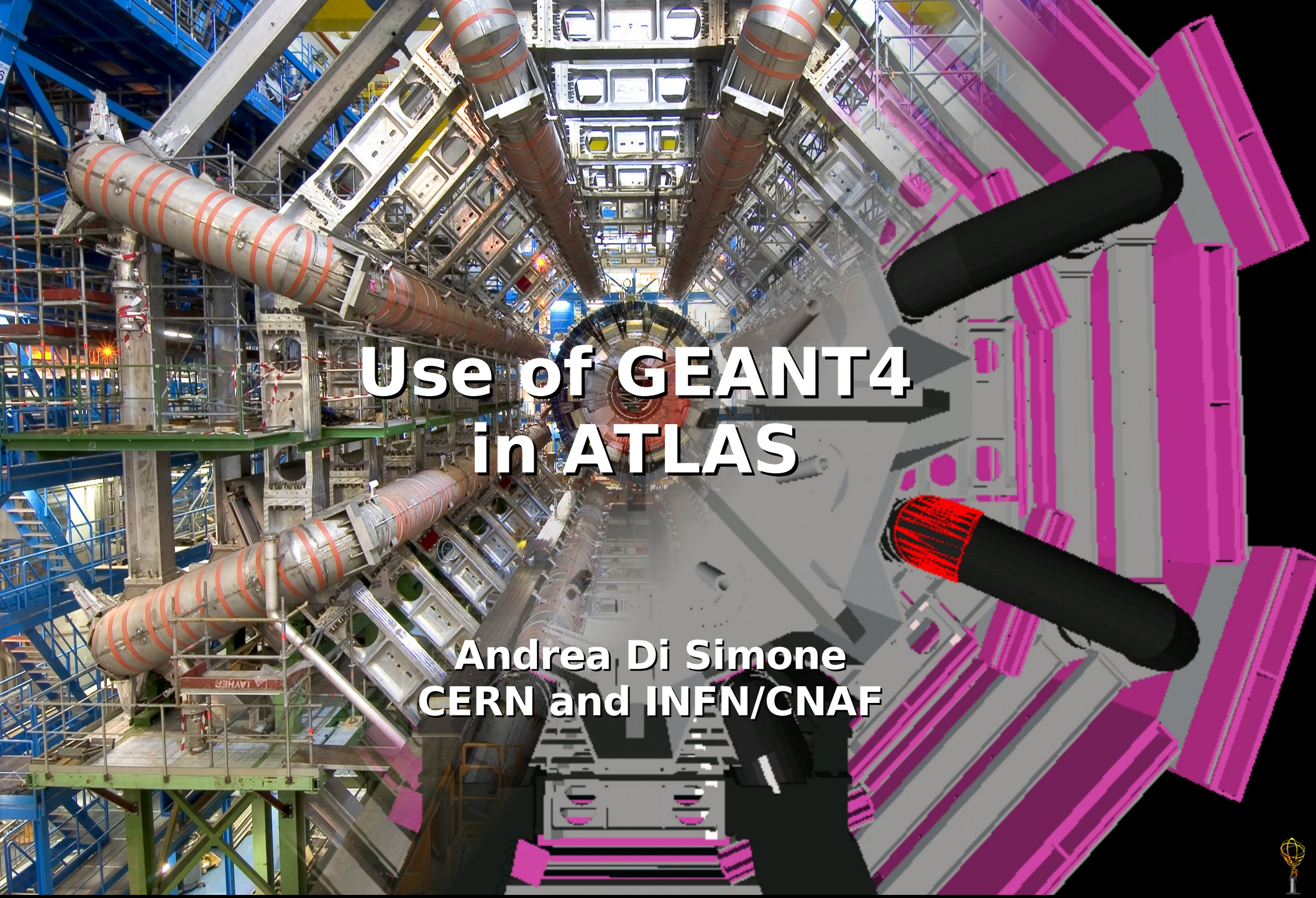


Use of GEANT4 in ATLAS

Andrea Di Simone
CERN and INFN/CNAF

Outline

- The ATLAS simulation software
  - Geometry description
  - Job configuration / user interface
  - Misaligned detector
  - Shower parameterization
- Computing performance
  - CPU time per event
  - Memory usage @ runtime
  - Eta dependence
  - G4.8.0 tests
  - Automated tests

The ATLAS simulation software

- Since 2002, the old G3 simulation has been replaced by a new G4-based framework as the official ATLAS simulation software
- Fully integrated in the Gaudi-based ATLAS offline framework (Athena)
- Used for the simulation of many different setups:
  - Full ATLAS
  - 2004 Combined test beam
  - Stand alone test beams
  - Cosmic commissioning
  - ...

The ATLAS simulation software

- Used for massive production on the grid:
  - Data challenges
  - CTB production
- Features include:
  - G4 geometry built at run time starting from an independent description
  - Choice between different setups done at runtime
    - The same code is used to run all the setups
  - Full job configurability through python scripts

Geometry description

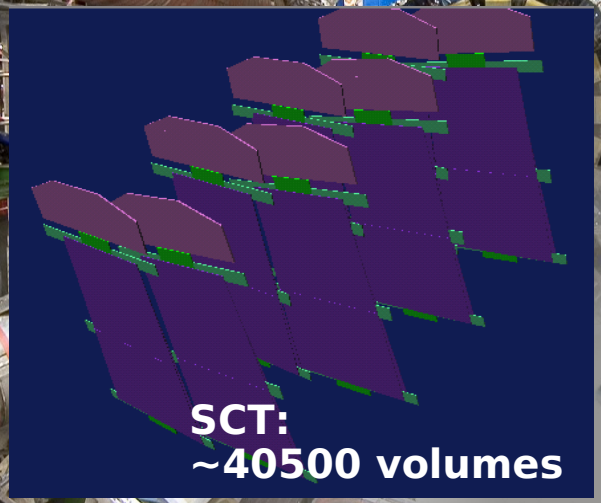
- The geometrical description of the ATLAS detector is implemented using a dedicated set of classes (GeoModel) with no dependencies on G4
  - *Exactly the same* description is used by the reconstruction jobs.
- The GeoModel description is built at run time starting from the geometry data bases, and automatically translated to a Geant4 geometry
- It is optimized to handle a complex geometry as the one of the ATLAS detector:
  - number of volumes is kept to the minimum
  - extensive use of parameterized volumes

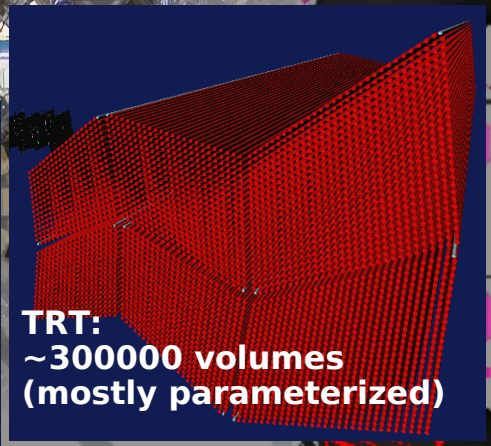
In spite of the optimizations, the memory usage of the description is quite large:

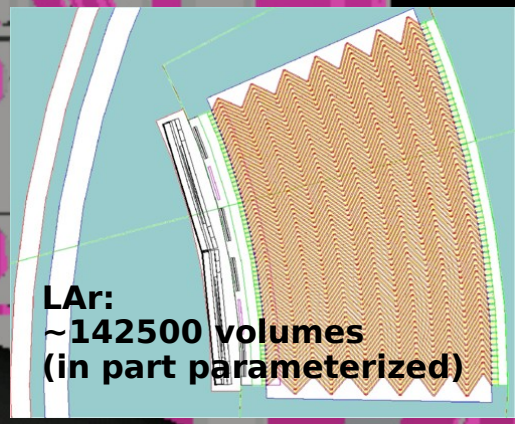
Task	Total memory (MBytes)
Pixel	5.6
SCT	9.1
TRT	3.1
InDet material	1.0
LAr	54.4
Tile	1.1
Muon	21.3
GeoModel total	95.7

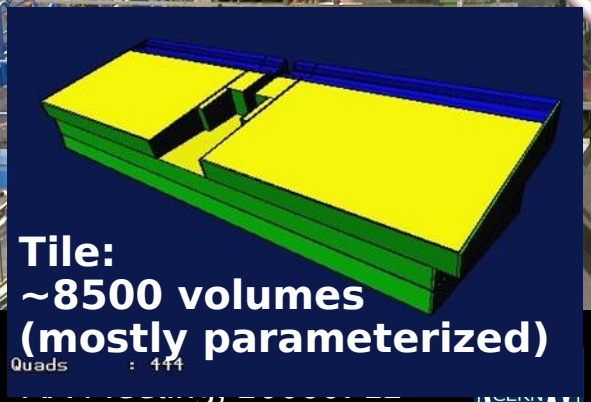
Geometry description

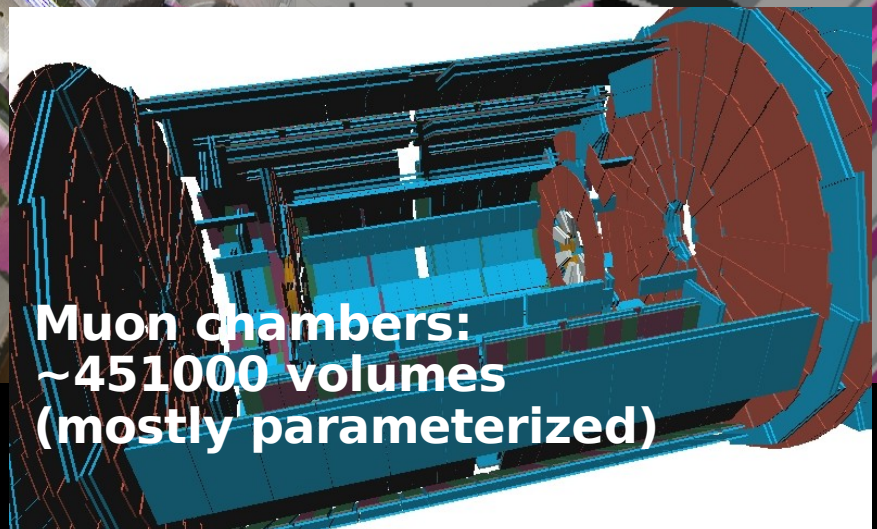
Pictures obtained using simulation display tools

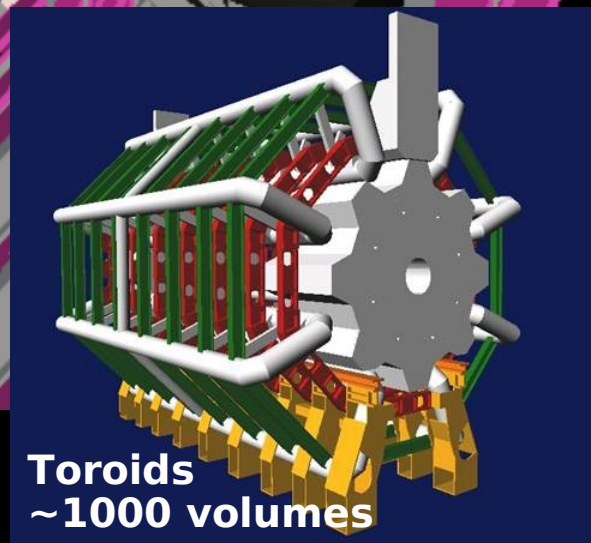










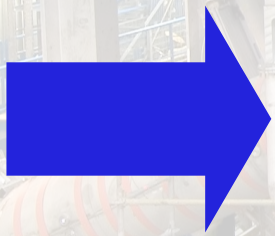


Job configuration

- All the ATLAS offline software is integrated in a Gaudi based framework (ATHENA)
- ATHENA uses the python language as a frontend to the final users
  - a job is controlled via a python script (jobOption)
    - input file(s)
    - which algorithms to run, in what order, with what configuration/parameters
    - output file
- The user has all the advantages of a full-featured scripting language
    - flexibility
    - ease of use
    - C++ bindings
- G4 has no native interface to python
  - Simulation configuration is normally achieved passing specific command lines to G4
    - several command lines can be grouped in *macro files*

Job configuration

- For the simulation of a complex experiment such as ATLAS, the number of G4 *macros* involved tends to diverge rapidly
  - lack of flexibility
  - maintainability issues
- Moreover, the *macro* based interface does not integrate in the jobOptions mechanism



A python layer has been implemented, using bindings to the underlying C++ simulation code, which allows to fully configure a job with python commands

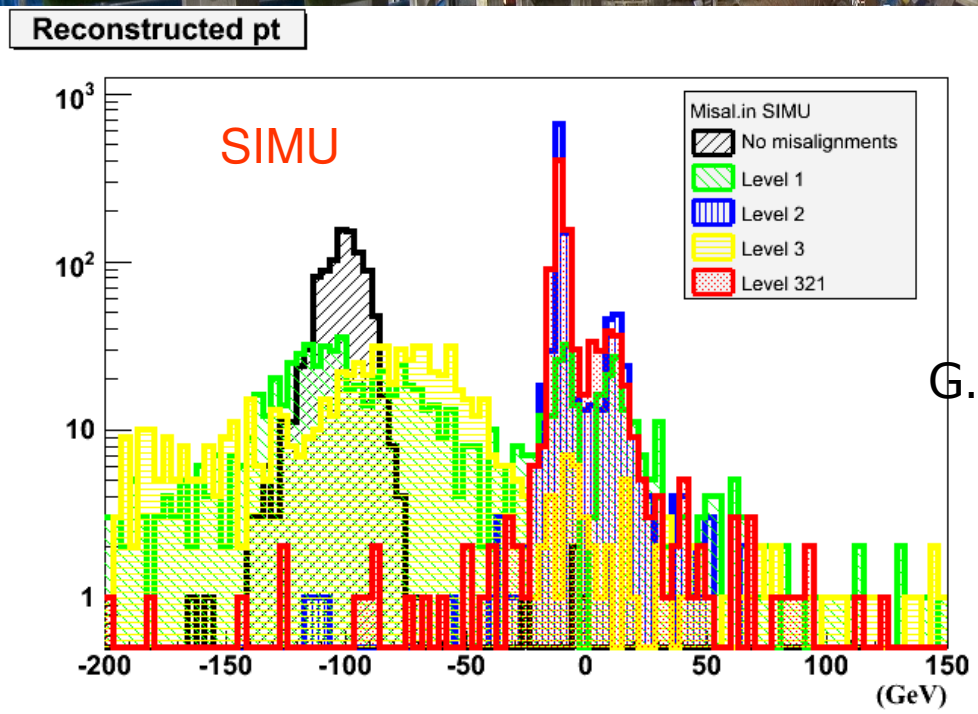
- With the resulting user interface (PyG4Atlas), a simulation job can be controlled and configured (**also interactively**) from the normal ATHENA prompt
- The effort requested to the user in order to interact with the simulation is thus reduced to a minimum

Detector misalignments

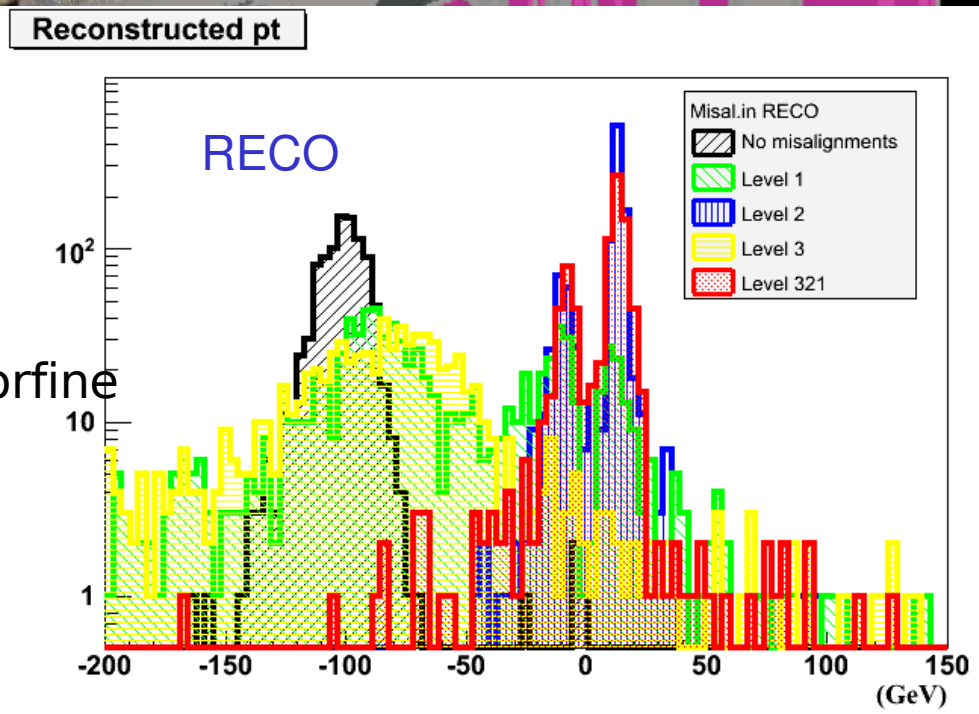
- Detector geometry as installed is of course slightly different from the design one
- Need to understand the impact of these misalignments on the physics performance, and include them in both simulation and reconstruction
- Misalignment values stored in the geometry DB
  - Retrieved at the beginning of the job using the run number
  - GeoModel uses that info to build the misaligned geometry

Detector misalignments

- Study done by the Inner Detector:
  - Single electrons, Pt=100GeV
  - Perform misaligned reconstruction on nominal simulation and vice versa

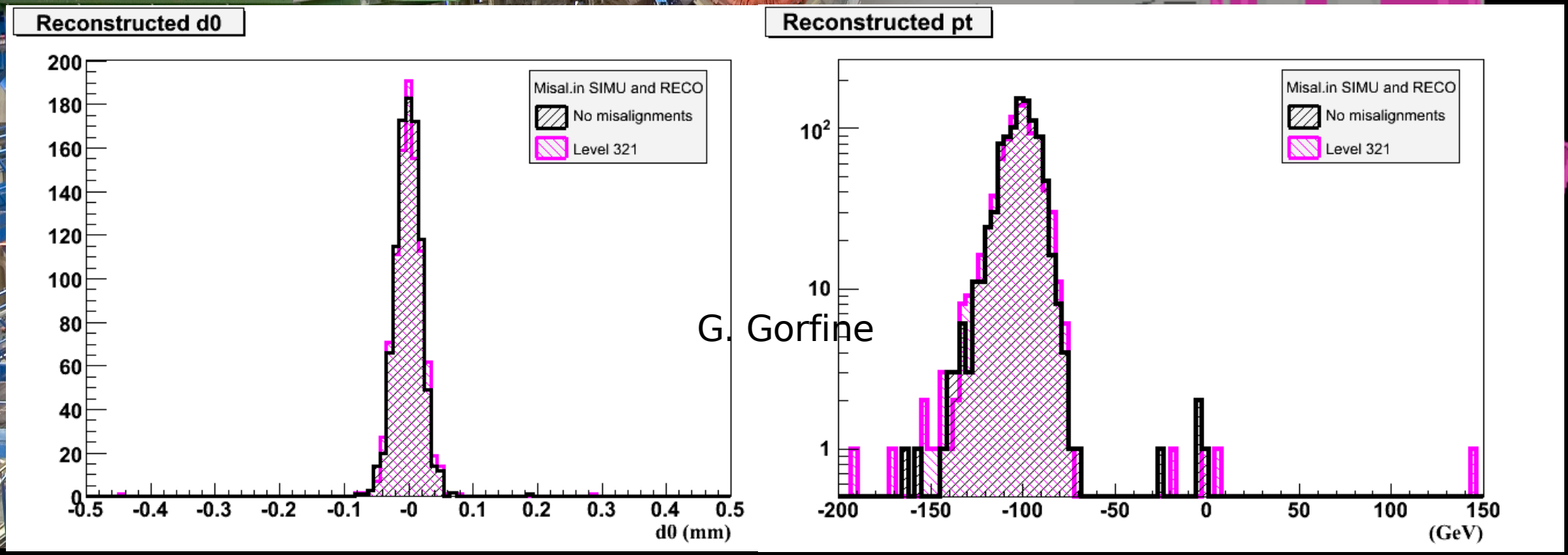


G. Gorfine



Detector misalignments

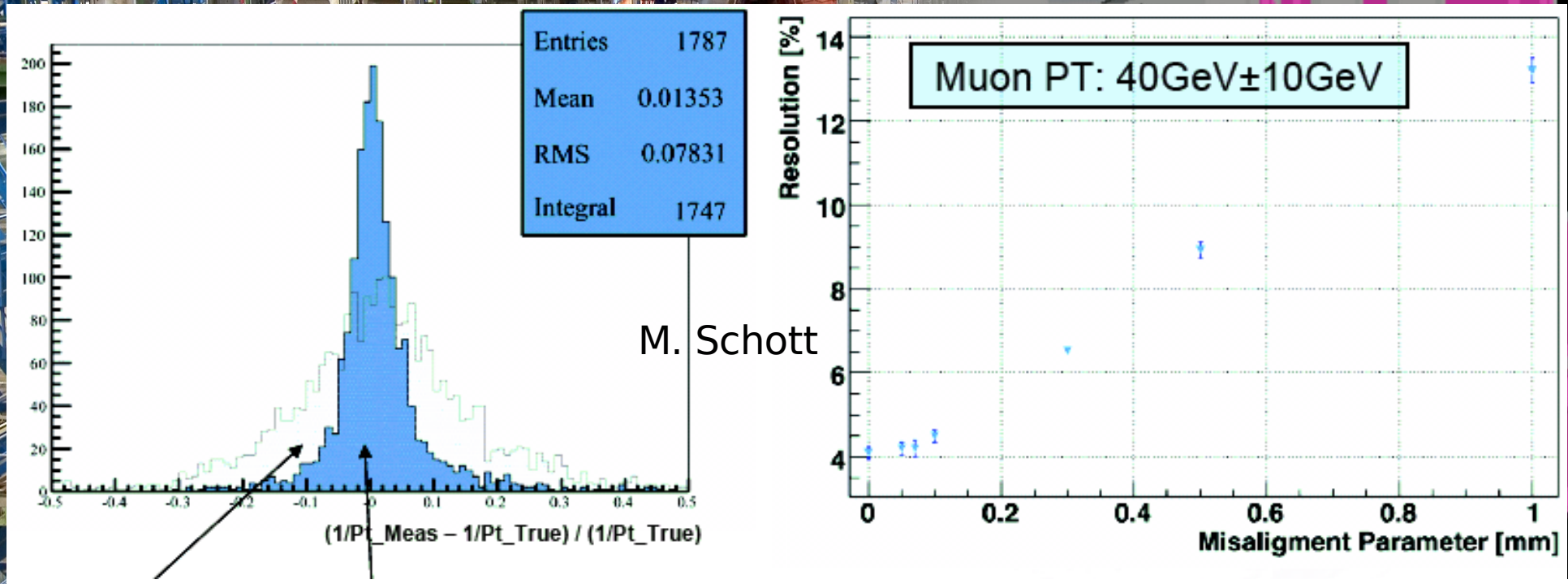
- When misaligning both reconstruction and simulation, good physics performance is obtained.



G. Gorfine

Detector misalignments

- Effect of chamber misalignment on Pt resolution in muon spectrometer
- Muon chambers shifted and rotated randomly



Shower parameterization

- Most of the simulation time is spent for EM showers in the calorimeters
- Shower parameterization would allow to speed up this process
  - replace the full simulation of the shower propagation with ad-hoc placement of energy depositions according to parameterized shower shape
- Parameterisation used in ATLAS is based on Grindhammer & Peters, *arXive:hep-ex/0001020v1h*
- Work presented in the following slides done by A. Waugh and E. Barberio

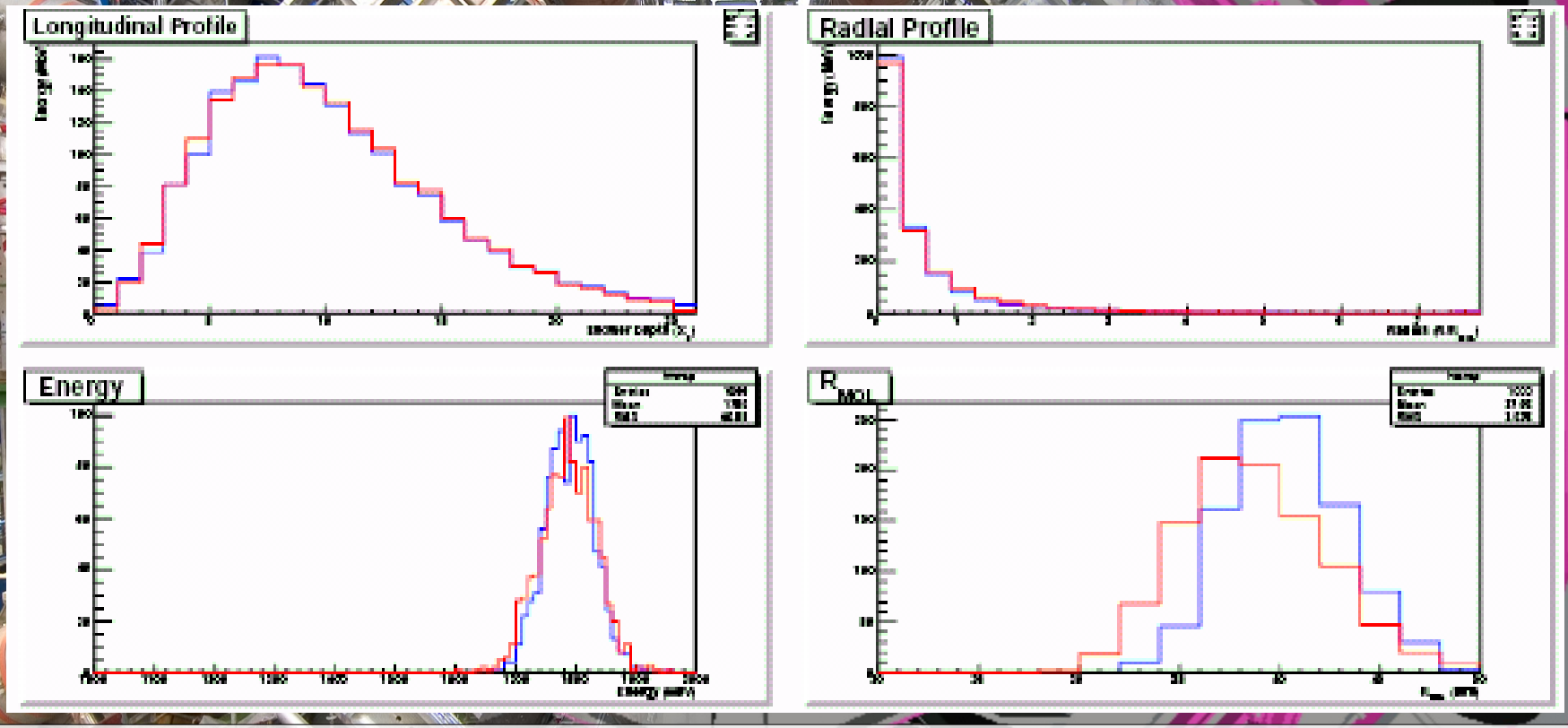
Shower parameterization

- Conditions for parameterization
  - Only electrons/positrons are parameterized.
  - Photons are returned to full sim and the first electron produced is parameterized.
  - Particle must be within allowable energy range, if not it is returned to full simulation
  - Shower must be contained within the calorimeter, checked at Z(95%) and R(95%).
- Handling of Leakage
  - If shower not contained, returned to full simulation; when the shower produces a fully contain electron, it is parameterized
- Showers starting before the calo
  - Particles are tracked using full simulation until they enter the calo, then each is parameterized separately

Shower parameterization

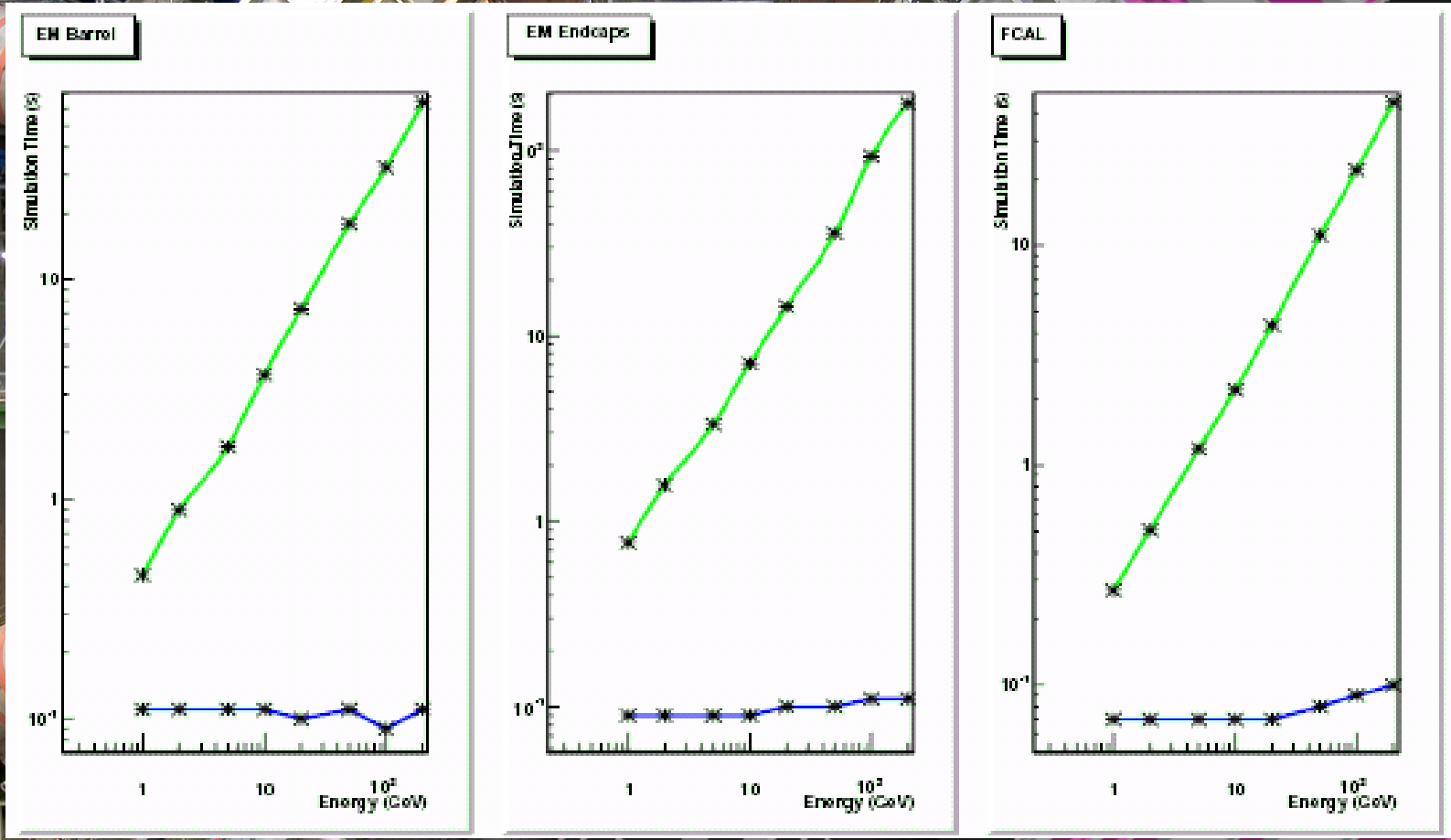
- If the conditions are satisfied
  - the track is killed
  - generate fake steps of a chosen length (10μm) along the initial electron trajectory; calculate energy deposited and number of spots for each step
  - applying sampling fluctuation, taking into account the calorimeter resolution, calculate energy and position of each spot
  - generate hits and fill fake step to feed the sensitive detector
  - follow the full simulation chain to process the hits
  - loop until the total shower energy is deposited

Shower parameterization



- EM barrel calo, 10GeV electrons, 1000 events
- Full and Fast simulations
- Results obtained with analysis at the G4Hit level
- Small disagreement on R_{MOL}, still to be clearly understood

Shower parameterization

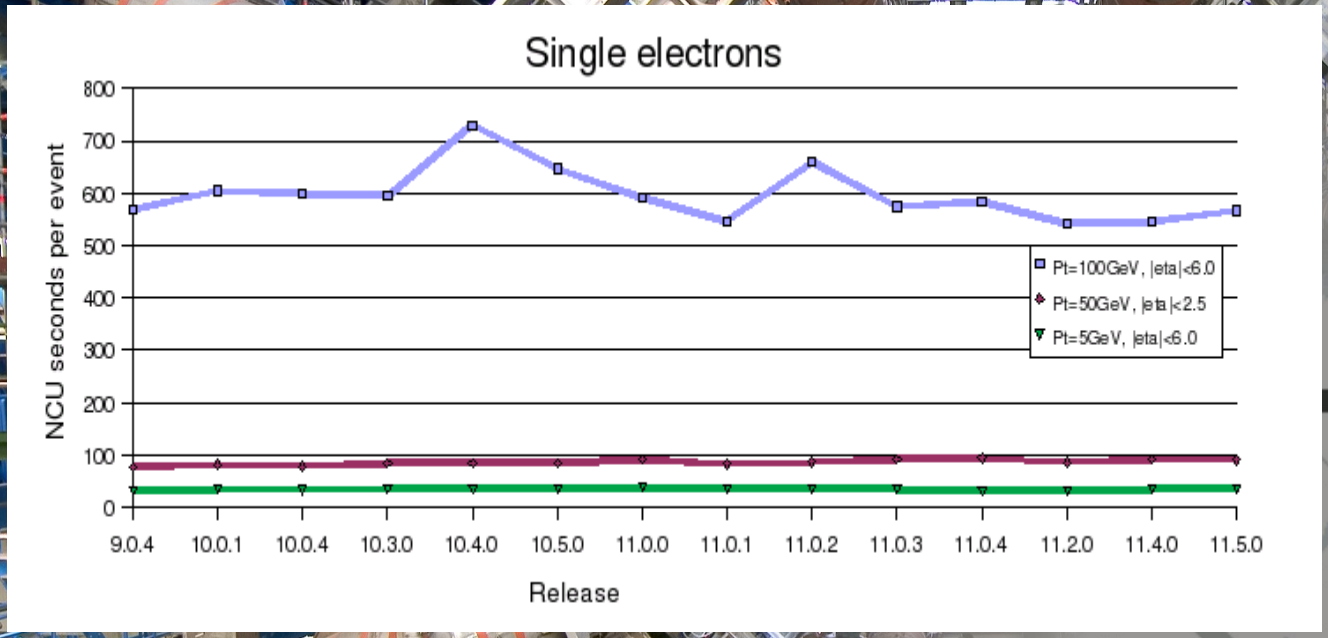


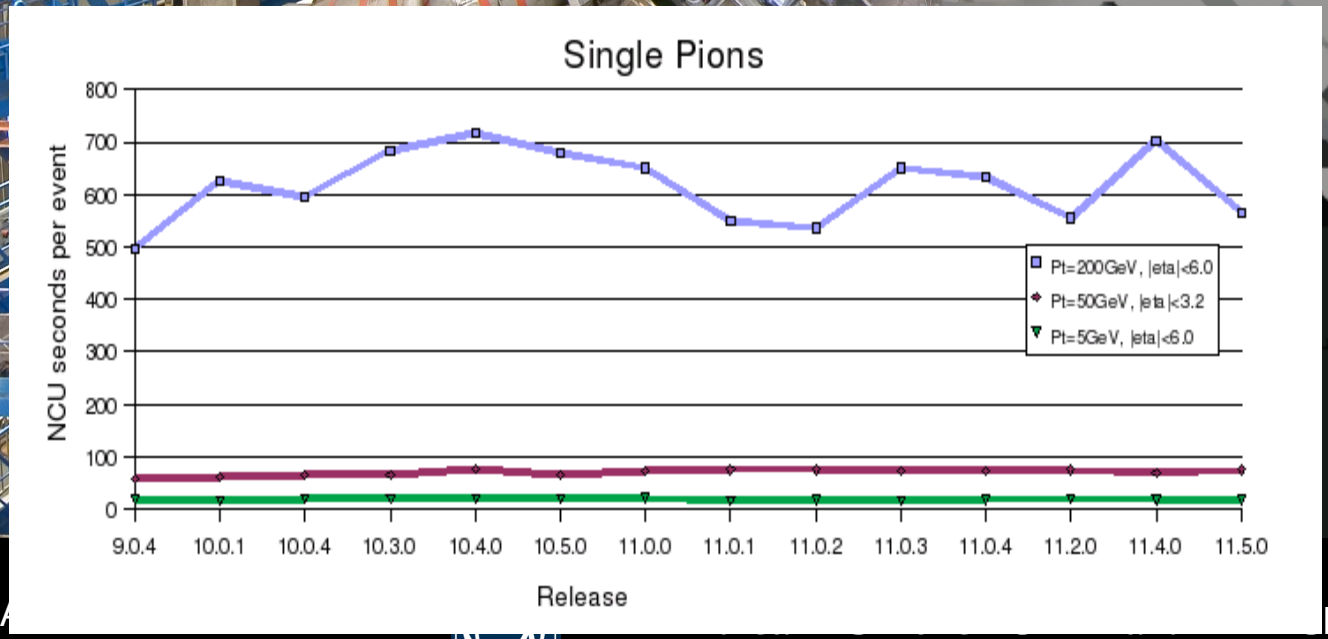
- Simulation time for single electrons as a function of the energy
- Full and Fast simulations

Computing performance

- Computing performance is kept under continuous monitoring
  - CPU time per event
    - Measured using different samples, both single particles and full physics events
  - Memory at beginning of first event
    - Contributions from each initialization step are measured
  - Memory at end of run
    - Check the absence of leaks

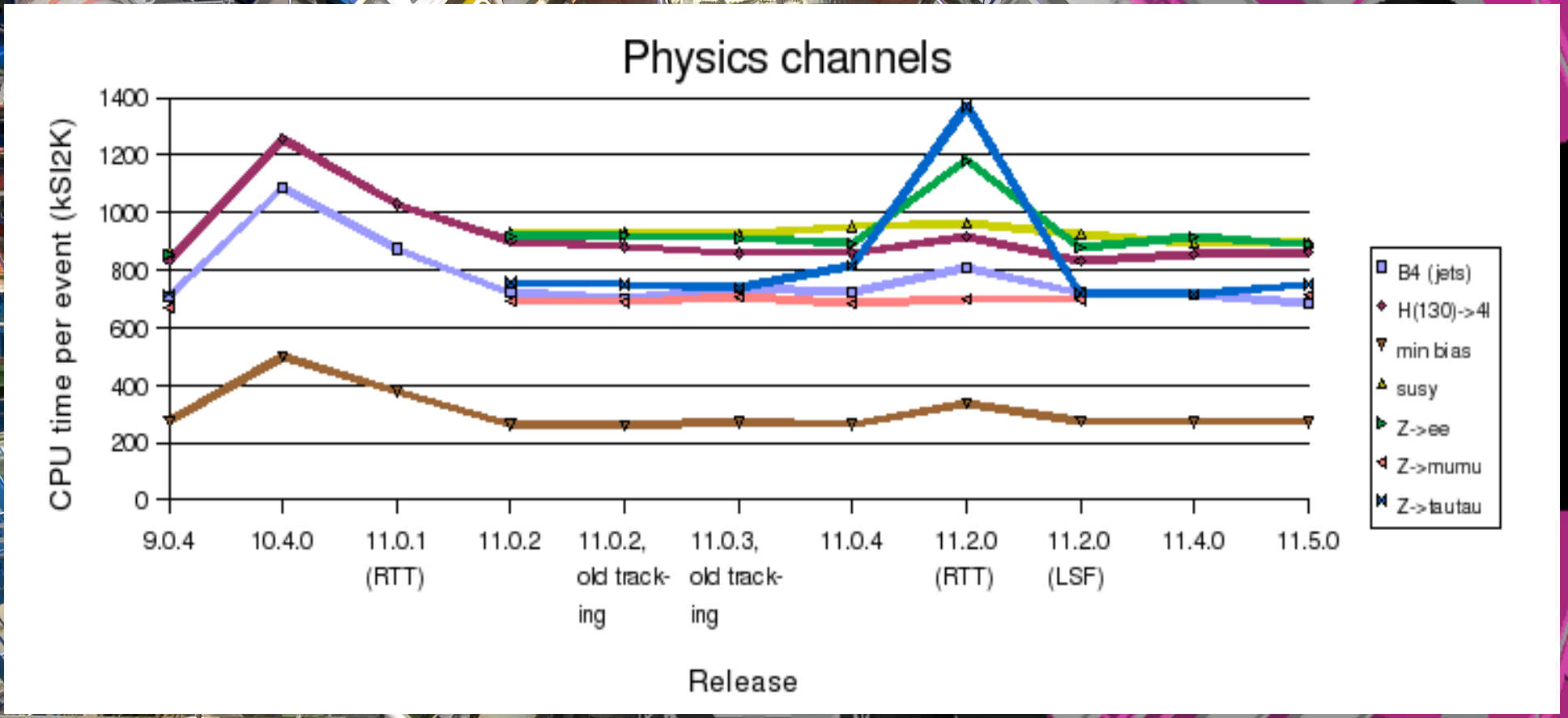
CPU time per event: single particles





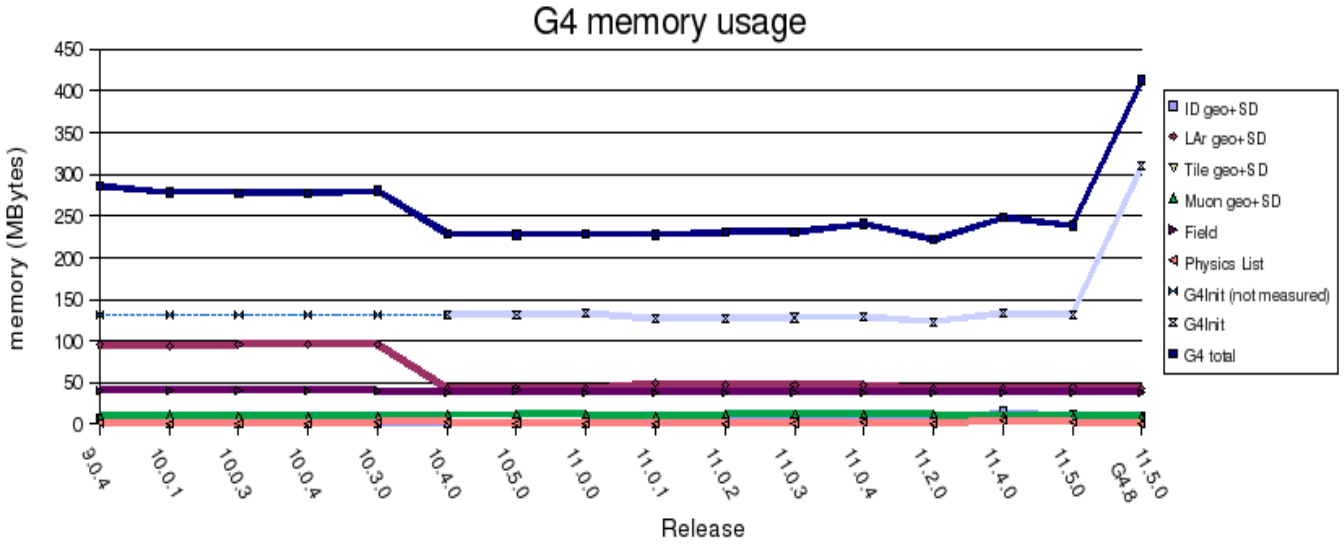
- Single particle performance well under control since the last big production (~1.5 years ago)
- Similar plot available for single muons

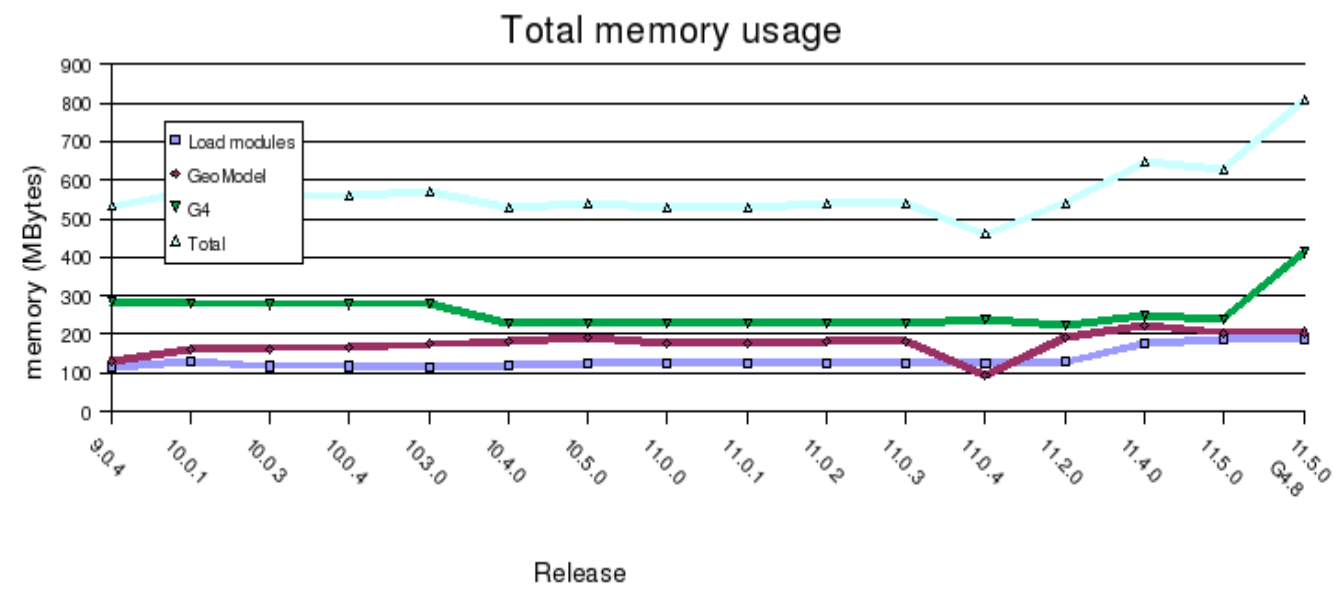
CPU time per event: physics events



- Performance of full physics events is also compatible with the one of release 9.0.4
- Some big oscillations are due to problems (fully understood) in the CPU time measurement

Memory usage



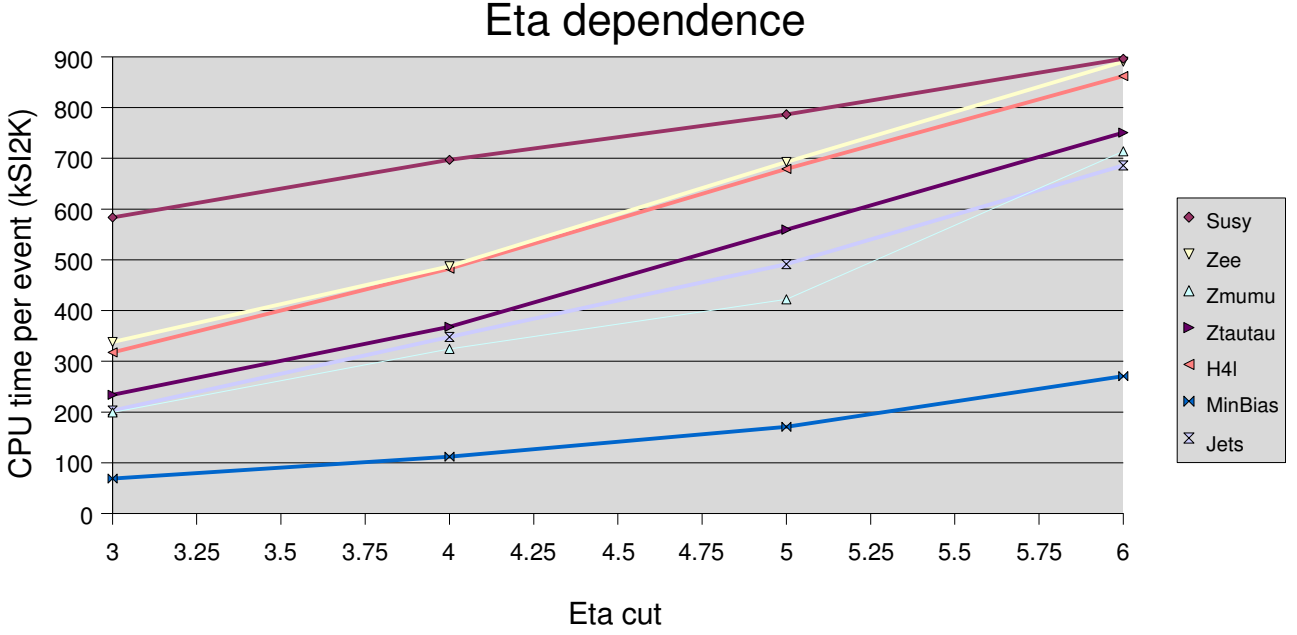


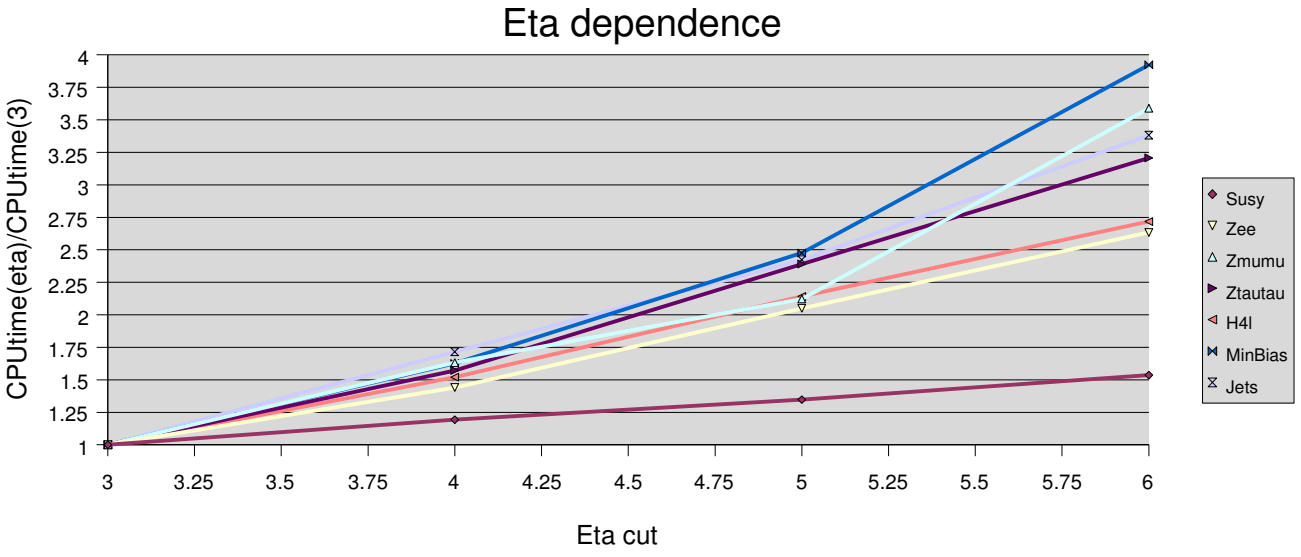
- Memory usage variations observed up to now are not worrying, and are however fully understood
- Small problem with g4.8.0.p01 (now fixed)

Eta dependence

- Users can decide at runtime to limit the simulation only to a certain eta interval
- This has a strong impact on performance
- G4 ATLAS simulation done by default in the eta range (-6,6)
- Older simulation (G3) used to work with a different eta range (-3,3)
- A clear understanding of the eta dependence of the simulation time allows to:
    - Identify the regions where most of the CPU time is being spent
    - Better compare performance with the one by G3

Eta dependence





- Average CPU time per event is measured for different eta intervals using full physics samples.
- As expected, the effect is bigger in minimum bias events. This is clearly visible in the lower plot, where the CPU time is normalized to the time needed in -3<eta<3.

G4.8.0 tests

- Several tests done in order to understand the impact on computing performance of the new msc implementation
- The same ATLAS simulation software (Athena release 11.5.0) has been built twice, using G4.7.1.p01 and G4.8.0.p01
- G4.8.0 was tested with several different configurations:
  - Default: with the new msc and ATLAS standard cuts
  - Special cuts: new msc and 1mm cut for all volumes
  - Msc71: plugging in g4.8.0 the msc implementation from g4.7.1
  - Nsl: same as “Default” but inhibiting the step limitation by the msc

G4.8.0 tests

CPU time per event (kSI2K)

	G4.7	G4.8	G4.8 1mm	G4.8 msc71	G4.8 nsl
Susy	896.46	2019.66	1690.29		849.62
Zee	890.47	1916.37	1573.31	850.41	760.2
Zmumu	713.76	1369.27	1201.99	642.02	671.32
Ztautau	750.73	1427.59	1253.83	743.69	677.34
H4l	862.15	1788.29	1429.86	884.07	783.73
Jets	685.8	1442.15	1364.75	701.05	753.6

	Ratio	Ratio 1mm	Ratio msc71	Ratio nsl
Susy	2.25	1.89		0.95
Zee	2.15	1.77	0.96	0.85
Zmumu	1.92	1.68	0.9	0.94
Ztautau	1.9	1.67	0.99	0.9
H4l	2.07	1.66	1.03	0.91
Jets	2.1	1.99	1.02	1.1

- Timing results for full physical events are shown, as obtained in the different configurations. Ratios wrt G4.7.1 timing results are reported as well.

G4.8.0 tests

- The increase in time is really due only to the new msc implementation, and it is connected with the step limitation
- Setting all the production cuts to 1mm does not help in reducing the processing time
- In order to have timing results compatible with the ones we used to have with g4.7, we would probably have to choose between
  - Using the old msc implementation
  - Using the new msc implementation, switching off the step limitation

Automated tests

- RTT (RunTimeTester) is the tool provided by Athena for automated tests of the software functionality
- Since last year, it is also used to test the G4 ATLAS simulation:
  - "Fast" tests with single particle samples done for each nightly build
    - Time per event
    - Memory usage at each event
  - Tests using full physical events done at each Athena release
  - Results archived and presented on a web page for easy consultation.

Conclusions

- The G4 based ATLAS simulation is nowadays fully-featured:
  - Allows detector misalignment
  - Physics validation of shower parameterization ongoing
- Its computing performance is continuously monitored:
  - Since the last big production, both CPUtime per event and memory usage remained constant, in spite of the addition of new features
- Computing performance tests with G4.8 already done. Physics performance tests ongoing