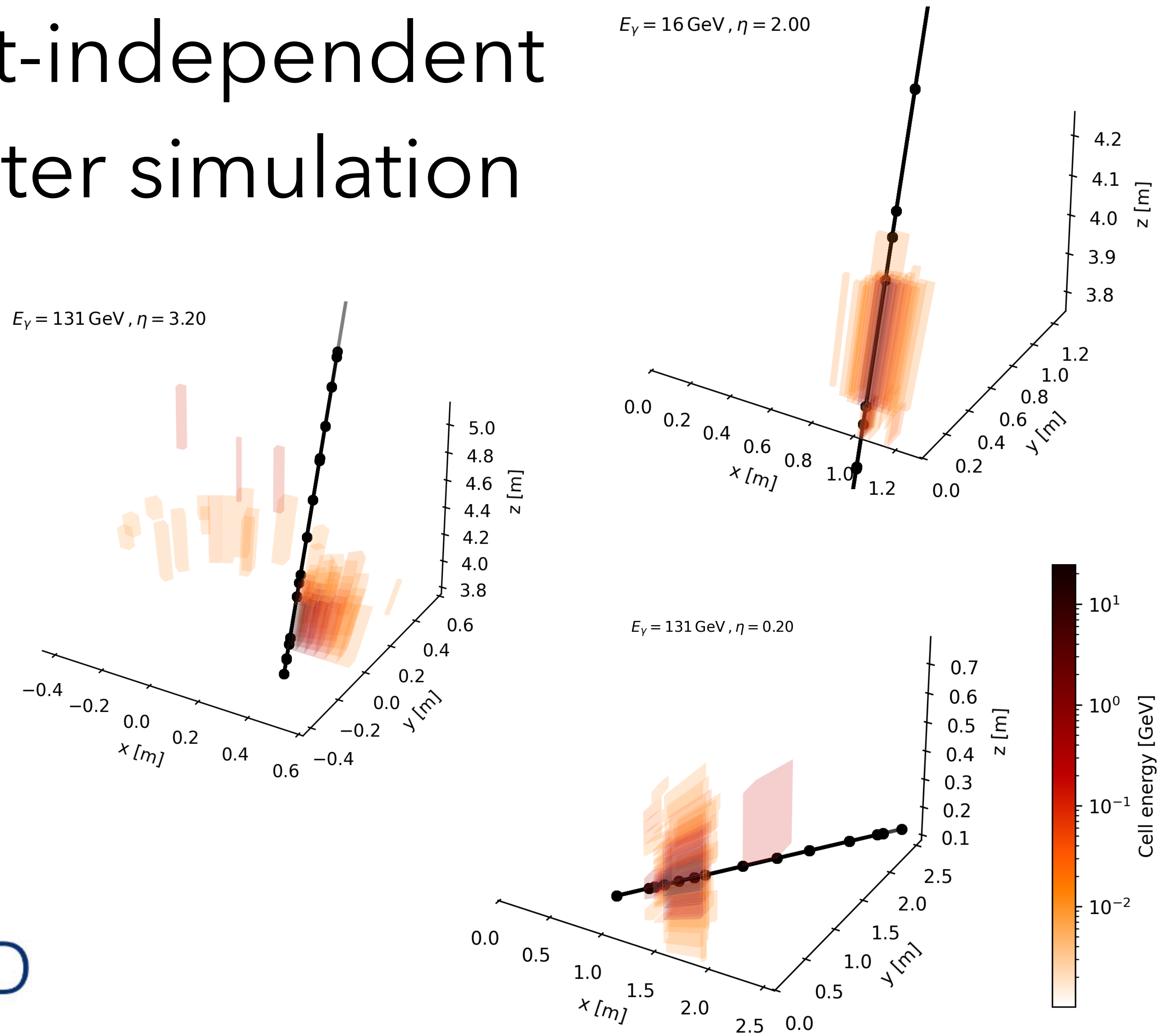
CHEP 2024
KRAKOW, POLAND

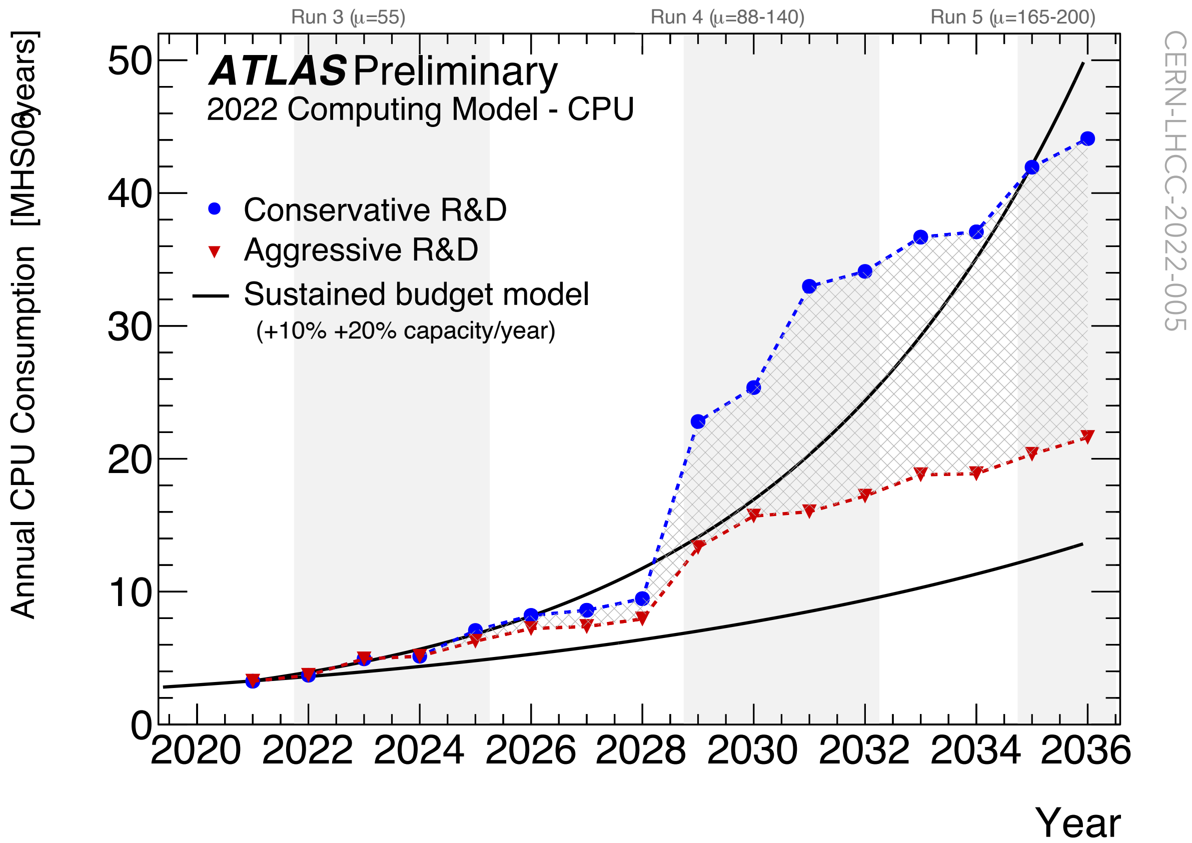
Towards an experiment-independent toolkit for fast calorimeter simulation

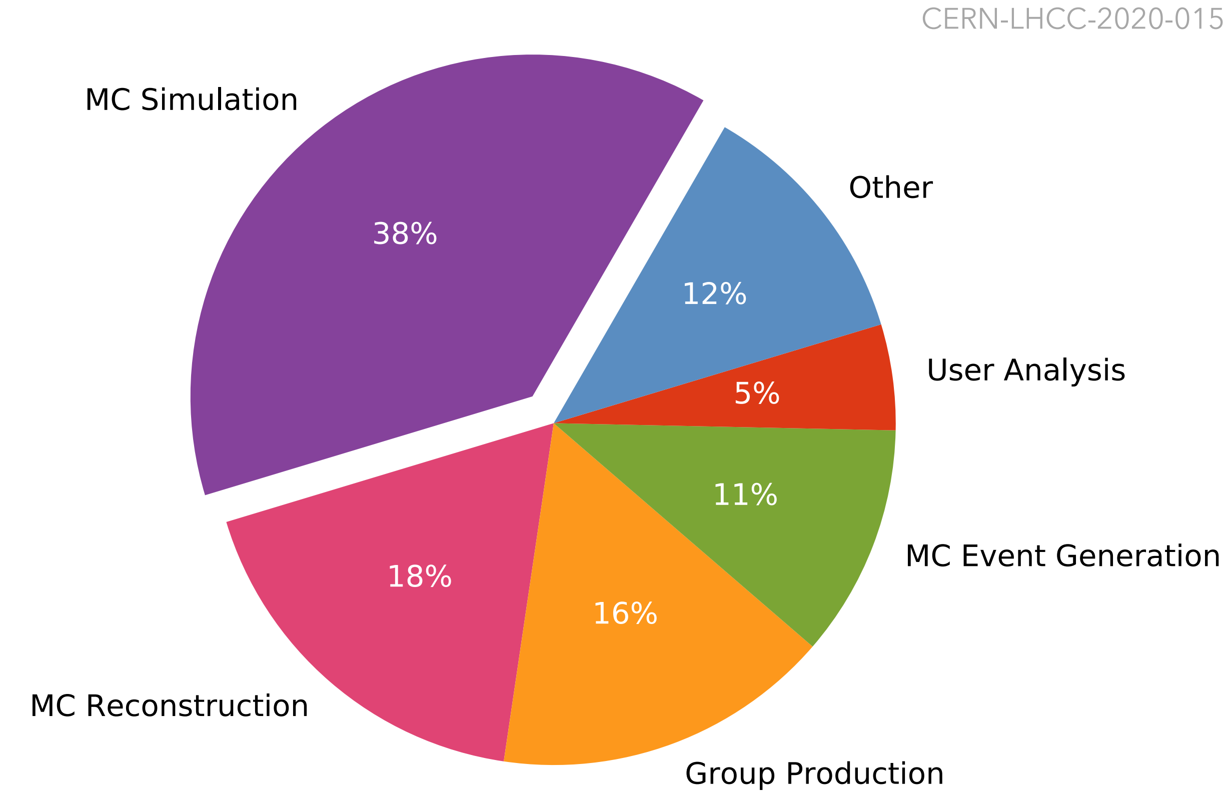
JOSHUA F. BEIRER (CERN)

ON BEHALF OF THE ATLAS COLLABORATION



Motivation: Fast Simulation





- ATLAS needs to produce billions of MC events, but is limited by CPU constraints
- MC simulation has largest single share of total CPU usage
- About 80 - 90 % of CPU time spent on simulation of showers in calorimeter system

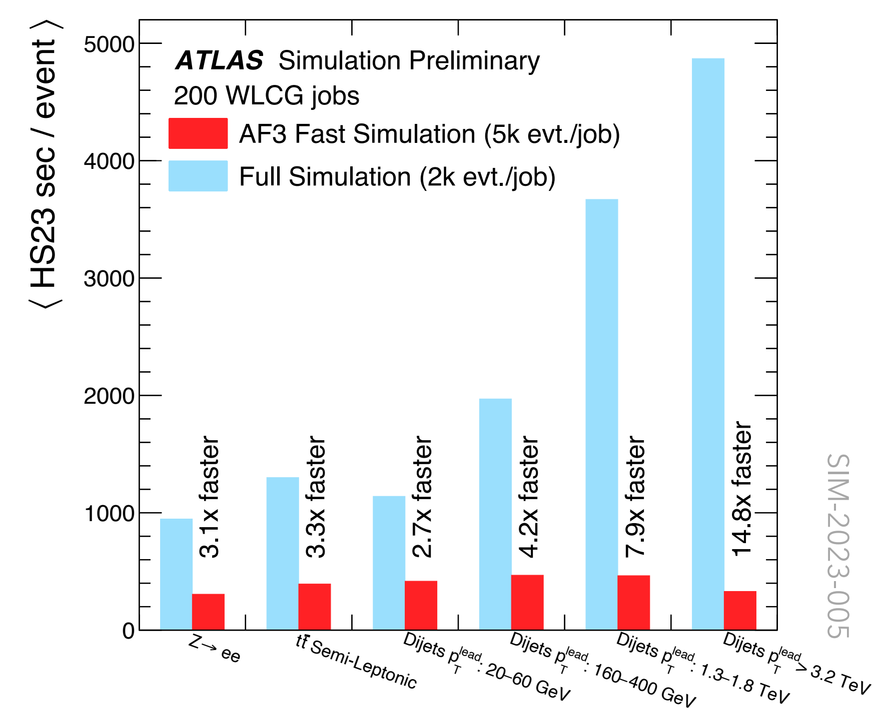
→ **use (fast) parametric models to reproduce Geant4 simulation as accurately as possible**

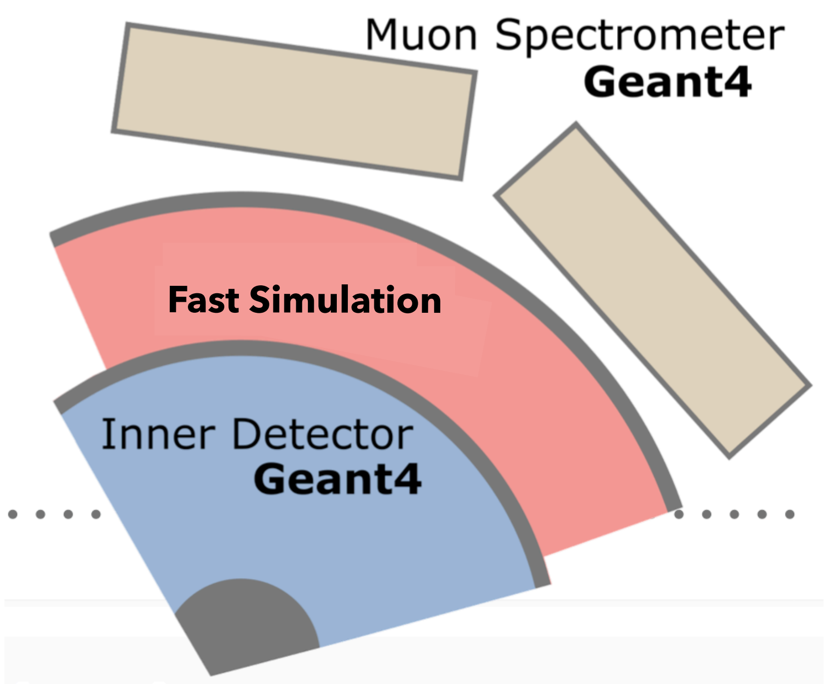
AtlFast3: State-of-the-Art Fast Simulation

Current State-of-the-Art fast simulation tool in ATLAS · **AtlFast3!**

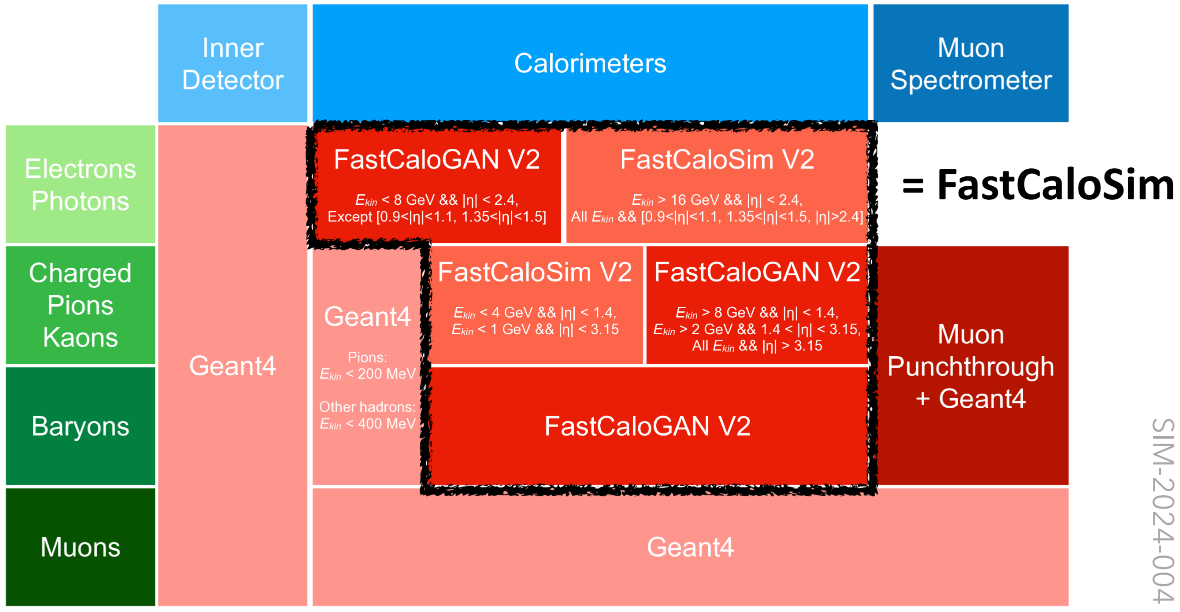
- **Basic Principle:** instead of tracking each particle in calorimeter showers, parametrise energy response with single particles
- Two distinct approaches of shower generation:
  - ***FastCaloSimV2:*** histogram-based parametrised modelling
  - ***FastCaloGAN:*** Generative Adversarial Network
- **3-15x** increase in simulation speed with respect to Geant4
- AF3 offers drastically improved physics performance with respect to Run 2 predecessor

3-15 x increase in simulation speed





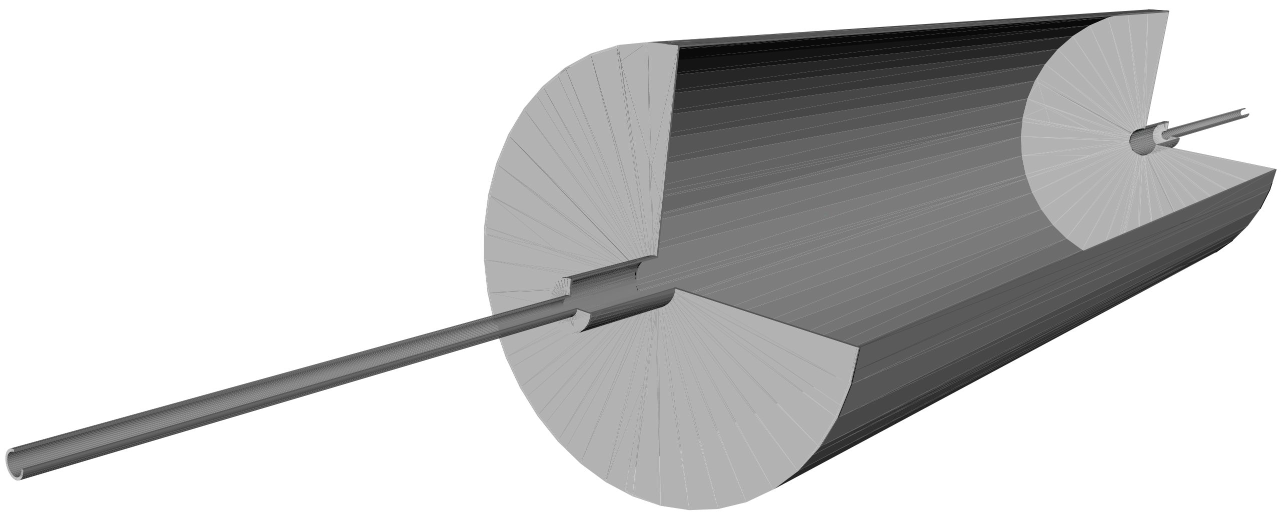
Run 3 Configuration of AtlFast3



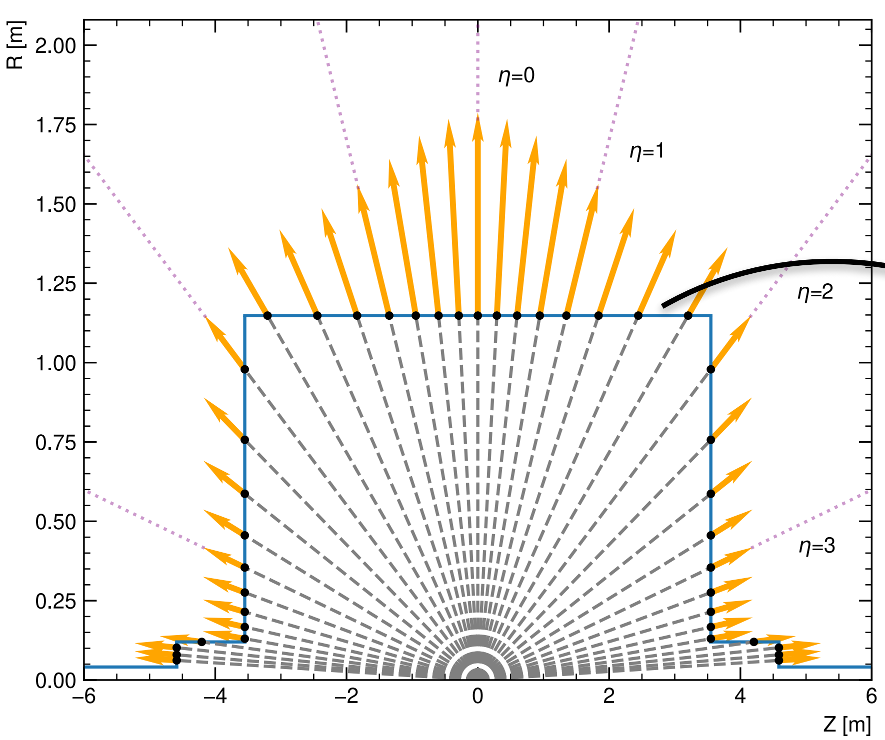
Simulation time fully dominated by Geant4 simulation in ID!

AtlFast3: Parametrisation

Boundary between Inner Detector and Calorimeter System



- Three separate parametrisations
  - **EM showers:**
    1. Photons γ
    2. Electrons / Positrons $e^{\pm}$
  - **Hadronic Showers:**
    3. Charged Pions $\pi^{\pm}$ (+ dedicated p GANs)



Particles generated at ID-Calo boundary

CERN-THESIS-2023-096

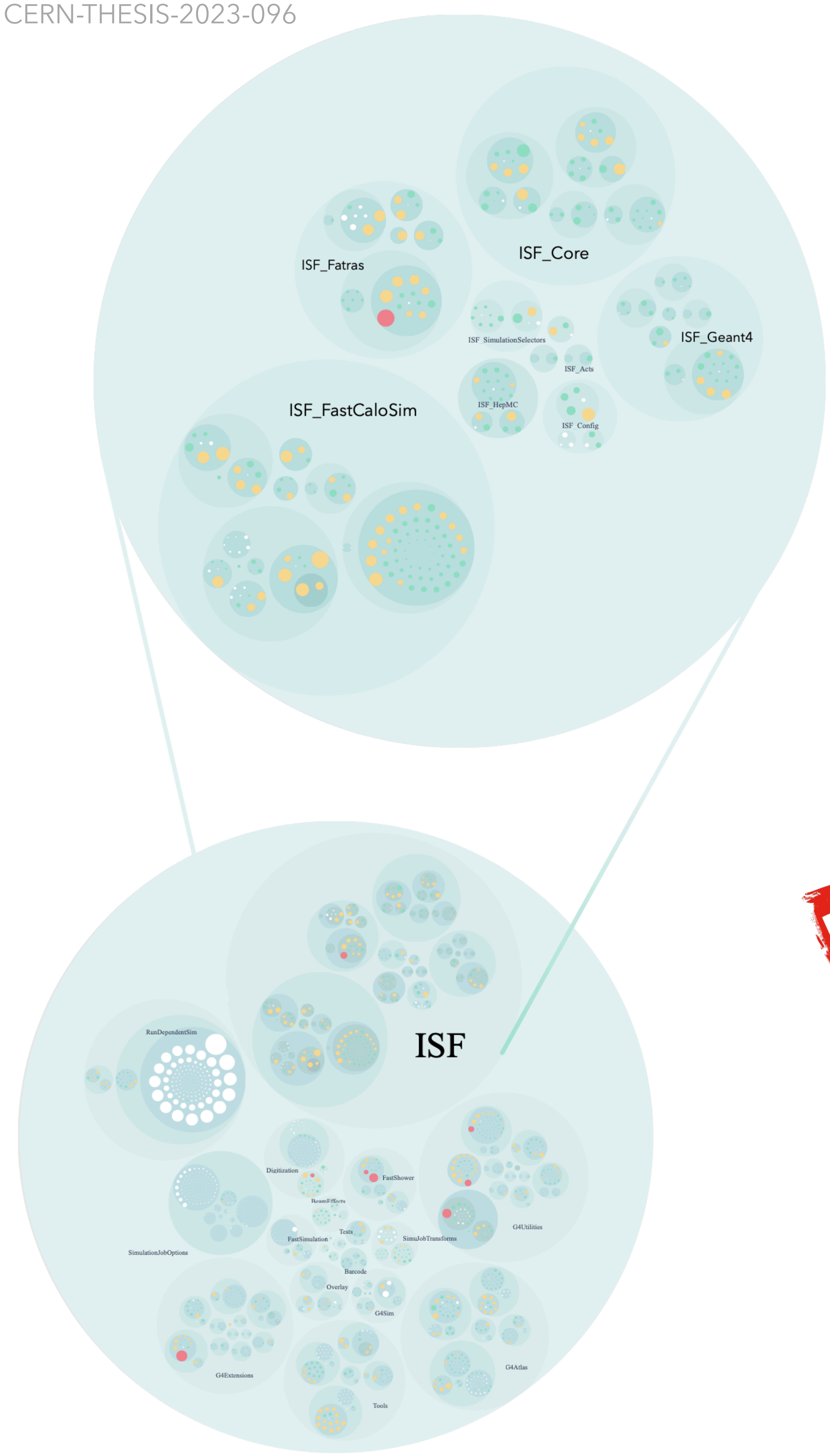
- Particles for parametrisation generated at the boundary between Inner Detector and Calorimeter System

- Parametrisation depending on incident particle energy and direction:
  - 17 log-bins of **truth momentum** from 64MeV to 4TeV
  - 100 bins of |η| from 0 to 5.0

More information on AtlFast3 in Federico's talk

Motivation: Current Fast Simulation Infrastructure

CERN-THESIS-2023-096



- **FastCaloSim** currently implemented in the Integrated Simulation Framework (ISF) in **athena** to combine with multiple simulators

- ISF is flexible, but very complex
  → originally designed for complex use-cases that are not needed in ATLAS
  → disproportionate growth in complexity over the years
  → increasingly hard to maintain for the collaboration

- Our goal is to severely simplify our simulation infrastructure and at the same time make FastCaloSim available to future experiments:

DONE

Implement FastCaloSim as Geant4 fast simulation model

Promote FastCaloSim to an experiment-independent library

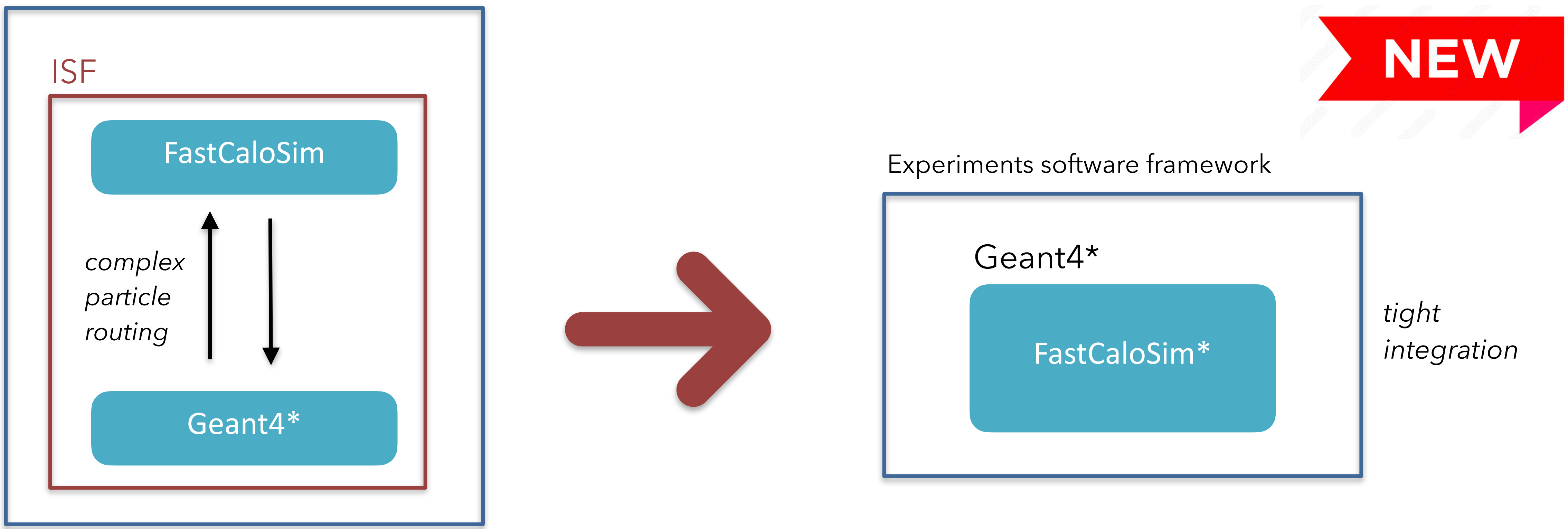
More on this in my ACAT 2024 talk

Goal: **simple** and **streamlined** ISF-independent ATLAS simulation!

Fast Simulation: Now and then

athena

* external library



experiment-dependent ISF implementation

experiment-independent Geant4 implementation

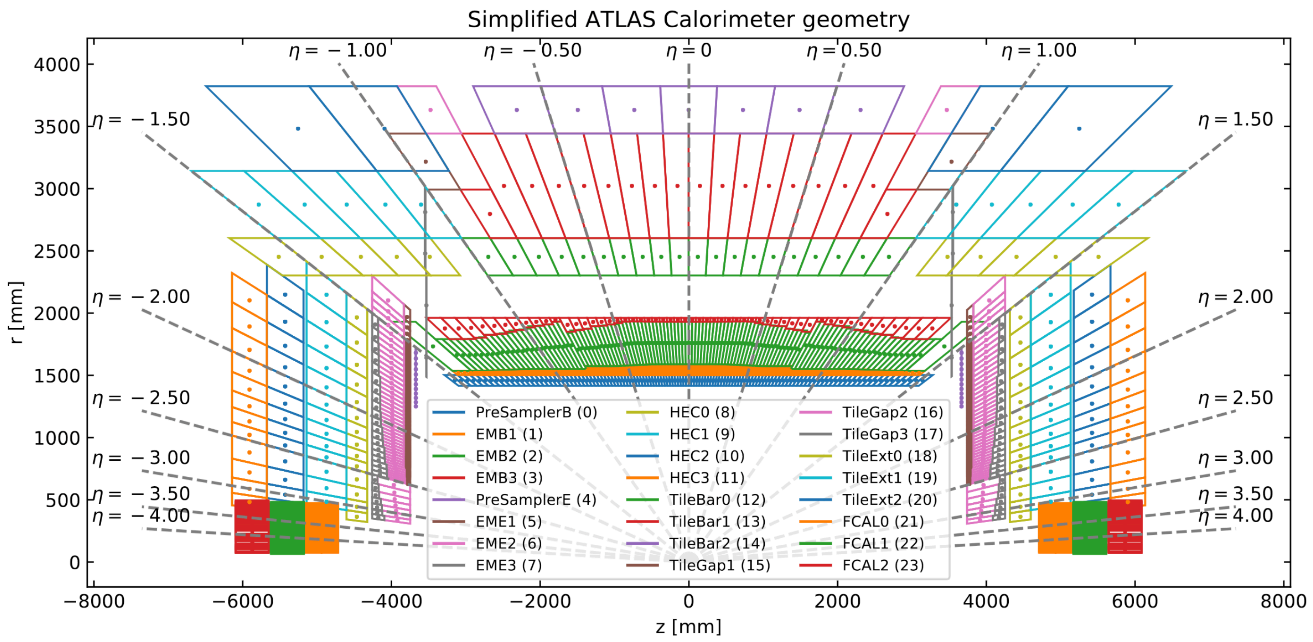
Advantages:

1. Fully eliminates the need for ISF, severely reducing codebase to maintain
2. Fully experiment-independent without athena or ISF dependencies
3. Allows fast local code development in containers
4. Allows to enforce modern best practices in controlled environment
5. Allows for contributions of other HEP experiments

What was required for the restructuring?

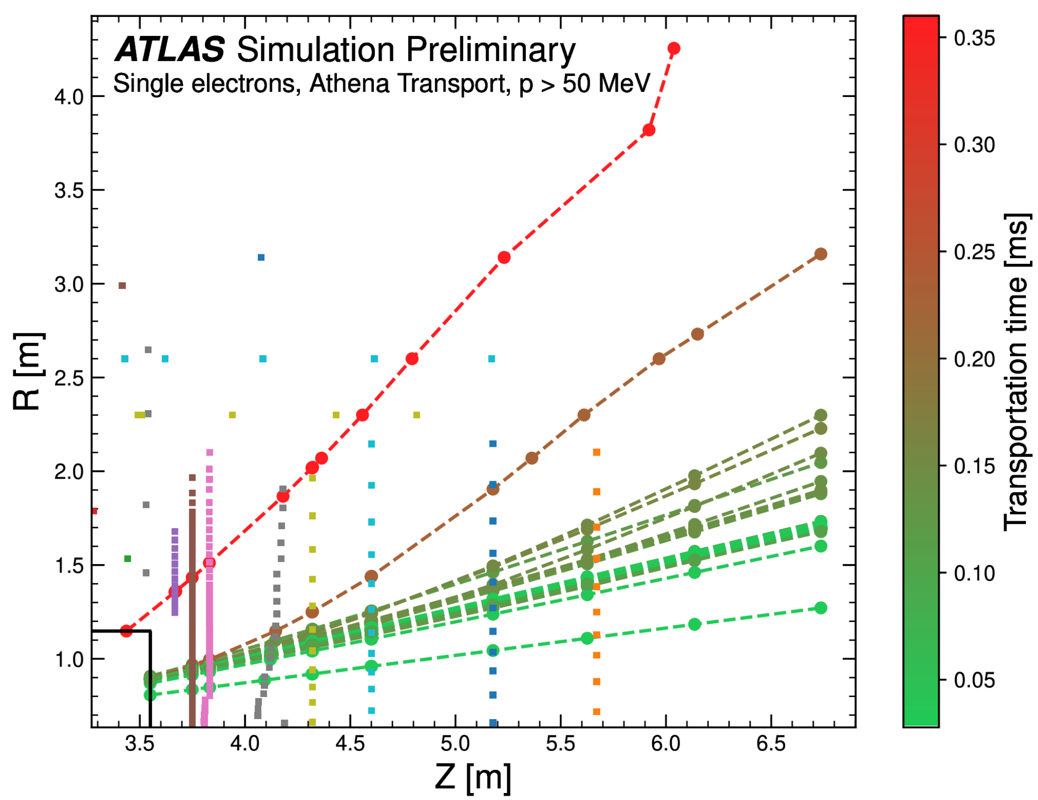
Step I: Replacement of ATLAS Track Transport

EPJ Web Conf., 245 (2020) 10003



- FastCaloSim relies heavily on an accurate determination of the shower centre position in each calorimeter layer
- To determine the positions, tracks need to be transported through the ATLAS calorimeter system, taking into account magnetic field

SIM-2024-004



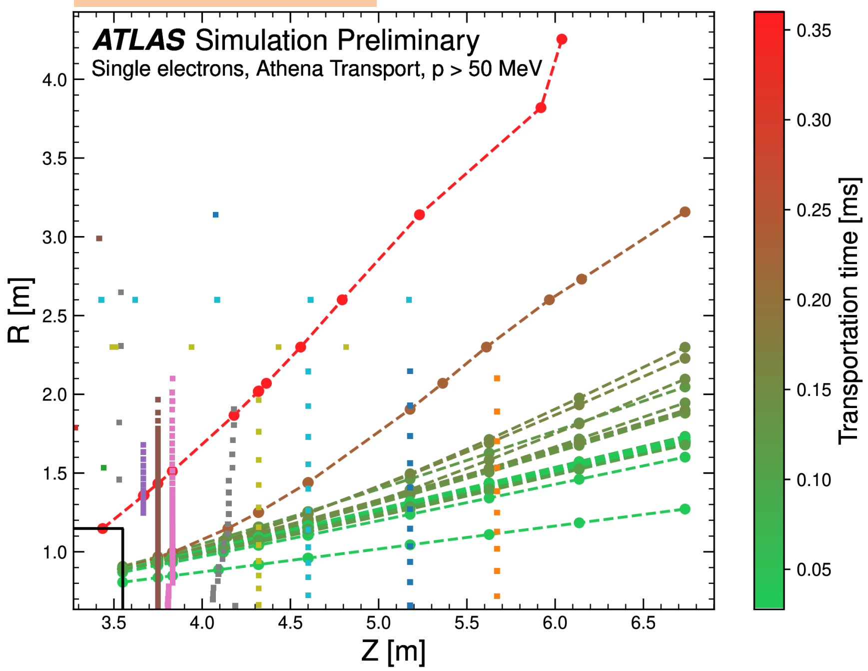
- FastCaloSim currently uses proprietary **Athena tracking tools** to transport particles
- Intersections with active calorimeter layers given as input to FastCaloSim

Athena transport needs to be replaced with experiment-independent Geant4 solutions

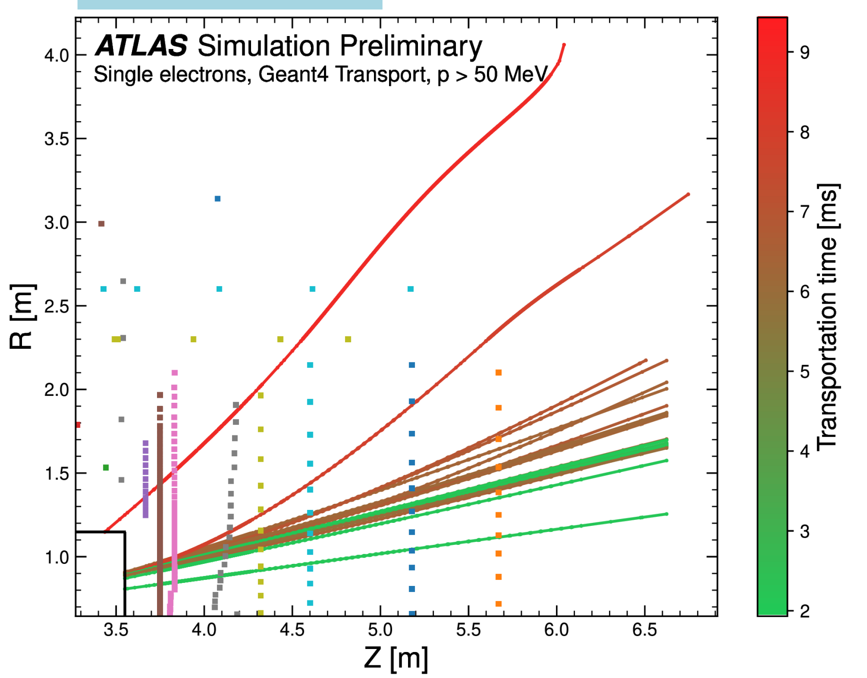
Step I: Replacement of ATLAS Track Transport

SIM-2024-004

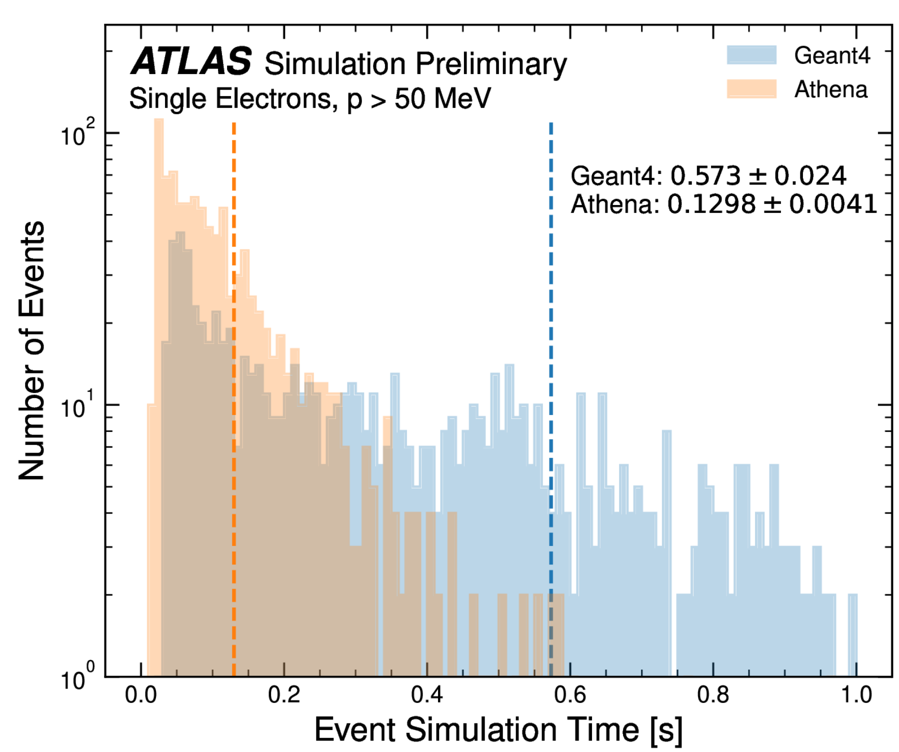
Athena Transport

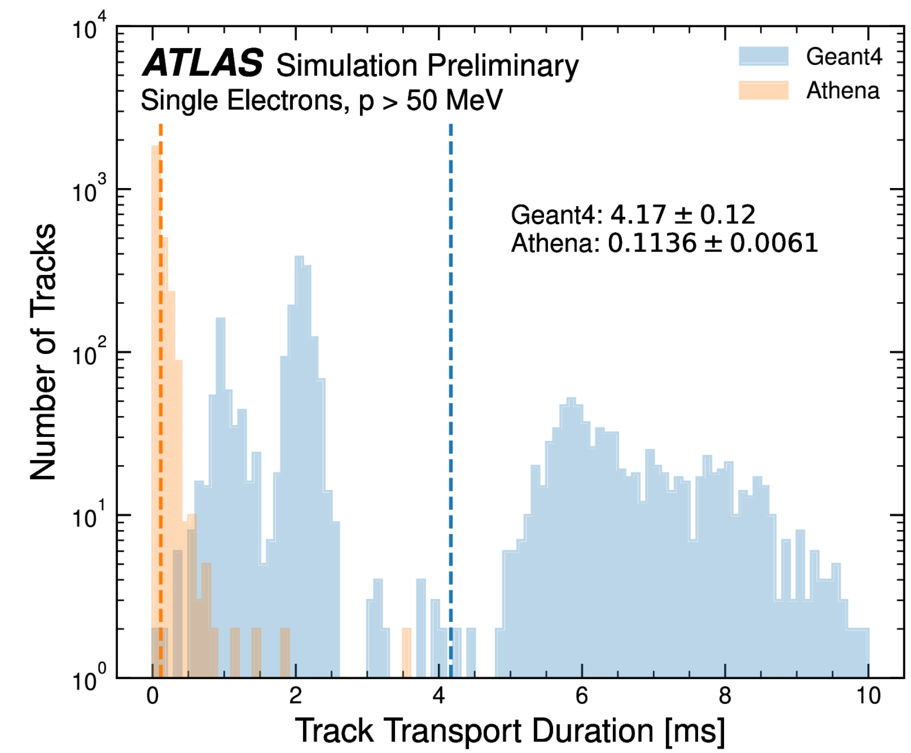


Geant4 Transport



First attempt: Exploit default Geant4 engine for track transportation





Single electron event simulation time:
x 3 slower with Geant4 transport

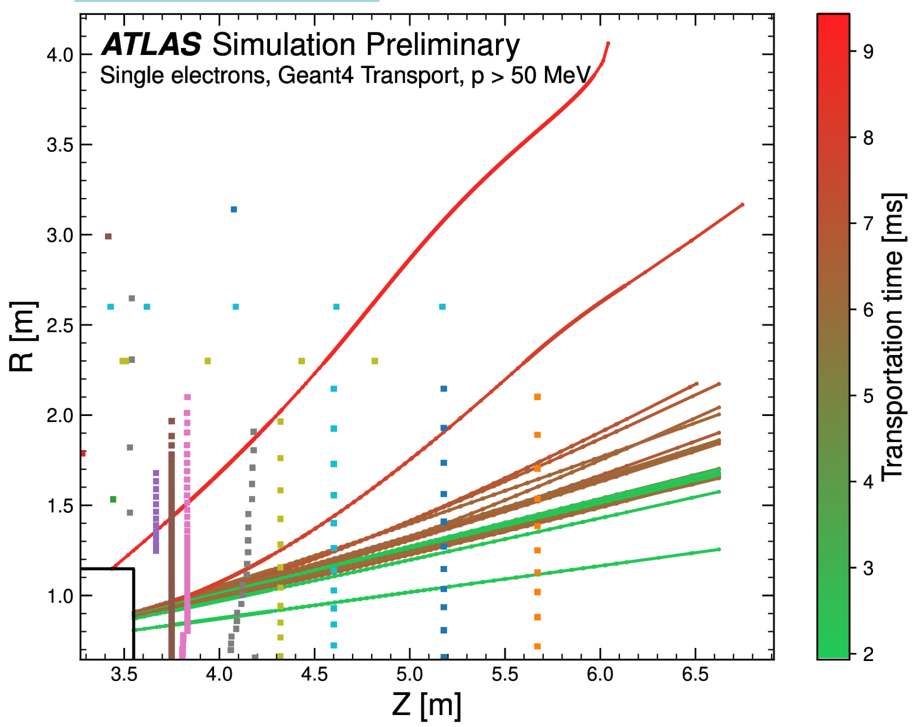
Transport duration per track:
x 34 slower with Geant4 transport

First very naive approach of Geant4 transport **prohibitively slow**

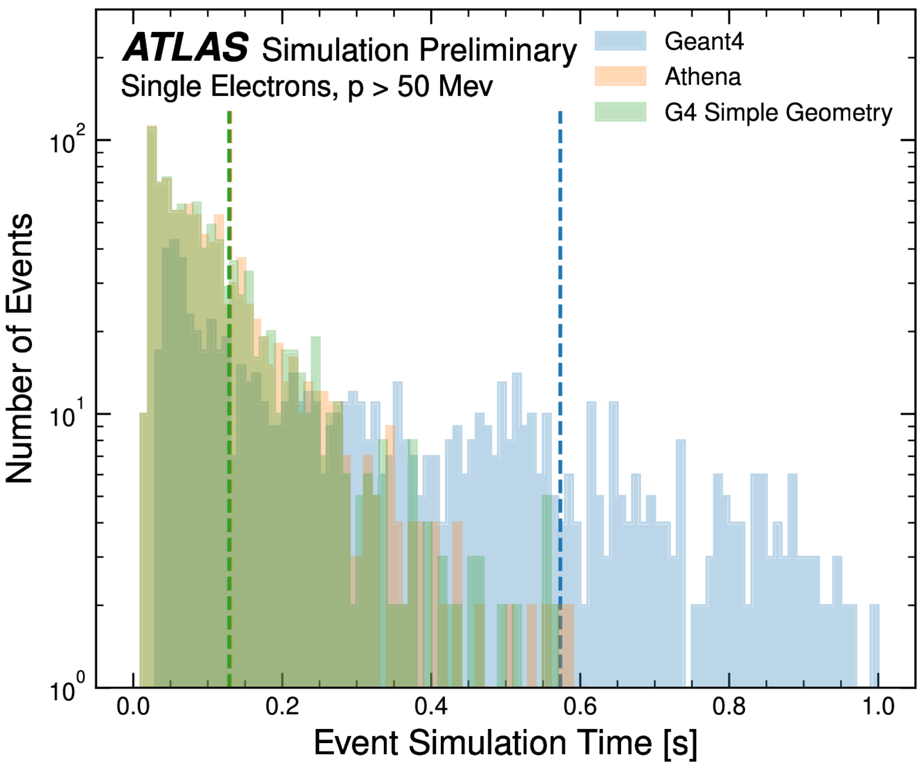
How can we do better?

Step I: Replacement of ATLAS Track Transport

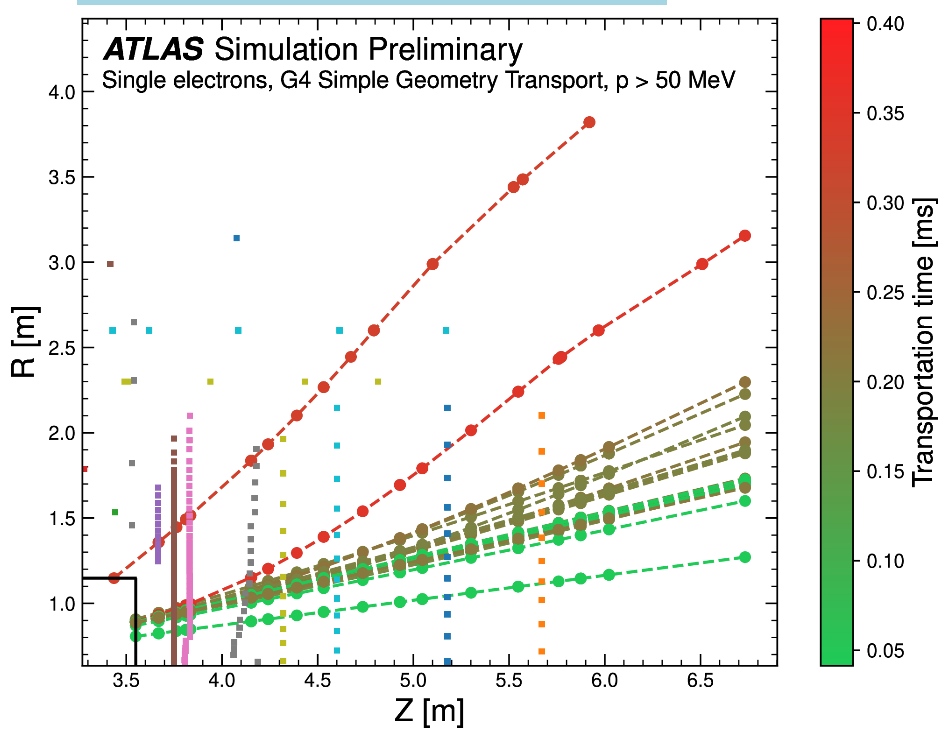
Geant4 Transport



SIM-2024-004



Geant4 Transport in simplified geometry

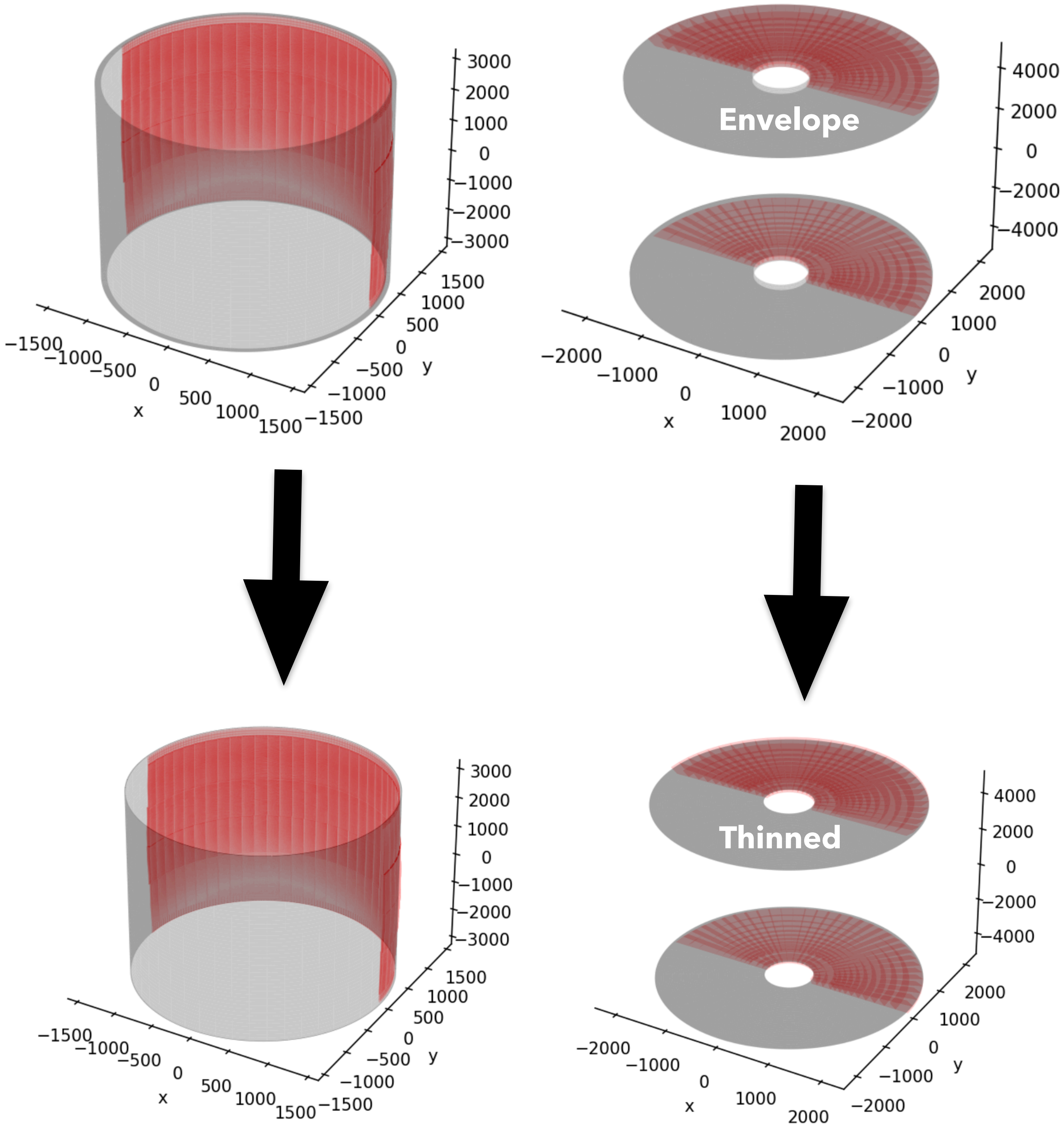


- Interested only in track position at entry and exit of each calorimeter layer
- Instead of transporting through full calorimeter geometry, do navigation in simplified (layer-based) geometry

Event simulation time of Athena tracking tools recovered with Geant4 navigation in simplified geometry

How to construct simplified geometry?

Simplified Geometry: Construction



Goal: *Experiment-independent* package to automatically build simplified geometry based on detector cells where:

- Layers are modelled as cylinders
- Cylinder surfaces approximately correspond to real entry and exit of detector layers
- Clash-free

Approach (simplified):

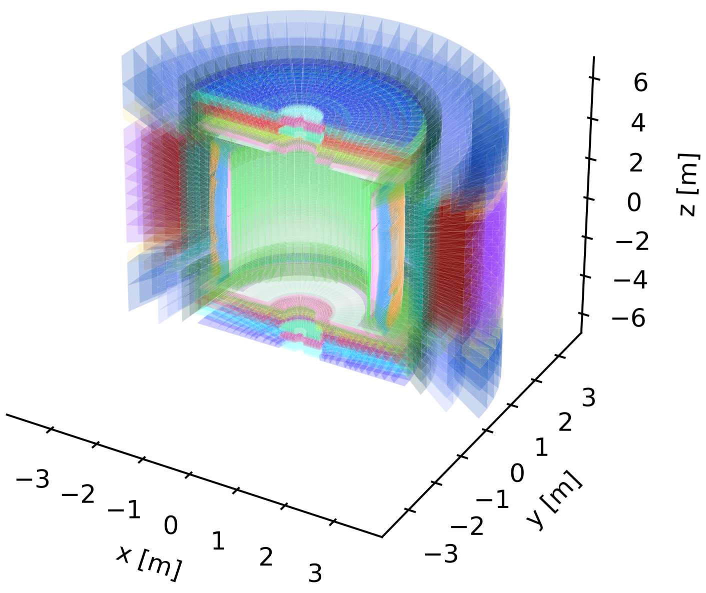
1. Model all layers as cylindrical hulls (='envelopes') from the maximum geometric extension of the cells

2. *Thin down* hulls to generate clash-free geometry
→ for barrel layers thinned $r = r^{\mathrm{hull}}_{\mathrm{mid}}$
→ for endcap layers thinned $z = z^{\mathrm{hull}}_{\mathrm{mid}}$

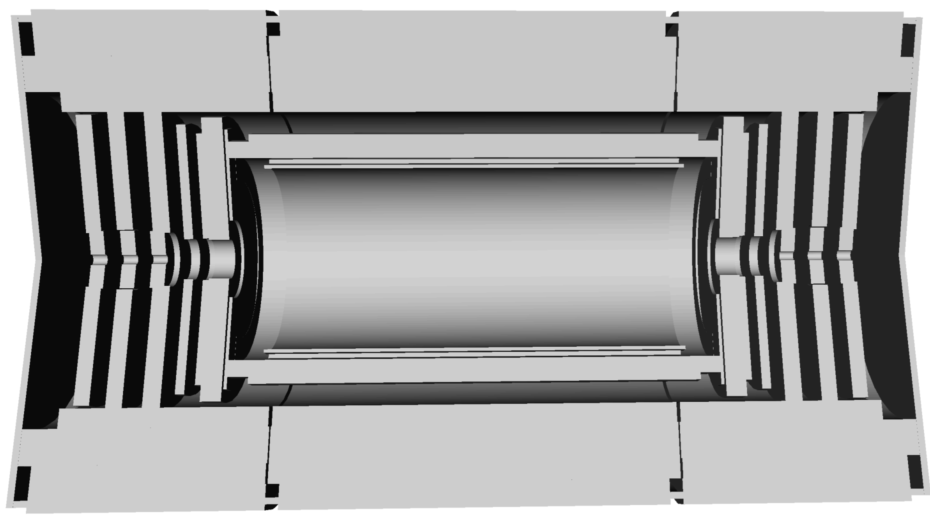
3. Attempt to grow back thinned down layers to original size of envelopes (constraint: only grow up to the limiting layer, i.e. w/o creating overlaps)

Simplified Geometry: Construction

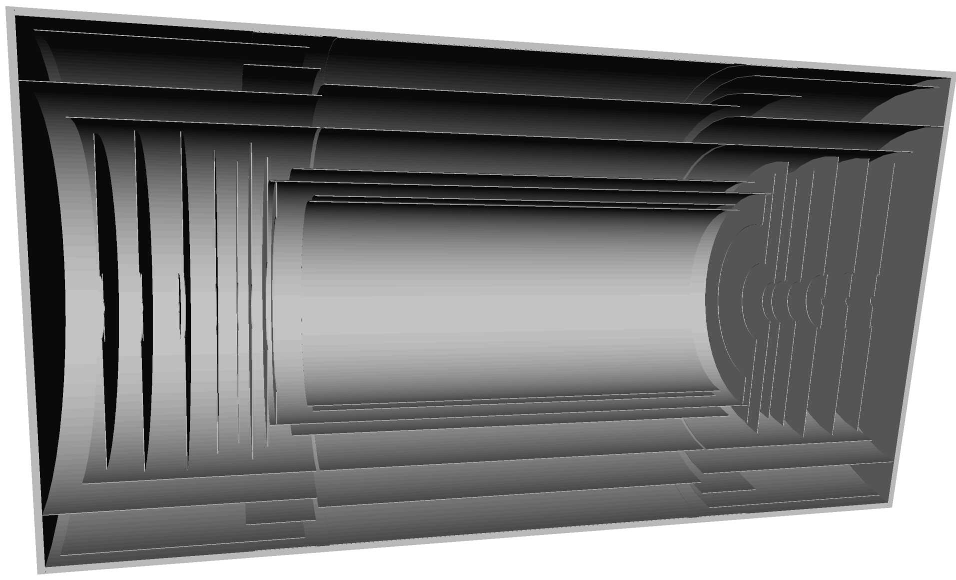
1) Calorimeter Cells



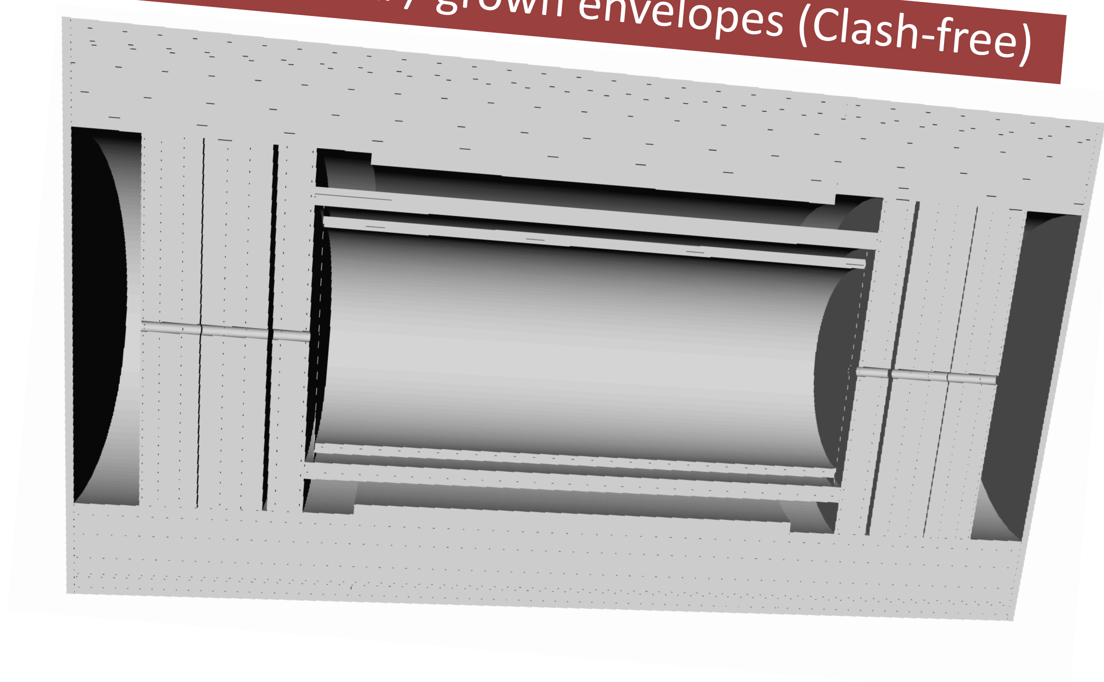
2) Cell Envelopes (**NOT** clash-free)



3) Thinned envelopes (Clash-free)



4) Processed / grown envelopes (Clash-free)

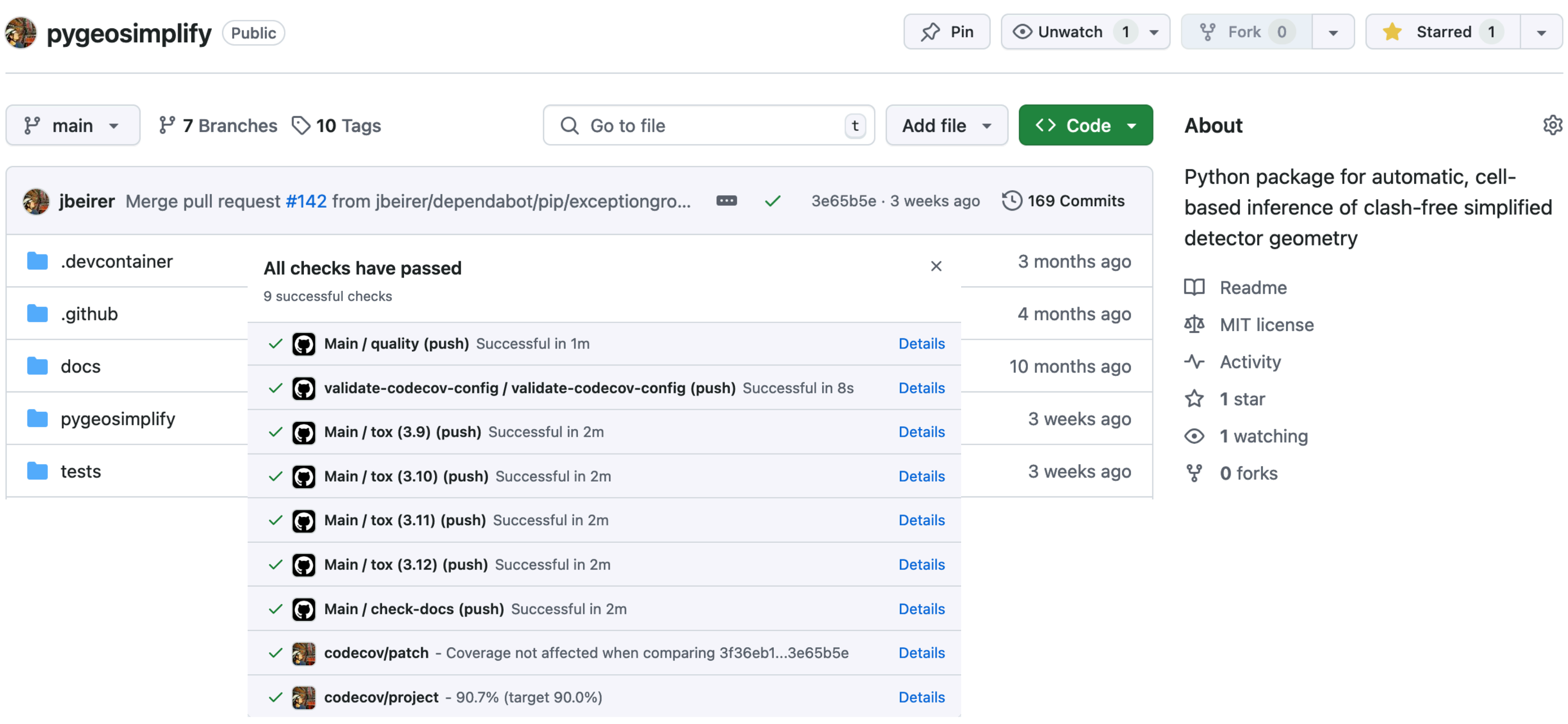


```
**** Real time elapsed   : 0.00852796
**** User time elapsed   : 0.01
**** System time elapsed : 0

   Size of G4SolidStore            : 50
   Size of G4LogicalVolumeStore    : 50
   Size of G4PhysicalVolumeStore   : 50

   Number of volumes : 50 (2 levels)
   Number of volumes checked : 50
   Number of clashes detected: 0
```

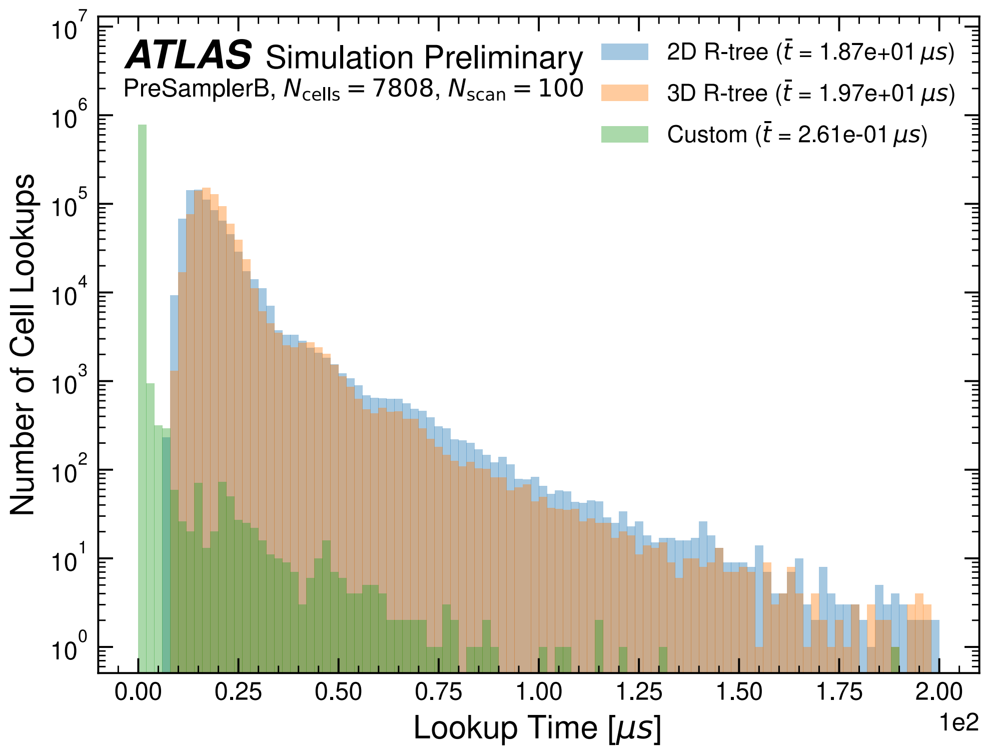
Python package: pyGeoSimplify



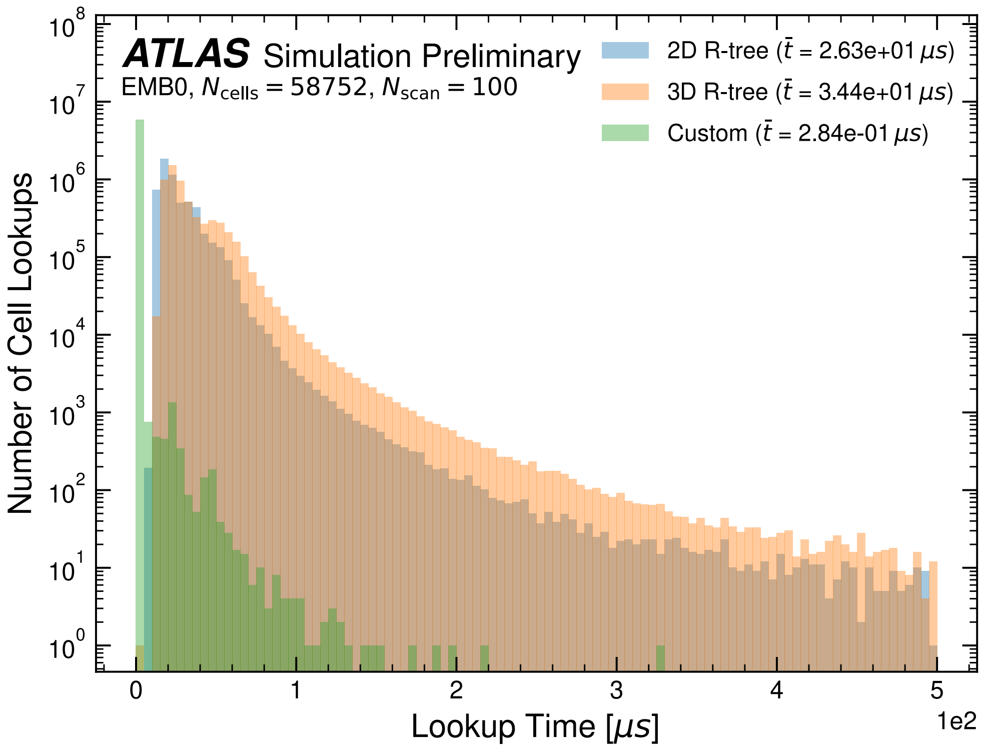
- All functionality integrated in python package names pyGeoSimplify
- **Input:** ROOT file with cell positions and dimensions
- **Output:** GDML file of clash-free simplified detector
- Extensive testing with over 90% test coverage
- pyGeoSimplify available on PyPi: `pip install pygeosimplify`

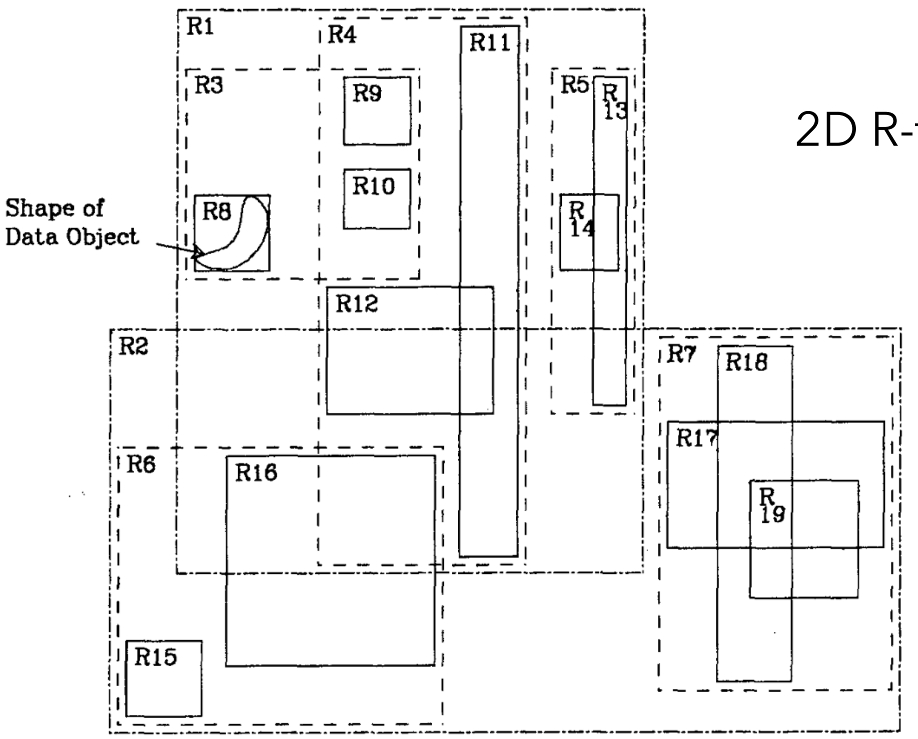
Step II: Experiment-independent cell matching

SIM-2024-009



SIM-2024-009





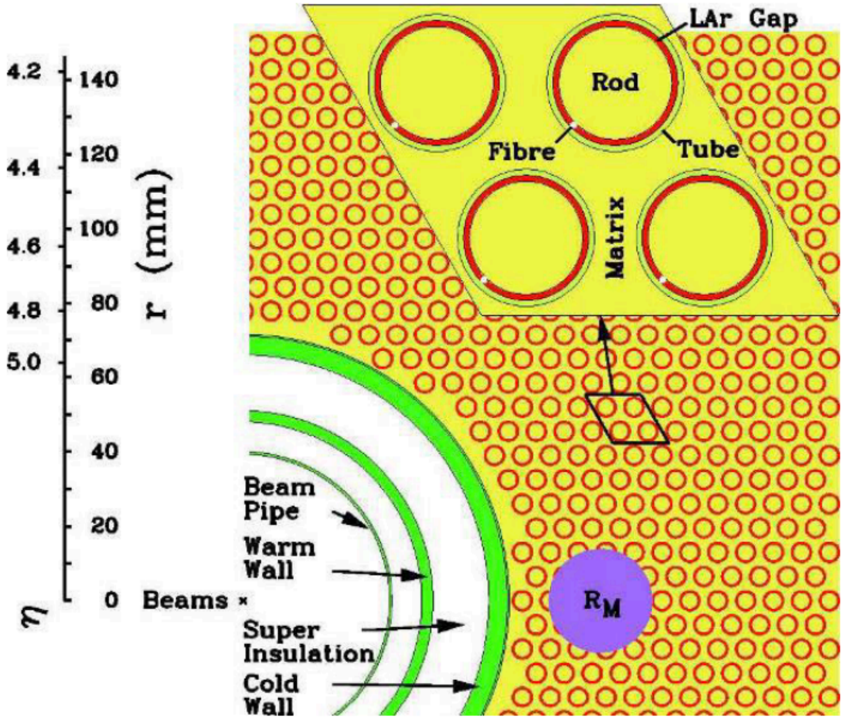
2D R-tree example

10.1145/971697.60226

- During simulation, need to match sampled hits to best-matching cells
- Cells used to be found by storing complex, experiment-dependent geometry maps
- Re-implemented hit-to-cell matching by organizing cell information into R-trees for **fast experiment-independent lookup**, allowing to remove thousands of lines of code
- Experiment-independent lookup up to 100x slower compared to maps, but not expected to become a bottleneck

Step II: Experiment-independent cell matching

ATLAS (irregular) FCAL geometry



10.1109/NSSMIC.2006.355989

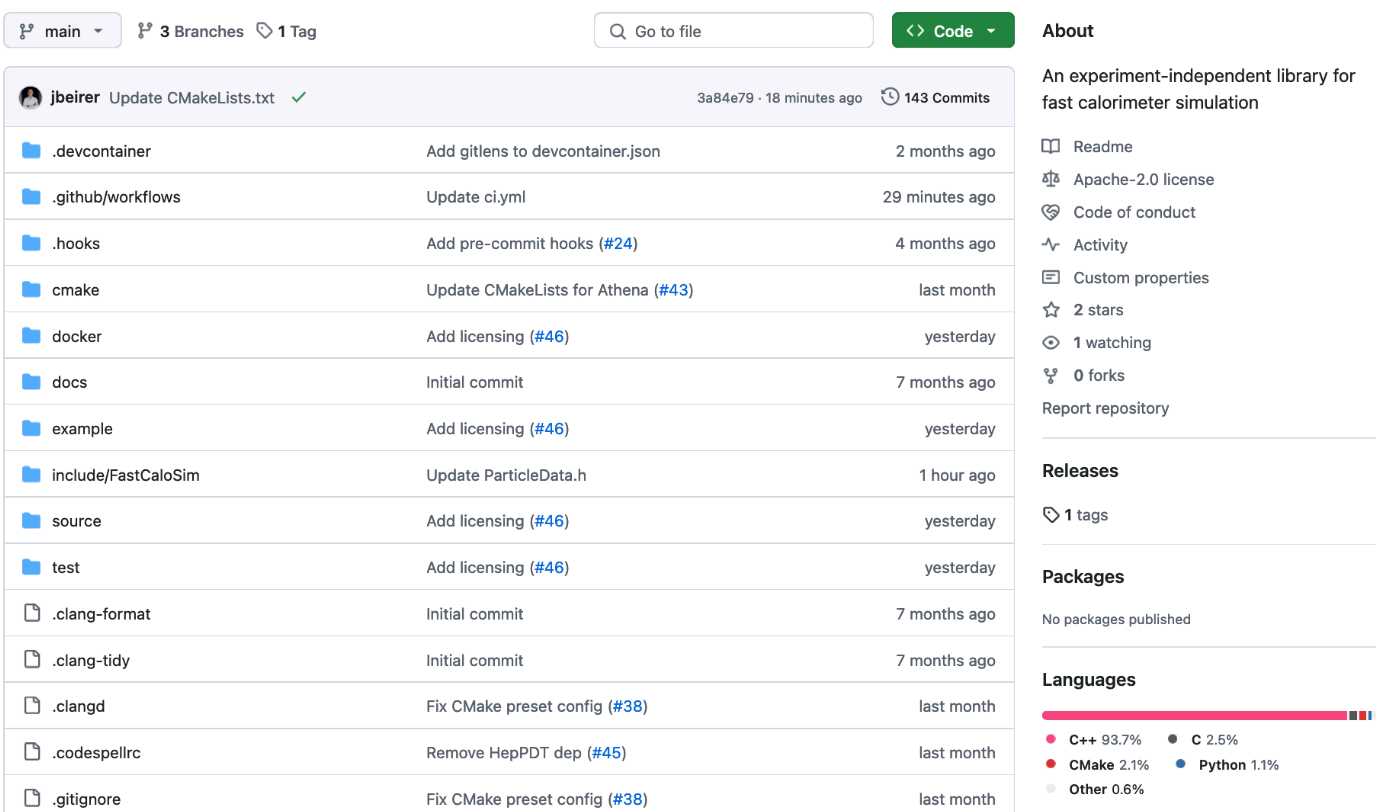
- The hit-to-cell matching assumes detector cells can approximately be described as cuboids in some coordinate system
- If this approximation is not good enough (or experiments need a faster cell lookup), custom geometry handlers can be easily implemented
- For ATLAS, the cells of the forward calorimeter are highly irregular and cuboid description not sufficient:

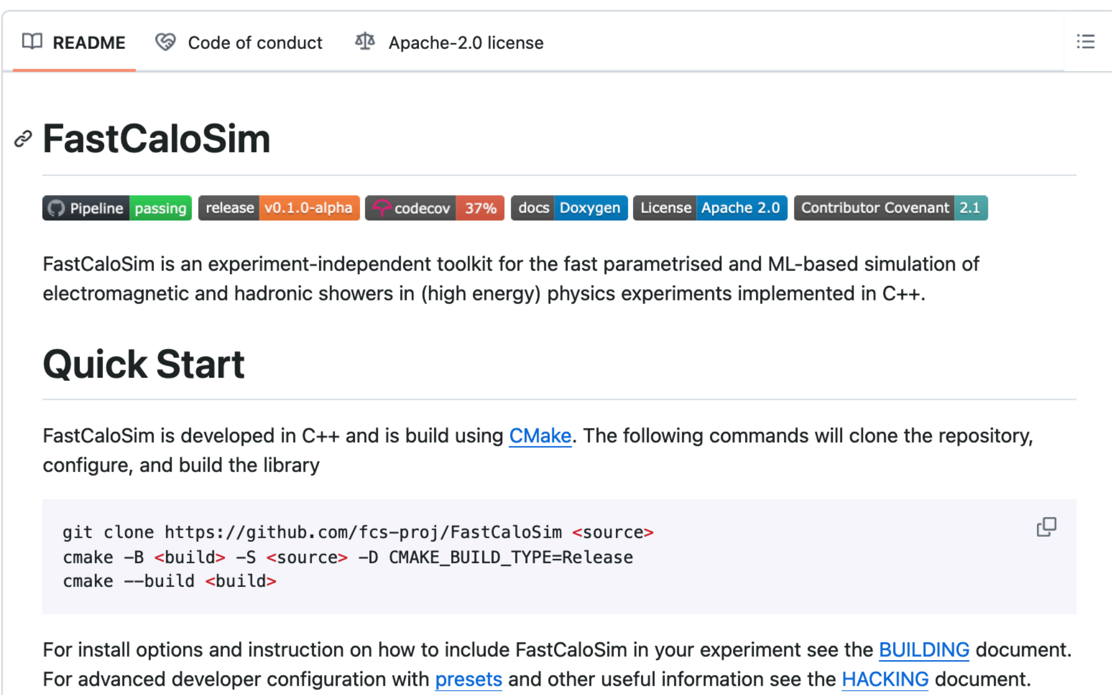
```
// Create alternative geometry handler for FCal
std::shared_ptr<FCal> fcal_geo = std::make_shared<FCal>();
// Load the FCal geometry from the files
fcal_geo->load(AtlasGeoTestsConfig::FCAL_ELECTRODE_FILES);
// Cast the FCal geometry handler to the geo interface
std::shared_ptr<CaloGeo> fcal_geo_handler =
    std::static_pointer_cast<CaloGeo>(fcal_geo);
// Set the alternative geometry handler for the FCal layers (21 - 23)
geo->set_alt_geo_handler(21, 23, fcal_geo_handler);
```

Class that implements method to return cell id, given a position in detector

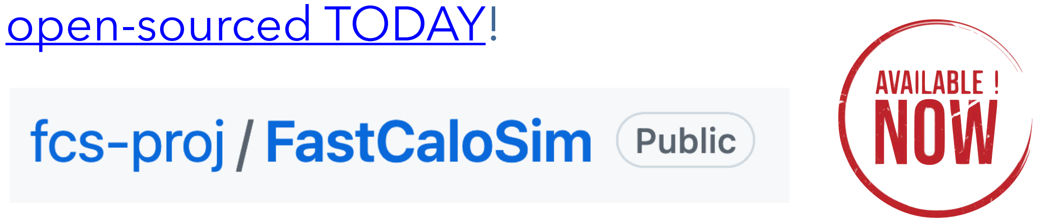
Alternative geo handler can be set for arbitrary detector layers

Repository: Overview





- The new experiment-independent FastCaloSim was open-sourced TODAY!

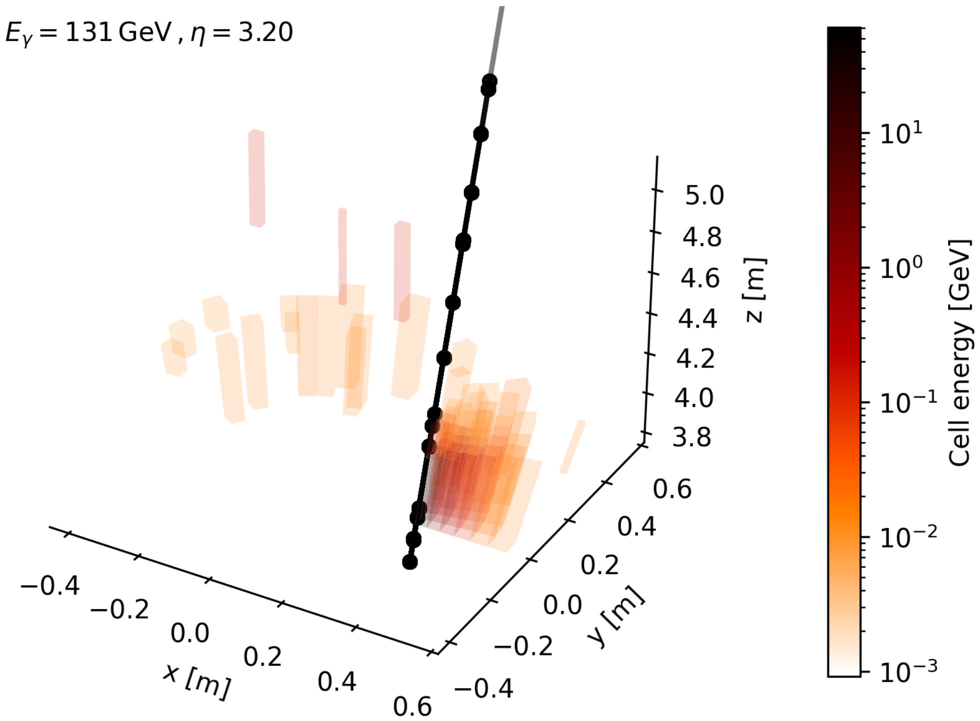


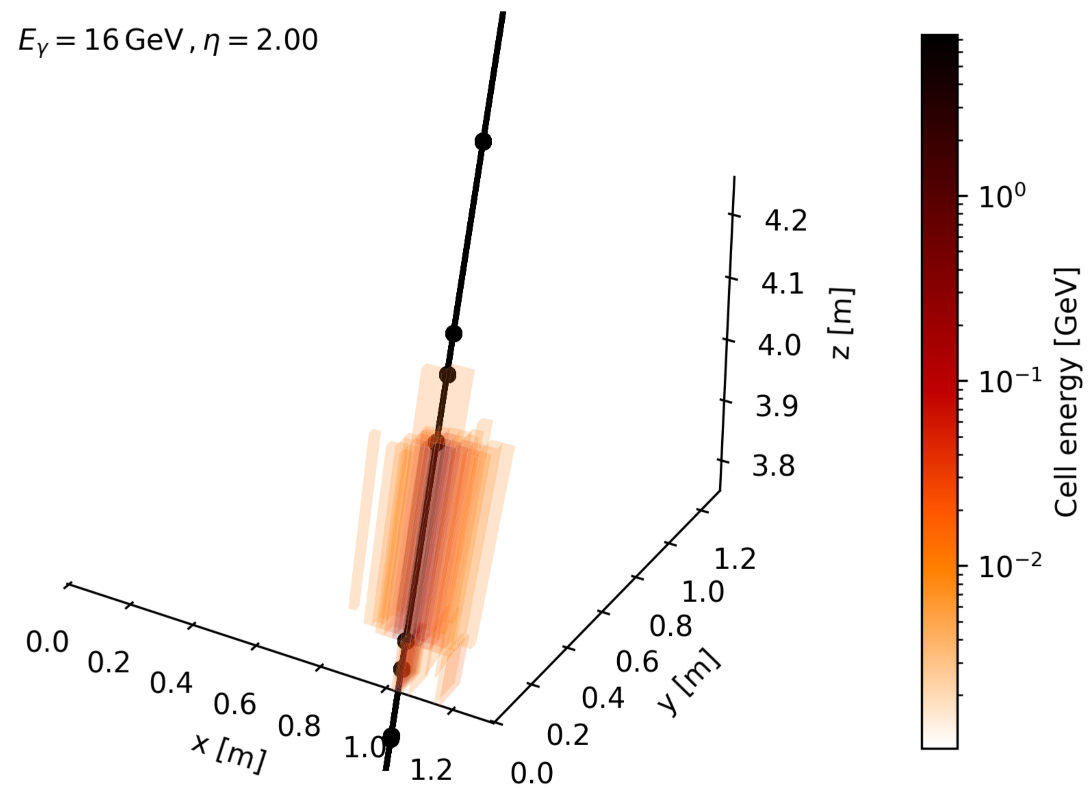
- Built with modern development practices in mind:

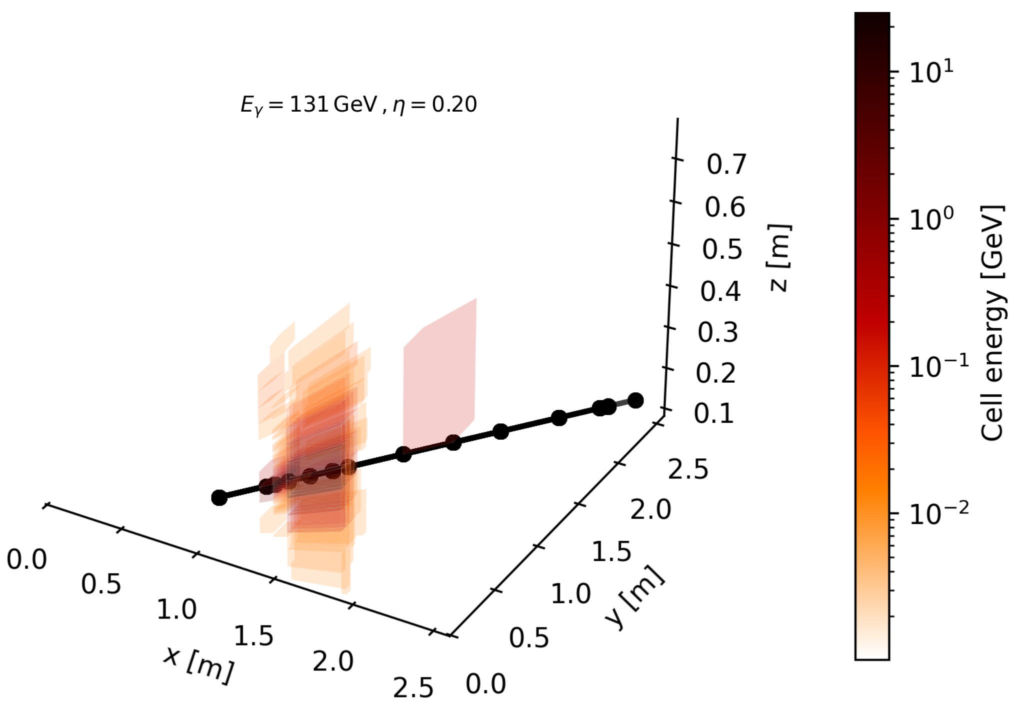
  1. Modern CMake configuration with CMake presets
  2. Unit testing with googletest
  3. Static code analysis with clang-tidy and cppcheck
  4. Code linting and formatting with clang-format
  5. Spell check with codespell
  6. LCov to provide test coverage information
  7. Docker images to create reproducible test and development environment

Continuously tested on alma9, ubuntu24 and LCG releases

Unit Tests: Simulation







- A full example implementation for FastCaloSim for ATLAS as Geant4 fast simulation model is provided
- The full simulation chain is unit-tested (transport, extrapolation, simulation)
- Visualisations of the shower simulation are created on-the-fly in the CI

Summary & Outlook

Summary

- For the first time, FastCaloSim is implemented as single external library that can be used outside Athena without any ATLAS geometry dependencies
- pyGeoSimplify allows to automatically generate clash-free simplified geometry needed by FastCaloSim
- New FastCaloSim repository follows modern practices and allows for far more efficient development
- Very much on track to fully phasing out ISF, creating a simple and streamlined (fast) simulation in ATLAS

Outlook

- Core library experiment-independent, but tools to allow easy creation of parametrization still in development
- Plan is to use Open Data Detector (ODD) as proof-of-concept implementation starting from generation of Geant4 input samples, creation of parametrization and simulation

Thank You

BACKUP