NATIONAL CENTRE FOR NUCLEAR RESEARCH

DOCTORAL THESIS

New simulation software and machine learning technologies in the LHCb experiment to evaluate physics performance of Run 3

Author:
Michał MAZUREK

Supervisor:
Wojciech WIŚLICKI
Auxiliary supervisor:
Wojciech KRZEMIEŃ

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Warsaw LHCb Group
High Energy Physics Division
Department of Fundamental Research

September 12, 2024

Declaration of Authorship

I, Michał MAZUREK, declare that this thesis titled, "New simulation software and machine learning technologies in the LHCb experiment to evaluate physics performance of Run 3" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the National Centre for Nuclear Research.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at the National Centre for Nuclear Research or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

NATIONAL CENTRE FOR NUCLEAR RESEARCH

Abstract

New simulation software and machine learning technologies in the LHCb experiment to evaluate physics performance of Run 3

Michał MAZUREK

The main goal of the work presented in this thesis was to explore new software and machine learning-based technologies to improve the performance of the LHCb data processing applications and, in particular, the simulation framework of LHCb, GAUSS.

GAUSSINO is the new core simulation framework that was created by extracting all the experiment-independent functionalities of GAUSS. In this work, the GAUSSINO framework was moved from the advanced prototype stage to the production-ready framework, which can act as an ideal test bed for all the new simulation and detector developments in a standalone mode, as well as a robust core simulation framework for experiments in high-energy physics (HEP). GAUSS-ON-GAUSSINO is the new version of the LHCb simulation framework, based on GAUSSINO. In addition to ensuring the smooth transition to the new simulation framework for Run 3 and beyond, the work included integrating a new interface to fast simulations, adding support for new detector description toolkits (DD4HEP and EXTERNALDETECTOR), as well as new visualization tools and web-based documentation.

In addition to the improvements in the software of the simulation framework, new machine learning-based technologies were explored and integrated into the simulation framework. In particular, infrastructure for running Generative AI (GenAI) models for calorimeter fast simulations in GAUSSINO was integrated. Moreover, the performance of the first, production-ready CALOML+VAE model trained on the LHCb electromagnetic calorimeter data was evaluated. An exploration of the additional use of the machine learning (ML) models in object detection algorithms for cluster energy reconstruction in the LHCb electromagnetic calorimeter was also presented.

Finally, validation of the new simulation framework and machine learning-based fast simulation techniques was presented using a few relevant LHCb decay channels in the last chapter of this thesis. Validation was done with respect to samples produced with GAUSS framework when using the well-tested GEANT4 toolkit.

NATIONAL CENTRE FOR NUCLEAR RESEARCH

Streszczenie

New simulation software and machine learning technologies in the LHCb experiment to evaluate physics performance of Run 3

Michał MAZUREK

Głównym celem przedstawionej rozprawy było zbadanie nowych technologii oprogramowania oraz metod opartych na uczeniu maszynowym w celu poprawy wydajności przetwarzania danych w LHCb, ze szczególnym naciskiem na oprogramowanie do symulacji w LHCb (GAUSS).

GAUSSINO jest nowym oprogramowaniem symulacyjnym, które zostało stworzone poprzez wyodrębnienie wszystkich komponentów niezależnych od eksperymentu. W tej pracy, GAUSSINO zostało udoskonalone i udostępnione jako gotowe narzędzie w środowisku produkcyjnym, zarówno jako idealne środowisko testowe dla nowych symulacji i rozwoju detektorów oraz jako solidny framework symulacyjny dla eksperymentów w fizyce wysokich energii. GAUSS-ON-GAUSSINO to nowa wersja oprogramowania symulacyjnego LHCb oparta na GAUSSINO. Płynne przejście na nowe, wydajniejsze oprogramowanie symulacyjne w Runie 3 było jednym z celów pracy. Dodatkowo praca obejmuje dodanie nowego interfejsu do szybkich symulacji, wsparcie dla nowych narzędzi do opisu detektorów (DD4HEP i EXTERNALDETECTOR) oraz nowe narzędzia wizualizacyjne i dokumentację.

Nowe technologie symulacyjne oparte na uczeniu maszynowym (ML) zostały również zbadane w tej pracy. W szczególności zbudowana została infrastruktura niezbędna do uruchamiania modeli generatywnej sztucznej inteligencji (GenAI) dla szybkiej symulacji w kalorymetrze w GAUSSINO, a na jej podstawie dodano i przetestowano pierwszy model CALOML+VAE wytrenowany na danych z kalorymetru elektromagnetycznego LHCb. Dodatkowo zbadano możliwość wykorzystania modeli ML w algorytmach detekcji obiektów do rekonstrukcji energii klastrów w kalorymetrze elektromagnetycznym LHCb.

Na koniec przedstawiono wyniki symulacji fizycznych wybranych kanałów rozpadu istotnych dla eksperymentu LHCb w celu walidacji szybkich symulacji opartych na uczeniu maszynowym z wykorzystaniem modelu CALOML+VAE. Walidacja została przeprowadzona przy użyciu próbek wyprodukowanych za pomocą oprogramowania GAUSS.

Acknowledgements

I would like to express my very great appreciation to my supervisors at the National Centre for Nuclear Research, Prof. Wojciech Wiślicki and Dr. Wojciech Krzemień, for their professional guidance and support throughout my doctoral research.

I would also like to extend my sincere *grazie mille* to Dr. Gloria Corti, my supervisor during my Doctoral Studentship at CERN. I am truly grateful for the time and effort she invested in explaining every detail with such clarity. The opportunity to spend three years at CERN has been an enriching experience.

I also wish to thank all my colleagues in the LHCb collaboration for their assistance and encouragement.

Finally, I am deeply thankful to my family for their constant support, especially to my beloved Klaudusia, whose encouragement has been key in helping me through difficult decisions.

Contents

List of Figures

List of Tables

Kochanej Klaudusi

Introduction

Physicists working in the field of high-energy physics (HEP) aim to understand the fundamental particles and forces that constitute the Universe. Large HEP experiments are designed to recreate conditions similar to those just after the Big Bang, providing insights into the origins and evolution of the Universe.

The Large Hadron Collider (LHC) at CERN is the world's largest and most powerful particle accelerator. It was built to probe the fundamental structure of matter by colliding protons and heavy ions at unprecedented energies. The primary goal of the experiments at the LHC is to examine the validity and limitations of the Standard Model of particle physics and to search for New Physics (NP) beyond it, such as dark matter and supersymmetry.

Experiments at LHC have achieved remarkable successes, including the discovery of the Higgs boson, pentaquarks, and many more. They rely heavily on simulations to interpret experimental data, optimize detector design, and test theoretical models. The traditional approach to these simulations involves Monte Carlo (MC) event generators and detailed particle propagation and interaction with the material of the detector using toolkits like GEANT4. Although very effective, these methods are computationally expensive, especially given the increasing luminosity and complexity of the experiments. The role of MC simulations in HEP experiments is explained in detail in Chapter 1.

One of the large experiments at LHC is the one being carried out by the LHCb Collaboration. It is a single-arm spectrometer designed to study the properties of particles containing beauty (b) or charm (c) quarks. Originally designed to make precision studies of *CP* violation and very rare decays in *B*-meson systems by exploiting proton-proton collisions at the LHC as the most copious source of *b*-hadrons in the world, the LHCb experiment's physics programme has been extended to include a wide range of measurements in the field of heavy flavour physics and beyond. The results gathered by LHCb so far have demonstrated that the Standard Model effectively describes phenomena up to an energy scale of 1-10 TeV. The LHCb experiment and its current status (during Run 3 of data taking at the LHC) is described in more detail in Chapter 2.

The LHCb experiment is facing one of the most challenging trigger rates among the LHC experiments with the detector and data acquisition system that have been upgraded. Its simulations are critical for these developments and since Run 2 of data taking take 90% of all the allocated computing resources. The main goal of the work presented in this thesis was to explore new software and machine learning-based technologies in order to improve the performance of the LHCb data processing applications, and in particular, the simulation framework of LHCb named GAUSS.

In view of the increasing demands at LHCb, the simulation framework, GAUSS, was redesigned to be more efficient and flexible, allowing for the integration of new simulation techniques and technologies. The core simulation framework, GAUSSINO, was developed by extracting the core functionalities of GAUSS, and making them available as a standalone library. GAUSSINO and the new version of GAUSS, also called GAUSS-ON-GAUSSINO, is described in more detail in Chapter 3.

Moving the GAUSSINO and GAUSS-ON-GAUSSINO framework from an advanced prototype stage to a production-ready framework was one of the main tasks of this thesis. Moreover, the work included integrating a new interface to fast simulations, adding support for new detector description toolkits (DD4HEP and EXTERNALDETECTOR), and developing new visualization tools along with web-based documentation.

Despite improvements in the simulation framework, the computational cost of simulating the detector response to particle interactions remains a significant challenge. To address it, modern simulation techniques based on generative artificial intelligence (GenAI), have emerged as a promising solution. Generative models can learn the underlying patterns and correlations in the data, enabling them to generate realistic simulations with significantly reduced computational costs. Integrating machine learning-based (ML) models into the simulation framework of experiments can be complex and time-consuming, requiring adaptation to the specific environment of the simulation framework. The GAUSSINO framework, with its modular design, is ideal for exploring generic GenAI models. GAUSSINO can be used in a standalone mode for rapid prototyping and testing of new models, as well as running them in various experiment configurations. Once trained and tested on experiment-agnostic datasets, the models can be easily adapted to the specific requirements of the experiment's simulation framework. Chapter 4 presents a few applications that were explored in this thesis. The interface to the ML libraries for running inference directly in the simulation framework is presented in Section 4.1. Integration of the infrastructure necessary to run GenAI models for calorimeter fast simulations in GAUSSINO, as well as the performance of the first production-ready CALOML+VAE model trained on the LHCb electromagnetic calorimeter data is presented in Section 4.2. Brief exploration of the additional use of the ML models in object detection algorithms for cluster energy reconstruction in the LHCb electromagnetic calorimeter is presented in Section 4.3.

The last chapter of this thesis, Chapter 5, presents the results of the simulation of a few relevant LHCb decay channels to validate the methods and tools developed in the previous chapters. In particular, a detailed physics performance of the first, production-ready GenAI model for the electromagnetic calorimeter simulation is presented. Validation was done with respect to samples produced with GAUSS framework when using the detailed and comprehensively tested GEANT4 toolkit, before reconstructed data from Run 3 were available.

Additional performance plots of the GAUSSINO and GAUSS-ON-GAUSSINO simulation frameworks were placed in Appendix A. Selected sub-detector visualizations used during the validation process of the new detector description in GAUSS-ON-GAUSSINO were put in Appendix B. Figures representing performance of the interface to ML backends in GAUSSINO were added in Appendix C. Finally, additional plots related to the ML-based fast simulation in GAUSSINO and GAUSS-ON-GAUSSINO were placed in Appendix D.

1

Simulations in experimental high energy physics

High energy physics (HEP), also known as particle physics, or physics of particles and fields, is a leading field of research that stands at the forefront of unraveling the fundamental building blocks of our Universe. It aims to understand the nature of matter, the forces that govern it, and the laws that dictate its behavior. In this chapter, a very brief introduction to high-energy physics will be given. Moreover, experimental techniques in high energy physics will be discussed, and in particular, the role of Monte Carlo simulations.

1.1 Particles at high energies

In everyday life, humans interact with objects of various shapes and sizes, often comparable to their own scale. However, at much smaller scales, matter consists of very tiny chunks and vast empty spaces in between [1]. These tiny chunks, also called *elementary particles*, come in a small number of types, and it is the way they are arranged that gives rise to the rich variety of complex entities we see around us.

Classical mechanics has been used for centuries to describe the motion of macroscopic objects. However, it is not sufficient to describe the motion of particles at very small scales, and at very high speeds. In some specific conditions, the classical rules of mechanics break down, and other types of mechanics are needed to describe the motion of particles. Figure 1.1 shows all the known realms of mechanics and theory of interactions represented by the so-called Bronstein-Zelmanov-Okun cube. Each axis of the cube represents a particular condition, which when starts dominating, changes the type of mechanics. For example, when objects travel at speeds comparable to the speed of light, special relativity is needed to describe their motion. If in addition to that, the objects are also very heavy, then *general relativity* is needed to describe their motion.

Elementary particles are very small, and when they travel at very high speeds, they are best described by *quantum field theory* [2, 3], a union of *quantum mechanics* and *special relativity*. A collection of theories describing elementary particle interactions except gravity, but including strong and electroweak interactions, is called the *Standard Model* of particle physics. All elementary particles are briefly described in Section 1.1.1, and the Standard Model is discussed in Section 1.1.2.

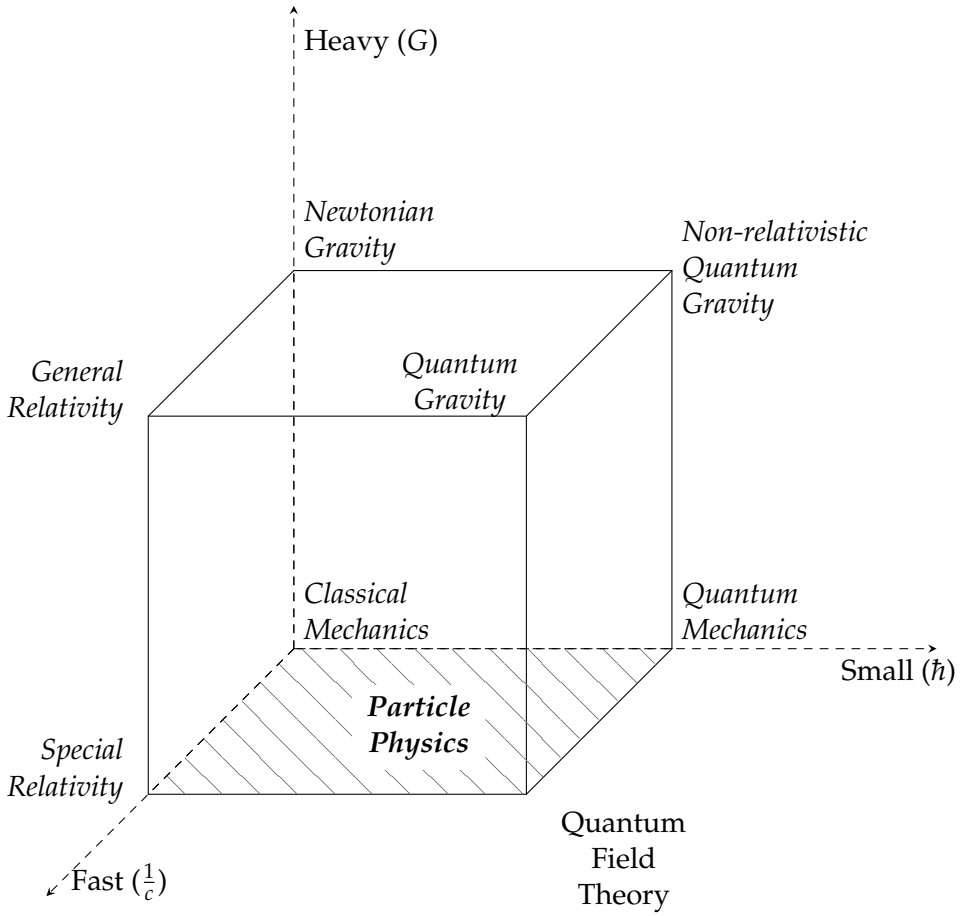


FIGURE 1.1: Known realms of mechanics represented by the Bronstein-Zelmanov-Okun cube. Each corner of the cube represents a different type of mechanics depending on the conditions.

1.1.1 Elementary particles

Our visible Universe is made of *fermions*, half-odd-integer spin ($\frac{1}{2}, \frac{3}{2}, ...$) particles that obey the Pauli exclusion principle. They can be then divided into two groups: *quarks* and *leptons*, depending on whether they interact via strong interaction or not. There are 3 generations of quarks and 3 generations of leptons:

- 1st generations:
  - up (u) and down (d) quarks,
  - electron (e) and electron neutrino (ν_e);
- 2nd generations:
  - charm (c) and strange (s) quarks,
  - muon (μ) and muon neutrino (ν_μ);
- 3rd generations:
  - top (t) and bottom or beauty (b) quarks,
  - tau (τ) and tau neutrino (ν_τ).

Each fermion has a corresponding antiparticle, which has the same mass, but opposite all other additive quantum numbers, e.g. electric charge, strangeness, baryonic numbers, etc.

Gauge bosons are the force carriers of the interactions in the Standard Model. They carry integer spins and mediate the interactions between other particles. There are in total 12 gauge bosons [4, 5] in the Standard Model:

- 8 gluons (g) for the strong interaction,
- 1 photon (γ) for the electromagnetic interaction,
- 3 weak bosons ($W^{\pm}$, Z^0) for the weak interaction.

There is also one *scalar boson*, the *Higgs boson* [6–10], which is produced by the excitation of the Brout-Englert-Higgs field that is responsible for particle masses.

1.1.2 Standard Model

The Standard Model (SM) [1] of particle physics is a mathematical framework that unifies electromagnetic, strong and weak interactions. Gravity is not included in the scope of this theory, and in any case, it is too weak to play any significant role in processes measurable by present-day HEP experiments. In the SM [11], all fundamental interactions derive from *local gauge invariance* of the symmetry group

$$SU(3) \times SU(2) \times U(1) \tag{1.1}$$

where:

- $SU(3)$ is the symmetry group of the strong interaction, formed in theory known as *quantum chromodynamics* (QCD),
- $SU(2) \times U(1)$ is the symmetry group of the electroweak interaction;

in which the Lagrangian does not change under transformations belonging to these groups. Thus, the lagrangian density is given by

$$\mathcal{L} = \mathcal{L}_{\text{QCD}} + \mathcal{L}_{\text{EW}} + \mathcal{L}_{\text{Higgs}} \tag{1.2}$$

where:

- $\mathcal{L}_{\text{QCD}}$ is the lagrangian density of quantum chromodynamics,
- $\mathcal{L}_{\text{EW}}$ is the lagrangian density of the electroweak interaction,
- $\mathcal{L}_{\text{Higgs}}$ is the lagrangian density of the Higgs mechanism.

Despite the fact that the SM is a very successful theory, it is not complete and it does not explain all the phenomena that physicists observe [12]. As mentioned before, it does not include gravity, nor does it explain the existence of dark matter and dark energy, which are believed to constitute the majority of the Universe. Moreover, it does not explain why neutrinos have mass and why they oscillate from one type to another. Neither does it explain why there is more matter than antimatter in the Universe. Beyond Standard Model (BSM) theories, is a term used to describe theories that attempt to explain these phenomena.

1.2 Experiments in high energy physics

In Figure 1.2, a schematic diagram of a typical experimental workflow in particle physics is presented. The absolute minimum of any experimental setup is a *source of particles* of interest, an *experimental apparatus* that allows to detect their interactions, and a *readout system* that allows to record and store the outcomes for further analysis.

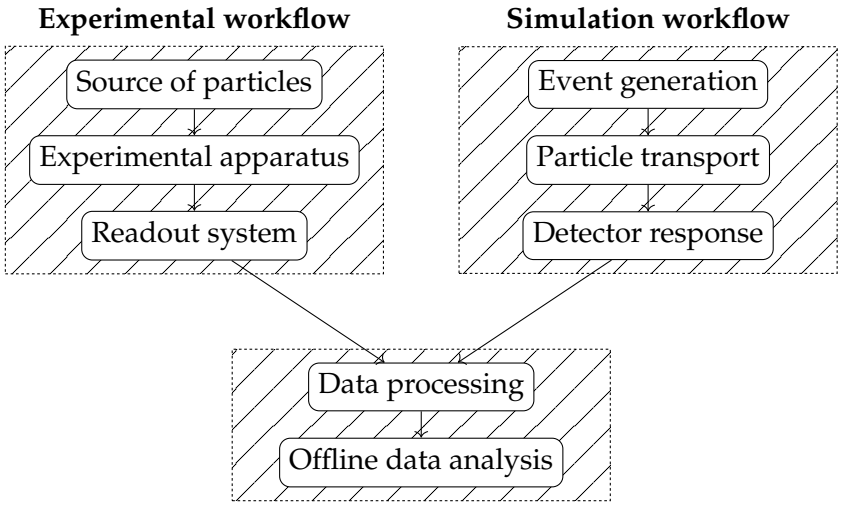


FIGURE 1.2: The standard workflow of a high energy physics experiment. It represents two possible ways of obtaining data: from a real, physical experiment (left) or from a simulation (right).

1.2.1 Particle accelerators

Particle accelerators can be used to accelerate particles to high energies and then collide them with each other or with a fixed target. The largest and the highest-energy accelerator ever built is the Large Hadron Collider (LHC) [13, 14] at CERN at the Franco-Swiss border near Geneva in Switzerland. It was designed as a proton-proton (pp) collider with a capacity to run collisions with center-of-mass energy up to $\sqrt{s} = 14$ TeV. In Figure 1.3, the evolution of high energy particle colliders is shown as a function of time. The plan to build machines able to provide even higher energies and luminosities is already in place. High-Luminosity Large Hadron Collider (HL-LHC or HiLumi LHC) [15] is a project to upgrade the LHC to increase its number of collisions by a factor of 5 to 7.5 with respect to the nominal LHC design. Following the HL-LHC, an ambitious plan to build a completely new accelerator, the Future Circular Collider (FCC), is being developed. First, the FCC would operate as a lepton (e^+e^-) collider (FCC-ee) [16] with a maximal center-of-mass energy of 365 GeV, and then as a hadron (pp) collider (FCC-hh) [17] with a maximal center-of-mass energy of 100 TeV. Projects to build linear colliders, such as the International Linear Collider (ILC) [18] or Compact Linear Collider (CLIC) [19], are also being considered.

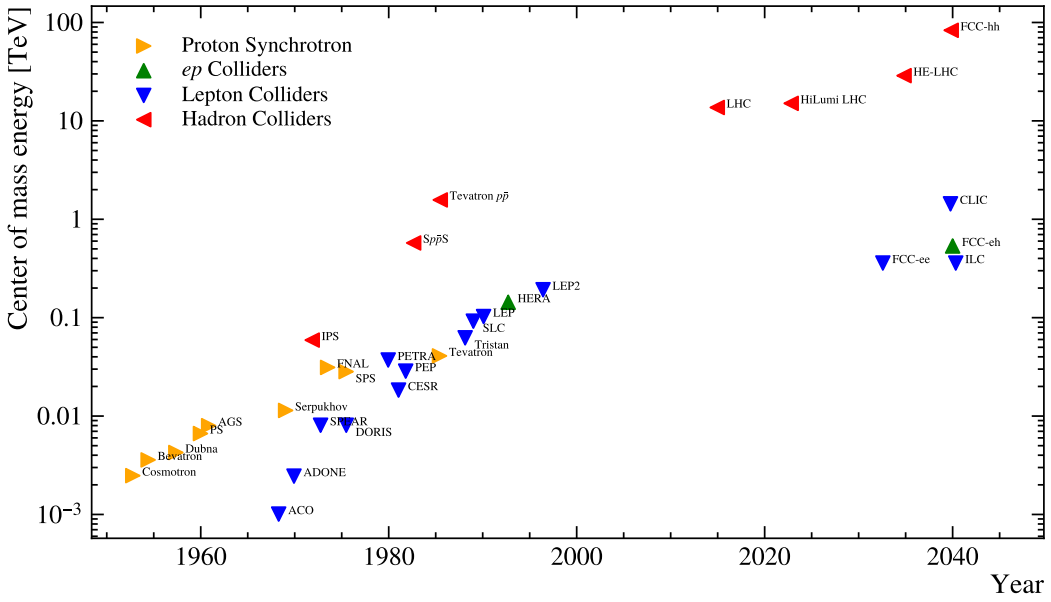


FIGURE 1.3: The evolution of particle colliders [20]. Particle energy in the center of mass is shown as a function of time.

1.2.2 Experimental apparatus

Once the particles are produced, they have to be detected by the experimental apparatus. Modern detectors [21] consist of a series of various detectors, each designed to measure a specific property of particles. Combined together, they allow to build a full picture of how particles interacted. In particular, they allow to reconstruct the trajectories of particles, identify them, and measure their charge and energy. Tracking detectors are usually in a few places in the experiment, and are designed to record the trajectory of particles as they pass through the detector. Curvature of tracks in a magnetic field enables determination of particles' momenta. A very special type of tracking detectors are silicon tracking detectors, in which charged particles traverse the silicon layers and generate electron-hole pairs. Vertex detectors are a specific subtype of silicon detectors, placed very close to the *interaction point* (IP) — a point in space where particles collide and produce secondary particles. Gaseous detectors are also used as tracking detectors in HEP experiments. In these detectors traversing particles ionize the medium of the detector, causing electrons and ions to drift in the electric field towards the electrodes, where the detection takes place. An example of such a detector is the Time Projection Chamber (TPC), particularly common in low-luminosity experiments. Calorimeters absorb incident particles in order to measure their energy lost in absorbing material and released as an electromagnetic or hadronic cascade. If the particle travels faster than the phase velocity of light in a medium with high refractive index, it produces a shock wave of light known as Cherenkov radiation, its apex forming a cone. In Ring-Imaging Cherenkov (RICH) detectors, the light cone is reflected by mirrors and focused into rings that trigger a photomultiplier tube. The Cherenkov light is emitted with an angle proportional to the phase velocity thus enabling determination of particle's velocity. Therefore, by using information on the particles velocity, Cherenkov detectors can be used to identify particles, such as electrons, muons, and pions. The only particles left in the outermost layers of the

detector are muons which interact with matter with a small cross section, and which are detected by massive muon detectors, usually located farther from the IP.

1.2.3 Readout system

Detectors produce electric signals that are then recorded by front-end electronics. Abundance of particles produced in high-energy collisions, as well as high collision rates, require the data acquisition systems to be very fast and selective. *Trigger systems* [22] are usually multi-layer systems that are designed to select only the events that are of interest to physicists and, in addition, meeting the storage and CPU limitations of the experiments. Figure 1.4 shows how the trigger rate has been changing over the years for various experiments. Once filtered, the data is then passed on to the *reconstruction software*. The aim of the reconstruction software is to convert the low-level electronic signals, called the raw data, into the high-level physics objects, such as tracks, vertices, and particles. Only then the data, once persisted in storage, is ready to be analyzed offline by physicists.

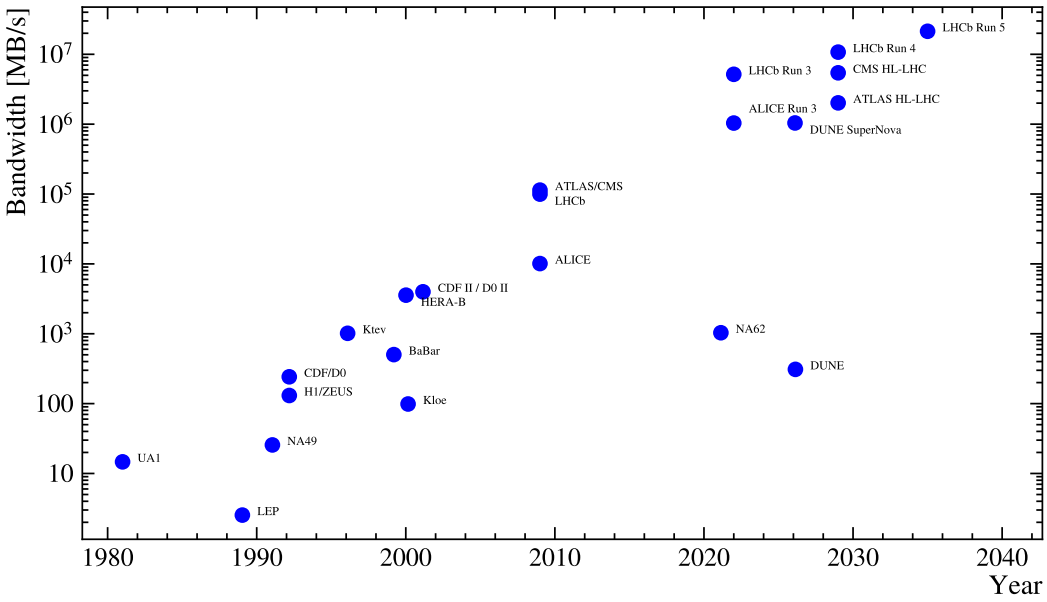


FIGURE 1.4: The evolution of trigger systems in high energy physics experiments [23]. The trigger rate is shown as a function of time.

Data, carefully collected and correctly reconstructed by the experiments, is of vital importance to any new physics search. However, the data is often contaminated by various sources of background, such as misidentified particles, detector noise, etc. As it turns out, constructing a parallel track along the standard experimental workflow presented in Figure 1.2 is possible. This parallel track is known as *Monte Carlo simulations*, subject of the next section, which provides indispensable tools for the physicists to understand the data, test physical hypotheses, and plan new detectors.

1.3 Monte Carlo simulations

Experiments in physics can be very complex and expensive in terms of time, money, and human resources. Careful planning in order to maximize the physics

performance is thus very important. Once the physics experiment is built, it is also very important to understand the data collected and compare it with what is expected from the theory. Physicists use many different types of simulations to achieve these goals. In particular, in high-energy physics, the model itself has to also reflect the stochastic nature of the physics processes. There are usually many possible outcomes, many of which are very rare, and thus require a large number of events to be simulated. Some events can be very complex, and thus require a lot of computing power to be simulated. This is where Monte Carlo simulations come into play. Their role in the scientific process in high-energy physics is summarized in Figure 1.5.

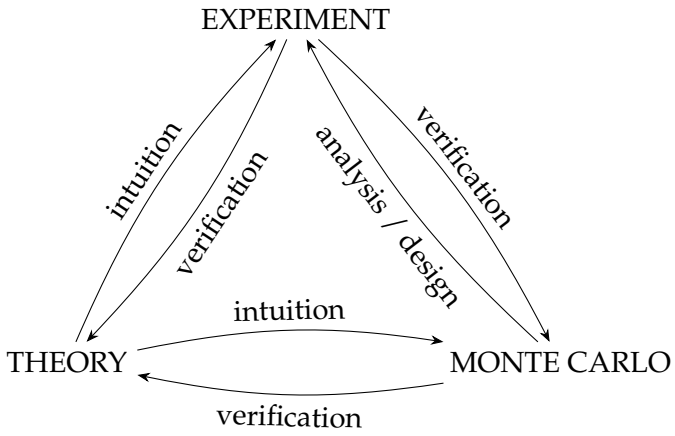


FIGURE 1.5: The role of Monte Carlo simulations in the scientific process in high energy physics inspired by [24].

1.3.1 Statistical foundations of Monte Carlo simulations

A Monte Carlo simulation, informally, is any simulation that relies on random numbers to solve a problem. This does not mean that the problem itself must be of stochastic nature, but it must be possible to reformulate the problem in such a way that the random numbers are used to obtain its solution. More formally, the Monte Carlo method can be defined as representing the solution of a problem as a set of parameters of a hypothetical population. Sequences of random numbers can be used to construct a sample of that population, from which statistical estimates of the parameter can be obtained [25].

The power of Monte Carlo simulations resides in repeated sampling from the same probability distribution function *many times*. The more samples n are taken, the more the distribution of the random variable will resemble the distribution of the underlying population. This comes as the result of the *law of large numbers*. The law of large numbers provides the very important information about what happens when n tends to infinity, but it does not say how it is distributed for finite n. This, in turn, is described by the *central limit theorem*, which states that the sum of large number of independent random variables is normally distributed.

The result of the central limit theorem is crucial for Monte Carlo simulations, as it provides a concrete formula for the error of the Monte Carlo estimate, i.e. $\sigma/\sqrt{n}$, where σ is the square root of variance of the sampled distribution. More importantly, this also implies that the Monte Carlo estimate does not depend on the dimensionality of the problem. As Monte Carlo can be slower than other methods

in lower dimensions, it might be significantly more efficient if the dimensionality of the problem is high and that is usually the case in high-energy physics.

1.3.2 Event generation

Simulation of multiparticle systems is an extremely challenging problem due to the complexity of both the theory and detecting apparatus usually reflected by the dimensionality of the problem and correlations between random variables [26, 27]. For each single collision with k outgoing particles, the phase-space volume element is the $(3k\text{-}4)$-dimensional* momentum space element. If k becomes larger than 3, the complexity of the integration of the relativistic phase space of multiparticle interactions becomes too complex to be solved analytically. At the LHC, the number of particles produced in a single collision can go up to hundreds of all different species of particles of the SM, and in most cases their momenta span over many orders of magnitude.

Modern techniques allow to factorize the full process into different regimes, depending on the amount of momentum transfer:

- at the high scales of energy or momentum transfers, *hard subprocesses* dominate, in which the constituents of the incoming beam particles interact to produce relatively few, but very energetic outgoing particles, and these interactions can be modeled using perturbation theory;
- at the intermediate scales, many additional partons are produced in the form of initial- and final-state radiation (*parton showers*), which still can be modeled using perturbative QCD;
- at the low scales, *soft subprocesses* dominate, in which the incoming partons remain confined in the beams and the outgoing partons interact non-perturbatively, however, they still have to be modeled in non-perturbative theories.

Models and algorithms ranging from hard to soft processes can be simulated and are implemented in the so-called *event generators*. The most important multi-purpose generators are:

- PYTHIA [28, 29] (the most popular in LHC simulations),
- HERWIG [30],
- SHERPA [31].

The event generators usually cannot be used as is without prior tuning and validation [26]. Validation is performed globally in order to make sure that the models describe the underlying physics. RIVET [32] is a Monte Carlo validation tool commonly used in high-energy physics experiments. It also provides a set of experimental analyses useful for MC generator development. Tuning usually boils down to adjusting the free parameters of the models to improve the description of the relevant data. An example of such a tuning tool is the PROFESSOR [33] tool. PROFESSOR works on the output of MC validation analyses where it optimizes the parameters to achieve the best possible fit to the data.

*This includes all the 3 momentum components of particles, subtracted by the 4 constraints of energy-momentum conservation.

In most of the cases, event generators have to be interfaced with other programs in order to provide a complete simulation of an experiment. The output of such a generator is usually written in an *event record*. A very popular format for event records is the HEPMC [34, 35] format. It gives the possibility to store general information about conditions, pseudorandom seeds, as well as data objects such as particles and vertices. Once the event generator fills the event records, they can be then passed to the following step in the simulation chain, described in the next section.

1.3.3 Detector simulation

After the collision takes place, all the new and in many cases still unstable particles traverse the detector and interact with its material. Only then they can be registered via the readout system. As described in Section 1.2, the detector is usually a complex system of many different subdetectors, each designed to use different physics mechanisms to detect properties of the particles. Modeling of particles traversing geometries of significant complexity and with a large number of volumes, as well as all the physical interactions, extending from high-energy particles emitted in collisions down to interactions of eV-scale photons and electrons, is the main task of the *detector simulation* software [36]. GEANT4 [37, 38], FLUKA [39] and MARS [40] are some of the most popular detector simulation tools used in high-energy physics experiments.

Each transportation process [24, 36, 41] is split into a series of *steps*, and at each of these steps the Monte Carlo method requires [42]:

- cross sections in the current material for any possible physical interaction,
- an algorithm to select which interaction will take place,
- a method that applies the effects of each physics interaction such as generation of secondary particles, etc.

In practice, some interactions that take place below a certain energy cut, such as in Bremsstrahlung and delta-ray production processes, may not be sampled individually, but only their collective effect is taken into account [43]. If an external electromagnetic field exists, the Lorentz equation is used to obtain the equations of motion. The equations of motion are then solved using, in most cases, numerical integration methods such as the Runge-Kutta method.

The simulation of electromagnetic interactions in the detector effectively reduces to modeling the interactions of photons and charged particles [44, 45]. Simulation of photon interactions is usually considered much simpler, than the simulation of charged particles, as they occur at discrete points and can modeled this way. Simulation of electromagnetic interactions of photons includes processes such as: Rayleigh scattering, photoelectric effect, Compton scattering, and gamma conversion. As for the charged particles, and in particular electrons and positrons, the most important processes are ionization, Bremsstrahlung, and Coulomb scattering. Simulation of charged particles is much more complicated than photons, mostly due to the large cross sections of elastic and ionization interactions.

Simulation of hadronic interactions, contrary to the electromagnetic interactions, can only be modeled using measured data and phenomenological models [36, 44, 46]. The largest hadronic cross section at low energies is taken by the elastic interaction. They are typically parameterized from data. Intranuclear cascade models describe the hadronic interactions at intermediate energies, modeled as a

succession of independent collisions of the projectile with nucleons. It can be either described as an ensemble of nucleons placed randomly in the nucleus such as in the GEANT4 Binary cascade model, or as a number of shells of constant density, such as in the GEANT4 Bertini cascade model. At really high energies, the simulation relies on phenomenological descriptions of soft QCD interactions. In GEANT4, the Quark Gluon String (QGS) model or the Fritiof (FTF) model are used for this purpose. On top of that, special treatment is required for the simulation of neutrons as they are usually abundantly produced and can have many elastic scatterings with nuclei before they are captured.

2

LHCb experiment in Run 3

In the previous chapter, the role of Monte Carlo simulations in high-energy physics was presented. The techniques described there are crucial for the understanding of the experimental data collected by the experiments and the design of the detectors. This chapter will focus on the LHCb experiment and its setup, with particular emphasis on the description of sub-detectors and the physics programme of Run 3 of the LHC data taking.

2.1 Large Hadron Collider

The LHCb experiment is one of the large experiments at the Large Hadron Collider (LHC) [13, 14, 47]. LHC is, as of 2023, the largest and the most powerful particle accelerator in the world, located at CERN, near Geneva, Switzerland. It is 27 km long in circumference and was designed primarily for the study of proton-proton collisions at a maximal center-of-mass energy of $\sqrt{s}$ = 14 TeV.

The LHC is also capable of colliding heavy ions, such as lead ions. The accelerator consists of two beam pipes, with one proton beam circulating in each of them. Each beam consists of a sequence of up to 2808 bunches, with around 10^{11} protons in each bunch. The beams are accelerated in opposite directions and are brought into collision at four interaction points, which is where the experiments are located. There are four main large experiments at the LHC: ATLAS [48], CMS [49], LHCb [50] and ALICE [51].

The LHC operates in a series of interchanging periods of data taking and maintenance. The data-taking periods are called Runs and are followed by Long Shutdowns, during which the accelerator is upgraded and maintained. The operation of the LHC is planned in advance and the schedule is published by CERN. The LHC is currently in Run 3, which started in 2022 and is planned to last until 2026. It will then be followed by the Long Shutdown 3 (LS3), which is scheduled to last for three years, and will result in the upgrade of the LHC to the High-Luminosity LHC (HL-LHC) [15].

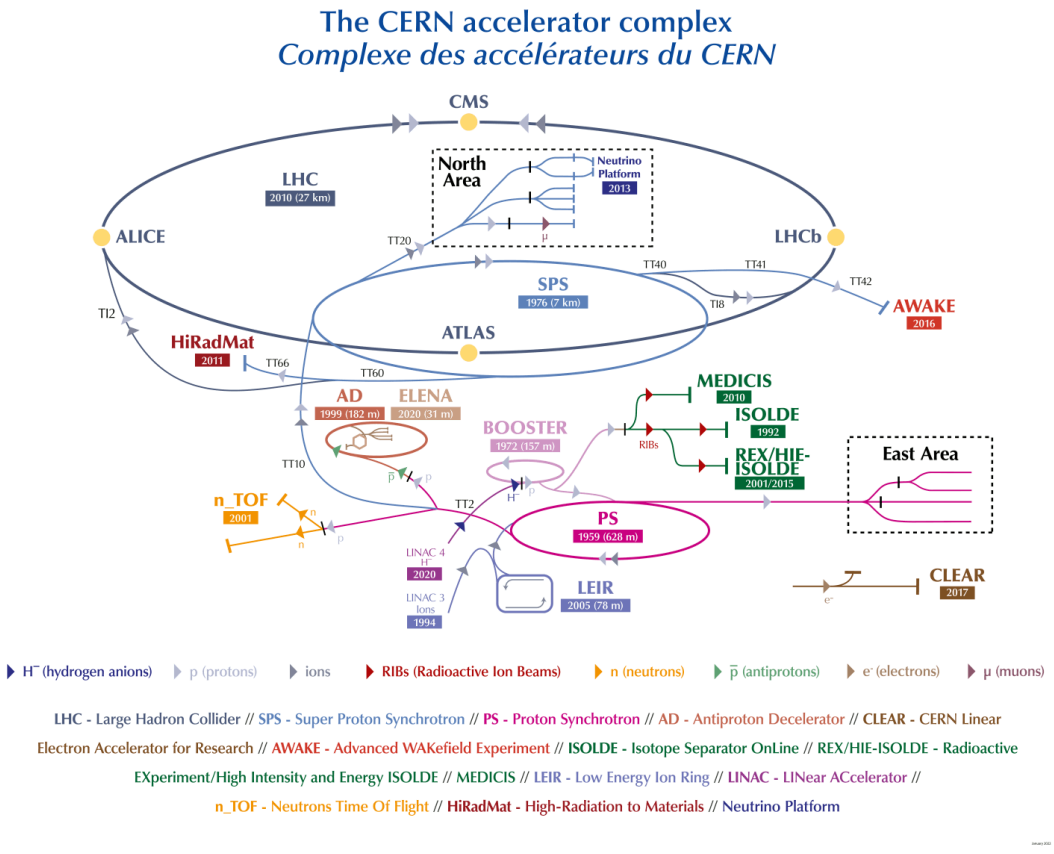


FIGURE 2.1: The CERN accelerator complex [52]. It is a succession of smaller accelerators, each boosting the energy of the particles before they are injected into the LHC (gray ring).

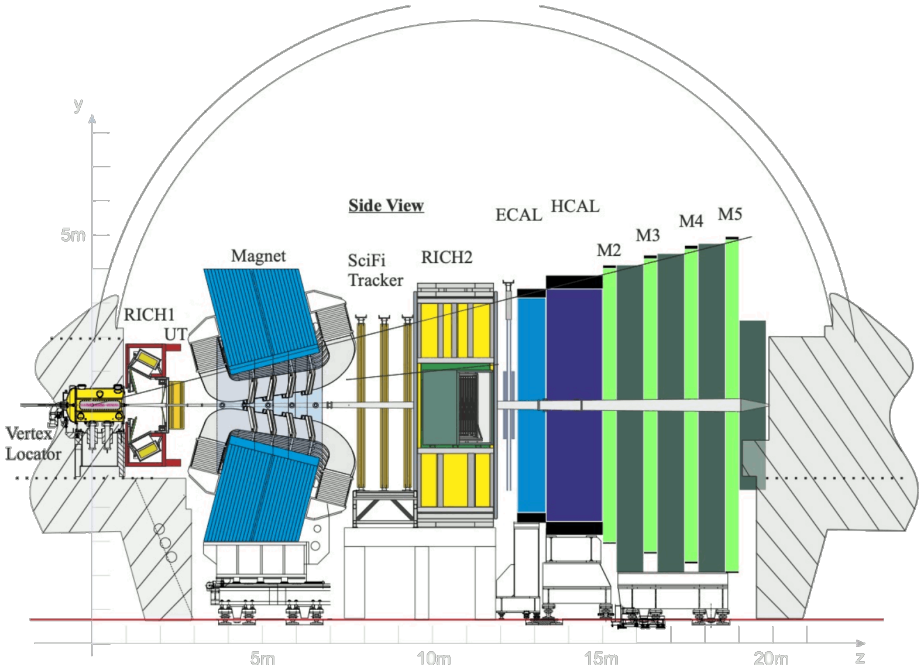
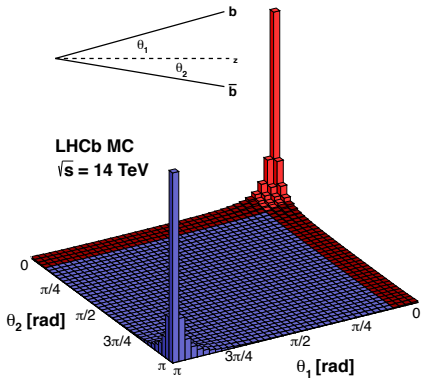


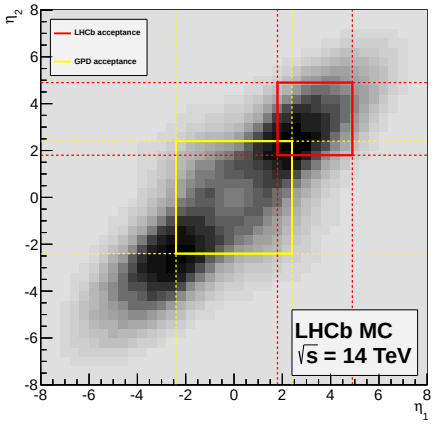
FIGURE 2.2: The layout of the LHCb spectrometer starting from Run 3 of data taking [53, 54].

2.2 The LHCb detector

The LHCb detector [50] is located at the LHC interaction point 8 of the LHC. It is a single-arm spectrometer, with a dipole magnet with a forward angular coverage of 10 mrad up to 300 mrad in the bending plane and up to 250 mrad in the non-bending plane. The particular layout of the detector, significantly different from the general-purpose detectors, is due to the fact that the LHCb experiment is designed to study the decays of heavy flavour hadrons, containing b and c quarks, produced in the forward region of the LHC. The reason for this is that the dominant $b\bar{b}$ production mechanism at the LHCb is through gluon fusion [55]. In this case, the ratio of incoming parton momenta is strongly asymmetric in the laboratory frame, which results in the center-of-mass energy of the $b\bar{b}$ pair being boosted in the direction of the higher momentum proton. Figures 2.3b and 2.3a show the distribution of $b\bar{b}$ production angles at the LHCb experiment obtained via Monte Carlo simulation in terms of pseudorapidity and polar angle, respectively. In addition, the difference in the production angles of $b\bar{b}$ pairs at the LHCb and general-purpose detectors is shown in Figure 2.3b.



(A) $b\bar{b}$ production angles in terms of polar angles.



(B) $b\bar{b}$ production angles in terms of pseudorapidity for the LHCb (red) and general-purpose detectors (yellow).

FIGURE 2.3: Distributions of $b\bar{b}$ production angles at the LHCb experiment obtained via Monte Carlo simulation [56].

LHCb detector consists of a set of sub-detectors, each designed for a different task. The whole spectrum of various detectors used in high energy physics experiment was presented in Section 1.2. In a nutshell, particles leave characteristic signatures in the detectors, which can be then used to identify and measure their properties. In LHCb, the tracking system is used to determine the trajectories of charged particles, and measure their momentum based on their curvature in the dipole field. The VErtex LOcator (VELO) is a tracking sub-detector that sits the closest to the interaction point and is used to reconstruct the primary and secondary vertices with high precision. The Ring Imaging Cherenkov (RICH) system is used to identify charged particles based on the properties of the Cherenkov radiation they produce. The calorimeters are used to absorb the energy of particles inducing electromagnetic and hadronic showers in their heavy metallic bodies. Muons are detected in the muon system based on the hits they leave in the muon chambers.

2.2.1 Run 1 and Run 2 setup

The first version of the LHCb detector operated in Run 1 (2010-2012) and Run 2 (2015-2018) data taking [57] periods. The tracking system consisted of a silicon microstrip Turicensis Tracker (TT) sub-detector upstream of the dipole magnet, and three tracking stations downstream of the magnet: the silicon microstrip Inner Tracker (IT) in the inner part and the Outer Tracker (OT) in the outer part of the stations. VELO consisted of a series of silicon double-sided strip modules providing measurements of r and φ coordinates. The RICH system consisted of two sub-detectors: RICH1 and RICH2, each filled with different media in order to cover different momentum ranges. RICH1 was placed upstream of the magnet and RICH2 was placed downstream of the magnet. The calorimetry system, located downstream of the magnet and RICH2, consisted of four sub-detectors: the electromagnetic calorimeter (ECAL), the hadronic calorimeter (HCAL), the preshower (PS) and the scintillating pad detector (SPD). The SPD calorimeter was used to mark the presence of charged particles and the PS was used to identify the start of electromagnetic showers, necessary in separating electrons, photons and pions. One muon station M1 was installed in front of the calorimeters and four other muon stations (M2-M5) were placed behind the calorimeters, at the far end of the detector. Moreover, the experiment was eqipped with a level-0 (L0) hardware trigger with an output rate of 1 MHz, followed by a High-Level Trigger (HLT) software trigger.

Until the end of Run 2, the LHCb detector collected a total of 9 fb^{-1} of pp, 200 nb^{-1} of pPb and around 30 nb^{-1} of $PbPb$ collision data. The LHCb detector was originally designed to operate at a nominal luminosity of $2 \times 10^{32}\ \text{cm}^{-2}\text{s}^{-1}$ with a pile-up close to unity, but it was able to operate at a luminosity of $4 \times 10^{32}\ \text{cm}^{-2}\text{s}^{-1}$ already in Run 1.

Nevertheless, the precision of the measurements in key physics observables during Run 1 and Run 2 was still limited by the statistics of the data sample. The design of the Run 1-2 detector would not allow for operation at higher luminosities, not only due to the limitations of the level-0 trigger, but also because it could not cope with the increased occupancy and radiation load. Therefore, a major upgrade [58] of the LHCb detector was planned in order to allow for the operation at higher luminosities and to improve the performance of the detector in general.

2.2.2 Run 3 setup

The LHCb detector was upgraded in view of the Run 3 of the LHC data taking [53, 54]. This upgrade is referred to as Upgrade I in order to distinguish it from Upgrade II [59], which is planned for Run 5 of the HL-LHC. The layout of the new detector is shown in Figure 2.2. The luminosity of the experiment has been increased to $2 \times 10^{33}\ \text{cm}^{-2}\text{s}^{-1}$, and the L0 hardware trigger has been completely removed resulting in a fully-software trigger, which plays a crucial role in real-time event reconstruction and selection, efficiently processing data at 40 MHz collision rate. The first trigger stage (HLT1) is implemented on GPUs and is executed by the ALLEN application. HLT1 applies a partial reconstruction and selection on raw data. The second trigger stage (HLT2) is implemented on CPUs and is executed by the MOORE application. HLT2 performs a full reconstruction and the selection of specific decay channels. Further processing of the events for physics analysis is performed at distributed sites offline.

The readout electronics and the data acquisition system were almost completely replaced. In the tracking system, the silicon-strip VELO was replaced with a new silicon pixel detector, the TT was replaced with a new Upstream Tracker (UT) and both IT and OT were replaced with a new Scintillating Fibre (SciFi) Tracker. The new tracking system is described in more detail in Section 2.2.4. SPD and PS were removed from the calorimetry system, and more information on the remaining calorimeters, ECAL and HCAL, is presented in Section 2.2.6. The RICH system was also upgraded, and the details are presented in Section 2.2.5. Finally, the muon system was also modified, and more information is provided in Section 2.2.7.

2.2.3 Infrastructure

Magnetic field is needed in particle physics detectors to bend the trajectories of charged particles in order to measure their momentum. In LHCb, a dipole magnet [50, 60] is used for this purpose. It consists of two saddle-shaped coils, placed mirror-symmetrically to each other, and mount in a window-frame yoke in order to fit in the detector acceptance. Each coil is built of 15 pancakes, arranged in 5 triplets and made of aluminium. The magnet provides an integrated magnetic field of 4 Tm.

The beam pipe [50, 53, 61] in the LHCb detector is a particularly delicate component since the LHCb detector is a forward spectrometer, and the particle density is the highest in the forward region. The amount of passive material in this region must be minimized in order to reduce the number of secondary particles. The most critical part of the beam pipe (< 12 m), at least when it comes to transparency, is made of beryllium, which is a material that fits best the requirements thanks to its high radiation length. The beryllium part of the beam pipe is attached to the spherical VELO exit window, which is made of a thin aluminium foil. In the remaining part of the beam pipe (> 12 m), the beam pipe is made of stainless steel.

2.2.4 Tracking system

Vertex reconstruction is crucial for the LHCb experiment. The VErtex LOcator [50, 62, 63] (VELO) is a unique sub-detector that was designed to provide precise measurements of the primary and secondary vertices. Displaced secondary vertices are very common in b-hadron decays. VELO covers the full momentum and angular range of the downstream detectors in LHCb. At trigger level, the VELO is used to identify the tracks with high-impact parameter. The performance of VELO at the reconstruction level is equally important, as its excellent secondary vertex resolution is key in B_s^0 oscillations and time-dependent *CP* violation measurements. The former version of the VELO was used in Run 1 and Run 2 data taking periods, and the upgraded version has been put into operation before Run 3. Sensors in each module of the upgraded VELO are arranged in a rotated 'L' shape and were designed to tolerate a high and non uniform fluence.

The Scintillating Fibre (SciFi) Tracker [54] is a new tracking system placed downstream of the magnet. It consists of three stations, each with four tracking planes. Each tracking plane is made of modules containing scintillating fibres, which are read out by silicon photomultipliers (SiPMs). The SciFi Tracker was designed to provide a high detection efficiency, low material budget and high resolution in the downstream part of the detector.

The Upstream Tracker (UT) [64, 65] is a silicon microstrip detector placed between VELO and SciFi, just upstream of the magnet. The sensitive area of UT

consists of four tracking planes where the inner two are placed at a small angle to the beam axis. Each tracking plane consists of four types of sensors in order to cope with high occupancy around the beam pipe. In addition to VELO and SciFi, UT plays an important role in the trigger, as it reduces the number of ghost tracks and increases the efficiency of the trigger by enabling measurement of low momentum charged particles before the magnet.

2.2.5 RICH system

As explained in Section 1.2, the Cherenkov radiation is produced by charged particles when they travel through a medium with a velocity greater than the phase velocity of light in that medium. Information about the Cherenkov angle of a particle and its momentum from the tracking system can be used to retrieve the particle's mass and thus its identity.

The Ring Imaging Cherenkov (RICH) system [66–68] in LHCb consists of two sub-detectors: RICH1 and RICH2, each placed in a different location and filled with different media in order to cover different momentum ranges. The RICH1 sub-detector is located upstream of the magnet and downstream of VELO and is filled with C_4F_{10} gas. It is used to identify charged particles with momenta in the range from 2 to 50 GeV/c. On the other hand, the RICH2 sub-detector is located downstream of the magnet, just before the calorimeters and is filled with CF_4 gas. It is used to identify charged particles with momenta in the range from 15 to 100 GeV/c. They have a similar design: a tilted spherical mirror, a secondary flat mirror and a photon detector plane, situated outside of the spectrometer acceptance. The photon detector plane consisted of hybrid photon detectors (HPDs) for the Run 1 & Run 2 version of the RICH detectors, and multi-anode photomultiplier tubes (MaPMTs) for the upgraded version. The readout system was also replaced for Run 3 in order to handle the increased data rates and to improve the performance of the RICH detectors.

The main purpose of the identification system provided by the RICH detectors is to distinguish charged hadrons such as pions, kaons and protons, which is very important in order to reduce the combinatorial background in flavour physics experiments. The RICH system also plays an important role in distinguishing finals states with similar topologies, flavour tagging and trigger-level selections.

2.2.6 Calorimeters

The calorimetry system [53, 69, 70] in LHCb is responsible for the identification and measurement of the deposited energy of hadrons, electrons, positrons and photons. It consists of two sub-detectors: the electromagnetic calorimeter (ECAL) and the hadronic calorimeter (HCAL). Additional two calorimeters, the preshower (PS) and the scintillating pad detector (SPD), were used in Run 1 and Run 2, but were removed for Run 3 in view of their limited role in the new purely software trigger. The ECAL measures the energy of electron, positrons, photons and π^0 mesons, whereas the HCAL measures the energy of hadrons. The energy resolution of the ECAL can be represented with the following formula:

$$\frac{\sigma_E}{E} = \frac{(9.0 \pm 0.5)\%}{\sqrt{E}} \oplus (0.8 \pm 0.2)\% \oplus \frac{0.003}{E \sin\theta}, \tag{2.1}$$

where E is the energy of the particle in GeV, and θ is the polar angle of the ECAL cell. The second term in the formula 2.1 is a constant term that depends on the

mis-calibrations, energy leakage and other effects, and the third term stands for the electronic noise. The energy resolution of the HCAL, on the other hand, is given by the following formula:

$$\frac{\sigma_E}{E} = \frac{(67 \pm 5)\%}{\sqrt{E}} \oplus (9 \pm 2)\%. \tag{2.2}$$

The layout of both ECAL and HCAL remains the same in the upgraded version of the LHCb experiment, while the readout electronics was changed to allow for 40 MHz readout.

	SPD	**PS**	**ECAL**	**HCAL**
number of channels	6016	6016	6016	1488
cell size (mm) Inner	39.7	39.8	40.4	131.3
cell size (mm) Middle	59.5	59.76	60.6	-
cell size (mm) Outer	119	119.5	121.2	262.6

TABLE 2.1: Cell sizes and other parameters of the calorimeters in the LHCb experiment [70].

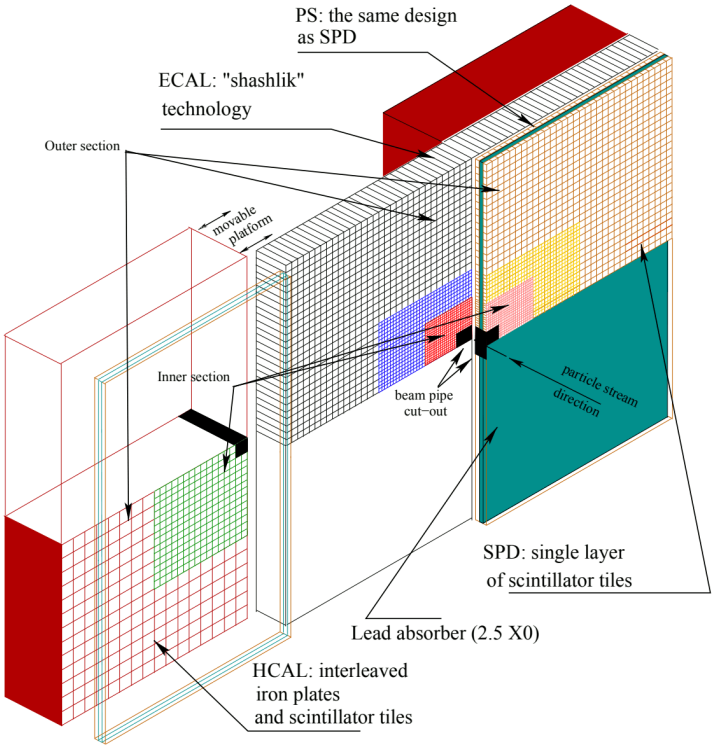


FIGURE 2.4: The layout of the calorimeter system [70] in the LHCb experiment up to Run 2. The hadronic calorimeter (HCAL) is located at the back, followed by the electromagnetic calorimeter (ECAL) and the preshower (PS) and the scintillating pad detector (SPD) in the front. Both the PS and the SPD were removed for Run 3 of data taking.

The calorimeter system is placed downstream of the magnet and the tracking system, perpendicularly to the beamline. The calorimeters are assembled in two halves (A and C side) and can move out horizontally for assembly and maintenance. They are segmented into square cells of different sizes, depending on the density

of particles entering the calorimeter and the energy resolution required. The inner region of the calorimeters is the most finely granular one, as it is placed directly around the beam pipe. The cells in the outer region are larger and are located the farthest from the beam pipe. The ECAL is also equipped with a middle region, with an intermediate granularity. The visual representation of the granularity of the calorimeters is shown in Figure 2.4 and the details of the calorimeter cells and other parameters are presented in Table 2.1.

The calorimeters are built as a succession of absorber, lead for the ECAL and iron for the HCAL, and scintillator layers along the direction of the beam. Particles hitting the absorber layers produce showers of secondary particles, which are then detected by the scintillator layers. The scintillators emit light, which is in turn collected by wavelength shifting fibres (WLS) and read out by photomultiplier tubes (PMTs). The readout system of the calorimeters was replaced for Run 3 to handle the increased data rates and to improve the performance of the calorimeters.

2.2.7 Muon system

A dedicated system is required to identify muons, as they are particles capable of penetrating the calorimeters and reach the outermost part of the LHCb detector. The muon system [71–73] in Run 3 is composed of four stations (M2-M5) interleaved with iron absorbers and is placed the farthest from the interaction point, downstream of the calorimeters. An additional station, M1, was used in Run 1 and Run 2, which was placed upstream of the calorimeters, but was removed for Run 3 as not needed in the new fully software-based trigger.

The muon stations are equipped with multi-wire proportional chambers (MWPCs), which are used to measure the positional information of charged particles. Each MWPC consists of four layers, also called gaps, each consisting of anode wires between two cathode planes. Each station can be further divided into four regions (R1, R2, R3, R4) with increasing distance from the beam. The logical pad segmentation of the cathode planes of the muon chambers is finer in the horizontal direction than in the vertical direction, due to the bending of the muon tracks in the magnetic field that is performed horizontally. The iron absorber plates, also called muon filters, which are used to filter low energy particles. The readout electronic of the muon system has been completely replaced for Run 3.

2.3 Physics programme

The LHCb experiment is a renowned, world's leading flavour physics facility. Originally designed to make precision studies of *CP* asymmetries and very rare decays in the B-meson systems by exploiting the LHC as the most copious source of b-hadrons in the world [74], the LHCb experiment's physics programme has been extended to include a wide range of measurements in the field of heavy flavour physics and beyond. The results gathered by LHCb so far have demonstrated that the Standard Model effectively describes phenomena up to an energy scale of 1-10 TeV [75]. Selected key observables and their uncertainties for the LHCb experiment up to 2018, as well as anticipated uncertainties for the upcoming data taking periods, are presented in Table 2.2.

Observable	End of 2018 ($9\ \text{fb}^{-1}$)	Run 3 ($23\ \text{fb}^{-1}$)	Run 4 ($50\ \text{fb}^{-1}$)	Upgrade II ($300\ \text{fb}^{-1}$)
CKM tests				
γ ($B \to DK$, etc.)	4°	1.5°	1°	0.35°
ϕ_s ($B_s^0 \to J/\psi\phi$)	32 mrad	14 mrad	10 mrad	4 mrad
$\lvert V_{ub}/V_{cb}\rvert$ ($\Lambda_b^0 \to p\mu^-\bar{\nu}_\mu$, etc.)	6%	3%	2%	1%
Charm				
ΔA_{CP} ($D^0 \to K^+K^-, \pi^+\pi^-$)	29×10^{-5}	13×10^{-5}	8×10^{-5}	3.3×10^{-5}
A_Γ ($D^0 \to K^+K^-, \pi^+\pi^-$)	11×10^{-5}	5×10^{-5}	3.2×10^{-5}	1.2×10^{-5}
Δx ($D^0 \to K_S^0\pi^+\pi^-$)	18×10^{-5}	6.3×10^{-5}	4.1×10^{-5}	1.6×10^{-5}
Rare Decays				
$\mathcal{B}(B^0 \to \mu^+\mu^-)/\mathcal{B}(B_s^0 \to \mu^+\mu^-)$	69%	41%	27%	11%
Lepton Universality Tests				
R_K ($B^+ \to K^+\ell^+\ell^-$)	0.044	0.025	0.017	0.007
R_{K^*} ($B^0 \to K^{*0}\ell^+\ell^-$)	0.12	0.034	0.022	0.009
$R(D^*)$ ($B^0 \to D^{*-}\ell^+\nu_\ell$)	0.026	0.007	0.005	0.002

TABLE 2.2: Selected key flavour observables and their uncertainties in the LHCb experiment up to 2018, as well as anticipated uncertainties for the upcoming data taking periods [59, 76].

2.3.1 *CP* violation and the CKM matrix

CP violation [74], or in other words, a violation of a combination of charge conjugation and parity symmetries at the same time, was first observed by V. L. Fitch , J. H. Cronin, J.H. Christensen and R. Turlay in 1964 in the decays of neutral kaons [77], for which Fitch and Cronin received a Nobel Prize in Physics in 1980. This type of violation is one of the three conditions that were proposed by A. Sakharov in 1967 [78] as necessary for the generation, at the very early stages of the Universe, of the observed matter-antimatter asymmetry. In the Standard Model, one of just a few sources of *CP*-violating phenomena in the quark sector is described by the Cabibbo-Kobayashi-Maskawa (CKM) [79] quark-mixing matrix:

$$V_{\text{CKM}} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix}, \tag{2.3}$$

where V_{ij} are the elements of the matrix, and are related to the relative strengths of the transition of down-type quarks ($j \in \{d, s, b\}$) to up-type quarks ($i \in \{u, c, t\}$) as in the following equation:

$$\begin{pmatrix} d' \\ s' \\ b' \end{pmatrix} = V_{\text{CKM}} \begin{pmatrix} d \\ s \\ b \end{pmatrix}. \tag{2.4}$$

The complex elements of the CKM matrix can be represented as vectors in the complex plane, and the unitarity conditions force these vectors to form a triangle, known as the unitarity triangle as shown in Figure 2.5. The position of the apex of that triangle governs the amount of *CP* violation in the quark sector of the Standard Model and thus precise measurements of the angles and sides of the unitarity triangle are crucial for the understanding of the origin of *CP* violation and in the search of new physics. The apex can be determined using entirely tree-level processes (CKM angle γ and the ratio of the sides $|V_{ub}/V_{cb}|$), or via flavour-changing

neutral-current loop processes (FCNC, CKM angle β and neutral B-meson oscillation rates Δm_d and Δm_s).

2.3.2 *CP*-violation measurements

The LHCb collaboration determined the precision of the CKM angle γ to be of 4° [80, 81] with the data collected until the end of Run 2. Measurements of $|V_{ub}|$ and $|V_{cb}|$ were also performed by the LHCb experiment using exclusive decays of Λ_b^0 [82] and B_s^0 [83] hadrons, and the precision of these measurements based on the data collected up to the end of Run 2 is of 6 %. Decay-time-dependent *CP* asymmetries in the B_s^0 system using $b \to c\bar{c}s$ transitions are sensitive to the CKM phase $\beta_s = \arg(-V_{ts}V_{tb}^*/V_{cs}V_{cb}^*)$, which is related to the experimental observable $\phi_s = 2\beta_s$ provided that penguin loop contributions to the decay are negligible. The weak phase ϕ_s is very precisely predicted in the SM, and thus the measurement of ϕ_s is a sensitive probe of new physics. LHCb has already provided very precise measurements of ϕ_s via $B_s^0 \to J/\psi\phi$ decays that are listed in Table 2.2. Another source of *CP*-violating new physics can be found in the parameters $(a_{sl}^{d,s})$ used in $B_{(s)}^0 - \bar{B}_{(s)}^0$ mixing, typically determined using semileptonic decays such as in $B_s^0 \to D_s^- \mu^+ \nu_\mu$ channels [84, 85].

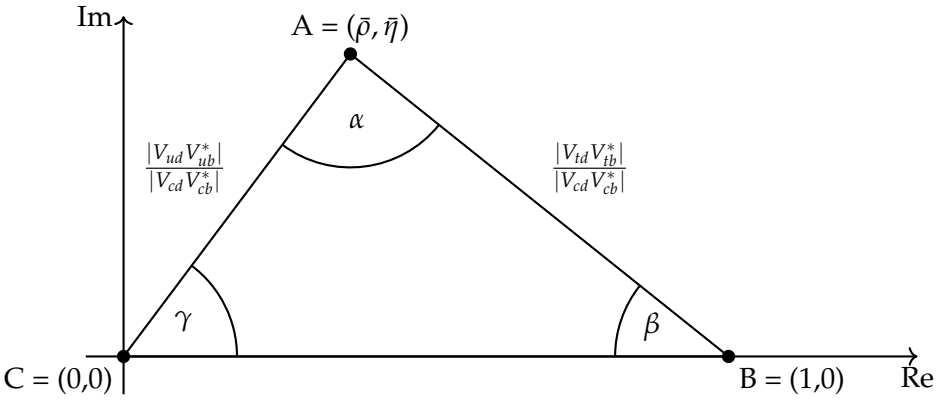


FIGURE 2.5: The CKM unitarity triangle represented in the complex plane [74].

2.3.3 Charm physics

The SM level of indirect *CP* violation in charm systems is expected to be very small and any asymmetry significantly larger than that would be a clear signature of new physics [59, 76, 86, 87]. There is already a huge dataset available at LHCb and more data will be collected with the future upgrades of the detector. The very first observation of *CP* violation in charm decays was made by the LHCb collaboration [88] by measuring the ΔA_{CP} asymmetry parameter. Combined with measurements of other parameters, such as A_Γ [89] and Δx [90] measured by the LHCb collaboration, the constraints on the fundamental parameters of *CP* violation in charm mixing can be obtained. The LHCb experiment is also involved in studies of direct *CP* violation in many charm decays (e.g. $D^0 \to K^+K^-$ or $D^0 \to \pi^+\pi^-$).

2.3.4 Rare decays

Rare or very rare decays can be very useful in the search for new physics and despite several deviations with respect to the SM predictions that were observed, more measurements are necessary in order to verify their nature. This study is central to the LHCb programme and includes rare decays of beauty and charm hadrons, as well as τ leptons [59, 76]. For example, LHCb and CMS collaborations made the first observation of $B_s^0 \to \mu^+\mu^-$ decays and even more suppressed $B^0 \to \mu^+\mu^-$, whose branching fraction ratio is a powerful test of new physics [91–93]. LHCb is also involved in measuring a number of flavour-changing $b \to s\ell^+\ell^-$ and $b \to d\ell^+\ell^-$ transitions, as well as lepton-flavour, lepton-number and baryon-number violating processes (e.g. search for $B \to e^\pm\mu^\mp$), radiative beauty and charm decays (e.g. $B^0 \to K^{*0}e^-e^+$), rare (semi-)leptonic charm decays (e.g. $D^0 \to \mu^+\mu^-$), and rare decays of strange hadrons with Σ and Λ hyperons.

2.3.5 Lepton flavour universality

Interaction strengths of gauge bosons with all the three generations of charged leptons – e, μ and τ – are identical, and this comes more as an accidental symmetry rather than any fundamental axiom of the SM. The only feature there that actually distinguishes leptons from one another is their mass. Therefore, any NP theories where the universality of the lepton couplings is violated would be a clear sign of new physics. At LHCb, the lepton universality is tested in the decays of beauty hadrons. Quantity R_X, defined as the ratio of the decay rates of $B \to X\mu^+\mu^-$ and $B \to Xe^+e^-$, is a particularly interesting test of lepton flavour universality. For example, a recent measurement of R_K [94] (the ratio of $B^+ \to K^+\mu^+\mu^-$ and $B^+ \to K^+e^+e^-$ decay rates) shows a deviation from the SM prediction at the level of 3.1 σ.

2.3.6 Hadron spectroscopy

The quark model describes hadrons as bound states of quarks and gluons, where baryons consist of three quarks and mesons from a quark and an antiquark. It has been very successful in predicting not only the properties of mesons and baryons, but also the existence of more exotic states such as tetraquarks and pentaquarks. The LHCb experiment has been involved in the discovery of many of these multi-quark states. This includes the exotic P_c^+ ($c\bar{c}uud$) [95, 96] pentaquark states, and the Z_{cs} ($c\bar{c}u\bar{s}$) [97], $X_{0,1}(2900)$ ($cs\bar{u}\bar{d}$) [98, 99], T_{cc}^+ ($cc\bar{u}\bar{d}$) [100, 101], and $X(6900)$ ($cc\bar{c}\bar{c}$) [102] tetraquark states, as well properties of other states such as the $\chi_{c1}(3872)$ state [103–105]. Although the existence of the multi-quark states was predicted by the quark model, a detailed analysis of their properties is still needed.

2.3.7 High-p_T, fixed-target and dark sector physics

The physics programme of the LHCb experiment has a broad scope and includes many other areas of research, greatly extending the programme in its first shape. Thanks to the unique capabilities of its detector, which provides access to a kinematic region that is not covered by the other LHC experiments, the LHCb collaboration developed a unique programme of studies in the area of top physics [106, 107], but also the W [108] and Z [109] bosons that play an important role in reducing the uncertainties of PDFs that are crucial in Higgs boson measurements and NP

searches, complementing the measurements performed by the ATLAS and CMS collaborations.

The installation of SMOG (System for Measuring Overlap with Gas) in 2015 [110–113], allowed the LHCb collaboration to pioneer beam-gas fixed-target physics in addition to the already existing beam-beam collisions. This includes both proton and lead beams impinging on gaseous targets, such as He, Ar, Ne, etc.

In the dark or hidden sectors [114], the LHCb employed a broad programme of searches for new particles that might interact very weakly with the SM particles. In fact, LHCb has proven to be the most sensitive experiment to visible dark-photon decays [115, 116] and to GeV-scale Higgs-portal scalars [107, 117].

3

New simulation software

MC simulations play a crucial role in high energy physics. They are essential for designing experiments, developing data analysis techniques, and interpreting the results of measurements. The precision of these simulations directly impacts the quality of the physics results. The increasing complexity and volume of data from the LHC and future experiments make the need for efficient and accurate simulations more pressing than ever. The new simulation framework for the LHCb experiment, GAUSS-ON-GAUSSINO, was designed to meet these challenges, and is described in this chapter.

3.1 From GAUSS to GAUSS-ON-GAUSSINO

In LHCb, data collected by the detector is processed using a set of custom applications based on the GAUDI [118, 119] core software framework. A simplified view on the sequencing of the LHCb data processing applications is shown in Figure 3.2. LHCb applications identically process events collected by the detector itself or events produced by the simulation software. Simulated events are first handled by GAUSS [120] that performs the event generation and particle transport through the detector. The BOOLE application then provides signal digitization. It mimics the specific sub-detector technologies and electronics response, providing the same digital output of the data acquisition system.

Producing necessary simulated samples for physics analyses in LHCb consumed around 90% [121–123] of all the distributed computing resources available to LHCb during Run 2. The increase in the number of events in Upgrade I, and in future upgrades, will require to simulate even more events (Figure 3.1).

The version of the simulation software used for the MC productions of Run 2 data analysis, SIM10 being the latest, was not designed to meet the requirements imposed by the upgrade of the detector. Dependencies of the SIM10 GAUSS framework are shown in Figure 3.3a. It is a single-threaded application that is not able to take advantage of the multi-core CPUs available in modern computing systems. Moreover, the size of the code base and the complexity of the framework made it difficult to maintain. The age of some parts of the code, which evolved from a version created almost 20 years ago, required pruning to make it easier to extend.

Other major changes in the software stack the simulation software had to adapt to include the introduction of DD4HEP [124] for geometry description, multi-threading in GEANT4 and GAUDI [125, 126]. The introduction of new fast simulation models was also a necessity to reduce the time to produce simulated

samples. At the same time, GAUSS still has to provide support for the older geometries when producing simulated samples for Run 1 and Run 2 physics analyses.

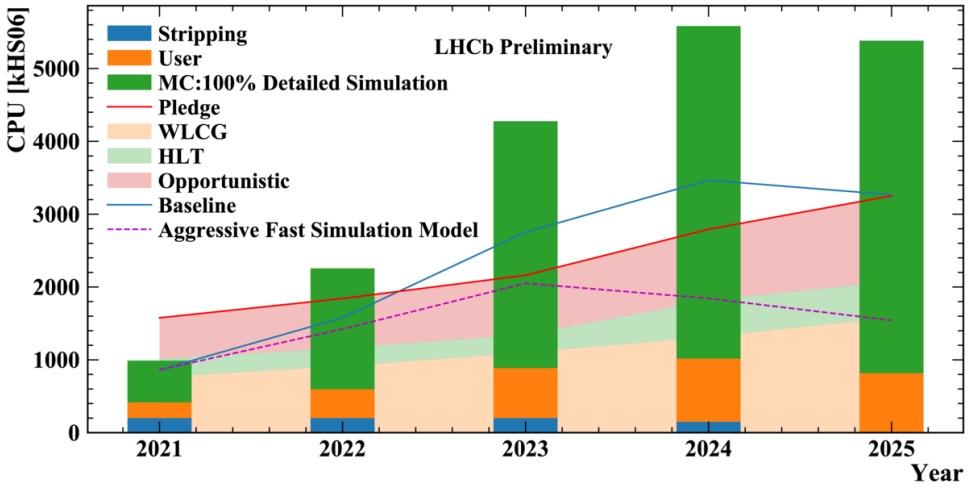


FIGURE 3.1: Projection of the computing resources available to the LHCb [121] experiment. The CPU needs for producing 100% of the required samples with the detailed simulation is shown by the green bars. The CPU needs in two scenarios with different fractions of events produced with detailed, fast and ultra-fast simulations are shown by the blue and dashed red lines respectively.

In order to address the challenges mentioned above, the LHCb simulation team decided to move all the LHCb-independent components from the simulation software and place it in a separate project, called GAUSSINO [128–135], as a core simulation framework, on which all the new versions of GAUSS would be built. It is also possible to run GAUSSINO as a standalone application, and use it to explore new software technologies and simulation techniques. Dependencies in the new LHCb version of the framework, called here GAUSS-ON-GAUSSINO for clarity, are illustrated in Figure 3.3b. GAUSSINO follows the GAUDI's inter-event-based parallelism of the event loop, in which algorithms are scheduled in a way that guarantees thread-safety. GAUSSINO communicates with GEANT4 objects by creating corresponding object factories that act as GAUDI *tools*. The multi-threaded approach in GAUSS-ON-GAUSSINO has already made it possible to simulate more events in time by limiting the memory consumption of each event as shown in an earlier paper [130]. Nevertheless, this was still not enough to meet the requirements imposed by the upgrade of the experiment. Further tuning of the simulation software was needed, as well as the introduction of fast simulation models with the use of ML-based models emerging as an innovative alternative to classical, algorithmic parametrizations.

In the simulation itself, propagating particles through matter dominates the time used by the application. The time spent by the SIM10 version of GAUSS in the simulation in the Run 3 LHCb detector is shown in Figure 3.5. Most of the time is spent in the calorimeters (around 60 %) and the RICH detectors (around 25 %). A finer analysis of the relative time spent in each sub-detector of the upgrade geometry is presented in Figure A.19a for the current version of the framework. Consistent results are obtained for GAUSS-ON-GAUSSINO, as expected and shown in Figure A.19b. Additional information about the time spent by each particle is given in Figure A.21 and Figure A.20 in Appendix A.

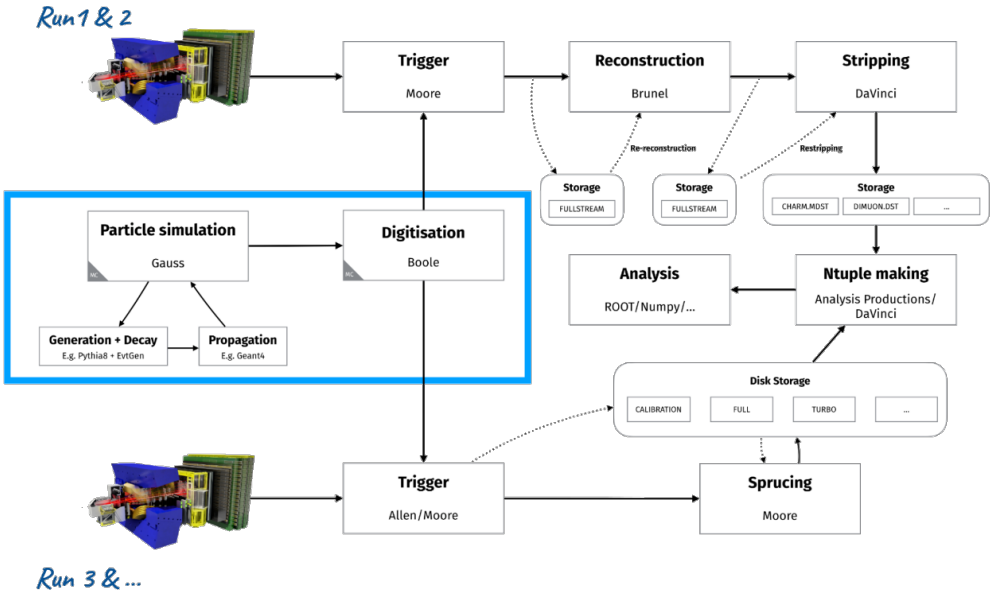
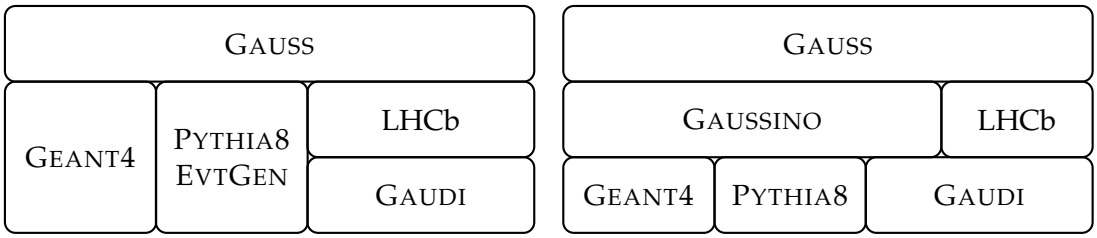


FIGURE 3.2: The data flow in the LHCb experiment [125] in different data taking periods: Run 1 & 2 (top) and in Run 3 (bottom).



(A) GAUSS Run 1 and Run 2 framework. (B) GAUSS-ON-GAUSSINO framework

FIGURE 3.3: Dependencies [127] in the simulation software stack before and after upgrade. Additional LHCb-specific configuration for GEANT4, PYTHIA and EVTGEN is not shown in the dependency graph.

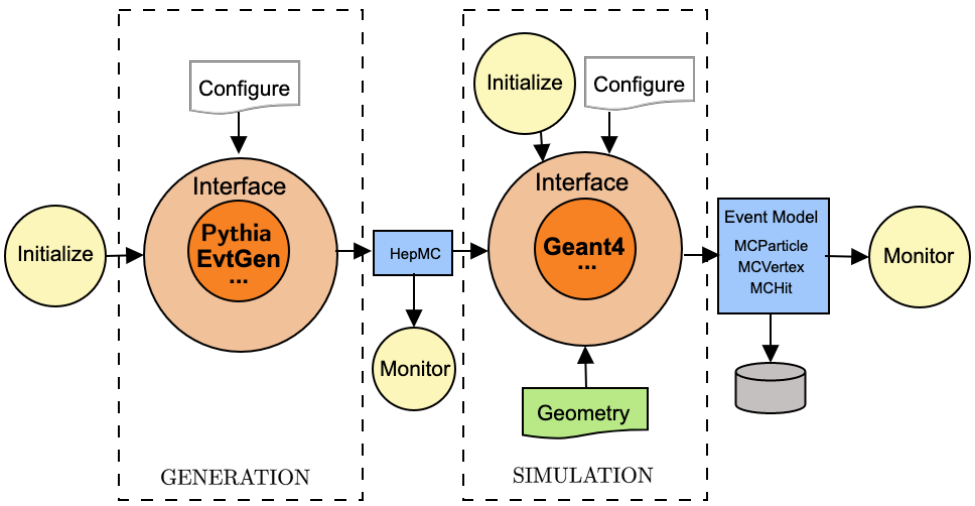


FIGURE 3.4: Dataflow in GAUSSINO with two main phases: generation and (detector) simulation [136, 137].

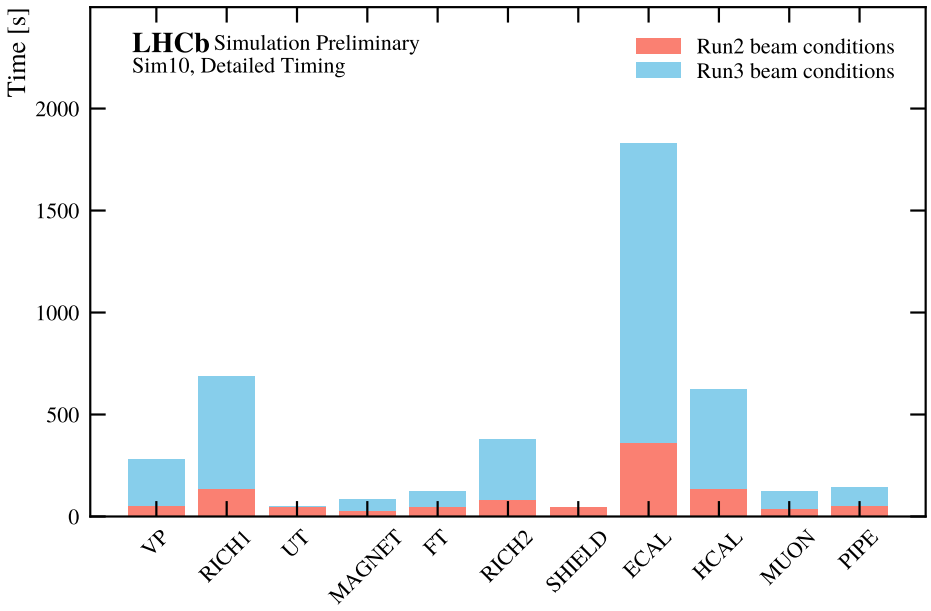


FIGURE 3.5: Detailed timing breakdown per detector in the LHCb Run 3 simulation. Most of the time is spent in the calorimeters (around 60%) and the RICH detectors (around 25%).

Scenarios with different fractions of the samples produced with detailed simulation, fast, and ultra-fast models would allow to fulfill the needs within the resource forecast. In addition, hardware accelerators [131] like GPUs are being investigated for the electromagnetic calorimeter simulation with software packages ADEPT [138] and CELERITAS [139], as well as optical photon propagation in the Cherenkov detectors with MITSUBA3 [140]. A campaign with the goal to introduce a palette of fast simulation models to complement the detailed simulation was launched. ReDecay [141] is a technique, in which the underlying pp interaction is reused in the simulation of the detector multiple times, with an independently decaying signal for each event. Lamarr [142] is an in-house, ultra-fast parametrization framework that extends up to the reconstruction level and provides high-level reconstruction objects in the output. When it comes to the calorimeters, a fast simulation model of ECAL based on a point library [143], as well as ML-based fast simulation models [134] are being developed. Nevertheless, a special interface is needed in GAUSSINO to exploit them via the fast simulation mechanisms available in GEANT4.

3.2 Generation

Event generation is the very first step in the simulation software, and it is responsible for generating interactions which are the primary subjects of the study of physics in this experiment. These scatterings and decays are generated taking into account the kinematics and known dynamics of interactions. Particles in the final states are further known as the primary particles. The exact shape of the event generation process depends on the type of physics processes that are being simulated. In the first chapter 1.3.2, most of the components of the event generation step were described in detail, and in particular: hard subprocesses, parton showers and soft

subprocesses. In LHCb, the additional decays of unstable, heavy-flavour particles is performed using a dedicated event generator: EvtGen [144].

As it turns out, in the simulation software, all these processes can be realized by calling various generator packages, each covering a different aspect of the event generation process. The exact sequence and the level of precision in which the event generation is performed can be usually controlled by the user and depends on how the generator packages are interfaced to the simulation software. Moreover, additional *tools* providing other functionalities, such as generator-level cuts, pile-up interactions, or beam parameters, can be used to further enhance the event generation process.

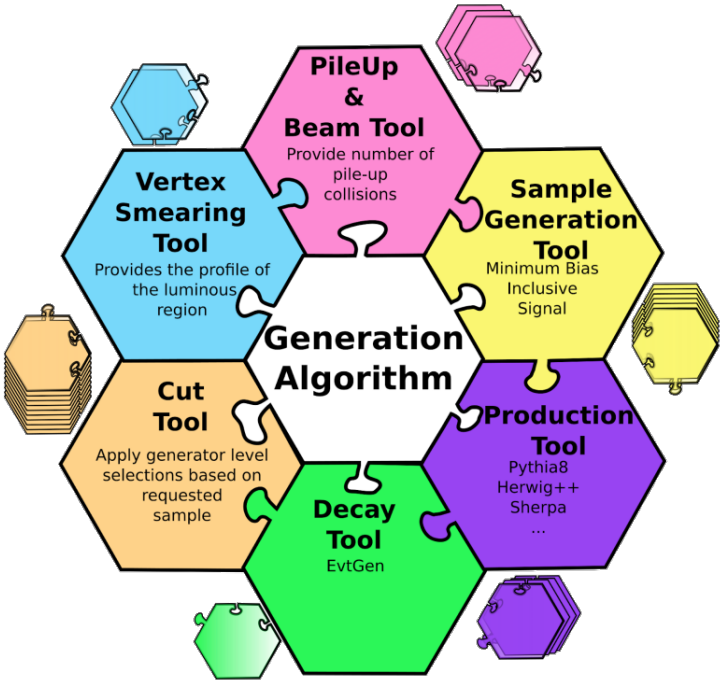


FIGURE 3.6: A graphical representation [145] of the main generation algorithm used in GAUSSINO and GAUSS-ON-GAUSSINO and the additional tools that are called to attach additional functionalities to the event generation process.

3.2.1 Main generation algorithm

GAUSSINO's implementation of the event generation is heavily based on the SIM10 version of the GAUSS framework, i.e. one GAUDI algorithm with additional tools that are called to attach additional functionalities to the event generation process. Each tool has a generic interface with several implementations depending on the provided configuration. A graphical representation of the main generation algorithm `Generation` used in LHC collisions in GAUSSINO with the additional tools is shown in Figure 3.6.

`Production Tool` is the most general representation of the physics processes as depicted in Figure 3.7, which is responsible for all the steps required to generate collisions: simulation of the hard process, parton evolution, the underlying event and hadronization. A concrete implementation of the `Production Tool` is usually realized by interfacing an external generator, such as PYTHIA8 [28, 29],

or HERWIG [30]. In SIM10 version of GAUSS, other external generators were also used, for example BCVEGPY [146] for the production of the B_c meson. GAUSSINO is equipped with two interfaces to PYTHIA8 depending on how the multi-threading is handled in the generator: shared `Pythia8Interface` and thread-local `Pythia8MTInterface`. Other external generators have not yet been ported to GAUSSINO or GAUSS-ON-GAUSSINO, as their use still has to be adapted to the multi-threading scheme used in the new version of the framework. The scope of operation of the `ProductionTool` is based on what was used in SIM10 version of GAUSS, however, another way of interfacing the physics processes was investigated [147] in GAUSSINO, i.e. separating the hard process from the showering and hadronization processes in the production tool, in order to make the GAUSSINO flexible enough to interface MADGRAPH explicitly, in view of using its GPU version currently under development.

`Decay Tool` is another important type of generation tools that is used to decay unstable hadrons produced by the `Production Tool`. In most cases, the EVTGEN generator is usually the default choice for the `Decay Tool` in GAUSS-ON-GAUSSINO, because the LHCb experiment needs a very detailed simulation of B decays, taking into account CP violation effects or angular correlations in decay chains.

`Sample Generation Tool`, on the other hand, defines what type of events are generated in the simulation. The most common types are:

- `Minimum Bias`, which keeps all the events generated by the production generator,
- `Inclusive` that only keeps events containing a particle out of a configurable list of particle types, e.g. in GAUSS-ON-GAUSSINO inclusive events are defined as events containing at least one charm hadron or one beauty hadron,
- `Signal`, which operates in 3 modes:
  - `SignalPlain`, that keeps all the vents produced by the production generator containing a 'signal' particle of a given type, e.g. in LHCb containing B^+, B^0, J/ψ, etc.,
  - `SignalForcedFragmentation`, which does the same as `SignalPlain` but uses forced fragmentation to obtain the 'signal' particle relatively quickly,
  - `SignalRepeatedHadronization`, which again is the same as `SignalPlain` but re-hadronizes the same event several times until the correct type of 'signal' particle is found.
- and `Special` that is applied to produce events either generated with special generators settings or specific generators, e.g. H^0, Z, W or t-physics in LHCb.

`Pile-Up Tool` is used to generate additional interactions in a given bunch crossing. The additional interactions are generated by adding minimum-bias interactions on top of the main interaction. The `Pile-Up Tool` then obtains the number of interactions in one event, N_{int}, following a Poisson law with a mean value ν from the formula:

$$\nu = \frac{\mathcal{L} \cdot \sigma_{\text{tot}}}{f}, \tag{3.1}$$

where $\mathcal{L}$ is the instantaneous luminosity, σ_{tot} is the total cross-section of the collision, and f is the collision frequency of the LHC bunches. The `Pile-Up Tool` can be used in 3 modes:

- `FixedLuminosity` that generates the pile-up events with a fixed luminosity,
- `VariableLuminosity` that uses an exponentially decreasing luminosity profile to generate the pile-up events,
- `FixedNInteractions` that generates a fixed number of interactions per event.

`Beam Tool` is responsible for computing the beam parameters based on the kinematics of two particle beams or a single beam against a fixed target.

If cuts can be applied at the generator level in order to reject the events that would not participate in the physics analysis, then they can be provided using the `Cut Tool`. Concrete implementations exist in GAUSS-ON-GAUSSINO, for example:

- `LHCbAcceptance` that rejects signal particles that do not travel in the acceptance of the LHCb detector,
- `DaughtersInLHCb` that ensures that the direction of decay products of 'signal' particles is within the acceptance of the LHCb detector.

`VertexSmearingTool` is used to implement the generation of the luminous region of the collisions by applying one of the following modes:

- `BeamSpotSmearVertex` that smears the position of the interaction point around the mean collision point by following normal distributions in x, y, and z and fixed time t,
- `FlatZSmearVertex` that smears the position of the interaction point around the mean collision point by following normal distributions in x, y, and flat distribution in z and fixed time t,
- `BeamSpot4D` that smears the position of the interaction point around the mean collision point by following normal distributions in x, y, z, and t.

The main generation algorithm in GAUSSINO is realized in three main actions:

1. **Initialization:** this step mainly deals with the configuration obtained from PYTHON configurables,
2. **Event loop execution:** the result of this step is the generation of one physics event per thread, which is then stored in the HEPMC3 format and transferred to the next step,
3. **Finalization:** monitoring counters and histograms are produced at this stage.

3.2.2 Particle guns

In addition to the `Generation` algorithm, which is responsible for the generation of the events with collisions, there is also another algorithm called `ParticleGun` that is used to generate an arbitrary number of particles with a given momentum and position from a given vertex. This algorithm is very useful when it comes to generating particles for the calibration of the detector or validation of the simulation and reconstruction algorithms. It can also be used to simulate cosmic rays when the detector is taking data, but the beam is not present.

`ParticleGun` is compatible with the `Decay Tool`, `VertexSmearingTool` and `Cut Tool` in the same way as the `Generation` algorithm. The main

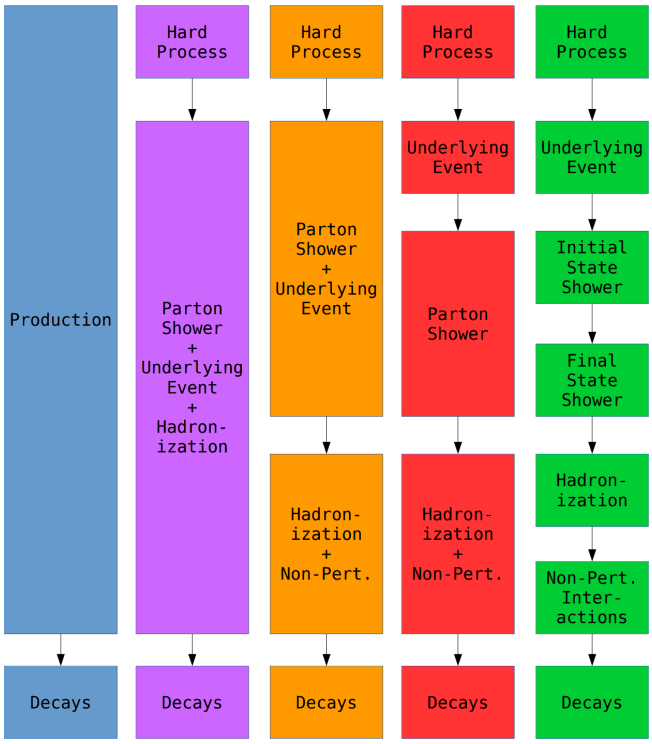


FIGURE 3.7: Possible implementations of the sequencing to event generation processes that were considered in GAUSSINO [147] for direct control through specific interfaces. The blue and purple are the final choices depending on the choice of generators and processes to be simulated.

difference between the `ParticleGun` and the `Generation` algorithm is that the `ParticleGun` does not use external generators to produce the particles, but instead generates them directly in the simulation software. This is achieved by using various `Particle Gun Tools`:

- `FixedMomentum` that generates particles with a fixed momentum and identity,
- `FlatPtRapidity`, which generates particles by sampling the transverse momentum and rapidity from a flat distribution,
- `GaussianTheta` that can be used to generate particles by sampling the polar angle from normal distribution, and the azimuthal angle and momentum from a flat distribution,
- `MomentumRange` that generates particles by sampling the polar angle, azimuthal angle, and momentum from a flat distribution,
- `MomentumSpectrum`, which generates particles with momentum sampled from a custom distribution,
- `BeamShape` that emulates the beam of particles based on the beam parameters such as the beam spot size, emittance, etc.
- `MaterialEval`, which is a tool that generates a grid of particles either defined in Cartesian or cylindrical coordinates.

The main PYTHON configurable responsible for the event generation part in GAUSSINO is called `GaussinoGeneration`, whereas an equivalent in GAUSS-ON-GAUSSINO is called `GaussGeneration`. `GaussinoConfigurable` can be also used to generate events based on particle guns.

3.3 Particle transport

In Section 1.3.3, the processes happening in the detector once the primary particles are generated were described in detail. In GAUSSINO and GAUSS-ON-GAUSSINO, the main software engine responsible for the particle propagation through the detector is GEANT4. The simulation with GEANT4 physics-based models is referred to as the *detailed simulation*. GEANT4 provides the way of simulating the detector response, which is the most precise and resembling reality because it simulates physics interactions in detectors. Parametrization of the detector response is another way of doing that, and it is referred to as the *fast simulation*.

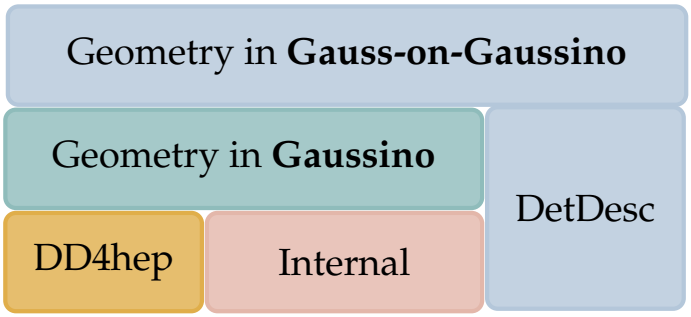


FIGURE 3.8: Dependencies of GAUSSINO and GAUSS-ON-GAUSSINO on various detector description libraries.

3.3.1 Geometry description

The geometry of the detector, described in its native format has to be converted to C++ objects understandable by GEANT4. Therefore, dedicated tools are needed to convert the geometry description to the format that can be used by GEANT4. The dependencies of GAUSSINO and GAUSS-ON-GAUSSINO on different description tools are shown in Figure 3.8. GAUSS supported the legacy geometry description in LHCb that was used for Run 1 and Run 2 of the data taking: DETDESC [148] – a toolkit that was used to read the geometry description written in XML format. The LHCb upgrade required the introduction of a new geometry description toolkit in the software stack, DD4HEP [124], which speeds up the geometry readout by moving part of the logic to the C++ code. DD4HEP is a common toolkit that can be used to describe the geometry of any detector in HEP. An interface to steer the conversion of DD4HEP objects to native GEANT4 geometry objects was implemented in GAUSSINO allowing for the description of the Run 3 LHCb detector within GAUSS-ON-GAUSSINO, as well as for other detectors using pure GAUSSINO. GAUSSINO is also equipped with an internal geometry description package, EXTERNALDETECTOR, that allows to add simple volumes to the existing geometry at runtime, or can be used in GAUSSINO standalone mode to provide a simple geometry description for the simulation of the detector response.

3.3.2 Detailed simulation with GEANT4

Interface

GEANT4 is a toolkit that was designed to work in a standalone mode. In order to pass the data between GAUSSINO and GEANT4, a special interface was developed [128]. A special customization of the main run manager used by GEANT4 (`G4MTRunManager`) and thread-local run managers (`G4WorkerRunManager`) was introduced in GAUSSINO: `GiGaMTRunManager` and `GiGaWorkerRunManager` respectively. The main execution of the detector transport algorithm is done in the `GiGaAlg` algorithm. In addition to that, a special `GiGaMT` service is used to initialize GEANT4 and manage event-by-event communications with GEANT4. The interplay between each of the components is shown in Figure 3.9. GAUDI tools work as factories that create GEANT4 objects and manage them throughout the simulation process. Once the event generation phase is finished, the generated event is placed in a FIFO queue, from which the GEANT4 worker threads take the event and perform the simulation.

Configuration

Modeling of the interactions between particles and the detector material is done by using the so-called *physics lists*. They are used to define the set of models that are used to simulate the interactions between particles and the detector material. The models were described schematically in Section 1.3.3. The choice of the physics list depends on the experiment and usually boils down to the trade-off between the precision of the simulation and the time needed to perform it. In GAUSSINO, the physics lists are defined in the `GiGaMT` service, and the choice of the physics list is made in the PYTHON configuration file via the `GaussinoSimulation` configurable. In GAUSS-ON-GAUSSINO, a set of predefined physics lists is available, optimized for the LHCb detector, and the choice of the physics list is made configurable in the `GaussSimulation`.

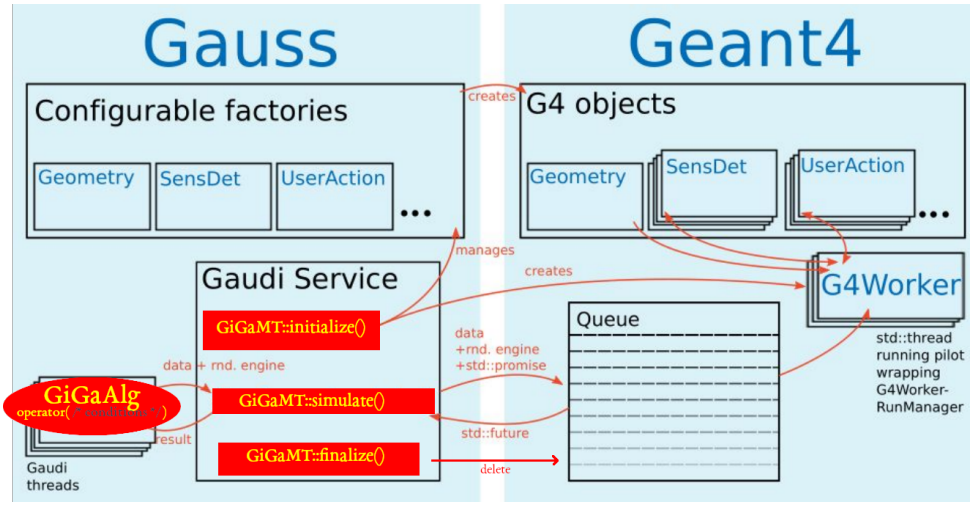


FIGURE 3.9: Integration and workflow between GAUDI and GEANT4 simulation frameworks in GAUSSINO.

3.3.3 Interfacing fast simulations with GEANT4

GAUSSINO provides a generic FASTSIMULATION interface to GEANT4 objects in order to minimize the work spent in the future on implementing fast simulation models, and also to guarantee the integrity of the simulated data. Following the convention already present in GAUSSINO, the FASTSIMULATION interface consists of object factories ensuring GEANT4 objects are configured properly and at the right moment when running the application. A set of the most important factories and their GEANT4 counterparts are presented in Figure 3.10. Most of the work necessary to implement a fast simulation model itself can be pushed to the configuration in python files. The most optimistic scenario is that the developer will only have to implement a `G4VFastSimulationModel::DoIt()` callback method in C++ that is the key component of the whole interface and actually describes the whole process of generating fast hits.

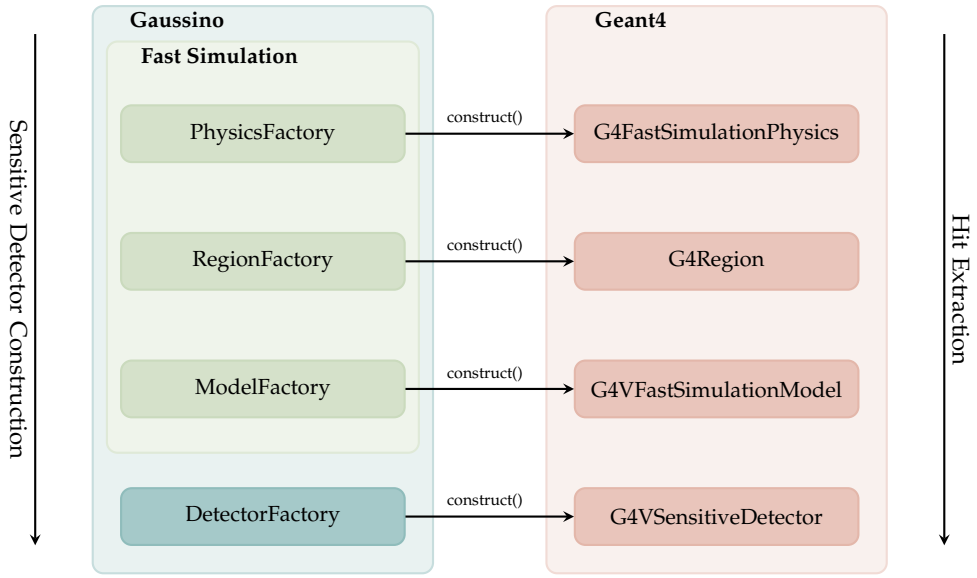


FIGURE 3.10: A simplified model of the FastSimulation interface with a set of dedicated factories that construct the corresponding Geant4 objects.

Two, purely abstract, fast simulation models are introduced in GAUSSINO in order to measure the performance of the interface, and to mark off a lower bound in terms of time spent on the simulation for all further fast simulation models. An `ImmediateDeposit` model generates one hit per particle that intercepts the region where the `ImmediateDeposit` model is active and deposits all of its energy in that hit. `ImmediateDeposit` gives useful information about the timing needed for the infrastructure itself to call the fast simulation methods. The `ShowerDeposit` model works in a similar manner, but it splits the energy of a particle into a selected number of hits, and generates them randomly around the position where the particle intercepted the region. `ShowerDeposit` provides the minimum amount of time needed to generate a specific number of hits with no additional calculations.

A comparison between these two different models using a particle gun that creates a grid of 3328 evenly-spaced photons originating at the LHCb interaction point, is presented in Figure 3.11. Time spent in the `ImmediateDeposit` model is comparable across different photon energies and works as expected. On the

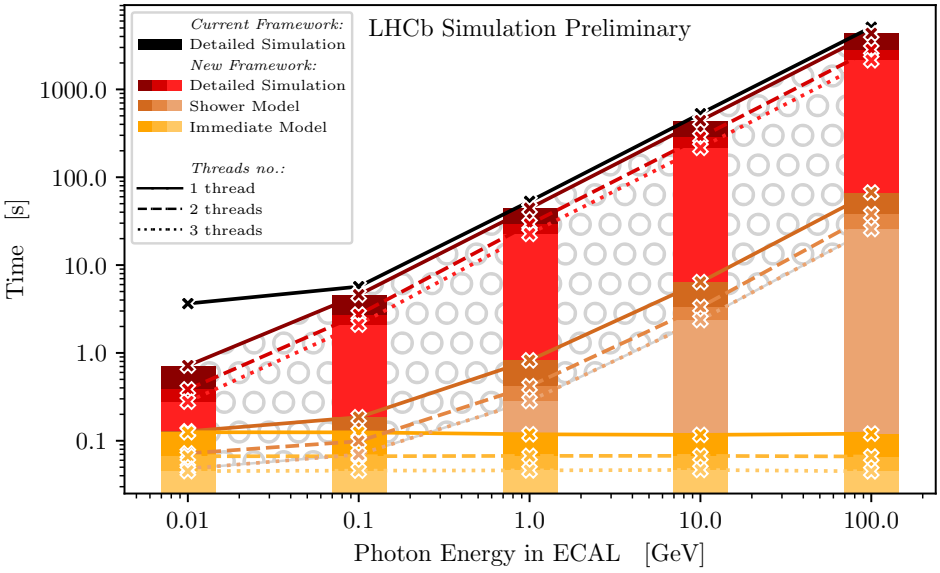


FIGURE 3.11: Comparison of the time [127] spent by different fast simulation models (`ImmediateDeposit` and `ShowerDeposit`) and a detailed simulation with GEANT4 in the electromagnetic calorimeter. In each of the models tested, a particle gun generates a grid of evenly-spaced photons of a particular energy. For the detailed simulation the time of the current version of GAUSS is also given as reference.

other hand, the time spent in the `ShowerDeposit` model increases with the energy of photons. Naturally, this is caused by a larger number of hits generated in each shower. In the worst case, a 100 GeV photon generates 21558 hits on average in the calorimeter. Only around 25 seconds are needed for the fast simulation infrastructure in GAUSSINO with 3 threads to simulate 3328 x 21558 hits. In a detailed simulation, the time needed to simulate the same number of hits rises to 2145 seconds. The results prove that the infrastructure, used by the FASTSIMULATION interface, provides the possibility to significantly improve the time spent by the simulation software in the detector, provided the fast simulation model gives a similar level of precision in physics.

Fast simulation training datasets

Many of the advanced, fast simulation models require prior tuning or training on some input data in order to provide valid results. When implementing a fast simulation model, a developer specifies a region of the whole detector that does not necessarily have to coincide with the sub-detectors boundaries. The information required to train these models is not always available in a standard output file, as GAUSSINO stores only the minimum amount of information required for physics studies. Therefore, the developers of fast simulation models should be given the possibility to turn off any unneeded optimization features and gather particle information at any given place in the detector to train the model.

Information about the simulated objects can be easily obtained by introducing a new, virtual regions, imitating the sensitive sub-detector that would register hits of an abstract type with all the information needed to train the fast simulation model. A few difficulties may be encountered with this approach though. Since these

detectors will only be used in just a few, specific studies, setting them up should be configurable on the fly without having to introduce them in the existing detector description, hosted in a database in an xml format. This functionality is provided by a new package in GAUSSINO called EXTERNALDETECTOR, that allows for virtual sub-detectors of any shape to be inserted. An example of an external plane-like detector embedded in the LHCb geometry, as seen by GEANT4, is illustrated in Figure 3.13. A side view of the same setup is presented in Figure 3.14, together with information of simulated particles and their origin.

The user can also choose what kind of factories should be attached to an external detector. In principle, the external detector can become a sensitive detector (i.e. it will be activated to register hits) or the user can add a monitoring tool that will be launched when the simulation of the whole event is complete in order to verify the integrity of the collected data. MCCOLLECTOR provides a set of abstract sensitive detector factories that are easier to configure than those used in the standard simulation.

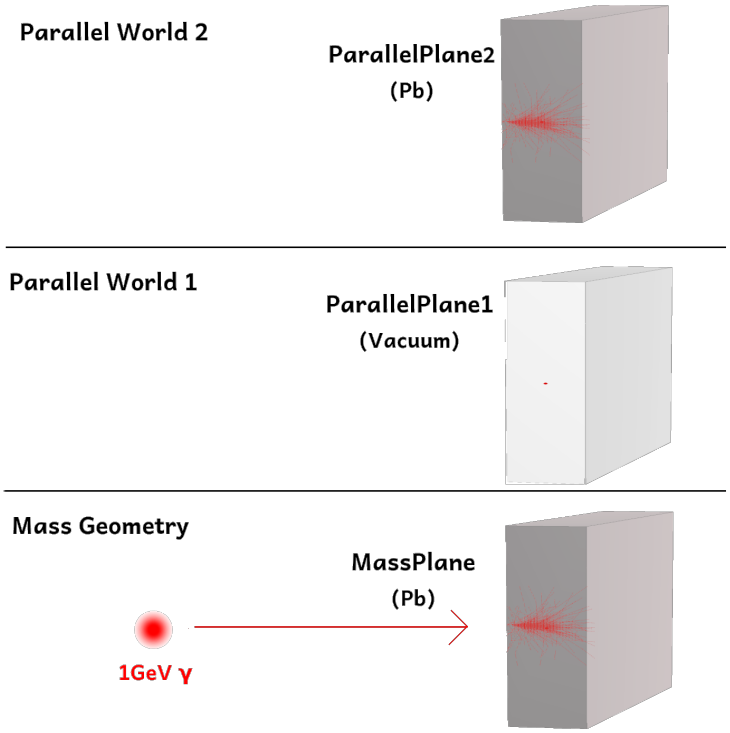


FIGURE 3.12: A simple example demonstrating the core functionality of the PARALLELGEOMETRY package. In GEANT4, multiple parallel worlds can be defined, each of them containing a separate geometry. The simulation takes place simultaneously in all of the defined geometries depending on the hierarchy of the worlds, location and materials of the volumes.

Finally, it might be the case that the external detector will overlap with other existing volumes in the geometry. PARALLELGEOMETRY exploits an abstract concept introduced by GEANT4 that allows for having multiple geometries in parallel, each of them performing the particle transport without interference from objects defined in other geometries. The mechanism of the PARALLELGEOMETRY package is illustrated in Figure 3.12.

The software provides a generic way of producing the training datasets for fast simulation models. Two simple examples are presented in this chapter in order to

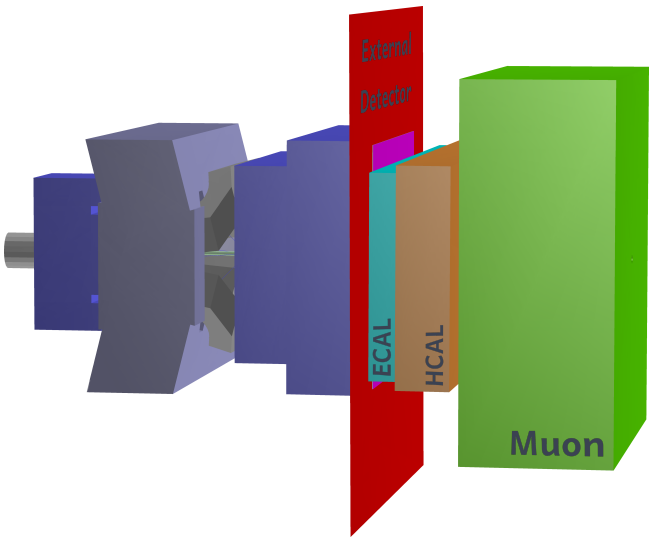


FIGURE 3.13: LHCb upgrade geometry [127] as seen by the GEANT4 toolkit with an example of a plane-like detector (red, thin plane), introduced by the EXTERNALDETECTOR package. When used as a collector of particle information, it may provide the source of training information about incident particles for all the sub-detectors placed downstream from it along the beamline: ECAL (cyan box), HCAL (orange box), or muon system (green box).

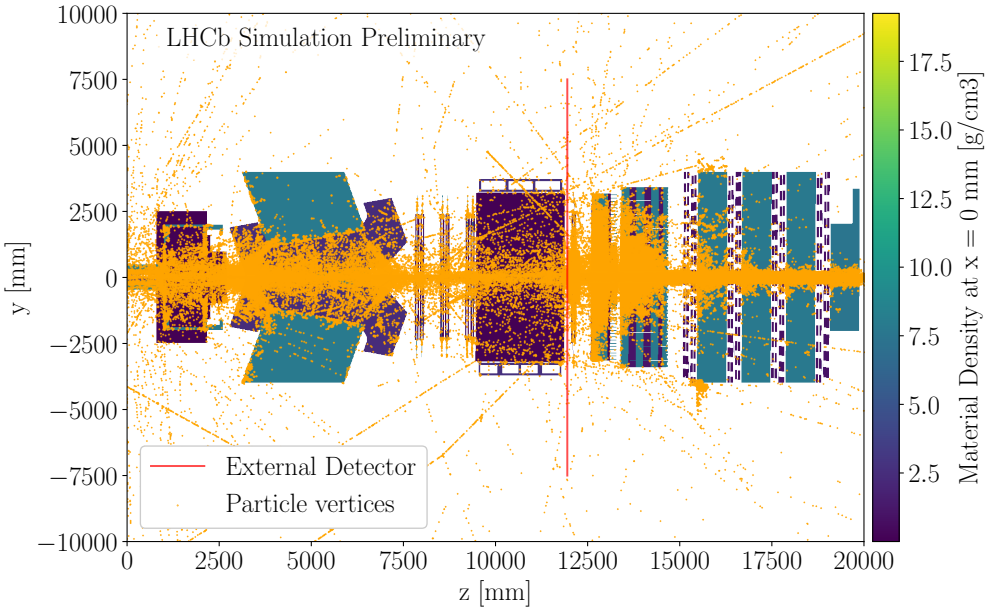


FIGURE 3.14: Particles generated [127] using the SIM10 framework in the simulation of a minimum bias event with the beam conditions as foreseen in the Run 3 data-taking period and the upgrade geometry. An external plane-like detector that collects information about traversing particles is depicted with the red color.

show how they can be used for the models under development for ECAL (e.g. point library [143] or ML-based [134]). Visual representations of the training datasets, produced by placing a collector plane in front of ECAL, are illustrated in Figure 3.15. In Figure 3.15a, a grid of 3328 evenly-spaced photons, similar to that used as an example when testing the performance of the interface, is shown. In Figure 3.15b, an input for the fast simulation studies requiring minimum bias events is presented.

3.4 Visualization

Visualization techniques in simulations are essential for verifying the integrity of the description of the detector geometry, as well as for understanding the data produced by the simulation. The natural choice for visualization in GAUSSINO is anything that is already supported by GEANT4, as it is the main particle transport engine used in the framework. Another possibility is to use the new experiment-independent framework for event and geometry visualization, already used in LHCb: PHOENIX. Both of these technologies were explored in GAUSSINO [149], and the results are presented in the following sections.

3.4.1 Integration of GEANT4 visualization in GAUSSINO

GEANT4 provides support for both interactive and batch visualization of the geometry and data produced by the simulation. The visualization can be enabled with the GEANT4 visualization drivers, which are in most cases external libraries that GEANT4 is linked against. The support for some of the libraries was provided in the previous versions of the GAUSS simulation framework in the form of the GEANT4 visualization manager. However, the support for the visualization was not enabled in the GAUSSINO framework, and it had to be re-implemented facing the challenges of the multi-threading environment. In particular, the visualization thread, spawned by GEANT4, had to be correctly synchronized with all the other GAUDI and GEANT4 threads.

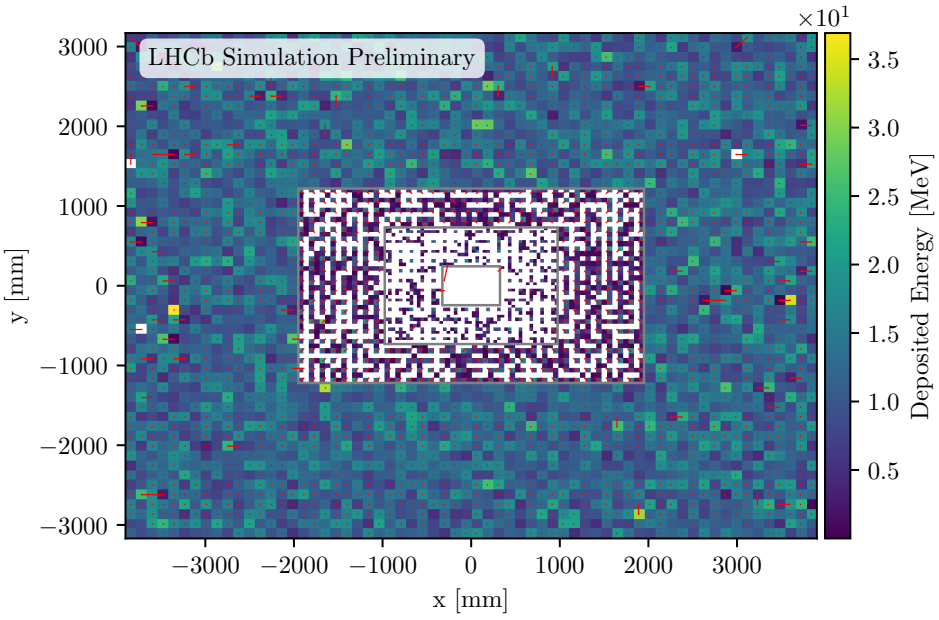
Available visualization drivers

A wide array of visualization drivers is available in GEANT4, each serving a distinct purpose. Some prioritize achieving the highest resolution, such as DAWN, while others, like OPENGL, emphasize interactivity. In GAUSSINO, four drivers have been explored and activated [150]: ASCIITREE, DAWN, HEPREP, and OPENGL.

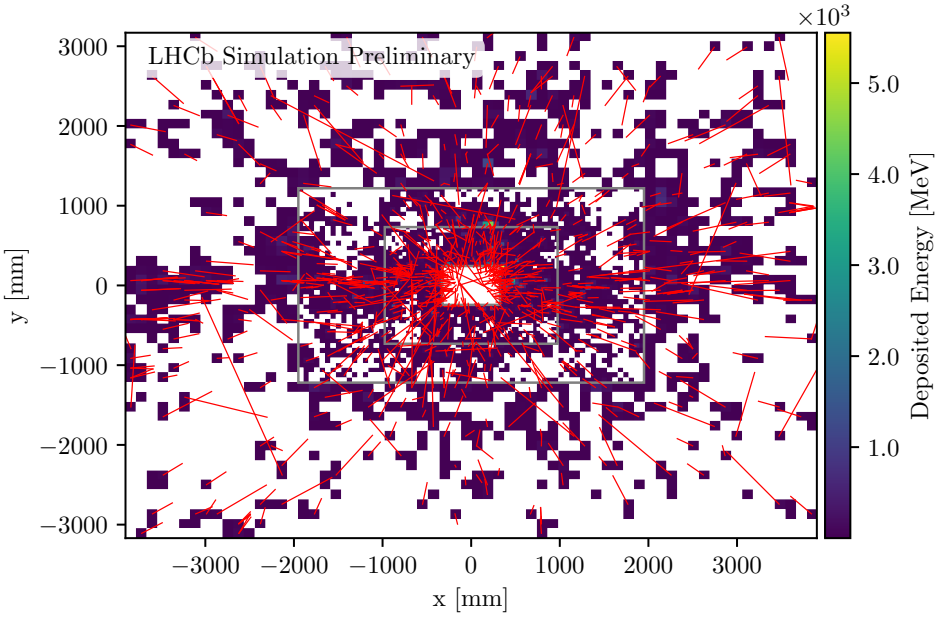
ASCIITREE serves as a non-graphical system for dumping geometry hierarchy into a text file. It provides a tree representation with control over the level of detail and offers calculations for mass and volume. Despite its speed and low computational cost, ASCIITREE lacks the capability to visualize trajectories or hits, restricting its use to extracting geometry information only.

DAWN is recognized as a high-quality technical renderer, aiming for high-resolution images through vector PostScript output. However, its computational intensity and reliance on specialized browsers limited interactivity and ease of use for viewing and exporting images.

Unlike its predecessors, HEPREP offers interactive features such as zooming, translation, and rotation. Its hierarchical view of geometry and data, coupled with control over data visibility, enhances user experience. Despite supporting various vector formats for export, photorealistic images are not produced by HEPREP, but is limited to a wireframe view and simple area fills in geometry.



(A) Particle gun with a grid of 3328 evenly-spaced 100 MeV photons.



(B) Minimum bias event with the beam conditions as foreseen in the Run 3 data-taking period and the upgrade geometry.

FIGURE 3.15: Visualization [127] of the training dataset produced by placing a collector plane in front of the electromagnetic calorimeter. Each of the images represent the ECAL energy deposits (hits) projected onto an xy-plane. The main role of the collector plane is to gather the positions of all particles intercepting the front face of ECAL. The particle positions are then linked (red lines) with the energetic centers of the showers generated by these particles.

OPENGL provides direct visualization of geometry and data from the GEANT4 command console, rendering photorealistic images with interactive features like zooming and rotation. While fast redraws are offered leveraging graphics hardware, additional GL libraries are required, and struggles with large numbers of steps or hits in visualization are encountered.

Activating the GEANT4 visualization driver in GAUSSINO is straightforward. In the Python configuration file, the visualization should be initialized, GEANT4 chosen as the visualization framework, and the desired Geant4 visualization driver selected.

Geometry visualization

Visualization of the detector geometry became the first priority when working on the integration of the visualization infrastructure in GAUSSINO. The geometry visualization is crucial for verifying the integrity of the detector description, and helped in identifying missing or incorrectly modeled parts of the geometry during the commissioning of the upgraded LHCb detector.

The geometry visualization works with all different geometry description tools used in GAUSSINO and GAUSS-ON-GAUSSINO. An example of the visualization of simple volumes using the EXTERNALDETECTOR package in GAUSSINO standalone. In Figure 3.16a, a cube block made of lead is visualized, in an attempt to mimic the behavior of the calorimeters, while in Figure 3.16b, a set of silicon planes is used to mimic the behavior of silicon detectors used for tracking in real experiments.

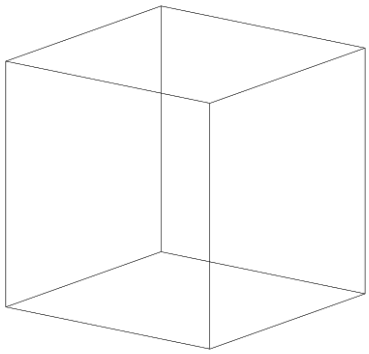
More advanced visualization of the LHCb detector is also possible with detector description tools available in GAUSS-ON-GAUSSINO. The upstream viewpoint with the LHCb detector description rendered with DETDESC is shown in Figure 3.16c, while the detector description rendered with DD4HEP is shown in Figure 3.16d. Similarly, the downstream viewpoint is shown in Figure 3.16e and Figure 3.16f for DETDESC and DD4HEP, respectively.

Simulated data visualization

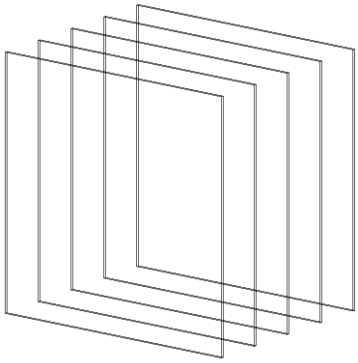
In the previous section, visualization with GEANT4 of the geometry was discussed. However, visualization of simulated data is also possible with GEANT4 visualization drivers and can be enabled in GAUSSINO and GAUSS-ON-GAUSSINO. It can be used to visualize trajectories, hits, and other data produced by the simulation.

In some cases, a special configuration in GAUSSINO is needed that enables the recording of additional simulation data information such as full trajectory or native hit information, which in regular simulations is not stored due to the large amount of data produced. When it comes to the amount of detail in the visualization of trajectories, the user can choose between the following options:

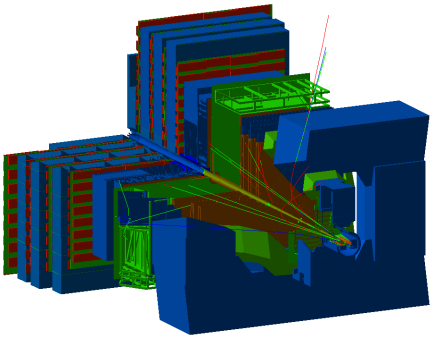
- `All`, when all trajectories are drawn,
- `Marked` that keeps only the trajectories of particles that hit the sensitive detectors,
- `Truth`, when only the trajectories of particles that were persisted in the simulation are drawn.



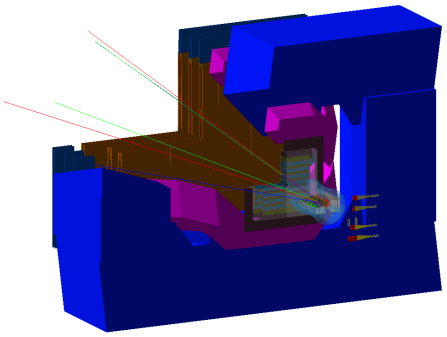
(A) Visualization of a cube block made of lead with the EXTERNALDETECTOR package in GAUSSINO standalone.



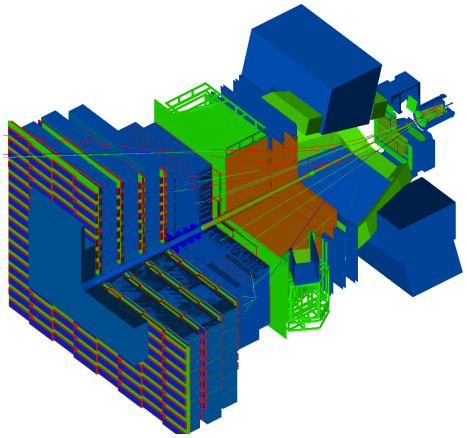
(B) Visualization of silicon tracking planes with the EXTERNALDETECTOR package in GAUSSINO standalone.



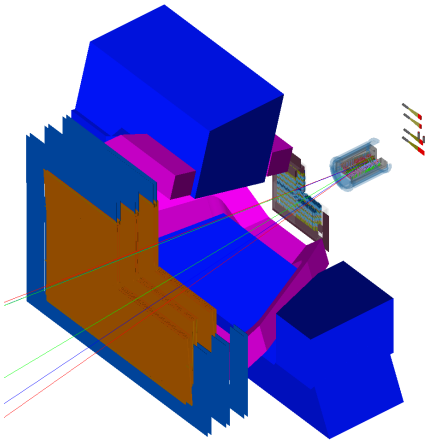
(C) Upstream viewpoint with the LHCb detector description rendered with DETDESC



(D) Upstream viewpoint with the LHCb detector description rendered with DD4HEP (simulation status as of August 2022).

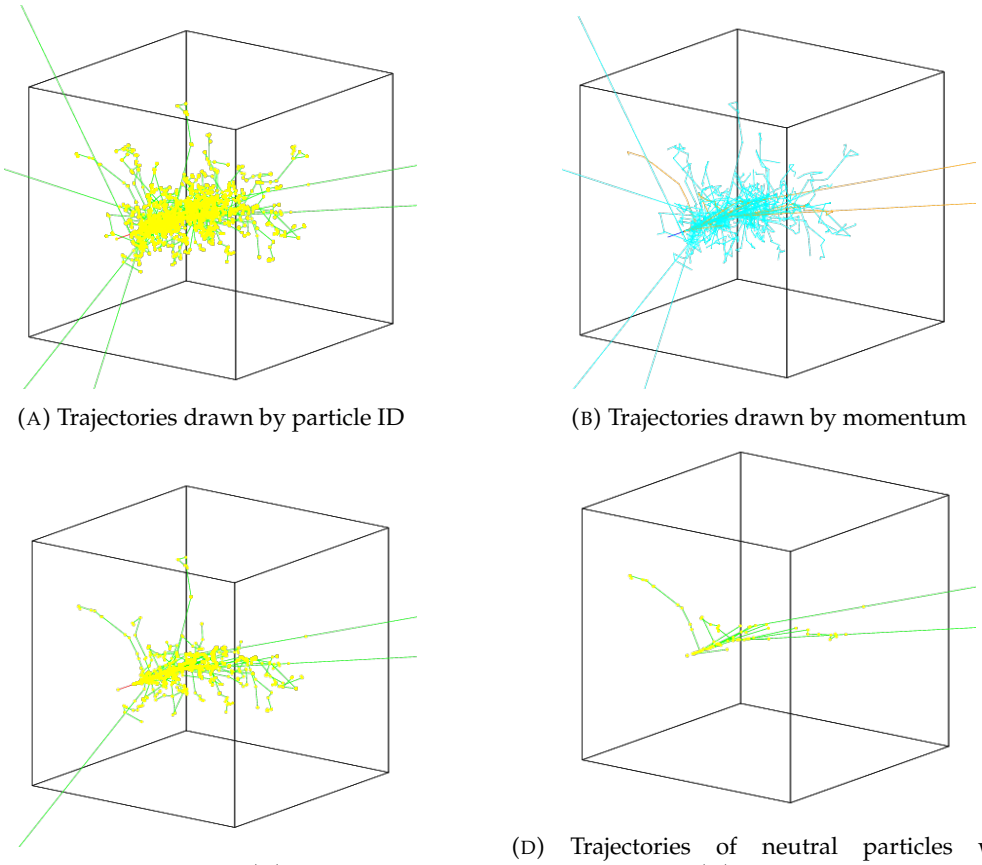


(E) Downstream viewpoint with the LHCb detector description rendered with DETDESC



(F) Downstream viewpoint with the LHCb detector description rendered with DD4HEP (simulation status as of August 2022).

FIGURE 3.16: Visualization of simple volumes with EXTERNALDETECTOR in GAUSSINO and the LHCb detector with either the DETDESC or the DD4HEP detector description toolkits in GAUSS-ON-GAUSSINO and the OPENGL visualization driver.



(A) Trajectories drawn by particle ID

(B) Trajectories drawn by momentum

(C) Trajectories with $|\vec{p}| > 10$ MeV

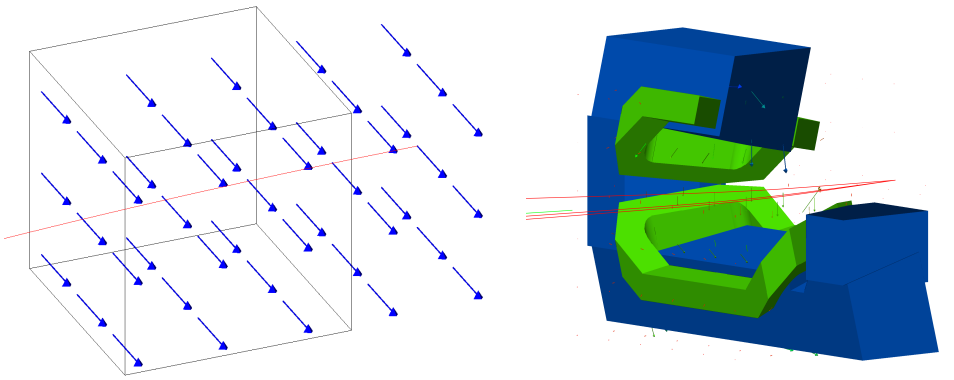
(D) Trajectories of neutral particles with $|\vec{p}| > 10$ MeV

FIGURE 3.17: Simulation of an electron hitting a lead cube target and the visualization of trajectories with different models and filters in GAUSSINO.

There are several ways to visualize the simulation data and GAUSSINO provides support for the majority of the options available in GEANT4. An example of the visualization of trajectories with different models and filters in GAUSSINO is shown in Figures 3.17. The user can choose to draw trajectories by particle ID, momentum, or any other attribute, and filter them by momentum, charge, pseudorapidity or any other attribute. Custom trajectory models and filters can also be created. Additional options are passed as dictionaries.

Magnetic field visualization

In addition to the visualization of the geometry and simulation data, the visualization of vector fields with GEANT4 is also possible in GAUSSINO. The magnetic field visualization is particularly useful for understanding how trajectories bend in the detector. In order to activate the magnetic field visualization, the user should set the `MagneticField` property of the `GaussinoVisualization` to a dictionary with the desired properties.



(A) Uniform magnetic field in a cube rendered with the EXTERNALDETECTOR package in GAUSSINO standalone.

(B) The dipole magnet in the LHCb detector and the visualization of its magnetic field in GAUSS-ON-GAUSSINO.

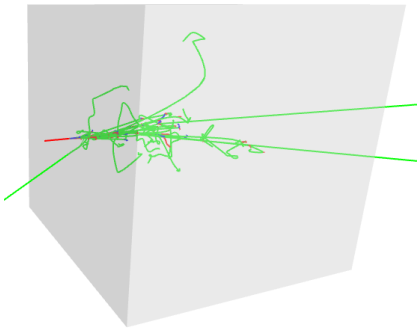
FIGURE 3.18: Visualization of the magnetic field in GAUSSINO.

3.4.2 PHOENIX visualization

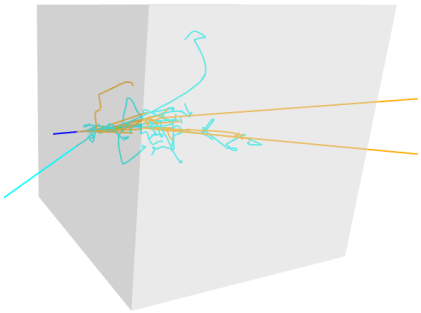
Having introduced the visualization of the geometry and simulation data with GEANT4, the next step was to explore the possibility of using the PHOENIX framework for visualization. PHOENIX [151] is an experiment-independent framework for event and geometry visualization, based on the THREE.JS [152] library and ANGULAR [153] framework. A key advantage of PHOENIX is that it enables direct comparisons between Monte Carlo (MC) simulated events and their reconstructed counterparts. It was designed to be used in a web browser, and it provides a user-friendly interface for the visualization of the detector geometry. In LHCb, PHOENIX has successfully been used for the visualization of the reconstructed data, in the form of an event display [154], and the possibility of visualizing the simulated data was yet to be explored [149].

Since it is a web-based framework, PHOENIX can only be used as an external visualization tool, based on the data produced by the simulation. The simulated data can be exported in JSON format, which is then used as input for the PHOENIX visualization. On the other hand, the geometry can be exported to GDML format with GAUSS-ON-GAUSSINO, and adapted to the format required by PHOENIX.

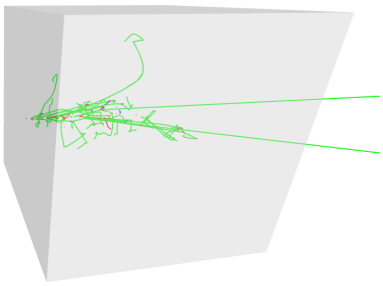
The user can choose to draw trajectories based on different properties, filter them by different attributes, and even create custom trajectory models. The only thing that changes is the fact that there are no drivers to choose from, as the visualization is done in the web browser. In the PHOENIX visualization, each trajectory, is a JSON object encompassing attributes necessary for rendering, such as the 'pos' attribute, which is an array detailing all points traversed by the particle. Additional information included covers the trajectory's initial momentum, transverse momentum, pseudorapidity, charge, and the designated color for depiction. PHOENIX supports various trajectory visualization models like `drawByCharge`, `drawByParticleID`, `drawByMomentum`, and `drawByKineticEnergy`, mirroring the GEANT4 features, thus eliminating the need for users to configure framework-specific settings. These models facilitate the selection of visualization preferences through a dedicated interface in GAUSSINO. Furthermore, PHOENIX allows for the filtering of trajectories, using criteria such as transverse momentum and pseudorapidity, although these filters require both minimum and maximum values.



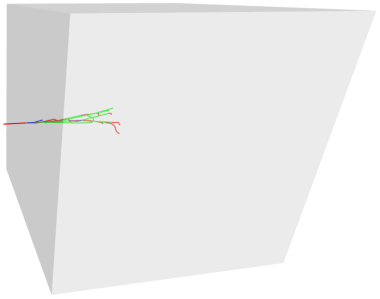
(A) Trajectories drawn by particle ID



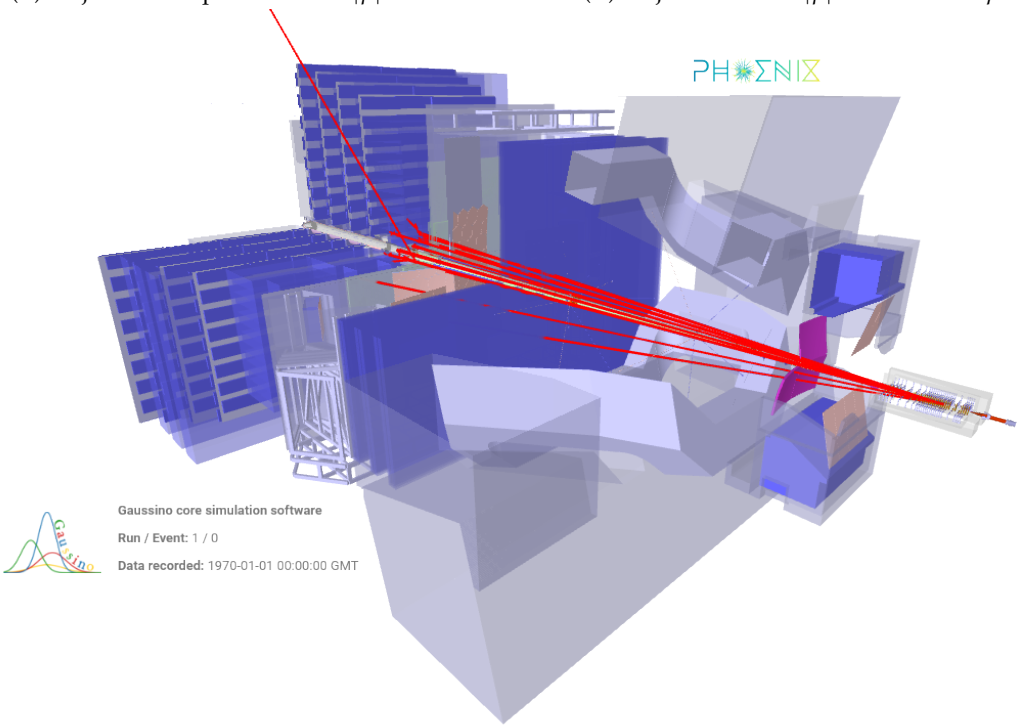
(B) Trajectories drawn by momentum



(C) Trajectories of particles with $|\vec{p}| > 25$ MeV



(D) Trajectories with $|\vec{p}| > 1$ MeV and $\eta < 3$



(E) Visualization of the LHCb detector geometry in PHOENIX.

FIGURE 3.19: Examples of trajectory filtering in Phoenix visualization.

3.5 Documentation

Both GAUSSINO and GAUSS-ON-GAUSSINO frameworks have been documented with the help of the SPHINX documentation generator [155]. Two websites have been created [156, 157], one for each framework. Each website contains installation instructions, simple examples, and detailed descriptions of some of the configuration options. The landing page of the GAUSSINO documentation website is shown in Figure 3.20, while the landing page of the GAUSS-ON-GAUSSINO documentation website is shown in Figure 3.21. The documentation is automatically generated from the framework source code, and each release has its own documentation version.

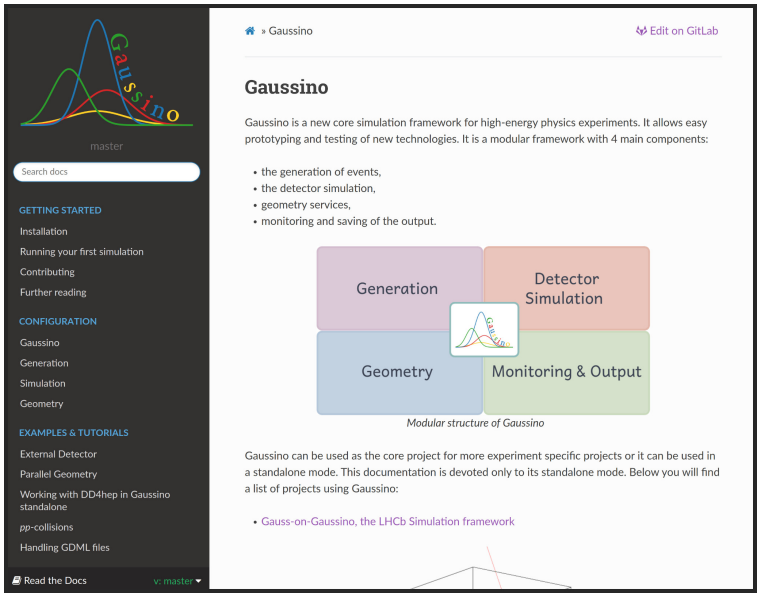


FIGURE 3.20: The GAUSSINO documentation website.

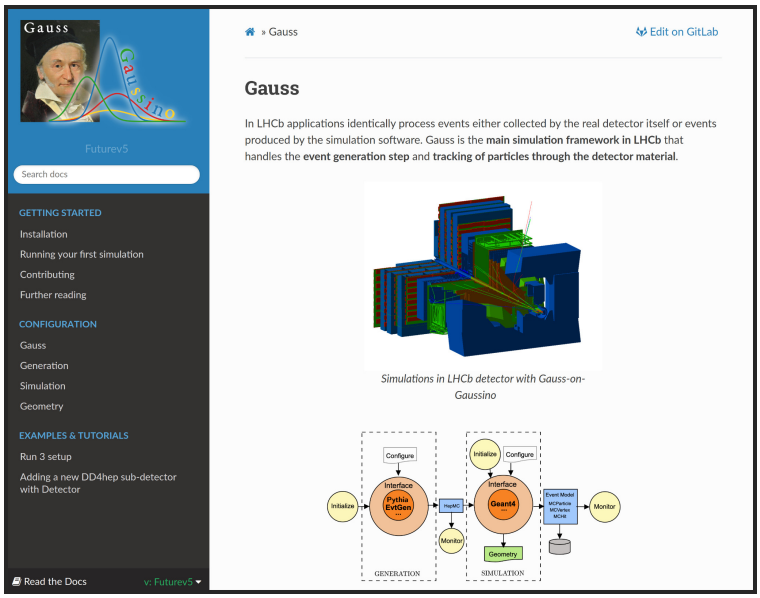


FIGURE 3.21: The GAUSS-ON-GAUSSINO documentation website.

4

Machine learning and simulations

Experiments in high-energy physics rely heavily on efficient computing systems to process the vast amounts of data produced by the detectors and simulations. As outlined in Chapters 1 and 3, the traditional approach to data processing is becoming increasingly challenging due to the growing complexity of the detectors and the increasing amount of data produced by the experiments. This is especially visible in the case of the LHCb experiment, which is expected to face one of the most challenging trigger rates among the LHC experiments in the upcoming runs (see Figure 1.4).

Given these challenges, machine learning (ML) techniques arise as an effective solution due to their ability to learn the underlying patterns and correlations in the data and produce the results efficiently. ML methods have already been widely used in high-energy physics for a variety of tasks, such as data analysis, event reconstruction, and Monte Carlo simulations. The path to the adoption of ML in high-energy physics is usually not straightforward, as the experiments require a relatively high level of precision, and at the same time, the algorithms have to be able to process vast amounts of data in a short time. Moreover, most of the ML tools and libraries used for training and inference are still being developed. In this chapter, three selected applications of ML are presented:

- the integration of ML libraries for running inference in the GAUSSINO simulation framework,
- the use of the Generative AI for generic calorimeter fast simulations in GAUSSINO, and its production-ready implementation in the simulation framework of the LHCb experiment: GAUSS-ON-GAUSSINO,
- and the application of ML-based object detection algorithms in the cluster reconstruction of the LHCb electromagnetic calorimeter.

4.1 Interfacing machine learning libraries in the simulation framework

4.1.1 Available backends

Introducing machine learning models in the simulation framework requires an efficient way to interface with the external libraries to be able to:

- load the model at the initialization stage (the model could be written in a different language, e.g. Python),
- perform the inference at the event processing stage.

In order to make this happen, the ML backend has to be written in C++, as the rest of the simulation framework. The multithreading environment should be taken into account, as the simulation framework is designed to run on multiple threads. Moreover, different random seeds should be set in threads in a way that allows for producing stable and reproducible results. Finally, the interface itself should be relatively simple and flexible to use, so that the user can easily switch between different models and libraries, ideally without the need to recompile the simulation framework.

Two main libraries have been considered in this work: C++ APIs for PyTorch [158] and ONNXRuntime [159]. There are also other libraries available including Tensorflow [160], Caffe [161], and others, which might be considered in the future. PyTorch C++ API consists of 5 main components:

- `ATen` the tensor library,
- `Autograd` the automatic differentiation library,
- `C++ Frontend`, which includes high-level constructs for training and evaluation,
- `TorchScript` the JIT compiler and interpreter,
- `C++ Extensions` with custom C++ and CUDA* routines.

`ATen` can be used in the simulation framework to convert the input data to the tensor format, which is the main data structure used in PyTorch. The `C++ Frontend` can be used to load the model and perform the inference. `TorchScript` can be used to convert the model to the intermediate representation, which can be then loaded in the simulation framework.

On the other hand, the ONNXRuntime was designed to be a cross-platform ML model accelerator with flexible interface to hardware-specific libraries. ONNX format is an open format for representing deep learning models, which allows for interoperability between different frameworks.

4.1.2 Integration in GAUSSINO

The integration of the libraries took place in the GAUSSINO framework, however, the interface was designed to be as generic as possible, so that it can be easily ported outside of the framework. Each interface consists of two main components:

*CUDA is a parallel computing platform enabling GPU acceleration for tasks.

- the `ModelServer` that inherits from `ModelServerBase` and depends only on the components provided by the external library,
- and `ModelServerSvc`, which inherits from GAUDI's `Service`, as well as the `IModelServerSvc` that acts as a wrapper around the `ModelServer`.

Because of the fact that the `ModelServer` is a pure C++ class, it can be easily tested in a standalone environment, without the need to run the simulation framework. The `ModelServer` exposes the main `evaluate` method, which is used to perform the inference on the loaded model. The `ModelServerSvc` is then used to provide the access to the `ModelServer` in the algorithm, as well as to give the possibility to steer the configuration of the model and the library via PYTHON configuration files. Moreover, `Service` is an entity that lasts throughout the whole run of the simulation framework, so the model is loaded only once at the initialization stage, and then the inference can be performed by multiple algorithms.

Both inputs and outputs of the `evaluate` are templated, which allows for a flexible way of passing the data to and from the model. The `evaluate` method is then used to perform the inference on the loaded model. The input data is passed to the model in the form of the tensor, which is the main data structure used in PyTorch and ONNXRuntime. If the model accepts a different type of input or different number of inputs, and the automatic type deduction is not possible, an error is thrown.

4.1.3 Performance tests

Both PyTorch and ONNXRuntime allow for setting the number of threads used for the inference. There are two types of parallelisations that can be used:

- *inter-op parallelism*, which allows for parallelizing the tasks that are forked within the application process,
- and *intra-op parallelism* that can be used to speed up element-wise operations on large tensors, etc.

The performance of the PyTorch and ONNX backends was tested in GAUSSINO by running the inference on a very simple model to measure the the actual performance of the infrastructure, not limited by the complexity of the model or particle transport. The model used for the tests consisted of a single matrix multiplication operation: $C = AB$ where $\{A, B, C\} \in \mathbb{R}^{1024\times1024}$. The operation was repeated 1000 times with different numbers of inter-op and intra-op threads. The tests were performed with a single machine equipped with two Intel Xeon E5-2630 v4 CPUs with 10 cores each and hyperthreading enabled, which gives a total of 40 threads. The results of the tests in terms of the total throughput ratio and the total virtual memory ratio are presented in Figures 4.1 and 4.2, respectively. In these tests, the number of the inter-op threads is always the same as the number of GAUDI threads to ensure the same number of events are processed concurrently. Additional, more detailed results are presented in Appendix C.

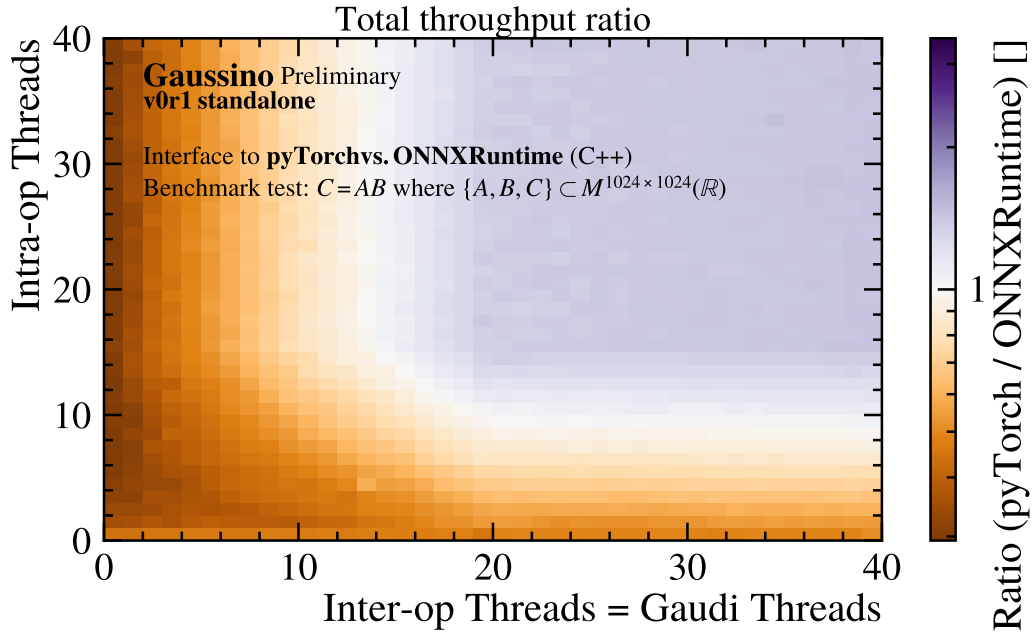


FIGURE 4.1: Total throughput ratio for the PyTorch and ONNX backends in GAUSSINO with different numbers of inter-op threads and intra-op threads.

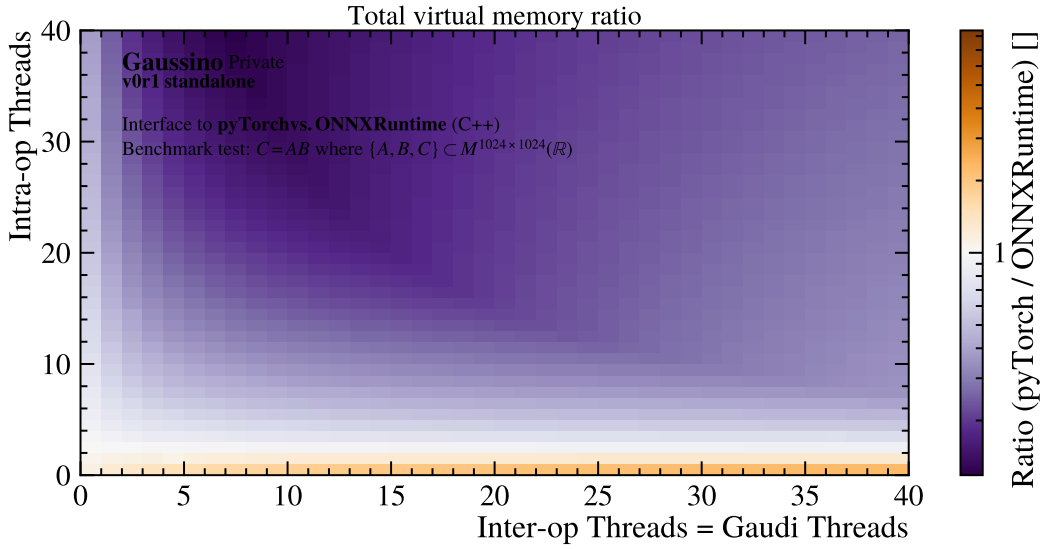
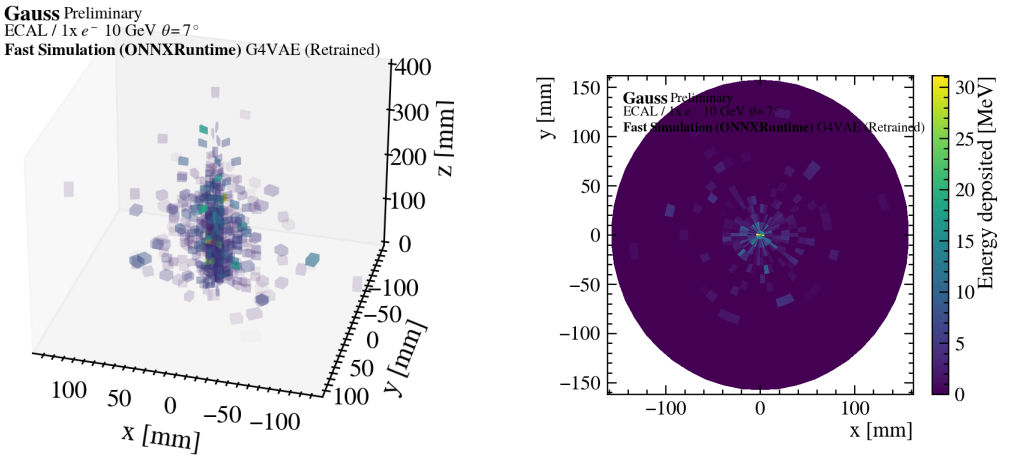


FIGURE 4.2: Total virtual memory ratio for the PyTorch and ONNX backends in GAUSSINO with different numbers of inter-op threads and intra-op threads.

4.2 Generative AI for calorimeter fast simulations

Generative AI [162] models have emerged as a promising solution to the computational challenges faced by fast simulations. These models can learn the underlying patterns and correlations in the data, enabling them to generate realistic simulations with significantly reduced computational costs. The models can be divided into two main categories: fully generative models and refinement techniques. Fully generative models replace classical simulation engines by taking generated particle data or random noise as input, while refinement techniques enhance the quality of simulated events by taking lower-quality simulations as input. State-of-the-art generative models include Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Normalizing Flows. Some of them are already being used in HEP experiments in production [134, 163, 164]. On the other hand, newer architectures like Transformer-based models [165] and Diffusion models [166] have shown promise in producing more accurate simulations [167–169].



(A) 3D visualization of the virtual energy deposits in the calorimeter.

(B) Projection of the virtual energy deposits in the calorimeter onto the x-y plane.

FIGURE 4.3: Visualization [134] of the virtual energy deposits generated by the modified VAE model in place of the detailed GEANT4 simulation.

ML models, once tested, have to be integrated into the simulation framework of the experiment. This process can be complex and time-consuming, as the models need to be adapted to the specific environment of the simulation framework that includes complex geometries, demanding multithreading, etc. GAUSSINO with its modular design is an ideal platform for testing and integrating generic generative AI models into the simulation framework.

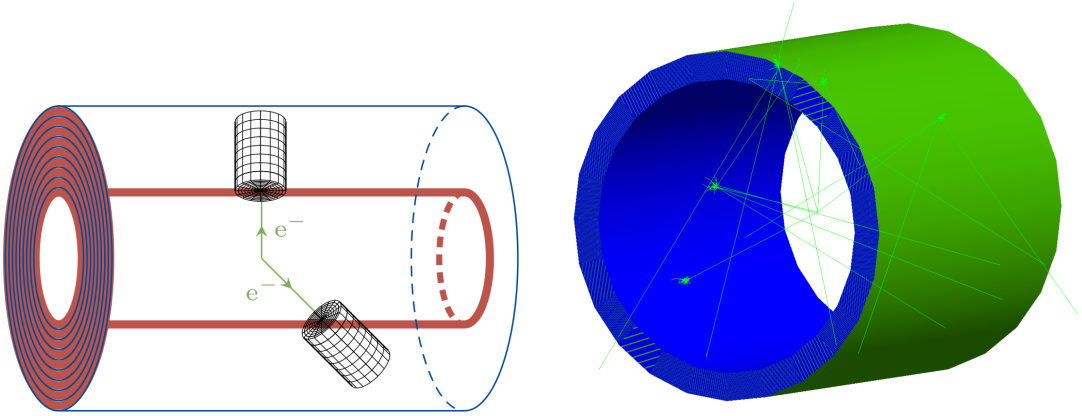
4.2.1 Generic calorimeter fast simulations in GAUSSINO

GAUSSINO with its ability to run standalone simulations, and at the same time being the building block of the simulation framework of the LHCb experiment [132–134], is an ideal platform for testing and integrating generic generative AI models into the LHCb simulation framework. The models can be trained on generic datasets, and

then adapted to the specific requirements of any experiment's simulation framework built on top of GAUSSINO.

CALOCHALLENGE in GAUSSINO

CALOCHALLENGE [170] is the first community-wide challenge for the development of fast and accurate calorimeter simulations. It provides three datasets, each with a different level of complexity. The setup for each of the datasets is very similar: particles are generated in the center of a cylindrical calorimeter, and the energy deposits are recorded in virtual concentric cylinders acting as detectors. The detectors are created on the fly, along the direction of the propagated particle in order to encapsulate the energy deposits. Each detector is segmented along its axial coordinate z, as well as the radial r and azimuthal ϕ variables. Visualization of the virtual energy deposits generated in the calorimeter is presented in Figure 4.4a.



(A) Visualization of the how the energy deposits are generated in CALOCHALLENGE datasets. The image is taken from [170]

(B) Visualization of the CALOCHALLENGE-comptible setup in GAUSSINO consisting of a cylindrical calorimeter and a particle gun located in the center.

FIGURE 4.4: CALOCHALLENGE setup for generating generic calorimeter training datasets.

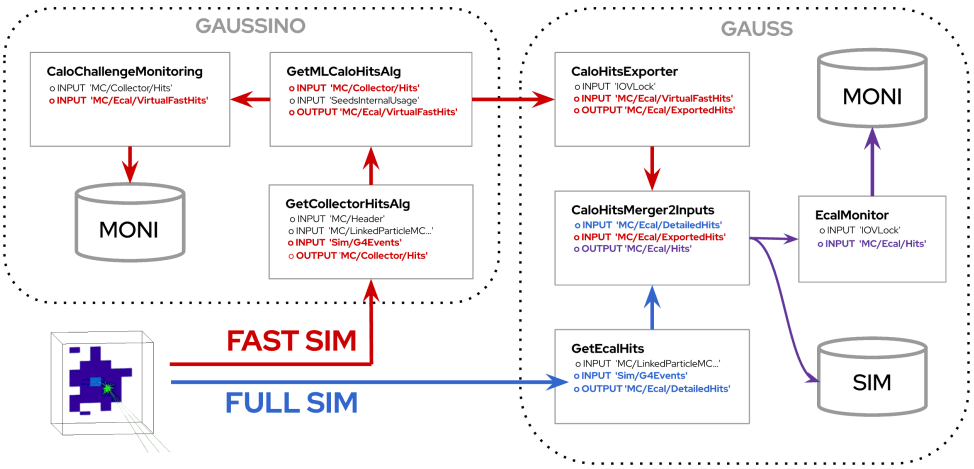


FIGURE 4.5: Data flow in the hybrid simulation setup in GAUSSINO. The production of ML-based data components (red arrows) is integrated with the traditional simulation chain (blue arrows).

The CALOCHALLENGE setup, although very simple in its nature, requires a bit of work to be adapted to the GAUSSINO simulation framework. The selection of particles that are fast simulated in the calorimeter is performed with the fast simulation interface described in Section 3.3.3. At inference time, the particles are stopped just in front of the calorimeter. GEANT4 performs then the rest of the simulation within the `GiGaAlg` algorithm as explained in Section 3.3.2. All the information about the particles is stored within the `CollectorHits`. They are then retrieved from `G4Events` object in `GetCollectorHitsAlg` algorithm that converts them into the hit format compatible with the event model used in GAUSSINO. The collector hits are then passed to the `GetMLCaloHitsAlg` that performs the inference on the ML model with the collector hits as input. The inference is performed using the interface to ML-backends introduced in Section 4.1.

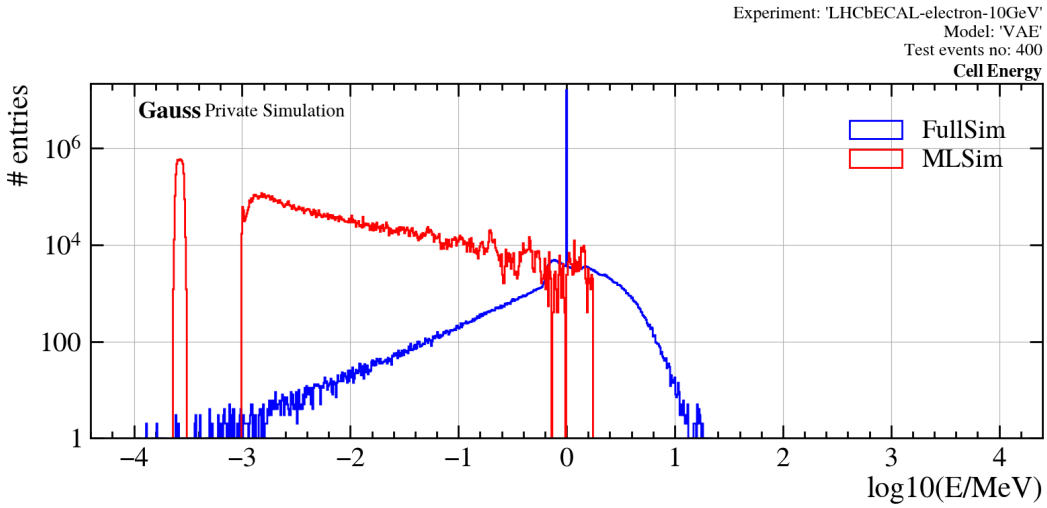


FIGURE 4.6: Energy distribution of a pure VAE model trained on the CALOCHALLENGE-compatible dataset produced in GAUSSINO.

Variational Autoencoder with Profiles

Variational Autoencoders (VAEs) [171] have been used as the first model in the CALOCHALLENGE to generate the energy deposits in the calorimeter. They are similar to pure Autoencoders, in the sense that they consist of two main components: the *encoder* and the *decoder*. The encoder takes the input data and maps it to the latent space (i.e. space containing learnt properties, cf. Figure 4.7 in pink), while the decoder takes the latent space and maps it back to the input space. In VAEs, the encoder and decoder are connected through a probabilistic latent space, i.e. the latent space is sampled from a probability distribution (Gaussian). The encoder and the decoder are trained simultaneously, usually by applying the *reparametrization trick*, which allows for backpropagation through the sampling process. At inference time, only the decoder is used to generate the data.

VAEs are trained in an unsupervised manner, i.e. they do not require labeled data. The virtual cylinders with energy deposits are fed to the model as input, and the model is trained to reconstruct the same deposits. Additional information about the particle is also fed to the model: the type encoded as a one-hot vector, the energy, and azimuthal θ and polar angles φ at which the particle entered the calorimeter.

VAEs are known to have problems in generating sparse data, which is a typical case for calorimeter simulations. Figure 4.6 shows the energy distribution of a

pure VAE model trained on the CALOCHALLENGE-compatible dataset produced in GAUSSINO. The energy distribution is not accurate, and the model smears the energy deposit over the whole cylinder.

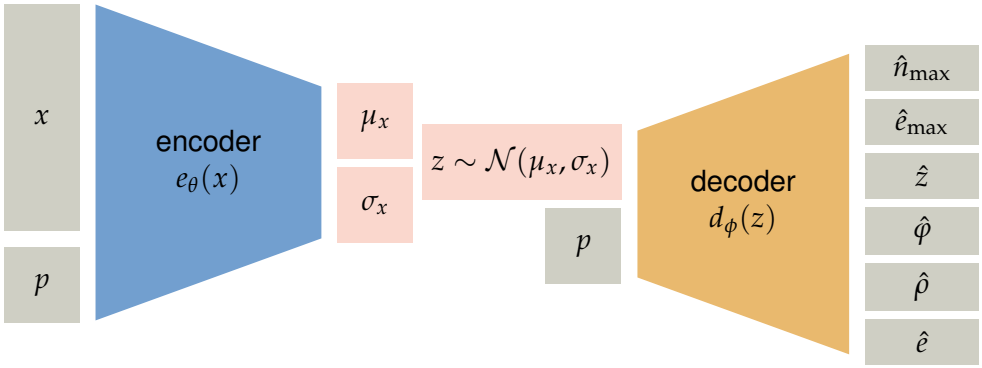


FIGURE 4.7: Architecture of a modified VAE model (VAEWithProfiles) used for the calorimeter fast simulations in GAUSSINO.

In order to improve the quality of the generated energy deposits, a modified VAE model (VAEWithProfiles) has been proposed. The architecture of the model is presented in Figure 4.7. Instead of trying to generate the energy deposits directly, the model is trained to generate the profiles of the energy deposits: ρ, φ, z, as well as the energy distribution e. The total energy deposit $e_{\max}$ and total number of hits $n_{\max}$ are also added in order to stabilize the postprocessing step, in which the energy deposits are reconstructed from the profiles. The loss function is a sum of binary cross-entropy (BCE) terms for the reconstructed profiles and variables, and KL-divergence term for the similarity of the latent space $z \sim \mathcal{N}(\mu, \sigma)$ to the Gaussian distribution $\mathcal{N}(0, 1)$, and can be written as

$$\begin{aligned}\mathcal{L}(n_{\max}, e_{\max}, \rho, \varphi, z, e) = \mathcal{L}_{\text{BCE}}(n_{\max}, \hat{n}_{\max}) + \mathcal{L}_{\text{BCE}}(e_{\max}, \hat{e}_{\max}) + \mathcal{L}_{\text{BCE}}(\rho_i, \hat{\rho}_i) \\ + \mathcal{L}_{\text{BCE}}(\varphi, \hat{\varphi}) + \mathcal{L}_{\text{BCE}}(z, \hat{z}) + \mathcal{L}_{\text{BCE}}(e, \hat{e}) + \mathcal{L}_{\text{KL}}(\mu, \sigma).\end{aligned} \tag{4.1}$$

The sampling process is not ideal because it assumes that all the profiles are independent from each other, which is not the case, but it allowed for a significant improvement in the quality of the generated energy deposits, as well as in the speed of the training process. The differences in the energy distributions can be neglected if the granularity of the target calorimeter is much lower than the granularity of the cylinder used in the CALOCHALLENGE datasets, which is the case for the LHCb calorimeter. The energy distribution of the model trained on the same dataset is presented in Figure 4.9. A 3D and 2D visualizations of the energy deposits generated by the model are presented in Figure 4.3.

4.2.2 Adaptation to the LHCb calorimeter in GAUSS

Additional steps are required to adapt the setup presented in the previous section to be functional in production simulations in LHCb. The electromagnetic calorimeter in LHCb is a planar calorimeter, and that means that a dedicated collector plane can be used to store information about the incident particles just in front of the calorimeter. Unfortunately, the calorimeter is not uniform and there is some passive material in the upstream area of the calorimeter, such as a so-called *beam plug* and a neutron shielding wall that can affect the energy deposits. Moreover, the calorimeter

is slightly tilted around the x axis of the LHCb reference frame, and the inner hole in the calorimeter introduces additional complexity. The configuration of this setup is presented in Figure 4.8. The collector plane is placed just in front of the neutron shielding (blue line). Since the position of the collector plane is roughly 500 mm away from the front face of the calorimeter, the particles are virtually transported to the beginning of the sensitive area of the calorimeter (red box) by calculating the distance x to the collector plane:

$$x = \frac{|x_{coll}|}{\hat{n} \cdot \hat{v}}, \tag{4.2}$$

where:

- x_{coll} is the shortest distance to the collector plane from the initial point P,
- $\hat{n}$ is the normal vector of collector plane,
- $\hat{v}$ is a vector describing the direction of a particle at point P.

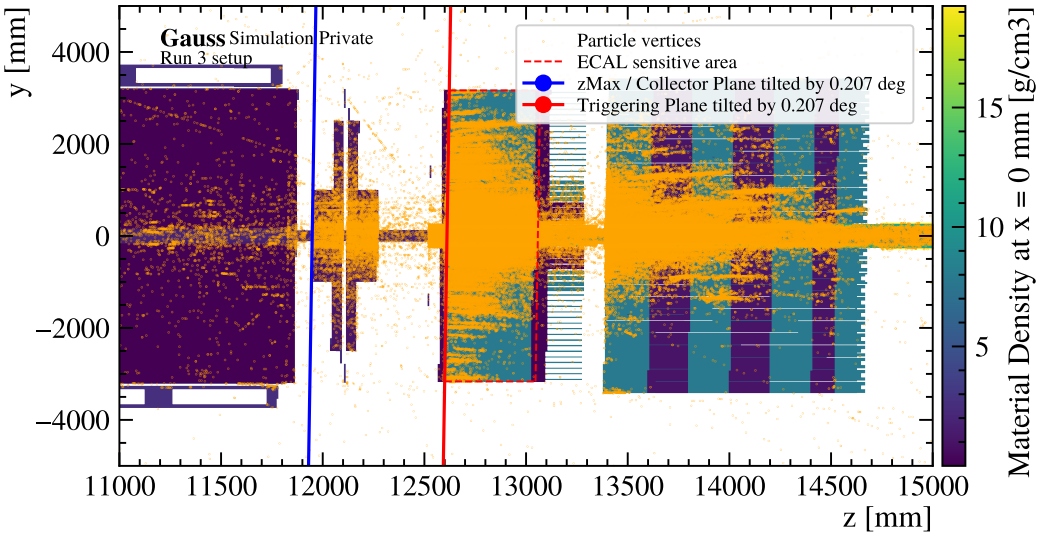


FIGURE 4.8: Visualization [134] of how the ML-based simulations can be implemented in production-ready simulations in LHCb for calorimeter showers. The particles of interest are stopped before the calorimeter (red line), and the ML-based simulations are used to simulate the energy deposits produced by particles in the sensitive area of the calorimeter (red, dotted line).

The right side of Figure 4.5 shows what happens in GAUSS-ON-GAUSSINO once the ML-based hits (`VirtualFastHits`) are generated in GAUSSINO. `VirtualFastHits` are energy deposits generated by a ML model in fictitious cylinders that have to be then mapped to the real calorimeter geometry. `CaloHitsExporter` is an algorithm that takes the `VirtualFastHits` as an input and created the `ExportedHits`, which are linked to physical cells in the calorimeter. The `ExportedHits` are then passed to the `CaloHitsMerger` algorithm that merges the `ExportedHits` with the `DetailedHits` produced by the Geant4 simulation. The output hits of the `CaloHitsMerger` are persisted and can be used in the subsequent steps of the simulation chain.

Training

The training of the model was performed on a total of 2 million events, with 1.5 million events used for training and 0.5 million for validation. Additional 4 thousand events were used for testing. The particles were generated with a `MomentumRange` particle gun. The dataset consists of 4 equal subsets, each with a different energy range: 10-100 MeV, 100-1000 MeV, 1-10 GeV, and 10-1000 GeV. Each subset consists of an equal number of particles with different types: electrons and photons. The training was performed on a single NVIDIA A100 GPU, and the training time was approximately 8 hours. The improved energy distribution obtained with the VAEWithProfiles model is presented in Figure 4.9. Total energy distribution is presented in Figure 4.10.

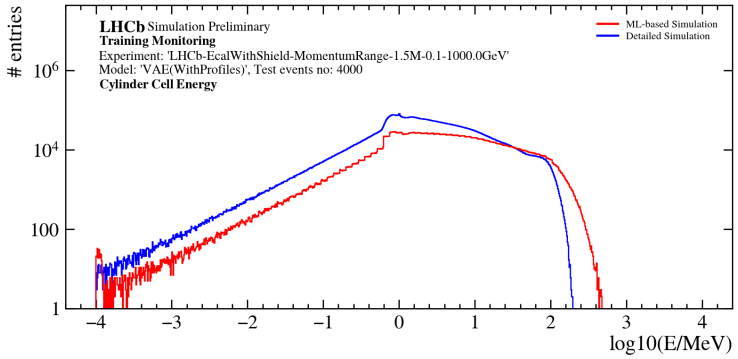


FIGURE 4.9: Energy deposit distribution of a modified VAE model (VAEWithProfiles) trained on the CALOCHALLENGE-compatible dataset produced in GAUSSINO.

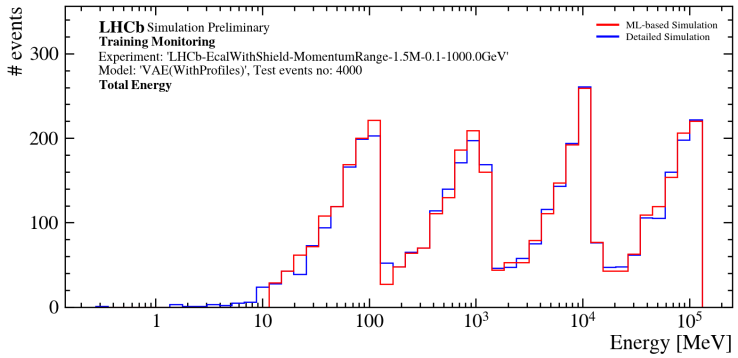


FIGURE 4.10: Total energy deposit distribution of a modified VAE model (VAEWithProfiles) trained on the CALOCHALLENGE-compatible dataset produced in GAUSSINO.

Validation

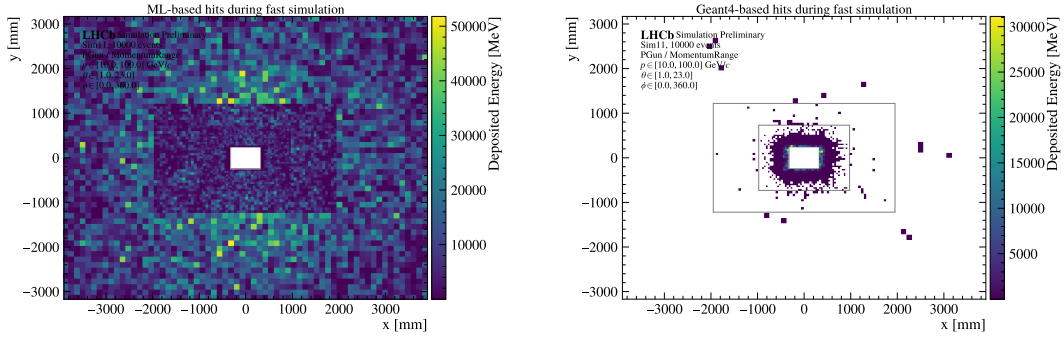
The performance of the model was validated with the following datasets:

- 4 datasets produced with `MomentumRange` particle gun, each with 1000 events, and energy ranges: 10-100 MeV, 100-1000 MeV, 1-10 GeV, and 10-1000 GeV,
- 10000 events produced with `MinimumBias` setup,

- dedicated physics channels, which are discussed in detail in Chapter 5.

The `MomentumRange` particle gun was used to scan the whole phase space of the calorimeter and check if the model is robust enough. `MinimumBias`, on the other hand, was used to check the performance of the models on more realistic data, similar to the one that is produced in real collisions. Comparison of the total simulation throughput with and without the ML-based simulation is presented in Figure 4.12b and is estimated to be up to 2 orders of magnitude faster. Figures 4.11a and 4.11b represent the remaining part of particles that still remain simulated with GEANT4 due to the difficulties of running ML-based simulation around the inner hole and with other particle species. As it turns out, the ML-based solution is able to capture around 40% of all the energy deposits in `MinimumBias` events due to relatively large input of hadrons and particles showers generated around the inner hole.

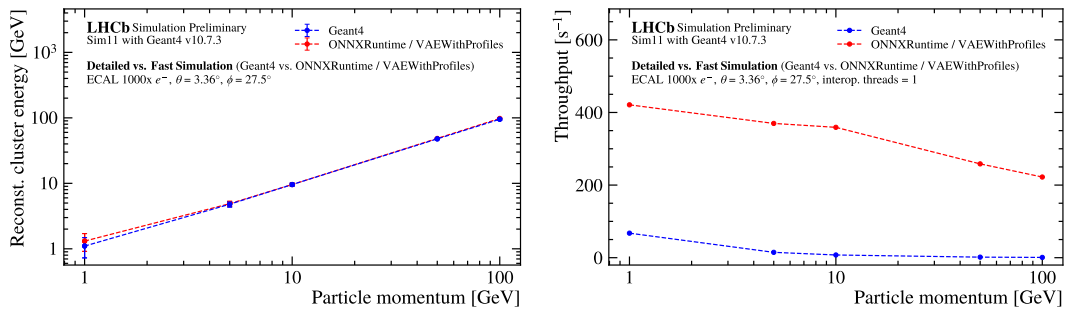
After the simulation, the whole reconstruction chain was run, and the performance of the model was compared using reconstructed variables. The reconstructed energy of the particles is presented in Figure 4.12a and is in good agreement with the reconstructed energy from the full Geant4 simulation: the difference is between 1% and 4% depending on the energy of the particle. Additional performance plots of the ML-based fast simulation in GAUSSINO and GAUSS-ON-GAUSSINO were placed in Appendix D.



(A) ML-based simulation that replaces the standard detailed simulation with Geant4.

(B) Geant4-based detailed simulation that still remains due to the difficulties of running ML-based simulation around the beam hole.

FIGURE 4.11: Energy deposits in the LHCb electromagnetic calorimeter produced by a particle gun during the fast simulation.



(A) Reconstructed energy of the particles.

(B) Total simulation throughput.

FIGURE 4.12: Performance comparison [134] between the full GEANT4 sim. and the ML-based sim. in the LHCb calorimeter.

4.3 Cluster energy reconstruction in the LHCb electromagnetic calorimeter

Reconstructing data in calorimeters typically involves solving a clustering problem, where energy deposits left by secondary particles are grouped together. These clusters of energy deposits are then used to infer more complex features in the reconstruction process, such as the energy of particles as explained in Section 1.2. The data reconstruction algorithms in calorimeters significantly impact the overall reconstruction time. In LHCb, the calorimeters account for approximately 25% of the total HLT2 processing time [172]. CELLULAR AUTOMATON [173] and GRAPH CLUSTERING [174] are examples of the classical, unsupervised algorithms that have been used in LHCb up to Run 2, and for Run 3 respectively.

Recent advancements in the field of computer vision and the use of more advanced, machine-learning based algorithms show that there is still significant improvement possible. Object detection algorithms share many similarities with the algorithms used for clusterization in high-energy physics such that on an abstract level both problems can be analysed in a similar way. In particular, for planar calorimeters, such as those used in LHCb, it is possible to construct an image or a graph of the detector with hits as pixels or nodes. Recent results (as of 2020) show that it is possible to improve the cluster reconstruction algorithm using recent developments in computer vision. In particular, there is a lot of potential in the frameworks based on convolutional neural networks (CNNs)[175–182] and graph neural networks (GNNs)[181, 183–186]. In Section 4.3.1, the early feasibility studies of CNN-based cluster energy reconstruction in the LHCb electromagnetic calorimeter are presented. In the following Section 4.3.3, another approach in the preparation of the training datasets with GAUSSINO, the experiment-agnostic core simulation framework introduced in Section 3.1, is discussed.

4.3.1 Early feasibility studies with convolutional neural networks

Convolutional neural networks

Computer vision and high-energy particle detectors are similar in the sense that the reconstruction algorithms operate either on a finely grained set of pixels or detector hits, respectively, and then try to infer more complex properties from them. The simplest way to represent the data from the planar calorimeter is to construct an image with energy deposits as pixels. In this case, a natural candidate that deals with this kind of input is the neural network based on convolutional layers. It employs a *convolution* that acts on subsets of the image in order to give each pixel additional knowledge about its neighbourhood [187]. The operation can be written as:

$$s(t) = (x * w)(t) = \int x(a)w(t-a)\mathrm{d}a, \tag{4.3}$$

where x is the input and w is the *kernel*. The kernel slides over the image and convolves with the pixels within the *receptive field*. The output of the convolutional layer is called a *feature map*. The motivation for using convolutional layers is threefold. Firstly, thanks to the kernel usually having narrower support than the input, the number of parameters is significantly reduced. This is sometimes referred to as *sparse connectivity*. Secondly, *parameter sharing* is employed, which means that the same parameter is used for more than one function in the model. Finally, with

convolutional layers the *equivariance* to translation is achieved, which means that the output of the convolutional layer is invariant to the translation of the input.

In practice, convolutional neural networks consist of at least three stages: convolutional layers, non-linear activation functions and *pooling layers*. The non-linear activation functions are used to introduce non-linearity to the model, which is essential for the network to be able to approximate complex functions. The pooling layers are used to reduce the spatial dimensions of the input, which helps to reduce the number of parameters in the network. For example, *max pooling* takes the maximum value from the subset of the input, while *average pooling* takes the average value.

You Only Look Once (YOLO)

You Only Look Once (YOLO) [175, 176, 179, 181] is a state-of-the-art, real-time, one-stage detection system based on convolutional layers. It applies the neural network to the whole image at once. The image is divided into a grid $S \times S$ and each cell of that grid is assigned B bounding boxes. In the full YOLOv3 model, there are 3 grids with different granularities.

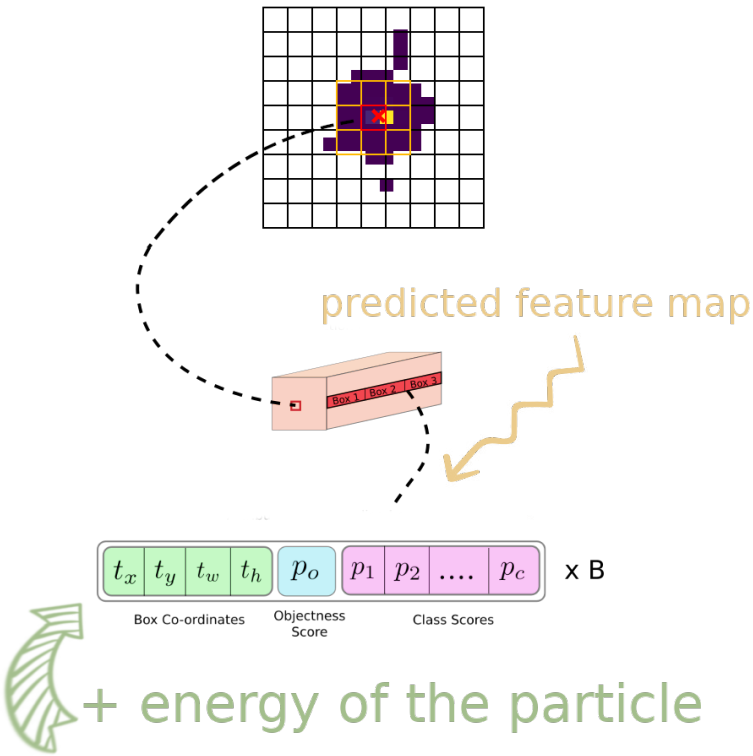


FIGURE 4.13: Graphical visualization of the feature map produced by the YOLO-like model for the cluster energy reconstruction.

At the training stage, the YOLOv3 model is trained on a set of images and the corresponding labels. The labels are transformed into the form of the bounding boxes and the class probabilities and attached to the grid cells. The backbone of the YOLOv3 model is the Darknet-53 architecture with 53 convolutional layers and 3 skip connections that allows to introduce grids of different granularities. At the end of the backbone, the feature map consists of bounding box priors assigned to the

grid cells. Additional refinement of the bounding box priors is performed according to the formula

$$\begin{aligned} b_x &= \sigma(t_x) + c_x, \\ b_y &= \sigma(t_y) + c_y, \\ b_w &= p_w e^{t_w}, \\ b_h &= p_h e^{t_h}, \end{aligned} \tag{4.4}$$

where:

- b_x, b_y are the coordinates of the center of the bounding box,
- b_w, b_h are the width and height of the bounding box,
- c_x, c_y are the offsets from the top left corner of the image,
- p_w, p_h are the width and height of the bounding box priors,
- t_x, t_y, t_w, t_h are the coordinates of the bounding box priors.

The loss function for the YOLOv3 model is a sum of minimum mean squared error and binary cross-entropy losses:

$$\begin{aligned} &L_{\text{YOLOv3}}(x, \hat{x}, y, \hat{y}, C, \hat{C}, p, \hat{p}) = \\ &\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ &+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\ &- \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i \log(\hat{C}_i) + (1 - C_i) \log(1 - \hat{C}_i)) \\ &- \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i \log(\hat{C}_i) + (1 - C_i) \log(1 - \hat{C}_i)) \\ &- \sum_{i=0}^{S^2} \mathbb{1}_{i}^{obj} \sum_{c \in classes} (p_i(c) \log(\hat{p}_i(c)) + (1 - p_i(c)) \log(1 - \hat{p}_i(c))). \end{aligned} \tag{4.5}$$

where:

- x_i, y_i are the coordinates of the centers of the bounding boxes,
- w_i, h_i are the width and height of the bounding boxes,
- C_i is the confidence score of whether the bounding box contains an object,
- p_i stands for the class probabilities,
- $\mathbb{1}_{ij}^{obj}$ and $\mathbb{1}_{ij}^{noobj}$ are the indicators of whether the bounding box contains an object or not.

At inference time, the model outputs the bounding boxes with the highest confidence scores. Since there are usually at least a few bounding boxes assigned to each cell, the *non-maximum suppression* (NMS) algorithm is used to remove the redundant bounding boxes. It is based on the intersection over union (IoU) metric,

which is defined as the ratio of the area of the intersection of the bounding boxes to the area of their union:

$$\text{IoU} = \frac{S(B_1 \cap B_2)}{S(B_1 \cup B_2)}. \tag{4.6}$$

The YOLO-like framework has been tested using the official Monte Carlo simulation data [181] for the LHCb calorimeter.

YOLO-like framework for the LHCb electromagnetic calorimeter

In order to apply the YOLO-like framework to the LHCb electromagnetic calorimeter, the energy deposits have been represented as pixels in the image. Since the granularity is not uniform, the image has been divided into a rectangular grid 384×312, with the cell size 2, 3, and 6 times smaller than the cell size of the inner, middle, and outer regions of the calorimeter, respectively. Moreover, with the help of three skip connections, the problem of hybrid granularities in calorimeters can be easily addressed by assigning bounding boxes with significantly different shapes to a grid that fits best their size. A graphical visualization of the feature map produced by the YOLO-like model for the cluster energy reconstruction is presented in Figure 4.13. The energy deposit has been added as one of the additional features in the bounding box candidates. Therefore, the loss function is a sum of the original YOLO loss and the mean squared error loss for the energy deposit E:

$$L_{\text{YOLOv3}}(x, \hat{x}, y, \hat{y}, C, \hat{C}, p, \hat{p}) + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (E_i - \hat{E}_i)^2. \tag{4.7}$$

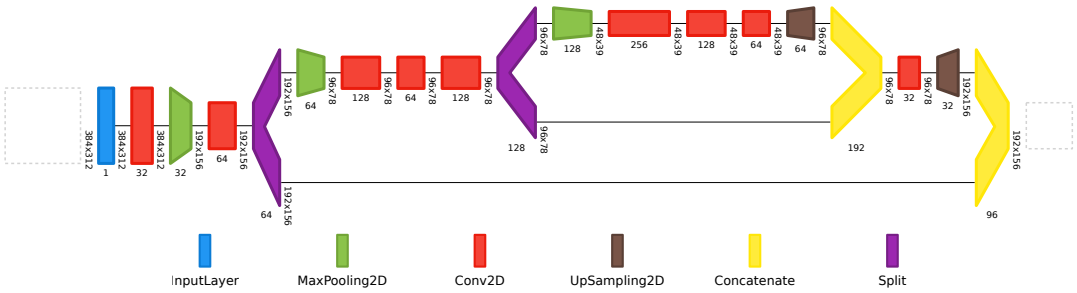


FIGURE 4.14: The backbone of the YOLO-like model for the LHCb calorimeter with three skip connections and rectangular image input.

For the training of the model, an official production with a dataset containing 10^5 minbias events was chosen. At that time the official algorithm used for cluster reconstruction was the CELLULAR AUTOMATON algorithm, and therefore the results for the current algorithm (GRAPH CLUSTERING) were not yet available. The sample had to be then exported from its native ROOT format to a more optimized TFRECORD[†] format. This allowed to prefetch the data during the training process and to avoid the bottleneck of the I/O operations and running out of memory. The training was performed on the Świerk Computer Center on nodes equipped with NVIDIA Tesla K80 GPUs with `MirroredStrategy`, which allows the training process to be distributed across multiple GPUs. The training was performed with the `Adam` optimizer with a learning rate of 10^{-4} and a batch size of 32. Additional callbacks were used to monitor the training process, such as the `ModelCheckpoint`

[†]Dedicated format for storing sequences of binary records.

callback to save the best model, the `EarlyStopping` callback to stop the training process if the validation loss does not improve for N epochs, and the `TensorBoard` callback to monitor the training process in real time.

At inference time, an additional matching algorithm on top of the regular NMS algorithm was used to match the predicted clusters with the MC truth. The idea was to estimate the number of ghost particles (false positives) and missed particles (false negatives) in the output of the model. This is a classical assignment problem that can be written as an optimization problem:

$$\begin{aligned} \min_{x} \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} = 1 \quad \forall i, \\ & \sum_{i=1}^{m} x_{ij} = 1 \quad \forall j, \\ & x_{ij} \geq 0. \end{aligned} \tag{4.8}$$

where c_{ij} is the distance between the predicted cluster i and the MC truth cluster j in 3D space (x, y and energy), and x_{ij} is the binary variable that indicates whether the predicted cluster i is matched with the MC truth cluster j. The Hungarian algorithm was used to solve this problem [181]. In addition to the matching algorithm, two additional constraints were added to prevent from matching too distant clusters:

- the centroid of the predicted cluster must be within twice the size of the predicted cluster from the MC truth cluster,
- the energy difference cannot differ by more than one order of magnitude.

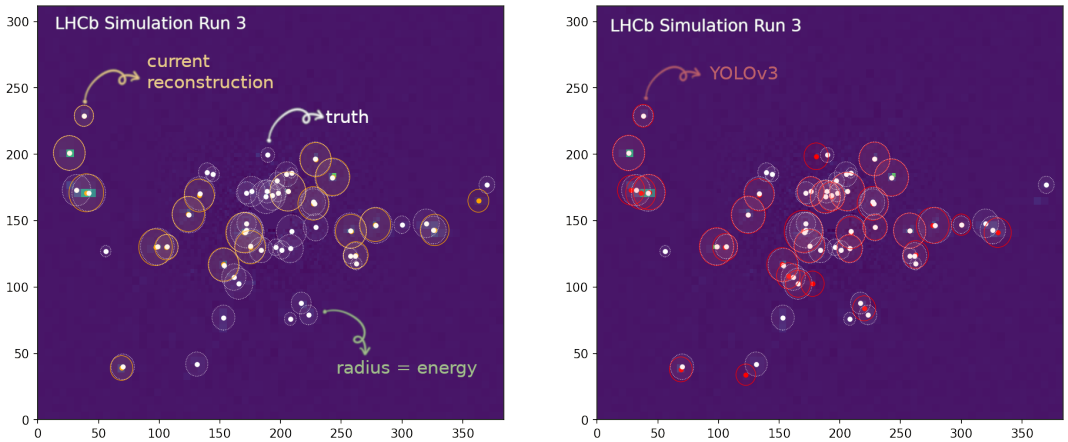


FIGURE 4.15: Comparison of the CELLULAR AUTOMATON and YOLO-like cluster reconstruction for a single event [181].

Around 2000 events were used at the testing stage and compared with the CELLULAR AUTOMATON algorithm. An exemplary event showing a comparison between the performance of the current and YOLO-like cluster reconstruction has been chosen from the subset of the preliminary results to show the potential of the framework and is presented in Figure 4.15. The number of clusters detected and the reconstructed energy as a ratio of the MC truth for the YOLO-like model

and CELLULAR AUTOMATON algorithm are presented in Table 4.1 and visualized in Figure 4.16. The missed rate and ghosts rate as a function of the MC truth energy for the YOLO-like model, as well as the energy resolution for the YOLO-like model and CELLULAR AUTOMATON algorithm are presented in Figure 4.17. The results show that the YOLO-like model is able to detect clusters with possibly higher efficiency than the CELLULAR AUTOMATON algorithm, however, the energy resolution is still not satisfactory. Moreover, high missed and ghost rates are observed for the YOLO-like model, which indicates that the model is not able to detect all the clusters and introduces many ghost particles.

	Clusters [%]		Energy [%]	
	YOLO-like	(current)	YOLO-like	(current)
≥1 GeV	97.6	70.1	95.7	84.6
≥2 GeV	97.8	83.6	95.7	90.5
≥3 GeV	97.6	90.7	95.2	94.3

TABLE 4.1: Reconstructed energy and number of clusters detected as a ratio to the MC truth for the YOLO-like model and CELLULAR AUTOMATON algorithm [181].

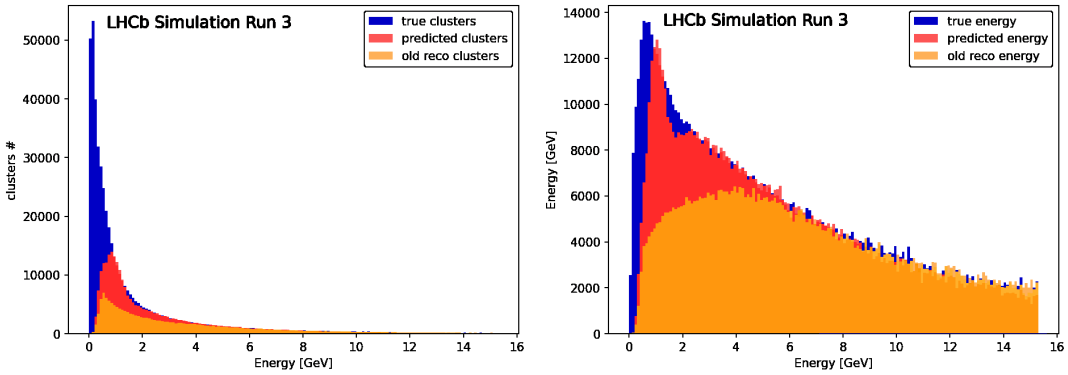


FIGURE 4.16: Number of clusters detected (left) and reconstructed energy (right) as a function of the MC truth energy for the YOLO-like model and CELLULAR AUTOMATON algorithm [181].

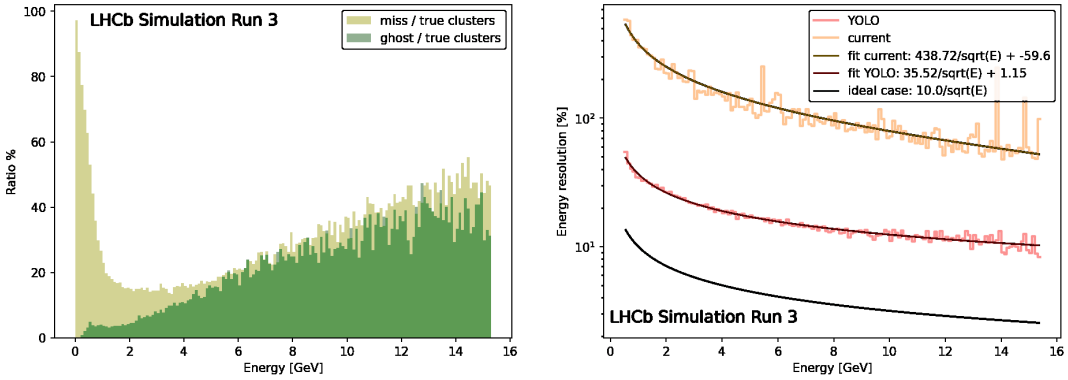


FIGURE 4.17: Missed rate and ghosts rate as a function of the MC truth energy for the YOLO-like model (left), as well as the energy resolution for the YOLO-like model and CELLULAR AUTOMATON algorithm (right) [181].

4.3.2 Limitations of the YOLO-like model

The high ghost and missed rates discussed in the previous section are not the only challenges that need to be addressed. Images generated based on the calorimeter output are usually not homogeneous enough and may consist of many sub-detectors with different granularities. Moreover, many neural networks with YOLO included, which use anchors for the bounding boxes, impose implicit constraints on the size of the objects. Showers do not have well-defined edges and hence the output of the calorimeter is usually a sparse matrix. In general, showers induced by the electromagnetic particles are highly concentrated around the entering point of the particle, however, the energy deposits caused by hadronic showers can be scattered all over the detector with different densities and magnitudes.

Another problem is the high overlap observed for Run 3 data, as shown in Figure 4.18. YOLO's performance is known to drop significantly in the case of small, grouped objects with high overlap. It is especially challenging in the inner region around the beam pipe.

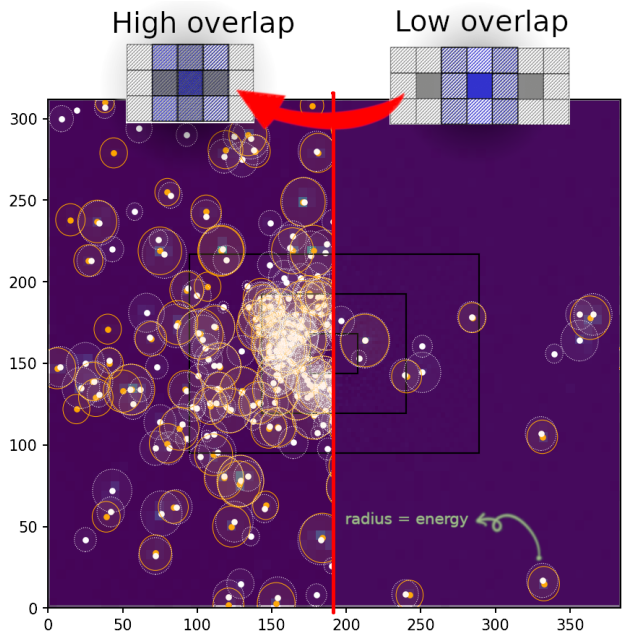


FIGURE 4.18: High overlap observed for Run 3 data.

Moreover, the information required to produce high-quality training datasets is not always available in the standard output file, as software applications in high-energy physics usually store only the minimum amount of information required for physics studies. For the training datasets, it is very important to turn off any unneeded optimization features and to give the possibility to gather information at any given place in the detector. For example, placing a virtual thin detector in front of the calorimeter could be used to collect precise information about incident particles. Another example is the fact in the LHCb simulation software, all hits generated by the particles traveling upstream from the hadronic calorimeter are assigned to the ancestor particle that entered the calorimeter, and therefore determining the size of the cluster is not straightforward. Both examples are illustrated in Figure 4.19.

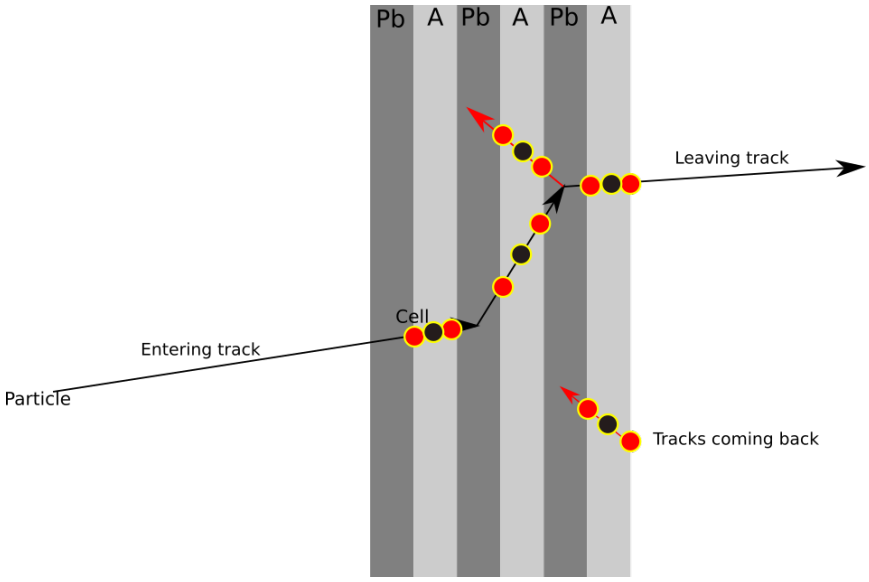
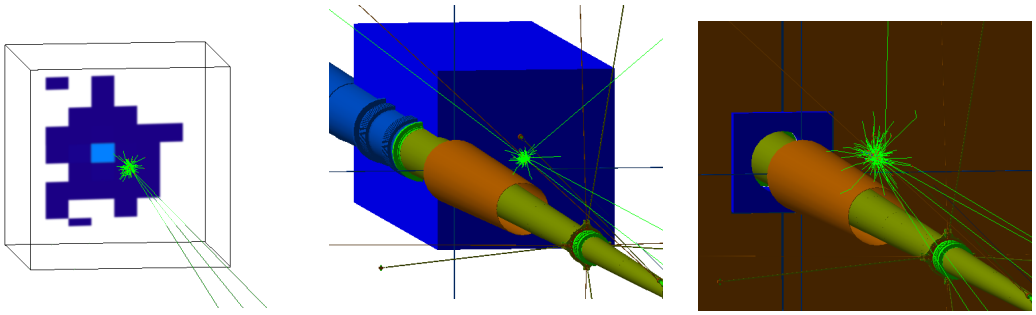


FIGURE 4.19: Illustration of a particle interaction within the LHCb electromagnetic calorimeter, showing the track of the particle as it enters the calorimeter from the simulation point of view. The alternating layers of lead (Pb) and scintillator (A) facilitate energy deposition and detection for cluster energy reconstruction.

4.3.3 Preparation of the training datasets with GAUSSINO

In order to address some of the challenges mentioned in the previous section, GAUSSINO with its custom simulation interface can be used to produce training datasets. The main advantage of this new framework is that it can be used to produce training datasets for very simple, toy models (Figure 4.20a), as well as very complex, full-scale high-energy experiments (Figure 4.20c). Moreover, a hybrid approach is also possible (Figure 4.20b), in which a toy model can be directly tested in the environment of the real detector. This gives the possibility to test the impact of the geometry and environment on the performance of the machine learning model in a seamless manner.

The performance of the YOLO-like model trained on a toy model placed inside the LHCb environment is presented in Figure 4.21a and 4.21b. The precision of the coordinates of the bounding boxes can be further improved by the use of the calorimeter with higher granularity. In GAUSS-ON-GAUSSINO, the same configuration with just a few modifications can be then used to test the neural network on the real LHCb electromagnetic calorimeter.

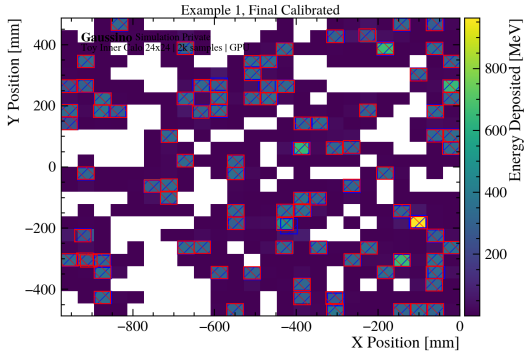


(A) Bare toy model

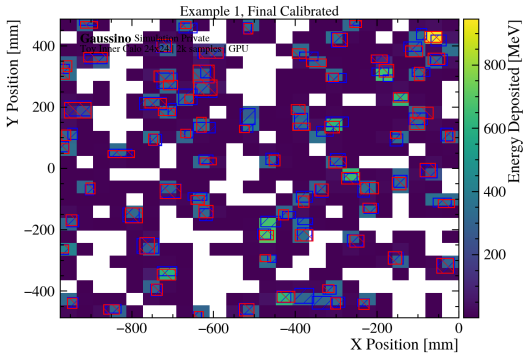
(B) Toy model in the environment of the experiment.

(C) Full-scale detector

FIGURE 4.20: Incremental approach for producing training datasets using Gaussino.



(A) Standard granularity used for getting the coordinates of the bounding boxes.



(B) The precision of the coordinates of the bounding boxes can be further improved by the use of the calorimeter with higher granularity.

FIGURE 4.21: Selected training datasets produced by Gaussino using a small, toy calorimeter inside the LHCb environment. Blue boxes represent true clusters as taken from the Monte Carlo simulation and red boxes represent reconstructed clusters by the YOLO-like model.

5

Physics validation

One of the final steps in the development of the new simulation software discussed in Chapter 3 and machine learning models described in Chapter 4 is to validate its physics performance for the LHCb Run 3 data. There are many changes and new features in the new version of the simulation framework, GAUSS-ON-GAUSSINO, and the list includes a new detector, updated detector description (DD4HEP), more recent GEANT4 version, as well as all the fast simulation components described in Chapter 4. Physics validation with selected decay channels is crucial to ensure that the new simulation framework with all the new additions is able to produce results suitable for physics analyses. Validation with simulated samples of the ML-based fast simulations is described in this Chapter.

5.1 Preliminary results

5.1.1 Preparing the simulation samples

The simulation samples were prepared using the new simulation GAUSS-ON-GAUSSINO framework, described in Chapter 3. At the time of preparing the physics validation studies, the first official minimum bias samples were available for the 2022 data-taking conditions, and therefore this setup was used to prepare the samples for the physics validation studies. In 2022, the UT detector was not yet installed and so it was not included in the simulation, and therefore the validation studies are performed without the UT detector. As of 2023, the new simulation framework, GAUSS-ON-GAUSSINO, was still missing the implementation of the RICH detectors physics processes, i.e. the simulation of the optical photons in the detector. Therefore, the identification system is incomplete and might lead to misidentified hadrons, such as pions and kaons.

The simulation software was configured to produce samples in two scenarios:

- *ML-based fast simulation*, in which the CALOML+VAE fast simulation described in Section 4.2 was turned on in the electromagnetic calorimeter, and detailed simulation with GEANT4 10.7.3 was used in the rest of the detector,
- *Detailed simulation*, in which the detailed simulation with GEANT4 10.7.3 was used in the whole detector.

Each scenario was repeated for each of the decay channels resulting in a total of 8 large samples, 1 million events each.

The digitization of the samples was performed on both simulated samples with BOOLE. The HLT trigger and reconstruction software (MOORE) was then run on the digitized samples. Following the sequence described in Figure 3.2, n-tuples were created with the DAVINCI framework. All simulated samples were produced and processed up to the trigger and reconstruction making use of the LHC computing grid.

Most of the developments described in Chapter 4 are focused on the electromagnetic calorimeter, and therefore the validation described in this chapter focuses on decays with photons and electrons in the final state. Therefore, there should be at least one decay channel with a photon in the final state and one decay channel with an electron in the final state, as well as a decay channel with both photon and electron in the final state.

Radiative B decays play an important role in search of new physics [188, 189]. In LHCb, the direct CP asymmetry can be searched with $B_{(s)} \to K^{(*,**)}\gamma$ decay channels. In particular, the ratio of branching fractions of $B^0 \to K^{*0}\gamma$ and $B^0 \to \phi\gamma$, or the direct CP asymmetry in $B^0 \to K^{*0}\gamma$ decays are observables that can be measured with good precision at LHCb [190]. $B^0 \to K^{*0}\gamma$ is also one of the channels that were used to validate the photon reconstruction performance of the electromagnetic calorimeter in the past [191]. In this channel, K^{*0} decays to $K^+\pi^-$, and using it in the validation studies may be problematic due to the missing RICH physics in GAUSS-ON-GAUSSINO, nevertheless, the decision was made to include this channel in the validation studies due to its importance. In order to avoid bias in the results from the mis-identification of hadrons, another radiative channel was included in the validation studies to provide a complementary check.

$B_{(s)} \to J/\psi\gamma$ radiative decays are very rare, and no significant signal has been detected using 3 fb^{-1} [192] of data. Nevertheless, the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay is a very useful channel to test the fast simulation of photons, and at the same time, the muon reconstruction performance with the new simulation framework. The $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay is also included in the validation samples to test simultaneously the electron reconstruction performance and the photon reconstruction performance based on the fast simulation samples. The choice of these two channels is also motivated by the fact that decays involving $b \to s\ell\ell$ transitions are used in tests of lepton universality [193, 194]. In particular, the ratio of branching fractions $\mathcal{B}(J/\psi \to e^+e^-)/\mathcal{B}(J/\psi \to \mu^+\mu^-)$ is an important observable, as it is constrained to unity by theory assuming the lepton flavour universality. In addition, decays including $J/\psi \to e^+e^-$ are also frequently used to measure the electron reconstruction efficiency. In LHCb [195], the $B^+ \to J/\psi(\to e^+e^-)K^+$ decay was previously used to determine the electron reconstruction efficiency using a tag-and-probe method and kinematics constraints. Therefore, the same decay channel is included in the validation studies in this work as a source of electrons.

5.1.2 Selected decay channels

All eight samples (four decay channels, and two simulation scenarios) were produced according to the setup described in the previous section. The results consist of the plots of invariant mass distributions of the B mesons, as well as the transverse momentum distributions of selected final state particles in the decay channels.

$B^+ \to J/\psi\,(\to e^+e^-)K^+$

The B^+ meson invariant mass distribution from the $B^+ \to J/\psi\,(\to e^+e^-)K^+$ decay is shown in Figure 5.1. Good agreement between the two simulation scenarios is observed, although the mass peak in the fast simulation scenario is slightly shifted towards lower mass values by around 10 MeV/c^2. The transverse momentum distribution of the e^+ of the same decay channel is shown in Figure 5.2. The sample with fast simulated showers shows a slightly lower number of events around the peak. Good agreement is observed in higher transverse momentum values.

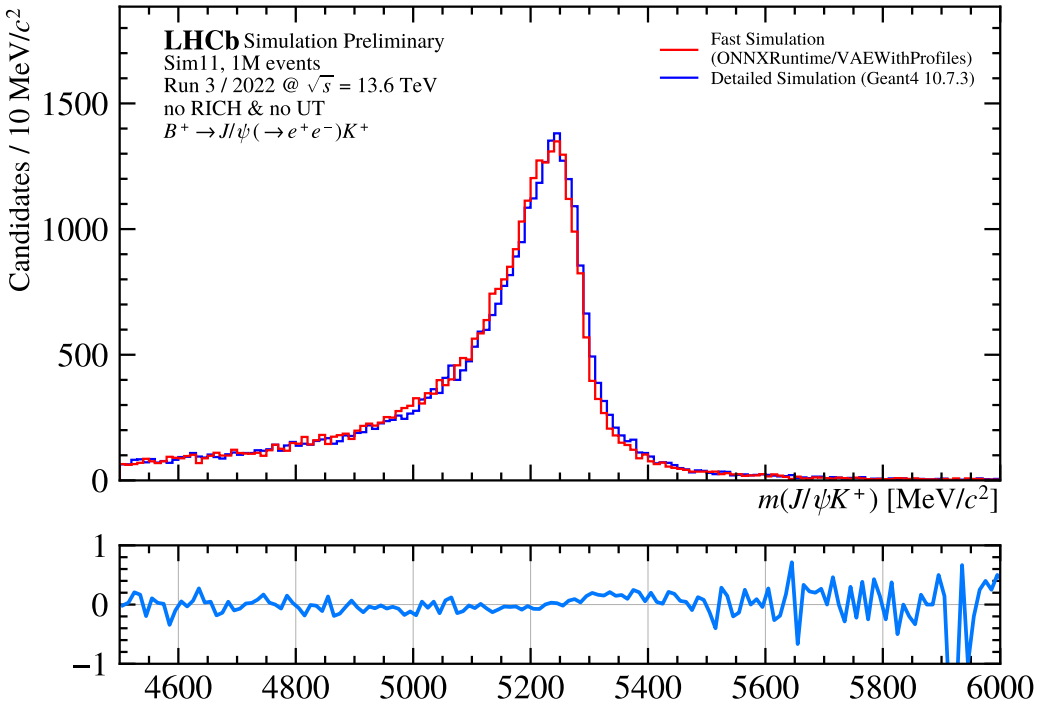


FIGURE 5.1: B^+ invariant mass distribution from the uncalibrated simulation sample of the $B^+ \to J/\psi\,(\to e^+e^-)K^+$ decay.

$B_s^0 \to J/\psi(\to e^+e^-)\gamma$

The B_s^0 meson invariant mass distribution from the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay is shown in Figure 5.3. In this case, the mass peak in the fast simulation scenario is much more shifted towards lower mass values by around 40 MeV/c^2. The transverse momentum distribution of the e^+ of the same decay channel is shown in Figure 5.4. Similarly to the previous decay channel, the fast simulation sample is underestimating the number of events around the peak. The transverse momentum distribution of the γ of the channel with B_s^0 meson is shown in Figure 5.5. The momentum is slightly underestimated in the fast simulation scenario across the whole range of transverse momenta.

$B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$

In the B_s^0 mass distribution in Figure 5.6, with J/ψ decaying to muons, an even larger shift of around 80 MeV/c^2 is observed. In addition, the mass peak is much higher in the fast simulation scenario. The transverse momentum distribution of the μ^+ of the

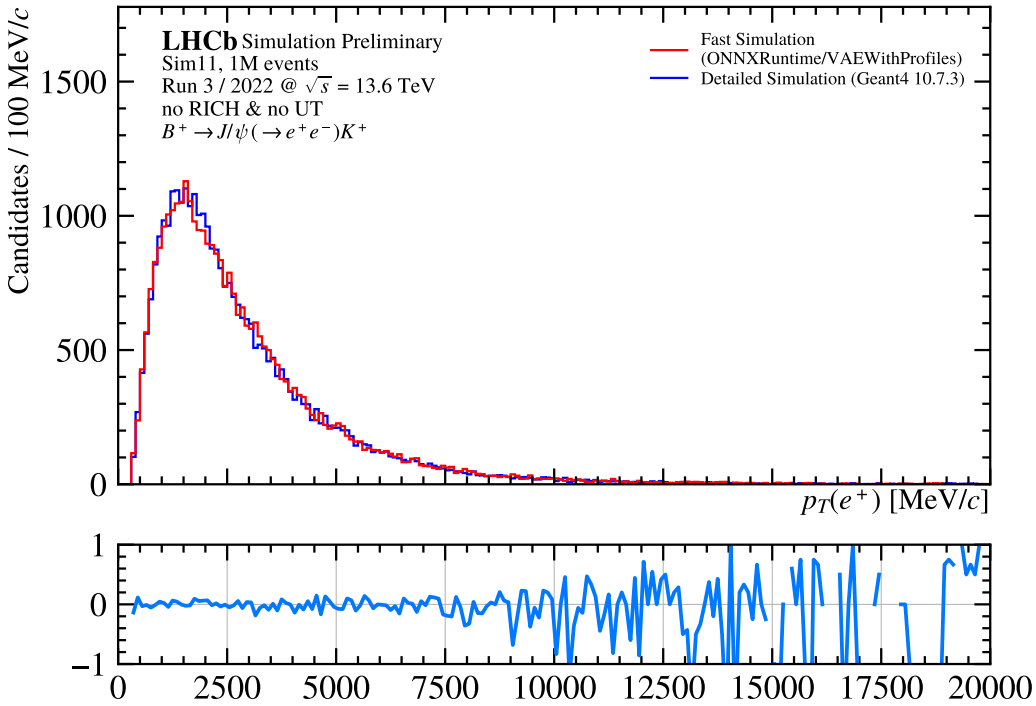


FIGURE 5.2: e^+ transverse momentum distribution from the uncalibrated simulation sample of the $B^+ \to J/\psi\ (\to e^+e^-)K^+$ decay.

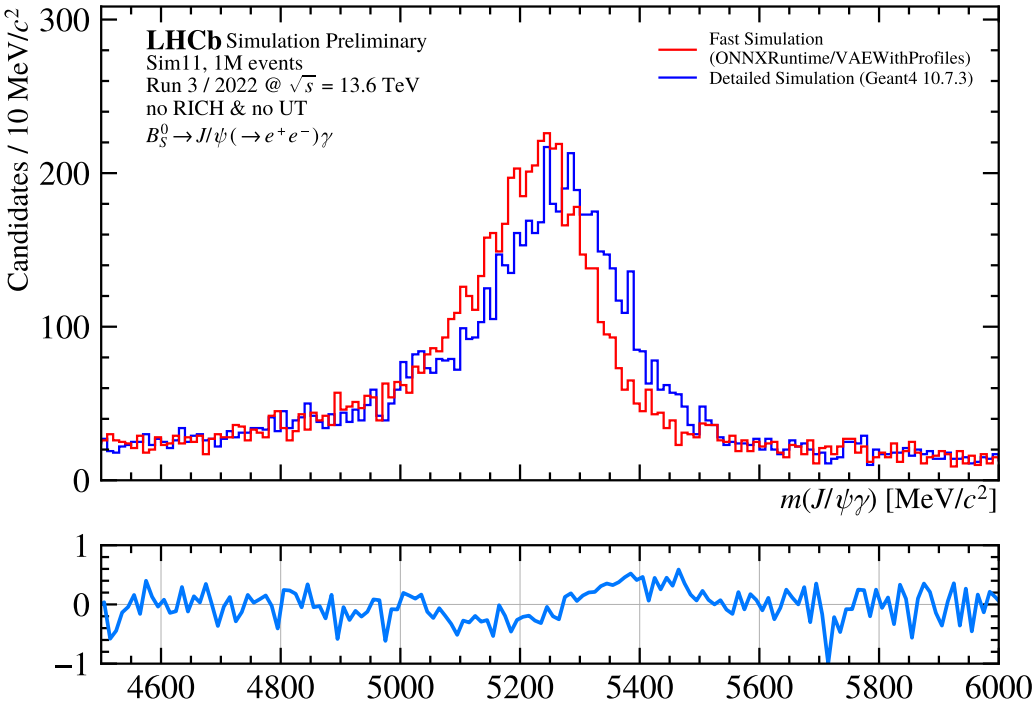


FIGURE 5.3: B_s^0 invariant mass distribution from the uncalibrated simulation sample of the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay.

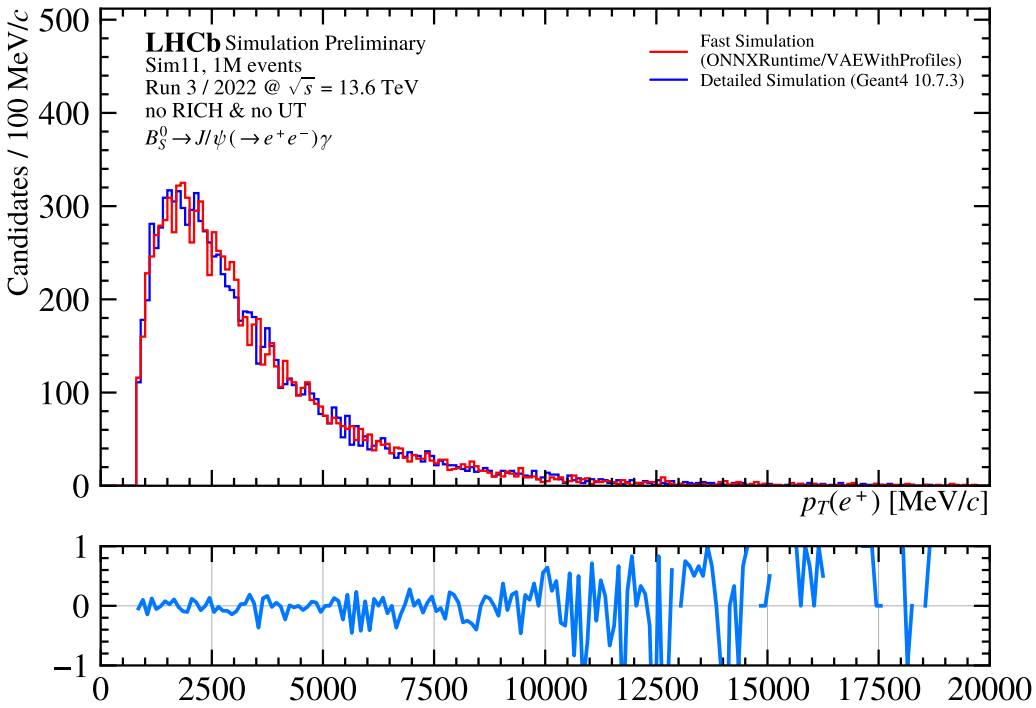


FIGURE 5.4: e^+ transverse momentum distribution from the uncalibrated simulation sample of the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay.

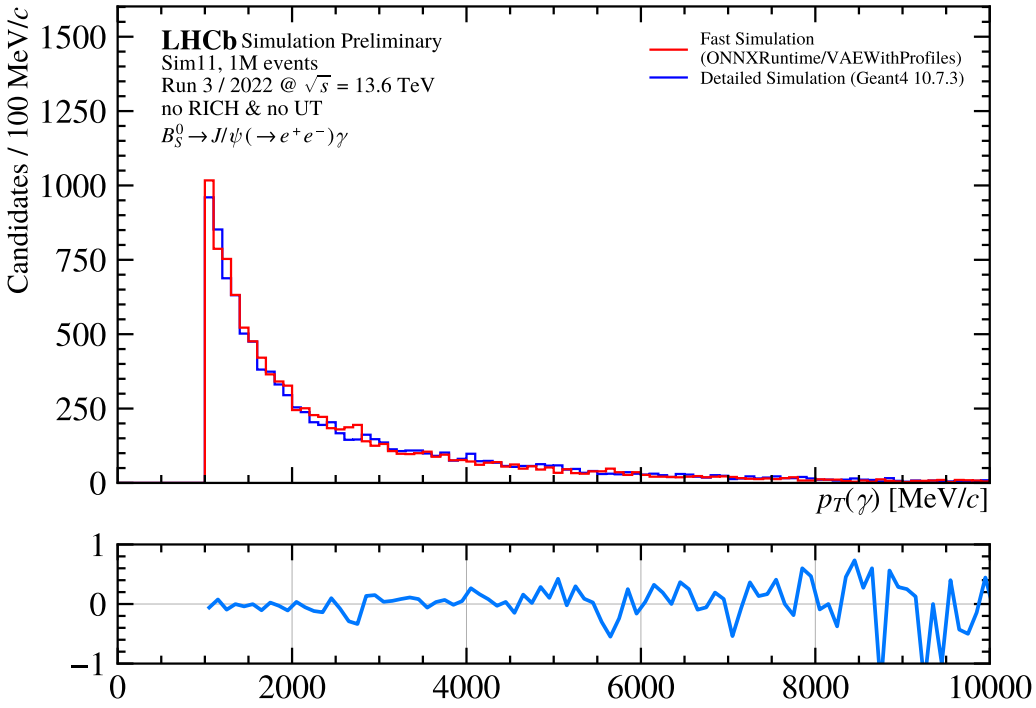


FIGURE 5.5: γ transverse momentum distribution from the uncalibrated simulation sample of the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay.

same decay channel is shown in Figure 5.7. Relatively good agreement is observed across the whole range of transverse momenta, which is expected as the muon simulation is not affected by the fast simulation happening in the electromagnetic calorimeter area. The transverse momentum distribution of the γ of the same decay channel is shown in Figure 5.8 and shows a similar trend as the B_s^0 meson invariant mass distribution.

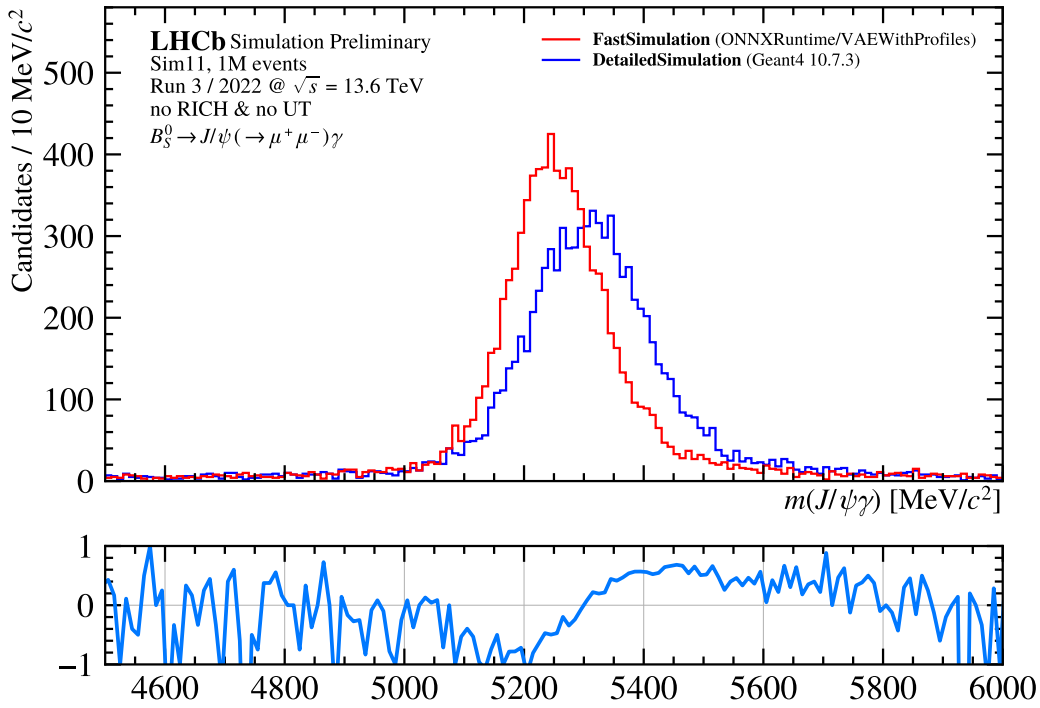


FIGURE 5.6: B_s^0 invariant mass distribution from the uncalibrated simulation sample of the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay.

$B^0 \to K^{*0}(\to K^+\pi^-)\gamma$

Finally, the B^0 meson invariant mass distribution from the $B^0 \to K^{*0}\gamma$ decay is shown in Figure 5.9. Similarly to the B_s^0 meson with J/ψ decaying to muons, the mass peak is shifted towards lower mass values by around 80 MeV/c^2 and the mass peak is larger in the fast simulation scenario. The transverse momentum distribution of the γ of the same decay channel is shown in Figure 5.10 and represents a relatively good agreement across the whole range of transverse momenta.

5.2 Calibrated results

Relatively large mass peak shifts, as well as the underestimated momenta in the fast simulation scenario presented in the previous section, were not expected, as the systematic error of the trained model was estimated to be between 1-4 %, depending on the particle momentum. Careful analysis of the preliminary n-tuples of reconstructed quantities revealed that it was necessary to reduce the systematic error of the model down to 0.01 % in order to achieve better agreement between the two simulation scenarios.

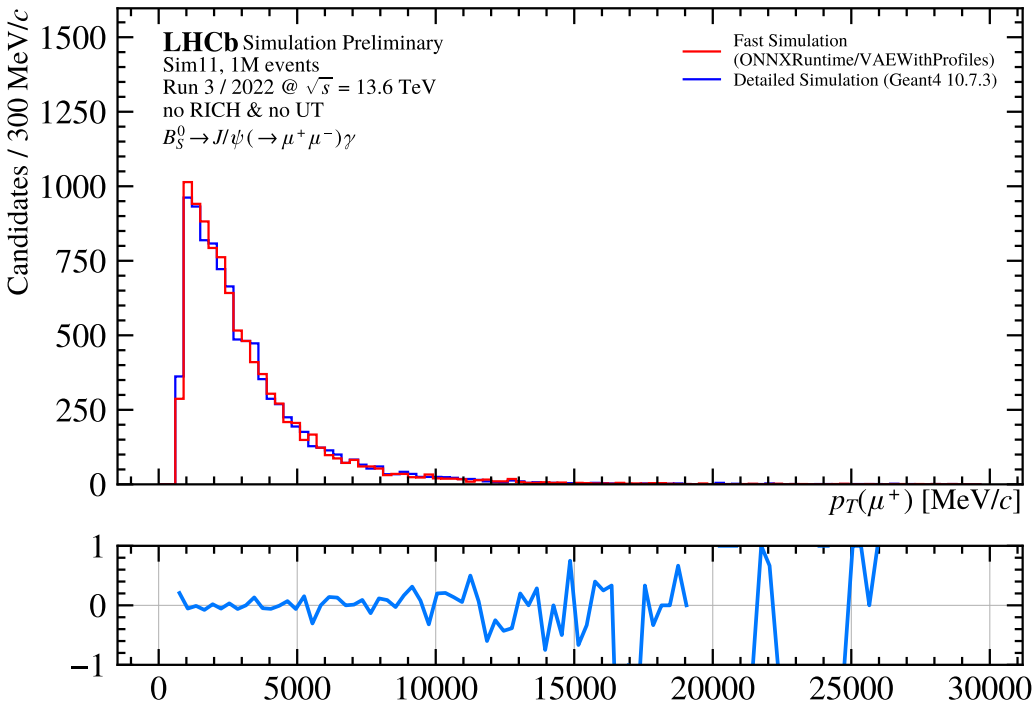


FIGURE 5.7: μ^+ transverse momentum distribution from the uncalibrated simulation sample of the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay.

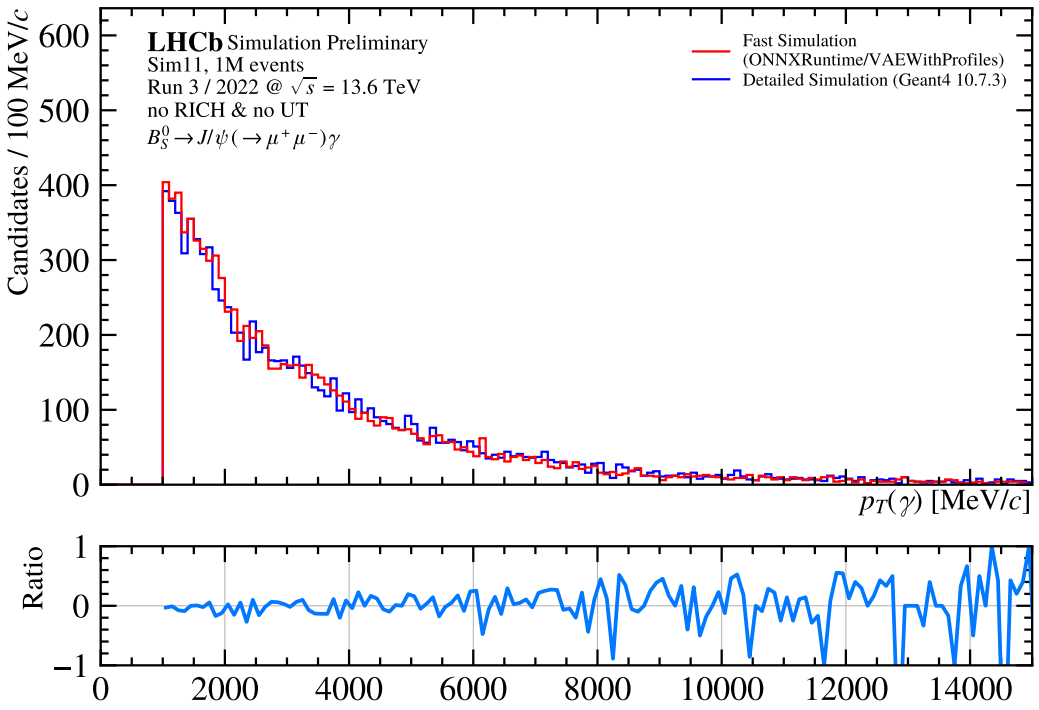


FIGURE 5.8: γ transverse momentum distribution from the uncalibrated simulation sample of the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay.

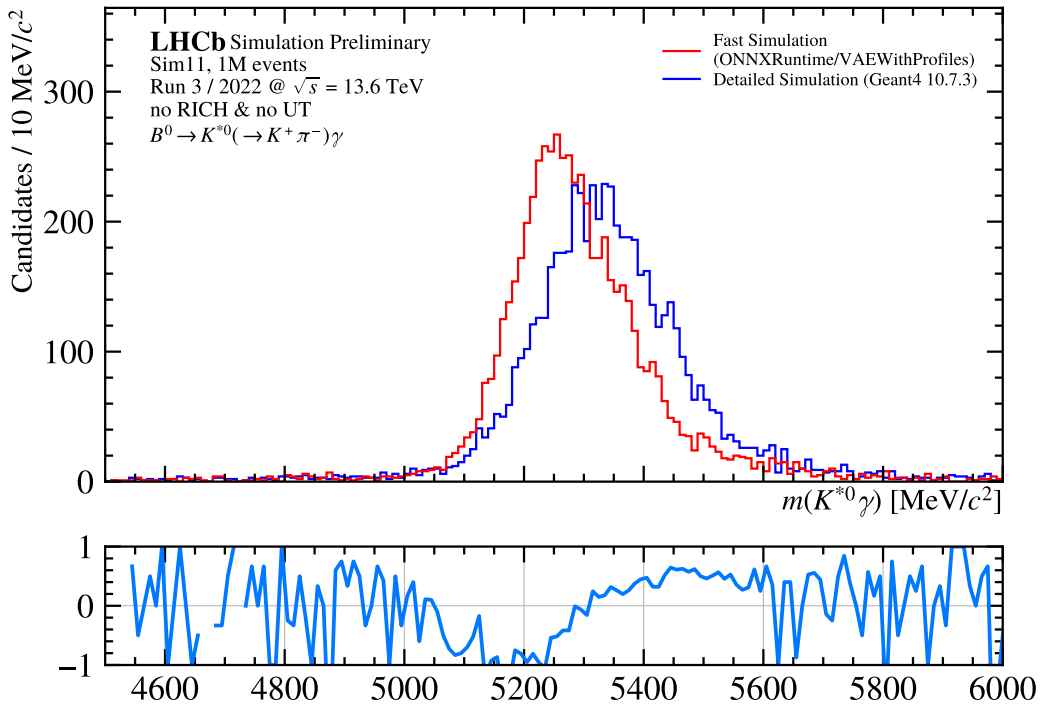


FIGURE 5.9: B^0 invariant mass distribution from the uncalibrated simulation sample of the $B^0 \to K^{*0}(\to K^+\pi^-)\gamma$ decay.

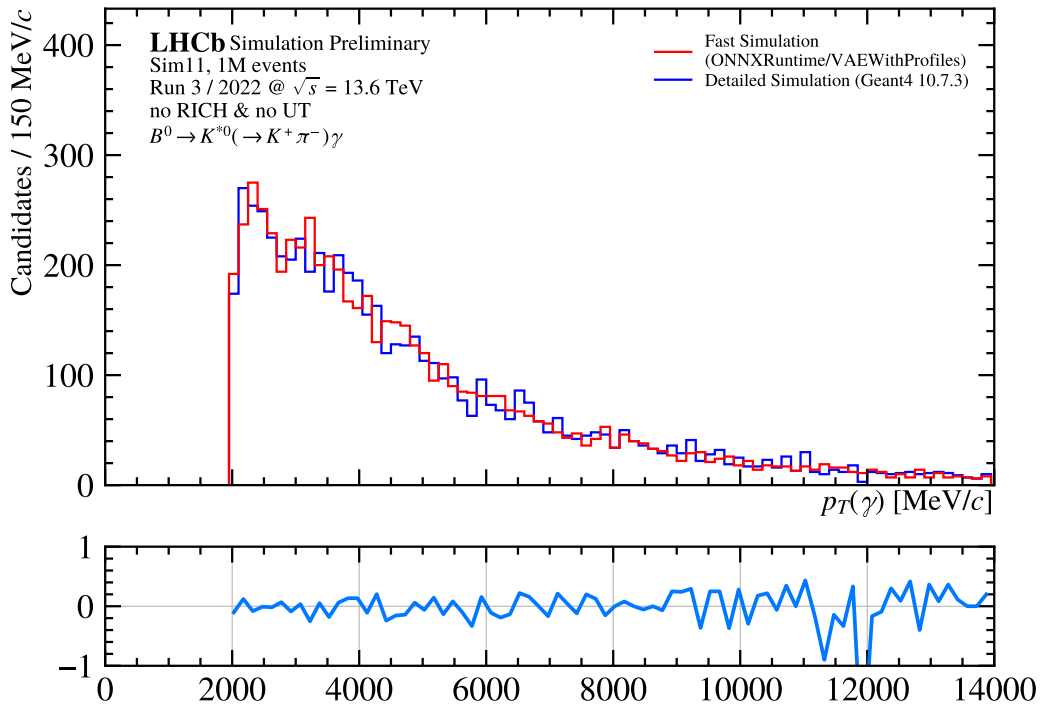


FIGURE 5.10: γ transverse momentum distribution from the uncalibrated simulation sample of the $B^0 \to K^{*0}(\to K^+\pi^-)\gamma$ decay.

5.2.1 Calibration of the simulation samples

Tuning of the CALOML+VAE model can be performed in two ways: either by tuning the model hyperparameters, or by tuning the total energy deposit e_{max} and total number of hits n_{max} of the postprocessing step described in Section 4.2.1. The second approach was chosen, as the procedure did not require retraining of the model, and therefore was faster to perform. Two parameters were introduced: e_{overflow} and n_{overflow}, which act as additional coefficients for e_{max} and n_{max}, respectively.

In total, 360 calibration points were selected to cover the parameter space of e_{overflow} and n_{overflow}. The n_{overflow} parameter was varied from 1.0 to 1.5 with a step of 0.05, and the e_{overflow} parameter was varied from 1.0 to 1.035 with a step of 0.001. For each combination of the two parameters, 4 samples with 10000 (MomentumRange particle gun) events were generated, one per magnitude of the momentum of the particle gun: $[0.1, 1.0)$, $[1.0, 10.0)$, $[10.0, 100.0)$, $[100.0, 1000.0)$ GeV/c. Each of the momentum ranges was split into 10 logarithmic bins, and systematic and random errors were calculated for each bin and plotted as a function of the momentum in Figure 5.11 for electrons and Figure 5.12 for photons, for the default values of e_{overflow} and n_{overflow} parameters. The same plots for the tuned values of e_{overflow} and n_{overflow} parameters are shown in Figure 5.13 for electrons and Figure 5.14 for photons.

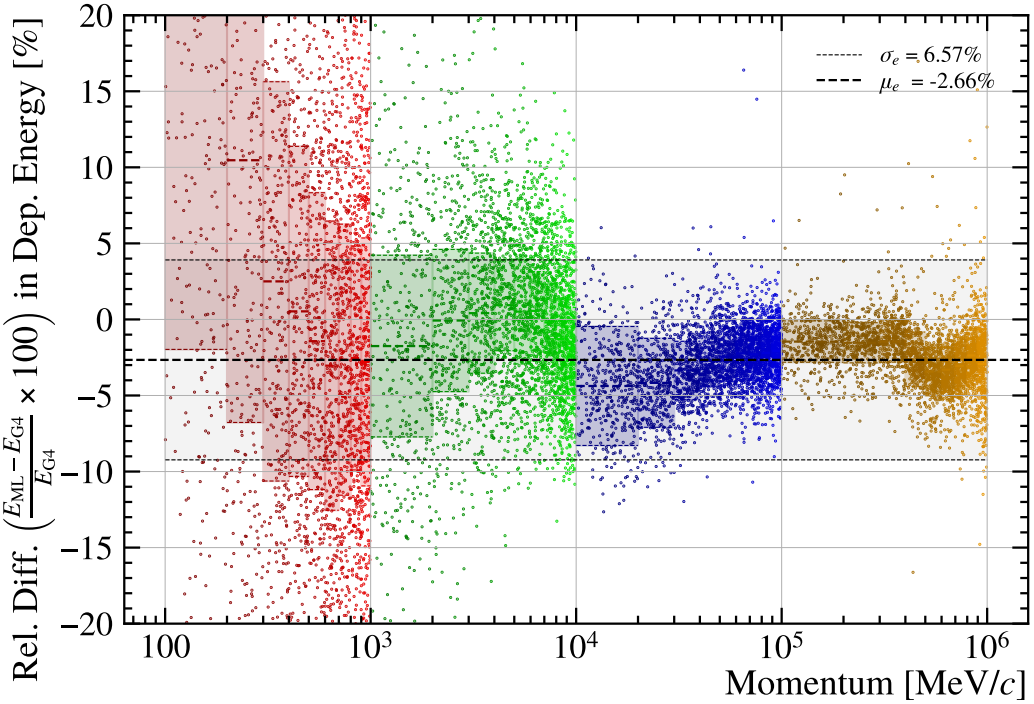


FIGURE 5.11: Systematic and random error of the CALOML+VAE model with the default values of e_{overflow} and n_{overflow} parameters for electrons.

5.2.2 Selected decay channels

The results of the calibration of the CALOML+VAE model described in the previous sections are presented in this section. Calibrated samples were prepared using the same setup as the uncalibrated samples, and the same plots were produced.

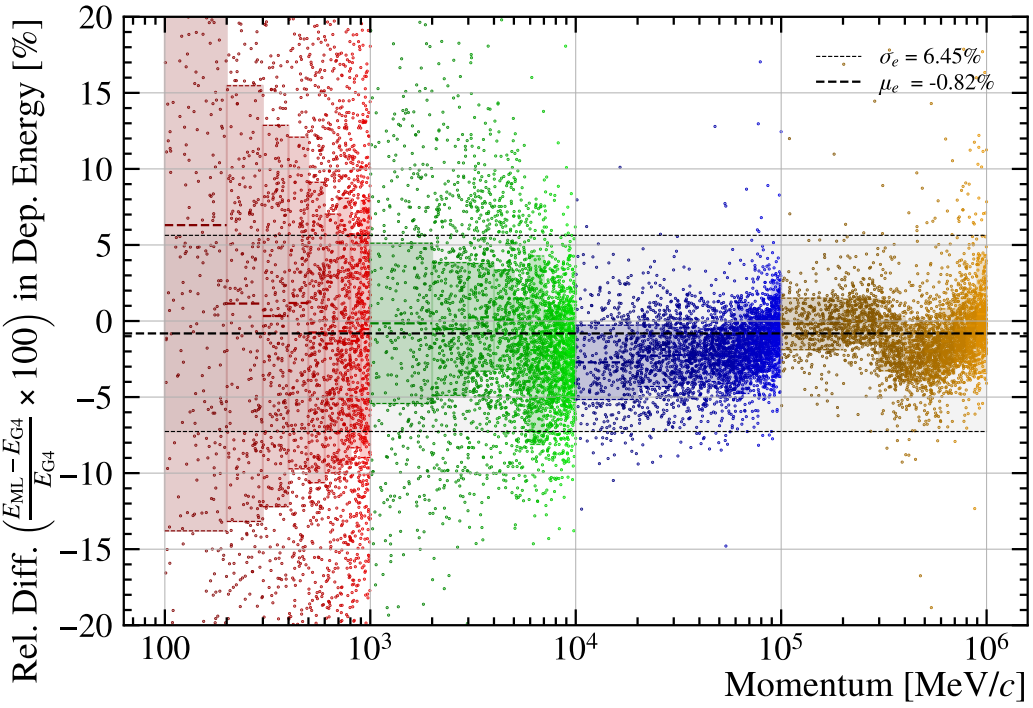


FIGURE 5.12: Systematic and random error of the CALOML+VAE model with the default values of e_{overflow} and n_{overflow} parameters for photons.

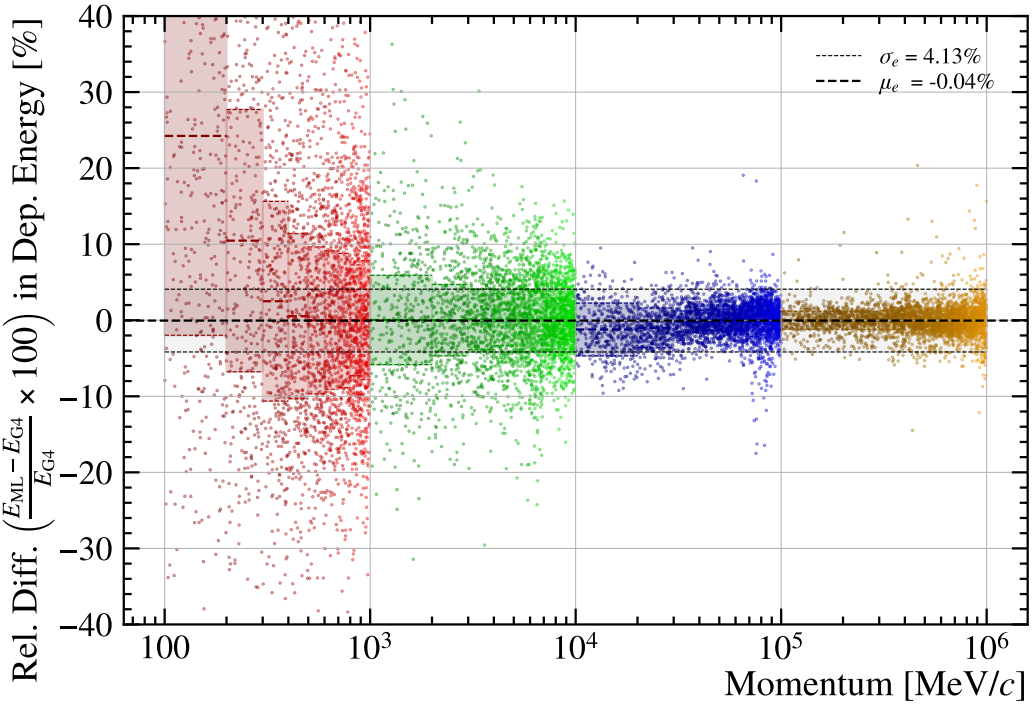


FIGURE 5.13: Systematic and random error of the CALOML+VAE model with the tuned values of e_{overflow} and n_{overflow} parameters for electrons.

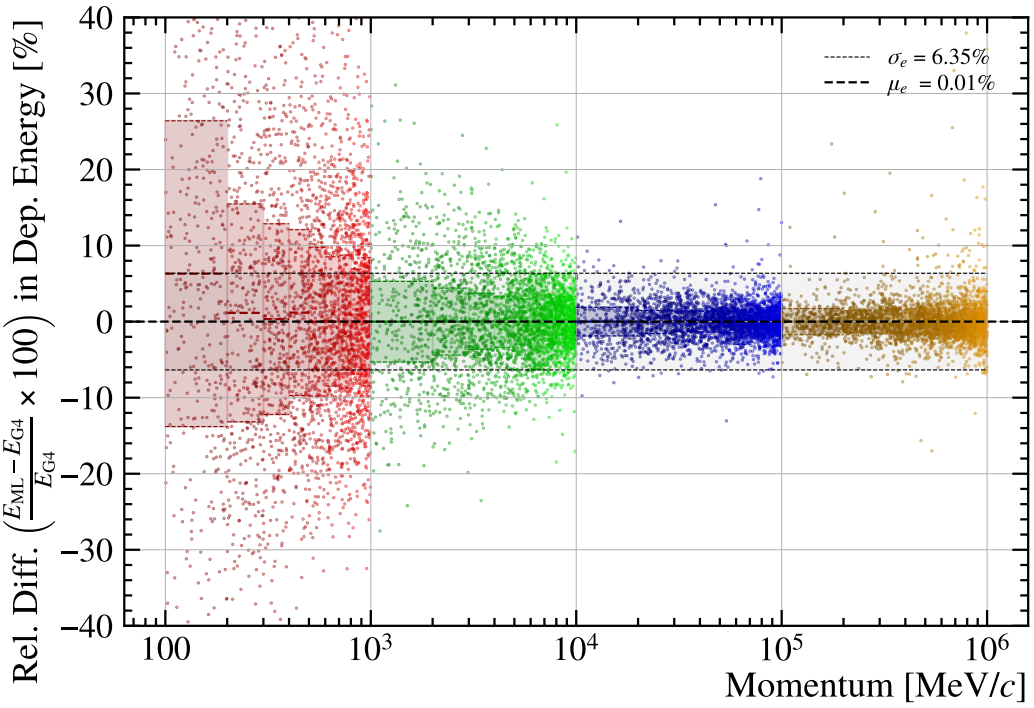


FIGURE 5.14: Systematic and random error of the CALOML+VAE model with the tuned values of e_{overflow} and n_{overflow} parameters for photons.

$B^+ \to J/\psi\ (\to e^+e^-)K^+$

Calibrated sample with $B^+ \to J/\psi\ (\to e^+e^-)K^+$ as the decay channel is represented by the B^+ invariant mass distribution in Figure 5.15 and e^+ transverse momentum distribution in Figure 5.16. Very good agreement between the two simulation scenarios is observed, and the mass peak in the fast simulation scenario is no longer shifted towards lower values. The transverse momentum of the e^+ coming from the fast simulation sample is almost exactly the same as the one coming from the detailed simulation sample.

$B_s^0 \to J/\psi(\to e^+e^-)\gamma$

B_s^0 invariant mass distribution from the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay is shown in Figure 5.17. The mass peak is no longer shifted towards lower values, and the statistical analysis shows that both distributions are almost indistinguishable, although a slight overshoot in the fast simulation scenario is observed around the peak. e^+ transverse momentum of the same decay channel is presented in Figure 5.18. Similarly to the $B^+ \to J/\psi\ (\to e^+e^-)K^+$ decay channel, the transverse momentum distribution of the e^+ is almost indistinguishable between the two simulation scenarios.

$B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$

In the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay channel, the mass peak of B_s^0 (Figure 5.20) is also no longer shifted towards lower values, however, the whole mass peak is slightly narrower and higher in the fast simulation scenario, despite the fact that

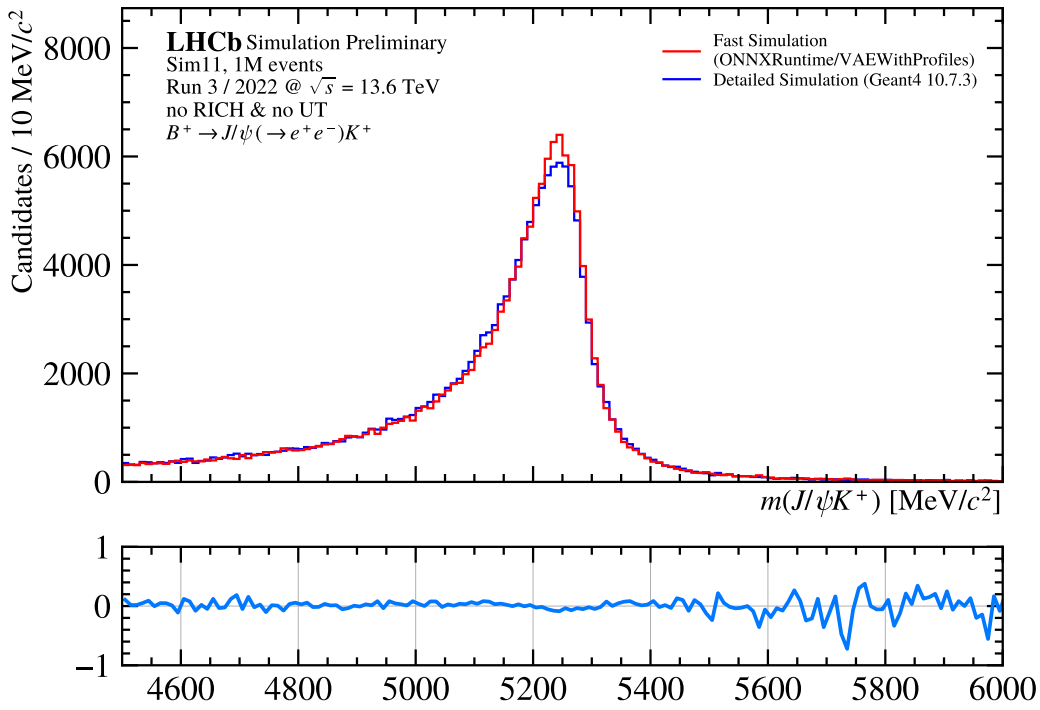


FIGURE 5.15: B^+ invariant mass distribution from the calibrated simulation sample of the $B^+ \to J/\psi\ (\to e^+e^-)K^+$ decay.

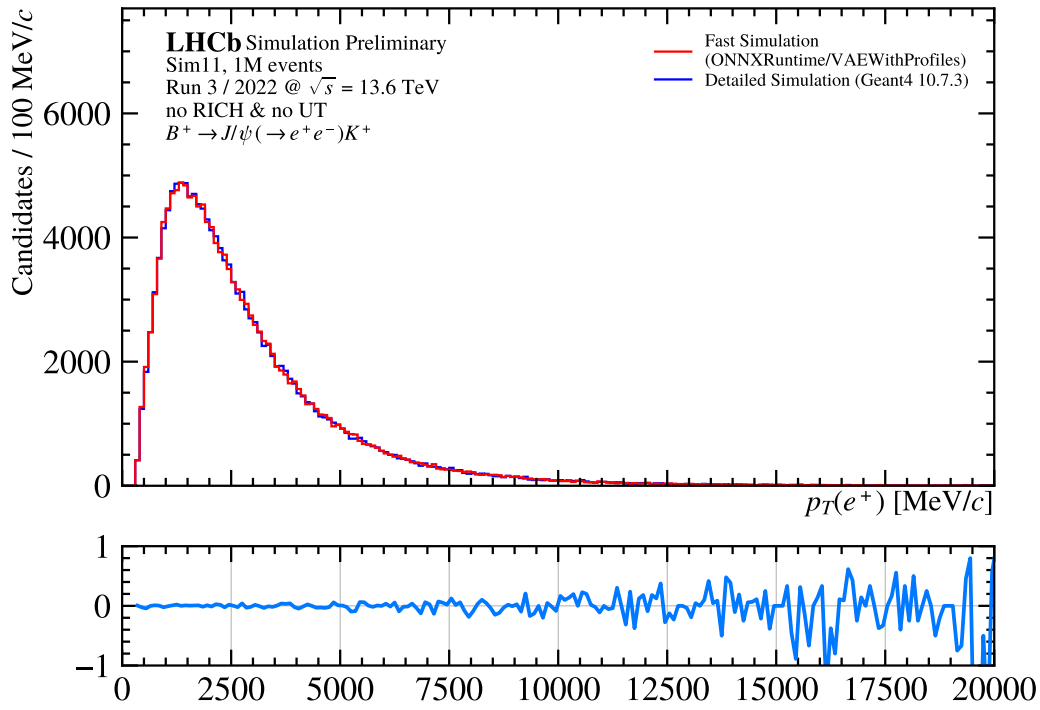


FIGURE 5.16: e^+ transverse momentum distribution from the calibrated simulation sample of the $B^+ \to J/\psi\ (\to e^+e^-)K^+$ decay.

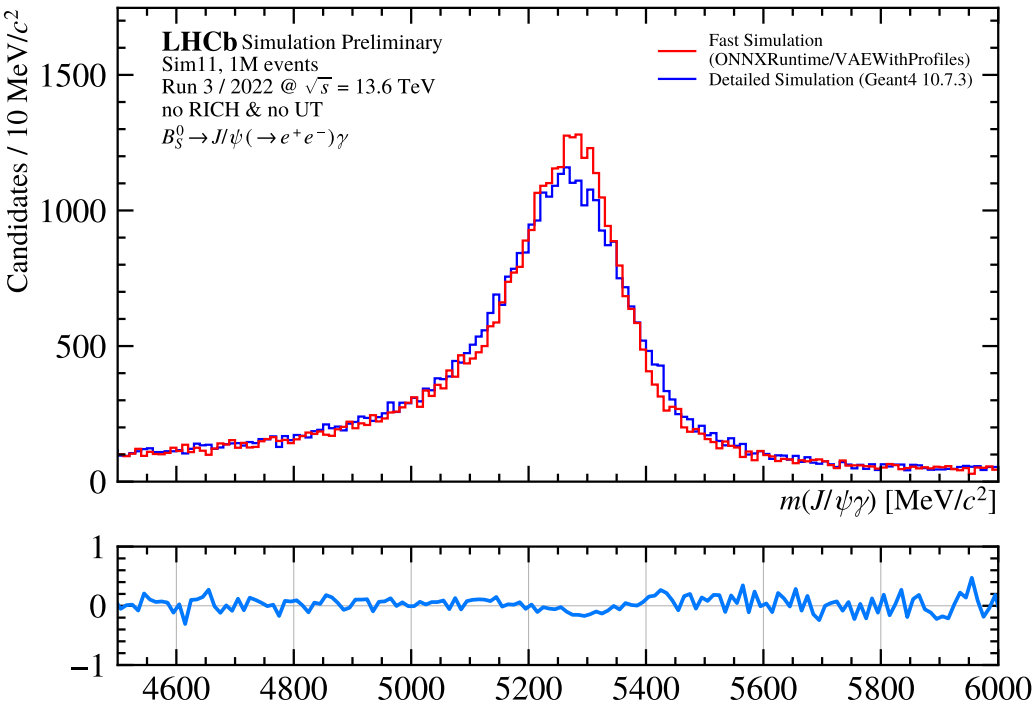


FIGURE 5.17: B_s^0 invariant mass distribution from the calibrated simulation sample of the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay.

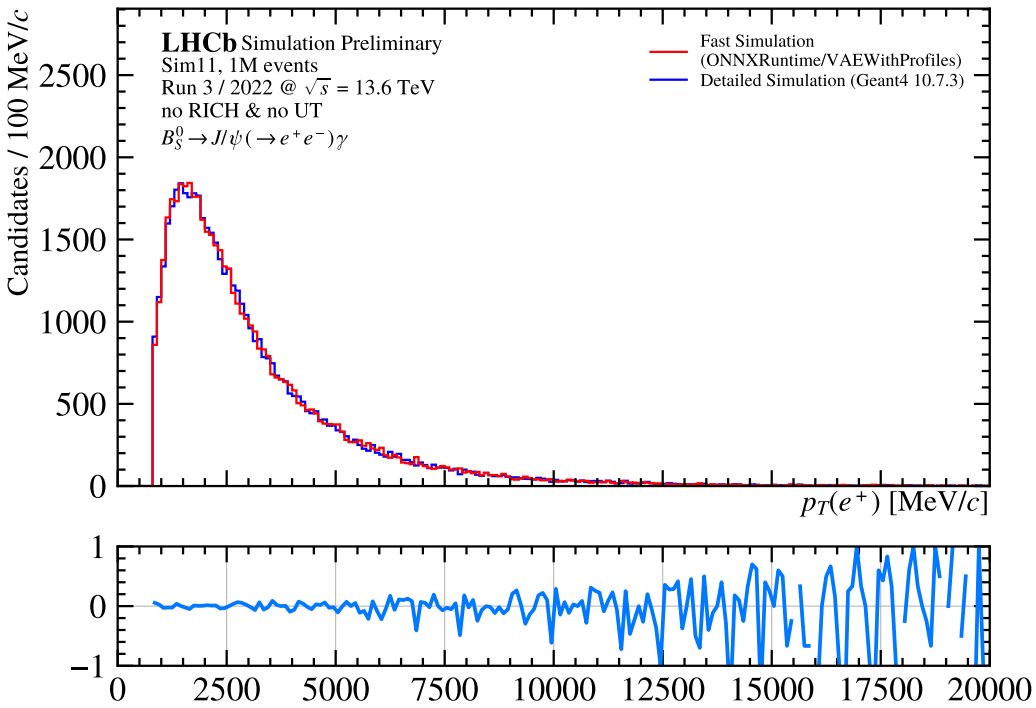


FIGURE 5.18: e^+ transverse momentum distribution from the calibrated simulation sample of the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay.

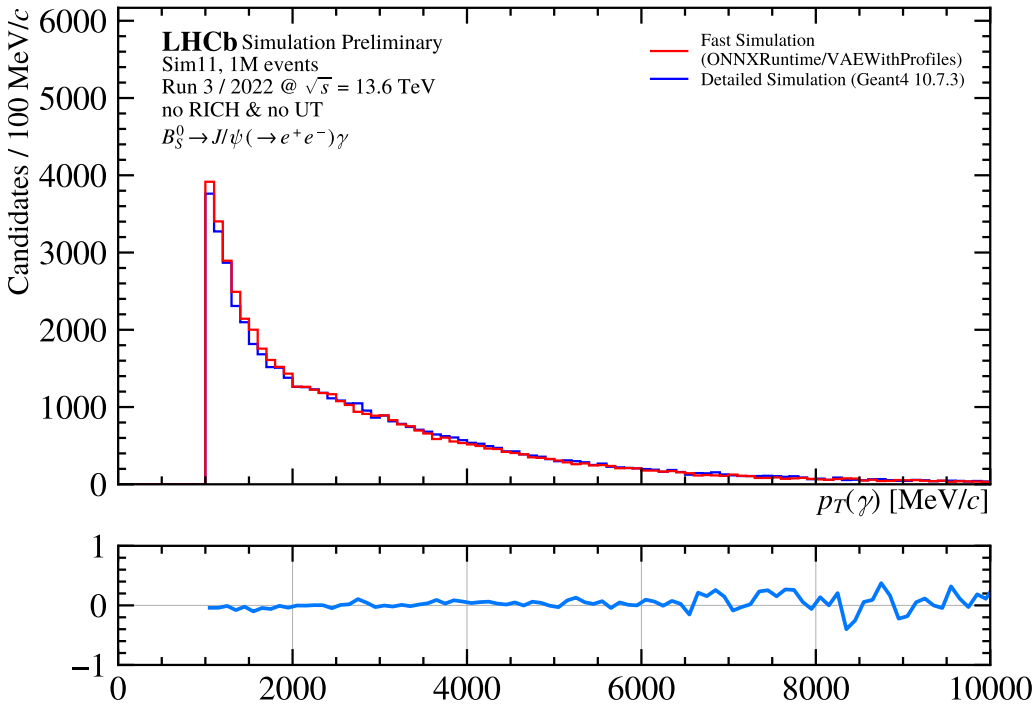


FIGURE 5.19: γ transverse momentum distribution from the calibrated simulation sample of the $B_s^0 \to J/\psi(\to e^+e^-)\gamma$ decay.

the statistical properties of the two distributions are very similar. p_T distribution of the μ^+ (Figure 5.21) represents a very good agreement between the two simulation scenarios, similarly to the uncalibrated sample. p_T distribution of the γ (Figure 5.22) is in a very good agreement across the whole range of transverse momenta.

$B^0 \to K^{*0}(\to K^+\pi^-)\gamma$

Moving to the last decay channel considered, the $B^0 \to K^{*0}(\to K^+\pi^-)\gamma$ decay, it is observed that the mass is also no longer shifted towards lower values, as shown in Figure 5.23 Despite having a slightly narrower mass peak, the statistical properties of the two distributions are very similar. p_T distribution of the γ (Figure 5.24) is indistinguishable between the two simulation scenarios.

5.2.3 Additional discussion

Results presented in this chapter, and in particular disagreements between the fast and detailed simulations show that detailed physics validation is an indispensable step in the development of the new simulation software and machine learning models. In particular, reconstruction algorithms and trigger selections are particularly sensitive to the quality of the simulation samples, and a major effort is needed to ensure that the fast simulation samples are as close as possible to those obtained with the GEANT4 modeling.

The calibration of the CALOML+VAE model presented in this chapter shows that decreasing the systematic error of the model is crucial in order to achieve good agreement between the fast and detailed simulations samples. Small differences in shapes of the invariant mass distributions are observed, and more understanding is needed in order to determine the source of these differences. One of the possible

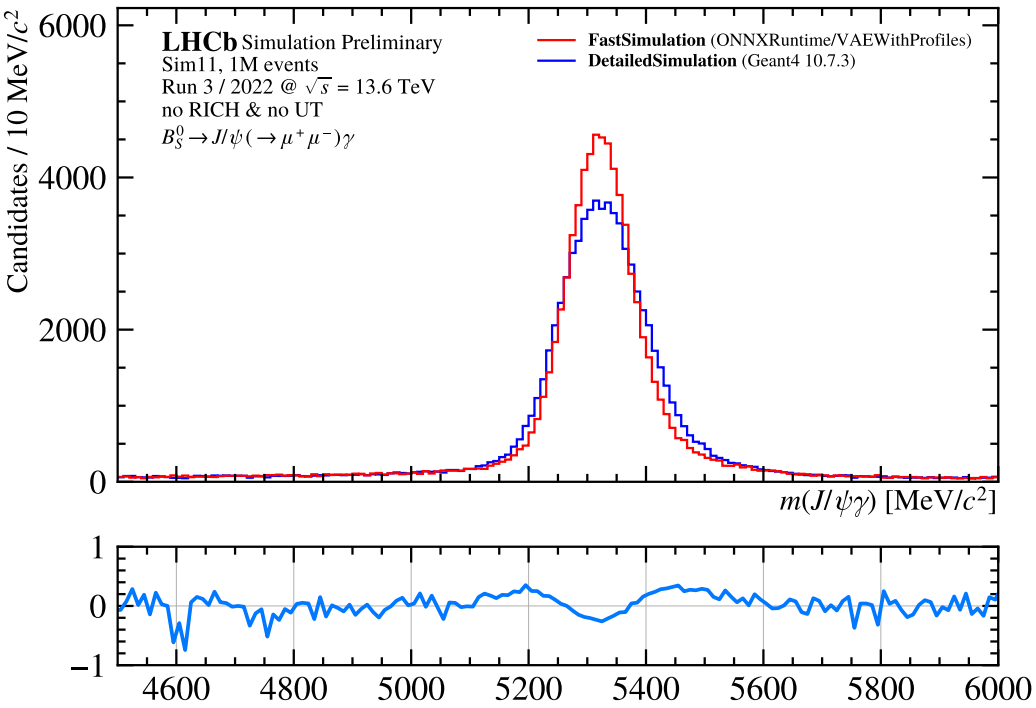


FIGURE 5.20: B_s^0 invariant mass distribution from the calibrated simulation sample of the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay.

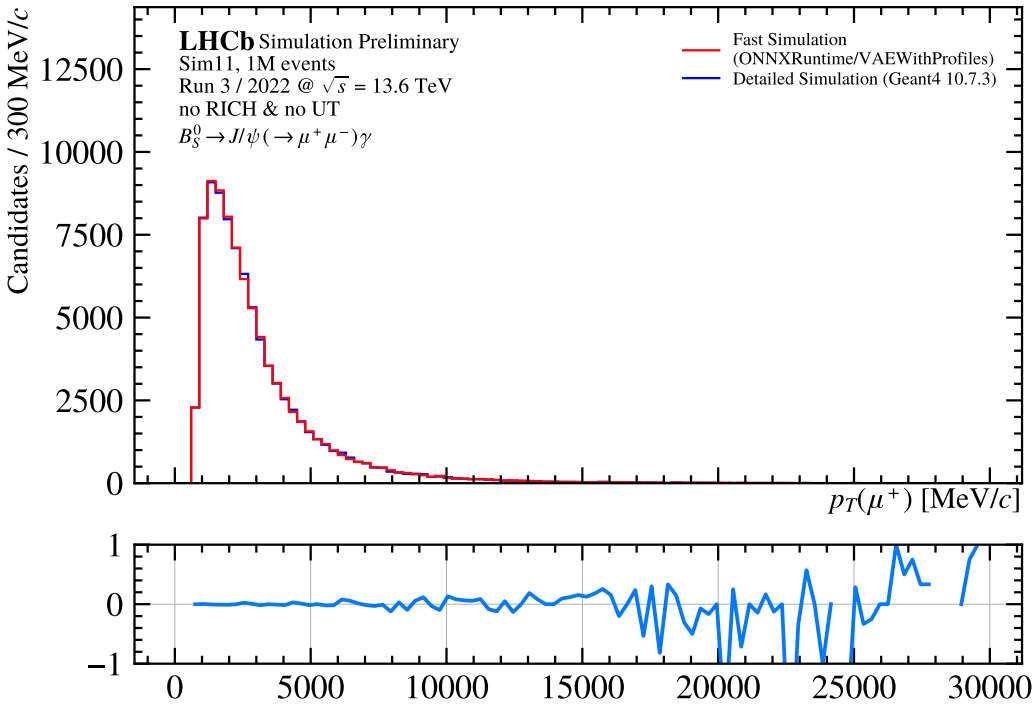


FIGURE 5.21: μ^+ transverse momentum distribution from the calibrated simulation sample of the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay.

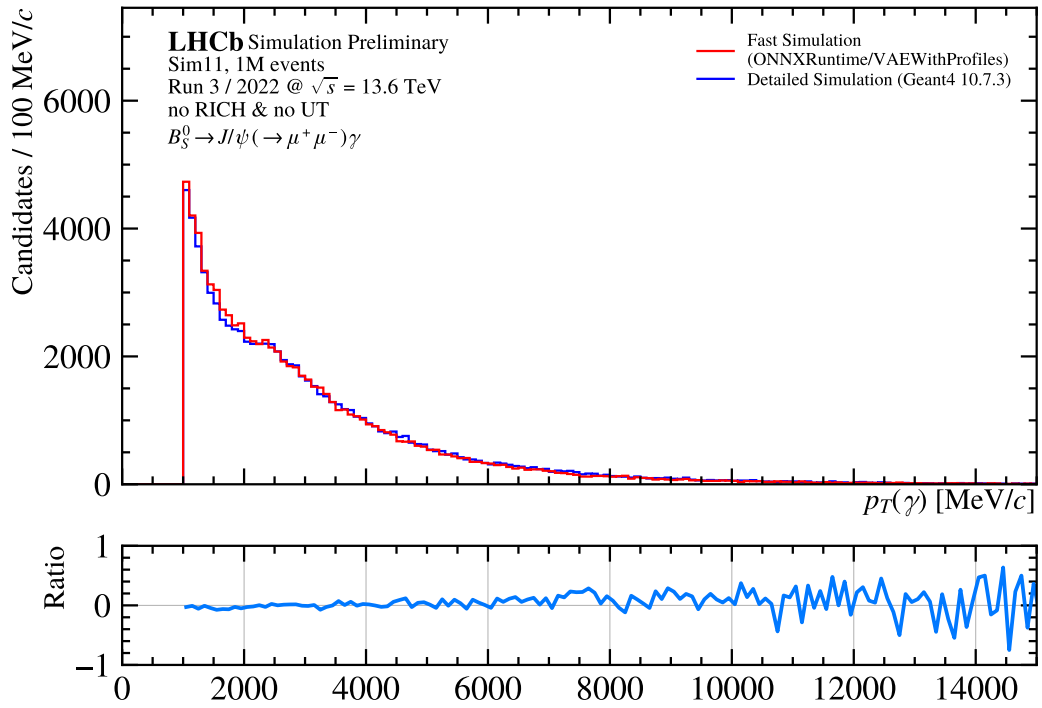


FIGURE 5.22: γ transverse momentum distribution from the calibrated simulation sample of the $B_s^0 \to J/\psi(\to \mu^+\mu^-)\gamma$ decay.

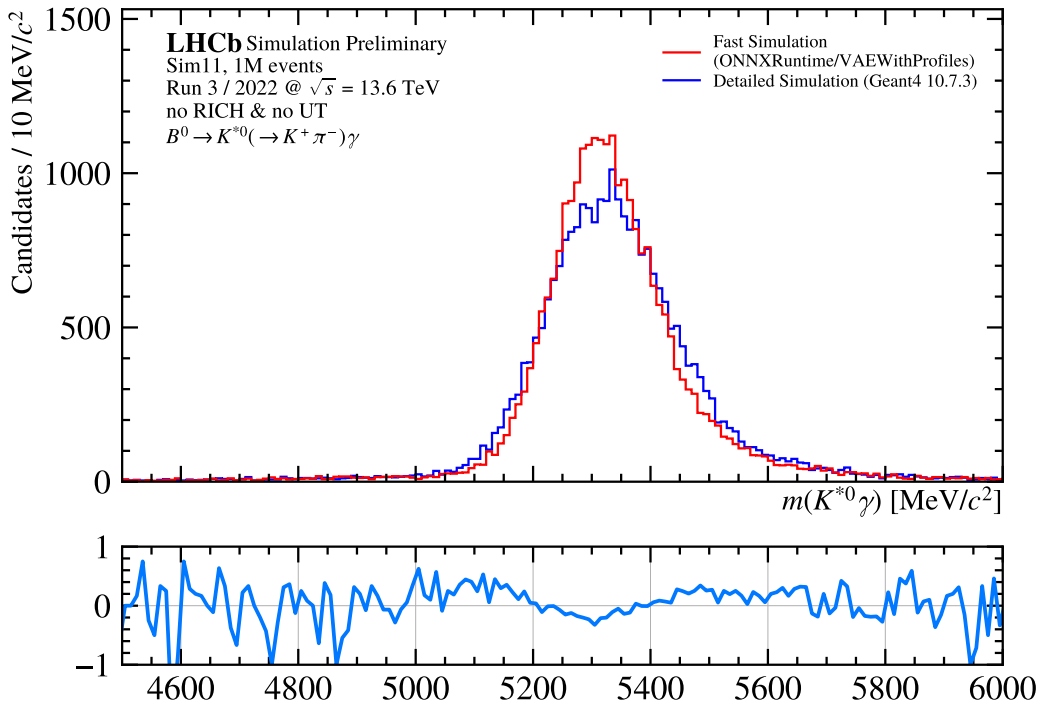


FIGURE 5.23: B^0 invariant mass distribution from the calibrated simulation sample of the $B^0 \to K^{*0}(\to K^+\pi^-)\gamma$ decay.

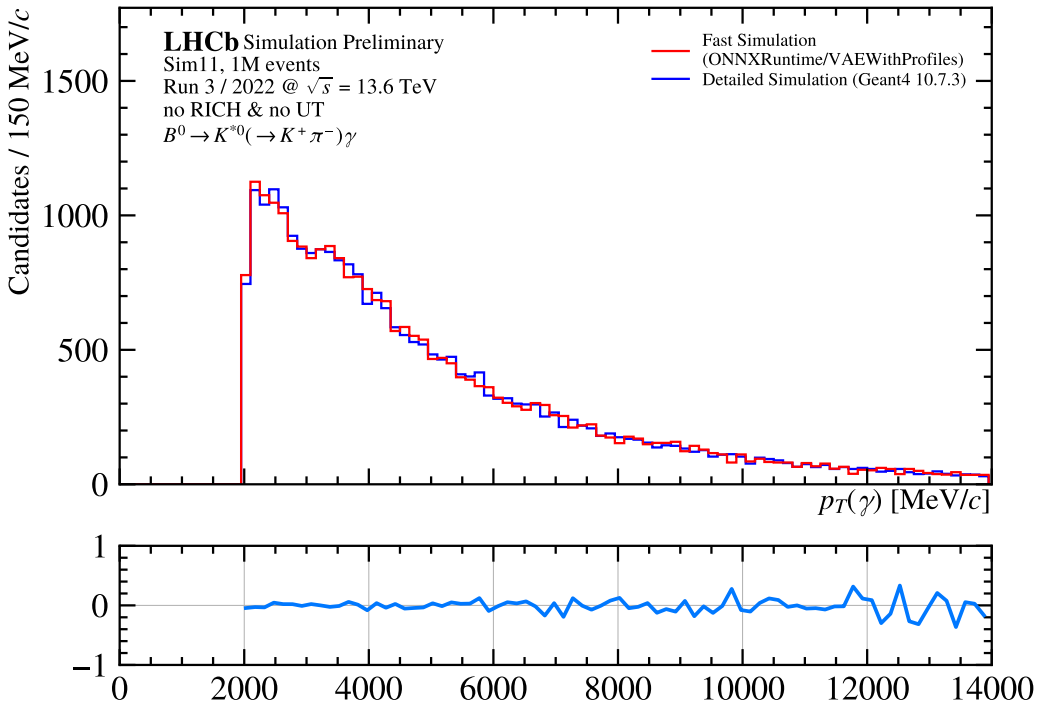


FIGURE 5.24: γ transverse momentum distribution from the calibrated simulation sample of the $B^0 \to K^{*0}(\to K^+\pi^-)\gamma$ decay.

sources of the differences is the fact that the CALOML+VAE model is relatively simple, and does not take into account the full complexity of the electromagnetic calorimeter, and as a result, the electromagnetic showers produced in the fast simulation scenario suffer from too low variability. A simple test was made to verify this hypothesis: selected points across the calorimeter phase space were chosen and each particle was shot 50 times at each point with different pseudorandom seeds. The results are represented as the ratio of the active energy to the total momentum of the particles as a function of the particle type in Figure 5.25, as a function of the ϕ angle in Figure 5.26, and as a function of the θ angle in Figure 5.27. In all the box plots, it is clearly visible that the variability of the electromagnetic showers produced by the CALOML+VAE model is much lower than the one produced by GEANT4. In particular, the variability of the showers produced with the CALOML+VAE model is almost constant across the whole phase space, and the whiskers are very short with much fewer outliers than in the case of GEANT4. The results of this test suggest that the CALOML+VAE model should be improved in order to produce more realistic electromagnetic showers, and therefore more realistic simulation samples, although the statistical analysis of the samples presented in this chapter shows that the differences between the fast simulation and detailed simulation samples are already very small. Studies conducted in this section show that the choice of the model, along with careful training and tuning, is crucial for achieving successful performance and high precision results.

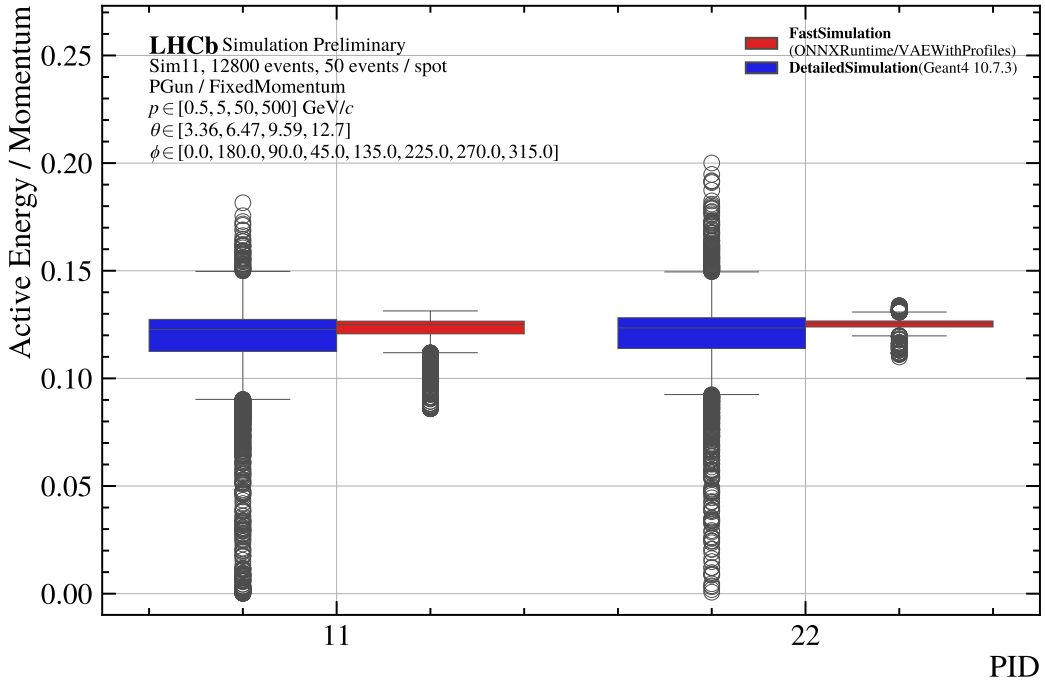


FIGURE 5.25: Variability of electromagnetic showers produced by the CALOML+VAE model and GEANT4 for different particle types.

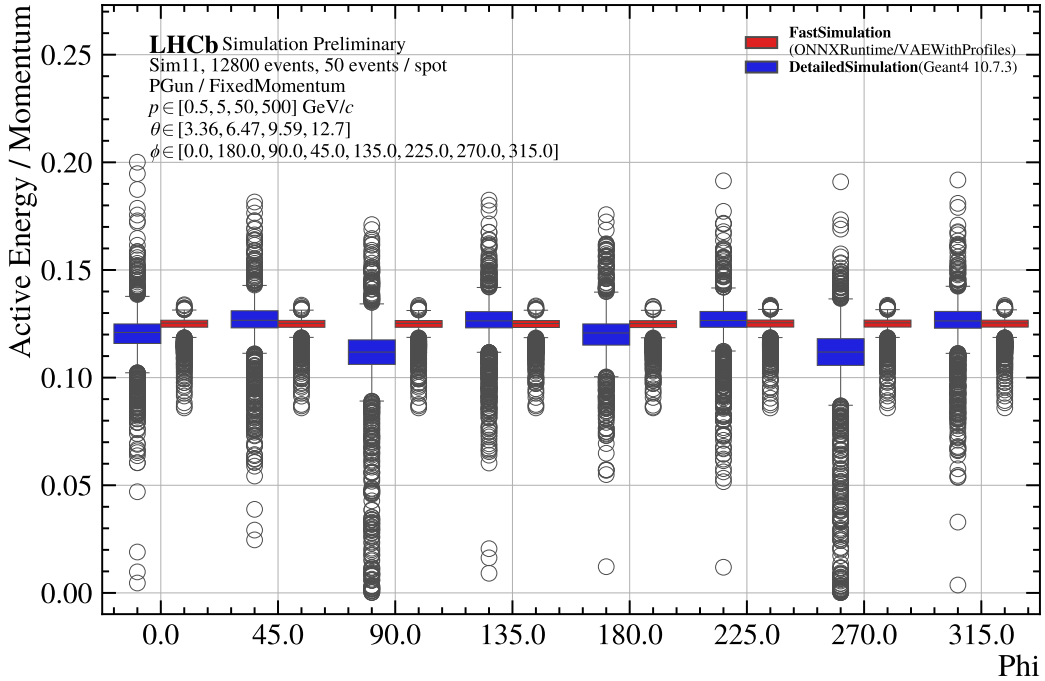


FIGURE 5.26: Variability of electromagnetic showers produced by the CALOML+VAE model and GEANT4 for different ϕ angles.

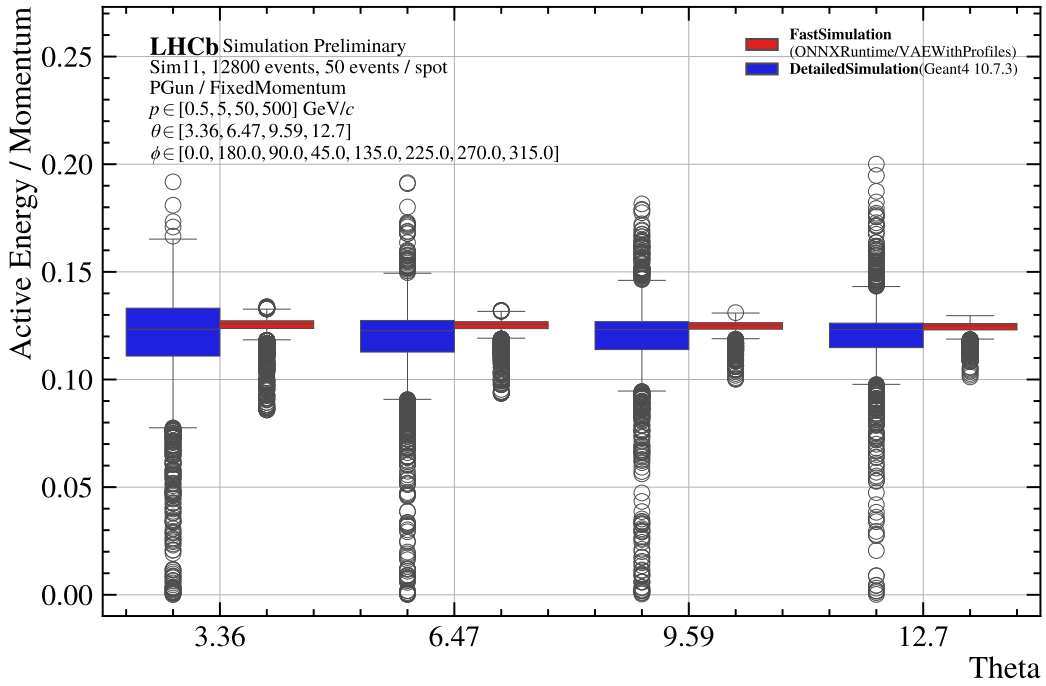


FIGURE 5.27: Variability of electromagnetic showers produced by the CALOML+VAE model and GEANT4 for different θ angles.

Bibliography

[1] David Griffiths. *Introduction to Elementary Particles*. Wiley, Dec. 1987. DOI: 10.1002/9783527618460. URL: https://doi.org/10.1002/9783527618460.

[2] Steven Weinberg. *The Quantum Theory of Fields*. Cambridge University Press, June 1995. DOI: 10.1017/cbo9781139644167. URL: https://doi.org/10.1017/cbo9781139644167.

[3] Michael E. Peskin. *An Introduction To Quantum Field Theory*. CRC Press, May 2018. DOI: 10.1201/9780429503559. URL: https://doi.org/10.1201/9780429503559.

[4] Steven Weinberg. *THE DISCOVERY OF SUBATOMIC PARTICLES*. 1984. ISBN: 9780521823517.

[5] Robert Mann. *An Introduction to Particle Physics and the Standard Model*. CRC Press, Nov. 2009. DOI: 10.1201/9781420083002. URL: https://doi.org/10.1201/9781420083002.

[6] Peter W. Higgs. "Broken Symmetries and the Masses of Gauge Bosons". In: *Phys. Rev. Lett.* 13 (1964). Ed. by J. C. Taylor, pp. 508–509. DOI: 10.1103/PhysRevLett.13.508.

[7] John Ellis. "The Higgs and the fate of the universe". In: *CERN Courier* 62.4 (2022), pp. 59–60. URL: https://cds.cern.ch/record/2815308.

[8] Gilad Perez. "The origin of particle masses". In: *CERN Courier* 62.4 (2022), pp. 53–54. URL: https://cds.cern.ch/record/2815308.

[9] CMS collaboration, Serguei Chatrchyan et al. "Observation of a New Boson at a Mass of 125 GeV with the CMS Experiment at the LHC". In: *Phys. Lett. B* 716 (2012), pp. 30–61. DOI: 10.1016/j.physletb.2012.08.021. arXiv: 1207.7235 [hep-ex].

[10] ATLAS collaboration, Georges Aad et al. "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC". In: *Phys. Lett. B* 716 (2012), pp. 1–29. DOI: 10.1016/j.physletb.2012.08.020. arXiv: 1207.7214 [hep-ex].

[11] Antonio Pich. "The Standard Model of Electroweak Interactions". In: *2010 European School of High Energy Physics*. Jan. 2012, pp. 1–50. arXiv: 1201.0537 [hep-ph].

[12] T. S. Virdee. "Beyond the standard model of particle physics". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2075 (Aug. 2016), p. 20150259. DOI: 10.1098/rsta.2015.0259. URL: https://doi.org/10.1098/rsta.2015.0259.

[13] Oliver Sim Brüning et al. *LHC Design Report*. CERN Yellow Reports: Monographs. Geneva: CERN, 2004. DOI: 10.5170/CERN-2004-003-V-1. URL: https://cds.cern.ch/record/782076.

[14] "LHC Machine". In: *JINST* 3 (2008). Ed. by Lyndon Evans and Philip Bryant, S08001. DOI: 10.1088/1748-0221/3/08/S08001.

[15] I. Zurbano Fernandez et al. "High-Luminosity Large Hadron Collider (HL-LHC): Technical design report". In: 10/2020 (Dec. 2020). Ed. by I. Béjar Alonso et al. DOI: 10.23731/CYRM-2020-0010.

[16] FCC collaboration, A. Abada et al. "FCC-ee: The Lepton Collider: Future Circular Collider Conceptual Design Report Volume 2". In: *Eur. Phys. J. ST* 228.2 (2019), pp. 261–623. DOI: 10.1140/epjst/e2019-900045-4.

[17] FCC collaboration, A. Abada et al. "FCC-hh: The Hadron Collider: Future Circular Collider Conceptual Design Report Volume 3". In: *Eur. Phys. J. ST* 228.4 (2019), pp. 755–1107. DOI: 10.1140/epjst/e2019-900087-0.

[18] "The International Linear Collider Technical Design Report - Volume 3.II: Accelerator Baseline Design". In: (June 2013). Ed. by Chris Adolphsen et al. arXiv: 1306.6328 [physics.acc-ph].

[19] CLIC collaboration, T. K. Charles et al. "The Compact Linear Collider (CLIC) - 2018 Summary Report". In: 2/2018 (Dec. 2018). Ed. by P. N. Burrows et al. DOI: 10.23731/CYRM-2018-002. arXiv: 1812.06018 [physics.acc-ph].

[20] P. Lebrun S. Gourlay L. Rossi. "Powering the field forward". In: *CERN Courier* 57.7 (2017), pp. 17–20. URL: https://cds.cern.ch/record/2815308.

[21] Andrew Sessler and Edmund Wilson. *Engines of Discovery*. WORLD SCIENTIFIC, June 2012. DOI: 10.1142/8552. URL: https://doi.org/10.1142/8552.

[22] LHCb collaboration. "LHCb trigger system: Technical Design Report". In: CERN-LHCC-2003-031 (2003).

[23] S. Perazzini D. vom Bruch L. Grillo. "LHCb looks forward to the 2030s". In: *CERN Courier* 63.2 (2023), pp. 22–25. URL: https://cds.cern.ch/record/2857133.

[24] David W. O. Rogers et al. "5 – Monte Carlo Techniques of Electron and Photon Transport for Radiation Dosimetry". In: 1990. URL: https://people.physics.carleton.ca/~drogers/pubs/papers/RB90.pdf.

[25] F James. "Monte Carlo theory and practice". In: *Reports on Progress in Physics* 43.9 (Sept. 1980), p. 1145. DOI: 10.1088/0034-4885/43/9/002. URL: https://dx.doi.org/10.1088/0034-4885/43/9/002.

[26] Andy Buckley et al. "General-purpose event generators for LHC physics". In: *Physics Reports* 504.5 (2011), pp. 145–233. ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2011.03.005. URL: https://www.sciencedirect.com/science/article/pii/S0370157311000846.

[27] Torbjörn Sjöstrand et al. "Introduction to Event Generators 1-4". In: *Lecture notes, CTEQ/MCnet School, DESY, July* (2016). URL: http://home.thep.lu.se/~torbjorn/talks/neu22a.pdf.

[28] Torbjörn Sjöstrand, Stephen Mrenna, and Peter" Skands. "A brief introduction to PYTHIA 8.1". In: *Comput. Phys. Commun.* 178 (2008), pp. 852–867. DOI: 10.1016/j.cpc.2008.01.036. arXiv: 0710.3820 [hep-ph].

[29] Torbjörn Sjöstrand, Stephen Mrenna, and Peter" Skands. "PYTHIA 6.4 physics and manual". In: *JHEP* 05 (2006), p. 026. DOI: 10.1088/1126-6708/2006/05/026. arXiv: hep-ph/0603175 [hep-ph].

[30] Johannes Bellm et al. "Herwig 7.0/Herwig++ 3.0 release note". In: *Eur. Phys. J. C* 76.4 (2016), p. 196. DOI: 10.1140/epjc/s10052-016-4018-8. arXiv: 1512.01178 [hep-ph].

[31] Enrico Bothmann et al. "Event generation with Sherpa 2.2". In: *SciPost Physics* 7.3 (Sept. 2019). ISSN: 2542-4653. DOI: 10.21468/scipostphys.7.3.034. URL: http://dx.doi.org/10.21468/SciPostPhys.7.3.034.

[32] Christian Bierlich et al. "Robust Independent Validation of Experiment and Theory: Rivet version 3". In: *SciPost Phys.* 8 (2020), p. 026. DOI: 10.21468/SciPostPhys.8.2.026. arXiv: 1912.05451 [hep-ph].

[33] Andy Buckley et al. "Systematic event generator tuning for the LHC". In: *Eur. Phys. J. C* 65 (2010), pp. 331–357. DOI: 10.1140/epjc/s10052-009-1196-7. arXiv: 0907.2973 [hep-ph].

[34] Matt Dobbs and Jørgen Beck Hansen. "The HepMC C++ Monte Carlo event record for High Energy Physics". In: *Computer Physics Communications* 134.1 (2001), pp. 41–46. ISSN: 0010-4655. DOI: https://doi.org/10.1016/S0010-4655(00)00189-2.

[35] Andy Buckley et al. "The HepMC3 event record library for Monte Carlo event generators". In: *Computer Physics Communications* 260 (2021), p. 107310. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2020.107310.

[36] J. Apostolakis. "Detector Simulation". In: *Particle Physics Reference Library: Volume 2: Detectors for Particles and Radiation*. Ed. by Christian Wolfgang Fabjan and Herwig Schopper. Cham: Springer, 2020, pp. 485–531. DOI: 10.1007/978-3-030-35318-6_11.

[37] Geant4 collaboration, John Allison et al. "Geant4 developments and applications". In: *IEEE Trans.Nucl.Sci.* 53 (2006), p. 270. DOI: 10.1109/TNS.2006.869826.

[38] Geant4 collaboration, S. Agostinelli et al. "Geant4: A simulation toolkit". In: *Nucl. Instrum. Meth.* A506 (2003), p. 250. DOI: 10.1016/S0168-9002(03)01368-8.

[39] Giuseppe Battistoni et al. "Overview of the FLUKA code". In: *Annals Nucl. Energy* 82 (2015), pp. 10–18. DOI: 10.1016/j.anucene.2014.11.007.

[40] Nikolai V. Mokhov. "The MARS code system user's guide version 13(95)". In: (Apr. 1995).

[41] Mihaly Novak. "Special EM processes and their role in Geant4-stepping". In: (2019). Lecture given at the Geant4 Workshop. URL: https://indico.cern.ch/event/776050/contributions/3240654/attachments/1789266/2914260/Geant4Stepping.pdf.

[42] Geant4, M. Asai et al. "Design of tracking and generic processes in Geant4". In: *International Conference on Advanced Monte Carlo for Radiation Physics, Particle Transport Simulation and Applications (MC 2000)*. Oct. 2000, pp. 1117–1122. ISBN: 978-3-642-18211-2.

[43] Martin J. Berger. "Monte Carlo calculation of the penetration and diffusion of fast charged particles". In: (1963). URL: https://public.websites.umich.edu/~bielajew/NewStuff/NERS544/Berger-1963.pdf.

[44] Geant4 Collaboration. *Geant4 Physics Reference Manual, Release 11.2*. 2024. URL: https://geant4-userdoc.web.cern.ch/UsersGuides/PhysicsReferenceManual/fo/PhysicsReferenceManual.pdf.

[45] Vladimir Ivanchenko. *Electromagnetic physics, Geant4 Advanced Course*. 2023. URL: https://indico.cern.ch/event/1304769/contributions/5487609/attachments/2679536/4754412/EMPhysics2023.pdf.

[46] Alberto Ribon. *Hadronic physics, Geant4 Advanced Course*. 2023. URL: https://indico.cern.ch/event/1304769/contributions/5487619/attachments/2679535/4754179/HadronicPhysics1.pdf.

[47] CERN. *Facts and figures about the LHC*. 2023. URL: https://home.cern/resources/faqs/facts-and-figures-about-lhc.

[48] ATLAS collaboration, G. Aad et al. "The ATLAS Experiment at the CERN Large Hadron Collider". In: *JINST* 3 (2008), S08003. DOI: 10.1088/1748-0221/3/08/S08003.

[49] CMS collaboration, S. Chatrchyan et al. "The CMS Experiment at the CERN LHC". In: *JINST* 3 (2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004.

[50] LHCb collaboration, A. A. Alves Jr. et al. "The LHCb detector at the LHC". In: *JINST* 3.LHCb-DP-2008-001 (2008), S08005. DOI: 10.1088/1748-0221/3/08/S08005.

[51] ALICE collaboration, K. Aamodt et al. "The ALICE experiment at the CERN LHC". In: *JINST* 3 (2008), S08002. DOI: 10.1088/1748-0221/3/08/S08002.

[52] Ewa Lopienska. "The CERN accelerator complex, layout in 2022. Complexe des accélérateurs du CERN en janvier 2022". In: (2022). CERN-GRAPHICS-2022-001-1. URL: https://cds.cern.ch/record/2800984.

[53] LHCb collaboration, Roel Aaij et al. "The LHCb Upgrade I". In: (2023). to appear in JINST. arXiv: 2305.10515 [hep-ex].

[54] LHCb collaboration. "Framework TDR for the LHCb Upgrade: Technical Design Report". In: CERN-LHCC-2012-007 (2012).

[55] Monica Pepe Altarelli and Frederic Teubert. "B Physics at LHCb". In: *Int. J. Mod. Phys. A* 23 (2008). Contribution to 'Perspectives on LHC Physics,' edited by G. Kane and A. Pierce, pp. 5117–5136. DOI: 10.1142/S0217751X08042791. arXiv: 0802.1901. URL: http://cds.cern.ch/record/1088975.

[56] LHCb collaboration, Christian Elsasser. *$\bar{b}b$ production angle plots*. URL: https://lhcb.web.cern.ch/lhcb/speakersbureau/html/bb_ProductionAngles.html.

[57] LHCb collaboration, R. Aaij et al. "LHCb detector performance". In: *Int. J. Mod. Phys.* A30 (2015), p. 1530022. DOI: 10.1142/S0217751X15300227. arXiv: 1412.6352 [hep-ex].

[58] *Letter of Intent for the LHCb Upgrade*. Tech. rep. Geneva: CERN, 2011. URL: https://cds.cern.ch/record/1333091.

[59] LHCb collaboration. "LHCb Framework TDR for the LHCb Upgrade II Opportunities in flavour physics, and beyond, in the HL-LHC era". In: CERN-LHCC-2021-012 (2022).

[60] LHCb collaboration. "LHCb magnet: Technical Design Report". In: CERN-LHCC-2000-007 (2000).

[61] L Leduc, G Corti, and R Veness. "Design of a Highly Optimised Vacuum Chamber Support for the LHCb Experiment". In: (2011). Linked to Poster-2011-196. URL: https://cds.cern.ch/record/1383810.

[62] LHCb collaboration. "LHCb VELO (VErtex LOcator): Technical Design Report". In: CERN-LHCC-2001-011 (2001).

[63] LHCb collaboration. "LHCb VELO Upgrade Technical Design Report". In: CERN-LHCC-2013-021 (2013).

[64] LHCb collaboration. "LHCb Tracker Upgrade Technical Design Report". In: CERN-LHCC-2014-001 (2014).

[65] LHCb, Matthew Scott Rudolph. "The LHCb Upstream Tracker Upgrade". In: *PoS* Vertex2019 (2020), p. 013. DOI: 10.22323/1.373.0013.

[66] LHCb collaboration. "LHCb RICH: Technical Design Report". In: CERN-LHCC-2000-037 (2000).
[67] M. Adinolfi et al. "Performance of the LHCb RICH detector at the LHC". In: *Eur. Phys. J.* C73 (2013), p. 2431. DOI: 10.1140/epjc/s10052-013-2431-9. arXiv: 1211.6759 [physics.ins-det].
[68] LHCb collaboration. "LHCb PID Upgrade Technical Design Report". In: CERN-LHCC-2013-022 (2013).
[69] LHCb collaboration. "LHCb calorimeters: Technical Design Report". In: CERN-LHCC-2000-036 (2000).
[70] C. Abellan Beteta et al. "Calibration and performance of the LHCb calorimeters in Run 1 and 2 at the LHC". In: (2020). submitted to JINST. arXiv: 2008.11556 [hep-ex].
[71] LHCb collaboration. "LHCb muon system: Technical Design Report". In: CERN-LHCC-2001-010 (2001).
[72] A A Alves Jr. et al. "Performance of the LHCb muon system". In: *JINST* 8 (2013), P02022. DOI: 10.1088/1748-0221/8/02/P02022. arXiv: 1211.1346 [physics.ins-det].
[73] F. Archilli et al. "Performance of the muon identification at LHCb". In: *JINST* 8 (2013), P10020. DOI: 10.1088/1748-0221/8/10/P10020. arXiv: 1306.0249 [physics.ins-det].
[74] LHCb collaboration. *LHCb: Technical Proposal*. CERN-LHCC-98-004. Geneva: CERN, 1998. URL: https://cds.cern.ch/record/622031.
[75] LHCb collaboration. "Future physics potential of LHCb". In: (2022).
[76] LHCb collaboration. "Physics case for an LHCb Upgrade II — Opportunities in flavour physics, and beyond, in the HL-LHC era". In: CERN-LHCC-2018-027 LHCb-PUB-2018-009 (2018). arXiv: 1808.08865 [hep-ex].
[77] J. H. Christenson et al. "Evidence for the 2π Decay of the K_2^0 Meson". In: *Phys. Rev. Lett.* 13 (4 July 1964), pp. 138–140. DOI: 10.1103/PhysRevLett.13.138. URL: https://link.aps.org/doi/10.1103/PhysRevLett.13.138.
[78] A. D. Sakharov. "Violation of CP Invariance, C asymmetry, and baryon asymmetry of the universe". In: *Pisma Zh. Eksp. Teor. Fiz.* 5 (1967), pp. 32–35. DOI: 10.1070/PU1991v034n05ABEH002497.
[79] Makoto Kobayashi and Toshihide Maskawa. "CP Violation in the Renormalizable Theory of Weak Interaction". In: *Prog. Theor. Phys.* 49 (1973), pp. 652–657. DOI: 10.1143/PTP.49.652.
[80] LHCb collaboration. *Updated LHCb combination of the CKM angle γ*. LHCb-CONF-2020-003. 2020.
[81] LHCb collaboration, R. Aaij et al. "Simultaneous determination of CKM angle γ and charm mixing parameters". In: *JHEP* 12 (2021), p. 141. DOI: 10.1007/JHEP12(2021)141. arXiv: 2110.02350 [hep-ex].
[82] LHCb collaboration, R. Aaij et al. "Determination of the quark coupling strength $|V_{ub}|$ using baryonic decays". In: *Nature Physics* 11 (2015), p. 743. DOI: 10.1038/nphys3415. arXiv: 1504.01568 [hep-ex].
[83] LHCb collaboration, R. Aaij et al. "First observation of the decay $B_s^0 \to K^-\mu^+\nu_\mu$ and measurement of $|V_{ub}|/|V_{cb}|$". In: *Phys. Rev. Lett.* 126 (2021), p. 081804. DOI: 10.1103/PhysRevLett.126.081804. arXiv: 2012.05143 [hep-ex].

[84] LHCb collaboration, R. Aaij et al. "Measurement of the semileptonic *CP* asymmetry in B^0–$\bar{B}^0$ mixing". In: *Phys. Rev. Lett.* 114 (2015), p. 041601. DOI: 10.1103/PhysRevLett.114.041601. arXiv: 1409.8586 [hep-ex].

[85] LHCb collaboration, R. Aaij et al. "Measurement of the *CP* asymmetry in B_s^0–$\bar{B}_s^0$ mixing". In: *Phys. Rev. Lett.* 117 (2016), p. 061803. DOI: 10.1103/PhysRevLett.117.061803. arXiv: 1605.09768 [hep-ex].

[86] Tommaso Pajero. "Recent advances in charm mixing and CP violation at LHCb". In: *Mod. Phys. Lett. A* 37.24 (2022), p. 2230012. DOI: 10.1142/S0217732322300129. arXiv: 2208.05769 [hep-ex].

[87] Sascha Stahl. "Charm Physics at LHCb". In: LHCb-TALK-2021-031. 2021. URL: https://cds.cern.ch/record/2759106.

[88] LHCb collaboration, R. Aaij et al. "Observation of *CP* violation in charm decays". In: *Phys. Rev. Lett.* 122 (2019), p. 211803. DOI: 10.1103/PhysRevLett.122.211803. arXiv: 1903.08726 [hep-ex].

[89] LHCb collaboration, R. Aaij et al. "Measurement of the *CP* violation parameter A_Γ in $D^0 \to K^+K^-$ and $D^0 \to \pi^+\pi^-$ decays". In: *Phys. Rev. Lett.* 118 (2017), p. 261803. DOI: 10.1103/PhysRevLett.118.261803. arXiv: 1702.06490 [hep-ex].

[90] LHCb collaboration, R. Aaij et al. "Observation of the mass difference between neutral charm-meson eigenstates". In: *Phys. Rev. Lett.* 127 (2021), p. 111801. DOI: 10.1103/PhysRevLett.127.111801. arXiv: 2106.03744 [hep-ex].

[91] CMS and LHCb collaborations, V. Khachatryan et al. "Observation of the rare $B_s^0 \to \mu^+\mu^-$ decay from the combined analysis of CMS and LHCb data". In: *Nature* 522 (2015), p. 68. DOI: 10.1038/nature14474. arXiv: 1411.4413 [hep-ex].

[92] LHCb collaboration, R. Aaij et al. "Analysis of neutral *B*-meson decays into two muons". In: *Phys. Rev. Lett.* 128 (2022), p. 041801. DOI: 10.1103/PhysRevLett.128.041801. arXiv: 2108.09284 [hep-ex].

[93] LHCb collaboration, R. Aaij et al. "Measurement of the $B_s^0 \to \mu^+\mu^-$ decay properties and search for the $B^0 \to \mu^+\mu^-$ and $B_s^0 \to \mu^+\mu^-\gamma$ decays". In: *Phys. Rev.* D105 (2022), p. 012010. DOI: 10.1103/PhysRevD.105.012010. arXiv: 2108.09283 [hep-ex].

[94] LHCb collaboration, R. Aaij et al. "Test of lepton universality in beauty-quark decays". In: *Nature Physics* 18 (2022), p. 277. DOI: 10.1038/s41567-021-01478-8. arXiv: 2103.11769 [hep-ex].

[95] LHCb collaboration, R. Aaij et al. "Observation of $J/\psi p$ resonances consistent with pentaquark states in $\Lambda_b^0 \to J/\psi p K^-$ decays". In: *Phys. Rev. Lett.* 115 (2015), p. 072001. DOI: 10.1103/PhysRevLett.115.072001. arXiv: 1507.03414 [hep-ex].

[96] LHCb collaboration, R. Aaij et al. "Observation of a narrow pentaquark state, $P_c(4312)^+$, and of two-peak structure of the $P_c(4450)^+$". In: *Phys. Rev. Lett.* 122 (2019), p. 222001. DOI: 10.1103/PhysRevLett.122.222001. arXiv: 1904.03947 [hep-ex].

[97] LHCb collaboration, R. Aaij et al. "Search for time-dependent *CP* violation in $D^0 \to K^+K^-$ and $D^0 \to \pi^+\pi^-$ decays". In: *Phys. Rev.* D104 (2021), p. 072010. DOI: 10.1103/PhysRevD.104.072010. arXiv: 2105.09889 [hep-ex].

[98] LHCb collaboration, R. Aaij et al. "Model-independent study of structure in $B^+ \to D^+D^-K^+$ decays". In: *Phys. Rev. Lett.* 125 (2020), p. 242001. DOI: 10.1103/PhysRevLett.125.242001. arXiv: 2009.00025 [hep-ex].

[99] LHCb collaboration, R. Aaij et al. "Amplitude analysis of the $B^+ \to D^+D^-K^+$ decay". In: *Phys. Rev.* D102 (2020), p. 112003. DOI: 10.1103/PhysRevD.102.112003. arXiv: 2009.00026 [hep-ex].

[100] LHCb collaboration, R. Aaij et al. "Observation of an exotic narrow doubly charmed tetraquark". In: *Nature Physics* 18 (2022), p. 751. DOI: 10.1038/s41567-022-01614-y. arXiv: 2109.01038 [hep-ex].

[101] LHCb collaboration, R. Aaij et al. "Study of the doubly charmed tetraquark T_{cc}^+". In: *Nature Communications* 13 (2022), p. 3351. DOI: 10.1038/s41467-022-30206-w. arXiv: 2109.01056 [hep-ex].

[102] LHCb collaboration, R. Aaij et al. "Observation of structure in the J/ψ-pair mass spectrum". In: *Science Bulletin* 65.23 (2020), p. 1983. DOI: 10.1016/j.scib.2020.08.032. arXiv: 2006.16957 [hep-ex].

[103] LHCb collaboration, R. Aaij et al. "Determination of the $X(3872)$ meson quantum numbers". In: *Phys. Rev. Lett.* 110 (2013), p. 222001. DOI: 10.1103/PhysRevLett.110.222001. arXiv: 1302.6269 [hep-ex].

[104] LHCb collaboration, R. Aaij et al. "Study of the line shape of the $\chi_{c1}(3872)$ state". In: *Phys. Rev.* D102 (2020), p. 092005. DOI: 10.1103/PhysRevD.102.092005. arXiv: 2005.13419 [hep-ex].

[105] LHCb collaboration, R. Aaij et al. "Study of the $\psi_2(3823)$ and $\chi_{c1}(3872)$ states in $B^+ \to (J/\psi\pi^+\pi^-)K^+$ decays". In: *JHEP* 08 (2020), p. 123. DOI: 10.1007/JHEP08(2020)123. arXiv: 2005.13422 [hep-ex].

[106] Alexander L. Kagan et al. "Top LHCb Physics". In: *Phys. Rev. Lett.* 107 (2011), p. 082003. DOI: 10.1103/PhysRevLett.107.082003. arXiv: 1103.3747 [hep-ph].

[107] LHCb collaboration, R. Aaij et al. "First observation of top quark production in the forward region". In: *Phys. Rev. Lett.* 115 (2015), p. 112001. DOI: 10.1103/PhysRevLett.115.112001. arXiv: 1506.00903 [hep-ex].

[108] LHCb collaboration, R. Aaij et al. "Measurement of the W boson mass". In: *JHEP* 01 (2022), p. 036. DOI: 10.1007/JHEP01(2022)036. arXiv: 2109.01113 [hep-ex].

[109] LHCb collaboration, R. Aaij et al. "Measurement of the forward-backward asymmetry in $Z/\gamma^* \to \mu^+\mu^-$ decays and determination of the effective weak mixing angle". In: *JHEP* 11 (2015), p. 190. DOI: 10.1007/JHEP11(2015)190. arXiv: 1509.07645 [hep-ex].

[110] LHCb collaboration. "LHCb SMOG Upgrade". In: CERN-LHCC-2019-005 (2019).

[111] Saverio Mariani. "Heavy-ion and fixed-target physics at LHCb". In: *56th Rencontres de Moriond on QCD and High Energy Interactions*. May 2022. arXiv: 2205.03315 [hep-ex].

[112] LHCb collaboration, R. Aaij et al. "Measurement of antiproton production from antihyperon decays in pHe collisions at $\sqrt{s_{\rm NN}} = 110$ GeV". In: *Eur. Phys. J.* C83 (2023), p. 543. DOI: 10.1140/epjc/s10052-023-11673-x. arXiv: 2205.09009 [hep-ex].

[113] LHCb collaboration, R. Aaij et al. "Measurement of antiproton production in pHe collisions at $\sqrt{s_{\rm NN}} = 110$ GeV". In: *Phys. Rev. Lett.* 121 (2018), p. 222001. DOI: 10.1103/PhysRevLett.121.222001. arXiv: 1808.06127 [hep-ex].

[114] Daniel Craik et al. "LHCb future dark-sector sensitivity projections for Snowmass 2021". In: *Snowmass 2021*. Mar. 2022. arXiv: 2203.07048 [hep-ph].

[115] LHCb collaboration, R. Aaij et al. "Search for $A' \to \mu^+\mu^-$ decays". In: *Phys. Rev. Lett.* 124 (2020), p. 041801. DOI: 10.1103/PhysRevLett.124.041801. arXiv: 1910.06926 [hep-ex].

[116] LHCb collaboration, R. Aaij et al. "Search for dark photons produced in 13 TeV pp collisions". In: *Phys. Rev. Lett.* 120 (2018), p. 061801. DOI: 10.1103/PhysRevLett.120.061801. arXiv: 1710.02867 [hep-ex].

[117] LHCb collaboration, R. Aaij et al. "Search for long-lived scalar particles in $B^+ \to K^+\chi(\mu^+\mu^-)$ decays". In: *Phys. Rev.* D95 (2017), p. 071101. DOI: 10.1103/PhysRevD.95.071101. arXiv: 1612.07818 [hep-ex].

[118] G. Barrand et al. "GAUDI — A software architecture and framework for building HEP data processing applications". In: *Computer Physics Communications* 140.1 (2001). CHEP2000, pp. 45–55. ISSN: 0010-4655. DOI: https://doi.org/10.1016/S0010-4655(01)00254-5.

[119] Marco Clemencic et al. "Recent developments in the LHCb software framework gaudi". In: *Journal of Physics: Conference Series* 219.4 (Apr. 2010), p. 042006. DOI: 10.1088/1742-6596/219/4/042006.

[120] M Clemencic et al. "The LHCb simulation application, Gauss: Design, evolution and experience". In: *J. Phys. Conf. Ser.* 331 (2011), p. 032023. DOI: 10.1088/1742-6596/331/3/032023.

[121] LHCb Collaboration. *LHCb CPU Usage Forecast*. https://cds.cern.ch/record/2696552h. LHCb Collaboration, 2019.

[122] Mark Peter Whitehead. *A Palette of Fast Simulations in LHCb*. https://cds.cern.ch/record/2630475. July 2018.

[123] LHCb collaboration. "Computing Model of the Upgrade LHCb experiment". In: CERN-LHCC-2018-014 (2018).

[124] M Frank et al. "DD4hep: A Detector Description Toolkit for High Energy Physics Experiments". In: *Journal of Physics: Conference Series* 513.2 (June 2014), p. 022010. DOI: 10.1088/1742-6596/513/2/022010.

[125] Concezio Bozzi, Sébastien Ponce, and Stefan Roiser. "The core software framework for the LHCb Upgrade". In: *EPJ Web Conf.* 214 (2019), p. 05040. DOI: 10.1051/epjconf/201921405040.

[126] Rosen Matev. *A 30 MHz software trigger for the LHCb upgrade*. https://cds.cern.ch/record/2631781. July 2018.

[127] LHCb Collaboration, LHCb Collaboration. *Performance of the fast simulation interface in Gauss-on-Gaussino*. https://cds.cern.ch/record/2781378. Sept. 2021.

[128] B. G. Siddi and D. Müller. "Gaussino - a Gaudi-Based Core Simulation Framework". In: *2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*. 2019, pp. 1–4. DOI: 10.1109/NSS/MIC42101.2019.9060074.

[129] Dominik Müller. "Adopting new technologies in the LHCb Gauss simulation framework". In: *EPJ Web Conf.* 214 (2019), p. 02004. DOI: 10.1051/epjconf/201921402004.

[130] LHCb Collaboration, LHCb Collaboration. *Performance of a multithreaded prototype for the future LHCb simulation framework (Gauss-on-Gaussino)*. https://cds.cern.ch/record/2694003. Oct. 2019.

[131] Adam Morris et al. "The Gaussino core simulation software". In: WLCG/HSF Workshop 2024 (). URL: https://indi.to/wtqsD.

[132] Michał Mazurek. "Gauss and Gaussino: the LHCb simulation software and its new experiment agnostic core framework". In: (CHEP2023). LHCb-TALK-2022-174. URL: https://cds.cern.ch/record/2815786.

[133] Michał Mazurek. "From prototypes to large scale detectors: how to exploit the Gaussino simulation framework for detectors studies, with a detour into machine learning". In: (CHEP2023). LHCb-TALK-2023-110. URL: https://cds.cern.ch/record/2859941.

[134] Michał Mazurek, Gloria Corti, and Mateusz Kmieć. "Performance of the Gaussino CaloChallenge-compatible infrastructure for ML-based fast simulation in the LHCb Experiment". In: ACAT2024 (). URL: https://indi.to/FYGgf.

[135] Michał Mazurek, Marco Clemencic, and Gloria Corti. "Gauss and Gaussino: the LHCb simulation software and its new experiment agnostic core framework". In: *PoS* ICHEP2022 (Nov. 2022), p. 225. DOI: 10.22323/1.414.0225.

[136] LHCb collaboration. "LHCb computing: Technical Design Report". In: CERN-LHCC-2005-019 (2005).

[137] LHCb collaboration. "LHCb Upgrade Software and Computing". In: CERN-LHCC-2018-007 (2018).

[138] G. Amadio et al. "Offloading electromagnetic shower transport to GPUs". In: *J. Phys. Conf. Ser.* 2438.1 (2023), p. 012055. DOI: 10.1088/1742-6596/2438/1/012055. arXiv: 2209.15445 [hep-ex].

[139] S. C. Tognini et al. "*Celeritas*: GPU-accelerated particle transport for detector simulation in High Energy Physics experiments". In: *Snowmass 2021*. Mar. 2022. arXiv: 2203.09467 [physics.data-an].

[140] Adam C. S. Davis et al. "Optical Photon Simulation with Mitsuba3". In: (Sept. 2023). arXiv: 2309.12496 [physics.comp-ph].

[141] Dominik Müller et al. "ReDecay: A novel approach to speed up the simulation at LHCb". In: *Eur. Phys. J.* C78 (2018), p. 1009. DOI: 10.1140/epjc/s10052-018-6469-6. arXiv: 1810.10362 [hep-ex].

[142] LHCb Collaboration, LHCb Collaboration. *Performance of the Lamarr Prototype: the ultra-fast simulation option integrated in the LHCb simulation framework*. https://cds.cern.ch/record/2696310. Oct. 2019.

[143] Matteo Rama and Giacomo Vitali. "Calorimeter fast simulation based on hit libraries LHCb Gauss framework". In: *EPJ Web Conf.* 214 (2019), p. 02040. DOI: 10.1051/epjconf/201921402040.

[144] D. J. Lange. "The EvtGen particle decay simulation package". In: *Nucl. Instrum. Meth.* A462 (2001), pp. 152–155. DOI: 10.1016/S0168-9002(01)00089-4.

[145] I. Belyaev et al. "Handling of the generation of primary events in Gauss, the LHCb simulation framework". In: *J. Phys. Conf. Ser.* 331 (2011), p. 032047. DOI: 10.1088/1742-6596/331/3/032047.

[146] Chao-Hsi Chang et al. "BCVEGPY: An Event generator for hadronic production of the B_c meson". In: *Comput. Phys. Commun.* 159 (2004), pp. 192–224. DOI: 10.1016/j.cpc.2004.02.005. arXiv: hep-ph/0309120.

[147] Ruben Pozzi. "Upgrade of the LHCb simulation framework with advanced particle event generators. Uppgradering av LHCb:s simulationsverktyg med avancerade partikelgeneratorer". CERN-THESIS-2023-272. KTH Royal Institute of Technology, 2023. URL: http://cds.cern.ch/record/2883035.

[148] G. Barrand et al. "The LHCb detector description framework". In: *11th International Conference on Computing in High-Energy and Nuclear Physics*. Feb. 2000, pp. 96–100.

[149] Filip Bilandzija. "Visualization of geometry and simulated events for Gaussino and Gauss-on-Gaussino". In: (2022). URL: https://cds.cern.ch/record/2836877.

[150] *Geant4 Documentation*. 2024. URL: https://geant4.web.cern.ch/docs/.

[151] Edward Moyse et al. "The Phoenix event display framework". In: *EPJ Web Conf.* 251 (2021), p. 01007. DOI: 10.1051/epjconf/202125101007.

[152] *Three.js*. 2024. URL: https://github.com/mrdoob/three.js/.

[153] *Angular*. 2024. URL: https://github.com/angular/angular.

[154] Andreas Pappas. "New Web Based Event Data and Geometry Visualization for LHCb". 2021. URL: https://cds.cern.ch/record/2792618.

[155] *Sphinx*. 2024. URL: https://github.com/sphinx-doc/sphinx.

[156] *Gaussino Documentation*. 2024. URL: https://gaussino.docs.cern.ch/master/index.html.

[157] *Gauss-on-Gaussino Documentation*. 2024. URL: https://lhcb-gauss.docs.cern.ch/master/index.html.

[158] Jason Ansel et al. "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation". In: *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, Apr. 2024. DOI: 10.1145/3620665.3640366. URL: https://pytorch.org/assets/pytorch2-2.pdf.

[159] *ONNX: Open Neural Network Exchange*. https://github.com/onnx/onnx. 2019.

[160] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[161] Yangqing Jia et al. "Caffe: Convolutional Architecture for Fast Feature Embedding". In: *arXiv preprint arXiv:1408.5093* (2014).

[162] Andreas Adelmann et al. "New directions for surrogate models and differentiable programming for High Energy Physics detector simulation". In: *Snowmass 2021*. Mar. 2022. arXiv: 2203.08806 [hep-ph].

[163] Michele Faucci Giannelli and Rui Zhang. "CaloShowerGAN, a Generative Adversarial Networks model for fast calorimeter shower simulation". In: (Sept. 2023). arXiv: 2309.06515 [physics.ins-det].

[164] Matteo Barbetti. "Lamarr: LHCb ultra-fast simulation based on machine learning models deployed within Gauss". In: *21th International Workshop on Advanced Computing and Analysis Techniques in Physics Research: AI meets Reality*. Mar. 2023. arXiv: 2303.11428 [hep-ex].

[165] Ashish Vaswani et al. "Attention Is All You Need". In: *31st International Conference on Neural Information Processing Systems*. June 2017. arXiv: 1706.03762 [cs.CL].

[166] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: (June 2020). arXiv: 2006.11239 [cs.LG].

[167] Piyush Raikwar et al. "Transformers for Generalized Fast Shower Simulation". In: *EPJ Web Conf.* 295 (2024), p. 09039. DOI: 10.1051/epjconf/202429509039.

[168] Piyush Raikwar, Renato Paulo Da Costa Cardoso, Anna Zaborowska, Dalila Salamani, Kristina Jaruskova, Sofia Vallecorsa, Kyongmin Yeo, Vijay Ekambaram, Nam Nguyen, Jayant Kalagnanam, Mudhakar Srivatsa.

"CaloDiT: Diffusion with transformers for fast shower simulation". In: ACAT2024 (). URL: https://indi.to/kDFtx.

[169] Erik Buhmann et al. "CaloClouds II: ultra-fast geometry-independent highly-granular calorimeter simulation". In: *JINST* 19.04 (2024), P04020. DOI: 10.1088/1748-0221/19/04/P04020. arXiv: 2309.05704 [physics.ins-det].

[170] *CaloChallenge*. 2024. URL: https://calochallenge.github.io/homepage/.

[171] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: (Dec. 2013). arXiv: 1312.6114 [stat.ML].

[172] LHCb collaboration. *Throughput and resource usage of the LHCb upgrade HLT*. https://cds.cern.ch/record/271521. LHCB-FIGURE-2020-007. 2020.

[173] V Breton, N Brun, and P Perret. *A clustering algorithm for the LHCb electromagnetic calorimeter using a cellular automaton*. Tech. rep. LHCb-2001-123. Geneva: CERN, Sept. 2001. URL: https://cds.cern.ch/record/681262.

[174] Núria Valls Canudas et al. *Graph Clustering: a graph-based clustering algorithm for the electromagnetic calorimeter in LHCb*. 2022. DOI: 10.48550/ARXIV.2212.11061. URL: https://arxiv.org/abs/2212.11061.

[175] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2016. DOI: 10.1109/cvpr.2016.91. URL: https://doi.org/10.1109/cvpr.2016.91.

[176] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: 1804.02767 [cs.CV].

[177] Ross Girshick et al. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2014. DOI: 10.1109/cvpr.2014.81. URL: https://doi.org/10.1109/cvpr.2014.81.

[178] Ross Girshick. "Fast R-CNN". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, Dec. 2015. DOI: 10.1109/iccv.2015.169. URL: https://doi.org/10.1109/iccv.2015.169.

[179] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017. DOI: 10.1109/cvpr.2017.690. URL: https://doi.org/10.1109/cvpr.2017.690.

[180] Kaiwen Duan et al. "CenterNet: Keypoint Triplets for Object Detection". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2019. DOI: 10.1109/iccv.2019.00667. URL: https://doi.org/10.1109/iccv.2019.00667.

[181] Michał Mazurek, Blaise Raheem Delaney, and Joao Coelho. *Deep learning solutions for 2D calorimetric cluster reconstruction at LHCb*. Oct. 2020. URL: https://indi.to/pgYNG.

[182] Núria Valls Canudas et al. "Deep Learning approach to LHCb Calorimeter reconstruction using a Cellular Automaton". In: *EPJ Web Conf.* 251 (2021), p. 04008. DOI: 10.1051/epjconf/202125104008. URL: https://doi.org/10.1051/epjconf/202125104008.

[183] Steven Farrell et al. *Novel deep learning methods for track reconstruction*. 2018. arXiv: 1810.06111 [hep-ex].

[184] Steven Farrell et al. "The HEP.TrkX Project: deep neural networks for HL-LHC online and offline tracking". In: *EPJ Web of Conferences* 150

(2017). Ed. by C. Germain et al., p. 00003. DOI: 10.1051/epjconf/201715000003. URL: https://doi.org/10.1051/epjconf/201715000003.

[185] Xiangyang Ju et al. *Graph Neural Networks for Particle Reconstruction in High Energy Physics detectors*. 2020. arXiv: 2003.11603 [physics.ins-det].

[186] Blaise Raheem Delaney. "Determination of the CKM ratio $|V_{ub}|/|V_{cb}|$ using semileptonic B_c^+ decays at LHCb". PhD thesis. DOI: 10.17863/CAM.87305. URL: https://www.repository.cam.ac.uk/handle/1810/339884.

[187] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[188] Tobias Hurth. "Inclusive rare B decays". In: *5th International Symposium on Radiative Corrections: Applications of Quantum Field Theory to Phenomenology*. Apr. 2001. arXiv: hep-ph/0106050.

[189] Tobias Hurth and Thomas Mannel. "Direct CP violation in radiative B decays". In: *AIP Conf. Proc.* 602.1 (2001), pp. 212–219. DOI: 10.1063/1.1435929. arXiv: hep-ph/0109041.

[190] LHCb, Albert Puig. "Rare radiative decays at LHCb". In: *CERN Proc.* 1 (2018). Ed. by David d'Enterria, Albert de Roeck, and Michelangelo Mangano, p. 243. DOI: 10.23727/CERN-Proceedings-2018-001.243.

[191] O. Deschamps et al. "Photon and neutral pion reconstruction". In: (Sept. 2003).

[192] LHCb collaboration, R. Aaij et al. "Search for the decays $B^0 \to J/\psi\gamma$ and $B_s^0 \to J/\psi\gamma$". In: *Phys. Rev.* D92 (2015), p. 112002. DOI: 10.1103/PhysRevD.92.112002. arXiv: 1510.04866 [hep-ex].

[193] LHCb collaboration, R. Aaij et al. "Test of lepton universality using $B^+ \to K^+\ell^+\ell^-$ decays". In: *Phys. Rev. Lett.* 113 (2014), p. 151601. DOI: 10.1103/PhysRevLett.113.151601. arXiv: 1406.6482 [hep-ex].

[194] LHCb collaboration, R. Aaij et al. "Test of lepton universality with $B^0 \to K^{*0}\ell^+\ell^-$ decays". In: *JHEP* 08 (2017), p. 055. DOI: 10.1007/JHEP08(2017)055. arXiv: 1705.05802 [hep-ex].

[195] LHCb collaboration, Roel Aaij et al. "Measurement of the electron reconstruction efficiency at LHCb". In: *JINST* 14 (2019), P11023. DOI: 10.1088/1748-0221/14/11/P11023. arXiv: 1909.02957 [hep-ex].

A

Additional performance plots of the GAUSSINO and GAUSS-ON-GAUSSINO simulation frameworks

This chapter provides additional performance plots of the GAUSSINO and GAUSS-ON-GAUSSINO simulation frameworks focusing on the performance analysis of multi-threaded interfaces to PYTHIA8 and GEANT4, two critical components implemented in GAUSSINO and described in Chapter 3.

Multi-threaded interface to PYTHIA8 is represented by Figure A.1, which illustrates the throughput of the generation step as a function of the number of threads in GAUSSINO. Figure A.2 shows the time per event as a function of the number of threads, while Figure A.3 depicts the virtual memory consumption for the same setup.

The multi-threaded interface to PYTHIA8 was also tested in LHCb conditions with GAUSS-ON-GAUSSINO. This is highlighted in Figures A.4, A.5, and A.6 in 2016 beam conditions. Performance in 2022 beam conditions is shown in Figures A.7, A.8, and A.9.

The multi-threaded interface to GEANT4 is examined in the GAUSSINO standalone setup with the EXTERNALDETECTOR package in Figures A.10, A.11, and A.12. These figures illustrate the throughput, time per event, and virtual memory consumption, respectively, for both the generation and particle transport steps.

The detector simulation of Run 3 using DD4HEP and DETDESC geometry descriptions is presented for 2016 beam conditions in Figures A.13, A.14, and A.15, and for 2022 beam conditions in Figures A.16, A.17, and A.18.

Comparison of the relative time spent in each sub-detector between the GAUSS-ON-GAUSSINO and the former version of GAUSS (SIM10) is presented in Figure A.19. Relative time spent by each particle in the sub-detectors measured with respect to the time spent in that detector for SIM10 is shown in Figure A.20. The same plot for GAUSS-ON-GAUSSINO is shown in Figure A.21.

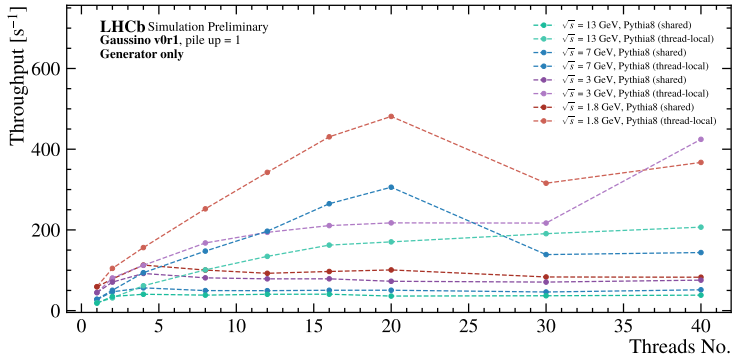


FIGURE A.1: Throughput of the generation step as a function of the number of threads in GAUSSINO using shared and thread-local interface to PYTHIA8 for head-on *pp*-collisions with varying beam energies.

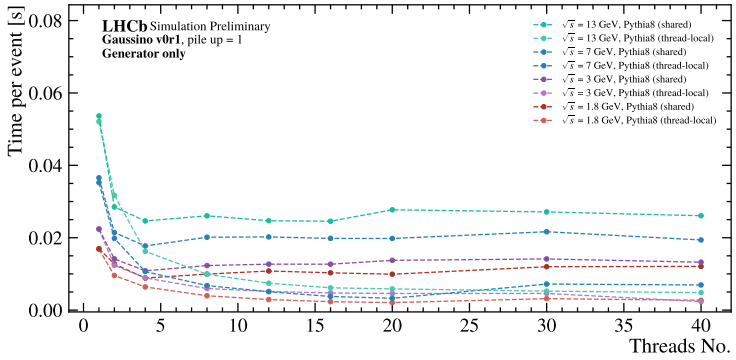


FIGURE A.2: Time per event of the generation step as a function of the number of threads in GAUSSINO using shared and thread-local interface to PYTHIA8 for head-on *pp*-collisions with varying beam energies.

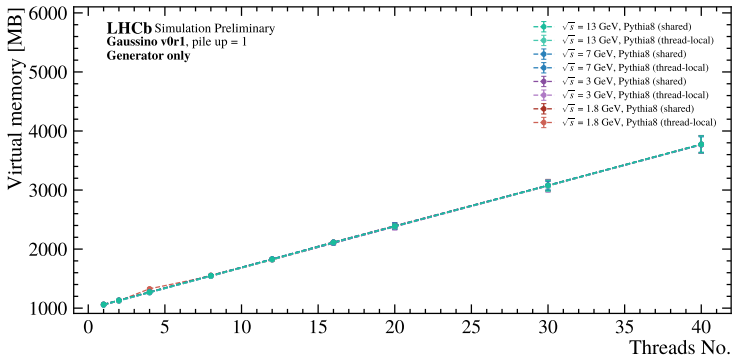


FIGURE A.3: Virtual memory consumption of the generation step as a function of the number of threads in GAUSSINO using shared and thread-local interface to PYTHIA8 for head-on *pp*-collisions with varying beam energies.

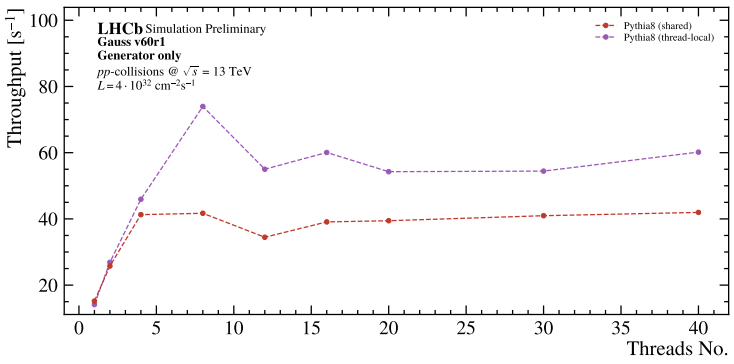


FIGURE A.4: Throughput of the generation step in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to PYTHIA8 for *pp*-collisions (minimum bias events) with the same beam conditions as during Run3 (2016) data-taking period in LHCb.

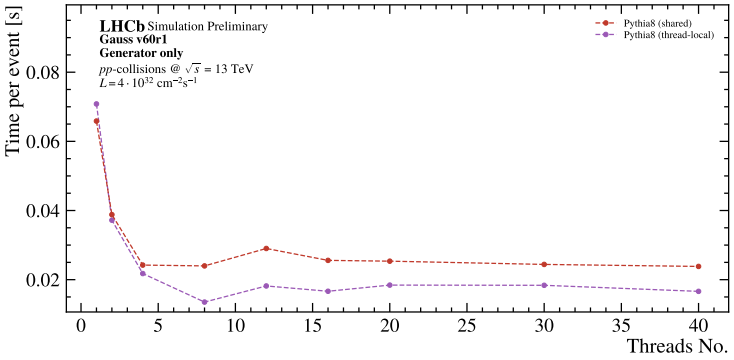


FIGURE A.5: Time per event of the generation step in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to PYTHIA8 for *pp*-collisions (minimum bias events) with the same beam conditions as during Run3 (2016) data-taking period in LHCb.

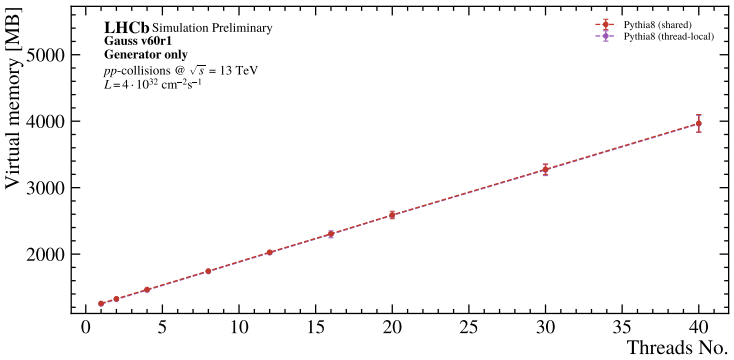


FIGURE A.6: Virtual memory consumption of the generation step in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to PYTHIA8 for *pp*-collisions (minimum bias events) with the same beam conditions as during Run3 (2016) data-taking period in LHCb.

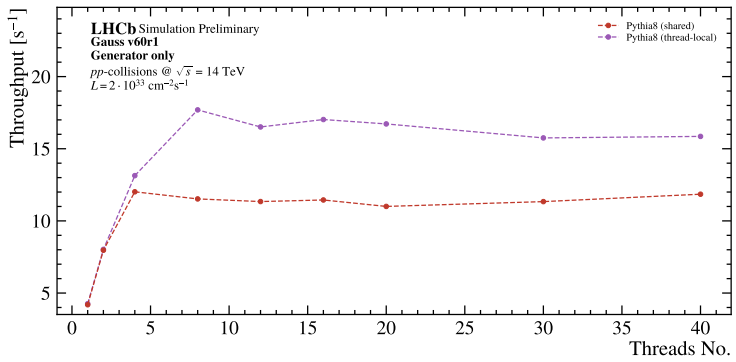


FIGURE A.7: Throughput of the generation step in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to PYTHIA8 for pp-collisions (minimum bias events) with the same beam conditions as during Run3 (2022) data-taking period in LHCb.

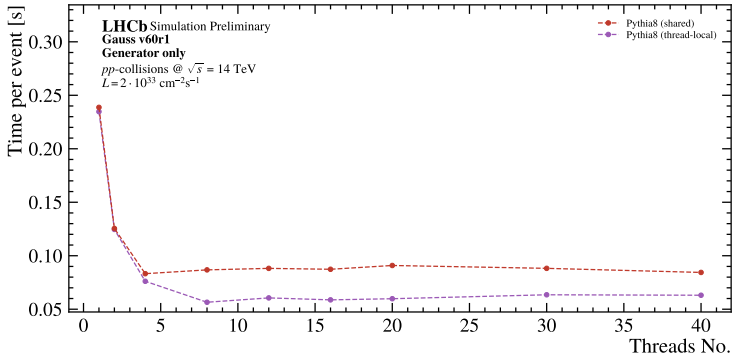


FIGURE A.8: Time per event of the generation step in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to PYTHIA8 for pp-collisions (minimum bias events) with the same beam conditions as during Run3 (2022) data-taking period in LHCb.

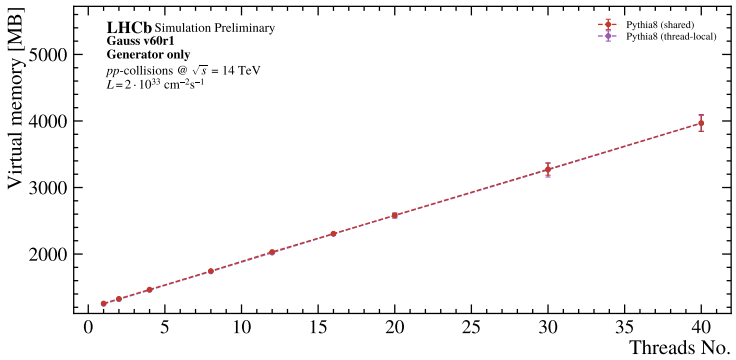


FIGURE A.9: Virtual memory consumption of the generation step in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to PYTHIA8 for pp-collisions (minimum bias events) with the same beam conditions as during Run3 (2022) data-taking period in LHCb.

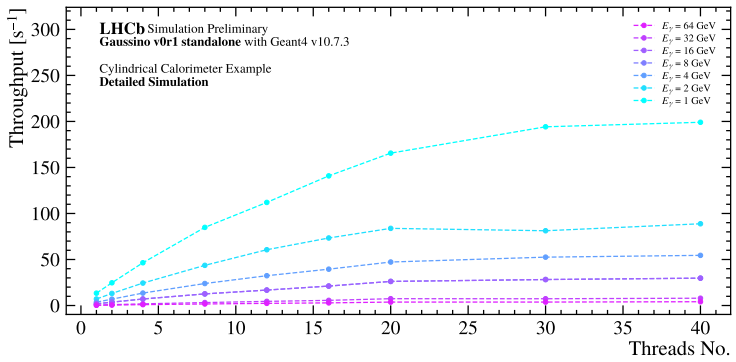


FIGURE A.10: Throughput of both the generation and particle transport steps as a function of the number of threads in GAUSSINO measured using a photon gun with varying energies shot at a simple cylindrical calorimeter.

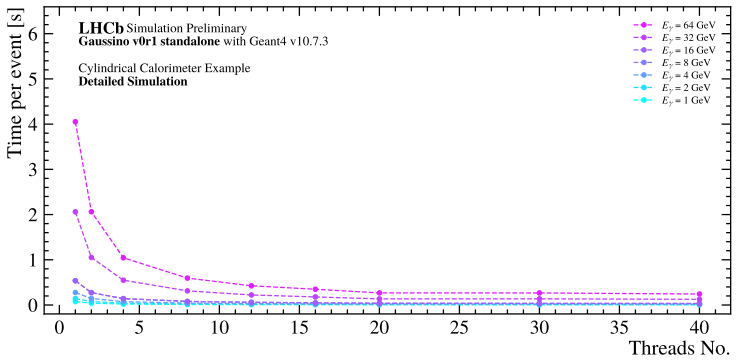


FIGURE A.11: Time per event of both the generation and particle transport steps as a function of the number of threads in GAUSSINO measured using a photon gun with varying energies shot at a simple cylindrical calorimeter.

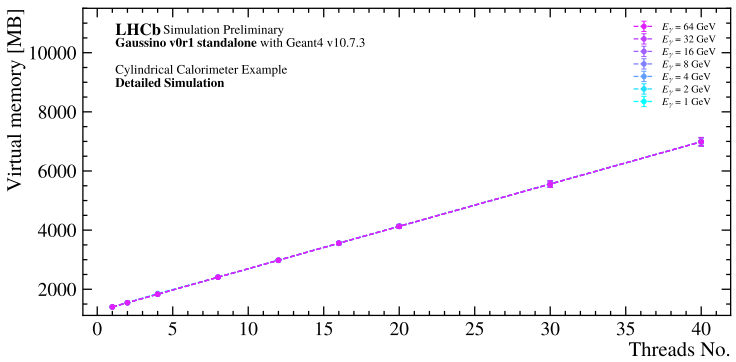


FIGURE A.12: Virtual memory consumption of both the generation and particle transport steps as a function of the number of threads in GAUSSINO measured using a photon gun with varying energies shot at a simple cylindrical calorimeter.

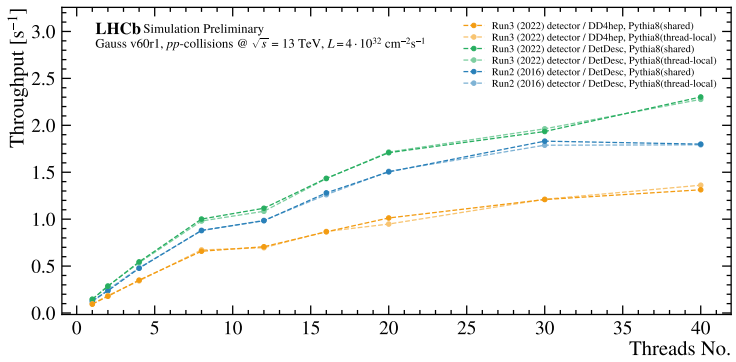


FIGURE A.13: Throughput of the generation and detector simulation steps in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to GEANT4 for pp-collisions (minimum bias events) with the same beam conditions as during Run3 (2016) data-taking period in LHCb.

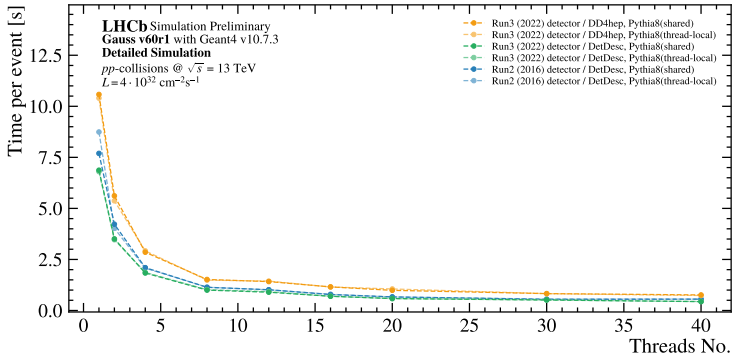


FIGURE A.14: Time per event of the generation and detector simulation steps in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to GEANT4 for pp-collisions (minimum bias events) with the same beam conditions as during Run3 (2016) data-taking period in LHCb.

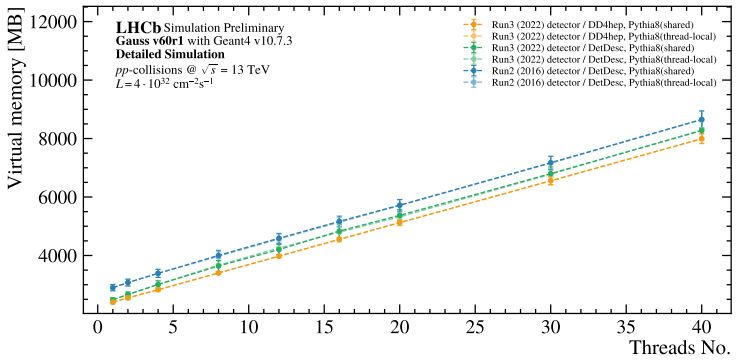


FIGURE A.15: Virtual memory consumption of the generation and detector simulation steps in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to GEANT4 for pp-collisions (minimum bias events) with the same beam conditions as during Run3 (2016) data-taking period in LHCb.

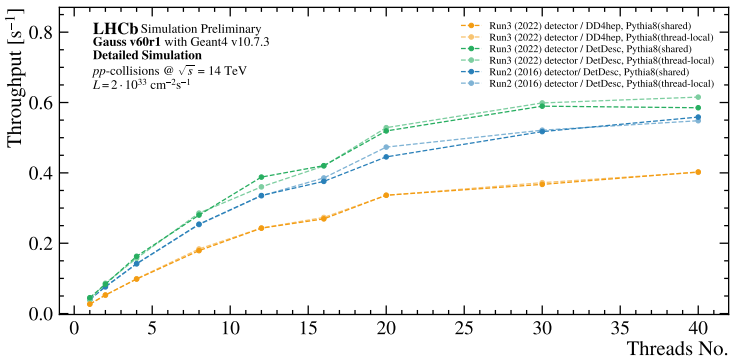


FIGURE A.16: Throughput of the generation and detector simulation steps in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to GEANT4 for *pp*-collisions (minimum bias events) with the same beam conditions as during Run3 (2022) data-taking period in LHCb.

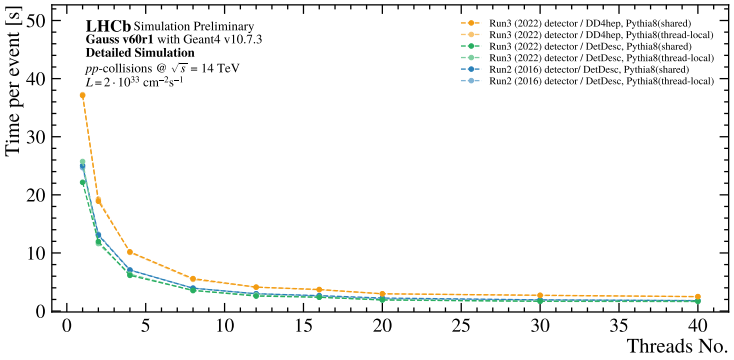


FIGURE A.17: Time per event of the generation and detector simulation steps in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to GEANT4 for *pp*-collisions (minimum bias events) with the same beam conditions as during Run3 (2022) data-taking period in LHCb.

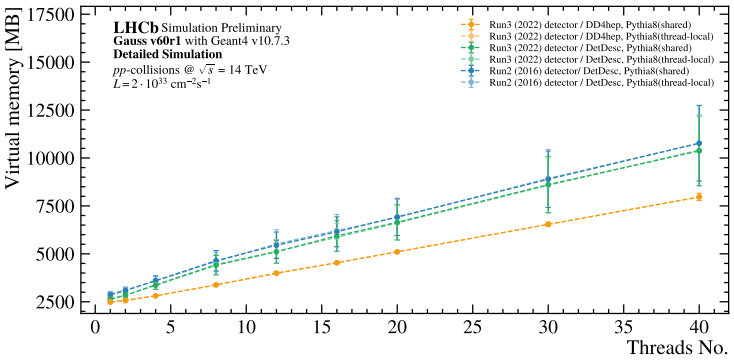
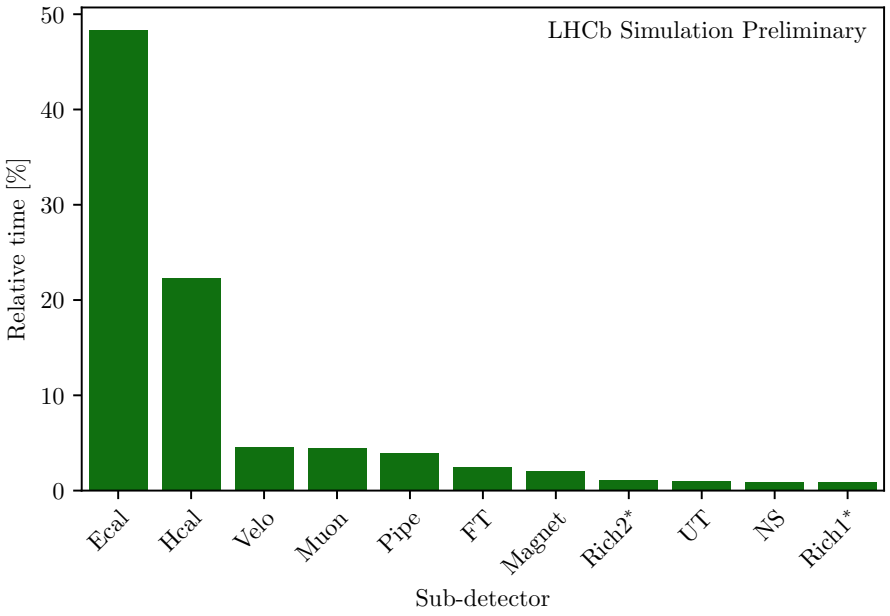
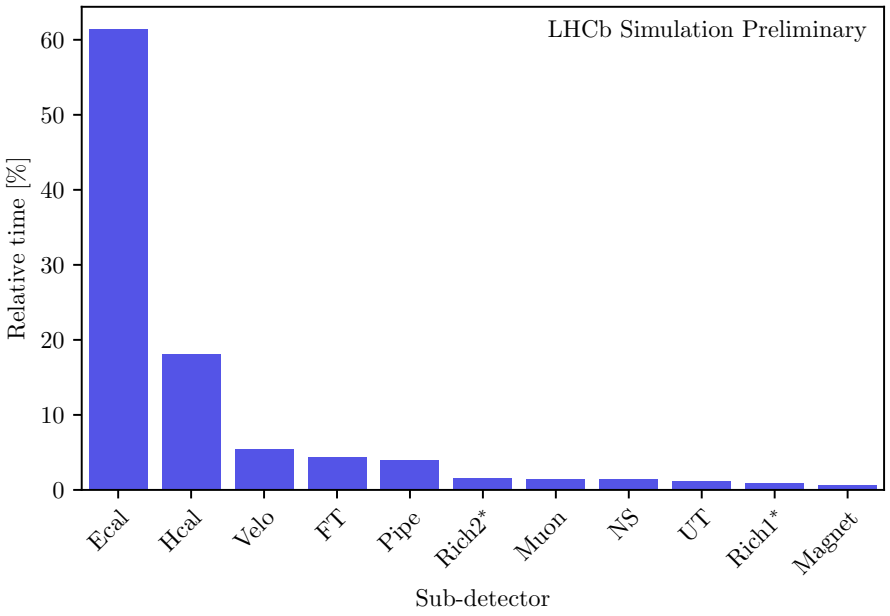


FIGURE A.18: Virtual memory consumption of the generation and detector simulation steps in GAUSS-ON-GAUSSINO as a function of the number of threads using shared and thread-local interface to GEANT4 for *pp*-collisions (minimum bias events) with the same beam conditions as during Run3 (2022) data-taking period in LHCb.

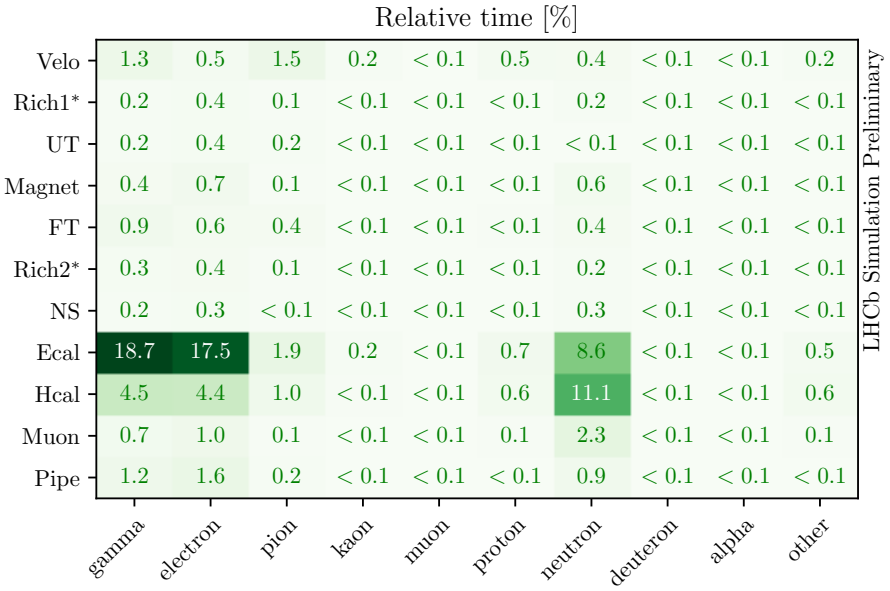


(A) SIM10 GAUSS framework with GEANT4 v10.6.

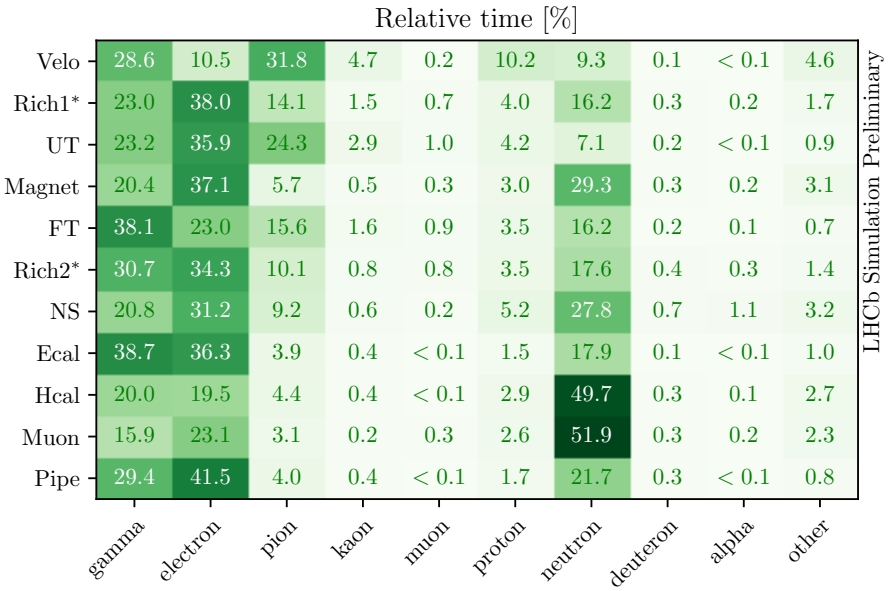


(B) New multi-threaded (1 thread) GAUSS-ON-GAUSSINO framework with GEANT4 v10.7.

FIGURE A.19: Relative time [127] spent in each sub-detector when simulating minimum bias events using different versions of the framework with the nominal beam conditions during the Run 3 data-taking period and the Run 3 geometry. The time spent in the calorimeters is very similar in the current and new versions of the framework as expected. Simulation of the optical photons in RICH1 and RICH2 was turned off in the SIM10 version of GAUSS as it is not yet available in GAUSS-ON-GAUSSINO, for easier comparison.

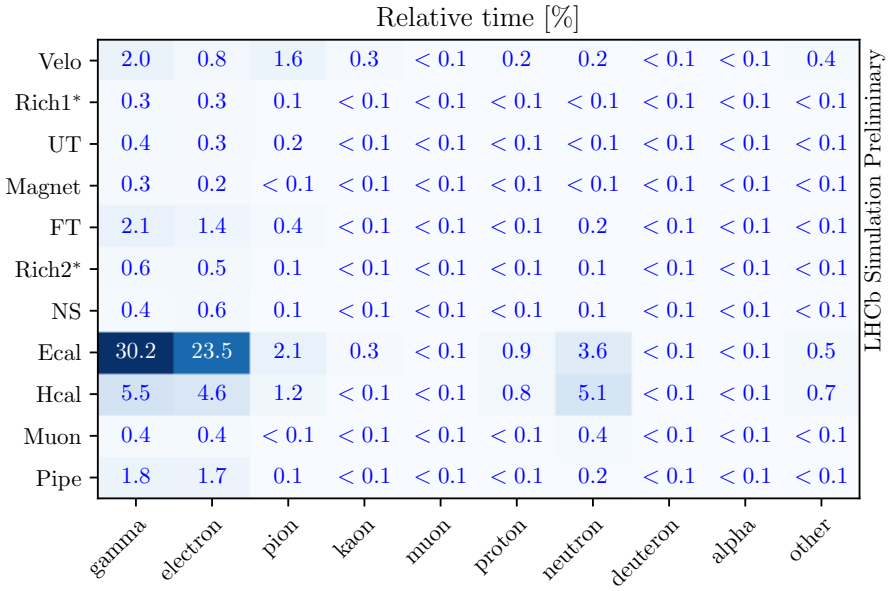


(A) Relative time spent by each particle in a sub-detector measured with respect to the total time spent on the simulation.

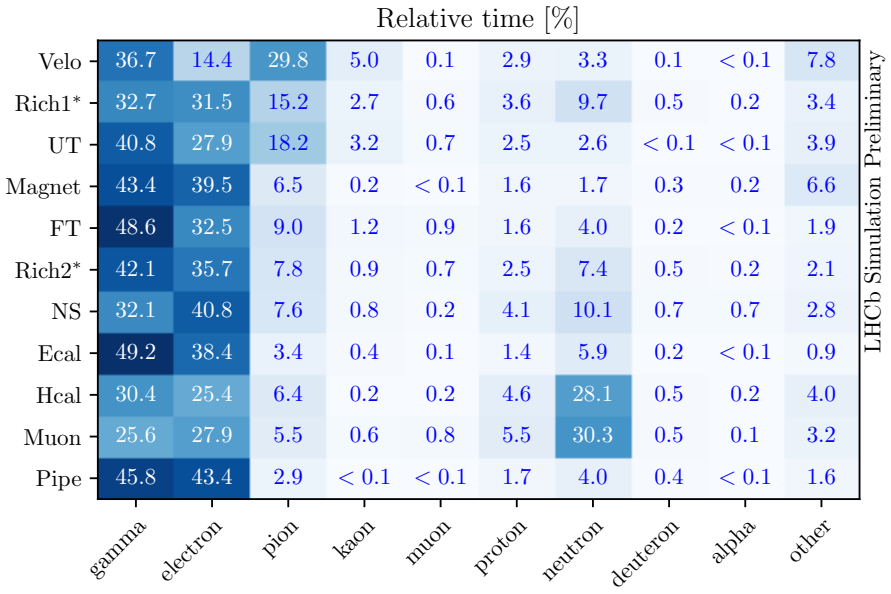


(B) Relative time by each particle in a sub-detector measured with respect to the time spent in that sub-detector.

FIGURE A.20: Detailed performance [127] of the SIM10 version of the GAUSS framework when simulating minimum bias events with the nominal beam conditions of the Run 3 data-taking period and the Run 3 geometry.



(A) Relative time by each particle in a sub-detector measured with respect to the total time spent on the simulation.



(B) Relative time by each particle in a sub-detector measured with respect to the time spent in that sub-detector.

FIGURE A.21: Detailed performance [127] of the new multi-threaded (1 thread) GAUSS-ON-GAUSSINO framework when simulating minimum bias events with the nominal beam conditions of the Run 3 data-taking period and the Run 3 geometry.

B

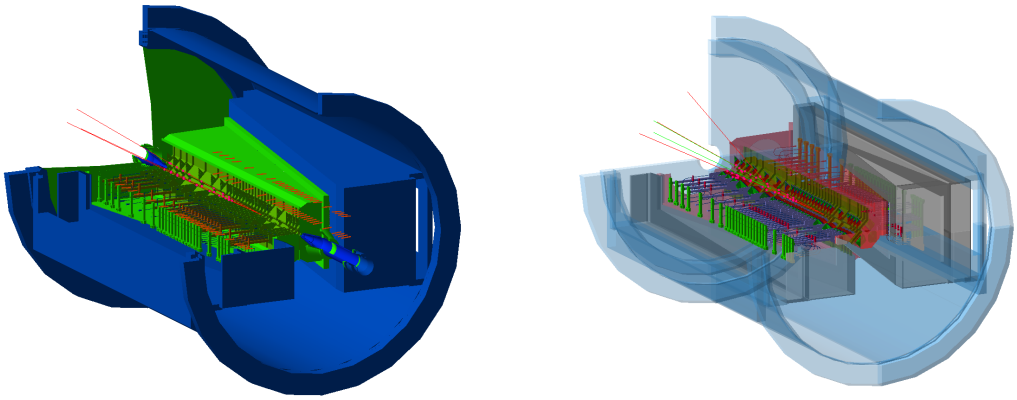
Selected sub-detector visualizations used in the validation of the DD4HEP detector description

This chapter provides a selection of sub-detector visualizations that were used for validating the DD4HEP detector description by comparing it with the DETDESC visualization as described in Chapter 3.

The comparison starts with the perspective view of the VP detector in both the DETDESC and DD4HEP visualizations, as shown in Figure B.1. Figure B.2 provides a zoomed-in perspective of the same detector for closer examination. The downstream view of the VP is shown in Figure B.3, followed by a zoomed-in version in Figure B.4. Lastly, Figure B.5 presents the A-side view of the VP detector in both visualizations.

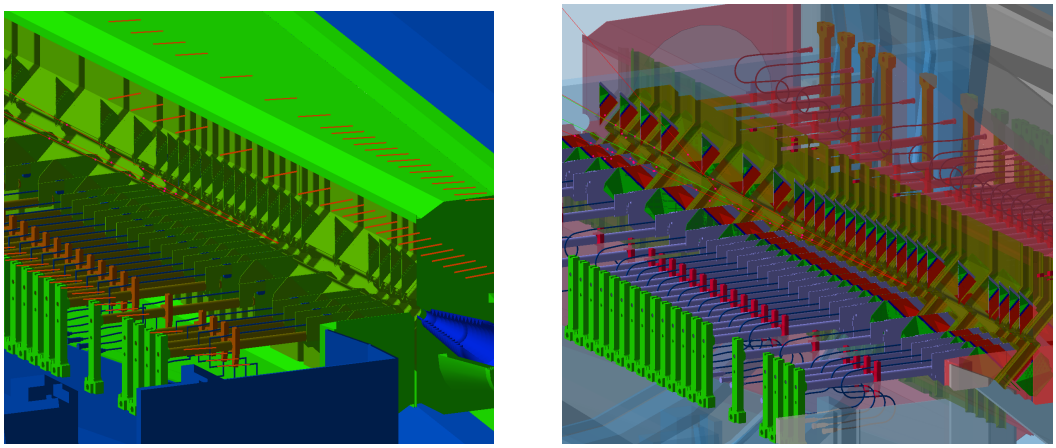
The next set of visualizations focuses on the FT detector. A perspective view is shown in Figure B.6, followed by the front view in Figure B.7, and the side view in Figure B.8.

Figures presented in this chapter are only a small subset of all the figures that were used to validate the new detector description. Missing volumes of the neutron shielding observed next to the FT detector, as well as missing volumes of the beam pipe in Figures representing the VELO detector, are some examples showing how indispensable the visualization tools in the simulation framework are.



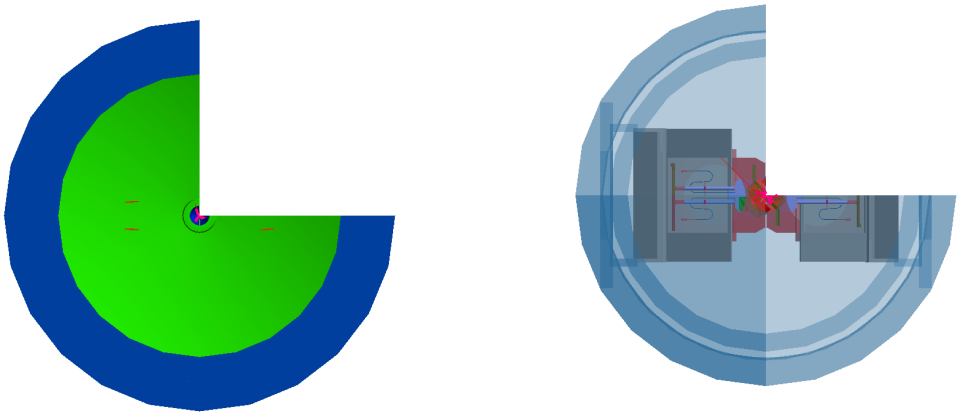
(A) VP in DETDESC (B) VP in DD4HEP

FIGURE B.1: Perspective view of the VP detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



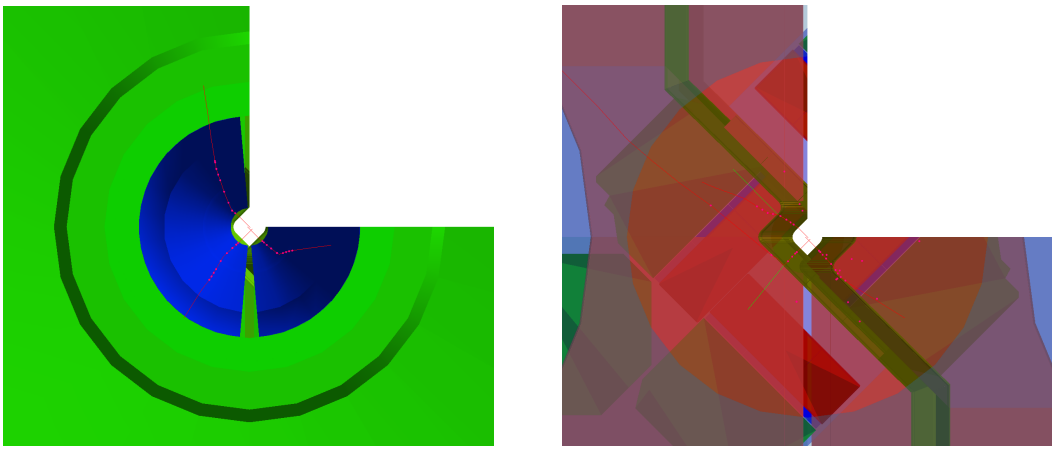
(A) VP in DETDESC (B) VP in DD4HEP

FIGURE B.2: Zoomed-in perspective view of the VP detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



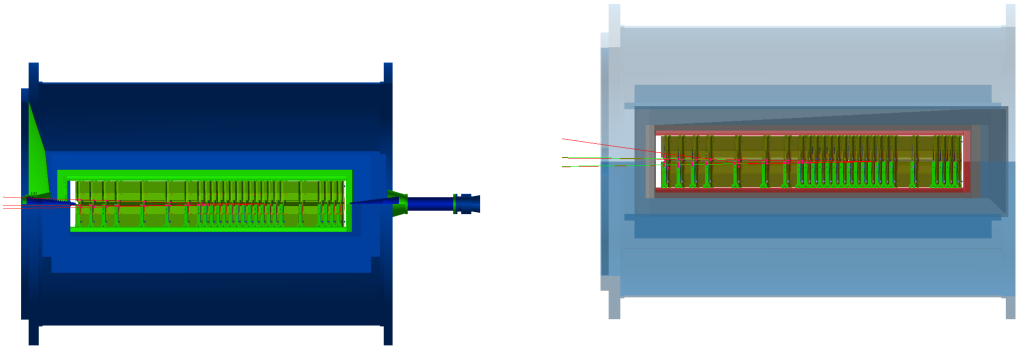
(A) VP in DETDESC (B) VP in DD4HEP

FIGURE B.3: Downstream view of the VP detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



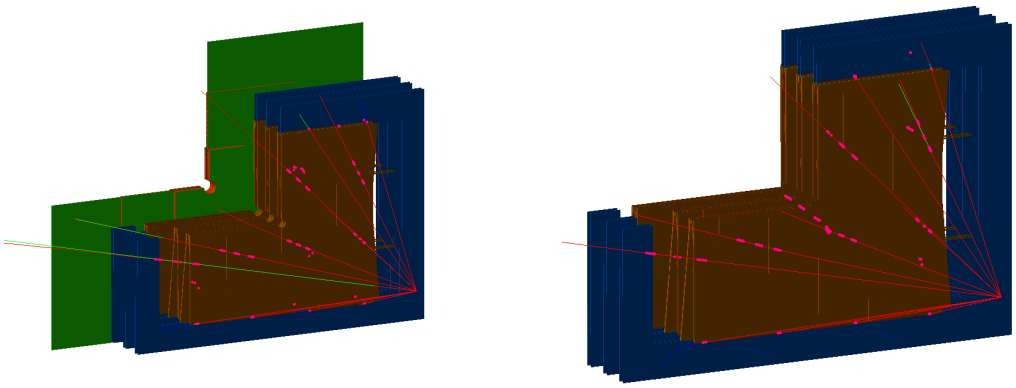
(A) VP in DETDESC (B) VP in DD4HEP

FIGURE B.4: Zoomed-in downstream view of the VP detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



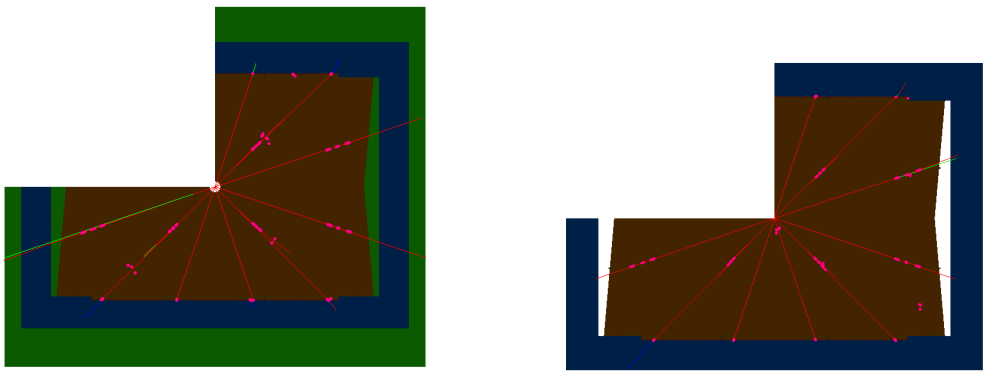
(A) VP in DETDESC (B) VP in DD4HEP

FIGURE B.5: A-side view of the VP detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



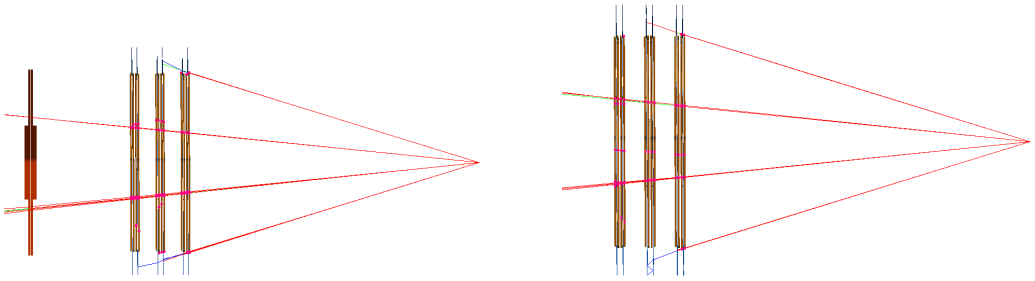
(A) FT in DETDESC (B) FT in DD4HEP

FIGURE B.6: Perspective view of the FT detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



(A) FT in DETDESC (B) FT in DD4HEP

FIGURE B.7: Front view of the FT detector in the DETDESC and DD4HEP visualizations (status as of August 2022).



(A) FT in DETDESC (B) FT in DD4HEP

FIGURE B.8: Side view of the FT detector in the DETDESC and DD4HEP visualizations (status as of August 2022).

C

Additional performance plots of the interface to PyTorch and ONNXRuntime backends in GAUSSINO

This chapter provides additional performance plots of the interface to PyTorch and ONNXRuntime backends implemented in the GAUSSINO framework, as described in Chapter 4. These plots focus on comparing the performance of the PyTorch and ONNXRuntime backends across various configurations, specifically the number of intra-op and inter-op threads, which are key factors in optimizing machine learning inference libraries.

Figure C.1 compares the total simulation time per event for the PyTorch and ONNX backends using different numbers of inter-op threads with one intra-op thread. Figure C.2 presents the inference throughput for the ONNX backend, while Figure C.3 shows the throughput per thread.

Inference time comparisons for the ONNX backend are given in Figure C.4 (total inference time per event) and Figure C.5 (time per thread). The total simulation throughput and time for the ONNX backend are shown in Figures C.6 and C.7, respectively. Figure C.8 provides a comparison of virtual memory usage.

For the PyTorch backend, inference throughput is compared in Figures C.9 (total) and C.10 (per thread). Figures C.11 and C.12 show inference time comparisons. The total simulation throughput and time for PyTorch are displayed in Figures C.13 and C.14, while virtual memory usage is depicted in Figure C.15.

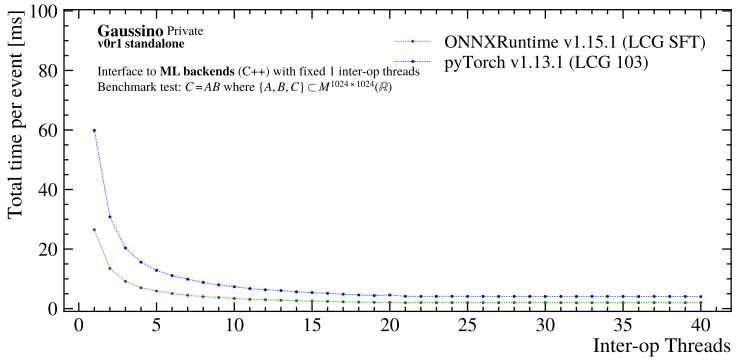


FIGURE C.1: Comparison of the total time per event for the PyTorch and ONNX backends in GAUSSINO with different numbers of inter-op threads and one intra-op thread.

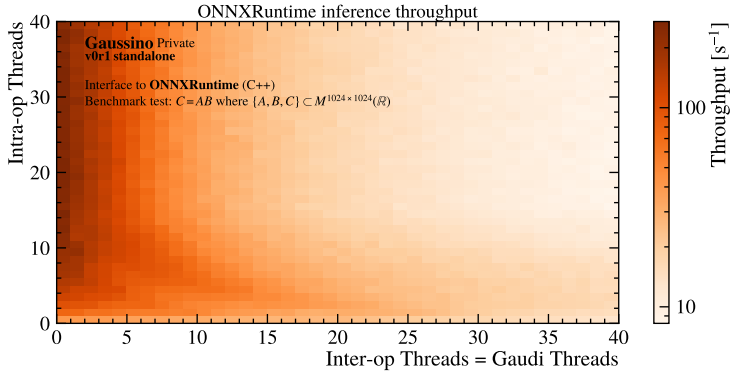


FIGURE C.2: Comparison of the inference throughput for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

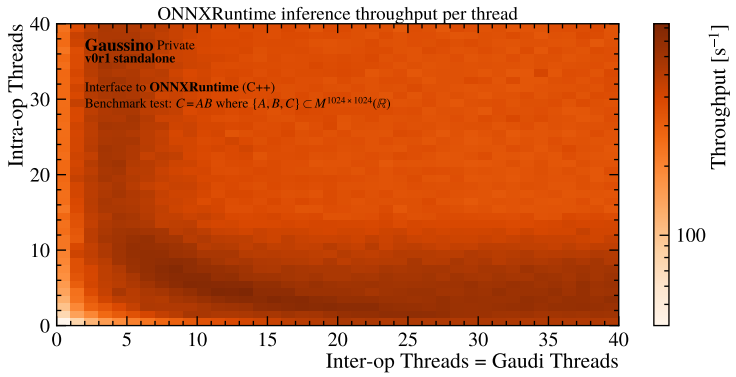


FIGURE C.3: Comparison of the inference throughput per thread for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

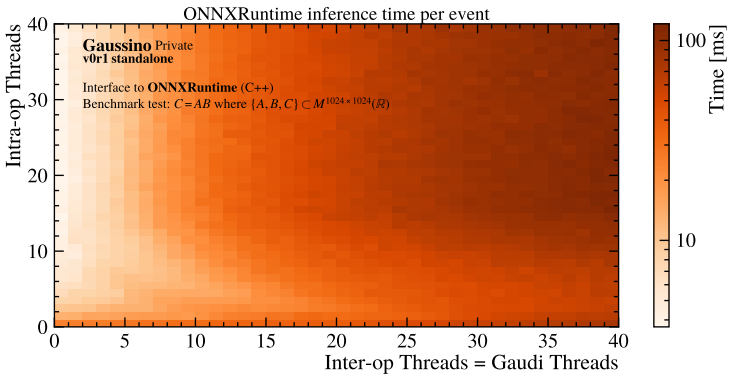


FIGURE C.4: Comparison of the inference time per event for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

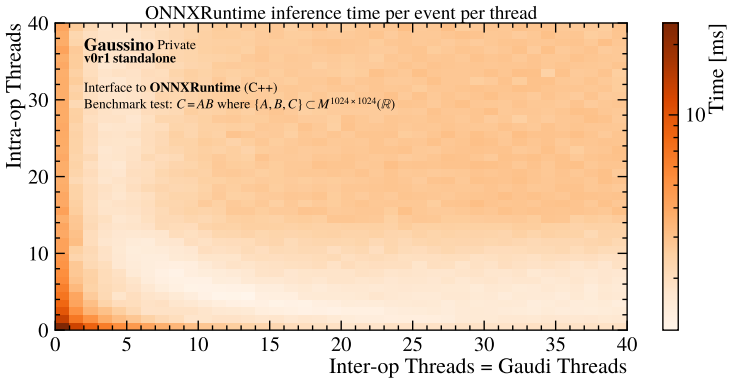


FIGURE C.5: Comparison of the inference time per event per thread for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

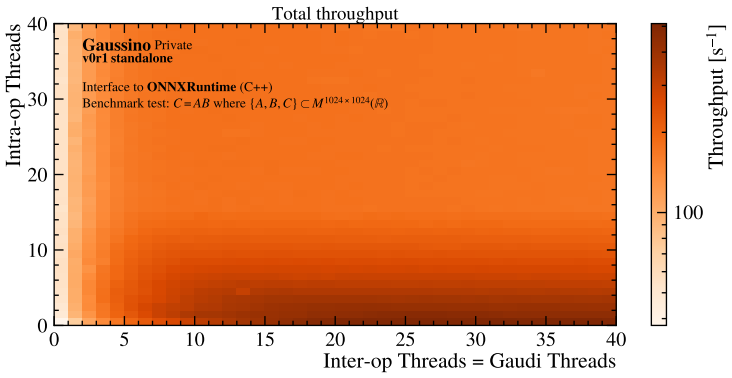


FIGURE C.6: Comparison of the total simulation throughput for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

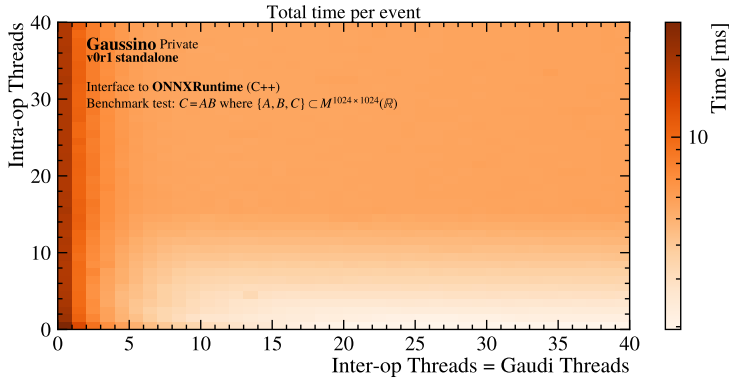


FIGURE C.7: Comparison of the total simulation time per event for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

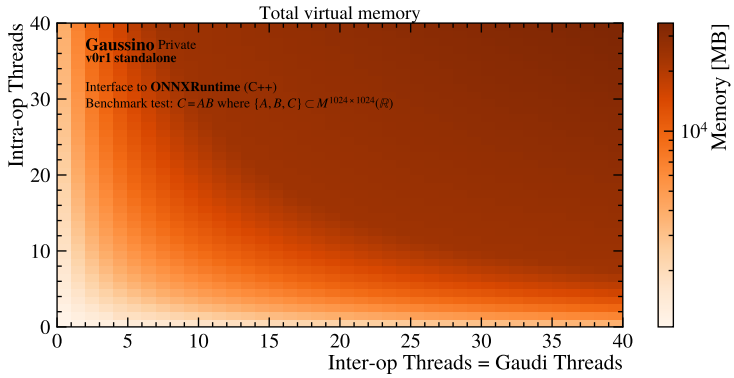


FIGURE C.8: Comparison of the total virtual memory usage for the ONNX backend in GAUSSINO with different number of intra-op and inter-op threads.

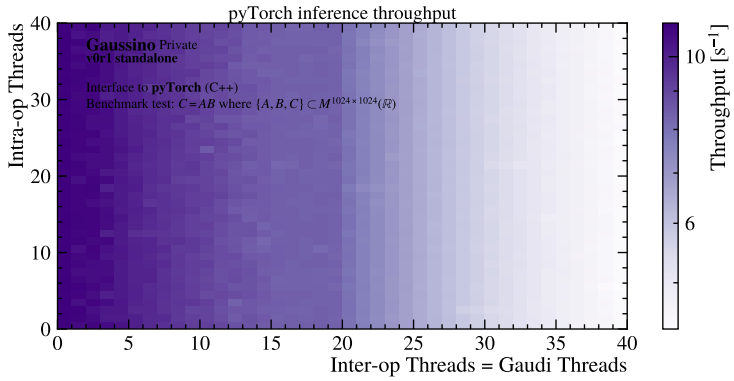


FIGURE C.9: Comparison of the inference throughput for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

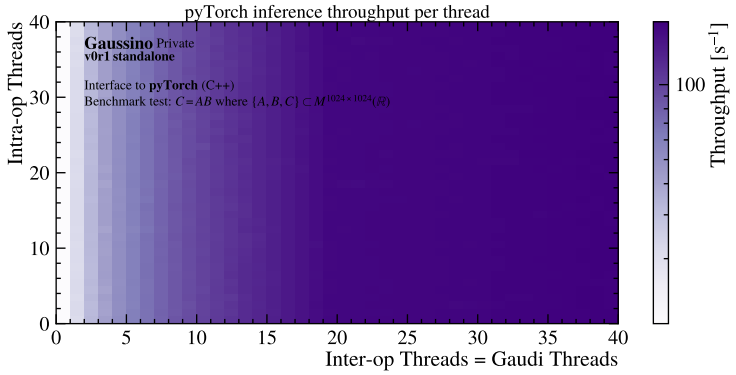


FIGURE C.10: Comparison of the inference throughput per thread for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

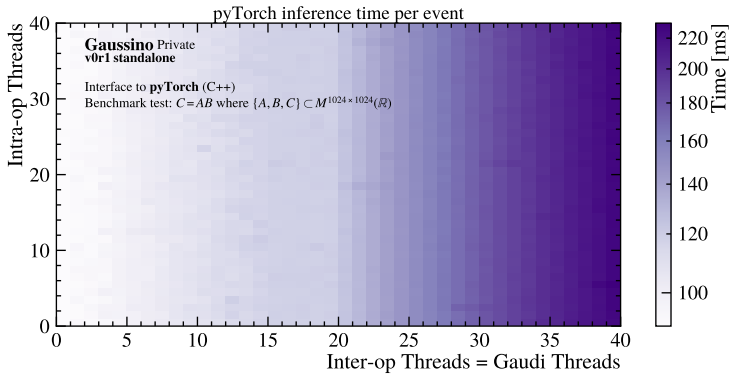


FIGURE C.11: Comparison of the inference time per event for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

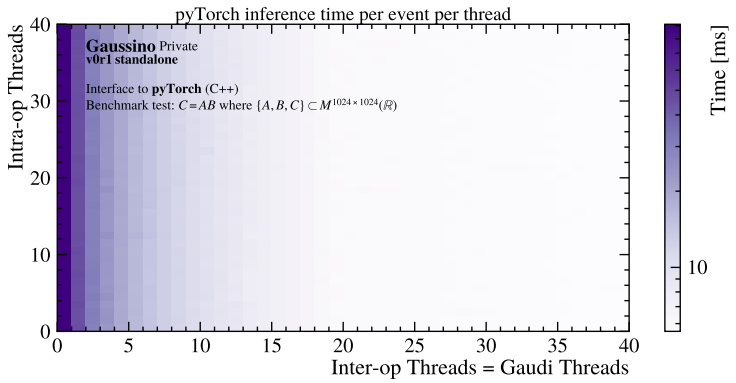


FIGURE C.12: Comparison of the inference time per event per thread for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

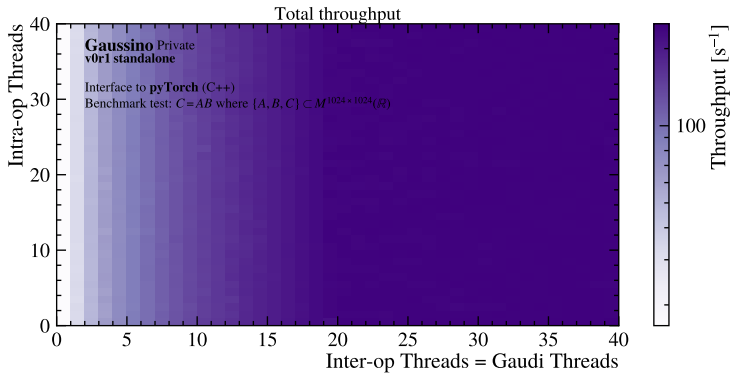


FIGURE C.13: Comparison of the total simulation throughput for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

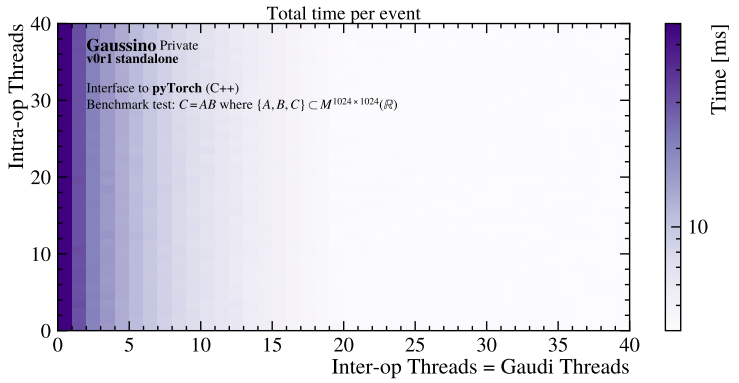


FIGURE C.14: Comparison of the total simulation time per event for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

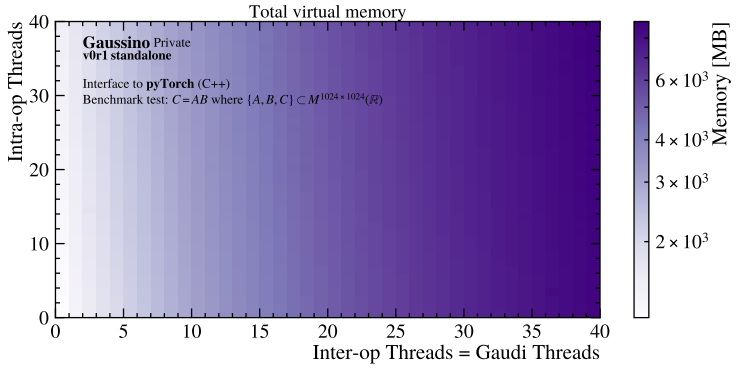


FIGURE C.15: Comparison of the total virtual memory usage for the PyTorch backend in GAUSSINO with different number of intra-op and inter-op threads.

D

Additional performance plots of the ML-based fast simulation in GAUSSINO and GAUSS-ON-GAUSSINO

This chapter provides additional performance plots of the ML-based fast simulation implemented in the GAUSSINO and GAUSS-ON-GAUSSINO simulation frameworks, as described in Chapter 4. The plots focus on the performance of the CALOML+VAE model.

Figure D.1 shows the throughput of the ML-based fast simulation (VAE model) when tested on a simple cylindrical calorimeter with varying photon energies. Figure D.2 provides the corresponding time per event, while Figure D.3 illustrates the potential speedup achieved using this fast simulation model. Virtual memory consumption is depicted in Figure D.4.

The longitudinal and lateral profile distributions of a modified VAE model, trained on a CALOCHALLENGE-compatible dataset produced in GAUSSINO, are presented in Figures D.5 and D.6, respectively.

Monitoring output for fast simulated 10 GeV electrons at two angles ($\theta = 3.36°$ and $\theta = 12.7°$) with both retrained and non-retrained VAE models is shown in Figures D.7, D.8, D.9, and Figure D.10 for fast simulated photons.

Visualization of the ML-based fast simulation tested on simple cylindrical and planar calorimeters is presented in Figures D.11 and D.12.

Figures D.13, D.14, and D.15 illustrate the validation of the ML-based fast simulation, showing comparisons between energy deposits left by ML-based and GEANT4-based simulations in the electromagnetic calorimeter, across different particle momenta and positions.

Finally, Figures D.16a, D.16c, D.16d, D.16e, and D.16f present the validation results for the ML-based fast simulation on LHCb minimum bias events.

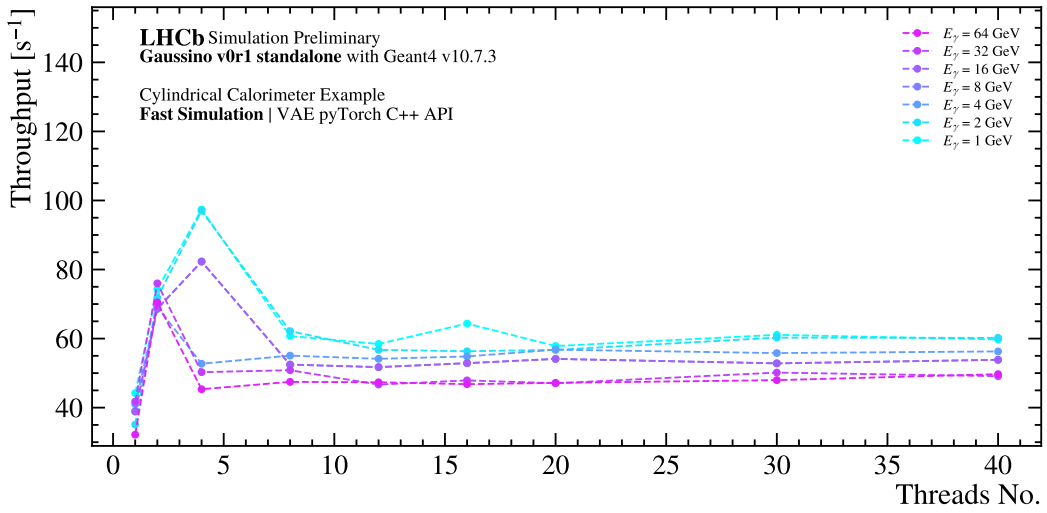


FIGURE D.1: Throughput of the ML-based fast simulation (VAE model) tested on a simple cylindrical calorimeter with varying photon energies.

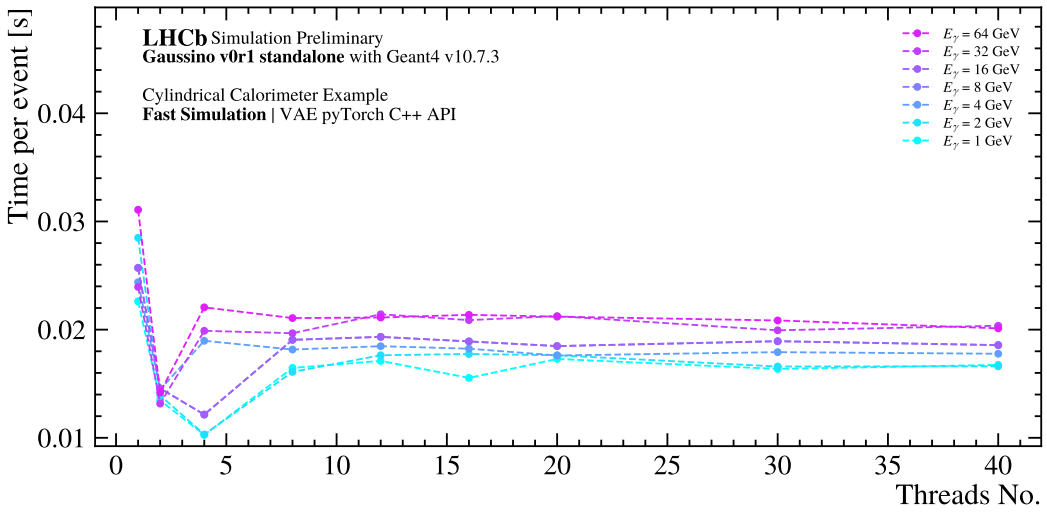


FIGURE D.2: Time per event of the ML-based fast simulation (VAE model) tested on a simple cylindrical calorimeter with varying photon energies.

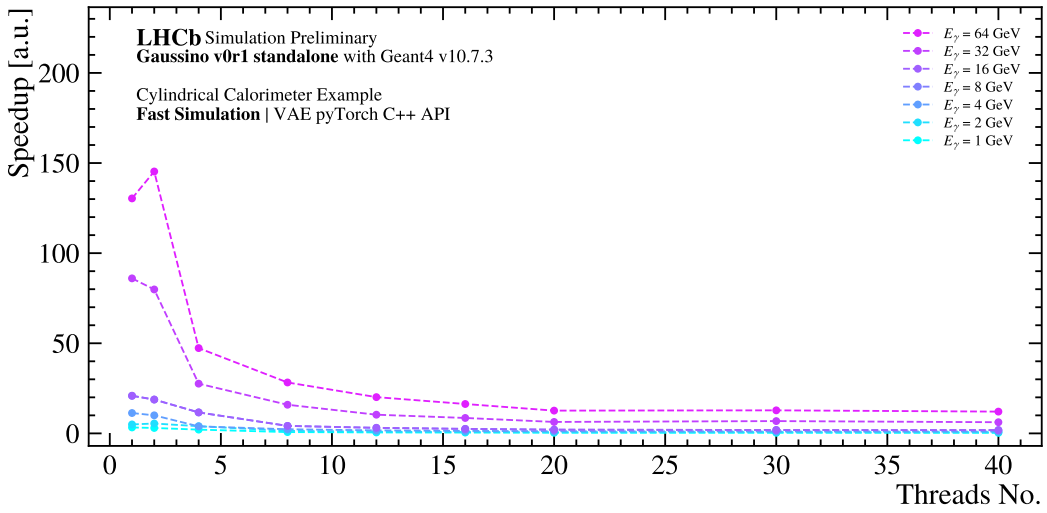


FIGURE D.3: Possible speedup obtained with the ML-based fast simulation (VAE model) tested on a simple cylindrical calorimeter with varying photon energies.

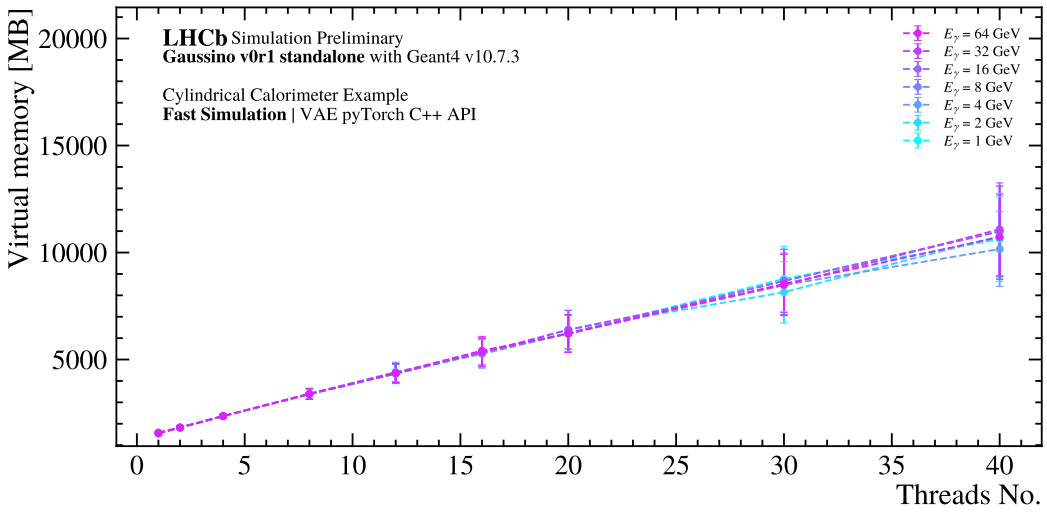


FIGURE D.4: Virtual memory usage of the ML-based fast simulation (VAE model) tested on a simple cylindrical calorimeter with varying photon energies.

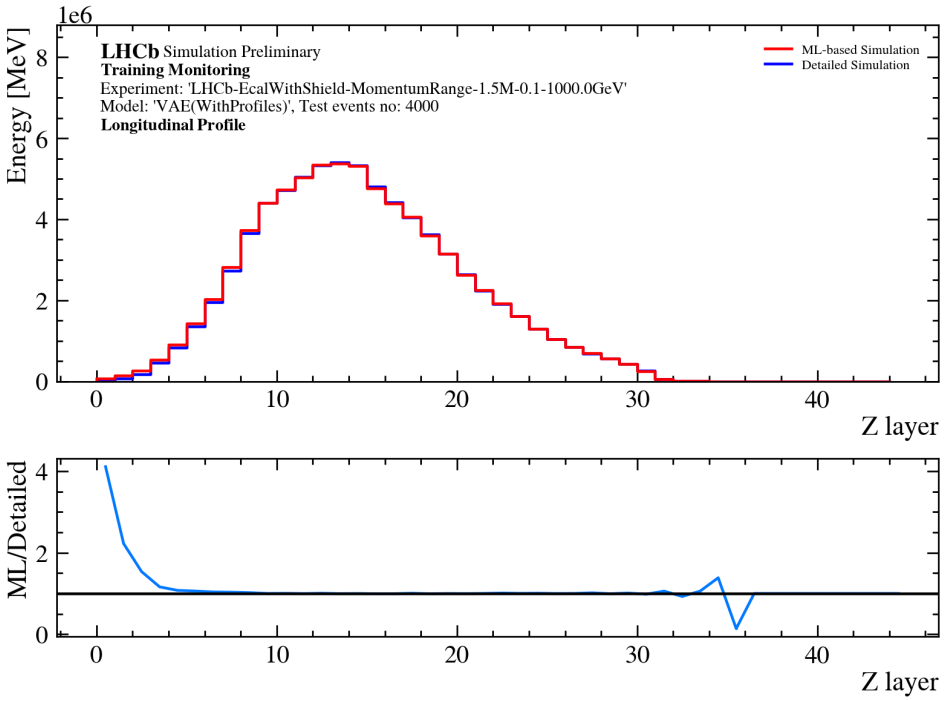


FIGURE D.5: Longitudinal profile distribution of a modified VAE model (VAEWithProfiles) trained on the CALOCHALLENGE-compatible dataset produced in GAUSSINO.

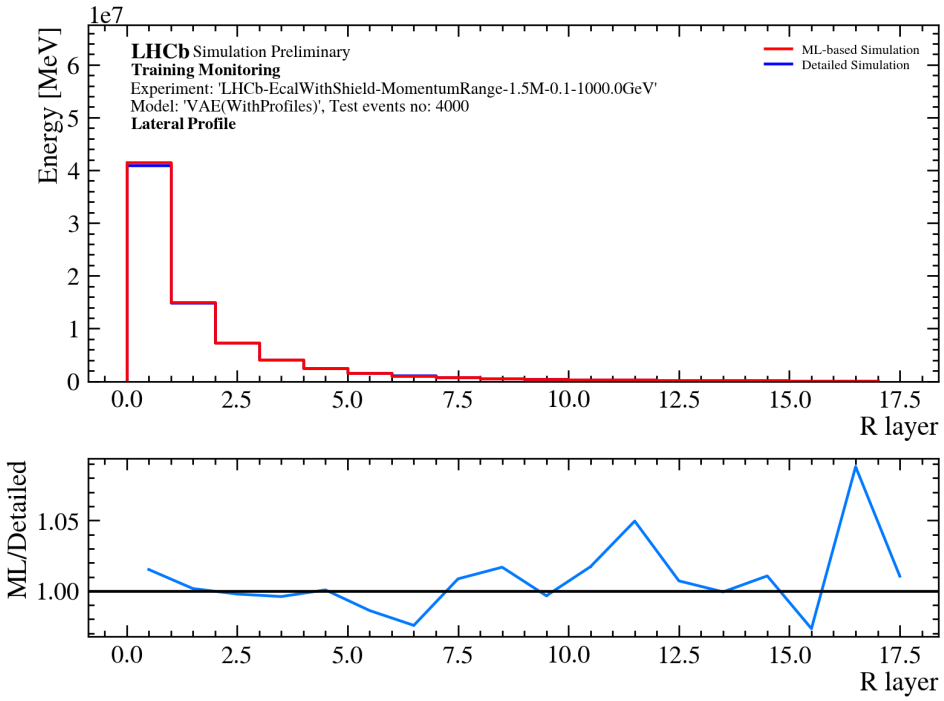


FIGURE D.6: Lateral profile distribution of a modified VAE model (VAEWithProfiles) trained on the CALOCHALLENGE-compatible dataset produced in GAUSSINO.

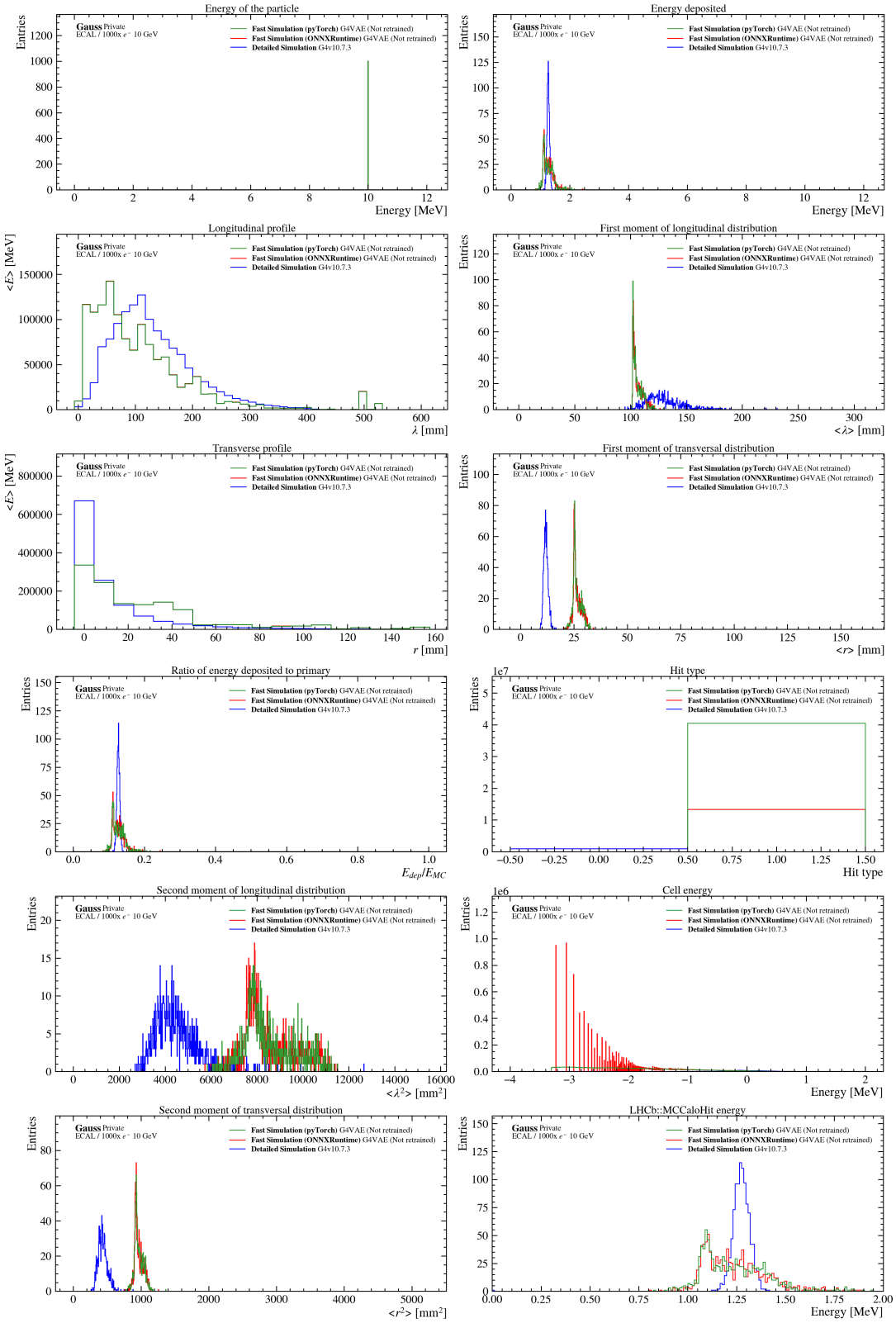


FIGURE D.7: Output plots of the monitoring algorithm of fast simulated 10 GeV electrons at $\theta = 3.36$ with not a retrained VAE model.

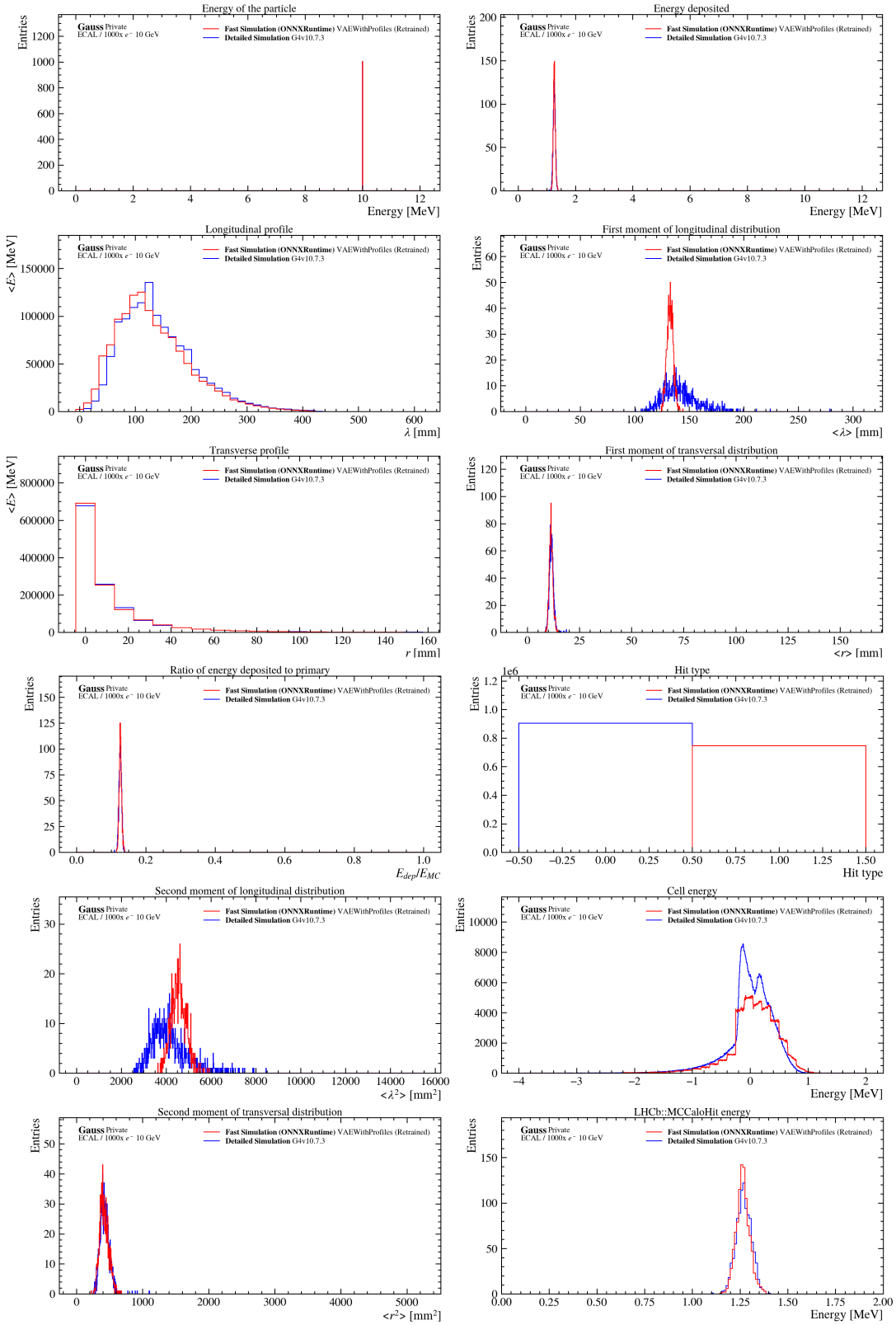


FIGURE D.8: Output plots of the monitoring algorithm of fast simulated 10 GeV electrons at $\theta = 3.36$ with a retrained VAE model.

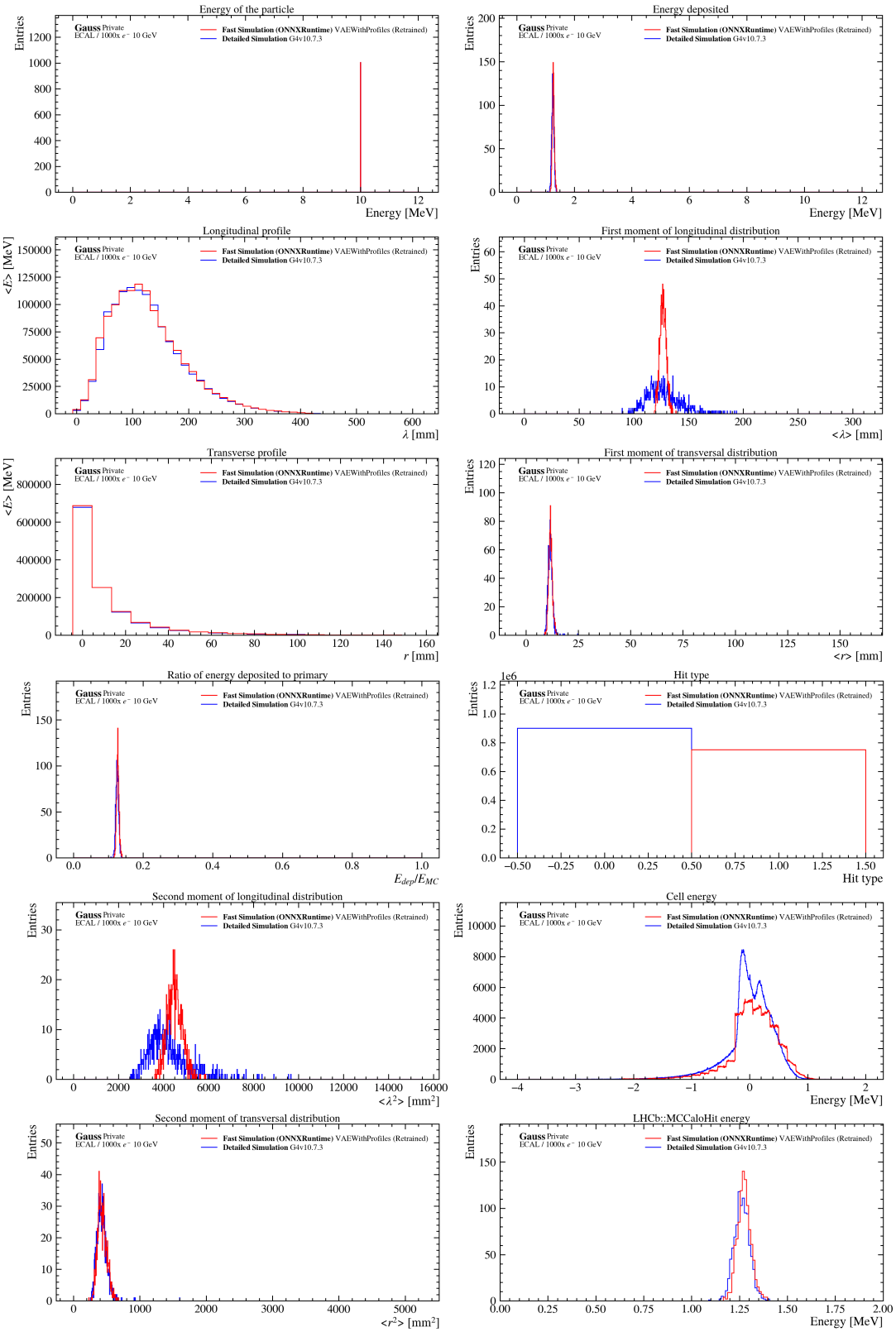


FIGURE D.9: Output plots of the monitoring algorithm of fast simulated 10 GeV electrons at $\theta = 12.7$ with a retrained VAE model.

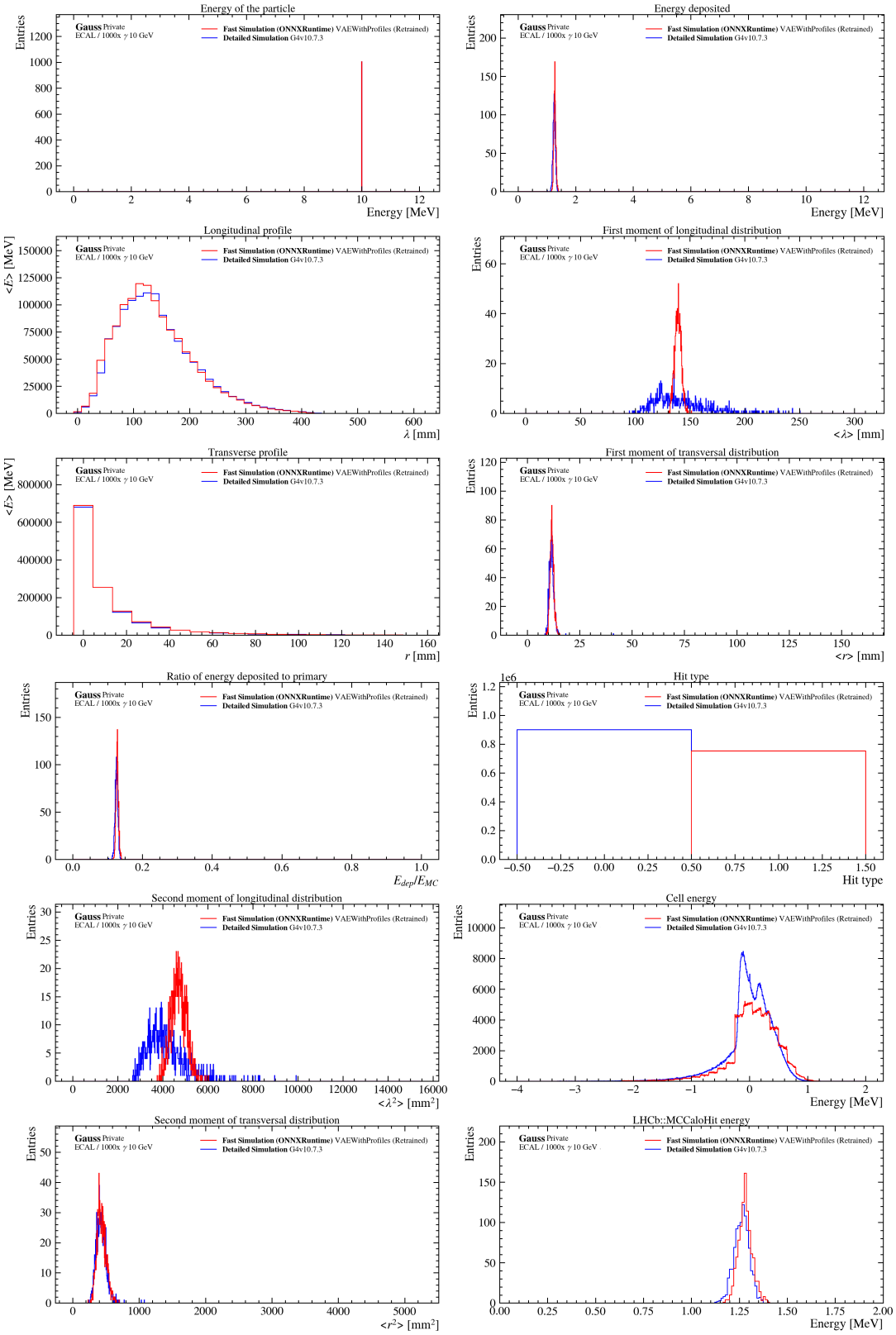


FIGURE D.10: Output plots of the monitoring algorithm of fast simulated 10 GeV photons at $\theta = 12.7$ with a retrained VAE model.

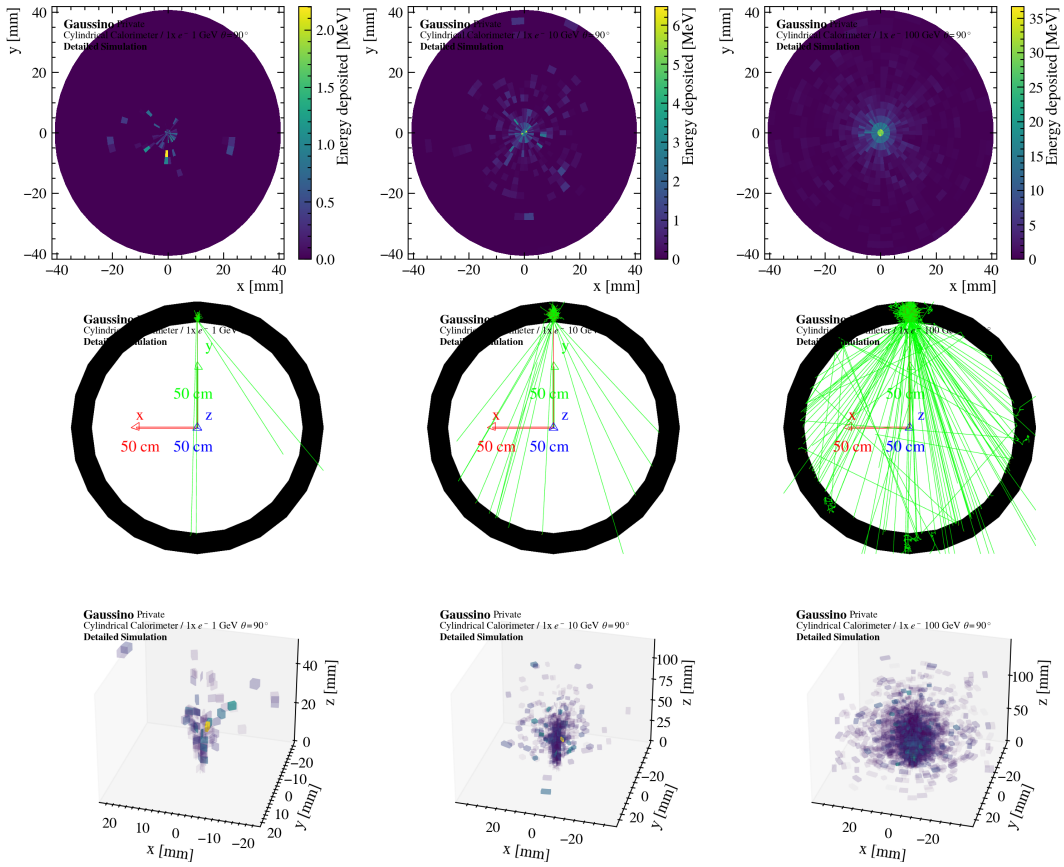


FIGURE D.11: Visualization of the ML-based fast simulation (VAE model) tested on a simple cylindrical calorimeter. The energy of the incident particle increases from left to right. Images in the top row represents a projection of the calorimeter response in the x-y plane of the virtual cylinder. The middle row consists of the GEANT4 visualizations of the showers as sees from the z axis of the main coordinate system. The bottom row represents 3D visualizations of the showers.

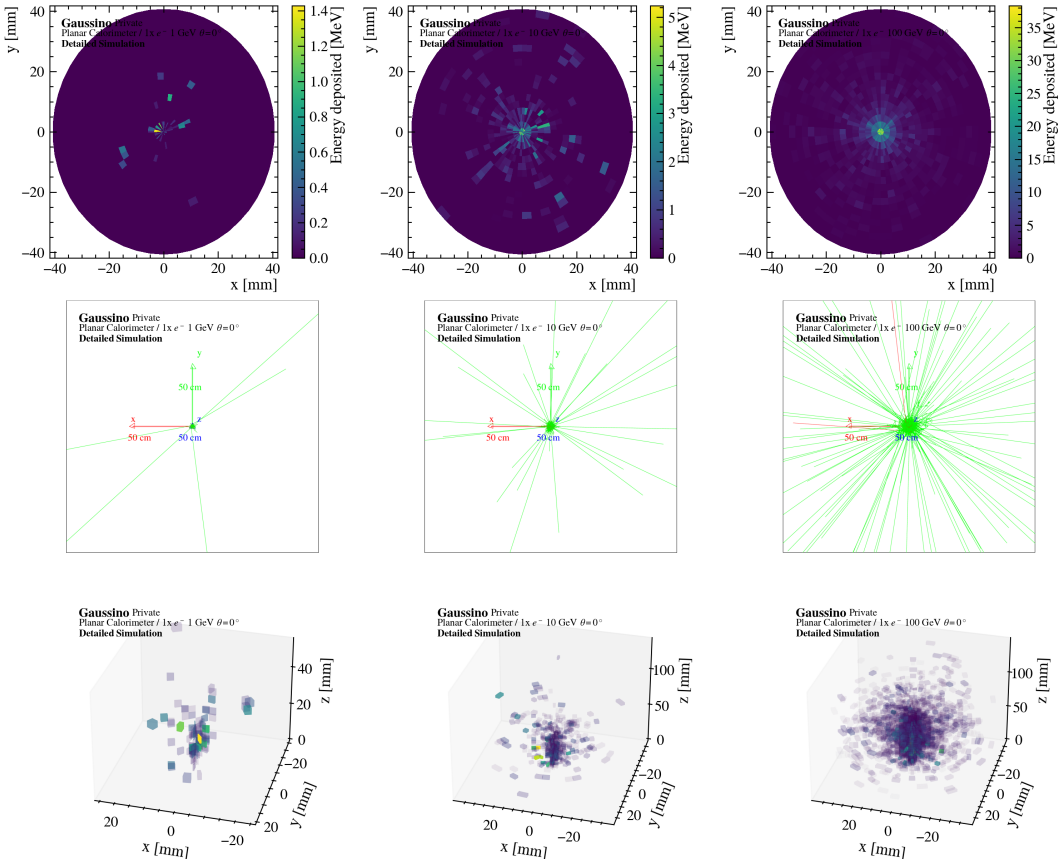
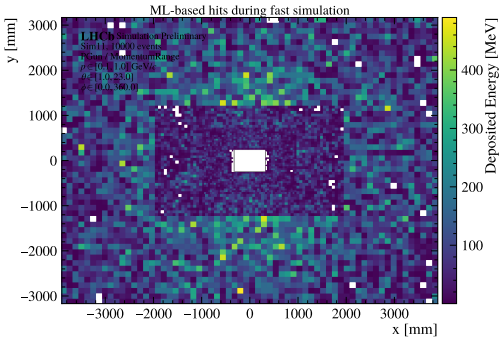
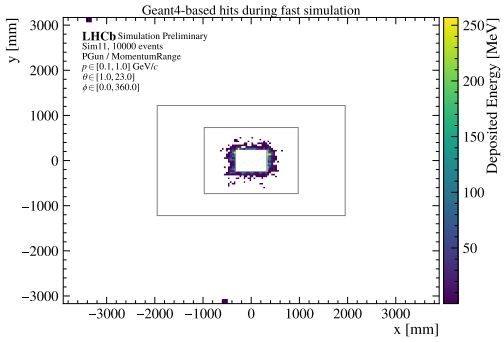


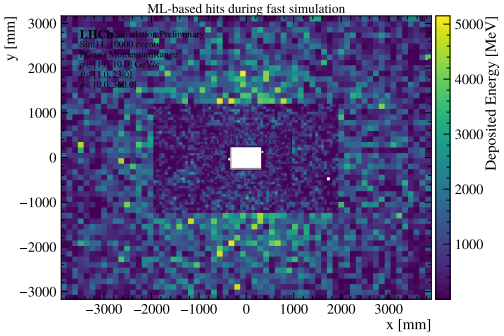
FIGURE D.12: Visualization of the ML-based fast simulation (VAE model) tested on a simple planar calorimeter. The energy of the incident particle increases from left to right. Images in the top row represents a projection of the calorimeter response in the x-y plane of the virtual cylinder. The middle row consists of the GEANT4 visualizations of the showers as sees from the z axis of the main coordinate system. The bottom row represents 3D visualizations of the showers.



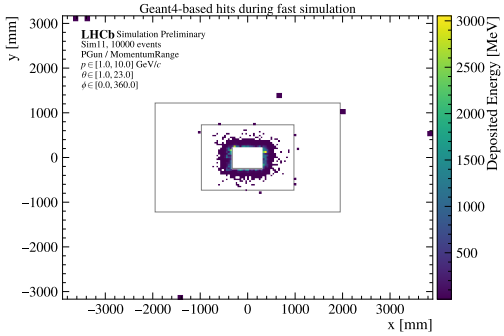
(A) ML-based sim. for $p \in [0.1, 1.0]$ GeV.



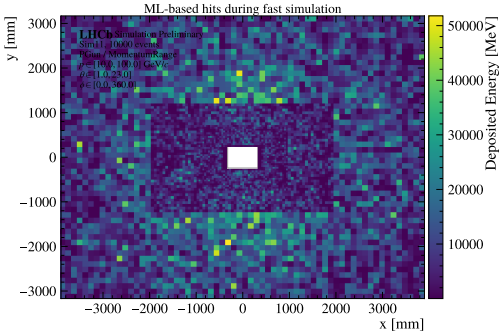
(B) GEANT4-based sim. for $p \in [0.1, 1.0]$ GeV.



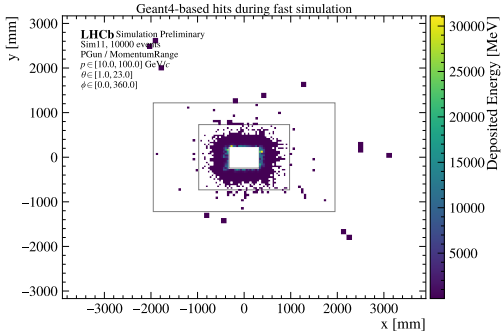
(C) ML-based sim. for $p \in [1.0, 10.0]$ GeV.



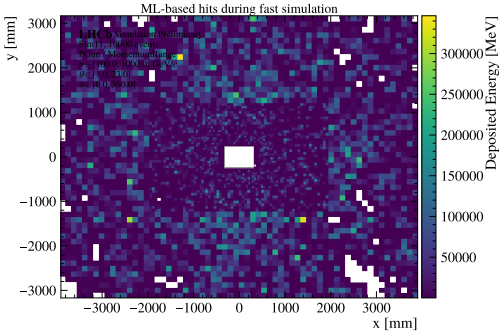
(D) GEANT4-based sim. for $p \in [1.0, 10.0]$ GeV.



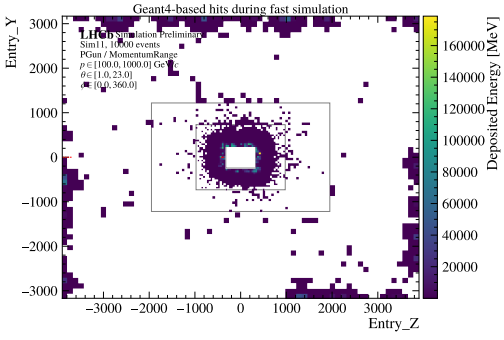
(E) ML-based sim. for $p \in [10.0, 100.0]$ GeV.



(F) GEANT4-based sim. for $p \in [10.0, 100.0]$ GeV.

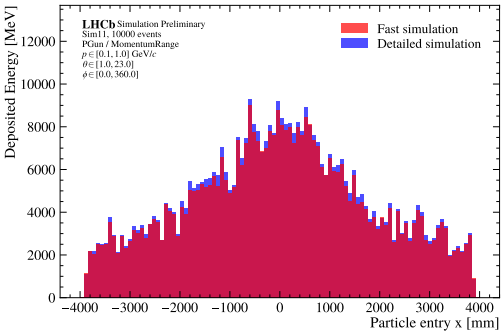


(G) ML-based sim. for $p \in [100.0, 1000.0]$ GeV.

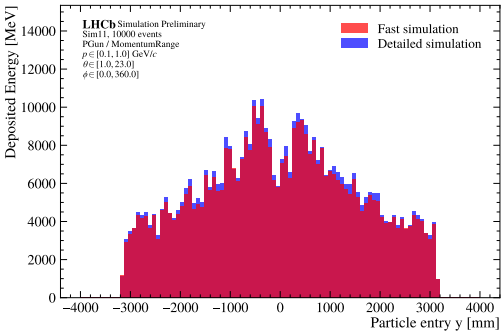


(H) GEANT4-based sim. for $p \in [100.0, 1000.0]$ GeV.

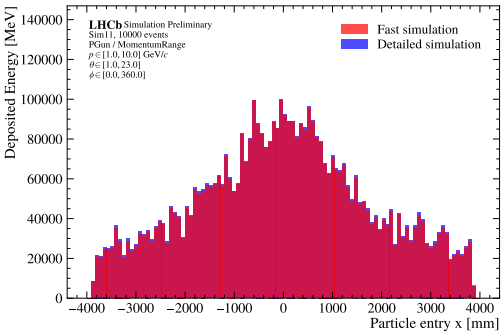
FIGURE D.13: Visualization of the energy deposits in the electromagnetic calorimeter left by the ML-based and GEANT4-based component when running fast simulation. The ML-based fast simulation is turned off around the beam pipe. Moreover, the fast simulation is only applied to photons and electrons with momenta in the range of 0.1-1000 GeV.



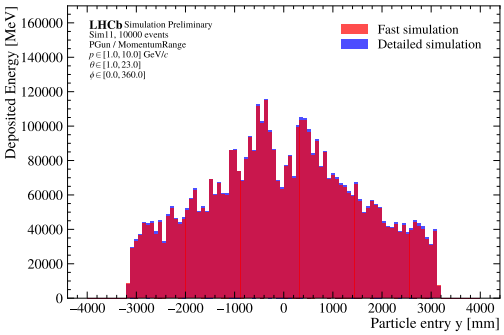
(A) x-scan for $p \in [0.1, 1.0]$ GeV.



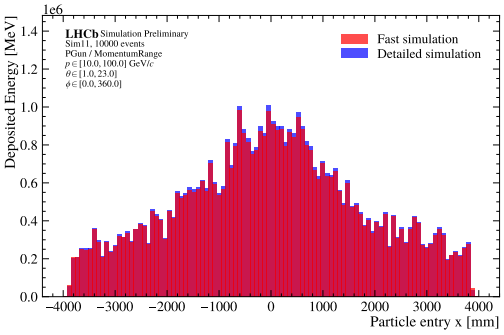
(B) y-scan for $p \in [0.1, 1.0]$ GeV.



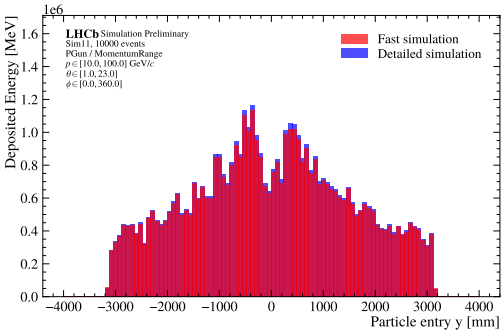
(C) x-scan for $p \in [1.0, 10.0]$ GeV.



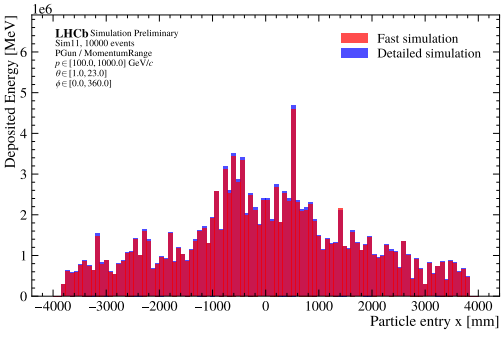
(D) y-scan for $p \in [1.0, 10.0]$ GeV.



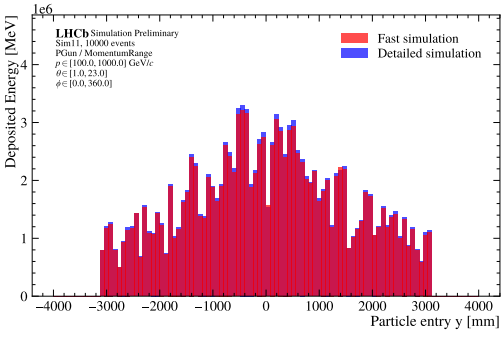
(E) x-scan for $p \in [10.0, 100.0]$ GeV.



(F) y-scan for $p \in [10.0, 100.0]$ GeV.

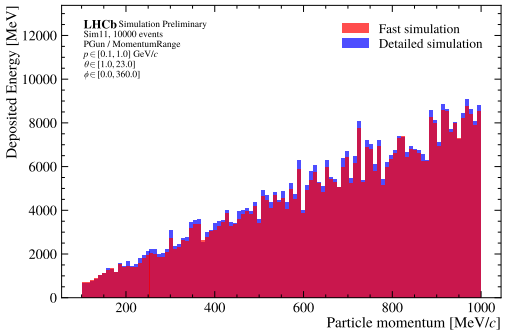


(G) x-scan for $p \in [100.0, 1000.0]$ GeV.

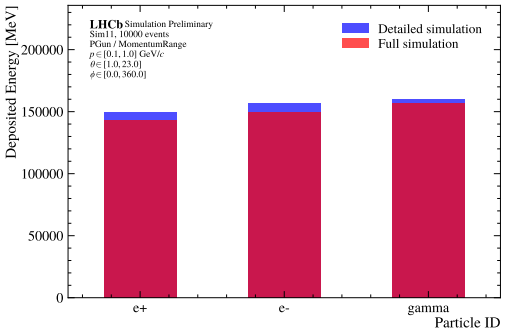


(H) y-scan for $p \in [100.0, 1000.0]$ GeV.

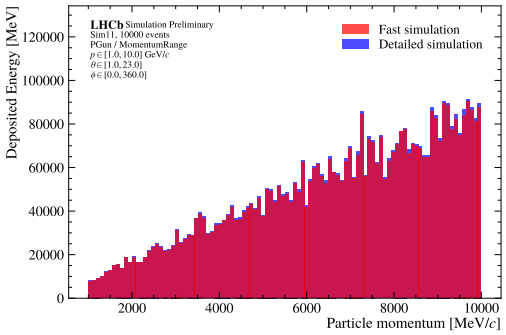
FIGURE D.14: Comparison of the energy deposited in the electromagnetic calorimeter by the ML-based fast simulation and detailed simulation as a function of the entry point of the particle.



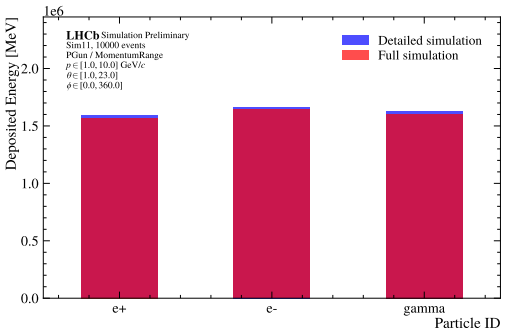
(A) p-scan in $[0.1, 1.0]$ GeV.



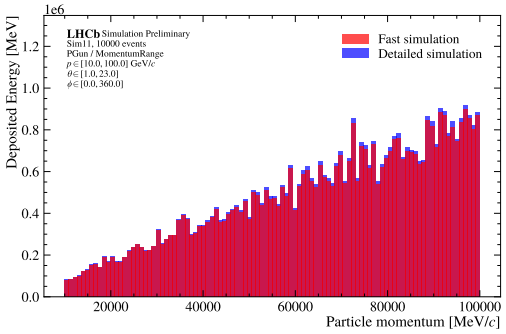
(B) PDG-scan for $p \in [0.1, 1.0]$ GeV.



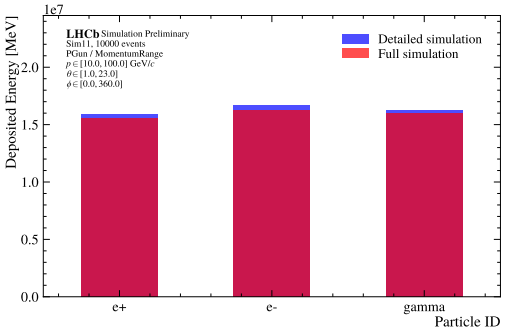
(C) p-scan in $[1.0, 10.0]$ GeV.



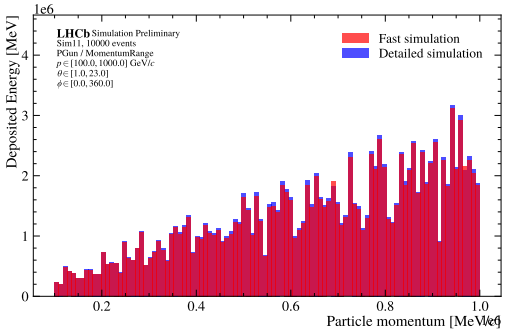
(D) PDG-scan for $p \in [1.0, 10.0]$ GeV.



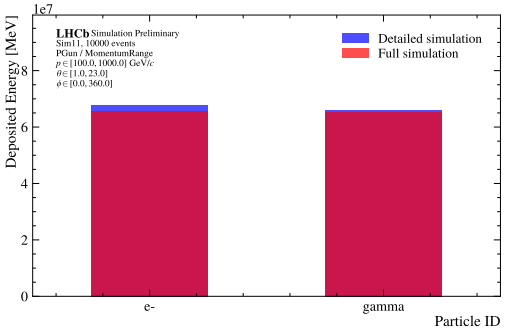
(E) p-scan in $[10.0, 100.0]$ GeV.



(F) PDG-scan for $p \in [10.0, 100.0]$ GeV.

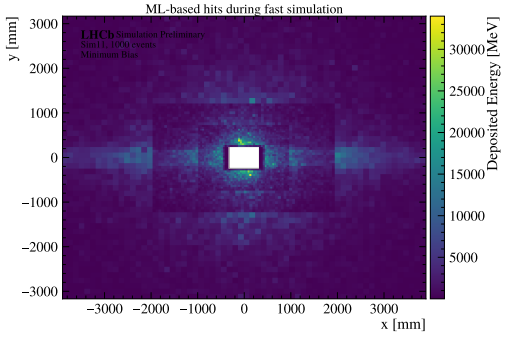


(G) p-scan in $[100.0, 1000.0]$ GeV.

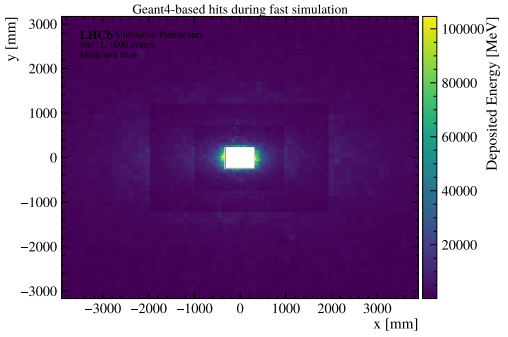


(H) PDG-scan for $p \in [100.0, 1000.0]$ GeV.

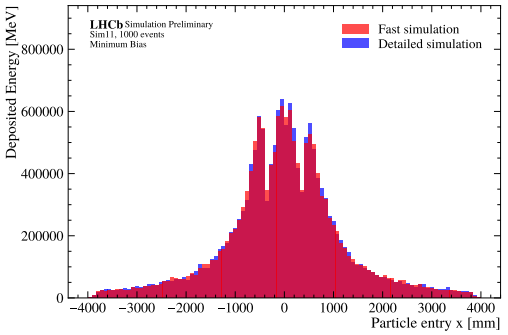
FIGURE D.15: Comparison of the energy deposited in the electromagnetic calorimeter by the ML-based fast simulation and detailed simulation as a function of the momentum of the particle and its PDG code.



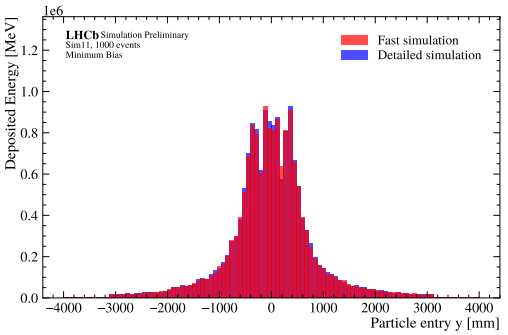
(A) ML-based sim. component when running fast simulation.



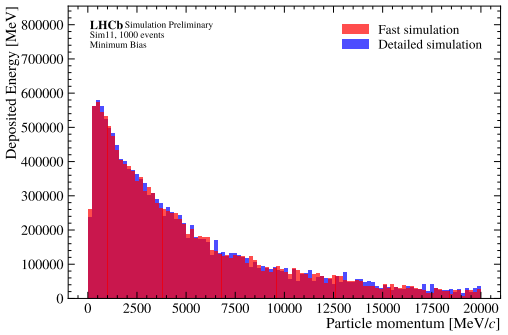
(B) GEANT4-based sim. component when running fast simulation.



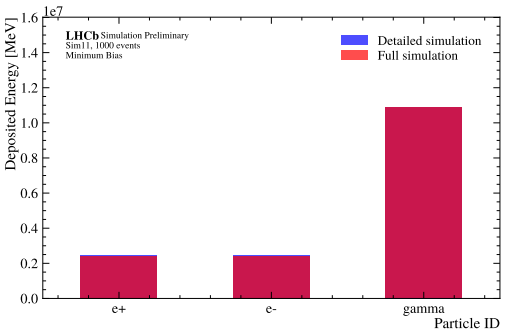
(C) x-scan.



(D) y-scan.



(E) p-scan.



(F) PDG-scan.

FIGURE D.16: Validation of the ML-based fast simulation using the CALOML+VAE model on the LHCb minimum bias events.