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

Matthew Feickert[a], Nikolai Hartmann[b], Lukas Alexander Heinrich[c], Alexander Held[a], Vangelis Kourlitis[c], Nils Erik Krumnack[d], Giordon Holtsberg Stark[e], Matthias Vigl[c], Gordon Watts[f] on behalf of the ATLAS Computing Activity

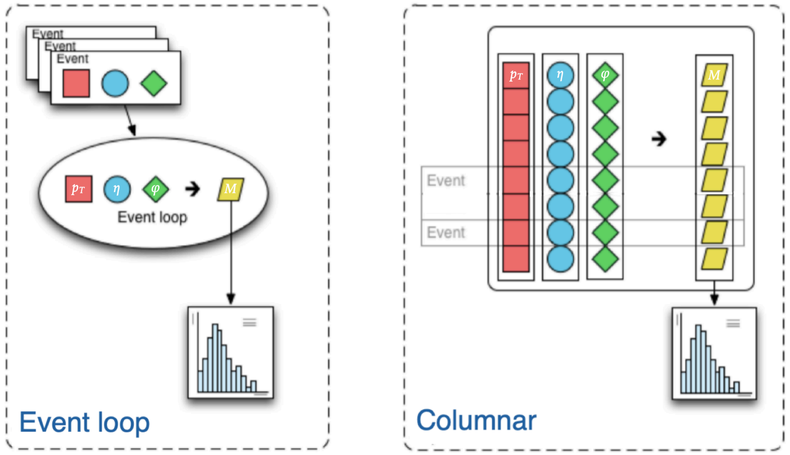
a: University of Wisconsin-Madison, b: Ludwig Maximilians Universitat, c: Technical University of Munich, d: Iowa State University, e: SCIPP, UC Santa Cruz, f: University of Washington

ACAT 2024, Charles B. Wang Center, Stony Brook University 11.03.2024 – 15.03.2024

Columnar Analysis

Columnar is the HEP term for **Array Programming**: Operators and functions abstract away the internal mechanics of looping over elements providing a **user-friendly and optimised API.**

- Integrates better with modern data science ecosystem
- Easier for new coders to start with
- Columnar analysis has already been demonstrated[1]



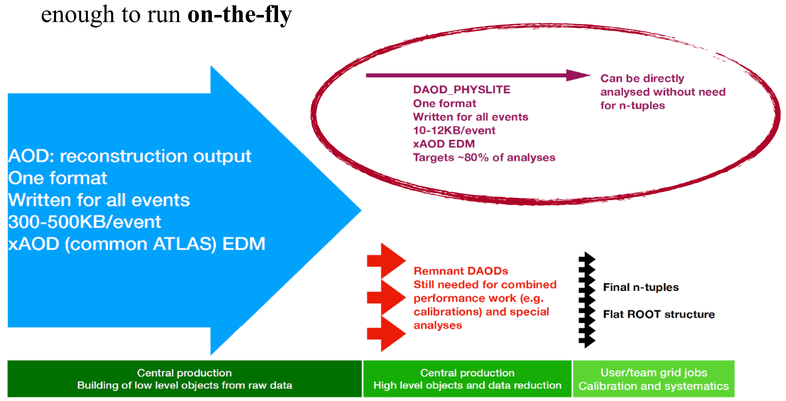
PHYSLITE[2] data format

Future format for Run 4 - used already in Run3:

- Looks much more like an analysis n-tuple while still being about as versatile as the original offline format
- Only selected set of variables is written out
- Contains already-calibrated objects

Goal: Run full analysis directly on PHYSLITE

- Avoid writing out intermediate n-tuples
- No need to write out systematic variations to disk
  - Challenge: make **systematic variation calculations** fast & simple enough to run **on-the-fly**

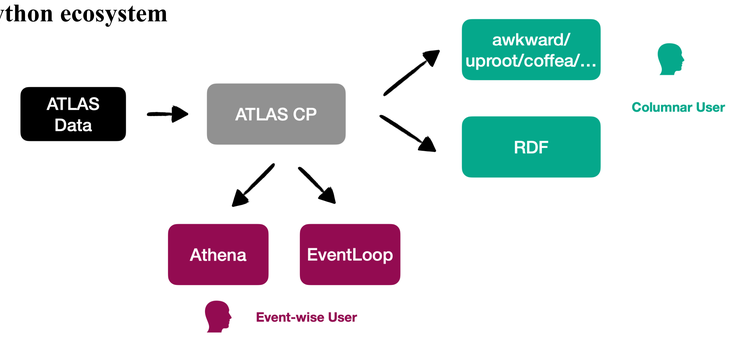


Modern CP Tools

The tools developed by the combined performance (CP) groups are provided to the ATLAS analysis teams to calibrate physics objects and estimate systematic uncertainties. These are written in C++ and perform intricate and complex calculations in the event-loop analysis environment.

In recent years, a shift towards use of data frames (tabular data structure) and array programming APIs for data analysis has been driven by users.

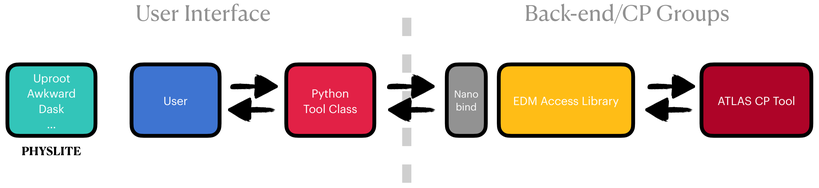
- In HEP, two common columnar environments: **RDataFrame** and **Scientific Python ecosystem**



Want to extend and improve the analysis experience for ATLAS by leveraging these columnar environments and their benefits

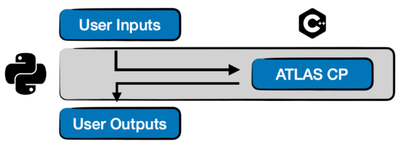
- Event-wise processing (Athena, EventLoop) will still be supported in addition to columnar environments

EDM Access Library and Python bindings



Access data through abstract interface[3]: Event Data Model (EDM) Access Library (same code in all environments).

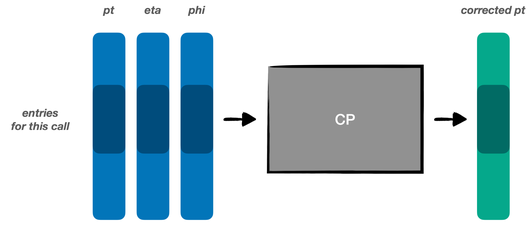
- [4]External array data from python, compute in C++ and ship the result back to python
- No slowdown due to *zero-copy* operation



$Z \to e^+e^-$ demo

Example of simple ATLAS analysis in columnar style in Python + `Dask` with **electron efficiency** and **corrections** calculated on-the-fly from PHYSLITE.

1. Load PHYSLITE to `awkward` arrays with `uproot`
2. Apply analysis selections
3. Compute systematics on-the-fly



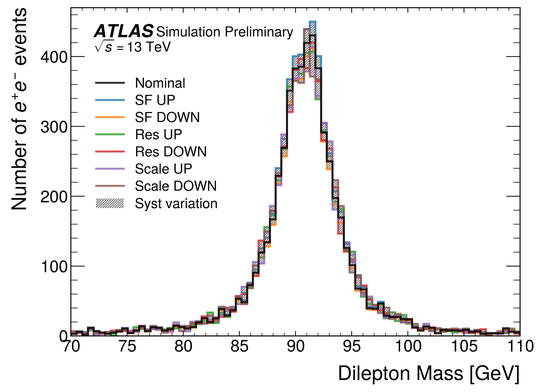

```
from atlascp import EgammaTools
```

- Can also think about having a "`pip install atlascp`" in the future

```
electrons.pt = energyCorrectionTool(electrons, sys="Res_up")
scale_factor = efficiencyCorrectionTool(electrons)
```

→ Systematic variations

4. Plot observables (can scale up with `Dask`)



References

[1]: "Columnar data analysis with ATLAS analysis formats", N. Hartmann, J. Elmsheuser, G. Duckeck (https://indico.cern.ch/event/948465/contributions/4324123/)

[2]: "PHYSLITE - a new reduced common data format for ATLAS", James Catmore, Johannes Elmsheuser, Jana Schaarschmidt, Lukas Alexander Heinrich, Nurcan Ozturk, Alaettin Serhan Mete, Nils Erik Krumnack (https://cds.cern.ch/record/2857821)

[3]: Triple-use Tools Prototype by N. Krumnack: https://gitlab.cern.ch/krumnack/columnarprototype/-/tree/master/

[4]: Python bindings by G. Stark, M. Feickert and L. Heinrich: https://gitlab.cern.ch/gstark/pycolumnarprototype/-/tree/main?ref_type=heads