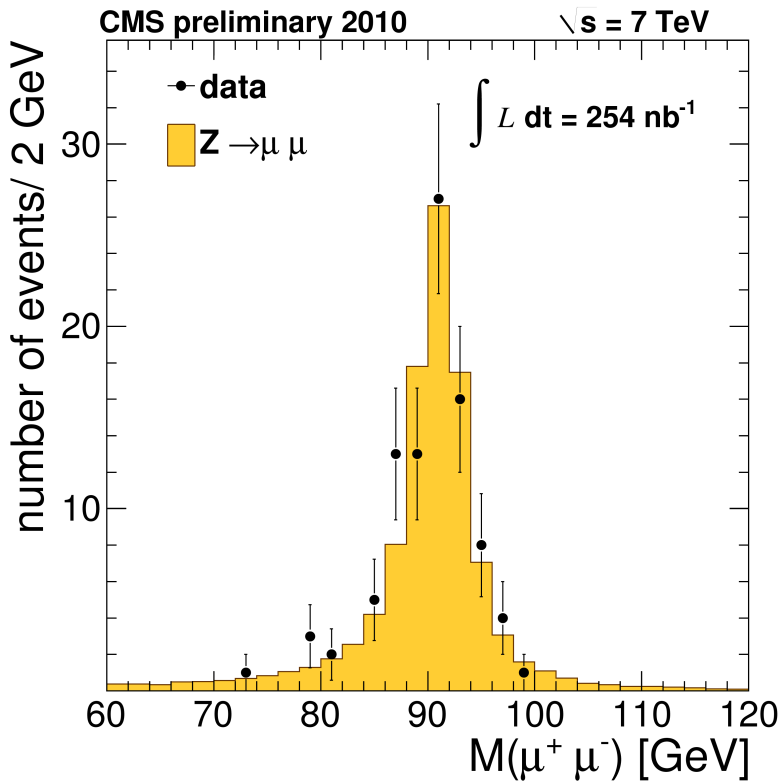




PAUL SCHERRER INSTITUT

Results – A physics example: $Z \rightarrow \mu\mu$

Reconstructed known resonances are also a measure of alignment quality



Shaded: simulation. Doing such things relies (among other factors) on a good alignment.

PAUL SCHERRER INSTITUT

Results – Alignment status of CMS inner tracker

From all these plots we know: The alignment is close to expectations.

What can we do now?

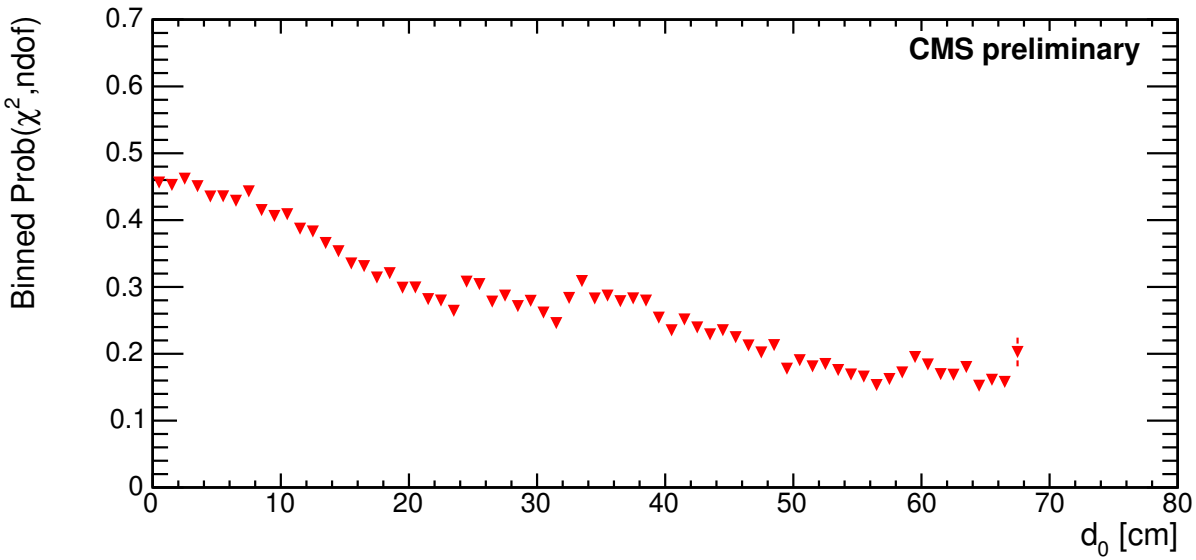
Going further: introducing more complex surface description

To what distortions of the tracker are we sensitive?

How can we improve?

Going further: the problem

The following plot lead the way to further developments:



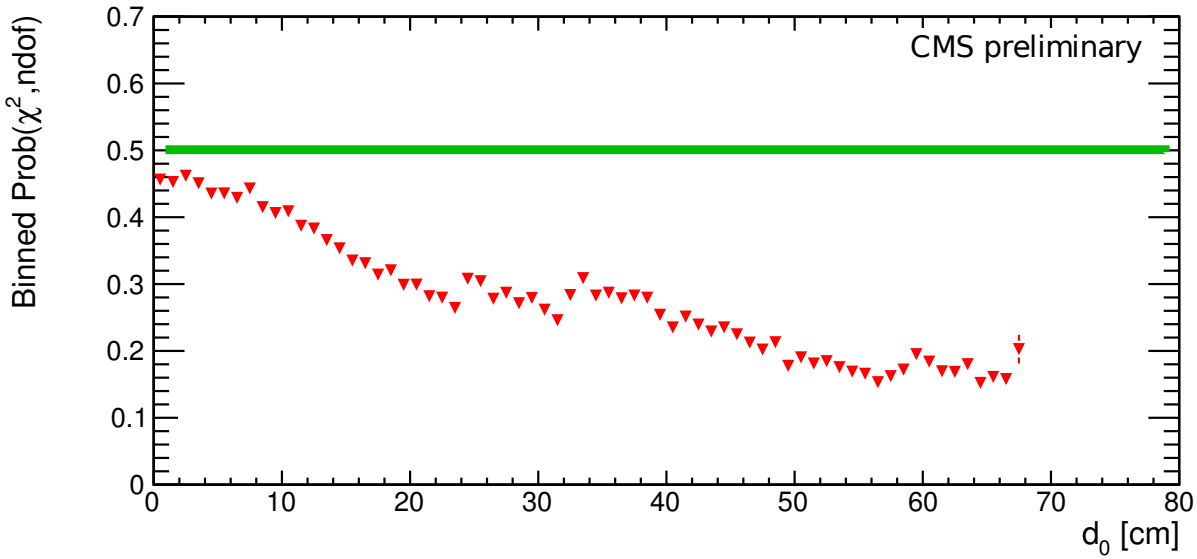
Distribution of the probability of the χ^2 vs. d_0:

Individual tracks from cosmic rays (CRAFT09) were fitted and binned. Shown are averages (markers) and RMS/$\sqrt{N}$ (error bars).

What does this mean?

PAUL SCHERRER INSTITUT

Going further: the problem

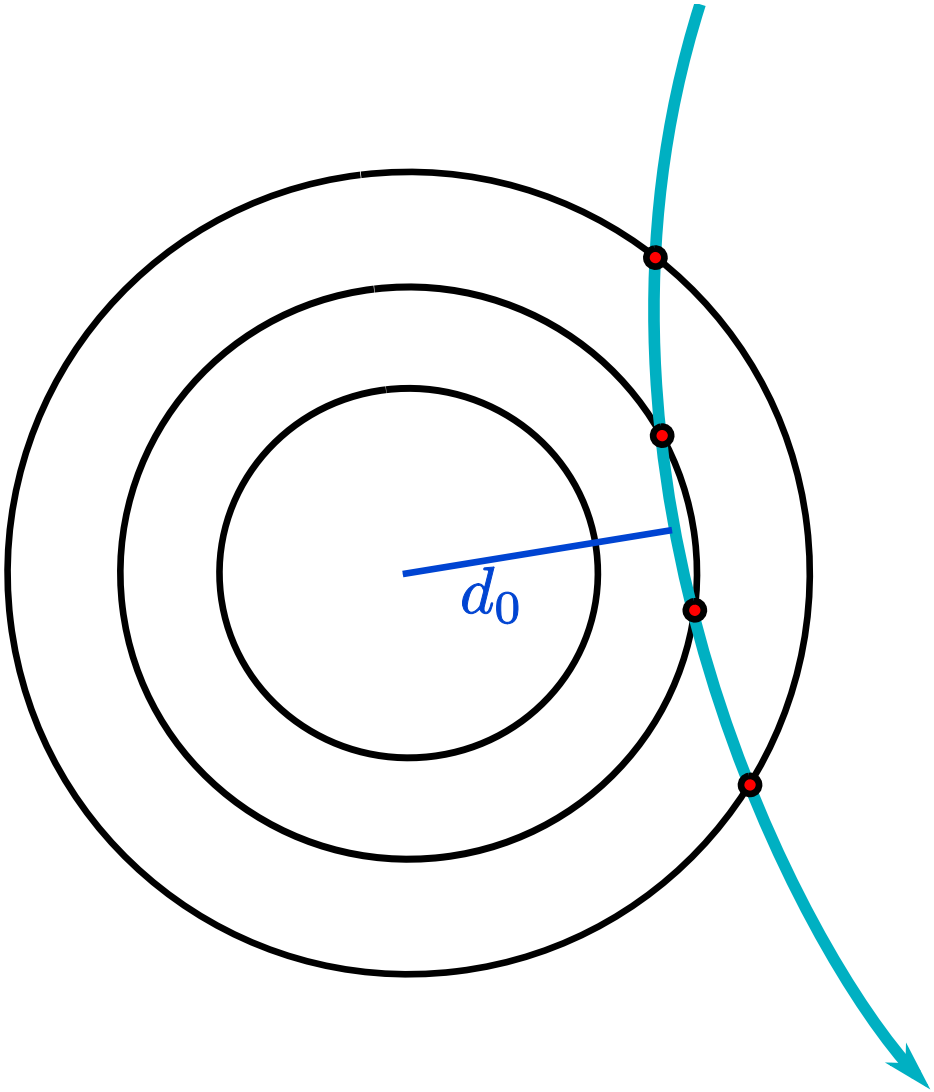


Whenever you plot $\langle \text{prob}(\chi^2, \text{ndof}) \rangle$ vs. some reasonable track parameter the expected result would be a flat distribution with

$$\langle \text{prob}(\chi^2, \text{ndof}) \rangle = 1/2$$

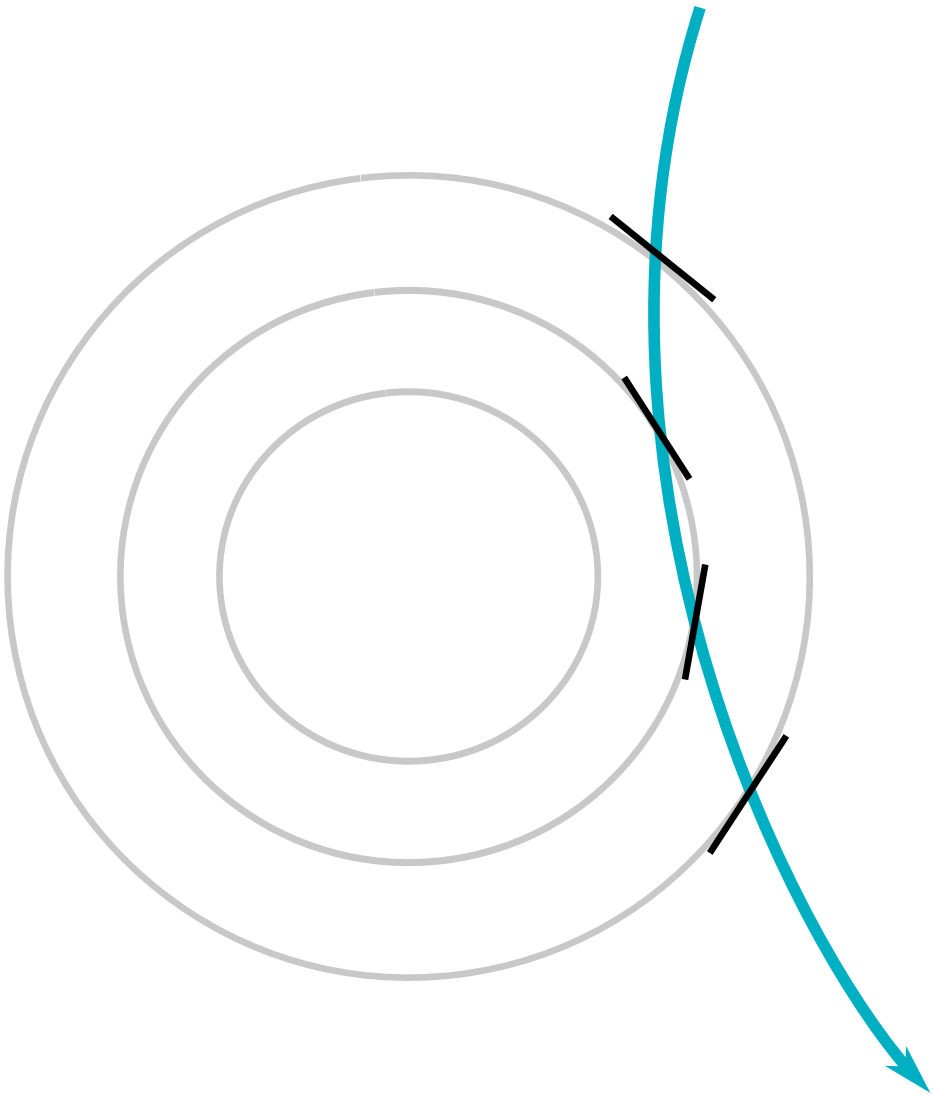
PAUL SCHERRER INSTITUT

Going further: the problem



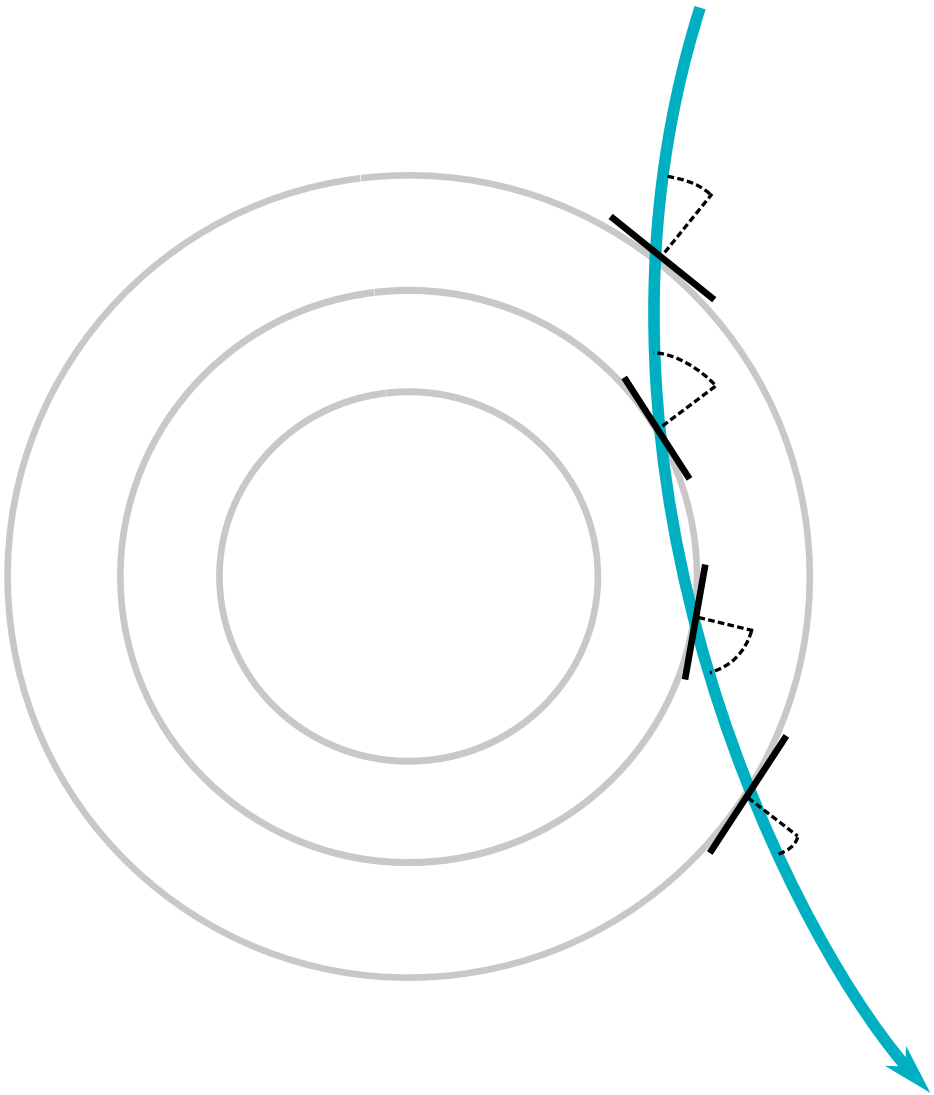
d_0 is one of the track parameters.
It describes the distance of closest approach to the origin of the coordinate system (our case: geometric center of the tracker)

Going further: the problem



Real modules differ from the circular shape, i.e. are flat

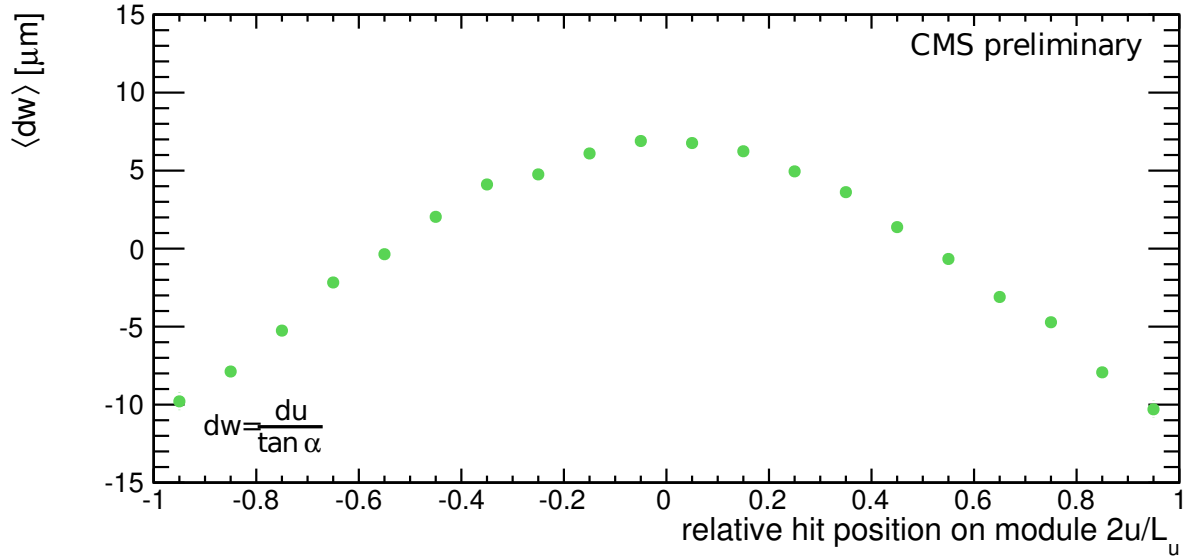
Going further: the problem



and so it is clear that d_0 maps to the incident angle w.r.t the module normal.

Going further: the solution to the problem

Analyzing the residuals versus the local module coordinates revealed the problem:



Shown here are the results for all modules of the two innermost layers in the strip tracker barrel (TIB).

- α: track angle to the normal
- *du*: residual of measured and predicted position (track fit)

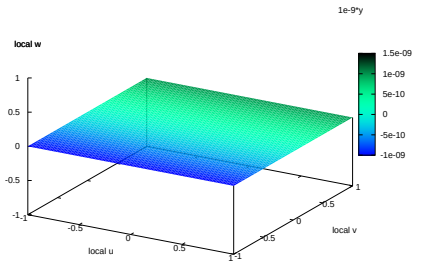
So we most likely have bowed sensor surfaces. No surprise for hardware people.

PAUL SCHERRER INSTITUT PSI

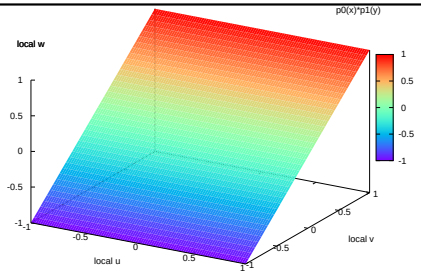
More complicated surface distortions

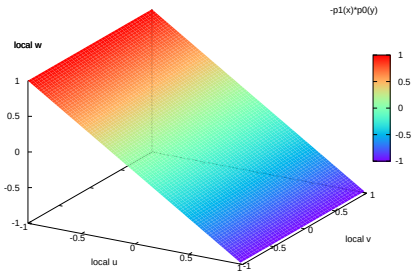
The movements in alignment are superpositions of

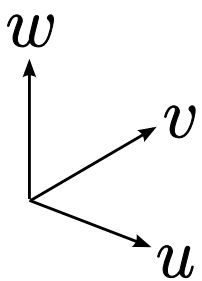
Shifts along the axes



Slopes (i.e. rotations)



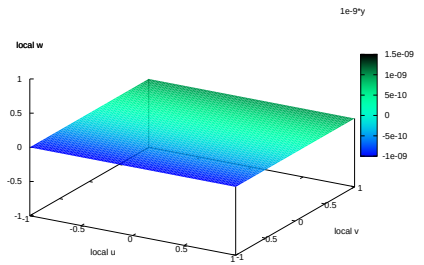


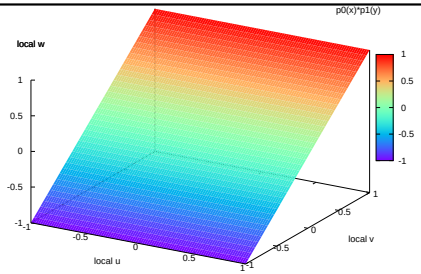


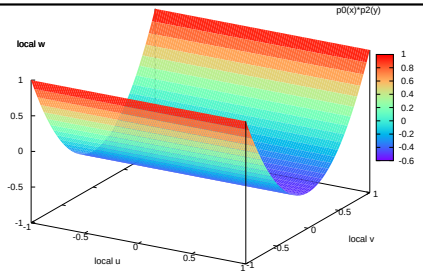
More complicated surface distortions

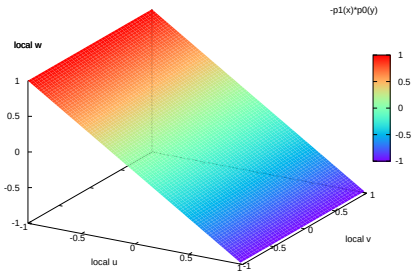
The movements in alignment are superpositions of

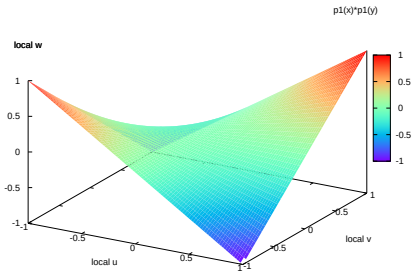
Shifts along the axes	Slopes (i.e. rotations)	bowed surfaces

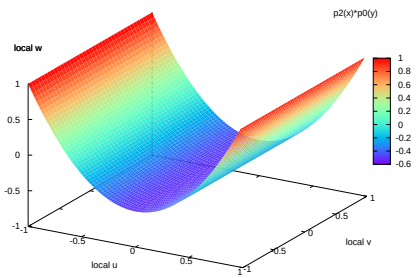


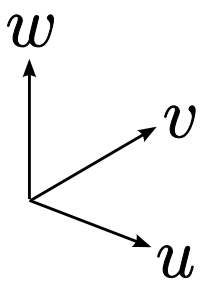








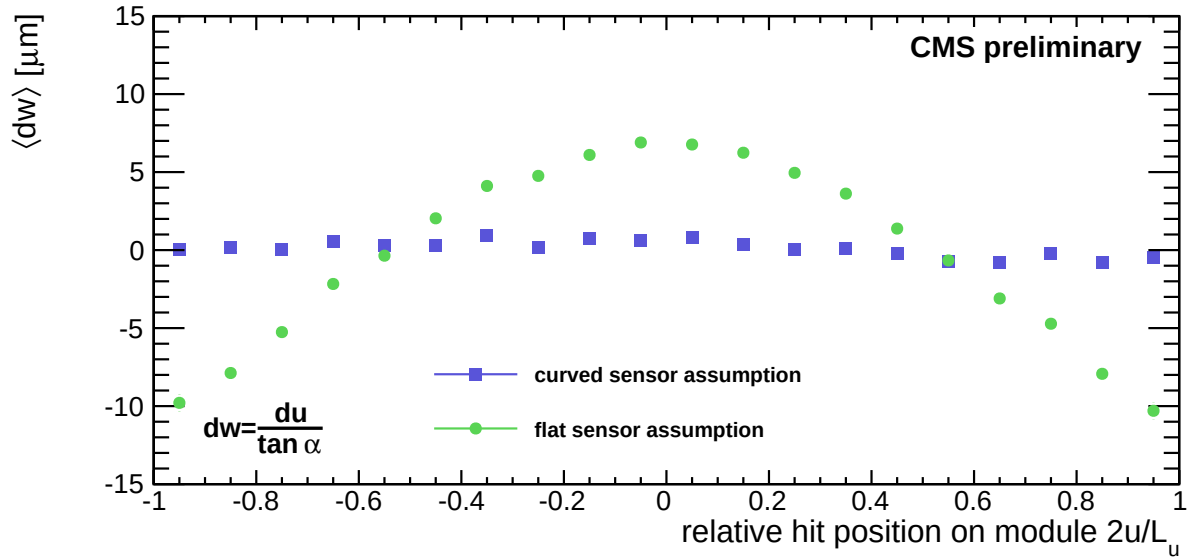




This adds 3 more parameters per module. Does this help?

More complicated surface distortions

Implementing this in alignment lead to the following result:

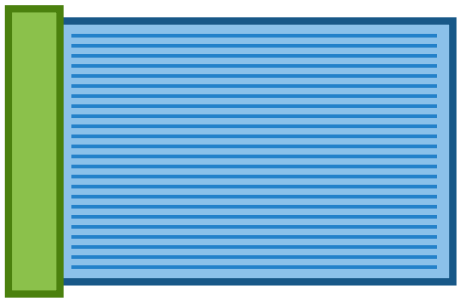


This shows the residuals when aligning for flat and curved sensor.
Observe: The bowed curve belongs to the flat sensor assumption.

- α: track angle to the normal
- *du*: residual of measured and predicted position (track fit)

More complicated surface distortions

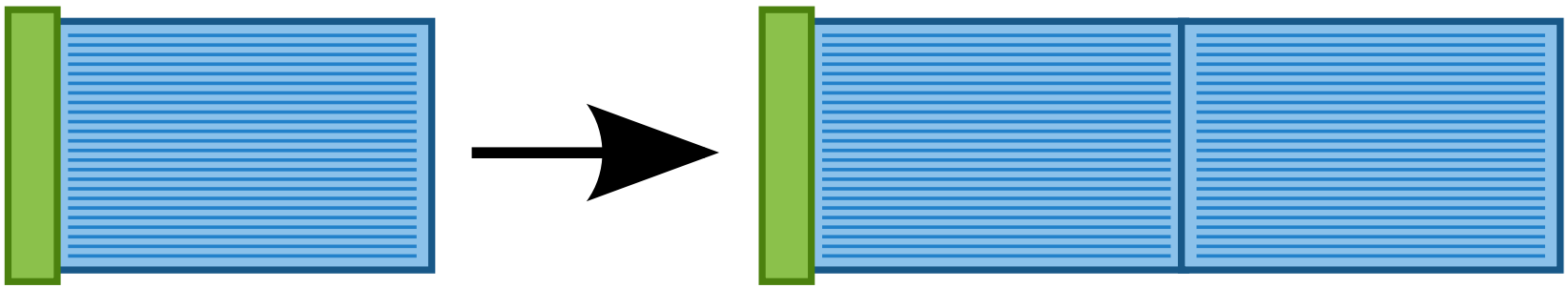
But we have another difficulty: The topologies of the strip modules differ in the regions of the tracker:



In the inner regions, the modules consist of one sensor

More complicated surface distortions

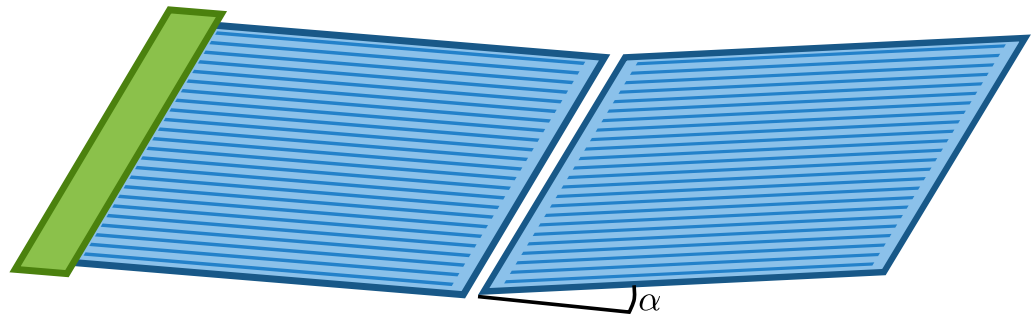
But we have another difficulty: The topologies of the strip modules differ in the regions of the tracker:



In the outer regions, the modules are made of two sensors, daisy-chained to the read-out electronics

More complicated surface distortions

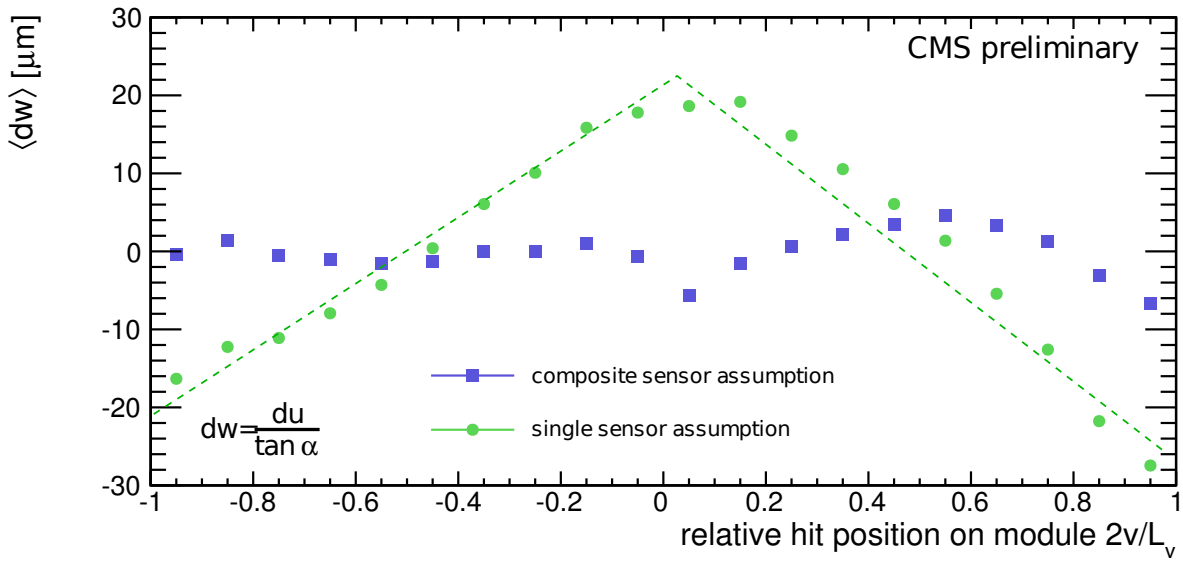
But we have another difficulty: The topologies of the strip modules differ in the regions of the tracker:



These composite modules may have a kink in between

More complicated surface distortions

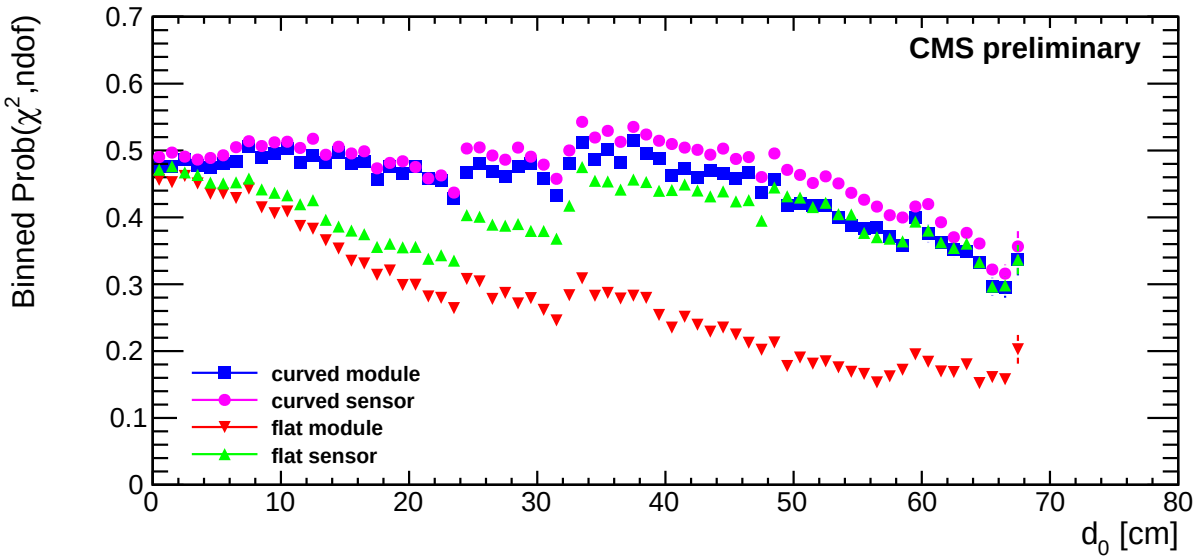
And for composite modules:



Again this confirms our findings. The bow in the right halve of the sensor is visible.

More complicated surface distortions

Distribution of the probability of the χ^2 vs. d_0:

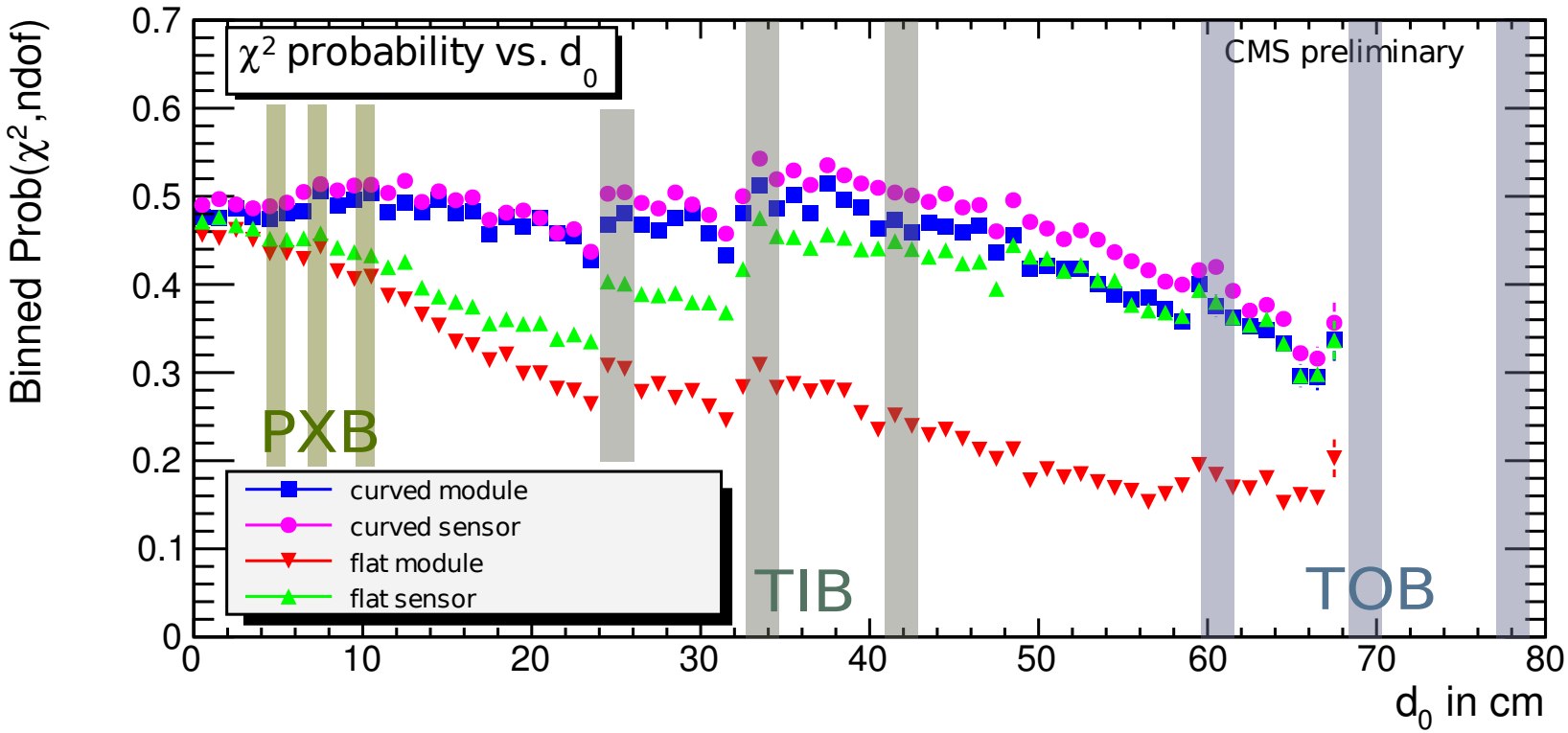


Aligning for bowed and split sensors corrects for the d_0 dependence up to about 50 cm.

This came with an increase of the number of parameters to align from 80 000 to 200 000. Alignment done in less than half a day (20 GB RAM, 7 cores in parallel).

PAUL SCHERRER INSTITUT

More complicated surface distortions

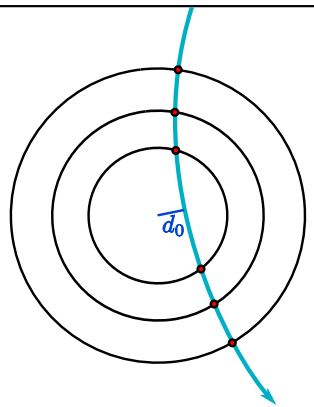


The jumps are an artifact of our detector topology.

More complicated surface distortions

To understand these jumps, we need to see what happens as d_0 grows:

small d_0	larger d_0	$d_0 > r_{\text{layer}}$	and larger

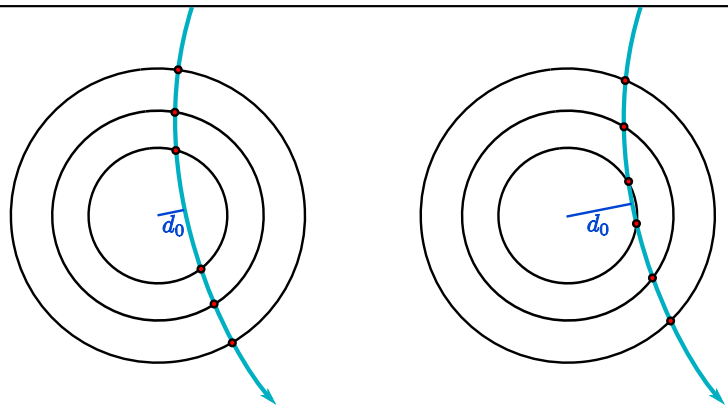


Such a track shows moderate track angles at every hit

More complicated surface distortions

To understand these jumps, we need to see what happens as d_0 grows:

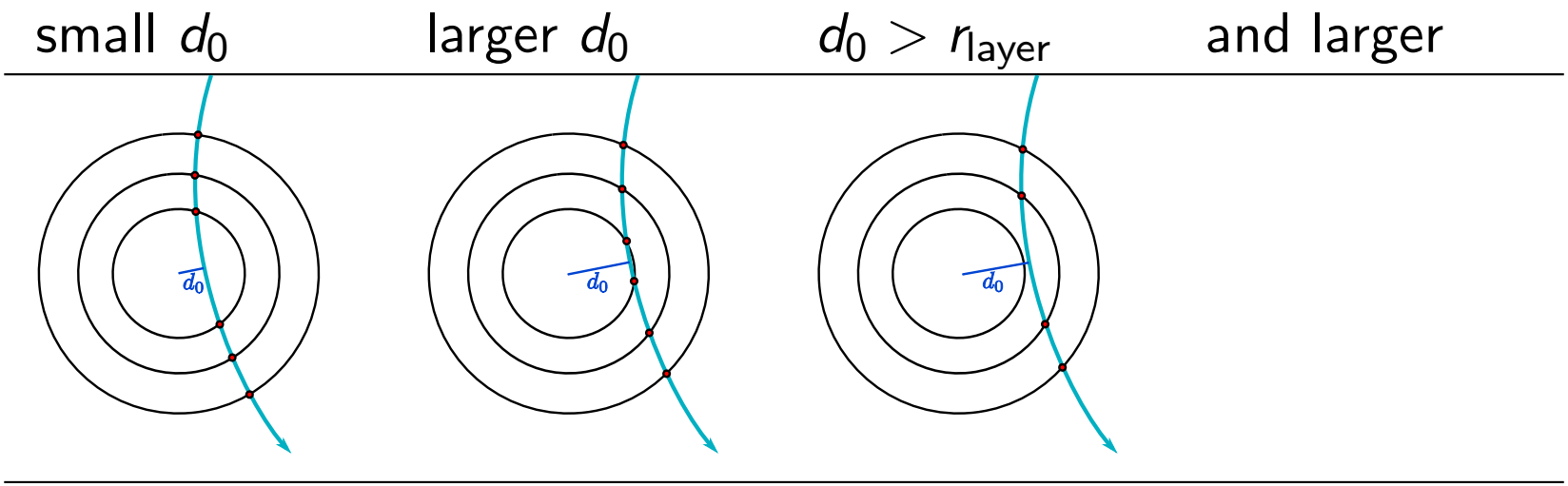
small d_0 | larger d_0 | $d_0 > r_{\text{layer}}$ | and larger



In this case, the track angles at the innermost hits are larger, the χ^2 gets deteriorated by a wrong surface description

More complicated surface distortions

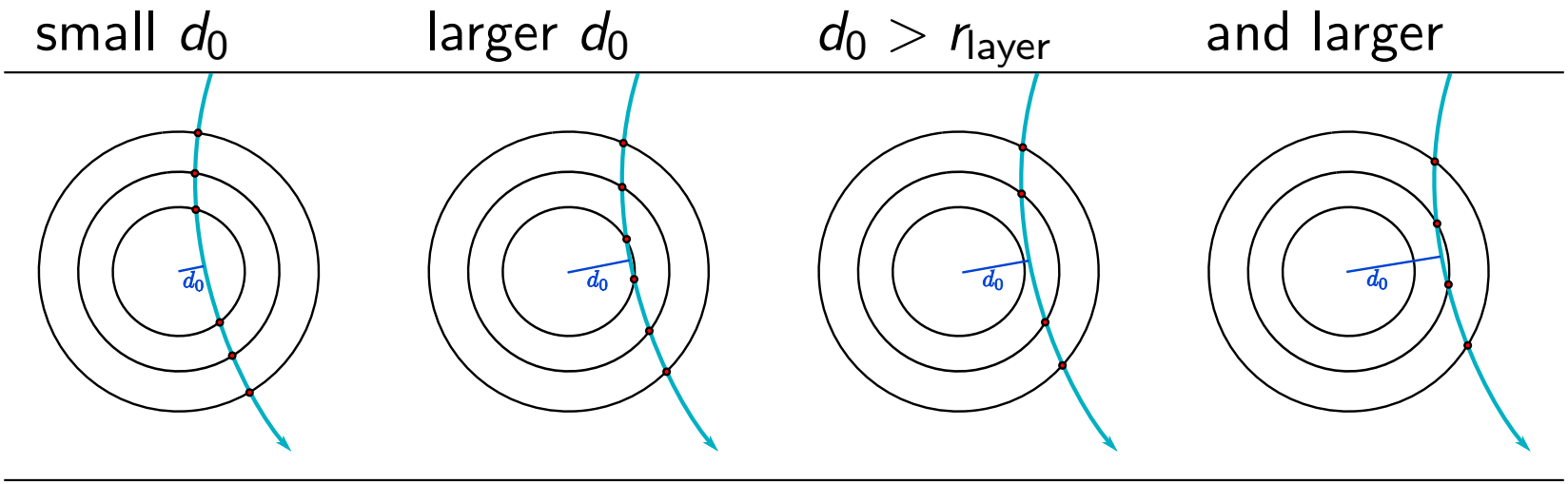
To understand these jumps, we need to see what happens as d_0 grows:



The innermost hits are lost, the χ^2 jumps to a better, though still lower than ideal, value

More complicated surface distortions

To understand these jumps, we need to see what happens as d_0 grows:



And the game starts over again

Lessons learned

What aspects of detector design helps aligners?

Lessons learned

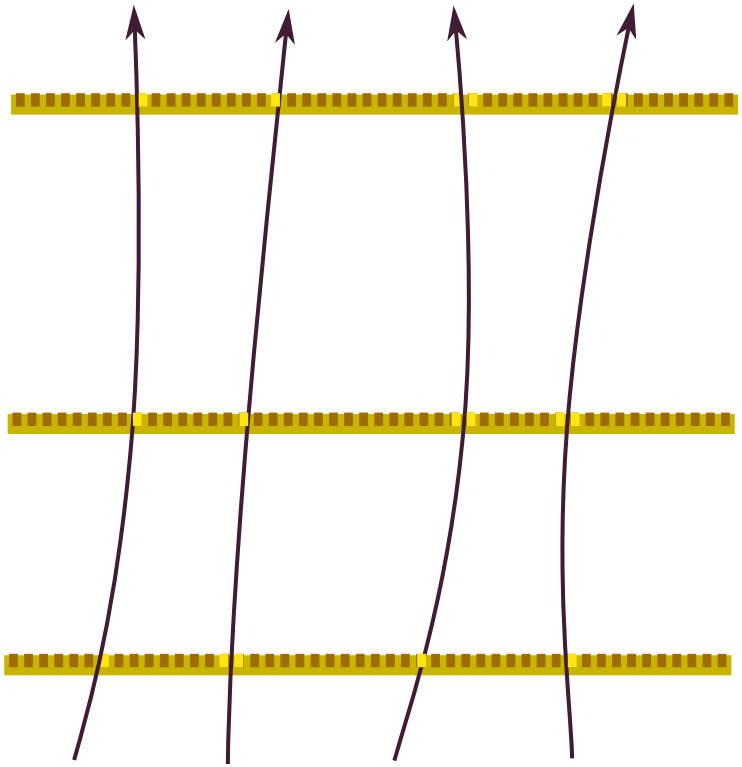
Here is my personal list of things to consider in future detectors

- Build detector modules with high resolution power
- Choose the module size as large as possible
- Mount your detector on a rigid structure (but don't waste money on precision mounting)
- Choose a detector design that is not too optimal
- Think of possibilities to measure tracks from non-standard origin
- Add visible survey marks
- Have overlap
- Omit unnecessary features

And now let me motivate some points of this list...

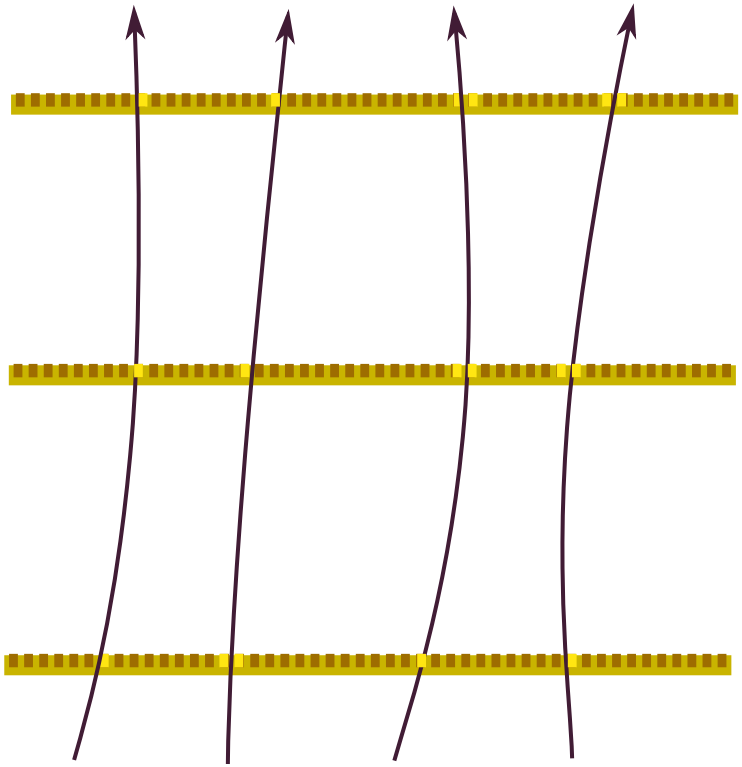
How to align the “third” dimension?

Aligning the distances between layers works on a subtle detail:



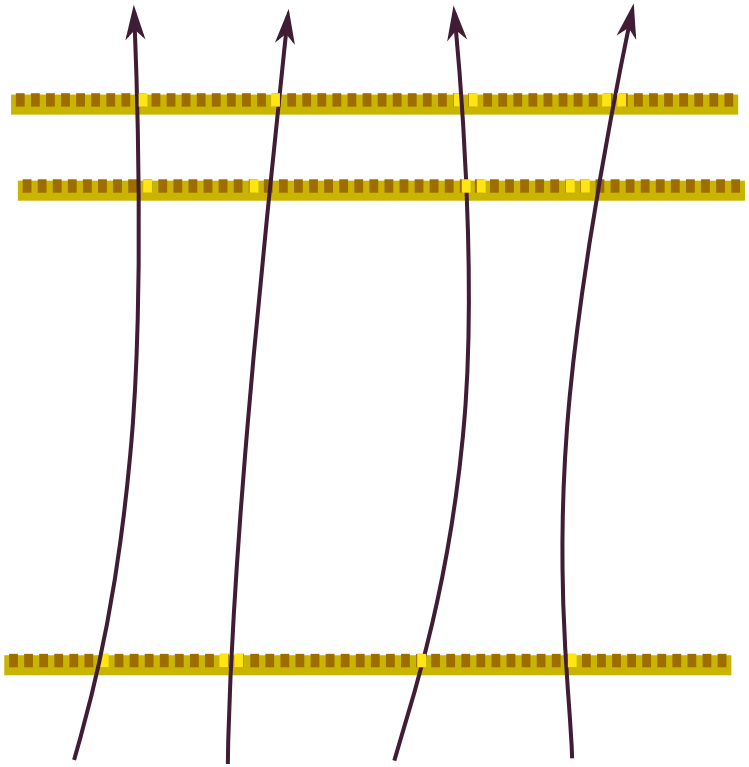
How to align the “third” dimension?

In case of tracks with similar direction



How to align the “third” dimension?

the distance cannot be determined precisely.

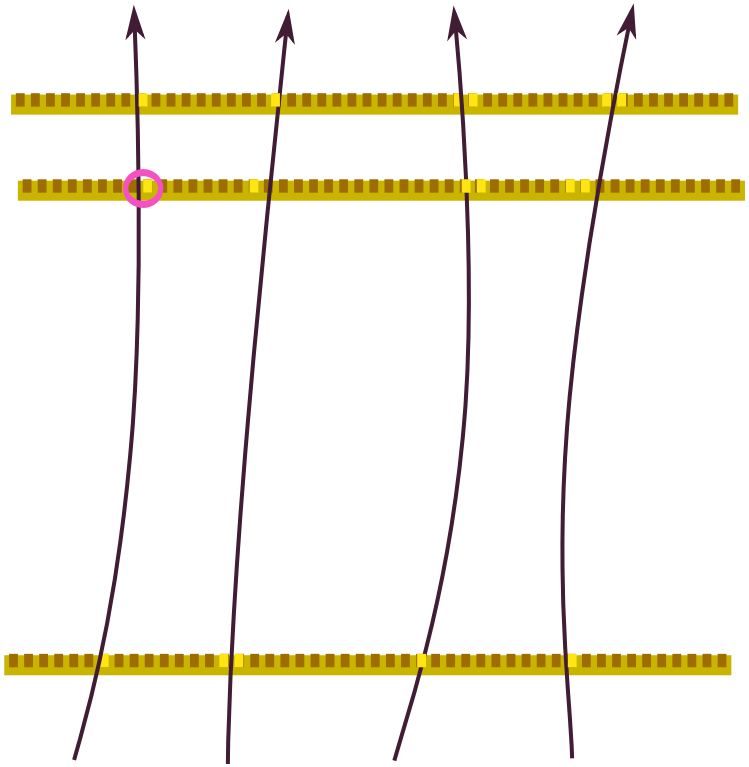


PAUL SCHERRER INSTITUT

PSI

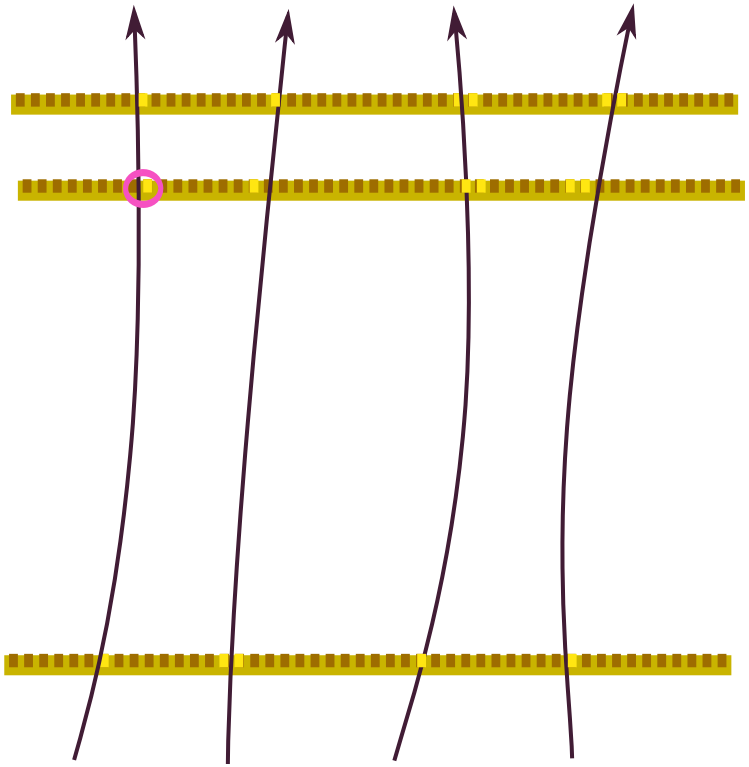
How to align the “third” dimension?

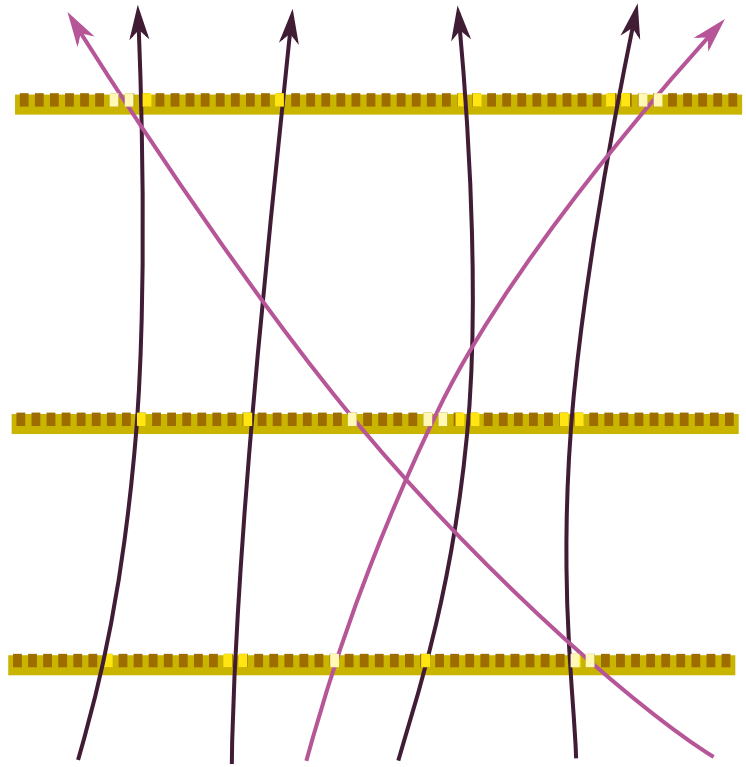
Only for large displacements we get a change in χ^2



How to align the “third” dimension?

Adding tracks from other sources



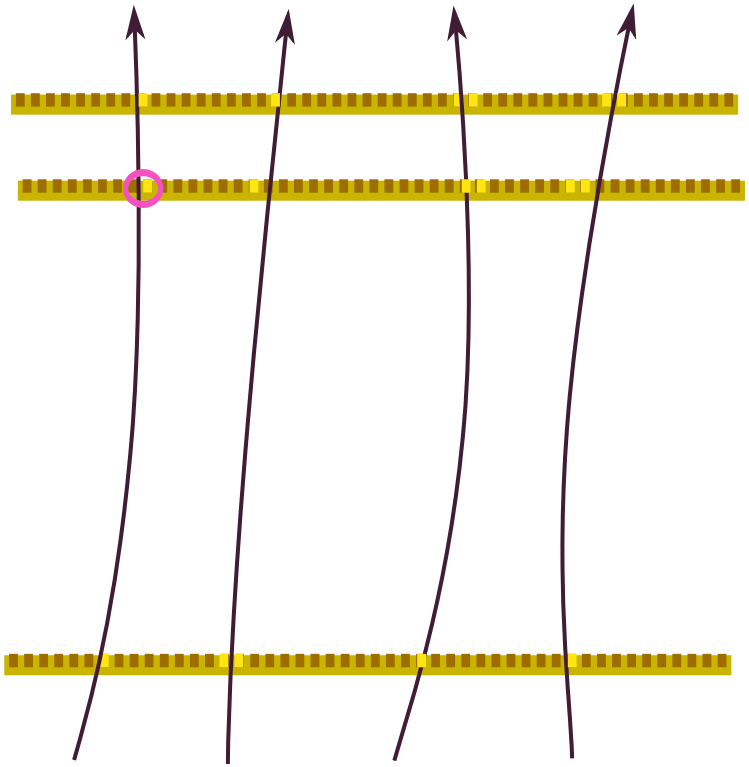


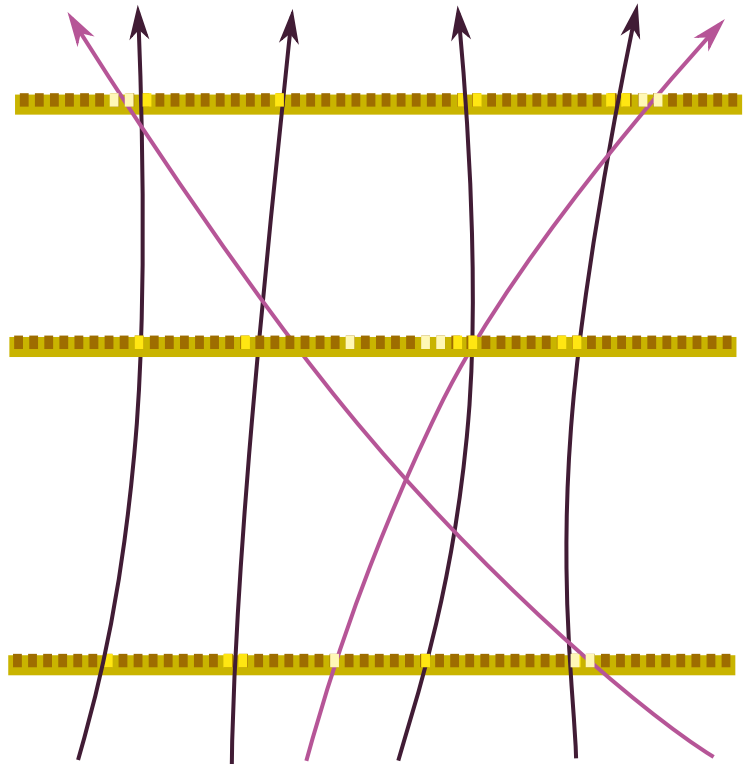
PAUL SCHERRER INSTITUT

PSI

How to align the “third” dimension?

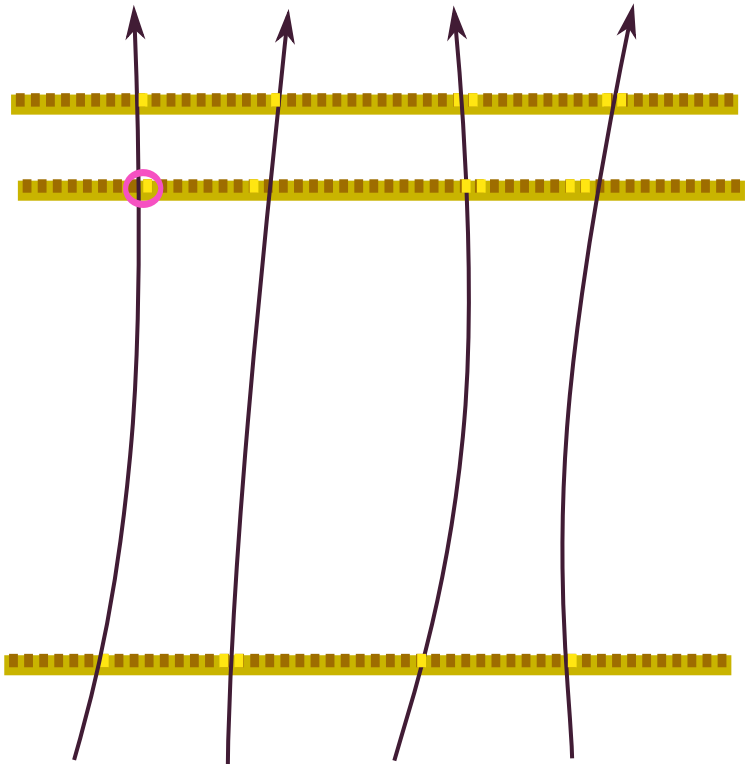
places much more restrictions

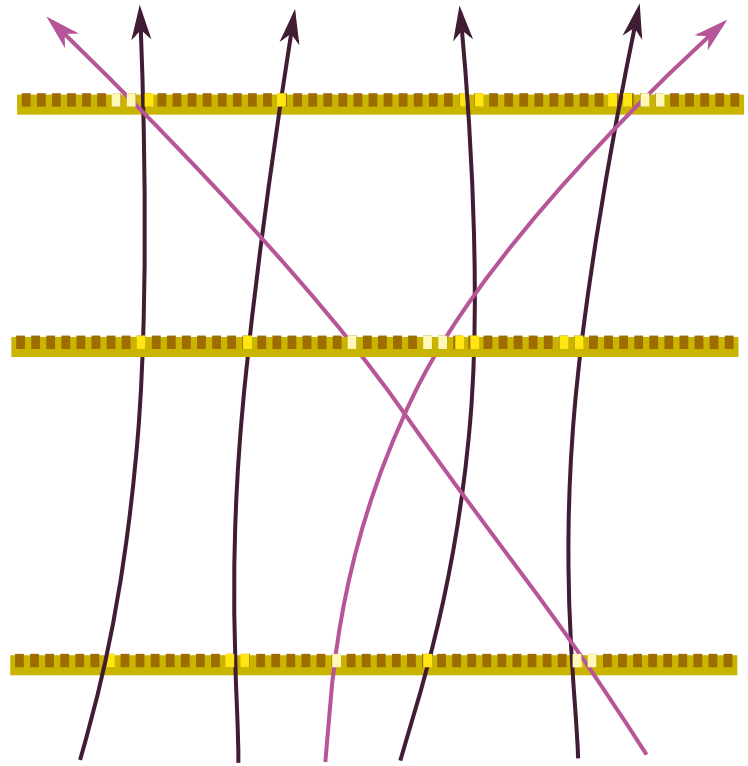




How to align the “third” dimension?

and alignment would need to distort the tracks beyond physical limits.

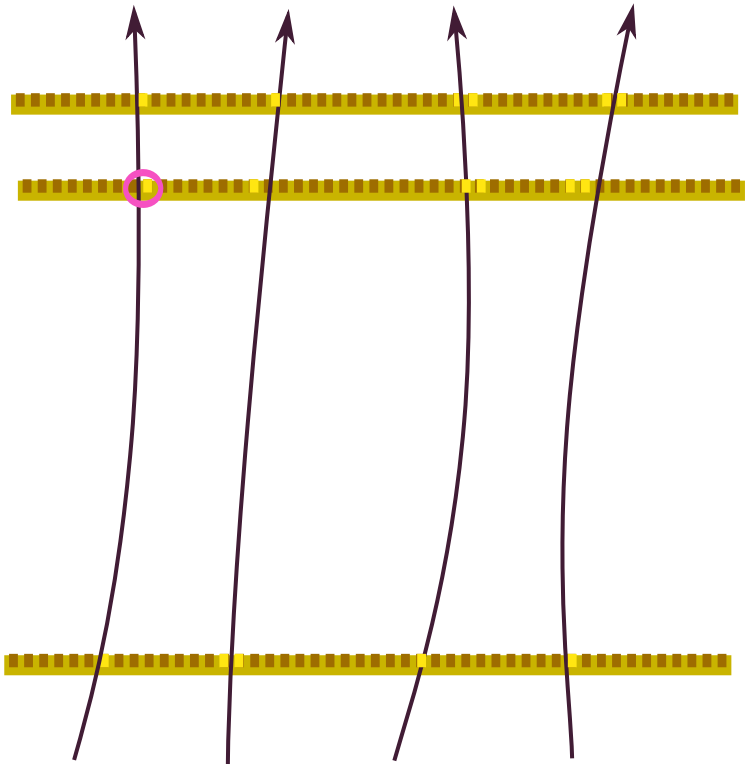


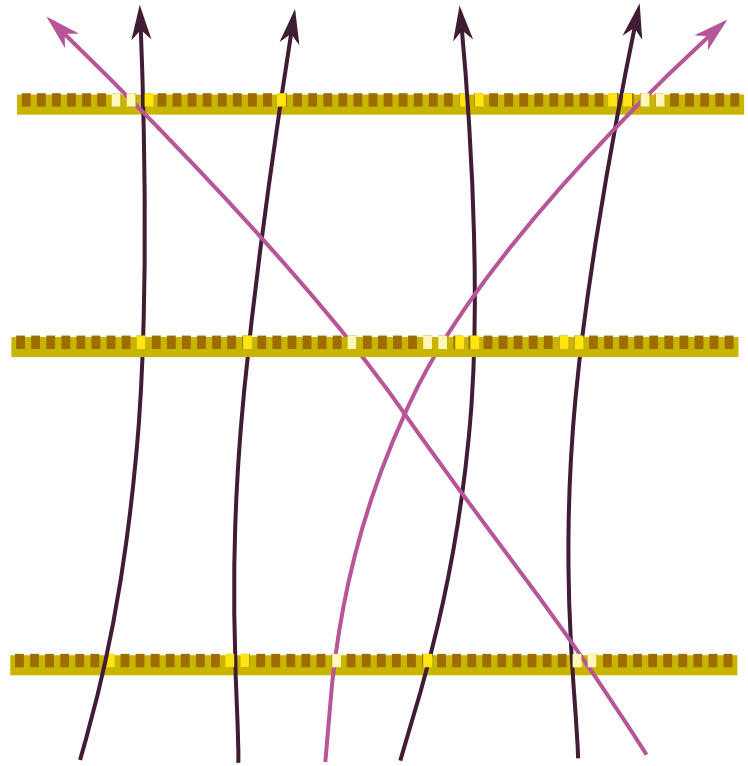


PAUL SCHERRER INSTITUT

How to align the “third” dimension?

and alignment would need to distort the tracks beyond physical limits.





The pitch keeps the distance of the hits on the sensor. And the pitch is probably the best constant over time.

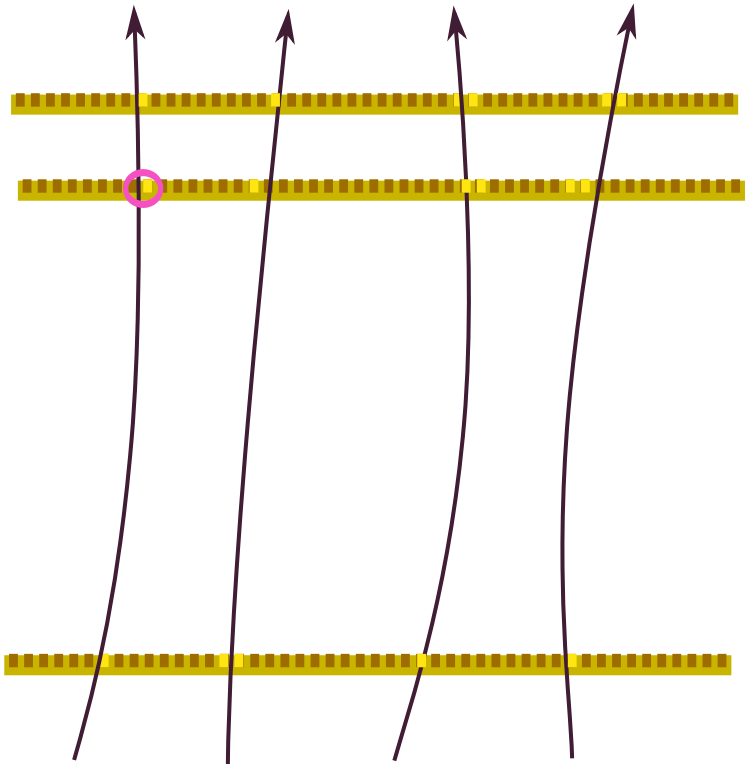
$\Rightarrow$ large modules and good resolution are important.

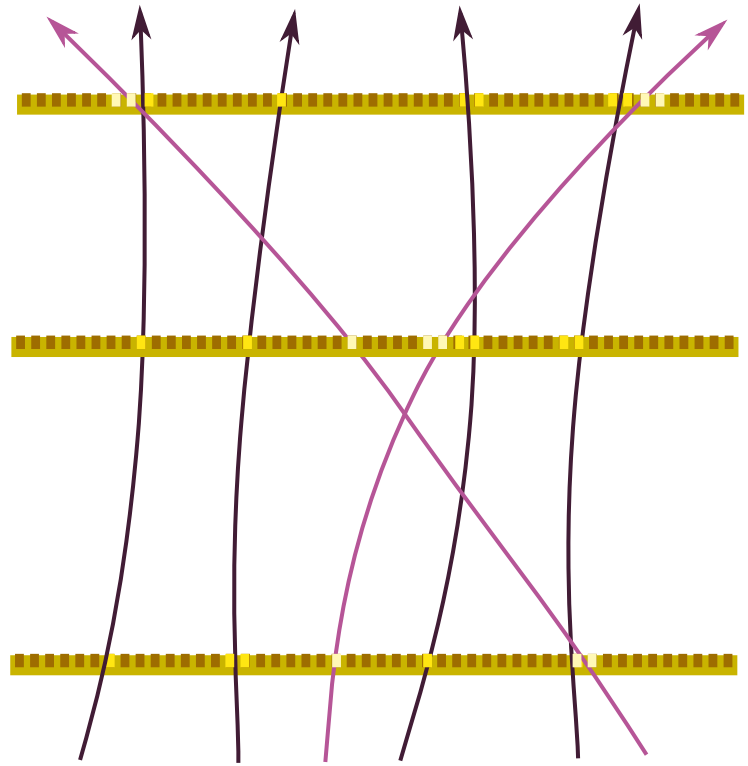
PAUL SCHERRER INSTITUT

PSI

How to align the “third” dimension?

and alignment would need to distort the tracks beyond physical limits.





Observe: Tracks from other sources give you very strong constraints.

Lessons learned – rigid mounting

Precision mounting is only necessary if

- you are sure that you constrain a mode
- you know that this stays stable throughout the whole lifetime of your detector

Our experience: With an alignment algorithm (properly chosen and setup) and a good selection of tracks we may align displacements of up to a few mm (sic!).

Lessons learned – omit unnecessary feature

Don't implement a hardware based alignment system unless you know that you

- really improve the alignment
- have no side effect (i.e. added material)

Conclusions

- Alignment of the CMS silicon tracker is well in shape. The performance is already close to design.
- We are sensitive to sensor bow and the substructures within modules
- We were able to align for sensor bow and kink

And my persoanl wish:
Construct future detectors with track-based alignment in mind.
This will help physics (and our budgets).

PAUL SCHERRER INSTITUT

Credits

We would like to give credits to the following people:

- Volker Blobel, Universität Hamburg (inventor of the MillePede-II algorithm): Very fruitful discussions.
- Claus Kleinwort, DESY: Investigated the problem by implementing the extensions needed for this analysis in CMSSW (higher order surfaces of sensors) and MillePede-II (error estimation). Performed several studies with Cosmics.
- Frank Meier, PSI: Spotted the initial problem while doing routine alignment. Performed studies with Cosmics and error estimation.
- Ernesto Migliore, Torino (and collaborators): Performed independent studies for confirmation based on tracking.
- Hans-Christian Kästli, PSI: Performed measurments of assembled pixel barrel modules on the microscope.
- Derek Feichtinger, T3 admin at PSI: Supplied computer nodes fulfilling our needs at a speed we never dreamed of.
- The Statistics Tools Group of the Analysis Centre of the Helmholtz Terascale Alliance for the implementation of compression in storage of the data and the parallelization of relevant parts of the algorithm.
- The whole tracker alignment group.
- All current and former members of CMS who worked on the implementation of MillePede-alignment in CMS.

PAUL SCHERRER INSTITUT

Backup slides

How the bowed surfaces are implemented

The bows were implemented using two-dimensional Legendre polynomials:

$$w(u,v) = \sum_{i=0}^{N} \sum_{j=0}^{i} c_{ij} L_j(u) L_{i-j}(v)$$

where

- $w(u,v)$ is the deviation from a plane at the origin in w direction as function of u, v
- N the maximal order of the Legendre polynomials. For $N \to \infty$ every possible surface may be described.
- c_{ij} coefficients
- $L_i(x)$ Legendre polynomial of i-th order

PAUL SCHERRER INSTITUT

How the bowed surfaces are implemented

The bows were implemented using two-dimensional Legendre polynomials:

$$w(u,v) = \sum_{i=0}^{N} \sum_{j=0}^{i} c_{ij} L_j(u) L_{i-j}(v)$$

Pro memoria:

- Legendre polynomials are orthogonal on $x \in [-1, 1]$
- The first three Legendre polynomials are
  - $L_0(x) = 1 \quad L_1(x) = x \quad L_2(x) = \frac{1}{2}(3x^2 - 1)$
- $N = 1$ is the same situation as in the past, just slopes instead of angles
- $N = 2$ allows for bends, the sagittae will be
  - $S_u = \frac{3}{2} c_{22} \quad S_{uv} = c_{21} \quad S_v = \frac{3}{2} c_{20}$

PAUL SCHERRER INSTITUT

Backup: Error estimate on parameters

How precise can we determine these bows? In principle, Millepede-II solves for $\mathbf{x}$ like in[1]

$$\mathbf{M}\mathbf{x} = \mathbf{y}$$

Inversion of $\mathbf{M}$ would give access to the covariance matrix, but inversion is of $O(n^3)$ and $n \approx 200\,000$.

[1] see backup slides for more

Backup: Error estimate on parameters

How precise can we determine these bows? In principle, Millepede-II solves for $\mathbf{x}$ like in[1]

$$\mathbf{M}\mathbf{x} = \mathbf{y}$$

Inversion of $\mathbf{M}$ would give access to the covariance matrix, but inversion is of $O(n^3)$ and $n \approx 200\,000$.
Observe that when solving for $\mathbf{x}$ in

$$\mathbf{M}\mathbf{x} = \delta_i$$

where δ_i is the Kronecker delta, x will be the i-th column of $\mathbf{M}^{-1}$.
Solving for $\mathbf{x}$ is $O(n^2)$.

[1] see backup slides for more

Backup: Error estimate on parameters

Calculation cost for one error:

- Basis: Alignment using 200 000 parameters (bows and composite modules)
- Memory footprint: 19 GB of RAM (grace to sparsity of the matrix)
- 12 minutes of wall-clock time (parallelized on 7 cores)
- 1.1 hrs of CPU time

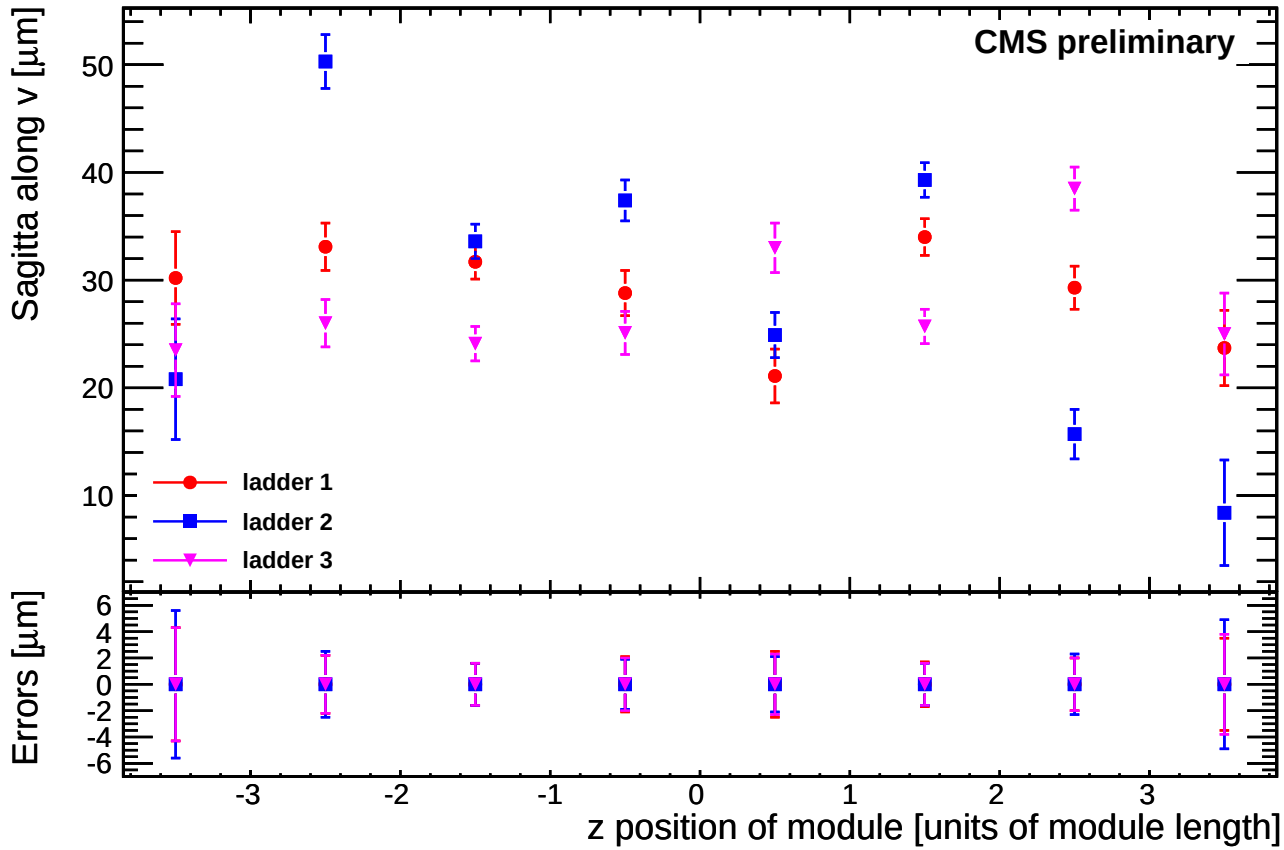
So determining the errors for all parameters would require more than 4 years...

Carried out on the Tier3 at PSI using one CPU with 8 cores (Intel Xeon Nehalem) and 24 GB of RAM.

PAUL SCHERRER INSTITUT

Results: Error estimate on parameters

Example: pixel modules in the innermost layers (sagitta in v):



Result obtained using several millions of collision and cosmic events.

Backup: How Millepede works

Millepede-II is a general solver for *linear least squares* problems with a special structure typical for alignment problems.
The expression for the χ^2 to be minimized is

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \mathbf{r}_{ij}^{T}(\mathbf{p},\mathbf{q}_j)\, \mathbf{V}_{ij}^{-1}\, \mathbf{r}_{ij}(\mathbf{p},\mathbf{q}_j) \tag{2}$$

where $\mathbf{p}$ denotes the alignment parameters describing the actual geometry and $\mathbf{q}_j$ denotes the track parameters of the j^{th} track.
Allow for the following identification:

- Alignment parameters ($\mathbf{p}$) $\mapsto$ global parameters
- Track parameters ($\mathbf{q}_j$) $\mapsto$ local parameters

Backup: How Millepede works

Nonlinear dependencies (angles) require local linearization:

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \frac{1}{\sigma_{ij}^2} \left(\mathbf{m}_{ij} - \mathbf{f}_{ij}(\mathbf{p}_0, \mathbf{q}_{j0}) - \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{p}} \Delta\mathbf{p} - \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{q}_j} \Delta\mathbf{q}_i \right)^2 \quad (3)$$

Here, $\mathbf{f}_{ij}$ is the hit position predicted by the track model from track reconstruction and $\mathbf{m}_{ij}$ is the measured hit position. Assuming uncorrelated measurements allows to replace th inverse covariance matrix by $\frac{1}{\sigma_{ij}^2}$.

To minimize this expression one does the obvious: rewrite is as a matrix expression and then differentiate and you will get corrections to your parameters as a vector $\mathbf{\Delta a}$.

PAUL SCHERRER INSTITUT

Backup: How Millepede works

Doing this ends up with[2]

$$\mathbf{\Delta a} = (\mathbf{A}^T \mathbf{W} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{W} \mathbf{r} \tag{4}$$

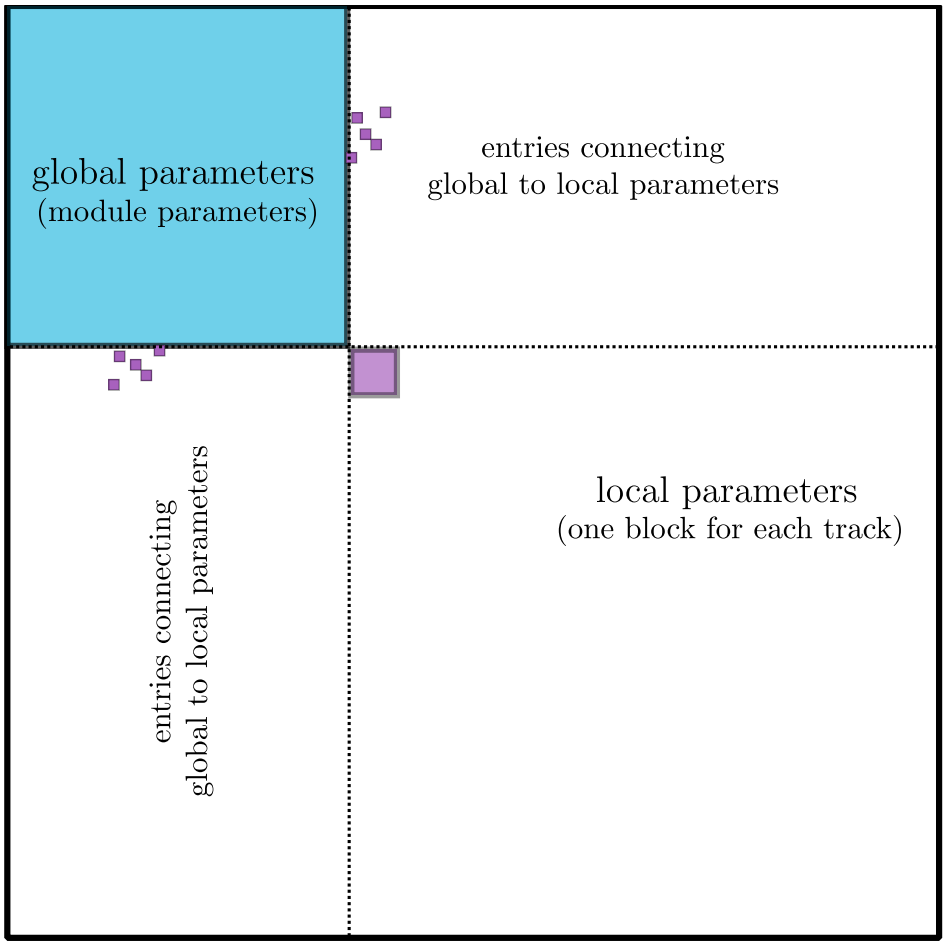
where $\mathbf{\Delta a}$ are the estimated correction of parameters, $\mathbf{A}$ the Jacobian, $\mathbf{W}$ the inverse covariance matrix of the measurements and $\mathbf{r}$ the residuals.
The estimate of the covariance matrix of the *parameters* $\mathbf{V}[\mathbf{\Delta a}]$ is then

$$\mathbf{V}[\mathbf{\Delta a}] = (\mathbf{A}^T \mathbf{W} \mathbf{A})^{-1} \tag{5}$$

If inversion is feasible, the errors are available.

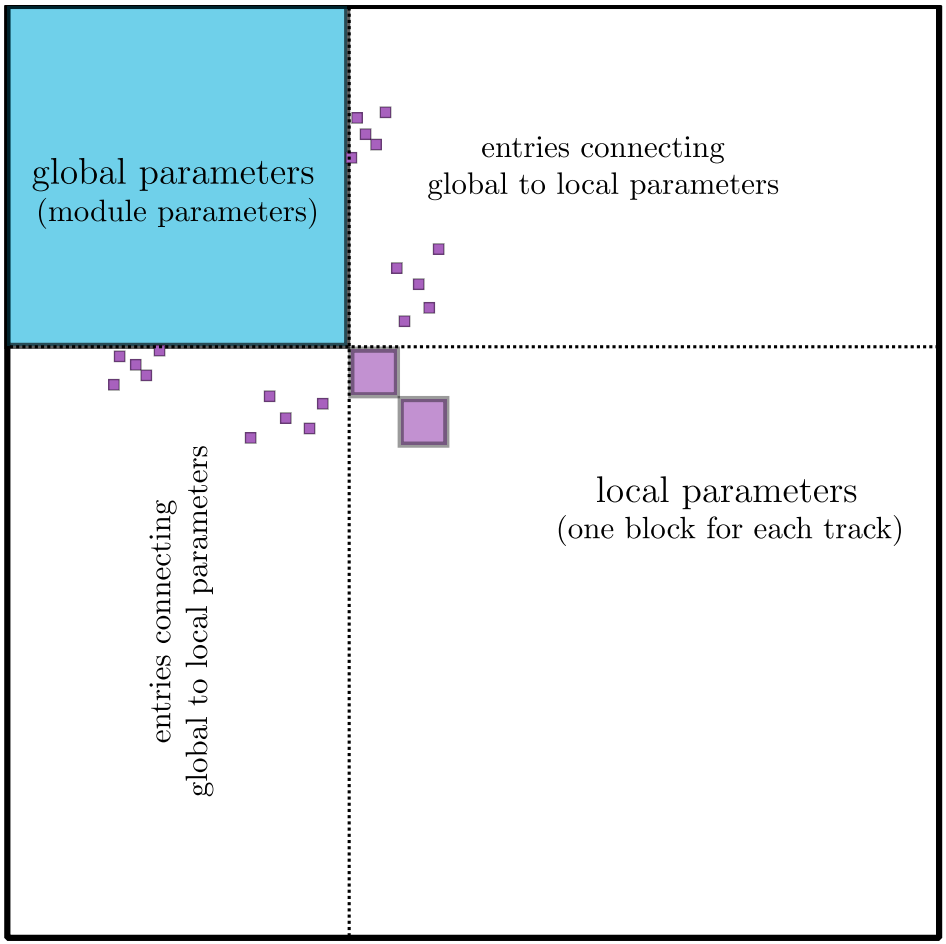
[2] following the notation in Blobel/Lohrmann

Backup: How Millepede works



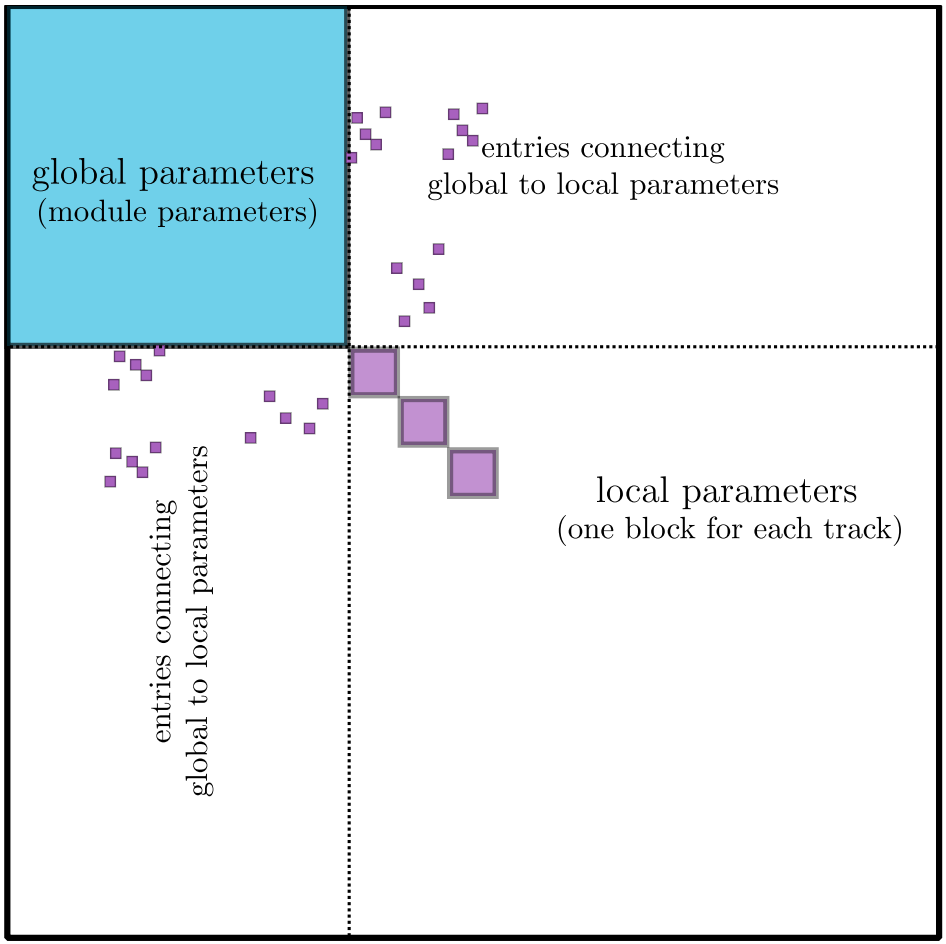
Arrange the matrix (and the vectors as well, but not shown here) in the following way: Put all global parameters at the begin, followed by the local parameters.

Backup: How Millepede works



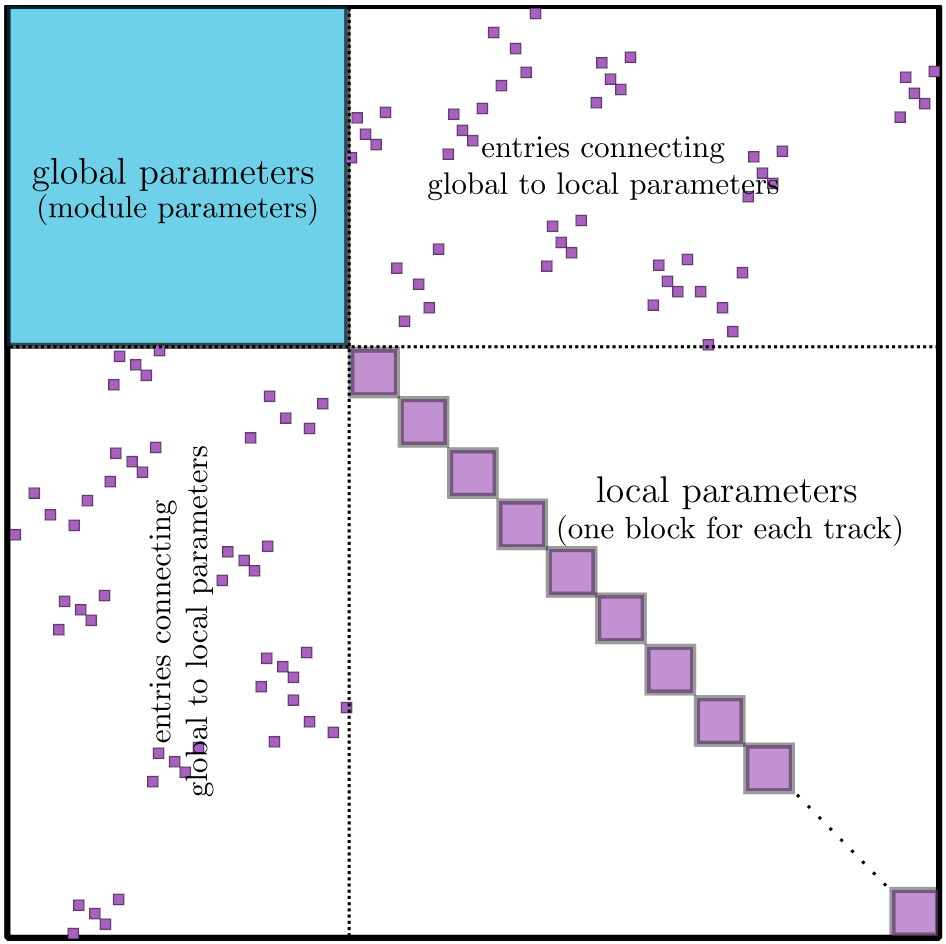
For each track a new block will be added. Entries in the global block are updated.

Backup: How Millepede works



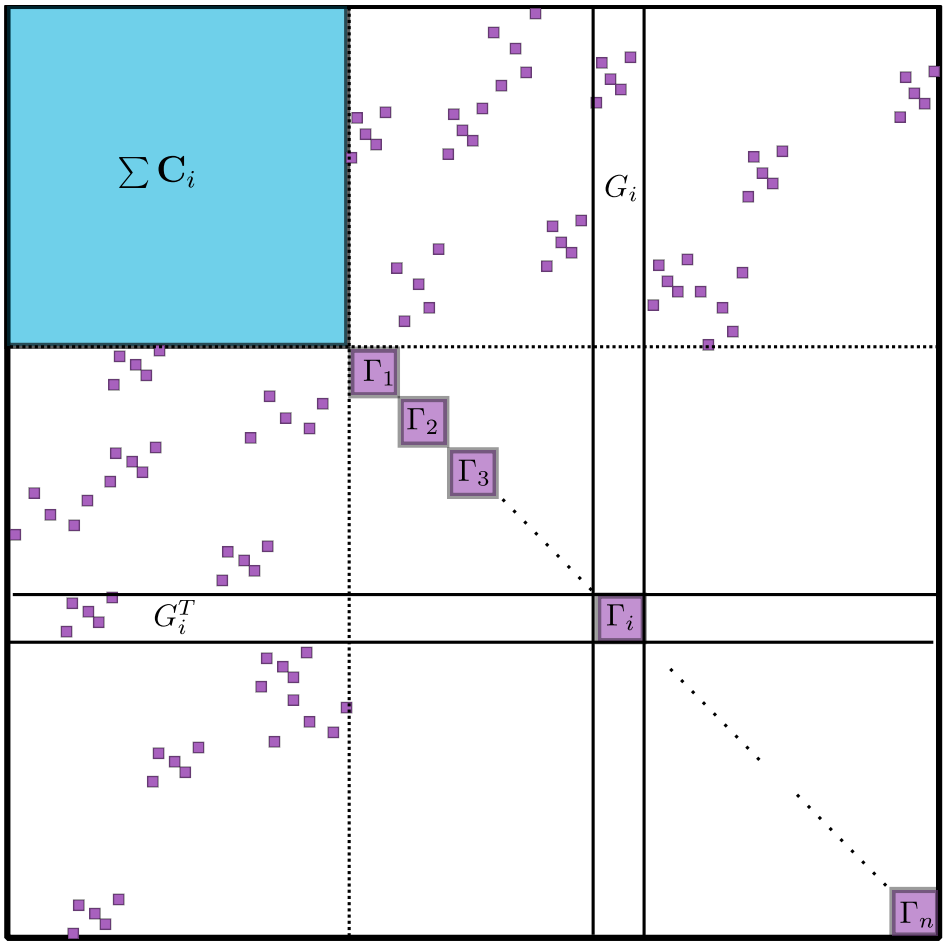
The entries for the local block are connected to the global parameters via the band parts of the matrix

Backup: How Millepede works



The pattern starts to appear: Each track contributes to the global and local parameters. The entries of the local parameters connect to the global parameters in the band outside

Backup: How Millepede works



More formally the matrix consists of the following parts:

- $\sum \mathbf{C}_i$: the block containining the sum of the contributions to the global parameters
- Γ_i: small blocks for each track, local parameters, disjoint between measurments
- G_i: band matrix connecting the local parameters of track i with the global parameters hit by the track

PAUL SCHERRER INSTITUT

Backup: How Millepede works

So you end up with this equation

$$\left(\begin{array}{c|ccc} \sum \mathbf{C}_i & \cdots & \mathbf{G}_i & \cdots \\ \hline \vdots & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{G}_i^T & \mathbf{0} & \mathbf{\Gamma}_i & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{0} & \ddots \end{array}\right) \cdot \left(\begin{array}{c} \mathbf{a} \\ \hline \vdots \\ \boldsymbol{\alpha}_i \\ \vdots \end{array}\right) = \left(\begin{array}{c} \sum \mathbf{b_i} \\ \hline \vdots \\ \boldsymbol{\beta}_i \\ \vdots \end{array}\right)$$

where the matrix has size

$$N = N_{\text{parameters}} + N_{\text{tracks}} \cdot N_{\text{track parameters}}$$

In alignment, you want to solve for the global parameters only, so there might be a possibility to reduce the problem to

$$N_{\text{parameters}}$$

PAUL SCHERRER INSTITUT

Backup: How Millepede works

Using some block matrix theorems (partitoning formulas for calculation of inverse matrices) you can reduce this problem to

$$\begin{pmatrix} \\ \mathbf{C}' \\ \\ \end{pmatrix} \cdot \begin{pmatrix} \\ \mathbf{a} \\ \\ \end{pmatrix} = \begin{pmatrix} \\ \mathbf{b}' \\ \\ \end{pmatrix}$$

where a new matrix and a new vector of size $N_{\text{parameters}}$ are used:

$$\mathbf{C}' = \sum \mathbf{C}_i - \sum \mathbf{G}_i \mathbf{\Gamma}_i^{-1} \mathbf{G}_i^T \qquad \mathbf{b}' = \sum \mathbf{b}_i - \sum \mathbf{G}_i (\mathbf{\Gamma}_i^{-1} \beta_i)$$

where $\mathbf{\Gamma}_i^{-1}$ is small and therefore can be calculated in reasonable time. Even though i might be large, the cost for solving the reduced equation is

$$N_{\text{pars}}^2 + N_{\text{tracks}} \cdot N_{\text{track pars}}^2 \ll (N_{\text{pars}} + N_{\text{tracks}} \cdot N_{\text{track pars}})^2$$

PAUL SCHERRER INSTITUT

Backup: How Millepede works

If you don't believe the impact, let us calculate:

$$N_{\text{pars}}^2 + N_{\text{tracks}} \cdot N_{\text{track pars}}^2 \ll (N_{\text{pars}} + N_{\text{tracks}} \cdot N_{\text{track pars}})^2$$

using the following typical values for an alignment in CMS:

- $N_{\text{pars}} = 200\,000$
- $N_{\text{tracks}} = 10^7$
- $N_{\text{track pars}} = 30$

this gives

$$5 \cdot 10^{10} \approx 4 \cdot 10^{10} + 10^7 \cdot 9 \cdot 10^2 \ll (2 \cdot 10^5 + 10^7 \cdot 30)^2 \approx 10^{17}$$