Report on Geant4 Usage in ATLAS

Vangelis Kourlitis

on behalf of the ATLAS Collaboration

January 20th 2022

Synopsis

Overview of Geant4 in ATLAS

Geant4 Optimization Task Force

Implemented & Validated

Ongoing R&D

New Ideas

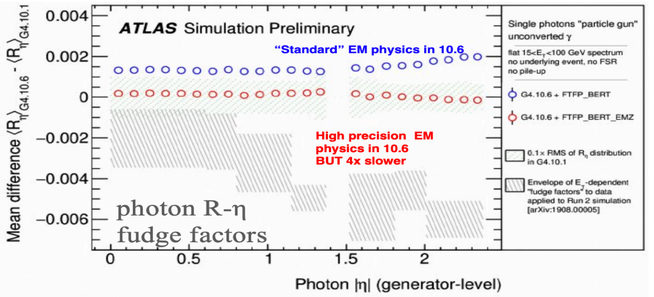
Geant4 Version

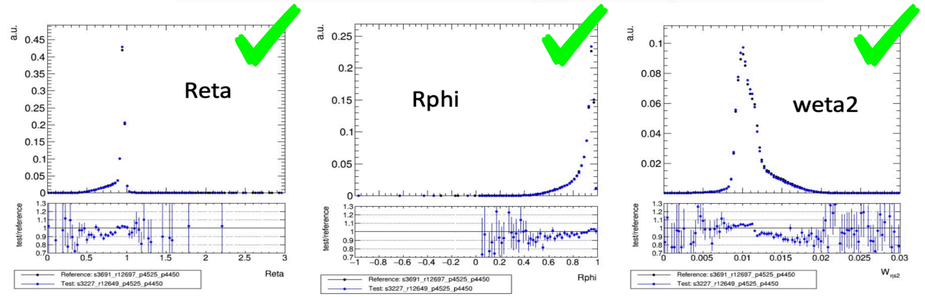
- For **Run 2 samples Geant4 10.1.patch03.atlas07** has been used in `x86_64-centos7-gcc62-opt` platform
- For **Run 3** ATLAS considers two versions in `x86_64-centos7-gcc11-opt` (C++17) platform:

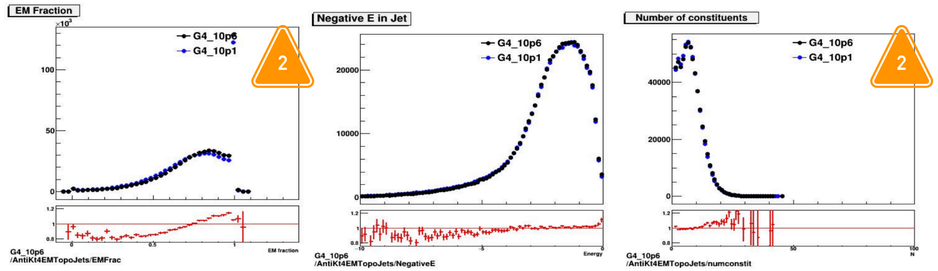
1. **Geant4 10.6.patch03.atlas03 (first samples for calibration)**
   - Ready to be used in Athena master
     - `atlas01`: AtlasRK4 stepper
     - `atlas02`: Magnetic integration driver patch
       to make comparison with G4 10.1 easier
     - `atlas03`: G4GammaGeneralProcess fix
       back-ported from G4 10.7
   - Physics differ - *not necessarily better for ATLAS*
     1. E/γ shower shapes in agreement
     2. Jet EM fraction and constituents increase
     3. Jet response decrease, opposed to data for |η| > 1.2
   - Birks' coefficient and physics list tuning ongoing
     (following G4 recommendations)

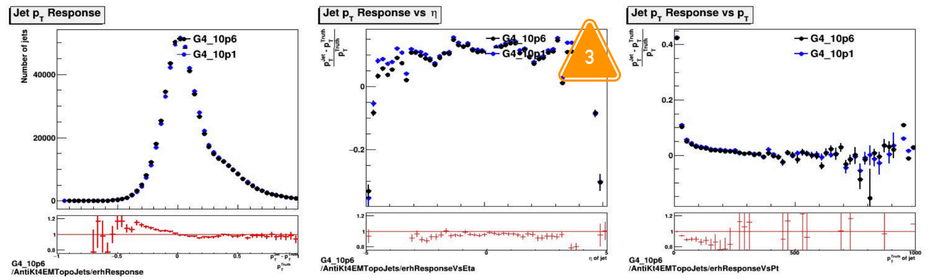
2. **Geant4 10.7.patch02.atlas01 (bulk production?)**
   - Physics ~ same as 10.6
   - Some differences are seen and currently under investigations

Geant4 10.6 Validation









Calorimeter Test Beam Integration to Geant Val

Geant4 validation program

Automatically validate Geant4 using hadronic and electromagnetic calorimeters test-beam data

Lorenzo Pezzotti & Alberto Ribon

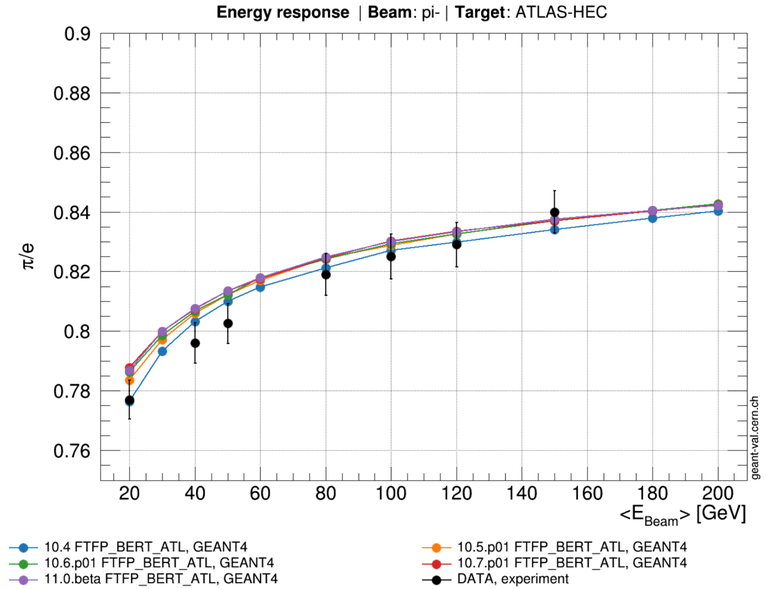
ATLAS data considered:

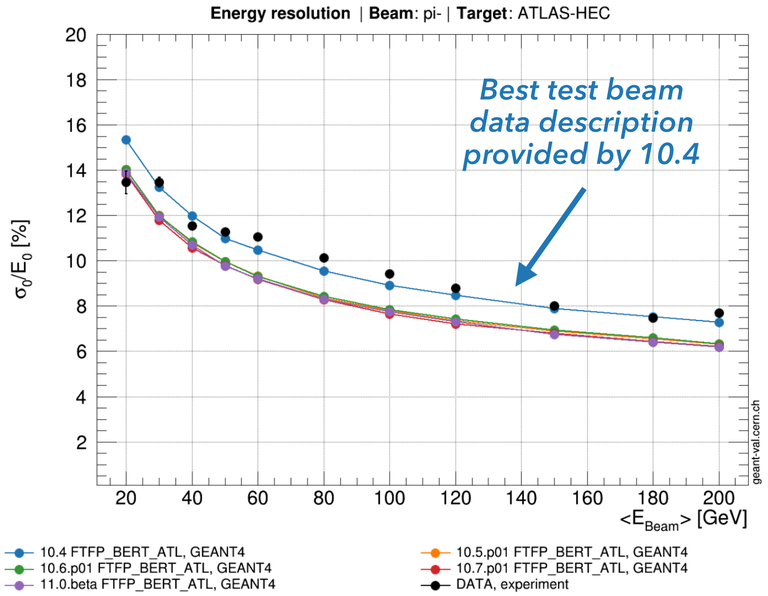
- Hadronic Endcap Calorimeter (HEC)
- Tile Calorimeter (TileCal)

Workflow:

1. Port the ATLAS HEC simulation into a new standalone Geant4 simulation
2. Perform Geant4 validation against the ATLAS HEC test-beam data
3. Porting the application into the Geant Val testing suite

Excellent example of collaboration between ATLAS and Geant4!

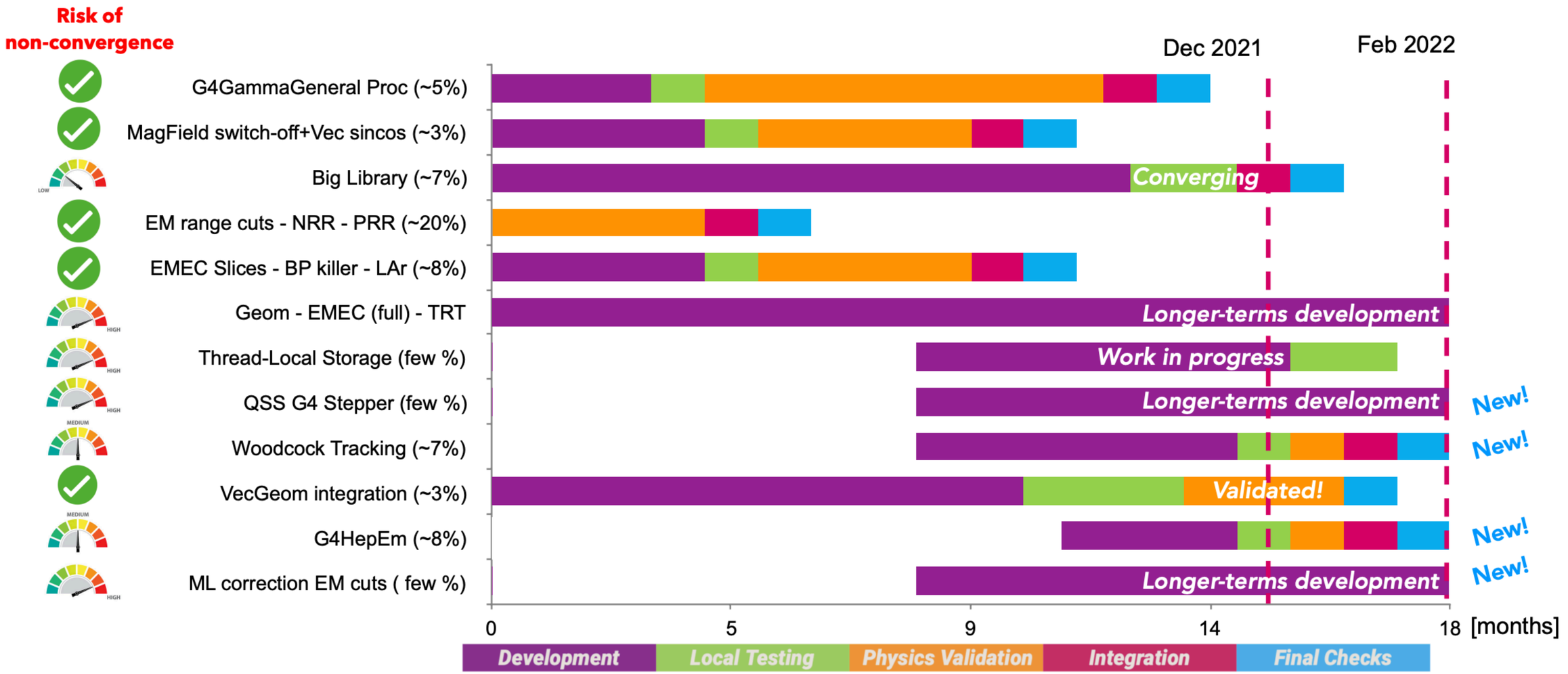




Geant4 Optimization Task Force

Geant4 Optimisation Task Force responsible for optimising the performance of the ATLAS Geant4 simulation software

One-year mandate to achieve for Run 3 >30% CPU time speed up w.r.t the comparable Run-2 simulation



Integrated & Validated

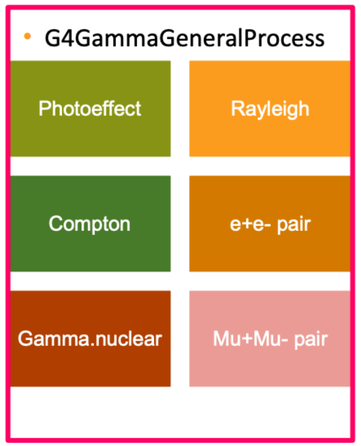
Intrinsic Geant4 Improvements

Gamma General Process

SteppingManager sees only 1 physics process for photons → reduced number of instructions

-4.3%

measured on 100 ttbar events in Athena

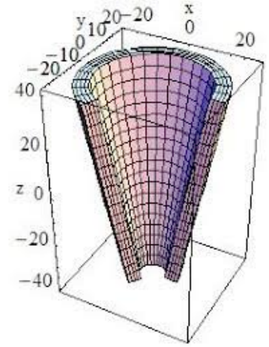


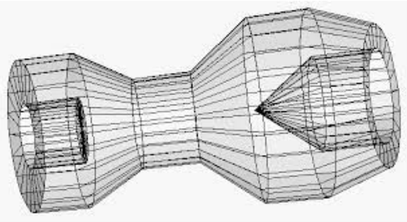
VecGeom

speed up using internal vectorisation for CPU – just for `G4Cons` & `G4Polycone`, no speed up measure considering all shapes

-1.5%

measured on 500 ttbar events in Athena

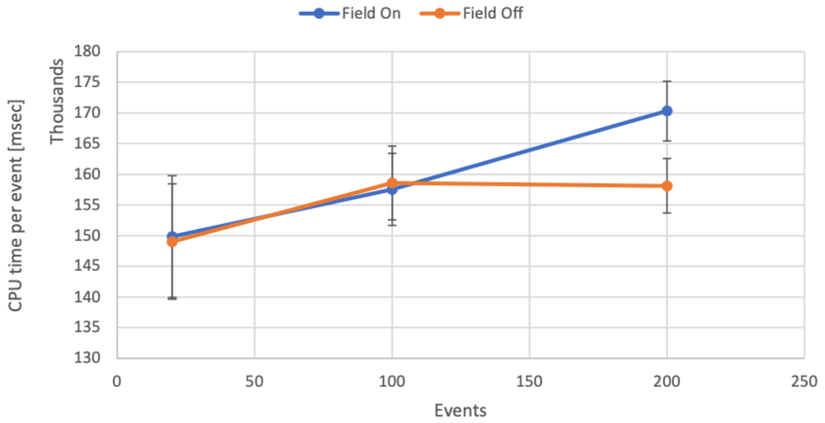




Reducing Operations

Magnetic Field Tailored Switch-OFF

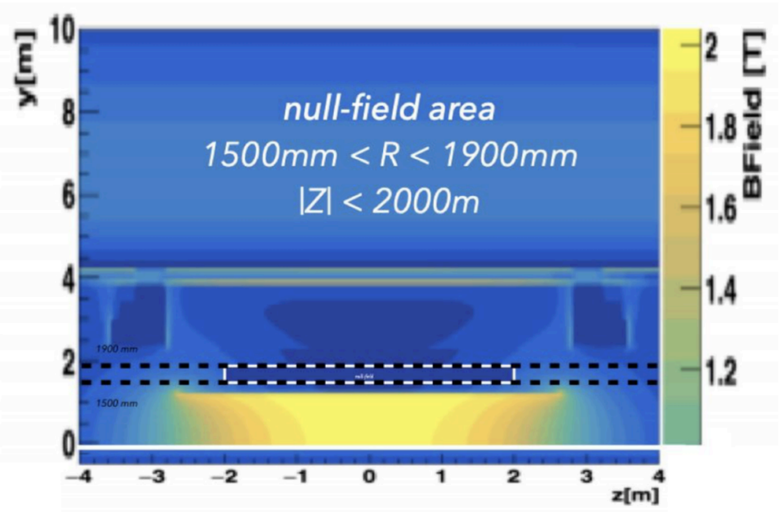
Speed up observed when switching-off magnetic field in LAr calorimeter (except for muons) without affecting shower shapes



Detailed studies showed smaller null-field area needed

- ~3% speed up for full ttbar events
- ~7% speed up for 1GeV e- on 0<n<0.17

Possibility to extend solution to other detector regions too



Vectorized sin/cos calculation in EMEC

calculates both sine and cosine in ElectroMagnetic EndCap geometry for a given radius, vectorization reduces operations needed

Both stand-alone and Athena timing shows a ~20% speed up in `LArWheelCalculator`.

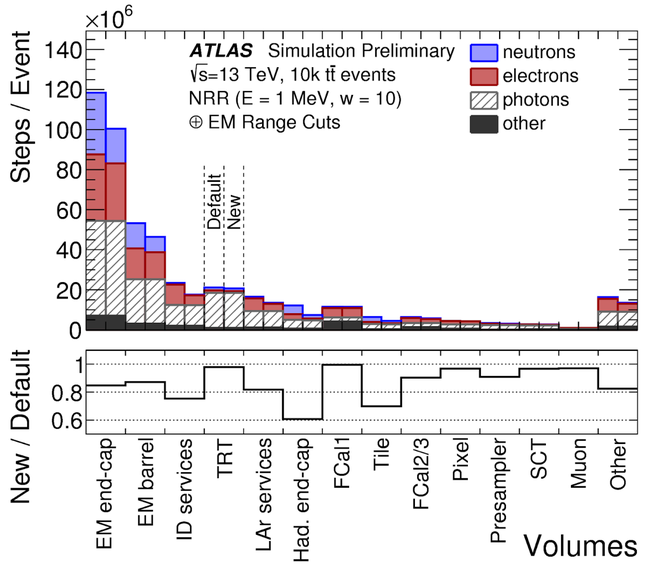
Difficult to assess overall speed up

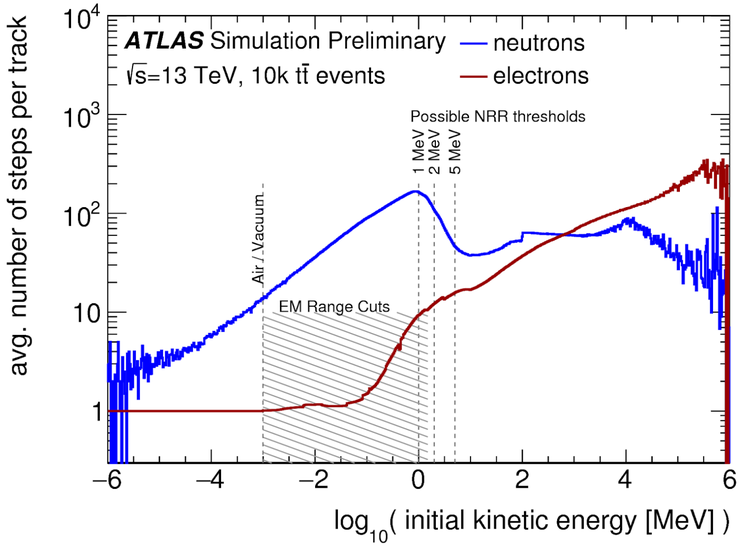
`LArWheelCalculator::parameterized_sincos` takes only ~1.5% of total CPU time

Russian Roulettes & EM Range Cuts

Russian Roulette

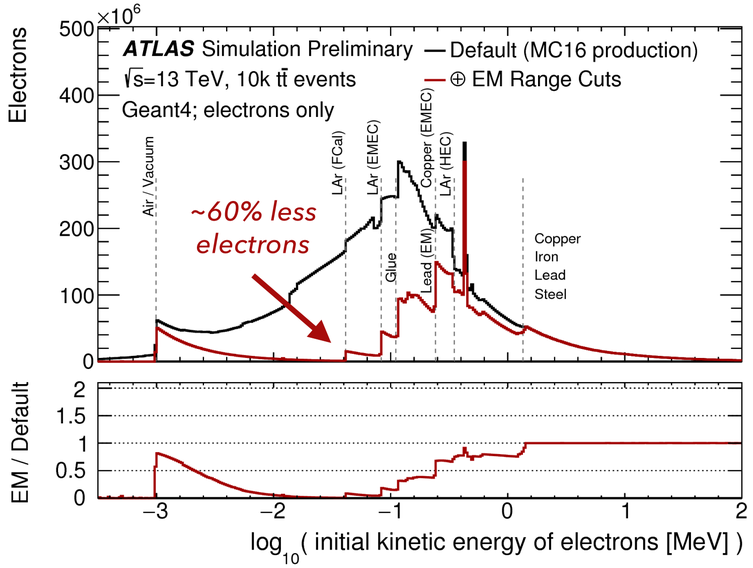
- Neutrons and photons take majority of CPU time
  EMEC most resource intensive
- Photon/Neutron Russian Roulette (PRR/NRR): randomly discard particles below energy threshold and weight the energy deposits of remaining particles accordingly
- NRR performance: 10% speed up with 2 MeV threshold for neutrons





EM Range Cuts

- OFF by default for three processes:
  Compton, conversion, photo-electric effect
- Turning them on provide ~6-7% speed up with negligible impact on physics



SIM-2019-001

Simplifying Geometries *(aka reducing G4Polycone usage)*

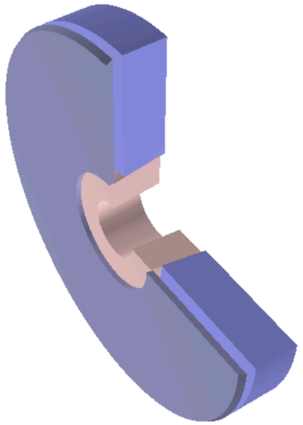
EMEC

Described by a custom Geant4 solid using `G4Polycone` for internal calculations (Bounding Shape). Re-Implemented custom solid variants:

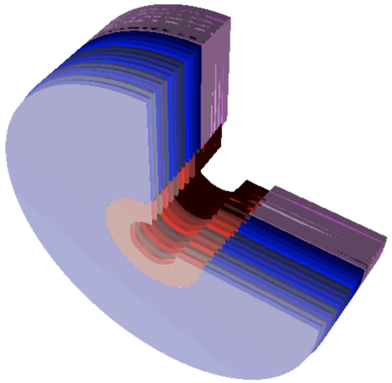
- Wheel: the default with G4Polycone
- Cone: improved shape using G4ShiftedCone – outer wheel divided into two conical-shaped sections
- Slices: new LArWheelSliceSolid – each wheel is divided into many thick slices along Z axis

Chosen variant Slices provided 5-6% speed up

Cone



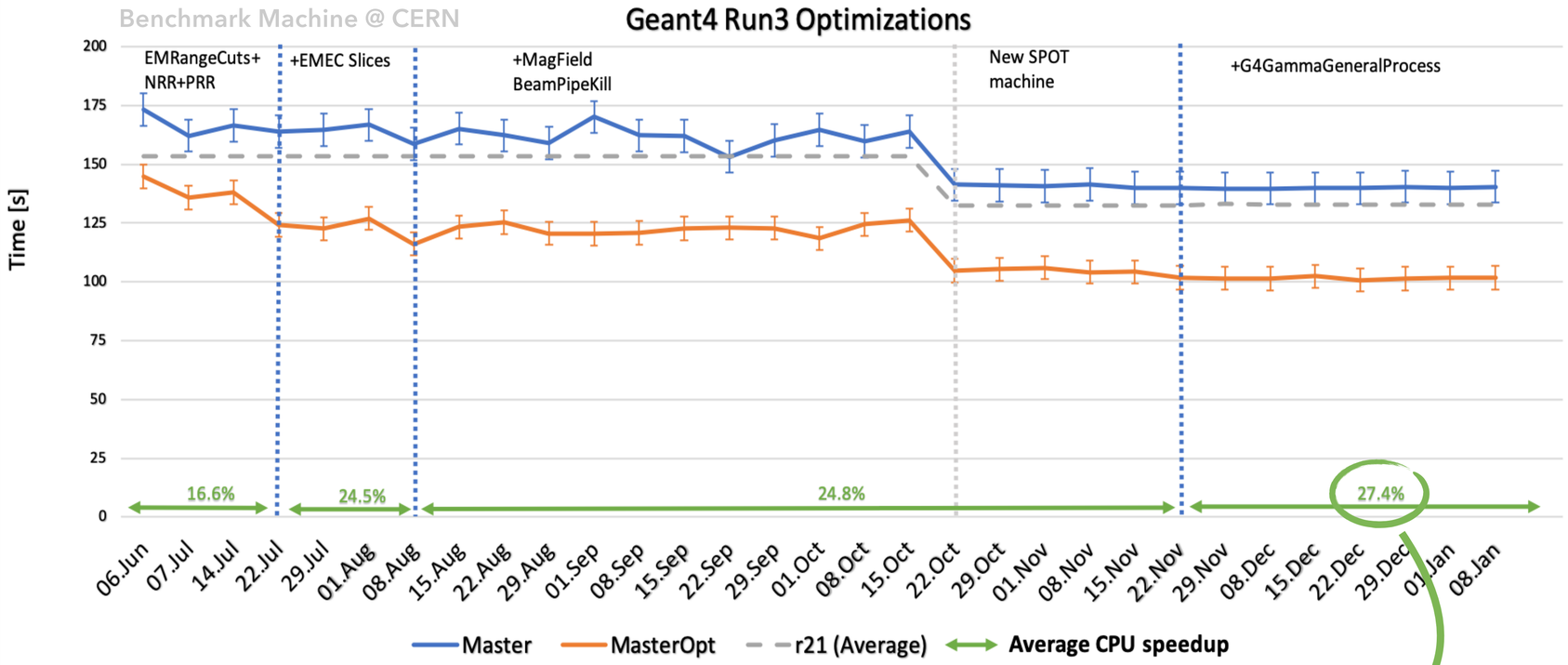
Slices



Geant4 Optimizations Benchmark

Close to achieve the target performance improvement

Improvement is also observed in "realistic" production conditions – grid sites

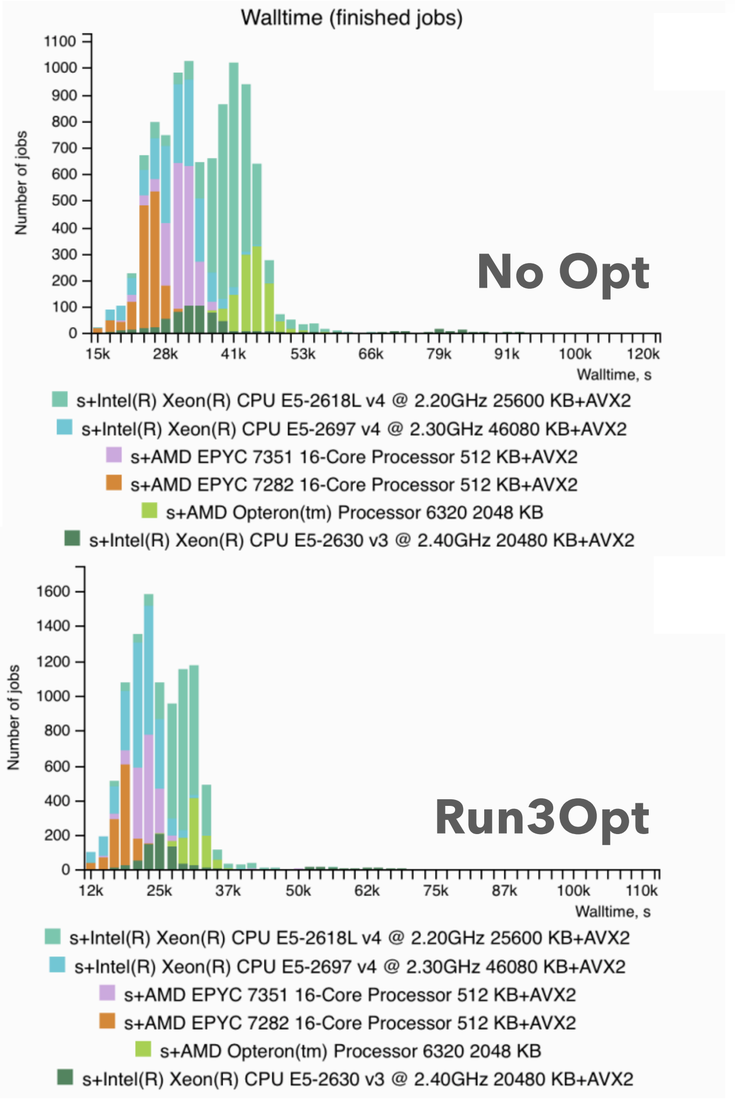


Corresponding to 38% higher throughput

"can now simulate 1.38 times more events using the same computational resources"

*CPU time speed up=[(t2-t1)/t1]*100*
*Throughput speed up=[(t1-t2)/t2]*100

Grid Benchmarks



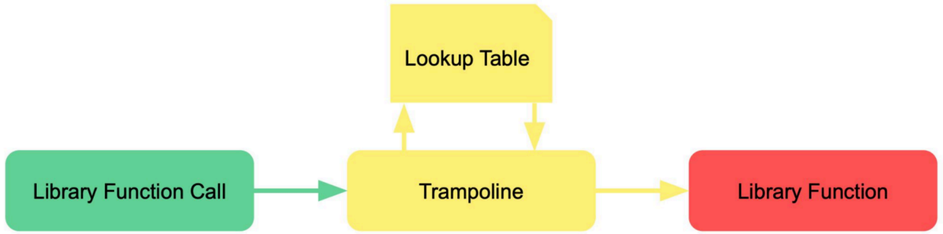
Ongoing R&D

Non-Physics Improvements

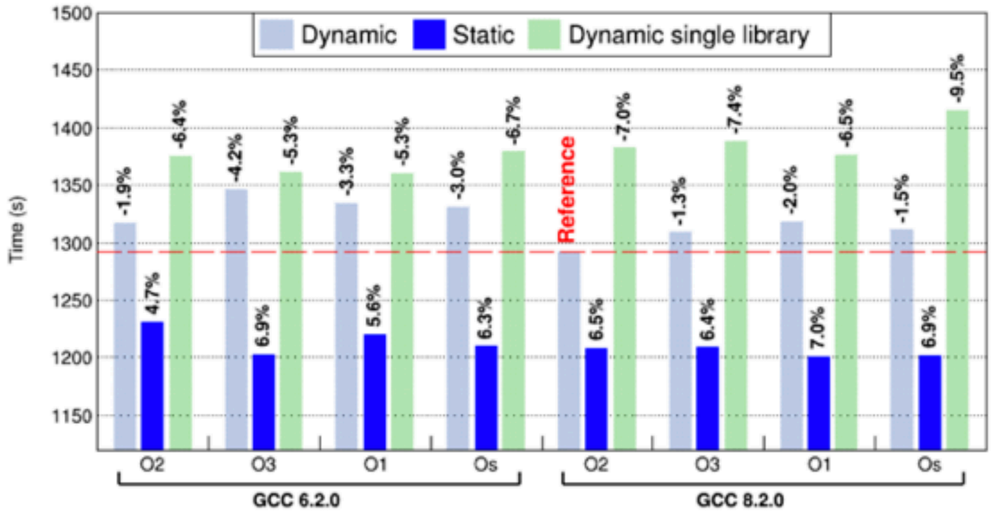
Big (static) Library

converging!

- Use Geant4 as static library(ies) to avoid "trampolines"



- Define a `BigSimulation` SHARED library, as a grouping of all libraries from packages that use Geant4



HepExpMT benchmark (Geant4 10.5.1) show 6-7% speed up

integration/testing into Athena ongoing ***no validation needed!***

Thread Local Storage (TLS)

- Athena profiling showed bottlenecks from usage of TLS
  - Going to MT in Athena/G4 cost ~5-10% due to TLS
- Both Athena & Geant4 are using TLS:
  - Athena → magnetic field
  - Geant4 → geometry data
- Work on reducing TLS usage is on-going from both sides
  - Athena → performance bug fix
  - Geant4 → investigating code restructure

TRT Geometry Optimization

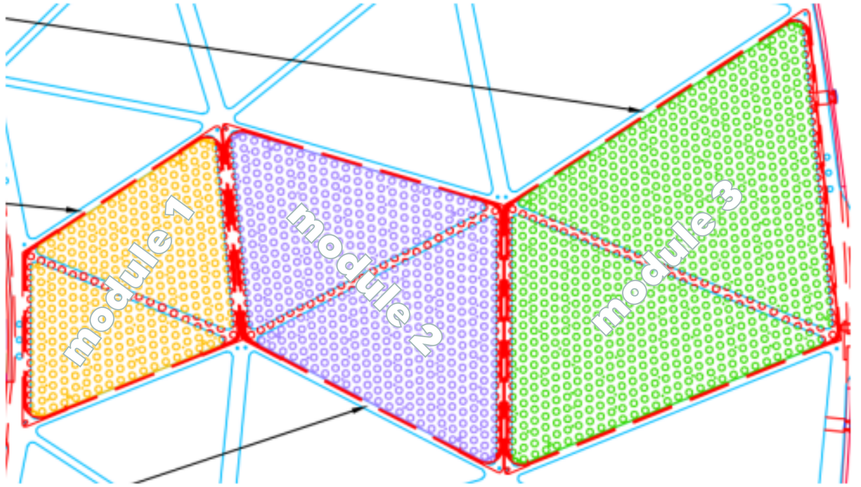
Currently the TRT geometry is described using **Boolean operations**

This approach is not optimal as Boolean operations are slow and they can cause tracking issues especially in presence of coincident surfaces

Describe these volumes using alternative shapes:

1. **arbitrary trapezoid (Arb8)**
   requires a total of 8 points to be specified – 4 vertices belonging to the −h/2 plane and 4 points belonging to the +h/2 plane
2. **the Boundary REPresentation (BRep)**
   requires the 4 vertices describing the trapezoid cross-section to be specified

JINST 3 P02014 (2008



96 trapezoidal modules grouped in 3 types characterized by an increasingly larger cross sectional area

Module shapes	Execution time (s)	Improvement
Boolean solids	1663	Reference
Arb8	1638	1.5%
BRep	1675	−0.7%

A speed up of 1.5% is observed for the **Arb8** representation, whereas the **BRep** solid exhibits a minor slowdown with respect to the reference boolean solids

GPU-Friendly EMEC

Description of the EMEC with Geant4/VecGeom standard shapes

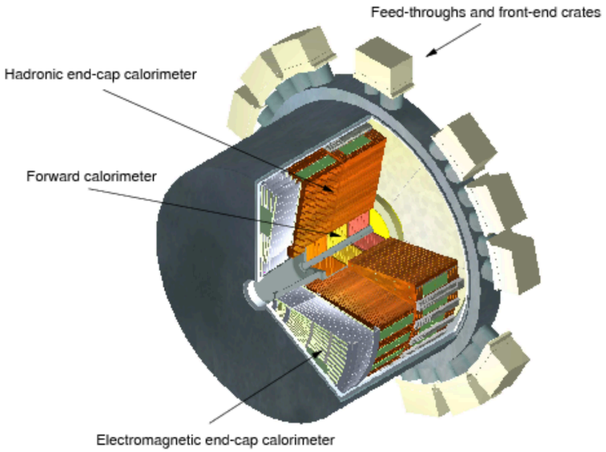
no accordion shape within the GEANT4 standard geometry shapes, defined a custom solid

- Possible speed up in VecGeom on CPU making use of internal vectorisation
- Possibility for the ATLAS geometry to be standard and GPU-friendly (see AdePT project)

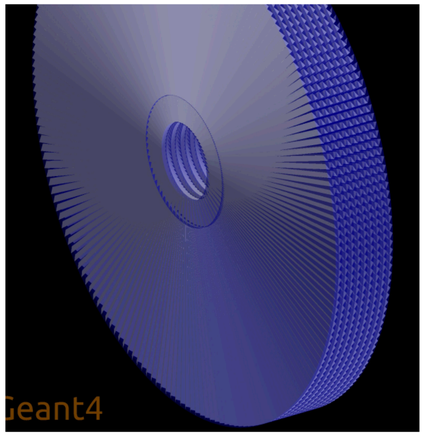
Status

- Repeated accordion volume implementations using:
  1. `G4GenericTrap` (converted from `G4TwistedTrap`)
  2. `Arb8` & `G4Trap`
- Good progress overall

▸ Repository: https://gitlab.cern.ch/avishwak/atlas_emec_g4



Wheel sliced into discs along z-axis



Benchmark run in FullsimLight for G4Trap and G4GenericTrap using 1000 events with 10 GeV electrons

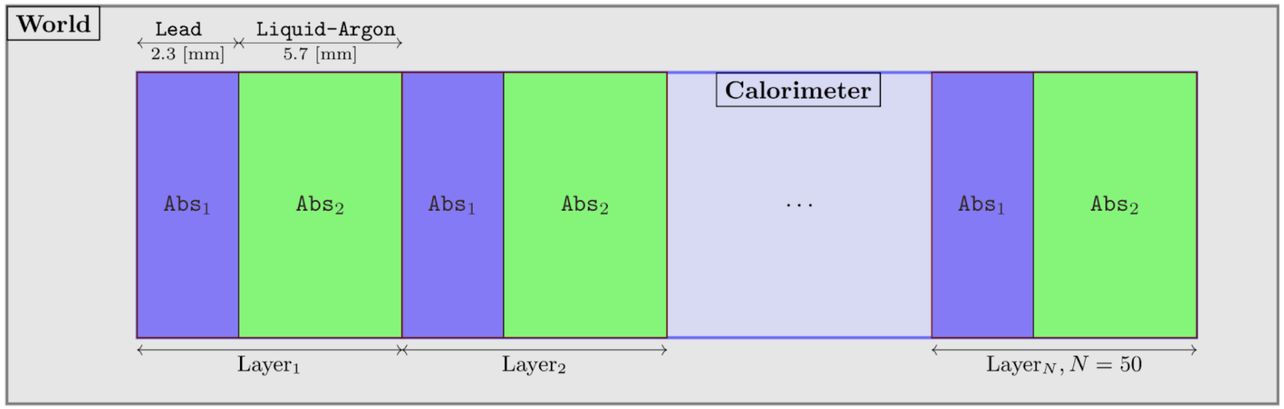
Geometry	Time (s)
G4Trap	111.67
G4GenericTrap	72.07

WOODCOCK TRACKING

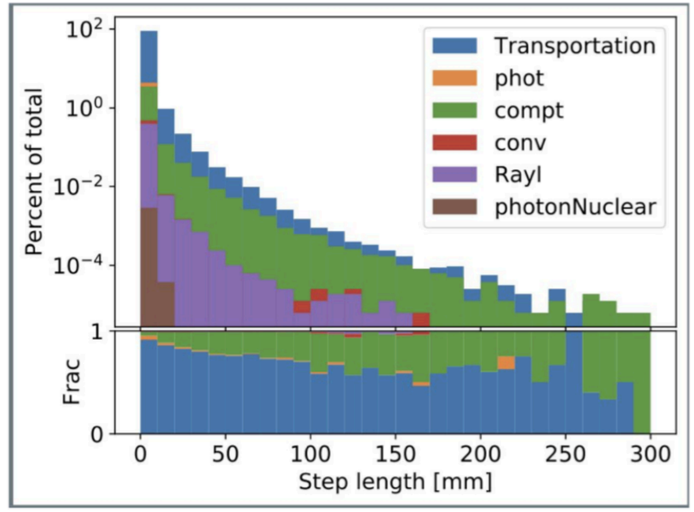
Reducing CPU time without approximations

Idea proposed by John Apostolakis

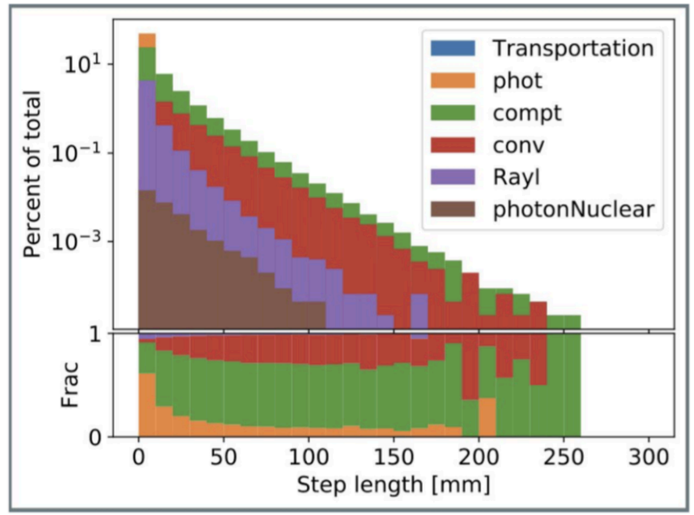
- Especially powerful in highly granular detectors (e.g, the EMEC) where geometric boundaries limit steps, rather than interactions
- Performs tracking in geometry with one material: the densest (Pb)
- Interaction probability is proportional to the cross section ratio between the real material and Pb
- Avoids many steps caused by geometric boundaries (*Transportation*) since there are no boundaries
- Up to 10% computational speed improvement for simplified layered Pb/LAr calorimeter (FullSimLight example by Mihaly Novak – image)
- Implementation for ATLAS EMEC ongoing



EMEC Total



EMEC Lead Only



Quantized State System Stepper

Background

- Quantized State System (QSS) numerical methods to solve the ordinary differential equations that govern the movement of particles in a field.
- QSS methods discretize the system state variables as opposed to traditional methods that discretize the time.
- Very efficient handling of discontinuities in the simulation of continuous systems.
- Based on: Efficient discrete-event based particle tracking simulation for high energy physics

Status

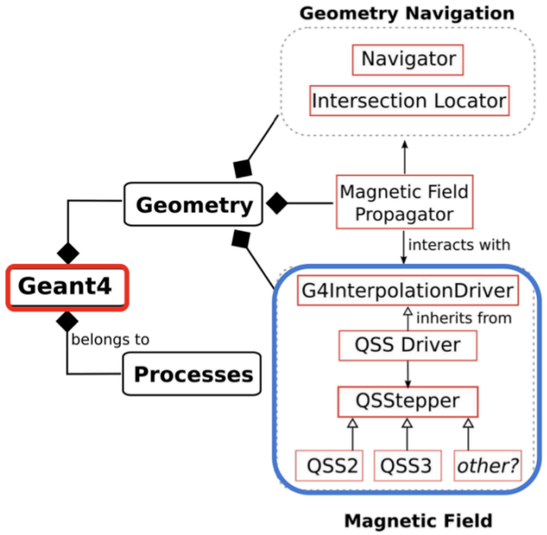
- Successfully ported QSS stepper from Geant4 v10.5 to v10.7.2

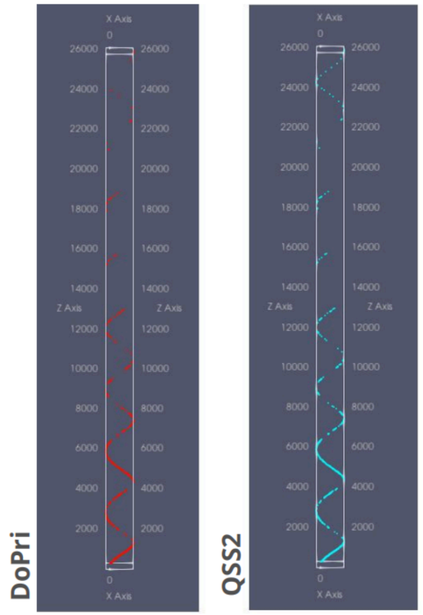
  to be added in G4 release

- Results using the N02 model qualitatively indistinguishable compared to those using the G4DormandPrince745
- Testing using FullSimLight

  ATLAS geometry & magnetic field map

- Performance profiling ongoing





G4HepEM Library Integration

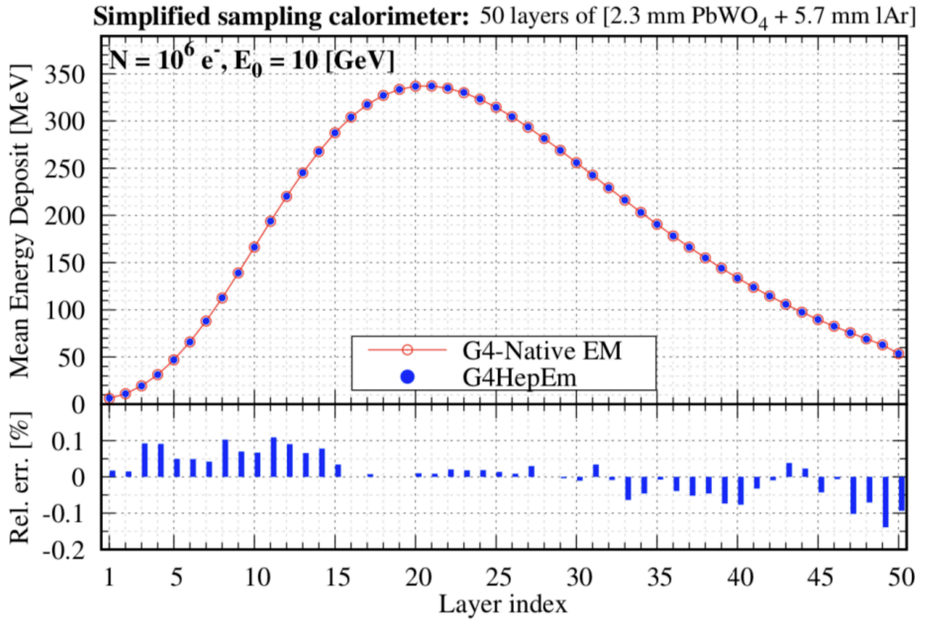
G4HepEM library is a new compact Geant4 EM library

Jonas Hahnfeld, Benjamin Morgan, Mihaly Novak

Optimized to be used for HEP electromagnetic showers development and transport

- more compact and GPU-friendly
- provides significant speed up with *Specialized Tracking*

Ongoing work of integration and benchmark, first in FullSimLight and then in Athena



max 0.1% change in simplified calorimeter observables

CMS detector configuration simulating ttbar events

	Physics List	Specialised Tracking	difference
G4NativeEm	2889 s	2747 s	-4.9 %
G4HepEm	2847 s	2660 s	-6.6 %
difference	-1.5 %	-3.2 %	-7.9 %

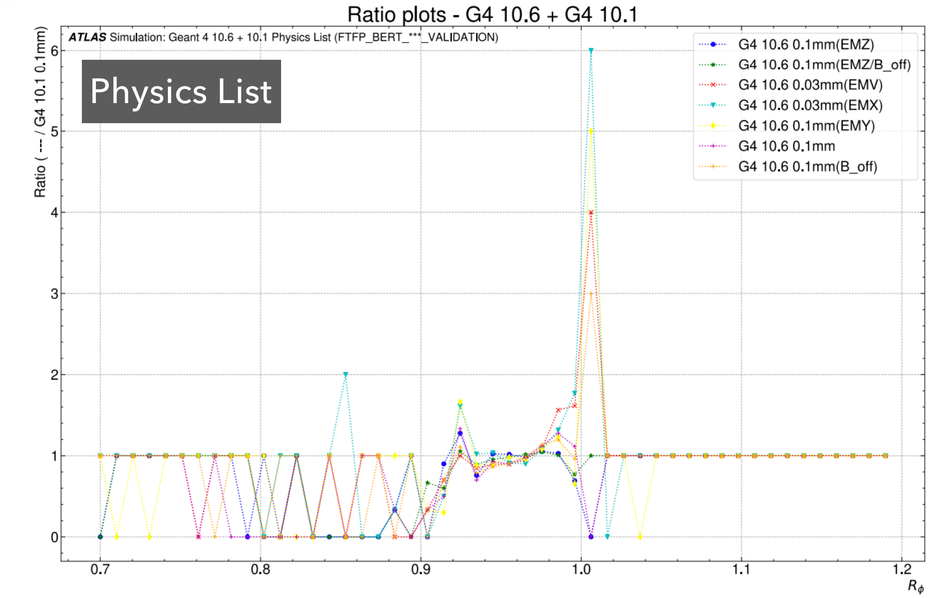
Note: significant performance gain due to the specialised tracking of e^-/e^+ and γ even already using GEANT4 native processes that is boosted further with G4HepEm (even in its current, preliminary phase)

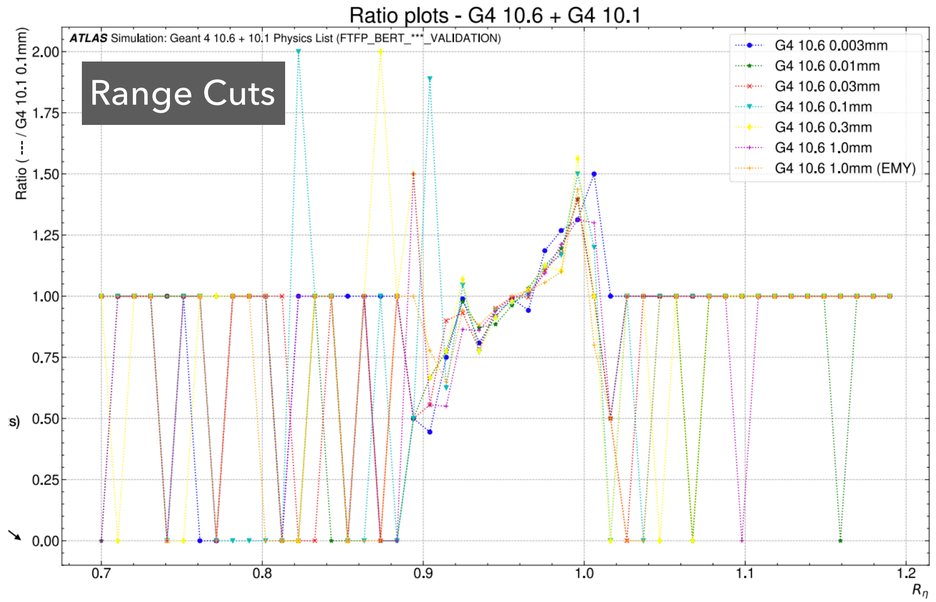
More info & data here

EM Physics Tuning

G4 simulation can run with different intrinsic tuning parameters

- **Goal:** find the best compromise between simulation accuracy and speed in order to improve data-mc agreement
- **Parameters to be optimized:**
  1. Physics List
  2. Range Cuts
  3. MSC Range Factors
- G4 simulation is compared to measurements available from the ATLAS combined performance groups, especially egamma and jet/etmiss
- Studies ongoing

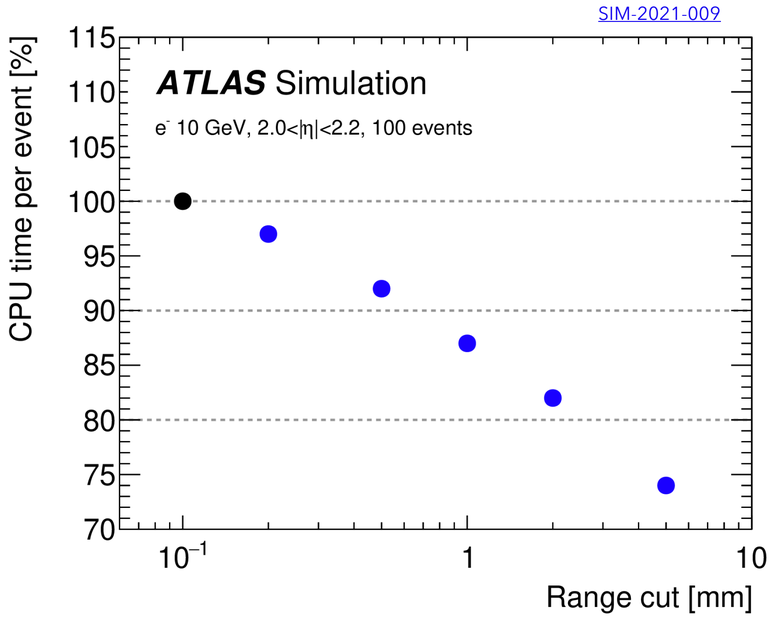




ML Correction for Aggressive Range Cuts

Increased range cuts can **reduce the number of photons**, thus reduce the transportation steps and **increase computational performance**

EM calorimeters dominate the simulation load due to low-energy photons from electron scattering, ~90% of these are transportation processes



Side-effect:
"High" range cuts can degrade the accuracy of the simulation

The ML correction applied as a post-processing step utilizing **batch processing** and **accelerator hardware** achieving **~15% speed up** in example geometries – ML inference time negligible compared to simulation time reduction.

Solution to be implemented/tuned for the ATLAS EMEC

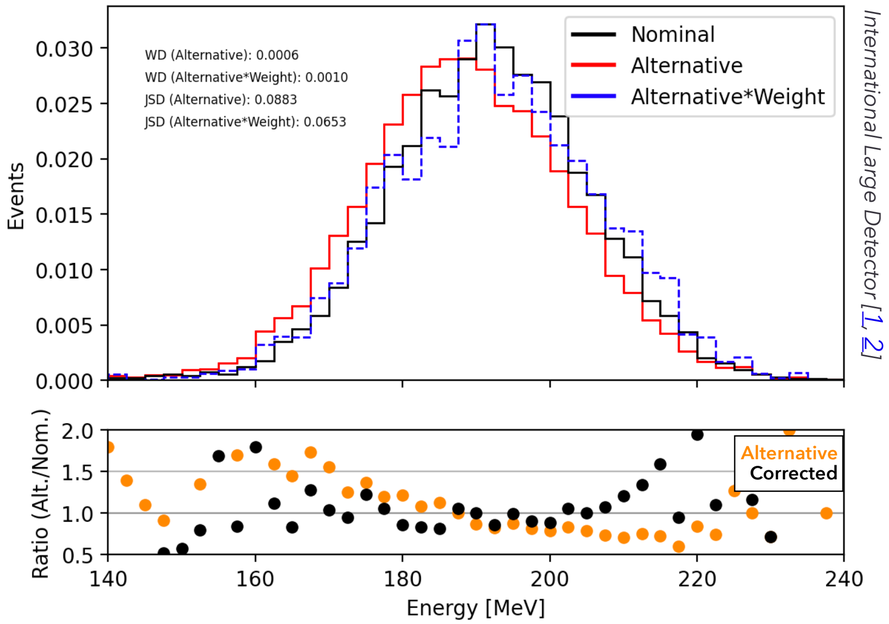
ML-based correction

Classification NN to learn correction weights [ref]

*Re-weight the **alternative** simulation to the **nominal** one by learning multi-dimensional weights considering all cell energy deposits*

$$r(\vec{x}) = \frac{p(\vec{x}\,|\,\theta_p)}{q(\vec{x}\,|\,\theta_q)}$$

θ be the range cut, $\boldsymbol{x}$ the energy deposits

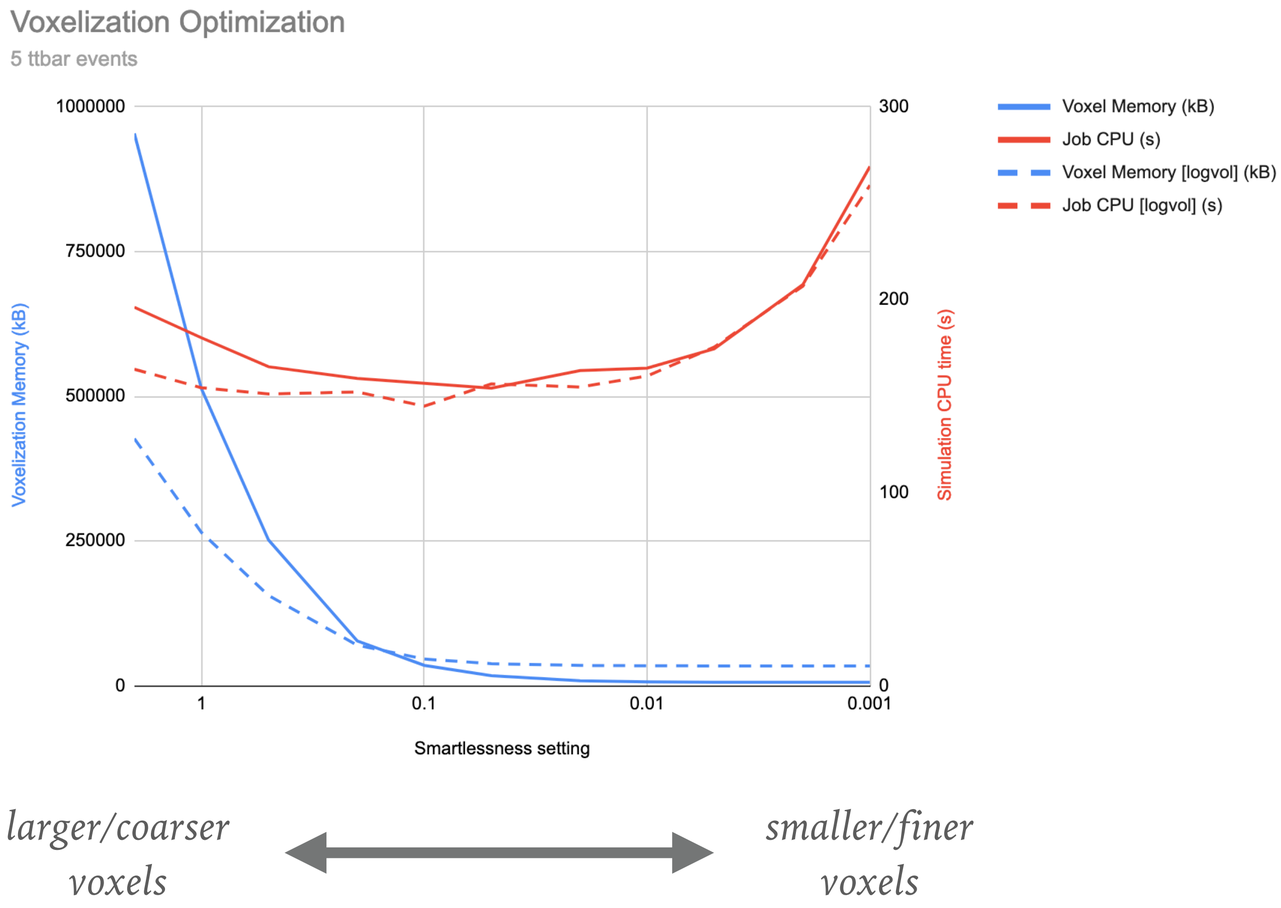


New Ideas

Voxel Density Tuning

Tracking can be optimized by voxelization, the size/granularity of the voxels can be tuned by the *Smartless* parameter

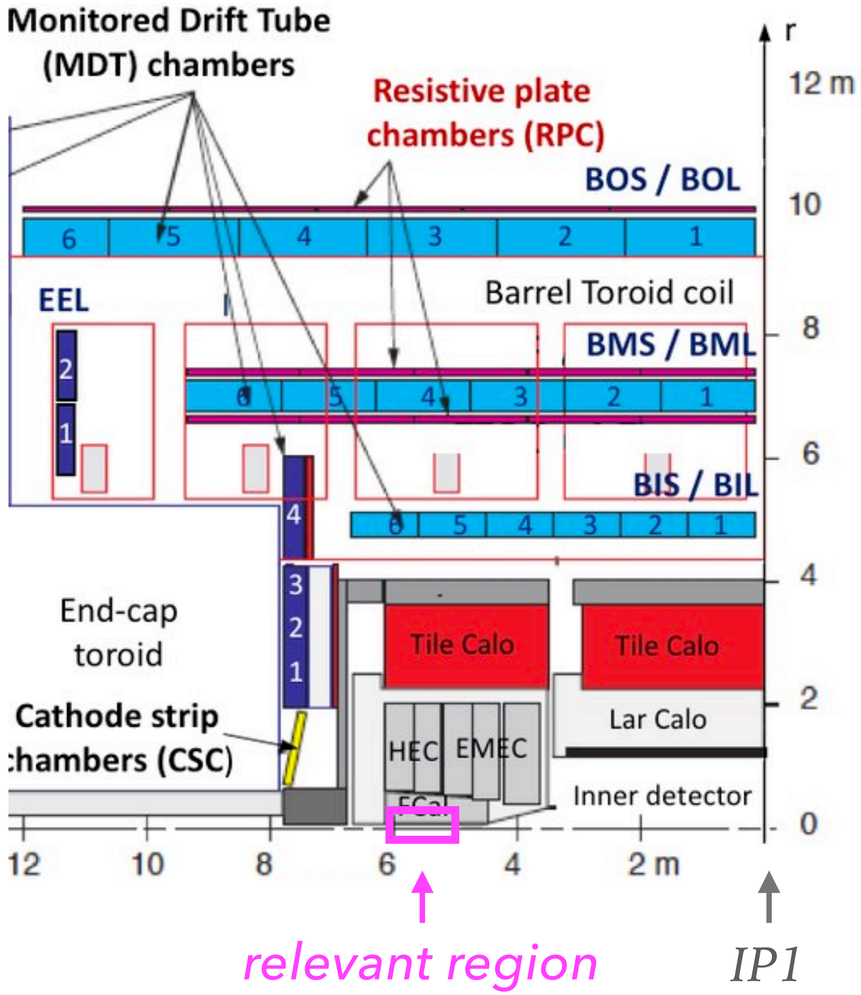
- **Goal:** Optimize the values of *Smartless* parameter for a balance between memory used for the detector description and CPU time for simulation
- Simulation accuracy should also be checked – although no effect is expected
- Initial studies targeting the Run4 ATLAS ITk sub-detector with many tracking elements
  Will also investigate for Run3 detector



New Particle Filter

Goal: Kill primary particles generating secondaries close to the beam-pipe at 5-6 m

- There is a huge amount of secondaries being created 5-6m away from (0,0,0), with small r – close to the beam pipe
- Many of these secondaries will never cause any energy in the calorimeters or a muon hit
  - The primary particles that caused these interactions could just be dropped directly
- Approach:
  1. generate a large sample of single particles with $4,5 < |\eta| < 6$ and different energies
  2. map out which η and E combinations can produce a relevant signal
  3. drop the rest directly with a new particle filter
  4. *Approach similar to Russian Roulette*
- We already kill all particles at $\eta>6$
  - Particles at $\eta>5$ and $p_T < 10$ GeV?
  - Or/and particles at $\eta>4$ and $p_T < 1$ GeV?
- Solution to be investigated



Conclusions

For ATLAS Run 2 samples Geant4 10.1 has been used

For Run 3 ATLAS considers two versions: 10.6 or 10.7

Able to achieve >27% CPU speed up with optimization so far

Translating to 38% higher throughput

1.38 times more events using the same computational resources

More optimizations upcoming

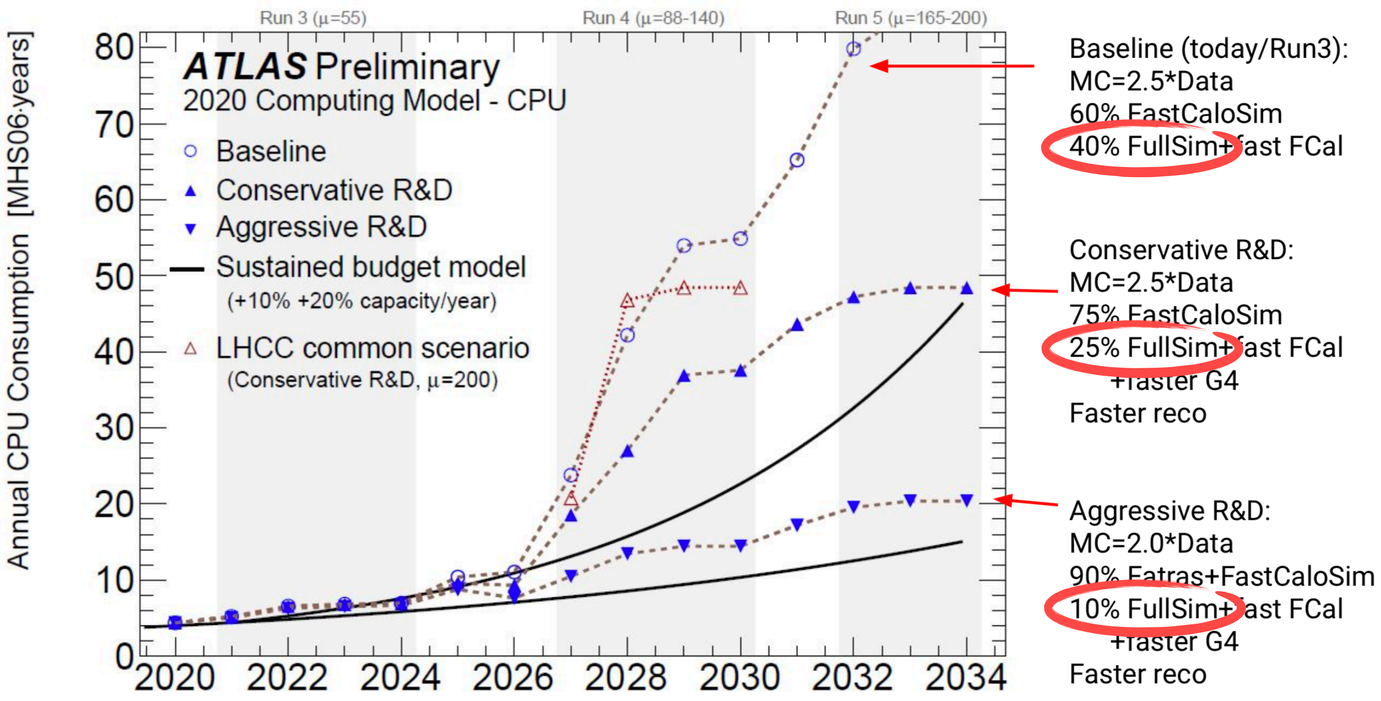
Either ongoing R&D or Future Ideas

We all thankful to Geant4 team for the excellent collaboration and support!

Backup

Detector Simulation Landscape

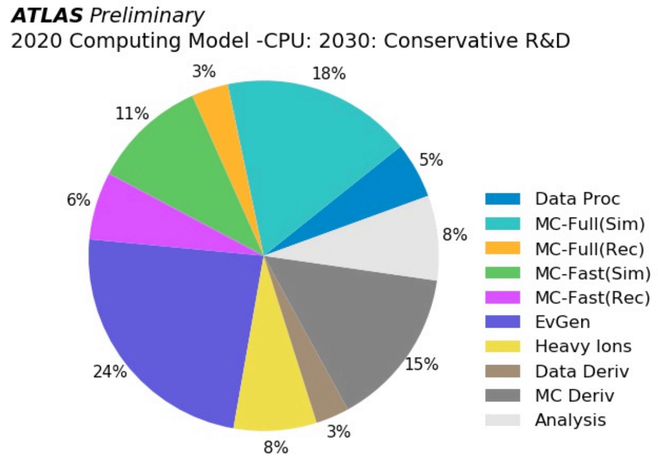
- LHC is about to start Run 3, ATLAS targets to FullSim ~25% of events
  during Run 2 ATLAS FullSim'ed ~40% (~21B) events
- **FullSim usage is unavoidable** (CP calibrations, FastSim training, etc.)
- In Run 4 and onwards FullSim requirements intensify as LHC plans to rise μ up to 200 and ATLAS to collect an order of magnitude more data
- Considerable CPU consumption O(10-20%) by Geant4

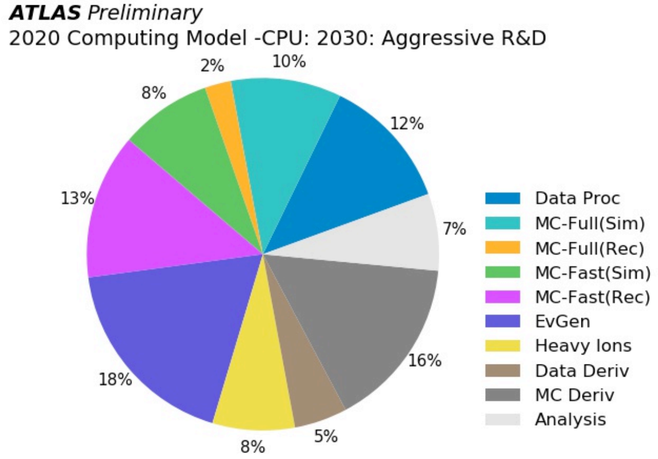


Baseline (today/Run3):
MC=2.5*Data
60% FastCaloSim
40% FullSim+fast FCal

Conservative R&D:
MC=2.5*Data
75% FastCaloSim
25% FullSim+fast FCal
+faster G4
Faster reco

Aggressive R&D:
MC=2.0*Data
90% Fatras+FastCaloSim
10% FullSim+fast FCal
+faster G4
Faster reco

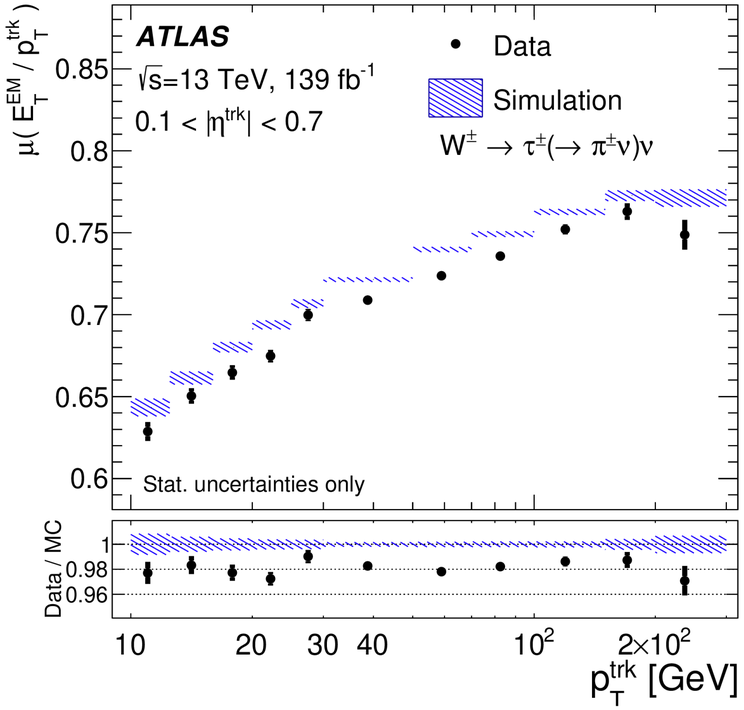


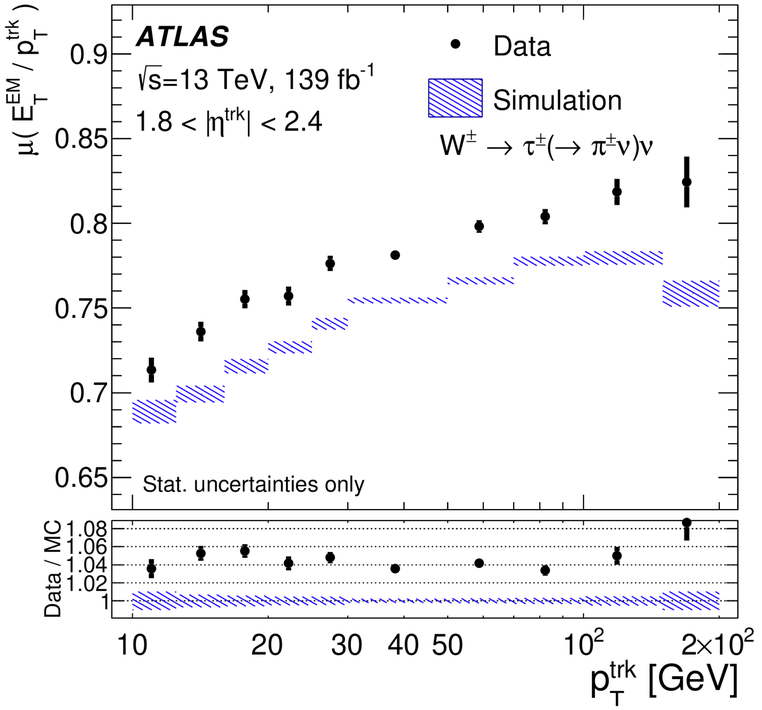


More info: ATLAS HL-LHC CDR

Data vs Geant4 10.1

- We see a 2% lower response in data in the barrel.
- We see a 4% higher response in data in the end-cap - this increases to 8% with a MIP selection.
- Electrons have 0.5-1% lower response in these two regions.
- We have heard reports in previous meetings on the test-beams from Tile and HEC.





Vectorized Sin/Cos Calculation in EMEC

- Parameterized sincos calculation in EMEC take a significant portion of the total Geant4 simulation time (2-3%),
- It could be re-written with explicit vectorization to speed it up,
- Successfully implemented multiple stand-alone versions of the sincos implementation, that appear to be consistently ~20% faster,
- Implemented in Athena, but difficult to test the exact speed-up because sincos is only a small portion of the total simulation time and because the vector implementation changes the outcome of the simulation due to fluctuations in the last digit of the double precision.

TRT Geometry Optimization

Results

- The different module representations were tested with the **FullSimLight benchmarking code**, run with a **single thread** on the CERN standalone machine;
- These studies target **only the TRT part** of the ATLAS detector geometry;
- Each test has been repeated 10 times. As **primary particles, 10^4 e– at 10 GeV** have been considered;
- The code has been compiled against **Geant4 v10.6.2 with GCC 8.2.0.**

Execution times for three types of solid representations. The current implementation with boolean solids is taken as reference.

Module shapes	Execution time (s)	Improvement
Boolean solids	1663	Reference
Arb8	1638	1.5%
BRep	1675	−0.7%

- A **positive improvement of 1.5%** is observed for the **Arb8 representation**, whereas the **BRep solid exhibits a minor degradation** with respect to the reference boolean solids.

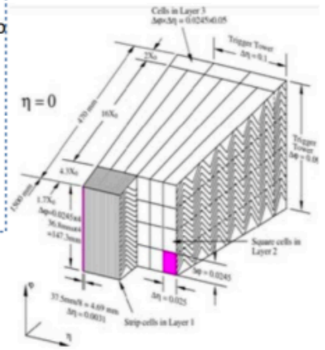
Physics and Performance Tuning

In order to see the photon shower shape distributions, plot the following observables:

- Reta, Rphi, weta2

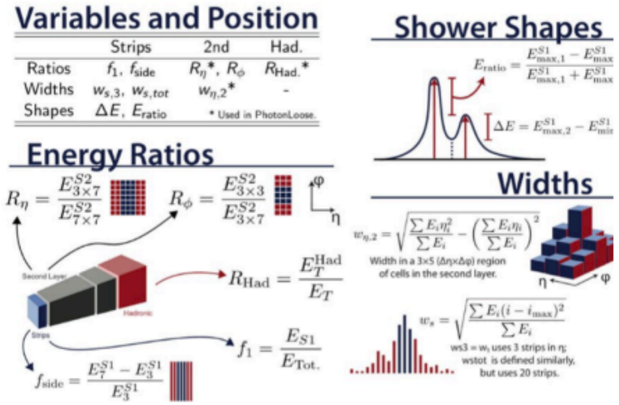
ATLAS and the EM calorimeter

- EM (EMB+EMEC)
  - Pre-sampler: Recover energy deposited before Calo
    - | η |<1.8, 0.025*0.1
  - Strips: Fine η granularity ensure good gamma/pi separation
    - | η |<3.2, typical 0.003*0.1
  - Middle: Most EM energies deposited
    - | η |<3.2, typical 0.025*0.025
  - Back: Recover e/g longitudinal energy leakage
    - |η| <2.5, 0.05*0.025
- HEC
  - Four layers: 1.5<|η| <2.5,0.1*0.1;2.5<|η| <3.2, 0.2*0.2
- FCAL
  - Three layers, 3.1 <|η| <4.9, Non projective



4

Particle identification

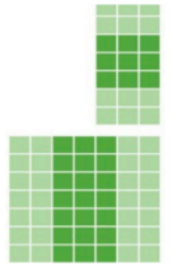


Lateral shower shapes (reminder)

- R_η, R_ϕ

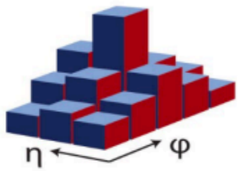
$$R_\phi = E_{3x3}/E_{3x7}$$

$$R_\eta = E_{3x7}/E_{7x7}$$



- $W_{\eta,i}$: shower "RMS" in layer *i*

$$W_{\eta 2} = \sqrt{\sum (E_i \eta_i^2) - (\sum (E_i \eta_i)/\sum (E_i))^2}$$



14

Source: Simulation of Electrons and Photons in ATLAS (November 2-3, 2020), a talk by Maarten Boonekamp

Geant4 Optimizations Benchmark

The throughput is the inverse of the CPU time (per event), so assuming that t1 is the initial cputime (per event) and t2 is the final one (after our optimizations) the 2 speed ups are:

$$\text{CPU time speed up} = [(t2-t1)/t1]*100$$

$$\text{Throughput speed up} = [(t1-t2)/t2]*100$$