Alignment of the Upgraded VELO Detector at LHCb

Thomas Johnson
(201399389)

A thesis submitted in partial fulfilment of the requirements for the degree of

Master of Physics

Under the supervision of David Hutchcroft

At the
Department of Physics

April 2023

Declaration of Authorship

I, THOMAS JOHNSON, declare that this thesis titled, 'Alignment of The Upgraded VELO Detector at LHCb' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: T. john

Date: 26/04/23

UNIVERSITY OF LIVERPOOL

Abstract

Alignment of The Upgraded VELO Detector at LHCb

Thomas Johnson

This study involved the development of a method for finding present misalignments, along the global x and y co-ordinate axes, of the LHCb VELO modules. By analysing simulated event data in a Python framework, optimisation was performed using a set of known misalignments and testing was then conducted on an unknown set. The re-alignment of modules using final misalignment estimates yielded an average absolute offset of (0.00697±0.00295)mm compared to 0.00967mm beforehand. The method produced represents the first step of an iterative process that may be further developed for the complete alignment of the detector.

Contents

List of Figures

List of Tables

1 Introduction

1.1 LHCb background

The LHCb (Large Hardon Collider beauty) experiment is one of the main ongoing experiments at CERN, with the primary goal of discovering new physics by measuring CP-violating interactions of B-hadrons. Other applications of the detector include the investigation of jets and electro-weak phenomena. Between 2010 and 2018, Runs 1 and 2 of the experiment took data with centre of mass energies ($\sqrt{s}$) ranging from 7TeV to 13TeV **[1].** During the second long shutdown of the experiment, significant upgrades have been made to the LHCb detector in preparation for run 3 which commenced in July 2022 and will increase centre of mass energy to $\sqrt{s} = 14$TeV **[2].**

In around 1% of proton-proton (pp) collisions, the gluons binding the quarks together interact to produce a pair of B-hadrons, which very quickly decay. Secondary vertex lengths (decay products) are often of the order of centimetres and below, so high-precision tracking is required to properly resolve them in order to accurately measure B-hadron lifetimes (**Figure 1.1 [3]**).

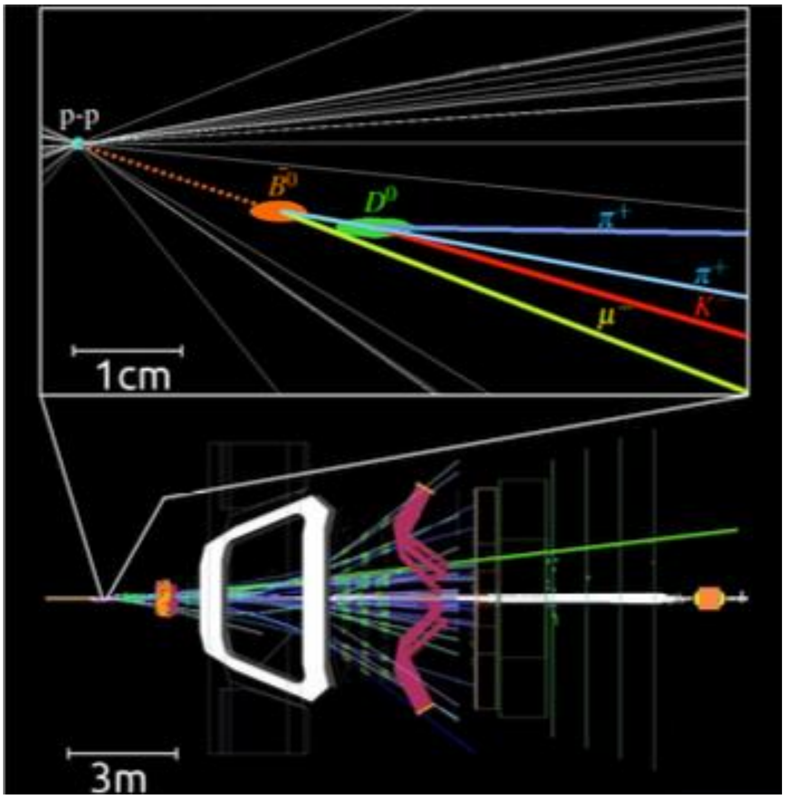


Figure 1.1 Example of a b-meson decay reconstruction at LHCb. The top half of the diagram shows the trajectories of the B-meson (dotted orange line) and its decay products: $B^0 \rightarrow D^{*+} + \tau^- + \bar{\nu_\tau}$. The very short-lived τ^- decays to a μ^- (yellow line) and the π^+ vertices (blue) and K^- arise from the following: $D^{*+} \rightarrow D^0 + \pi^+$, $D^0 \rightarrow K^- + \pi^+$. The bottom half displays the entirety of the LHCb detector from a side-on view. **[3]**

The detector itself was designed as a forward-arm spectrometer due to the fact that most B-hadron production occurs in cones directed forwards and backwards along the beam axis. In the adopted (global) right-handed co-ordinate system, the beam travels along the z-axis and the y-axis is directed upwards (**Figure 1.2 [4]**). The beam interaction point is located at z=0. The different sub-detectors of LHCb each have their own purpose, such as track reconstruction, energy measurement and particle identification.

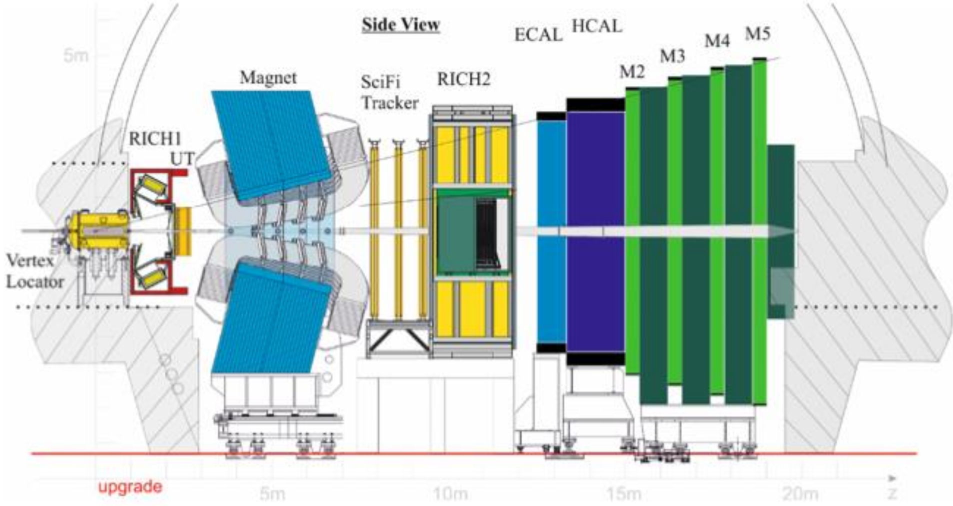


Figure 1.2 Side-view of the upgraded LHCb detector. It is approximately 20 metres in length, with a forward angular coverage from around 10mrad to 250mrad. **[4]**

Tracking sub-detectors include the VELO, UT (Upstream Tracker) and SciFi. Particle identification is performed by RICH1 and RICH2 (Cherenkov detectors) as well as ECAL and HCAL (Electromagnetic and Hadronic CALorimeters). Particle energy is also measured by ECAL and HCAL via hadronic showers. M2-M5 are muon chambers designed to identify and track muons.

1.2 The VELO Upgrade

The recently installed VELO (VErtex LOcator) upgrade is the innermost tracking detector of LHCb. It is a silicon pixel tracking detector, designed to reconstruct the primary and secondary vertices of pp collision products. The VELO modules were each constructed by the LHCb collaboration at NIKHEF and Manchester before being assembled in Liverpool and transported to CERN in 2022 for installation, in preparation for LHC run 3 (planned to span from 2022-2026).

The VELO received a significant upgrade during the second long shutdown, with a much faster readout rate (improved from 1MHz to 40MHz) and the ability to operate in closer proximity to the LHC beamline (5.1mm instead of 8.2mm). The new design makes use of semiconducting silicon pixels, rather than strips. The detector consists of 52 modules placed in halves either side of the beam (**Figure 1.3 [5]**), each holding 4 sensors (**Figure 1.4 [5]**). Each sensor is a single silicon device with a 200µm thickness, segmented into pixels with a 55 by 55µm pitch **[6]**.

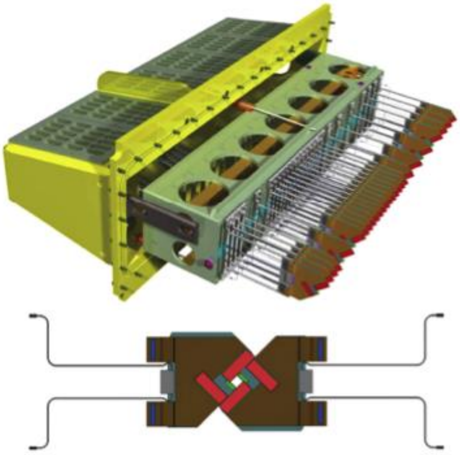


Figure 1.3 Top: One half of the VELO. Bottom: One station, consisting of left and right-sided modules. **[5]**

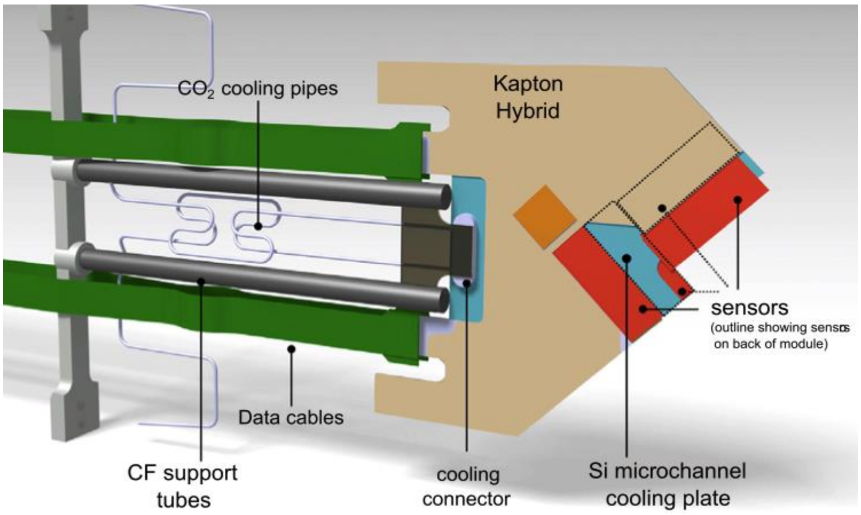


Figure 1.4 3D model of a single VELO module. The sensors themselves can be seen in red – there are two attached to both the front and back (see outline) to give four sensors in each module. **[5]**

The VELO modules cover around 1m of length along the z-axis, starting before the interaction region (where z=0) in order to detect backwards tracks (**Figure 1.5 [6]**). The modules are not distributed evenly along z as the layout was optimised, using a ray-tracing simulation, so that 99% of tracks produced within the correct region will pass through at least four stations **[6]**. For a track to be registered, it must have associated 'hits' in three or more modules.

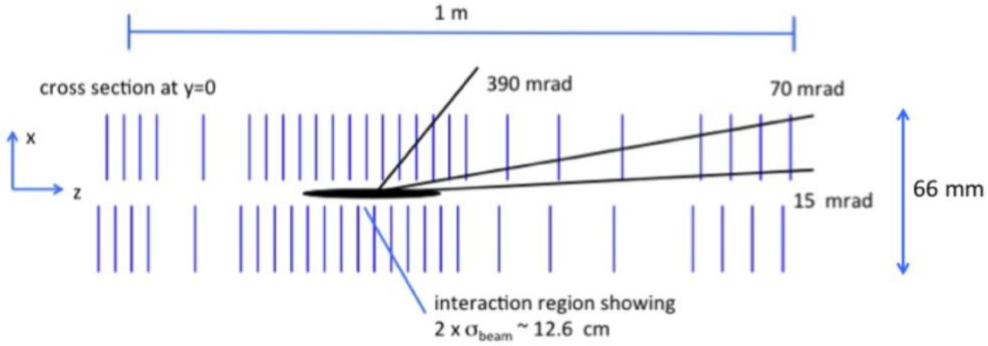


Figure 1.5 Z-layout of the VELO modules. Sensors are closest together around the beam interaction region (black). **[6]**

Two types of co-ordinate system are defined for the VELO – 'local' and 'global', both right-handed. The global system has its origin set at the beam interaction point, with z-axis directed upstream along the beamline and y-axis vertically upwards. Each VELO sensor has its own local frame defined, with the origin set at a corner and the x and y axes running along its edges. Sensors are attached to both sides of modules and at differing angles, so various rotations must be performed (in global x and y), together with translations along z to transform between local and global systems.

1.3 Monte Carlo Simulation

This project makes use of data from the Monte Carlo simulation that is used for the VELO in order to optimise the design of the detector and study its performance under different conditions. Using a somewhat simplified model of the detector's geometry (**Figure 1.6** **[6]**), pp collisions and subsequent events are simulated along with hits that would be registered by the VELO sensors. Modules can be modelled as being perfectly aligned, or with added misalignments.

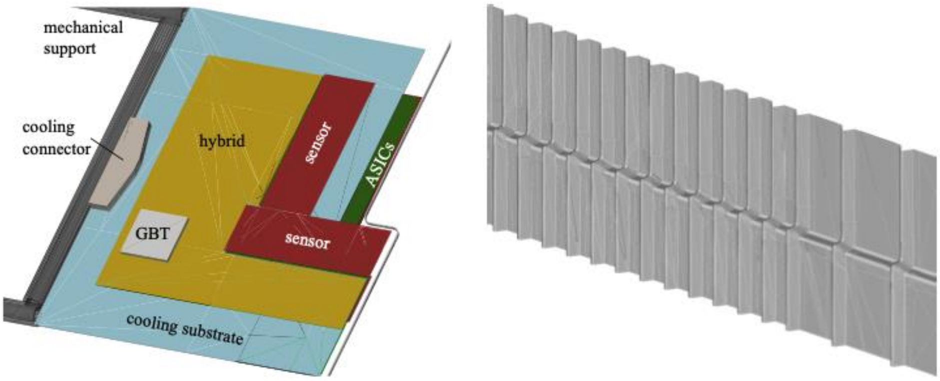


Figure 1.6 Left: A single module, as implemented by the simulation. Right: the modelled RF foil which separates the VELO from the machine vacuum. The thin cross-section of the foil can be seen on the left. **[6]**

The simulation includes the Geant4 package to model for small angle scattering that occurs as particles interact with the detector material. Sensors and the ASICs (Application Specific Integrated Circuit) between them are each modelled as 200µm thick silicon blocks **[6]**. Data available for analysis was written by the specialised Root (https://root.cern) programme suite and, for this project, was read out as a tuple in Python via the Uproot library. Variables relating to simulated events included (but were not limited to) those shown in **Table 1** below. For consistency, the variable names displayed are the same as those included in the Root files.

Name(s)	Description
Residual	The (signed) residual value for a given hit.
unbiasedResidual	The unbiased residual value for the given hit.
errResidual	The 1σ uncertainty of a given residual value.
clusX, clusY, clusZ,	A ‘cluster’ refers to a group of adjacent pixels that have registered hits from a passing particle. clusX, clusY and clusZ are the average spatial coordinates of a cluster.
node_X, node_Y, node_Z,	A ‘node’ is a point along a reconstructed biased track, corresponding to a given cluster (separated into global x, y and z components).
unbiasedNode_X, unbiasedNode_Y, unbiasedNode_Z	An ‘unbiased node’ is a point along a reconstructed unbiased track, corresponding to a given cluster (separated into global x, y and z components).
module	The module number (ranging from 0 to 51) corresponding to a given measured cluster.
sensor	The sensor number (ranging from 0 to 207) corresponding to a given measured cluster.
velonodenumber	The number of VELO sensors that have measured a given track (tracking only looks at one sensor per module).
eta	The pesudorapidity of a particle corresponding to a given track. This quantity is defined as $\eta = -\ln\left(\tan\frac{\vartheta}{2}\right)$, where θ is the polar angle of the track with respect to the beam axis (**Figure 1.7 [7]**).
p	The momentum of the measured particle in GeV.

Table 1 Names and descriptions of simulation output variables. Only those of relevance are listed.

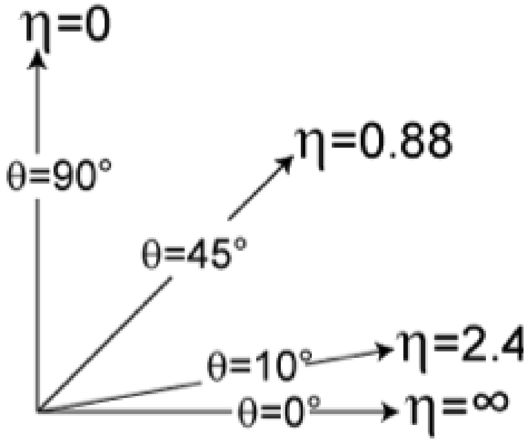


Figure 1.7 Examples of Pseudorapidity values, with the z-axis running horizontally. Tracks directed along the beam axis have $\eta = \infty$ while those running perpendicular have $\eta = 0$. **[7]**

1.4 Project Outline

As discussed earlier, for B-decay lifetimes to be measured with the desired resolution (around 20 to 200μm) **[8]**, track reconstructions must be extremely accurate (current measurements find for B_s, the lifetime $\tau = 1.445 \pm 0.016$ (stat) ± 0.008 (syst) ps) **[9]**. Therefore, the positions of sensors must be known with a high (micron-level) degree of precision. The aim of this project is to measure the effects of various misalignments on simulation data and to find the required corrections that must be applied, by estimating the present misalignments of modules.

More specifically, an attempt will first be made to develop a method of retrieving known misalignments from simulated event data. The method of alignment finding will then be optimised by performing various data cuts and studying how they affect the accuracy of misalignment predictions. These predictions may be tested by comparing estimated (which will have uncertainties) to the known values defined in the simulation. Once an optimised method has been produced, it shall be applied to simulated data where the modules have been given a new set of unknown misalignments. The success of the optimised model will then be judged by comparing the final misalignment estimates to the unknown values.

2 Track Fitting and Alignment

2.1 Track Fitting

In the context of particle physics, the purpose of track fitting is to reconstruct the trajectory of a particle in a detector so that its properties may be measured (e.g., determining charge by measuring the curvature of the trajectory in the presence of a magnetic field). Semiconducting trackers such as the VELO are compact, high-precision and offer a fast-readout, appropriate for environments such as near the collision point of the LHC. In this case, each time a particle interacts with a VELO sensor, a hit is registered wherever the interaction is measured within the material (see 'clusX' etc, **Table 1**). Once all hits for an interaction have been registered, a track-fitting algorithm may be applied to determine the best estimate for the trajectory of each particle (the process of collecting all hits from a single track before fitting is known as pattern recognition). Examples of track-fitting algorithms include least-squares regression and the Kalman filter (the method used in practice for VELO track fitting), which will each be discussed below.

Although there are minimal magnetic fields present in the region around the LHCb collision point, in practice the particles that interact with the detector do not follow perfectly straight paths due to random (Coulomb) scattering processes within the detector material (**Figure 2.1 [10]**). For a large number of small angle scatters the angular distribution becomes Gaussian, as explained by the central limit theorem. The RMS scattering angle for particles exiting a material can be described by:

$$\theta_{rms}^{plane} = \frac{13.6MeV}{\beta cp} z \sqrt{\frac{x}{X_0}} \left[1+0.038\ln\left(\frac{xz^2}{X_0\beta^2}\right)\right] \tag{1}$$

where z, βc and p are the charge number, speed and momentum of the travelling particle. x is the thickness of the material and X_0 is the radiation length, which is defined as the mean distance after which a high-energy electron will be left with a factor 1/e of its energy, due to bremsstrahlung losses.

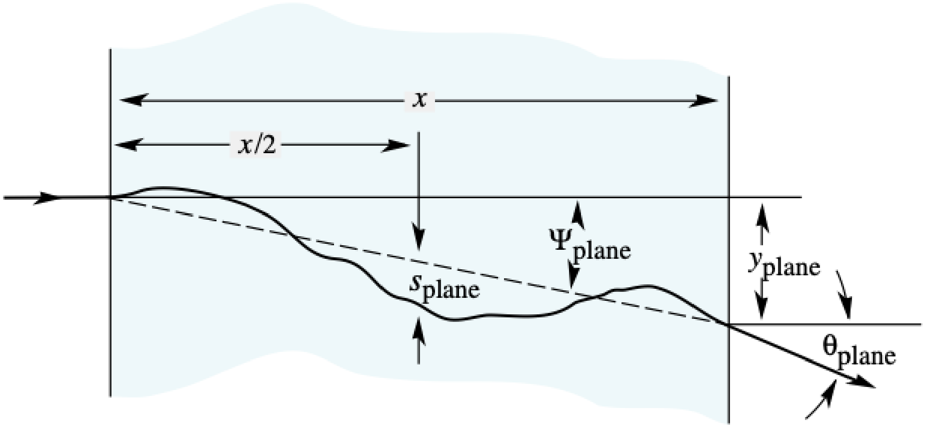


Figure 2.1 Example of a particle undergoing multiple Coulomb scattering within a material. Its path is a 'random walk' within the material, and it exits at an angle θ_{plane}. **[10]**

2.2 Comparison of Track Fitting Methods

2.2.1 Least Squares

Least-squares is a fitting method commonly used in data modelling and analysis. It simply aims to find the line (or curve) that minimises the sum of the squared differences between data points and a fitted line. For least-squares fitting in the case of the VELO, it would not be unreasonable to approximate tracks as straight lines.

If the data are represented by sets of points with co-ordinates {x_i} and {y_i} and the fitted line has equation y = a + bx, it is true that:

$$\bar{y} = \boldsymbol{A}\bar{m} \tag{2}$$

where $\bar{y}$ is the data vector that consists of values y_i, $\boldsymbol{A}$ is a matrix and $\bar{\boldsymbol{m}}$ is the vector containing model parameters **a** and **b**. The sum of squares of data misfit is therefore given by:

$$\textstyle\sum_i e_i^2 = (\bar{y} - \boldsymbol{A}\bar{m}) \cdot (\bar{y} - \boldsymbol{A}\bar{m}) \tag{3}$$

Minimising this quantity leads to the general least-squares solution for model parameters:

$$\bar{m} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}A^T\bar{y} \tag{4}$$

The least-squares method therefore offers a fast, straightforward solution for track fitting.

2.2.2 Kalman Filter

The Kalman filter is a much more complex method that may be used for track fitting – in general it uses a series of measurements made over time in order to estimate the state of a dynamical system, accounting for both process and measurement uncertainties (or noise). This is achieved via a cycle of predictions and corrections, shown in **Figure 2.2.** As the underlying equations are beyond the scope of this paper, a high-level description of the algorithm is instead given.

For VELO track fitting the Kalman filter repeatedly uses measurements made by each sensor to update an estimate of a particle's position and momentum. In the prediction step, the previously measured position and estimated momentum (or initial estimates at the start) are used to predict where the particle will pass through the current sensor. Here process noise is accounted for i.e., the random scattering angle. The 'Kalman gain' is computed in the update step, which is a weighted combination of the prediction and current measurement (the measurement weight relates to measurement noise i.e., detector resolution). The current estimate is then updated as a linear combination of the predicted state and the Kalman gain. Error covariances are also predicted and updated during these steps in order to continually quantify the uncertainty of the estimates and measurements.

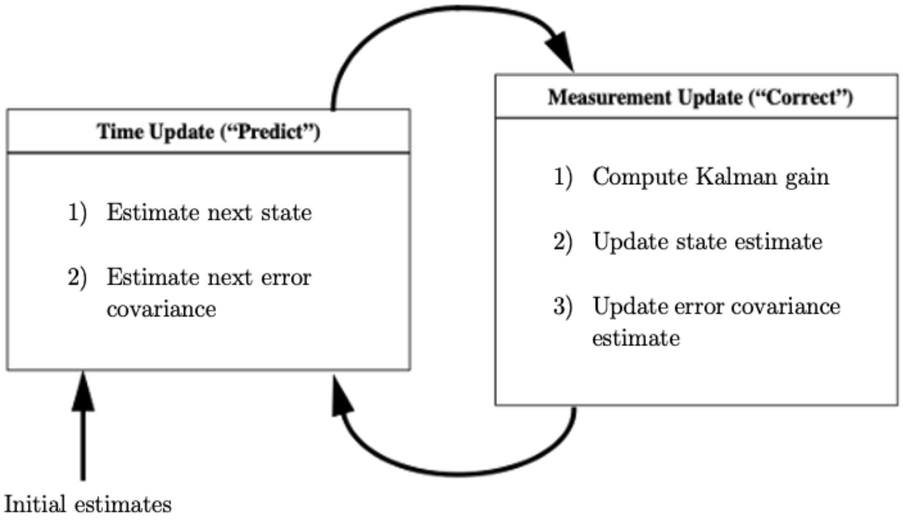


Figure 2.2 A simple diagram showing the Kalman filter 'predict-correct' cycle, with numbered operations.

By accounting for uncertainties in both the system and measurements, the Kalman filter is generally able to produce more accurate fitted tracks. The trade-off, however, is that it is far more computationally demanding than the least-squares method.

2.3 Residuals

After a track has been fitted to the hits in a detector, a residual is defined as the signed difference between a measured hit and the track. For the purpose of alignment, it is useful to separate a residual value into x and y components. Residuals may be classified as biased or unbiased. The distinction between the two is that for a biased residual, the hit for which it is being calculated is considered during track fitting (along with all other hits). The opposite is true for unbiased – all hits are considered in track fitting apart from that which corresponds to the residual of interest (**Figure 2.3 [11]).**

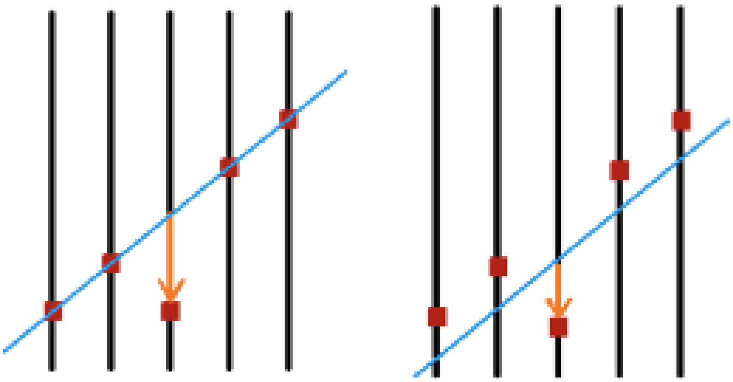


Figure 2.3 Two examples of measured hits (red) across five sensors with fitted tracks (blue). Orange arrows mark unbiased (left) and biased (right) residuals. **[11]**

2.4 Alignment

The way a residual relates to the alignment of a detector is conceptually straightforward – a misalignment of a detector along some axis will tend to induce residuals along the same axis, of equal magnitude and opposite in direction to the misalignment. This occurs because during track fitting, if misalignments are not accounted for in the design of the detector, hits in misaligned sensors will 'drag' fitted tracks out of position (**Figure 2.4 [11]**). The use of histograms to study the distribution of residuals for a given sensor can therefore provide insight into its misalignment. For a perfectly aligned detector with finite resolution, the residuals are normally distributed with a mean (μ) of zero and standard deviation (σ) related to the detector resolution. For a misaligned detector, the mean of such a histogram with n entries equals the magnitude of the misalignment in a given direction and has an uncertainty given by $\sigma/\sqrt{n}$ (standard error of the mean).

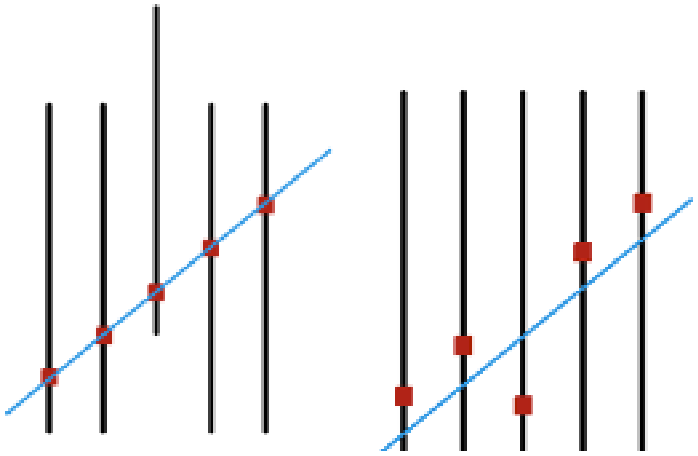


Figure 2.4 Left: the 'real' position of a track with one sensor vertically misaligned. Right: the track as seen by the reconstruction when no misalignment is assumed; the misalignment induces a residual. **[11]**

Once misalignments have been calculated, the alignments are not applied by physically moving the detector, but by accounting for them during data analysis.

3 Methodology

The following sections will discuss the methods used in order to develop an initial method of estimating module misalignments. Root files were used containing reconstructed tracks for a simulated data sample where the detector had a 4mm opening, with around 120,000 tracks present in total.

3.1 Co-ordinate Transformations

Before making misalignment predictions, a method was first developed for transforming between local and global co-ordinates for a given sensor. As the whole LHCb detector uses the same global frame but individual sensors have their own local origins defined at their bottom left corners, it was important to understand how these related to each other when considering the placement of detector pixels in Cartesian space. Although explicitly transforming between frames was not ultimately needed for the misalignment estimation process (as many variables in the tuple already had versions relating to both co-ordinate systems), this step proved an important part of the project as it provided a deeper understanding of the detector geometry. This understanding proved especially useful during parts of the project where troubleshooting was required.

A transformation between frames may be represented by a 2-dimensional rotation matrix (5) plus a possible sign-flip along z, depending on whether a sensor is attached to the front or back of a module.

$$R_\theta = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \qquad (5)$$

Due to the pattern of sensor orientations, necessary rotation angles were ±45° and ±135°. If transforming a set of points rather than vectors, sensor-dependent ‘offset’ values must also be added to account for where the sensors are stationed (e.g., how far along the beam axis and which side). By plotting both ‘clus’ and ‘clusLocal’ variables in x and y, corresponding hits in different co-ordinate systems could be compared:

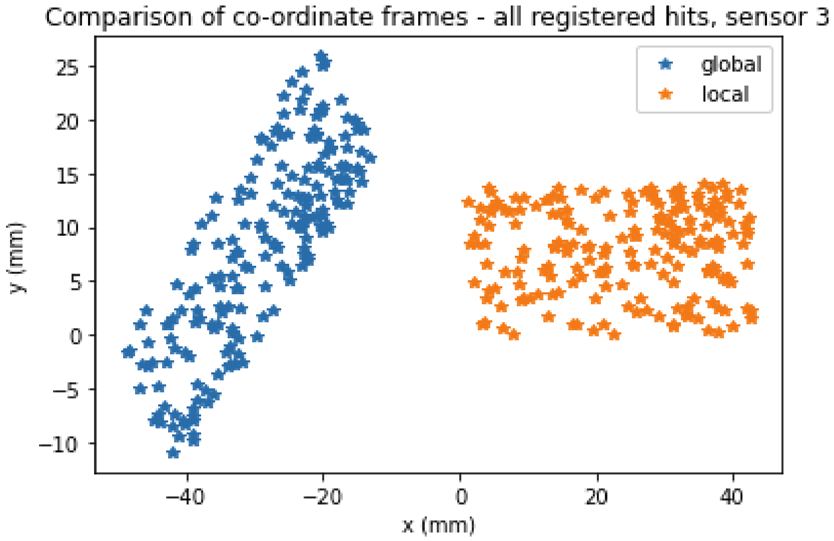
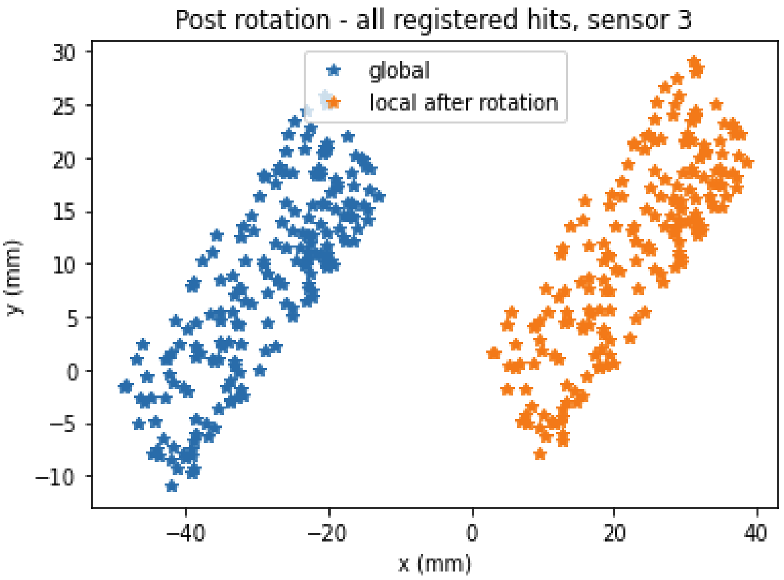


Figure 3.1 Total registered hits in sensor 3, shown in both global (blue) and local (orange) co-ordinate systems.

Transformations between frames could then be verified by reproducing the plots, after a transformation has been applied to a set of points:



```
X offset: 51.6636, Y offset: 3.1113, Z offset: 288.141
```

Figure 3.2 Total registered hits in sensor 3. Blue points again represent hits in the global co-ordinate system, and orange points represent hits after applying the appropriate rotation to points in the local system. Labelled offset values represent the difference between corresponding points.

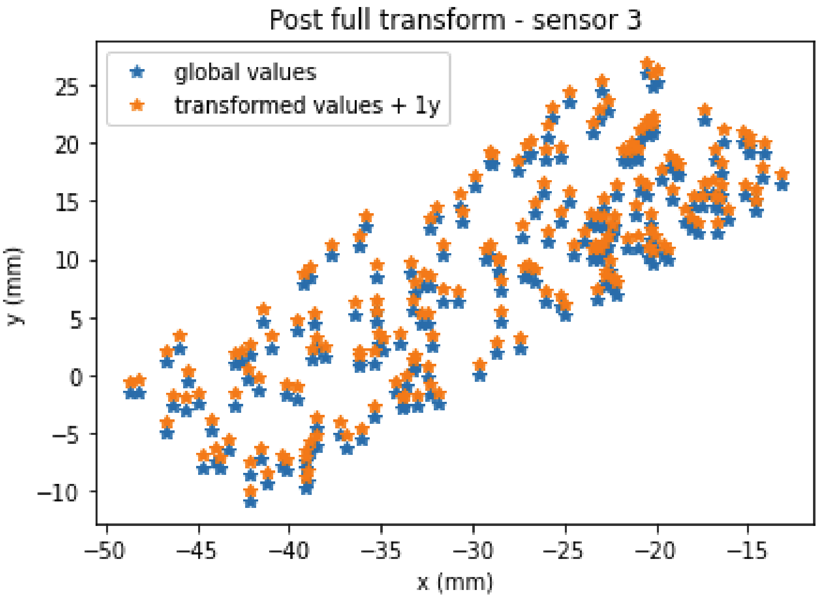


Figure 3.3 Again, total registered hits in sensor 3 are shown. Blue points again represent hits in the global co-ordinate system, and orange points now show hits after a full transformation from local to global co-ordinates has been applied (a rotation followed by subtracting the offset values shown in **Figure 3.2**). Orange points are plotted 1mm higher in y than their actual positions so that the blue points are not covered.

3.2 Plotting Tracks and Residuals

This section will provide simple examples of how residuals were visualised, as well as the locations of hits for a given event. An 'ideal' detector will first be considered (one with no misalignments).

3.2.1 Plotting Tracks

Figure 3.4 shows how the measured hits for a given track may be visualised. In the example given there are three measured hits, two plotted points corresponding to each one in order to show the residuals along x and y.

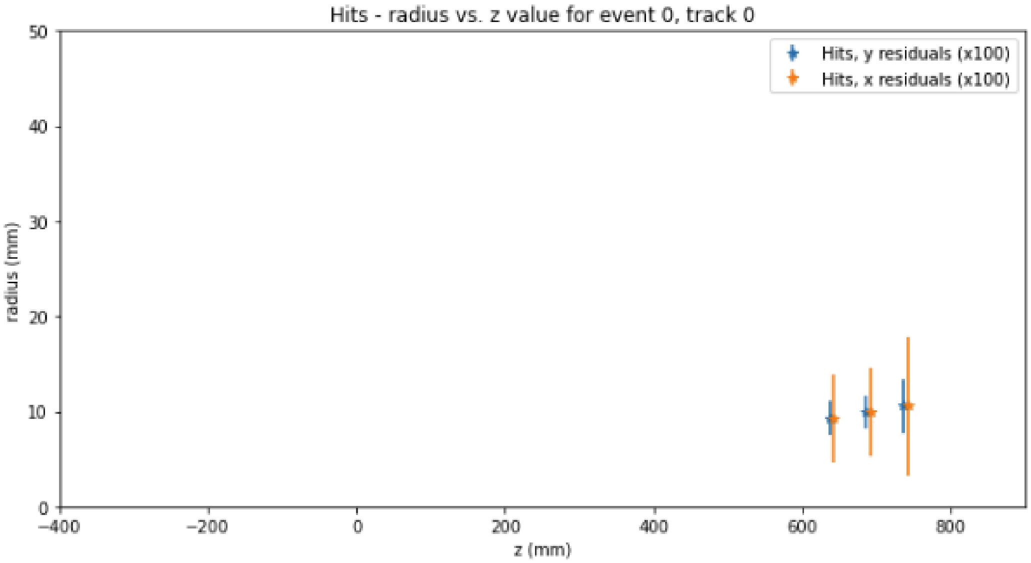
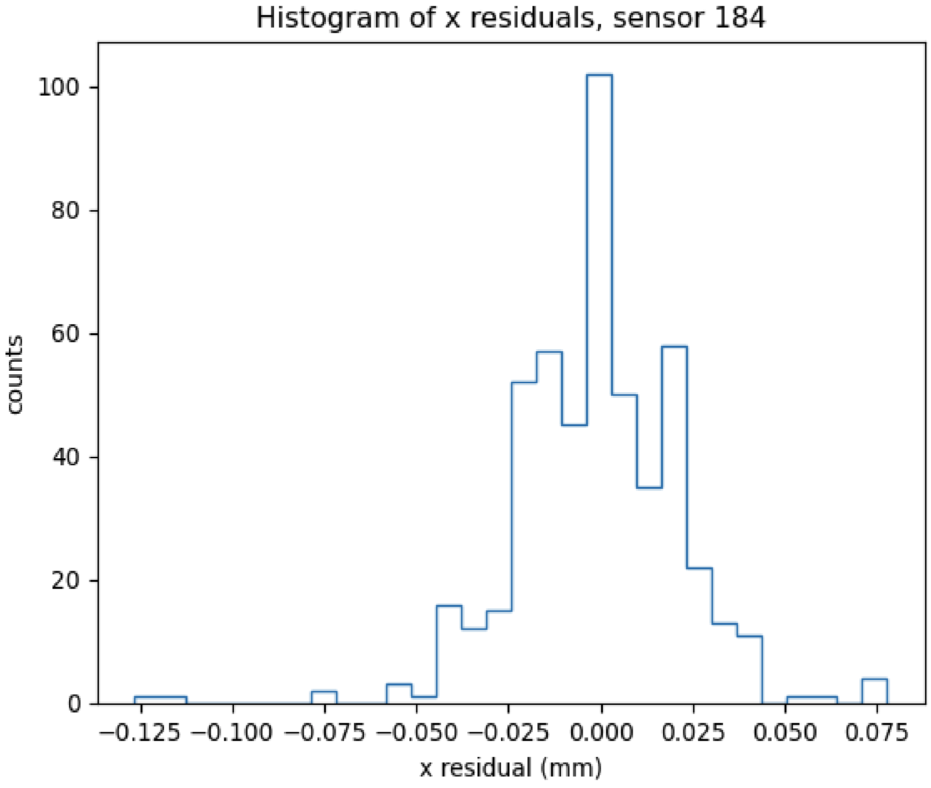


Figure 3.4 An example of the locations of measured hits for a given event. Radius (the perpendicular distance from the z axis) is plotted against z. There are three hits, with x (orange) and y (blue) residuals shown by error bars (exaggerated by a factor of 100). Pairs of points are slightly separated in z for readability. As expected, the radial positions of hits increase in magnitude for sensors further from the interaction point.

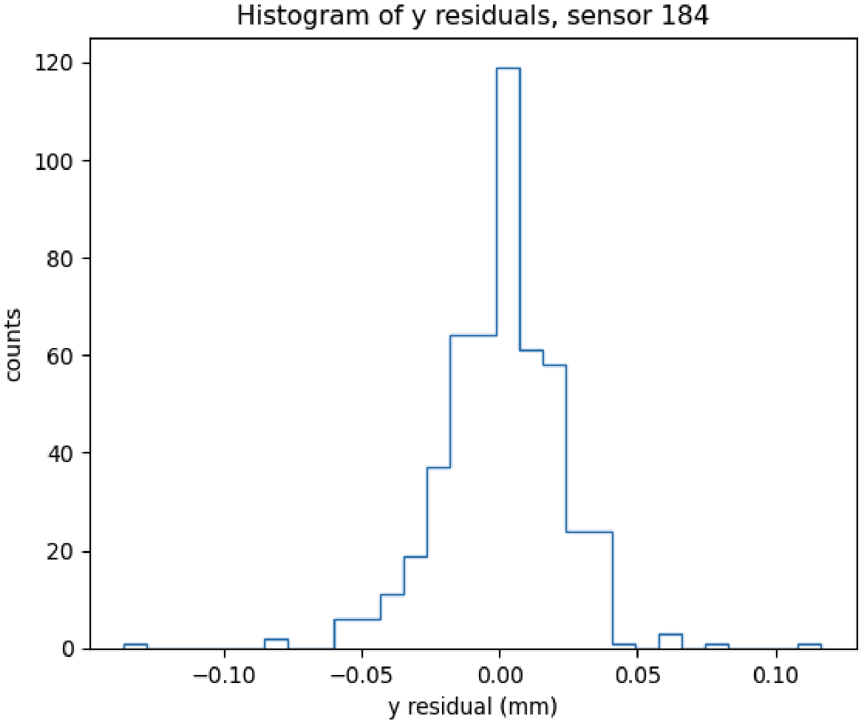
3.2.2 Residuals - Perfectly Aligned Detector

As mentioned earlier, for a perfectly aligned sensor the residuals should form a normal distribution with a mean of zero and standard deviation related to the detector resolution. **Figure 3.2** and **Figure 3.3** provide examples of such histograms for residuals along (global) x and y for a data sample where all modules are simulated as being perfectly aligned. As these axes are orthogonal, any measurements made along them are independent so residuals relating to both must be considered. Sensor 184 has been arbitrarily chosen – similar plots may be produced for all sensors.



```
N = 502
Mean = −0.000865
Std. dev = 0.022063
Std. error of mean = 0.000985
```

Figure 3.5 The simulated distribution of residuals (components along global x) for sensor 184, for a perfectly aligned detector. The mean is approximately zero, consistent with its standard error.

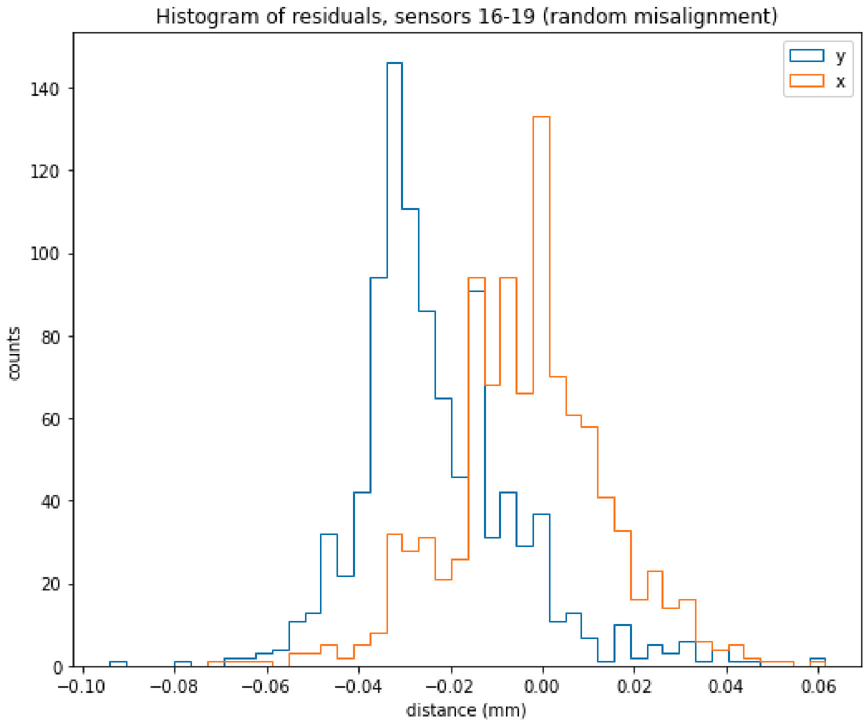


```
N = 502
Mean = 0.000171
Std. dev = 0.022182
Std. error of mean = 0.00099
```

Figure 3.6 The simulated distribution of residuals (components along global y) for sensor 184, for a perfectly aligned detector. The mean is approximately zero, consistent with its standard error.

3.2.3 Residuals - Misaligned Detector

A new Root file including event data with random (known) misalignments added to each module in x and y was then loaded. These values are given in **Table A1** of Appendix A. Using the global co-ordinate frame, it is possible to ‘add up’ histograms of sensors belonging to the same module. Since it is the modules that can be misaligned, rather than the sensors themselves, sensors belonging to the same module will have the same misalignments in the global frame (the same is not true in local frames as sensors are oriented differently from one another). Combining data in this way allows for less statistical uncertainty in the misalignment estimate due to a greater number of events. An example of such a histogram is shown below in **Figure 3.7**:



```
N = 978.0000
Even residual mean -0.0233 +-  0.0006 mm
Odd residual mean -0.0029  +-  0.0006 mm
```

Figure 3.7 The simulated distribution of residuals for sensors 16-19 (module 5), where the modules are randomly misaligned in x and y. “Even” refers to y, while “Odd” refers to x.

3.3 Retrieving Known Random Misalignments

After repeating the above method of analysing residuals for all sensors, predicted misalignments for each module were obtained (separately for x and y), along with corresponding uncertainties. Since in this case the random misalignments were known, it was useful to produce graphics (**Figure 3.8**) comparing the predicted values to the real ones. To provide more insight into the accuracy of this initial set of results, predicted misalignments and their uncertainties are shown for each sensor rather than per module. An equivalent plot to the one below, for predictions along y, is given in Appendix A (**Figure A1**).

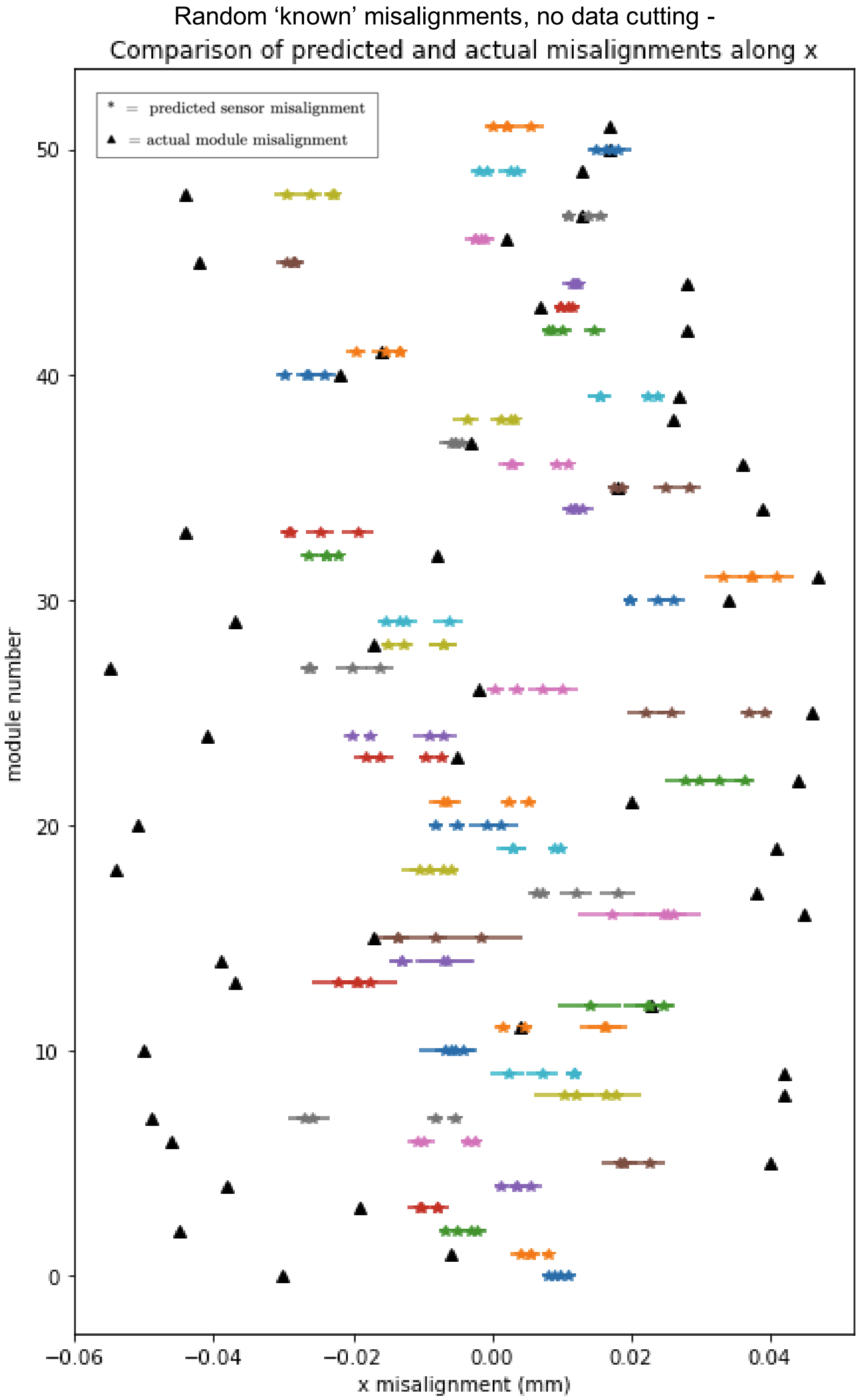


Figure 3.8 Comparison of predicted sensor misalignments (stars) to actual module misalignments (triangles). Sensors corresponding to a given module are marked in the same colour and modules are separated vertically, ranging from 0 (bottom) to 51 (top).

4 Optimisation

4.1 Chi-Squared

To properly measure the accuracy of misalignment predictions, χ^2 statistics were calculated, separately for misalignment estimates along the global x and y axes. The χ^2 statistic accounts for the different uncertainties of each misalignment estimate by including weights:

$$\chi^2 = \sum_i \frac{(T_i - P_i)^2}{\sigma_i^2} \tag{6}$$

where T_i is a true misalignment, P_i is an estimated value and σ_i is the uncertainty of the estimate. It is common to produce a reduced χ^2 value, obtained by dividing χ^2 by the number of degrees of freedom. Comparing this value to 1 then shows how well the predictions fit the data, given the errors present (values significantly larger than 1 indicate that the model fits the data poorly). In this case however, although χ^2 itself is well defined, it was difficult to determine the number of degrees of freedom present. This issue arose due to the fact that measurements across the 52 modules and their errors were correlated, while the χ^2 statistic assumes the opposite. For this analysis the decision was made to divide χ^2 values by 104 because, per co-ordinate axis, there was one misalignment estimate and one uncertainty value for each of 52 modules. This had no real effect on any results gathered, as the focus was simply on minimising χ^2, but resulted in more 'readable' values (around the order of 10^2 rather than 10^5).

4.2 Data Cutting

With the aim of improving misalignment estimates, various cuts were applied to simulated data prior to the estimation process. The ‘correct’ data cutting has the effect of removing poorly fitted tracks, which may correlate with certain measured variables. Appropriate variables for cutting were first identified, before varying the ranges over which the cuts were performed to find which values best minimised χ^2. An example of a cut is to discard all tracks with momentum below 10GeV.

χ^2 would ideally be minimised by simply making use of an optimisation function, such as the SciPy library's “optimise.minimise”, where a function has been defined to calculate χ^2 given cutting values for each variable of interest. This method was first considered and attempted but proved ineffective due to the fact that optimisation methods are designed for continuous functions. The calculated χ^2 value, however, does not vary completely smoothly for changes in cutting values due to the discrete nature of the data.

The approach was therefore taken to first produce a scatter plot for each variable to visualise changes in χ^2 for discrete changes in cutting bounds, which would hopefully show minima. The cuts were then ‘combined’ in groups of two (e.g., simultaneously varying the lower bounds for p and velonodenumber) to give contour plots for χ^2, again to look for minima. Cutting ranges to plot over were selected by examining the plots shown in **Figure 4.2**, **Figure 4.3** and **Figure 4.4**, along with their y-dimensional counterparts.

4.2.1 Identifying Cutting Candidates

To identify appropriate variables for cutting, correlations were looked for between each variable and the absolute difference between all residual values and their corresponding known misalignments (effectively the variable vs fitted track resolution). Plotting these parameters against each other for different variables allowed any correlations present to be viewed – variables for which cutting would provide no benefit showed no correlation.

For example, it was expected that cutting on the azimuthal angle ϕ (not included in **Table 1**) would be pointless since the production of particles at LHCb, and the resolution of VELO sensors, have no ϕ-dependence. **Figure 4.1** confirms this. For variables where such a correlation is present, the right cutting would then act to remove points that are further to the right on the described plots. Using variable names as given in **Table 1**, those identified as appropriate cutting candidates include p (**Figure 4.2**), eta (**Figure 4.3**) and velonodenumber (**Figure 4.4**). Appendix A includes figures equivalent to those shown below, for y rather than x (**Figure A2** – **Figure A5**).

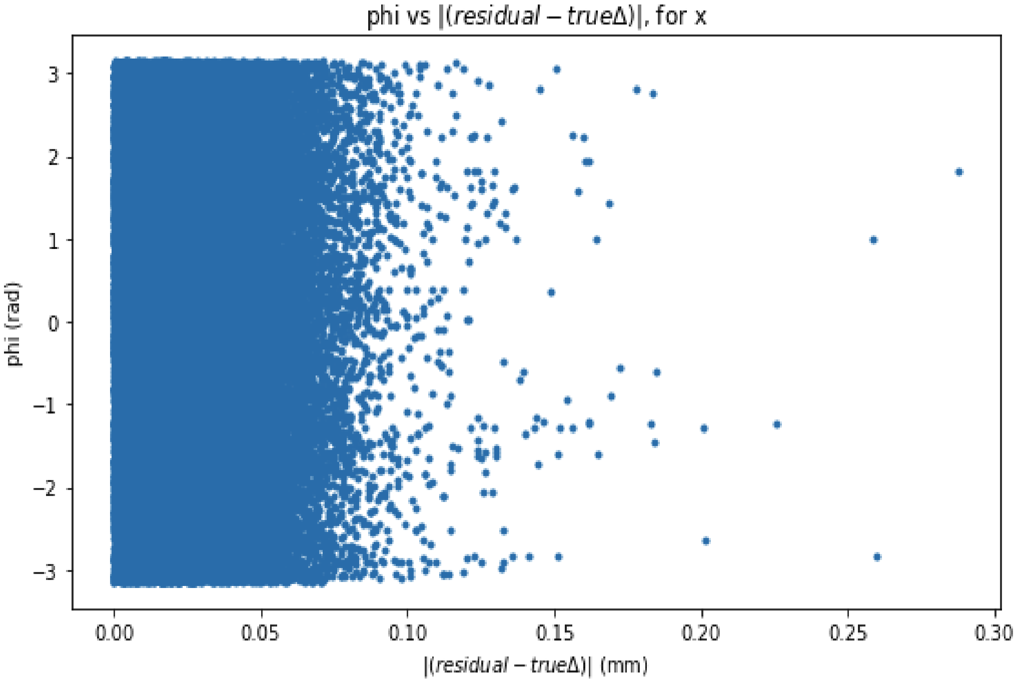


Figure 4.1 Comparison between track resolution and ϕ (for residuals along x). There is no correlation between the two so, combining this with a similar result for y, it is not possible that cutting the data on ϕ would remove any of the more poorly resolved tracks.

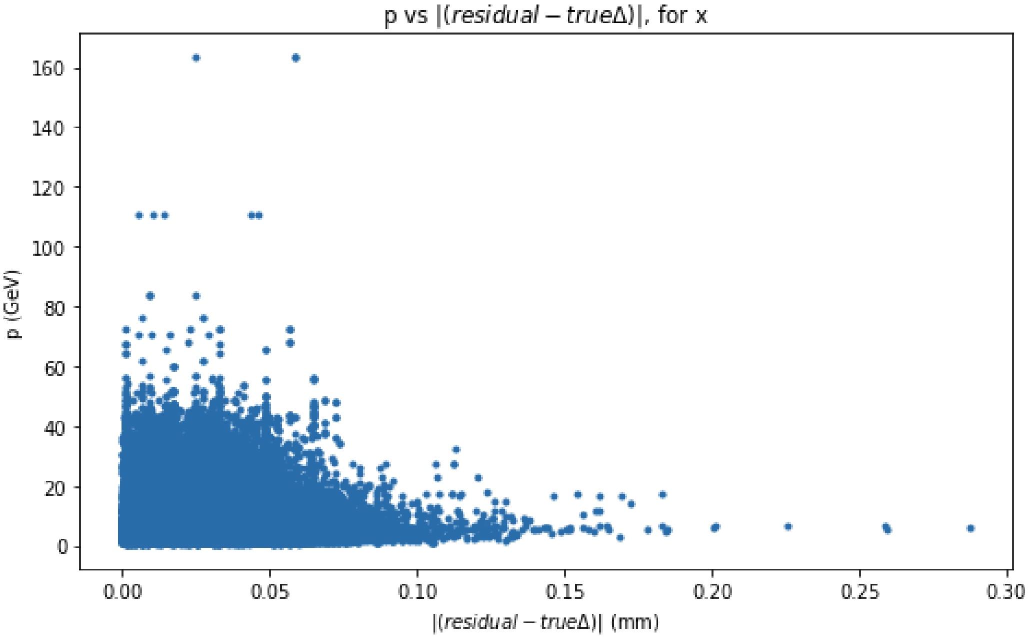


Figure 4.2 Comparison between track resolution and momentum, for x. There is some correlation between the two - cutting away tracks below approximately 20GeV would remove those that are most poorly resolved (right hand side). This was expected as higher momentum particles are less likely to exit a material at a large scattering angle (see equation 1), so straight-line track fitting provides a more accurate approximation.

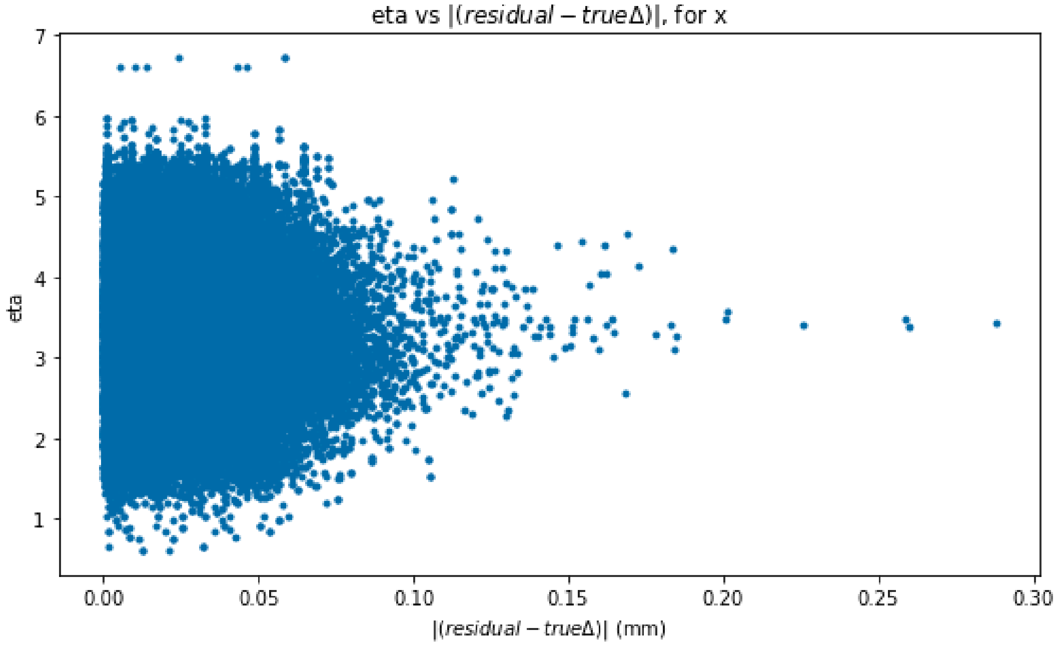


Figure 4.3 Comparison between track resolution and pseudorapidity, for x. Better resolved tracks appear to have relatively low and high eta values.

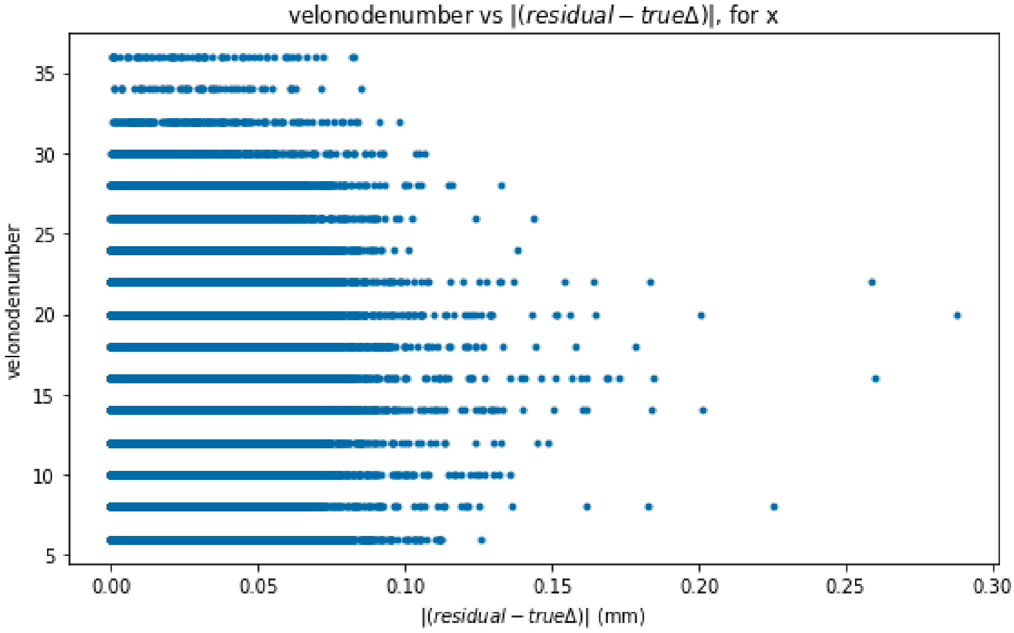


Figure 4.4 Comparison between track resolution and velonodenumber, for x. Here, more poorly resolved tracks appear more for lower values of velonodenumber. Horizontal lines appear because the variable only takes even-integer values, due to the fact that x and y are treated as separate variables (and therefore separate nodes).

4.2.2 Optimising Cutting Bounds – Scatter Plots

This section displays the one-dimensional plots that were produced to understand how cutting the selected variables, one at a time, affected χ^2. As mentioned earlier these plots would ideally each shown a minimum value, after which χ^2 begins to increase again, but the issue arose that cutting the data down too harshly resulted in entire modules with zero or very few registered hits. These plots are therefore not able to show exactly where the minimum occurs. Similarly, single sensors with only a few hits registered tended to have poorly defined mean residual values, which resulted in them disproportionately affecting χ^2. To solve this problem the decision was made to exclude single sensors from the calculations if they had below four hits registered.

Figure 4.5 shows how χ^2 decreases as the lower cutting bound for momentum increases – here the lower cutting bound means that anything below the value is rejected.

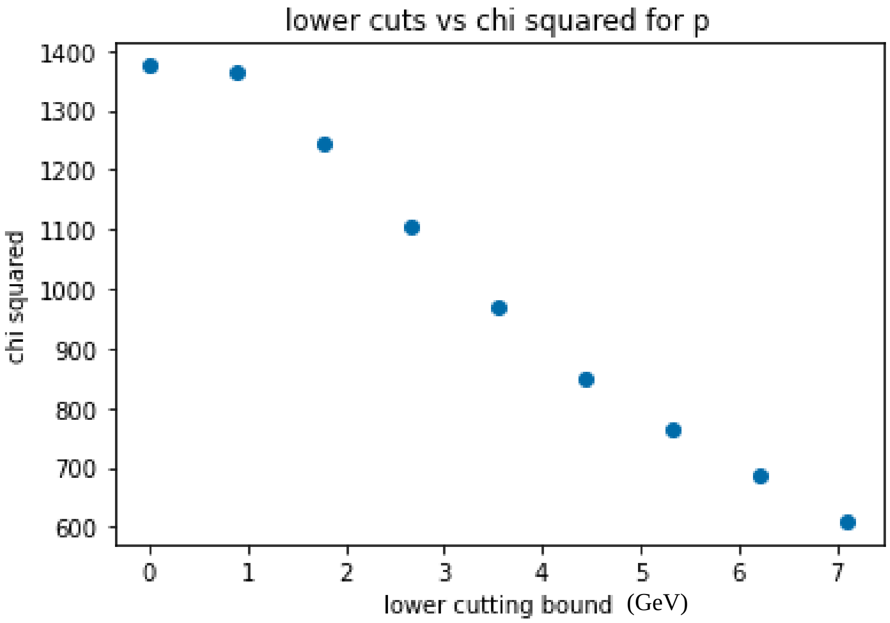


Figure 4.5 χ^2 vs lower cutting bound for track momentum. χ^2 improves as the bound increases, but it cannot be brought above approx. 7GeV as empty modules begin to appear.

From **Figure 4.3** it was concluded that high and low-eta tracks are better resolved so any cutting should remove those in the middle of the distribution. Since the variable is distributed symmetrically, two bounds could be defined either side of the mean μ with positions $\mu + a$ and $\mu - a$, where a is a number that can be varied. (**Figure 4.6**). Rejecting all tracks within the bounds would then leave the better resolved high and low-eta tracks remaining.

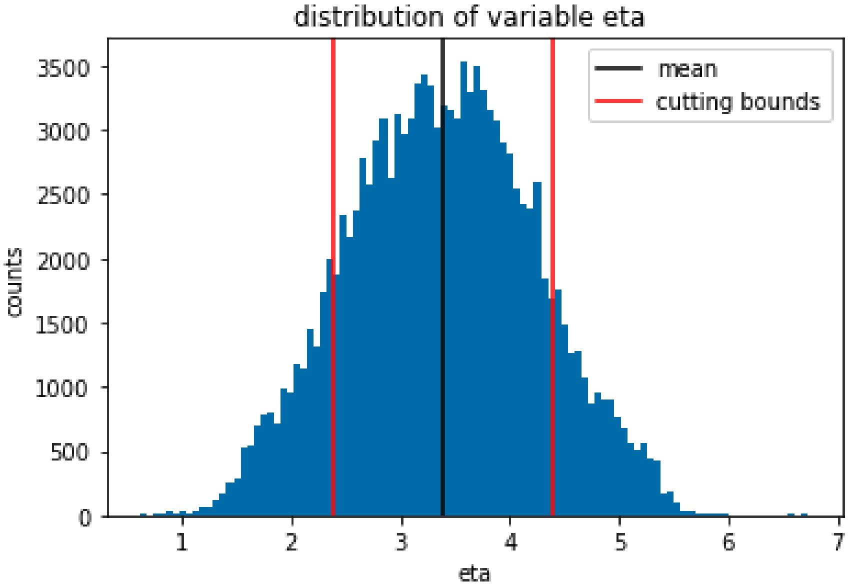


Figure 4.6 Distribution of the variable eta. It is symmetrical so the same value a can be used to define upper and lower bounds $\mu \pm a$ (red) with respect to the mean μ (black). Tracks with eta within these bounds are then rejected. Here a is arbitrarily set to 1.

The effect on χ^2 of varying a from 0 to around 1 is shown in **Figure 4.7**. Again, the appearance of empty modules limits how harshly the cutting may be taken (and so the size of the parameter a).

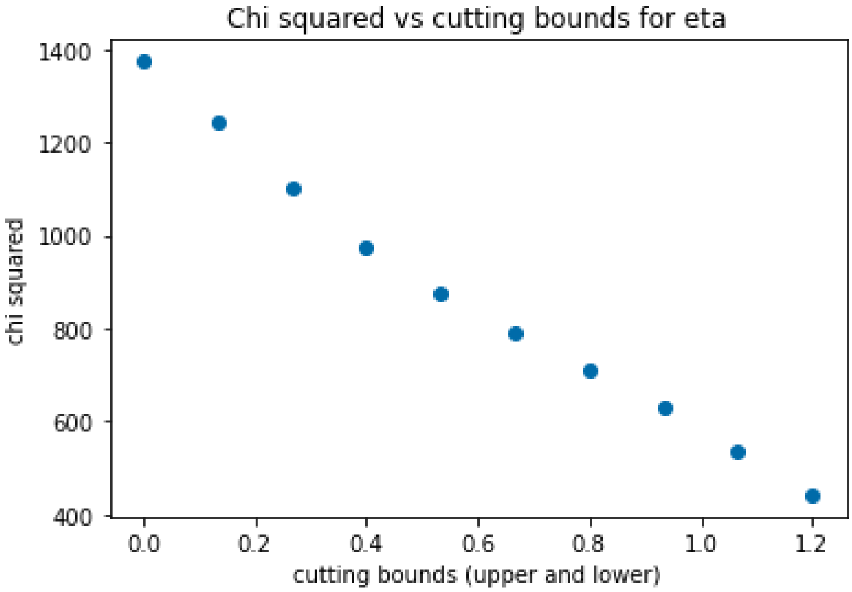


Figure 4.7 Effect on χ^2 of varying the defined cutting parameter a from 0 to 1.2, for variable eta. χ^2 improves as more of the tracks with mid-ranged eta values are discarded.

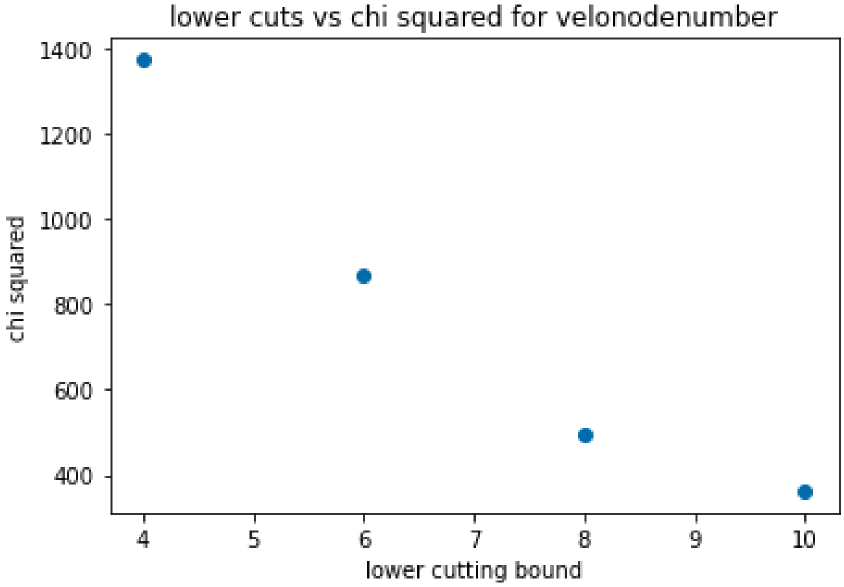


Figure 4.8 Effect on χ^2 of varying the lower cutting bound of velonodenumber from 4 to 10 (tracks with values below this threshold are discarded). χ^2 improves as this minimum bound is increased, as longer tracks are better resolved.

4.2.3 Optimising Cutting Bounds – Contour Plots

Figure 4.9 to **Figure 4.11** below are contour plots to show how χ^2 changes as the above cutting parameters are varied two at a time. The goal was to find regions on these plots that both minimised χ^2 and corresponded to the least 'harsh' cuts possible. This would ensure that as few tracks as possible were rejected when all three sets of cuts were finally combined, to again avoid empty sensors or modules.

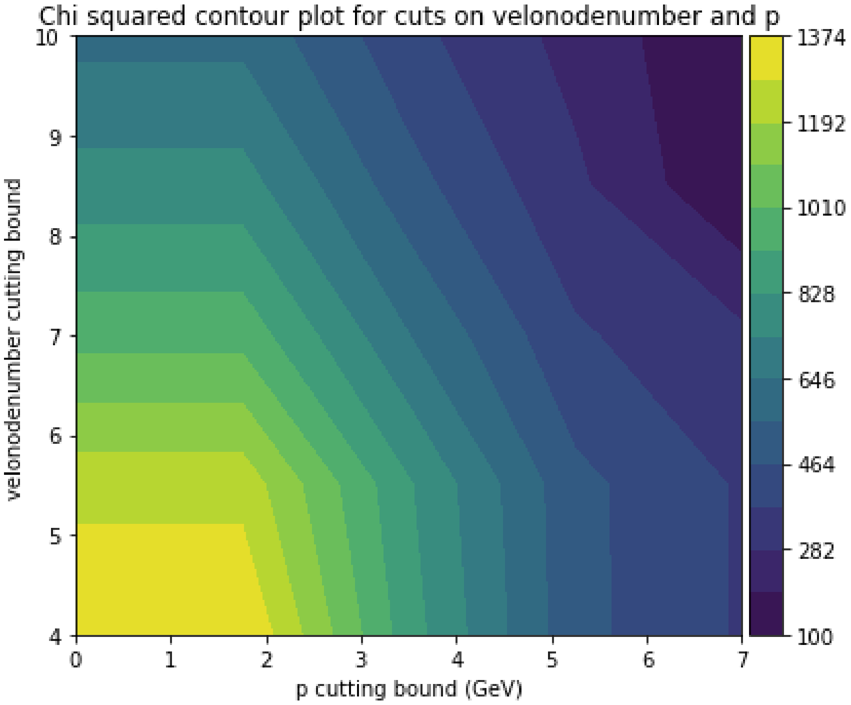


Figure 4.9 Effect on χ^2 of varying cuts on both momentum and nodenumber. Darker regions correspond to lower χ^2 values.

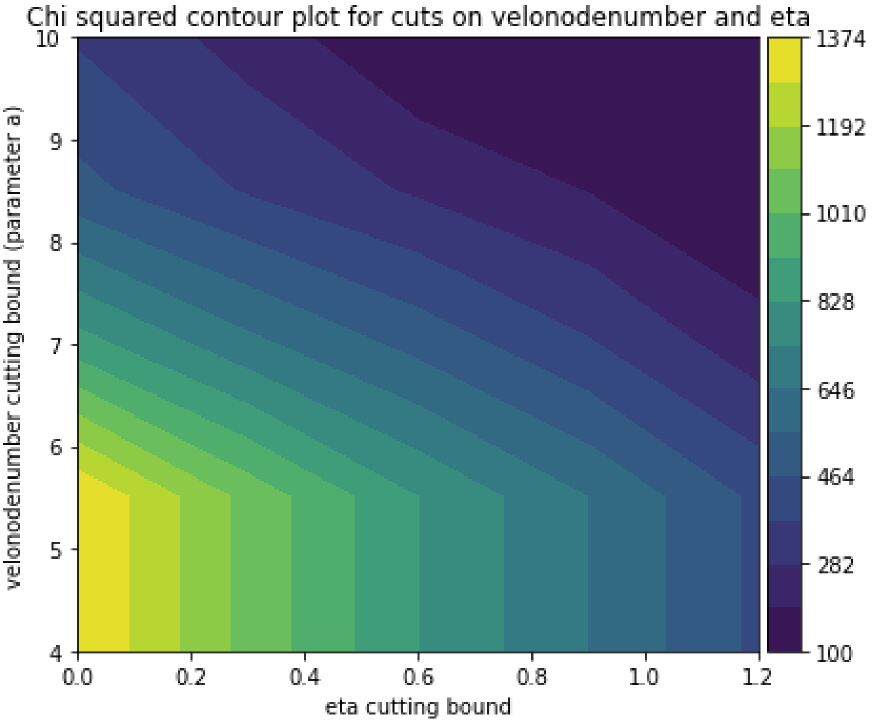


Figure 4.10 Effect on χ^2 of varying cuts on both eta and nodenumber. Darker regions correspond to lower χ^2 values.

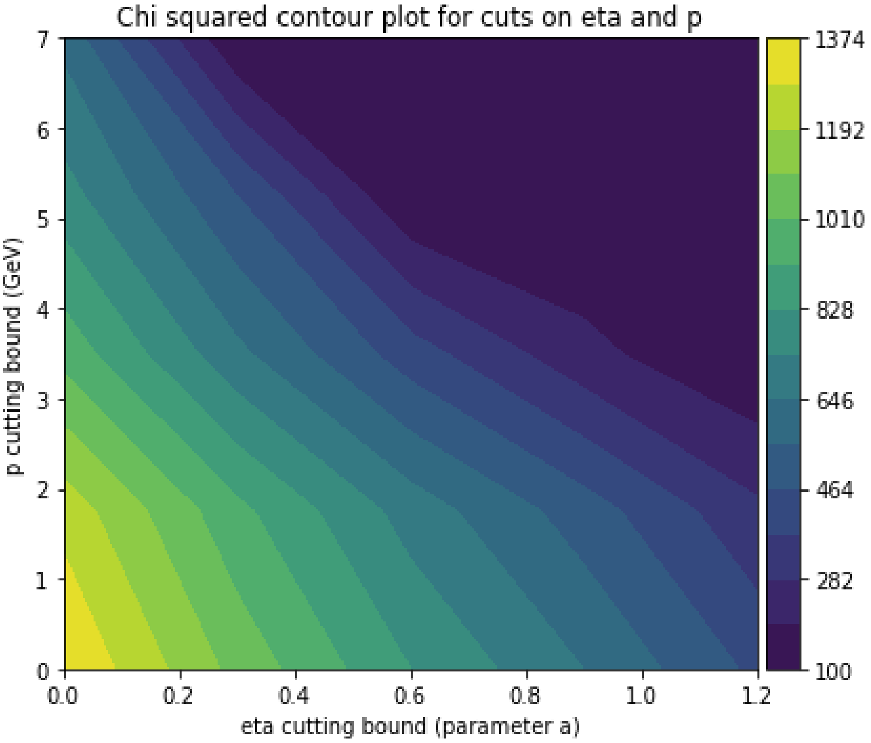


Figure 4.11 Effect on χ^2 of varying cuts on both eta and momentum. Darker regions correspond to lower χ^2 values.

It is clear from all of the above contour plots that again, χ^2 generally improves as the described cuts each become harsher. **Figure 4.11** indicates that the harshest possible cutting on eta and momentum may offer no benefit compared to, for example, having the parameter $a = 0.8$ and $p > 5$GeV. Any combination of cuts within the darkest regions of the contour plots were considered 'optimal' cuts. Unfortunately, it was not possible to simultaneously cut optimally on all variables at once (again, too few tracks appeared), so a further trial-and-error approach was taken. This involved trying different sets of cuts that were close to the optimal values (taken from the contour/scatter plots), but less harsh so that enough tracks were still present. As well as χ^2, the quantity 'absolute error' was defined to provide another metric for the accuracy of misalignment estimates, due to the fact that χ^2 could become smaller for worse estimates, if their errors have increased significantly. The absolute error represented the summed absolute differences between each modules misalignment estimate and the actual misalignment value. **Table 2** compares these metrics for when there was no cutting, and when the 'best' cutting had been applied:

	No cuts	**Best cuts**
Min. node number	0.0	8.0
Min. momentum (GeV)	0.0	3.1
Eta cutting parameter *a*	0.0	1.0
χ^2	1374.39	64.35
Abs. error (mm)	1.87	1.70

Table 2 Comparison of χ^2 and abs. error metrics for no cutting, and the 'best' cutting parameter values, using simulated events with known module misalignments. Cutting greatly improves χ^2 but only slightly improves the absolute error of the misalignment estimates.

Appendix **Figure A6** and **Figure A7** again give visual comparisons of the misalignment estimates to the known misalignments, this time with the ‘best’ cuts applied to the data. The estimates are similar to earlier, but with increased errors.

5 Results & Analysis

5.1 Results

Once the method of misalignment retrieval had been optimised by finding the 'best' data cuts, it was tested on a tuple containing event data with unknown module misalignments. The simulated misalignments were then 'revealed' to again allow a comparison with the predicted values so that the overall success of the method could be judged. This final set of random misalignments followed a uniform distribution with a maximum value of ±0.01mm.

Plots were again produced comparing individual sensor misalignment predictions, with error bars, to the corresponding actual module misalignments (**Figure 5.1, Figure 5.2**). A final χ^2 (summed for x and y), as well as an 'absolute error' (as described above), were also calculated. **Table 3** gives a final comparison of these values for the three cases: known misalignments with no cutting, known misalignments with optimal cutting (the optimised method), and unknown misalignments using the optimised method. The misalignment values, predictions and uncertainties are given in whole in **Table 2** of Appendix A.

Case	χ^2	**Abs. error (mm)**
known misalignments, no cutting	1374.39	1.87
known misalignments, optimised method	64.35	1.70
unknown misalignments, optimised method	42.59	0.304

Table 3 Comparison of misalignment prediction accuracy metrics for the three described cases.

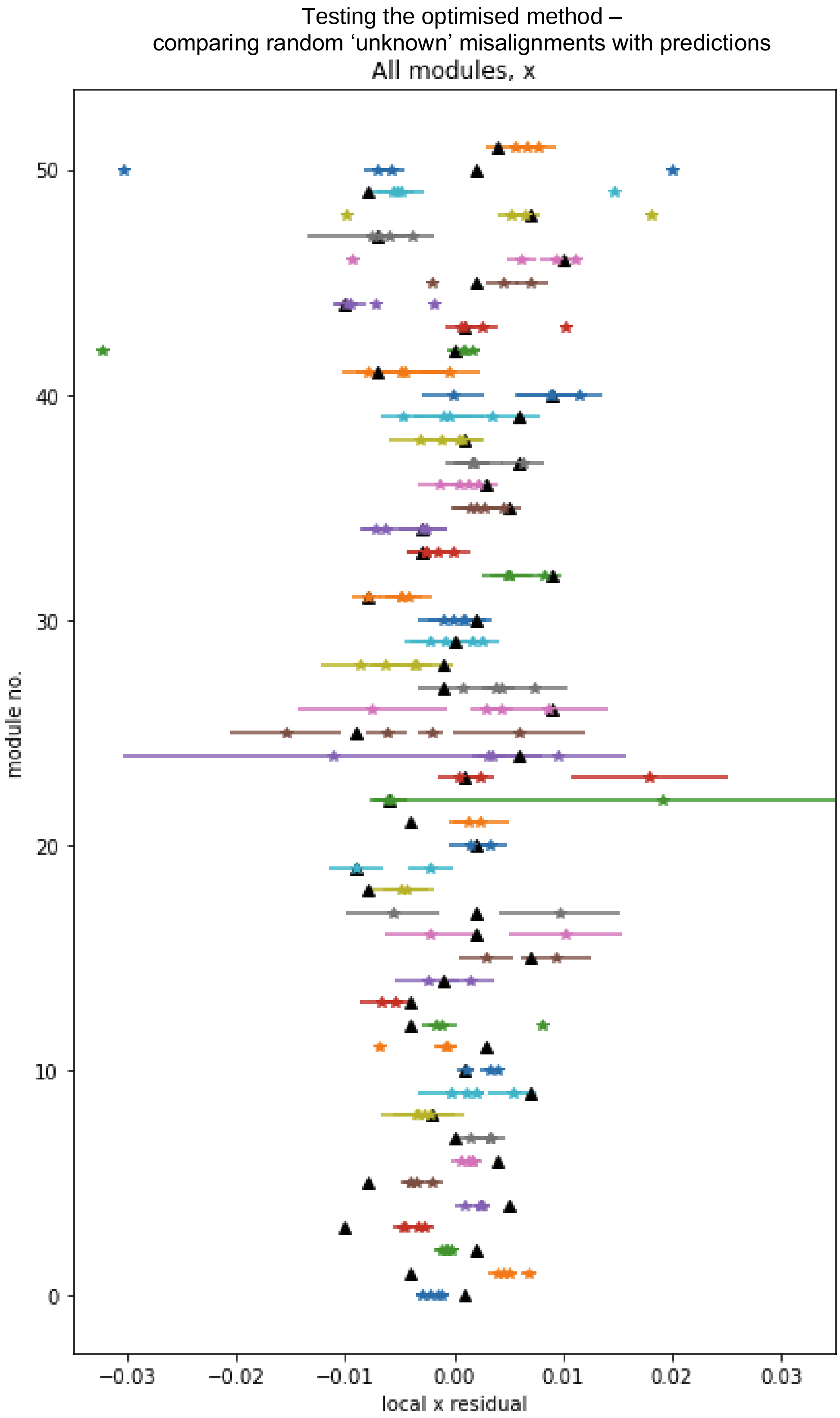


Figure 5.1 Comparison of predicted sensor misalignments (stars) to actual 'unknown' module misalignments (triangles), in x. Sensors corresponding to a given module are marked in the same colour and modules are separated vertically, ranging from 0 (bottom) to 51 (top).

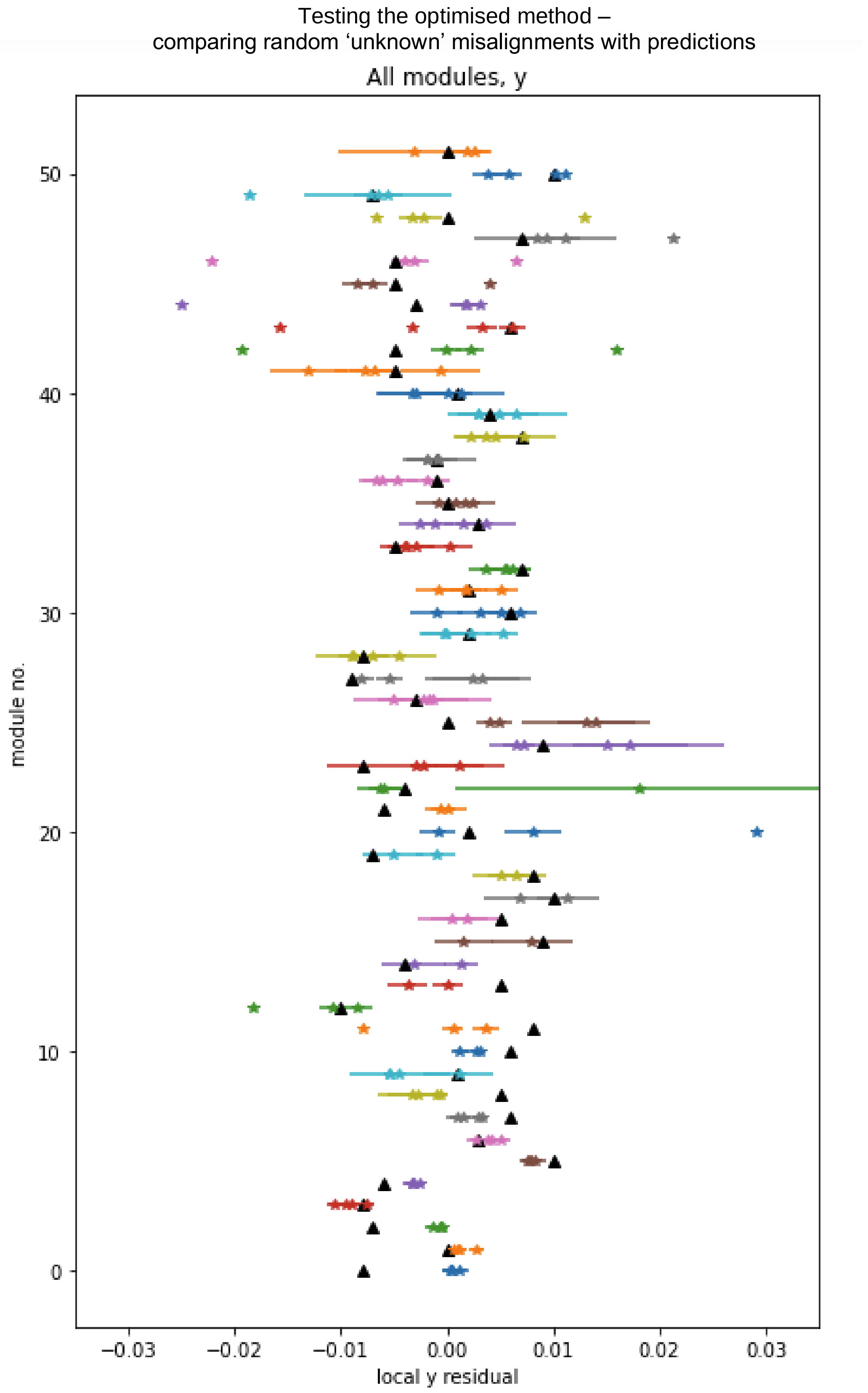


Figure 5.2 Comparison of predicted sensor misalignments (stars) to actual 'unknown' module misalignments (triangles), in y. Sensors corresponding to a given module are marked in the same colour and modules are separated vertically, ranging from 0 (bottom) to 51 (top).

5.2 Analysis

From the two figures above it is clear that the optimised method was able to yield many misalignment predictions that are consistent with the actual values, with some outliers. The outlying points that have suspiciously small uncertainties appear to be correlated with higher module numbers (>40), which are located the furthest down the beamline from the interaction point and so could be expected to register fewer hits. However, it is not known why these small uncertainties still arise, as the method was developed to discard sensors with very few registered hits and thus (counter-intuitively) underestimated errors.

By comparing the final data in **Figure 5.1** with both **Figure 3.8** (no cuts) and Appendix A **Figure A6*** (best cuts), it can be seen that the method generally appears to estimate misalignments much more accurately for smaller misalignment values. In any case, most misalignments greater than around 0.01mm are poorly resolved. The reason for this is that the pattern recognition algorithm looks for hits that lie within a certain 'window', so the more a module is misaligned the more likely it is a hit will lie outside this region and not be recognised as part of the track. Greater misalignments also increase the small probability of a 'wrong' hit being added (a hit from a different track). It therefore seems likely that, because the final set of misalignments on which the method was tested only includes values ranging to ±0.01mm, it performed better here than during earlier testing.

Table 3 shows that a final χ^2 value of 42.59 was achieved, a considerable improvement compared to the best value that could be produced (64.35) by optimising the model when the misalignments were known. The absolute error also improved substantially from 1.70mm to 0.304mm. The overall usefulness of misalignment estimates may also be judged by subtracting them from the present misalignment values. This should, on average, have the effect of ‘shifting’ residual distributions towards zero. If the estimates have the overall effect of bringing the modules closer towards being aligned, the mean of the absolute differences between estimated misalignment values and true values will be smaller than the mean of the absolute true values. These quantities can be referred to as ‘mean absolute misalignments’, before and after alignment. The final results of this analysis yielded a mean absolute misalignment of (0.00697±0.00295) mm, compared to 0.00967mm before alignment, so the method overall could likely be used to bring the detector towards alignment.

***For ease of reading, only figures corresponding to x-axis data have been mentioned.*

6 Conclusion

This study involved the development of a method for the alignment of the VELO tracking detector at LHCb, by analysing Monte Carlo simulated data in a Python framework. The method was optimised using a set of known misalignments and was then tested by attempting to retrieve an unknown set. Overall, it was successful, as it has shown the ability to provide stable, relatively accurate misalignment estimates when they are below the scale of 0.01mm. Although the estimates are not perfect, this was not unexpected as there are too many factors unaccounted for (such as the non-linear nature of tracks). At this current stage the solution does not represent a practical method for the alignment of the detector – it instead represents a necessary part of a wider solution for the full alignment of the detector.

The next steps that may be taken, which were beyond the scope of this project, could firstly involve analysing a much larger dataset. The analysis described above involved small sample sizes (due to time and resource constraints), while in reality the experiment can generate Petabytes of data per second. Studying larger datasets would result in more well-defined residual distributions (e.g., **Figure 3.5**) and therefore fewer misalignment estimates with large uncertainties. The addition of iterative stages, which would include 'feeding' estimated misalignment values back into the simulation and repeating the method, should cause the estimates to converge towards the true values. The 'mean absolute offsets' explained on the page above show that the method described in this analysis represents the first run of this iterative process, as module misalignments tended to be closer to zero after the subtraction of the estimated values.

Bibliography

[1] Belyaev, I., Carboni, G., Harnew, N. et al., “The History of LHCb,” EPJ H, vol. 46, no. 3, 2021.

[2] The LHCb Collaboration, “Framework TDR for the LHCb Upgrade: Technical Design Report,” CERN, 2012.

[3] G. Ciezarek, M. Sevilla, B. Hamilton, R. Kowalewski, T. Kuhr, V. Lüth and Y. Sato, “A Challenge to Lepton Universality in B Meson Decays,” Nature, vol. 546, 2017.

[4] LHCb Collaboration et al, “The LHCb Detector at the LHC,” JINST, vol. 3, no. S08005, 2008.

[5] K. Hennessy, “LHCb VELO Upgrade,” Nuclear Instruments and Methods in Physics Research, vol. 845, 2016.

[6] LHCb Collaboration, “LHCb VELO Upgrade Technical Design Report,” CERN, 2013.

[7] V. Chiochia, “Accelerators and Particle Detectors,” University of Zurich, 2010.

[8] L. Eklund, “Physics benchmarks of the VELO upgrade,” JINST, vol. 11, 2016.

[9] “Determining the lifetime of the Bs,” CERN.

[10] S. K. D.E. Groom, “Passage of Particles Through Matter,” 2021. [Online]. Available: https://pdg.lbl.gov/2022/reviews/rpp2022-rev-passage-particles-matter.pdf.

[11] A. M. P. Sjo̎gren, “Alignment of the VELO pixel detector,” CERN, 2017.

[12] G. B. G. Welch, “An Introduction to the Kalman Filter,” SIGGRAPH, 2001.

Appendix A

Module No.	X Misalignment (mm)	Y Misalignment (mm)
0	-0.030	-0.020
1	-0.006	0.043
2	-0.045	0.011
3	-0.019	0.044
4	-0.038	-0.031
5	0.040	-0.035
6	-0.046	0.041
7	-0.049	-0.010
8	0.042	0.049
9	0.042	-0.045
10	-0.050	-0.023
11	0.004	-0.008
12	0.023	0.048
13	-0.037	0.041
14	-0.039	-0.008
15	-0.017	-0.051
16	0.045	0.052
17	0.038	0.017
18	-0.054	0.035
19	0.041	-0.005
20	-0.051	0.011
21	0.020	0.050
22	0.044	-0.013
23	-0.005	-0.020
24	-0.041	0.028
25	0.046	-0.036
26	-0.002	-0.048
27	-0.055	-0.043
28	-0.017	0.045
29	-0.037	0.034
30	0.034	-0.028
31	0.047	0.007
32	-0.008	0.025

Module No.	**X Misalignment (mm)**	**Y Misalignment (mm)**
33	-0.044	-0.001
34	0.039	-0.053
35	0.018	-0.051
36	0.036	-0.025
37	-0.003	0.005
38	0.026	0.012
39	0.027	0.017
40	-0.022	0.013
41	-0.016	-0.028
42	0.028	-0.032
43	0.007	-0.037
44	0.028	-0.055
45	-0.042	-0.010
46	0.002	-0.046
47	0.013	0.010
48	-0.044	-0.033
49	0.013	0.051
50	0.017	0.010
51	0.017	-0.031

Table A1 All ‘known’ module misalignment values (used for optimising the method of misalignment finding).

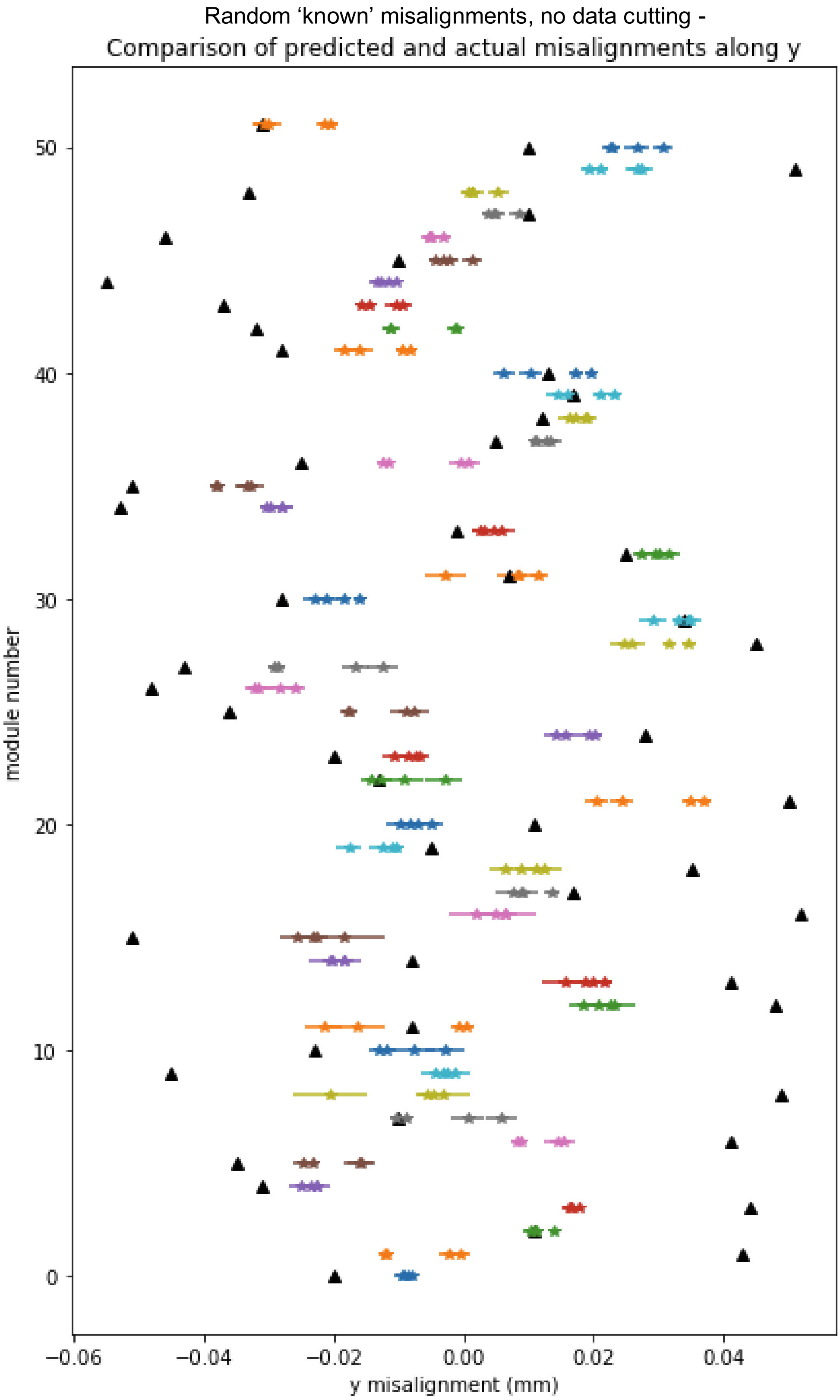


Figure A1 Comparison of sensor misalignment estimates with known misalignments, with no data cutting, for y.

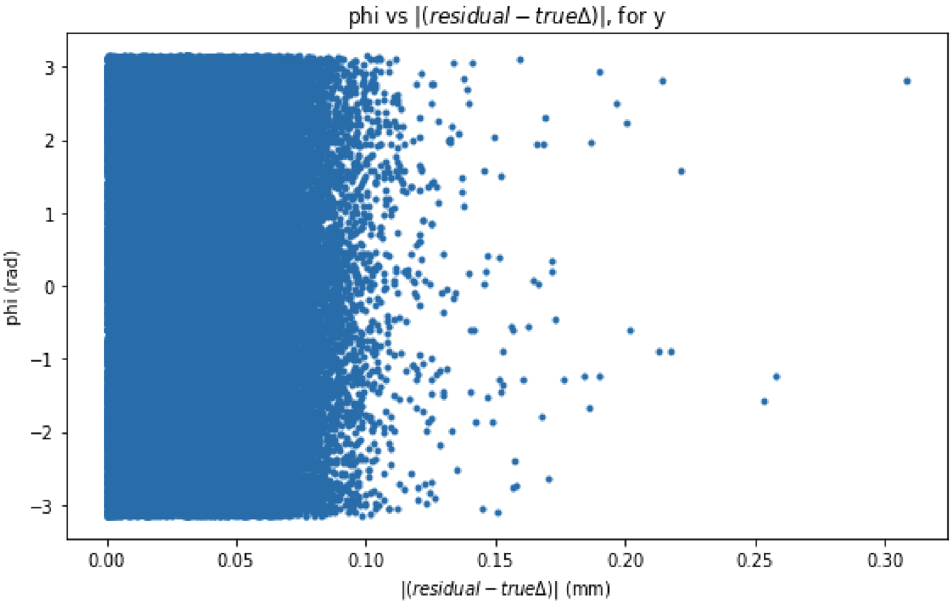


Figure A2 Comparison between track resolution and ϕ (for residuals along y). There is no correlation between the two so, combining this with a similar result for y, it is not possible that cutting the data on ϕ would remove any of the more poorly resolved tracks.

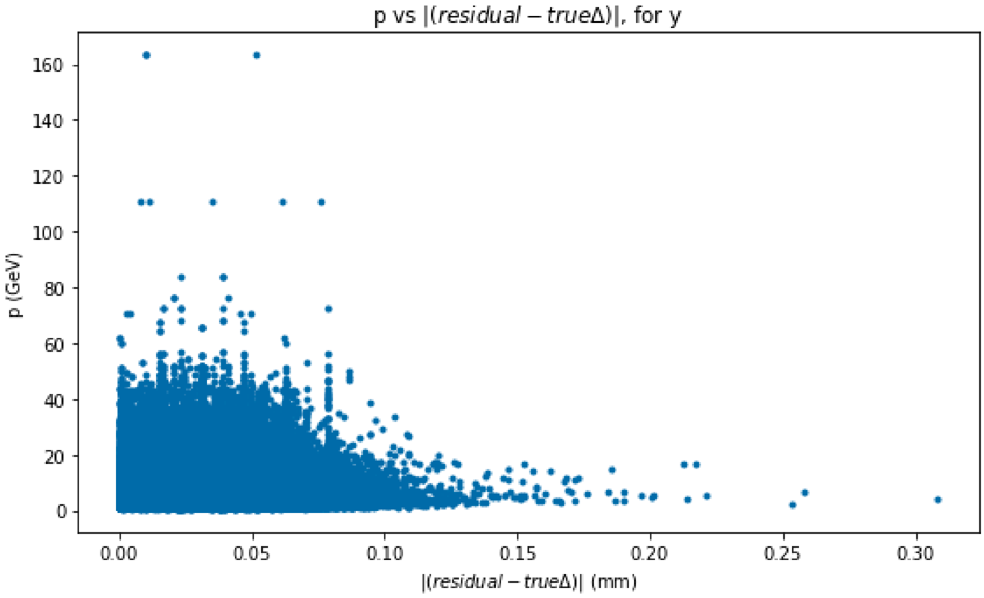


Figure A3 Comparison between track resolution and momentum, for y. There is some correlation between the two - it appears that cutting away tracks below approximately 20GeV would remove those that are most poorly resolved (right hand side). This was expected as higher momentum particles are less likely to exit a material at a large scattering angle (see equation 1), so straight-line track fitting provides a more accurate approximation.

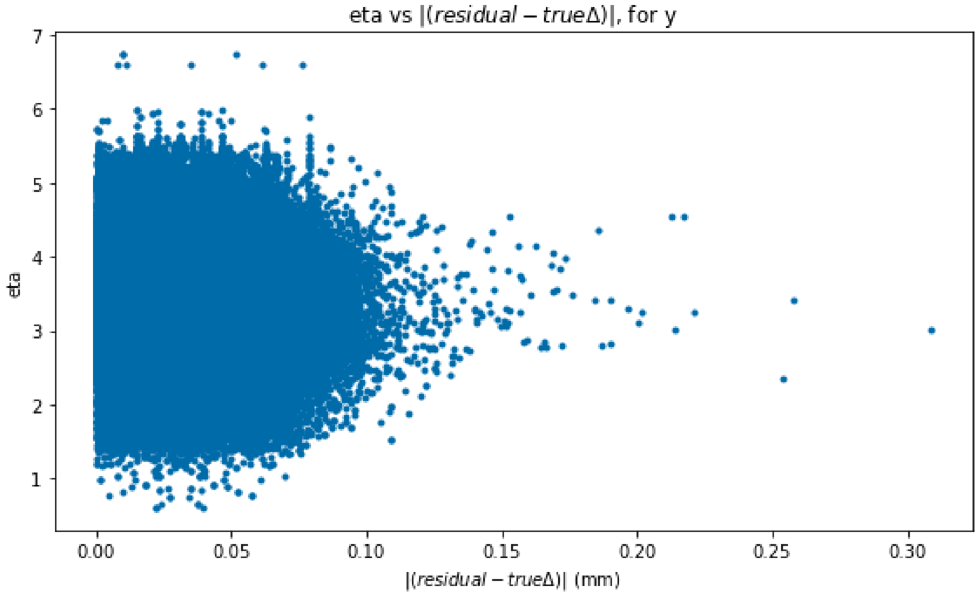


Figure A4 Comparison between track resolution and pseudorapidity, for y. Better resolved tracks appear to have relatively low and high eta values.

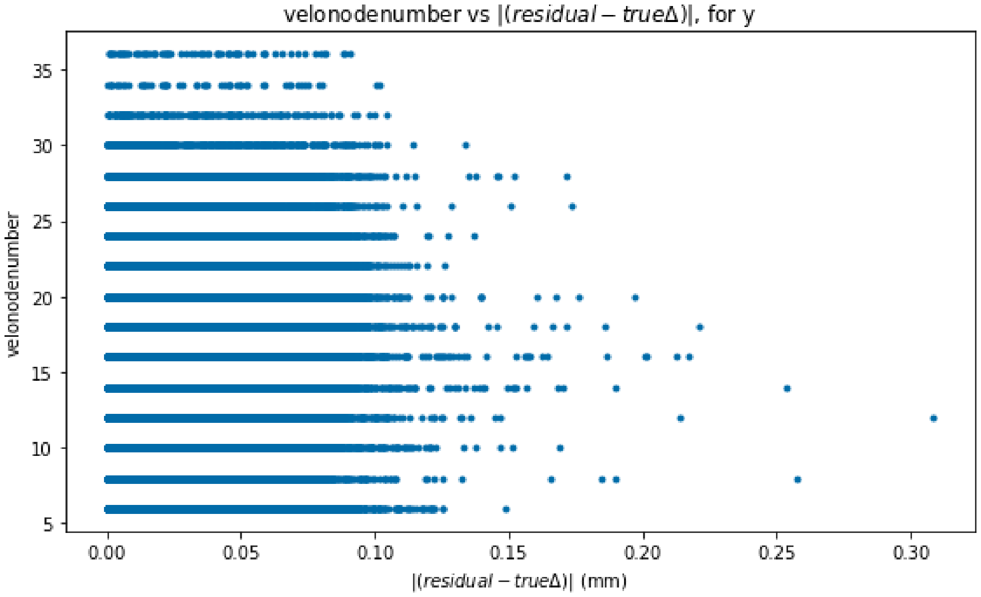


Figure A5 Comparison between track resolution and velonodenumber, for x. Here, more poorly resolved tracks appear more for lower values of velonodenumber. Horizontal lines appear because the variable only takes even-integer values, due to the fact that x and y are treated as separate variables (and therefore separate nodes).

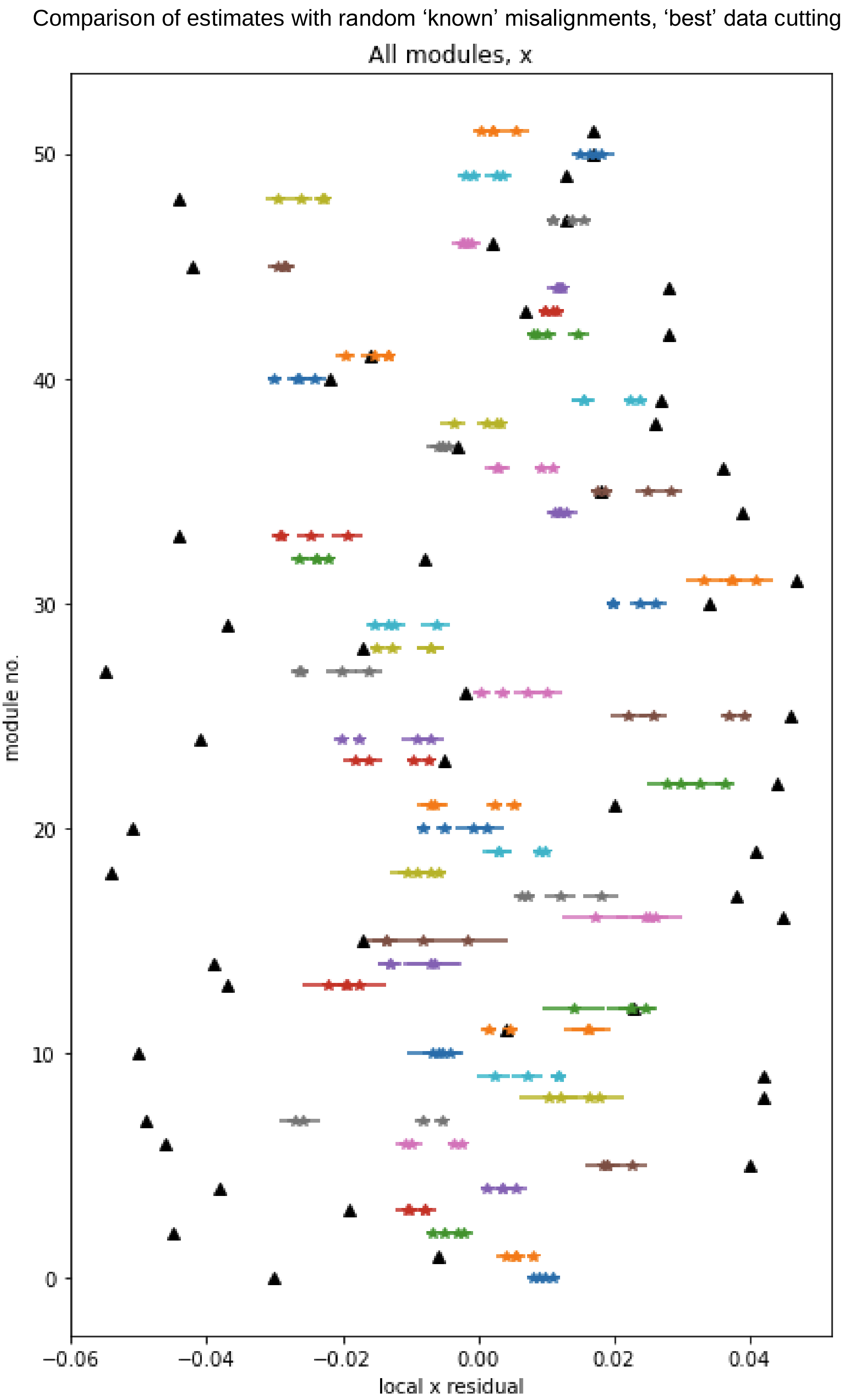


Figure A4 Comparison of sensor misalignment estimates with known misalignments, with ‘best’ data cutting, for x.

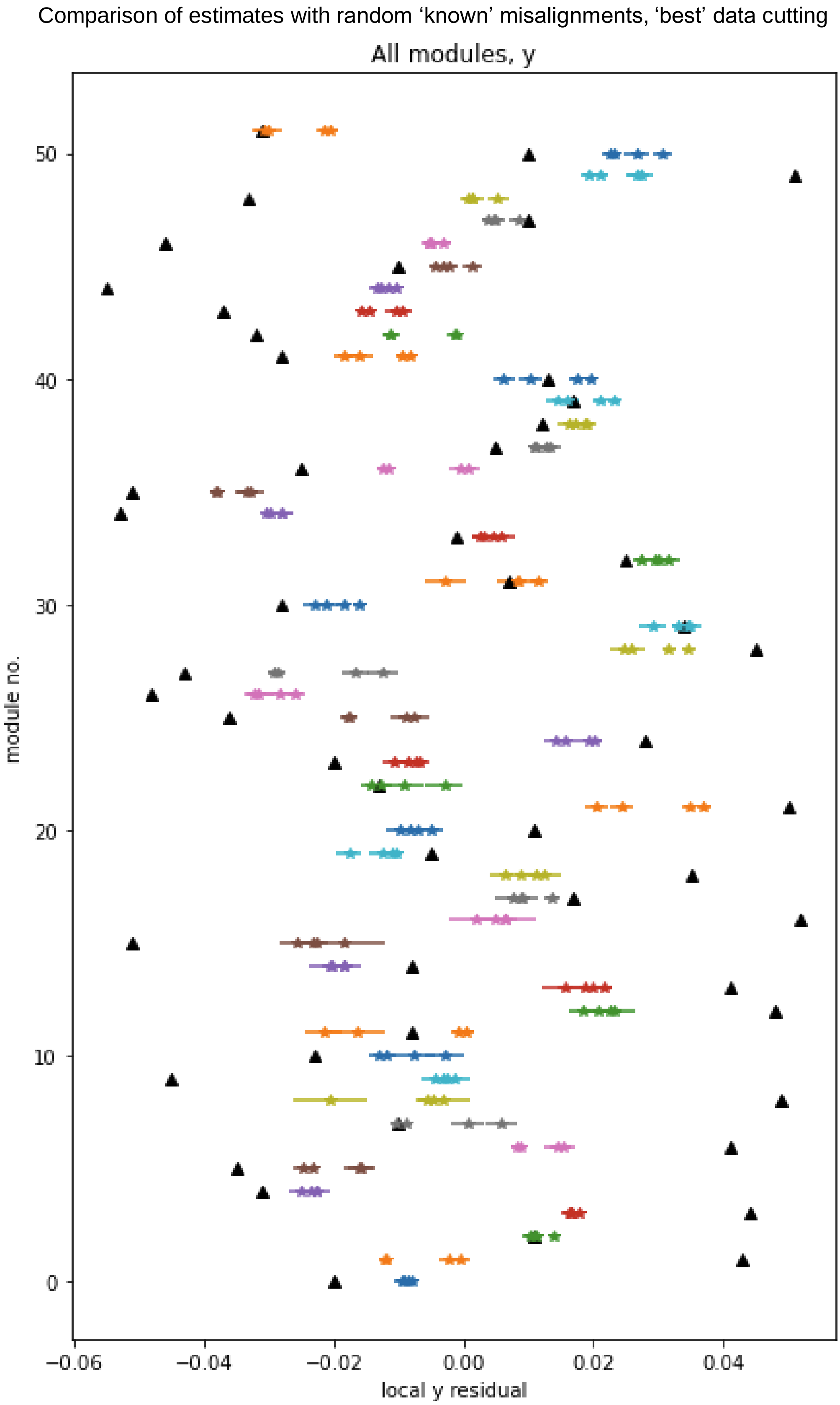


Figure A5 Comparison of sensor misalignment estimates with known misalignments, with ‘best’ data cutting, for y.

Module No.	**X Mis-alignment (mm)**	**X Prediction (mm)**	**Y Mis-alignment (mm)**	**Y Prediction (mm)**
0	0.001	-0.00216 ±0.00034	-0.008	0.00050±0.00035
1	-0.004	0.00510 ±0.00038	-0.000	0.00139±0.00037
2	0.002	-0.00068 ±0.00031	-0.007	-0.00075±0.00031
3	-0.010	-0.00386 ±0.00036	-0.008	-0.00924±0.00034
4	0.005	0.00223 ±0.00032	-0.006	-0.00303±0.00032
5	-0.008	-0.00363 ±0.00035	0.010	0.00782±0.00034
6	0.004	-0.00148 ±0.00032	0.003	0.00396±0.00032
7	0.000	-0.00316 ±0.00035	0.006	0.00281±0.00034
8	-0.002	-0.00312 ±0.00033	0.005	-0.00092±0.00034
9	0.007	-0.00162 ±0.00036	0.001	-0.00543±0.00036
10	0.001	0.00201 ±0.00060	0.006	0.00179±0.00061
11	0.003	-0.00077 ±0.00071	0.008	0.00168±0.00076
12	-0.004	-0.00144 ±0.00091	-0.010	-0.00965±0.00094
13	-0.004	-0.00589 ±0.00111	0.005	-0.00131±0.00110

Module No.	**X Mis-alignment (mm)**	**X Prediction (mm)**	**Y Mis-alignment (mm)**	**Y Prediction (mm)**
14	-0.001	0.00019 ±0.00172	-0.004	0.00040±0.00140
15	0.007	0.00527 ±0.00194	0.009	0.00369±0.00224
16	0.002	0.00251 ±0.00322	0.005	0.00113±0.00229
17	0.002	0.00021 ±0.00340	0.010	0.00948±0.00226
18	-0.008	-0.00460 ±0.00170	0.008	0.00575±0.00193
19	-0.009	-0.00499 ±0.00158	-0.007	-0.00218±0.00156
20	0.002	0.00259 ±0.00127	0.002	0.00164±0.00145
21	-0.004	0.00154 ±0.00128	-0.006	-0.00032±0.00109
22	-0.006	-0.00587 ±0.00107	-0.004	-0.00596±0.00123
23	0.001	0.00217 ±0.00106	-0.008	-0.00139±0.00115
24	0.006	0.00342 ±0.00091	0.009	0.00712±0.00084
25	-0.009	-0.00343 ±0.00095	0.000	0.00510±0.00084
26	0.009	0.00374 ±0.00088	-0.003	-0.00283±0.00085
27	-0.001	0.00424±0.00081	-0.009	-0.00619±0.00082
28	-0.001	-0.00515	-0.008	-0.00788±0.00096

Module No.	**X Misalignment (mm)**	**X Prediction (mm)**	**Y Misalignment (mm)**	**Y Prediction (mm)**
		±0.00095		
29	-0.000	0.00131±0.00089	0.002	0.00258±0.00094
30	0.002	-0.00003±0.00093	0.006	0.00443±0.00086
31	-0.008	-0.00587±0.00078	0.002	0.00240±0.00090
32	0.009	0.00607±0.00086	0.007	0.00508±0.00095
33	-0.003	-0.00160±0.00087	-0.005	-0.00278±0.00095
34	-0.003	-0.00508±0.00095	0.003	-0.00020±0.00098
35	0.005	0.00289±0.00087	-0.000	0.00115±0.00095
36	0.003	0.00077±0.00088	-0.001	-0.00517±0.00091
37	0.006	0.00269±0.00091	-0.001	-0.00143±0.00106
38	0.001	-0.00044±0.00103	0.007	0.00374±0.00108
39	0.006	-0.00240±0.00131	0.004	0.00337±0.00133
40	0.009	0.00786±0.00134	0.001	-0.00124±0.00158
41	-0.007	-0.00493±0.00146	-0.005	-0.00716±0.00146
42	-0.000	0.00082±0.00099	-0.005	0.00110±0.00096
43	0.001	0.00166±0.00100	0.006	0.00483±0.00095
44	-0.010	-0.00978±0.00091	-0.003	0.00178±0.00101
45	0.002	0.00584±0.00104	-0.005	-0.00786±0.00100
46	0.010	0.00773±0.00097	-0.005	-0.00363±0.00094
47	-0.007	-0.00647±0.00095	0.007	0.00960±0.00098

Module No.	X Mis-alignment (mm)	X Prediction (mm)	Y Mis-alignment (mm)	Y Prediction (mm)
48	0.007	0.00567±0.00094	0.000	-0.00287±0.00103
49	-0.008	-0.00522 ±0.00091	-0.007	-0.00649±0.00104
50	0.002	-0.00643 ±0.00096	0.010	0.00487±0.00100
51	0.004	0.00697±0.00095	-0.000	0.00209±0.00101

Table A2 Final results - all ‘unknown’ module misalignment values and their corresponding estimates, with uncertainties.