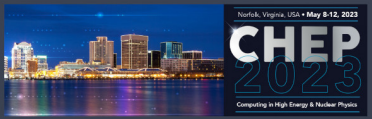


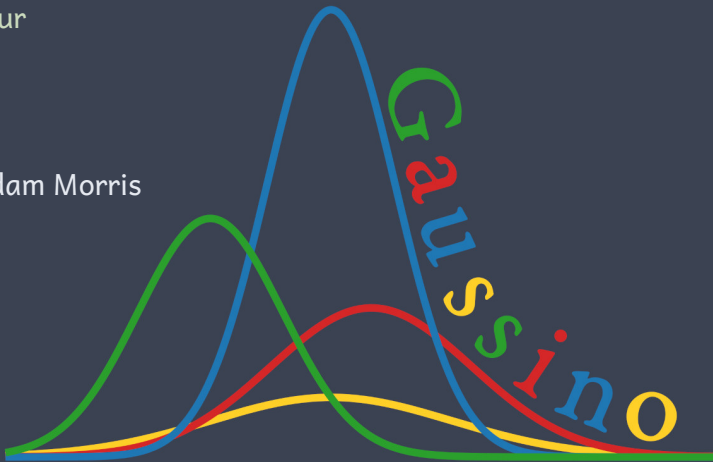
CERN

From prototypes to large scale detectors

how to exploit the **Gaussino** simulation framework
for detectors studies, with a detour
into machine learning

by **Michał Mazurek**
Gloria Corti, Marco Clemencic, Adam Morris

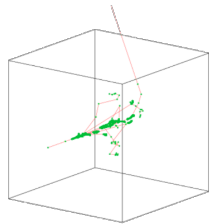
on 11 May, 2023
CHEP 2023, Norfolk VA, USA

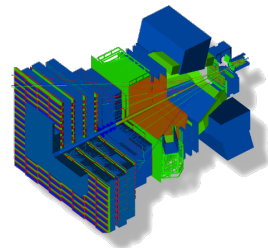


Simulations for detector R&D studies

👶 **test beams** / **early studies** with sub-detector prototypes in **Geant4 standalone**,

🧔 porting to **the experiment's main simulation framework** once tested



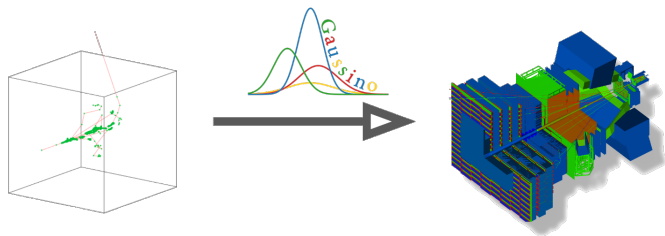


‼️ early simulations already very complex: optimizations, custom simulations, machine learning, etc.
😓 the path to the final detector is **not easy**: multithreading, complex geometries, etc.

Universal framework for simulations

🎯 use the **Gaussino** framework **for both**: test beams and final integration in the experimental setup!

- new core simulation framework to be used as a library or in a standalone mode,
- created by extracting experiment-independent components from Gauss,
- 👉 **more on Gauss-on-Gaussino for LHCb** in the previous talk!,
- in collaboration with the CERN SFT group / FCC,
- 👉 **more on Gaussino in Key4hep** in A. Salier's talk!,

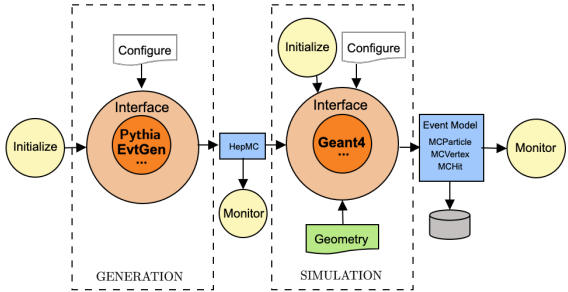


👉 **this talk** shows what **can already be done**!

Gaussino: keep what's good and works well…

LHCB-TDR-017 & LHCB-TDR-011

i.e. the complete simulation framework architecture
well-served in the LHCb experiment



- similar modularity,
- integrated generation and simulation phase,
- similar event model,
- Gaudi tools, algorithms etc.
- python configuration,

… and support new developments and ideas!

LHCB-TDR-017

Key concepts

- possibility to run in a standalone mode,
- higher-level configuration in python,
- multi-threaded event loop in Gaudi,
- multi-threaded Geant4,
- interfaces to generic detector description tools (DD4Hep),
- interface to fast simulations in Geant4,

Execution structure

- use Gaudi functional,
- every algorithm as a 'task',

Random numbers

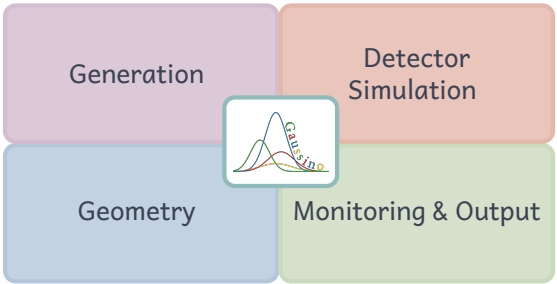
- ensure reproducibility,
- seed initialized with:
  - run #,
  - event #,
  - algorithm instance name,
- create random engines on the stack,

In the pipeline

- interface to ultra-fast simulations (Lamarr, more in M. Barbetti's talk) ,
- interface to generic event models (EDM4hep),
- interfaces to machine learning libraries,

Configuration

👉 python configurables steering C++ classes,
👉 modular structure with 4 main configurables, one for each module

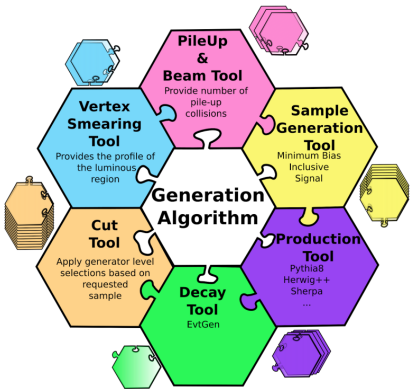


- Gaussino()
- GaussinoGeneration()
- GaussinoSimulation()
- GaussinoGeometry()
- \+ optional (ParticleGun, etc.)

Generation phase

LHCb-PROC-2010-056 & LHCB-FIGURE-2023-010

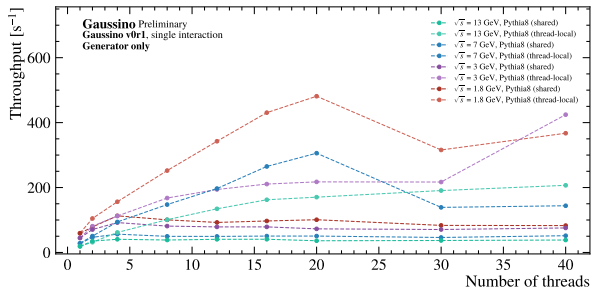
- highly modular as in Gauss,
- thread safety of generators,
- HepMC3 as an exchange format,



- shared and thread-local interface to Pythia8,
- various particle guns,

Performance

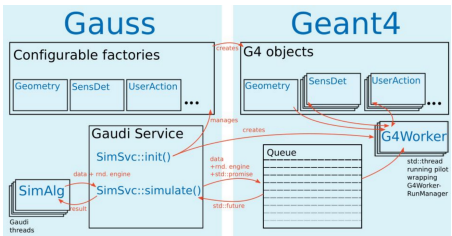
- throughput of the shared vs. thread-local Pythia8 interface for various beam energies,



Detector simulation

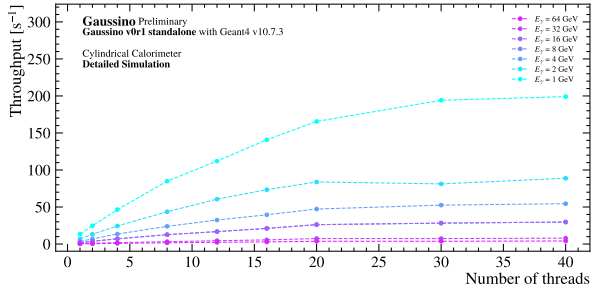
LHCB-FIGURE-2023-010

- Geant4 with multi-threading,
- Gaudi tools as factories for G4 objects,
- flexible python configuration:
  - in-time / out-of-time pileup,
  - signal / other particles,
  - fast simulations,

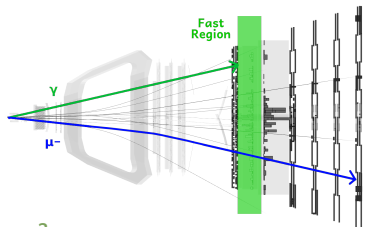


Performance

- throughput tests of the generic cylindrical calorimeter simulation,

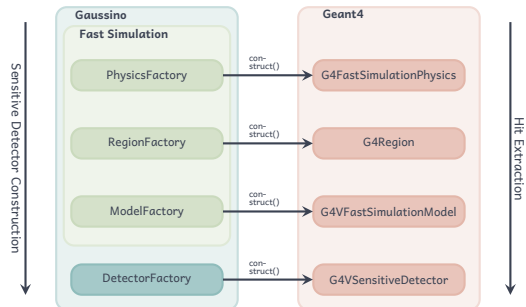


Interfacing ~~fast simulations~~ *custom physics* with G4



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,

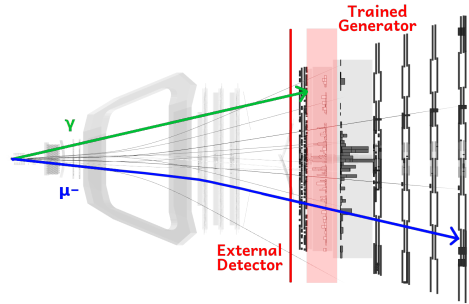
- high-level configuration available!



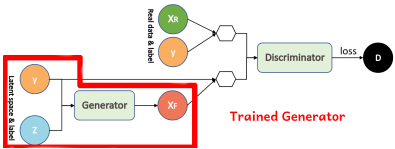
Machine learning in fast simulations

LHCB-FIGURE-2021-004

- (Example) Generative Adversarial Networks
  Idea: use GANs trained on the data produced by a detailed simulation to generate showers in ECAL



- use the trained generator of the network as a fast simulation model



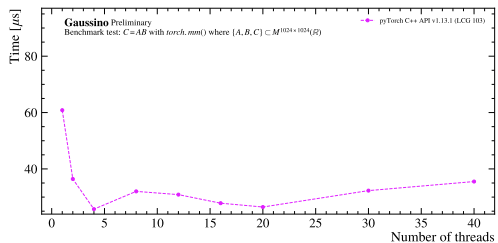
- interface to machine learning libraries needed!

Generic ML interface

LHCB-FIGURE-2023-010

- **ML model serving** using Gaudi's services,
  - handles loading of the model,
  - model accessible throughout the whole execution,
  - setting general properties:
    - threads no. in inter-op parallelism,
    - threads no. in intra-op parallelism,
- **ML model evaluation** using Gaudi's tools and algorithms,
  - pass the random generator seed to guarantee reproducibility,
  - fixed or automatic types for inputs & outputs,

- first candidate: **pyTorch C++ API**,
- preliminary throughput tests:



ML inference in fast simulations

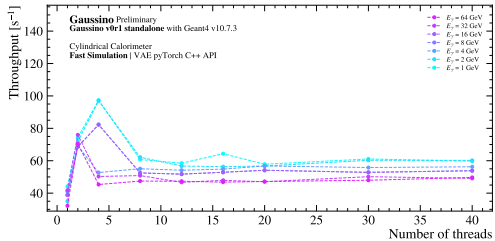
Geant4/ML4SIM & LHCB-FIGURE-2023-010

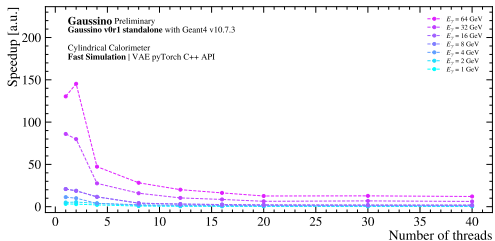
2 possibilities for ML inference

1. internally in Geant4,
2. in a Gaudi algorithm after the simulation algorithm takes place,

First benchmarks

- in a close collaboration with **Geant4/ML4Sim**,
- compatible with the **CaloChallenge** setup,
- generic cylindrical calorimeter setup,
- variational autoencoder for photons,
- Gaussino's interface to pyTorch C++ API,





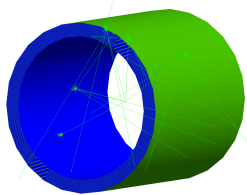
Geometry

LHCB-FIGURE-2021-004

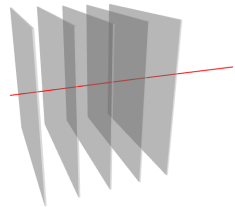
- **experiment-independent** services
- **DD4Hep** in **standalone mode**,
- import & export of **GDML** files,
- **internal** geometry service for simple volumes:
  - available in **standalone mode**,
  - can be **mixed** with other services,
  - possible **parallel worlds**,
  - used for **fast simulation hooks**,

💡 **Various setups possible!**

👉 **general-purpose detectors**



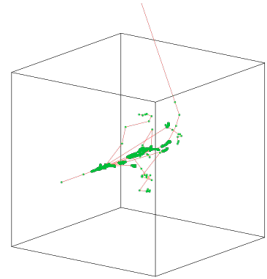
👉 **forward spectrometers,**

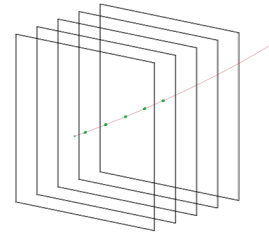


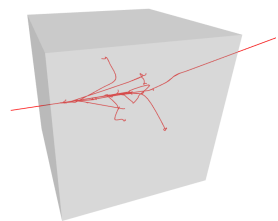
Visualization in Gaussino

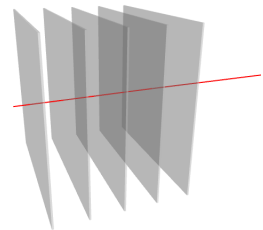
CERN-STUDENTS-Note-2022-205

- **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 experiment-specific** simulated data,
  - data exported to **JSON**,
  - **simulated vs. reconstructed** data comparison possible,





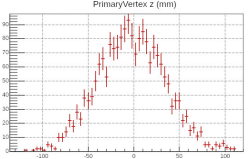


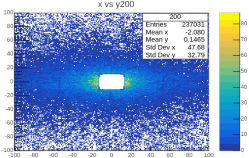


Monitoring & Output

D.Popov EPJ Web Conf., 214 (2019) 02043
M. Szymański, B. Couturier EPJ Web Conf., 214 (2019) 05014
LHCB-FIGURE-2021-004

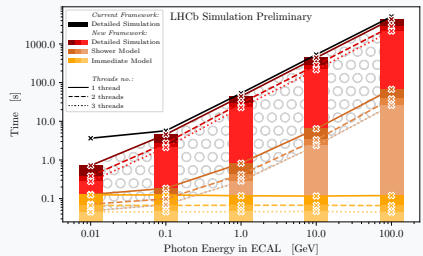
- various, persistent output formats possible with predefined contents
  - built-in event model,
  - consistent MCTruth:
    - from generators,
    - from Geant4 choosing what to keep,
  - histograms,
  - custom n-tuples,
- the output and performance of the simulation is monitored with an automatic tool from LHCb: LHCbPR





Example: timing in fast simulation

- Comparison with benchmark models:
  - ImmediateDeposit gives the timing needed for the infrastructure itself to call the fast simulation
  - ShowerDeposit provides the minimum amount of time needed to generate a specific number of hits with no additional calculations

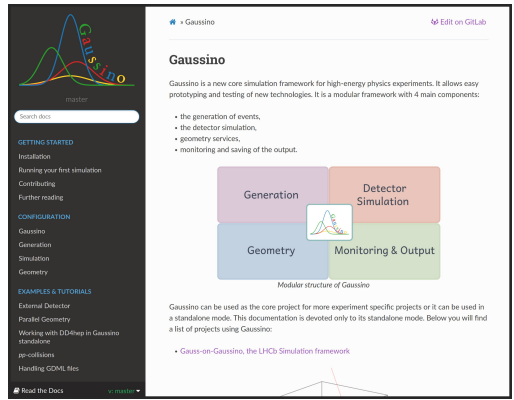


Documentation

https://gitlab.cern.ch/Gaussino/Gaussino

https://gaussino.docs.cern.ch/

- each new development in Gaussino is documented
- the documentation provides the description of:
  - how to install and run simulations
  - high-level python configuration
  - simple examples
- versioning of the documentation



Conclusions

- 👉 Gaussino is the new core simulation framework extracted from the LHCb simulation framework.
- 👉 Can be used for as the **core simulation toolkit** for detector frameworks in HEP…
- 👉 … **or** it can be used in a **standalone mode** as an ideal test bed for early detector studies:
  - 👉 easy configuration in python,
  - 👉 integrated generation and detector simulation phase,
  - 👉 internal geometry service,
  - 👉 support for fast simulation, machine learning, etc.
- 🎉 Gaussino is mature enough for its first beta release!

Thank you!