Chapter 1

Experimental Setup

1.1 The Large Hadron Collider

1.2 The Compact Muon Solenoid

1.2.1 The Magnet

1.2.2 Inner Tracker

Figure .. shows an r-z view of the CMS Tracker.

1.2.3 Electromagnetic Calorimeter

1.2.4 Hadronic Calorimeter

1.2.5 Muon System

1.3 Alignment of CMS tracker

The task of CMS tracker is to measure the trajectories of charged particle with high resolution in track position, angle and momentum. Excellent tracking performance is essential for analysis of physics processes. For example, searches for high mass resonances in leptonic final states require good momentum resolution for transverse momenta above 1TeV. In addition, excellent impact parameter resolution of reconstructed tracks is important for reconstruction of beam spot, primary vertices and b-jets tagging.

Tracker geometry refers to a complete set of parameters describing the geometrical properties (both location and angles) for the modules composing the tracker, and it is one of the most important input for track reconstruction. The uncertainty of tracker geometry should remain below the intrinsic silicon hit resolution of around 10μm for pixels and 30μm for strips.

There are many challenges to determine to tracker geometry, such as the limited accessibility of the tracker, the large number of modules to align, the high precision

required, and the constant changing in geometry due to changes in environment conditions. Track-based internal alignment is effective in solving these problems, and it is responsible for adjusting the positions and angles of the tracker modules relative to each other. Survey measurements of TOB is used to determine its global position relative to the beam axis and the other sub-components are aligned relative to TOB by track-based alignment algorithms.

1.3.1 The Problem

Track based alignment determine the module positions using the tracks reconstructed by the tracker geometry in situ, because residuals are sensitive to the tracker geometry used in track reconstruction relative to the true positions and angles of tracker modules. It can be treated as a least square minimization problem, where sum of the squares of track-hit residuals from a collection of tracks is minimized.

The objective function to minimize is:

$$\chi^2(\mathbf{p},\mathbf{q}) = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{measurements}} \left(\frac{m_{ij} - f_{ij}\left(\mathbf{p},\mathbf{q}_j\right)}{\sigma_{ij}} \right)^2, \tag{1.1}$$

where the measurement m_{ij} is the reconstructed hit positions on the modules, σ_{ij} is uncertainty of the measurement, and f_{ij} is the trajectory prediction of the track model at the position of the measurement, depending on the geometry parameters $\mathbf{p}$ and track parameters $\mathbf{q}_j$ (slope and curvature of the track).

1.3.2 Parameterization

1.3.3 Linear Least Squares Model

Since an initial geometry is used to determine the approximate track parameters, and we assume the corrections from the initial geometry to be small, and the fitted trajectories can be approximated with a straight line in the vicinity of the detector plane.

f_{ij} can be linearized around its initial values:

$$f_{ij}\left(\mathbf{p}+\Delta\mathbf{p},\mathbf{q}_j+\Delta\mathbf{q}_j\right)\simeq f_{ij}\left(\mathbf{p},\mathbf{q}_j\right)+\left(\frac{\partial f_{ij}}{\partial\mathbf{p}}\right)^T\Delta\mathbf{p}+\left(\frac{\partial f_{ij}}{\partial\mathbf{q}_j}\right)^T\Delta\mathbf{q}_j \tag{1.2}$$

We can combine geometry and track parameters as a vector $\mathbf{r}$ and write residual in vector form:

$$\boldsymbol{\varepsilon}(\mathbf{r}+\Delta\mathbf{r})\simeq\boldsymbol{\varepsilon}(\mathbf{r})+\mathbf{A}^T\Delta\mathbf{r} \tag{1.3}$$

where $\varepsilon(\mathbf{r})$ is the initial residual and $\varepsilon(\mathbf{r})$ is the residual with corrected parameters. A is the Jacobian matrix $\mathbf{A}=\nabla_{\mathbf{r}}\boldsymbol{\varepsilon}(\mathbf{r})$. For simplicity, $\boldsymbol{\varepsilon}(\mathbf{r})$ will be denoted $\boldsymbol{\varepsilon}$ below.

The objective function 1.1 becomes:

$$\chi^2(\mathbf{r}+\Delta\mathbf{r})\simeq\left(\boldsymbol{\varepsilon}+\mathbf{A}^T\Delta\mathbf{r}\right)^T\mathbf{V}^{-1}(\boldsymbol{\varepsilon}+\mathbf{A}^T\Delta\mathbf{r}) \tag{1.4}$$

where $\mathbf{V}$ is the covariance matrix of the measurements.

To derive the optimal correction vector $\Delta\mathbf{r}$ for minimization, we can simply differentiate the objective function w.r.t. $\Delta\mathbf{r}$ and equating to zero. This leads to the

normal equations to the linear least square solutions:

$$\mathbf{A}\mathbf{V}^{-1}\boldsymbol{\varepsilon} + (\mathbf{A}\mathbf{V}^{-1}\mathbf{A}^T)\Delta\mathbf{r} = 0 \tag{1.5}$$

or

$$\mathbf{C} \times \Delta\mathbf{r} = \mathbf{d} \tag{1.6}$$

where the symmetric matrix $\mathbf{C} = \mathbf{A}\mathbf{V}^{-1}\mathbf{A}^T$ and the vector $\mathbf{d} = -\mathbf{A}\mathbf{V}^{-1}\boldsymbol{\varepsilon}$.

To solve for the optimal correction $\Delta\mathbf{r}$, the straightforward way is to invert the matrix C.However, there are 16588×6 geometry parameters and millions of track parameters, leading to a huge n-by-n matrix C with $n > 10^6$. The computing time is $O(n^3)$ for matrix inversion, so the solution in this form is not practical in terms of computing time and storage space.

It's also important to note that sometimes the residual depend non-linearly on the track parameters. In this case a few iterations is needed.

1.3.4 Matrix Partitioning

The huge n-by-n matrix $\mathbf{C}$ in 1.6 can be partitioned into four submatrices:

$$\mathbf{C} = \left(\begin{array}{c|c} \mathbf{B} & \mathbf{G} \\ \hline \mathbf{G}^T & \boldsymbol{\Gamma} \end{array}\right) \tag{1.7}$$

Where the submatrix $\mathbf{B}$ is the symmetric p-by-p Hessian matrix for geometry parameters (p is the number of geometry parameters), the submatrix $\boldsymbol{\Gamma}$ is the symmetric q-by-q Hessian matrix for track parameters (q is the number of track parameters),

and the submatrix $\mathbf{G}$ is a q-by-p matrix relating the geometry parameters to track parameters. Since the tracks are independent of each other, $\mathbf{\Gamma}$ is mostly diagonal, i.e. only the hits on the same track are correlated.

The matrix equation 1.6 can be written in the form

$$\left(\begin{array}{c|c} \mathbf{B} & \mathbf{G} \\ \hline \mathbf{G}^T & \mathbf{\Gamma} \end{array}\right)\left(\begin{array}{c} \mathbf{\Delta p} \\ \hline \mathbf{\Delta q} \end{array}\right) = \left(\begin{array}{c} \mathbf{b} \\ \hline \boldsymbol{\beta} \end{array}\right) \tag{1.8}$$

Inverting the entire matrix $\mathbf{C}$ can be simplified by performing the block Gaussian elimination shown below.

The product of $\mathbf{C}$ with the lower triangular matrix $\mathbf{L}$ (defined below) can be written as the product of a upper triangular matrix $\mathbf{U}$ and a diagonal matrix $\mathbf{D}$:

$$\begin{aligned} \mathbf{CL} &= \left(\begin{array}{c|c} \mathbf{B} & \mathbf{G} \\ \hline \mathbf{G}^T & \mathbf{\Gamma} \end{array}\right)\left(\begin{array}{c|c} \mathbf{I_p} & \mathbf{0} \\ \hline -\mathbf{\Gamma}^{-1}\mathbf{G}^T & \mathbf{I_q} \end{array}\right) \\ &= \left(\begin{array}{c|c} \mathbf{B}-\mathbf{G}\mathbf{\Gamma}^{-1}\mathbf{G}^T & \mathbf{G} \\ \hline \mathbf{0} & \mathbf{\Gamma} \end{array}\right) \\ &= \left(\begin{array}{c|c} \mathbf{I_p} & \mathbf{G}\mathbf{\Gamma}^{-1} \\ \hline \mathbf{0} & \mathbf{I_q} \end{array}\right)\left(\begin{array}{c|c} \mathbf{B}-\mathbf{G}\mathbf{\Gamma}^{-1}\mathbf{G}^T & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{\Gamma} \end{array}\right) = \mathbf{UD} \end{aligned} \tag{1.9}$$

In this way we find a LDU decomposition for the block matrix $\mathbf{C}$:

$$\mathbf{C} = \mathbf{UDL}^{-1} = \left(\begin{array}{c|c} \mathbf{I_p} & \mathbf{G}\mathbf{\Gamma}^{-1} \\ \hline \mathbf{0} & \mathbf{I_q} \end{array}\right)\left(\begin{array}{c|c} \mathbf{B}-\mathbf{G}\mathbf{\Gamma}^{-1}\mathbf{G}^T & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{\Gamma} \end{array}\right)\left(\begin{array}{c|c} \mathbf{I_p} & \mathbf{0} \\ \hline \mathbf{\Gamma}^{-1}\mathbf{G}^T & \mathbf{I_q} \end{array}\right) \tag{1.10}$$

and the inverse of $\mathbf{C}$ can be expressed as:

$$\mathbf{C}^{-1} = \mathbf{L}\mathbf{D}^{-1}\mathbf{U}^{-1} = \left(\begin{array}{c|c} \mathbf{I_p} & \mathbf{0} \\ \hline -\mathbf{\Gamma}^{-1}\mathbf{G}^T & \mathbf{I_q} \end{array}\right) \left(\begin{array}{c|c} (\mathbf{B}-\mathbf{G}\mathbf{\Gamma}^{-1}\mathbf{G}^T)^{-1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{\Gamma}^{-1} \end{array}\right) \left(\begin{array}{c|c} \mathbf{I_p} & -\mathbf{G}\mathbf{\Gamma}^{-1} \\ \hline \mathbf{0} & \mathbf{I_q} \end{array}\right) \tag{1.11}$$

The inverse of $\mathbf{C}$ is now expressed in terms of two inverses of smaller matrices $(\mathbf{B}-\mathbf{G}\mathbf{\Gamma}^{-1}\mathbf{G}^T)$ and $\mathbf{\Gamma}$. $(\mathbf{B}-\mathbf{G}\mathbf{\Gamma}^{-1}\mathbf{G}^T)$ is the Schur complement $\mathbf{C}/\mathbf{B}$ of block $\mathbf{B}$ in matrix $\mathbf{C}$, and it will be denoted as $\mathbf{S}$ matrix below.

Expand 1.11 to get the inverse of C, and solve for the optimal correction in 1.8 :

$$\left(\begin{array}{c} \Delta\mathbf{p} \\ \hline \Delta\mathbf{q} \end{array}\right) = \left(\begin{array}{c|c} \mathbf{S}^{-1} & -\mathbf{S}^{-1}\mathbf{G}\mathbf{\Gamma}^{-1} \\ \hline -\mathbf{\Gamma}^{-1}\mathbf{G}^T\mathbf{S}^{-1} & \mathbf{\Gamma}^{-1}-\mathbf{\Gamma}^{-1}\mathbf{G}^T\mathbf{S}^{-1}\mathbf{G}\mathbf{\Gamma}^{-1} \end{array}\right) \left(\begin{array}{c} \mathbf{b} \\ \hline \boldsymbol{\beta} \end{array}\right) \tag{1.12}$$

This gives two set of equations, one for correction in geometry parameters (tracker alignment) and the other for correction for track parameters (track fitting):

$$\Delta\mathbf{p} = \mathbf{S}^{-1}(\mathbf{b}-\mathbf{G}\mathbf{\Gamma}^{-1}\boldsymbol{\beta}) \tag{1.13}$$

$$\Delta\mathbf{q} = (-\mathbf{\Gamma}^{-1}\mathbf{G}^T\mathbf{S}^{-1})\mathbf{b} + (\mathbf{\Gamma}^{-1}-\mathbf{\Gamma}^{-1}\mathbf{G}^T\mathbf{S}^{-1}\mathbf{G}\mathbf{\Gamma}^{-1})\boldsymbol{\beta} \tag{1.14}$$

It will be ideal to solve simultaneously for both geometry and track parameters, but it's not possible given the large the size of matrices. For tracker alignment, we are most interested in the geometry parameters $\Delta\mathbf{p}$, i.e. to solve 1.3.9. In order to do this, the first step is to build all the matrices and vectors from the measurements. An initial track fitting is required, which means we need to get an approximate solution for 1.14.

1.3.5 Single Track Fit

Starting from 1.14, we make the assumption that the residuals are independent of geometry parameters. This approximation assumption not always valid. If the starting tracker geometry has large deviation from the true tracker geometry, the track parameters calculated under this assumption will be too biased, and can lead to more biased geometry parameters calculated in next section 1.3.7. Furthermore, the entire linear least square model will fail with a poor starting geometry.

This assumption leads to a much simplified equation, with matrices $\mathbf{B}$, $\mathbf{G}$ and the vector $\mathbf{b}$ being set to zero. 1.14 becomes:

$$\Delta\mathbf{q} = \mathbf{\Gamma}^{-1}\boldsymbol{\beta} \tag{1.15}$$

It has been mentioned that the track parameters of different tracks are independent of each other, and the matrix $\mathbf{\Gamma}$ is a diagonal block matrix made of block $\mathbf{\Gamma}_j$ for each track. Therefore, we can fit an individual track, independent of other tracks. Instead of solving for $\Delta\mathbf{q}$, the track parameters $\Delta\mathbf{q}_j$ for track j will be solved.

$$\begin{aligned} &\Delta\mathbf{q}_j = \mathbf{\Gamma}_j^{-1}\boldsymbol{\beta}_j \\ &\mathbf{\Gamma}_j = \sum_i^{\text{hits}} \left(\frac{\partial f_{ij}}{\partial \mathbf{q}_j}\right)^T \left(\frac{\partial f_{ij}}{\partial \mathbf{q}_j}\right) \frac{1}{\sigma_{ij}^2} \quad \textit{and} \quad \boldsymbol{\beta}_j = \sum_i^{\text{hits}} \left(\frac{\partial f_{ij}}{\partial \mathbf{q}_j}\right)^T \frac{\varepsilon_{ij}}{\sigma_{ij}^2} \end{aligned} \tag{1.16}$$

The track parameter is updated to $\mathbf{q}_j + \Delta\mathbf{q}_j$, then $\mathbf{\Gamma}_j$ and $\boldsymbol{\beta}_j$ are also updated. The fitting process is repeated until convergence is reached ($\boldsymbol{\beta}_j$ becomes negligible).

1.3.6 Build the Matrices

After the fitting of all tracks, we can use them to construct all the matrices and vectors in 1.3.9 to solve for geometry parameters.

Using the converged covariance matrix $\mathbf{\Gamma}_j$ from each track, the full $\mathbf{\Gamma}$ matrix is formed by:

$$\mathbf{\Gamma} = \left(\begin{array}{c|c|c} \mathbf{\Gamma}_1 & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \ddots & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_j \end{array}\right) \tag{1.17}$$

The q-vector $\boldsymbol{\beta}$ is given by:

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\beta}_1 \\ \vdots \\ \boldsymbol{\beta}_j \end{pmatrix} \tag{1.18}$$

The $p \times p$ matrix $\mathbf{B}$ and the p-vector $\mathbf{b}$ of the geometry parameters in 1.8 can be formed by summing up contributions from all measurements of all tracks:

$$\mathbf{B} = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{hits}} \left(\frac{\partial f_{ij}}{\partial \mathbf{p}}\right)^T \left(\frac{\partial f_{ij}}{\partial \mathbf{p}}\right) \frac{1}{\sigma_{ij}^2} \tag{1.19}$$

$$\mathbf{b} = \sum_{j}^{\text{tracks}} \sum_{i}^{\text{measurements}} \left(\frac{\partial f_{ij}}{\partial \mathbf{p}}\right)^T \frac{\varepsilon_{ij}}{\sigma_{ij}^2} \tag{1.20}$$

For each track j, a rectangular $p \times q_j$ matrix $\mathbf{G}_j$, which correlate the track parameters with the geometry parameters can be formed by summing over all measurements of that track:

$$\mathbf{G}_j = \sum_{i}^{\text{hits}} \left(\frac{\partial f_{ij}}{\partial \mathbf{p}}\right)^T \left(\frac{\partial f_{ij}}{\partial \mathbf{q}_j}\right) \frac{1}{\sigma_{ij}^2} \tag{1.21}$$

The full correlation matrix $\mathbf{G}$ of dimension $p \times q$ is formed by joining them horizontally:

$$\mathbf{G} = (\mathbf{G}_1 \cdots \mathbf{G}_q) \tag{1.22}$$

1.3.7 The Approximate Solutions - Tracker Alignment Algorithms

Two approaches to determine the alignment(geometry) parameters are in use. One is the local approach, implemented in HipPy package, and the other one is the global approach, implemented in the MILLEPEFE II package.

1.3.7.1 Local Approach

The local approach solves the alignment problem iteratively. The correlation between geometry parameters and track parameters is ignored within a iteration, but it is taken into account when the single track fitting procedure in 1.3.6 is repeated and the alignment parameters are re-calculated for many iterations. Generally the correction step gets smaller and smaller, and the final geometry converges after 10 to 20 iterations.

There are cases when the convergence to true geometry cannot be reached within reasonable number of iterations, especially when the starting geometry is far away and the track number is small. This sometimes happens when the detector starts a new run, and the small collection of tracks available was reconstructed with a very

different old geometry. Many methods can be used to improve the performances and avoid deviating from true geometry, such as hierarchical alignment, differential alignment, adding constraints and properly mixing track collections, which will be discussed in the following sections. There's also a requirement of minimum number of hits on each detector module for it's geometry to be updates, and a cut on the final alignment uncertainly.

The matrix $\mathbf{G}$ accounts for the correlation between track parameters and geometry parameters, and it is approximated to be zero in the local approach. The normal equations for geometry parameter correction 1.3.9 is simplified to:

$$\Delta\mathbf{p} = \mathbf{B}^{-1}\mathbf{b} \tag{1.23}$$

We can further simplify the problem by solving the correction for each detector module independently, and take care of the correlation between detectors by multiple iterations.

For a single detector module, the geometry parameter is of dimension 6, as mentioned in 1.2.2:

$$\Delta\mathbf{p} = (\Delta u, \Delta v, \Delta w, \Delta\alpha, \Delta\beta, \Delta\gamma) \tag{1.24}$$

where the u-axis is along the precise coordinate of the sensor plane, the v-axis along the course coordinate, and the w-axis normal to the sensor. The angles α, β and γ are rotations around the axes u,v and w.

In stereo strip detectors and pixel detectors, each hit has two measurements, u_m and v_m, and in the non-stereo strip detectors there is only one measurement u_m.

Without loss of generality, we consider the first case.

The residual is a 2-vector for a hit on a specific detector module:

$$\boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_u \\ \\ \varepsilon_v \end{pmatrix} = \begin{pmatrix} u_m - u_f \\ \\ v_m - v_f \end{pmatrix} \tag{1.25}$$

It can be shown that the 6×2 Jacobian matrix for f_{ij}, the measurement i from track j, has the form below:

$$\bigtriangledown_{\mathbf{p}} f = \begin{pmatrix} \frac{\partial f_u}{\partial u} & \frac{\partial f_v}{\partial u} \\ \frac{\partial f_u}{\partial v} & \frac{\partial f_v}{\partial v} \\ \frac{\partial f_u}{\partial w} & \frac{\partial f_v}{\partial w} \\ \frac{\partial f_u}{\partial \alpha} & \frac{\partial f_v}{\partial \alpha} \\ \frac{\partial f_u}{\partial \beta} & \frac{\partial f_v}{\partial \beta} \\ \frac{\partial f_u}{\partial \gamma} & \frac{\partial f_v}{\partial \gamma} \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ tan\psi & tan\vartheta \\ v_x tan\psi & v_x tan\vartheta \\ u_x tan\psi & u_x tan\vartheta \\ v_x & -u_x \end{pmatrix} \tag{1.26}$$

where ψ is the angle between the track and the vw-plane, and ϑ is the angle between the track and the uw-plane.

So the matrix $\mathbf{B}$ (6×6 for a single track module) and vector $\mathbf{b}$ (2-vector) can easily be calculated following 1.19.

The iterative correction $\Delta\mathbf{p}$ can be solved by 1.23, by simply inverting a 6×6 matrix.

HipPy alignment algorithm is one of two major tracker alignment algorithms in CMS. The performance can be demonstrated by the CRUZET alignment in 2016.

0.2 million cosmic tracks with 3.6 million hits at 0T magnetic field is used. The alignment is done at high level ,aligning 6 detector units, which means there are only 6×6 geometry parameters, and the coordinates are the same as global tracker coordinate. For this alignment campaign,20 iterations are used.

The iterative correction in all parameters in each iterations is shown in Figure 1.1. It shows that the most significant movement is in z direction of BPIX, and the correction converge to small values in the end, which means the final geometry is stable. This kind of plot is referred to as "convergence plot" for HipPy.

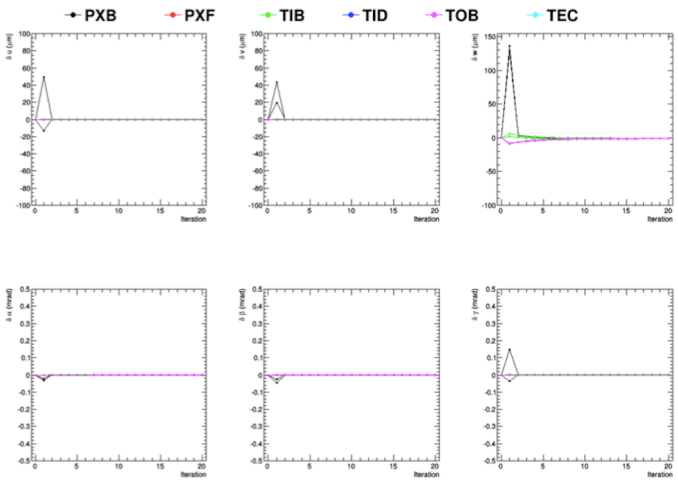


Figure 1.1: The convergence plot for 2016 CRUZET alignment, using HipPy package.

The convergence plot for a module level alignment is shown in 1.2. Alignment corrections pixel barrel detector (BPIX) is plotted, with each line correspond to one of the BPIX modules. Since v is along the course direction, the convergence is worse

than that of u direction, but it reaches the desired accuracy of $10\mu m$. The β direction is not aligned in this specific alignment.

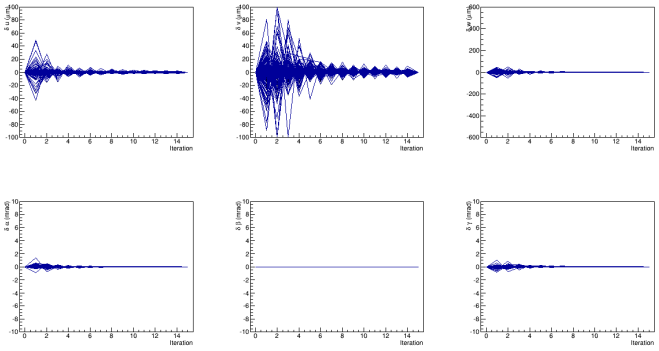


Figure 1.2: The convergence plot for module level alignment, using HipPy package.

The total correction relative to the initial geometry is shown in Figure 1.3. This is also referred to as "shift plot" for HipPy. It gives the similar information as "convergence plot", but provide a clearer view of the total movement. Here we see that the BPIX moved by about 130 μm in z direction.

The alignment error is obtained for each iteration, and the percentage error for each detector module is calculated and used to determine if the geometry of that module will be updated in that iteration. The alignment parameters with error bars is plotted for the final iteration as shown in Figure 1.4 for high level alignment.

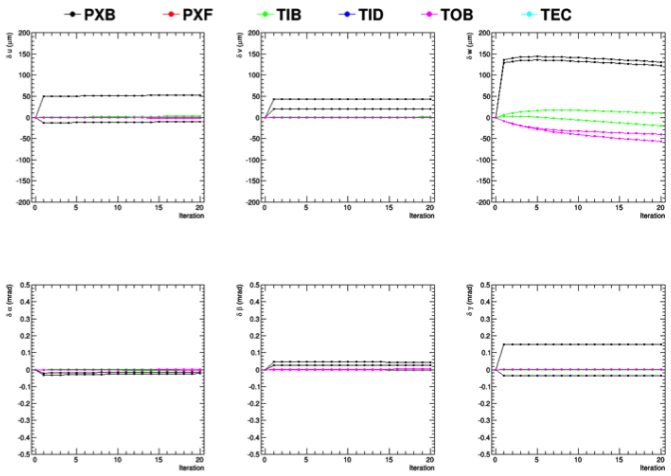


Figure 1.3: The shift plot for 2016 CRUZET alignment, using HipPy package.

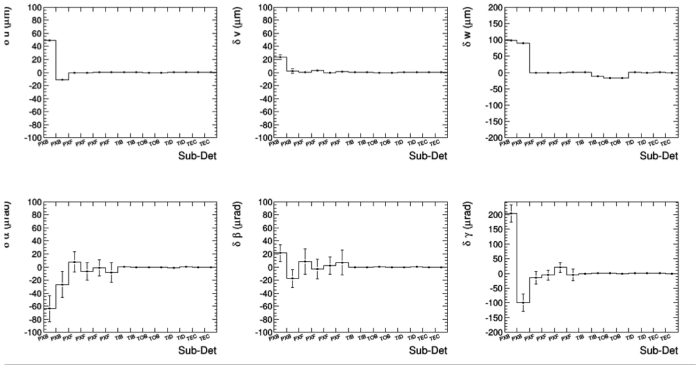


Figure 1.4: The fitted parameters with error in 2016 CRUZET alignment, using HipPy package.

1.3.7.2 Global Approach

The global approach solves the alignment problem in a single step (or a few iterations when needed). It takes into account the correlation between track parameters and geometry parameters (matrix $\mathbf{G}$), as well as the correlation between geometry parameters (full matrix $\mathbf{B}$). It is close to but not an exact solution to the normal equations 1.3.9, because it follows the same procedure of first fitting the single tracks based on the initial geometry, rather than fitting the track parameters together with the geometry parameters, so possible bias could be introduced in constructing matrices $\mathbf{B}$ and $\mathbf{G}$.

The full $\mathbf{\Gamma}$ matrix needs to be inverted in order to construct the Schur complement $\mathbf{S}$ to matrix $\mathbf{B}$, and the inverse of the Schur complement $\mathbf{S}$ itself need to be calculated in the end to determine the corrections. Both $\mathbf{\Gamma}$ and $\mathbf{S}$ are large sparse matrices given that millions of tracks are used.

1.3.8 Alignment Strategies

There are certain patterns of CMS tracker geometry that can be used to reduce the computing time and improve the alignment accuracy. Instead of solving the problem directly, two main strategies are used in routine alignments.

1.3.8.1 Hierarchical Alignment

The alignment purpose is to determine the positions of all modules of the tracker, but the alignment procedure can be performed at any hierarchy levels. i.e. different substructures of the detector. For example, the barrel pixel is made of two half barrels, which are made of three layer with each layer made of several ladders, and on each ladder, 8 modules are mounted on top. Alignment can be done at any of these "levels". To treat the translation and rotation of the substructure as a whole, the geometry parameters of each level of substructure can be defined, and the correlation between track measurements and these parameters can be obtained. This leads to a modified version of 1.26:

$$\nabla_{\mathbf{p}_h} f = \frac{\partial \mathbf{p}}{\partial \mathbf{p}_h} \cdot \nabla_{\mathbf{p}} f \tag{1.27}$$

where $\frac{\partial \mathbf{p}}{\partial \mathbf{p}_h}$ is the 6×6 Jacobian matrix relating the translation and rotation of high level structures to the movement of each module in local module coordinates.

The alignment of high level structure is very useful when the number of tracks is insufficient for determining the position of each module, or when there's a significant weak mode in module level alignment.

In the iterative local approach, higher level alignment is usually done before the full module level alignment, for faster and better convergence. In the global approach, the higher level alignment can be done simultaneous with the module level alignment, and linear equality constraints are used to eliminate redundant degrees of freedom, by

means of Lagrangian multipliers. The substructure selected for higher level alignment depends on the number of tracks available, and the specific alignment goals.

1.3.8.2 Differential Alignment

The "differential alignment" or "multi-IOV alignment" means some of the alignment parameters are treated as time-dependent, while the majority of the alignment parameters are treated as time-independent. In reality, the positions of high level structures are more sensitive to the changing environment, and the relative position of modules mounted on top remain more stable with time.

A interval of validity (IOV) is a period during which the detector environment remains stable, and we don't expect much change in the high level structure. It is sometimes determined by running preliminary alignments with tracks from each run, and compare their alignment parameters and validation results. All the tracks from a specific IOV are combined to determine the geometry of high level structure for that IOV.

In the local approach, the multi-IOV alignment is done in two steps. In the first step, the high level structure is aligned for each IOV, using only tracks from the corresponding IOV. The final aligned geometries for all IOVs are combined into a multi-IOV geometry, and is used as the starting geometry for the second step. In the second step, tracks from all IOVs are used together to determine the position of each module w.r.t. the high level structure. Each track pick the starting geometry

according to its own IOV, but the relative correction is calculated using all tracks.

This method allows to use the full statistics of the whole dataset while still taking into account the time dependence of higher structures.

1.3.9 Weak Modes

The main challenges of the alignment are the linear combinations of track parameters that leave the objective function χ^2 invariant. This will be reflected in singularity of the matrix **S** in These linear combinations, or geometry transformations, are called "weak modes", because their contribution to the final geometry can not be easily determined.

Weak mode arises when coherent change in the geometry parameters **p** can be compensated by change of the track parameters $mathbfq$. This is especially the case when the starting geometry is far from the true geometry, because the track parameters $mathbfq$ are determined prior to the alignment. Simple examples include that, the overall shift of the tracker can be compensated by the change of impact parameter of tracks, and the "layer rotation" of the tracker can be compensated by the change of curvature of the tracks. The weak modes will further lead to the bias in track parameters when the aligned geometry is used for track reconstruction, and this contribute significantly to the uncertainty in kinematic properties of the tracks.

The weak modes depend on a few factors, such as the geometry and segmentation of the detector, the geometry of tracks used, and the alignment parameters to solve.

A few typical weak modes and their solutions are discussed below.

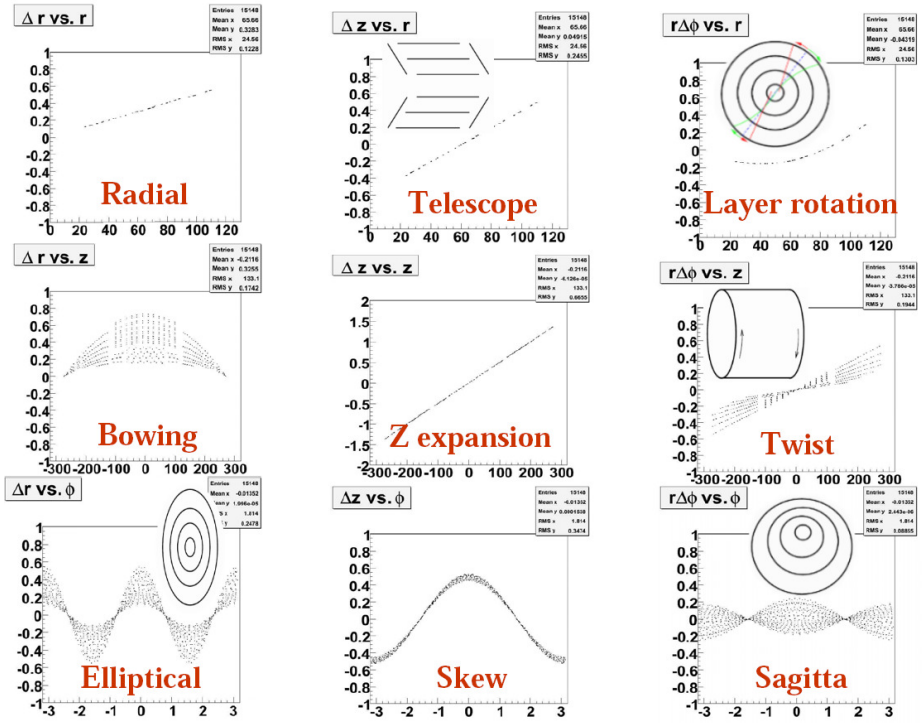


Figure 1.5: Possible weak modes in CMS tracker alignment.

1.3.9.1 Z-Expansion

Z-expansion ($\Delta z \propto z$) is the most common weak mode, and it happens when the alignment is done with collision tracks parallel to the beam line, and the z-movement of tracker modules will have small effect on the track-hit residuals. The most effective way to constrain this weak mode is to include cosmic tracks, with track parameters more sensitive to the z-movement of the detector. When there're limited the cosmic tracks, more weight need to be assigned to the cosmic tracks compared with the

collision tracks for a better constraint.

z-expansion doesn't change the transverse momenta of the reconstructed tracks. There are a few other weak modes that systematically affect the track transverse momenta. They can be categorized as change-symmetric and charge-asymmetric deformations, depending on if the change to the transverse momenta is same or opposite for the oppositely charged tracks.

1.3.9.2 charge-symmetric deformations

The systematic changes in radial direction will change the momentum of positive and negative charged tracks in the same way, so we call them charge-symmetric deformations. The examples are:

Simple radial expansion/contraction ($\Delta r \propto r$): constant Δp_T, no spatial dependence. Elliptical ($\Delta r \propto cos(2\phi + \phi_0)$): Δp_T has the same phi dependence Sagitta ($\Delta r \propto cos(\phi + \phi_0)$): Δp_T has the same phi dependence Bowing ($\Delta r \propto z$): Δp_T has the same z dependence

figure 1.6(a) shows the changes on reconstructed track curvature caused by sagitta deformation.

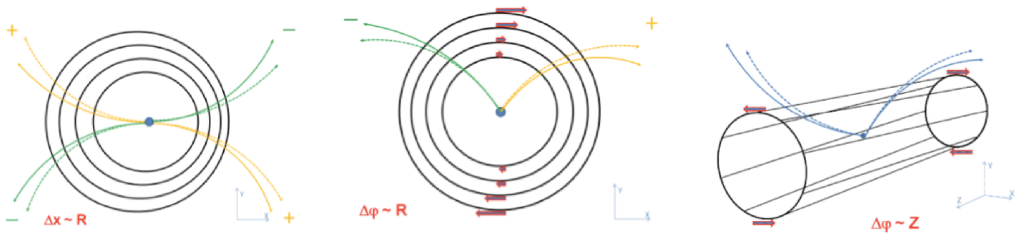
1.3.9.3 charge-asymmetric deformations

The systematic changes in ϕ direction will change the momentum of positive and negative charged tracks in opposite ways, so we call them charge-asymmetric

deformations. The examples are:

Simple layer rotation ($\Delta\phi \propto r$) : constant Δp_T, no spatial dependence. Layer rotation with phi dependence ($\Delta\phi \propto rf(\phi)$) : Δp_T has the same phi dependence Twist ($\Delta\phi \propto z$): Δp_T has the same z dependence

figure 1.6(b) and (c) show the changes on reconstructed track curvature caused by layer rotation and twist deformation.



(a) (b) (c)

Figure 1.6: The impact of weak modes on track reconstruction. The dashed lines are the true track trajectories, and the solid lines are the reconstructed tracks.

1.3.9.4 $Z\mu\mu$ validation and constraint

The information of a known resonance decaying into two charged particles can be used to detect as well as constrain certain weak modes. A common process used is the decay of Z boson into two muons, because high p_T muons are measured with high precision and efficiency by CMS.

In $Z\mu\mu$ validation, the reconstructed Z mass is compared with the theoretical

Z mass, and plotted against the track ϕ and η for both $\mu+$ and $\mu-$. Since the two muons have opposite charge, it can detect both charge-symmetric and charge-asymmetric weak modes. The ϕ and η dependence of the bias in reconstructed Z mass can also point to the types of existing weak modes.

A detailed dependence is mentioned in section 1.3.9.2 and 1.3.9.3 and plotted in the simulation study1.7

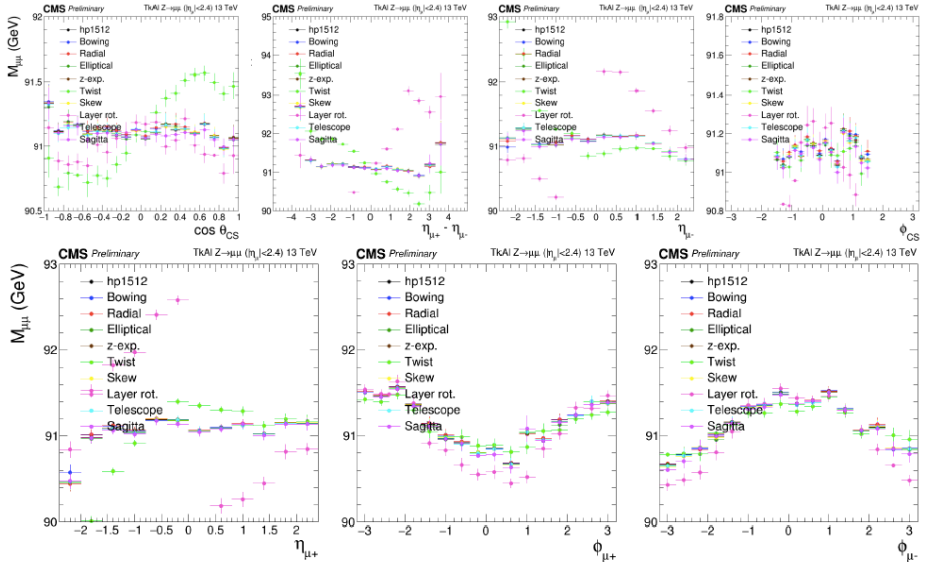


Figure 1.7: Possible weak modes in CMS tracker alignment.

The $Z\mu\mu$ constraint uses the Z mass and decay vertex as a constraint in alignment fit. The position of decay vertex,the momentum of resonance candidate,two decay angles in the rest frame of the resonance, and the mass of the resonance are used as input for alignment. It's a effective method to constrain the weak modes that can be detected by $Z\mu\mu$ validation.

1.3.10 Validation Methods

To estimate the statistical accuracy of the alignment results and to detect possible biases, many validation methods are used. In addition to $Z\mu\mu$ validation mentioned in 1.3.9.4, the track splitting, distribution of median of residuals, primary vertex and are commonly used.

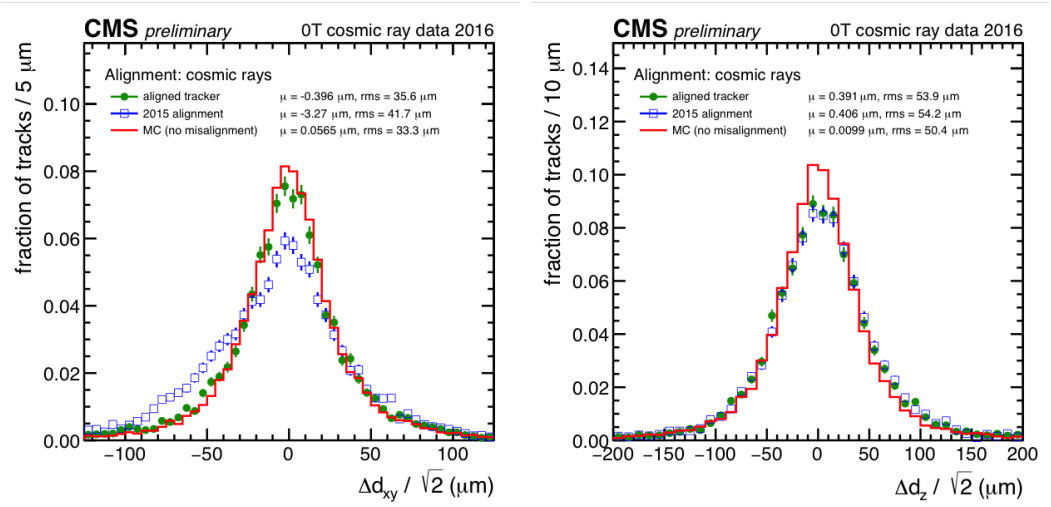
1.3.10.1 Track Splitting

In track splitting validation, cosmic tracks are split in half at the point of closest approach to beam line, then both halves are reconstructed independently and their parameters are compared at the splitting point. The normalized differences between track parameters of the two halves are histogrammed. If the relative position of the detector sub-structures used to reconstruct the two half-tracks are not determined correctly, the difference in the track parameters will have systematic deviation from zero. The width of the distribution measures the achieved alignment precision.

Figure. 1.8 shows an example of track splitting validation results.

1.3.10.2 Distribution of the Median of Residuals

Each track is refitted using the alignment constants under consideration, and the hit prediction for each module is obtained from all of the other track hits. The median of the distribution of unbiased hit residuals (DMR) is then taken for each module and is histogrammed.

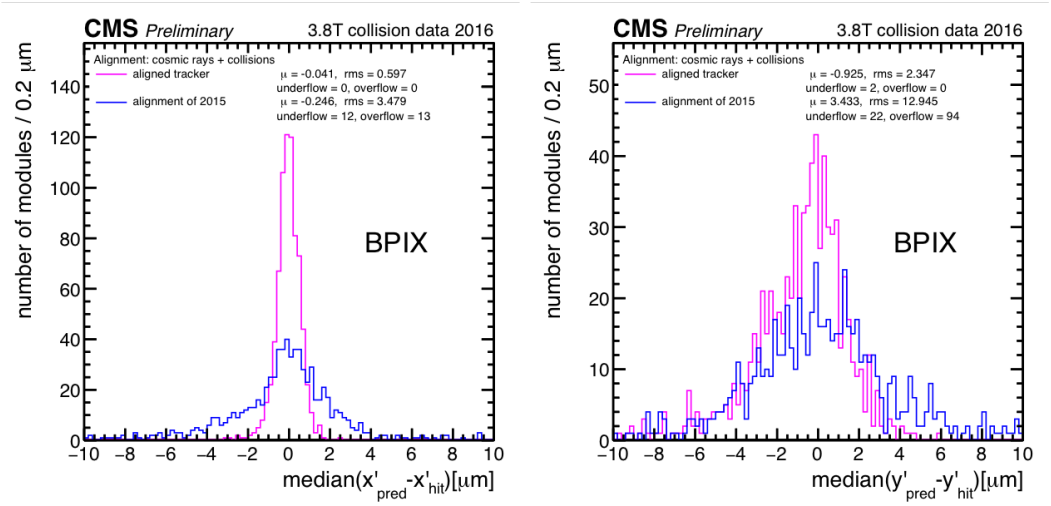


(a) (b)

Figure 1.8: The track splitting validation in impact parameters d_{xy} and d_z for 2016 CRUZET alignment. The observed precision using the aligned geometry (green circles), produced with the Millepede-II and HipPy algorithms using cosmic ray data at 0T, is a major improvement over the 2015 EOY geometry (blue empty squares).The precision comes close to that of the ideal Monte Carlo (red).

The width of this distribution of the medians of residuals is a measure of the statistical precision of alignment results; deviations from zero indicate possible biases. The width also has an intrinsic component due to the limited number of tracks, meaning that distributions can only be compared if they are produced with the same number of tracks, as is the case within each set of plots here.

Figure. 1.9 shows an example the DMR validation results.



(a) (b)

Figure 1.9: The DMR validations for the local x- and y-directions in the barrel pixel detector.The alignment shown in magenta was produced with the Millepede-II and HipPy algorithms using 3.8T cosmic ray and collision data collected in 2016. The blue line shows the starting geometry obtained at the end of 2015.

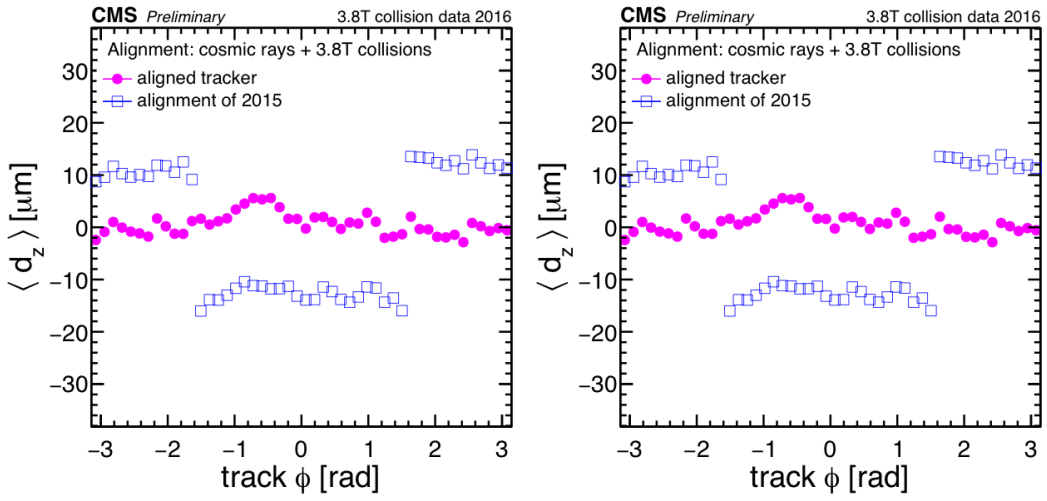
1.3.10.3 Primary Vertex

The resolution of the reconstructed vertex position is driven by the pixel detector since it is the closest detector to the interaction point and has the best hit resolution. The primary vertex residual method is based on the study the distance between the track and the vertex, the latter reconstructed without the track under scrutiny (unbiased track-vertex residual).

Events used in this analysis are selected online with minimum bias triggers. The fit of the vertex must have at least 4 degrees of freedom. For each of the vertices, the impact parameters are measured for tracks with more than 6 hits in the tracker, of which at least two are in the pixel detector, and at least one hit in the first layer of the Barrel Pixel or the first disk of the Forward Pixel, with Chi2/ndof of the track fit < 5. The vertex position is recalculated excluding the track under scrutiny. A deterministic annealing clustering algorithm is used in order to make the method robust against pileup, as in the default reconstruction sequence.

The distributions of the unbiased track-vertex residuals in the transverse plane, and in the longitudinal direction, are studied in bins of track azimuth ϕ and pseudo-rapidity η . Random misalignments of the modules affect only the resolution of the unbiased track-vertex residual, increasing the width of the distributions, but without biasing their mean. Systematic movements of the modules will bias the distributions in a way that depends on the nature and size of the misalignment and the and of the selected tracks.

Figure. 1.8 shows an example the primary vertex validation results in 2016.



(a) (b)

Figure 1.10: The primary vertex validation, with d_{xy} and d_z plotted in bins of track azimuth *phi*. The performance of a dedicated alignment (magenta) achieved with the Millepede-II and HipPy algorithms using cosmic ray and collision data at 3.8T is compared to the one of the alignment used to reprocess the collision data collected by CMS during 2015 (blue).

1.3.11 Conclusion

The track-based alignment is an effective method to align CMS tracker to a high precision close to the intrinsic silicon sensor resolution. The major computational challenge is the inversion of large matrices, due to large number of tracker geometry

parameters and track parameters. In order to reduce the size of problem, the matrix is partitioned and the alignment parameters are solved after track fitting. There are two approaches adopted by CMS alignment group. The local approach solves for each detector unit independently and reduce the matrix size to 6×6. The correlation between geometry parameters are taken into account by running multiple iterations. The global approach solves the large sparce matrix in one go, and include the correlation terms in calculation. The two approaches had shown similar performance, and the best aligned geometry is usually obtained by combining the two sequentially. Minimizing the objective χ^2 function cannot guarantee the aligned geometry to be correct, due to possible weak modes. Different form of weak modes have been discussed as well as the ways to detect and constrain them. Various validation methods are used to test the final alignment results, and to estimate the alignment precision. The sample validation plots from 2016 CRUZET and 2016 runB alignments are shown.

1.4 Summary