Using Legacy ATLAS C++ Calibration Tools in Modern Columnar Analysis Environments

Matthew Feickert[1], Nikolai Hartmann[2], Lukas Alexander Heinrich[3], Alexander Held[1], Evangelos Kourlitis[3], Nils Erik Krumnack[4], Giordon Holtsberg Stark[5], Matthias Vigl[3] and Gordon Watts[6]

[1]University of Wisconsin-Madison

[2]Ludwig Maximilians Universitat

[3]Technical University of Munich

[4]Iowa State University

[5]SCIPP, UC Santa Cruz

[6]University of Washington

E-mail: matthias.vigl@tum.de

Abstract. The ATLAS experiment at the LHC relies on crucial tools written in C++ to calibrate physics objects and estimate systematic uncertainties in the event-loop analysis environment. However, these tools face compatibility challenges with the columnar analysis paradigm that operates on many events at once in Python/Awkward or RDataFrame environments. Those challenges arise due to the intricate nature of certain tools, as a result of years of continuous development, and the necessity to support a diverse range of compute environments. In this work, we present the ATLAS R&D efforts to adapt these legacy tools to be used in both event-loop and columnar environments with minimal code modifications. This approach enables on-the-fly calibration and uncertainties calculations, minimizing the reliance on intermediate data storage. We demonstrate the functionality of this approach in a Python Jupyter notebook that reproduces a toy Z-boson-peak analysis.

1 Introduction

The ATLAS experiment [1] has traditionally relied on *event-loop* processing for analysing High Energy Physics (HEP) data, which typically contains variable-length lists of reconstructed objects and features per event: e.g. tracks, jets, leptons etc. On the other hand, the scientific Python ecosystem offers powerful tools that typically work with rectilinear arrays and are thus able to process multiple events at once. While the ROOT [2] data format, currently employed for DAOD_PHYSLITE [3], supports columnar reading of these single-jagged structures by storing event offsets separately from the data, *ATLAS calibration tools* are designed to operate as part of the ATLAS software stack, using the data access layer from this event-wise framework. In recent years, there has been a notable shift towards the use of data frames (tabular data structures) and array programming APIs for data analysis, driven by user demand. In the field of HEP, two common columnar environments have emerged: RDataFrame [4] and the Scientific Python ecosystem. In this sense, *columnar analysis* can be thought of as the HEP

equivalent of *array programming*, where the goal is to benefit from compact and expressive syntax for accessing, manipulating, and operating on data in arrays. With the advent of these *columnar data analysis* paradigms, sketched in Figure 1, there is a need to adapt the legacy correction tools to work efficiently in modern environments that leverage Python and columnar data processing libraries [5], such as Awkward [6] and RDataFrame. In practical terms, this means that correction tools must be able to operate on multi-object/event arrays, or columns. Additionally they need to be fast and simple enough to enable interactive use of these tools for on-the-fly analysis on PHYSLITE data. This would eliminate the need to write out intermediate n-tuples and systematic variations to disk, significantly reducing data storage requirements which is especially important for the HL-LHC era and beyond [7] [8].

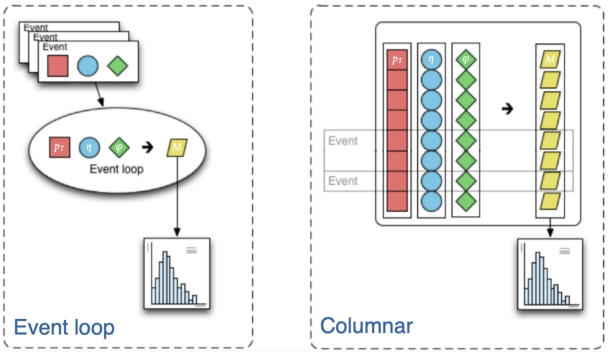

Figure 1: Comparison of Event Loop and Columnar Data Processing: the left diagram illustrates the traditional event loop processing where each event is processed sequentially, while the right diagram depicts the columnar data processing approach where data for each variable is stored in contiguous arrays, allowing for parallel processing. [9]

2 PHYSLITE data format

The PHYSLITE data format is designed as a common, monolithic, and unskimmed format, where only a selected set of variables is written out to disk. This format contains already calibrated objects that are loosely preselected, making it highly efficient and streamlined for fast analysis, with the plans of covering 80% of all ATLAS physics analyses. It will be the main format for ATLAS during HL-LHC and it's already available for both analysis and R&D purposes. The idea is to have frequent PHYSLITE productions, with the potential for up to 6-8 production cycles per year, ensuring that the data remains current and reflective of the latest experimental conditions and recommendations.

3 ATLAS calibration tools

Calibration tools are software algorithms, developed and maintained by the ATLAS collaboration, that ensure accurate measurement and calibration of various physics objects. All these tools are a result of continuous development, with each serving a specific purpose dealing e.g. with jet energy calibration, tagging efficiencies and systematic uncertainties. This group of tools, techniques, and procedures is referred to as ATLAS *combined performance* (CP). For this prototype we demonstrate a columnar conversion of the *EGamma* Scale Factor and Smearing tools [10], which are used to compute electron scale factors and momentum corrections. Given that objects in PHYSLITE are already calibrated, we are only interested in deriving systematic variations with this set of tools.

4 Columnar *Triple use* tools

ATLAS CP tools are primarily written in C++ and are designed to perform intricate and complex calculations within the event-loop analysis environment. While we want columnar CP tools to extend and enhance the analysis experience for ATLAS by leveraging columnar environments, it's also import to continue supporting event-wise processing through frameworks like *Athena* and *EventLoop*, as sketched in Figure 2. This goal should be achieved avoiding extensive re-writing of the existing CP tools but instead by converting them with minimal code modifications to allow usage in columnar frameworks.

4.1 EDM Access Library

The result of this ongoing effort is a *ColumnarProtoype* [1] in which data access is facilitated through an abstract interface, that has a compile time switch to adapt it for the different environments. In the event-wise framework it relies on the existing xAOD classes for data access. In columnar frameworks the prototype wraps raw pointers and indices directly, leading to minimal overhead for data access. Event boundaries are stored as separate offset columns, one for each object container. This model also provides an easy option for implementing additional data access modes, if needed in the future. The class interfaces and names of the prototype are chosen to match the existing xAOD classes closely: this makes it straightforward to migrate the parts of the CP tools that have a direct columnar equivalent, and focus the migration effort on the few places that need deeper changes to work in columnar environments.

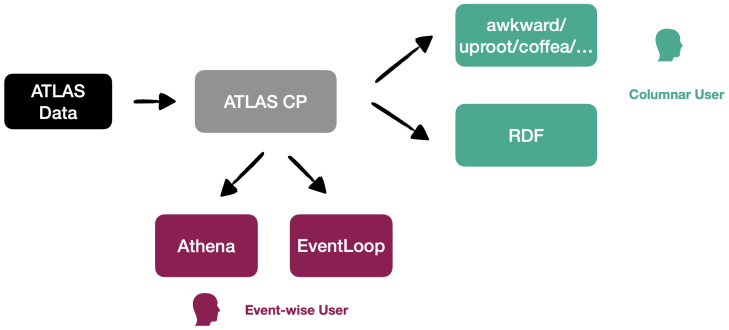

Figure 2: Schematic view of the event-wise and columnar data processing environments ATLAS CP tools need to support.

4.2 Python bindings

After building the C++ *ColumnarProtoype* infrastructure for tools conversion, we wrap the ATLAS CP tools in a user-friendly Python interface. This involves accessing external array data from Python, performing the necessary computations in C++, and then returning the results back to Python. This process is highly efficient, with no slowdown thanks to zero-copy operations. The Python binding [2] that glues the C++ code to the Python environment is achieved with *nanobind* [11].

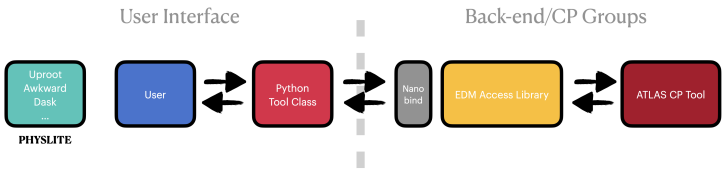

Figure 3: The user interacts with the core tools only through an API that's idiomatic within the Python ecosystem, while the intricacy of the C++ implementations are hidden and glued to the Python environment with nanobind.

5 Z→ e^+e^- demo

We demonstrate how such a prototype for columnar tools can be integrated into a simple analysis through a Python Jupyter notebook [3] that reproduces a toy Z-boson-peak. The overall analysis flow follows previous work [12] that demonstrated how to converted PHYSLITE into columnar form using Uproot [13] and Awkward Array, and presented a proof of concept for simple analysis applications. To showcase how the whole packaging would ideally be handled, we ship the two prototype tools in an *atlascp* package that

[1] https://gitlab.cern.ch/krumnack/ColumnarPrototype2/-/tree/master?ref_type=heads

[2] https://gitlab.cern.ch/gstark/pycolumnarprototype/-/tree/main?ref_type=heads

[3] https://gitlab.cern.ch/gstark/pycolumnarprototype/-/blob/py_el_tool_test/Zee_demo.ipynb?ref_type=heads

would contain all the relevant ATLAS tools in the final design. The user would then simply import tools from different CP groups from this package: in this case we import the two *EgammaTools* [4].

```
from atlascp import EgammaTools
```

These tools can then directly operate on both regular numpy or awkward arrays following the syntax in Figure 4, where we can compute e.g. the p_T correction for a chunk of electrons at once.

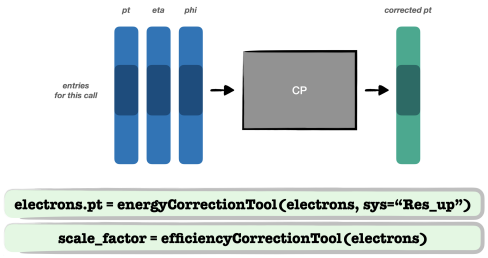

Figure 4: Columns are passed to CP tools which return new columns a chunk at a time.

In the notebook, data from PHYSLITE are loaded to awkward arrays with Uproot and the Coffea framework [9] and events are filtered by selecting e^+e^- final states. The scale factor tool is then applied to get the nominal Z-boson-peak shown in black in Figure 5, together with *up* and *down* systematic variations. These are then combined with systematics from the Smearing tool to finally get the full set of systematics shown in different colors in the same plot. While this prototype allows to scale up to multiple files using e.g. *dask-awkward*, testing and performance assessment of the dask implementation is still ongoing.

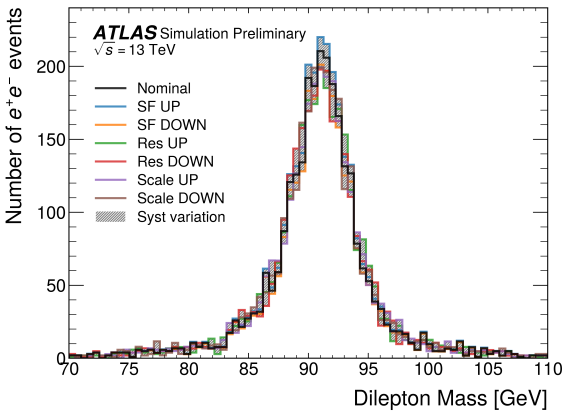

Figure 5: Distribution of the dilepton mass with different colors corresponding to the nominal distribution and various systematic variations, computed on-the-fly on PHYSLITE simulation with columnar CP tools prototypes. The hatched area represents the overall systematic uncertainty as the envelope of all variation. [14]

6 Conclusions

We demonstrate two EGamma tools successfully operating in a columnar fashion within a Python environment, computing corrections and uncertainties on-the-fly directly on PHYSLITE. This effort was part of the prototyping and review phase of the columnar data operations in ATLAS, which is anticipated to conclude by the end of 2024. Development and discussion on the precise technical pathway for developing columnar CP tools is currently ongoing.

[4] https://gitlab.cern.ch/gstark/pycolumnarprototype/-/blob/py_el_tool_test/atlascp/EgammaTools.py?ref_type=heads

References

[1] ATLAS collaboration. The ATLAS Experiment at the CERN Large Hadron Collider. *JINST*, 3:S08003, 2008.

[2] R. Brun and F. Rademakers. ROOT: An object oriented data analysis framework. *Nucl. Instrum. Meth. A*, 389:81–86, 1997.

[3] Jana Schaarschmidt, James Catmore, Johannes Elmsheuser, Lukas Heinrich, Nils Krumnack, Serhan Mete, and Nurcan Ozturk. PHYSLITE - A new reduced common data format for ATLAS. *EPJ Web Conf.*, 295:06017, 2024.

[4] Danilo Piparo, Philippe Canal, Enrico Guiraud, Xavier Valls Pla, Gerardo Ganis, Guilherme Amadio, Axel Naumann, and Enric Tejedor. RDataFrame: Easy parallel ROOT analysis at 100 threads. *EPJ Web Conf.*, 214:06029, 2019.

[5] Charles R. Harris et al. Array programming with NumPy. *Nature*, 585(7825):357–362, 2020.

[6] Jim Pivarski, Peter Elmer, and David Lange. Awkward Arrays in Python, C++, and Numba. *EPJ Web Conf.*, 245:05023, 2020.

[7] Evangelos Kourlitis. HL-LHC and Beyond Computing Challenges. *PoS*, LHCP2023:112, 2024.

[8] Johannes Elmsheuser et al. Evolution of the ATLAS analysis model for Run-3 and prospects for HL-LHC. *EPJ Web Conf.*, 245:06014, 2020.

[9] Nicholas Smith et al. Coffea: Columnar Object Framework For Effective Analysis. *EPJ Web Conf.*, 245:06012, 2020.

[10] ATLAS collaboration. Electron and photon energy calibration with the ATLAS detector using LHC Run 2 data. *JINST*, 19(02):P02009, 2024.

[11] Wenzel Jakob. nanobind: tiny and efficient c++/python bindings, 2022. https://github.com/wjakob/nanobind.

[12] Nikolai Hartmann, Johannes Elmsheuser, and Günter Duckeck. Columnar data analysis with ATLAS analysis formats. *EPJ Web Conf.*, 251:03001, 2021.

[13] Jim Pivarski, Henry Schreiner, Angus Hollands, Pratyush Das, Kush Kothari, Aryan Roy, Jerry Ling, Nicholas Smith, Chris Burr, and Giordon Stark. Uproot, February 2024. 10.5281/zenodo.10699405.

[14] Vangelis Kourlitis, Matthias Vigl, Nils Erik Krumnack, Matthew Feickert, Giordon Holtsberg Stark, Lukas Alexander Heinrich, Gordon Watts, Alexander Held, and Nikolai Hartmann. Using Legacy ATLAS C++ Calibration Tools in Modern Columnar Analysis Environments - Poster for ACAT 2024. https://cds.cern.ch/record/2905022.