Data Handling and Analysis in ATLAS

...and some other things

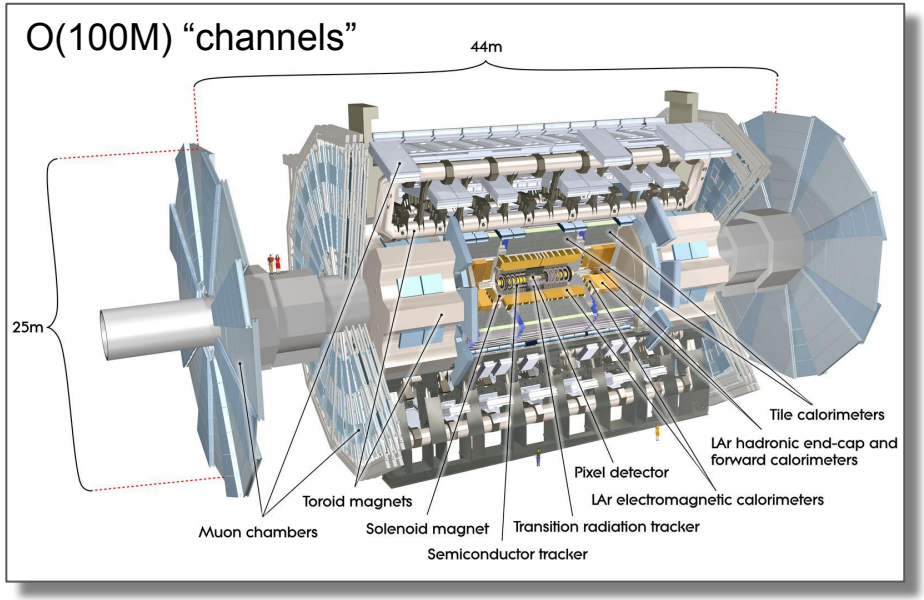
Attila Krasznahorkay

Overview

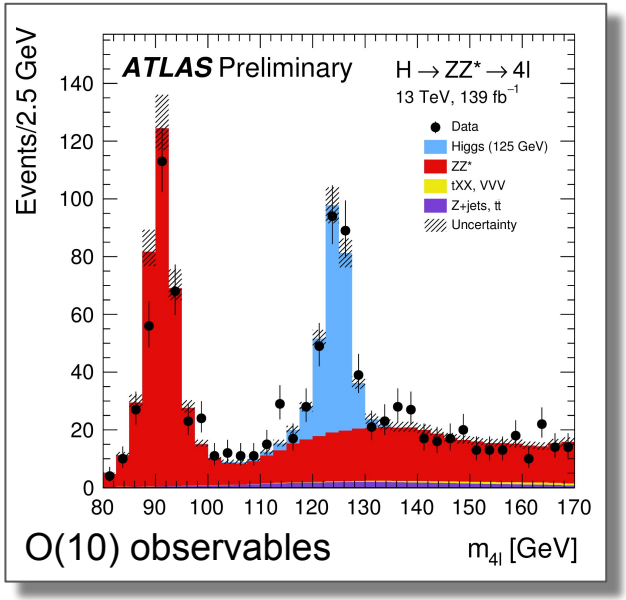
- Overview of HEP data analysis
- Status quo of HEP data analysis
- Future prospects for HEP data analysis
- Data analysis at “different” scales

HEP Data Analysis

- As a reminder: How we operate today was informed by decades of development. Most of the things that we do have a very good reason for being as they are…
- The goal is “simple”:

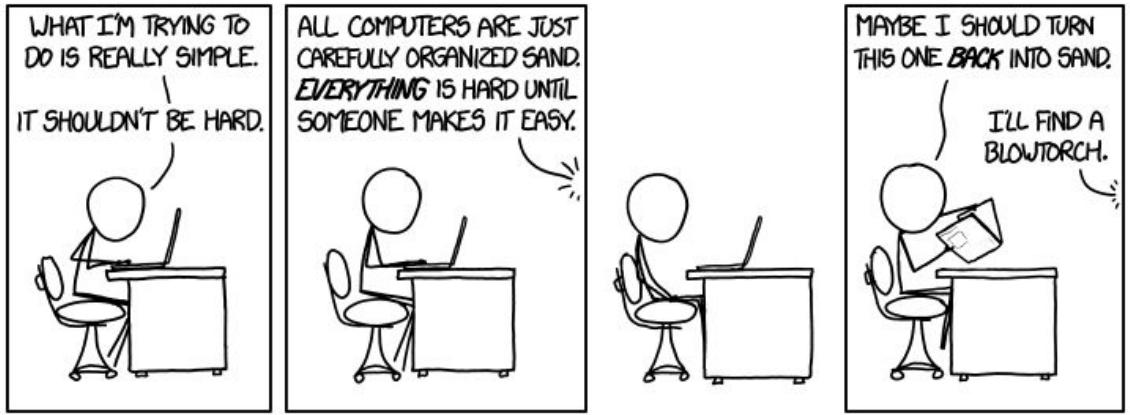


?



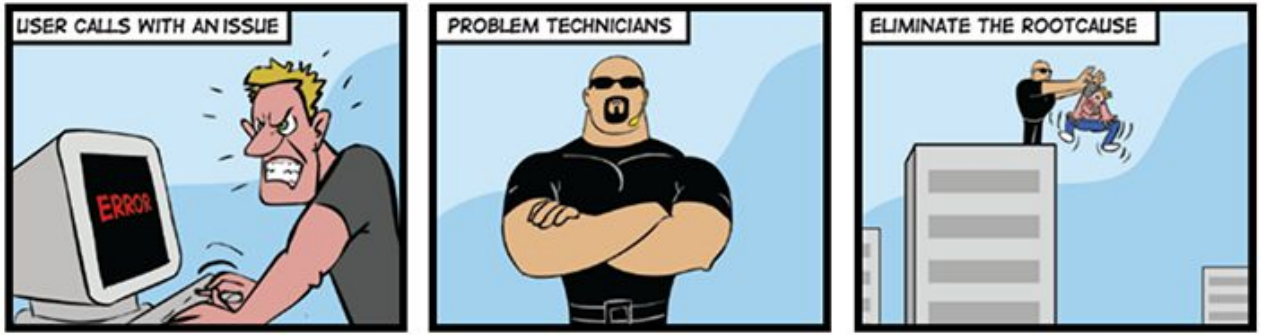
What Does a Physicist Want?

- To be able to go from ideas to results as quickly as possible
  - It should be possible to see the effects of “non-fundamental” changes to an analysis in O(15 min)
    - If it takes hours (or even more) to get updated histograms after some minor change, the analysis will suffer
- To be able to express ideas in as easily understandable of a form as possible
  - Most physicists are not interested in computer science, it must be possible for them to reason about the collected/simulated data in a way that is as close to physics as possible



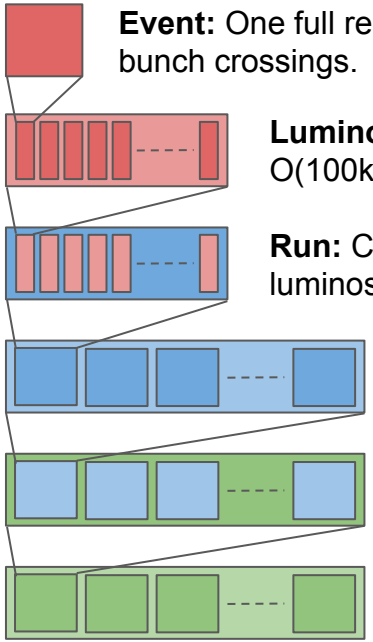
What Do “Software/IT People” Want?

- To fit into their allocated budget… 😛
  - We have finite storage and computing resources available. The physics goals must be met while fitting into these.
- This is best served by:
  - Having a well defined data analysis model
  - Having the data processing code be as bug-free as possible
  - I.e. “reasonably” limiting what people can do…



ATLAS’s (Current) Analysis Model

Data Organisation in ATLAS



Event: One full readout of the detector. May contain information from multiple p-p interactions and bunch crossings.

Luminosity block: Unit of (near) constant instantaneous luminosity. O(1) minute of data taking, O(100k) events.

Run: Continuous period of ATLAS data taking. Usually corresponds to an LHC fill. O(1k) luminosity blocks. Never resets. The run+event number uniquely identifies all ATLAS events.

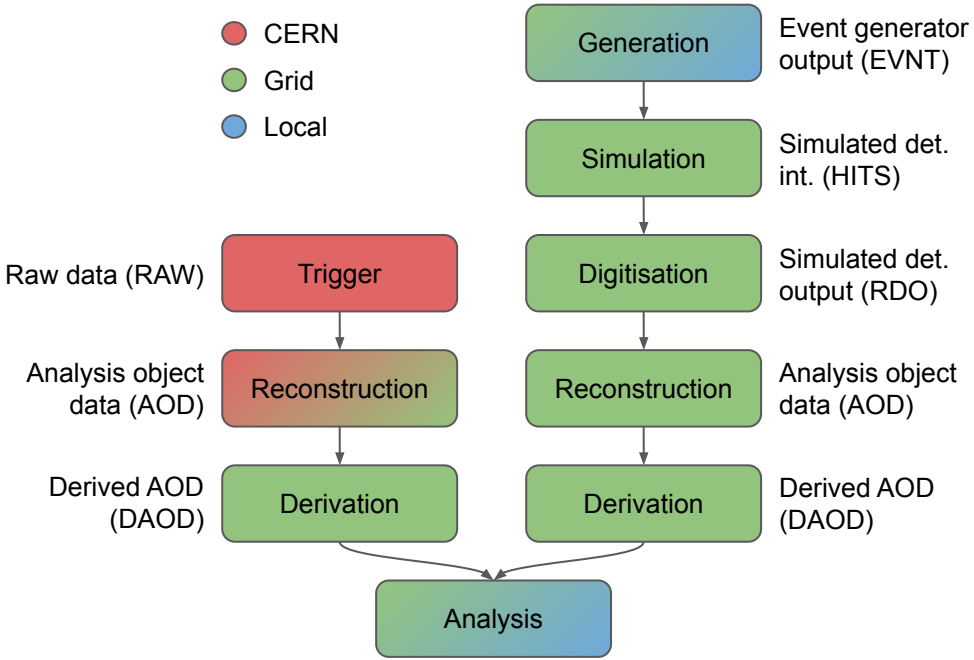
Sub-period: Group of runs taken with **very** similar conditions. O(10) runs.

Period: Group of sub-periods taken with similar conditions. O(10) sub-periods.

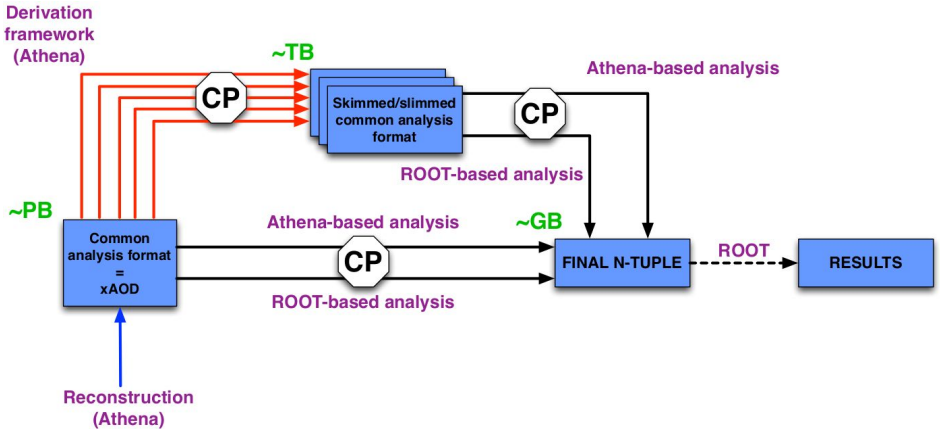
LHC run: Long periods of data taking between LHC long shutdowns.

The ATLAS Data Processing Model

- We do what all other large HEP experiments do
  - Process the data in multiple steps, handling MC simulations in the “same way” as real data after the detector simulation
- With all steps producing specific file types, designed to efficiently hold that particular kind of information

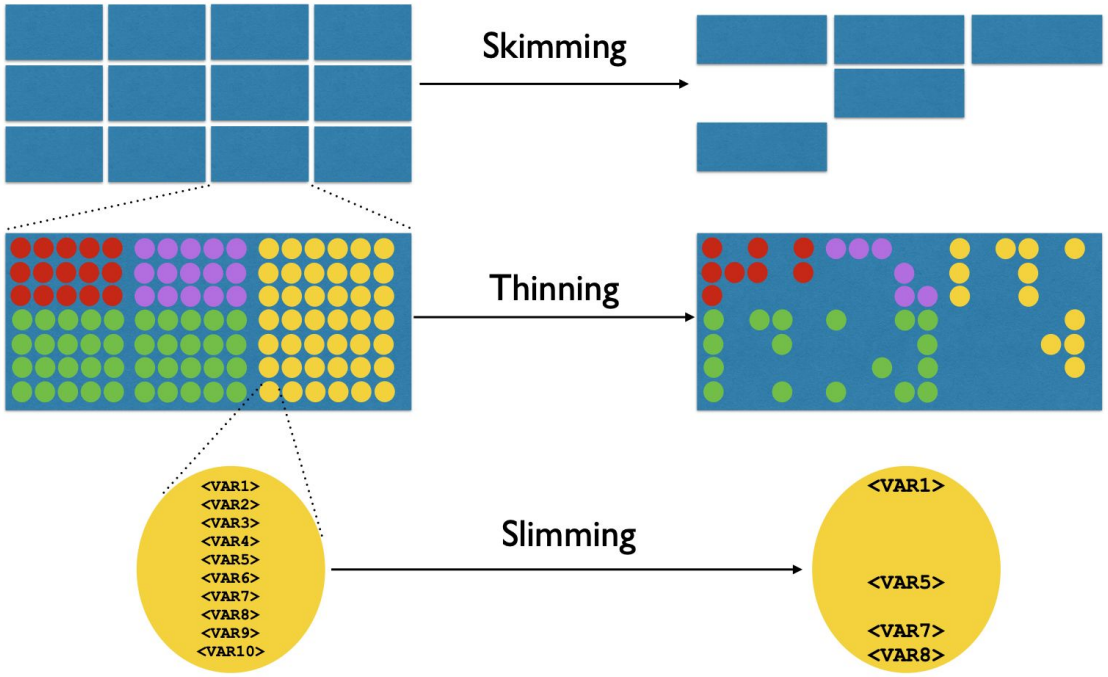


The ATLAS Analysis Model



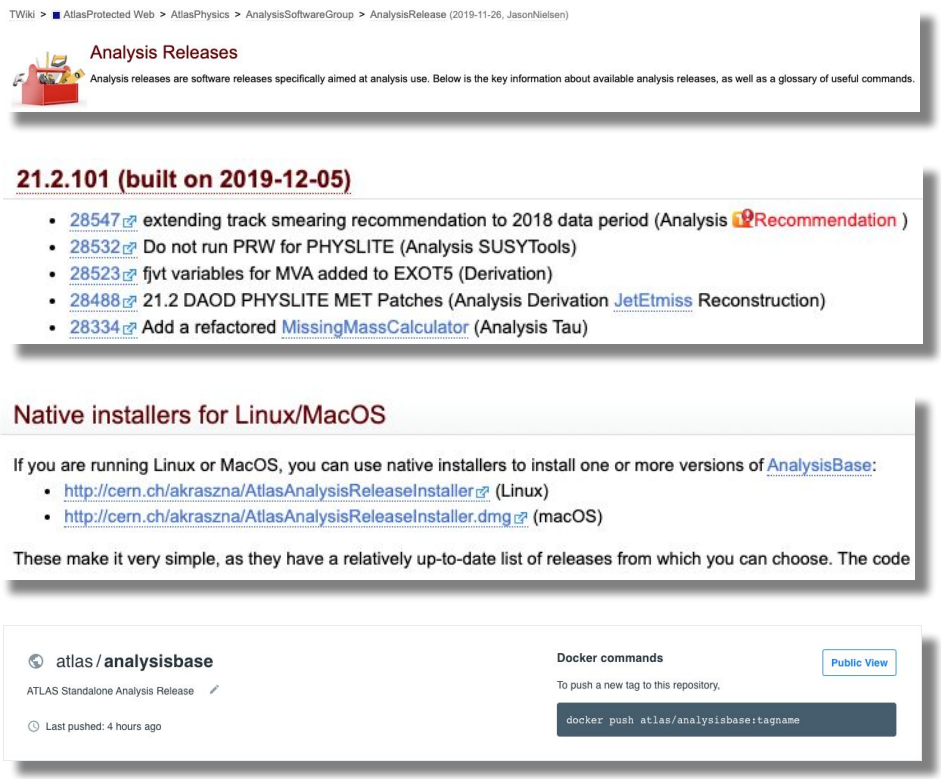
- Analysers are provided files/datasets with a single Event Data Model (EDM)
  - The same EDM can be used to create many different file formats with different amounts of information, object and event selections
- It is up to each analysis to apply the following during the processing of the (D)xAOD datasets
  - Final 4-momentum calibrations;
  - Efficiency corrections;
  - Systematic variations to the previous values.
- Most analyses write a custom small ntuple with the results of these operations

ATLAS Derivations



- Performs all types of data reduction operations possible, while potentially applying fixes to the data not possible to do during analysis

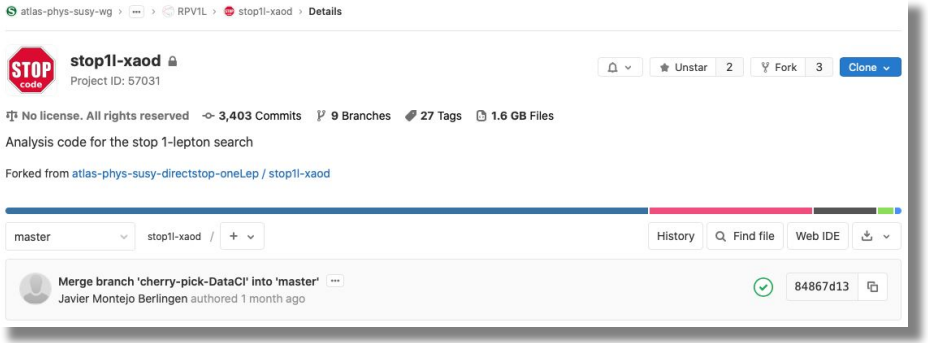
ATLAS Analysis Software Releases

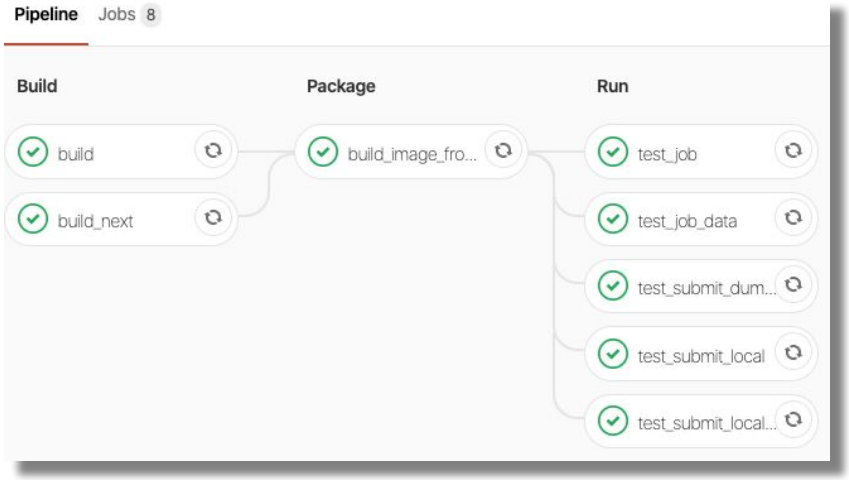


- As mentioned earlier, physicists need to perform non-trivial calibrations as part of their analysis
  - The “analysis releases” provide common code for doing so
- Built out of the same https://gitlab.cern.ch/atlas/athena repository as all offline/trigger code of ATLAS
  - But distributed in multiple additional ways on top of the “usual” installations on `/cvmfs/atlas.cern.ch` for SLC6/CentOS7

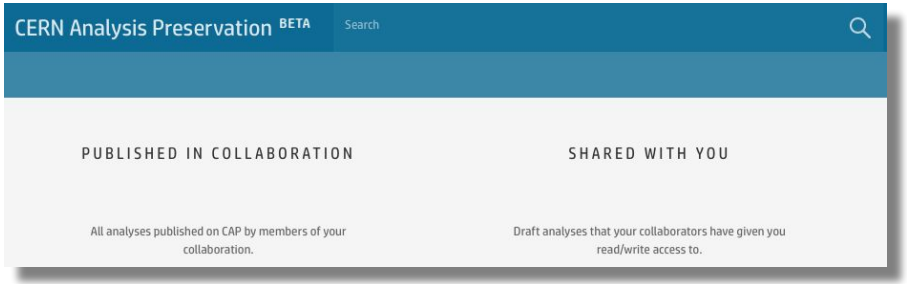
Analysis Software (Repositories)

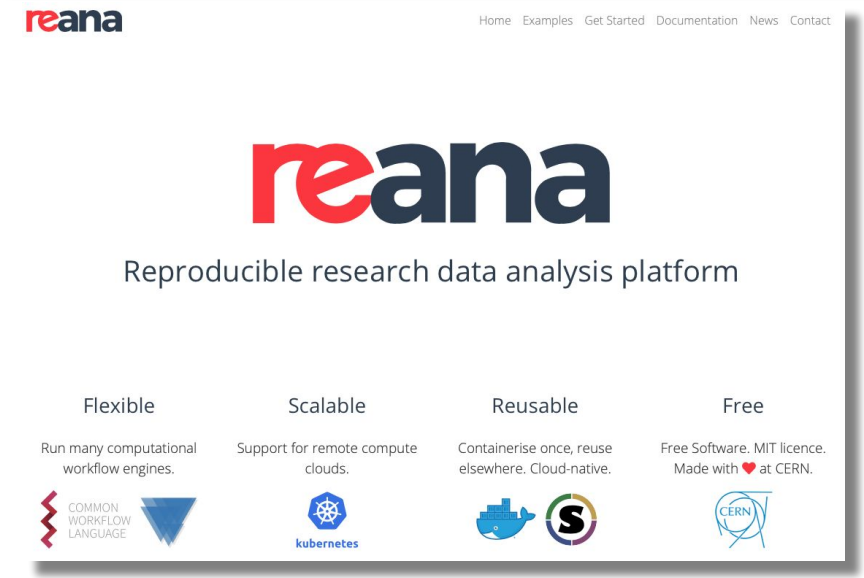
- As said earlier, the first step of most analyses produces a custom (ROOT ntuple) file format using the analysis release
  - These codes became a **lot** better since we started using Git, CMake, Docker, and a number of modern development tools
- Later steps of the analyses, operating on the custom ntuples, are usually a bit less organised
  - More on that later…





Analysis Preservation in ATLAS

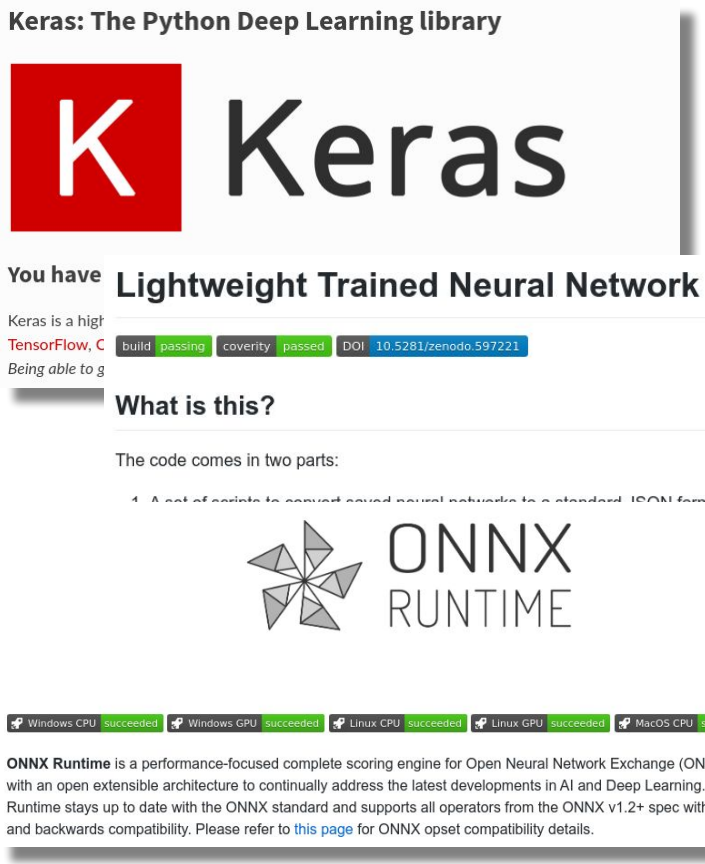




- Analysis preservation is a big topic since a while
  - We need to do a better job with archiving how all our analyses were performed, so future generations would not have to re-learn everything from scratch
- The policy in ATLAS at the moment is to
  - Save a “runnable” version of the code operating directly on DxAOD files
    - And of all the code coming after
  - Save O(1 TB) of ntuples per analysis, on which the final plotting was done
- The goal is to make REANA/RECAST a success
  - Note that this is an ATLAS+CMS project by now...

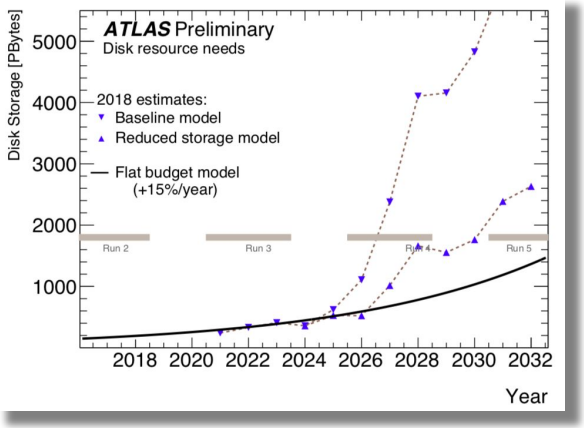
Machine Learning (in Analysis)

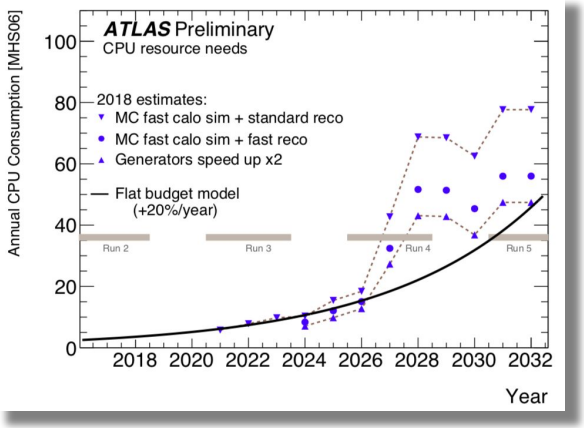
- Is a quite active field of course
- Used mainly for classification tasks currently
  - To separate different kinds of events/objects/etc.
- But is being used for regression tasks more and more
  - For instance to estimate energy correction values from a number of input variables
- Training is always happening outside of experimental frameworks
  - Even with TMVA this was always the case, and with Python based tools it didn't change
- Experimental-software-wise the challenge is implementing inference
  - Using the ML libraries directly is often not the best approach



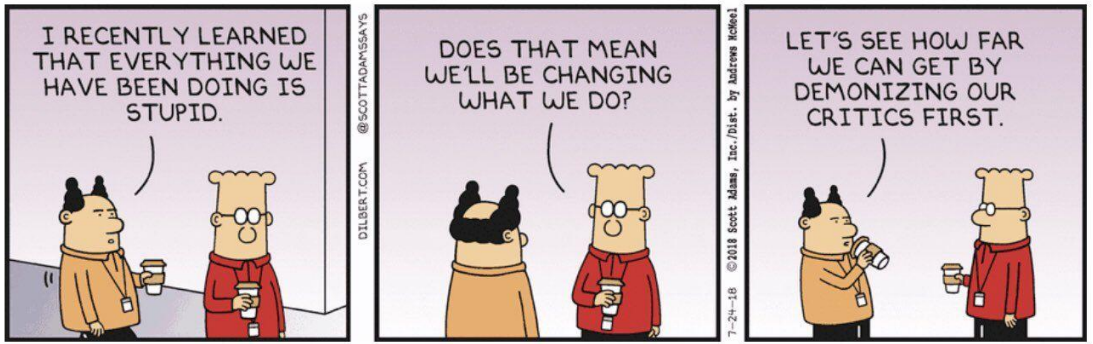
LHC Run-3 and Beyond

Coping With More Data / Complexity





- Our current analysis model is serving us well at the moment
- But it will not scale to the HL-LHC
  - In Run-3 we could still survive without larger changes, but that would only delay the issue
- So we are updating our model once again…



DAOD_PHYS

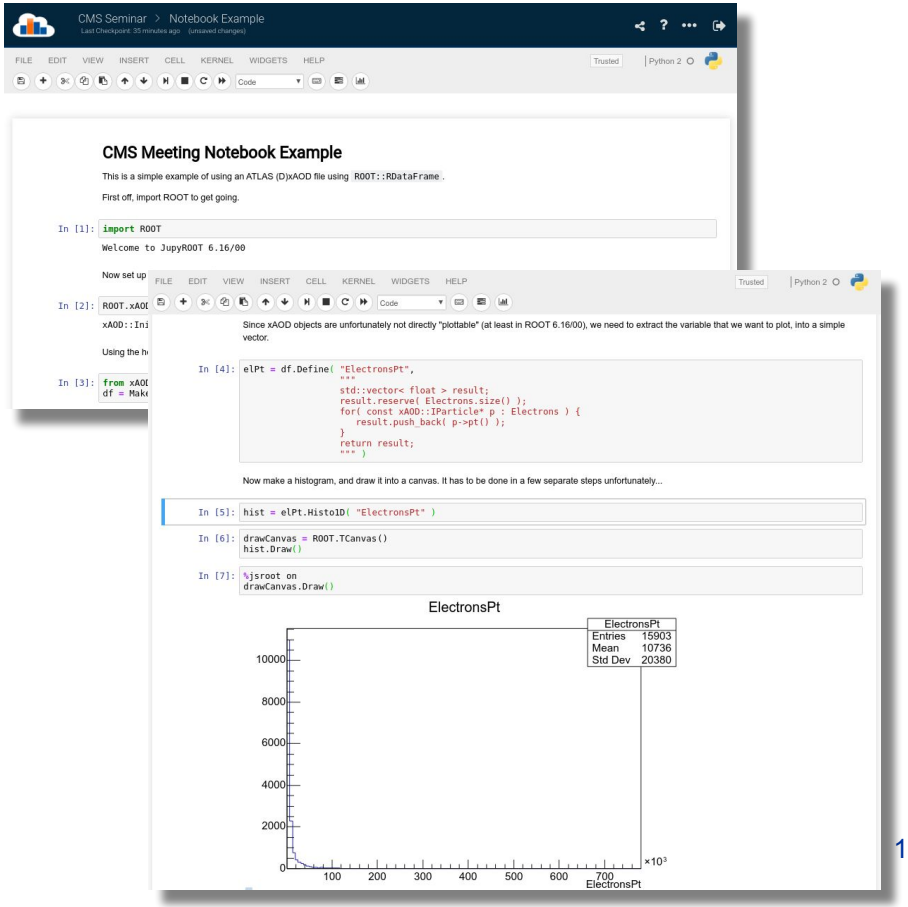
- Has a very similar goal as CMS's mini-AOD
  - Provide a single data format that could cover the vast majority of analyses
    - With a <50 kB/event size
  - Will (hopefully) allow us to reduce the number of DAOD types from the current O(100) to O(40)
    - A large number of technical formats will still need to remain
- Fundamentally the same kind of file as all of our current DAODs
  - But will only contain the variables absolutely necessary to perform all user level calibration tasks (with some very few additions)
    - This is where our EDM shines! We can keep using the same EDM while dropping variables. The EDM can also tell us which variables are used by a set of analysis tools.
- Will not require a major change in analyses
  - They will be expected to use DAOD_PHYS datasets in the same way as the current physics DAODs
- Will be our main workhorse for Run-3

DAOD_PHYSLITE

- Has a similar goal to CMS's nano-AOD
  - O(10 kB)/event
  - But still using the xAOD EDM!
    - Holding "nominally" calibrated objects instead of the "uncalibrated" ones shipped with DAOD_PHYS
- Users will still need to use tools provided by the analysis releases to get the systematic variations on the pre-calibrated objects
  - So the analysis releases will still play a big role
  - But these tools are much lighter weight than the ones applying nominal calibrations, it will be possible to perform analyses in more ways than it is now
- We will provide DAOD_PHYSLITE datasets already during Run-3
  - We have to figure out how to use them efficiently, as they will have to become our main way of physics analysis for Run-4

Declarative Analysis

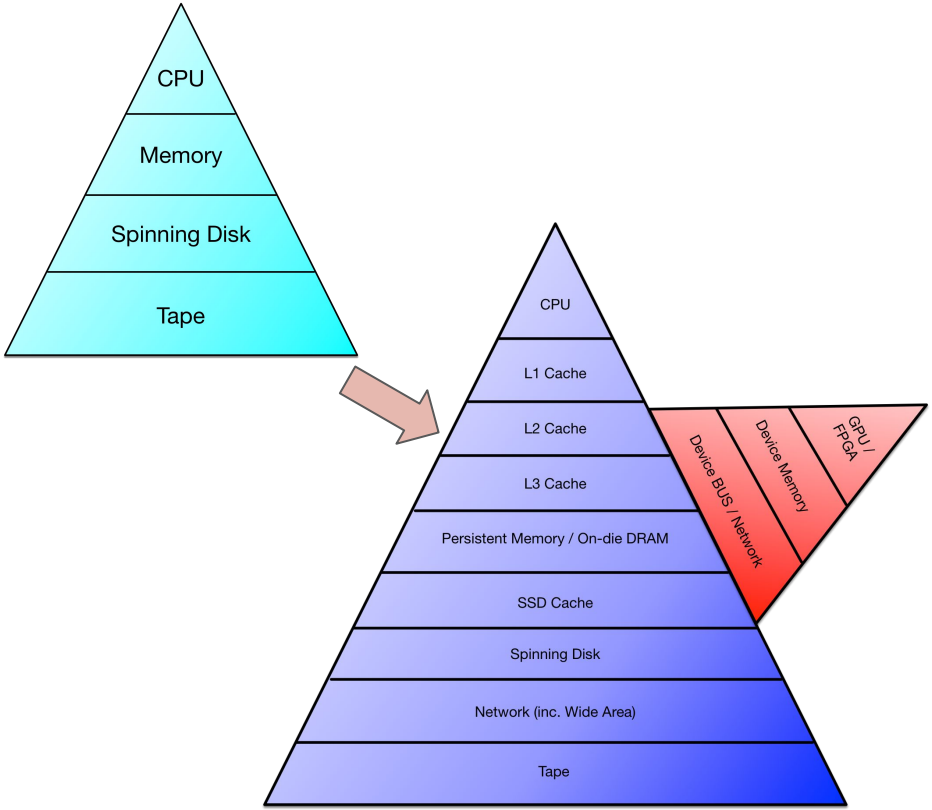
- You probably heard about Directional Acyclic Graphs (DAGs) recently…
  - This is one of the new buzzwords…
  - There is a growing trend in HEP to try to express all the analyses that we do in DAGs
- One of the easiest to use examples of this is ROOT::RDataFrame
  - Will probably make more and more use of such tools as we go forward
- But do exercise a bit of skepticism!
  - People are now trying to re-invent the exact same “analysis language” that we used with PAW…



Analysis Centres

- ALICE is currently evaluating the usage of dedicated “analysis centres” for Run-3
  - All physics analyses would be done in a small number of dedicated sites
  - One idea is to use these with a UI similar to SWAN
    - Though I can’t imagine how they would survive without providing a classical batch system on these centres as well 😕
- This is certainly gaining some traction in HSF
  - “Closely controlled” analysis centres would make it easier to make use of heterogeneous hardware for physics analysis
- ALICE claims an amazing O(PB)/hour analysis speed in their R&D on specialised setups
  - Will have to see how they can build an analysis model around this concept that will be compatible with how physicists develop code, and with analysis preservation in general

Heterogeneous Computing



- Computing is quite a bit more complicated these days than it used to be… ☹
- All the LHC experiments are trying to figure out how to best make use of new types of hardware (GPUs, FPGAs, etc.) for data processing
- I do believe that how well we manage to figure this out, will make a big impact on how we analyse data on the Run-4 timescale

Analysis at a Smaller Scale

The “X17 Experiment”

- Don’t want to go into any real detail about the experiment itself, just want to highlight a few computational details
- The size of this is many-many orders of magnitude smaller than those of the LHC experiments
  - Even much smaller than the planned FASER Experiment
  - O(50) parameters read out for one “event”
- Its data processing uses many tools developed for much larger experiments though

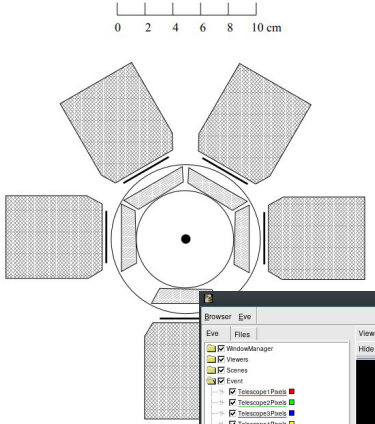
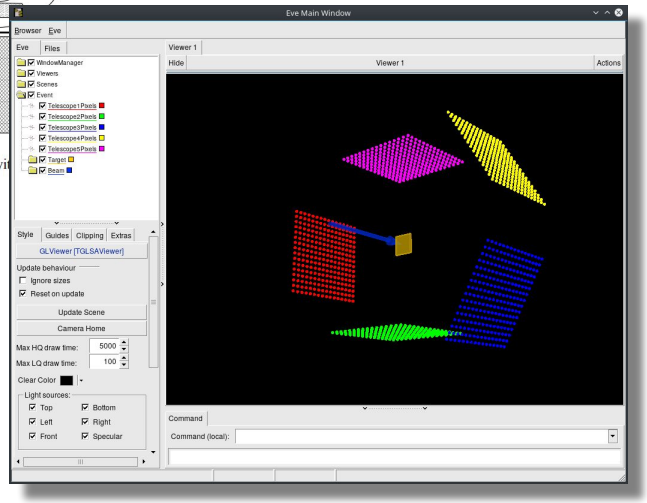
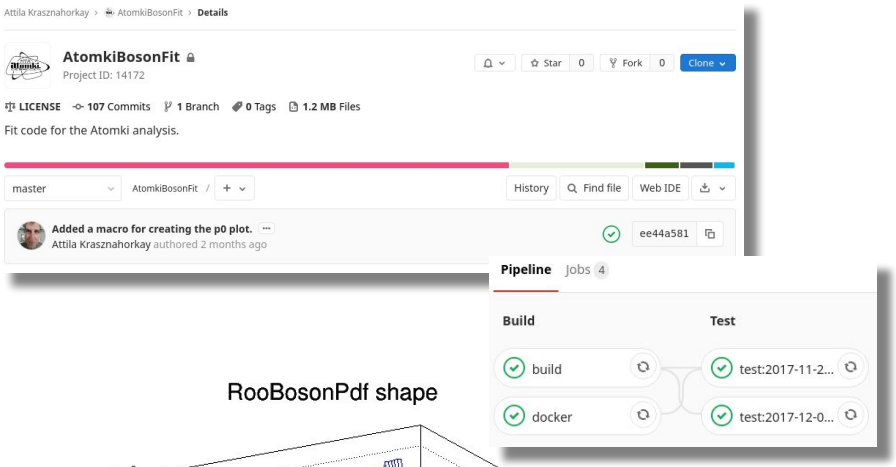
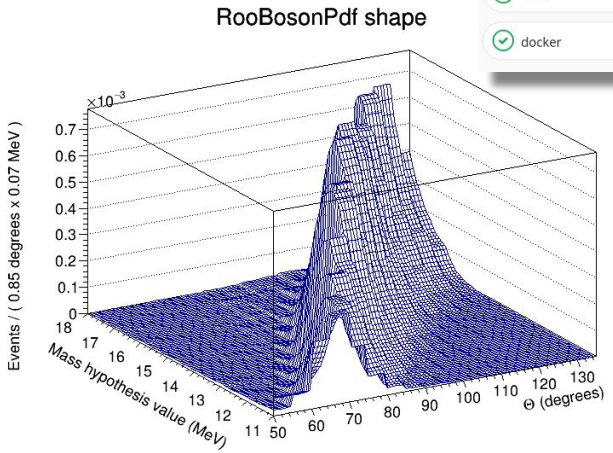


Figure 4: Final schematic arrangement wi



Statistical Analysis (for X17)





- Apart from writing the DAQ software for the experiment, I wrote the code performing the statistical interpretation for it
  - Using many of the same coding skills that I learned / developed for ATLAS
- The C++/Python code used for the fitting / significance estimation lives in gitlab.cern.ch
  - The repository's CI tests both the build of the code, and that it can successfully run on older distributions already analysed by it (histogram inputs are kept on EOS)

Summary

- The way we analyse data at the LHC is once again changing
  - What we did in Run-1 and 2 **were** the right thing to do. But now that we understand our detectors better, we need to do something else.
- The computing landscape around us is changing
  - Many smaller / simpler computing cores are becoming the norm. Our code has to adapt, as it currently performs very poorly on these systems.
- Machine learning can definitely help us along this road, but is not a silver bullet
  - For now just continue using it for the “classical” classification and regression applications, until somebody comes up with something better.

http://home.cern