psi.ch/mu3e

Some thoughts on alignment

Frank Meier
Universität Heidelberg

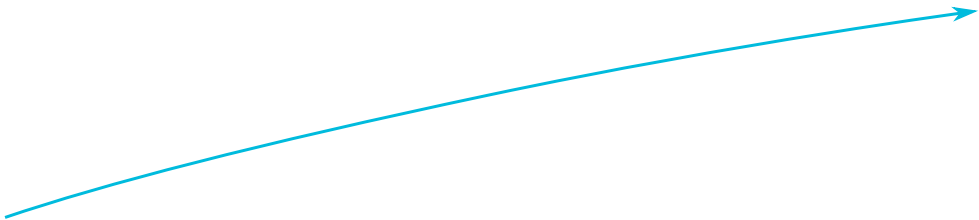
May 1, 2017

Preface

- This is a collection of slides – I will breeze over some of them but am happy to revisit
- Just for starting the discussion
- Although I'm with Mu3e today, I used to be „Alignment Convener" with CMS
- I'm an outsider, so please apologise for being mostly a „linear collider agnostic".

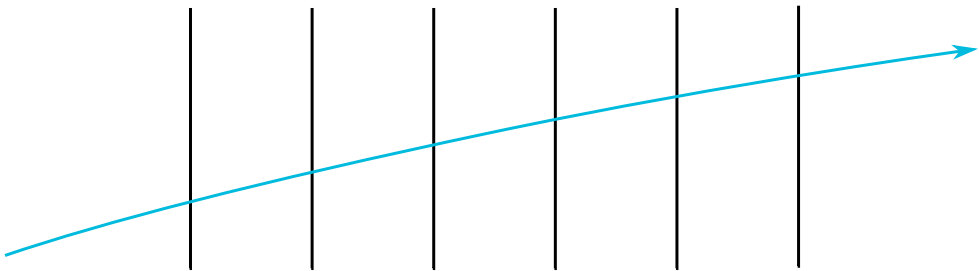
Disclaimer: Material shown here has a strong CMS bias and was shown and public occasions back during my time with CMS.

Alignment of a simple toy tracker



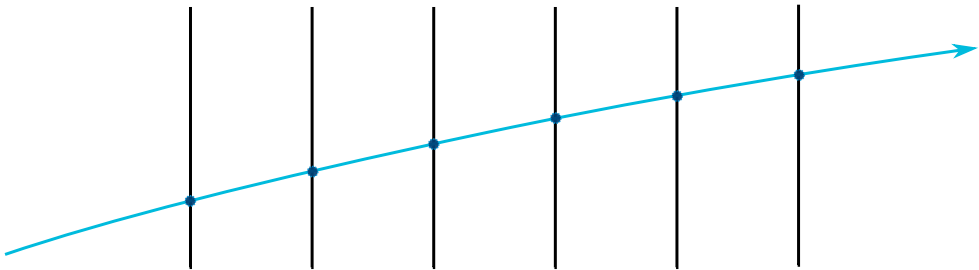
You want to track a charged particle in a magnetic field

Alignment of a simple toy tracker



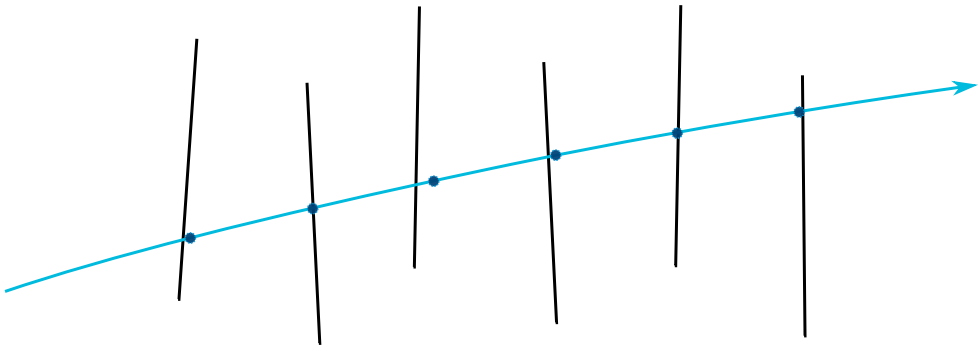
So you take six modules of your high-precision tracking detectors

Alignment of a simple toy tracker



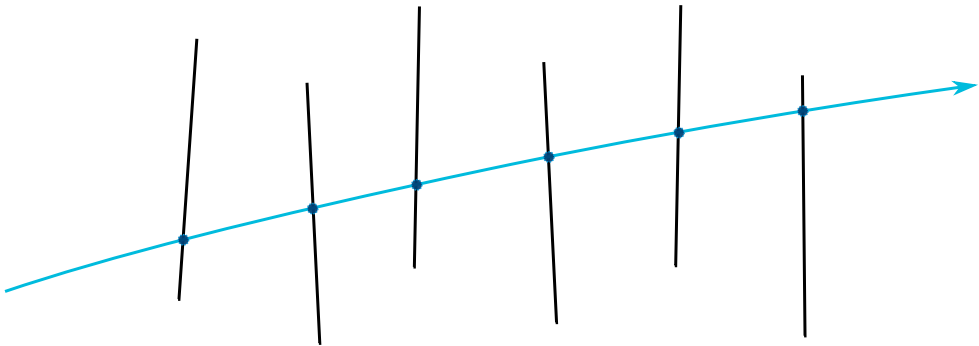
and you think you have recorded the right signal.

Alignment of a simple toy tracker



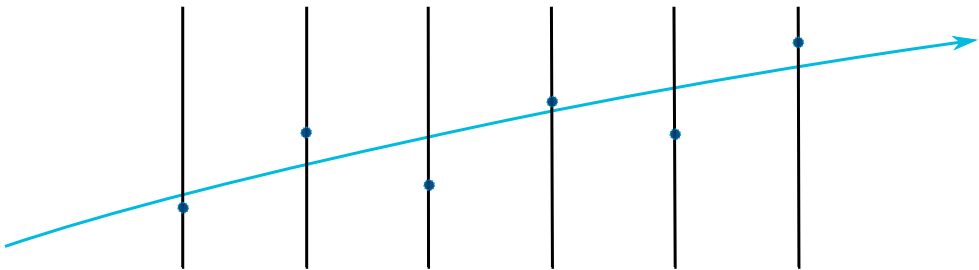
But unfortunately the modules were not mounted precisely where needed

Alignment of a simple toy tracker



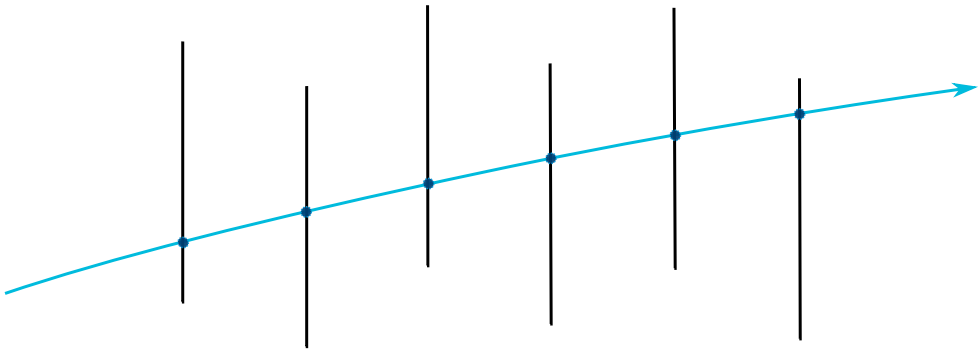
and your hits are not where you thought

Alignment of a simple toy tracker



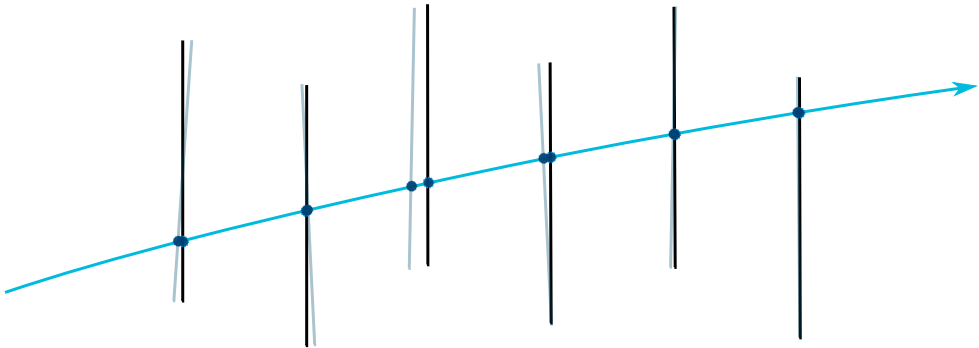
You still assume an ideal tracker and you know that the particle follows a smooth trajectory

Alignment of a simple toy tracker



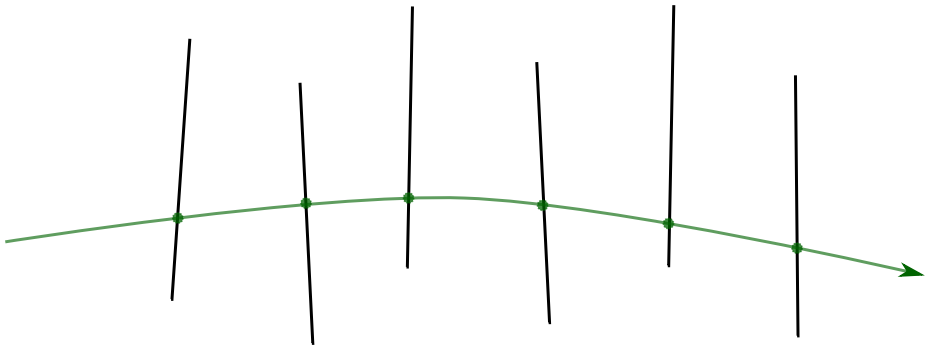
so you correct the modules' positions.

Alignment of a simple toy tracker



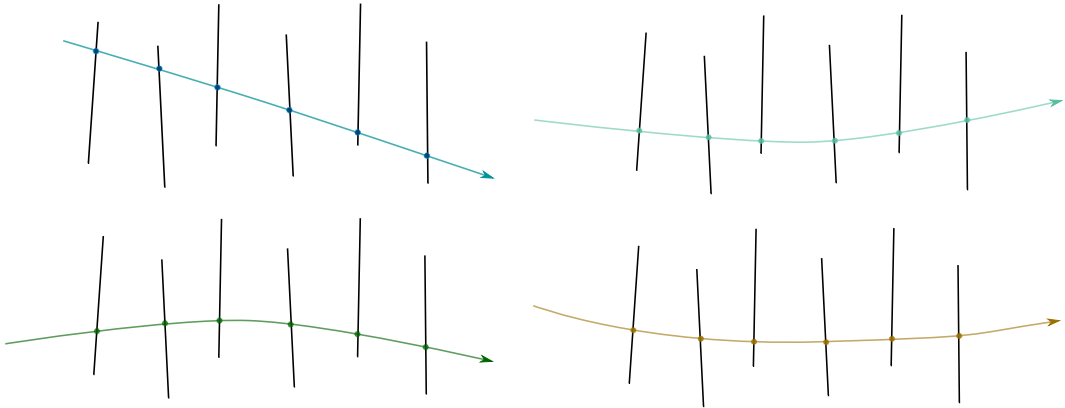
But compared to the reality, you are still off.

Alignment of a simple toy tracker



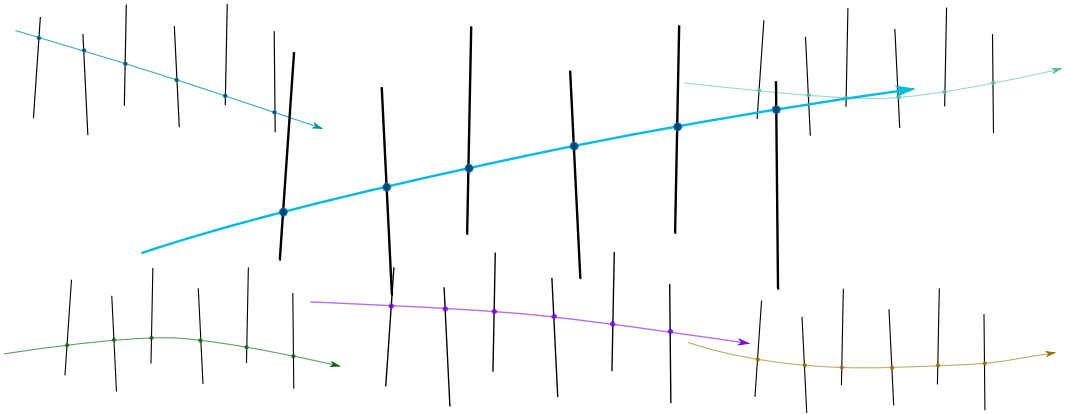
So you record more tracks

Alignment of a simple toy tracker



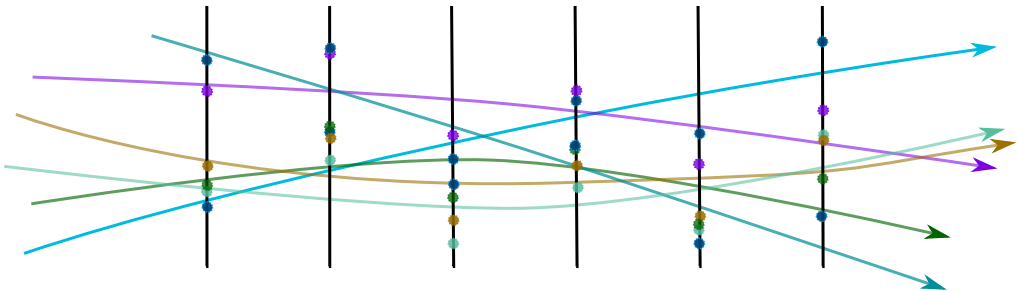
and more

Alignment of a simple toy tracker



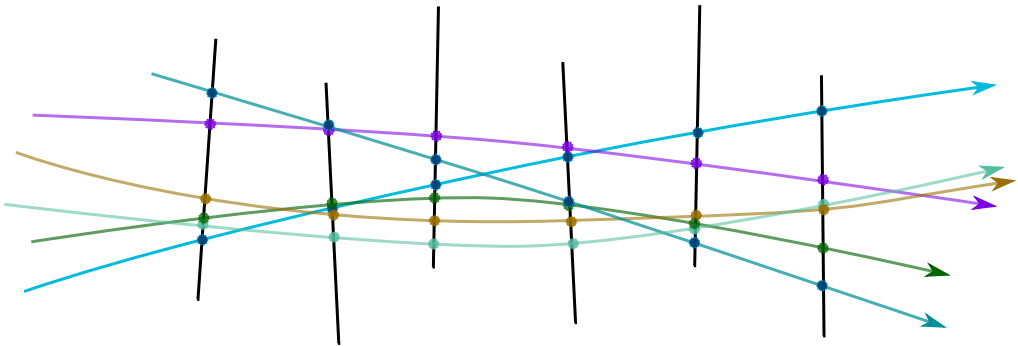
and even more

Alignment of a simple toy tracker



you take them all into account and deduce the true position

Alignment of a simple toy tracker



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

Algorithms

- Alignment can be done as a variant of a *linear least squares* problem

Algorithms

- Alignment can be done as a variant of a *linear least squares* problem
- The expression to be minimized is

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

where

- $\mathbf{p}$: alignment parameters describing the actual geometry
- $\mathbf{q}_j$: track parameters of the j^{th} track
- $\mathbf{r}_{ij}$: residual, distance between measured hit and position predicted by track fit
- $\mathbf{V}_{ij}^{-1}$: 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

- Alignment can be done as a variant of a *linear least squares* problem
- The expression to be minimized is

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

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

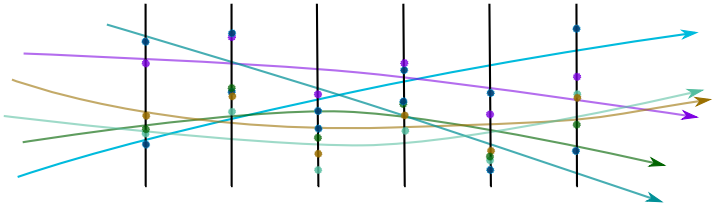
$$\left\langle \frac{\chi^2(\mathbf{p},\mathbf{q})}{\text{ndof}} \right\rangle = 1$$

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

Algorithms

CMS uses two competing algorithms:

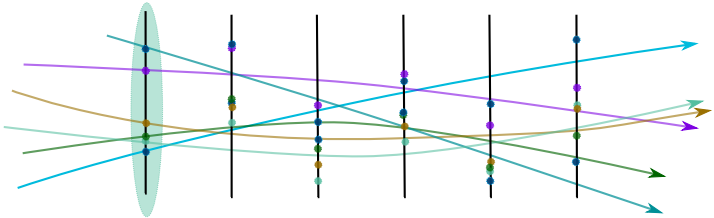
- one that works iteratively: HIP



Algorithms

CMS uses two competing algorithms:

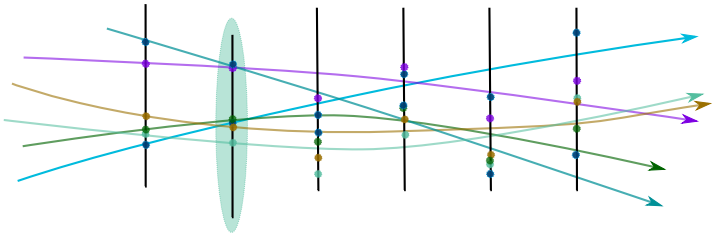
- one that works iteratively: HIP



Algorithms

CMS uses two competing algorithms:

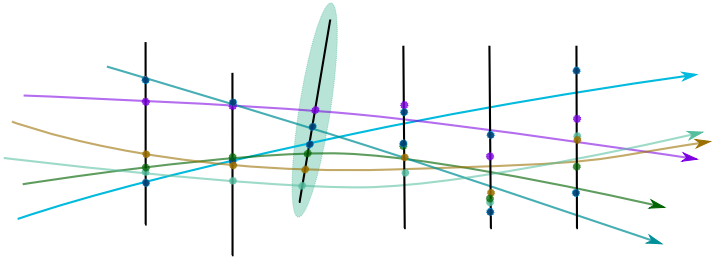
- one that works iteratively: HIP



Algorithms

CMS uses two competing algorithms:

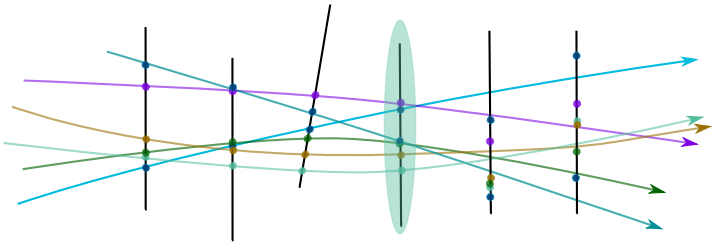
- one that works iteratively: HIP



Algorithms

CMS uses two competing algorithms:

- one that works iteratively: HIP



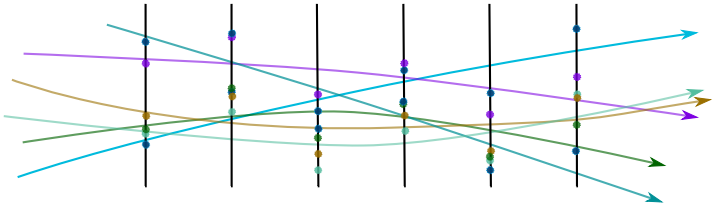
i.e. the sum in the χ^2-equation (1) runs over all tracks passing through one module, optimizes the parameters $\mathbf{p}$ of this module.

- Benefit: small problems to solve in each step
- Advantage: only local residuals contribute to χ^2, tf. insensitive to large misalignment in rest of tracker $\rightarrow$ startup alignment
- Disadvantage: a lot of iterations needed (recall: 16 588 modules with 5 to 6 d.o.f.)

Algorithms

CMS uses two competing algorithms:

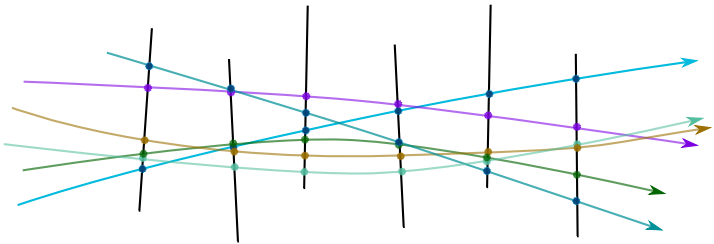
- one that works iteratively: HIP
- and one that does it in one step: Millepede-II



Algorithms

CMS uses two competing algorithms:

- one that works iteratively: HIP
- and one that does it in one step: Millepede-II



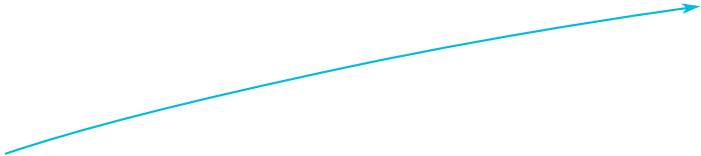
i.e. the sum in the χ^2-equation (1) runs over all tracks and modules

- Benefit: everything in one go
- Disadvantage: huge problem to solve
  (16 588 modules, 5 to 6 d.o.f.$\Rightarrow$ 80 000 $\times$ 80 000-matrix to handle)

Material effects

Two effects we have to take into account:

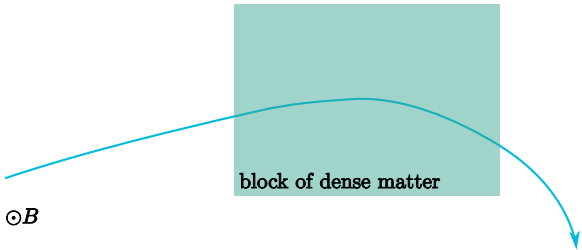
- Energy loss



Material effects

Two effects we have to take into account:

- Energy loss

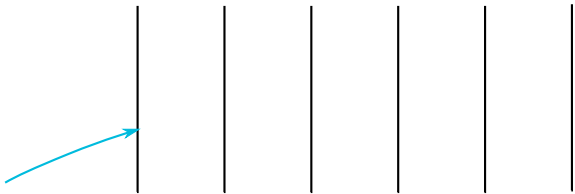


The charged track looses energy while passing through matter
$\Rightarrow$ particle looses momentum

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering

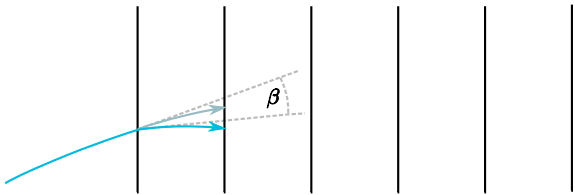


A charged particle hits material

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering

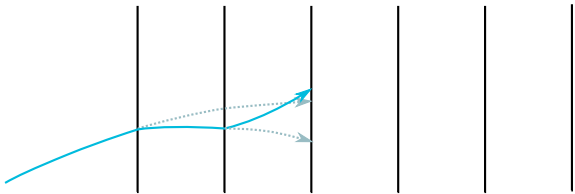


A charged particle hits material and scatters off some atoms in the material

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering

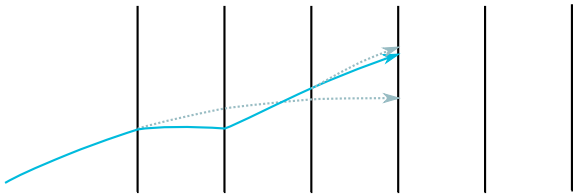


A charged particle hits material and scatters off some atoms in the material which changes its direction

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering

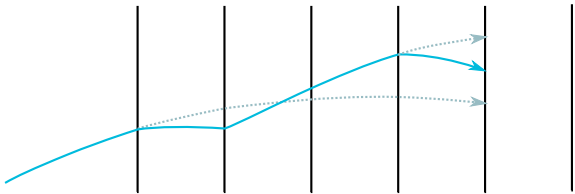


A charged particle hits material and scatters off some atoms in the material which changes its direction in a random fashion

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering

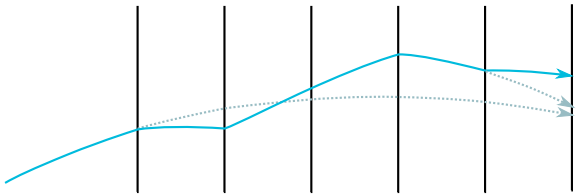


A charged particle hits material and scatters off some atoms in the material which changes its direction in a random fashion at every plane made of matter

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering

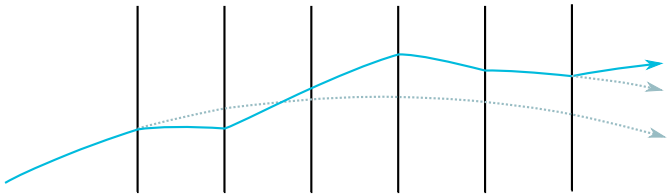


A charged particle hits material and scatters off some atoms in the material which changes its direction in a random fashion at every plane made of matter and ends up

Material effects

Two effects we have to take into account:

- Energy loss
- Multiple scattering



A charged particle hits material and scatters off some atoms in the material which changes its direction in a random fashion at every plane made of matter and ends up at a significantly different place and direction.

Multiple scattering

The mean of the deflection angle is $\langle\beta\rangle = 0$, but the $\sigma(\beta)$ can be approximated by[1]:

$$\sigma(\beta) = \frac{13.6\,\text{MeV}}{vp} z\sqrt{x/X_0}\,[1 + 0.038\ln(x/X_0)] \tag{2}$$

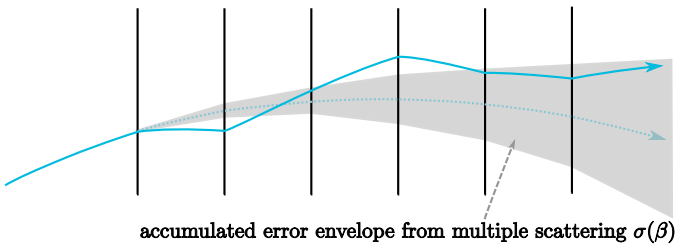
- $v = \beta c$ velocity of particle (here β is the relativistic velocity factor)
- p its momentum
- z its charge
- x/X_0 thickness of traversed material in radiation lengths

[1] formula as found in PDG booklet

Multiple scattering

How to take this into account?

- First attempt: Error envelope around whole track without multiple scattering

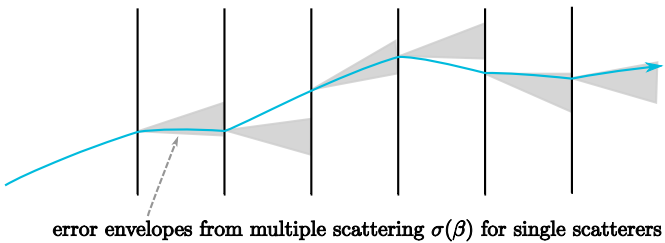


Disadvantage: σ of hits larger than expected, tf. $\langle \chi^2/\text{ndof} \rangle < 1$
Difficult to judge result as χ^2-distribution will be distorted.
This approach would work if correlations introduced by multiple scattering are taken into account. This is not implemented in Millepede-II.

Multiple scattering

How to take this into account?

- First attempt: Error envelope around whole track without multiple scattering
- Improvement: **Broken lines**
  Allow for kinks within error envelope for each track segment and use $\langle \beta \rangle = 0$



Advantage: Track model matches reality, $\langle \chi^2/\text{ndof} \rangle = 1$

Multiple scattering

How to take this into account?

- First attempt: Error envelope around whole track without multiple scattering
- Improvement: **Broken lines**
  Allow for kinks within error envelope for each track segment and use $\langle \beta \rangle = 0$
  Advantage: Track model matches reality, $\langle \chi^2/\text{ndof} \rangle = 1$
- Have to pay a price for proper implementation of multiple scattering: correlated hits or more track parameters.
  Choice of broken lines leads to bordered band matrix structure for the local fit $\Rightarrow$ fast solution by Cholesky decomposition.

Results

With this improved track model available in CMS-software, we did alignment using Millepede-II and got

- nice results using track data from **collisions**:

$$\langle \chi^2/\text{ndof} \rangle \approx 1.04$$

This is good when compared to simple envelope where $\langle \chi^2/\text{ndof} \rangle \approx 0.34$

Results

With this improved track model available in CMS-software, we did alignment using Millepede-II and got

- nice results using track data from **collisions**:

$$\langle \chi^2/\text{ndof} \rangle \approx 1.04$$

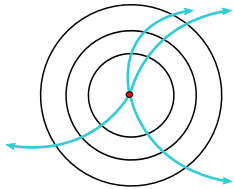
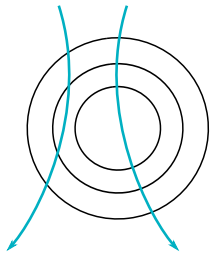
- unusable results using track data from **cosmic rays**:

$$\langle \chi^2/\text{ndof} \rangle > 1.6$$

  Millepede-II outlyer treatment requires fraction of outliers $< 1/3$. In this case, it aborted before finish.

Results

What are the differences between the two classes of tracks?[2]

	collision tracks	cosmic tracks
		

Results

What are the differences between the two classes of tracks?[2]

	collision tracks	cosmic tracks
Momentum	small, larger contribution of multiple scattering	medium
Track angle (in barrel)	medium, driven by $q/p \cdot R_{\text{hit}}$	large, driven by d_0/R_{hit}

- q/p: charge signed inverse momentum
- R_{hit}: radial distance of hit to the tracker center
- d_0: distance of closest approach of track to tracker center

Results

What are the differences between the two classes of tracks?[2]

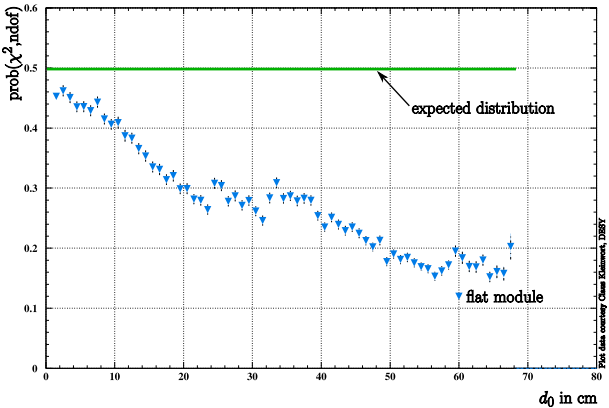
	collision tracks	cosmic tracks
Momentum	small, larger contribution of multiple scattering	medium
Track angle (in barrel)	medium, driven by $q/p \cdot R_{\text{hit}}$	large, driven by d_0/R_{hit}

- q/p: charge signed inverse momentum
- R_{hit}: radial distance of hit to the tracker center
- d_0: distance of closest approach of track to tracker center

Note: Plots shown on following slides are old but have educational qualities.

Results

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 (error bars).

Results

We clearly have a problem.

- First suspect: calculation of the hit errors at large angles.

Results

We clearly have a problem.

- First suspect: calculation of the hit errors at large angles.
  But this was not the solution. Just shifted/distorted the pattern.
  But this allowed for a hot fix anyway.

Results

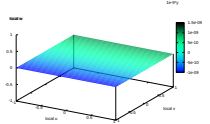
We clearly have a problem.

- First suspect: calculation of the hit errors at large angles.
  But this was not the solution. Just shifted/distorted the pattern.
  But this allowed for a hot fix anyway.
- May it be that we have a more complicated surface than flat sensors?

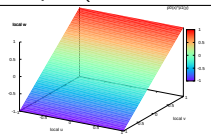
More complicated surface distorions

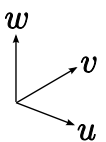
The movements in alignment are superpositions of

Shifts along the axes



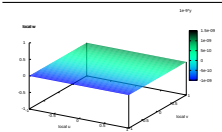
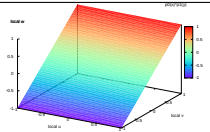
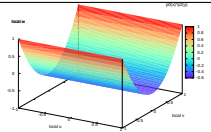
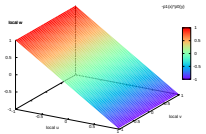
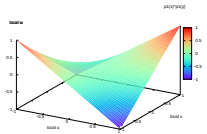
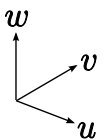
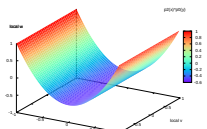
Slopes (i.e. rotations)





More complicated surface distorions

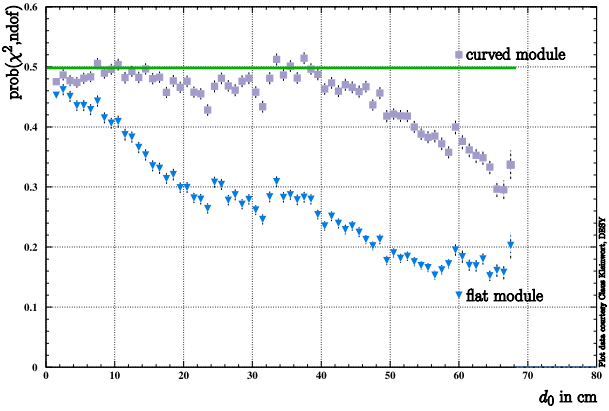
The movements in alignment are superpositions of

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

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

More complicated surface – more results

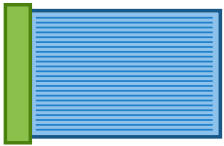
The same distribution of the probability of the χ^2 vs. d_0 looks far better now:



But for $d_0 \gtrapprox 40\,\text{cm}$ still something happens

Results

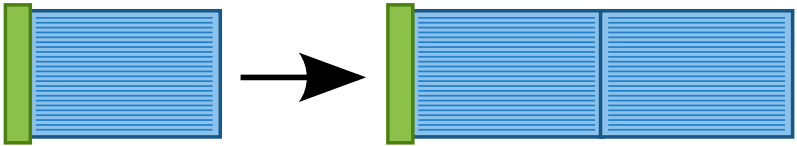
The topologies of the strip modules differ in the regions of the tracker:



In the inner regions, the modules consist of one sensor

Results

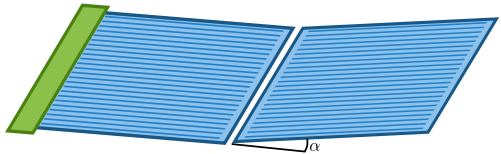
The topologies of the strip modules differ in the regions of the tracker:



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

Results

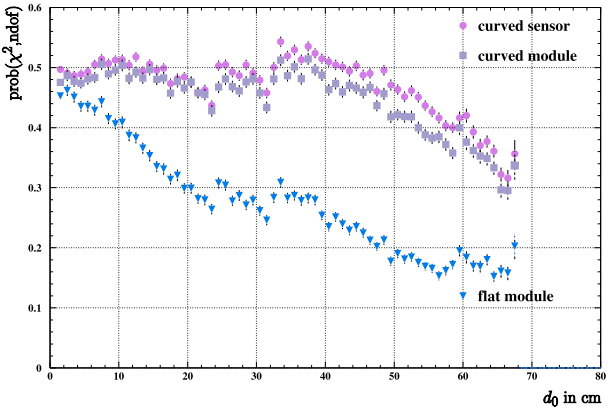
The topologies of the strip modules differ in the regions of the tracker:



These composite modules may have a kink in between

More complicated surface – more results

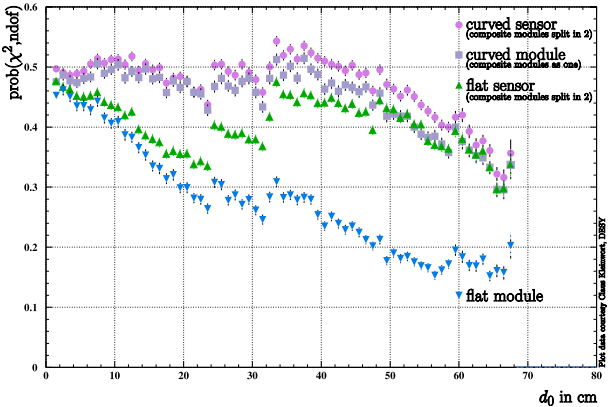
Splitting the composite sensors in two gives an improvement:



Observe that the track-χ^2 depends on the full track, not only the nearest hit.

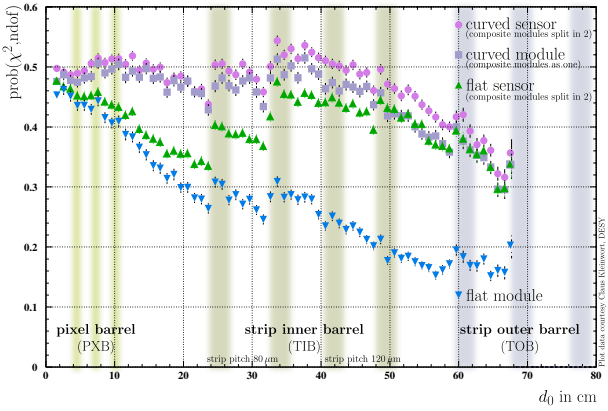
More complicated surface – more results

Here we have the full picture: Allow for bows in the sensor surface and for kinks between composite modules:



More complicated surface – more results

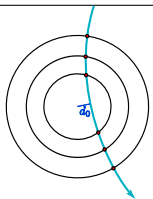
Here we have the full picture: Allow for bows in the sensor surface and for kinks between composite modules:



Adding the boundaries of the barrel layers is instructive and shows that an explanation for the jumps.

More complicated surface – more results

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

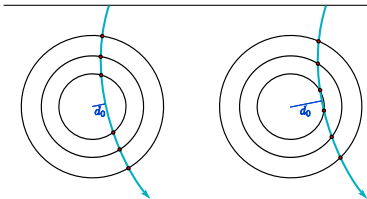
small d_0	larger d_0	$d_0 > r_{\text{layer}}$	and larger
			

Such a track shows moderate track angles at every hit

More complicated surface – more results

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

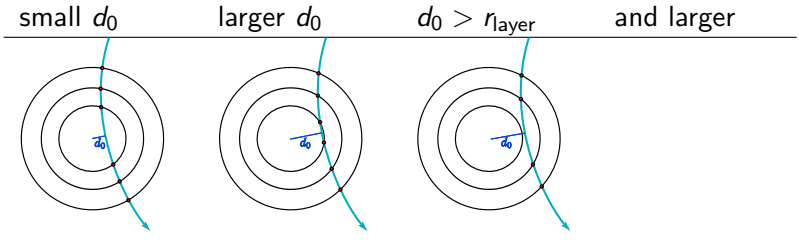
small d_0 | larger d_0 | $d_0 > r_{\text{layer}}$ | and larger



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 – more results

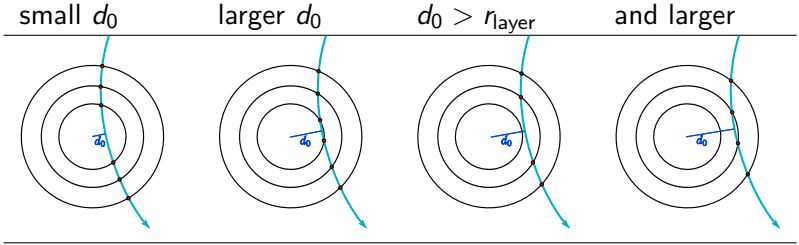
To understand these jumps, we need to see what happens as d_0 grows:



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

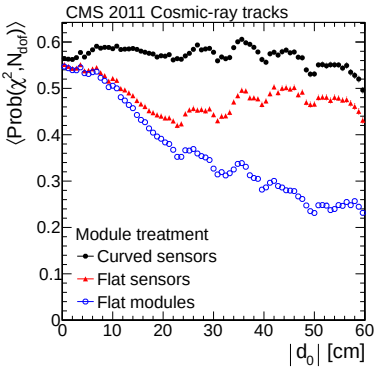
More complicated surface – more results

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



And the game starts over again

Results



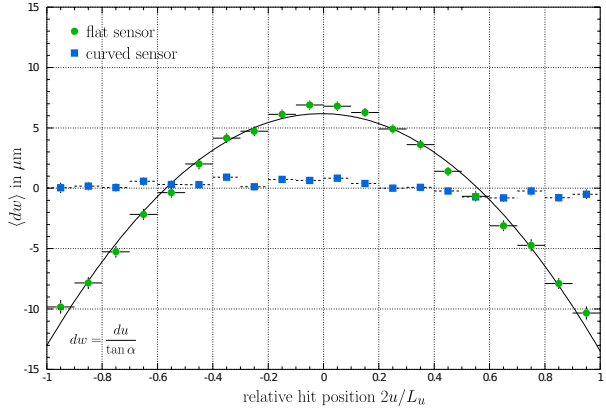
And this is the plot from the paper TRK-11-002, recently published.

`http://arxiv.org/abs/1403.2286`

With strong DESY involvement – big THANKS on behalf of the CMS TkAlignment conveners!

Results

Another look at the same problem:

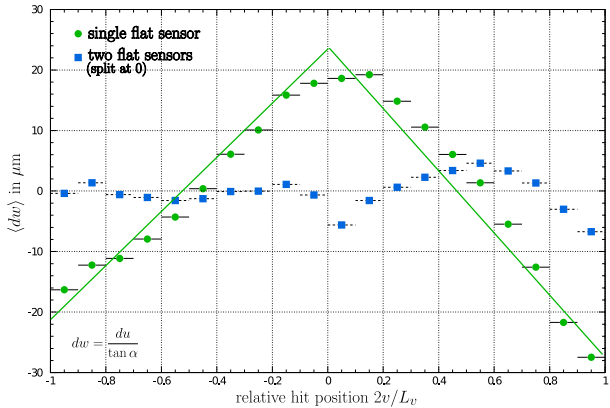


This shows the residuals when aligning for flat and curved sensor. Observe: The bowed curve belongs to the flat sensor assumption.

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

Results

And for composite modules:



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

Thoughts

Allow for a few personal comments on alignmnet of silicon detectors:

- **Survey** comes labelled with an expiration date (but you can't read it).
  Tracks in low-occupancy regimes – Cosmics – can be reconstructed even when heavily misaligned

Thoughts

Allow for a few personal comments on alignmnet of silicon detectors:

- **Survey** comes labelled with an expiration date (but you can't read it).
  Tracks in low-occupancy regimes – Cosmics – can be reconstructed even when heavily misaligned
- **Scale** comes to a large extent implicit when properly implemented
  Sensor pitch is a very precise ruler (photolithography)

Thoughts

Allow for a few personal comments on alignmnet of silicon detectors:

- **Survey** comes labelled with an expiration date (but you can't read it).
  Tracks in low-occupancy regimes – Cosmics – can be reconstructed even when heavily misaligned
- **Scale** comes to a large extent implicit when properly implemented
  Sensor pitch is a very precise ruler (photolithography)
- **Laser alignment systems** suffer from narrow distributions
  Particle tracks come with a distribution, laser beams not.

Thoughts

Allow for a few personal comments on alignmnet of silicon detectors:

- **Survey** comes labelled with an expiration date (but you can't read it).
  Tracks in low-occupancy regimes – Cosmics – can be reconstructed even when heavily misaligned
- **Scale** comes to a large extent implicit when properly implemented
  Sensor pitch is a very precise ruler (photolithography)
- **Laser alignment systems** suffer from narrow distributions
  Particle tracks come with a distribution, laser beams not.
- **Mechanical stability** must be sufficient, but that is enough.
  Movements slower than the time to collect tracks for alignment are perfectly ok.

Thoughts

Allow for a few personal comments on alignmnet of silicon detectors:

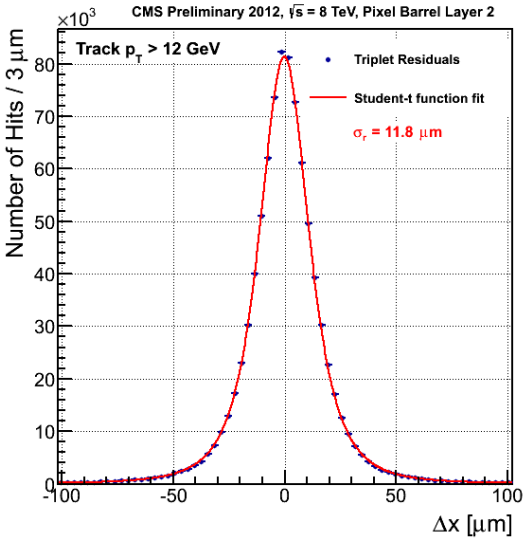
- **Survey** comes labelled with an expiration date (but you can't read it).
  Tracks in low-occupancy regimes – Cosmics – can be reconstructed even when heavily misaligned
- **Scale** comes to a large extent implicit when properly implemented
  Sensor pitch is a very precise ruler (photolithography)
- **Laser alignment systems** suffer from narrow distributions
  Particle tracks come with a distribution, laser beams not.
- **Mechanical stability** must be sufficient, but that is enough.
  Movements slower than the time to collect tracks for alignment are perfectly ok.
- **Weak modes** are a concern
  but can be handled by global alignment algorithms when **all** available information is used.

Thoughts

Allow for a few personal comments on alignmnet of silicon detectors:

- **Survey** comes labelled with an expiration date (but you can't read it).
  Tracks in low-occupancy regimes – Cosmics – can be reconstructed even when heavily misaligned
- **Scale** comes to a large extent implicit when properly implemented
  Sensor pitch is a very precise ruler (photolithography)
- **Laser alignment systems** suffer from narrow distributions
  Particle tracks come with a distribution, laser beams not.
- **Mechanical stability** must be sufficient, but that is enough.
  Movements slower than the time to collect tracks for alignment are perfectly ok.
- **Weak modes** are a concern
  but can be handled by global alignment algorithms when **all** available information is used.
- **Non-standard events** are extremely helpful, e.g. resonances (CMS: $\Upsilon, Z \to \mu\mu$)

Achievable resolution

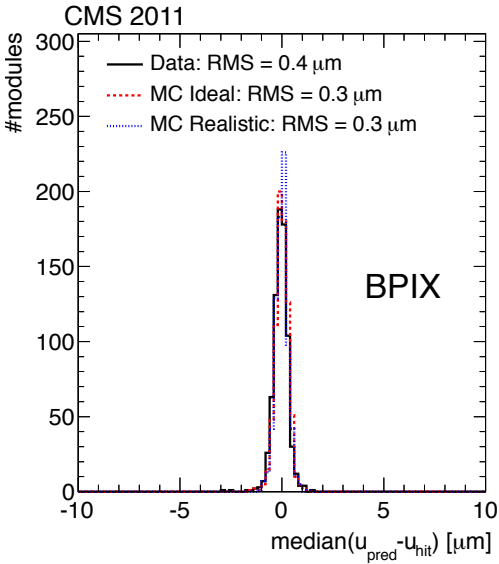


This shows the resolution measured for CMS pixel layer 2.

This demonstrates the power of pulse-height readout and make use of charge-sharing.

How precisely can you align this detector?

Achievable resolution

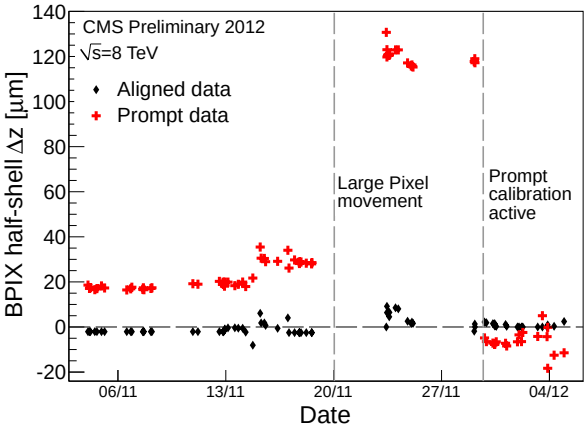


This is what you can reach.

We believed this to be a little overkill, but the heavy ion people loved this.

The precision comes from statistics and sound understanding of *all the details* of your detector.

Example for prompt alignment



The CMS pixel barrels are not fixed, they are standing on rails.

Changes in tempertaure or magnetic field can trigger movements.

A few minutes of data is sufficient to align this (if triggers wouldn't be that prescaled, a few seconds would do...)

Conclusions

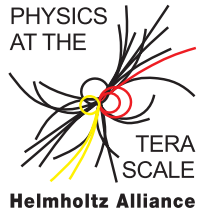
- Introduction of an improved track model for use with the Millepede-II alignment algorithm in CMS lead to problems with Cosmics
- More complex models introduced and cover now:
  - bowed sensors
  - composite modules with two sensor surfaces
  - Lorentz angles
  - Backplane correction
- Quite some infrastructure and organization needed.

Conclusions

- Introduction of an improved track model for use with the Millepede-II alignment algorithm in CMS lead to problems with Cosmics
- More complex models introduced and cover now:
  - bowed sensors
  - composite modules with two sensor surfaces
  - Lorentz angles
  - Backplane correction
- Quite some infrastructure and organization needed.
- This has entered routine

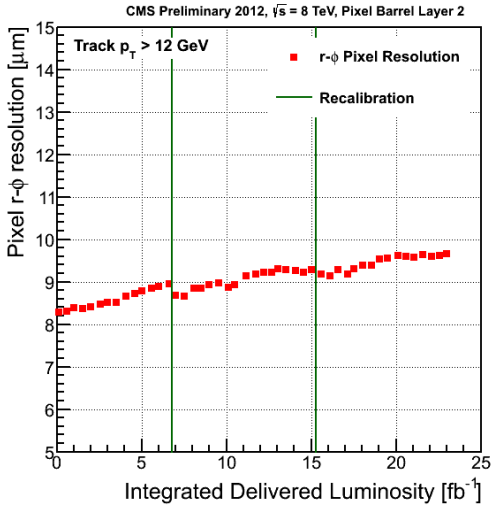
Acknowledgements

The success of CMS alignment is largely (!) based on work carried out at University of Hamburg and DESY (Volker Blobel, Claus Kleinwort, Gero Flucke, Rainer Mankel + students).
MP-II and GBL are supported by DESY/Terascale.



ENCORE

Resolution degradation



Backup: 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

Backup: 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
  - $S_u = \frac{3}{2} c_{22} \quad S_{uv} = c_{21} \quad S_v = \frac{3}{2} c_{20}$

Backup: Error estimate on parameters

How precise can we determine these bows? In principle, Millepede-II solves for $\mathbf{x}$ like in[3]

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

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

[3] 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[3]

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

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 $O(n^2)$.

[3] see backup slides for more

Backup: Error estimate on parameters

Calculation cost for one error:

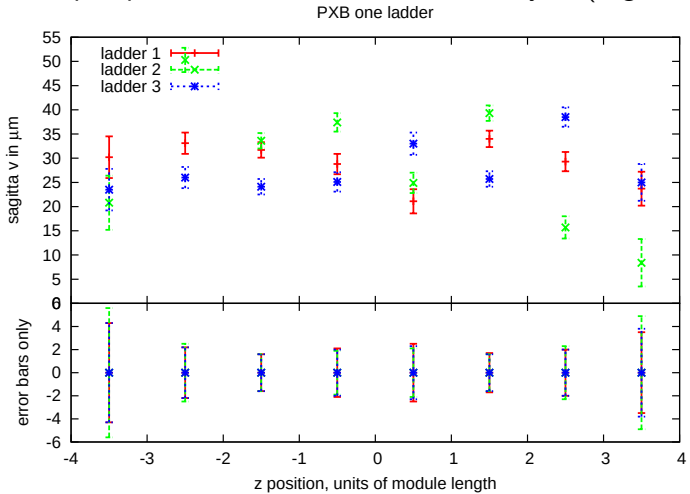
- Basis: Alignment using 200 000 parameters (bows and composite modules)
- Memory footprint: 19 GB of RAM (grace to sparsity of the matrix)
- 12 minutes of wall-clock time (parallelized on 7 cores)
- 1.1 hrs of CPU time

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.

Backup: Error estimate on parameters

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



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

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \mathbf{r}_{ij}^{T}(\mathbf{p},\mathbf{q}_j)\, \mathbf{V}_{ij}^{-1}\, \mathbf{r}_{ij}(\mathbf{p},\mathbf{q}_j) \tag{3}$$

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 require local linearization:

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \frac{1}{\sigma_{ij}^2} \left(\mathbf{m}_{ij} - \mathbf{f}_{ij}(\mathbf{p}_0, \mathbf{q}_{j0}) - \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{p}} \Delta\mathbf{p} - \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{q}_j} \Delta\mathbf{q}_i \right)^2 \quad (4)$$

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 $\mathbf{\Delta a}$.

Backup: How Millepede works

Doing this ends up with[4]

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

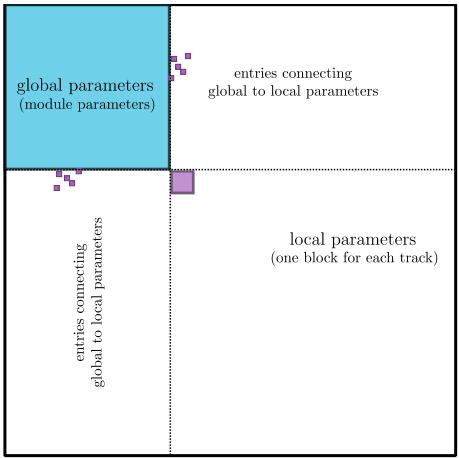
where $\mathbf{\Delta a}$ are the estimated correction of parameters, $\mathbf{A}$ the Jacobian, $\mathbf{W}$ the inverse covariance matrix of the measurements and $\mathbf{r}$ the residuals.
The estimate of the covariance matrix of the *parameters* $\mathbf{V}[\mathbf{\Delta a}]$ is then

$$\mathbf{V}[\mathbf{\Delta a}] = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1} \tag{6}$$

If inversion is feasible, the errors are available.

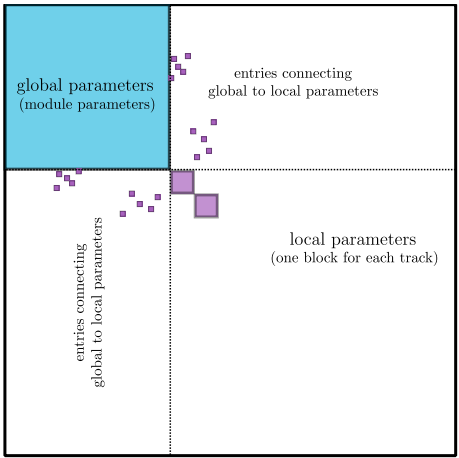
[4] following the notation in Blobel/Lohrmann

Backup: How Millepede works



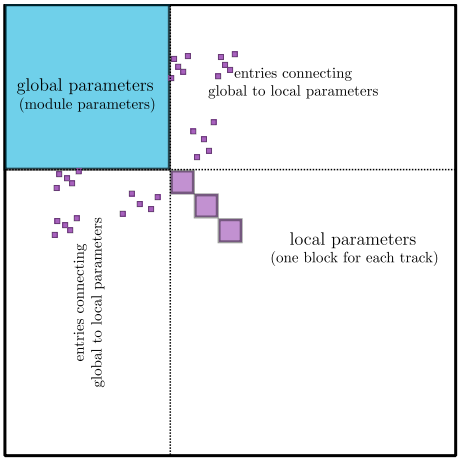
Arrange the matrix (and the vectors as well, but not shown here) in the following way: Put all global parameters at the begin, followed by the local parameters.

Backup: How Millepede works



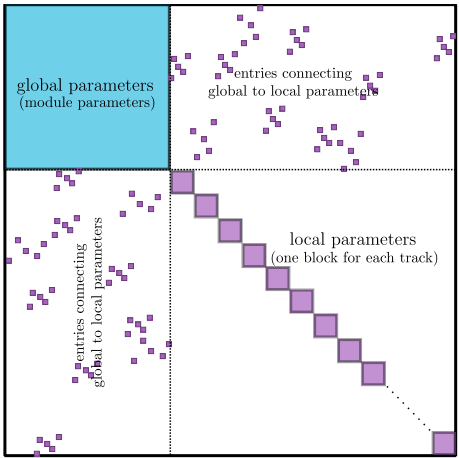
For each track a new block will be added. Entries in the global block are updated.

Backup: How Millepede works



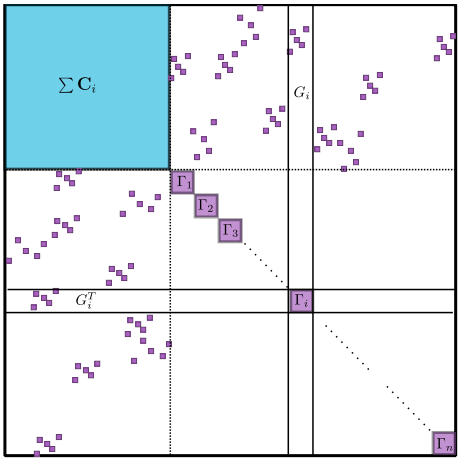
The entries for the local block are connected to the global parameters via the band parts of the matrix

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

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

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

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

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

Backup: How Millepede works

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

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

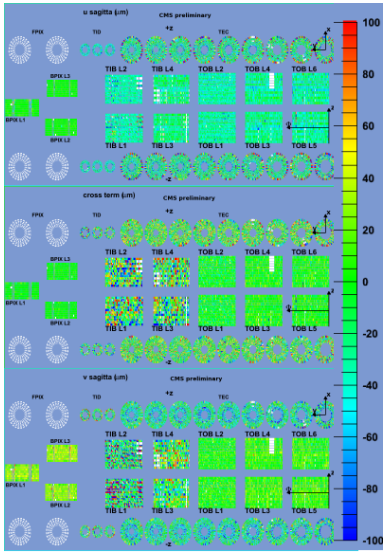
using the following typical values for an alignment in CMS:

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

this gives

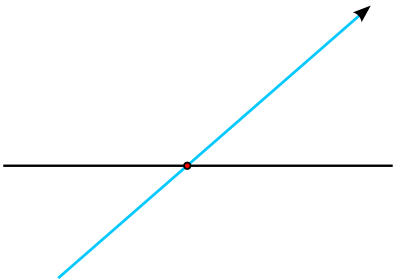
$$5 \cdot 10^{10} \approx 4 \cdot 10^{10} + 10^7 \cdot 9 \cdot 10^2 \ll (2 \cdot 10^5 + 10^7 \cdot 30)^2 \approx 10^{17}$$

Results



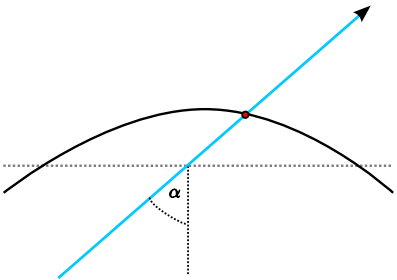
This is a rough overview of the bows determined using alignment.
Showed are the sagittae in μm.

What is the impact?



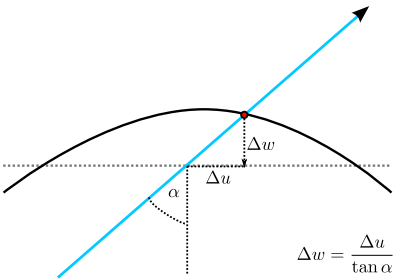
In a flat sensor, the hit is where it is.

What is the impact?



As soon as there is a bow, the track angle α determines the shift due to a deviation from the anticipated position.
For large α typically observed in cosmic tracks, this gets substantial.

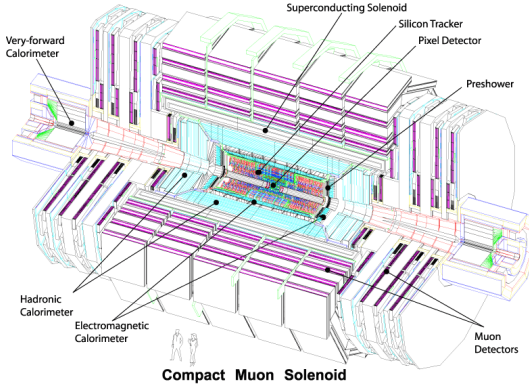
What is the impact?



As soon as there is a bow, the track angle α determines the shift due to a deviation from the anticipated position.
For large α typically observed in cosmic tracks, this gets substantial.

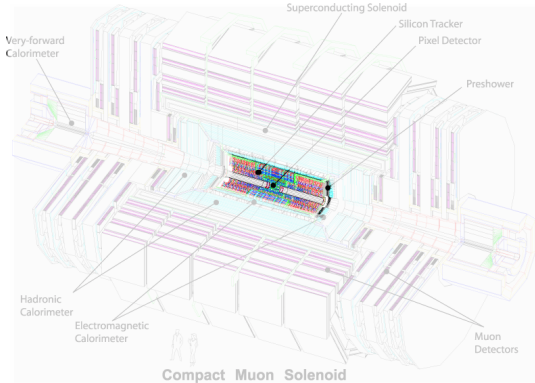
- The intrinsic resolution is as low as 15 μm (pixel detector).
- If Δw is of the same order, there will be a systematic error in the tracking.
- **Worst case:** sagitta of 30 μm, $\eta = 2.5 \Rightarrow \tan \alpha = 6$ (far end of pixel barrel, track from interaction point):
  $du = 30\,\mu\text{m} \cdot 6 = 180\,\mu\text{m}$

A brief detour: The CMS tracker



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

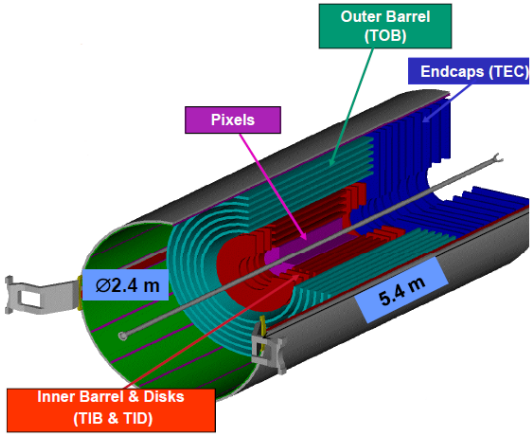
A brief detour: The CMS tracker



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

- 1440 silicon pixel modules
- 15 148 silicon strip modules

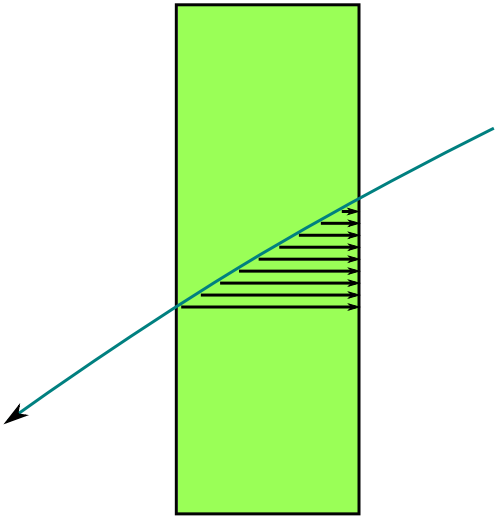
A brief detour: The CMS tracker



This is a closer view of the tracker. Observe the layerwise structure in the barrel and endcap.

Image courtesy Steven Lowette

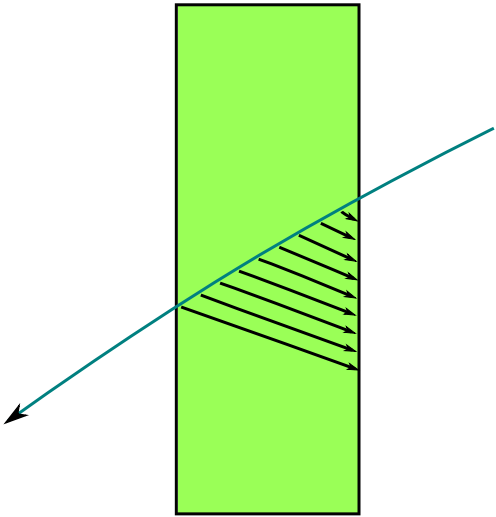
More parameters



A sensor (pixel, strip) collects charge from an ionizing particle traversing the active volume.

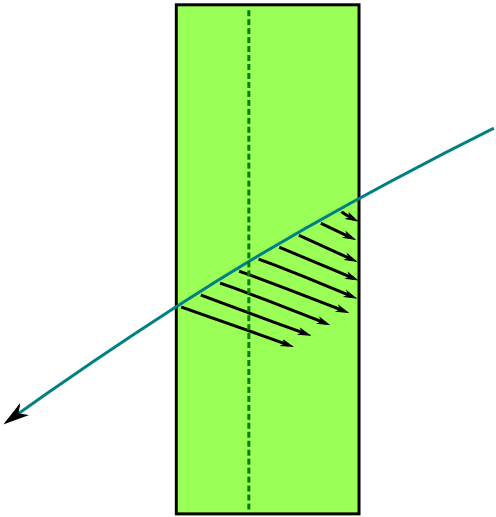
Without any B-field, the charge propagates along the E-field and gets collected.

More parameters



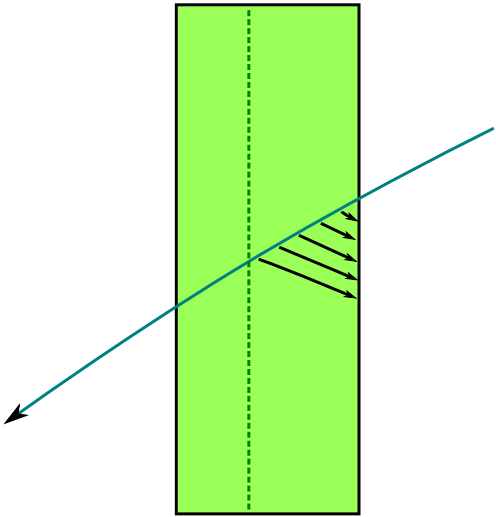
With B-field, another component is added to the drift. Usually described by a so called Lorentz angle (LA).

More parameters



In case of an integration time shorter than the drift time, not all charge reaches the read-out amplifier.

More parameters



This can be described by a (hypothetical) backplane inside th detector.

This is the case for strip sensors in CMS (when run in deconvolution mode).

Computing challenges

What are the challenges in aligning the CMS tracker?

- Prompt calibration loop

Computing challenges

What are the challenges in aligning the CMS tracker?

- Prompt calibration loop
  - Runs during datataking
  - Every about 10 000 good tracks a simple alignment is started
  - If movements are above threshold: check with larger sample, update conditions DB if manifest
  - Reason: Pixel barrel movements

Computing challenges

What are the challenges in aligning the CMS tracker?

- Prompt calibration loop
- Full alignment

Computing challenges

What are the challenges in aligning the CMS tracker?

- Prompt calibration loop
- Full alignment
  - Align for the full detector (16k objects)
  - 6 position parameters (3 coordinates and 3 angles)
  - 3 bow parameters
  - kink parameters for composite modules
  - end up with 200k parameters
  - requires 32 MB of memory

Computing challenges

What are the challenges in aligning the CMS tracker?

- Prompt calibration loop
- Full alignment
- Lorentz angle and BP alignment

Computing challenges

What are the challenges in aligning the CMS tracker?

- Prompt calibration loop
- Full alignment
- Lorentz angle and BP alignment
  - Even more parameters
  - Calls for even bigger machines (was done at DESY last year, now we have 256 MB machines at Cern)

Computing challenges

200 000 parameters $\rightarrow$ 200 000 $\times$ 200 000 matrix $= 40 \cdot 10^{12}$ numbers... How to handle this?

- Matrix is sparse (about 30% of off-diagonal elements are $\neq 0$)
- Apply compression
- Parallelize task using OpenMP
- Equation system solved using MINRES and not inversion
- Make use of recent hardware

One example: Full alignment using 22 Mio tracks runs for about 10 wallclock hours on an 8 core CPU with 48 GB of RAM memory