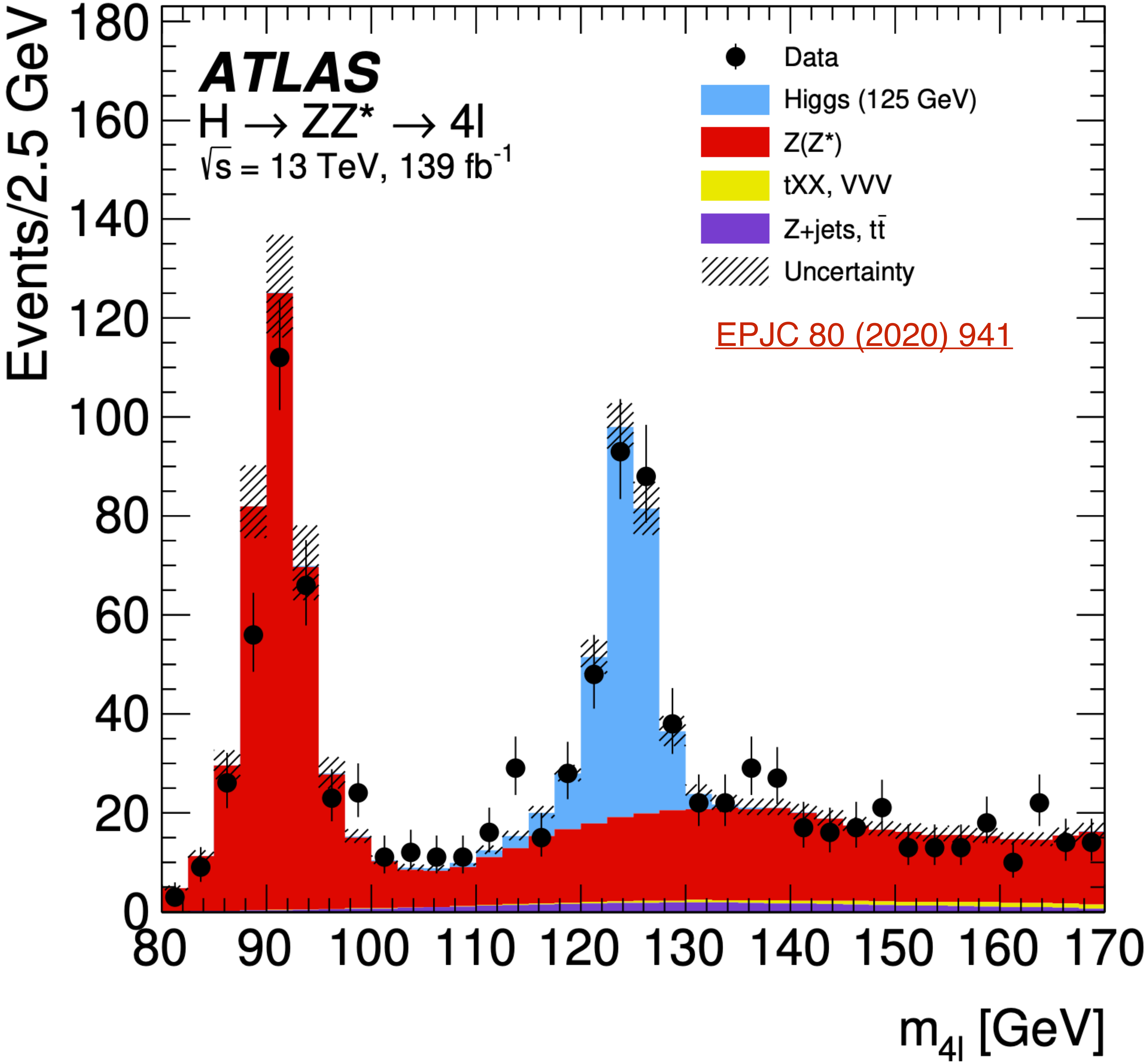
Optimisation of the Inner Detector tracking for the ATLAS High Level Trigger

Mark Sutton
University of Sussex
on behalf of the
ATLAS Collaboration

28th October 2020

Preface



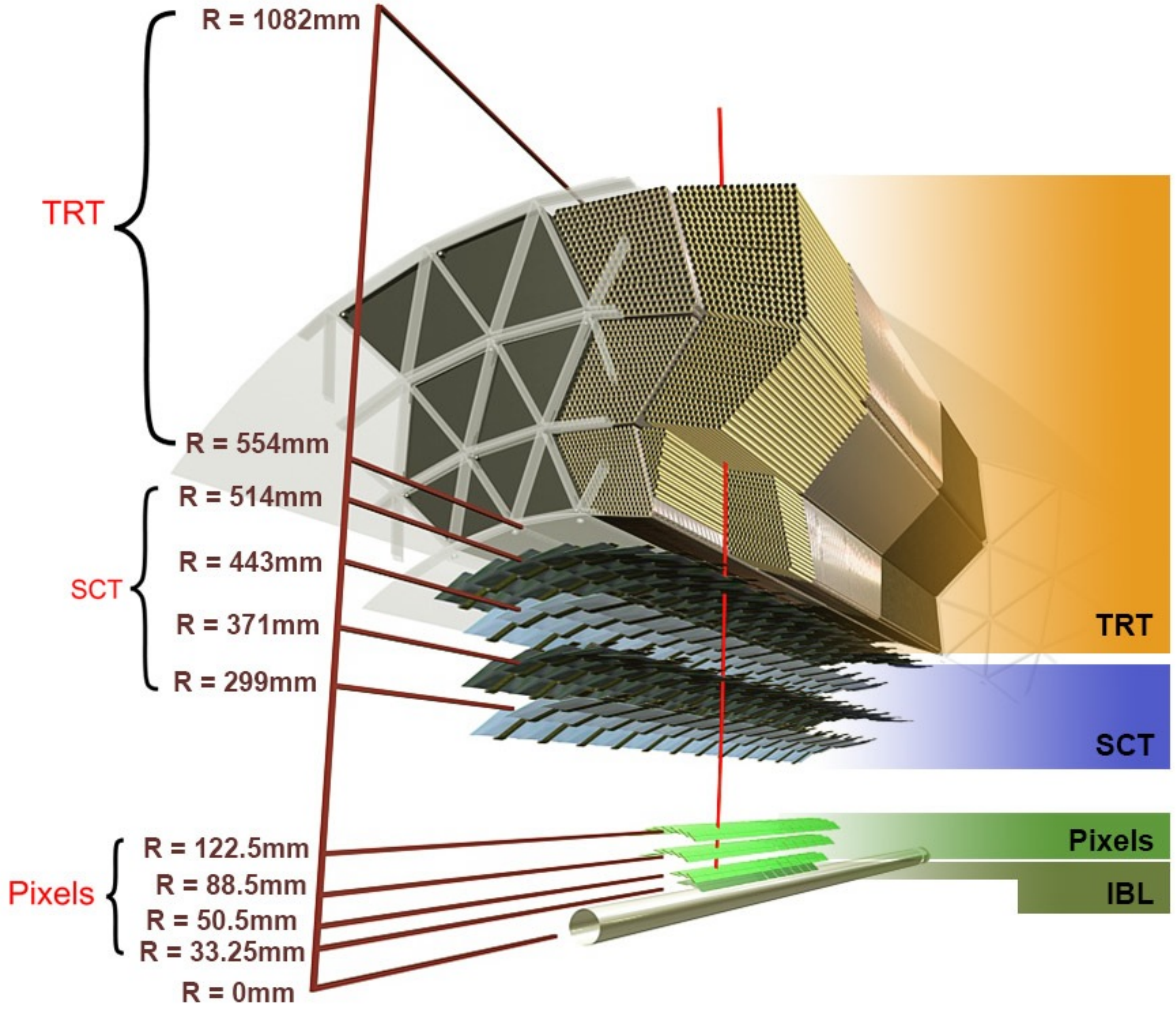
- The Inner Detector Trigger is an essential component of the ATLAS Trigger
- Without it, it would be impossible to achieve the required rate reduction with sufficient purity to fulfil the ATLAS physics programme

- Example H → 4 lepton analysis, only possible because of the high performance of the leptonic triggers
  - Every event used in the analysis was accepted following the successful reconstruction of the leptons in the trigger

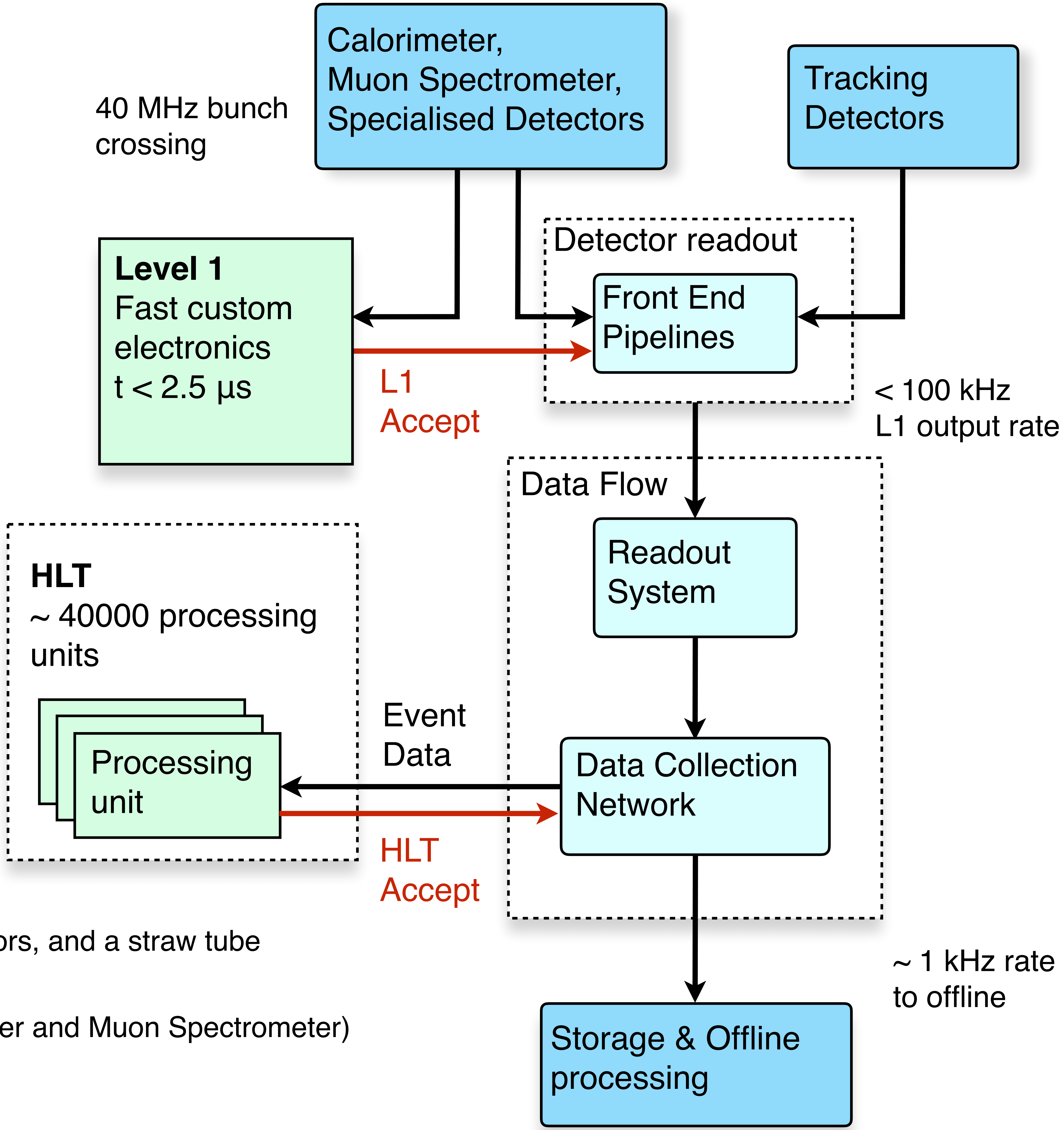
Rubric

- Was requested to address some specific questions …
  - How the resource estimates for online reconstruction were obtained and then validate
    - To what extent this can be extended to an e+e- experiment
  - What the implications are for interfacing ‘real time world’ to ‘offline world’ In terms of software techniques and DAQ
  - Any ATLAS extrapolations for the cost of HLT processing into the future

- Will attempt to address each of these, to a greater or lesser extent



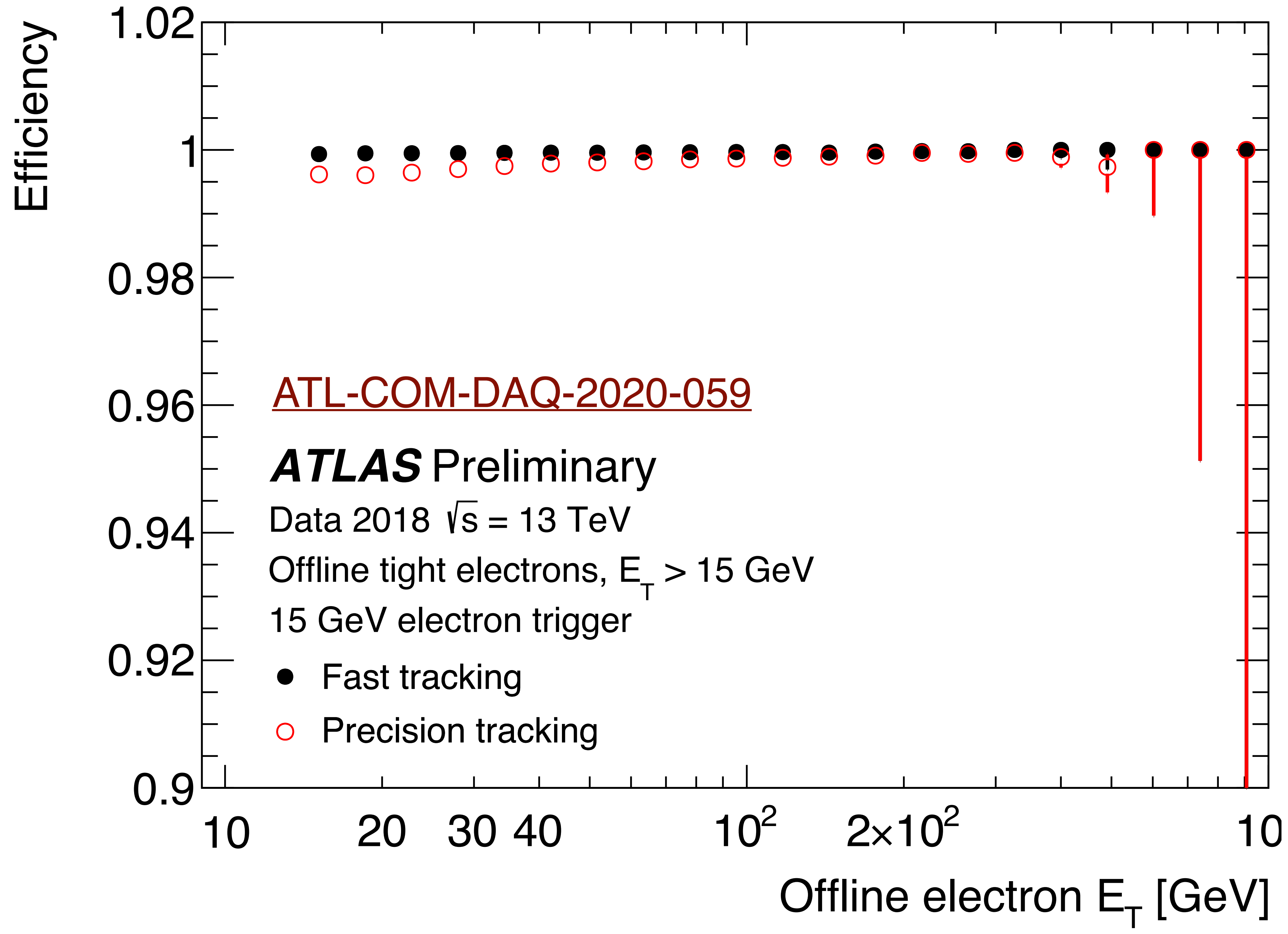
The ATLAS Trigger and the Inner Detector

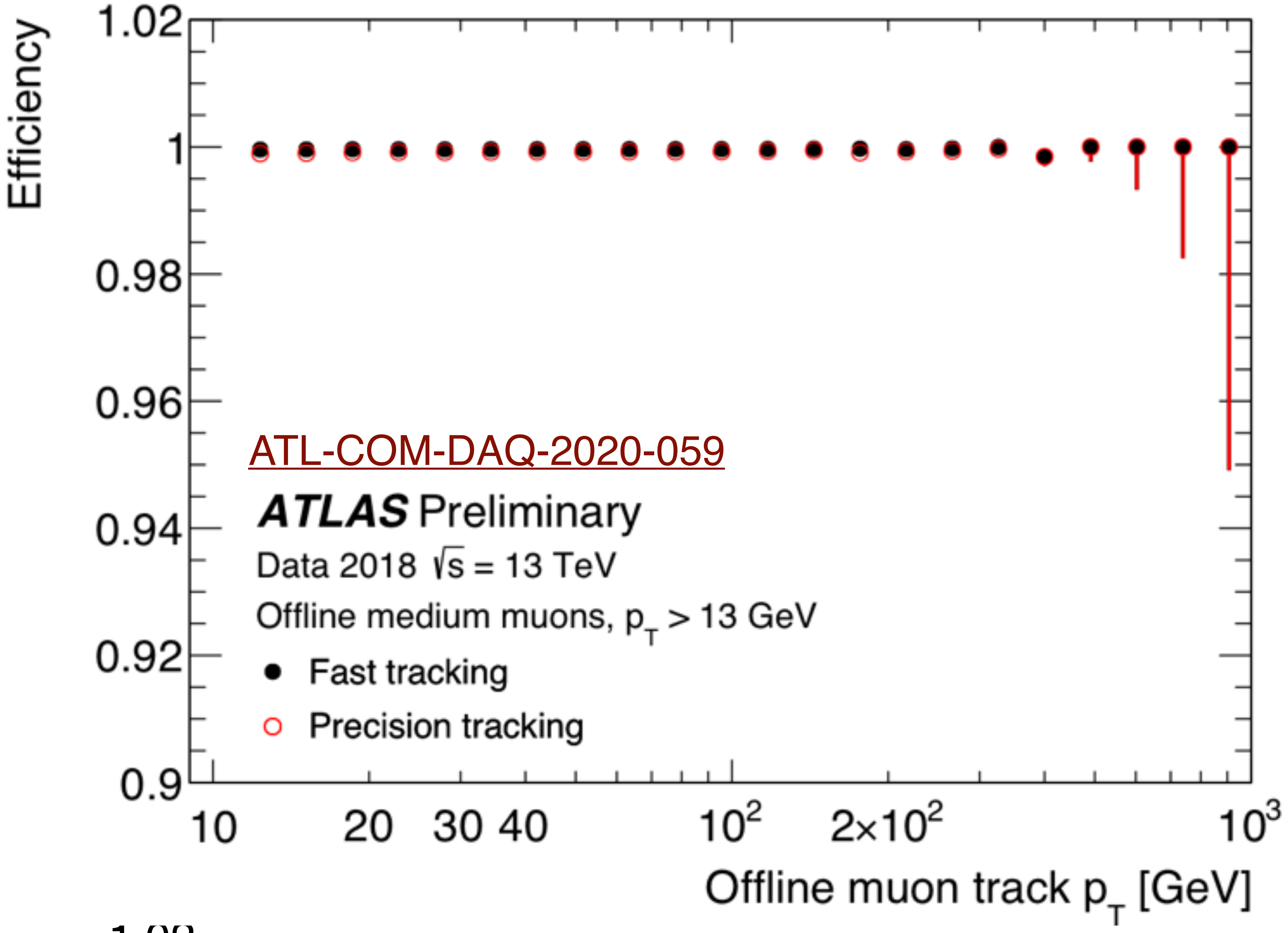


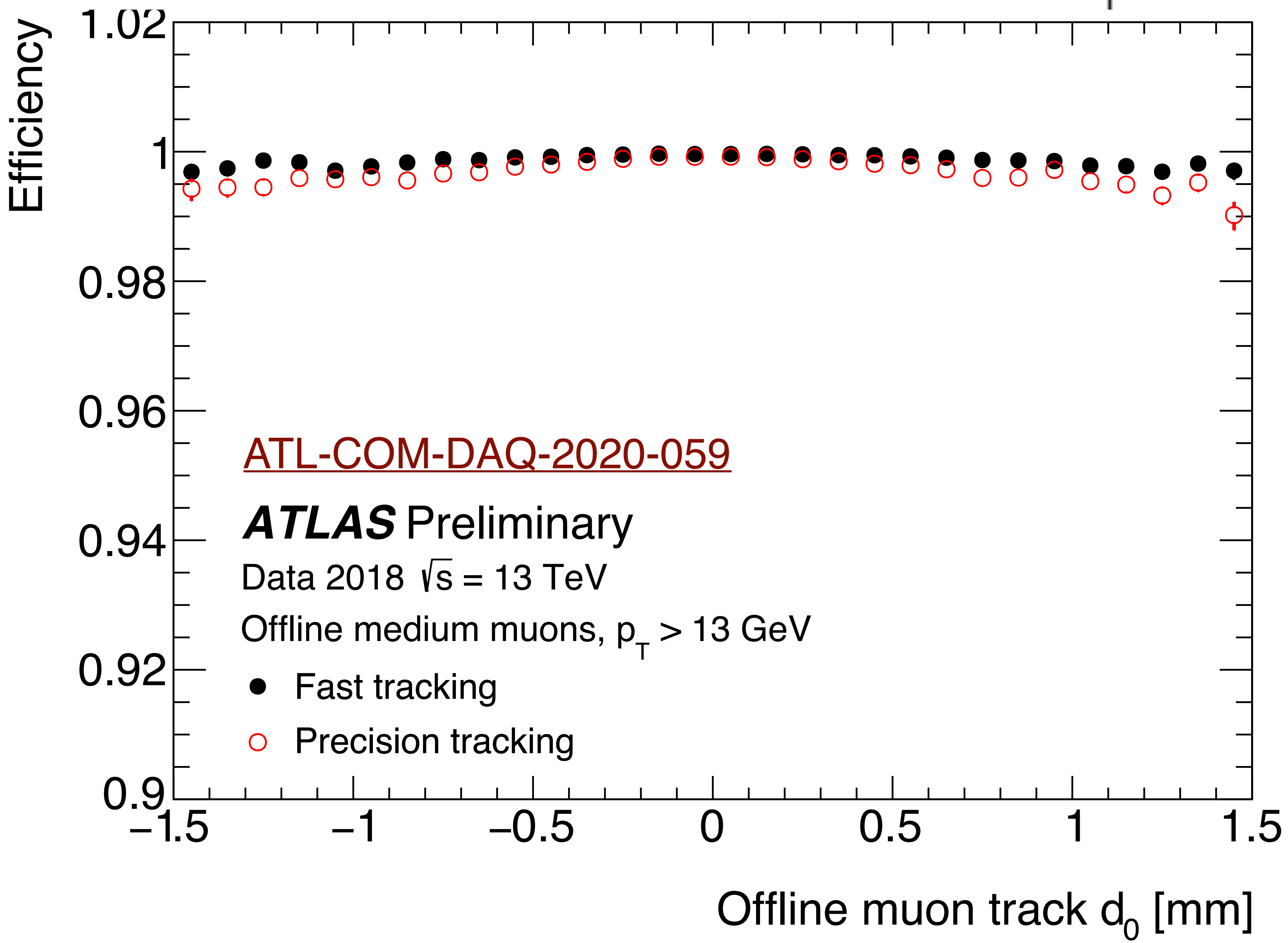
- The Inner Detector (ID) consists of a **PIXEL** detector, microstrip detector (**SCT**) silicon detectors, and a straw tube transition radiation detector (**TRT**)
- Level 1 runs fast reconstruction with dedicated limited granularity detector readout (Calorimeter and Muon Spectrometer)
  - Identifies Regions of Interest (RoI) for processing in the HLT with full detector granularity
- Silicon detector readout not fast enough for use in Level 1 …
  - Used for the first time in the DAQ chain in the High Level Trigger (HLT)

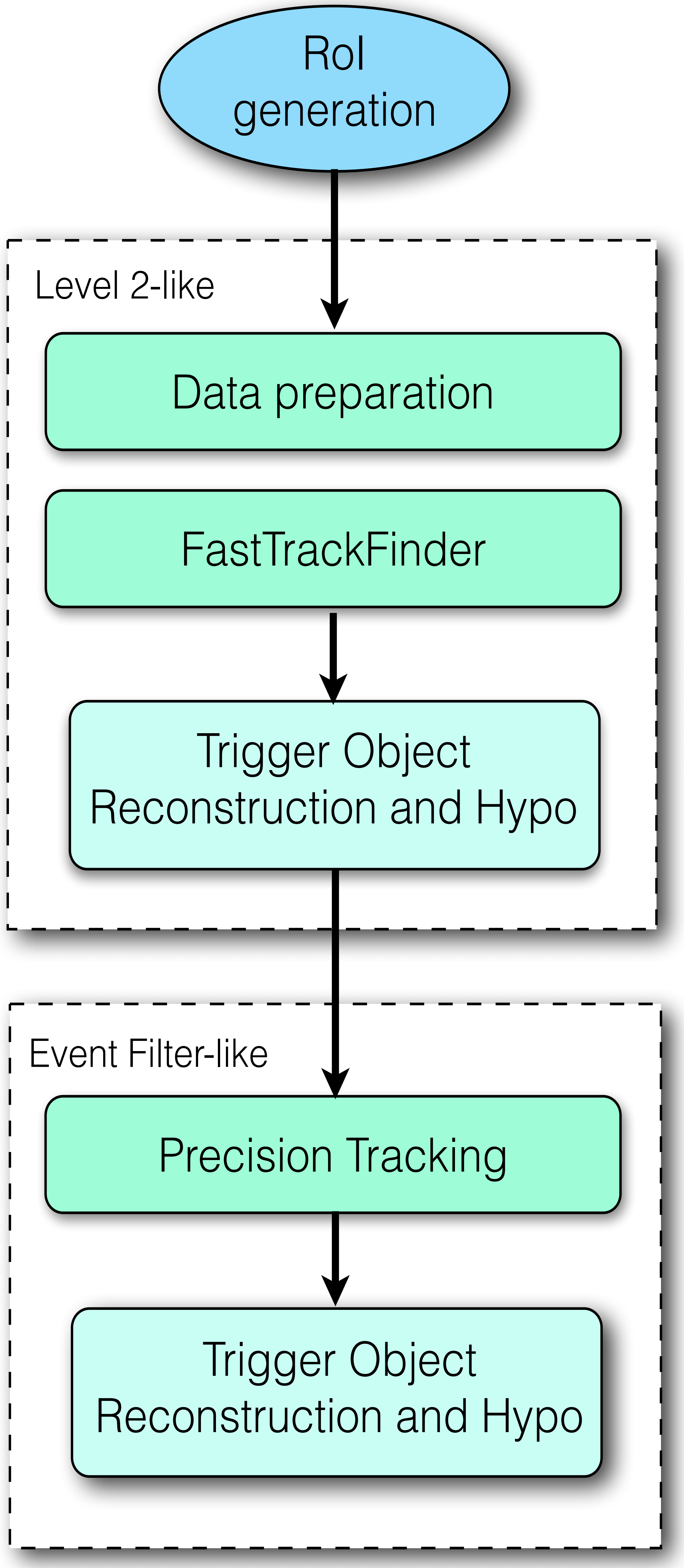
ID Trigger overview

- The raw detector data within the RoI requested from the Readout system
- The data preparation then runs once per RoI
- Pixel and SCT clustering, transformation to spacepoints
  - **Fast tracking** is expected (custom seeding, combinatorial track following, fast track fit)
  - Fast tracks used to seed the **Precision Tracking**
    - Resolved ambiguities in the pattern recognition, rejects potential fake tracks, runs the offline track fit
- Trigger signatures, used in the nearly all trigger signatures, muons, electrons etc







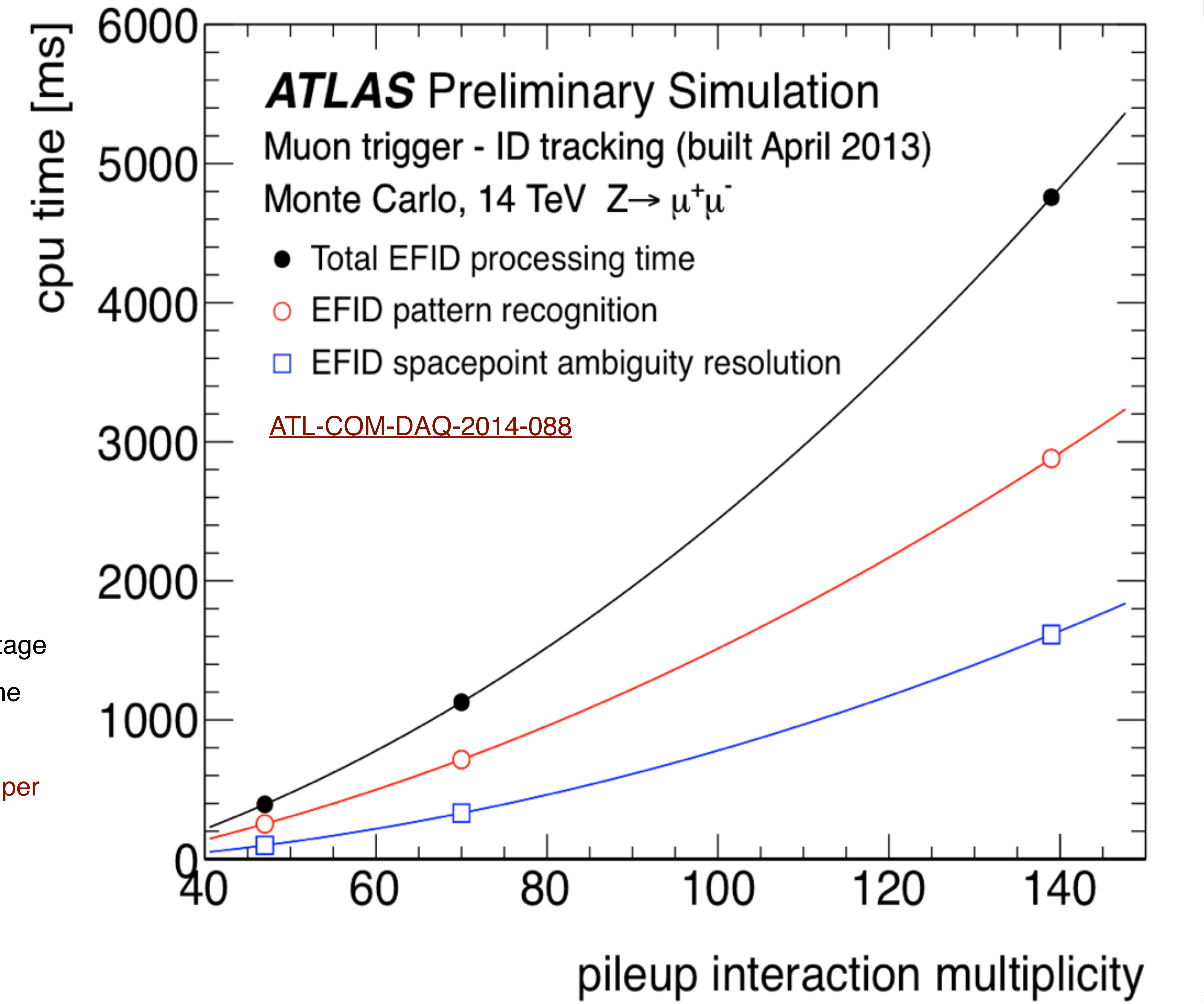


Offline to online …

- General desire to run trigger reconstruction as close to the offline reconstruction as possible

- For LHC Run 1 (2009 - 2013) ran a **two level** HLT
  - Level 2 running faster, but less accurate track reconstruction
  - Event Filter (the third trigger level) running a modified version of the offline tracking - EFID
    - Reruns data preparation
    - Offline pattern recognition
    - Offline ambiguity solver - including offline track fit

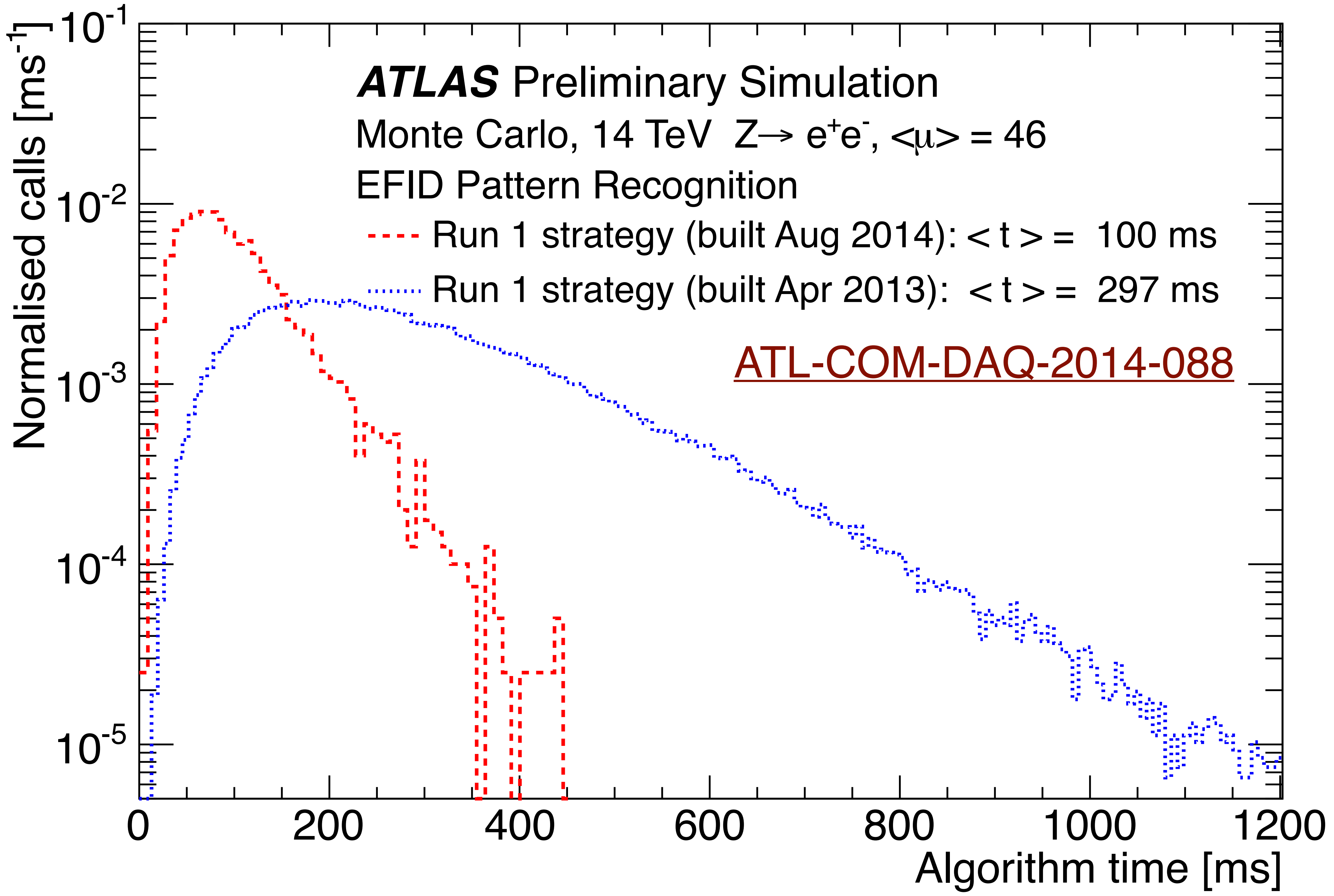
- For Run 2 (2015-2018) L2 and EF stages combined into a single HLT stage
- Avoid running the data preparation and pattern recognition a second time
  - More than 50 % of CPU in the offline algorithm in pattern recognition
  - Doesn't scale well with pileup multiplicity around 50 - 60 interactions per bunch crossing at the start of a fill in Run 2
  - Too slow for the trigger

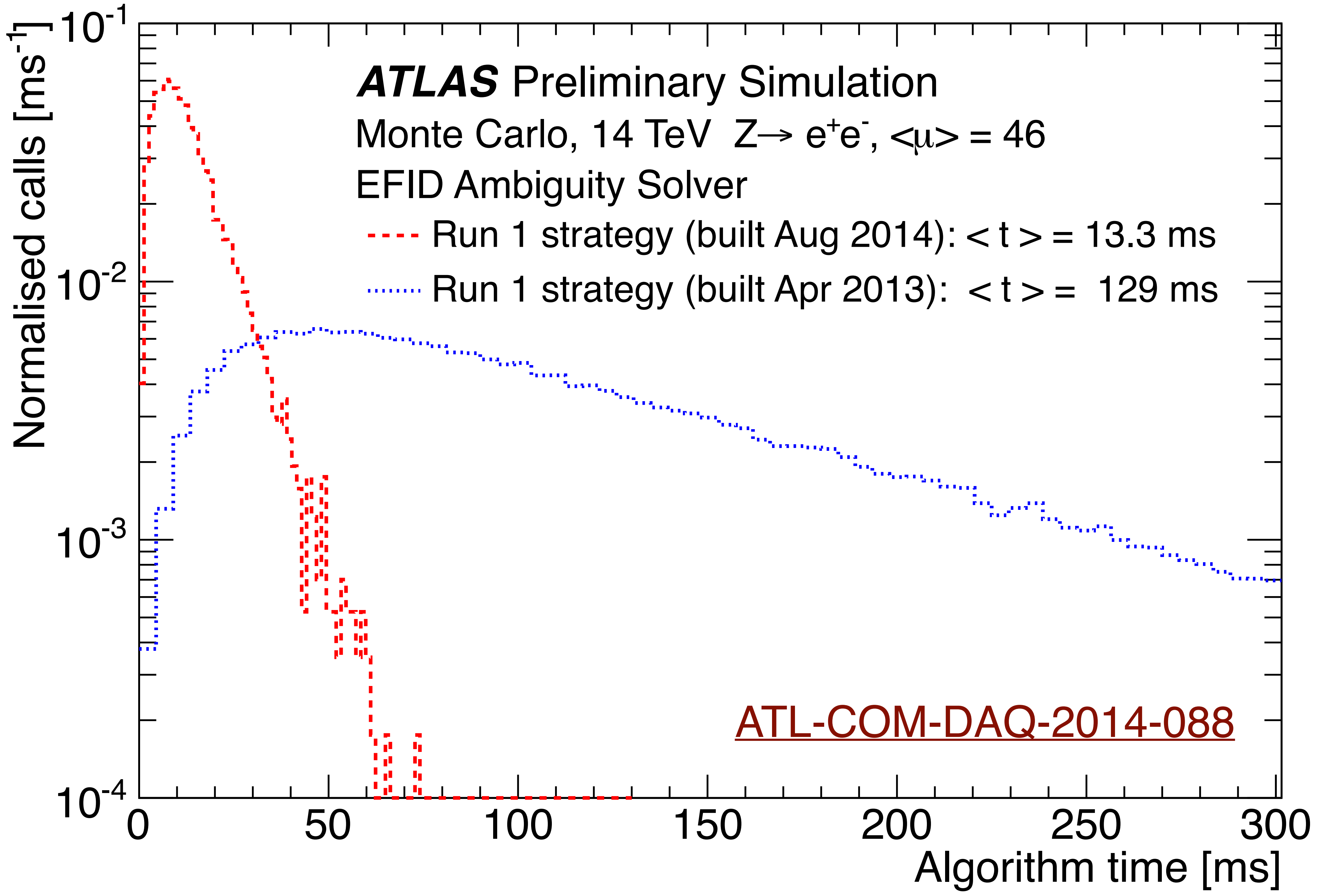


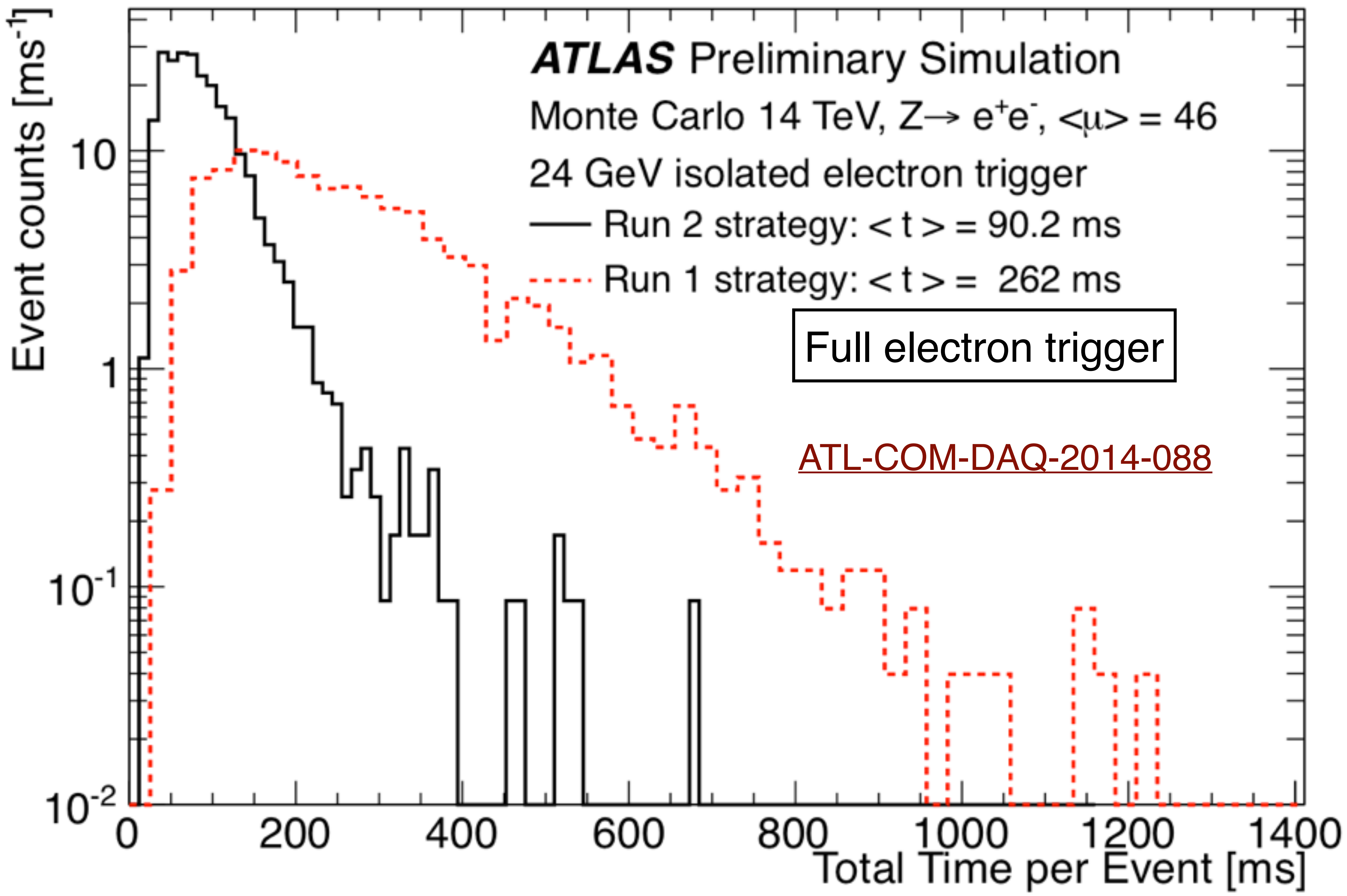
Improvements to the offline tracking

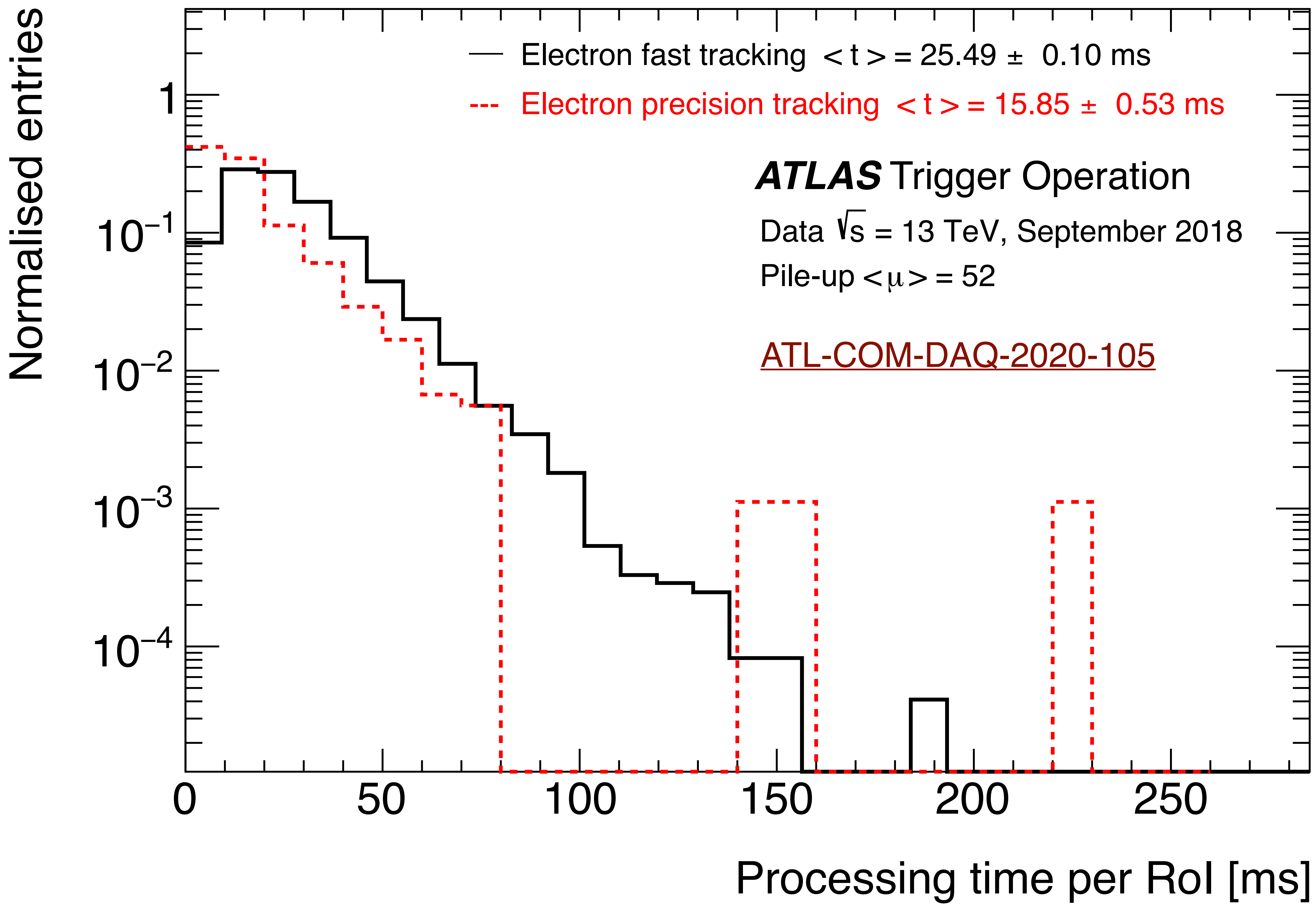
- Offline tracking itself is costly
  - After trigger selection , events written to offline storage and subject to the full offline reconstruction in the ATLAS Tier 0
  - Offline reconstruction must not fall too far behind data taking from the detector
  - Although not time critical in the same way as the trigger tracking, is a limiting factor in the Tier 0 reconstruction
- For Run 2 there were improvements in the offline execution time:
  - Improvements to the computing infrastructure:
    - Switch to running on a 64-bit, rather than 32-bit kernel
    - Newer compiler, switched to gcc 4.8 from 4.3
    - Modifications to the algorithm execution, replacement of the CLHEP [45] linear algebra library by the Eigen library
- These immediately lead to benefits for the Trigger EFID tracking, but still rather too slow for trigger use

- A new strategy was needed for Run 2





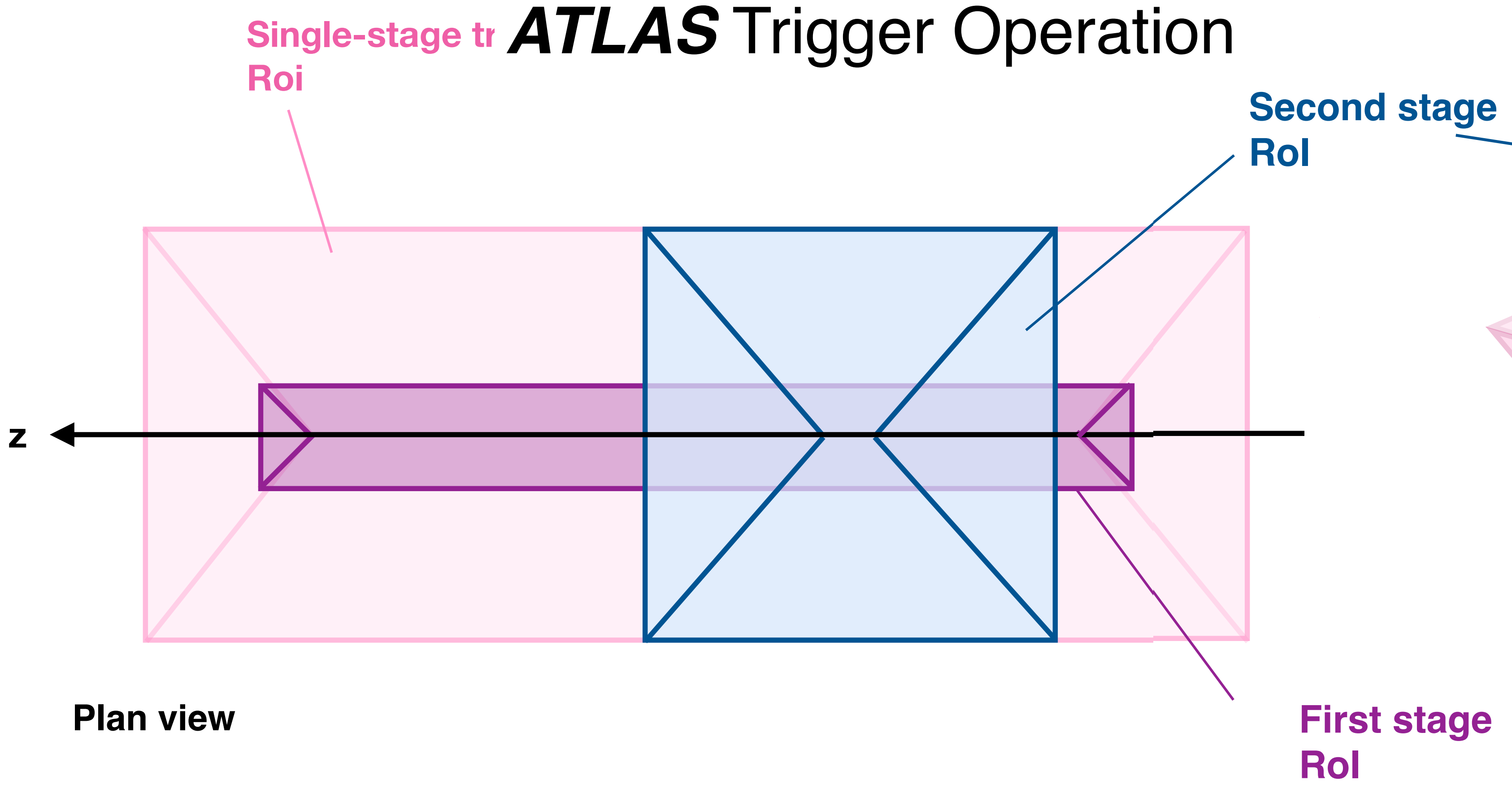


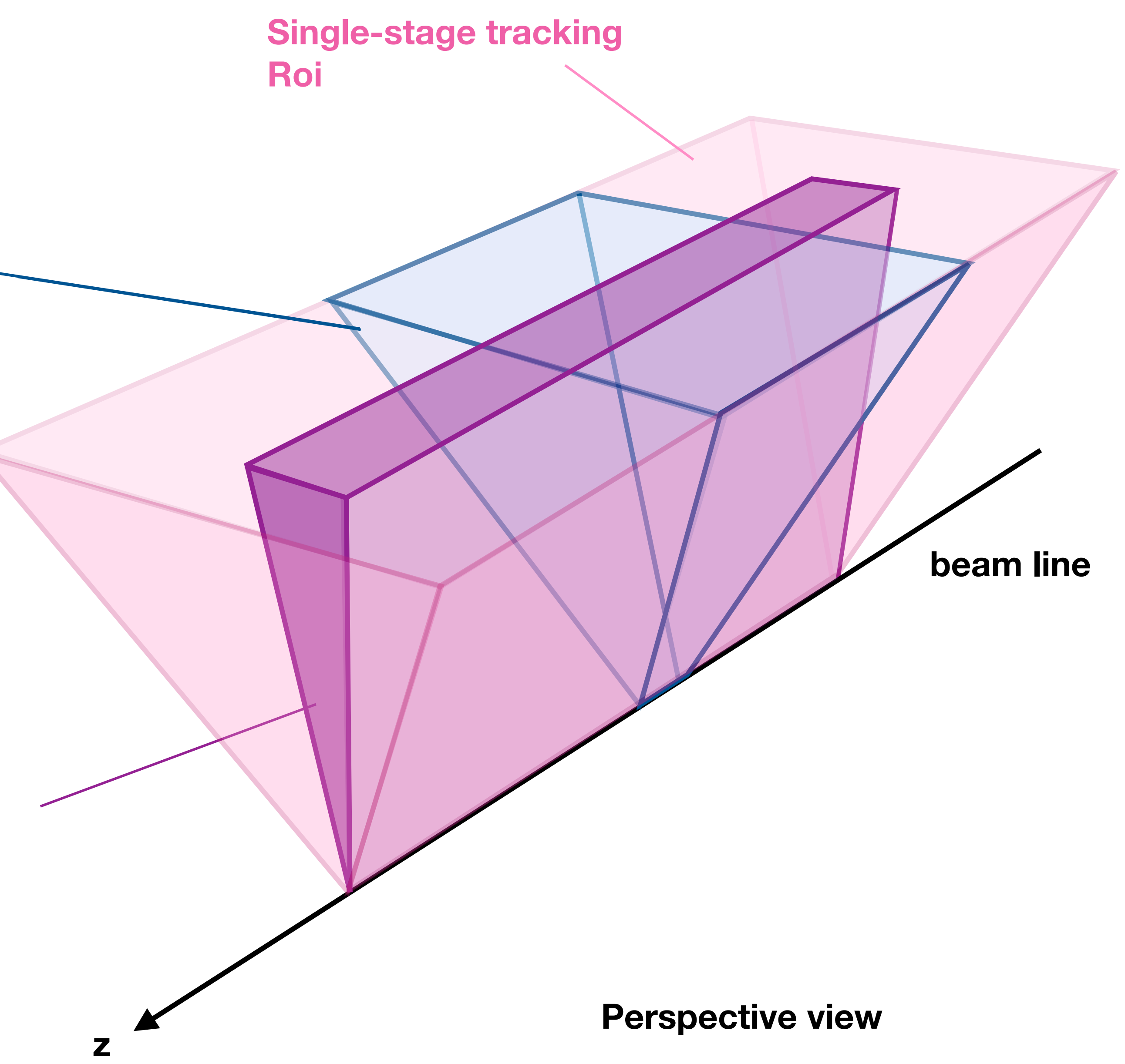


Run 2 tracking strategy

- New strategy for Run 2: redesigned the fast L2 reconstruction for use in the single stage HLT:
  - Fast track reconstruction - customs seeding but with offline track following
  - Seed the offline ambiguity solver with the tracks from the fast reconstruction as explained in the ID Trigger overview

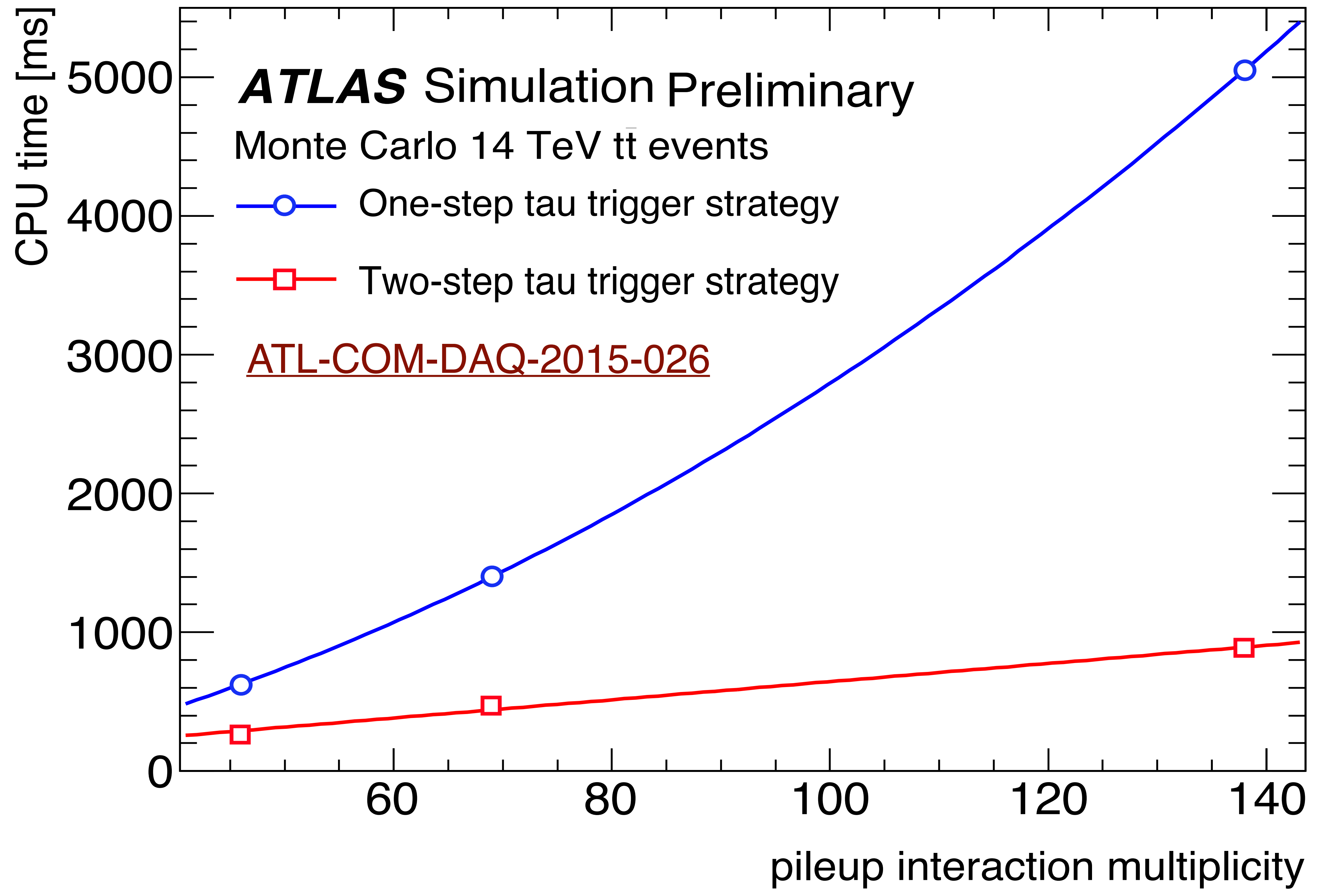
- Custom Fast tracking and seeding gives significant reduction in CPU with respect to the Run 1 strategy, running the offline pattern recognition

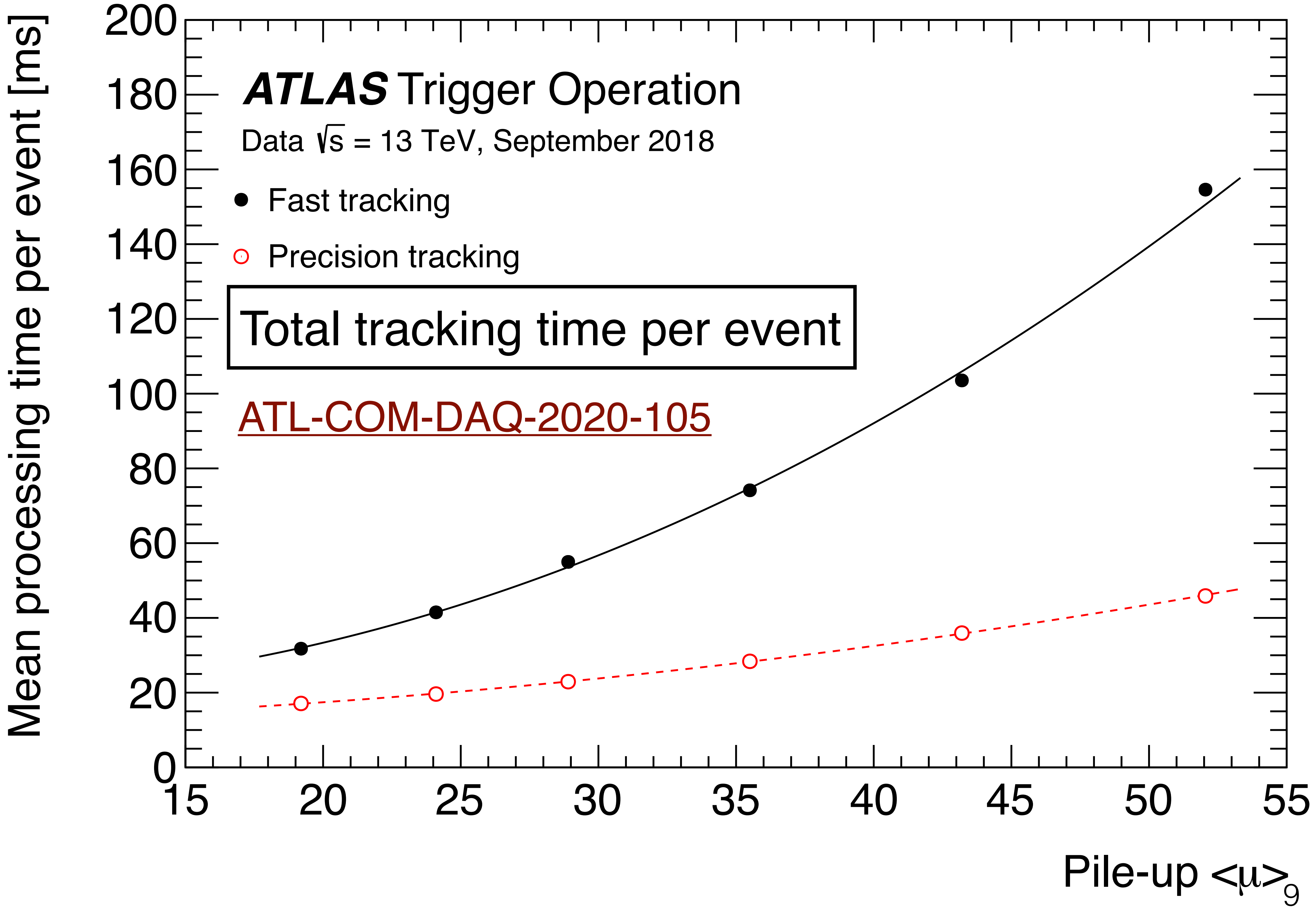




Two stage tracking

- Despite using custom pattern recognition for some signatures, eg Taus and b-jets, even this may still be too time consuming because of the wide RoIs, needed, eg 0.8 × 0.8 for the jets
- Adopt a two stage approach
  - First stage:
    - Run the fast tracking in narrow RoIs, but extended the full length of the luminous region in z
      - For the tau trigger identify the likely leading tau decay product track vertex
      - For b-jets run tracking only in the jet core for all jets, use the tracks to reconstruct the event vertex
  - Second stage
    - Run the fast tracking again, but now with the full width RoI, but tight around the z position identified in the first stage
- Signifiant saving in the overall CPU for the tau and b-jet triggers
- Overall time to run the actual tracking algorithms around 160 ms per event — 290 ms including all data preparation etc





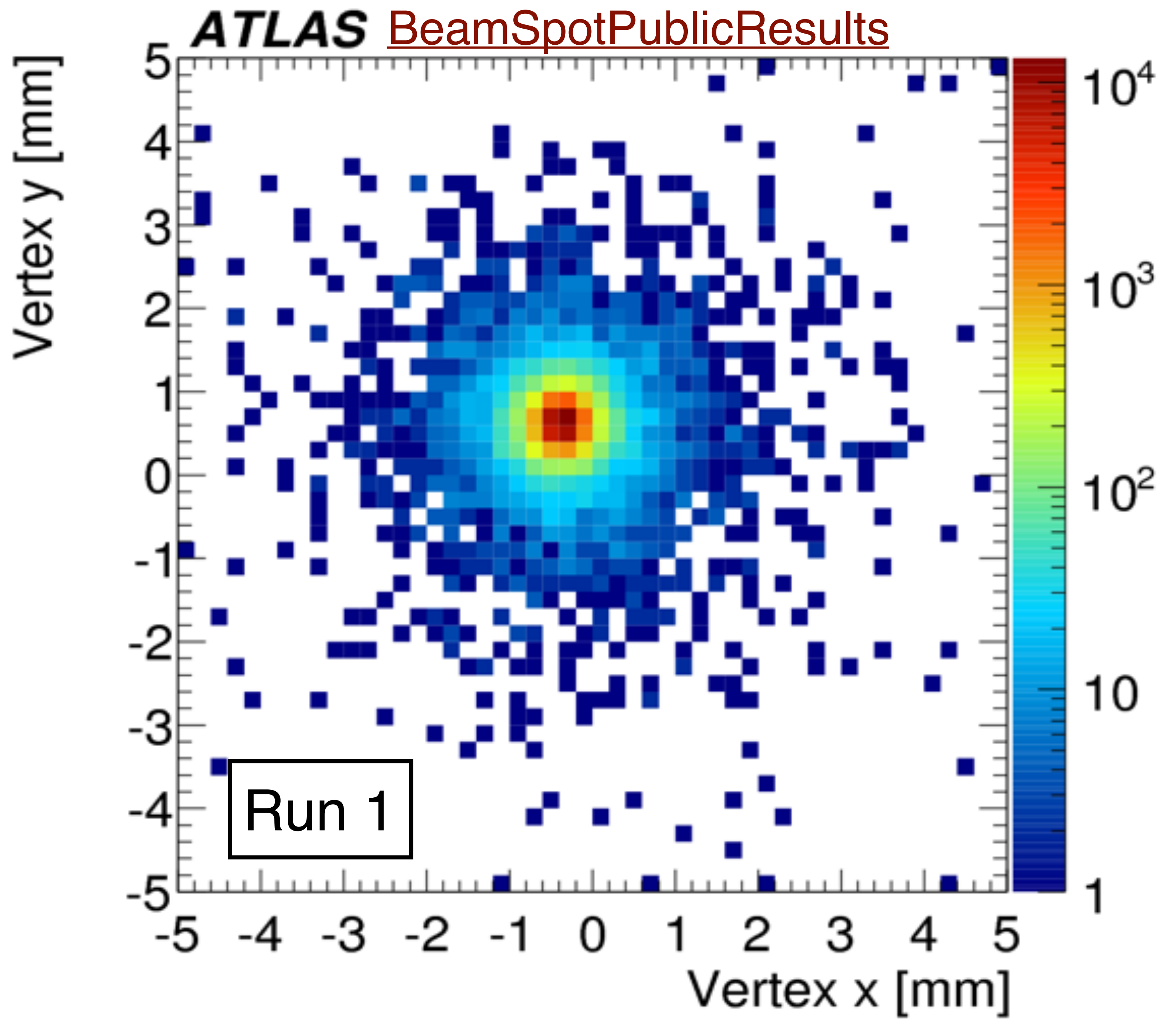
Additional offline considerations

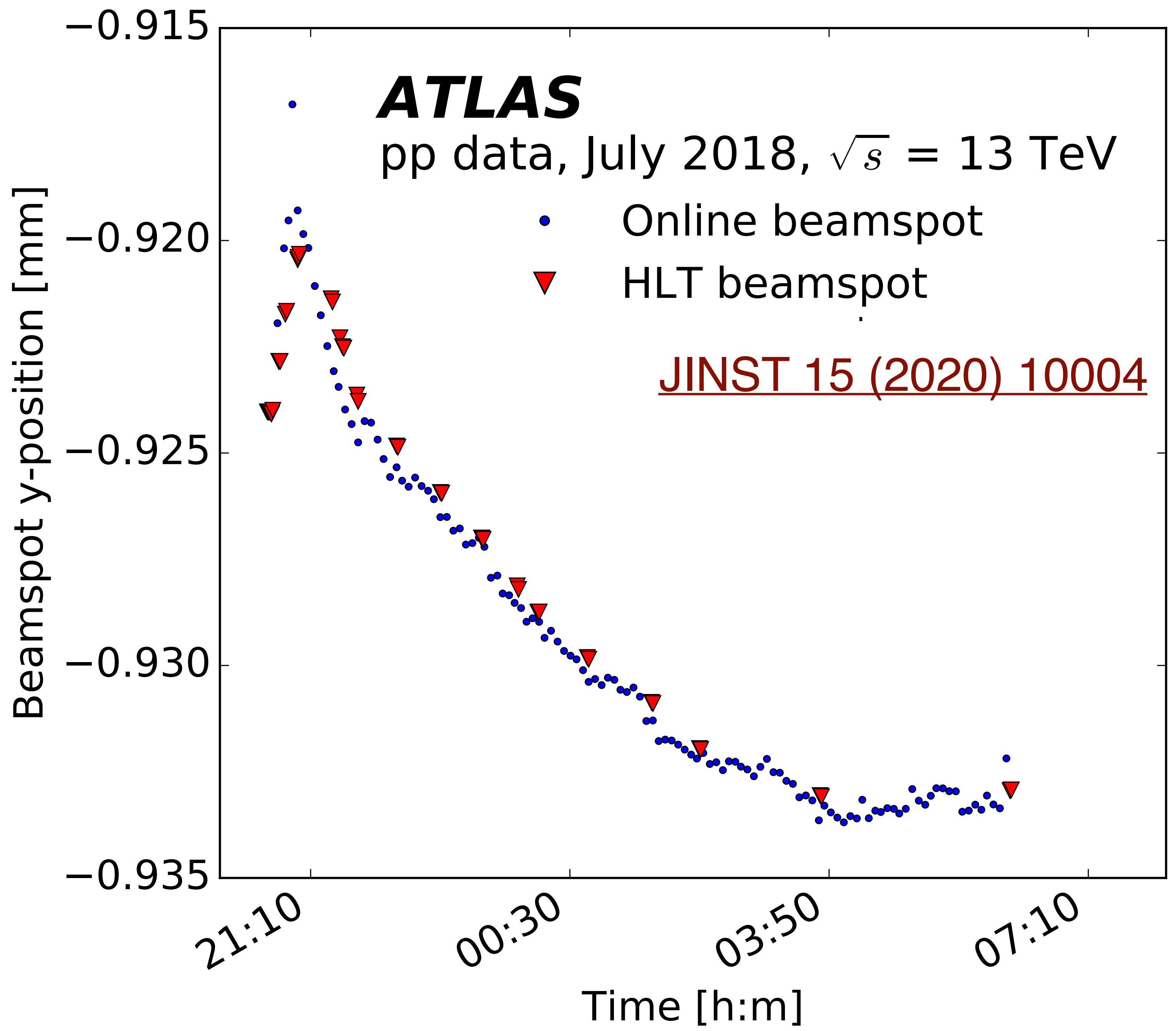
- Other offline consideration that need modifying
  - Offline reconstruction has a first express stream pass - limited event sample, useful for determining
    - **Track based alignment** and determination of the **beam line position** (beam position moves throughout a run), **instantaneous luminosity**
    - Subsystem Calibration
    - Determination of dead or inactive modules, hot cells etc

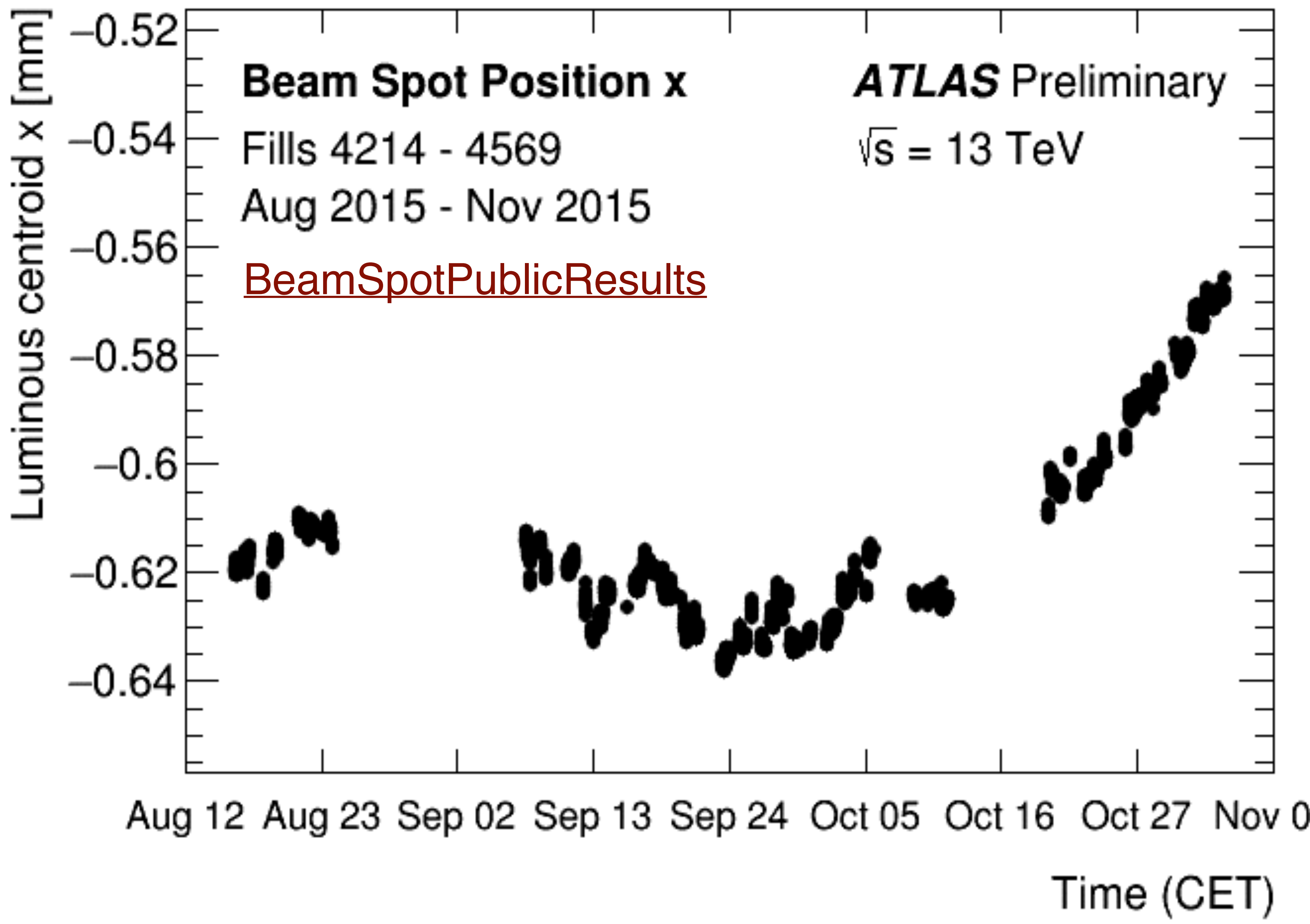
- ATLAS used the concept of the **Luminosity Block**
  - Fixed duration block of luminosity ~ typically 1 or 2 minutes — over which the detector conditions are assumed to be constant
    - Calibration constant, alignment etc determined per Lumi Block

- Of course, cannot use the detector calibrations per Lumi Block during data taking as the data has not been processed
  - More limited use of detector conditions and alignment - most up to date version fixed at start of the physics run

- A notable exception is the **beamspot** algorithm …

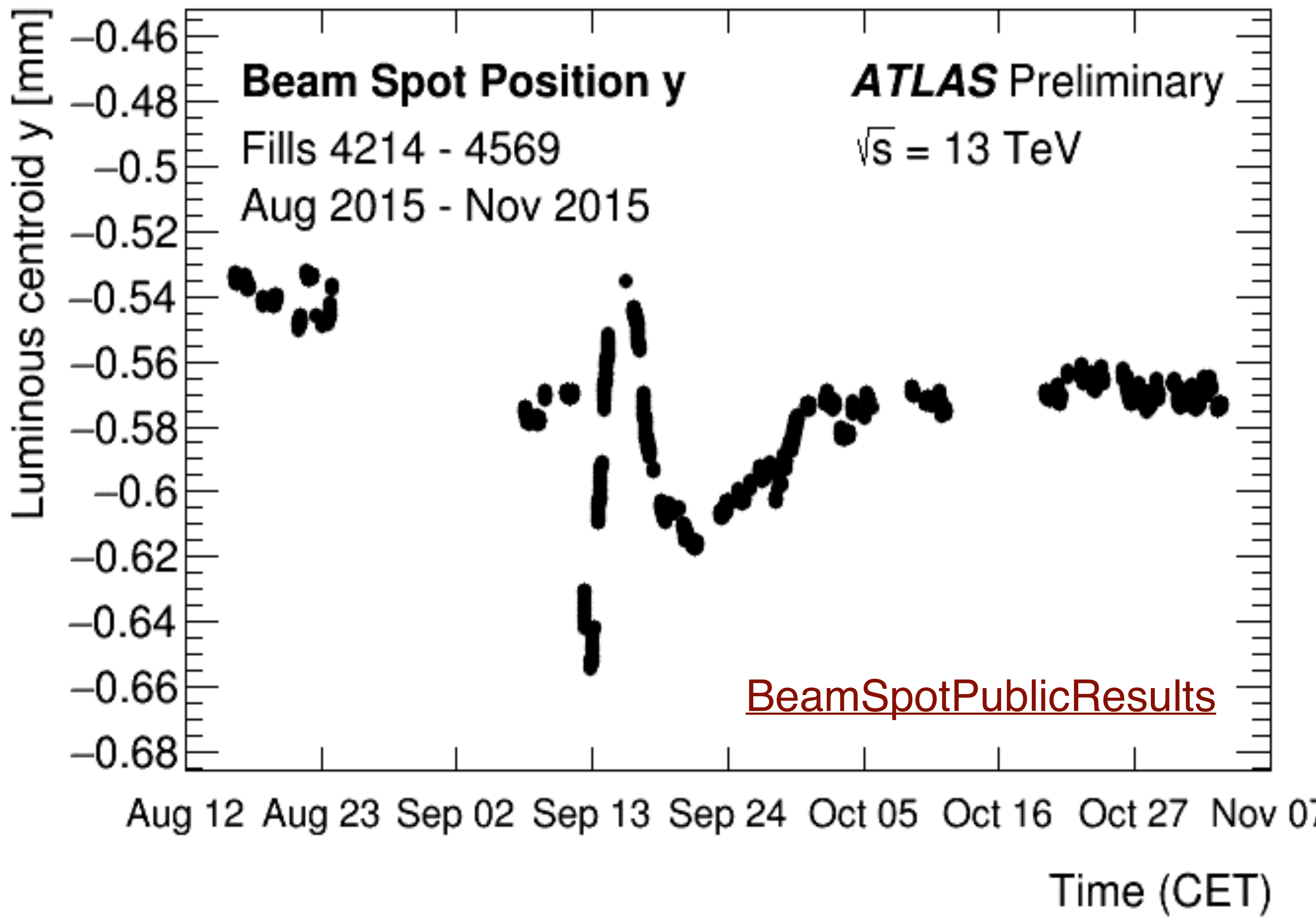






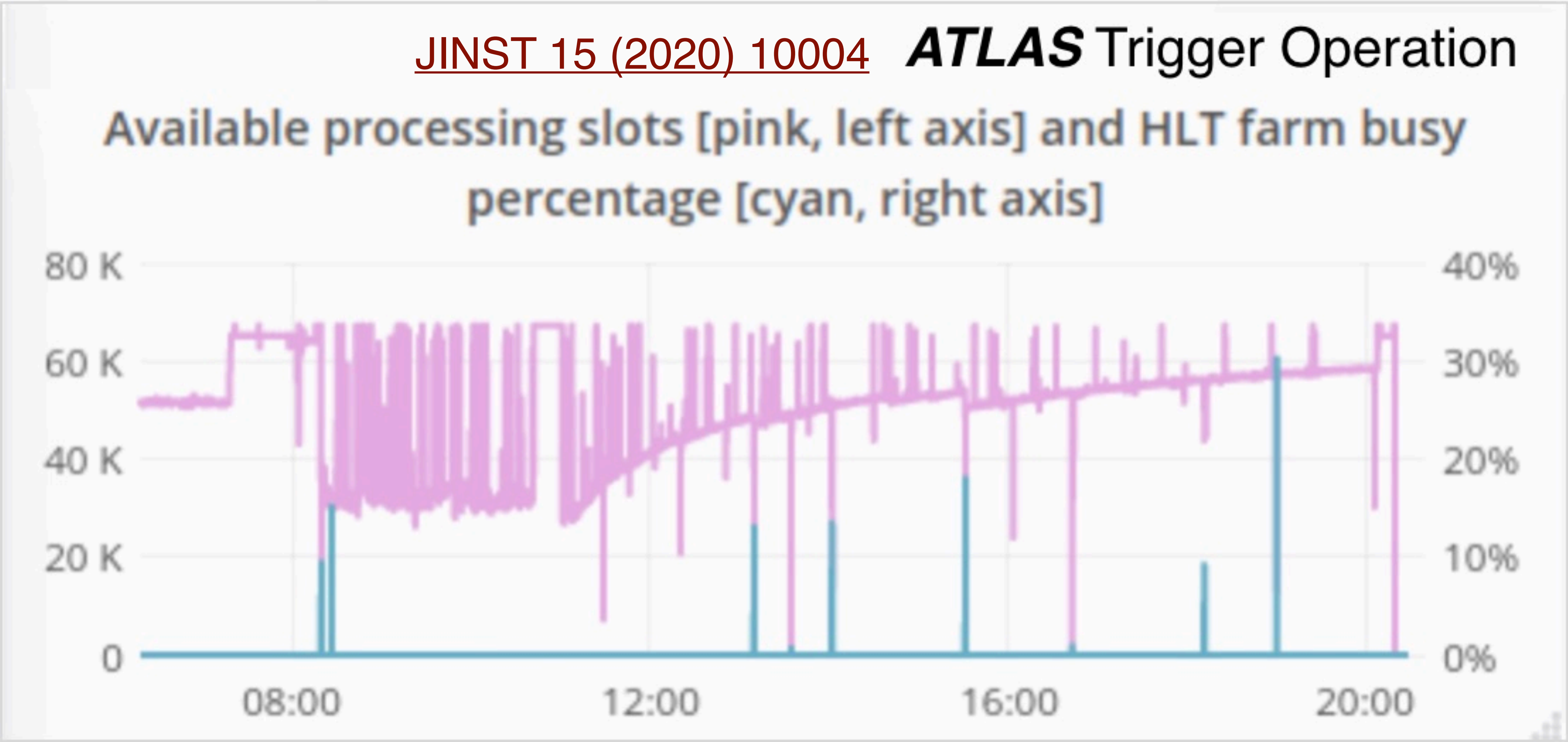
The beamspot and luminosity

- Beamspot needed for optimal tracking performance, but also for accurate determination of the track transverse impact parameter for efficient **secondary vertex finding** and **b-tagging**
- Luminosity for pileup corrections etc
- Statistics are accumulated throughout the Lumi Block
- For beamspot, have a dedicated beamspot chain, with its own instance of the tracking
- Once the beam has moved in x, y beyond some threshold, the updated beamspot position is broadcast to the entire HLT farm
  - The entire farm must be updated at the same time …
  - The beamspot is distributed along with synchronisation information and all the HLT nodes update and start using the new beamspot information at the start of the next Lumi Block



Resource estimation

- Lots of algorithm execution times shown so far, but where do they come from ? How do we estimate required CPU resources to ensure we can to run the tracking ?
  - For example, if an algorithm takes 1 second to execute, but only in 10 out of every 1000 events, that is only a 10 ms average contribution to the total processing time
- Trigger cost monitoring
  - Used to monitor the execution time for each algorithm run as a step in the overall trigger processing
- Nightly tests
  - For every nightly build for the code release, run tests for each signature (~ 40), with multiple Monte Carlo samples
  - Internal timers for the different stage of processing, overall algorithm execution times from the cost monitoring
  - Allows full spectrum evaluation for any change in the code - quickly identifies losses in efficiency or slower algorithm processing for any code changes
  - Detailed performance estimates for the efficiency and resolution
- Data taking from the ATLAS detector
  - Cost monitoring times all algorithms, samples 1 in every 10 event processed by the HLT
  - Timing information written to the event stream for later reconstruction off line
  - Monitoring HLT Farm occupancy in detail - number of cores occupied at any one time, events waiting to be processed etc - establish operation points for future running



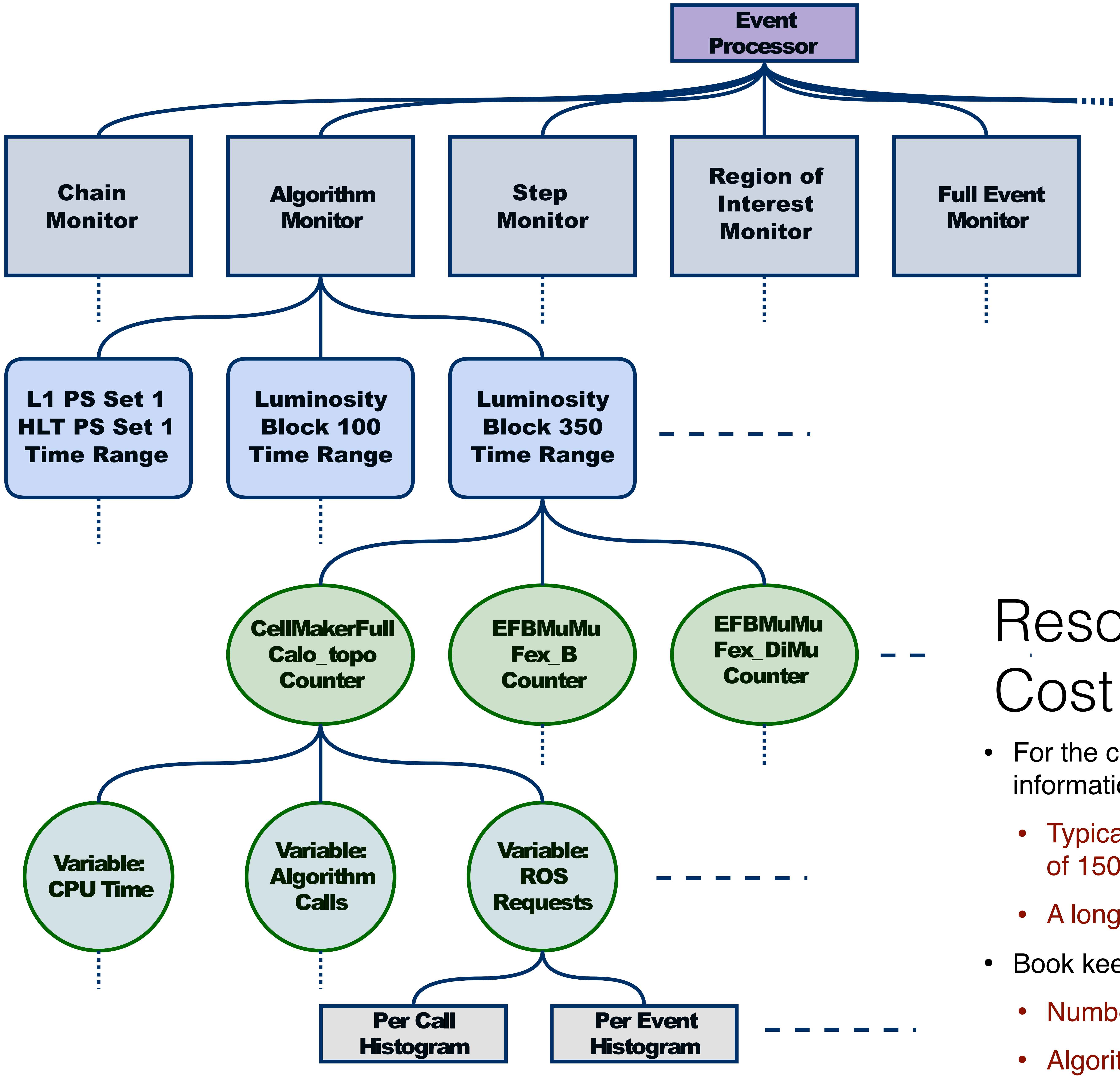
- Trigger reprocessing
  - Runs the trigger offline on the **grid** - data from special **Enhanced Bias** runs with effectively a large HLT trigger pass through rate used to give realistic spectrum of events input to the the HLT

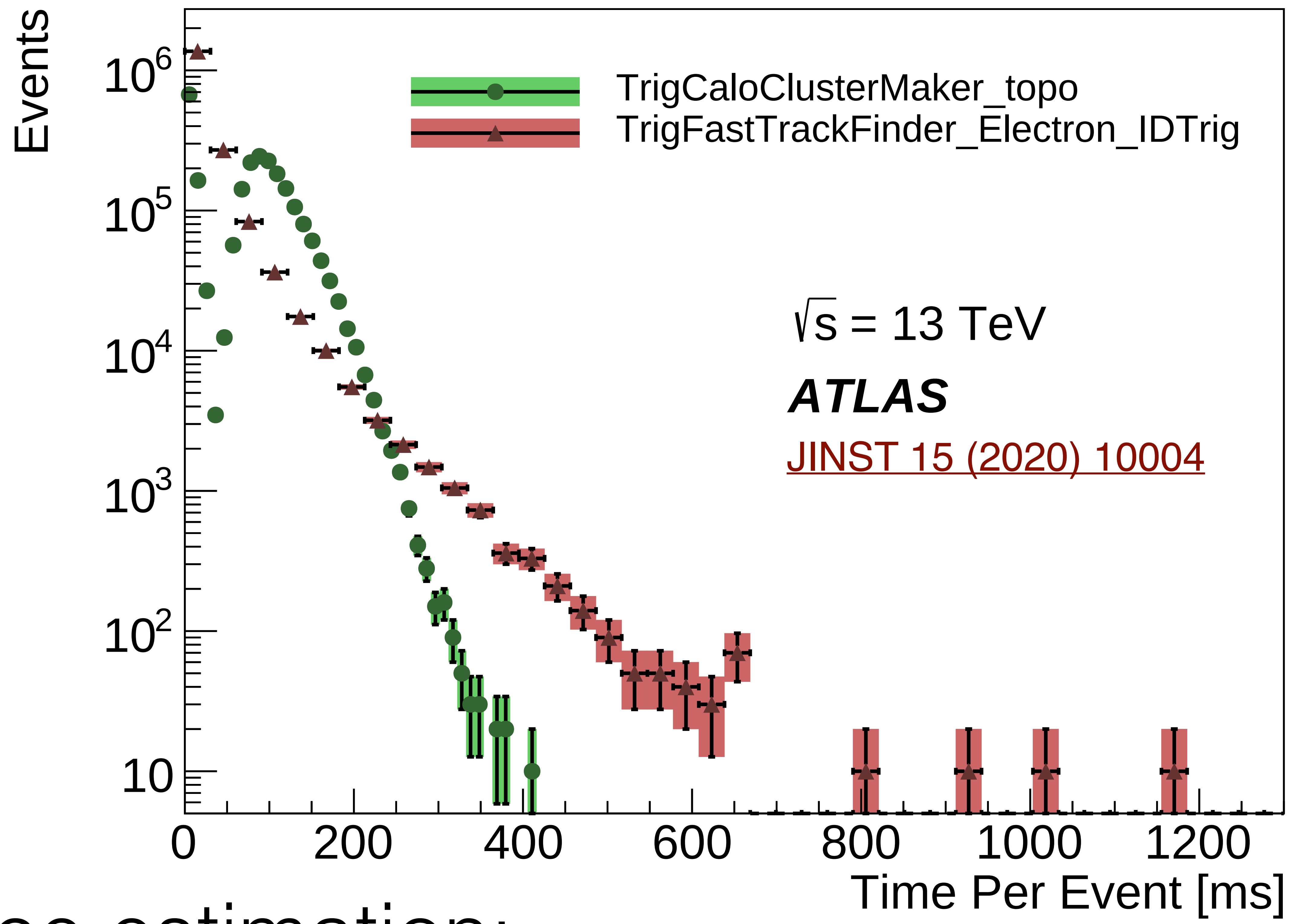
Nightly tests - id trigger branch view

available analyses in branch master - Athena

all_ttbar_pu40	2020-10-15	Fri 16 Oct : 05:10	2020-10-14	Thu 15 Oct : 02:10	2020-10-13	Wed 14 Oct : 07:10	2020-10-12	Tue 13 Oct : 02:10	2020-10-11	Mon 12 Oct : 02:10	2020-10-10	Sun 11 Oct : 05:10
all_ttbar_pu40_mt	2020-10-15	Fri 16 Oct : 01:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 11:10	2020-10-11	Mon 12 Oct : 11:10	2020-10-10	Sun 11 Oct : 11:10
all_ttbar_pu80	2020-10-15	Fri 16 Oct : 07:10	2020-10-14	Thu 15 Oct : 04:10	2020-10-13	Wed 14 Oct : 07:10	2020-10-12	Tue 13 Oct : 03:10	2020-10-11	Mon 12 Oct : 04:10	2020-10-10	Sun 11 Oct : 06:10
all_ttbar_pu80_mt	2020-10-15	Fri 16 Oct : 12:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 12:10	2020-10-11	Mon 12 Oct : 01:10	2020-10-10	Sun 11 Oct : 12:10
all_ttbar_pu80_mt2	2020-10-15	Fri 16 Oct : 12:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 11:10	2020-10-11	Mon 12 Oct : 11:10	2020-10-10	Sun 11 Oct : 11:10
all_ttbar_pu80_mt3	2020-10-15	Fri 16 Oct : 12:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 11:10	2020-10-11	Mon 12 Oct : 11:10	2020-10-10	Sun 11 Oct : 11:10
bjet_pu40	2020-10-15	Fri 16 Oct : 03:10	2020-10-14	Thu 15 Oct : 02:10	2020-10-13	Wed 14 Oct : 03:10	2020-10-12	Tue 13 Oct : 03:10	2020-10-11	Mon 12 Oct : 01:10	2020-10-10	Sun 11 Oct : 01:10
bjet_pu40_mt	2020-10-15	Fri 16 Oct : 12:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 01:10	2020-10-12	Tue 13 Oct : 11:10	2020-10-11	Mon 12 Oct : 11:10	2020-10-10	Sun 11 Oct : 11:10
el_jpsiee_pu40	2020-10-15	Fri 16 Oct : 09:10	2020-10-14	Fri 16 Oct : 12:10	2020-10-13	Thu 15 Oct : 12:10	2020-10-12	Tue 13 Oct : 10:10	2020-10-11	Mon 12 Oct : 07:10	2020-10-10	Sun 11 Oct : 09:10
el_jpsiee_pu40_mt	2020-10-15	Fri 16 Oct : 07:10	2020-10-14	Thu 15 Oct : 04:10	2020-10-13	Wed 14 Oct : 06:10	2020-10-12	Tue 13 Oct : 06:10	2020-10-11	Mon 12 Oct : 02:10	2020-10-10	Sun 11 Oct : 02:10
el_singlee_7-80	2020-10-15	Fri 16 Oct : 06:10	2020-10-14	Thu 15 Oct : 02:10	2020-10-13	Wed 14 Oct : 06:10	2020-10-12	Tue 13 Oct : 04:10	2020-10-11	Mon 12 Oct : 02:10	2020-10-10	Sun 11 Oct : 03:10
el_singlee_7-80_larged0	2020-10-15	Fri 16 Oct : 02:10	2020-10-14	Thu 15 Oct : 04:10	2020-10-13	Wed 14 Oct : 05:10	2020-10-12	Tue 13 Oct : 03:10	2020-10-11	Mon 12 Oct : 03:10	2020-10-10	Sun 11 Oct : 03:10
el_singlee_7-80_larged0_mt	2020-10-15	Fri 16 Oct : 12:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Wed 14 Oct : 04:10	2020-10-12	Tue 13 Oct : 01:10	2020-10-11	Mon 12 Oct : 12:10	2020-10-10	Sun 11 Oct : 12:10
el_singlee_7-80_larged0_pu	2020-10-15	**Sat 17 Oct : 01:10**	2020-10-14	Fri 16 Oct : 04:10	2020-10-13	Thu 15 Oct : 08:10	2020-10-12	Wed 14 Oct : 03:10	2020-10-11	Tue 13 Oct : 03:10	2020-10-10	Sun 11 Oct : 10:10
el_singlee_7-80_larged0_pu_mt	2020-10-15	Fri 16 Oct : 04:10	2020-10-14	Thu 15 Oct : 06:10	2020-10-13	Thu 15 Oct : 03:10	2020-10-12	Tue 13 Oct : 06:10	2020-10-11	Mon 12 Oct : 07:10	2020-10-10	Sun 11 Oct : 10:10
el_singlee_7-80_mt	2020-10-15	Fri 16 Oct : 01:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 12:10	2020-10-11	Mon 12 Oct : 01:10	2020-10-10	Sun 11 Oct : 12:10
el_singlee_7-80_pu40	2020-10-15	Fri 16 Oct : 07:10	2020-10-14	Thu 15 Oct : 04:10	2020-10-13	Wed 14 Oct : 05:10	2020-10-12	Tue 13 Oct : 04:10	2020-10-11	Mon 12 Oct : 03:10	2020-10-10	Sun 11 Oct : 10:10
el_singlee_7-80_pu40_mt	2020-10-15	Fri 16 Oct : 01:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 01:10	2020-10-11	Mon 12 Oct : 02:10	2020-10-10	Sun 11 Oct : 12:10
el_zee_pu40	2020-10-15	**Sat 17 Oct : 05:10**	2020-10-14	Fri 16 Oct : 04:10	2020-10-13	Thu 15 Oct : 07:10	2020-10-12	Wed 14 Oct : 05:10	2020-10-11	Tue 13 Oct : 04:10	2020-10-10	Mon 12 Oct : 02:10
el_zee_pu40_mt	2020-10-15	Fri 16 Oct : 05:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Thu 15 Oct : 04:10	2020-10-12	Tue 13 Oct : 04:10	2020-10-11	Mon 12 Oct : 04:10	2020-10-10	Sun 11 Oct : 05:10
fsjet_ml_pu40	2020-10-15	Fri 16 Oct : 04:10	2020-10-14	Thu 15 Oct : 03:10	2020-10-13	Wed 14 Oct : 05:10	2020-10-12	Tue 13 Oct : 02:10	2020-10-11	Mon 12 Oct : 02:10	2020-10-10	Sun 11 Oct : 01:10
fsjet_pu40	2020-10-15	Fri 16 Oct : 02:10	2020-10-14	Thu 15 Oct : 02:10	2020-10-13	Wed 14 Oct : 06:10	2020-10-12	Tue 13 Oct : 02:10	2020-10-11	Mon 12 Oct : 03:10	2020-10-10	Sun 11 Oct : 01:10
fsjet_pu40_mt	2020-10-15	Fri 16 Oct : 01:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 03:10	2020-10-12	Tue 13 Oct : 03:10	2020-10-11	Mon 12 Oct : 11:10	2020-10-10	Sun 11 Oct : 12:10
minbias	2020-10-15	Fri 16 Oct : 12:10	2020-10-14	Thu 15 Oct : 12:10	2020-10-13	Wed 14 Oct : 01:10	2020-10-12	Tue 13 Oct : 02:10	2020-10-11	Mon 12 Oct : 11:10	2020-10-10	Sun 11 Oct : 11:10
minbias_mt	2020-10-15		2020-10-14		2020-10-13		2020-10-12		2020-10-11		2020-10-10	
mu_bphys	2020-10-15	Fri 16 Oct : 08:10	2020-10-14	Thu 15 Oct : 07:10	2020-10-13	Wed 14 Oct : 01:10	2020-10-12	Tue 13 Oct : 04:10	2020-10-11	Mon 12 Oct : 06:10	2020-10-10	Sun 11 Oct : 02:10
mu_bphys_mt	2020-10-15	Fri 16 Oct : 02:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 01:10	2020-10-11	Mon 12 Oct : 12:10	2020-10-10	Sun 11 Oct : 11:10
mu_singlemu_larged0	2020-10-15	**Sat 17 Oct : 04:10**	2020-10-14	Fri 16 Oct : 04:10	2020-10-13	Thu 15 Oct : 06:10	2020-10-12	Wed 14 Oct : 04:10	2020-10-11	Tue 13 Oct : 01:10	2020-10-10	Mon 12 Oct : 01:10
mu_singlemu_larged0_mt	2020-10-15	Fri 16 Oct : 08:10	2020-10-14	Thu 15 Oct : 08:10	2020-10-13	Wed 14 Oct : 09:10	2020-10-12	Tue 13 Oct : 06:10	2020-10-11	Mon 12 Oct : 03:10	2020-10-10	Sun 11 Oct : 08:10
mu_singlemu_larged0_pu	2020-10-15	Fri 16 Oct : 03:10	2020-10-14	Thu 15 Oct : 05:10	2020-10-13	Thu 15 Oct : 12:10	2020-10-12	Tue 13 Oct : 05:10	2020-10-11	Mon 12 Oct : 04:10	2020-10-10	Sun 11 Oct : 03:10
mu_singlemu_larged0_pu_mt	2020-10-15	Fri 16 Oct : 03:10	2020-10-14	Thu 15 Oct : 07:10	2020-10-13	Wed 14 Oct : 02:10	2020-10-12	Tue 13 Oct : 01:10	2020-10-11	Mon 12 Oct : 12:10	2020-10-10	Sun 11 Oct : 03:10
mu_zmumu_pu40	2020-10-15		2020-10-14		2020-10-13		2020-10-12		2020-10-11		2020-10-10	
mu_zmumu_pu40_mt	2020-10-15	Fri 16 Oct : 02:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Wed 14 Oct : 04:10	2020-10-12	Tue 13 Oct : 01:10	2020-10-11	Mon 12 Oct : 01:10	2020-10-10	Sun 11 Oct : 12:10
mu_zmumu_pu40_mt-old	2020-10-15	Fri 16 Oct : 04:10	2020-10-14	Thu 15 Oct : 03:10	2020-10-13	Wed 14 Oct : 06:10	2020-10-12	Tue 13 Oct : 02:10	2020-10-11	Mon 12 Oct : 01:10	2020-10-10	Sun 11 Oct : 12:10
tau_ztautau_pu46	2020-10-15	Fri 16 Oct : 11:10	2020-10-14	Fri 16 Oct : 12:10	2020-10-13	Wed 14 Oct : 08:10	2020-10-12	Tue 13 Oct : 10:10	2020-10-11	Mon 12 Oct : 05:10	2020-10-10	Sun 11 Oct : 06:10
tau_ztautau_pu46_mt	2020-10-15	Fri 16 Oct : 02:10	2020-10-14	Thu 15 Oct : 01:10	2020-10-13	Wed 14 Oct : 04:10	2020-10-12	Tue 13 Oct : 02:10	2020-10-11	Mon 12 Oct : 01:10	2020-10-10	Sun 11 Oct : 12:10

ATLAS Trigger





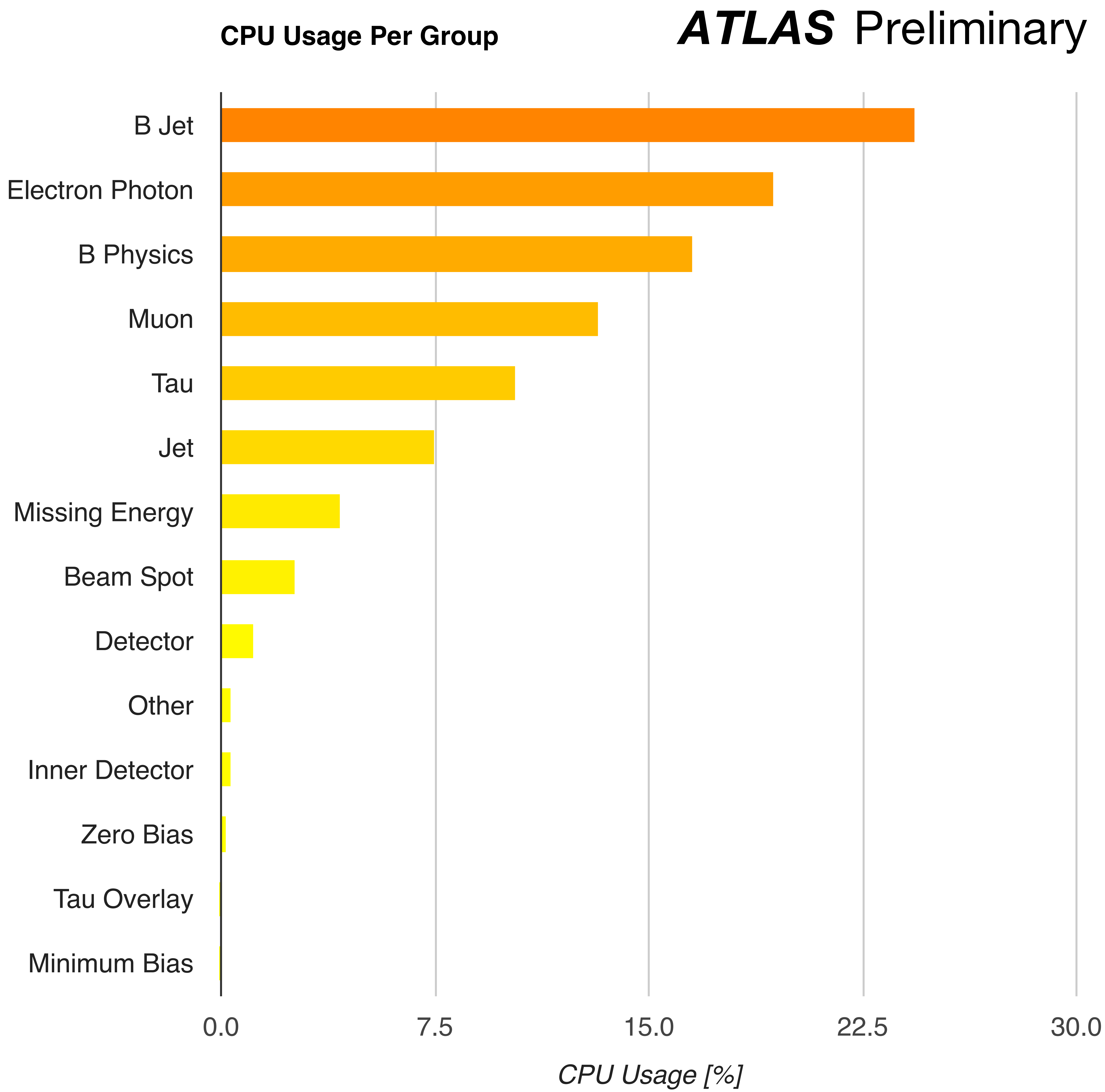
Resource estimation: Cost monitoring

- For the cost monitoring In 10 % of events during specific Lumi Blocks the timing information for **all algorithms** is sampled and written to the event stream
  - Typically the first three Lumi Blocks after luminosity is declared, and at regular intervals of 150 LumiBlocks
  - A long run will typically generate 2.75 TB of cost data
- Book keeping data stored for several different categories
  - Number of RoIs per chain
  - Algorithm timing per call and per event, and per call for each chain
  - Chain time per event

Resource estimation for Run 2

PUB-TRIG-2016-02

- B-jet reconstruction the most costly overall
  - With these initial estimates this was driven by the large tracking burden
  - Inner Detector apparently small
    - Most ID Trigger operation included already within the other signatures
    - Unique ID Trigger contributions from dedicated chains
- By studying cost data can determine which algorithms are most costly
  - Allows chains and stages to be reordered to reduce the rate at which we run costly algorithms
  - Identify which algorithms are most worth trying to speed up



Enhanced bias data

PUB-TRIG-2016-02

- In practice five dedicated HLT chains are added to the menu for the Enhanced Bias data with a large pass through rate
  - Approximately 300 Hz of additional triggers for 1 hour in the run, for around 10^6 Enhanced bBias events
  - Trigger menu is invertible - ie a single event weight $w_{\mathrm{EB}}(e)$ can be computed for each individual event to correct for the prescales $w_C(e)$ for the chains used at Level 1 and restore the zero bias spectrum

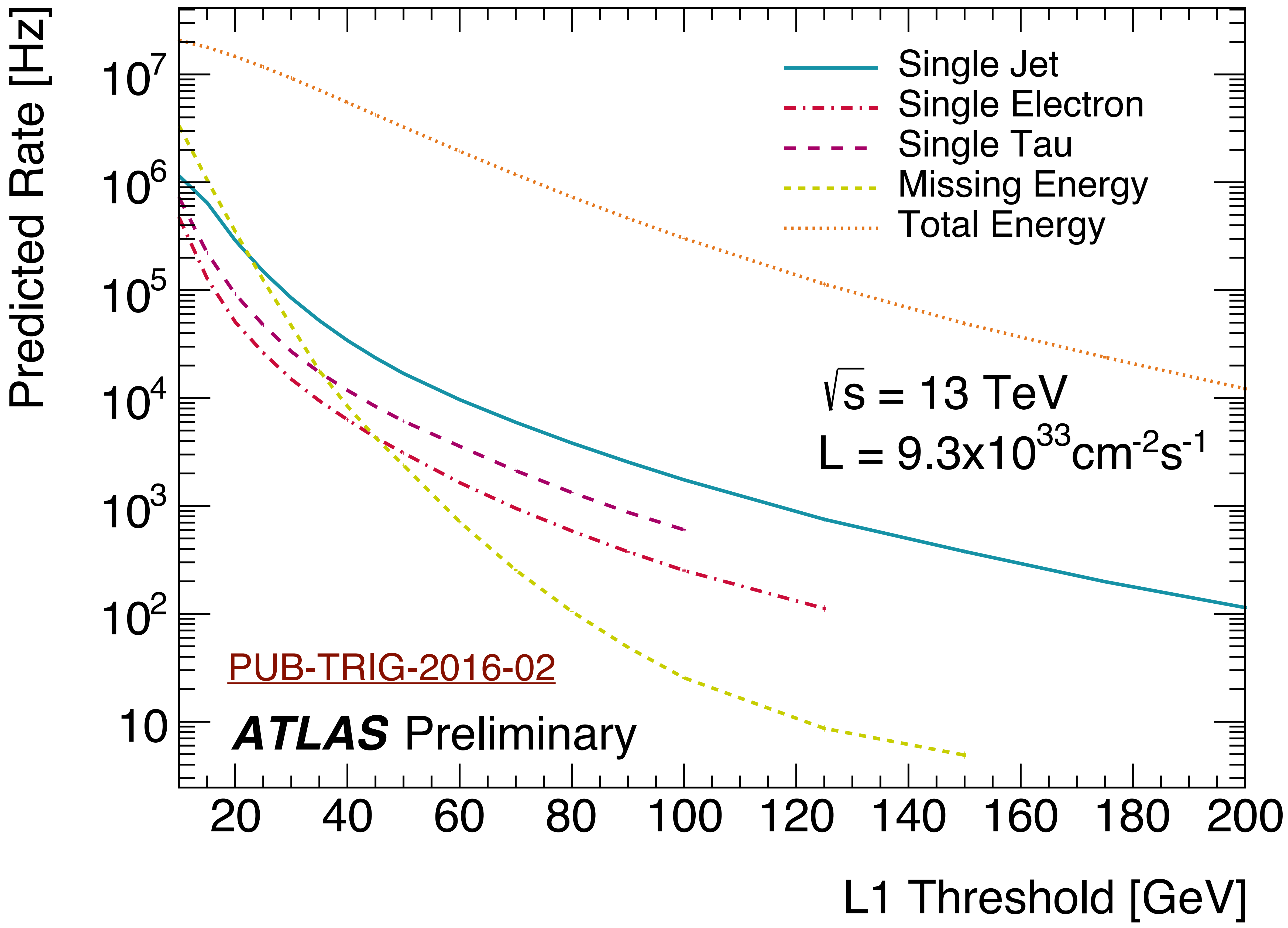
$$\mathrm{Rate}(c) = \frac{\sum_{e=1}^{N} w_{\mathrm{EB}}(e) w_C(e)}{\Delta t} \qquad \frac{1}{w_{\mathrm{EB}}(e)} = 1 - \prod_{j=1}^{\mathrm{EBChains}} \left[1 - \frac{r_{je}}{p_j}\right]$$

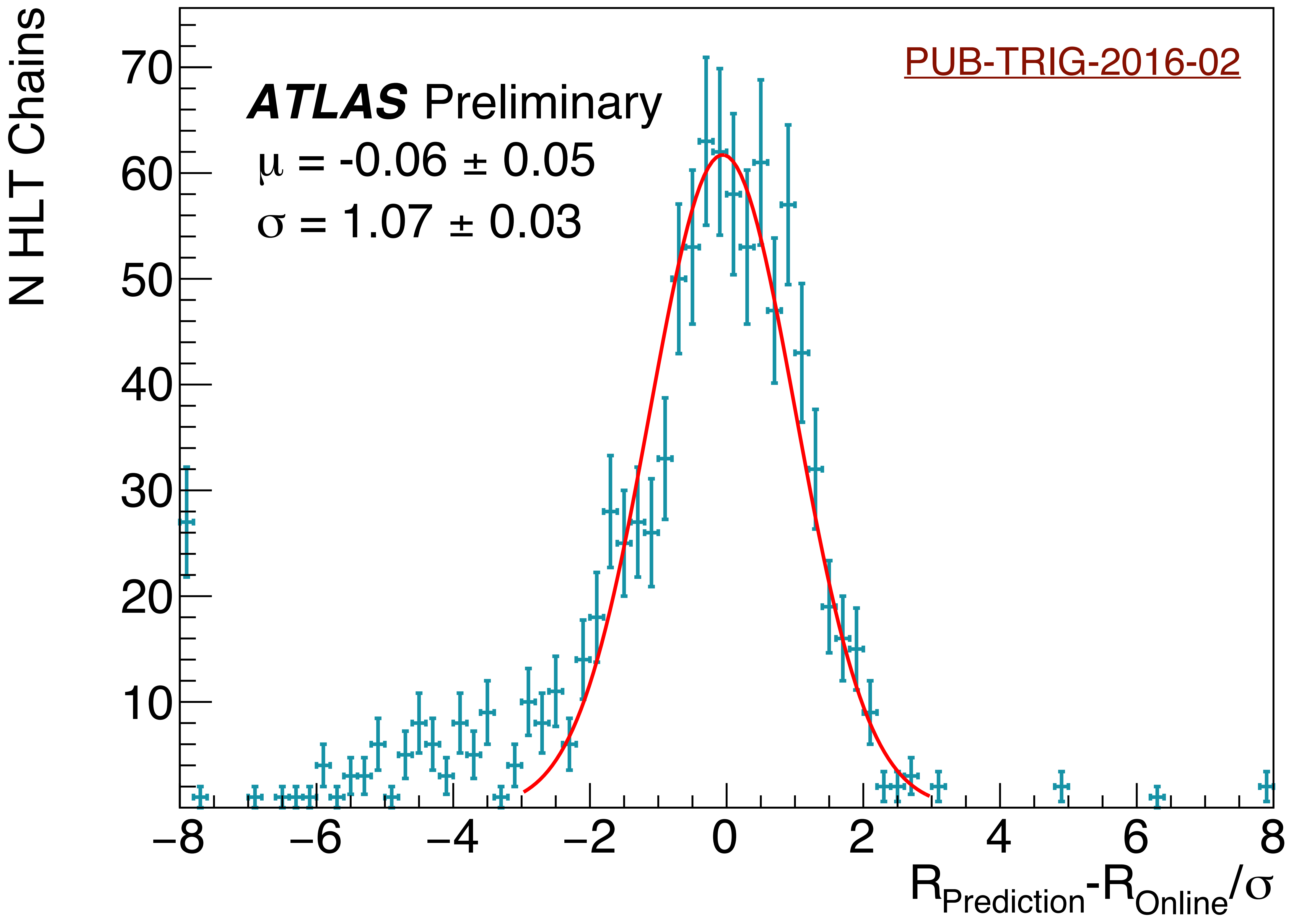
  - with raw chain rate r_{je} for each of the EB chains with prescale p_j

Name	**Seeding**	**Output [Hz]**	**L1 Seed Rates [kHz]**
Random	Random	60	> 500
Low	Random	60	50–500
Medium	Random	60	20–50
Primary	Direct	110	0.1–20
High	Direct	10	< 0.1

- When reprocessing the data on the grid, the events are all processed with no prescale for rate estimation, but with prescale for CPU estimation

- Subsequently, any set of prescales can be applied to the rates from each trigger to estimate the actual trigger rate for that percale set





Rate and CPU estimation

- The overall rates can be estimated with some degree of accuracy
- For CPU estimation, prescales should be applied during execution, sine the trigger caches the results for an algorithm running in an RoI if that RoI has already been processed by that algorithm for a different chain
  - Need to get the number of times each RoI is processed correct, else the overall time all be incorrect
  - Converting the prescale corrected processing mean time per event to a rate estimate can determine the mean number of processor cores required to execute a given HLT chain, or full menu

Outlook

- For the Run 3, had planned to make use of hardware accelerated tracking - the Fast Tracker (FTK), which was however cancelled
  - Attempting to extend the software ID Trigger to address the use cases which would have benefited from the hardware tracking
  - Currently modifying the ID Trigger code and evaluating the performance to see how fast we can make it
    - Also complete redesign of the ATLAS software framework to run multi threaded code - Gaudi Hive
  - The cost monitoring is being used to evaluate the performance, and all the tools described used to estimate the required resources wit the new software
    - Still reasonably early days
      - Code still under development
      - Trigger Menu still evolving
    - Estimates available within the ATLAS collaboration but until everything has stabilised, too soon to present to the wider community

- What lessons would we learn for the future ?
  - Nightly testing is very important for the very fine grained study of the **performance**, and **execution times** for the individual algorithms across the entire trigger, not just the ID Trigger
  - Frequent reprocessing of Enhanced Bias runs is essential for estimating the overall CPU load for the full Trigger Menu, with realistic prescales
  - Important to understand the behaviour as a function of the pile up — combinatoric algorithms do not scale linearly with pile up
  - Early rejection in the trigger is important — can experiment by the movement of different algorithms, most costly algorithms should be executed less frequently
    - Can be run in smaller regions for preselection to achieve even faster execution

- All of these would be important for any experiment, with any trigger
  - Perhaps the behaviour with larger pile up is less relevant for a lepton collider, but detailed evaluation and estimation of CPU load will be very important for any high rate trigger system