VERTEX FINDING AND B-TAGGING ALGORITHMS FOR THE ATLAS INNER DETECTOR

A. Wildauer, Leopold-Franzens-University, Innsbruck, Austria*
F. Åkesson, CERN, Geneva, Switzerland†

Abstract

For physics analysis in ATLAS, reliable vertex finding and fitting algorithms are important. In the harsh environment of the LHC (∼23 inelastic collisions every 25 ns) this task turns out to be particularly challenging. One of the guiding principles in developing the vertexing packages is a strong focus on modularity and defined interfaces using the advantages of object oriented C++. The benefit is the easy expandability of the vertexing with additional fitting strategies integrated in the Athena framework.

Various implementations of algorithms and strategies dedicated to primary and secondary vertex reconstruction using the full reconstruction of simulated ATLAS events are presented.

Primary and secondary vertex finding is essential for the identification of b-jets in a reconstructed event. Results from a modular and expandable b-tagging algorithm are shown using the presented strategies for vertexing.

INTRODUCTION

The new event data model of the ATLAS experiment allows and encourages the development and implementation of reusable code and objects. In this report vertexing tools fully integrated in the ATLAS Athena framework [1] and compatible with the ATLAS Event Data Model [2] are presented. Two use-cases are discussed to display the chain from track reconstruction to analysis objects. The first use-case is the reconstruction of the primary vertex in an event and the second is the finding of b-flavored jets (b-tagging).

THE ATLAS EVENT DATA MODEL

The ATLAS Experiment will collect up to one Peta-Byte of data per year. The vast amount of data prohibits the distribution of raw data to all collaborators. To enable physicists to analyse the data at remote sites, two additional layers of datasets will be produced:

- The Event Summary Data (ESD) contains the detailed output of the detector reconstruction and will be produced from the raw data. The target size for the ESD is 500 kB per event.
- The Analysis Object Data (AOD) contains a summary of the reconstructed event which will be sufficient for most physics analyses. Several tailor made streams of AOD's are foreseen for the different needs of the physics community. The AOD can be produced from the ESD and thus makes it unnecessary in general to navigate back and process the raw data. The target size for the AOD is 100 kB per event.

The final content of ESD and AOD will be defined taking detailed feedback from the users and the physics community into account.

VERTEX FINDIND AND FITTING

Vertex finding and fitting is important for many physics analyses. A common vertexing package which makes use of the new ATLAS Event Data Model has been developed. As a first example application a primary vertex finder has been written. Many physics analyses rely on a well reconstructed primary vertex in order to reconstruct the physics process of the event. The primary vertex is therefore a core part of a reconstructed event and will be included in the ESD as well as the AOD.

Implementation

The vertex fitting methods are implemented as Athena tools. This enables a client to execute the fitter on demand and as many times as needed. In addition, several instances with different settings can be used simultaneously. The C++ implementation follows an inheritance structure which makes it possible to add new fitting methods easily and to be used by the client without the need of any change in the client code (Figure 1). The general strategy is that the base class takes care of framework related issues and provides a common interface, whereas the derived classes only need to implement the actual mathematical fitting method.

The Billoir Method

Within this new common vertexing framework two vertex fitting methods have been implemented. Both are based on the "Billoir" method [3]. It makes use of a local parameterisation of the track parameters $\mathbf{q} = (q_1, q_2, q_3, q_4, q_5)$ at a fixed point, e.g. the perigee, as a function of the vertex position $V(x, y, z)$ and the momenta $\mathbf{p}_i$ of the tracks:

$$F(V^0 + \delta V, \mathbf{p}_i^0 + \delta \mathbf{p}_i) = F(V^0, \mathbf{p}_i^0) + \mathbf{D}_i \delta V + \mathbf{E}_i \delta \mathbf{p}_i \quad (1)$$

With this linearisation of the track parameters the following χ^2 is minimised:

$$\chi^2 = \sum_i (\mathbf{q}_i^{meas} - F(V, \mathbf{p}_i))^T \mathbf{W}_i (\mathbf{q}_i^{meas} - F(V, \mathbf{p}_i)) \quad (2)$$

---

* andreas.wildauer@cern.ch
† fredrik.akesson@cern.ch

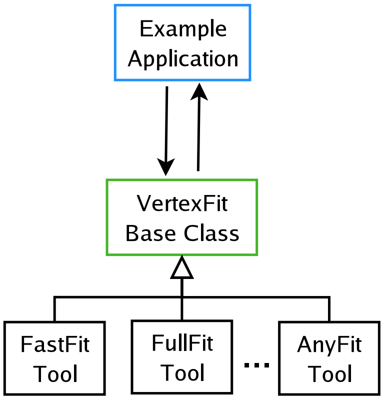

Figure 1: The inheritance structure of the vertex fitting methods ensures common interfaces for all applications in need of fitting vertices.

In the *fullfit* method the track parameters are re-evaluated with the constraint that they pass through the fitted vertex. In the *fastfit* method only the vertex position is fitted and the track parameters are constant. Mathematically this means that for the *fastfit* method the last term of eq. 1 is zero.

The track perigee parameters used in ATLAS are:

- d_0 - The transverse impact parameter, the distance of the closest approach of helix to beampipe
- z_0 - The longitudinal impact parameter, the z value at the point of closest approach
- ϕ_0 - The azimuthal angle of the momentum at point of closest approach, measured in the range $[-\pi,\pi)$
- θ - The polar angle in the range $[0,\pi]$
- q/p - The charge over momentum magnitude.

The Perigee parameters are the helix parametrisation in a frame where the z-axis is parallel to the magnetic field.

Primary Vertex Finder

The primary vertex finder is an example application which makes use of the new vertex fitting package described in the previous sections. It retrieves tracks from the transient event store, performs a basic track selection (minimal requirements on p_t, number of hits, etc.) and passes them to the vertex fitting tools. The fitting method returns the vertex and a list of tracks used. With this information the vertex finder does a second track selection based on the χ^2 of the track in the fit to eliminate outliers. Tracks with $\chi^2 > 4$ contributions are removed and the vertex is refitted with the remaining tracks.

In events with pile-up the primary vertex of the physics event has to be found among other (∼23 at high luminosity) vertices originating from minimum bias events. In this case a histogram method is used to find clusters of tracks along the z-direction. The tracks in the clusters are subsequently fitted to separate vertices using the primary vertex finder. The vertex with the highest p_t-sum of the fitted tracks is defined to be the primary vertex.

Results

The resolution of the reconstructed vertex in events without pile-up from the process $\mathrm{WH}(120)\rightarrow\mu\nu\mathrm{u}\bar{\mathrm{u}}$ in z-direction is shown in Fig. 2. The resolutions in the x and y directions for this process are about 14 μm and 37 μm in z-direction.

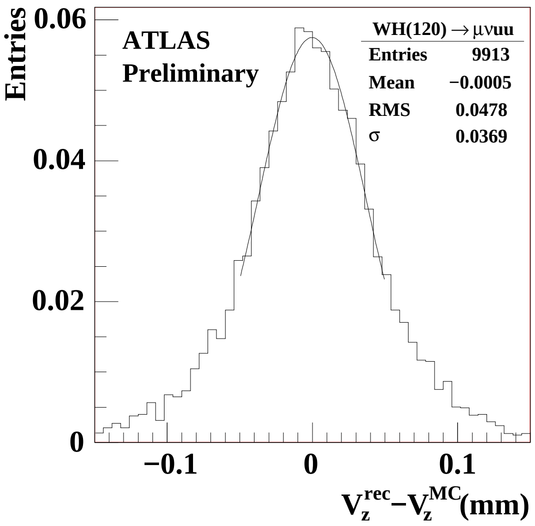

Figure 2: The resolution of the reconstructed primary vertex in events from the process $\mathrm{WH}(120)\rightarrow\mu\nu\mathrm{u}\bar{\mathrm{u}}$ without additional pile-up events in the z-direction.

In Fig. 3 the resolution of the reconstructed primary vertex from the vertex fit in $\mathrm{H}(130)\rightarrow\ell\ell\ell\ell$ events with high luminosity pile-up is shown. The resolution of the reconstructed vertex in high-luminosity pile-up events differ little from the ones calculated in non-pile-up events. The resolutions in the x and y directions for this process are about 16 μm and 46 μm in z-direction. In 88 % of the cases the reconstructed primary vertex is within the three-dimensional distance of 0.5 mm of the simulated true primary vertex of the high luminosity pile-up physics event.

B-TAGGING

A further important use case of the new vertexing package is the tagging of jets with decaying B-mesons (b-tagging). Due to the relatively long life-time of the b-flavored mesons the decay will on average take place a few mm from the primary vertex. The signature in the detector will be a secondary vertex which can be reconstructed with the tools presented here. In addition to the secondary vertex finding other characteristics of jets, such as jet-kinematics, semi-leptonic decays of the B-mesons, and impact parameters can be used to separate b-flavored jets from jets with lighter flavors. The resulting b-tagging objects will be part of the AOD production and will support physicists searching for processes with final state b-jets.

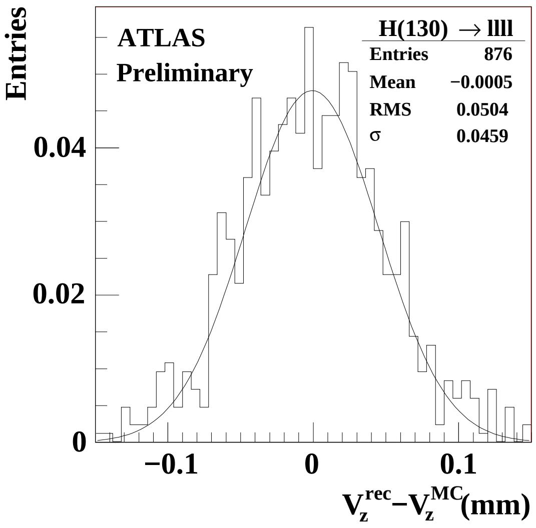

Figure 3: The resolution of the reconstructed primary vertex in events from the process $H(130) \rightarrow \ell\ell\ell\ell$ with additional high-luminosity pile-up in the z-direction. The reconstructed primary vertex with the highest sum of track-p_t is defined to be the primary vertex. The correct primary vertex of the physics event is found in 88 % of the cases.

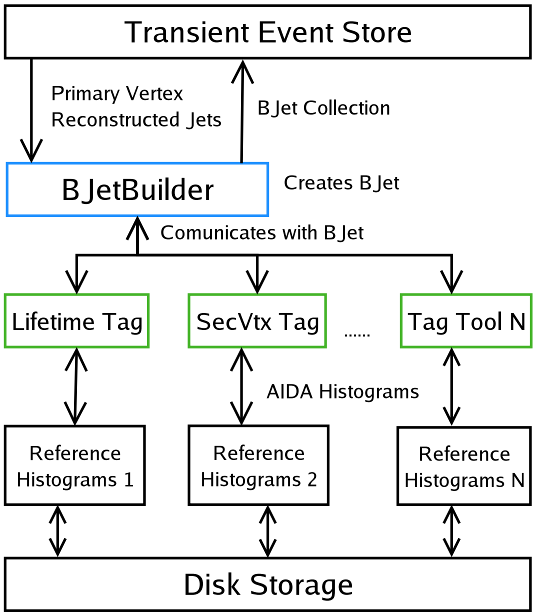

Figure 4: The schematic information flow for the b-tagger. The BJetBuilder algorithm reads the reconstructed primary vertex and jets from the transient event store, creates a BJet which is passed on for tagging to the different Athena tagging tools.

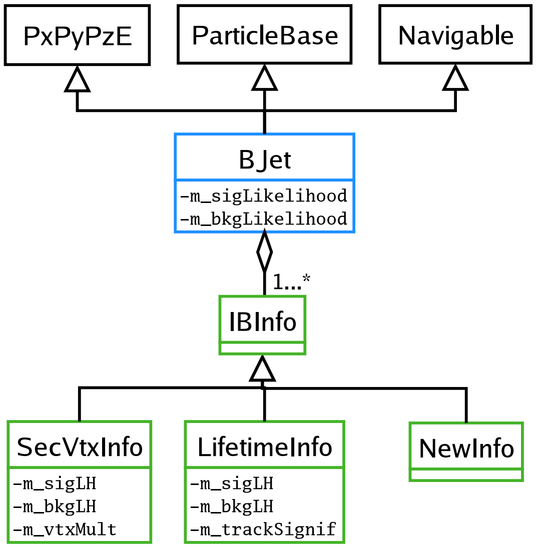

Figure 5: The inheritance structure of the BJet ensures common interfaces with the other AOD objects. The BJet has a vector of IBInfo objects which contain the tagging information from the various tagging tools.

Implementation

The b-tagging algorithm ("b-tagger") is implemented as an Athena algorithm using different and user-selectable Athena tools which are specialised tags for different characteristics. The BJetBuilder algorithm retrieves reconstructed jets and the primary vertex from the transient event store, performs a basic quality selection of the tracks and creates a BJet object (Fig. 5) for each jet to be tagged. Subsequently it calls the different tagging tools specified by the user. The schematic information flow of the b-tagging procedure can be seen in Fig. 4. After the execution of all tools the results are combined with a likelihood method [4] and the collection of BJets is written back to the transient event store.

The tag tools inherit from a common base-class with a well defined interface. The modularity is granted since for every new tag an adequate info object to store the specific information of this tag has to be written. All info objects inherit from an abstract IBInfo base class which provides a common interface. After each tag the tool itself adds this info object to the vector of IBInfo objects which the BJet contains.

Internally the tools can use likelihood or neural network methods to calculate the tag. Reference histograms can, for example, be read in as AIDA histograms.

The BJet objects are included in the AOD stream and written to the AOD Pool file at the end of the event loop.

Results

To display the functionality of the full chain from reconstructed tracks to b-tagging of jets two tools were implemented and used.

- Lifetime Tag Tool - Uses the lifetime signed impact parameters divided by their errors from the tracks in the jet, and combines the values with a likelihood method (Fig. 6).

- Secondary Vertex Tool - A build-up method of finding secondary vertices in the jets is used. One of the used discriminating variables after the fit is shown in Fig. 7. All discriminating variables are internally combined with a likelihood method.

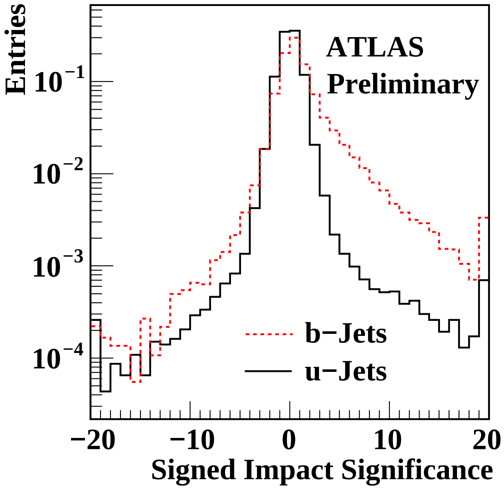

Figure 6: The lifetime signed impact parameter divided by its error as produced by the lifetime tag tool.

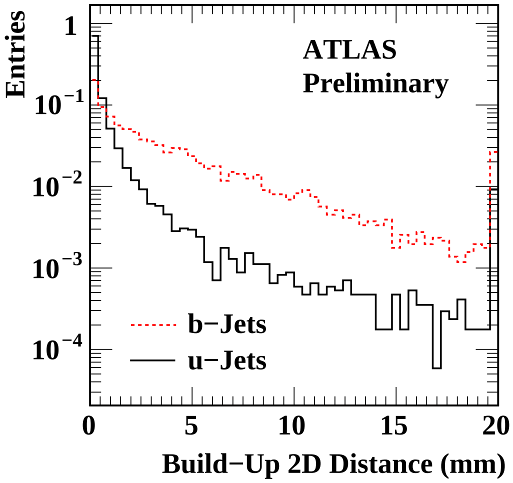

Figure 7: The distance of the secondary vertex to the primary vertex in the $r - \phi$ plane.

The subsequent combination of the likelihoods by the b-tagging algorithm gains the results as shown in Fig. 8. The efficiency at a given likelihood cut for b-jets and light-flavored jets is presented in Fig. 9.

CONCLUSIONS

We have shown that within the ATLAS Athena framework it is possible to design and implement functional chains from tracking to reconstruction of physical objects.

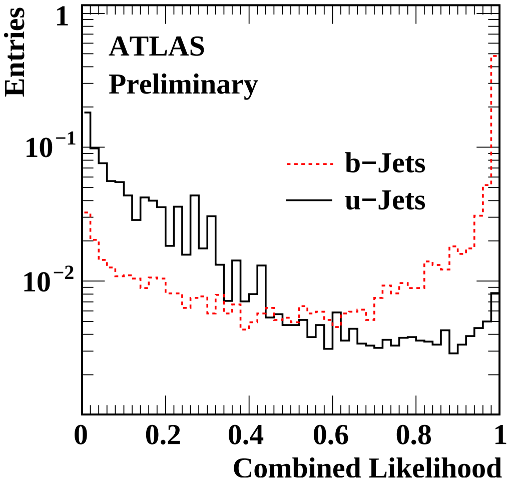

Figure 8: The resulting likelihood output of the b-tagging for b-jets (dashed line) and light-flavored jets (solid line).

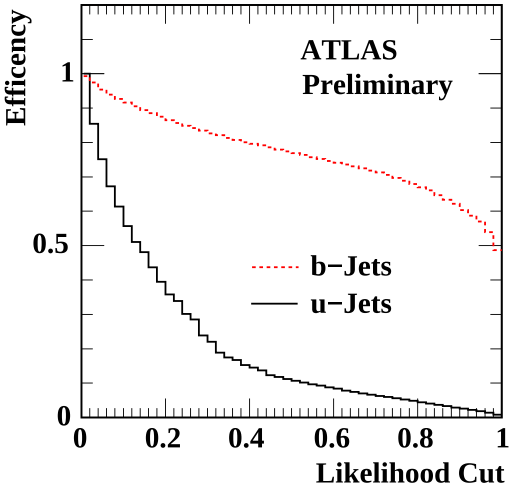

Figure 9: The efficiency versus likelihood cut for b-jets (dashed line) and light-flavored jets (solid line).

New modular and expandable vertexing and b-tagging algorithms have been developed and prototype implementations of vertex fitting and b-tagging have been introduced. The software uses the new ATLAS Event Data Model which supports the development of common tools. Preliminary results the vertex and b-tagging software have been presented.

REFERENCES

[1] http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/architecture/index.html.

[2] E. Moyse and F. Åkesson, “Event Data Model in ATLAS”, Presented at CHEP 2004, Interlaken, Switzerland, September 27th - October 1st, 2004

[3] P. Billoir and S. Qian, Nucl. Instr. and Meth. A311 (1992) 139-150.

[4] R. J. Barlow, “Statistics”, John Wiley & Sons, New York (1989).