ATLAS muon reconstruction from a C++ perspective: a road to the Higgs
Hendriks, P.J.

Citation for published version (APA):
Hendriks, P. J. (2000). ATLAS muon reconstruction from a C++ perspective: a road to the Higgs

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
The reconstruction of events in the ATLAS muon spectrometer is built on top of the AMBER framework (see section 3.2), utilizing the classes provided by the DRT (see section 3.3), with the actual reconstruction algorithm implemented in terms of the GDL (see section 3.4). From a global perspective, the reconstruction looks like the component diagram shown in figure 4.1.

**Figure 4.1** Global view of the muon reconstruction algorithm.

The hits in the trigger chambers (RPCs and TGCs) are retrieved from the detector layers with the help of a ContainerStubInterface descendant (cf. figure 3.18). These hits are used to build the regions of activity called trigger roads to which the subsequent reconstruction of the precision chambers is confined. The pattern recognition creates track segments out of the precision hits, which are when possible combined into tracks. And finally, a global fit of the track segments and the trigger hits is performed to determine the exact track parameters.

1. The precision-chamber reconstruction is not directly coupled to the TriggerRoad class, and in fact any descendant of the DRT class RegionOfActivity will do.
4.1 Trigger Chamber Reconstruction

The first step in the reconstruction of tracks in the muon spectrometer is the creation of trigger roads, i.e. regions of activity based on the hit information in the trigger chambers. They are needed to guide the reconstruction of the precision chambers because of the following reasons (see also section 2.3):

- The high background environment in the precision chambers requires the presence of a selection criterion with a high capability of rejecting the background hits if the execution time of the algorithm is to be kept in check;

- The large drift times of the precision chambers relative to the bunch spacing of the LHC make an efficient tagging of the bunch crossing to which a given track belongs by the chambers themselves impossible;

- The MDT chambers do not measure the azimuthal coordinate along the wire, which is needed to calculate the real drift time of a hit;

Because of their fast read-out and very low occupancy, the trigger chambers are very well suited for these tasks.

The algorithm for finding the trigger roads mimics part of the work that is performed by the level-2 trigger [45]. However, instead of trying to determine the momentum of the passing muon as is the task of the trigger, the goal of the algorithm here is to define a road that contains all the muon hits and a minimum of background hits. This is essential because all subsequent processing is limited to hits that lie inside the road.

The main advantage of the trigger algorithm as it is implemented here is that it is fast because it uses only the geometrical properties of the trigger chambers, and does not require any knowledge of the hits in the precision chambers, nor of the magnetic field. It is described in sections 4.1.3 and 4.1.4 for the low- and high-$p_T$ trigger respectively, but first the reconstruction of the individual chambers is explained.

4.1.1 The RPC Chambers

The RPC chambers provide the trigger information in the barrel of the muon spectrometer. They consist of two layers, each one based on a gas gap around which two strip planes provide respectively the $\phi$- and $\eta$-coordinate of a track (see figure 4.2). Each such plane is represented by a detector in AMBER’s detector description, and its digits are the individual strips that are hit by a particle.

When such a particle crosses a strip close to its edge, neighbouring strips can also fire, resulting in multiple digits being generated by a single track. Therefore, the first step in the reconstruction is to cluster adjoining digits (see figure 4.3). A cluster is based on the ErrorPoint class provided by the DRT. It stores a position corresponding to the centre of gravity of the digits
4.1. Trigger Chamber Reconstruction

that make up the cluster, and a 3-dimensional error based on their extent. When the goal of the trigger reconstruction would have been to calculate the position of the track, a scale factor of $\sqrt{2}$ or higher could have been applied to this extent. But as we are searching for the boundaries within which the particle has traversed the detector, the whole extent must be taken into account.

Based on the ATLAS trigger definition as described in chapter 2, one cluster in each projection is the minimum requirement for a trigger signal to be generated by an RPC chamber. This means that from this point on there are two possible ways to proceed. The first is to combine the clusters of the $\eta$- and $\phi$-planes that make up a layer. This approach fails however when a particle generates a hit in only one of the two planes. When such a hit is combined with one of the uncorrelated clusters in the other plane, its size is incorrectly restricted in the dimension that is measured by the second cluster. Therefore, the only solution is to keep all original clusters, but this not only increases the number of combinatorials in the reconstruction that follows, but also requires an extra step at its end in which duplicate clusters have to be removed.

The other strategy, and the one that has been adopted, is to first combine the clusters in the planes that have the same orientation, i.e. either the $\phi$- or the $\eta$-planes. In this case only the

---

2. When a cluster contains two digits, the position of the track can be inferred to have been close to the boundary between the two strips, except of course when one of the digits was caused by a $\delta$-ray or any other source of background.
original clusters that have not been used need to be saved for the next step. The algorithm for the reconstruction of such a doublet of detector planes is shown in figure 4.4.

![Diagram of the reconstruction algorithm](image)

**Figure 4.4** Reconstruction of a doublet of detector element planes.

When two clusters, one from each plane, overlap or are close enough as defined by the user, they are added together into a single cluster. Because the clusters, like the digits they are based on are sorted, a SortedCombinatorials dataview can be used for this process, in conjunction with a Transformer. The latter takes as its input the pairs of clusters coming out of the SortedCombinatorials and for each pair calculates their total extent and sets the origin equal to their centre. As a last step, a Merger concatenates the list of these clusters with the original ones that were not used in the combinatorials.

The final step in the reconstruction of an RPC chamber is to take the combinatorials of the $\eta$- and $\phi$-clusters. Because according to the ATLAS trigger logic at least one hit is required in each projection, and because there is no way to determine which clusters belong together, all combinations of the clusters of the two doublets must be taken (see figure 4.5). Such a combination is formed by calculating the weighted sum of the two clusters, which results in a cluster the size of their overlap region.

3. The SortedCombinatorials dataview both constructs the combinatorials of the values of its two inputs and applies a filter on the created pairs. Because of its knowledge about the ordering of the input values it can use a binary search algorithm, which makes it (much) faster than when these two operations were applied separately (see also section 3.4.2).
The RPC layers

The RPC chambers are arranged in three cylindrical layers, consisting both of large chambers in the odd $\phi$-sectors, and of small chambers in the even sectors (see figure 2.3). As a particle coming from the interaction point can cross such a layer only once, it makes sense to combine the reconstructed clusters from the chambers that make up a layer into a single stream (see figure 4.6). The clusters are sorted in $\phi$ and not in $\eta$, because the roads are much narrower in the $\phi$-projection where there is hardly any magnetic field that can cause the tracks to bend. As a result, a filtered combinatorials on $\phi$ later on in the reconstruction will give the most reduction in the number of combinations.

4.1.2 The TGC Chambers

In the endcaps the trigger information is provided by the TGC chambers. These are multiwire proportional chambers of which three different types are used, depending on their position within ATLAS (cf. figure 2.7). In the innermost TGC0 layer, the chambers consist of only two wire planes. These wires, which measure the azimuthal coordinate are grouped together, 4 to 20 at a time. Such a wire-group behaves just like a strip from the perspective of the
reconstruction, and hence the algorithm described above for the RPCs can be reused here. Only, the building of the combinatorials out of the two different projections as shown in figure 4.5 must of course be skipped. For the second type of TGC chambers, which consist of two wire planes in conjunction with two strip planes (see figure 4.7b) even this small deviation from the RPC algorithm is not needed.

![Figure 4.7](image-url)

**Figure 4.7** Schematic view of a triplet (a) and of a doublet (b) of TGCs (the gas gap is not shown to scale).

That leaves the triplets, chambers consisting of two strip and three wire planes. For the strips, the standard algorithm can be used, but for the wires a specialized version must be developed. The ATLAS trigger logic states that a 2 out of 3 coincidence is required (cf. section 2.3), and so all three the combinatorials of 2 layers each are taken (see figure 4.8). Of course, this procedure overestimates the number of real clusters, as a track can create hits in all three layers. Therefore, as a last step, the clusters that are compatible with each other are combined into a single one.

The result of the reconstruction so far is the creation of 11 trigger layers (three in the barrel and four in each endcap), which can be used to create the trigger roads. The low-$p_T$ roads are constructed first, after which an attempt is made to extend them into the high-momentum regime. These algorithms are in no way dependent on the RPC or TGC background of the trigger layers, and use only the positions and sizes of the generic trigger clusters.

### 4.1.3 Low-$p_T$ Trigger

The low-$p_T$ trigger is a 6 GeV trigger based on a 3 out of 4 coincidence in each projection in the two middle RPC or in the two outer TGC layers (see figure 2.7). Because the individual chambers were reconstructed based on a 1 out of 2 coincidence per projection, by combining
the clusters of the two trigger layers, the reconstruction algorithm is capable of finding all tracks that pass the ATLAS trigger, and in addition the ones that leave only one hit per projection in each layer.

The algorithm itself is quite straightforward. When two trigger clusters are close enough in $\phi$ and $\eta$, they are combined into a trigger road (see figure 4.9). As the clusters are sorted in $\phi$, the combinations are created with the help of a SortedCombinatorials dataview. The filtering

![Diagram of trigger reconstruction](image)

**Figure 4.8** Reconstruction of the three wire planes of a TGC triplet.

![Diagram of low-$p_T$ trigger reconstruction](image)

**Figure 4.9** Low-$p_T$ trigger reconstruction.
in $\eta$ is subsequently performed by a regular Filter class, after which a Transformer is responsible for the creation of the roads in the form of TriggerRoad objects.

The TriggerRoad class is derived from DRT's ErrorCone, which represents a three-dimensional cone with variable $\phi$- and $\eta$-shapes (see figure 4.10). For the low-$p_T$ trigger, the $\phi$-shape has the form of an hourglass with a width and opening angle based on the size of the two clusters. In the $\eta$- or bending plane a chalice shape is used. Both of its sides are helices aimed away from the axis of the cone. For ATLAS these are set to 6 GeV trajectories in a 0.5 Tesla field. An actual estimate of the momentum of the track from the two clusters that form the road is not possible because their separation too small compared to their size [46].

![Figure 4.10](image)

**Figure 4.10** The different trigger road shapes used by the reconstruction, viz. hourglass (a), chalice (b) and helix (c). All of them can be used in both the $\phi$- and $\eta$-projections, but as the toroidal field of the ATLAS muon spectrometer only bends tracks in $\eta$, (a) is used as the $\phi$-shape, while either (b) or (c) form the $\eta$-shape.

As a final step in the low-$p_T$ reconstruction, compatible roads that can be found in the overlap regions of the small and large chambers are combined.

### 4.1.4 High-$p_T$ Trigger

The results from the RPC and TGC low-$p_T$ triggers are combined into a single list, and an extension into the high-$p_T$ regime is attempted. To that end, the clusters in the RPC3 and TGC1 layers are merged together and subsequently sorted in $\phi$ (the high-$p_T$ clusters dataview in figure 4.11). To match the clusters to the trigger roads, a SortedCombinatorials on $\phi$ and a Filter on $\eta$ are used. When both are successful, the cluster is added to the road. As a result the shape in the $\phi$-projection is narrowed, and the $\eta$-shape is changed into a helix form when the sign of the track’s charge could be determined. Otherwise, it remains a chalice-shape, all be it a reduced one.

The list of newly created trigger roads is augmented by the original low-$p_T$ ones that could not be extended into the high-$p_T$ regime. As a last step, a network identical to the one shown in figure 4.11 is used to try to refine these roads with the clusters found in the TGC0 layer, i.e. the innermost TGC chambers that only measure the azimuthal coordinate.
4.2 MDT Pattern Recognition

The trigger reconstruction is followed by the pattern recognition in the precision chambers. The ATLAS detector contains two types of these chambers, viz. the MDTs and the CSCs (cf. figure 2.4). The latter, which are only used in the inner forward regions, are not implemented by the ATLAS simulation program and have been replaced with MDTs instead. The reconstruction as implemented by AMBER will therefore do the same.

4.2.1 Local MDT Reconstruction

All MDT chambers consist of either one or two multilayers, containing three to four tube layers each (cf. figure 2.6). In the barrel these chambers are arranged in so-called ladders, i.e. rows of chambers adjacent in z (i.e. the beam axis), that belong to the same detector layer (i.e. cylinder), \( \phi \)-sector and side of the muon spectrometer (cf. figure 2.4). The corresponding entity in the endcaps is a sector of a MDT wheel, but for the remainder of this chapter, it too will be referred to as a ladder.

Seen from the interaction point, a ladder is a surface that a track can pass only once. Furthermore, the chambers are so close together in z that a particle can easily cross two
neighbouring chambers. To exploit this first feature, and to make the reconstruction independent of the second effect, the digits from identical tube layers of all the chambers in a ladder are grouped together (see figure 4.12). These layers are attached to the merger in such a way that the digits coming out of it are sorted in z for the barrel and in r for the endcaps.

![Diagram](image)

**Figure 4.12** Definition of the reconstruction algorithm for the i-th tube layer in a ladder.

A chamber can have anywhere between three and eight of these layers, which means that there are just as many lists of digits in the reconstruction of a ladder. To simplify the pattern recognition that is to follow, these lists are combined into a single one (see figure 4.13). A downside of this approach is that topology requirements on the hits can only be checked by querying the digits for their identifiers\(^4\).

![Diagram](image)

**Figure 4.13** Reconstruction of a MDT ladder.

\(^4\) An example is the requirement that in a track segment at least one hit should come from each multilayer.
The current region of activity is then used to discard all digits that do not lie (partly) within it. In normal operating mode these ROAs are the trigger roads calculated by the trigger reconstruction (see the previous section), but any other source will do just as well. The digits that pass this selection are transformed into hits. It is not possible to convert all digits at the beginning of the reconstruction, because a ROA is needed for the determination of the second-coordinate position. This is the same ROA as was used by the filter. In fact, a single region of activity is used throughout the whole local MDT reconstruction and pattern recognition. Only when all tracks that can be created have been built, is the next region retrieved (see also section 4.3).

The creation of a MDT hit starts by subtracting from the digit’s drift time the time it takes the signal to propagate along the MDT wire to the front-end electronics. To determine this time, the centre of the overlap region of the wire with the ROA is used, together with a user-definable signal speed. Subsequently, the time is corrected for the time-of-flight of the particle from the interaction point to the MDT tube. Here a straight line approximation of the track is used, which introduces an error well below the resolution of the detector. The resulting drift time is then converted to a distance by the detector to which the digit belongs, and a correction for the Lorentz angle is applied.

The error on the drift distance is determined based on the error in the r-t relation and the length of the section of the wire that falls inside the ROA. The latter has an effect on both the signal propagation time and on the time-of-flight correction.

### 4.2.2 Pattern Recognition

From the list of hits that lie inside a region of activity, all possible track segments are created by considering every combination of two hits. First a check is performed to determine whether the pair is valid, i.e.:

1. It is not part of any previously created track segment.

2. The line connecting the wire positions of the two hits points within a certain error to the interaction point. Because the hits lie so far from the origin, there is no need to take the drift circles into account in this step.

When a hit pair passes these tests, the four possible combinations of their left/right ambiguities are examined. For each, the following tasks are performed:

3. An initial track segment is created given by the formula (see figure 4.15 for an explanation of the variables used):

   \[
   \alpha = -\tan\left(\frac{x_2 - x_1}{y_2 - y_1}\right) + n \cdot \sin\left(\frac{r_1 + m \cdot r_2}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}\right) - n \cdot \frac{\pi}{2}
   \]

   \[
   b = y_1 - x_1 \tan(\alpha)
   \]

   (4.1)
with \( m = \pm 1 \) for the left/right side of the first hit, and \( n = \mp 1 \) for the left/right side of hit number 2. Furthermore, \( y_2 \) must be larger than \( y_1 \).

4. The hits that lie within a certain user-definable distance from the segment are added to it. All hits within the list are tested for their compliance to this rule, which should not be a problem as in most cases the number of hits inside a ROA is small.
When the track segment still has only two hits, it is discarded and the next one is tried.

5. A straight line is fitted through the hits as described in the next section.

From the four created track segments out of each original pair of hits only the best one is kept, where “best” is determined based on the quality of the fit and the topology of the hits. As a last step, the direction orthogonal to the drift plane is added to the segments by copying it from the current region of activity. They are then passed on to the global reconstruction of the precision chambers as described in section 4.3.

**Figure 4.14** Classes involved in the straight-line fit through a number of drift-circle hits (cf. figure 3.12).
4.2.3 Drift-Circle Fit

A straight-line fit to the drift-circle hits belonging to a track segment is implemented with the help of the COM mechanism as described in section 3.3.1. The actual fit is performed by DRT's DriftCircleFitter class through its execute method (see figure 4.14). It takes an \texttt{IUnknown} object (e.g. a track or a COM interface to a track) as its argument, which is queried for its \texttt{IDriftCircleFit} interface. When it does not exist, the fit terminates with an error. The fitter uses the \texttt{IDriftCircleFit} interface to retrieve the hit information from the track on the one hand, and to store the results of the fit on the other hand. For AMBER's TrackSegment an XTrackSegmentDriftCircleFit implementation exists to provide the required functionality.

As the drift-circle hits are to all intents and purposes two-dimensional, so is their fit. The straight-line track segment is therefore given by

\[ y = \tan(\alpha) \cdot x + b \]  

(4.2)

with \( \alpha \) and \( b \) the free parameters. The XTrackSegmentDriftCircleFit defines the \( x \)-axis as the pitch direction of the chamber, i.e. the global \( z \)-axis in the barrel and the radial direction in the endcaps. The \( y \)-axis is defined along the chamber's height, i.e. \( r \) in the barrel and \( z \) in the endcaps.

![Figure 4.15 Definitions of the variables used by the drift-circle fit.](image)

To determine the chi-squared of the track, a new coordinate system is chosen that lies alongside it. In this system, the distance of closest approach between a MDT wire and the track is given by the \( y_i^* \) coordinate of that wire. Therefore,
with the drift distance being a signed quantity. It is positive for the right side (in x) and negative for the left side. It is not possible to change the side of the hit during the fit, because that would compromise the stability of the algorithm: The chi-squared would have multiple minima at the various combinations of $\pm r_i (i = 1 \ldots n)$, and especially in the case of small drift distances the fit would start to oscillate between them.

The derivative of the chi-squared with respect to $b$ leads to the first of the two equations that are used to solve $\alpha$ and $b$, viz.

$$ (b \Delta - S_y) \cos \alpha + S_x \sin \alpha = -S_r $$

with

$$ S_\mu = \sum_{i=1}^{n} \frac{\mu_i}{\delta r_i} \quad S_{\mu v} = \sum_{i=1}^{n} \frac{\mu_i v_i}{\delta r_i}$$

(4.5)

The other equation follows from the derivative of the chi-squared with respect to $\alpha$, which after substituting the expression for $b$ from equation 4.4 becomes

$$ \bar{S}_{xy} \cos (2\alpha) + \frac{1}{2} (\bar{S}_{yy} - \bar{S}_{xx}) \sin (2\alpha) = \bar{S}_{xr} \cos \alpha + \bar{S}_{yr} \sin \alpha $$

(4.6)

with the constant factors defined as

$$ \bar{S}_{\mu v} = S_{\mu v} \Delta - S_{\mu} S_{v} $$

(4.7)

This equation cannot be solved analytically, and hence it must be done iteratively:

1. The value of $\alpha$ is guessed based on the centres of the first and last hits.

2. This value is substituted in the right-hand side of equation 4.6, which can be solved to give a new value for $\alpha$, viz.

$$ \alpha = -\frac{1}{2} \beta + \frac{1}{2} \arcsin \left[ \frac{\Lambda}{\sqrt{\bar{S}_{xy} + \frac{1}{4} (\bar{S}_{yy} - \bar{S}_{xx})^2}} \right] $$

(4.8)
with $\Lambda$ the result of the right-hand side of equation 4.6, and $\beta$ defined as

$$\beta = \text{atan} \left[ \frac{2S_{xy}}{S_{yy} - S_{xx}} \right] \quad (4.9)$$

3. Step 2 is repeated until $\alpha$ converges or until a user-defined maximum number of iterations has been reached. In most cases this method is found to converge within 3 to 4 steps.

The errors in the track parameters can most easily be calculated when the $(x', y')$ coordinate system is used because in it, equations 4.4 and the one leading to 4.6 can be linearized in $\alpha$ and $b$ without loss of accuracy. Rewriting them in a matrix formalism shows that the inverse of the covariance matrix is given by

$$C(\alpha', b')^{-1} = \begin{bmatrix} S_{x'x'} - S_{y'y'} + S_{y'y'} & S_{x'} \\ S_{x'} & \Delta \end{bmatrix} \quad (4.10)$$

To determine the errors at the centre of gravity of the hits, a shift is applied to the $x_i'$ values such that $S_{x'}$ becomes zero. Inverting the covariance matrix is then a trivial matter, and results in the following errors\(^{5,6}\):

$$\sigma_\alpha = \sqrt{1/(S_{x'x'} - S_{y'y'} + S_{y'y'})}$$

$$\sigma_b = \sqrt{1/\Delta} \quad (4.11)$$

The covariances are of course zero.

As an additional feature, the DriftCircleFitter is capable of removing hits from a track when their chi-squared exceeds a certain threshold. The max chi squared method can be used to set this value, and when the chi-squared of the worst hit is higher, it is removed and the $S_{\mu\nu}$ factors are updated. The advantage of the algorithm described here is that these factors do not have to be recalculated from scratch, but instead follow from the original ones by merely subtracting the contribution from the bad hit. After that, the iterative process described above can restart.

The DriftCircleFitter class continues to remove hits as long as one of them has too large a chi-squared, and the number of hits that will be left is at least equal to two. When multiple hits are removed in this fashion, there is no guarantee that the final track created by the fitter is the best one based on the original set of hits. Instead, this responsibility has been delegated to the pattern recognition. The Pattern Recognition dataview stores a history of all created

5. The errors in $\alpha$ and $\alpha'$ are the same as the two angles differ only by a constant factor.

6. The stated error in $b$ is actually the offset error perpendicular to the track. In the rotated $(x', y')$ frame these two are identical, but not so in the original coordinate system. However, it is this standard error that is used throughout the remainder of this thesis.
tracks; tracks that do not contain the hits that were removed during the fit. Hence, tracks based on those hits will be created in the subsequent processing steps of the pattern recognition. This means that in the end all possible track segments have been built by it, after which a filter can be applied to select only the best one(s).

### 4.3 Global Reconstruction

After having reconstructed the regions of activity from the trigger hits, and having used them to find the track segments in the individual precision chambers, the final step in the reconstruction is to match these segments together and to build the global tracks. To this end, the pattern recognition is followed by a filter to select only those segments that pass certain cuts. The default criteria applied by this filter are defined as follows:

- If a segment crosses both multilayers, its number of hits must be higher than or equal to the number of layers in the chamber minus 1;

- If it crosses only one multilayer, it must have at least the same amount of hits as the number of layers in that multilayer.

The segments created by the MDT ladders that belong to the same detector layer\(^7\) are grouped together, and these layers form the inputs to the global reconstruction algorithm (see figure 4.16). The track builder is responsible for matching the various track segments and adding them to a track skeleton. This process is started with the segments in the outermost layers, because they have the lowest occupancy. The track builder is capable of applying certain

![Figure 4.16 Global reconstruction algorithm.](image)

7. A cylinder (BIS, BIL, etc.) in the barrel and a wheel (EIS, EIL, etc.) in the endcaps (see the glossary in appendix C).
The Global Fit

The global fit of the precision and trigger hits is based on an iterative algorithm in which least-squared corrections are applied to the track parameters via the first derivatives of the residuals of each track constituent [57]. For \( m \) independent measurements and \( n \) track parameters \( p \), the track residuals vector \( r \) and derivative matrix \( D \) are defined as

\[
r = \begin{bmatrix}
r_1 \\
\vdots \\
r_m \\
\end{bmatrix}, \quad D = \begin{bmatrix}
\frac{\partial r_1}{\partial p_1} & \frac{\partial r_1}{\partial p_n} \\
\vdots & \vdots \\
\frac{\partial r_m}{\partial p_1} & \frac{\partial r_m}{\partial p_n}
\end{bmatrix}
\]

The change to the track parameters is then given by

\[
\delta p = (D^T \cdot D)^{-1} \cdot (D^T \cdot r)
\]

with the covariance matrix equal to

\[
C = (D^T \cdot D)^{-1}
\]

The five independent parameters for the reconstruction of tracks in the ATLAS muon spectrometer are the \( R \phi \) and \( z \) positions, the \( \phi \) and \( \theta \) angles, and the inverse of the transverse momentum \( 1/p_T \), all at some fixed radius \( R \).

Since the inhomogeneity of the magnetic field prevents the analytic calculation of the residual and derivative matrices, the tracks must be propagated through the magnetic field to the position of each individual track constituent. They are therefore sorted in increasing distance along the track. From the position of closest approach of the track to a constituent, the residual can be calculated directly, while the derivatives can be retrieved from the transport matrix as it was created by the magnetic field propagation (cf. section 3.3.2).

The calculation of these positions of closest approach is performed by a list of so-called fit modules created out of the constituents of a track (see figure 4.17). In the case of the MDT hits, the tracks are propagated to the wire plane of the layer to which the hit belongs. Then a

---

8. In addition to a hit, a constituent can also be a vertex, a multiple scattering point, etc. (cf. section 3.3.1).
straight-line approximation is used to determine the point of closest approach, after which the residual is determined based on the drift time of the hit and the track's coordinate along the wire. For the trigger clusters, the point of closest approach is determined in the plane defined by the two directions in which it has the largest extent, i.e., the directions of its member strips and/or wire groups. And as a cluster measures a track’s position in two independent directions, it adds two rows to the residual and derivative matrices, one for each of these directions.

The resulting list of fit modules is controlled by the ModuleFitter class. It operates on a track through the IModuleFit COM interface (cf. section 3.3.1) and repeatedly executes the modules until the fit either converges or until a certain number of iterations has been performed. This convergence is determined by a SolverModule object, which is automatically added to the end of the module list. In its execute method it calculates the chi-squared, solves equation 4.13 and updates the track parameters and their covariance matrix.

As a starting point for the fit, the track’s position and direction are copied from the track segment that was added last, which as a result of the definition of the Track Builder (see figure 4.16) is the innermost segment. Except for possible misalignments of the chambers, and the bending of very low energy tracks, these values accurately define the first four parameters of the global track.

The fifth parameter, i.e., the magnitude of the particle’s momentum, cannot be determined from that one track segment. Instead, it is estimated based on the relative position and orientation of all segments, assuming a helical trajectory in a perfect and constant toroidal field. In the
barrel a magnetic field value equal to the average of the values at the centres of the track segments is taken. In the endcaps where the toroids lie between the stations, a constant field of 1 Tesla between the inner and middle stations is assumed. A different method would be to use a lookup table indexed on the track’s $\eta$ and $\phi$ coordinates, and using its sagitta. In any case, it turns out that the track fit is to a large extent independent of the accuracy in the initial guess of the momentum, and in most cases converges after 3 to 7 steps.