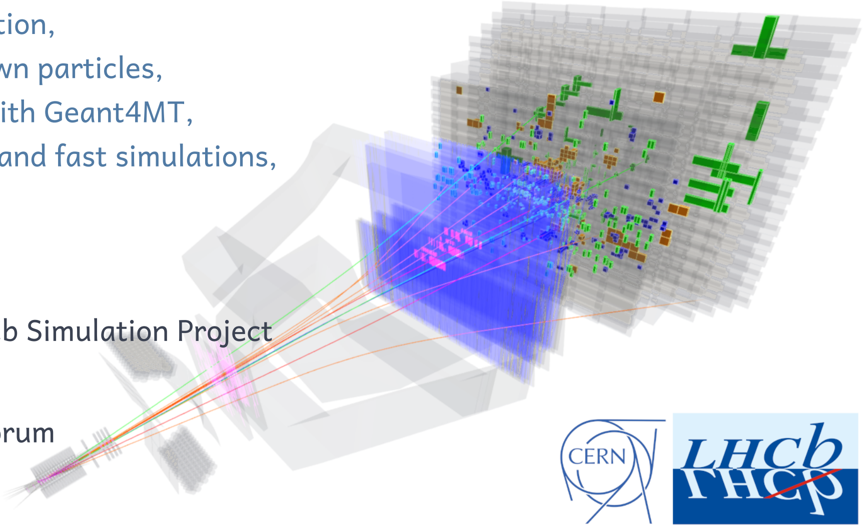
Simulation in LHCb

- 🎉 timing optimization,
- 🎉 handling unknown particles,
- 🎉 visualizations with Geant4MT,
- ⌛ custom physics and fast simulations,



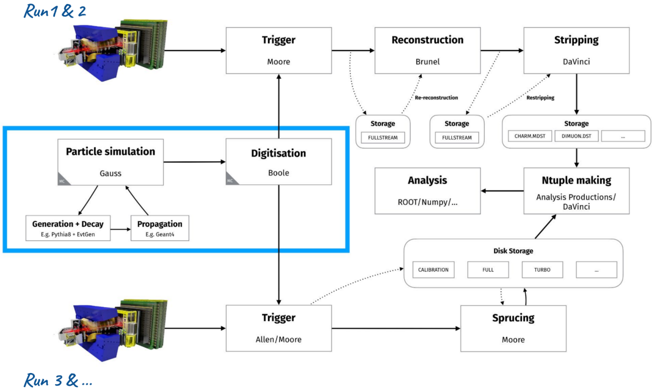
by Michał Mazurek
on behalf of the LHCb Simulation Project

on February 2, 2023
Geant4 Technical Forum

Geant4 in MC Productions

Production campaigns

- Sim09: Geant4 9.6.p04
  - x86_64-slc6-gcc48-opt,
  - in use since 2016, being retired,
- Sim10: Geant4 10.6.p02
  - x86_64_v2-centos7-gcc11-opt,
  - in use since 2020 for Run3 and 2022 for Run2,
  - intermediate developments with 10.2, 10.3, 10.4 used for test productions,
- Sim11: aim to move to Geant4 10.7 for production
  - x86_64_v2-centos7-gcc11-opt,
  - already used in fast simulation developments,
  - will require extensive physics validation,



Simulation in LHCb

Timing optimization

- 🎉 performed for **Sim09 and Sim10**,
- 🎉 with **monitoring & performance** tracking tools:
  - 👉 Simulation tests in LHCbPR,
  - 👉 Gauss Metrics (timing and memory),
  - 👉 detailed timing in G4 volumes,

Timing: Sim10 vs Sim09 for Run2 detector

G. Corti, D. Popov, A. Valassi et al.

Overall timing is from start to stop of a complete simulation job

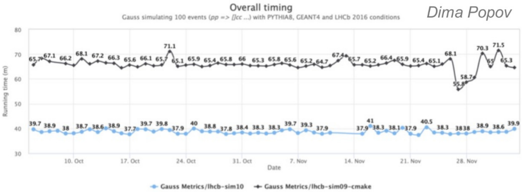
- *disclamer: measurements done in LHCb nigthly system*

Results from nigthly builds with latest candidate pacthes for release

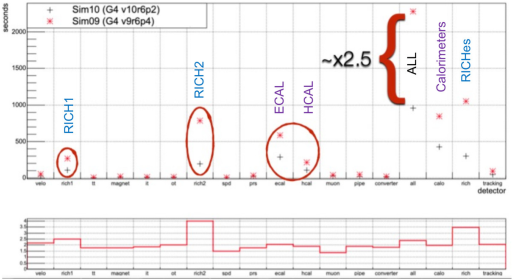
Sim09 – G4 9.6.p04
Sim10 – G4 10.6.p02

builds in LHCb stack
with some patches as-needed

Dima Popov



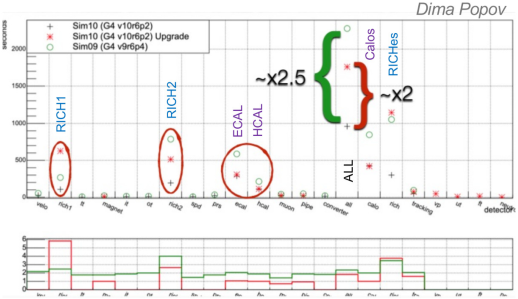
Dima Popov



Big improvement in Sim10 from G4 EM physics and LHCb optimization, in particular for RICH backgroud processes, also new compilers, rewritten code,…

G4 detailed timing: Sim09 vs Sim10 vs Run2/Run3

G. Corti, D. Popov, A. Valassi et al.



Sim09 – G4 9.6.p04
Sim10 – G4 10.6.p02

builds in LHCb stack
with some patches as-needed

Comparing only detector effects
- *identical 2016 running conditions*

The difference in the RICHes (x 5 and x 6) reflects in x 2 for Run3 vs Run2 detector in Sim10 ?

Vincenzo Innocente profiled the LHCb Grid workload in the context of the WLCG HepSCORE Benchmarking Task Force and reported that a single G4 10.6 function *accounted for 40% of the CPU time, i.e. G4LogicalBorderSurface::GetSurface*

He also noted that this function changed from G4 10.6 to 10.7

Speeding up Sim10

G. Corti, D. Popov, A. Valassi et al.

Moving to Geant4 10.7 (or higher) is not feasible

- Requires thorough physics validation, $\mathcal{O}$(months)

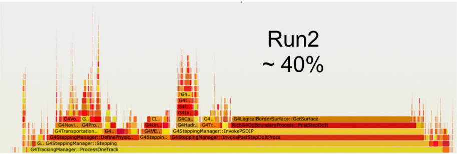
Porting the change to Sim10

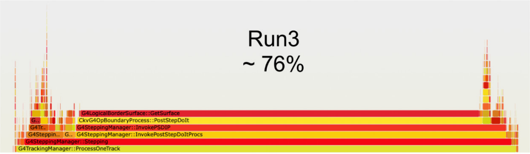
- Andrea Valassi prepared a patch for the LHCb G4 10.6.p02 build taking `G4LogicalBorderSurface` from G4 10.7
    - Very encouraging results from his simple tests on few events
- Deploy the patch in the nightly slots and perform in depth checks

Stack trace sampling

G. Corti, D. Popov, A. Valassi et al.

Dima Popov





Stack analysis shows one large increase in calls

G4LogicalBorderSurface::GetSurface

Related to RICH

RichG4OpBoundaryProcess
CkvG4OpBoundaryProcess

Sajan Easo

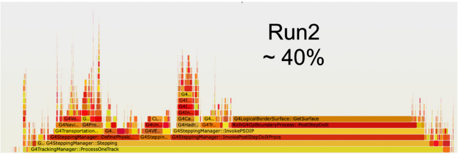
In Run3 RICH simulation

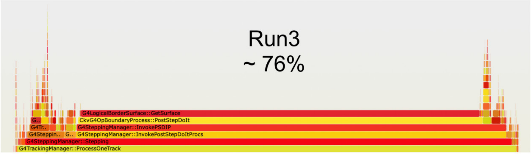
- Average yields is higher
- More optical surfaces

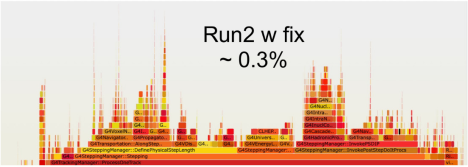
Run2: ~ 500 HPDs
Run3: ~ 3000 MaPMTs

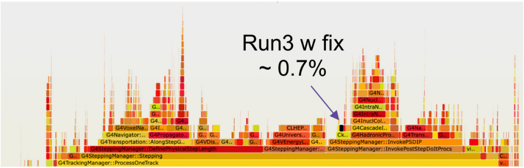
Sim10 stack trace sampling

G. Corti, D. Popov, A. Valassi et al.









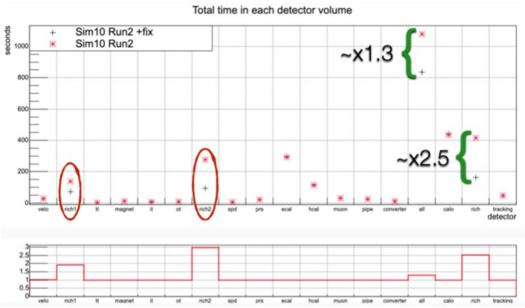
`G4LogicalBorderSurface::GetSurface` surfaces stored in `G4LogicalBorderSurfaceTable`

in 10.6.x `std::vector` -> loop over dynamic array to find element

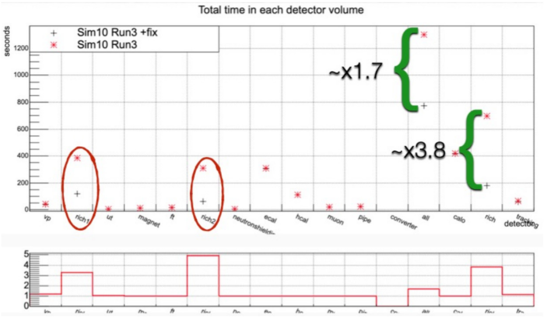
in 10.7.x `std::map` -> lookup element: average $\mathcal{O}(\log n)$

G4 detailed timing with the fix

G. Corti, D. Popov, A. Valassi et al.



Run2



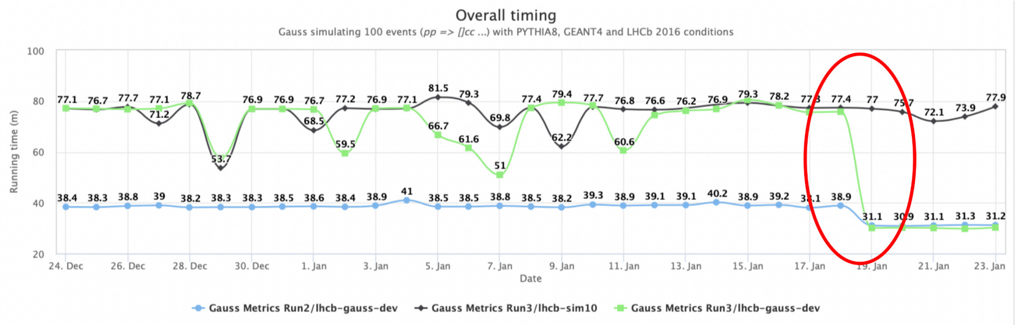
Run3

Comparing only detector effects

- *identical 2016 running conditions*

Sim10 overall timing with the fix

G. Corti, D. Popov, A. Valassi et al.



Sim10 Run2 (w fix) vs Sim10 Run3 (w fix) vs Sim10 Run3 (w/o fix)

⚠

Overall timing is from start to stop of a complete simulation job

- *disclamer: measurements done in LHCb nigthly system*

Comparing only detector effects

- *identical 2016 running conditions*

Simulation in LHCb

Handling unknown particles

🎉 characteristic for LHCb physics program,

🎉 solution to avoid many of classes for each particle…

Particles unknown to Geant4

M. Veltri et al.

LHCb generation ➔ *pp*, *PbPb* collisions ⊕ *pHe*, *pAr* fixed target

$\underbrace{\text{i.e. exotic particles, excited B, heavy ions, ...}}_{\text{❗ unknown to Geant4!}}$

❗ **Geant4:** each particle = unique static object (G4ParticleDefinition)

👉 too many particles!!!

💡 **New approach:**

- 👉 create the produced "unknown" particles/ions dynamically,
- 👉 intercepted right after the event generation phase,
- 👉 for particles instantiate a `G4ParticleDefinition` and **attach the processes**:
  - ⊕ transportation,
  - ⊕ (pre-assigned) decay,
  - ⊕ ionization/multiple scattering,

Particles unknown to Geant4 *continued*

M. Veltri et al.

❗ **But**, adding processes to particles need to be done already at initialization!

- 👉 **Solution:** implement at initialization all the particles known to the `LHCb::ParticlePropertyService` and create an instance of `G4ParticleDefinition`,

👉 **Unknown ions are treated differently:**

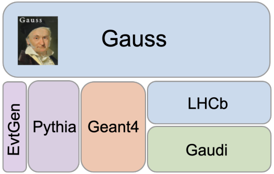
- 👉 **abstract prototype exists**: `G4GenericIon`,
- 👉 new ions can dynamically created using `GetIon` method of the `G4IonTable`,
- ❓ a similar approach for `G4GenericParticle`...

Simulation in LHCb

Visualizations with Geant4MT

- 🎉 to be integrated with **Sim11 (Gauss-on-Gaussino)**,
- 👉 **runtime visualization** in Geant4 not easy in the **mutli-threaded** environment of Geant4 & Gaudi,
- 🎉 Phoenix event display with the simulated data,

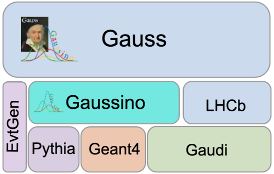
Transition from Sim10 to Sim11



Gauss

EvtGen Pythia Geant4 LHCb Gaudi

(a) Gauss **(Sim10)** current dependencies



Gauss

EvtGen Gaussino LHCb Pythia Geant4 Gaudi

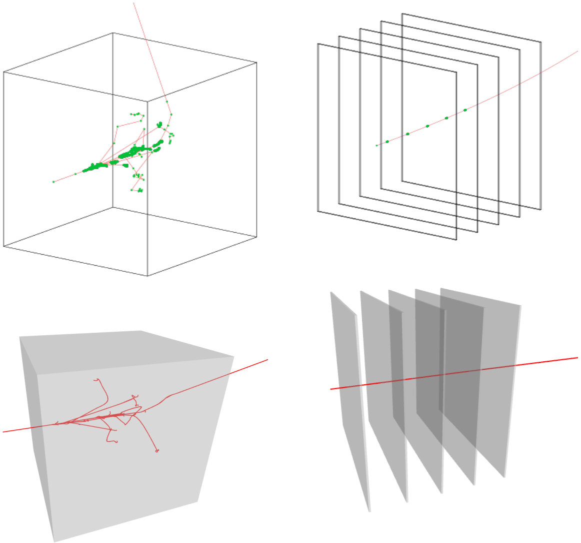
(b) Gauss-on-Gaussino **(Sim11)** dependencies

- Sim11 (Gauss-on-Gaussino)
  - first version to support both **DetDesc & DD4hep**,
  - **multithreaded** event loop,
  - based on **Gaussino**'s core functionalities,
- Gaussino
  - only experiment-independent components,
  - ideal **test bed** for new developments (new sub-detectors),

Visualization in Gaussino

F. Bilandžija, M. Mazurek

- **Geant4** visualization drivers
  - available at **runtime**,
  - volume **overlap checks** possible,
  - **G4 data** only,
  - drivers: ASCIITree, OpenGL, DAWN, HepRep
- **Phoenix** event display
  - available in an **external** tool,
  - geometry to be converted from **GDML** to a dedicated format,
  - both **G4 or LHCb** simulation data,
  - using **JSON exporter**,
  - **simulation vs. reconstruction** data comparison possible,



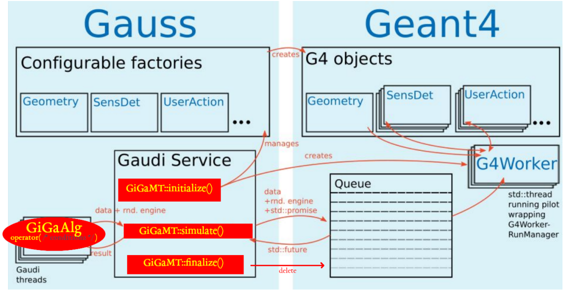
Visualization backend in Gaussino/Gauss

Possible data containers

- **Geant4:** G4Trajectories, G4Hits, ...
- **LHCb:** MCHits, MCCaloHits, ...
- **EDM4hep** in the future?
- additional optimization
  - only MCTruth,
  - only trajectories from tracking, etc.

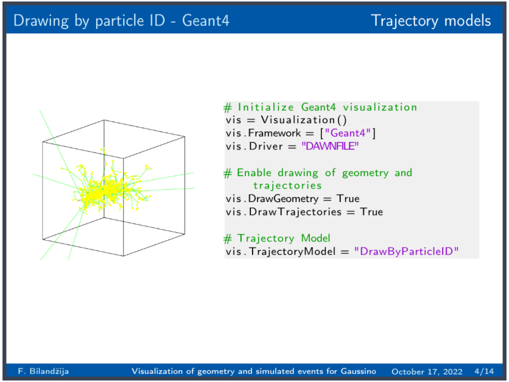
Recipe for Gaudi & Geant4MT

- implement `G4VisManager` that spawns an additional `G4VIS` thread,
- ensure `G4Event` lives **long enough** for
  - G4 main simulation,
  - Gaudi algorithms,
  - visualization postprocessing,
- **information exchange** between custom `G4RunManagers` at the **right moment**,



High-level Python configuration

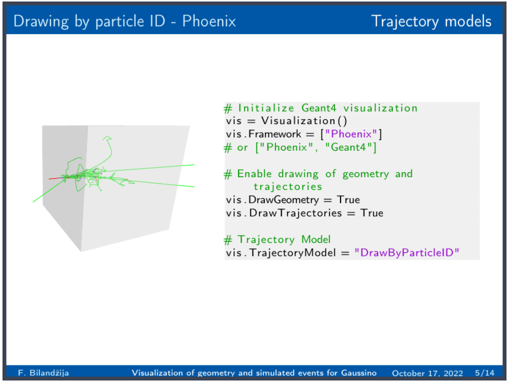
F. Bilandžija CERN-STUDENTS-Note-2022-205



Drawing by particle ID - Geant4 Trajectory models

```
# Initialize Geant4 visualization
vis = Visualization()
vis.Framework = ["Geant4"]
vis.Driver = "DAWNFILE"

# Enable drawing of geometry and
    trajectories
vis.DrawGeometry = True
vis.DrawTrajectories = True

# Trajectory Model
vis.TrajectoryModel = "DrawByParticleID"
```

F. Bilandžija Visualization of geometry and simulated events for Gaussino October 17, 2022 4/14



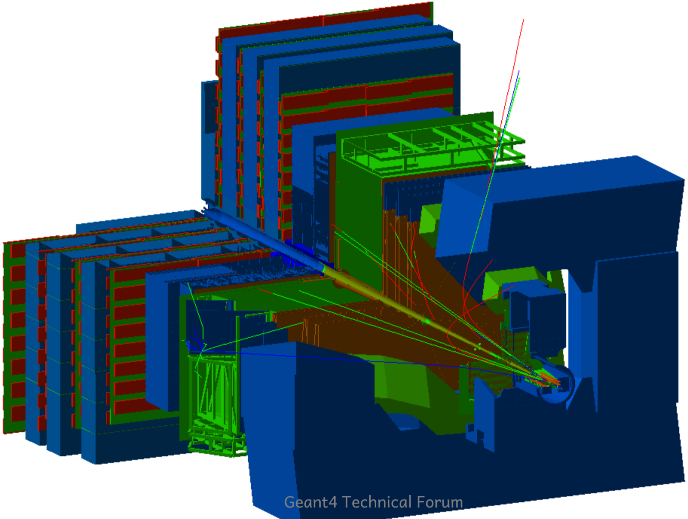
Drawing by particle ID - Phoenix Trajectory models

```
# Initialize Geant4 visualization
vis = Visualization()
vis.Framework = ["Phoenix"]
# or ["Phoenix", "Geant4"]

# Enable drawing of geometry and
    trajectories
vis.DrawGeometry = True
vis.DrawTrajectories = True

# Trajectory Model
vis.TrajectoryModel = "DrawByParticleID"
```

F. Bilandžija Visualization of geometry and simulated events for Gaussino October 17, 2022 5/14

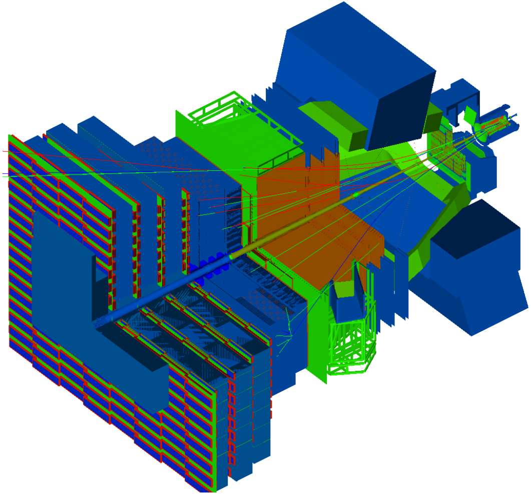
Visualization in Gauss-on-Gaussino (Geant4)

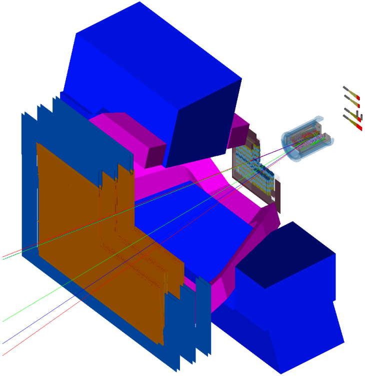
F. Bilandžija, M. Mazurek



Legacy vs. DD4hep (as of 08/22)

F. Bilandžija, M. Mazurek

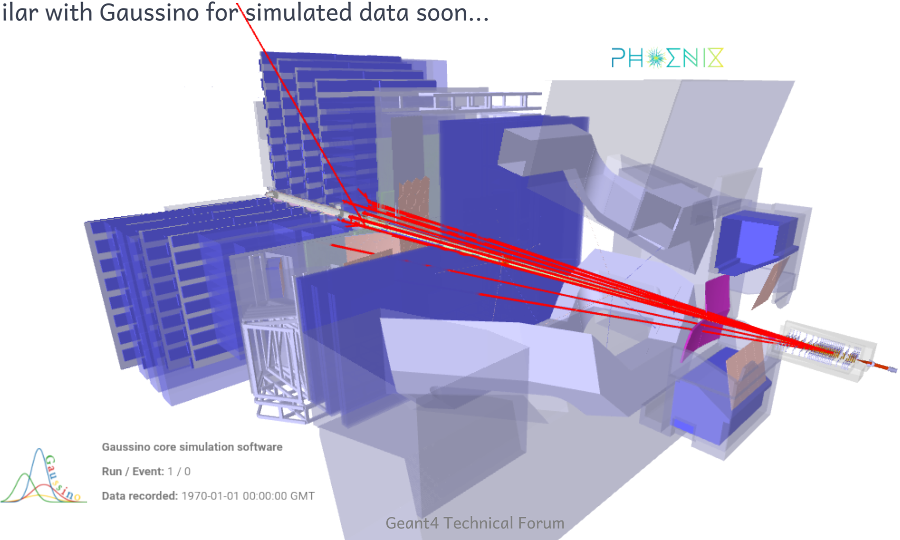




Visualization in Gauss-on-Gaussino (Phoenix)

F. Bilandžija, M. Mazurek

- current LHCb event display with reconstructed data: https://lhcb-eventdisplay.web.cern.ch/
- similar with Gaussino for simulated data soon...



Gaussino core simulation software

Run / Event: 1 / 0

Data recorded: 1970-01-01 00:00:00 GMT

Simulation in LHCb

Custom physics and fast simulations

👉 supporting the infrastructure for **fast simulation models** and **custom physics of future detectors**,

👉 helping in integrating **machine learning** libraries,

Supporting fast simulation developments

G. Corti, M. Mazurek, et al.

👉 work in parallel to current developments for Run 3

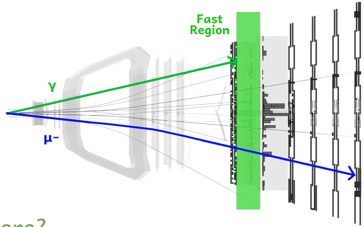
🔥 fast simulations to kick in Gauss-on-Gaussino seamlessly in 2023!

🔥 all but for G4 with ML already in Gauss Sim10!

Model	Generation	Decay	Propagation	Status in G-on-G
ReDecay	✓	✓	✓	done
ParticleGun	✓	✓	✓	done
SplitSim	✓	✗	✓	done
RICHless	✗	✗	✓	under tests
TrackerOnly	✗	✗	✓	under tests
Lamarr	✗	✗	✓	(**NEW**) in progress
Point library	✗	✗	✓	(**NEW**) in progress
GANs	✗	✗	✓	(**NEW**) in progress

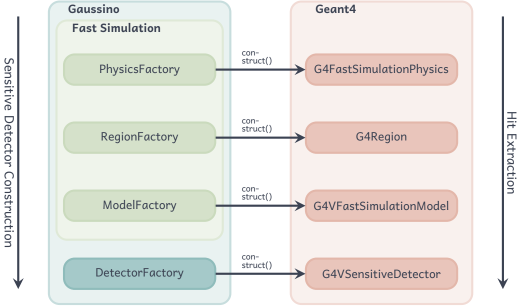
- ➔ **Interfacing fast simulations**
- ➔ **Interfacing ML libraries**
- ➔ **Monitoring infrastructure**
- ➔ **Production of training datasets**

Interfacing ~~fast simulations~~ *custom physics* with G4



1. Where?
   - ✓ region where the fast simulation takes place
2. What?
   - ✓ what types of particles should be tracked
3. How?
   - ✓ conditions when to fast simulate,
   - ✓ fast hit generation algorithm,

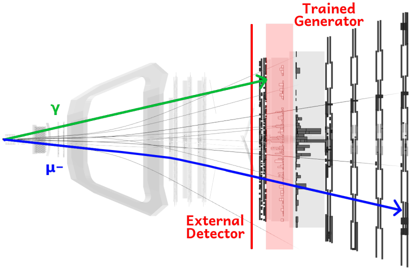
➔ high-level configuration available!



Interfacing ML libraries

G. Corti, M. Mazurek et al.

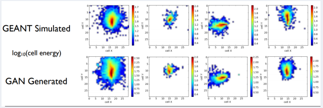
- development of Gaussino’s interface to ML libraries:
  - libtorch (PyTorch C++ API),
  - ONNX etc.
- collaboration with **CERN SFT / ML4Sim**,



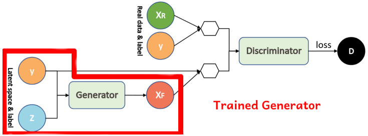
Example use case

Generative Adversarial Networks (GANs)

Idea: use GANs trained on the data produced by a detailed simulation to generate showers in ECAL

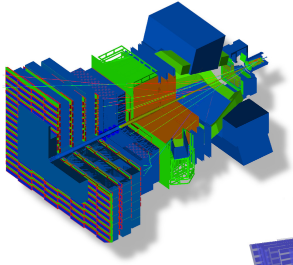


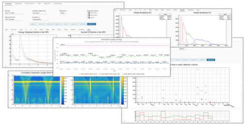
- use the trained generator part for the inference of a fast simulation model

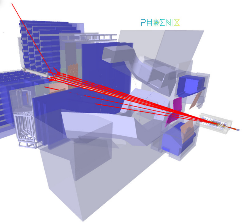


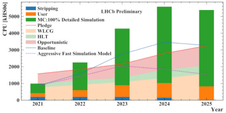
Conclusions

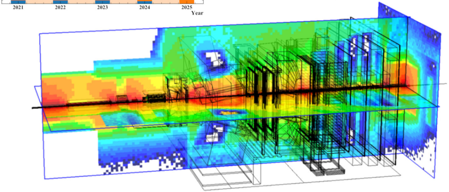
- Run3 detector is different from that of Run1&2,
- Simulation software stack is evolving,
- revised recent Geant4 developments in LHCb for Sim10 and the future:
  - timing optimization,
  - handling unknown particles,
  - new visualizations techniques,
  - custom physics and fast simulations,
  - and many more!











Thank you!