GENERATING CALORIMETER SHOWERS AS POINT CLOUDS

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Simon Patrik Schnake, M.Sc.

aus

Minden

Berichter: Prof. Dr. Kerstin Borras
Prof. Dr. Michael Krämer

Tag der mündlichen Prüfung: 31.01.2025

To my parents, whose love, dedication, and guidance have been the foundation of everything I have achieved.

ABSTRACT

Simulating particle interactions within high-energy physics detectors is essential for interpreting experimental data and advancing our understanding of fundamental physics. Calorimeters measure particle energies through cascades known as *showers*, but their complex responses and the computational intensity of traditional simulation tools like *Geant4* pose significant challenges. These are further compounded in *high-granularity calorimeters*, which consist of millions of cells yet register energy deposits in only a sparse subset, rendering full-scale simulations impractical.

This thesis addresses the need for efficient and accurate calorimeter simulations by developing novel generative *machine learning* models that leverage the inherent sparsity and point-like nature of calorimeter data. Initial efforts using *voxel-based generative adversarial networks (GANs)*—which model data in a discrete grid structure—encountered scaling issues due to data sparsity and high dimensionality. By shifting to a *point cloud* representation, the *CaloPointFlow* model was developed, marking the first application of point cloud generative techniques to calorimeter simulation. This approach reduces data complexity by treating showers as collections of points in space rather than densely populated grids.

Although *CaloPointFlow* marked a significant advancement, limitations such as inadequate point-to-point information exchange and difficulties in modeling discrete coordinate positions were observed. To overcome these challenges, *CaloPointFlow II* introduced *DeepSetFlow*, a novel *normalizing flow* architecture that enables direct interactions between points, capturing complex dependencies within the data. Additionally, a new *dequantization strategy* called *CDF-Dequantization* was implemented to better map discrete cell positions to continuous space, along with a mitigation strategy to handle multiple energy deposits per cell.

Despite these improvements, purely point cloud-based models struggled to ensure one hit per calorimeter cell. To resolve this, *CaloHit* was introduced, a hybrid approach that combines voxel-based and point cloud methodologies. *CaloHit* first generates a hitmap to identify active cells using a voxel-based method and then predicts the energies of these hits with a point cloud-based model. This two-stage process effectively addresses the primary limitations of previous models by ensuring that all calorimeter cells are sampled without replacement and that the one-hit-per-cell constraint is maintained.

Evaluations of these models demonstrated their potential to accurately reproduce calorimeter showers while significantly reducing computational resources compared to traditional methods. Preliminary tests of the *CaloHit* approach showed promising results, indicating the feasibility of scaling this method to more complex and higher-dimensional datasets.

In conclusion, this thesis contributes to the advancement of calorimeter *surrogate modeling* by introducing and refining innovative generative models that effectively handle the complexities of high-granularity, sparse data. The proposed methods lay the groundwork for scalable, efficient, and experimentally validated simulation tools. Future work will focus on further improving these models, exploring more sophisticated techniques such as *diffusion models* or

conditional flow matching, and validating their performance in real experimental settings. The developments presented here hold significant potential for enhancing the efficiency and accuracy of particle physics simulations, ultimately aiding in the pursuit of new discoveries in the field.

ZUSAMMENFASSUNG

Die Simulation von Teilchenwechselwirkungen in Hochenergiephysik-Detektoren ist entscheidend, um experimentelle Daten zu interpretieren und das Verständnis der fundamentalen Physik voranzutreiben. Kalorimeter messen die Energie von Teilchen durch Kaskaden, die als *Showers* bekannt sind, jedoch stellen ihre komplexen Reaktionen und die rechnerische Intensität traditioneller Simulationswerkzeuge wie *Geant4* erhebliche Herausforderungen dar. Diese Herausforderungen werden bei *hochgranularen Kalorimetern*, die aus Millionen von Zellen bestehen, aber nur in einem spärlichen Teil Energieeinträge registrieren, noch verstärkt, was Vollsimulationen zunehmend unpraktikabel macht.

In dieser Arbeit wird die Notwendigkeit effizienter und genauer Kalorimetersimulationen durch die Entwicklung neuartiger generativer *maschineller Lernmodelle* adressiert, die die inhärente Sparsität und punktartige Natur von Kalorimeterdaten nutzen. Erste Ansätze mit *voxelbasierten generativen adversarialen Netzwerken (GANs)*, die Daten in einer diskreten Gitterstruktur modellieren, stießen aufgrund von Datenlücken und hoher Dimensionalität auf Skalierungsprobleme. Durch den Wechsel zu einer *Punktwolken*-Darstellung konnte *CaloPointFlow*, das erste Modell, das generative Techniken für Punktwolken auf Kalorimetersimulationen anwendet, entwickelt werden. Dieser Ansatz reduziert die Datenkomplexität, indem Showers als Sammlungen von Punkten im Raum behandelt werden, anstatt als dicht besetzte Gitter.

Obwohl *CaloPointFlow* einen bedeutenden Fortschritt darstellte, zeigten sich Einschränkungen wie unzureichender Informationsaustausch zwischen den Punkten und Herausforderungen bei der Modellierung diskreter Koordinatenpositionen. Um diese Probleme zu beheben, wurde mit *CaloPointFlow II* eine neuartige *Normalizing-Flow* Architektur namens *DeepSetFlow* eingeführt, die direkte Interaktionen zwischen Punkten ermöglicht und komplexe Abhängigkeiten in den Daten erfasst. Zusätzlich wurde eine neue *Dequantisierungsstrategie* namens *CDF-Dequantisierung* implementiert, um die Zuordnung zwischen diskreten Zellpositionen und kontinuierlichem Raum zu verbessern, sowie eine Strategie entwickelt, um das Problem mehrerer Energieeinträge pro Zelle zu lösen.

Trotz dieser Verbesserungen hatten rein punktwolkenbasierte Modelle Schwierigkeiten, sicherzustellen, dass jede Kalorimeterzelle nur einen Treffer erhält. Zur Lösung dieses Problems wird in dieser Arbeit *CaloHit* vorgestellt, ein hybrider Ansatz, der voxelbasierte und punktwolkenbasierte Methoden kombiniert. Zunächst wird mit einer voxelbasierten Methode eine Hitmap zur Identifikation aktiver Zellen generiert und anschließend werden die Energien dieser Treffer mit einem punktwolkenbasierten Modell vorhergesagt. Dieser zweistufige Prozess behebt die Hauptbeschränkungen vorheriger Modelle, indem sichergestellt wird, dass alle Kalorimeterzellen ohne Wiederholung abgetastet werden und die Ein-Treffer-pro-Zelle-Bedingung eingehalten wird.

Die Bewertung dieser Modelle zeigte ihr Potenzial, Kalorimeter-Showers genau zu reproduzieren und dabei den Rechenaufwand im Vergleich zu traditionellen Methoden erheblich zu reduzieren. Vorläufige Tests des *CaloHit*-Ansatzes zeigten vielversprechende Ergebnisse und deuten darauf hin, dass dieser Ansatz auf komplexere und höherdimensionale Daten-

sätze skalierbar ist.

Abschließend trägt diese Arbeit zur Weiterentwicklung der *Surrogatmodellierung* von Kalorimetern bei, indem innovative generative Modelle eingeführt und weiterentwickelt werden, die die Komplexität hochgranularer, spärlicher Daten effektiv bewältigen. Die vorgeschlagenen Methoden schaffen eine Grundlage für skalierbare, effiziente und experimentell validierte Simulationswerkzeuge. Zukünftige Arbeiten werden sich auf die weitere Verbesserung dieser Modelle, die Erkundung fortschrittlicherer Techniken wie *Diffusionsmodelle* oder bedingte Flow Matching-Verfahren und die Validierung ihrer Leistung in realen experimentellen Umgebungen konzentrieren. Die hier vorgestellten Entwicklungen bieten erhebliches Potenzial zur Steigerung der Effizienz und Genauigkeit von Simulationen in der Teilchenphysik und tragen letztlich zur Entdeckung neuer Erkenntnisse in diesem Bereich bei.

ACKNOWLEDGEMENTS

This thesis marks the culmination of a significant period of research and personal growth, a journey made possible by the support, guidance, and encouragement of many individuals and institutions. I wish to express my sincere gratitude to all those who have contributed to this work.

First and foremost, I extend my deepest appreciation to my primary supervisor, Prof. Dr. Kerstin Borras. Thank you for granting me the invaluable freedom to explore the research avenues that truly captivated my interest, rather than strictly adhering to the initial scope. Your trust and guidance were instrumental in shaping this project.

I am also grateful to my co-supervisor, Prof. Dr. Michael Krämer, for his kind support and for allowing me the independence to pursue my research path while being available when needed. My thanks also go to Prof. Dr. Stefan Schael and Prof. Dr. Oliver Pooth for serving on my thesis committee and for their valuable time and feedback.

A special and significant thank you goes to Dr. Dirk Krücker. Your mentorship, insightful guidance, and stimulating discussions were incredibly helpful and sparked many of the ideas explored in this thesis. Your support during my time at DESY was invaluable.

I would also like to thank my fellow PhD candidate, Benno Käch. The journey was certainly more enjoyable with your camaraderie, humour, and the many fruitful discussions we shared. I also acknowledge Moritz Scham for his contributions as part of our research group.

This research would not have been possible without the institutional and financial support provided by DESY. I am grateful for the opportunity to conduct my research within such a stimulating environment. Furthermore, I acknowledge the crucial role of the DESY Maxwell Cluster, which provided the computational resources essential for the analyses presented herein.

On a personal note, I owe immense gratitude to my parents for their unwavering support throughout my life, which laid the foundation for all my endeavors.

To Josefine Brauer, the love of my life, thank you. Your constant emotional support, patience, and understanding carried me through the many hills and valleys of this PhD. Thank you for enduring my moods, for your kindness, and for sharing this journey with me.

My sincere thanks extend to all my friends for their encouragement and for providing welcome distractions and perspective.

To everyone mentioned and unmentioned who played a part in this journey, thank you.

CONTENTS

1 INTRODUCTION

The fundamental objective of physics is to deepen our understanding of the underlying structure of the universe. In particle physics, research focuses on investigating elementary particles to uncover the principles governing their interactions. The Standard Model has been remarkably successful in describing and predicting fundamental forces, offering an unprecedented level of precision. One of its most significant achievements is the discovery of the Higgs boson, a milestone that marks a pivotal advancement in the development of the Standard Model.

However, despite its experimental precision and theoretical coherence, the Standard Model remains incomplete, as it fails to account for several phenomena, including dark matter and gravitation. Contemporary collider experiments, such as those conducted at the LHC, still hold the potential for groundbreaking discoveries. To realize these discoveries, the experiments must analyze recorded collision data and compare the results with theoretical predictions, a highly complex task.

Particle physicists rely extensively on Monte Carlo simulations to address this challenge. A full LHC experiment simulation begins with event generation, which involves calculating the initial collision and hard processes to predict the primary particles produced. The next step in the simulation chain is hadronization, which models the formation of hadrons. This process results in a cloud of particles that traverse the detector, requiring the entire detector response to be accurately modeled.

This entire simulation chain requires a substantial amount of computational time and resources, with the simulation of calorimeters—used to measure the energy of the produced particles—consuming a significant fraction of these resources.

A calorimeter is a large detector volume designed to measure the energy of particles passing through it. Calorimeters can detect cascades of secondary particles produced by incoming particles, commonly referred to as showers. Simulating these showers requires sophisticated, multi-step computational techniques, with Geant4 (Allison et al. [Geant4] 2003) being one of the most commonly employed simulation toolkits for this purpose.

Although Geant4 provides unparalleled precision, it is also highly computationally demanding, often requiring seconds per event on a conventional CPU. Given these limitations, the feasibility of conducting comprehensive, intricate detector simulations for every event is becoming increasingly questionable (Albrecht et al. 2019; Boehnlein et al. 2022; Zurbano Fernandez et al. 2020).

Generative machine learning models are designed to learn and sample from the underlying distributions of datasets. Over the past decade, numerous novel generative models have emerged. The introduction of Generative Adversarial Networks (Goodfellow et al. 2014) and Variational Autoencoders (Kingma & Welling 2013a) has enabled the generation of complex, multi-dimensional data. Additionally, Normalizing Flows (Papamakarios et al. 2021) have gained recognition for their ability to model complex distributions through a

sequence of invertible transformations. Most recently, Diffusion Models (Sohl-Dickstein et al. 2015) have also gained significant attention.

These generative models have been applied across a wide range of particle physics applications, particularly in the context of fast simulations for calorimeters. Initially, Generative Adversarial Networks (Paganini et al. 2018a) were the primary focus of research, with Variational Autoencoders (Buhmann et al. 2020) and Normalizing Flows (Krause & Shih 2021) also explored. More recently, Diffusion Models (Mikuni & Nachman 2022, 2024) have emerged as a new area of interest in this field. The majority of these models are based on a fixed data geometry, where a calorimeter is represented by a set of voxels—the three-dimensional equivalent of pixels—with each voxel corresponding to a calorimeter sensor. High-granularity calorimeters, such as the proposed upgrade for the CMS hadron calorimeter (CMS Collaboration [CMS] 2017), consist of millions of cells, necessitating the development of models capable of handling such large datasets. However, in most particle showers, only a subset of these cells typically receive energy deposition.

My initial approach to employing generative models for calorimeter showers involved constructing a Generative Adversarial Network-based model for high-granularity calorimeters. The primary challenge was the sparsity of the showers, which proved difficult to learn and reproduce effectively, leading to suboptimal early solutions. Most voxel-based approaches have since adopted a method developed by Krause et al., where small amounts of energy are distributed across all empty cells to mitigate sparsity, allowing non-sparse showers to be learned. The sparsity is then reintroduced by applying an energy threshold.ld.

An alternative methodology was developed by leveraging the sparsity of the dataset to advantage. Rather than modeling all cells, the focus was shifted to modeling the distribution of hits, transforming the data structure into a point cloud. This approach offers several advantages. First, the need to model all empty cells is eliminated, making it feasible to simulate very high-granularity calorimeters. Second, it allows for easier adaptation to complex detector geometries. However, the generative modeling of point clouds presents a greater challenge than traditional voxel-based methods. The CaloPointFlow model (Schnake et al. 2022) was one of the first point cloud-based generative models developed for calorimeter simulations. The PointFlow architecture was adapted specifically for this purpose, resulting in a novel model.

Subsequent publications and research within the field have since been published (Acosta et al. 2023; Buhmann, Diefenbacher, Eren, et al. 2023; Buhmann, Gaede, Kasieczka, et al. 2023; Käch et al. 2023). Several disadvantages of the point cloud-based approach were identified, leading to the development of a second, updated version of the model, CaloPointFlow II (Schnake et al. 2024), to address these issues. This version incorporates a new point cloud normalizing flow architecture and a novel dequantization strategy.

In addition, a novel hybrid approach, termed *CaloHit*, is introduced to address the primary limitation of point cloud-based models. A proof of concept is presented to demonstrate the feasibility of this approach.

This thesis is organized as follows: It begins with an overview of the Standard Model (Chapter 2) as the foundation of particle physics, followed by a review of high-energy physics, the CMS experiment, and the Large Hadron Collider (LHC) (Chapter 3). Subsequently, an overview of calorimetry and its simulation using Geant4 is provided (Chapter 4). The fun-

damental concepts of machine learning are then introduced (Chapter 5), followed by the construction of the generative models utilized in this investigation (Chapter 6). The main chapter (Chapter 8) delves into the CaloPointFlow architecture, detailing the model's training process and sampling techniques. The feasibility of the CaloHit approach is also demonstrated (Chapter 9). The model's performance is evaluated, comparing the updated version to the original. The thesis concludes with a summary and outlook (Chapter 10).

THE STANDARD MODEL OF PARTICLE PHYSICS

2

The *Standard Model* (SM) of particle physics is a comprehensive *Quantum Field Theory* (QFT) that describes all known elementary particles and their interactions. It encompasses the electromagnetic, weak, and strong forces. The SM extends the concept of gauge invariance, initially postulated for the electromagnetic field in *Quantum Electrodynamics* (QED), to include other fields. The validity of the Standard Model has been reinforced by numerous predictions that were later confirmed, most notably the discovery of the top quark in 1995, the tau neutrino in 2000, and the Higgs boson in 2012. Moreover, the Standard Model accurately predicted the properties of weak neutral currents as well as the W and Z bosons.

However, the Standard Model is not without limitations. While the SM is theoretically self-consistent, it remains an incomplete theory of all fundamental interactions. It does not fully explain the observed asymmetry between baryons and antibaryons, nor does it incorporate a comprehensive theory of gravity or account for the observed accelerating expansion of the universe. Additionally, the SM lacks a viable dark matter candidate and fails to adequately address neutrino oscillations and their non-zero masses.

2.1 Historical Context and Development

The landscape of QFT underwent significant transformations between the 1950s and the early 21st century. This period marked the development and consolidation of what is now known as the Standard Model of particle physics. In 1954, Chen-Ning Yang and Robert Mills introduced gauge theory for non-Abelian groups, extending the scope of gauge theory beyond Abelian groups, such as those found in Quantum Electrodynamics, to encompass the complexities of strong interactions (C. N. Yang & Mills 1954).

In 1961, Sheldon Glashow made a significant contribution to the evolution of the Standard Model by combining electromagnetic and weak interactions (Glashow 1961). This theoretical advancement paved the way for further unification within the model. In 1967, Weinberg 1967 and Salam 1968 incorporated the Higgs mechanism (Englert & Brout 1964; Guralnik et al. 1964; Higgs 1964) into the electroweak interaction, building on the foundation laid by Glashow. This development gave the electroweak interaction its current form. Empirical support for the electroweak theory came with the discovery of neutral weak currents via Z boson exchange at CERN by Hasert et al. 1973.

The discovery of the Higgs boson at the *Large Hadron Collider* (LHC) in 2012 marked a major milestone in particle physics, confirming key predictions of QFT.

The term "Standard Model" was first introduced by Pais and Treiman 1975 in the context of electroweak theory, which at that time included four quarks (Cao 1998). This term encapsulates the framework that describes the fundamental forces and particles of the universe, representing decades of theoretical development and experimental validation.

2.2 Particles of the Standard Model

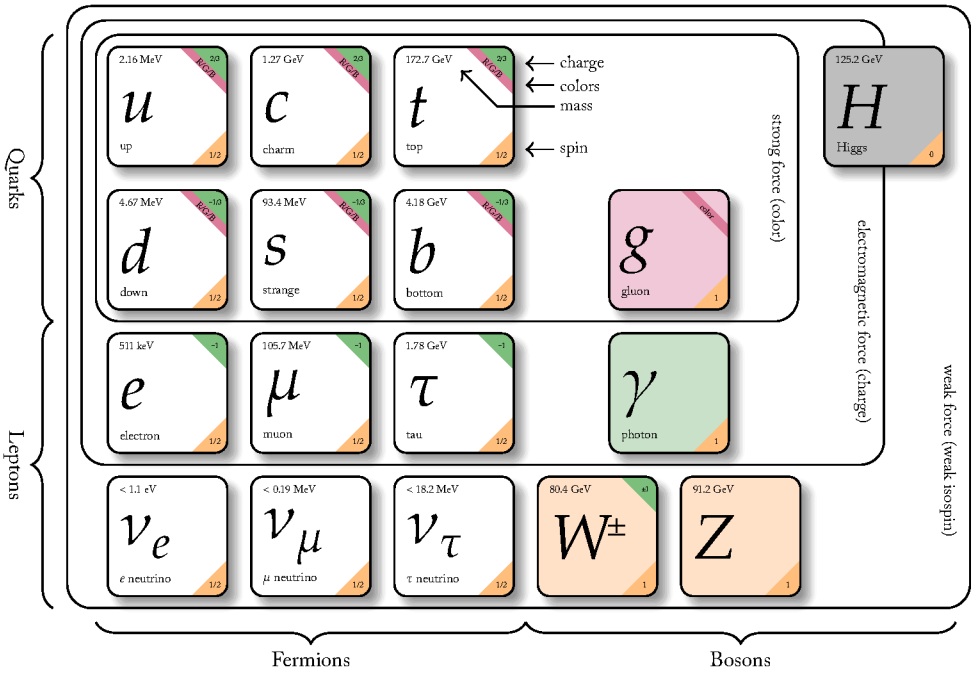

Figure 2.1: Diagram of the Standard Model of Particle Physics, adapted from Galbraith (2013) and using quantities from Hagiwara et al. (2002). Uncertainties have been omitted for readability.

The Standard Model of particle physics classifies fundamental particles into two main groups: bosons and fermions. Bosons, characterized by integer spin, obey Bose-Einstein statistics, while fermions, with half-integer spin, follow Fermi-Dirac statistics. The model includes 12 fermions, divided into two categories: six quarks and six leptons. These fermions are further organized into three generations, ordered by their mass and lifetime.

The lepton family consists of three doublets: the electron (e) and the electron neutrino (ν_e), the muon (μ) and the muon neutrino (ν_μ), and the tau (τ) and the tau neutrino (ν_τ). Leptons are massive particles, while neutrinos were initially assumed to be massless in the SM. However, the discovery of neutrino oscillations suggests that neutrinos possess small but non-zero masses. Leptons interact through the electromagnetic and weak forces, carrying both electric and weak hypercharges. Neutrinos, which have no electric charge, interact only through the weak force. The muon and tau are unstable and decay, while neutrinos of all generations are stable but rarely interact with baryonic matter.

Quarks, another category of fermions, are also divided into three generations: up (u) and down (d) quarks, charm (c) and strange (s) quarks, and top (t) and bottom (b) quarks. Up-type quarks (u, c, t) have an electric charge of +2/3, while down-type quarks (d, s, b) have a charge of −1/3. Quarks are unique in that they carry a color charge and interact via the strong force, leading to the phenomenon of color confinement. This interaction enables quarks to form composite particles, such as mesons (composed of a quark and an anti-quark) and baryons (composed of three quarks). The two lightest baryons, protons and neutrons, are

made up of quarks and interact both electromagnetically and through the weak interaction.

In the SM, force-carrying particles are the bosons. The model includes four gauge bosons with spin 1, which act as force carriers. The electromagnetic interaction is mediated by massless photons (γ), while the weak interaction involves the massive bosons $W^{\pm}$ and Z^0. The strong interaction is governed by massless gluons (g), which exist in eight varieties. Unlike fermions, gauge bosons do not obey the Pauli exclusion principle.

A central component of the SM is the Higgs boson (H), a scalar spin-0 particle. Proposed by Peter Higgs in 1964, the Higgs boson is essential for the mass acquisition of gauge bosons and charged fermions through the Higgs mechanism, which is associated with the spontaneous breaking of electroweak gauge symmetry. This scalar particle has no intrinsic spin and interacts with itself due to its mass.

2.3 Gauge Invariance

The Standard Model is characterized by its particle content and inherent symmetries. According to Noether's theorem (Noether 1918), any continuous symmetry in a Lagrangian corresponds to the existence of a conserved current. This theorem asserts that the Lagrangian remains unchanged under the equations of motion. The SM is a non-Abelian field theory, meaning it is a gauge-invariant or gauge-symmetric field theory, where gauge transformations—transforming the field into another form—do not alter any measurable quantity. Consequently, symmetry transformations within the SM are non-commutative.

Local gauge invariance is central to the SM, as the model is defined by the symmetry group

$$\mathrm{SU}(3)_C \otimes \mathrm{SU}(2)_L \otimes \mathrm{U}(1)_Y,$$

where the subscripts denote color, left-handedness, and hypercharge, respectively.

2.4 Electroweak Theory

2.4.1 *Quantum Electrodynamics*

Quantum Electrodynamics is the first QFT applied to physics problems, explaining the interaction of charged particles via the emission and absorption of virtual photons. QED is notable for being renormalizable, that is, infinity terms in mathematical computations are absorbed by the virtual photons. Its symmetry group is $U(1)$, characterizing it as an abelian gauge theory and the QFT counterpart of classical electrodynamics.

The Lagrangian of the relativistic spin-1/2 Dirac fermion is given by

$$\mathscr{L} = \bar{\psi}(x)(i\gamma^{\mu}\partial_{\mu} - m)\psi(x), \tag{2.1}$$

where $\psi(x)$ is the Dirac spinor, γ^μ denotes the Dirac matrix, and $\bar{\psi}(x)$ is defined as $\psi^\dagger(x)\gamma^0$. This Lagrangian is not invariant under local $U(1)$ gauge transformations, $\psi'(x) = e^{iq\theta(x)}\psi(x)$. To maintain gauge invariance, a new spin-1 field, $A_\mu(x)$, the photon field, is introduced. It transforms

$$A_\mu(x) \rightarrow A_\mu(x) + \frac{1}{e}\partial_\mu\theta(x). \tag{2.2}$$

The Lagrangian for QED (Feynman 1950)

$$\begin{aligned}\mathscr{L} =& \mathscr{L}_{\text{Dirac}} + \mathscr{L}_{\text{EM}} + \mathscr{L}_{\text{Int}} \\ =& \bar{\psi}(x)(i\gamma^\mu\partial_\mu - m)\psi(x) \\ & - \frac{1}{4}F_{\mu\nu}(x)F^{\mu\nu}(x) \\ & + eqA_\mu(x)\bar{\psi}(x)\gamma^\mu\psi(x),\end{aligned} \tag{2.3}$$

consists of three terms: a relativistic term for fermionic spin-1/2, denoted as $\mathscr{L}_{\text{Dirac}}$, a gauge invariant kinetic term, denoted as $\mathscr{L}_{\text{EM}}$, and an electromagnetic interaction term, denoted as $\mathscr{L}_{\text{Int}}$.

Here m is the lepton mass, ψ is the lepton field, $F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu$, represents the electromagnetic field tensor, and q is the charge. Since all leptons carry the same absolute electromagnetic charge, only one field tensor, ψ, is considered. The electromagnetic coupling strength is defined by the fine structure constant ($\alpha = e^2/4\pi$), which is the strength of interaction between charged particles and photons. The mass term of the photon field, $1/2m^2A_\mu A^\mu$, would not be gauge invariant, therefore the photon has to be massless.

2.4.2 *Electroweak Unification*

The 1960s brought a change in the understanding of weak interactions. The previously accepted V-A theory was replaced by the theory of electroweak interaction. This new theory unifies two of the four fundamental forces of nature: the electromagnetic and weak interactions. The unification occurs above an energy of 246 GeV, a value derived from the effective Fermi theory (Langacker 1986). The theory was developed by Glashow 1961, Weinberg 1967, and Salam 1968.

The local gauge group theory is $\text{SU}(2)_L \otimes \text{U}(1)_Y$. Similar to QED, the weak interaction can be described by requiring local invariance under transformations in SU(2) Here, the generator of the weak interaction T is an element of the associated algebra of the gauge group $\text{SU}(2)_L$. It can be expressed in the basis of the Pauli matrices as $\text{T} = \frac{1}{2}\sigma$. The weak coupling constant is g_W. This interaction is known as weak isospin, which couples only with left-handed fields.

In order to achieve flavor-changing properties, the left-handed fermions and right-handed antifermions are arranged in an isospin doublet The weak hypercharge serves as the electromagnetic generator of $\text{U}(1)_Y$. The mediators in this scenario are three W gauge bosons W_1, W_2, W_3 of the weak isospin and B of the weak hypercharge, all of which are initially massless.

The electroweak symmetry would be broken by the masses of the observed gauge bosons $W^\pm$, Z, and γ. The Higgs mechanism allows such terms through spontaneous symmetry

breaking. Notably, the electromagnetic charge remains unaffected by the Higgs mechanism, while the photon and the bosons are mixed by the generators,

In contrast, $W^{\pm}$ is a combination of W_1 and W_2

The Lagrangian of the electroweak theory is therefore given by

$$\mathscr{L} = \mathscr{L}_g + \mathscr{L}_f + \mathscr{L}_y + \mathscr{L}_h, \tag{2.4}$$

where $\mathscr{L}_g$ is the interaction between vector boson fields. The kinetic term $\mathscr{L}_f$ accounts for left and right-handed fermions, $\mathscr{L}_y$ describes Yukawa couplings connecting matter and the Higgs field, and $\mathscr{L}_h$ describes the Higgs field interaction with the gauge boson field.

2.4.3 *Electroweak Unification*

The 1960s marked a shift in the understanding of weak interactions. The previously accepted V-A theory was replaced by the theory of electroweak interaction, which unifies two of the four fundamental forces of nature: the electromagnetic and weak interactions. This unification occurs above an energy scale of 246 GeV, a value derived from effective Fermi theory (Langacker 1986). The theory was developed by Glashow 1961, Weinberg 1967, and Salam 1968.

The local gauge group governing the electroweak interaction is $\mathrm{SU}(2)_L \otimes \mathrm{U}(1)_Y$. Similar to QED, the weak interaction is described by requiring local invariance under transformations in SU(2):

$$\phi \rightarrow \exp\left[ig_W \alpha(x)\mathrm{T}\right]\phi. \tag{2.5}$$

Here, the generator of the weak interaction, T, is an element of the associated algebra of the gauge group $\mathrm{SU}(2)_L$ and can be expressed in the basis of the Pauli matrices as $\mathrm{T} = \frac{1}{2}\sigma$. The weak coupling constant is denoted as g_W. This interaction, known as weak isospin, couples only with left-handed fields.

In order to achieve flavor-changing properties, left-handed fermions and right-handed antifermions are arranged in an isospin doublet:

$$\phi_L = \begin{pmatrix} \nu_e(x) \\ e^-(x) \end{pmatrix}. \tag{2.6}$$

The weak hypercharge serves as the electromagnetic generator of $\mathrm{U}(1)_Y$. The mediators in this interaction are three W gauge bosons W_1, W_2, W_3 of the weak isospin, and B of the weak hypercharge, all of which are initially massless.

Electroweak symmetry is broken by the masses of the observed gauge bosons $W^{\pm}, Z$, and γ. The Higgs mechanism facilitates this symmetry breaking through spontaneous symmetry breaking. Notably, the electromagnetic charge remains unaffected by the Higgs mechanism, while the photon and the bosons are mixed by the generators as follows:

$$\begin{pmatrix} \gamma \\ Z^0 \end{pmatrix} = \begin{pmatrix} cos\theta_W & sin\theta_W \\ -sin\theta_W & cos\theta_W \end{pmatrix} \begin{pmatrix} B \\ W_3 \end{pmatrix}. \tag{2.7}$$

In contrast, the $W^{\pm}$ bosons are combinations of W_1 and W_2:

$$W^{\pm} = \frac{1}{\sqrt{2}} (W_1 \mp iW_2) . \tag{2.8}$$

The Lagrangian of the electroweak theory is given by:

$$\mathscr{L} = \mathscr{L}_g + \mathscr{L}_f + \mathscr{L}_y + \mathscr{L}_h, \tag{2.9}$$

where $\mathscr{L}_g$ describes the interaction between vector boson fields, $\mathscr{L}_f$ accounts for left- and right-handed fermions, $\mathscr{L}_y$ represents the Yukawa couplings connecting matter and the Higgs field, and $\mathscr{L}_h$ describes the Higgs field's interaction with the gauge boson fields.

2.4.4 *The Higgs Mechanism*

The Higgs mechanism (Englert & Brout 1964; Guralnik et al. 1964; Higgs 1964) explains how particles acquire mass, relying on the concept of symmetry breaking. Its Lagrangian is given by:

$$\mathscr{L}_{\text{Higgs}} = (D_\mu \phi)^\dagger (D^\mu \phi) - V(\phi), \tag{2.10}$$

where ϕ is a complex doublet under $\text{SU}(2)_L$. This doublet can be expressed as:

$$\phi = \frac{1}{\sqrt{2}} \begin{pmatrix} \phi^+ \\ \phi^0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} \phi_1 + i\phi_2 \\ \phi_3 + i\phi_4 \end{pmatrix}. \tag{2.11}$$

The potential $V(\phi)$ in the Lagrangian is defined as:

$$V(\phi) = \mu^2 \phi^\dagger \phi + \lambda \left(\phi^\dagger \phi \right)^2 , \tag{2.12}$$

where μ is the mass of the complex scalar field, and λ describes the self-interactions of the two complex doublets.

The shape of the Higgs potential $V(\phi)$ has degenerate minima, as shown in Figure 2.2. Electroweak symmetry breaking arises from selecting a vacuum state away from the unstable point at $\phi^\dagger \phi = 0$. The field configuration minimizes V when $\phi^+ = 0$ and $\phi^0 = v$, leading to the vacuum expectation value of the field:

$$\langle \phi \rangle_0 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v \end{pmatrix}. \tag{2.13}$$

Small perturbations around this minimum lead to:

$$\langle \phi \rangle_0 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + h(x) \end{pmatrix}, \tag{2.14}$$

where $h(x)$ denotes the Higgs field. Using the measured values of m_W and g_W, the vacuum expectation value of the Higgs field is predicted to be $v = 246\,\text{GeV}$.

A direct consequence of this mechanism is the observable Higgs boson. Its most precise mass measurement to date is $m_H = 125.38 \pm 0.14\,\text{GeV}/c^2$, as measured by the CMS collaboration (CMS 2020), through a combination of the four-lepton decay channel ($H \to ZZ \to 4l$) and the diphoton decay channel ($H \to \gamma\gamma$).

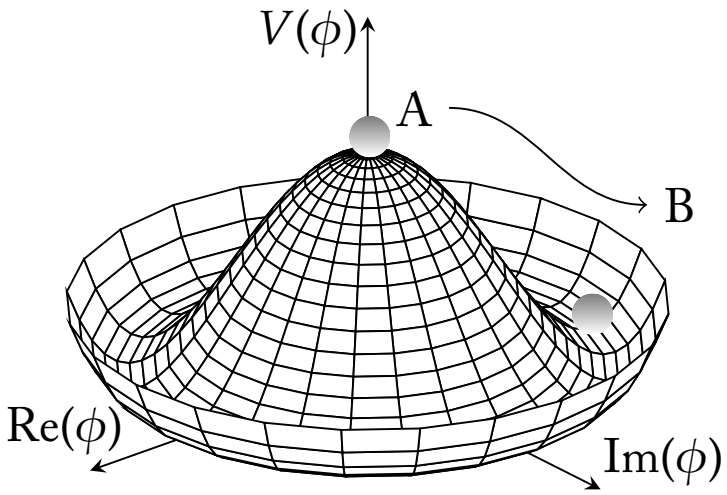

Figure 2.2: Diagram of the Higgs potential $V(\phi)$. Adaptation of Riebesell (2022).

2.5 Quantum Chromodynamics

Quantum Chromodynamics (QCD) describes the strong interaction, initially proposed by Gell-Mann (1961, 1964) and Zweig 1964, introducing the concept of quarks. Originally, three species of quarks were identified, later termed colors. QCD explains the interactions between particles that carry color charge.

Gluons are massless, neutral particles that mediate the force between color-charged particles in QCD. Notably, gluons can also interact with other gluons. Gluons and quarks are never observed in an unbound state, a phenomenon known as color confinement.

The Lagrangian of QCD remains invariant under arbitrary global transformations of the $SU(3)_C$ group in color space. This global symmetry is extended to a local one through the introduction of QCD covariant derivatives:

$$D_\mu = \partial_\mu + ig_s T^a G^a_\mu, \tag{2.15}$$

where T^a represents the eight generators of the $SU(3)_C$ symmetry group, and G^a_μ denotes the eight gauge boson fields corresponding to the gluons, indexed by $a = 1, \dots, 8$. The QCD Lagrangian is therefore expressed as:

$$\mathscr{L}_{\text{QCD}} = -\frac{1}{4} G^{\mu\nu}_a G^a_{\mu\nu} + \bar{q}\left[i\gamma^\mu D_\mu - m_q\right]q, \tag{2.16}$$

which describes the interaction between the quark field q and the anti-quark field $\bar{q}$. The first term of the Lagrangian explains the kinematics of gluon fields and their self-interaction, while the second term includes the free Dirac fermion term and the interaction between quarks and gluons.

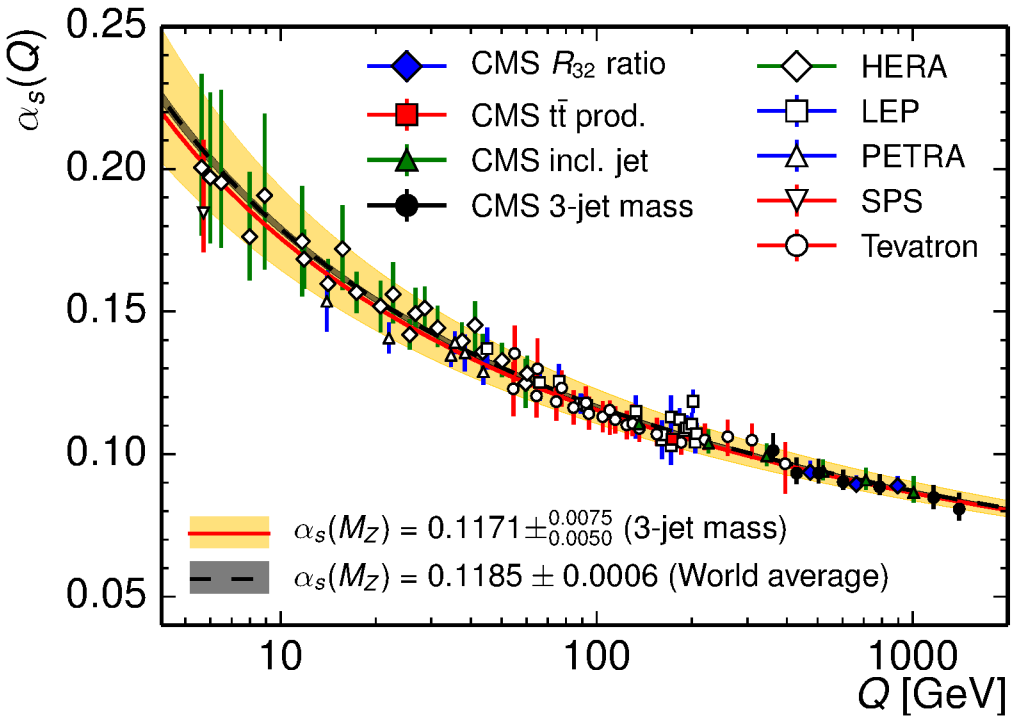

Figure 2.3: The evolution of $\alpha_s(Q)$. The figure was created by CMS Collaboration (2015). For further details, please refer to this reference.

The coupling constant α_s of QCD is defined as:

$$\alpha_s = \frac{g_s^2}{4\pi} \sim \left[\log\left(\frac{Q^2}{\Lambda^2}\right)\right]^{-1}, \tag{2.17}$$

where Λ is the QCD scale, approximately 200 MeV. The coupling constant is not fixed but varies with the energy scale Q, a phenomenon known as the running of the coupling constant. This variation has been precisely measured in experiments, such as those conducted by CMS (see Figure 2.3). At high energies ($Q \gg \Lambda$), corresponding to small distances, the coupling decreases, enabling perturbative calculations. This phenomenon in QCD is known as asymptotic freedom.

At smaller distances, quarks and anti-quarks form bound states known as hadrons, resulting in a diverse collection of mesons and baryons.

2.6 Limitations of the Standard Model

Several phenomena remain unexplained by the Standard Model, prompting investigations into physics beyond the SM. Below is a summary of key phenomena:

The *hierarchy problem* pertains to the question of why gravity is significantly weaker than the electroweak force. This issue is linked to the unexpectedly low mass of the Higgs boson compared to predictions from the Standard Model. The discrepancy necessitates fine-tuning and

suggests the existence of new physics beyond the SM. The Higgs boson lacks the symmetry protections that other particles enjoy, making its mass susceptible to large variations.

Matter-antimatter asymmetry poses a significant challenge in cosmology and particle physics. According to the Standard Model, matter and antimatter should have been created in nearly equal amounts, which is contradicted by the universe's predominant matter composition, as confirmed by astronomical observations. This asymmetry cannot be explained by the small amount of CP violation in the SM alone, and no significant antimatter signatures, such as gamma rays from annihilation events, have been observed.

Dark matter and dark energy account for approximately 95% of the universe's mass-energy content and present major challenges to the SM. Dark matter, inferred from galactic rotation rates and gravitational effects that exceed what can be explained by visible matter, has no candidate particle in the SM. Hypothetical particles such as WIMPs, axions, and sterile neutrinos are being explored. Meanwhile, dark energy, which drives the universe's accelerating expansion and is exemplified by the cosmological constant problem, cannot be explained by current physical laws. This suggests the need for alternative theories or modifications to general relativity.

Neutrino oscillations—the phenomenon of neutrinos changing flavor as they travel through space—represent a significant departure from the Standard Model's original assumption of massless neutrinos. This was first indicated by discrepancies in solar and atmospheric neutrino observations and later confirmed by experiments such as Super-Kamiokande and the Sudbury Neutrino Observatory. The discovery of neutrino mass raises important questions about the nature and scale of neutrino mass and its implications for our understanding of the early universe and fundamental forces.

Beyond these issues, the Standard Model faces further limitations, emphasizing the need for an extended or entirely new theoretical framework. One major shortcoming is the failure to fully incorporate *gravity* into the model. Additionally, the lack of clarity regarding *neutrino mixing angles* and *CP violation* in the quark sector suggests that the SM could be a low-energy approximation of a more fundamental theory. Detected *irregularities* in certain particle decays and properties—such as the *anomalous magnetic moment of the muon*—further indicate the existence of physics beyond the Standard Model. The search for a more comprehensive theory continues to drive both experimental and theoretical research in particle physics.

HIGH ENERGY PHYSICS EXPERIMENTS

3

The pursuit of *High Energy Physics* (HEP) has always aimed to unravel the fundamental nature of the universe. Experimental HEP focuses on observing physics at the smallest possible scales. To make an object visible, it must interact with other objects, typically through the interference with light (photons). The resolving ability of a photon, or any particle, is determined by its wavelength. According to de Broglie 1923, the relationship between the wavelength (λ) and the momentum (p) is given by:

$$\lambda = \frac{h}{p}. \tag{3.1}$$

This equation implies that achieving higher resolution (smaller λ) requires higher momentum and, consequently, higher energy.

As discussed previously, particle physics is fundamentally governed by Quantum Field Theory. In QFT, particles are considered excitations or *quanta* of underlying fields. Energy is required to excite these quanta, which, according to Einstein 1905, can be expressed as:

$$E = mc^2. \tag{3.2}$$

To generate heavy fundamental particles, such as the Higgs boson or the top quark, substantial energy is necessary.

The early universe was characterized by exceedingly high energy levels, similar to those recreated in modern particle accelerators. These high-energy environments offer insights into the early universe and allow for investigations into its origin.

Particle accelerators are essential to HEP and are designed to operate at increasingly higher energies. The most powerful of these is the LHC, which accelerates and collides bunches of protons. These collisions enable scientists to probe the fundamental nature of the universe, providing insight into the conditions and processes that shaped the early universe.

3.1 Scattering Experiments

The simplest process in QED is the scattering of electrons and positrons (e^+e^- scattering). This process offers a straightforward method to discover new particles in accelerators. The total cross-section of the scattering is proportional to the square of the electromagnetic coupling strength α^2.

For instance, at tree-level in QED, the total cross-section for $e^+e^- \to \mu^+\mu^-$ is given by:

$$\sigma = \frac{4}{3}\pi\frac{\alpha^2}{s} \approx \frac{86.8\,\text{nb}}{s[\text{GeV}^2]}, \tag{3.3}$$

where s is the square of the total energy in the center-of-mass frame.

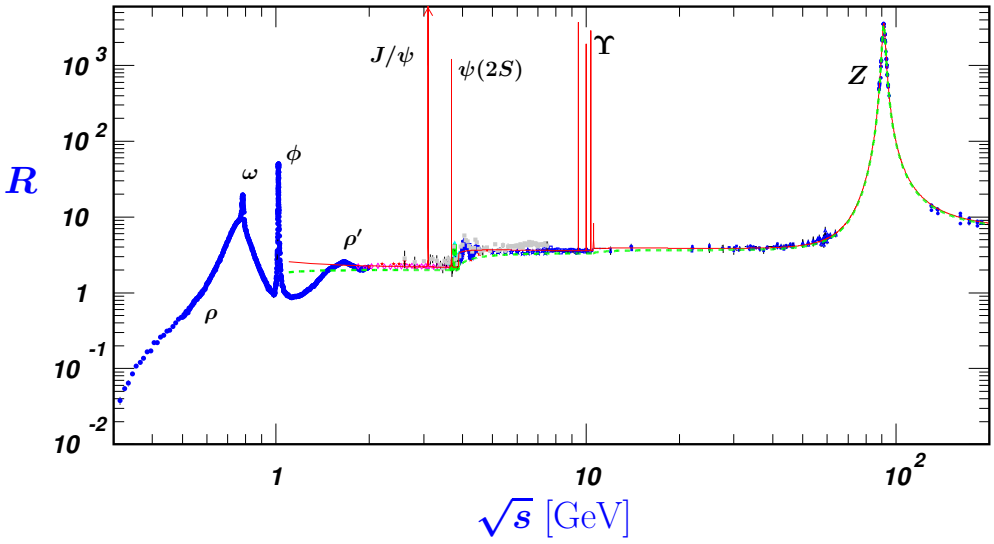

Figure 3.1: Graphical representation of the ratio R, illustrating the total cross section of electron-positron annihilation into hadrons relative to muons. The figure is by Ezhela et al. (2003). In that reference, there is also a comprehensive description of the methodology for extracting the R ratio.

Resonances at specific energies hint at the existence of particles with an invariant mass equal to $\sqrt{s}$. Figure 3.1 shows the ratio:

$$R(s) = \frac{\sigma(e^+e^- \to \text{hadrons}, s)}{\sigma(e^+e^- \to \mu^+\mu^-, s)}, \tag{3.4}$$

representing the cross-section as a function of $\sqrt{s}$, providing insight into these resonances. This figure demonstrates one of the notable successes of particle physics experiments in the past century.

Synchrotron radiation is emitted when charged particles are accelerated to relativistic speeds and forced to travel in curved paths by magnetic fields. As the particles are bent in their trajectories by the collider magnets, they emit synchrotron radiation, which leads to energy loss. The energy loss per revolution is proportional to γ^4, where:

$$\gamma = \frac{E}{mc^2}. \tag{3.5}$$

To achieve high-energy collisions in a circular collider, it becomes necessary to select particles with a high mass to minimize the energy loss due to synchrotron radiation. A natural choice for this purpose is to accelerate protons.

3.2 Physics of Proton-Proton Collisions

The LHC primarily collides protons to explore fundamental particles. Historically, other particles have also been used in collider experiments. For instance, electron-positron colliders, such as the *Large Electron-Positron Collider* (LEP) (CERN 1984), which previously occupied the same tunnel as the LHC, and proton-electron colliders, such as DESY's *Hadron-Electron*

Ring Accelerator (HERA) (Maidment 1986), played crucial roles in probing the structure of the proton.

Protons are not fundamental particles; they are composed of quarks and gluons, collectively referred to as partons. The proton's charge and baryon number are determined by its valence quarks. Specifically, a proton consists of two up quarks and one down quark, classifying it as a hadron. In addition to these valence quarks, sea quarks—quark-antiquark pairs spontaneously created and annihilated through vacuum fluctuations—also exist. The strong force binding quarks within the proton is mediated by gluons.

Understanding the structure of protons is essential for analyzing proton-proton (pp) collisions. The internal structure of the proton has been extensively studied, notably at facilities such as HERA (Abramowicz et al. 2015) in the pre-LHC era. During collision processes, the transferred momentum Q is used to probe the proton's inner structure. The momentum of the proton is shared among its constituent partons, and this distribution is described by the variable Bjorken-x, which represents the fraction of the proton's total momentum carried by a specific parton. Bjorken-x quantifies each parton's contribution to the proton's momentum during a given interaction, influencing the energy transfer and collision outcomes.

The center-of-mass energy of two colliding partons ($\sqrt{\hat{s}}$) depends on the proton collision energy ($\sqrt{s}$) and the partons' momentum fractions (x_1 and x_2). The *Parton Density Functions* (PDFs) describe the probability of finding a parton of a particular type with a given momentum fraction x at a specific momentum transfer Q. These PDFs are specific to each type of parton.

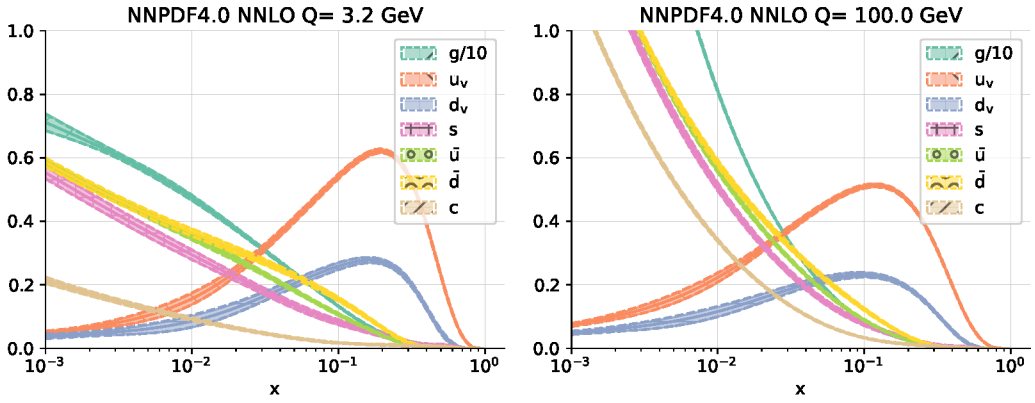

Figure 3.2: The NNPDF4.0 NNLO PDFs at $Q = 3.2\,\text{GeV}$ (left) and $Q = 100\,\text{GeV}$ (right). For more details, see the reference and source of the figure Ball et al. (2021).

Figure 3.2 presents examples of PDFs for momentum transfers at $Q = 3.2\,\text{GeV}$ (left) and $Q = 100\,\text{GeV}$ (right) (Ball et al. 2021). These examples show that interactions in pp collisions are predominantly influenced by valence quarks at higher x values and by sea quarks and gluons at lower x values.

The cross-section for a specific final state X in a proton-proton collision, $\sigma_{\text{pp}\to\text{X}}$, can be calculated using collinear factorization (Collins 2023). This process involves the convolution of the PDFs f_i and f_j with the parton interaction cross-section $\hat{\sigma}_{ij}$, where i and j represent the flavors of the initial state partons. The total cross-section is then obtained by integrating

over the momentum fractions x_1 and x_2, and summing over all possible initial state parton flavors:

$$\sigma_{\text{pp}} = \sum_{i,j} \iint f_i(x_1, q^2) f_j(x_2, q^2) \hat{\sigma}_{ij \to X}(x_1, x_2, q^2) \, \mathrm{d}x_1 \, \mathrm{d}x_2 \, . \tag{3.6}$$

In this equation, $f_i(x_1, q^2)$ and $f_j(x_2, q^2)$ are the PDFs for partons of flavors i and j, carrying momentum fractions x_1 and x_2 of the protons, evaluated at the factorization scale q^2. The term $\hat{\sigma}_{ij}(x_1, x_2, q^2)$ is the hard scattering cross-section of the parton interaction.

3.3 The Large Hadron Collider

The LHC is the most powerful particle accelerator in existence and represents a pinnacle of achievement in particle physics. Located beneath the France-Switzerland border near Geneva, the LHC operates under the auspices of *European Organization for Nuclear Research* (CERN, Conseil européen pour la Recherche nucléaire) and occupies the tunnel that once hosted LEP. This engineering marvel spans 26.7 kilometers and ranges in depth from 45 to 175 meters below the surface, enabling two proton beams to travel in opposite directions within adjacent vacuum tubes.

While the LHC primarily accelerates protons, it is also capable of accelerating heavy ions. This is achieved using superconducting radio frequency cavities that accelerate the particles, as well as superconducting magnets that maintain their circular trajectories and focus the beams along the tunnel. The proton acceleration process begins by extracting protons from hydrogen molecules through ionization. These protons are then accelerated through a series of precursors to the LHC, including the LINAC 2, the Booster, the Proton Synchrotron (PS), and the Super Proton Synchrotron (SPS). Through this process, proton energies are gradually increased to 450 GeV before they are injected into the LHC, where they are further accelerated to the final energy of 6.8 TeV.

Protons collide in bunches at intervals of 25 ns, contributing to the LHC's high luminosity and interaction rate. The luminosity (L) and cross section (σ) determine the number of particle interactions per unit time:

$$\frac{\mathrm{d}N}{\mathrm{d}t} = L\sigma. \tag{3.7}$$

The LHC hosts four primary experiments, as shown in Figure 3.3: ATLAS (A Toroidal LHC ApparatuS) (ATLAS Collaboration [ATLAS] 2008) and CMS (Compact Muon Solenoid) (CMS 2008), which are multipurpose detectors designed to study a variety of physical processes and search for new phenomena; ALICE (A Large Ion Collider Experiment) (ALICE Collaboration [ALICE] 2008), which specializes in heavy ion collisions to study the quark-gluon plasma and provides insights into the state of the universe shortly after the Big Bang; and LHCb (Large Hadron Collider beauty) (LHCb Collaboration [LHCb] 2008), which focuses primarily on studying B hadrons and investigating the mechanisms of CP violation.

In 2012, the LHC made a major discovery with the detection of the Higgs boson (ATLAS 2012; CMS 2012), marking a milestone in understanding the fundamental structure of matter.

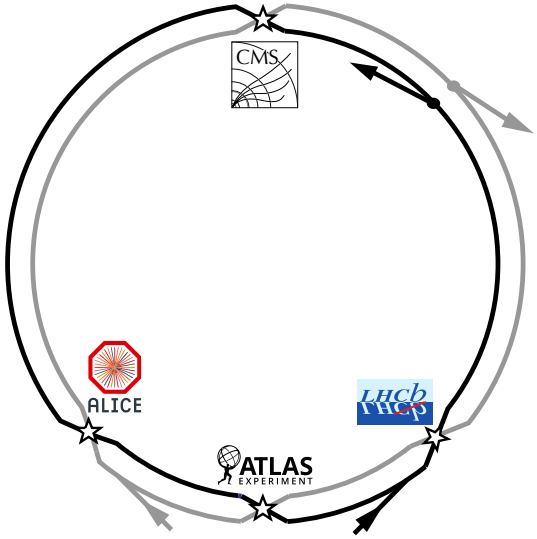

Figure 3.3: The Large Hadron Collider ring, surrounded by the logos of the LHC experiments: ATLAS, CMS, LHCb, and ALICE.

The LHC was designed to achieve a maximum center-of-mass energy of $\sqrt{s} = 14\,\text{TeV}$ and a luminosity of $L = 10 \times 10^{35}\,\text{cm}^{-2}\text{s}^{-1}$. Luminosity allows the calculation of particle collision rates, and the instantaneous luminosity of the LHC varies according to the specific requirements of each experiment:

$$L = f\frac{N_a N_b}{4\pi\sigma_x\sigma_y}, \tag{3.8}$$

where f is the collision frequency, N_a and N_b are the number of particles in bunches a and b, and σ_x and σ_y are the transverse beam sizes in the x- and y-directions, respectively.

The *primary vertex* (PV) is the point where hard scattering events occur within a proton-proton (pp) bunch crossing. Accurately associating particle tracks and energy deposits with the PV, while minimizing contamination from secondary collisions (known as *pile-up*), is a key challenge. This challenge arises from the multiple inelastic pp-collisions expected in each bunch crossing.

3.4 The CMS Experiment

The CMS Experiment

The CMS detector is a multipurpose apparatus designed to study proton-proton collisions at a center-of-mass energy of 14 TeV. It has a cylindrical design that encircles the interaction point, with a radius of 15 m and a length of 21.6 m. The CMS is engineered to record and analyze the wide spectrum of particles produced during collisions, including providing indirect detection of neutrinos.

The detector consists of several sub-detectors, each designed for specific tasks in particle identification and measurement. These sub-detectors include a tracker, an *Electromagnetic*

Calorimeter (ECAL), a *Hadronic Calorimeter* (HCAL), and a muon system. The superconducting solenoid magnet, with a diameter of approximately 6 m and a field strength of up to 3.8 T, is positioned between the muon chambers and the HCAL. Figure 3.4 provides a visualization of the overall detector structure.

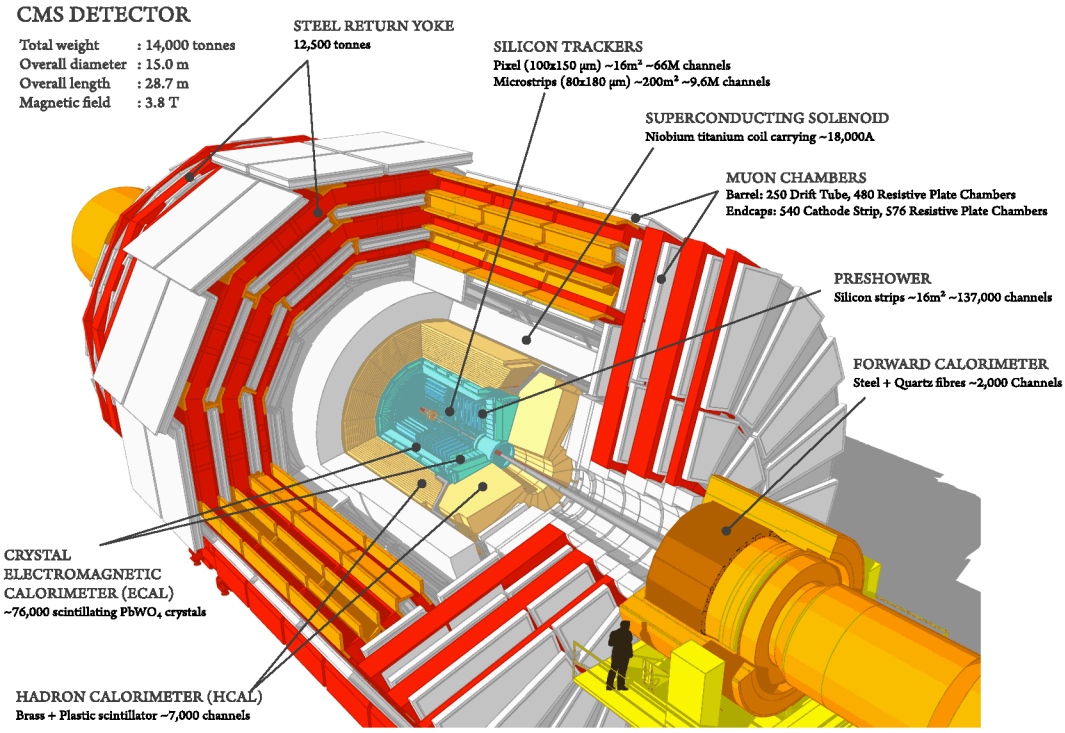

Figure 3.4: An overview of the CMS detector and its subdetectors. The figure is adapted from Ref. CMS (2019).

3.4.1 *Coordinate System and Conventions*

To describe the functionality and structure of the CMS sub-components, we first introduce the coordinate system used in the CMS experiment, along with other relevant physical conventions.

The CMS experiment employs a right-handed Cartesian coordinate system, with its origin at the collision point of the proton beams. The z-axis points along the beam direction, the y-axis points upwards, and the x-axis points towards the center of the accelerator. The radial distance r is measured from the interaction point at the center of the detector. The polar angle θ is defined relative to the z-axis, while the azimuthal angle ϕ spans the x-y-plane.

In particle physics analysis, it is essential to use variables that are Lorentz invariant under boosts along the z-axis. One such quantity is the transverse momentum:

$$p_T = \sqrt{p_x^2 + p_y^2}. \tag{3.9}$$

The azimuthal angle ϕ remains Lorentz invariant under boosts along z, unlike the polar

angle θ. To address this, pseudorapidity η is introduced as an alternative:

$$\eta = -\ln\left(\tan\left(\frac{\theta}{2}\right)\right). \tag{3.10}$$

Differences in η are Lorentz invariant under boosts in z. The angular separation between two objects in the detector is defined by ΔR, which is derived from invariant quantities:

$$\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}. \tag{3.11}$$

3.4.2 *Tracking Systems*

The CMS tracking system is an assembly of silicon-pixel and silicon-strip detectors arranged in concentric layers around the interaction point. This system detects and tracks charged particles by generating measurable signals as the particles pass through sensitive silicon diodes. Using advanced pattern recognition techniques, these signals—referred to as tracker hits—are combined to reconstruct particle tracks with high precision.

The CMS tracking system consists of two main components: a high-resolution pixel detector near the interaction point, and a larger, less granular tracker surrounding the pixel detector. The pixel detector comprises four layers in the barrel region and three layers in each endcap, with the first pixel layer positioned just 2.9 cm from the LHC beam. This proximity, coupled with the pixel size of 100 μm×150 μm, significantly enhances the precision of primary vertex reconstruction. The outer tracker, made of silicon strips, extends over 5 m in length and encompasses approximately 200 m^2 of silicon.

3.4.3 *Electromagnetic Calorimeter*

The ECAL is one of the core subdetectors of the CMS detector. For a detailed explanation of electromagnetic calorimeters, see Section 4.3.1. The ECAL consists of 75,848 lead tungstate ($PbWO_4$) crystals and is designed for precise measurement of electron and photon energies. It is a homogeneous calorimeter with a dense, radiation-hard structure, ensuring excellent energy resolution and fast response. Its primary function is to fully contain and measure the energy of electromagnetic showers initiated by electrons and photons.

The ECAL is divided into two main components: the ECAL Barrel (EB) and the ECAL Endcaps (EE). The EB comprises two half-cylinders, each made up of 18 super-modules. Each super-module contains 1,700 crystal bars and covers a significant solid angle with a granularity of $0.0174[\eta]\times0.0174[\phi]$. The EB spans a solid angle up to $|\eta| = 3$, although there is a gap in the instrumentation between $1.479 < |\eta| < 1.653$, which affects the reconstruction of electrons and photons (CMS 2008).

Lead tungstate crystals were chosen for their short radiation length ($X_0 = 0.89$ cm), high density ($\rho = 8.29$ g/cm^3), and small Molière radius ($R_m = 2.2$ cm), which are crucial for transverse shower containment. The rationale behind these choices will be discussed in greater detail in Chapter 4. This configuration allows for the collection of over 81

When an electron or photon passes through the ECAL, it emits energy through bremsstrahlung and pair production of $e^{\pm}$. The scintillators measure the photon energy using photodiodes, with the relative energy resolution σ/E, where σ represents the resolution and E is the measured energy (in GeV) (CMS 2008):

$$\frac{\sigma}{E} = \frac{2.8\%}{\sqrt{E/[\text{GeV}]}} \oplus \frac{0.12\%}{E/[\text{GeV}]} \oplus 0.3\%. \tag{3.12}$$

3.4.4 *Hadron Calorimeter*

The purpose of the HCAL in the CMS detector is to measure the energy of hadrons. For a detailed explanation of hadron calorimeters, see Section 4.3.2. It complements the ECAL by focusing on particles that interact via the strong force. The HCAL is designed to capture and measure the energy of hadrons, such as protons, neutrons, and pions, that undergo inelastic reactions with the detector material, creating hadronic showers.

Hadronic showers are more complex than electromagnetic showers due to their varied interaction mechanisms and the types of secondary particles produced. The spatial development of these showers is characterized by significant fluctuations, often resulting in missing energy. Accurately measuring hadronic showers is thus a challenging task. Notably, hadrons may begin interacting within the ECAL, depositing around 30

The HCAL is positioned between the ECAL and the superconducting magnet, with its design shaped by the constraints of the magnet's geometry. The HCAL operates as a sampling calorimeter, consisting of alternating layers of brass absorbers and scintillating plastic tiles, allowing it to effectively measure energy from both charged and neutral hadrons.

The HCAL is segmented into four regions: the central barrel (HB), the outer barrel (HO), the endcaps (HE), and the forward region (HF). The HB, HO, and HE regions have a granularity of $0.087[\eta] \times 0.087[\phi]$, providing broad coverage. In contrast, the HF region offers a finer angular resolution of $0.0175[\eta] \times 0.0175[\phi]$, tailored for detecting particles in the forward direction.

The scintillators produce light, which is collected by 1 mm diameter optical fibers that shift the wavelength for efficient light transmission. The HCAL has an average depth of about 11 hadronic interaction lengths (λ_i), allowing it to absorb and measure the energy of penetrating hadronic showers effectively.

However, the HCAL has a significantly lower energy resolution compared to the ECAL. The energy resolution (CMS 2008) is given by:

$$\frac{\sigma}{E} = \frac{115.3\%}{\sqrt{E/[\text{GeV}]}} \oplus 5.5\%, \tag{3.13}$$

where σ/E represents the relative energy resolution and E is the energy measured in GeV.

3.4.5 *Solenoid*

The 'S' in CMS stands for the superconducting solenoid, which is critical for the detector's ability to accurately measure and identify particles. The solenoid's cylindrical magnet coil has a diameter of 6 meters, a length of 12.5 meters, and weighs 220 tons. It generates a magnetic field of up to 4 Tesla inside the coil, storing approximately 2.6 GJ of energy.

The magnetic field produced by the CMS bends the trajectories of charged particles in the transverse plane. This curvature is essential for measuring particle momenta with high precision, thereby improving particle identification and reconstruction. The process is governed by the Lorentz force equation, $\vec{F}_L = q(\vec{v} \times \vec{B})$, where q represents the particle's charge, $\vec{v}$ its velocity, and $\vec{B}$ the magnetic field. For accurate estimation of a particle's charge and momentum, it is crucial that the magnetic field within the detector is precisely known, especially in the inner regions where particle trajectories are meticulously recorded.

The magnet coil is made from an alloy of niobium and titanium (NbTi), which has superconducting properties allowing it to conduct a nominal current of approximately 19.14 kA. To maintain these superconducting properties, the magnet is cooled using a cryostat system that employs liquid helium to reach a temperature of 4.7 Kelvin.

3.4.6 *The Muon System*

The "M" in CMS stands for muon, as the muon system is a cornerstone of the detector. This system is crucial for detecting muons, which are commonly produced in high-energy physics interactions, including the decay of heavier particles. These events are of great interest as they may reveal new physics. Muons, with a mass of 105.7 MeV, are less likely to interact with the calorimeters compared to other particles, allowing them to pass through the inner components of the detector nearly unscattered and reach the muon chambers, which form the outermost layer of the CMS detector.

The muon system is designed to identify muons and measure their momentum with high precision. It works in conjunction with the CMS's superconducting solenoid magnet, which bends muon trajectories in the transverse plane, improving momentum measurement. The system incorporates several technologies: Drift Tubes (DTs) in the barrel region ($|\eta| < 1.2$), Cathode Strip Chambers (CSCs) in the endcap regions ($1.0 < |\eta| < 2.4$), and Resistive Plate Chambers (RPCs). The DTs and CSCs are essential for muon identification and momentum measurement, while the RPCs, with excellent timing capabilities, provide a robust triggering system, despite having relatively less precise position measurements (CMS 2008).

The muon spectrometer can detect muons within a pseudorapidity range of $|\eta| < 2.4$. The design includes multiple measuring planes per station—12 in the barrel and 6 in the endcap regions. This enables the reconstruction of a global track by combining hits from both the inner tracker and the muon stations.

Muon momentum is estimated by analyzing the sagitta of its trajectory. Combining data from the inner tracker and muon stations results in exceptional momentum resolution. This integrated approach allows the CMS to achieve a momentum resolution of approximately 1

3.4.7 *The Trigger System*

Collisions in the CMS detector occur at a frequency of approximately 40 MHz, generating 40 million events per second. Each event involves around 25 simultaneous proton-proton interactions, producing data volumes averaging 1 MB per event. This results in a total data rate of about 40 TB per second, far exceeding the available storage capacity of the CMS collaboration. Consequently, an efficient filtering system is required to manage this massive data influx, and this is achieved through the trigger system.

The trigger system plays a crucial role in data pre-selection by identifying and recording only potentially interesting events for further analysis, effectively reducing the data volume to a manageable level. Storing every event is impractical, so the trigger system is designed to filter out irrelevant or less significant events.

CMS employs a two-stage trigger system to refine this process. The first stage, known as the Level-1 (L1) trigger, operates using programmable hardware processors and can process data at rates of up to 100 kHz. This stage rapidly evaluates incoming data for specific particle signatures or characteristics, such as identifying muons within the muon system, to determine whether an event warrants further scrutiny.

After the L1 trigger, the High-Level Trigger (HLT) performs a more detailed and comprehensive reconstruction of the event using data from all detector components. The HLT employs algorithms similar to those used in later data analysis to ensure that only events meeting specific selection criteria are saved for in-depth study. This stage reduces the event rate from 40 MHz to a more manageable 200 Hz to 1 kHz, balancing the need for comprehensive data capture with storage constraints. Unlike the L1 trigger, the HLT uses sophisticated algorithms executed on a computing farm composed of tens of thousands of standard CPU cores. This allows for thorough event analysis before data is stored for long-term usage and full event reconstruction.

CALORIMETRY

4

The measurement of particle energy is a fundamental aspect of a wide range of particle physics experiments. Calorimeters are a category of detectors specifically designed for this purpose. In modern experiments, their role includes accurately reconstructing four-vectors and determining the energy flow in complex event signatures, such as jets and missing transverse energy.

Calorimeters measure the energy of incident particles through a destructive process. As particles pass through the material of the calorimeter, they interact with it and produce a cascade of secondary particles, known as a shower. Figure A illustrates an example of such a shower.

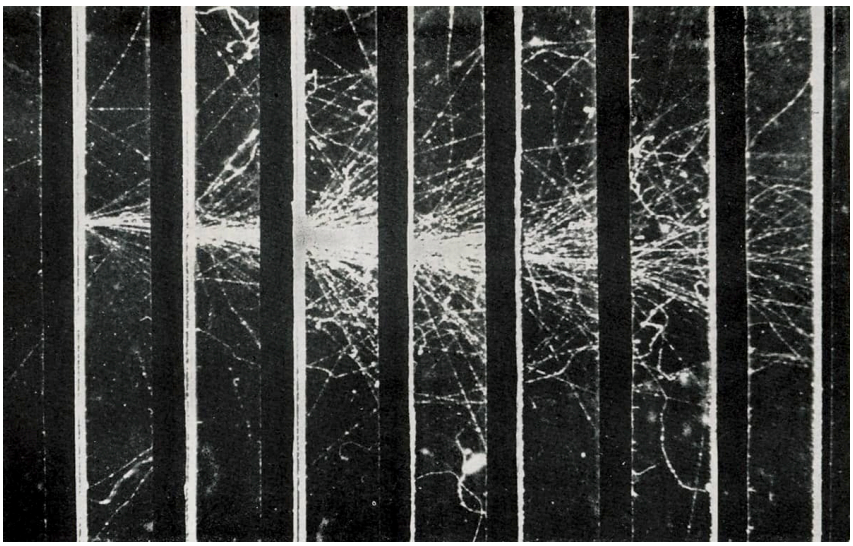

Figure 4.1: Image of a high-energy electron shower. The figure is adapted from (S. Lee et al. 2018).

This particle avalanche generates detectable signals within the calorimeter, allowing for the measurement of the total energy of the incident particle. It is important to note that the sum of the measured energy can differ significantly from that of the original particle. Additionally, the development of the particle cascade is strongly influenced by the type of incident particle, as each particle induces specific interactions within the calorimeter. These aspects will be examined in greater detail in the subsequent sections.

4.1 Interactions of Particles with Matter

4.1.1 *Electromagnetic Interactions*

Electrons and positrons deposit energy in material primarily through ionization and *Bremsstrahlung* (Kolanoski & Wermes 2016). Ionization occurs when these particles transfer energy to

atoms, leading to excitation or the ejection of atomic electrons. This process dominates at lower energies. Conversely, at higher energies, Bremsstrahlung becomes significant. Bremsstrahlung refers to the radiation emitted when a charged particle is decelerated by the Coulomb field of an atomic nucleus. At high energies, Bremsstrahlung losses dominate, increasing linearly with energy, whereas ionization losses grow logarithmically with particle energy (Workman et al. [PDG] 2022). Figure 4.2 illustrates the energy deposition of electrons and positrons at different energies.

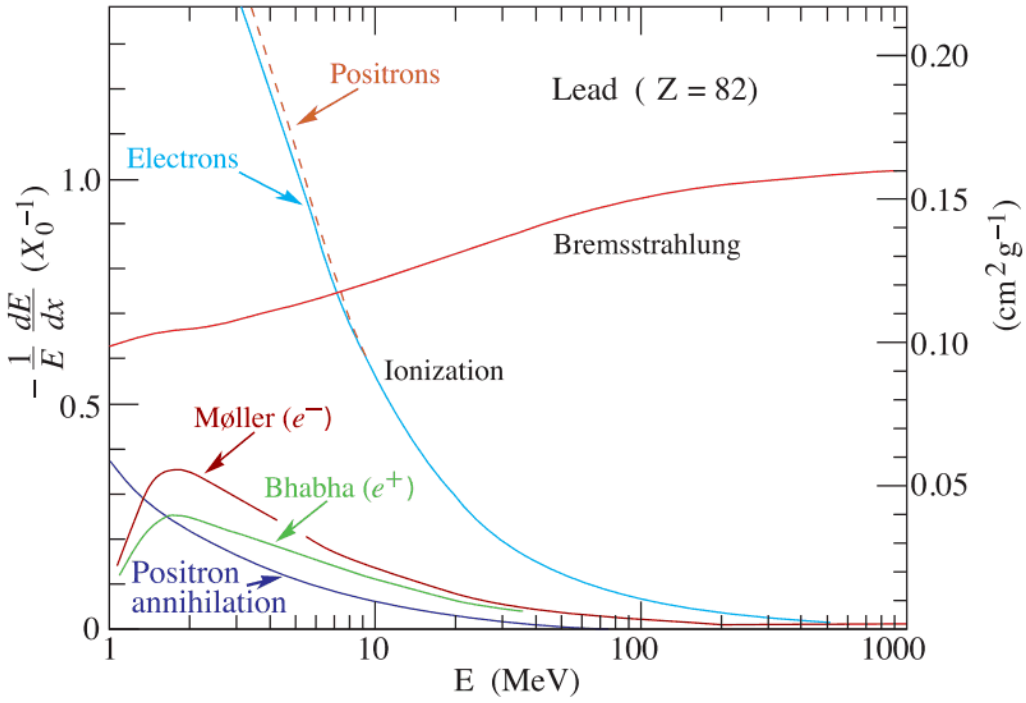

Figure 4.2: Illustration of the different energy loss fractions of electrons and positrons when passing through lead (PDG 2022).

For heavy charged particles, such as muons or charged hadrons, ionization and excitation dominate energy loss, as Bremsstrahlung is suppressed by the particle mass ($1/m^4$). The *Bethe-Bloch* formula provides the average energy loss per unit distance for heavy charged particles:

$$-\left\langle \frac{\mathrm{d}E}{\mathrm{d}x} \right\rangle = Kz^2 \frac{Z}{A} \frac{1}{\beta^2} \left[\frac{1}{2} \ln \frac{2m_e c^2 \beta^2 \gamma^2 T_{\max}}{I^2} - \beta^2 - \frac{\delta(\beta\gamma)}{2} \right]. \tag{4.1}$$

At low particle energies, the $1/\beta^2$ term dominates, resulting in a fixed range for particles that lose energy primarily through ionization. The Bragg peak represents the point of maximum energy deposition, located near the end of this range. This is depicted in Figure 4.3.

Photons interact with matter through several processes, including the photoelectric effect, Rayleigh scattering, the Compton effect, and pair production (Wigmans 2018). The cross section of a photon interacting with gold is shown in Figure 4.4. At low energies, the photoelectric effect dominates, with the cross-section proportional to E^{-3}, resulting in the ejection of an electron from an atomic shell. At higher energies, Rayleigh scattering, a coherent elastic scattering where the photon is deflected without energy loss, and the Compton effect, where a photon scatters off a free or quasi-free electron, become more significant. The

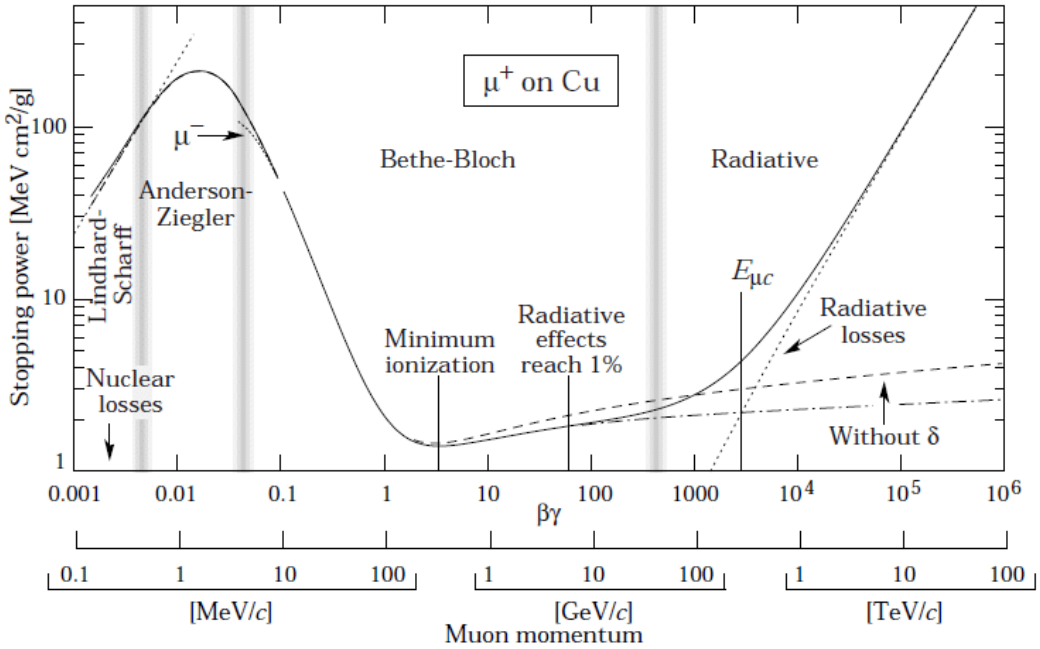

Figure 4.3: Graphical representation of the stopping power of muons in copper as a function of momentum, illustrating various energy loss mechanisms and transitions, adapted from PDG (2022).

cross-section for the Compton effect is proportional to $1/E$, making it dominant between a few hundred keV and around 5 MeV. Pair production occurs when the photon's energy exceeds twice the electron mass, allowing the creation of an electron-positron pair in the Coulomb field of a nucleus, which absorbs the recoil to conserve momentum. This process cannot occur in a vacuum (Wigmans 2018). At energies around 10 MeV, photons can excite atomic nuclei to resonant states, leading to the emission of nucleons and high-energy photons, though the cross-section for nuclear interaction is small compared to pair production. Hadronic interactions with atomic nuclei occur primarily via incident hadrons.

4.1.2 *Hadronic Interactions*

The electromagnetic interactions of charged hadrons are primarily governed by ionization, with Bremsstrahlung playing a much smaller role in energy deposition. The main mechanism for energy loss in hadronic interactions occurs through inelastic hard scattering with the nuclei of the traversed material, mediated by the strong force. In these interactions, the incident hadron transfers a significant portion of its momentum to the nucleus, which can lead to various nuclear reactions.

Spallation is the most common outcome of hadronic interactions and is characterized as a two-stage process involving an initial intranuclear cascade, followed by an evaporation phase (Wigmans 2018). During the intranuclear cascade, the incident hadron interacts quasi-freely with atomic nucleons, initiating a series of scatterings. The hadron scatters off

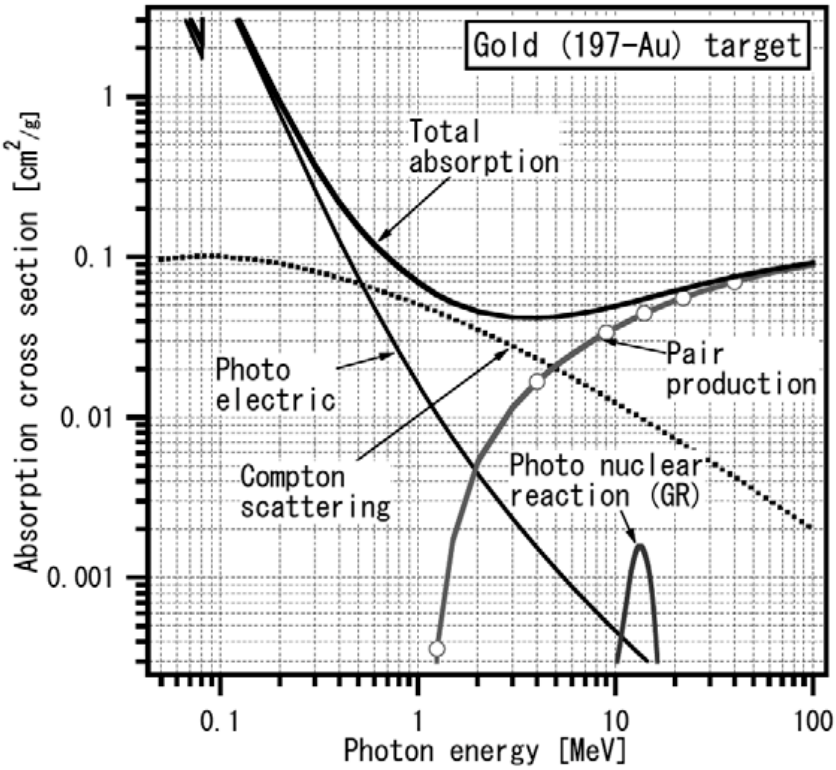

Figure 4.4: Illustration of the different cross-section fractions for photons interacting with gold (Miyamoto & Horikawa 2008).

quasi-free nucleons within the nucleus, causing these nucleons to propagate further and scatter other nucleons. Some of these nucleons may escape the nucleus, and if the incident energy is high enough, the cascade may lead to nuclear fission. The intranuclear cascade generates a variety of particles within the nucleus, some of which escape and propagate through the surrounding medium.

During the cascade, pions and other unstable hadrons are produced. The particles that escape the nucleus and propagate through the medium typically have energies in the GeV range (Fabjan & Gianotti 2003).

The evaporation phase follows the cascade. In this stage, the excited nucleus de-excites through the isotropic emission of free nucleons and photons. These particles generally have energies around 1 MeV. After the evaporation phase, the remaining energy in the nucleus is released through photon emission. Notably, neutral pions—constituting approximately one-third of all pions produced during spallation—decay almost instantly (with a mean lifetime at rest of approximately 10^{-16} s) into a pair of photons.

The determination of hadronic cross sections relies on phenomenological models and empirical measurements, as Quantum Chromodynamics (QCD) is not well described in the low-energy regime, where most hadronic interactions occur.

4.2 Development of Particle Showers

As particles pass through a material, they lose energy due to interactions with the atoms in the medium. This lost energy is transferred to secondary particles, which collectively form a particle shower. Only certain types of particles reach the calorimeters before decaying. Electrons, positrons, and photons ($e^{\pm}$, γ) induce electromagnetic showers, while charged and neutral hadrons—such as pions ($\pi^{\pm}$), kaons ($K^{\pm}$, K^0), protons (p), and neutrons (n)—produce hadronic showers. In contrast, muons (μ) exhibit minimal showering due to their weak interactions with the medium, and neutrinos (ν) are even less likely to interact, rarely producing detectable showers in calorimeters.

Given the complexity and variety of interactions involved in particle shower formation, it is impractical to trace all interactions at the microscopic level. Therefore, for most calorimetry applications, a macroscopic description of shower development is used. This approach emphasizes the general characteristics of the showers, such as their longitudinal and transverse development, rather than accounting for every individual interaction.

4.2.1 *Electromagnetic Cascades*

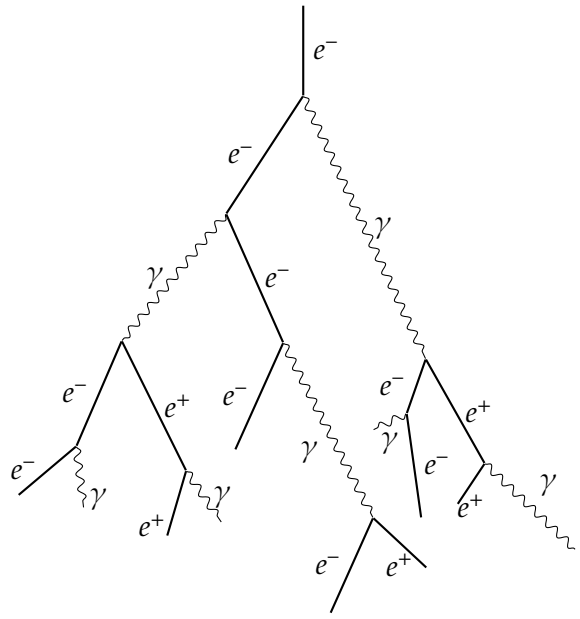

Figure 4.5: Illustration of the development of an electromagnetic shower.

Electromagnetic cascades are fundamental phenomena that occur when high-energy electrons, positrons, or photons penetrate dense matter. The initiation and development of these cascades are primarily governed by two processes: Bremsstrahlung and pair production. Bremsstrahlung is the dominant mechanism for electrons and positrons ($e^{\pm}$), resulting in the creation of photons. Conversely, pair production predominates for photons (γ), leading to the generation of new $e^{\pm}$ pairs. These secondary particles in turn produce additional particles, initiating a cascade with decreasing energies. This multiplicative process culminates in the formation of a particle shower, as illustrated in Figure 4.5.

As $e^{\pm}$ penetrate a material, they lose energy. To quantify this energy loss, the concept of radiation length (X_0) is introduced. X_0 represents the average distance over which a high-energy electron loses approximately 63.2

$$X_0 = \frac{716A}{Z(Z+1)\ln\left(\frac{287}{\sqrt{Z}}\right)} \left[\frac{\mathrm{g}}{\mathrm{cm}^2}\right] \tag{4.2}$$

where Z is the atomic number and A is the mass number of the nucleus, making X_0 a material-dependent parameter. The longitudinal development of an electromagnetic shower is proportional to $\ln(E_0)$, indicating that the shower length increases with the logarithm of the initial energy of the particle.

For photons, the interaction length (λ_{pair}) is defined as the mean free path before a γ decays into an $e^{\pm}$ pair. At high photon energies, the γ interaction length is approximately proportional to 7/9 of the radiation length X_0.

The energy density per depth segment of a longitudinal shower can be approximated by a gamma distribution. The multiplication of shower particles ceases when the particles' average energy drops to the critical energy ϵ_c, where energy losses due to ionization and Bremsstrahlung balance out. This critical energy is material-dependent and inversely proportional to the atomic number Z, as shown by:

$$\epsilon_c = \frac{610\,\mathrm{MeV}}{Z + 1.24}. \tag{4.3}$$

For example, the critical energy for electrons passing through lead is approximately 7 MeV. Unlike electrons, muons in the GeV energy regime predominantly lose energy via ionization, resulting in a critical energy much higher than that of electrons.

The multiplication of particles within the cascade scales roughly with E_0/ϵ_c, and the cascade growth diminishes as the average electron energy falls below ϵ_c. The transverse development of the shower, relative to the main propagation axis, is caused by multiple scattering of low-energy electrons in the Coulomb fields of atoms and back-scattering from Compton scattering, which contribute to the transverse spread of the shower.

The Molière radius (R_m) is introduced as a measure of this transverse spread. It defines the radius of a hypothetical cylinder around the shower axis that contains 90

$$R_m = 21.2\,\mathrm{MeV}\,\frac{X_0}{\epsilon_c}. \tag{4.4}$$

4.2.2 *Hadronic Cascades*

The development of hadronic showers is significantly more complex than that of electromagnetic showers, largely due to the multitude of interactions associated with the strong force. In contrast to the limited number of processes in electromagnetic showers, hadronic showers involve various mechanisms during their evolution (Wigmans 2018). A key factor in this

complexity is the nuclear interactions experienced by the struck nucleus, as opposed to electromagnetic showers, where only the small binding energy of electrons to nuclei is relevant (Wigmans 2018).

Charged hadrons lose part of their energy via ionization as they traverse a medium, eventually producing high-energy secondary particles through inelastic processes. For neutral hadrons, it is primarily the inelastic processes that dominate, highlighting a distinct difference in energy deposition mechanisms between charged and neutral hadrons (Fabjan & Gianotti 2003; Wigmans 2018). The distance between successive hadronic interactions is characterized by the hadronic interaction length (Fabjan & Gianotti 2003).

The nuclear interaction length within an absorber medium refers to the average distance a high-energy hadron travels before undergoing a nuclear interaction. This is analogous to the mean free path for high-energy photons (Wigmans 2018). The probability of a particle traveling a distance t in the medium without inducing a nuclear interaction provides insight into the interaction dynamics.

The total cross-section for nuclear interactions, σ_{tot}, is inversely related to the nuclear interaction length, λ_{int}, and directly proportional to the atomic weight A of the nuclei involved. This relationship is given by:

$$\sigma_{\text{tot}} = \frac{A}{N_A \lambda_{\text{int}}}, \tag{4.5}$$

where N_A is Avogadro's number. The cross-section is influenced by both the size of the projectile and the target nuclei, with the cross-section of the target scaling with the square of its radius. The nuclear interaction length λ_{int} scales with $A^{1/3}$ when expressed in units of g cm^{-2} (Wigmans 2018).

Secondary hadrons produced in these inelastic collisions propagate through the detector until they are absorbed, contributing to the ongoing development of the shower. During nuclear spallation, hadrons can decay into photons and neutral pions, which in turn decay into photons before engaging in hadronic interactions. This results in a portion of the hadronic shower energy being converted into an electromagnetic sub-shower. The fraction of energy transferred to the electromagnetic component increases with the energy of the incoming hadrons (Fabjan & Gianotti 2003).

The electromagnetic component of a hadronic shower is subject to considerable fluctuations, as the fraction of electromagnetic energy depends on the initial processes at the start of the shower (Wigmans 2018). Moreover, hadronic showers tend to have a much larger spatial extent than electromagnetic showers, particularly in materials with a high nuclear charge. This difference arises from the variations in the cross-sections for electromagnetic and strong interactions.

Unlike electromagnetic showers, not all of the energy from a hadronic shower is detectable. Some shower products—such as delayed photons, soft neutrons, and the binding energy of hadrons and nucleons—are invisible to traditional energy measurement techniques. Furthermore, hadronic showers exhibit significantly larger spatial expansion compared to elec-

tromagnetic showers, especially in materials with a large nuclear charge. These differences underscore the distinct nature of hadronic showers (Wigmans 2018).

4.3 Calorimeter

Calorimeters are instruments used to measure the energy of incident particle showers through their interactions within the calorimeter material. Their design and operational principles are specifically tailored to absorb particles and determine their energy. There are two primary types of calorimeters: electromagnetic and hadronic, each optimized for measuring different types of particle interactions. This distinction arises from the differing interaction mechanisms and resulting shower shapes, as discussed in the preceding sections.

Calorimeters are further classified into two types based on their construction and energy measurement method: homogeneous and sampling calorimeters.

Homogeneous calorimeters use a single material to both absorb the incident particles and generate a measurable signal. These calorimeters typically utilize heavy inorganic scintillation crystals or non-scintillating Cherenkov radiators to measure the energy of incoming particles. The material must provide a clear and measurable response to particle interactions, enabling precise energy measurements (PDG 2022).

Sampling calorimeters are constructed with alternating layers of active and passive materials. The passive layers scatter and slow down the particles, spreading them out to ensure a more uniform energy deposition across the active layers. The active layers then generate a signal through processes such as ionization or scintillation, which allows the energy deposited by the particle shower to be measured. Passive layers often consist of heavy metals like lead, iron, copper, or uranium, chosen for their high density and atomic number, which enhance particle absorption. Active layers, on the other hand, may use liquid noble gases or organic and inorganic scintillators, selected for their ability to produce a measurable signal in response to particle interactions (PDG 2022).

4.3.1 *Electromagnetic Calorimeter*

Electromagnetic calorimeters are essential tools in high-energy physics experiments for precisely measuring the energy of electrons, positrons, and photons. The construction and performance of these devices are heavily influenced by the depth of the material required to fully absorb the incident particles. A notable example is the crystal ECAL at the CMS detector, which has a depth of 23 cm. This corresponds to approximately 26 radiation lengths (X_0), optimizing the calorimeter to effectively contain electromagnetic showers within a compact volume. The depth is crucial, as it is designed to accommodate the full extent of particle showers, ensuring that the calorimeter captures the total energy of the incident particles.

The performance of electromagnetic calorimeters can be quantitatively described by their

relative energy resolution, expressed as:

$$\frac{\sigma}{E} = \frac{a}{\sqrt{E}} \oplus b \oplus \frac{c}{E} \tag{4.6}$$

where $\oplus$ represents the square sum of the individual components contributing to the resolution. This equation includes three primary terms: the *stochastic term*, the *constant term*, and the *noise term*.

The *stochastic term* accounts for fluctuations in the number of charged tracks within the active medium, which significantly affects the overall resolution. In sampling calorimeters, this term is influenced by the thickness of the absorber (t) in units of radiation length (X_0) and varies inversely with the square root of the incident energy (E). This relationship emphasizes the impact of shower development and the geometry of the calorimeter on its energy resolution, assuming that the number of charged tracks in individual layers are independently distributed and follow a Gaussian distribution (Amaldi 1981).

The *noise and leakage term* arises from electrical noise during the signal processing phase, contributing a baseline level of uncertainty to the energy measurement.

The *constant term* represents energy-independent effects, such as inhomogeneities in the detector structure, fabrication inaccuracies, temperature gradients, and radiation damage. These factors can introduce systematic uncertainties, impacting the calorimeter's resolution and underscoring the importance of meticulous design and construction practices to minimize their effect (Fabjan & Gianotti 2003).

4.3.2 *Hadronic Calorimeter*

Hadronic calorimeters face unique challenges in accurately measuring the energy of hadronic showers due to the nature of the interactions occurring within them. Unlike electromagnetic showers, where nearly all the energy is detectable, hadronic showers include components such as delayed photons, soft neutrons, and the binding energy of hadrons and nucleons, which are invisible to standard energy measurement techniques (Fabjan & Gianotti 2003). This results in a systematically lower signal for hadrons compared to electrons, affecting the calorimeter's ability to accurately measure hadronic energy.

The efficiency of energy measurement in hadronic calorimeters is often quantified by the ratio of the response to electrons (e) and hadrons (h), denoted as e/h. Ideally, this ratio should equal one for a calorimeter to be considered *compensating*, meaning it would deliver equivalent signals for both hadrons and electrons (Wigmans 2018). However, achieving compensation is inherently difficult due to the internal properties of the calorimeter, which influence the e/h ratio but cannot be measured directly. Instead, the e/π ratio, representing the signal response to electrons and pions, is used as a proxy to estimate e/h through the relationship:

$$\frac{e}{\pi} = \frac{e/h}{1 - f_{\text{em}} - e/h}, \tag{4.7}$$

where f_{em} denotes the fraction of the electromagnetic shower component, and its value depends on the energy of the incident pion (Wigmans 2018).

Compensation is critical for improving both the linearity and resolution of hadronic calorimeters. Non-compensating calorimeters suffer from non-linear responses, as the proportion of the electromagnetic component of the shower increases with the energy of the incident particle, leading to stronger signals for higher-energy particles. This non-linearity, coupled with fluctuations in the electromagnetic shower component, can significantly degrade the calorimeter's resolution. Therefore, achieving compensation, where e/h is close to 1, is a vital design goal for hadronic calorimeters, leading to more accurate and reliable energy measurements (Fabjan & Gianotti 2003; Kolanoski & Wermes 2016; Wigmans 2018).

To approach compensation, strategies focus on reducing the electromagnetic signal while enhancing the hadronic signal. Utilizing absorber materials with high nuclear charge effectively diminishes the electromagnetic component. This reduction occurs because a substantial portion of the electromagnetic shower energy is absorbed through low-energy photons, which, in materials with high nuclear mass, generate electrons that fail to reach the active medium and thus do not contribute to the signal. Conversely, increasing the hadronic fraction involves enhancing the detection of cold evaporation neutrons. Since the energy transfer from neutrons is inversely proportional to the nuclear mass (A) of the material, neutrons can traverse a passive medium with minimal energy loss and efficiently transfer their energy to an active medium with a lower A or one containing hydrogen. Mechanisms such as varying the thicknesses of active and passive layers or enriching the active medium with hydrogen aim to boost the signal from the nuclear components of the shower, thereby improving the overall performance and accuracy of the hadronic calorimeter.

4.4 Geant4

Geant4 is a simulation toolkit that provides a platform for simulating the passage of particles through matter (Geant4 2003, 2006, 2016). Geant4, short for "Geometry and Tracking," offers a comprehensive framework for conducting such simulations.

The simulation process begins with establishing the environment, which involves defining the geometry of the calorimeter. This step specifies the shapes, sizes, and arrangement of the detector components. Next, the detector materials are defined, typically requiring the selection of both active materials, such as plastic scintillators, and passive materials.

A critical step in the process is defining the list of physical processes to be simulated. This list determines the types of interactions that will be modeled within the environment. Central to any simulation is the primary particle, which requires the configuration of a primary particle generator. This generator can source particles from an external simulator.

Finally, the simulation is executed. This involves configuring specific actions that influence various stages of the run, including individual events and the stepping mechanism that dictates how the simulation advances step by step.

4.4.1 *Geometry*

In Geant4, geometry is defined through a hierarchical, component-based approach, allowing for efficient and flexible design of detector setups. Complex structures are broken down into smaller, reusable components. For example, a single detector module can be defined once and then instantiated multiple times in different locations within a broader experimental setup.

Geant4 differentiates between *logical volumes* and *physical volumes* in its geometry definitions. A logical volume specifies the material properties and shape of a given component, serving as a blueprint. In contrast, the physical volume determines the exact position and orientation of the component within the overall geometry. This distinction allows multiple physical volumes to share the same logical volume definition but differ in placement and orientation.

The organization of volumes in Geant4 follows a hierarchical structure, where a volume may contain other volumes. The outermost volume is referred to as the *mother volume*, while the volumes it contains are called *daughter volumes*. The positioning of daughter volumes relative to their mother volume simplifies the placement of detector components in the simulation.

To enhance the efficiency of tracking particles through complex geometries, Geant4 employs voxelization techniques. This involves dividing the mother volume into smaller regions, or *voxels*, which reduces the computational complexity of navigating large volumes.

The complete geometry is enclosed within a *world volume*, which serves as the outer boundary of the detector setup. All other volumes are nested within this world volume. The Geant4 coordinate system is right-handed, with its origin typically located at the center of the world volume. This configuration ensures consistency and precision when defining and navigating the geometry throughout the simulation.

4.4.2 *Materials*

In the Geant4 simulation environment, the properties of materials dictate how particles interact during the simulation. Each material in Geant4 is defined based on its constituent chemical elements, which are characterized by their atomic number Z and atomic mass A. A material may consist of one or more elements or molecules, and is described by several key properties, including its density, state (solid, liquid, or gas), and the proportions of its constituent elements.

Materials in Geant4 can be defined as pure elements, such as lead or iron, or as compound materials, like water (H_2O). In the case of compound materials, the components are mixed in specific proportions, although they are not chemically bound together. Additionally, Geant4 provides access to a comprehensive database of predefined materials, utilizing data from the National Institute of Standards and Technology (NIST) database, which simplifies material definition.

In Geant4, the density of a material is expressed in grams per cubic centimeter (g/cm^3), and it plays a crucial role in determining how particles interact with the material, influencing

processes such as scattering and absorption. Furthermore, the temperature and pressure of a material can be specified. Geant4 also allows for the definition of isotopes, which is beneficial for simulating materials that are isotopically enriched.

4.4.3 *The Sensitive Detector*

In the Geant4 framework, the term *sensitive detectors* refers to components that capture and record detailed information about particle interactions within specified materials during a simulation. While calorimeter literature typically refers to such components as *active materials*, Geant4 defines them as sensitive detectors, integral to detecting hits, energy deposition, and other relevant quantities. The processing of data from these detectors can be customized to meet the specific goals of the simulation.

Sensitive detectors are linked to *logical volumes*, representing the materials within the simulation. This association enables the detection of interactions occurring within defined volumes. During each simulation step, information about particles—such as position, energy, and momentum within the sensitive volume—can be extracted and recorded.

The primary data collected by sensitive detectors are known as *hits*. A hit corresponds to a single interaction event within the sensitive detector and provides critical information such as the location and energy deposited by a particle at a specific point within the detector.

In Geant4, *scoring* refers to the process of collecting, analyzing, and consolidating data from sensitive detectors. Scoring allows for calculating various metrics, including total energy deposition, the number of hits, track length, and other relevant parameters.

4.4.4 *Physics Lists*

In Geant4, the *physics list* defines the set of physical processes that particles undergo during a simulation. It specifies how particles interact with materials, the types of interactions that can occur, and how these interactions are modeled within the simulation environment. Essentially, the physics list provides instructions to Geant4 on which physical processes to apply as particles traverse different materials, encompassing interactions such as electromagnetic scattering, ionization, hadronic interactions, decay processes, and more.

The selection of physical processes and models in the physics list significantly influences both the accuracy and computational efficiency of the simulation. A well-chosen physics list strikes a balance between these factors, tailored to the specific requirements of the simulation. The physics list starts by defining particles such as electrons, photons, protons, neutrons, and others. For each particle, the list outlines a set of processes that describe how it interacts with matter. These processes are typically categorized into several types, including electromagnetic processes, hadronic processes, decay processes, and optical processes.

The choice of models for a given process depends on factors such as the energy range and type of interaction. For instance, in hadronic physics, different models are used for low-energy

interactions, like neutron thermal scattering, compared to high-energy interactions, such as those involving quark-gluon plasma formation.

Geant4 offers a variety of predefined modular physics lists that cover a broad range of applications, catering to diverse simulations in fields such as high-energy physics and medical physics. For more specialized use cases, users have the option to create custom physics lists, allowing for precise control over the specific processes and models included. This customization capability is especially valuable in non-standard cases, enabling an optimal balance between computational performance and physical accuracy.

4.4.5 *Primary Particle Generators*

In Geant4, *Primary Particle Generators* define how a simulation begins by setting the initial conditions for the particles introduced into the simulation environment. The primary particle generator allows users to specify various attributes of these particles, including their type, energy, position, and direction.

4.4.6 *The Structure of a Simulation*

To illustrate the simulation process in Geant4, consider a sampling calorimeter setup, where an electron traverses air and passes through alternating layers of passive material (lead) and active material (scintillator), a common calorimeter configuration.

At the start of the event, Geant4 initializes the simulation by creating data structures for storing information about tracks, hits, and interactions. The primary particle generator produces an electron with specified initial conditions such as energy, position, and direction.

The electron is then converted into a *track*, representing its path through the detector. This track is divided into *steps*, with each step corresponding to the distance traveled between interaction points or boundaries between materials.

As the electron moves through the air, it follows a straight path unless interactions occur. The interactions are determined based on the cross sections for the particle's specific conditions, such as energy or type. As the electron progresses, it may ionize air molecules, generating secondary electrons (delta rays) or emit bremsstrahlung photons if decelerating in the electric field of the molecules, resulting in energy loss.

For each secondary particle generated, a new track is created. Secondary particles are added to a stack for further processing during the event. All tracks are processed similarly to the primary electron. After each step, Geant4 assesses whether the electron or any secondary particle should continue based on remaining energy and the distance to the next interaction point or boundary.

Upon reaching the lead layer, the system detects the boundary crossing and updates the material properties associated with the track. The electron interacts with the denser lead, undergoing multiple scattering and producing more secondary electrons and bremsstrahlung

photons due to lead's high atomic number. These secondary particles are assigned new tracks and propagated accordingly. If bremsstrahlung photons are of sufficiently high energy, they may undergo pair production, creating electron-positron pairs, which are tracked as well.

Once the electron enters the scintillator layer, Geant4 identifies the material change. In the scintillator, the electron excites and ionizes molecules, emitting scintillation light. This light is detected by the sensitive detector, which records the energy deposition as a *hit*.

The electron continues to traverse alternating lead and scintillator layers, losing energy through ionization and bremsstrahlung in the lead, while depositing energy in the scintillator. Additional secondary particles, such as delta rays and bremsstrahlung photons, are created and tracked independently.

As the electron's energy decreases and falls below a predefined threshold, the track is terminated, and no further steps are processed.

At the end of the event, Geant4 gathers all hits recorded by the sensitive detector. These hits include data such as energy deposition and time of occurrence. The total energy deposited in the scintillator layers is then calculated, providing a measure of the electron's total energy loss as it passed through the material. The simulation results are subsequently saved or used for further analysis.

MACHINE LEARNING

5

Traditionally, in the field of *Machine Learning* (ML), individual features within a dataset were manually crafted, requiring domain expertise and bespoke feature design tailored to each specific application. However, the advent of deep learning, a subset of ML, has revolutionized the handling of complex, high-dimensional data. Unlike traditional methods, deep learning represents a form of *representation learning*, where machines are presented with raw data and autonomously tasked with discovering the necessary features for a given task. This is achieved through the composition of non-linear layers, which progressively refine the representations into higher levels of abstraction. This approach enables machines to learn from unstructured, low-level data and extract discriminative, high-level features. Deep learning models, particularly those based on *Neural Network* (NN)s, have demonstrated remarkable success across various tasks, often surpassing traditional ML algorithms (LeCun et al. 2015).

5.1 Density Estimation

Many problems in ML can be understood as alignment problems, where the objective is to infer a distribution that is unknown *a priori* and for which the density cannot be directly evaluated. The task is to approximate this distribution using a machine learning model, essentially aligning the model distribution with the true distribution inferred from data samples.

Several common scenarios can be framed as alignment problems. For instance, in *regression*, the goal is typically to infer a one-dimensional distribution. In *classification*, the aim is to infer the parameters of a binary distribution. *Generation* tasks, on the other hand, involve modeling complex, multi-dimensional distributions and generating new samples from this learned distribution.

In each of these cases, a set of samples $(x_i, y_i)_{i=1}^n$ is provided, where y_i are instances of a random variable Y that follows an unknown distribution $p_y^\star\big(y|x\big)$, conditioned on a variable X. The objective is to model this distribution using the model's distribution $p_y\big(y|x, \boldsymbol{\theta}\big)$, which is conditioned on X and the model's parameters $\boldsymbol{\theta}$. In ML terminology, x_i refers to the data points, and y_i represents the labels. The primary task is to infer the label distribution given the input data.

5.1.1 *KL Divergence and Maximum Likelihood*

When viewed as a distribution alignment problem, the objective is to minimize the difference between the true data distribution, $p_y^\star\big(y|x\big)$, and the model distribution, $p_y\big(y|x, \boldsymbol{\theta}\big)$. A widely

used measure of the divergence between two distributions is the `glsKL`:

$$D_{\mathrm{KL}}\left(p_y^\star(y|x)\middle\|p_y(y|x,\boldsymbol{\theta})\right) = \int_Y p_y^\star(y|x)\log\frac{p_y^\star(y|x)}{p_y(y|x,\boldsymbol{\theta})}\,\mathrm{d}y\,. \tag{5.1}$$

It is important to note that the *Kullback-Leibler* (KL) *Divergence* divergence is asymmetric, meaning that $D_{\mathrm{KL}}\left(p_y^\star(y|x)\middle\|p_y(y|x,\boldsymbol{\theta})\right) \neq D_{\mathrm{KL}}\left(p_y(y|x,\boldsymbol{\theta})\middle\|p_y^\star(y|x)\right)$. Minimizing the KL divergence divergence with respect to the model's parameters, $\boldsymbol{\theta}$, is mathematically equivalent to maximizing the likelihood of the observed data:

$$\begin{aligned}
&\min_{\boldsymbol{\theta}} D_{\mathrm{KL}}\left(p_y^\star(y|x)\middle\|p_y(y|x,\boldsymbol{\theta})\right)\\
&= \min_{\boldsymbol{\theta}} \int_Y p_y^\star(y|x)\log\frac{p_y^\star(y|x)}{p_y(y|x,\boldsymbol{\theta})}\,\mathrm{d}y\\
&= \min_{\boldsymbol{\theta}}! \left[\int_Y p_y^\star(y|x)\log p_y^\star(y|x)\,\mathrm{d}y - \int_Y p_y^\star(y|x)\log p_y(y|x,\boldsymbol{\theta})\,\mathrm{d}y\right]\\
&= \min_{\boldsymbol{\theta}}! \left[\underbrace{\left\langle\log p_y^\star(y|x)\right\rangle_{p_y^\star}}_{\text{constant}} - \left\langle\log p_y(y|x,\boldsymbol{\theta})\right\rangle_{p_y^\star}\right]\\
&= \max_{\boldsymbol{\theta}} \left\langle\log p_y(y|x,\boldsymbol{\theta})\right\rangle_{p_y^\star}.
\end{aligned} \tag{5.2}$$

Since the true data density $p_y^\star(y|x)$ and its expectation value are not directly accessible, it becomes necessary to approximate the expectation of $p_y(y|x,\boldsymbol{\theta})$. Given a sample set $(x_i, y_i)_{i=1}^n$ from the target distribution $p_y^\star(y|x)$, the expectation value can be approximated as:

$$\left\langle p_y(y|x,\boldsymbol{\theta})\right\rangle_{p_y^\star} \approx \frac{1}{n}\sum_{i=1}^n p_y(y_i|x_i,\boldsymbol{\theta})\,. \tag{5.3}$$

5.1.2 *Mean Squared Error*

In the context of regression tasks, it is often assumed that the labels follow a locally Gaussian distribution with a constant variance σ^2. The task of the model is to predict the expected mean parameter $\mu(x,\boldsymbol{\theta})$ of this normal distribution. Minimizing the KL divergence diver-

gence for this problem leads to the following derivation:

$$\begin{aligned}
&\min_{\boldsymbol{\theta}} D_{\mathrm{KL}}\left(p_{\hat{y}}^{\star}\left(y|x\right)\middle\|p_y\left(y|x,\boldsymbol{\theta}\right)\right)\\
&=\max_{\boldsymbol{\theta}}\left\langle\log p_y\left(y|x,\boldsymbol{\theta}\right)\right\rangle_{p_{\hat{y}}^{\star}}\\
&=\max_{\boldsymbol{\theta}}\left\langle\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\frac{(y-\mu(x,\boldsymbol{\theta}))^2}{\sigma^2}\right)\right)\right\rangle_{p_{\hat{y}}^{\star}}\\
&=\max_{\boldsymbol{\theta}}\left\langle-\log\left(\sqrt{2\pi\sigma^2}\right)-\frac{1}{2}\frac{(y-\mu(x,\boldsymbol{\theta}))^2}{\sigma^2}\right\rangle_{p_{\hat{y}}^{\star}}\\
&=\min_{\boldsymbol{\theta}}\left\langle(y-\mu(x,\boldsymbol{\theta}))^2\right\rangle_{p_{\hat{y}}^{\star}}\\
&\approx\min_{\boldsymbol{\theta}}\frac{1}{n}\sum_{i=1}^{n}(y_i-\mu(x_i,\boldsymbol{\theta}))^2.
\end{aligned}\tag{5.4}$$

This result is known as the *Mean Squared Error* (MSE), a widely-used loss function in machine learning. By minimizing the MSE, the assumption is made that the underlying data is locally normally distributed with a constant variance σ^2.

5.1.3 *Binary Cross-Entropy*

In a basic binary classification task, the possible outcomes are typically represented as $y \in \{0,1\}$. As a result, the output distribution $p_y\left(y|x,\boldsymbol{\theta}\right)$ must follow a Bernoulli distribution. This distribution is characterized by a single parameter $\rho(x,\boldsymbol{\theta})$, which is inferred from the data x and the model's parameters $\boldsymbol{\theta}$:

$$p_y\left(y|x,\boldsymbol{\theta}\right)=\begin{cases}1-\rho(x,\boldsymbol{\theta}) & \text{for } y=0\\ \rho(x,\boldsymbol{\theta}) & \text{for } y=1\end{cases}\tag{5.5}$$

As discussed previously, minimizing the KL divergence divergence is equivalent to maximizing the log-likelihood of the model's predictions, which further translates into minimizing the binary cross-entropy loss function:

$$\begin{aligned}
\left\langle\log p_y\left(y|x,\boldsymbol{\theta}\right)\right\rangle_{p_{\hat{y}}^{\star}}&\approx\frac{1}{n}\sum_{i=1}^{n}\log p_y\left(y|x,\boldsymbol{\theta}\right)\\
&=\frac{1}{n}\sum_{i=1}^{n}\left[y_i\log p_x\left(y_i=1|x,\boldsymbol{\theta}\right)+(1-y_i)\log p_x\left(y_i=0|x,\boldsymbol{\theta}\right)\right]\\
&=\frac{1}{n}\sum_{i=1}^{n}\left[y_i\log\left(\rho(x,\boldsymbol{\theta})\right)+(1-y_i)\log\left(1-\rho(x,\boldsymbol{\theta})\right)\right].
\end{aligned}\tag{5.6}$$

Thus, binary cross-entropy serves as an effective and widely-used loss function for binary classification tasks.

In the case of classification tasks involving multiple classes, the output must represent the probabilities of each class. This is achieved using the *softmax* function:

$$\sigma(\mathbf{x}) = \frac{\exp(\mathbf{x})}{\sum_i \exp(x_i)}, \tag{5.7}$$

which ensures that the outputs, which can range from $-\infty$ to ∞, are normalized to be non-negative and sum to 1, thereby producing valid probability distributions.

5.1.4 *The General Case*

The commonly used loss functions—MSE and binary cross-entropy—are both derived from the KL divergence divergence, assuming specific target distributions. Similarly, other loss functions can be formulated by tailoring the assumptions to different types of distributions. In a more general scenario, $p_y^\star\left(y\middle|x\right)$ may represent a complex multidimensional distribution, where the dimensions of Y are interdependent and exhibit correlations. Addressing this added complexity necessitates more sophisticated modeling techniques. Further exploration of this issue will be presented in Chapter 6, which discusses approaches to effectively handle such intricate distributions.

The next section will outline suitable architectures designed to address these tasks.

5.2 Multilayer Perceptron

In ML, the primary objective is to construct a predictor F that maps an input X to an output Y, which is represented as:

$$F : X \to Y. \tag{5.8}$$

There are various techniques to create such predictors. In the context of deep learning, this multivariate function, denoted as $\hat{Y}(X)$, is constructed through a layered composition of affine transformations and non-linear functions. For each layer l, let $\sigma^{[1]}, \ldots, \sigma^{[L]}$ denote the non-linear activation functions applied at every layer. The affine transformation at layer l can be expressed as:

$$\sigma_{W,b}^{[l]} : \mathbf{a} \mapsto \sigma^{[l]}\left(W^{[l]}\mathbf{a} + \mathbf{b}^{[l]}\right), \tag{5.9}$$

where $W^{[l]}$ and $b^{[l]}$ represent the weight matrix and bias for the l-th layer, respectively. This defines a Deep Predictor as a composite mapping, formally expressed as:

$$\hat{Y} : \mathbf{x} \mapsto \left(\sigma_{W,b}^{[L]} \circ \cdots \circ \sigma_{W,b}^{[1]}\right)(\mathbf{x}). \tag{5.10}$$

The structure of the model is captured through a computational graph. For the i-th node in the l-th layer, the following computations are performed:

$$z_i^{[l]} := \sum_{j=1}^{N^{[l]}} W_{ij}^{[l]} a_j^{[l-1]} + b_i^{[l]}, \tag{5.11}$$

$$\begin{aligned} a_i^{[l]} &:= \sigma_i^{[l]}\left(z_i^{[l]}\right) \\ &= \sigma_i^{[l]}\left(\sum_{j=1}^{N^{[l]}} W_{ij}^{[l]} a_j^{[l-1]} + b_i^{[l]}\right), \end{aligned} \tag{5.12}$$

$$\mathbf{a}^{[0]} := \mathbf{x}, \tag{5.13}$$

$$\hat{Y}(\mathbf{x}) := \mathbf{a}^{[L]}, \tag{5.14}$$

where $N^{[l]}$ is the number of nodes in the l-th layer. The final output $\hat{Y}(\mathbf{x})$ is computed after passing through all layers.

5.3 Backpropagation

Backpropagation is a technique used to optimize NNs, enabling precise adjustment of model parameters to improve prediction accuracy. The objective of a NN is to minimize the function:

$$\mathscr{L} : \left(\hat{Y}(X), Y\right) \mapsto [0, \infty), \tag{5.15}$$

which measures the difference between the predicted output $\hat{Y}(X)$ and the actual output Y. Optimization seeks to minimize $\mathscr{L}$ by adjusting the model's parameters, denoted by $\boldsymbol{\theta}$, moving through the parameter space toward the minimum of $\mathscr{L}$. The first-order Taylor expansion of the loss function provides an approximation:

$$\mathscr{L}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \approx \mathscr{L}(\boldsymbol{\theta}) + \frac{\partial \mathscr{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \Delta\boldsymbol{\theta}. \tag{5.16}$$

For small values of $\Delta\boldsymbol{\theta}$, the leading-order term suffices. The optimal direction of parameter update occurs when $\Delta\boldsymbol{\theta}$ is antiparallel to the gradient of the loss function, $\partial\mathscr{L}/\partial\boldsymbol{\theta}$. The parameter update rule can therefore be expressed as:

$$\boldsymbol{\theta} \to \boldsymbol{\theta} - \eta \frac{\partial \mathscr{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \tag{5.17}$$

where η is a small positive scalar known as the learning rate, typically treated as a hyperparameter.

The parameters $\boldsymbol{\theta}$ are iteratively updated until a predefined minimization criterion is met. This approach, based on gradient computations, is commonly referred to as *Steepest Descent* or *Gradient Descent* (Cauchy 1847).

Within NN, backpropagation applies by calculating the chain of derivatives for layer l as:

$$\delta_j^{[l]} := \frac{\partial \mathscr{L}}{\partial z_j^{[l]}}. \tag{5.18}$$

where $z_j^{[l]}$ is the input to the activation function at node j in layer l. The derivatives of the loss function with respect to the weights and biases can then be calculated as:

$$\frac{\partial \mathscr{L}}{\partial W_{jk}^{[l]}} = \delta_j^{[l]} a_k^{[l-1]}, \tag{5.19}$$

$$\frac{\partial \mathscr{L}}{\partial b_j^{[l]}} = \delta_j^{[l]}. \tag{5.20}$$

$$\delta_j^{[l]} = \frac{\partial \mathscr{L}}{\partial z_j^{[l]}} = \sum_{k=1}^{N^{[l+1]}} \frac{\partial \mathscr{L}}{\partial z_k^{[l+1]}} \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} = \sum_{k=1}^{N^{[l+1]}} \delta_k^{[l+1]} \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}}. \tag{5.21}$$

Given that $z_k^{[l+1]}$ is dependent on $z_j^{[l]}$, the relationship is:

$$\begin{aligned} z_k^{[l+1]} &= \sum_{s=1}^{N^{[l]}} W_{ks}^{[l+1]} \sigma(z_s^{[l]}) + b_k^{[l+1]}, \\ \Rightarrow \quad \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} &= W_{kj}^{[l+1]} \sigma_i^{[l]}(z_j^{[l]}). \end{aligned} \tag{5.22}$$

Thus, the expression for $\delta_j^{[l]}$ becomes:

$$\delta_j^{[l]} = \sum_{k=1}^{N^{[l+1]}} \delta_k^{[l+1]} W_{kj}^{[l+1]} \sigma_l^{[l]}(z_j^{[l]}). \tag{5.23}$$

As indicated by (5.12), the computation of $a^{[l]}$ depends on the values of $a^{[l-1]}$. Therefore, the computation of the entire predictor proceeds in a *forward pass* through the network. The gradient for layer l requires the gradient from layer $l+1$, meaning that a *backward pass* through the computation graph is needed to propagate errors backward. This forms the core of the backpropagation algorithm, which utilizes the chain rule of calculus to efficiently compute gradients. Importantly, this method ensures that computing the gradients is no more computationally expensive than the forward pass.

The concept of minimizing error via gradient descent predates NNs and dates back to the 1960s (Amari 1967; Bryson 1961; A. Bryson & Ho 1969; A. E. Bryson & Denham 1961; Dreyfus 1962; Kelley 1960; Pontryagin et al. 1961; Wilkinson 1965). These algorithms were recognized for their efficiency, as deriving the derivatives was no more costly than computing the forward evolution of the system (Schmidhuber 2015). Efficient error backpropagation for arbitrary networks was introduced by Linnainmaa (1970, 1976), and Werbos 1981 first applied these ideas to Neural Networks. The potential of backpropagation to produce meaningful internal representations in deeper network layers was later demonstrated by Rumelhart et al. 1986.

5.4 Optimization Algorithms

Several algorithms have been developed for optimizing NNs. Gradient Descent, previously discussed, updates parameters using the gradient of the loss function across the entire dataset.

While effective, this method can be computationally expensive and impractical for large datasets. As an alternative, *Stochastic Gradient Descent* (SGD) and its variations are widely used in deep learning. SGD approximates the full gradient by computing it on a *mini-batch*, a randomly selected subset of the dataset. This reduces computational overhead and increases robustness against local minima, though it requires more steps to reach convergence.

One notable variation is *SGD with momentum* (Rumelhart et al. 1986), which introduces an *exponentially weighted moving average* of gradients. This method incorporates not only the current gradient but also prior updates, adding inertia to overcome local minima more effectively. The update rule for parameters $\boldsymbol{\theta}_t$ in SGD with momentum is expressed as:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \eta \mathbf{g}_{t+1}, \\ \mathbf{g}_{t+1} &= \beta \mathbf{g}_t + (1-\beta) \frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}_t}, \end{aligned} \tag{5.24}$$

where $0 < \beta < 1$ is a hyperparameter controlling the momentum's influence.

To address varying gradient magnitudes across different parameters, *RMSprop*, introduced by Hinton et al. 2014, dynamically adjusts learning rates based on a moving average of the squared gradients. The update rule for RMSprop is:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\left\langle \mathbf{g}_{t+1}^2 \right\rangle + \epsilon}} \frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}_t}, \\ \left\langle \mathbf{g}_{t+1}^2 \right\rangle &= \gamma \left\langle \mathbf{g}_t^2 \right\rangle + (1-\gamma) \left(\frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}_t} \right)^2, \end{aligned} \tag{5.25}$$

where $0 < \gamma < 1$ controls the moving average.

Adagrad (Duchi et al. 2011) offers a distinct approach, adjusting the learning rate individually for each parameter based on its historical squared gradient. This method is particularly well-suited for sparse data and gradients.

Adaptive Moment Estimation (Adam)(Kingma & Ba 2014) combines the benefits of RMSprop and SGD with momentum. By using bias-corrected estimates of the first and second moments of gradients, Adam adaptively adjusts learning rates. The update rule for Adam is:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \eta \frac{\hat{\mathbf{m}}_{t+1}}{\sqrt{\hat{\mathbf{v}}_{t+1}} + \epsilon}, \\ \hat{\mathbf{m}}_{t+1} &= \frac{\mathbf{m}_{t+1}}{1-\beta_1}, \\ \hat{\mathbf{v}}_{t+1} &= \frac{\mathbf{v}_t}{1-\beta_2}, \\ \mathbf{m}_{t+1} &= \beta_1 \mathbf{m}_t + (1-\beta_1) \frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}_t}, \\ \mathbf{v}_{t+1} &= \beta_2 \mathbf{v}_t + (1-\beta_2) \left(\frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}_t} \right)^2, \end{aligned} \tag{5.26}$$

where $0 < \beta_1, \beta_2 < 1$ are hyperparameters that control the decay rates of the first and second moment estimates.

5.5 Convolutional Neural Networks

Convolutional Neural Network (CNN)s are a specialized category of NNs designed to process data with a grid-like topology, such as images. The key concept behind CNNs is their ability to recognize spatial hierarchies of features, which makes them particularly effective in tasks involving images, where the arrangement of features, like edges or textures in images, is essential for effective pattern recognition. By focusing on the spatial relationships within the data, CNNs can identify patterns that might not be apparent when features are examined in isolation.

The mathematical foundation of CNNs lies in the convolution operation between two functions. Given an input function $x(t)$ and a kernel $w(t)$, the convolution operation is defined as:

$$(x * w)(t) = \int x(a)w(t-a)\,\mathrm{d}a\,. \tag{5.27}$$

In this context, a *kernel* refers to a small matrix of weights that moves across the input data, performing the convolution to detect patterns within the data. The specific features identified by the kernel, such as edges or textures, are determined by its size and the values of its weights. When CNNs operate on discrete inputs, such as digital images, the convolution transforms into a summation:

$$(x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \tag{5.28}$$

For multidimensional data, such as images, the convolution is applied across all dimensions of the input. If $I(i,j)$ represents a two-dimensional image and $K(m,n)$ denotes the convolutional kernel (filter), the two-dimensional convolution is expressed as:

$$(K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n). \tag{5.29}$$

A notable feature of CNNs is *sparse connectivity*, achieved by applying small kernels instead of fully connected layers. This reduces computational complexity and memory requirements. Additionally, CNNs utilize *weight sharing*, where the same kernel is applied across different regions of the input, which introduces *translational invariance*. This ensures that the model can detect patterns irrespective of their position in the input space.

In a convolutional layer, multiple kernels are applied to the input data, and the outputs are passed through a non-linear activation function. Afterward, *pooling* operations are used to reduce the spatial dimensions of the data while preserving critical information. Pooling is commonly performed using *max-pooling*, which selects the maximum value in a region, or *average-pooling*, which computes the average of the values within a region. These operations reduce the size of the feature maps, enhance the model's robustness to minor variations in the input, and help prevent overfitting by lowering the number of parameters and computations.

The origins of CNNs can be traced back to Hubel and Wiesel 1959, who discovered neurons in the visual cortex that respond to specific regions in the visual field, leading to the hierarchical processing of visual stimuli. This inspired the development of CNNs. The introduction of the Neocognitron by Fukushima 1980 formalized these principles into a computational model for hierarchical feature detection.

Advances in CNNs were further propelled by the adoption of backpropagation for training. Waibel et al. applied this technique in their Time Delay Neural Network for speech recognition (Hampshire & Waibel 1989; Waibel et al. 1990). LeCun et al. 1989 demonstrated the effectiveness of backpropagation-trained CNNs for tasks such as handwritten digit recognition, highlighting the versatility and potential of CNNs for a wide range of pattern recognition tasks.

5.6 Symmetries in Machine Learning

Incorporating prior knowledge of a problem's structure is a fundamental strategy in designing effective ML algorithms. This prior knowledge often manifests as *Symmetries* within the problem's representation, which can reduce the architectural complexity and improve the model's generalization performance (Cohen & Welling 2016). The earlier discussion on CNNs offers a clear example of this principle. CNNs exploit the translation symmetry inherent in image data. By aligning a NN's architecture with these underlying symmetries, the learning process is greatly simplified, enhancing both model efficiency and performance.

Symmetries can be incorporated into NNs by ensuring the network respects certain invariances or equivariances. To explain these concepts, consider a function $f : X \to Y$ and a symmetry group g, where T_X and T_Y represent the actions induced by g on the spaces X and Y, respectively. A function is considered invariant under the action of g if:

$$f(T_X x) = f(x) \quad \forall\, x \in X, \tag{5.30}$$

meaning that the application of a symmetry transformation does not alter the function's output. This concept is visualized in Figure 5.1.

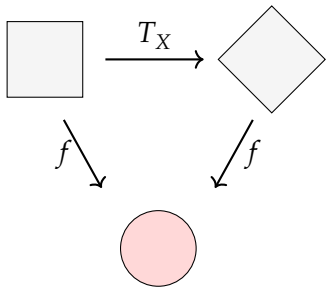

Figure 5.1: Illustration of Invariance of function f under the group g.

Conversely, a function exhibits *Equivariance* to g if the application of the group actions commutes with the function:

$$f(T_X x) = T_Y f(x) \quad \forall\, x \in X \text{ and } \forall\, f(x) \in Y. \tag{5.31}$$

In this case, changing the order of applying g and f does not alter the outcome. When $X = Y$, this implies that $T_X = T_Y$. This concept is illustrated in Figure 5.2.

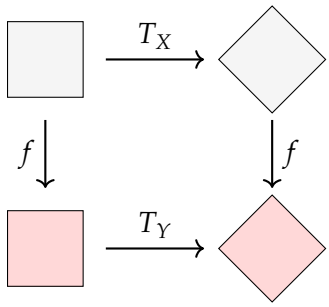

Figure 5.2: Illustration of equivariance of function f under the group g.

5.7 Point Clouds

In various domains of physics, point clouds provide an effective means of representing data, particularly when the data is best described as a collection of discrete points in space. These representations capture both the spatial distribution and properties of physical phenomena. This is especially relevant for the chapters that follow, where datasets are represented as point clouds. In this study, calorimeter showers are modeled as point clouds rather than voxel-based data, marking a significant methodological shift.

Point clouds consist of collections of points, each potentially carrying additional attributes such as color, intensity, or vector direction. Importantly, these collections lack any inherent order, making them *permutation invariant*. In other words, the representation of a point cloud remains unchanged under any reordering of the points.

Mathematically, point clouds are treated as sets. A set can be formally defined as $X = \{x_i\}_{i=1}^n$, where each $x_i \in \mathfrak{X}$ represents a unique element within X, and $n \in \mathbb{N}_+$ is the cardinality of the point cloud. For this study, uniqueness is not a critical concern and will not be further considered. A function $f : \mathfrak{X}^n \to Y$ is considered permutation invariant if, for any permutation π, it satisfies:

$$f([x_1, x_2, \dots, x_n]) = f\left([x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)}]\right), \tag{5.32}$$

where the elements of $\mathfrak{X}^n$ are represented as tuples. A function $f : \mathfrak{X}^n \to \mathfrak{Y}^n$ on a set is considered permutation equivariant if the permutation commutes with f,

$$f([x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)}]) = [f_{\pi(1)}(x), f_{\pi(2)}(x), \dots, f_{\pi(n)}(x)]. \tag{5.33}$$

Functions acting on point clouds must also handle varying cardinalities since point clouds can differ in size. Leveraging point clouds in machine learning requires NN architectures capable of processing these unordered and structurally flexible datasets.

One useful way to conceptualize point clouds is to imagine them as graphs where each point is treated as a node. Initially, these graph have no predefined edges, implying there are no predetermined connections between nodes. As a result, each element of the set must be treated as independent. The absence of predefined edges suggests that the relationships between nodes must be learned. This approach aligns with the perspective articulated by Bronstein et al. 2021. The subsequent section will explore two approaches to learning these relationships.

5.8 Deep Sets

Deep Sets (Zaheer et al. 2017) introduced one of the first NN architectures specifically designed to handle sets by incorporating a permutation-invariant structure. This architecture effectively addresses the challenge of processing unordered data through a two-step approach.

In the first step, a Neural Network ϕ independently maps each element of the input set X into a higher-dimensional latent space. This transformation is applied element-wise, ensuring the process is independent of the order of the elements. After this transformation, a permutation-invariant pooling operation, such as summation, mean, or max-pooling, is applied across the transformed elements. This step reduces the set of high-dimensional representations into a single aggregated form. Formally, this process can be expressed as:

$$X \mapsto \rho\left(\bigoplus_{x \in X} \phi(x)\right), \tag{5.34}$$

where $\bigoplus$ denotes the pooling operation and ρ represents the transformation applied to the pooled output. The result is a compressed representation of the aggregated information, suitable for subsequent tasks such as classification or regression.

From a graph-theoretical viewpoint, the Deep Sets architecture can be interpreted as aggregating the information from all nodes into a single induced node, as illustrated in Figure 5.3.

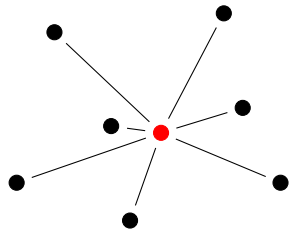

Figure 5.3: Illustration of Deep Sets from a graph perspective. The black nodes represent the points in the point cloud, while the red node represents the introduced information aggregation node.

This architecture can be extended to achieve permutation equivariance by transforming each point in the set into a linear combination of the original point and the aggregated information from the set:

$$x_i \mapsto \beta x_i + \lambda \rho\left(\bigoplus_{x \in X} \phi(x)\right), \tag{5.35}$$

where $\beta, \lambda \in \mathbb{R}$. A similar method was proposed by Buhmann, Kasieczka, and Thaler 2023, who applied transformations to all points in the set by concatenating the aggregated information with the original data.

5.9 Attention

The concept of *Attention* was introduced by Vaswani et al. 2017 in their seminal paper, *Attention is All You Need*, which presented the *Transformer* architecture. This architecture centers around the Attention mechanism, which has become pivotal in various NN models.

Originally designed for *Natural Language Processing* (NLP), the Transformer model was built as a sequence-to-sequence model trained to predict subsequent elements in a sequence. Since the Attention mechanism is permutation invariant, masking was introduced to prevent information flow from later elements to earlier ones. In NLP, words are embedded into tokens, though token embedding is not directly relevant to point clouds and will not be covered here.

The following description of Transformers and Attention is inspired by the work of J. Lee et al. 2019 on *Set Transformers*, which explores how transformers can handle unordered collections, such as point clouds.

In the graph view of point clouds, previously introduced, point clouds are treated as collections of nodes without predefined edges. In contrast, attention introduces a complete edge set, connecting each node to all others. This setup allows the model to represent all relationships between nodes without requiring prior knowledge of these relationships. Specifically, this architecture is known as *Self-Attention*, in which all points in a cloud X are connected to each other. This differs from *Cross-Attention*, where the relationship between two different clouds X and Y is modeled.

Attention is computed by assigning learned weights to the edges between nodes. These weights are calculated after mapping the point clouds into high-dimensional spaces known as embeddings. Three types of embeddings—*Query* (Q), *Key* (K), and *Value* (V)—are calculated as follows:

$$\begin{array}{llll} Q = [q_i] \in \mathbb{R}^{n\times d} & q_i = W_Q x_i & x_i \in \mathbb{R}^f & W_Q \in \mathbb{R}^{d\times f} \\ K = [k_i] \in \mathbb{R}^{m\times d} & k_i = W_K y_i & y_i \in \mathbb{R}^g & W_K \in \mathbb{R}^{d\times g} \\ V = [v_i] \in \mathbb{R}^{m\times b} & v_i = W_V y_i & & W_V \in \mathbb{R}^{b\times g} \end{array}$$

Here, W_Q, W_K, and W_V represent learned weight matrices, while x_i and y_i are individual points in the point clouds X and Y. These embeddings facilitate the computation of attention weights that define the relationships between different points.

Attention is then computed as follows:

$$\mathrm{Att}(Q, K, V, \omega) = \omega\left(QK^T\right)V, \tag{5.36}$$

where ω is an activation function. First, the dot product of all values q_i in Q and k_j in K is calculated, which measures the alignment of the points in the latent space. This process is referred to as "values attending to each other." The resulting alignment matrix has dimensions of $n \times m$. When multiplied by V, this produces a new point cloud with the cardinality of Q and the dimensionality of V, where each point is a linear combination of V with mixing factors from $\omega(QK^T)$.

The activation function ω determines how much each vector in V contributes based on the alignment scores from QK^T. The softmax function is commonly used, as the alignment scores may vary widely, and softmax ensures non-negative contributions that sum to one. The scores are scaled by the square root of the embedding dimension d to adjust for variance:

$$\omega_i(x) = \sigma_i\left(\frac{x}{\sqrt{d}}\right) = \frac{\exp\left(x_i/\sqrt{d}\right)}{\sum_j \exp\left(x_j/\sqrt{d}\right)}.$$

Thus, if ω is the scaled softmax, the attention function becomes:

$$\mathrm{Att}(Q, K, V) = \sigma\left(\frac{QK^T}{\sqrt{d}}\right)V, \tag{5.37}$$

with the activation function ω omitted for simplicity.

The paper *Attention is All You Need* (Vaswani et al. 2017) highlights the use of multiple independent linear representations of the input, called *heads*, in multi-head attention. After applying attention, the results of the different heads are concatenated.

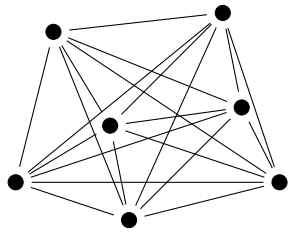

Figure 5.4: Illustration of Self-Attention. The black nodes represent the points in the point cloud. Each node shares its information with every other node.

Figure 5.4 visualizes Self-Attention in the graph view. Self-Attention is permutation equivariant because

$$\begin{aligned}
&\mathrm{Att}(PQ, PK, PV)\\
&= \sigma\left(\frac{PQ(PK)^T}{\sqrt{d}}\right)PV\\
&= \sigma\left(\frac{PQK^TP^T}{\sqrt{d}}\right)PV\\
&= P\sigma\left(\frac{QK^T}{\sqrt{d}}\right)P^TPV\\
&= P\sigma\left(\frac{QK^T}{\sqrt{d}}\right)V\\
&= P\ \mathrm{Att}(Q, K, V).
\end{aligned} \tag{5.38}$$

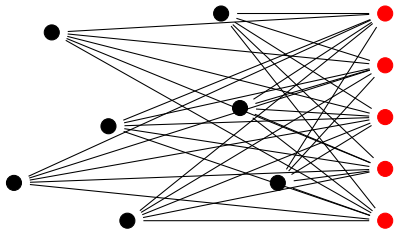

Figure 5.5: Illustration of Cross-Attention. The black nodes represent the points in the point cloud. Each node shares its information with every other node.

On the other hand, Cross-Attention, as visualized in Figure 5.5, is permutation-invariant, as demonstrated by

$$\begin{aligned}
&\text{Att}(Q, PK, PV) \\
&= \sigma\left(\frac{Q(PK)^T}{\sqrt{d}}\right)PV \\
&= \sigma\left(\frac{QK^TP^T}{\sqrt{d}}\right)PV \\
&= \sigma\left(\frac{QK^T}{\sqrt{d}}\right)P^TPV \\
&= \sigma\left(\frac{QK^T}{\sqrt{d}}\right)V \\
&= \text{Att}(Q, K, V).
\end{aligned} \tag{5.39}$$

J. Lee et al. 2019 demonstrated that attention can map point clouds of varying cardinalities to a fixed learned point cloud, potentially reducing the point cloud to a single point. In this extreme case, attention can be interpreted as a permutation-invariant pooling operation, similar to the Deep Sets architecture. If $Q = 0$, attention reduces to the following averaging operation:

$$\text{Att}(0, K, V) = \sigma\left(\frac{0 \cdot K^T}{\sqrt{d}}\right)V = \frac{\mathbb{1}}{n}V = \frac{1}{n}\sum_{i=1}^{n} v_i. \tag{5.40}$$

The *Induced Set Transformer* model introduced by J. Lee et al. 2019 approximates full attention by using cross-attention on a smaller learned point cloud, which is then mapped back to the original cloud.

GENERATIVE MODELS

6

6.1 Normalizing Flows

Building on the concepts introduced in Section 5.1, where specific cases like regression and classification were discussed, this section addresses the general case of density estimation using *Normalizing Flow* (NF)s. Here, the assumption of a locally Gaussian distribution is lifted, enabling a more flexible approach to mapping between complex, continuous distributions.

Without assuming that the random variable $X \in \mathbb{R}$ follows a local Gaussian distribution, consider the problem of regressing X. To address this, an auxiliary random variable U, uniformly distributed over the interval $[0,1]$, is introduced. By applying the *Cumulative Distribution Function* (CDF) $F_X(X)$ to X, the variable can be mapped to U. The inverse of the CDF, known as the quantile function $F_X^{-1}(U)$, then maps U back to X. A proof of this mapping can be found in Section A.1.

Given two random variables $X, Z \in \mathbb{R}$, an invertible mapping $f = F_X^{-1} \circ F_Z$ between any two continuous distributions can be established by first applying the CDF of Z, F_Z, and then the quantile function of X, F_X^{-1}. This requires access to both the cumulative distribution and quantile functions of the distributions.

The probability of an infinitesimally small interval is considered:

$$p(x_0 \leq x \leq x_0 + \mathrm{d}x) = p(x_0)\,\mathrm{d}x\,.$$

Applying the transformation $f^{-1}$and preserving the probability interval, leads to:

$$\begin{aligned} p(x_0)\,\mathrm{d}x &= p(x_0 \leq x \leq x_0 + \mathrm{d}x) \\ &= p\Big(f^{-1}(x_0) \leq z \leq f^{-1}(x_0 + \mathrm{d}x)\Big) \\ &= p\left(z_0 \leq y \leq z_0 + \left.\frac{\mathrm{d}f^{-1}}{\mathrm{d}x}\right|_{x=x_0} \mathrm{d}x\right) \quad \text{with } z_0 \coloneqq f^{-1}(x_0) \\ &= p(z_0)\left.\frac{\mathrm{d}f^{-1}}{\mathrm{d}x}\right|_{x=x_0} \mathrm{d}x\,. \end{aligned} \tag{6.1}$$

By expanding $f^{-1}(x_0 + \mathrm{d}x)$ using a Taylor series and neglecting higher-order terms, the general one-dimensional change of variable formula is obtained:

$$p(x) = p(z)\frac{\mathrm{d}f^{-1}}{\mathrm{d}x}.$$

This derivation, though illustrative, is a typical physicist's loose derivation meant for visualization, rather than being mathematically rigorous.

In the context of generative modeling, this framework extends to multidimensional distributions. For multidimensional random variables $X, Z \in \mathbb{R}^n$, the change in $\mathrm{d}\mathbf{z}$ introduces the

Jacobian determinant, which captures the scaling of infinitesimal volume changes under the transformation f:

$$p(\mathbf{x}) = p(\mathbf{z})\left|\det \frac{\partial f^{-1}}{\partial \mathbf{x}}\right|. \tag{6.2}$$

Since neural networks are universal function approximators (Hornik et al. 1989), they can be employed to approximate f. However, the network must satisfy the conditions of a diffeomorphism. Diffeomorphisms, being composable, allow for the construction of a complex transformation f as a composition of simpler diffeomorphisms, $f = f_n \circ ... \circ f_1$. This transformation can be optimized by maximizing the log-likelihood of mapping a complex distribution to a simpler base distribution, enabling both density estimation in one direction and sampling in the other.

This approach forms the basis of a model known as a NF (Tabak & Turner 2013; Tabak & Vanden-Eijnden 2010). A visualization of a NF is presented in Figure 6.1. In essence, NFs transform a simple base distribution into a more flexible one through a series of bijective transformations. For further details and a comprehensive overview of common architectures used in constructing NFs, refer to Kobyzev et al. 2021 and Papamakarios et al. 2021.

In the following sections, various architectures for constructing NFs will be examined. The architecture choice is constrained by the requirement for the mapping to be a diffeomorphism and the need for efficient computation of the logarithm of the Jacobian determinant.

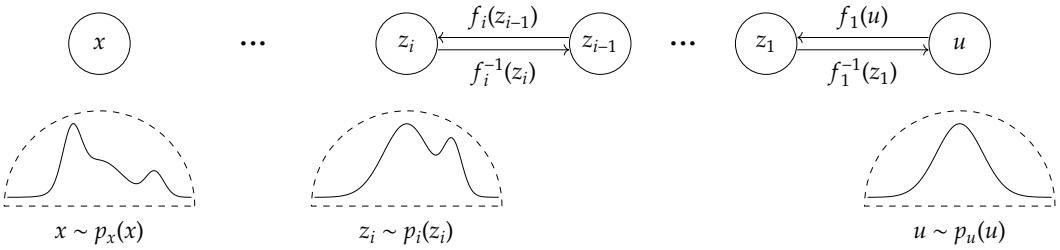

Figure 6.1: Visualization of a Normalizing Flow in one dimension.

6.1.1 *Univariate Invertible Functions*

In the univariate case, where the goal is to describe the probability distribution $p(x)$ for a one-dimensional variable x, a NF can be constructed using invertible mappings. For multidimensional variables, $\mathbf{x} = (x_i)_{i=1}^n$, where each dimension is independent, the joint probability distribution is expressed as $p(\mathbf{x}) = p(x_1) ... p(x_n)$. In such cases, each dimension x_i is treated independently, and the invertible mapping is applied to each dimension separately. This mapping is composed of any invertible parameterized function $f_\varphi(x)$, where the inverse function f_φ^{-1} must be accessible, and the derivative $\mathrm{d}f_\varphi/\mathrm{d}x$ must be computable.

Affine Functions

One of the simplest invertible functions is an affine mapping:

$$f_\varphi(x) = e^\alpha \cdot x + \beta, \qquad \varphi = \{\alpha, \beta\}. \tag{6.3}$$

In this function, the scaling factor e^α ensures that the function remains invertible by preventing the scaling from being zero. Additionally, the logarithm of the absolute derivative is straightforward to compute:

$$\ln \left| \frac{\mathrm{d}f_\varphi}{\mathrm{d}x} \right| = \ln |e^\alpha| = \alpha. \tag{6.4}$$

Monotonic Rational Quadratic Splines

Another method for constructing invertible functions is the monotonic *Rational Quadratic Spline* (RQS). These splines are defined within a bounded region $(-B, B)$, where the spline is applied, while outside this region, an identity transformation is used. The bounded region is divided into $K+1$ coordinates, denoted as $\{x^{(k)}, y^{(k)}\}_{k=0}^{K}$, referred to as *knots*. Each knot k is associated with a derivative $\delta^{(k)}$, which must be positive to ensure the monotonicity of the function. At the boundaries $-B$ and B, the derivative is set to one in order to match the identity transformation outside the domain, i.e., $\delta^{(0)} = \delta^{(K)} = 1$.

The quadratic monotone rational splines between two knots can be constructed by first defining the total slope between the knots as $s^{(k)} = (y^{(k+1)} - y^{(k)})/(x^{(k+1)} - x^{(k)})$. The input variable is expressed as the fraction of x between knots k and $k+1$, denoted $\xi(x) = (x - x^{(k)})/(x^{(k+1)} - x^{(k)})$. The rational quadratic spline, $f^{(k)}(\xi) = \alpha^{(k)}(\xi)/\beta^{(k)}(\xi)$, within the k^{th} bin is given by:

$$\frac{\alpha^{(k)}(\xi)}{\beta^{(k)}(\xi)} = y^{(k)} + \frac{\left(y^{(k+1)} - y^{(k)}\right)\left[s^{(k)}\xi^2 + \delta^{(k)}\xi(1-\xi)\right]}{s^{(k)} + \left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right]\xi(1-\xi)}. \tag{6.5}$$

The derivative of the spline, $\mathrm{d}f^{(k)}(\xi)/\mathrm{d}x$, is given by

$$\frac{\left(s^{(k)}\right)^2\left[\delta^{(k+1)}\xi^2 + 2s^{(k)}\xi(1-\xi) + \delta^{(k)}(1-\xi)^2\right]}{\left[s^{(k)} + \left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right]\xi(1-\xi)\right]^2}. \tag{6.6}$$

The inverse function $f^{-1}(y) = \xi(x) = 2c/(-b - \sqrt{b^2 + 4ac})$ can be expressed in terms of the parameters as follows:

$$a = \left(y^{(k+1)} - y^{(k)}\right)\left[s^{(k)} - \delta^{(k)}\right] + \left(y - y^{(k)}\right)\left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right], \tag{6.7}$$

$$b = \left(y^{(k+1)} - y^{(k)}\right)\delta^{(k)} - \left(y - y^{(k)}\right)\left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right], \tag{6.8}$$

$$c = -s^{(k)}\left(y - y^{(k)}\right). \tag{6.9}$$

Thus, the function $f_\varphi(x)$ is defined in a piecewise manner, with the parameters $\varphi = \{x^{(k)}, y^{(k)}, \delta^{(k)}\}_{k=0}^{K}$ and B. The above formulation follows the notation and presentation of Durkan et al. 2019. A rational quadratic spline for some arbitrarily chosen knots is illustrated in Figure 6.2.

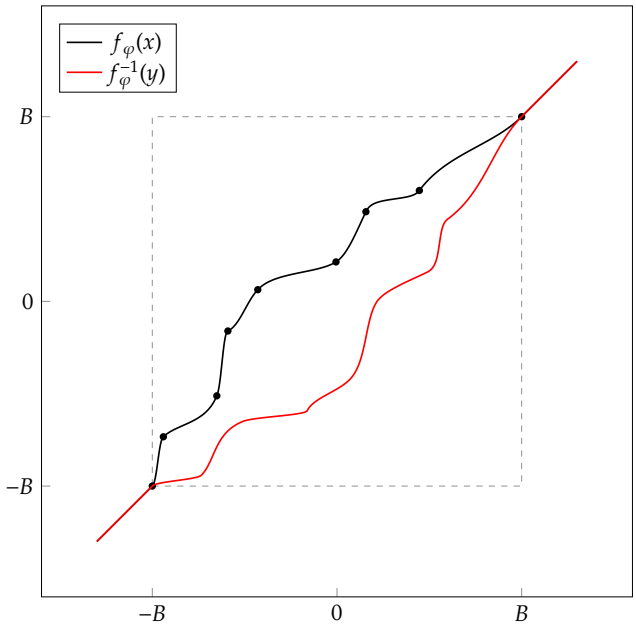

Figure 6.2: The black curve shows a monotonic rational quadratic spline for some arbitrary knots. The knots are shown as black dots. The derivatives are not shown. The red curve is the inverse of the spline.

Univariate Flows

To define the density of a univariate variable x, a composition of multiple univariate invertible functions is constructed. Each of these functions is paired with a neural network that predicts the parameters φ given the conditional features. These neural networks are trained to maximize the probability of a sample of $p(x)$ mapping to a base distribution. Several other options for invertible univariate functions exist; for further details, refer to the review by Papamakarios et al. 2021. This section, however, only discusses the specific functions used in this thesis.

6.1.2 *Autoregressive Flows*

In the case of a two-dimensional variable $\mathbf{x} = (x_1, x_2)$, where x_1 and x_2 are correlated, the probability density $p(\mathbf{x}|c)$ can be factorized as:

$$p(\mathbf{x}|c) = p(x_1, x_2|c) = p(x_1|c)\, p(x_2|x_1, c)\,,$$

where c is an optional variable that $\mathbf{x}$ could be conditioned on.

First, the density $p(x_1|c)$ is mapped using a univariate mapping f_{φ_1}. Here, in a slight abuse of notation, φ_1 represents the parameters of the mapping, and $\varphi_1(c)$ denotes the network that generates these parameters with c as the input. Subsequently, $p(x_2|x_1, c)$ can also be viewed as a univariate case. In this instance, x_1 is part of the conditional features. The variable is transformed by a univariate function f_{φ_2}, where the value of x_1 is provided as a conditional feature $\varphi_2(x_1, c)$.

The logarithm of the Jacobian determinant is then given by:

$$\begin{aligned}\ln\left|\det\frac{\partial f}{\partial \mathbf{x}}\right| &= \ln\left|\det\begin{pmatrix}\frac{\partial f_{\varphi_1}}{\partial x_1} & 0\\ \frac{\partial f_{\varphi_2}}{\partial x_1} & \frac{\partial f_{\varphi_2}}{\partial x_2}\end{pmatrix}\right| \\ &= \ln\left|\frac{\partial f_{\varphi_1}}{\partial x_1}\cdot\frac{\partial f_{\varphi_2}}{\partial x_2}\right| \\ &= \ln\left|\frac{\partial f_{\varphi_1}}{\partial x_1}\right| + \ln\left|\frac{\partial f_{\varphi_2}}{\partial x_2}\right|.\end{aligned} \tag{6.10}$$

Both derivatives $\partial f_{\varphi_i}/\partial x_i$ are obtained by differentiating the univariate functions f_{φ_i} without differentiating the associated neural networks φ_i. Consequently, they are relatively straightforward to compute.

This approach can be extended to n variables $(x_1, \dots, x_n)$ as:

$$p(x_1, \dots, x_n|c) = p(x_1|c)\cdot p(x_2|x_1, c)\cdots p(x_n|x_1, \dots, x_{n-1}, c)\,.$$

For computing the logarithm of the Jacobian determinant, a triangular Jacobian is first defined as:

$$J^f_{i,j} = \begin{pmatrix}\frac{\partial f_{\varphi_i}}{\partial x_i} & 0 & 0 & \cdots & 0\\ \frac{\partial f_{\varphi_{i+1}}}{\partial x_i} & \frac{\partial f_{\varphi_{i+1}}}{\partial x_{i+1}} & 0 & \cdots & 0\\ \frac{\partial f_{\varphi_{i+2}}}{\partial x_i} & \frac{\partial f_{\varphi_{i+2}}}{\partial x_{i+1}} & \ddots & \ddots & \vdots\\ \vdots & \ddots & \ddots & \ddots & 0\\ \frac{\partial f_{\varphi_j}}{\partial x_i} & \cdots & \frac{\partial f_{\varphi_j}}{\partial x_{j-2}} & \frac{\partial f_{\varphi_j}}{\partial x_{j-1}} & \frac{\partial f_{\varphi_j}}{\partial x_j}\end{pmatrix}, \tag{6.11}$$

where the determinant of $J^f_{i,j}$ is

$$\det J^f_{i,j} = \prod_{k=i}^{j}\frac{\partial f_{\varphi_k}}{\partial x_k}. \tag{6.12}$$

Thus, the Jacobian of the transformation in this n-dimensional case is:

$$\ln\left|\det\frac{\partial f}{\partial \mathbf{x}}\right| = \ln\left|\det J^f_{1,n}\right| = \ln\left|\prod_{i=1}^{n}\frac{\partial f_{\varphi_i}}{\partial x_i}\right| = \sum_{i=1}^{n}\ln\frac{\partial f_{\varphi_i}}{\partial x_i}. \tag{6.13}$$

The last equality holds because it is a required condition that all f_{φ_i} be monotone univariate functions. The order of the variables is arbitrary, allowing the mapping of multiple transformations to be composed, each with a different permutation of variables.

The transformation function $f_{\varphi_1}(x_1)$ is defined as $f_1(x_1, \varphi_1(c))$, where φ_1 depends only on the conditioning variable c. For the second variable, the transformation $f_{\varphi_2}(x_2)$ depends on both x_1 and c, hence it is set as $f_2(x_2, \varphi_2(x_1, c))$. This pattern continues such that for the i-th variable, the transformation function $f_{\varphi_i}(x_i)$ is defined as:

$$f_{\varphi_i}(x_i) = f_i(x_i, \varphi_i(x_{<i}, c)), \tag{6.14}$$

where $x_{<i} = \{x_1, \dots, x_{i-1}\}$ represents all preceding variables.

This sequential architecture allows the model to utilize any recurrent architecture, such as *Long Short-Term Memory* (LSTM) networks (Hochreiter & Schmidhuber 1997), which can take the current variable as input and predict the parameters for transforming the next variable.

Recurrent architectures within autoregressive flows have been employed by Kingma et al. 2016 and Oliva et al. 2018. This approach requires n passes through the networks in both directions; however, all necessary entries can be computed efficiently in one pass by masking out all unavailable information. This masking technique was utilized by Germain et al. 2015 and further developed to create *Masked Autoregressive Flow* (MAF) by Papamakarios et al. 2017.

The inverse transformation is more challenging since the necessary information may not be available. Therefore, evaluating the model recursively requires n passes through the network, resulting in both fast and slow dimensions within the network.

6.1.3 *Coupling Flows*

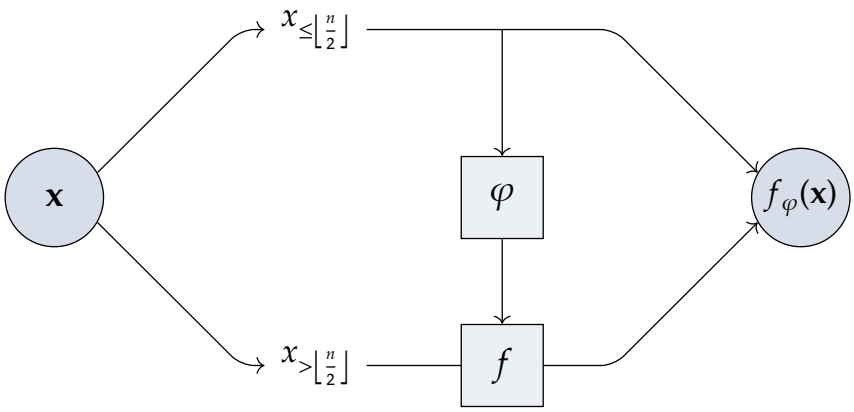

Figure 6.3: Illustration of a coupling layer, in which the variable $\mathbf{x}$ is transformed. The first half of its features $x_{\leq \lfloor \frac{n}{2} \rfloor}$ remain unchanged and are input to the network φ, which provides the parameters for the transformation f. The input to the transformation f are the second half of the features $x_{> \lfloor \frac{n}{2} \rfloor}$.

Instead of transforming each variable individually, variables can be transformed in groups, with each group conditioned on another. This model family is referred to as *Coupling Flows* (CFs) (Dinh et al. 2014, 2017). One common approach is to split the variable $\mathbf{x}$ into two halves:

$$x_{\leq \lfloor \frac{n}{2} \rfloor} = \left\{ x_1, \dots, x_{\lfloor \frac{n}{2} \rfloor} \right\}, \tag{6.15}$$

$$x_{> \lfloor \frac{n}{2} \rfloor} = \left\{ x_{\lfloor \frac{n}{2} \rfloor + 1}, \dots, x_n \right\}. \tag{6.16}$$

The probability density can then be expressed as

$$p(x_1, \dots, x_n | c) = p\left(x_{\leq \lfloor \frac{n}{2} \rfloor}, x_{> \lfloor \frac{n}{2} \rfloor} \middle| c \right) = p\left(x_{> \lfloor \frac{n}{2} \rfloor} \middle| x_{\leq \lfloor \frac{n}{2} \rfloor}, c \right) \tag{6.17}$$

by transforming one half based on the other half, which remains unchanged.

The Jacobian of a CF has a triangular structure, which allows for efficient computation of the logarithm of the transformation:

$$\begin{aligned}\ln\left|\det\frac{\partial f}{\partial \mathbf{x}}\right| &= \ln\left|\det\begin{pmatrix}\mathbb{1} & 0\\ \cdot & J^{f}_{\lfloor\frac{n}{2}\rfloor+1,n}\end{pmatrix}\right| = \ln\left|\prod_{i=1}^{\lfloor\frac{n}{2}\rfloor}1\cdot\prod_{i=\lfloor\frac{n}{2}\rfloor+1}^{n}\frac{\partial f_{\varphi_i}}{\partial x_i}\right| \\ &= \ln\prod_{i=\lfloor\frac{n}{2}\rfloor+1}^{n}\frac{\partial f_{\varphi_i}}{\partial x_i} = \sum_{i=\lfloor\frac{n}{2}\rfloor+1}^{n}\ln\frac{\partial f_{\varphi_i}}{\partial x_i}.\end{aligned} \tag{6.18}$$

Each transformation f_{φ_i} takes the form

$$f_{\varphi_i}(x_i) = f_i\left(x_i, \varphi_i\left(x_{\leq\lfloor\frac{n}{2}\rfloor}, c\right)\right), \quad \forall\, i \in \left\{\left\lfloor\frac{n}{2}\right\rfloor+1, \dots, n\right\}. \tag{6.19}$$

To simplify, the same univariate transformation f and network φ can be applied to all variables x_i:

$$f_{\varphi}(x_i) = f\left(x_i, \varphi\left(x_{\leq\lfloor\frac{n}{2}\rfloor}, c\right)\right). \tag{6.20}$$

The use of a CF layer allows for the prediction of the parameters of univariate functions applied to all transformed features using a single neural network. This provides flexibility in the choice of architecture while maintaining consistent computational speed in both directions. A visualization of such a layer in a CF is illustrated in Figure 6.3.

A variety of models based on coupling flows have been proposed for different applications (Ho et al. 2019; Kim et al. 2018; Kingma & Dhariwal 2018; Prenger et al. 2019).

6.2 Variational Autoencoder

A Normalizing Flow can be considered a mapping between a data space $\mathbf{x}$ and a latent space $\mathbf{z}$. Since flows are bijective, they do not alter the dimensionality of the data. It is often assumed that the data $\mathbf{x}$ resides on a manifold within the space in which it is defined, implying the existence of a lower-dimensional representation in the latent space $\mathbf{z}$. The goal of dimensionality reduction is to create a more compact representation of the data while retaining its essential information. This lower-dimensional representation facilitates the generation of samples on the manifold and their expansion into the data space. Consequently, compressing the data set is a desirable outcome.

Autoencoders (Baldi & Hornik 1989; Bourlard & Kamp 1988; Rumelhart et al. 1986) use neural networks to learn compact representations of data in a lower-dimensional latent space $\mathbf{z}$. The data $\mathbf{x}$ is compressed (encoded) by one neural network, referred to as the *Encoder*, and subsequently decompressed (decoded) by another neural network, called the *Decoder*, back to the data space. This model is trained in an unsupervised manner by minimizing the reconstruction error between the original data and the reconstructed data. However, unlike the NF case, the latent representation in a standard autoencoder does not have a closed form.

As a result, it is not possible to directly sample from the latent representation, limiting its suitability as a generative model.

A probabilistic formulation of the autoencoder is the *Variational Autoencoder* (VAE) (Kingma & Welling 2013b; Rezende et al. 2014). In a VAE, a joint distribution $p(\mathbf{x}, \mathbf{z})$ is defined, from which the distribution $p(\mathbf{x})$ can be obtained by marginalizing out $\mathbf{z}$:

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \int_{\mathbf{z}} p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})\, d\mathbf{z}. \tag{6.21}$$

Here, $p(\mathbf{x}|\mathbf{z})$ represents the likelihood, and the objective is to train the Decoder to maximize $p(\mathbf{x}|\mathbf{z})$. It is necessary to define a prior distribution $p(\mathbf{z})$ and to determine the posterior $p(\mathbf{z}|\mathbf{x})$. However, $p(\mathbf{z}|\mathbf{x})$ is intractable because generating all possible values of the latent variable $\mathbf{z}$ is not feasible.

To train VAEs, (amortized) *variational inference* (Blei et al. 2017; Jordan et al. 1999; Wainwright, Jordan, et al. 2008) is employed, whereby a function $q(\mathbf{z}|\mathbf{x})$ is introduced to approximate the posterior:

$$q(\mathbf{z}|\mathbf{x}) \approx p(\mathbf{z}|\mathbf{x}). \tag{6.22}$$

This approximation implies the need to minimize

$$\begin{aligned} D_{\mathrm{KL}}\left(q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}|\mathbf{x})\right) &= \left\langle \log \frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})} \right\rangle_{q(\mathbf{z}|\mathbf{x})} = \left\langle \log \frac{q(\mathbf{z}|\mathbf{x}) p(\mathbf{x})}{p(\mathbf{x}, \mathbf{z})} \right\rangle_{q(\mathbf{z}|\mathbf{x})} \\ &= \log p(\mathbf{x}) - \underbrace{\left\langle \log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right\rangle_{q(\mathbf{z}|\mathbf{x})}}_{\text{ELBO}}. \end{aligned} \tag{6.23}$$

Here, the *Evidence Lower Bound* (ELBO) has been introduced. Maximizing the ELBO minimizes the gap $D_{\mathrm{KL}}\left(q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}|\mathbf{x})\right)$. The ELBO can be expressed as

$$\begin{aligned} \text{ELBO} &= \left\langle \log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right\rangle_{q(\mathbf{z}|\mathbf{x})} = \left\langle \log \frac{p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right\rangle_{q(\mathbf{z}|\mathbf{x})} \\ &= \left\langle \log p(\mathbf{x}|\mathbf{z}) \right\rangle_{q(\mathbf{z}|\mathbf{x})} + \left\langle \log \frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right\rangle_{q(\mathbf{z}|\mathbf{x})} \\ &= \left\langle \log p(\mathbf{x}|\mathbf{z}) \right\rangle_{q(\mathbf{z}|\mathbf{x})} - D_{\mathrm{KL}}\left(q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})\right). \end{aligned} \tag{6.24}$$

The VAE is optimized by maximizing the likelihood of the reconstruction while minimizing the KL divergence between the approximate posterior and the induced probability distribution on the latent variable $\mathbf{z}$. For a trained VAE, the latent variable $\mathbf{z} \sim p(\mathbf{z})$ can be sampled and decoded by the Decoder into the data space $\mathbf{x}$.

To minimize $D_{\mathrm{KL}}\left(q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})\right)$, it is necessary to sample from $\mathbf{z}$. In order to train the VAE using backpropagation, it is therefore necessary to differentiate through the sampling process, which is not directly feasible. To address this issue, Kingma and Welling 2013b developed the *reparameterization trick*. If $q(\mathbf{z}|\mathbf{x})$ is defined as a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, with $\boldsymbol{\mu}(\mathbf{x})$ and $\boldsymbol{\sigma}^2(\mathbf{x})$ both predicted by the *Encoder* given a sample $\mathbf{x}$, sampling from $q(\mathbf{z}|\mathbf{x})$ can be done as

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}, \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbb{1}). \tag{6.25}$$

With this construction, backpropagation can be performed through $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. If $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbb{1})$, the KL can be calculated as

$$\begin{aligned} D_{\mathrm{KL}}\left(q(\mathbf{z}|\mathbf{x})\middle\|p(\mathbf{z})\right) &= D_{\mathrm{KL}}\left(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)\middle\|\mathcal{N}(\mathbf{0}, \mathbb{1})\right) \\ &= \log(\boldsymbol{\sigma}) + \frac{1 + \boldsymbol{\mu}^2}{2\boldsymbol{\sigma}^2} - \frac{1}{2} \end{aligned} \tag{6.26}$$

The formulation presented here represents the most basic case; however, it can be readily extended to other cases, which are not discussed in this thesis as

6.2.1 *Latent Normalizing Flows*

One limitation of the VAE framework, as previously discussed, is the restriction of the prior to a simple normal distribution $\mathcal{N}(\mathbf{0}, \mathbb{1})$. This restriction also limits the flexibility of the amortized posterior $q(\mathbf{z}|\mathbf{x})$, thereby constraining the minimization of the gap $D_{\mathrm{KL}}\left(q(\mathbf{z}|\mathbf{x})\middle\|p(\mathbf{z}|x)\right)$.

To introduce greater flexibility, a Normalizing Flow can be employed to transform the prior distribution. Let f represent the mapping of the NF, which transforms the base variable $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbb{1})$, with $\mathbf{z} = f(\mathbf{w})$. The ELBO can then be restructured as

$$\begin{aligned} \text{ELBO} &= \langle \log p(\mathbf{x}|\mathbf{z}) \rangle_{q(\mathbf{z}|\mathbf{x})} - D_{KL}\left(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})\right) \\ &= \langle \log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x}) \rangle_{q(\mathbf{z}|\mathbf{x})} \\ &= \langle \log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}) \rangle_{q(\mathbf{z}|\mathbf{x})} - \mathscr{H}\left(q(\mathbf{z}|\mathbf{x})\right) \\ &= \left\langle \log p(\mathbf{x}|\mathbf{z}) + \log p(f^{-1}(\mathbf{z})) + \log \left| \det \frac{\partial f^{-1}}{\partial \mathbf{z}} \right| \right\rangle_{q(\mathbf{z}|\mathbf{x})} - \mathscr{H}\left(q(\mathbf{z}|\mathbf{x})\right). \end{aligned} \tag{6.27}$$

Thus, the latent Normalizing Flow can be optimized in conjunction with the VAE. Here, $\mathscr{H}\left(q(\mathbf{z}|\mathbf{x})\right)$ represents the entropy of the approximate posterior. Given that it is normally distributed, the entropy can be derived as

$$\mathscr{H}\left(q(\mathbf{z}|\mathbf{x})\right) = \frac{n}{2}\left(1 + \log(2\pi)\right) + \sum_{i=1}^{n} \log \sigma_i. \tag{6.28}$$

The concept of implementing a NF in this way was initially proposed by Rezende and Mohamed 2015. However, our derivation does not directly follow theirs. By integrating NFs, the flexibility and capability of the VAE are enhanced, enabling the modeling of more complex data distributions and improving generative performance.

6.3 PointFlow

In this section, the components described in the preceding sections are integrated to construct the model, which will be utilized primarily in the subsequent chapters. The *PointFlow* model, originally developed by G. Yang et al. 2019, is based on continuous NFs. For this discussion, the focus is on the NFs detailed in Section 6.1, as they offer the advantage of faster

generation times by avoiding the need to solve differential equations during sampling. A variant of the PointFlow model employing non-continuous NFs has been devised by Klokov et al. 2020.

Consider a point cloud X as described in Section 5.7. PointFlow is designed as a VAE that autoencodes the point cloud. Therefore, X must first be mapped to a latent representation z using an Encoder. For a point cloud, the Encoder must be permutation invariant; thus, a Deep Sets based Encoder is chosen. This involves first mapping the point cloud into a high-dimensional space, followed by a pooling operation. While the original PointFlow model used max pooling, mean pooling is chosen here. From this pooled high-dimensional representation, the parameters μ and σ of the latent representation z are derived, allowing for sampling of z as explained in Section 6.2. Additionally, a latent normalization flow is included within the latent space, as described in Section 6.2.1.

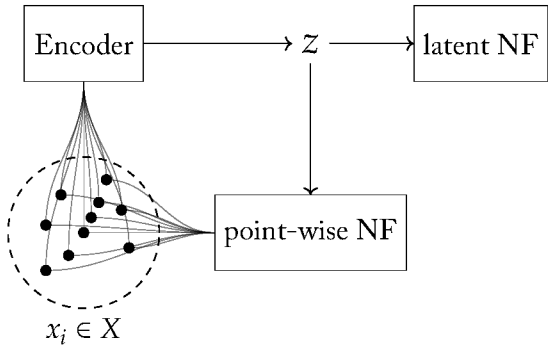

Figure 6.4: Visualization of a the PointFlow training setup. The point cloud X is encoded with the Encoder to z. The latent Normalizing Flow (NF) is trained on z. Conditioned on z the point-wise NF learns the points individually.

To complete the model, a suitable Decoder is required to model $p(X|z)$. The assumption underlying the PointFlow model is that points are *independent and identically distributed* (i.i.d.). Consequently, $p(X|z)$ can be decomposed as

$$p(X|z) = \prod_{i=1}^{n} p(x_i|z). \tag{6.29}$$

A point-wise NF is used to model each $p(x_i|z)$, conditioned on the latent representation z.

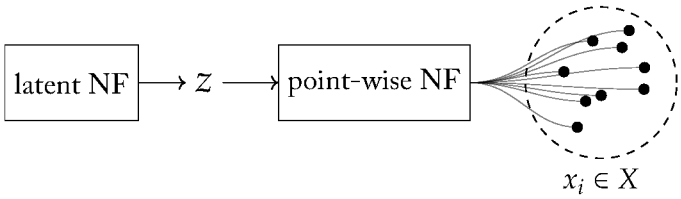

Figure 6.5: Visualization of a the PointFlow sampling setup. The latent NF samples z. Conditioned on z the point-wise NF samples each points individually.

The model is trained to maximize the ELBO, thereby training both flows on maximum likelihood and training the encoder to produce a high-quality latent representation. The training setup is illustrated in Figure 6.4.

To obtain a sample from the model, a latent representation z is first sampled from the latent NF, and then each point is sampled individually, conditioned on z. The sampling setup is illustrated in Figure 6.5.

This approach implies that there is no information exchange between the points during the generation process, restricting the model to capturing only global correlations within the point cloud, rather than one-to-one correlations between different points. This limitation will be further discussed in Chapter 8. The original model by G. Yang et al. 2019 was designed for point clouds sampled i.i.d. from a volume, thus not encountering this intricacy.

FAST SIMULATION FOR THE CMS CALORIMETERS

7

The CMS experiment relies extensively on simulations to analyze high-energy particle collisions, assess detector performance, and enable precision measurements. Geant4 serves as the primary framework for these simulations, providing highly accurate modeling of particle interactions with detector materials in what are known as *Full Simulation* (FullSim). However, the computational cost of these full simulations increases significantly with experimental demands, such as finer detector granularity and higher luminosity. To address this challenge, the CMS collaboration has developed and continually updates an alternative framework known as *Fast Simulation* (FastSim), which simplifies or replaces certain aspects of the simulation to achieve greater computational efficiency.

Recent efforts within the CMS FastSim group have focused on developing generative machine learning-based surrogate models intended to replace the current parametrized FastSim algorithms. These models aim to further improve the speed and accuracy of the fast simulation process.

This chapter provides an overview of fast simulation techniques within the CMS framework. It begins with a general description of FastSim, outlining its role in enhancing computational efficiency by streamlining elements of the full simulation process. The discussion then narrows to the specifics of calorimetry, examining the methods used to simulate particle interactions with calorimeter components. Subsequently, the currently employed parametrized model, based on the *GFlash* algorithm, is introduced, with an explanation of its functionality and limitations. Finally, the chapter concludes with an outline of the approach to integrating machine learning-based methods into the FastSim pipeline, highlighting how this innovative strategy can significantly improve both the speed and accuracy of simulations.

7.1 Fast Simulation in CMS

FastSim is embedded within the CMS software framework and provides substantial improvements in simulation efficiency, achieving up to a 100-fold speed increase, or a 20-fold increase when accounting for both simulation and reconstruction stages. FastSim is particularly valuable for tasks requiring extensive model scans, such as supersymmetry (SUSY) searches, exotic particle studies, and systematic analyses for top quark measurements.

To ensure reliability, the results produced by FastSim are frequently validated against those yielded by FullSim. While some discrepancies are expected due to the simplified methodology, these are closely monitored and minimized to maintain FastSim as a reliable tool for physics analyses.

The FastSim process begins with the same generated particles as the FullSim, simulating energy deposits along particle trajectories to produce *Simulated Hits* (SimHits). The efficiency of FastSim stems from its simplified framework, geometry, and parameterized material in-

teractions. The input to FastSim is a list of particles from either the EventGenerator or ParticleGun, each characterized by momentum and origin vertex. Particles are propagated through the magnetic field to various CMS sub-detector volumes, where interactions may occur. The CMS tracker geometry is approximated by infinitely thin cylinders and disks, with materials placed on the surface. Each layer interaction is described by material thickness in terms of radiation length and interaction length, and particle propagation is simplified using an approximated magnetic field.

As particles pass through each sub-detector layer, interactions can produce secondary particles, which are then added to the list of particles continuing to propagate through the detector.

FastSim also accounts for pile-up interactions—additional collisions occurring during the same bunch crossing—by incorporating particles drawn from pre-generated files, added to the event according to a Poisson distribution. The simulation considers various interactions, including Bremsstrahlung, photon conversion, ionization energy loss of charged particles, multiple scattering, and nuclear interactions. Additionally, electron, photon, and hadron showering are parameterized in the electromagnetic and hadron calorimeters. The first five processes are applied as particles traverse the thin tracker layers, while the latter are reserved for the calorimeter simulation.

7.2 Calorimetry in CMS FastSim

7.2.1 *Parametrized Model*

For calorimeter shower simulation, CMS FastSim (Giammanco 2014) employs a parametrized model based on the GFlash algorithm, originally developed by (Grindhammer & Peters 1993) for the H1 Experiment at HERA and later ported to Geant4.

In GFlash, the shower is sampled from the parameterized energy density distribution. The particle's track is followed in Geant4 until the first inelastic interaction, which defines the shower starting point. From that point, GFlash takes over, parameterizing the subsequent development of the shower. The longitudinal energy density distribution is numerically integrated, and energy is deposited perpendicular to the shower track according to the lateral distribution density.

GFlash uses the average material properties of the volume in which the shower develops, allowing for a simplified geometry. In the case of sampling calorimeters, an effective average material property is defined, considering atomic number, charge number, radiation length, absorption length, and mean density.

Electromagnetic and hadronic showers are modeled in Geant4 using the GFlash algorithm, which parametrizes both longitudinal and lateral shower profiles for homogeneous and sampling calorimeters.

In GFlash, the hadronic shower is divided into three components: the purely hadronic part,

the π^0 fraction from the first interaction, and the electromagnetic fraction produced through subsequent interactions during shower development.

The total deposited energy of a shower is parametrized using the following distribution function:

$$E_{dp} = \int dE_{dp}(\mathrm{r}) = \int_V E_{dp} f(\mathrm{r})\, dV \tag{7.1}$$

where the condition

$$\int_V f(\mathrm{r})\, dV = 1 \tag{7.2}$$

ensures normalization of the distribution.

The volumetric separation of deposited energy $dE_{dp}(\mathrm{r})$, is given by

$$dE_{dp}(\mathrm{r}) = E_{dp} f(\mathrm{r}) dV = E_{dp} f_z(z) dz f_r(r) dr f_\phi(\phi) d\phi \tag{7.3}$$

where the function $f(\mathrm{r})$ is factored into three independent functions, f_z, f_r, and f_ϕ, representing the longitudinal, radial, and azimuthal energy distribution densities, respectively.

The longitudinal energy distribution is modeled using three correlated gamma distributions. For the lateral energy distribution, the radial energy density is described as

$$f_r(r) = \frac{2rR_{50}^2}{(r^2 + R_{50}^2)}. \tag{7.4}$$

where the free parameter R_{50} scales with the interaction length for hadronic showers and the Molière radius for electromagnetic showers. The expected value and variance of R_{50} are parametrized as a function of energy and shower depth.

The showers are assumed to be symmetric around the beam axis, resulting in a uniform azimuthal distribution. The azimuthal distribution density is given by

$$f_\phi(\phi) = \frac{1}{2\pi}. \tag{7.5}$$

For electron and photon showers, the GFlash parametrization treats the CMS ECAL as a homogeneous medium. This approximation is justified due to the uniform structure of the PbWO4 crystal calorimeters in CMS. The simulation process occurs in two stages: first, the shower is modeled using the GFlash parametrization, and then the deposited energy is integrated over two longitudinal slices, each with a thickness of one radiation length (X_0).

The main advantage of GFlash over stepwise methods like Geant4 is its substantial speed improvement. The computational cost of the GFlash model scales with the track length of the primary particle, and thus with its energy. As the spatial extent of a particle shower increases logarithmically with energy, the computational demand also increases, but at a much slower rate compared to FullSim.

GFlash is capable of efficiently modeling both the longitudinal and lateral energy density distributions for hadronic and electromagnetic showers. The spatial extent of a particle shower increases logarithmically with energy, as does the computational demand.

7.2.2 *Machine Learning Based Simulation*

The current CMS FastSim framework samples SimHits from the predefined GFlash distribution. However, this distribution is simplified in ways that lead to inaccuracies. Factors such as the particle angle or the extent to which the shower is contained within the calorimeters can alter the energy deposition distribution, but these factors are not fully captured in the GFlash model. Additionally, the hadron calorimeter is treated as a single, simplified volume, which does not reflect the true complexity of the detector. These limitations have motivated the introduction of machine learning models, which have the potential to predict SimHit distributions more accurately by accounting for specific local conditions within the calorimeter.

When integrating machine learning into FastSim, two main strategies can be considered. One option is to generate SimHits, following a similar approach to Geant4, where the model aims to replicate the distribution of SimHits throughout the detector. The second option is to directly target rec hits, focusing on the final energy deposits in the active detector materials.

In the Geant4 approach, SimHits are generated by sampling from a parameterized probability distribution. Geant4 simulates the particle steps and places SimHits along the particle tracks as they propagate through the detector. Adopting a machine learning strategy to replicate these distributions would allow for greater flexibility and potentially better modeling accuracy. However, SimHits vary significantly depending on the material properties of the detector, leading to discontinuities or gaps in the hit distribution. Moreover, the number of SimHits generated per shower can be very large—up to 100,000—making the simulation process computationally expensive, especially when only a fraction of these hits are eventually used in the reconstruction process. While machine learning might not offer a speedup over the current method for SimHits generation, it promises improved accuracy and scalability.

Directly targeting *Reconstructed Hits* (RecHits), on the other hand, offers a more computationally efficient approach. By focusing only on the energy deposits in the active detector volumes, the overall number of points that need to be modeled is significantly reduced. This approach can simplify the simulation process, eliminating the need to map SimHits back to active cells, which is both time-consuming and computationally expensive. However, challenges exist, such as ensuring that the model generates only one RecHit per cell and dealing with gaps between active detector materials, particularly in the HCAL. These challenges will require specific strategies to ensure accuracy and smoothness in the hit distribution. By directly targeting RecHits, the computationally expensive process of mapping SimHits back to active volumes can be bypassed, thus improving overall efficiency.

Considering these factors, the preference leans toward directly targeting RecHits. By bypassing the intermediate step of generating SimHits and focusing on the final data products required for reconstruction, the simulation process can be simplified and accelerated. While the normalizing flow models required to generate SimHits offer some flexibility, the direct modeling of RecHits presents clear benefits in terms of both speed and precision, making it the more efficient approach.

In conclusion, while both approaches—targeting SimHits or RecHits—offer unique advantages, targeting RecHits presents a clear path toward greater computational efficiency and precision. By directly modeling the final energy deposits in the detector's active volumes,

the overall complexity can be reduced, and the simulation process can be streamlined. The remaining challenges, such as ensuring one hit per cell and addressing gaps between detector materials, will be explored in detail in the following chapters, where the machine learning models being developed and the strategies employed to overcome these hurdles will be outlined.

CALOPOINTFLOW

8

This chapter examines the central contribution of this thesis, the *CaloPointFlow* (CPF) architectural model. Initially, the *CaloPointFlow I* (CPF I) (Schnake et al. 2022) model was developed, and subsequently, its updated version, *CaloPointFlow II* (CPF II) (Schnake et al. 2024).

The models will not be presented in the order of their historic origins, with the first model followed by the second. Instead, they will be presented together, with an explicit focus on the differences and similarities between them.

This chapter is structured as follows: First, we conduct an analysis of the benefits of point cloud representations in calorimeter simulation surrogate models (Section 8.1). Subsequently, an overview of the *CaloChallenge* datasets will be presented (Section 8.2). The subsequent sections will describe the architecture (Section 8.3), the training process (Section 8.4), the sampling process (Section 8.5), and the pre- and post-processing (Sections 8.6 and 8.7). Sections 8.3 to 8.7 are structured to first outline the common elements in both architectures and then examine their differences. The chapter will conclude with an evaluation of both models in comparison to Geant4 and to each other (Section 8.8).

8.1 Point Cloud Representation

In Section 4.4, the process by which Geant4 generates calorimeter showers is explained. Geant4 uses a stepwise approach to model particle interactions within the material, resulting in the generation of simulated hits along the particle paths. These simulated hits are then assigned to volumes and clustered into energy deposits within the active material.

Most generative deep learning models have treated calorimeter showers not as point clouds but as voxels (Hashemi & Krause 2023). This approach is primarily driven by the fact that generative models were originally developed with image data in mind, and voxels are a natural extension of image data. By using voxels, existing machine learning research can be leveraged.

In the early stages of this work, a progressively growing GAN was developed using voxelized data. The idea was to fade in new layers in the generator and discriminator while increasing the dimensionality of the data. However, this approach was not successful. The model was unable to retain learned features from the early stages of training in the later stages. The sparsity of the data posed a significant challenge; this sparsity changes dramatically as the dimensionality is reduced or increased, making it difficult to learn the empty cells. Other models also struggled with this sparsity (Paganini et al. 2018b). One possible solution to deal with sparsity is to add uniform noise to all cells without energy and introduce a cutoff energy in post-processing that sets all cells below that energy to zero. This approach was first presented by Krause et al. 2022, and their preprocessing method has been adopted by several models (Amram & Pedro 2023; Buckley et al. 2023; Cresswell et al. 2022; Ernst et al. 2023;

Krause et al. 2022; Mikuni & Nachman 2022, 2024; Pang et al. 2023).

It is common practice to develop models using toy calorimeters that are small or have a regular geometry. However, this approach is not directly applicable to real calorimeter detectors, which often have complex structures. One possible solution is to create a regular cylindrical, highly granular volume in which the simulated hits can be clustered. This volume can then be mapped onto the non-regular detector geometry, allowing models to learn this geometry. A major advantage of this technique is its versatility: by defining different mappings between the cylindrical volume and a detector, it can be used for multiple detectors. An alternative approach used by ATLAS (2018, 2022, 2024) is to divide the calorimeter into several sections, each assigned a specific model.

In this work, another approach is taken. The calorimeter showers are represented by point clouds, as defined in Section 5.7. This representation can be achieved by employing either the simulated hits, which are inherently represented as point clouds, or the clustered energy entries within calorimeter cells as points. The CaloPointFlow I (Schnake et al. 2022) model was one of the first models to facilitate this approach, and others have also studied this method (Acosta et al. 2023; Buhmann, Diefenbacher, Eren, et al. 2023; Buhmann, Gaede, Kasieczka, et al. 2023; Käch et al. 2023).

Both voxel-based and point cloud-based representations have distinct advantages. The voxel-based approach allows for the application of existing machine learning techniques, whereas the point cloud-based approach provides a more precise representation of the sparsity and irregular geometry.

8.2 Datasets

The datasets used in this chapter are the second and third dataset from the CaloChallenge (Giannelli et al. 2022a). The first dataset was excluded due to its irregular geometry, which would have introduced additional complexities in the dequantization process. Furthermore, the calorimeter in the first dataset has a relatively small number of cells, resulting in low sparsity of the showers. This characteristic makes our approach less advantageous for this dataset.

Each dataset consists of 200,000 showers initiated by electrons, divided equally for training and evaluation purposes. Each data point in the dataset contains the incident energy of the incoming particle $E_{i,\text{in}}$ and the energy values in the cells of the calorimeter, denoted as

$$E_{i,z,\alpha,r}. \tag{8.1}$$

Here, i is the shower index in the dataset, ranging from 1 to $n = 100,000$. The incident energy is log-uniformly distributed between 1 GeV and 1 TeV.

Datasets 2 (Giannelli et al. 2022b) and 3 (Giannelli et al. 2022c) are simulated using the same physical detector, which consists of concentric cylinders with 90 layers of absorber and sensitive (active) materials, specifically tungsten (W) and silicon (Si), respectively. Each sublayer consists of 1.4 mm of W and 0.3 mm of Si, resulting in a total detector depth of 153 mm. The detector's inner radius is 80 cm.

The readout segmentation is determined by the direction of the particle entering the calorimeter. This direction defines the z-axis of the cylindrical coordinate system, with the entrance position in the calorimeter set as the origin $(0, 0, 0)$. The voxels (readout cells) in both datasets 2 and 3 have identical sizes along the z-axis but differ in segmentation in radius (r) and angle (α).

For the z-axis, the voxel size is 3.4mm, corresponding to two physical layers (W-Si-W-Si). Considering only the absorber value of the radiation length 1 $X_0(W) = 3.504$ mm, the z-cell size equates to $2 \times 1.4\,\text{mm}/3.504\,\text{mm} = 0.8\,X_0$. In the radial dimension, the cell sizes are 2.325 mm for dataset 3 and 4.65 mm for dataset 2. Approximately, considering the Molière radius of tungsten (W), this corresponds to $0.25\,R_m$ for dataset 3 and $0.5\,R_m$ for dataset 2. The minimum energy threshold for the readout per voxel in datasets 2 and 3 is set to 15.15 keV.

The calorimeter geometry of Dataset 2 comprises 45 concentric cylindrical layers stacked along the direction of particle propagation (z). Each layer is further divided into 16 angular bins (α) and nine radial bins (r), resulting in a total of $45 \times 16 \times 9 = 6480$ voxels.

Dataset 3 from the CaloChallenge features higher granularity compared to Dataset 2. Each layer in Dataset 3 consists of 18 radial bins and 50 angular bins, resulting in a total of $45 \times 50 \times 18 = 40500$ voxels.

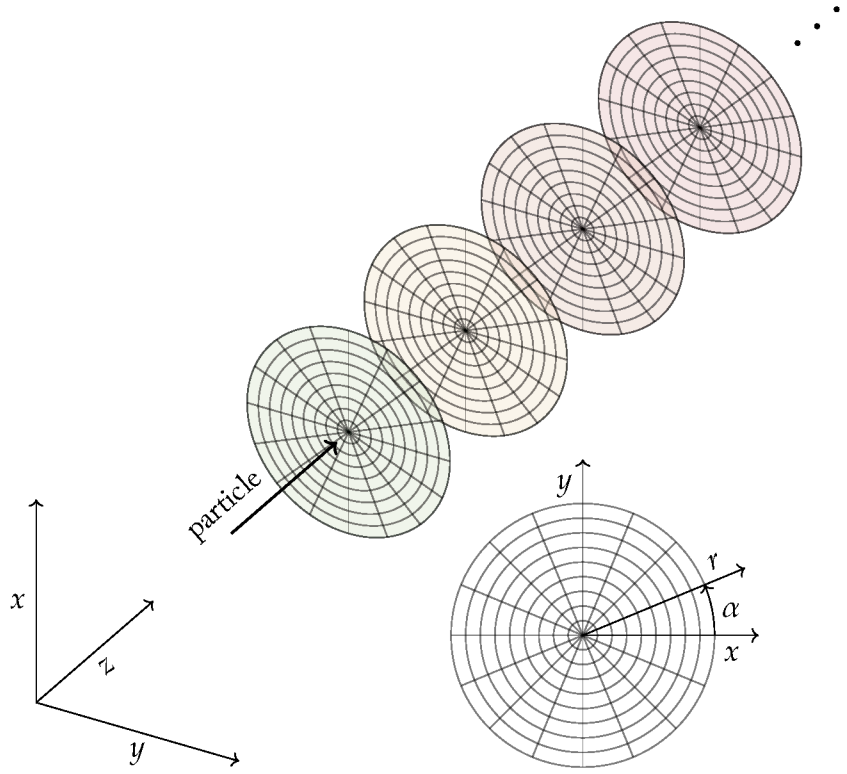

Figure 8.1: Illustration of the CaloChallenge Coordinate System. A 3D outline of the first four calorimeter layers is presented in the longitudinal direction z. Next to the calorimeter outline, the front view one disk in z is presented. Each layer in z is a disk are segmented in the angular direction α and in the radial direction r. The segmentation here corresponds to the one find in Dataset II. A cartesian coordinate system of the dimensions (x, y, z) is also presented.

Figure 8.1 illustrates the coordinate systems of the detectors. Each coordinate of a cell in the detectors is given as a tuple of discrete cell positions (z, α, r). For both datasets, the longitudinal index z ranges from 1 to $n_z = 45$. The angular index α ranges from 1 to $n_\alpha = 16$ for Dataset II and $n_\alpha = 50$ for Dataset III. The radial index r ranges from 1 to $n_r = 9$ for Dataset II and $n_r = 18$ for Dataset III. The cell positions can also be expressed in Cartesian coordinates [1] as

$$x = r \cos\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right), \tag{8.2}$$

$$y = r \sin\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right). \tag{8.3}$$

In Figure 8.2, the average shower in all layers along the z-axis is shown, illustrating the general detector behavior. The corresponding figure for Dataset 2, Figure B.16, is included in Chapter B. These figures primarily differ in terms of granularity.

8.3 Architecture

The underlying structure of both CaloPointFlow models is presented in this section, followed by a discussion of the specific differences between the two versions.

The objective of these models is to generate point clouds with the probability density $p(\mathbf{X})$, where $\mathbf{X} \in \mathbb{R}^{n,m}$ represents the possible showers. The models discussed here are based on the point flow model described in Section 6.3.

The basic elements of the architecture are illustrated in Figures 8.5 and 8.6: Encoder, LatentFlow, PointFlow, and CondFlow. The training process requires encoding the entire shower $\mathbf{X}$ using an encoder $q(z|\mathbf{X})$, resulting in the latent representation z. The encoder is based on the DeepSets architecture described in Section 5.8. It uses two *Multi-Layer Perceptron* (MLP)s for its function. The first MLP maps each point to a 1024-dimensional space. The dimensions of the layers in this MLP are shown in Table 8.1. The second MLP maps the averaged high-dimensional point representation to a 128-dimensional latent space z. The dimensions of the layers in this MLP are shown in Table 8.2. The latent representation is mapped to two 128-dimensional variables, μ and σ. The latent z is then sampled using the reparametrization trick, as described in Section 6.2.

Table 8.1: Dimensions of the layers of the MLPs used point-wise by Encoder.

Layer	Input	1	2	3	Latent
Dimensions	4	64	128	512	1024

The LatentFlow model is introduced to capture the distribution $p(z|n_{\text{hits}}, E_{\text{sum}}, E_{\text{in}})$ of the latent representation z conditioned on the number of hits n_{hits}, the total energy E_{sum}, and the incident energy E_{in}. The LatentFlow is modeled as a Coupling Flow, as detailed in Section 6.1.3, and utilizes eight transforms. In each transform, a randomly selected half of the

1 The CaloChallenge utilizes the terms η and ϕ to denote the Cartesian coordinates. However, I have chosen to refrain from using this nomenclature to prevent any potential confusion between η and the pseudorapidity.

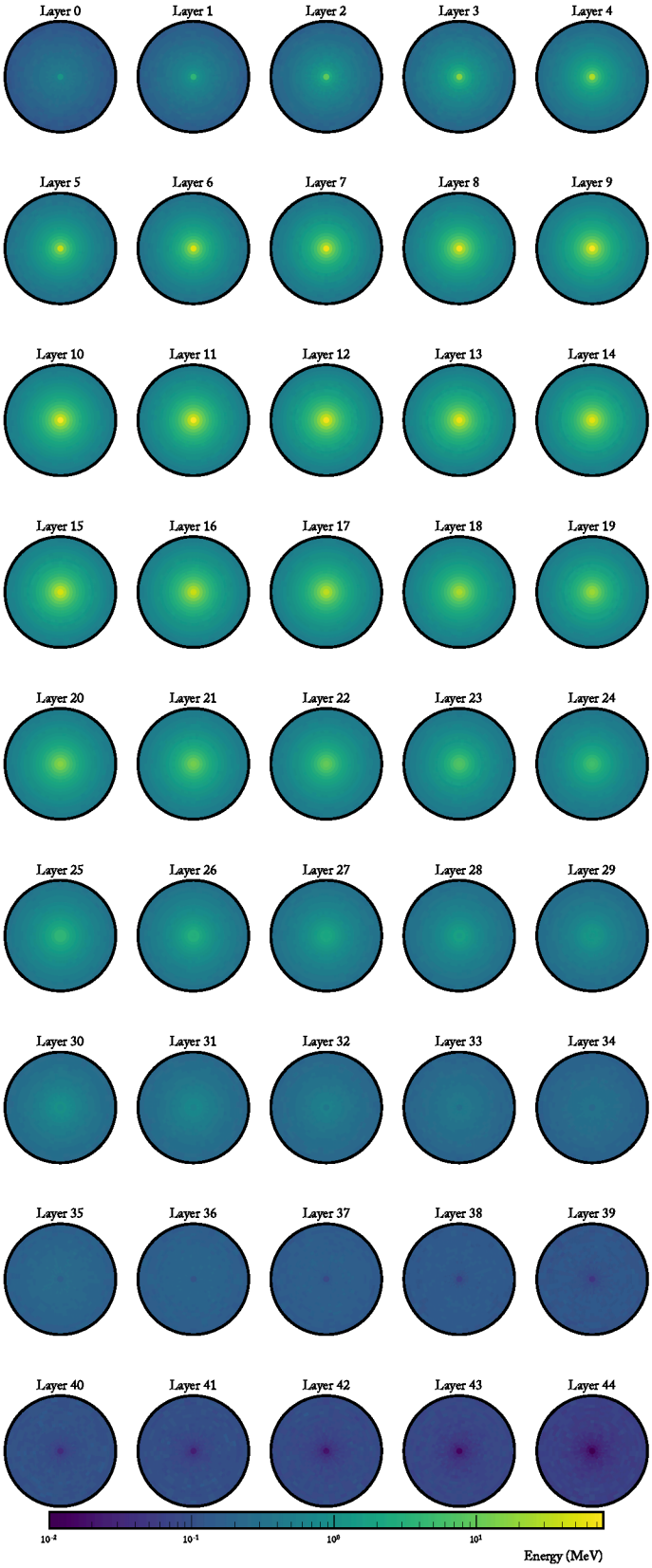

Figure 8.2: The average energy deposition of showers in all layers of z for Dataset 3 is illustrated. The figure has been reproduced with the code provided by the CaloChallenge (Giannelli et al. 2022a).

Table 8.2: Dimensions of the layers of the MLPs used globally by Encoder.

Layer	Input	1	2	Latent
Dimensions	1024	512	512	128

features is transformed. The transforms employ monotonic RQS, discussed in Section 6.1.1, as their univariate functions. Each RQS transform consists of eight knots per dimension, spread over the interval between -5 and 5. The parameters of these knots are determined by MLPs. These MLP take as input all non-transformed features along with n_{hits}, E_{sum}, and E_{in}. Each MLP comprises three hidden layers, with their dimensions listed in Table 8.3.

Table 8.3: Dimensions of the layers of the MLPs used in the LatentFlow that provide the RQS parameters.

Layer	Input	1	2	3	RQS param.
Dimensions	67	512	512	512	1472

For the generation process, the number of hits n_{hits} and the total energy E_{sum} must be known. Therefore, an additional normalizing flow, termed CondFlow, is incorporated to model the distribution $p(n_{\text{hits}}, E_{\text{sum}}|E_{\text{in}})$, as illustrated in Figure 8.6. The CondFlow is structured as a coupling flow and utilizes twelve transforms. Each transform employs a monotonic RQS as its univariate function. In every transform, one of the variables is randomly selected for transformation. The conditional MLPs take as input the non-transformed variable along with the incident energy E_{in}. Each MLP consists of three hidden layers, with their dimensions listed in Table 8.4.

Table 8.4: Dimensions of the layers of the MLPs used in the CondFlow that provide the RQS parameters.

Layer	Input	1	2	3	RQS param.
Dimensions	2 or 3	128	128	128	(2 or 1) · 23

The decoder differs between the two models. In both models, it is based on a NF architecture and is therefore referred to as the *PointFlow*. The PointFlow represents the distribution $p(\mathbf{X}|z, n_{\text{hits}}, E_{\text{sum}}, E_{\text{in}})$.

8.3.1 *CaloPointFlow I*

In the point-wise NF PointFlow of the CaloPointFlow I architecture, all points are assumed to be sampled i.i.d.,

$$p(\mathbf{X}|z, n_{\text{hits}}, E_{\text{sum}}, E_{\text{in}}) = \prod_{i=1}^{n_{\text{hits}}} p(x_i|z, n_{\text{hits}}, E_{\text{sum}}, E_{\text{in}})\,. \tag{8.4}$$

This assumption significantly simplifies the modeling process, as it removes the need to model interactions between points. However, this assumption does not fully align with reality. For example, if one point has high energy, it influences the probability of other points in

the shower also having high energy. Despite this limitation, the effectiveness of this approach was investigated.

As discussed in Section 8.7, an additional issue arises from the lack of information exchange between points. Since the points do not have knowledge of each other's coordinates, multiple points may end up having identical coordinates.

The PointFlow is structured as a coupling flow, where each point is transformed individually through a series of twelve distinct transformations. Each transformation utilizes monotonic RQS as univariate functions with eight knots positioned between -5 and 5, and randomly transforms two of the four variables associated with each point. The parameters of these transformations are determined by MLPs. The dimensions of each MLPs are listed in Table 8.5. The input to each MLP network includes the remaining two features of each point, the latent representation z, n_{hits}, E_{sum}, and E_{in}.

Table 8.5: Dimensions of the layers of the MLPs used in the PointFlow of the CPF I architecture that provide the RQS parameters.

Layer	Input	1	2	3	RQS param.
Dimensions	133	128	128	128	46

8.3.2 *CaloPointFlow II*

As previously described, one limitation of the PointFlow model for the intended application is its assumption that the points are i.i.d.. To accurately model the physical processes, an architectural design is required that does not impose this constraint and allows for information exchange among the points as they undergo transformation. To address this challenge, a NF architecture called *DeepSetFlow* (DSF) is proposed.

To facilitate the exchange of information, a Deep Sets architecture is employed within a Coupling Flow. Each coupling layer within the DSF can be divided into two sections. The first section aggregates information, as illustrated in Figure 8.3. The second section serves as an information dissemination step, as shown in Figure 8.4.

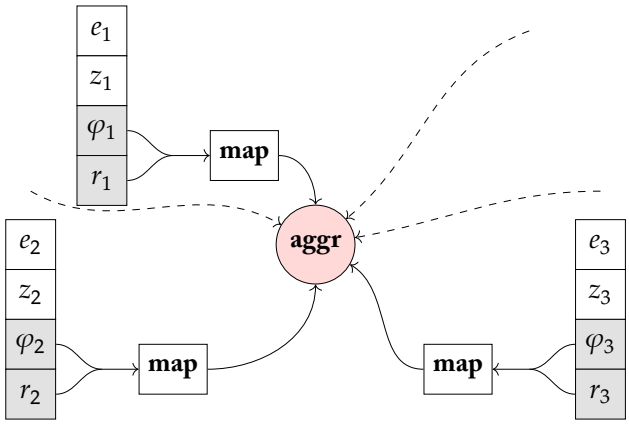

Figure 8.3: Illustration of the DeepSetFlow network aggregation step.

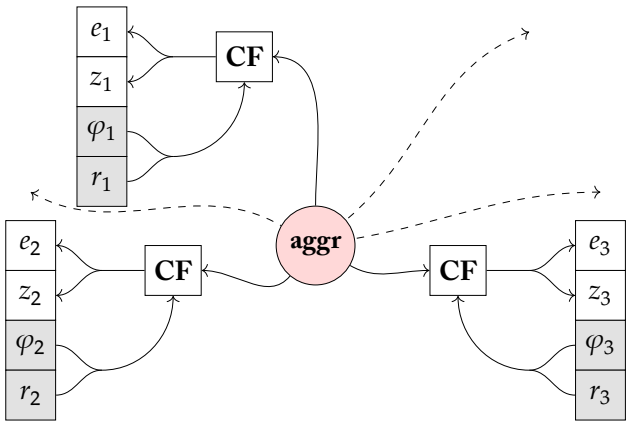

Figure 8.4: Illustration of the DeepSetFlow network dissemination step.

In this section, the architecture will be described along with the figures. The PointFlow model employs a point-wise CF that divides the features into two equal parts, with one half undergoing transformation based on the other half. To achieve permutation equivariance, each point must be treated identically. Consequently, half of the features of each point—the identical features for each point—are selected to transform the other half.

A Deep Sets approach is employed, whereby the features that are not transformed are mapped to a higher-dimensional space, as represented by the **map** box. The information is then aggregated, as indicated by the **aggr** circle, through the application of a mean pooling operation and further transformation of the aggregated features.

Table 8.6: Dimensions of the layers of the MLP used for the **map** of each point to a high-dimensional space.

Layer	Input	1	2	high-dim. space
Dimensions	2	64	128	512

Specifically, for the CPF II model, a **map** is chosen consisting of a MLP with the struc-

ture outlined in Table 8.6. The further transformation is also performed by a MLP, with dimensions outlined in Table 8.7.

Table 8.7: Dimensions of the layers of the MLP used for further transformation in the **aggr** step of the DSF.

Layer	Input	1	aggr. space
Dimensions	512	256	128

Subsequently, the objective is to disseminate the information. Here, the approach is similar to that employed with the PointFlow, where a point-wise CF is applied, symbolized by the **CF** box. However, what distinguishes this approach is that the CF also receives the aggregated information as an input, enabling an exchange of information between the points during the transformation. The pooled representation, combined with the non-transformed features and the conditional variables, serves as the input to the final MLP, which provides the parameters for the transformation in the CF architecture. The dimensions of this MLP are listed in Table 8.8.

Table 8.8: Dimensions of the layers of the MLPs used in the PointFlow of the CPF II architecture that provide the RQS parameters.

Layer	Input	1	2	3	RQS param.
Dimensions	133	128	128	128	46

Graphically, the architecture resembles a central node that gathers information from all nodes, distributes the information evenly, and is linear in the number of points.

The DSF, the PointFlow in the CPF II architecture, comprises twelve transformations, each employing a monotonic RQS univariate transformation with eight knots positioned between -5 and +5.

The new flow architecture is capable of capturing the point-to-point correlations. This approach shares similarities with previous work by Buhmann, Kasieczka, and Thaler 2023, who constructed the EPiC-GAN, a model that applied Deep Sets to create a permutation-equivariant generative model. Mikuni et al. 2023 also employed a similar technique of information aggregation within a diffusion model to learn point clouds.

In another instance, Käch and Melzer-Pellmann 2023 developed a model that combines all information into a single node, referred to as the mean field. Their model's main distinguishing factor from previous ones is the use of cross-attention to update the information of the mean field.

Finally, Liu et al. 2019 crafted a Graph NF. In creating a coupling layer for graphs, they suggested the same feature-splitting across points as in the current model.

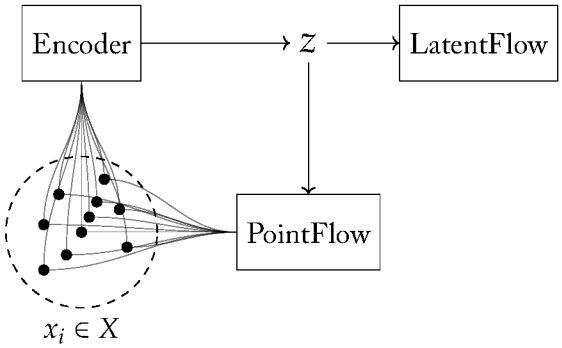

Figure 8.5: A schematic view of the models training setup. The *Encoder* encodes the points into the latent representation z. The *LatentFlow* is optimized to learn z, while the *PointFlow* is optimized to learn the distribution of points x_i conditioned on z.

8.4 Training Process

The data flow during the training process is visualized in Figure 8.5. The loss function of the CaloPointFlow models is based on the loss function derived for the PointFlow model, as detailed in Section 6.3. The loss function for the CaloPointFlow models is given by

$$\begin{aligned}\mathscr{L} = &\underbrace{\langle \log p(\mathbf{X}|z, n_{\text{hits}}, E_{\text{sum}}, E_{\text{in}}) \rangle_q}_{\mathscr{L}_{\text{recon}}} + \underbrace{\langle \log p(z|n_{\text{hits}}, E_{\text{sum}}, E_{\text{in}}) \rangle_q}_{\mathscr{L}_{\text{prior}}} \\ &+ \underbrace{\langle \log p(n_{\text{hits}}, E_{\text{sum}}|E_{\text{in}}) \rangle}_{\mathscr{L}_{\text{cond}}} - \mathscr{H}(q(z|\mathbf{X})).\end{aligned} \tag{8.5}$$

The loss function consists of four terms. The first term, $\mathscr{L}_{\text{recon}}$, represents the reconstruction error of the shower $\mathbf{X}$. Minimizing $\mathscr{L}_{\text{recon}}$ is equivalent to generating hits of the shower with a high likelihood. The second term, $\mathscr{L}_{\text{prior}}$, is the expectation value of the prior distribution conforming to the approximated distribution of the encoder. Minimizing $\mathscr{L}_{\text{prior}}$ ensures that the latent vector z is generated with a high probability. The third component, $\mathscr{L}_{\text{cond}}$, represents the loss of the CondFlow. Although the CondFlow could be trained separately, it is trained alongside the entire model. The final term, $\mathscr{H}(q(z|\mathbf{X}))$, represents the entropy of the encoded values and acts as a regularization term on the latent distribution. The entropy is calculated as described in Equation (6.28).

In each training step, the data were preprocessed as described in Section 8.6. The entire point cloud was encoded in z. The entropy and likelihoods of all flows were then computed. The parameters of the models were updated using the Adam optimizer (Kingma & Ba 2014). The models were trained for 100 epochs.

The training process is identical for both models.

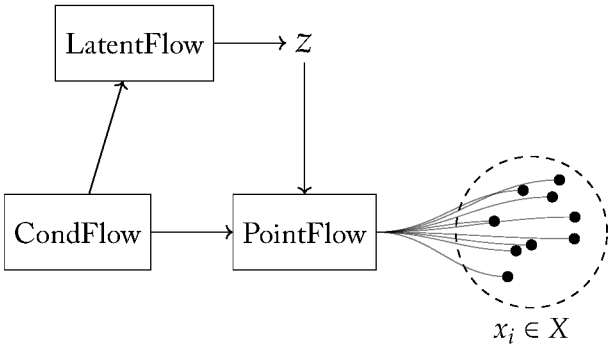

Figure 8.6: A schematic view of the models sampling setup. First, the conditional variables E_{sum} and n_{hits} are generated. Then, the *LatentFlow* generates the latent representation z, which is used to generate the points with the *PointFlow*.

8.5 Sampling Process

The sampling process for the CaloPointFlow models is illustrated in Figure 8.6. Initially, the sum of energy E_{sum} and the number of hits n_{hits} are sampled by the CondFlow. Subsequently, the latent variable z is sampled by the LatentFlow. The PointFlow then generates the shower $\mathbf{X}$, which consists of n_{hits} points, as determined by the CondFlow. Each generated point undergoes post-processing, as described in Section 8.7.

The sampling process is identical for both models.

8.6 Pre-Processing

In the pre-processing stage, the dataset is formatted specifically for the models. As previously mentioned, the provided voxelized dataset is converted into a point cloud dataset. Each shower is represented by a combination of energy values e and coordinates (r, α, z), with all points for which $e > 0$ included in the point cloud. Consequently, the resulting data structure comprises lists of four-dimensional points of varying cardinalities, depending on the number of hits n_{hits} in the shower.

The transformation of coordinates differs between the models discussed in subsequent sections. However, the energy transformation and conditional transformation are consistent across both models.

Initially, each energy value e is divided by the total energy to obtain fractions, yielding values between zero and one. These values are then min-max scaled and subjected to a logit transformation, which spreads the values over the interval $(-\infty, \infty)$. This processing follows the approach described by Krause and Shih 2021.

For the conditional variables, n_{hits} is dequantized by adding uniformly sampled noise between zero and one. These values are then divided by the square root of the incident energy, based on the observation that the number of hits scales with the square root of the inci-

dent energy. Subsequently, a log transformation is applied. Finally, the number of hits is normalized to achieve a mean of zero and a standard deviation of one.

A similar transformation is applied to E_{sum}. Initially, it is divided by E_{in}, followed by a log transformation and normalization. The incident energy E_{in} is log-transformed and normalized.

8.6.1 *CaloPointFlow I*

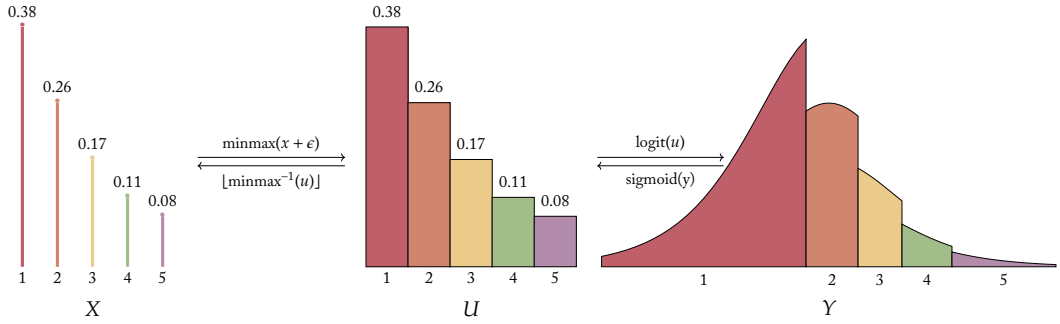

Figure 8.7: Visualization of an example for the classical dequantization process.

The generative models developed in Chapter 6 were primarily designed for continuous distributions or distributions in continuous spaces. However, the coordinate positions in (z, α, r) are discrete values. The models presented thus far are not suitable for learning discrete values, as they would treat them as delta distributions in space, making it challenging to learn the different volumes of the delta peaks. Therefore, a strategy is required to effectively handle these discrete values.

The strategy employed is referred to in the literature as dequantization (Dinh et al. 2017; Salimans et al. 2017; Uria et al. 2014).

The dequantization and its inverse process are visualized in Figure 8.7. The objective is to reversibly lift the discrete distribution to one defined on a continuous space. To illustrate, consider a discrete distribution X with the indices 1 to 5. For this example, assume that the *Probability Mass Function* (PMF) is

$$p(x_i) = \pi_i = [0.38, 0.26, 0.17, 0.11, 0.08]. \tag{8.6}$$

The first step in dequantization is to add uniformly distributed noise, $\epsilon \sim U(0,1)$ (Uria et al. 2014). Next, the values are min-max scaled to constrain them within $[0,1]$. To expand them to the full space, the logit transformation is applied (Dinh et al. 2017) to the values, resulting in Y, thus transforming the support from the interval $[0,1]$ to $(-\infty, \infty)$.

The continuous distribution Y can then be learned using one of the models described above. For generation, a sample from Y is obtained from the model, and the dequantization process is inverted by first applying the sigmoid transformation, which is the inverse of the logit transformation, followed by inverting the min-max scaling, and finally applying a floor operation. The dequantization and its inverse process are visualized in Figure 8.7.

One issue that arises is that if the probability masses of different indices differ, this leads to different heights in the resulting distribution hypercubes, resulting in a non-smooth distribution. This makes it difficult for the model to learn these discontinuous areas. To address this, Ho et al. 2019 introduced variational dequantization. Instead of assigning uniform noise, this technique incorporates an additional normalizing flow that learns and applies structured, non-uniform noise to the data, effectively increasing the modeling complexity.

For the pre-processing stage of the CaloPointFlow I model, the coordinate positions in (z, α, r) are dequantized as described above. Uniform noise $u \sim U(0,1)$ is added to each dimension of the discrete index value, and the values are divided by the size of the cells in that dimension. This ensures that all values fall between zero and one.

In initial attempts, difficulties were encountered in modeling the symmetry around α. This was due to the fact that, in the datasets, one value of α is assigned to the first index, and the last index is adjacent to the first. This posed a challenge for the dequantization strategy, as applying the logit transformation would result in the last index being spread to $+\infty$ and the first index being spread to $-\infty$.

To address this issue in the modeling process, a coordinate transformation is performed from the dequantized (α, r) coordinates in polar form to Cartesian coordinates. The values are then shifted and scaled as follows:

$$x' = \frac{x}{2} + \frac{1}{2}, \tag{8.7}$$

$$y' = \frac{y}{2} + \frac{1}{2}. \tag{8.8}$$

Subsequently, a logit transformation is applied, and the values are normalized.

8.6.2 *CaloPointFlow II*

In the pre-processing stage for the CaloPointFlow II model, the coordinates are not transformed to Cartesian form. Instead, the rotational symmetry of the calorimeter is leveraged, and the points are generated without an α-component. Therefore, in the pre-processing step, the α-component of each shower is removed, as explained in Section 8.7.2.

For the remaining coordinates, a newly developed dequantization strategy called CDF-Dequantization is employed. This method was developed for this thesis and is published in Schnake et al. 2024.

In typical circumstances, normalizing flows transform complex distributions into a normal distribution. The proposed approach here is to dequantize all discrete distributions into a normal distribution, ensuring that the marginal distributions are inherently aligned while allowing the model to learn correlations between features.

In Section 6.1, the transformation of one-dimensional distributions was discussed. Specifically, how a given distribution A can be transformed into another distribution B. This transformation is achieved by applying the CDF of A, followed by the inverse CDF of B, $F_B^{-1} \circ F_A$. For a bijective mapping between two distributions, access to both the CDF and inverse CDF of each distribution is required.

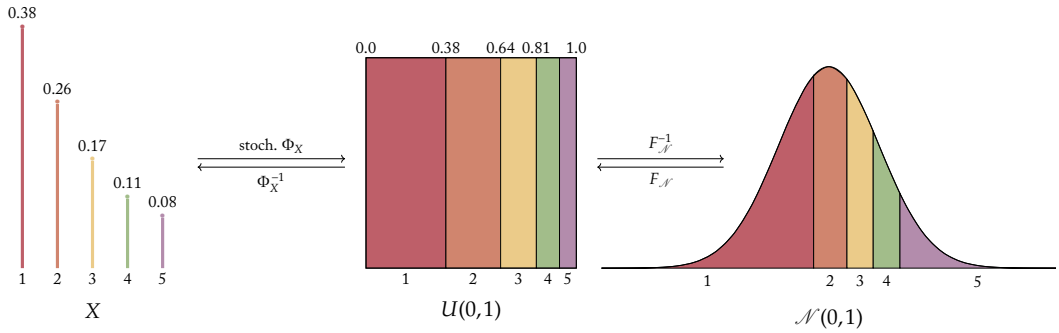

Figure 8.8: Visualization of an example for the CDF-Dequantization process.

The sigmoid and logit functions are the CDF and inverse CDF of a logistic distribution, respectively. However, it is evident from Figure 8.9 that the logistic and normal distributions exhibit notable differences in their tails.

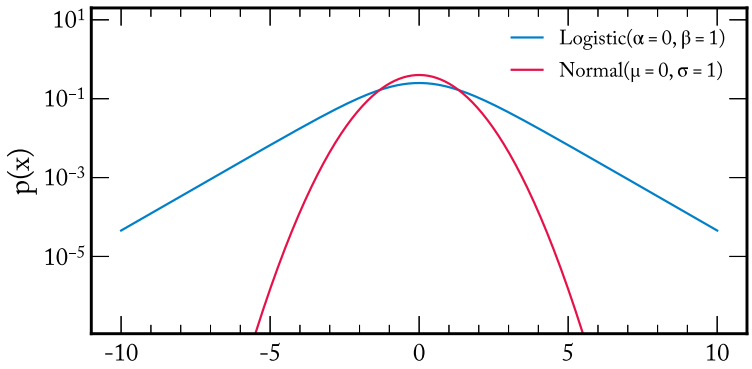

Figure 8.9: Comparison of the logarithmic declines of the logistic and normal distributions.

The CDF of a standard normal distribution is given by

$$F_{\mathcal{N}}(x) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right], \tag{8.9}$$

where $\operatorname{erf}(x)$ is the error function. Conversely, the quantile function of the standard normal distribution, often referred to as the probit function, is

$$F_{\mathcal{N}}^{-1}(y) = \sqrt{2}\operatorname{erf}^{-1}(2y - 1), \tag{8.10}$$

where $\operatorname{erf}^{-1}(x)$ is the inverse error function.

By substituting the standard CDF and probit function with the sigmoid and logit combination, $U(0,1)$ can be directly transformed into the normal distribution. The remaining task is to transform the discrete distribution into $U(0,1)$.

The inverse process will be addressed initially, which transforms $U(0,1)$ into a discrete distribution. If the PMF is available, the interval $[0,1]$ can be divided into discrete intervals

based on the PMF, and each sampled value $u \sim U(0,1)$ can be discretized according to the index of the interval it falls into. This forms a surjective mapping where intervals in $[0,1]$ are associated with specific discrete values. However, this mapping is not injective and therefore not bijective or directly invertible.

Nielsen et al. 2020 demonstrated how VAEs, normalizing flows, and surjective mappings can be integrated into a unified framework. They showed that surjective mappings can be used if a sufficient stochastic inverse is found. Within this framework, Nielsen et al. demonstrated that adding uniform noise can be interpreted as a stochastic inverse of the floor operation, denoted as $\lfloor x \rfloor$. This implies that $p(x|z) = \delta_{x,\lfloor z \rfloor}$. Its stochastic inverse, $q(z|x)$, has support in $\mathscr{B}(x) = \{x + u \mid u \in [0,1]\}$.

In our specific case, the PMF can be described as $p(x_i|u) = \delta(F_X(x_i) \leq u \leq F_X(x_{i+1}))$, which is a delta function that triggers if u is located in the interval $[F_X(x_i), F_X(x_{i+1})]$. Its stochastic inverse, $q(u|x_i)$, has support over

$$\begin{aligned}\mathscr{B} &= \{u|F_X(x_i) \leq u \leq F_X(x_{i+1})\} \\ &= \{F_X(x_i) + u|0 \leq u \leq F_X(x_{i+1}) - F_X(x_i)\} \\ &= \{F_X(x_i) + u|0 \leq u \leq p_i\} \\ &= \{F_X(x_i) + u \cdot p_i|u \in [0,1]\}\end{aligned} \tag{8.11}$$

Therefore, the stochastic mapping can be composed as $\Phi_X(x_i) = F_X(x_i) + u \cdot p_i$, where $u \sim U(0,1)$.

If all p_i are available, this transformation provides a mapping from discrete data to a uniform distribution. The values of p_i can be approximated by counting the frequencies of x_i in the data, thereby obtaining a straightforward approximation.

In conclusion, a relatively simple transformation has been devised to map discrete data to a normal distribution. This considerably simplifies the task of the normalizing flow, which is now solely responsible for learning the correlations in the data, rather than the general shape. For a visual representation, refer to Figure 8.8.

The algorithms for both directions of the CDF-Dequantization are provided below.

Algorithm 1 Forward Transformation ($X \rightarrow \mathscr{N}$)

for $x_i \in X$ **do**
  sample $u \sim U(0,1)$
  $y_i = F^{-1}_{\mathscr{N}}(\text{CDF}(x_i) + \text{PDF}(x_i) \cdot u)$
end for
Return $Y = \{y_1 \dots y_n\}$

Algorithm 2 Inverse Transformation ($\mathscr{N} \rightarrow X$)

for $y_i \in Y$ **do**
  $u_i = F_{\mathscr{N}}(y_i)$
  $x_i = \text{find first CDF} \geq u_i$
end for
Return $X = \{x_1 \dots x_n\}$

8.7 Post-Processing

In the post-processing stage, the generated data are transformed back to the voxelized format. Initially, the values sampled by the CondFlow are transformed by reversing the normalization and applying the exponential function. Specifically, E_{sum} is multiplied by E_{in}, and n_{hits} is multiplied by $\sqrt{E_{\text{in}}}$. The number of hits n_{hits} is discretized by applying the floor operation.

The post-processing of the points is specific to each individual model and is described in detail in the corresponding sections. Generally, the energy values are first unnormalized, followed by the application of a sigmoid function. The min-max scaling is then reverted, and the point clouds are transformed back into voxelized datasets. For each point, the voxel at the corresponding coordinate is assigned the point's energy value, while all other voxels are set to zero energy.

One significant challenge of the CaloPointFlow architecture is that the coordinates are not unique, allowing multiple points to have the same coordinates. This issue is a fundamental limitation of the point cloud approach.

In cases where multiple points have the same coordinate, one point is randomly selected, and all others are discarded. Although this strategy reduces the number of hits compared to the generated points, it addresses the problem to some extent. After assigning the energy values to the voxels, each value is scaled by E_{sum}. This scaling ensures that the total energy remains consistent despite the discarding of points.

8.7.1 *CaloPointFlow I*

In the post-processing stage, the coordinate values are initially unnormalized and subsequently transformed using a sigmoid function. The Cartesian coordinates are then converted back to polar coordinates as defined in Equations (8.2) and (8.3):

$$x = 2x' - 1, \tag{8.12}$$

$$y = 2y' - 1, \tag{8.13}$$

$$\alpha = \frac{\arctan2(y, x)}{2\pi}, \tag{8.14}$$

$$r = \sqrt{y^2 + x^2}. \tag{8.15}$$

The coordinates (r, α, z) are scaled in proportion to the dimensions of the dataset. A floor operation is then applied to each dimension to discretize the values.

8.7.2 *CaloPointFlow II*

In the post-processing stage for CPF II, the generated z and r values of the dataset are quantized as explained in Section 8.6.2. One of the main issues with the first model was its inability to enforce the constraint of only one hit per cell, due to PointFlow's lack of

awareness of the positions of other points. This issue is mitigated by allowing point-to-point information exchange within the DSF architecture.

To simplify the model's task, the constraint is relaxed by considering the showers to be symmetric around the z-axis in the α coordinates, resulting in a flat distribution in α. Ignoring any potential inner shower structure, random positions are assigned in α. If there are more points than available cells, these points are discarded. This allows for up to 16 points with the same (z, r) coordinates in Dataset II and up to 50 in Dataset III.

This approach confines the model to datasets with rotational symmetry, without accounting for any inner α distribution. While this constraint is acceptable for electron showers due to their homogeneity, it would not be suitable for hadronic showers. The improvements resulting from this approach will be discussed in the next section.

8.8 Evaluation

The evaluation of surrogate models in particle physics is a task of considerable complexity (Butter et al. 2019; Das et al. 2023; Kansal et al. 2023). This inherent complexity is also evident in calorimeter surrogate models, where no single metric can sufficiently assess model performance. In real calorimeters employed by experiments, performance evaluation often relies on high-level variables such as jet variables and b-tagging performances, which assist in evaluating the impact on analyses (ATLAS 2022). However, for the CaloChallenge datasets, access to such variables is limited, necessitating the estimation of performance based on more general criteria.

To evaluate the performance of the CPF models, a set of metrics has been defined. Specifically, the simulated showers generated by Geant4 are compared to those generated by both CaloPointFlow models in order to assess the quality of the CPF samples. The objective is for the CPF samples to be indistinguishable from the Geant4 samples.

All models are inherently uncertain. In the absence of direct access to these uncertainties, the bootstrapping method (Efron 1979) is employed to estimate errors. Bootstrapping is a statistical method used to estimate the distribution of a metric by resampling the data with replacement, assuming the dataset is representative of the entire distribution. Multiple bootstrap samples are generated from the original dataset, each created by randomly selecting data points with replacement.

The generated and simulated datasets each consist of 100,000 showers. For each simulation model, 1,000 bootstrapped datasets are sampled, each containing 100,000 showers. The average value and standard deviation for the considered metrics are then measured and reported. When dealing with comparison metrics calculated on two datasets, a combination of 1,000 bootstrapped samples is used to obtain the values for the metrics.

To estimate the internal uncertainty of the Geant4 simulation and to identify potential avenues for improvement, a comparison of 1,000 combinations of Geant4 bootstrap samples is conducted. This approach allows for an estimate of the internal uncertainty associated with the finite Geant4 sample.

One of the difference measures used in the literature is the *Normalized Difference* (ND):

$$\mathrm{ND}(v_a, v_b) = \frac{v_a - v_b}{v_a + v_b}, \tag{8.16}$$

which is a dimensionless quantity used to measure the relative difference between two values, v_a and v_b. It is particularly useful when values have different scales, as is often the case with the quantities investigated. The ND is employed in a variety of disciplines, including remote sensing and ecology (Robinson et al. 2017).

The use of the ND is preferred over the relative difference $(v_a - v_b)/v_b$ due to the absence of true values in our simulations, making a symmetric difference measure a more suitable choice. Additionally, ratios such as v_a/v_b are avoided, as they introduce different scaling for values below and above 1. The ND is used in the following sections for comparing histograms of models and data.

To ascertain the discrepancy between two given histograms, a χ^2 test is defined to compare two histograms (Cramér 1946; Gagunashvili 2007; Pearson 1904). The test hypothesis is that of homogeneity. The two histograms, with entries $h_{1,i}$ and $h_{2,i}$, have indices i representing the different bins. The total number of entries are $H_1 = \sum_{i=1}^{n} h_{1,i}$ and $H_2 = \sum_{i=1}^{n} h_{2,i}$ for n different bins. If both histograms originate from the same distribution with frequencies p_i, the best likelihood estimator of p_i is

$$\hat{p}_i = \frac{h_{1,i} + h_{2,i}}{H_1 + H_2}.$$

Therefore, one can define:

$$\begin{aligned} X^2 &= \sum_{i=1}^{n} \frac{(h_{1,i} - H_1\hat{p}_i)^2}{H_1\hat{p}_i} + \sum_{i=1}^{n} \frac{(h_{2,i} - H_2\hat{p}_i)^2}{H_2\hat{p}_i} \\ &= \frac{1}{H_1 H_2} \sum_{i=1}^{n} \frac{H_2(h_{1,i} - H_1\hat{p}_i)^2 + H_1(h_{2,i} - H_2\hat{p}_i)^2}{\hat{p}_i} \\ &= \frac{1}{H_1 H_2} \sum_{i=1}^{n} \frac{H_2^2 h_{1,i}^2 - 2H_1 H_2 h_{1,i} h_{2,i} + H_1^2 h_{1,i}^2}{h_{1,i} + h_{2,i}} \\ &= \frac{1}{H_1 H_2} \sum_{i=1}^{n} \frac{\left(H_2 h_{1,i} - H_1 h_{2,i}\right)^2}{h_{1,i} + h_{2,i}} \end{aligned} \tag{8.17}$$

For our purposes, only histograms with the same cardinality, $H_1 = H_2$, are compared. In this case, Equation (8.17) simplifies to:

$$X^2 = \sum_{i=1}^{n} \frac{(h_{1,i} - h_{2,i})^2}{h_{1,i} + h_{2,i}}. \tag{8.18}$$

X^2 follows approximately a χ^2_{n-1} distribution (Cramér 1946) since the total number of entries is fixed. Equation (8.18) is also known as the triangular discrimination or harmonic mean divergence in the statistics literature (Topsoe 2000). In some high-energy physics literature, it is referred to as the Separation Power (Diefenbacher et al. 2020; Nachman 2016). This measure has the advantage of being a distance measure ranging between 0 and 1, where 1 indicates no overlap between the histograms, and 0 denotes total agreement.

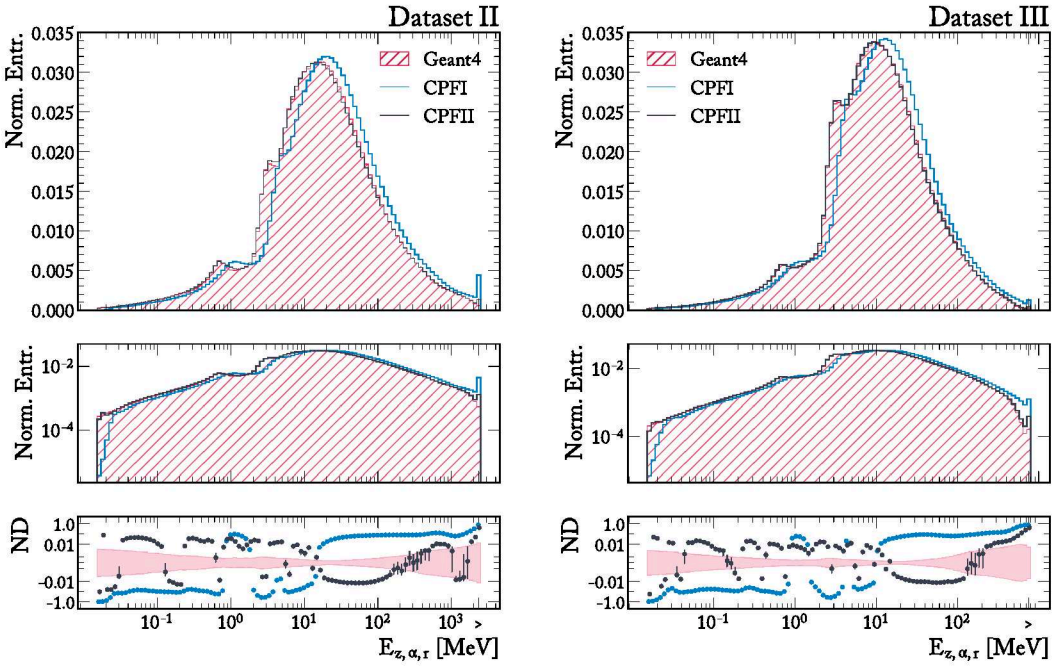

Figure 8.10: Cell energy distributions for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I and CPF I models. The upper graphs show the normalized entries of the cell energy values, binned logarithmically. The entries are normalized to sum to 1. The middle graphs show the same data as the upper graphs, but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models andGeant4. The red band shows the in-sample uncertainty of Geant4.

8.8.1 *Cell Energy*

The description of each calorimeter shower is defined in terms of the energy values in the cells of the calorimeter, denoted as

$$E_{i,z,\alpha,r}. \tag{8.19}$$

Here, i represents the shower index in the dataset, ranging from 1 to $n = 100,000$. For both datasets, the longitudinal index z ranges from 1 to $n_z = 45$. The angular index α ranges from 1 to $n_\alpha = 16$ for Dataset II and $n_\alpha = 50$ for Dataset III. The radial index r ranges from 1 to $n_r = 9$ for Dataset II and $n_r = 18$ for Dataset III.

The marginal distribution of all $E_{i,z,\alpha,r}$ values is illustrated in Figure 8.10. The values are binned between the cutoff energy of $E_{\text{min}} = 15.15\,\text{keV}$ and the upper limits, which are defined as 2500 MeV for Dataset II and 800 MeV for Dataset III. These bounds are set such that approximately 99.99% of all energy values fall below them. An overflow bin is included to capture all energy values above these thresholds. To evaluate the performance of the CPF models, a set of metrics has been established. A logarithmic binning approach is employed to accommodate the extensive range of values, with the entries subsequently normalized so that the sum of all 100 bins is equal to 1. The left side of the figure presents the results for Dataset II, while the right side shows the results for Dataset III. Given the considerable number of entries, the error bars on the normalized entries are negligible and not visible.

For CPF I, the values are shifted with respect to Geant4, which is not the case for CPF II.

This discrepancy can be attributed to the occurrence of double hits, which will be discussed in detail in Section 8.8.3. CPF II accurately models the cutoff energy due to the displacement of energy values, ensuring that no values fall below the cutoff. Additionally, CPF II has significantly fewer entries in the overflow bin compared to CPF I. The normalized difference band for Geant4 data is very narrow, and the CPF values do not achieve optimal agreement. This is further evidenced by the X^2 in Table 8.9, where CPF II results are two orders of magnitude higher than the possible mean values from Geant4, while CPF I results are as much as four orders of magnitude away.

Table 8.9: X^2 for all three models in both datasets of the histograms shown in Figure 8.10.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	0.26 ± 0.16	2150 ± 20	59.4 ± 1.3
III	0.18 ± 0.14	2340 ± 20	25.7 ± 0.8

8.8.2 *Relative Energy Sum*

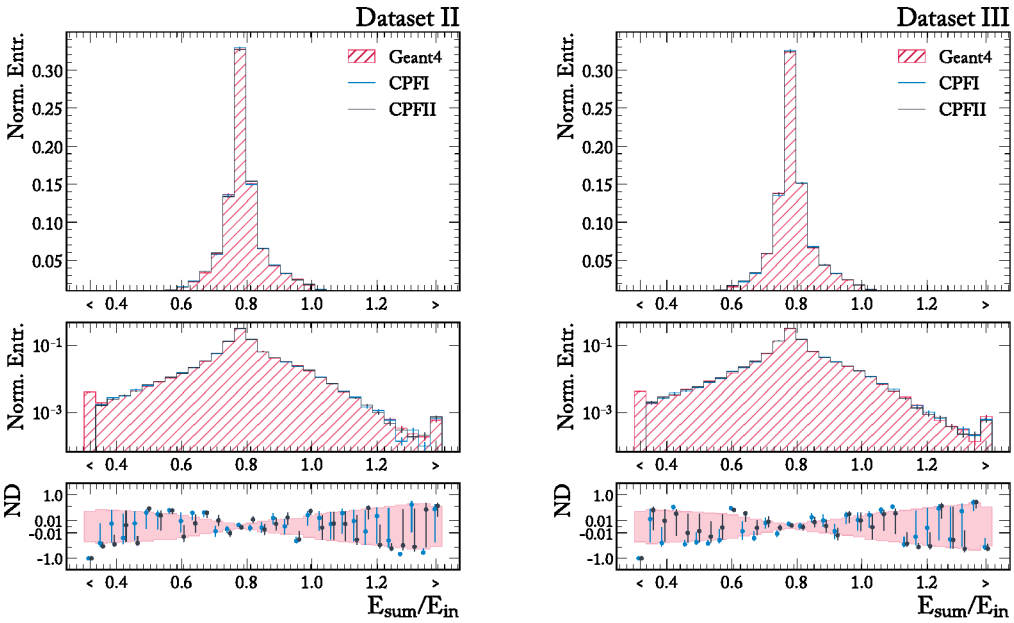

Figure 8.11: Relative energy sum distributions for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show the normalized entries of the energy sum, linearly binned. The entries are normalized so that the sum of all bins equals 1. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The sum of energy $E_{i,\text{sum}}$ is calculated by summing over the indices:

$$E_{i,\text{sum}} = \sum_{z=1}^{n_z} \sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} E_{i,z,\alpha,r}. \tag{8.20}$$

In Figure 8.11, a histogram of the energy sum divided by the incident energy is shown:

$$\frac{E_{i,\text{sum}}}{E_{i,\text{in}}}. \tag{8.21}$$

This division is performed because the energy sum scales approximately linearly with the incident energy. As a result, the resulting quantity is largely independent of the incident energy.

The histograms for both Dataset II (left) and Dataset III (right) in Figure 8.11 demonstrate that the energy sums are consistent between the datasets, differing only due to statistical fluctuations resulting from their different granularity. Since both CPF I and CPF II use a CondFlow with identical architecture, no significant differences are observed between them. Overall, the histograms do not exhibit substantial discrepancies between the Geant4 simulations and the CPF models.

Upon examining the ND, all values from the CPF models have uncertainties that overlap with the uncertainty band of Geant4, as shown in the lower halves of the figures. In Table 8.10, the X^2 for the histograms in Figure 8.11 is presented. The CPF models exhibit a X^2 nearly an order of magnitude greater than that of the Geant4 dataset, suggesting potential areas for improvement. The limitations in generation quality are attributed to the dataset size.

Table 8.10: X^2 for all three models in both datasets of the histograms shown in Figure 8.11.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	30.3 ± 7.7	208 ± 18	214 ± 19
III	30.5 ± 8.0	229 ± 20	198 ± 18

8.8.3 *Number of Hits*

The number of hits $n_{i,\text{hits}}$ quantifies the number of voxels in the calorimeter with an energy value greater than zero. Formally, it is defined as

$$\text{hit}\left(E_{i,z,\alpha,r}\right) = \begin{cases} 1 & \text{if } E_{i,z,\alpha,r} > E_{\text{min}} \\ 0 & \text{else} \end{cases} \tag{8.22}$$

$$n_{i,\text{hits}} = \sum_{z=1}^{n_z} \sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} \text{hit}\left(E_{i,z,\alpha,r}\right). \tag{8.23}$$

Therefore, $\max(n_{i,\text{hits}})$ is equal to the total number of voxels in the calorimeter dataset.

Figure 8.12 presents the distribution of n_{hits}. The histogram bins are linearly spaced between 0 and the 99.99th percentile of the distribution, with an additional overflow bin capturing values above this threshold. The distribution of n_{hits} peaks at a low value for both datasets and then decreases until reaching an upper limit of approximately 5200 for Dataset II and 17000 for Dataset III. There are very few events with values above these limits.

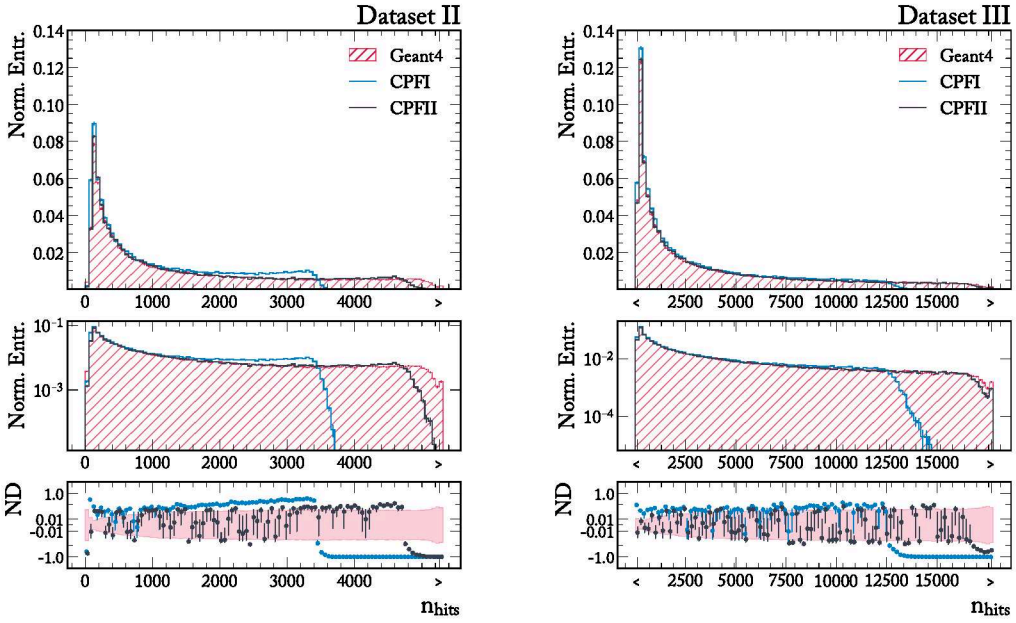

Figure 8.12: Number of hits distributions for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show the normalized entries of the number of hits, linearly binned. The entries are normalized so that the sum of all bins equals 1. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

Both models reproduce the general shape of the distribution; however, CPF I does not achieve the higher values observed in the Geant4 data. Similarly, CPF II underestimates the number of hits for Dataset II, although it performs better with the larger Dataset III. This discrepancy arises from the occurrence of double hits in the CPF models, where multiple points share the same coordinate. For both models, only one value per coordinate is retained, and all other points with the same coordinate are discarded, resulting in a lower number of hits.

In contrast, the CPF II model generates only the z and r coordinates and assigns the α index randomly, allowing more possibilities and reducing the likelihood of discarding points. Points are discarded only when n_α indices with the same (z, r) coordinates are reached. Since n_α is significantly higher in Dataset III, fewer points are discarded. In addition, CPF II incorporates a point-to-point information exchange mechanism, allowing points to communicate their positions and align accordingly, which potentially improves model performance.

Table 8.11: X^2 for all three models in both datasets of the histograms shown in Figure 8.12.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	98.2 ± 13.7	18800 ± 200	2720 ± 70
III	98.6 ± 13.9	7830 ± 100	304 ± 33

Point cloud models are optimal for sparse data, as expected from high granularity calorimeters. Dataset II, however, is not sparse and therefore does not represent a typical example. The CPF II model benefits from this difference, resulting in clearly improved performance.

The number of hits also provides insight into the observed shift in the cell energy distribution. Both the number of points generated and the energy sum of the shower are produced by CondFlow, and during post-processing, all points are scaled to the energy sum. A reduction in the number of points results in excessive scaling of the remaining points, thereby altering the fractional distribution and explaining the shift in the energy distribution.

8.8.4 *Average Energy Sum*

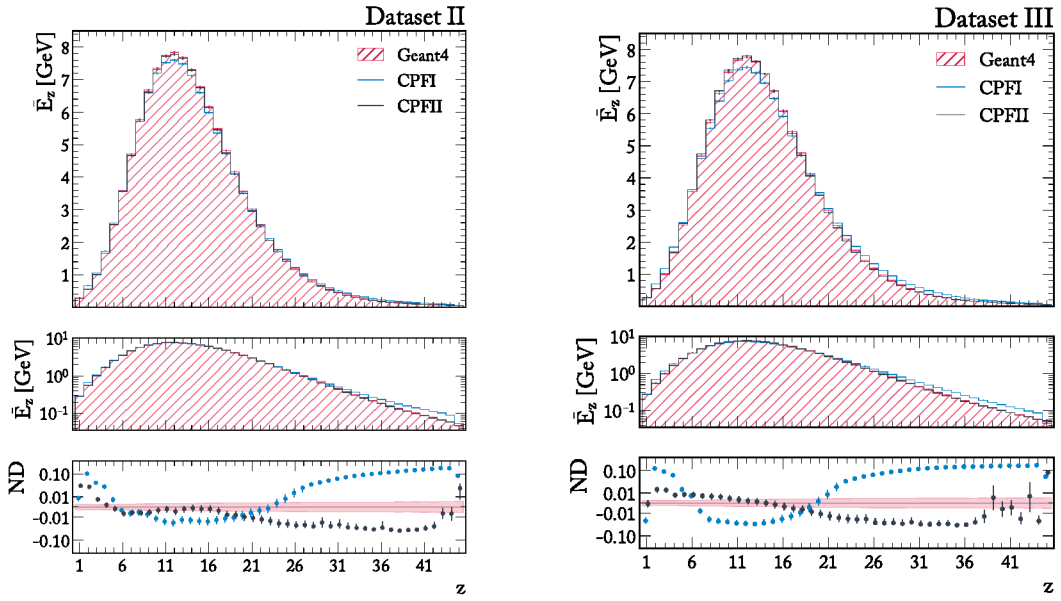

Figure 8.13: Average energy in each longitudinal layer z for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show the average energy values in MeV. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The total energy in a longitudinal layer z is defined as

$$E_{i,z} = \sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} E_{i,z,\alpha,r}. \tag{8.24}$$

The average energy in a layer z is then computed as

$$\bar{E}_z = \frac{1}{n} \sum_{i=1}^{n} E_{i,z}. \tag{8.25}$$

The average energy sum in each longitudinal layer z for both datasets is presented in Figure 8.13. The average shower energies closely approximate a gamma distribution, and both CPF models follow this distribution pattern. However, CPF I shows a greater deviation from Geant4 compared to CPF II, with this discrepancy becoming more pronounced in the

second dataset. This deviation cannot be sufficiently explained by the uncertainties on the average energies.

When examining the ND values, it is evident that CPF II performs significantly better in the first and last indices, which correspond to regions of lower average energies. In the high-energy region, specifically between indices 11 and 16, CPF II values align more closely with the Geant4 error band. This suggests that the second model more accurately captures the overall average longitudinal energy distribution.

The improved performance of CPF II is attributed to the utilization of the CDF-Dequantization method, as described in Section 8.6.2. This method facilitates a more precise modeling of the different indices within the datasets.

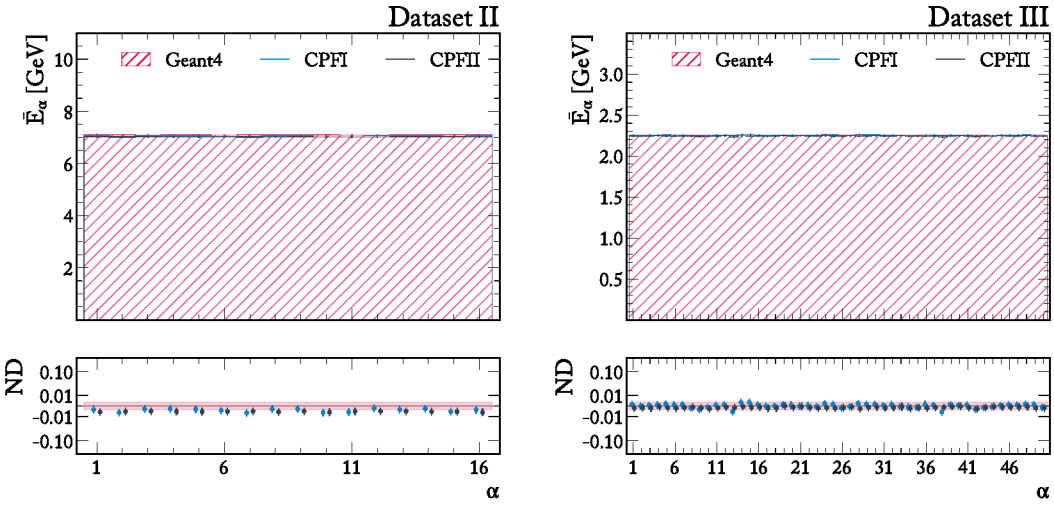

Figure 8.14: Average energy in each angular layer α for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show the average energy values in MeV. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The total energy in each angular layer α is defined as

$$E_{i,\alpha} = \sum_{z=1}^{n_z} \sum_{r=1}^{n_r} E_{i,z,\alpha,r}.$$

The average energy sum in each angular layer α for both datasets is computed as

$$\bar{E}_\alpha = \frac{1}{n} \sum_{i=1}^{n} E_{i,\alpha}.$$

This quantity for both datasets is presented in Figure 8.14. Since the showers should be invariant under rotations around α, the energy distribution is expected to be flat. Both models exhibit this feature. Originally, the CPF I model encountered issues in modeling, which necessitated switching to Cartesian coordinates. As observed, the distributions now match. For the CPF II model, α coordinates are randomly assigned, ensuring a flat distribution by construction. For Dataset II, the ND plots indicate a slight shift in the generated data. In

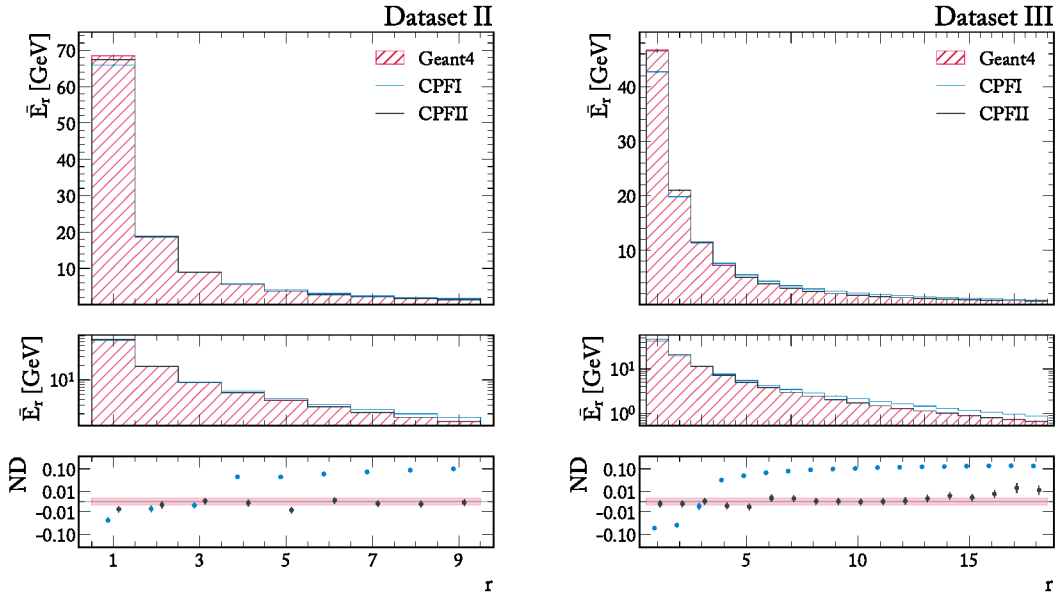

Figure 8.15: Average energy in each radial layer r for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show the average energy values in MeV. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

the third dataset, there is nearly perfect agreement between the averages and uncertainty in ND, leaving minimal room for improvement.

The total energy in each radial layer r is defined as

$$E_{i,r} = \sum_{z=1}^{n_z} \sum_{\alpha=1}^{n_\alpha} E_{i,z,\alpha,r}.$$

In Figure 8.15, the average energy sum in the radial layers r is presented as

$$\bar{E}_r = \frac{1}{n} \sum_{i=1}^{n} E_{i,r}.$$

The energy distribution decreases with increasing r, and both CPF models capture this trend. The CPF I model exhibits lower energy in the first index and an overshoot at the higher indices, which is clearly visible in the ND plots and is even more pronounced in the third dataset. For the CPF II model, only minor discrepancies are observed in the first index, with most ND values aligning well with the Geant4 data. This demonstrates the improved performance due to the CDF-Dequantization method.

8.8.5 *Energy Sum Distribution*

The analysis of the energy sums in each index (z, α, and r) has thus far focused exclusively on the mean values, leaving the full marginal distribution of E_z, E_α, and E_r unexplored. Each index value is associated with a distinct marginal energy distribution, and it is imperative to

evaluate the accuracy with which these marginal distributions are modeled. Although the individual histograms of these marginals will not be analyzed or presented in this section, they will be provided in Chapter B for reference.

Instead, the X^2 is calculated for each histogram, and the results are combined into a single graph. The binning for each histogram consists of 50 logarithmically spaced bins covering the interval between the 0.01th percentile and the 99.99th percentile, with additional underflow and overflow bins beyond these limits. Uncertainty is estimated by calculating the X^2 over 1,000 bootstrapped samples. For instance, in Figure 8.16, a total of $2 \times 3 \times 45 \times 1,000 = 270,000$ histograms were computed. A lower separation power indicates better performance. To establish a lower bound for the X^2, it is computed between different bootstrapped Geant4 histograms, which helps quantify the potential for further improvement.

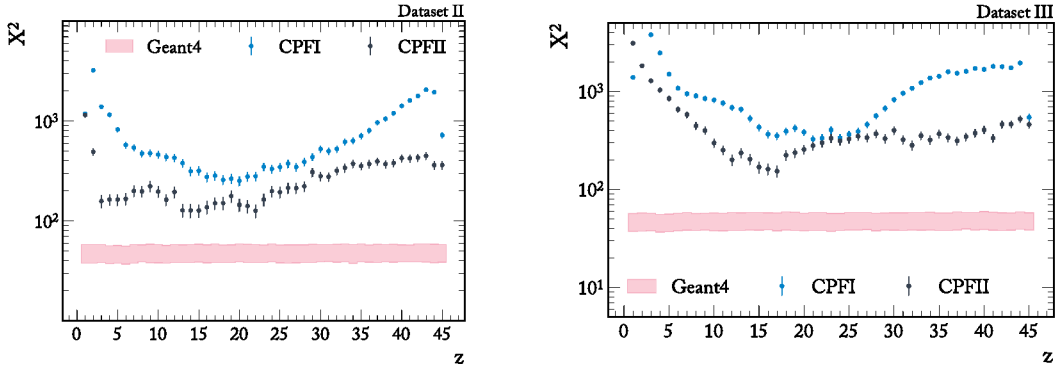

Figure 8.16: X^2 of the energy histograms in each longitudinal layer z for Dataset II (left) and Dataset III (right). The histograms used to calculate the separation power are shown in Figures B.4 to B.6 for Dataset II and Figures B.7 to B.9 for Dataset III. The values for both CPF models are shown as blue and dark gray points, respectively, with error bars indicating the standard deviation of the separation power. The red error band illustrates the uncertainty in the Geant4 data.

In Figure 8.16, the separation power in the longitudinal layer z is presented. Overall, CPF II exhibits a lower separation power than CPF I, indicating better performance. By examining the averages in Figure 8.13, better separation is expected in the central region with higher energy values than in the tails. This behavior is observed for CPF I in both Dataset II and Dataset III. However, CPF II does not follow this pattern. In Dataset II, CPF II performs worst in the first and last indices, with no clear pattern in between. In Dataset III, the separation power for CPF II initially decreases and then remains relatively stable. Nevertheless, all separation powers are significantly higher than the Geant4 band, indicating room for improvement.

The X^2 in the angular dimension is presented in Figure 8.17, where no differences are visible in individual bins, as expected due to rotational symmetry. Interestingly, the CPF I model outperforms the CPF II model in this aspect. It is hypothesized that this discrepancy is due to the random assignment of α in CPF II.

Finally, Figure 8.18 shows the X^2 for the radial layers r. Here, CPF II demonstrates much lower separability from Geant4 compared to CPF I. In Dataset II, the separation power for CPF I remains mostly constant, while CPF II shows an initial decrease followed by stability. These values are relatively close to the Geant4 band. In Dataset III, both CPF I and CPF II start with similar separation powers, with CPF II slightly better. However, as the radial index

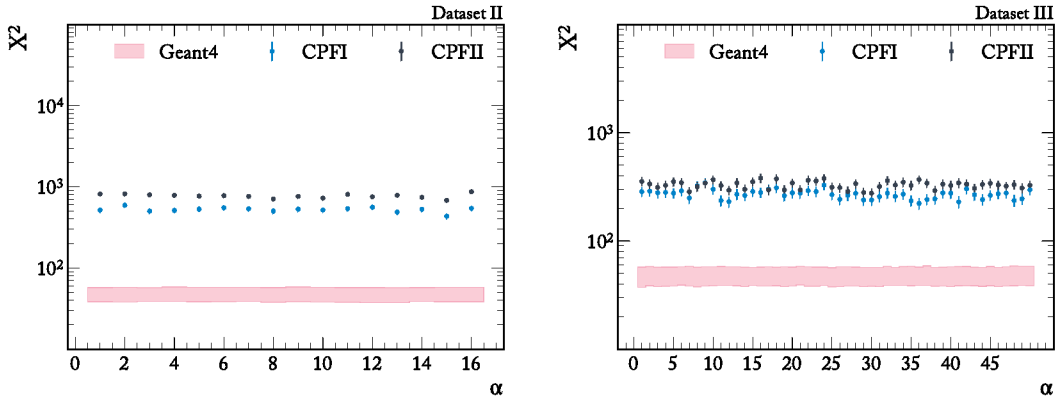

Figure 8.17: X^2 of the energy histograms in each azimuthal layer α for Dataset II (left) and Dataset III (right). The histograms used to calculate the separation power are shown in Figures B.1 and B.2 for Dataset II and Figures B.12 to B.15 for Dataset III. The values for both CPF models are shown as blue and dark gray points, respectively, with error bars indicating the standard deviation of the separation power. The red error band illustrates the uncertainty in the Geant4 data.

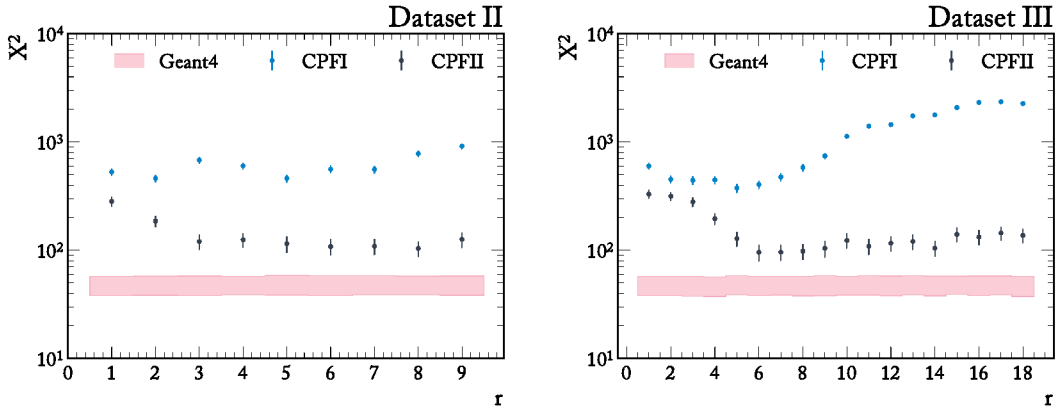

Figure 8.18: X^2 of the energy histograms in each longitudinal layer r for Dataset II (left) and Dataset III (right). The histograms used to calculate the separation power are shown in Figure B.3 for Dataset II and Figures B.10 and B.11 for Dataset III. The values for both CPF models are shown as blue and dark gray points, respectively, with error bars indicating the standard deviation of the separation power. The red error band illustrates the uncertainty in the Geant4 data.

increases, CPF II exhibits a decrease in separation power, while CPF I shows an increase.

Overall, CPF II provides a better model for the marginal energy sum distributions compared to CPF I, demonstrating its superior performance in this aspect.

8.8.6 *Shower Centers*

To gain a deeper understanding of the internal structure of the showers within the detector, the distribution of the shower center in the longitudinal z direction is analyzed. Specifically, the shower center in the longitudinal z direction is defined as

$$\mu_{i,z} = \frac{\sum_{z=1}^{n_z} z \cdot E_{i,z}}{E_{i,\text{sum}}}, \tag{8.26}$$

which represents the average z coordinate of all hits in the shower, weighted by their energy.

In Figure 8.19, a histogram of the shower centers μ_z in the longitudinal direction is presented, utilizing a linear binning scheme with 100 bins spanning from the 0.1th percentile to the 99.9th percentile of the distribution, and including both overflow and underflow bins.

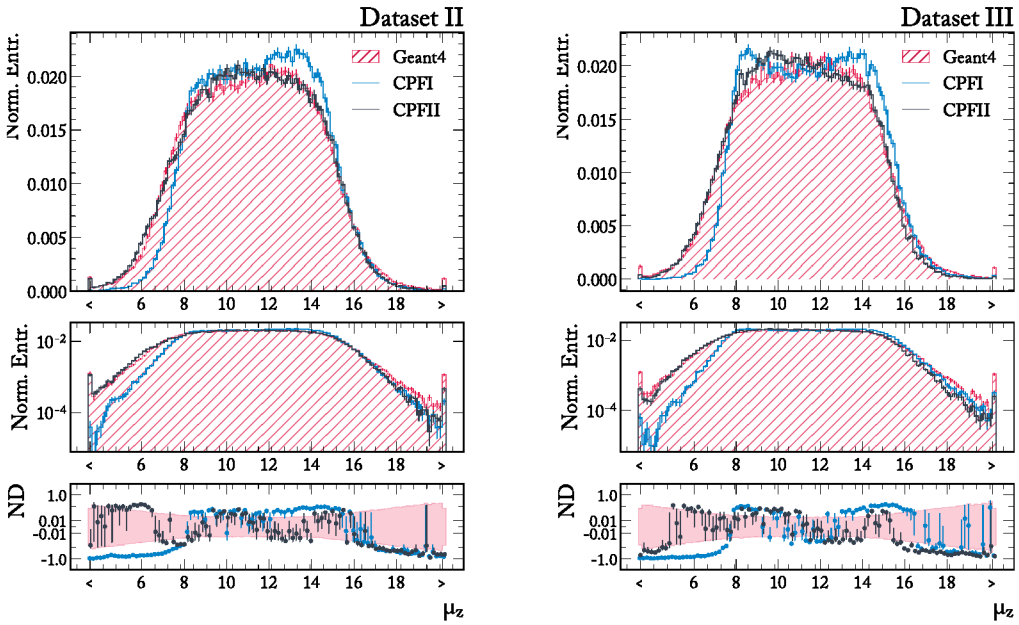

Figure 8.19: Shower centers μ_z in longitudinal direction z for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The Geant4 distribution exhibits an exponential rise, which saturates into a plateau between $z = 8$ and $z = 15$, followed by an exponential decline at higher values. Notably, there is no significant difference in the Geant4 distribution between Dataset II and Dataset III.

In comparing the models, CPF II demonstrates superior performance compared to CPF I overall. For Dataset II, CPF I shows a noticeable shift in the rising flank, with the plateau region not being well modeled. Although CPF I improves in capturing the falling distribution, it struggles with accurately modeling the plateau region, particularly near the end, where it overshoots. In contrast, CPF II effectively models the rising distribution and accurately captures the underflow bin. Within the plateau region, CPF II reaches the uncertainty level of Geant4. However, CPF II slightly undershoots in the falling distribution and the overflow bin. Overall, CPF II models the shower centers in the z direction reliably.

Table 8.12: X^2 for all three models in both datasets of the histograms shown in Figure 8.19.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	97.8 ± 14.5	2870 ± 100	429 ± 37
III	97.9 ± 13.6	3850 ± 110	677 ± 48

In Dataset III, CPF I again displays a shifted increase similar to that observed in Dataset II. Both edges of the plateau are overshot, and the decline is initially too gradual. Meanwhile, CPF II mostly aligns with the Geant4 uncertainty in the central region, although it exhibits worse performance in the tails, particularly in the overflow bin.

In summary, CPF II provides a more accurate representation of the shower centers in the z direction across both datasets, although there are still areas, particularly at the distribution tails, where further improvements are needed.

The analysis of the shower center can also be extended to the radial r direction. The shower center in this direction is defined as

$$\mu_{i,r} = \frac{\sum_{r=1}^{n_r} r \cdot E_{i,r}}{E_{i,\text{sum}}},$$

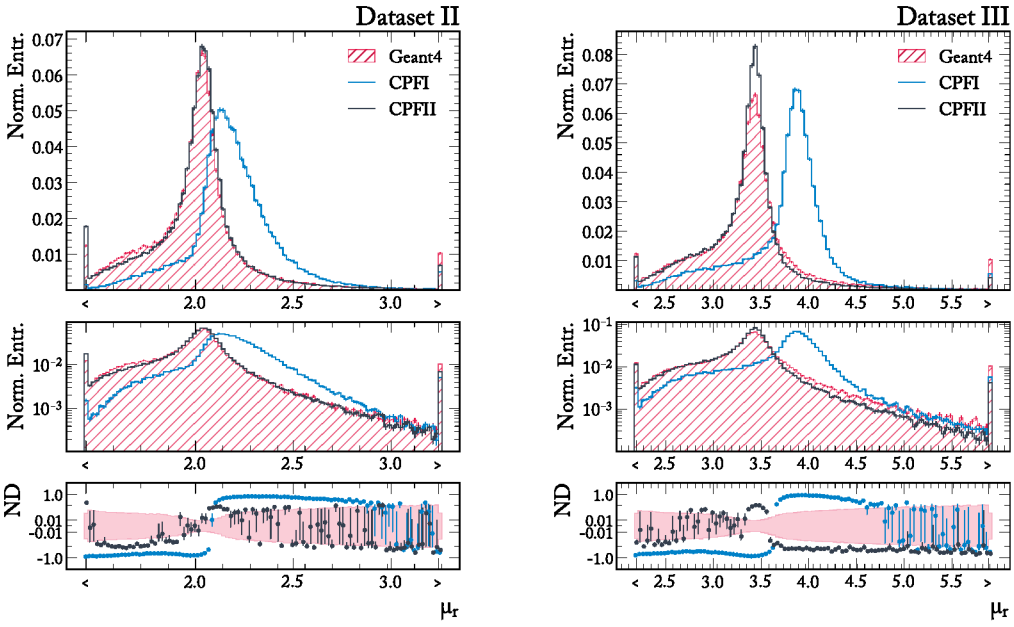

Figure 8.20: Shower centers μ_r in radial direction r for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

which represents the average r coordinate of all hits in the shower, weighted by their energy.

In Figure 8.20, the distribution of the shower centers μ_r in the radial direction is presented, employing the same binning strategy as used for μ_z—100 linear bins spanning from the 1st to the 99th percentile, including overflow and underflow bins.

The distribution increases until it peaks at $r \approx 2$ for Dataset II and $r \approx 3.4$ for Dataset III, with the different peak positions explained by the varying granularity in r between the datasets. Following the peak, the distribution exhibits an exponential decline for both datasets.

The CPF I model, however, demonstrates a peak that is too wide in the outer regions of the detector, failing to closely follow the Geant4 distribution. In contrast, CPF II more accurately captures the peaking structure, particularly in Dataset II, although it slightly overestimates the peak in Dataset III. Additionally, the distribution before the peak is somewhat too flat, and there are more entries in the underflow bin. However, the decline after the peak is well modeled by CPF II.

The shift of the shower center peak towards the outer regions in CPF I is likely due to the high probability of double hits in the central coordinates, where most hits in the shower occur. The discarding of these hits in CPF I reduces the energy in the center of the detector, leading to the observed shift. This issue is less pronounced in CPF II, which exhibits fewer double hits and thus provides a more accurate representation of the shower center in the radial direction.

Overall, CPF II offers better modeling of the shower centers in the radial direction across both datasets, although some discrepancies remain, particularly in the flatness of the distribution before the peak and the performance in Dataset III.

Table 8.13: X^2 for all three models in both datasets of the histograms shown in Figure 8.20.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	$(1.00 \pm 0.15) \times 10^{-3}$	$(4.92 \pm 0.04) \times 10^{-1}$	$(6.50 \pm 0.48) \times 10^{-3}$
III	$(9.98 \pm 1.46) \times 10^{-4}$	$(6.71 \pm 0.04) \times 10^{-1}$	$(1.54 \pm 0.08) \times 10^{-2}$

The evaluation of the shower centers can also be extended to the Cartesian coordinate frame, specifically in the x and y directions. First, an auxiliary energy sum is introduced by summing over the z index:

$$E_{i,\alpha,r} = \sum_{z=1}^{n_z} E_{i,z,\alpha,r}. \tag{8.27}$$

The shower center in the y direction is then defined as

$$\mu_{i,y} = \frac{\sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} r \cos\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right) \cdot E_{i,\alpha,r}}{E_{i,\text{sum}}}. \tag{8.28}$$

In Figure 8.21, the distribution of the shower center in the y direction is presented. The histogram is binned into 100 linear bins spanning from the 1st to the 99th percentile, including both overflow and underflow bins.

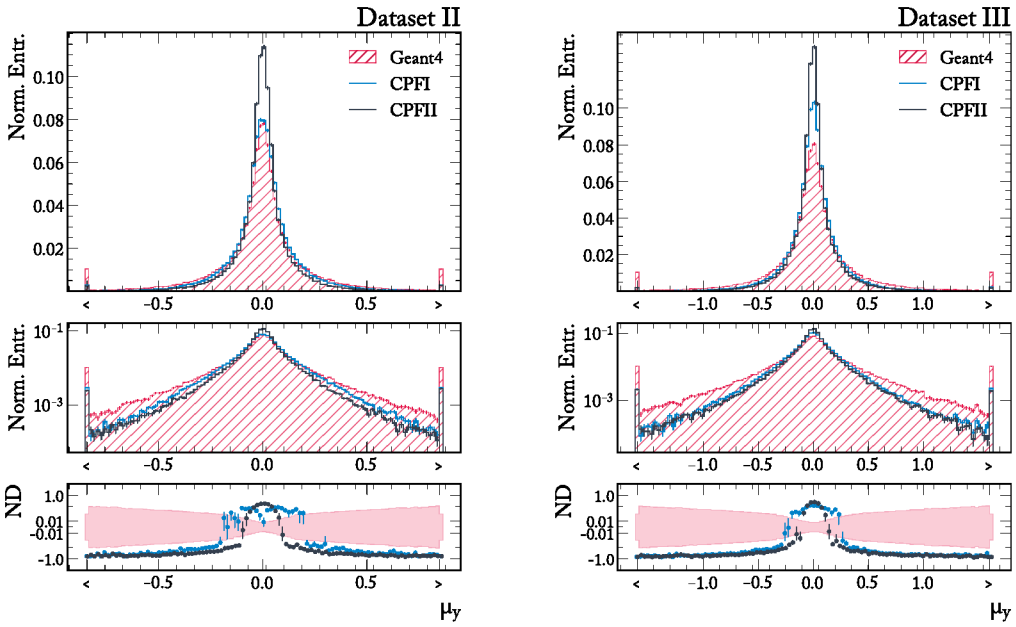

Figure 8.21: Shower centers μ_y in direction y for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

Due to the position of the incident particle, the center of the shower is expected to peak at $y = 0$. The distributions for both models reflect this expectation, with the peaks centered at $y = 0$ and falling off exponentially in both the positive and negative directions.

For Dataset II, the CPF I model accurately captures the peak, though it slightly underestimates the mass in the tails on both sides. In contrast, CPF II produces a peak that is too high, resulting in even less mass in the tails, clearly indicating that CPF I outperforms CPF II in this aspect. For Dataset III, a similar structure is observed, although here, CPF I also exhibits a peak that is too high. It is noteworthy that CPF I models the center of the energy distribution in the y coordinate more effectively, particularly for Dataset II.

Table 8.14: X^2 for all three models in both datasets of the histograms shown in Figure 8.21.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	$(1.00 \pm 0.15) \times 10^{-3}$	$(2.00 \pm 0.08) \times 10^{-2}$	$(6.33 \pm 0.15) \times 10^{-2}$
III	$(9.96 \pm 1.45) \times 10^{-4}$	$(4.31 \pm 0.12) \times 10^{-2}$	$(8.48 \pm 0.16) \times 10^{-2}$

The shower center in the x direction is also analyzed, where the x coordinate is defined as

$$x = r \cos\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right). \tag{8.29}$$

The shower center in the x direction is defined as

$$\mu_{i,x} = \frac{\sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} r \sin\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right) \cdot E_{i,\alpha,r}}{E_{i,\text{sum}}}. \tag{8.30}$$

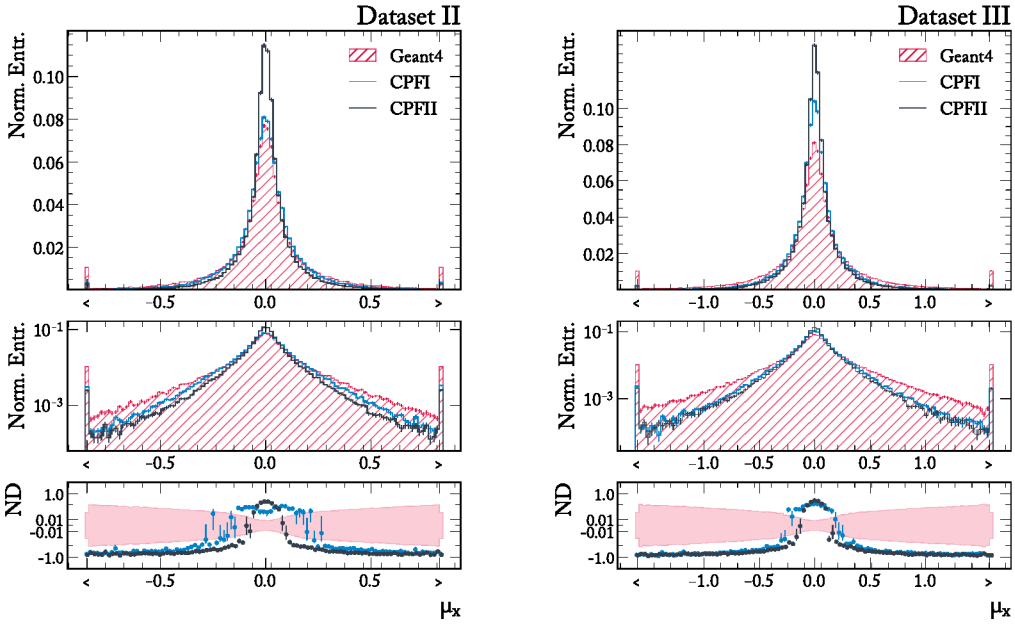

Figure 8.22: Shower centers μ_x in direction x for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

In Figure 8.22, the distribution of the shower center in the x direction is presented, using the same binning strategy as for μ_y.

Since the shower should be invariant under rotation, the distribution of the shower center in the x direction is expected to mirror that of the y direction. This expectation is confirmed, as the same behavior is observed in both directions.

Overall, CPF I models the shower center in both the y and x directions better than CPF II. This is likely because the data were transformed and learned in the Cartesian coordinate frame with CPF I, making the model more sensitive to this coordinate system.

Table 8.15: X^2 for all three models in both datasets of the histograms shown in Figure 8.22.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	$(9.99 \pm 1.42) \times 10^{-4}$	$(1.72 \pm 0.08) \times 10^{-2}$	$(6.46 \pm 0.15) \times 10^{-2}$
III	$(9.99 \pm 1.42) \times 10^{-4}$	$(4.56 \pm 0.12) \times 10^{-2}$	$(8.60 \pm 0.17) \times 10^{-2}$

8.8.7 *Shower Width*

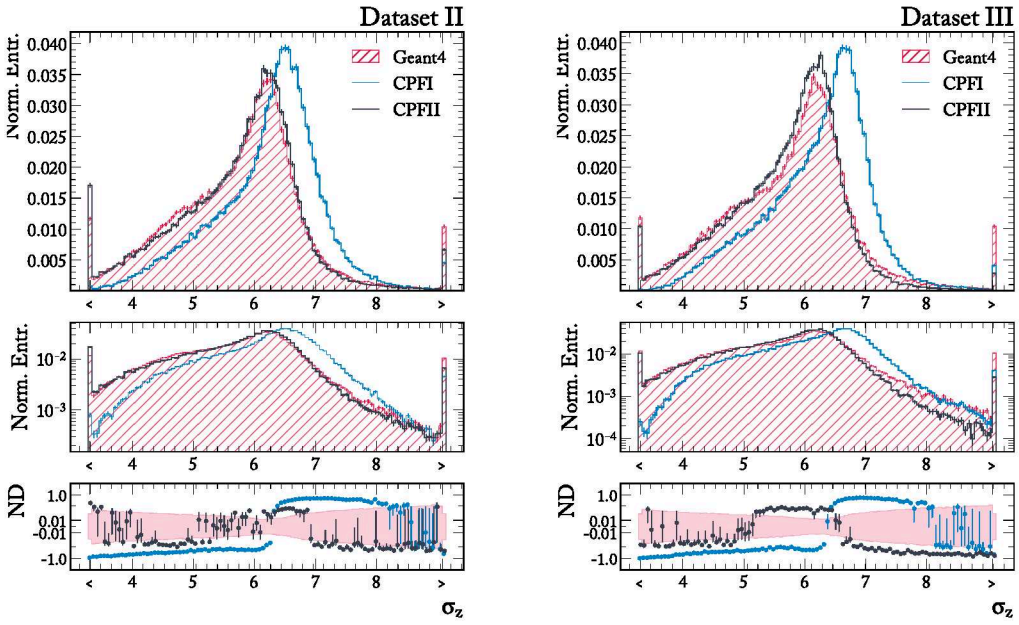

Figure 8.23: Shower width σ_z in direction z for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The evaluation of the shower characteristics also includes the analysis of the shower width. The shower width in the z direction is defined as

$$\sigma_{i,z} = \sqrt{\frac{\sum_{z=1}^{n_z} z^2 \cdot E_{i,z}}{E_{i,\text{sum}}} - \mu_{i,z}^2}. \tag{8.31}$$

In Figure 8.23, the distribution of the shower width in the z direction is presented. The binning is selected to match that used for the mean values. For both datasets, the Geant4 distribution increases to approximately 6 and then exhibits an exponential decline. However, both datasets reveal a misalignment in the shower width for the CPF I model, with the peak shifted to higher values. Conversely, the CPF II model accurately captures the distribution in Dataset II, while its performance decreases in Dataset III. The CPF II model also adequately models both tails in Dataset III, whereas in Dataset II, the right-side falling tail is slightly underestimated. Overall, the CPF II model demonstrates superior performance, which is further supported by the lower X^2 values, as presented in Table 8.16.

Table 8.16: X^2 for all three models in both datasets of the histograms shown in Figure 8.23.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	$(10.00 \pm 1.40) \times 10^{-4}$	$(1.67 \pm 0.02) \times 10^{-1}$	$(4.90 \pm 0.40) \times 10^{-3}$
III	$(9.97 \pm 1.39) \times 10^{-4}$	$(1.83 \pm 0.03) \times 10^{-1}$	$(1.37 \pm 0.07) \times 10^{-2}$

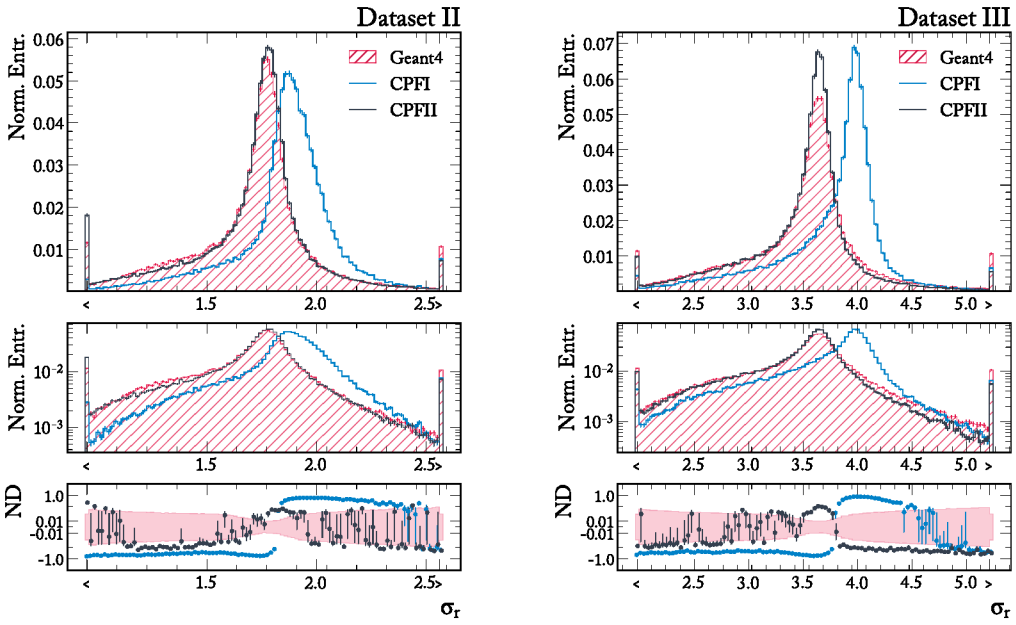

Figure 8.24: Shower width σ_r in direction r for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The distribution of the width in the r direction, defined as

$$\sigma_{i,r} = \sqrt{\frac{\sum_{r=1}^{n_r} r^2 \cdot E_{i,r}}{E_{i,\text{sum}}} - \mu_{i,r}^2}, \tag{8.32}$$

is shown in Figure 8.24. The shower width in the r direction exhibits a similar shape to that in the z direction, peaking at approximately 1.75 for Dataset II and 3.5 for Dataset III. The same pattern is observed in the CPF models, including the peak shift for CPF I. Table 8.17 presents the X^2 for these histograms, highlighting the significantly lower X^2 for the CPF II model compared to CPF I.

Table 8.17: X^2 for all three models in both datasets of the histograms shown in Figure 8.24.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	99.1 ± 13.8	39200 ± 400	1110 ± 70
III	101 ± 14	50900 ± 400	1850 ± 80

In Cartesian coordinates, the width in the y direction is defined as

$$\sigma_{i,y} = \sqrt{\frac{\sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} \left(r \cos\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right)\right)^2 \cdot E_{i,\alpha,r}}{E_{i,\text{sum}}} - \mu_{i,y}^2}. \tag{8.33}$$

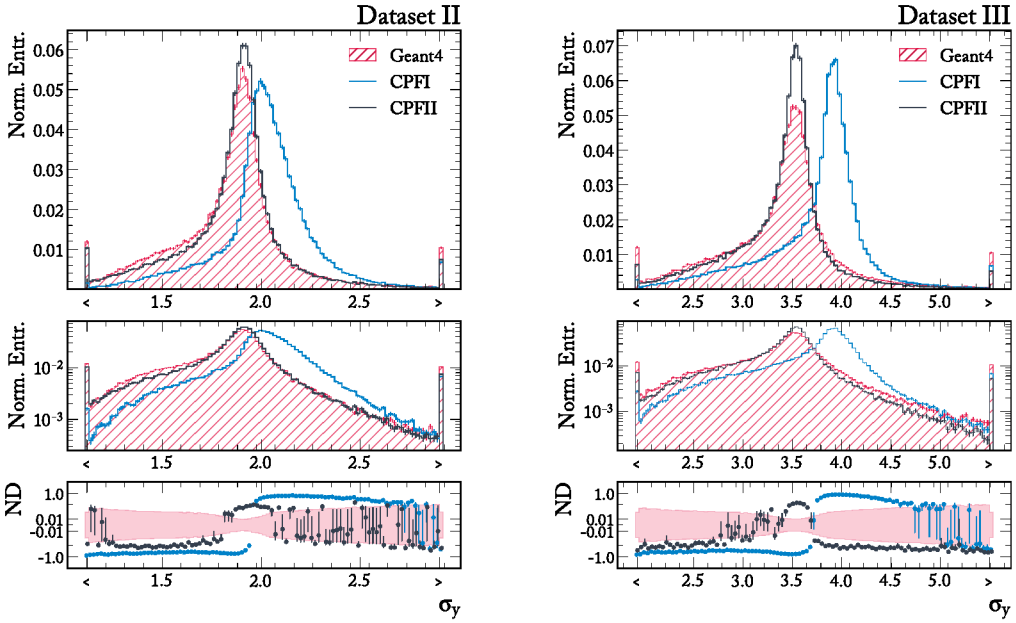

Figure 8.25: Shower width σ_y in direction y for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The shower width in the y direction is presented in Figure 8.25. Similarly, the width in the x direction is defined as

$$\sigma_{i,x} = \sqrt{\frac{\sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} \left(r \sin\left(\frac{2\pi \cdot \alpha}{n_\alpha}\right)\right)^2 \cdot E_{i,\alpha,r}}{E_{i,\text{sum}}} - \mu_{i,x}^2}. \tag{8.34}$$

The corresponding width in the x direction is shown in Figure 8.26. No discernible differences are observed between the x and y directions, which is expected due to the rotational symmetry of the shower. The behavior observed in the r and z directions is similarly reflected in the y and x directions, indicating a consistent performance pattern across these dimensions.

Table 8.18: X^2 for all three models in both datasets of the histograms shown in Figure 8.25.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	98.7 ± 14.3	39200 ± 300	1100 ± 70
III	101 ± 14	50900 ± 400	1850 ± 80

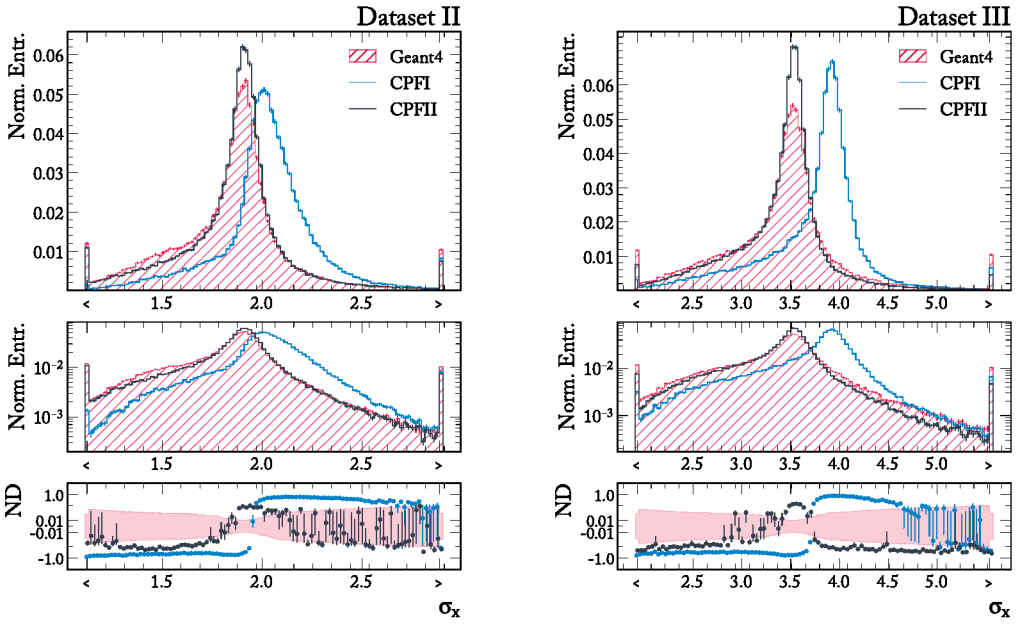

Figure 8.26: Shower width σ_x in direction x for Dataset II (left) and Dataset III (right) comparing Geant4, CPF I, and CPF II models. The upper graphs show histograms of the shower centers. The middle graphs present the same data as the upper graphs but with a logarithmic scale on the y-axis. The lower graphs show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

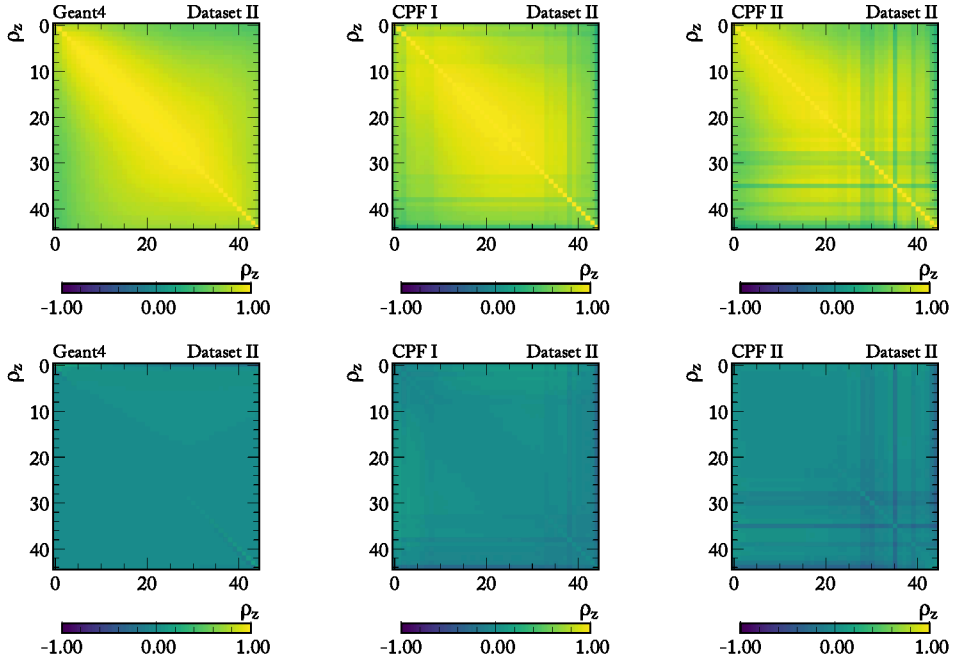

Figure 8.27: Pearson correlation coefficients ρ in direction z for Dataset II comparing Geant4, CPF I, and CPF II models. Upper row shows correlation coefficients. Lower row shows the ND of the correlation coefficients

Table 8.19: X^2 for all three models in both datasets of the histograms shown in Figure 8.26.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	100 ± 14	39900 ± 400	1170 ± 60
III	100 ± 14	50300 ± 400	1860 ± 90

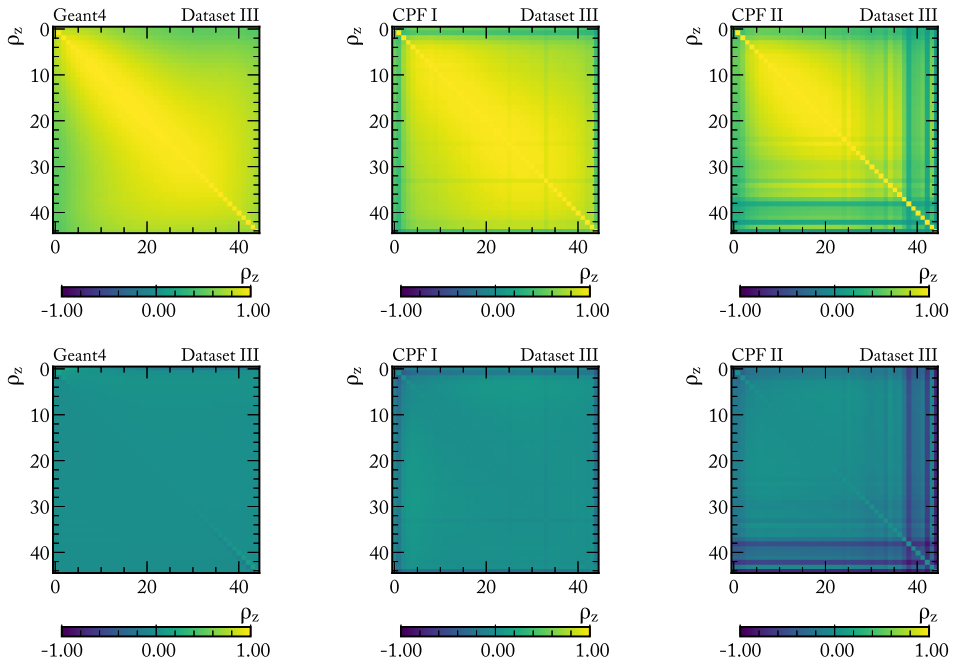

Figure 8.28: Pearson correlation coefficients ρ in direction z for Dataset III comparing Geant4, CPF I, and CPF II models. Upper row shows correlation coefficients. Lower row shows the ND of the correlation coefficients

8.8.8 *Layer Correlation Coefficients*

In a calorimeter, the energy deposited in different layers is expected to exhibit correlations, reflecting the underlying physics of particle showers. To evaluate the capability of the CPF models in capturing these correlations, the Pearson correlation coefficient, ρ_{z_j,z_k}, between two longitudinal layers z_j and z_k is studied. The correlation coefficient is defined as follows:

$$\Delta E_{i,z_j} = E_{i,z_j} - \bar{E}_{z_j}, \tag{8.35}$$

$$\rho_{z_j,z_k} = \frac{\sum_{i=1}^{n} \Delta E_{i,z_j} \Delta E_{i,z_k}}{\sqrt{\sum_{i=1}^{n} \Delta E_{i,z_j}^2 \sum_{i=1}^{n} \Delta E_{i,z_k}^2}}. \tag{8.36}$$

Here, $z_j, z_k \in [1, n_z]$. Figure 8.27 presents ρ_{z_j,z_k} for Geant4, CPF I, and CPF II for Dataset II. Figure 8.28 shows the same for Dataset III. The top row of the figure shows the average correlation coefficients over 1,000 bootstrapped samples, while the bottom row displays the

ND calculated from 1,000 combinations. Since both Dataset II and Dataset III share the same longitudinal segmentation, the results are consistent between these datasets.

For the Geant4 data, the highest correlations are observed between adjacent layers, with correlations decreasing as the distance between the layers increases. This behavior is consistent with the localized nature of particle showers. Both CPF models generally follow this pattern; however, certain layers in both models exhibit visibly lower correlations, deviating from the expected pattern. This deviation is more pronounced in the CPF II model.

To quantify the overall agreement between the models and Geant4, the Frobenius norm of the ND of the correlation matrices is computed:

$$\mathrm{Frob}\left(\rho, \rho'\right) = \sqrt{\sum_{z_j=1}^{n_z} \sum_{z_k=1}^{n_z} \left(\mathrm{ND}\left(\rho_{z_j,z_k}, \rho'_{z_j,z_k}\right)\right)^2} \tag{8.37}$$

The results, shown in Table 8.20, indicate that the CPF I model better captures the correlations within the calorimeter than CPF II. The Frobenius norms clearly demonstrate that CPF I has a closer match to the Geant4 correlation structure, reflecting its superior performance in modeling these essential correlations. The reason for the better performance of CPF I is not entirely clear; it is possible that the more complex architecture of the DSF could benefit from additional training volume.

Table 8.20: Frobenius norm of the ND between the ρ values fo all three models in both datasets of the histograms shown in Figure 8.24.

Dataset	**Geant4**	**CPF I**	**CPF II**
II	1.32 ±0.05	3.47 ±0.65	5.28 ±1.46
III	1.29 ±0.02	3.74 ±0.69	12.56 ±0.90

8.8.9 *Classifier Scores*

A classifier is trained to distinguish between simulated Geant4 data (Geant4) and the generated samples from the CPF (CPF). This approach is understood as a two-sample test (Lopez-Paz & Oquab 2016). The classifier is trained using binary cross-entropy (refer to Section 5.1.3), and an ideal classifier therefore predicts the probability function

$$f(x) = p(\text{Geant4}|x) = \frac{p(x|\text{Geant4})p(\text{Geant4})}{p(x|\text{Geant4})p(\text{Geant4}) + p(x|\text{CPF})p(\text{CPF})} \tag{8.38}$$

Since two samples of the same size are taken, it is assumed that $p(\text{Geant4}) = p(\text{CPF})$, simplifying the expression to

$$f(x) = \frac{p(\text{Geant4}|x)}{p(\text{Geant4}|x) + p(\text{CPF}|x)}. \tag{8.39}$$

From this, the transformation can be constructed as

$$\begin{aligned}\frac{f(x)}{1-f(x)} &= \frac{\frac{p(\text{Geant4}|x)}{p(\text{Geant4}|x)+p(\text{CPF}|x)}}{1-\frac{p(\text{Geant4}|x)}{p(\text{Geant4}|x)+p(\text{CPF}|x)}} \\ &= \frac{\frac{p(\text{Geant4}|x)}{p(\text{Geant4}|x)+p(\text{CPF}|x)}}{\frac{p(\text{CPF}|x)}{p(\text{Geant4}|x)+p(\text{CPF}|x)}} \\ &= \frac{p(\text{Geant4}|x)}{p(\text{CPF}|x)}.\end{aligned} \tag{8.40}$$

This expression represents the likelihood ratio, which, according to the Neyman-Pearson lemma (Neyman & Pearson 1933), is the most powerful test in binary hypothesis testing.

For calorimeter simulation surrogate models, classifier scores have been proposed as a metric by (Krause & Shih 2021), and (Das et al. 2023) demonstrated their utility in the context of particle physics generative models. To facilitate better comparability with other models, the CaloChallenge classifiers (Giannelli et al. 2022a) are employed in this evaluation. Two classifiers are utilized: the low-level classifier and the high-level classifier.

The low-level classifier classifies each shower based solely on the energy values $E_{i,z,\alpha,r}$. The high-level classifier incorporates low-level information along with additional features such as $E_{\text{sum}}/E_{\text{in}}$, μ_z, μ_r, μ_y, μ_x, σ_z, σ_r, σ_y, σ_x, and the sparsity, defined as

$$\text{sparsity} = 1 - \frac{n_{\text{hits}}}{n_z n_r n_\alpha}.$$

The *Receiver Operating Characteristic* (ROC) curve plots the true positive rate as a function of the false positive rate. The classifier score is reported as the *Area Under the Curve* (AUC), where a random classifier yields an AUC of 0.5, and a perfect separation results in an AUC of 1.0.

To obtain reliable results, ten randomly initialized classifiers are trained. The AUC scores for both low-level and high-level classifiers are presented in Table 8.21. The results indicate that the CPF II model consistently achieves lower classifier scores compared to the CPF I model, implying that the CPF II model is harder to distinguish from the Geant4 sample. Nevertheless, the classifier is able to reasonably distinguish between the CPF II results and the Geant4 results, despite the challenges discussed above.

Table 8.21: Low Level and High Level CaloChallenge Classifier AUC Scores for CPF I and CPF II for Dataset II and Dataset III.

Dataset	Model	low level	high level
II	CPF I	0.945 ± 0.004	0.927 ± 0.003
	CPF II	0.826 ± 0.006	0.785 ± 0.009
III	CPF I	0.786 ± 0.019	0.947 ± 0.003
	CPF II	0.709 ± 0.040	0.934 ± 0.003

8.8.10 *Ablation Study*

To assess the impact of individual modifications on the performance of the CPF II model in comparison to the CPF I model, an ablation study was conducted. The CPF II model incorporates several alterations relative to the CPF I model. To evaluate the contribution of each modification, the performance of different model variations was incrementally assessed.

The study compared the following models:

- The base CPF I model (CPF I).
- The CPF I model incorporating the DSF approach as defined in Section 8.3.2 (+DSF).
- The CPF I model incorporating the DSF approach and the random assignment of the α coordinate as described in Section 8.7.2 (+3d).
- The complete CPF II model, which includes the DSF approach, random assignment of α, and CDF-Dequantization as described in Section 8.6.2 (CPF II).

The evaluation was conducted using Dataset II, focusing solely on the CaloChallenge classifier scores. Each model variant was trained with 10 different initializations for 10 epochs. Subsequently, a classifier was trained for each variation to determine the AUC scores for both low-level and high-level classifiers.

The results, presented in Table 8.22, report the average AUC scores for the different model variations, and these scores are also illustrated in Figure 8.29. The study demonstrated a consistent decrease in the average AUC score with each incremental modification, indicating an enhancement in model performance. However, the variance also increased, suggesting that the additional features may require more extensive training.

Although some performance improvements were within the margin of error, particularly in the transition from "+3d" to the complete CPF II model, an overall performance enhancement was observed when considering both classifiers together. This improvement exceeded the margin of error, thereby confirming the effectiveness of the modifications introduced in the CPF II model.

Table 8.22: The average CaloChallenge classifier AUC scores, low level and high level, are presented for the different model variations.

Model	low level	high level
I	0.966 ± 0.001	0.945 ± 0.001
+DSF	0.960 ± 0.001	0.928 ± 0.002
+3d	0.892 ± 0.002	0.881 ± 0.003
II	0.874 ± 0.004	0.833 ± 0.002

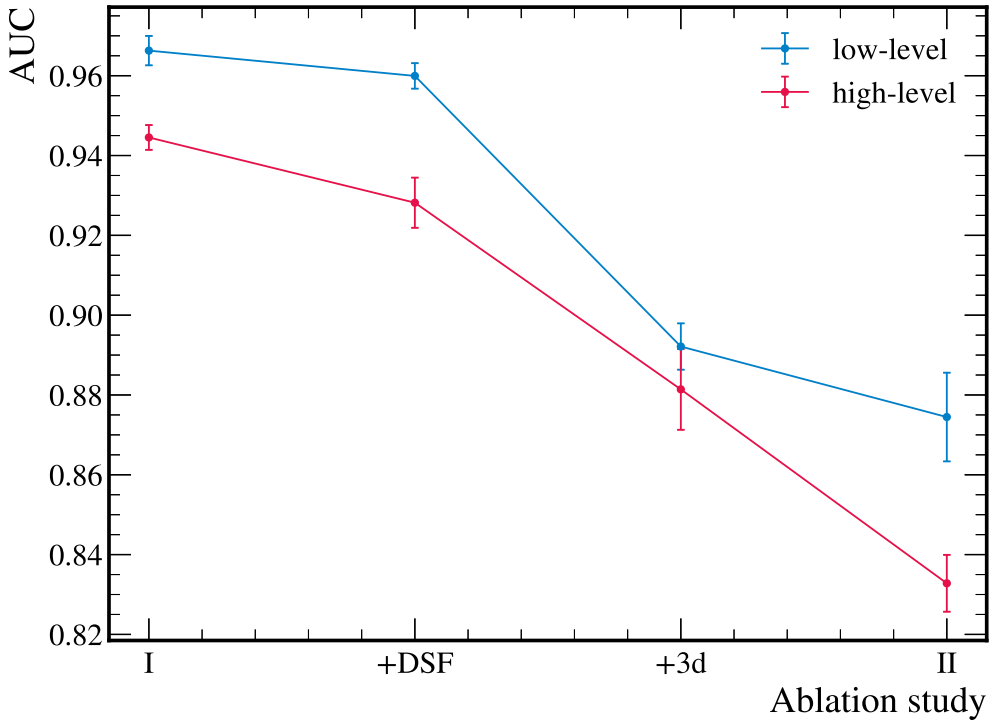

Figure 8.29: The average CaloChallenge classifier AUC scores, low level and high level, are presented for the different model variations.

8.8.11 *Timing*

The primary objective of surrogate models in particle physics is to expedite the generation of calorimeter showers. Consequently, timing is a critical factor in their evaluation. However, providing a definitive assessment of the generation speed is challenging due to several factors. The generation time is heavily dependent on the specific hardware employed and the model settings. Furthermore, the models can be further optimized to enhance their speed. In addition, direct access to the Geant4 simulation times and the corresponding hardware configurations is unavailable, which limits the comparability of the results.

According to Buckley et al. 2023, the Geant4 simulation time for Dataset II and III is on the order of 100 s. In comparison, the CPF models require approximately 1.5 ms on an NVIDIA A100 GPU and around 50 ms on a single core of an AMD EPYC 7543 CPU to produce one shower. These findings demonstrate that the surrogate models operate several orders of magnitude faster than Geant4. Thus, speed is not the limiting factor in the generation process; instead, the focus should remain on enhancing the quality of the generated samples.

CALOHIT

9

In the previous section, the primary issues encountered with two point flow-based calorimeter surrogate models were discussed, particularly the challenge of learning coordinates in a continuous space and mapping them back to discrete positions. This approach introduces significant complications, especially in ensuring accurate tracking of which positions are already occupied. As a result, multiple points may be erroneously mapped to the same coordinate, creating conflicts in position assignment. While existing strategies offer partial solutions to mitigate these issues, they do not fully resolve the conflicts inherent to this architectural approach. Consequently, a more effective solution is required to address these limitations and ensure reliable simulation outcomes.

In this section, a novel architecture is proposed to prevent the aforementioned problems entirely. This architecture, termed the *CaloHit approach*, represents a hybrid solution that integrates voxel-based and point cloud-based strategies. The core idea is to decouple the generation process into two distinct phases: one responsible for generating hit positions and another for assigning energies to these hits. By definition, the hit positions are discrete, while the energies remain continuous. This separation ensures that the challenges associated with conflicting positions in continuous space are avoided, allowing for a more accurate and reliable simulation outcome.

To implement this, two complementary models are introduced: one based on a voxel approach to generate hit positions, and another that uses a point flow-based method for energy assignment. The first step in this approach involves establishing the mathematical framework to effectively manage the hit map, utilizing the Gumbel-dequantization technique developed in this work. This method ensures that the discrete hit positions are handled efficiently while maintaining the continuous nature of energy distribution.

This architecture is then illustrated using a case study on a small dataset, specifically the smallest dataset from the CaloChallenge. This section details the design of two models based on the CaloHit approach, demonstrating how these models effectively address the issues identified in the previous architecture. Finally, potential extensions of this architecture are discussed, outlining the next steps and identifying any remaining challenges in the current stage of the research.

9.1 Dataset 1: Photon-Initiated Showers

To evaluate the CaloHit approach, the first dataset from the CaloChallenge, derived from the ATLAS Geant4 Open datasets, was used. The CaloChallenge provides two datasets: one where the incident particle is a photon and another where the incident particle is a pion. In this study, the analysis is restricted to the photon dataset, referred to as Dataset I, while the pion dataset is excluded from the investigation. Dataset I represents a voxelized configuration of the current ATLAS detector, specifically within the η range of 0.2 to 0.25 (ATLAS 2022). The detector geometry is cylindrical, with the coordinate system illustrated

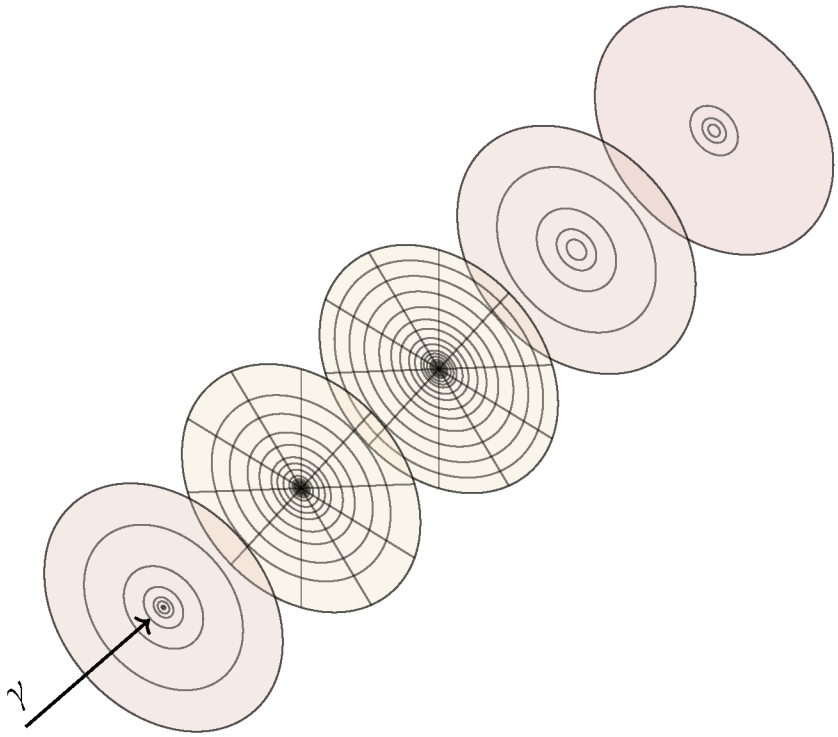

Figure 9.1: Segmentation of the 5 layers in Dataset I. The radius of all layers is set to the same size for easier visualization. The distance between the layers is also arbitrary.

in Figure 8.1. The incident photons propagate along the z-axis, and the detector is divided into five layers, each containing a varying number of radial and angular bins. Segmentation details are provided in Table 9.1 and Figure 9.1, resulting in a total of 368 voxels.

The dataset records energy depositions within each voxel in units of MeV. The energies of the incident particles range from 2^8 to 2^{22}, with energy depositions distributed across 15 distinct levels, each separated by powers of two. For each energy level, the dataset includes 10,000 events, except at the highest energies, where fewer events are available due to the extended simulation time required for large showers. In total, 2x121,000 events are recorded. Half of these events were split into training and validation sets, while the remaining half was reserved for testing.

Table 9.1: Nominal binning for photons with $0 < \eta < 1.3$ range.

Layer	r edges [mm]	α bins
1	0, 5, 10, 30, 50, 100, 200, 400, 600	1
2	0, 2, 4, 6, 8, 10, 12, 15, 20, 30, 40, 50, 70, 90, 120, 150, 200	10
3	0, 2, 5, 10, 15, 20, 25, 30, 40, 50, 60, 80, 100, 130, 160, 200, 250, 300, 350, 400	10
4	0, 50, 100, 200, 400, 600	1
5	0, 100, 200, 400, 1000, 2000	1

9.2 Gumbel-Dequantization

In designing the generative model for the hit distribution within the calorimeter, the objective is to sample hit cells while ensuring that each hit position is unique, thereby avoiding the overlap issues identified in earlier models.

To accomplish this, a dequantization strategy that allows for sampling from discrete distributions without replacement is required. This objective is achieved through the application of the Gumbel Top-k trick (Kool et al. 2019; Vieira 2014). The Gumbel-Dequantization technique offers several advantages. In principle, developing a model that samples without replacement requires generating output probabilities for each class. For each sample, the model must predict the probabilities while ensuring that previously sampled classes have a probability of zero, thereby preventing repeated selections. This approach presents significant challenges, as recalculating the probabilities for each sample requires traversing n paths through the network, where n represents the number of samples. Furthermore, the model must handle a dynamically changing output space for each combination, which adds to the complexity of development.

The Gumbel-Dequantization technique simplifies the process of sampling without replacement by directly predicting the order in which elements are selected, thereby eliminating the need for multiple recalculations and reducing the complexity of the model architecture.

This approach is referred to as dequantization because, in the output space, each data point is represented by values $p_i \in \{0, 1\}$, with $\sum p_i = n$. The goal is to transform these discrete values into continuous ones, making them amenable to learning through one of the previously discussed models.

Before delving into the details of this technique, it is important to introduce the following concepts: the Gumbel distribution, the Gumbel Max trick, the Gumbel Top-k trick, and, finally, the Gumbel Top-k dequantization technique.

9.2.1 *Gumbel Distribution*

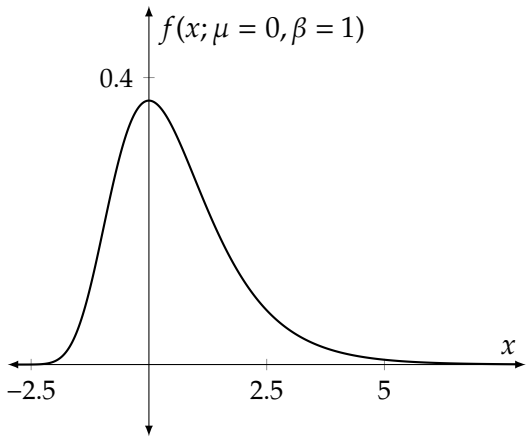

Figure 9.2: Gumbel PDF with $\mu = 0$ and $\beta = 1$.

The Gumbel distribution (Gumbel 1935, 1941), denoted as $G(\mu, \beta)$, is characterized by two parameters, the location parameter $\mu \in \mathbb{R}$ and the scale parameter $\beta \in \mathbb{R}_{\geq 0}$. A random variable following to this distribution is represented as $G_{\mu,\beta}$.

The PDF of the Gumbel distribution is given by

$$g_{\mu,\beta}(x) = \frac{1}{\beta} \exp\Big(-(x-\mu)/\beta - \exp\Big(-(x-\mu)/\beta\Big)\Big) \tag{9.1}$$

and the CDF is

$$G_{\mu,\beta}(x) = \exp\Big(-\exp\Big(-(x-\mu)/\beta\Big)\Big). \tag{9.2}$$

The inverse of the CDF is expressed as

$$G^{-1}_{\mu,\beta}(u) = \mu - \beta \ln\left(-\ln(u)\right). \tag{9.3}$$

9.2.2 *Gumbel-Max Trick*

Consider a categorical distribution $\mathrm{Cat}(\pi)$ that is defined over N classes. Each category i within this distribution is associated with a corresponding probability π_i, where $i \in \{1, \dots, N\}$. For the more general case where the probabilities are not normalized, π_i can be expressed in terms of unnormalized probabilities x_i as

$$\pi_i = \frac{x_i}{\sum_{j \in N} x_j}. \tag{9.4}$$

The Gumbel-Max trick enables exact sampling from a categorical distribution $\mathrm{Cat}(\pi)$, since

$$I = \underset{i}{\operatorname{argmax}} \left\{\ln x_j + g^{(j)}\right\} \sim \mathrm{Cat}(\pi). \tag{9.5}$$

Here, $g^{(i)} \sim G_{0,1}$ are i.i.d. random variables for all i in N, and the probability of selecting any category i is exactly π_i. It follows that

$$\ln x_j + g^{(j)} = g^{(j)}_{\ln x_j}, \tag{9.6}$$

where $g^{(j)}_{\ln x_j} \sim G_{\ln x_j, 1}$.

The derivation of the probability $P(I = i) = \pi_i$ follows from the work of Huijben et al. 2023.

$I = i$ holds if $g_{\ln x_i} > g_{\ln x_j}\ \forall j \in N \setminus \{i\}$. Since all $g_{\ln x_j}$ are i.i.d., it can be factorized as

$$\begin{aligned}
P(I = i) &= \int_{-\infty}^{\infty} g_{\ln x_i,1}(z) \prod_{j \in N \setminus \{i\}} p(g_{\ln x_j} < z)\, \mathrm{d}z \\
&= \int_{-\infty}^{\infty} g_{\ln x_i,1}(z) \prod_{j \in N \setminus \{i\}} \exp\Big(-\exp\big(\ln x_j - z\big)\Big)\, \mathrm{d}z \\
&= \int_{-\infty}^{\infty} \exp\left(\ln x_i - z - \exp(\ln x_i - z)\right) \exp\left(-\sum_{j \in N \setminus \{i\}} \exp\big(x_j - z\big)\right) \mathrm{d}z \\
&= \int_{-\infty}^{\infty} \exp\left(\ln x_i - z\right) \exp\left(-\sum_{j \in N} \exp\big(\ln x_j - z\big)\right) \mathrm{d}z \\
&= x_i \int_{-\infty}^{\infty} \exp(-z) \exp\left(-\exp(-z) \sum_{j \in N} x_j\right) \mathrm{d}z \\
&= \frac{x_i}{\sum_{j \in N} x_j} \\
&= \pi_k.
\end{aligned} \tag{9.7}$$

9.2.3 *Gumbel-Top-k Trick*

The Gumbel-Top-k trick extends the Gumbel-Max trick to allow for the selection of $k \leq n$ categories without replacement. Given an (ordered) sample without replacement from $\mathrm{Cat}(\pi)$, denoted as $I_1, ..., I_k$, the probability of any realization $i_1, ..., i_k$ is given by

$$P(I_1 = i_1, ..., I_k = i_k) = \prod_{m=1}^{k} \frac{x_{i_m}}{\sum_{l \in N_m} x_l}. \tag{9.8}$$

where $N_m = N \setminus \{i_1, ..., i_{m-1}\}$ represents the domain without replacement for the m-th sampled element.

The conditional probability for selecting the k-th element, given the previous selections, is expressed as

$$\begin{aligned}
&P\left(I_k = i_k | I_1 = i_1, ..., I_{k-1} = i_{k-1}\right) \\
=&\left(i_k = \arg\max_{i \in N_k} g_{\ln x_i} | I_1 = i_1, ..., I_{k-1} = i_{k-1}\right) \\
=&P\left(i_k = \arg\max_{i \in N_k} g_{\ln x_i} | \max_{i \in N_k} g_{\ln x_i} < g_{\ln x_{k-1}}\right) \\
=&P\left(i_k = \arg\max_{i \in N_k} g_{\ln x_i}\right) \\
=&\frac{x_{i_k}}{\sum_{l \in N_k} x_l}.
\end{aligned} \tag{9.9}$$

The Gumbel-Top-k trick can be proven by induction on k. Assuming the result holds for $k-1$, the k case is demonstrated as follows:

$$\begin{aligned} P(I_1 = i_1, ..., I_k = i_k) &= P(I_k = i_k | I_1 = i_1, ..., I_{k-1} = i_{k-1}) \\ &\quad \cdot P(I_1 = i_1, ..., I_{k-1} = i_{k-1}) \\ &= \frac{x_{i_k}}{\sum_{l \in N_k} x_l} \prod_{m=1}^{k-1} \frac{x_{i_m}}{\sum_{l \in N_m} x_l} \\ &= \prod_{m=1}^{k} \frac{x_{i_m}}{\sum_{l \in N_m} x_l} \end{aligned} \tag{9.10}$$

The base case $k = 1$ corresponds to the Gumbel-Max trick. The derivation presented here follows the approach by Kool et al. 2019.

9.2.4 *Gumbel-Dequantization*

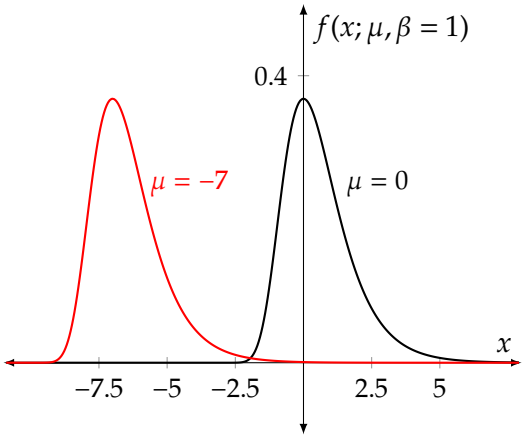

Figure 9.3: Gumbel PDF with $\mu = 0$ and $\beta = 1$.

The data consists of samples drawn without replacement. For example, with $n = 3$ and $k = 2$, the possible samples are $\{(1,1,0),(1,0,1),(0,1,1)\}$. The output space for the model can be defined to represent the probabilities of each class, utilizing the softmax function. For $n = 3$, the output is represented as (π_1, π_2, π_3).

A challenge arises during the sampling stage. By using the Gumbel top-k trick, the unnormalized log probabilities are manipulated by adding values sampled from the Gumbel distribution. Formally, this is expressed as:

$$(\log x_1 + g^{(1)}, \log x_2 + g^{(2)}, \log x_3 + g^{(3)}) = (g^{(1)}_{\log x_1}, g^{(2)}_{\log x_2}, g^{(3)}_{\log x_3}) \tag{9.11}$$

Consider an extreme example where only one valid option exists in the data, such as $(1,1,0)$. The distorted unnormalized probabilities are $(g^{(1)}, g^{(2)}, g^{(3)}_{-\infty})$. The last value poses a problem, as a center of $-\infty$ is not defined. To address this, a small probability ϵ is introduced for incorrect sampling, resulting in probabilities of $(g^{(1)}, g^{(2)}, g^{(3)}_{\log \epsilon})$.

In real datasets, where multiple combinations exist, this sampling and value-setting process is applied to each value, thus achieving dequantization. The inverse process, quantization,

involves selecting the top-k values. There is always a minor probability (corresponding to the value of ϵ) of selecting an incorrect value, which is accepted.

The dequantized values can then be learned as before. For visualization purposes, the densities of the Gumbel distribution around 0 are presented, with $\log \epsilon = -7$, in Figure 9.3.

9.3 CaloHit Model

9.3.1 *Energy Layer Flow Model*

The first component of the CaloHit Model is the *Energy Layer Flow*, designed to learn the joint distribution of the summed energies and the number of hits in the five longitudinal layers of the calorimeter detector. This model is crucial due to the varying granularity across the detector layers, which results in significantly different distributions of hits and energy in each layer. Accurately modeling these distributions is essential for capturing the complex energy deposition patterns within the detector, which, in turn, affects the overall performance of the simulation.

The architecture of the Energy Layer Flow model is inspired by the CaloFlow architecture, which also begins with a model that captures the layer-wise distributions. A coupling flow is incorporated to transform a 10-dimensional feature vector, consisting of the energies and the number of hits in each of the five layers (i.e., five layer energies plus five layer hit counts). The coupling flow is well-suited for capturing complex dependencies and interactions between these features.

To condition the model on the incident energy of the particle, an embedding table is used to map each discrete incident energy value to a 32-dimensional vector. The dataset contains 15 discrete incident energy values, and each is represented by a unique embedding vector. This embedded incident energy vector captures the specific characteristics associated with each energy level and allows the model to adapt its transformations accordingly.

The Energy Layer Flow model incorporates a coupling flow with 12 sequential transformations. Each transformation applies a univariate monotonic RQS, allowing for flexible and expressive transformations of the data. In this setup, transformations with 16 knots spanning the relevant data range are utilized, enabling the model to capture complex, nonlinear relationships within the features.

The parameters of the coupling flow are predicted by a series of MLPs. The dimensions of the layers in these MLPs are detailed in Table 9.2. The input layer receives the concatenation of the five conditioning features (either energies or number of hits) and the 32-dimensional embedded incident energy vector. The MLPs then process this input through successive layers of increasing dimensions, culminating in the parameters required for the coupling transformations. The final layer outputs the parameters for the transformations applied to the other five features.

The number of hits in each layer is dequantized using the CDF-Dequantization method, as

Table 9.2: Dimensions of the layers of the MLPs used in the Energy Layer Flow.

Layer	Input	1	2	3	4	Transf. Param.
Dimensions	5 + 32	64	128	256	256	5×47

described in Section 8.6.2. This technique facilitates handling discrete variables within the flow-based model by mapping them into a continuous space, allowing the flow transformations to be applied.

One challenge faced by the model is learning the zero-energy entries, which are common in layers with no energy deposition. The model tends to struggle with zero energies due to the nature of the transformations applied. To mitigate this, a small amount of noise is added to each layer with zero energy. This noise follows a log-normal distribution with a standard deviation of 10^{-3}. Since layers with zero energy also have zero hits ($n_{\text{hits}} = 0$), these layers can be easily identified for noise addition.

The layer energies undergo several preprocessing steps before being input to the model. First, they are min-max scaled to map the energies into a specific range, typically between 0 and 1. Next, a logit transformation is applied. The logit transform is defined as

$$\text{logit}(x) = \log\left(\frac{x}{1-x}\right), \tag{9.12}$$

which maps the scaled energies from the interval $(0,1)$ to the real line. Finally, the energies are normalized by standardizing them to have zero mean and unit variance.

The Energy Flow model serves as a foundational step in the CaloHit Model, providing the layer-wise energy and hit distributions that are essential for the subsequent modeling of individual hits within the detector.

9.3.2 *Hit Flow Model*

The second model, termed the *Hit Flow*, is designed to learn the distribution of hits within a shower event. The architecture of this model is also based on a coupling flow.

The output of the model consists of the Gumbel-dequantized values for the 368 calorimeter cells, representing whether each cell is hit.

To Gumbel-dequantize the hits, a Gumbel-distributed value is sampled for each cell in each data point. If a cell is not hit, the value is shifted by minus 7. The distribution is then normalized in each dimension.

The model is conditioned on the incident energy, as well as the number of hits and energies in each layer. Each input feature is mapped to a 128-dimensional context space. The mapped input features are summed to construct the conditional vector.

The Hit Flow model incorporates a coupling flow with eight sequential transformations.

Each transformation facilitates a univariate monotonic RQS transformation with four knots spanning the range between -5 and +5.

The parameters of each transformation are predicted by an MLP. The MLPs receive as input half of the features (those not being transformed) and the transformed incident energy vectors. The dimensions of the layers used in the MLPs are shown in Table 9.3.

Table 9.3: Dimensions of the layers of the MLPs used in the Hit Flow.

Layer	Input	1	2	3	4	Transf. Param.
Dimensions	185+32	512	512	512	512	$185 \cdot 11$

A relatively simple coupling flow architecture has been chosen for this model to present the overall concept of the CaloHit model.

The process of sampling from the Hit Flow model involves several steps. First, 368-dimensional vectors are sampled from the coupling flow. The predicted number of hits in each layer is then used to determine how many cells in the calorimeter were hit during the event.

Finally, the normalization of the residual values is reversed to restore their original scale. For each layer, the top-n_{hits} cells are set to one, while all remaining cells are set to zero.

9.3.3 *Energy Per Hit Model*

The *Energy Per Hit* model is designed to predict the energy values associated with each hit coordinate in a calorimeter shower. This model operates under the assumption that each energy value is independent of the others, thereby allowing for a simplified architecture that is both efficient and fast, albeit with the limitation of not capturing interactions between energy values.

The Energy Per Hit model employs a univariate flow featuring four sequential transformations. The first and last transformations are monotonic affine transformations, which provide linear shifts and scaling to the energy values. The two middle transformations are monotonic RQS transformations. The rationale for this setup is that the affine transformations adjust the scale and shift of the distribution of features, allowing the rational quadratic splines to perform more effective transformations within their defined range.

The model includes a single MLP to process all conditional features and map them into a shared latent space. All parameters of the univariate transformations are affine mappings of the latent vector. The dimensionality of the MLP is provided in Table 9.4.

Table 9.4: Dimensions of the layers of the MLP used in the Energy Per Hit Flow.

Layer	Conditional	1	2	3	4	5	Latent
Dimensions	1024	512	256	128	64	64	64

The model conditions the energy predictions on both global and local features. The global features include the input energy, the total number of hits, the energies of all layers, and the overall hit profile within the shower. The specific coordinate of the hit is used as a local feature. The global features are embedded into higher-dimensional spaces. The input energy and number of hits are embedded into 1024-dimensional vectors, while the hit profile is mapped into a 1024-dimensional space by an affine transformation. The energies in each layer and the coordinate of the hit are also embedded into a 1024-dimensional space. All global information is added to the embedded coordinate vector. These vectors are then used as the inputs for the MLP.

The generated energy predictions are first divided by the energy sum. The energies are then min-max scaled within each voxel, resulting in a distribution bounded between 0 and 1. Subsequently, a logit transformation is applied, and the values are normalized. This process mirrors the approach used in the Carlo Flow architecture, ensuring consistency with previous models while adapting it to the unique requirements of energy prediction.

9.4 Evaluation

The evaluation of the CaloHit architecture follows a structure similar to that employed for the CaloFlow model. Simulated showers generated using Geant4 are directly compared to those generated by the CaloHit model, with the objective of making the CaloHit model's output indistinguishable from the Geant4 simulations. As in previous evaluations, the bootstrapping approach (Efron 1979) is employed to assess the uncertainties. Specifically, 1,000 bootstrapped datasets, each containing 121,000 points, are evaluated. Differences between datasets are reported using the Normalized Difference, and the X^2 is reported for all histograms.

9.4.1 *Cell Energy*

The calorimeter showers are defined in terms of the energy values $E_{i,z,\alpha,r}$, where the index i ranges from 1 to $n = 121,000$, representing individual showers. The z index denotes the layer of the cell, with values ranging from 1 to $n_z = 5$. The angular (α) and radial (r) indices vary depending on z, and their maximum values n_α and n_r are detailed in Table 9.5.

Table 9.5: The maximum of the index j defined as n_j as depedent on z.

z	$n_\alpha(z)$	$n_r(z)$
1	1	8
2	10	16
3	10	19
4	1	5
5	1	5

The marginal distribution of the $E_{i,z,\alpha,r}$ values is shown in Figure 9.4. A total of 100 bins are set logarithmically between the 1st and the 99th percentile, including both overflow and

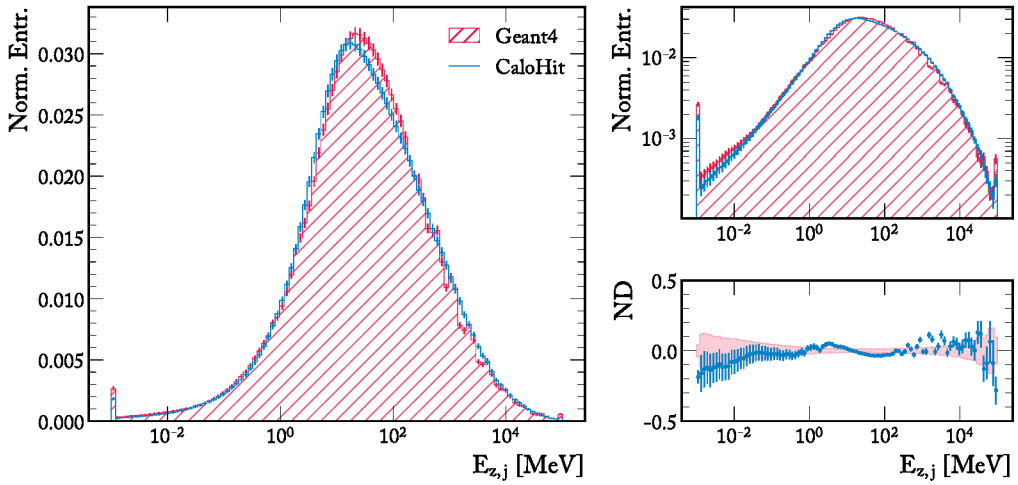

Figure 9.4: Cell energy distributions for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the normalized entries of the cell energy values, binned logarithmically. The entries are normalized to sum to 1. The right upper graph shows the same data as the left graphs, but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

underflow bins. The marginal energy distribution shows a slight shift between the models, and the ND of the CaloHit model overlaps with the Geant4 uncertainty band. This indicates that the CaloHit model does not perfectly model the energy distribution, with the X^2 remaining within a factor of two of the Geant4 values, as illustrated in Table 9.6.

Table 9.6: X^2 for Geant4 and the CaloHit model on Dataset I of the histograms shown in Figure 9.4.

Geant4	**CaloHit**
$(9.81 \pm 1.44) \times 10^{-4}$	$(3.47 \pm 0.33) \times 10^{-3}$

9.4.2 *Energy Sum*

The energy sum $E_{i,\text{sum}}$ is defined as follows:

$$E_{i,\text{sum}} = \sum_{z=1}^{n_z} \sum_{\alpha=1}^{n_\alpha(z)} \sum_{r=1}^{n_r(z)} E_{i,z,\alpha,r}.$$

In Figure 9.5, a histogram of the energy sum normalized by the incident energy is presented:

$$\frac{E_{i,\text{sum}}}{E_{i,\text{in}}}. \tag{9.13}$$

The binning consists of 100 linearly spaced bins between the 1st and 99th percentiles, including both overflow and underflow bins. The distribution peaks slightly below 1, indicating a small discrepancy.

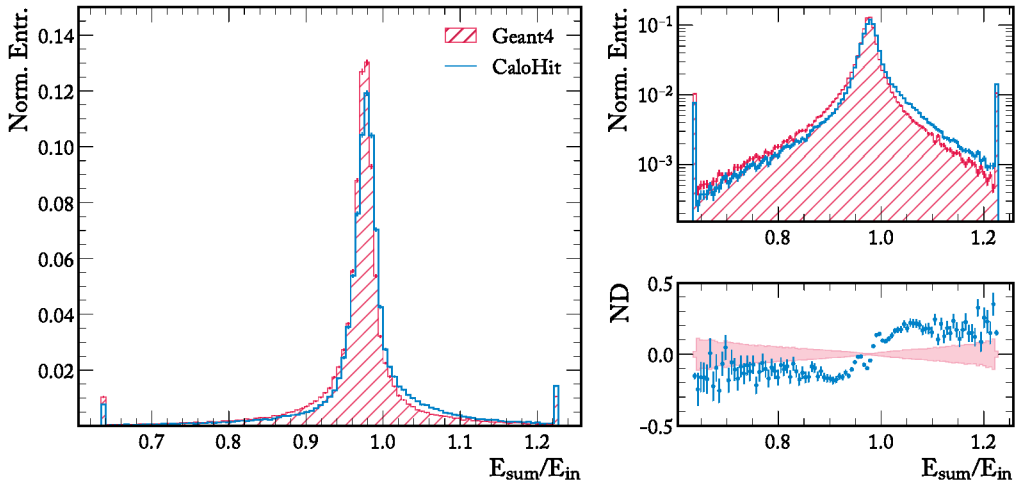

Figure 9.5: Energy sum distributions for Dataset I comparing Geant4 and the CaloHit model. The left graphs show the normalized entries of the energy sum, linearly binned. The entries are normalized so that the sum of all bins equals 1. The right upper graphs present the same data as the left graphs but with a logarithmic scale on the y-axis. The right lower graph show the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The CaloHit model closely replicates the overall distribution, although minor mismodeling is observed near the center. The tails of the distribution generally align with Geant4 within the margins of error. This alignment is further confirmed by the X^2, shown in Table 9.7, which remains within a factor of four above the Geant4 baseline.

Table 9.7: X^2 for Geant4 and the CaloHit model on Dataset I of the histograms shown in Figure 9.5.

Geant4	**CaloHit**
$(9.82 \pm 1.44) \times 10^{-4}$	$(2.59 \pm 0.10) \times 10^{-2}$

9.4.3 *Number of Hits*

The number of hits $n_{i,\text{hits}}$, as defined in Equation (8.23), is shown in Figure 9.6. This metric quantifies the number of voxels in the calorimeter that register an energy value greater than the threshold energy. The distribution of $n_{i,\text{hits}}$ ranges from scenarios with no hits to cases where all cells are hit. The histogram exhibits a crowded spectrum, with the bin representing the entire detector being hit containing the most entries.

Distinct peaks are observed within the distribution, corresponding to different discrete incident energies in the spectrum. The CaloHit model reproduces the overall shape of the distribution relatively well. No significant mismodeling is visible.

The X^2, as shown in Table 9.8, provides a quantitative measure of these discrepancies. Despite the overall good agreement, the X^2 indicates that there is room for improvement in the model's ability to replicate the fine structure observed in the Geant4 data.

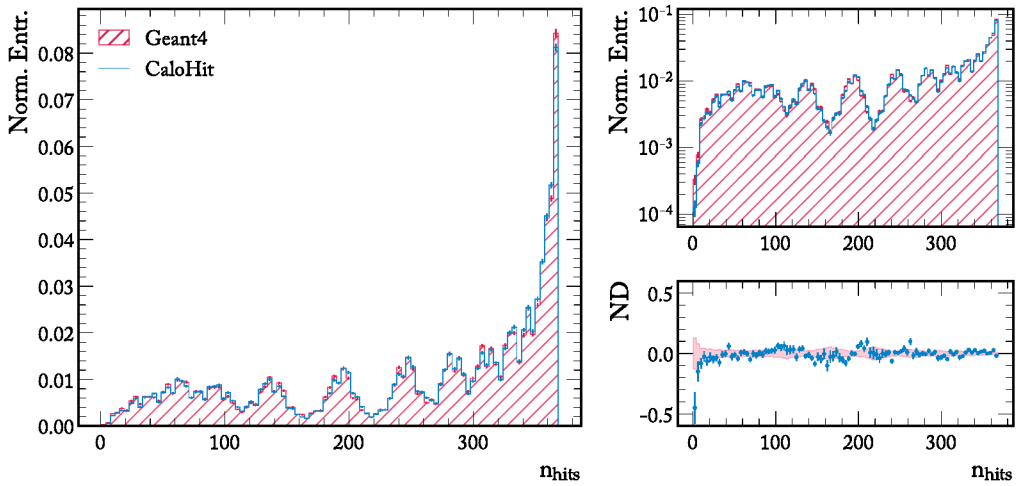

Figure 9.6: Number of hits distributions for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the normalized entries of the number of hits, linearly binned. The entries are normalized so that the sum of all bins equals 1. The right upper graph shows the same data as the left graphs, but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

Table 9.8: X^2 for Geant4 and the CaloHit model on Dataset I of the histograms shown in Figure 9.6.

Geant4	**CaloHit**
$(9.74 \pm 1.43) \times 10^{-4}$	$(2.98 \pm 0.31) \times 10^{-3}$

The number of hits also reflects the observed shifts in other metrics, such as the cell energy distribution. Both the number of points generated and the total energy sum are determined by the model, and during post-processing, the energy of all points is scaled to match the total energy sum. If the number of points is reduced, the remaining points are scaled excessively, leading to distortions in the energy distribution, which are reflected in the number of hits.

9.4.4 *Average Energy Sum*

The total energy in a longitudinal layer z is defined as

$$E_{i,z} = \sum_{\alpha=1}^{n_\alpha} \sum_{r=1}^{n_r} E_{i,z,\alpha,r}. \tag{9.14}$$

The average energy in a layer z, $\bar{E}_z$, is computed using Equation (8.25). The average energy sum in each longitudinal layer z for Dataset I is presented in Figure 9.7.

The distribution of average energies across the layers reveals a prominent feature: the energy deposited in the central layer, $z = 3$, is significantly higher than in the other layers.

The CaloHit model demonstrates strong performance in modeling the energy distribution across all longitudinal layers. The average energy values closely align with the Geant4 data,

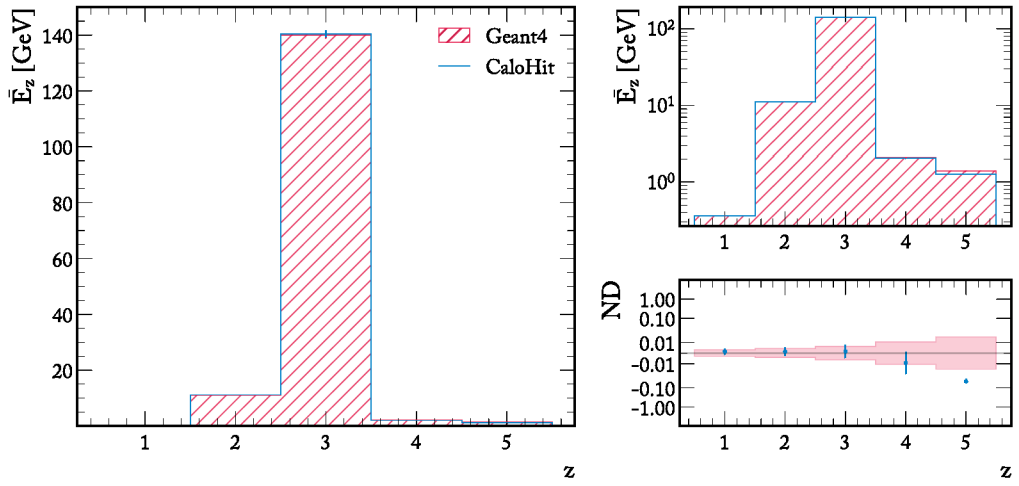

Figure 9.7: Average energy in each longitudinal layer z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the average energy values in MeV. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

particularly at the central energy peak where the energy deposition is highest. In most layers, the CaloHit model effectively replicates the energy deposition observed in the Geant4 simulations, indicating a high level of accuracy in capturing the expected energy patterns.

A slight discrepancy is observed in the final layer, where the CaloHit model marginally underestimates the energy compared to the Geant4 data. This minor difference is reflected in the Geant4 values, which remain low across all layers but show a small increase in the final layer. Despite this slight deviation, the overall agreement between the CaloHit model and Geant4 is excellent, suggesting that the model effectively captures the energy distribution throughout the detector.

9.4.5 *Energy Sum Distribution*

In contrast to the previous analysis, which focused solely on the mean values of energy sums, this section investigates the complete distribution across all five longitudinal layers. This detailed approach provides a better understanding of how the CaloHit model captures the energy distribution within each layer, highlighting specific areas where the model may excel or require improvement.

For the first index ($z = 1$) in Figure 9.8, no significant shift in the energy values is observed. This indicates that the CaloHit model accurately replicates the energy distribution in this layer. The entire distribution is well-modeled, as is particularly evident in the ND plot, which shows no significant deviations from the Geant4 reference data.

For the second index ($z = 2$), shown in Figure 9.9, the energy distribution reveals multiple peaks. The CaloHit model captures the general shape of the distribution effectively.

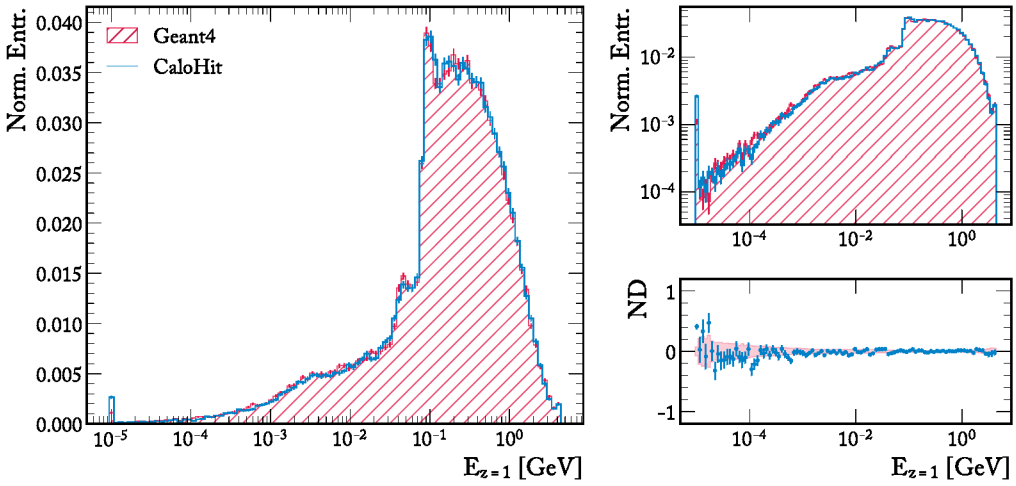

Figure 9.8: Distribution of the energy sum in the first index in longitudinal direction z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the average energy values in MeV. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

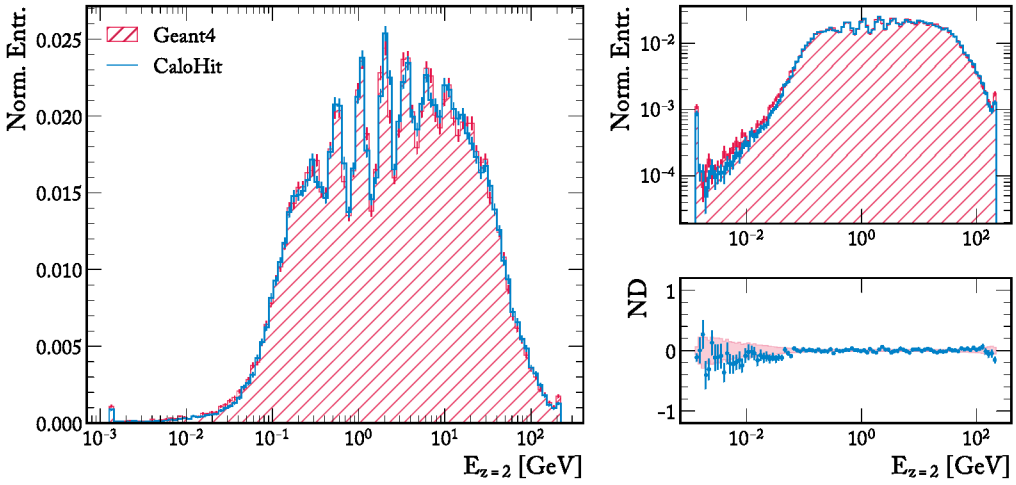

Figure 9.9: Distribution of the energy sum in the second index in longitudinal direction z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the average energy values in MeV. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

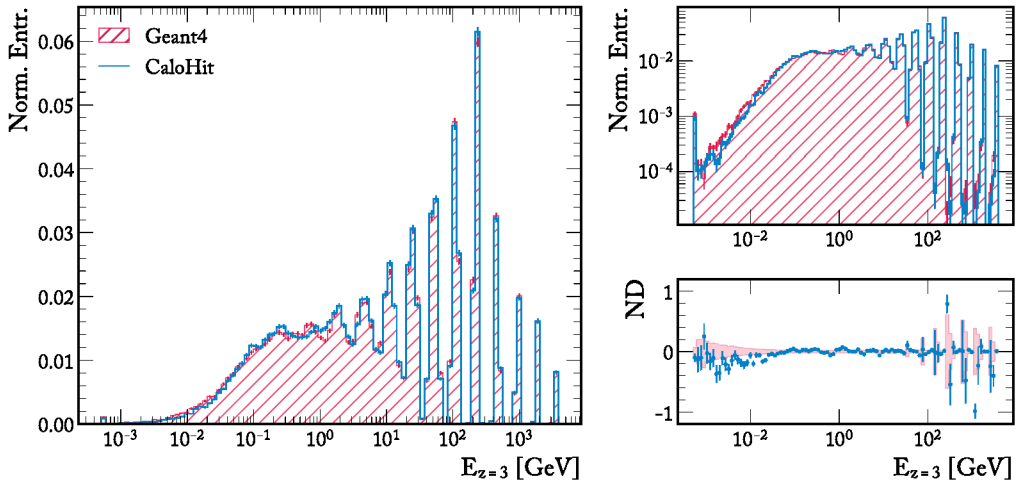

Figure 9.10: Distribution of the energy sum in the central index in longitudinal direction z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the average energy values in MeV. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The central index ($z = 3$), depicted in Figure 9.10, shows a combination of a bulk section and several peaks. The CaloHit model accurately models this distribution, performing as well as in the other indices. The bulk section is reasonably captured, and the peaks are well-matched, with the model nearly perfectly aligning with the Geant4 data.

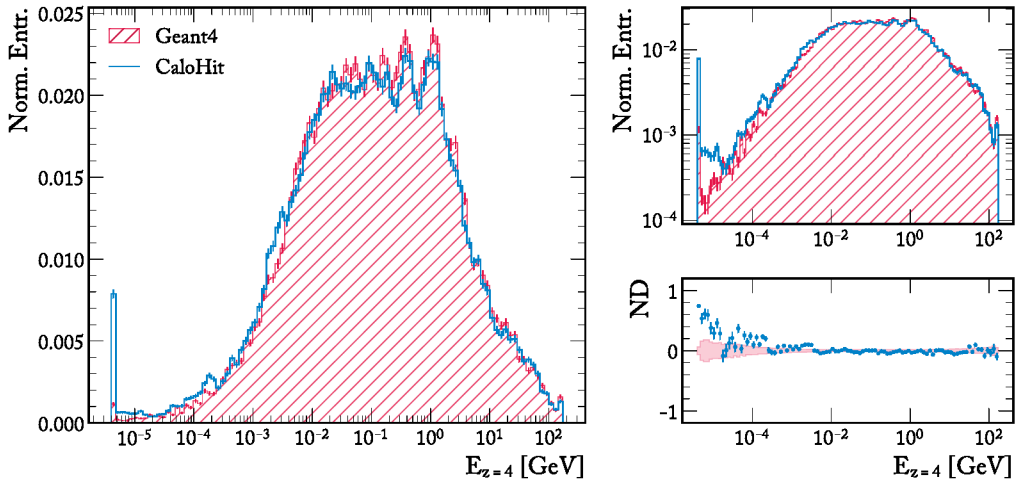

Figure 9.11: Distribution of the energy sum in the fourth index in longitudinal direction z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the average energy values in MeV. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

For the fourth index ($z = 4$), illustrated in Figure 9.11, the energy distribution is reasonably

well-modeled by the CaloHit model, although some discrepancies are present that are not observed in the Geant4 data. The model exhibits a tail towards lower energies that is not present in the Geant4 data, as seen in the underflow bin.

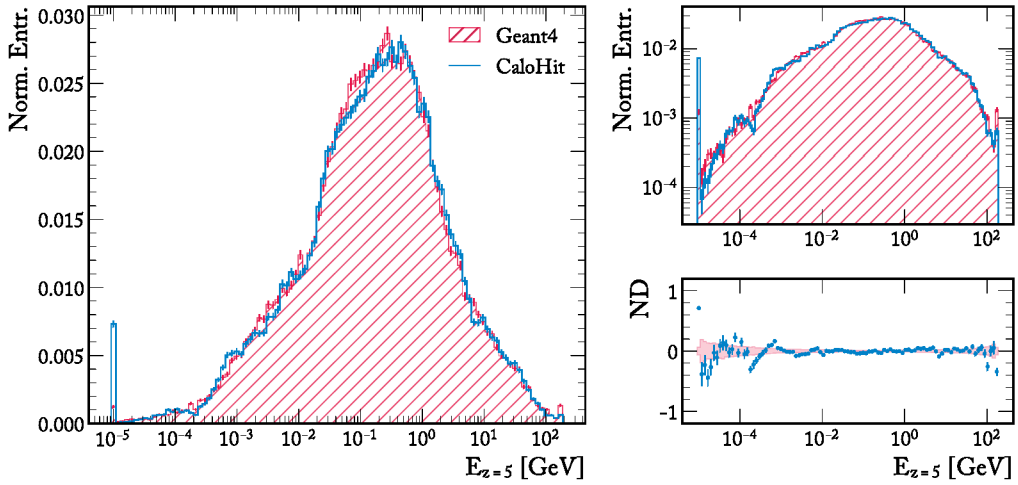

Figure 9.12: Distribution of the energy sum in the last index in longitudinal direction z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the average energy values in MeV. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The energy sum distribution for the last index ($z = 5$) is presented in Figure 9.12. Here, the CaloHit model performs relatively well; however, a noticeable tail towards lower energies is observed compared to the Geant4 data. This shift results in the model peaking in the underflow bin.

The X^2 values for all z layers, shown in Figure 9.13, confirm the observations made in the individual layer analyses. The X^2 is less than one order of magnitude higher than the expected band for the Geant4 data. While the CaloHit model generally captures the energy distribution well, there remains room for improvement, particularly in the tails in the last two layers.

9.4.6 *Shower Centers*

In Figure 9.14, the histogram of shower centers μ_z is presented. The Geant4 distribution exhibits a characteristic exponential rise, peaking just before $z = 3$, followed by an exponential decline. This pattern reflects the distribution of energy within the detector, with the peak corresponding to the densest region of energy deposition.

The CaloHit model replicates this distribution accurately, with only minor differences observed in the underflow and overflow bins, which are slightly smaller. This agreement is further emphasized in the ND plot, where no significant deviations from the Geant4 reference are evident.

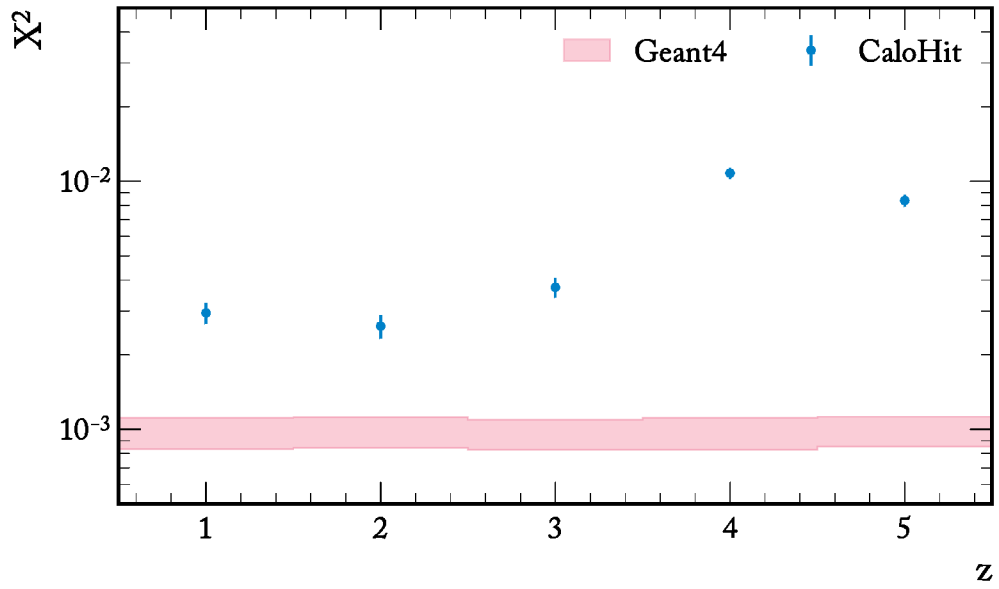

Figure 9.13: X^2 of the energy histograms in each longitudinal layer z for Dataset I. The histograms used to calculate the X^2 are shown in Figures 9.8 to 9.12. The values for the CaloHit model are shown as blue with error bars indicating the standard deviation of the X^2. The red error band illustrates the uncertainty in the Geant4 data.

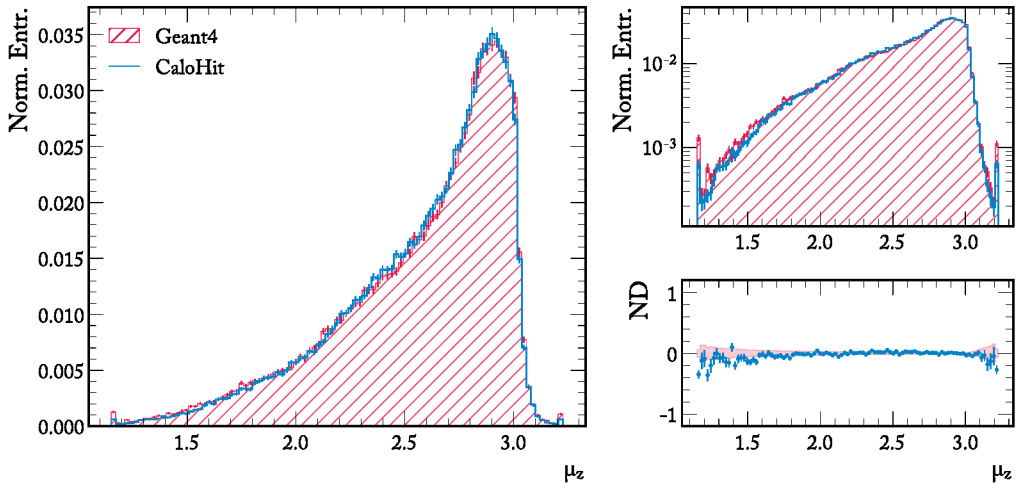

Figure 9.14: Shower centers μ_z in longitudinal direction z for Dataset I comparing Geant4 and the CaloHit model. The left graph shows the histogram of the shower centers. The right upper graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The right lower graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

Table 9.9: X^2 for Geant4 and the CaloHit model on Dataset I of the histograms shown in Figure 9.14.

Geant4	CaloHit
$(9.79 \pm 1.42) \times 10^{-4}$	$(2.87 \pm 0.29) \times 10^{-3}$

The X^2, shown in Table 9.9, quantitatively supports these observations. The X^2 values are not significantly higher for the CaloHit model, indicating that there is no substantial room for improvement in replicating the shower centers in the z direction.

9.4.7 *Shower Width*

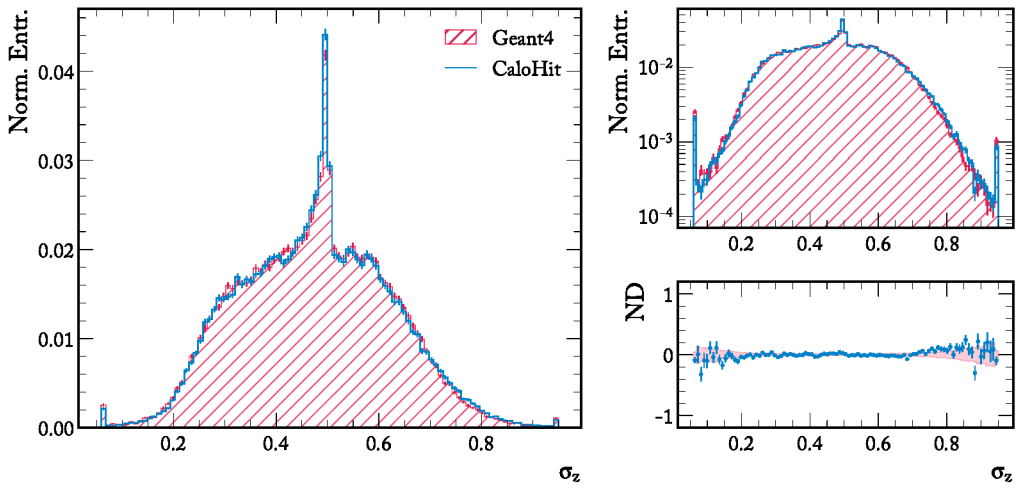

Figure 9.15: Shower width σ_z in direction z for Dataset I comparing Geant4 and CaloHit model. The left graph shows the histogram of the shower width. The upper right graph presents the same data as the left graph but with a logarithmic scale on the y-axis. The lower left graph shows the ND between the models and Geant4. The red band shows the in-sample uncertainty of Geant4.

The shower width in the longitudinal z direction is a critical metric for evaluating the spread of energy within a calorimeter shower. It is defined in Equation (8.31). This formula quantifies the standard deviation of the energy distribution along the z axis, providing insight into how concentrated or dispersed the energy is within a given shower.

The distribution of the shower width σ_z for Dataset I is presented in Figure 9.15. The Geant4 distribution typically increases to a plateau before gradually decreasing, with a pronounced peak on the plateau. This shape reflects the expected distribution of energy spread across the longitudinal direction, where the showers maintain a consistent width before tapering off.

The CaloHit model demonstrates behavior similar to that of Geant4. The distribution closely follows the Geant4 reference, indicating that the CaloHit model effectively captures the energy spread within the showers, particularly in the region where the distribution stabilizes.

The ND plot further illustrates this agreement, showing no significant deviations between the CaloHit model and the Geant4 reference data. This agreement is quantitatively reflected

in the X^2 values, as shown in Table 9.10. The X^2 for the CaloHit model is only slightly higher, indicating good performance in modeling the shower width compared to Geant4.

Table 9.10: X^2 for Geant4 and the CaloHit model on Dataset I of the histograms shown in Figure 9.15.

Geant4	**CaloHit**
$(9.89 \pm 1.41) \times 10^{-4}$	$(2.52 \pm 0.03) \times 10^{-1}$

9.4.8 *Correlation Coefficients*

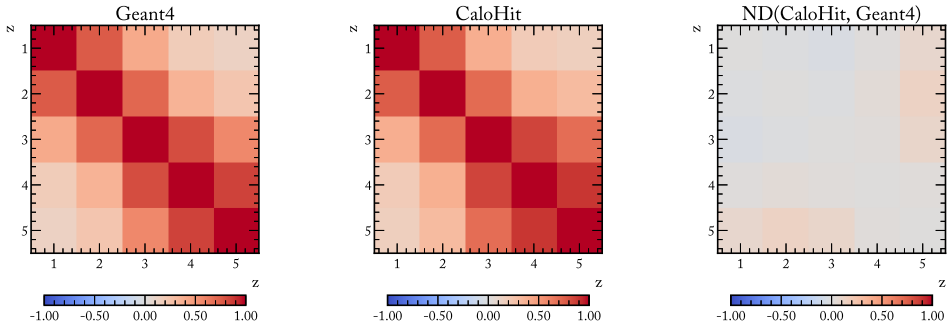

Figure 9.16: Pearson correlation coefficients ρ in direction z for Dataset II comparing Geant4, CPF I, and CPF II models. Upper row shows correlation coefficients. Lower row shows the ND of the correlation coefficients

To assess the internal consistency of the energy deposition within the calorimeter, the Pearson correlation coefficients ρ_{z_j,z_k} between different longitudinal layers, z_j and z_k, are examined. The correlation coefficient, which quantifies the linear relationship between the energy deposits in two layers, is defined in Equation (8.36).

In Figure 9.16, the correlation coefficients ρ_{z_j,z_k} for Dataset I are presented, comparing Geant4 with the CaloHit model. The Geant4 data exhibit the highest correlations between adjacent layers, which gradually decrease as the distance between the layers increases. This pattern reflects the localized energy deposition within the particle showers. The CaloHit model generally captures this trend but shows some discrepancies, particularly in the outer layers where the correlations deviate more significantly from the Geant4 data.

To quantify these discrepancies, the Frobenius norm of the ND of the correlation matrices is computed, providing a measure of the overall difference between the model and the reference data. The equation is given in Table 9.11.

Table 9.11: Frobenius norm of the ND between the ρ values for the Geant4 dataset and Calohit model shown in Figure 9.16.

Geant4	**CaloHit**
$(7.69 \pm 3.76) \times 10^{-2}$	$(5.01 \pm 0.44) \times 10^{-1}$

9.4.9 *Classifier Scores*

In evaluating the performance of calorimeter simulation surrogate models, classifier scores are utilized as a key metric. As discussed in Section 8.8.9, these scores are derived using binary classifiers that differentiate between simulated showers generated by the CaloHit model and those produced by Geant4. Two types of classifiers are employed:

1. **Low-level classifier:** This classifier relies solely on the energy values $E_{i,z,\alpha,r}$ within each shower.
2. **High-level classifier:** This classifier incorporates both the low-level energy values and additional features, such as the ratio $E_{\text{sum}}/E_{\text{in}}$, the shower centers $(\mu_z, \mu_r, \mu_y, \mu_x)$, the shower widths $(\sigma_z, \sigma_r, \sigma_y, \sigma_x)$, and the sparsity.

The classifiers' effectiveness is assessed using the AUC, with a score of 0.5 indicating random classification and 1.0 representing perfect separation between the two distributions.

For Dataset I, the classifier scores are presented in Table 9.12. The high AUC scores for both low-level and high-level classifiers indicate that the classifiers can almost perfectly distinguish between the CaloHit-generated samples and those from Geant4.

Table 9.12: Low Level and High Level CaloChallenge Classifier AUC Scores for the CaloHit model on Dataset I.

low level	high level
0.9871 ± 0.0177	0.8268 ± 0.1046

The near-perfect scores observed in the low-level classifier are likely attributable to its sensitivity to subtle features. Although the high-level classifier demonstrates improved performance, it still does not achieve perfect accuracy, suggesting that some residual discrepancies between the datasets remain detectable. This indicates that, despite the improved alignment of energy distributions, further refinement of the CaloHit model is necessary to minimize these detectable differences, particularly at higher classification levels.

10 CONCLUSION

This thesis has presented a comprehensive investigation into calorimeter surrogate models facilitating point clouds, particularly focusing on the CaloPointFlow and CaloHit models.

The research began with the development of generative models based on voxelized calorimeter data by initially experimenting with a progressive growing approach, starting with a low granularity representation and progressively increasing the granularity until reaching high-resolution calorimeter shower data.

However, it quickly became apparent that scaling this approach posed significant challenges. The primary issue was that in highly granular calorimeter data, the showers exhibit considerable sparsity. When attempting to downscale the granularity by clustering the cells, this sparsity became altered, complicating the modeling process. Despite various attempts, a convincing strategy to effectively manage this sparsity could not be devised.

In response, a different approach was pursued, interpreting the calorimeter showers as point clouds and exploring point cloud generative models. This led to the development and publication of one of the first models utilizing this methodology: the CaloPointFlow architecture.

Nevertheless, this model had limitations. The most significant drawback was the absence of a mechanism for direct point-to-point information exchange within the architecture. Additionally, the coordinate positions of the points were discrete, but the model was designed for continuous data, necessitating a mapping between discrete and continuous spaces. A major challenge arose from the data structure itself, which allowed only one hit per calorimeter cell, a constraint that was difficult to model accurately.

To address these limitations, the second iteration, the CaloPointFlow II architecture, was developed. This version introduced a new normalizing flow architecture for point clouds, named DeepSetFlow, which incorporates deep sets within the coupling layers to facilitate point-to-point information exchange. A novel dequantization strategy, CDF-Dequantization, was also implemented, significantly improving the mapping between discrete cells and continuous space. Additionally, a mitigation strategy specifically designed to address the multiple hit problem was employed.

A thorough evaluation of both models was conducted, employing a bootstrapping approach to assess uncertainties and identify areas for improvement. Although substantial progress was made, the issue of multiple hits per cell persisted, and efforts continued to design an architecture that inherently produced only one hit per cell.

This effort culminated in the development of the CaloHit architecture, which represents a hybrid approach combining aspects of both point cloud-based and voxelized methods. The generation process was divided into two stages: first, generating the hitmap using a voxelized approach, and second, generating the energies for the hits using a point cloud-based approach. In this model, the positions of the points were predetermined, and only the energies were generated, allowing effective sampling of all calorimeter cells without replacement

using the Gumbel Top-k trick.

Preliminary tests of this approach, conducted on a basic model with the first dataset from the CaloChallenge, demonstrated its potential. This approach appears highly promising, but scaling it to handle higher dimensionality remains a challenge, particularly in terms of scaling the hitmap approach. A proposed future direction involves developing a low-granularity hitmap for all calorimeter cells, learning the hit regions, and focusing only on the hit cells for high-resolution modeling. This would enable the development of a sparse super-resolution approach. Furthermore, while the current energy model is relatively simple, incorporating more sophisticated models, such as those based on diffusion or conditional flow matching, could potentially enhance performance. Finally, it will be essential to test this approach in a real experimental setup to evaluate its performance, speed, and usability. With further refinement, this hit-based approach could offer a robust solution to the challenges encountered in calorimeter surrogate modeling.

In conclusion, this thesis has laid the groundwork for new directions in calorimeter surrogate modeling by introducing and iteratively refining models that leverage point cloud and voxelized data representations. The pursuit of a robust, scalable, fast, and experimentally validated calorimeter surrogate model remains an open challenge, but the progress made in this thesis represents a meaningful step towards that goal.

INVERSE TRANSFORMATION

A

A.1 Inverse Transformation for continuous distributions

Let's consider X as a random variable that has a cumulative distribution function (CDF) represented as $F_X(x) = P(X \leq x)$. The CDF of a random variable is an essential element in probability and statistics, as it provides the probability that a random variable will take a value less than or equal to a specific value.

By the properties of the CDF, we know that if $x < y$, then $F_X(x) \leq F_X(y)$, because the probability that X is less than or equal to x is always less than or equal to the probability that X is less than or equal to y. Moreover, F_X is strictly increasing where the probability density function $P_X(x) > 0$, indicating that the probability increases as the value of X increases.

Now, let's suppose that F_X is strictly increasing. Then, for any $u \in (0,1)$, the equation $F(x) = u$ has exactly one solution. We denote this solution as $x = F_X^{-1}(u)$, where F_X^{-1} is the inverse function of F_X. In such a situation, we can say that

$$\Phi_X^{-1}(u) = \inf\{x|F_X(x) \geq u\} = \inf\{x|F_X(x) = u\} = F_X^{-1}(u) \tag{A.1}$$

This shows that the Smirnov transformation is essentially the inverse of the CDF, provided F_X is strictly increasing.

Furthermore, the inverse function F_X^{-1} is also strictly increasing on the interval $(0,1)$. This is because if F_X is strictly increasing, then the inverse function will also preserve this property.

Let's now define a new random variable $Y = F_X^{-1}(U)$. For this random variable, we can express the cumulative distribution function of U as $F_U(F_X(x)) = P(U \leq F_X(x)) = F_X(x)$.

Given that F_X^{-1} is strictly increasing, we can proceed and apply this property to our inequality. Specifically, $P(U \leq F_X(x)) = P(F_X^{-1}(U) \leq F_X^{-1}(F_X(x)))$. Since $F_X^{-1}(F_X(x))$ simplifies to x, this can be re-written as $P(F_X^{-1}(U) \leq x) = P(Y \leq x) = F_Y(x)$. Hence, we have derived that $F_Y(x) = F_X(x)$.

In conclusion, given these results and by the definition of the equality of random variables, we can say that $X = Y$.

As we've established, the equality $X = F_X^{-1}(U)$ and its inverse $U = F_X^{-1}(X)$ provide an invertible mapping between the standard uniform distribution and any continuous distribution without gaps.

A.2 Inverse Transformation for discrete distributions

Suppose X is a discrete random variable with $p_i = P(X = x_i)$ for $i \in 1, ..., n$, where p_i is the probability that X equals x_i and n is the total number of possible outcomes. The cumulative distribution function $F_X(x_n)$ is given by $\sum_i^n p_i$.

Let's consider an arbitrary interval $[a, b]$ such that $0 \leq a \leq b \leq 1$. In the standard uniform distribution, the probability that U falls within this interval is

$$P(a \leq U \leq b) = P(U \leq b) - P(U \leq a) = b - a. \tag{A.2}$$

Now, for all indices $i < j$, we have $0 \leq F_X(x_i) \leq F_X(x_j) \leq 1$ due to the properties of the cumulative distribution function. Consequently, the probability that U lies between $F_X(x_i)$ and $F_X(x_{i+1})$ equals $P(F_X(x_i) \leq U \leq F_X(x_{i+1})) = F_X(x_{i+1}) - F_X(x_i) = p_{i+1}$. This probability is precisely the probability that the discrete random variable X equals x_{i+1}.

Based on these results, we can construct the following function that acts as the inverse transform for the discrete distribution

$$\Phi_X^{-1}(u) = \begin{cases} x_1, & \text{if } U \leq F_X(x_1) \\ x_2, & \text{if } F_X(x_1) \leq U \leq F_X(x_2) \\ \vdots & \\ x_n, & \text{if } F_X(x_{n-1}) \leq U \leq F_X(x_n) \end{cases}. \tag{A.3}$$

In other words, $\Phi_X^{-1}(u)$ assigns the value x_i to u if it falls within the interval $[F_X(x_{i-1}), F_X(x_i)]$. With this piece-wise defined function, we can express the inverse transform as

$$\Phi_X^{-1}(u) = \inf\{x_i | u \leq F_X(x_i)\} \tag{A.4}$$

which provides us with the required mapping from the standard uniform distribution U to the discrete random variable X. Hence, we have successfully demonstrated the inverse transform method.

ADDITIONAL RESULTS

B

B.1 Marginal Energy Distributions

In Figures B.1 to B.6 show marginal distributions of E_z, E_α, E_r for all possible values in Dataset 2 and 3. Each figure follows the typical design of this thesis. Comparisons are made between Geant4, CPF I, and CPF II. The upper sections of the figures present histograms of the values, where the lines follow the average of bootstrapped datasets, and the error bars show the standard deviation for bootstrapped datasets. The red dashed area shows the value for Geant4, the blue line indicates the value for CPFI, and the dark grey line represents the value for CPFII. Error bars are provided for each data point; however, in some cases, the error bars are so small that they are not visible. The lower half of the figures presents the ND as defined in Equation (8.16). The red areas illustrate the anticipated value for the ND of Geant4 bootstrapping samples in relation to one another. The blue dots represent the observed values of the ND between the CPF I and Geant4 data, while the dark grey values indicate the ND between the CPF II and Geant4 data. For enhanced visual clarity, we slightly moved the values for CPFI and CPFII left and right so that they do not overlap. This adjustment is solely for better visibility and has no other implications.

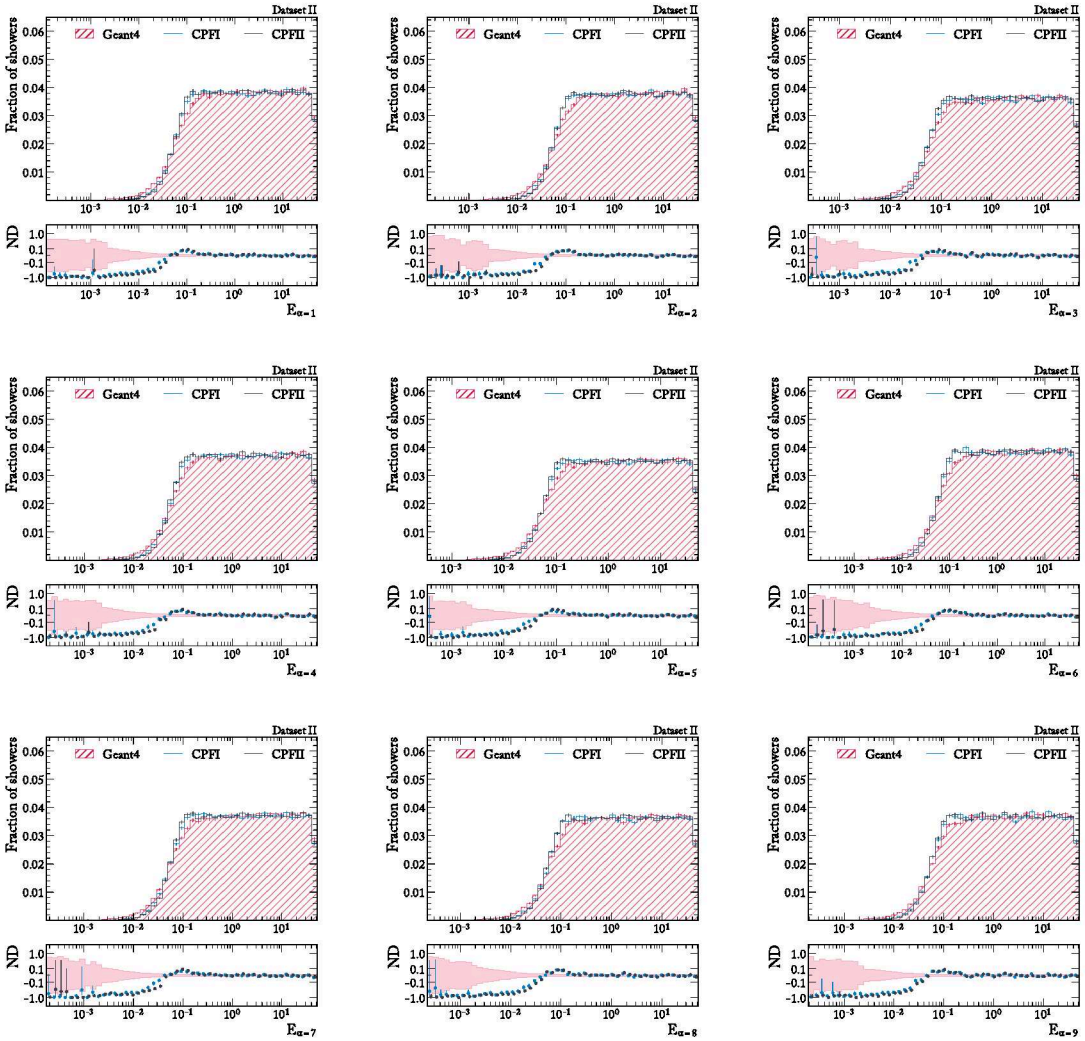

Figure B.1: The marginal energy distributions for the layers in α from 1 to 9 for Dataset II. See the accompanying text for an explanation of how to read the figures.

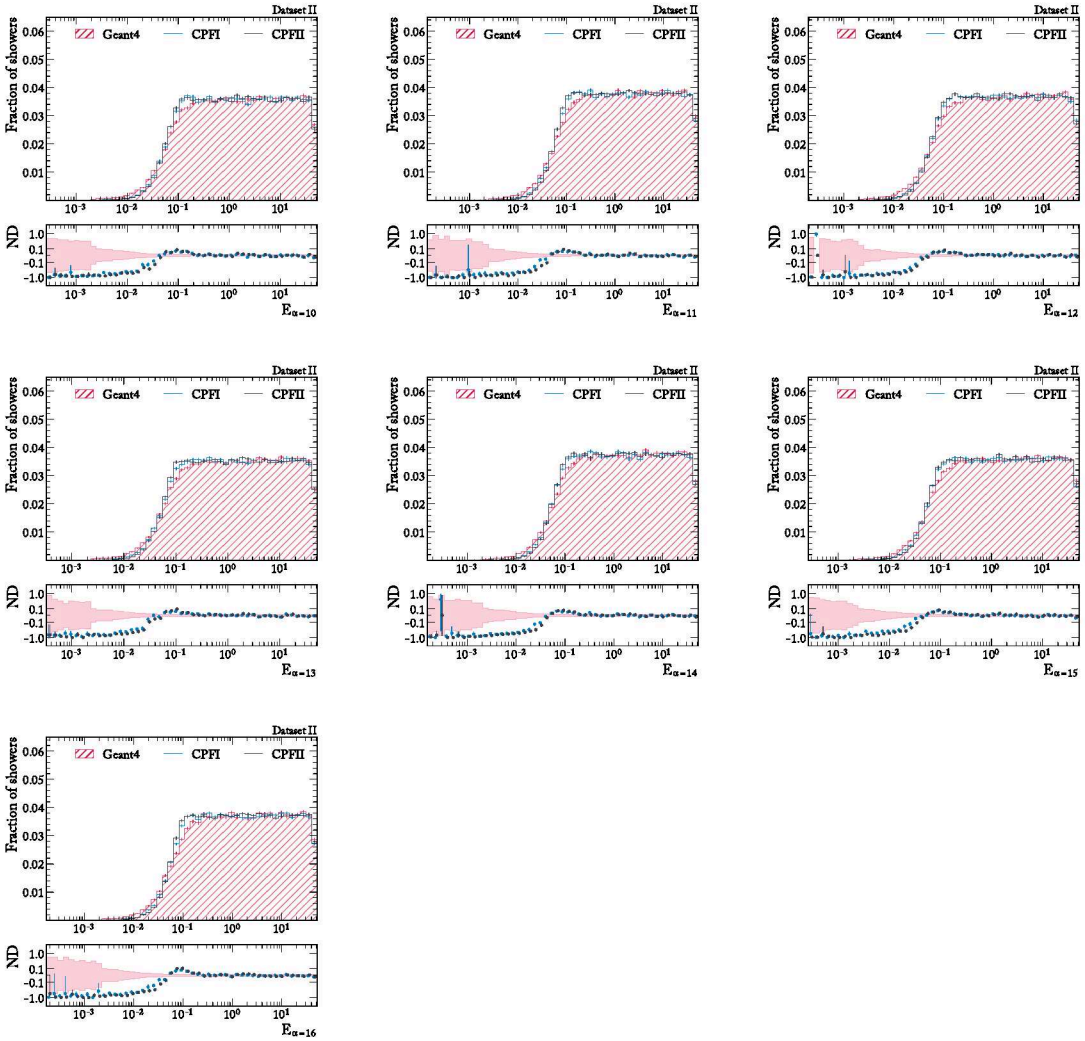

Figure B.2: The marginal energy distributions for the layers in α from 10 to 16 for Dataset II. See the accompanying text for an explanation of how to read the figures.

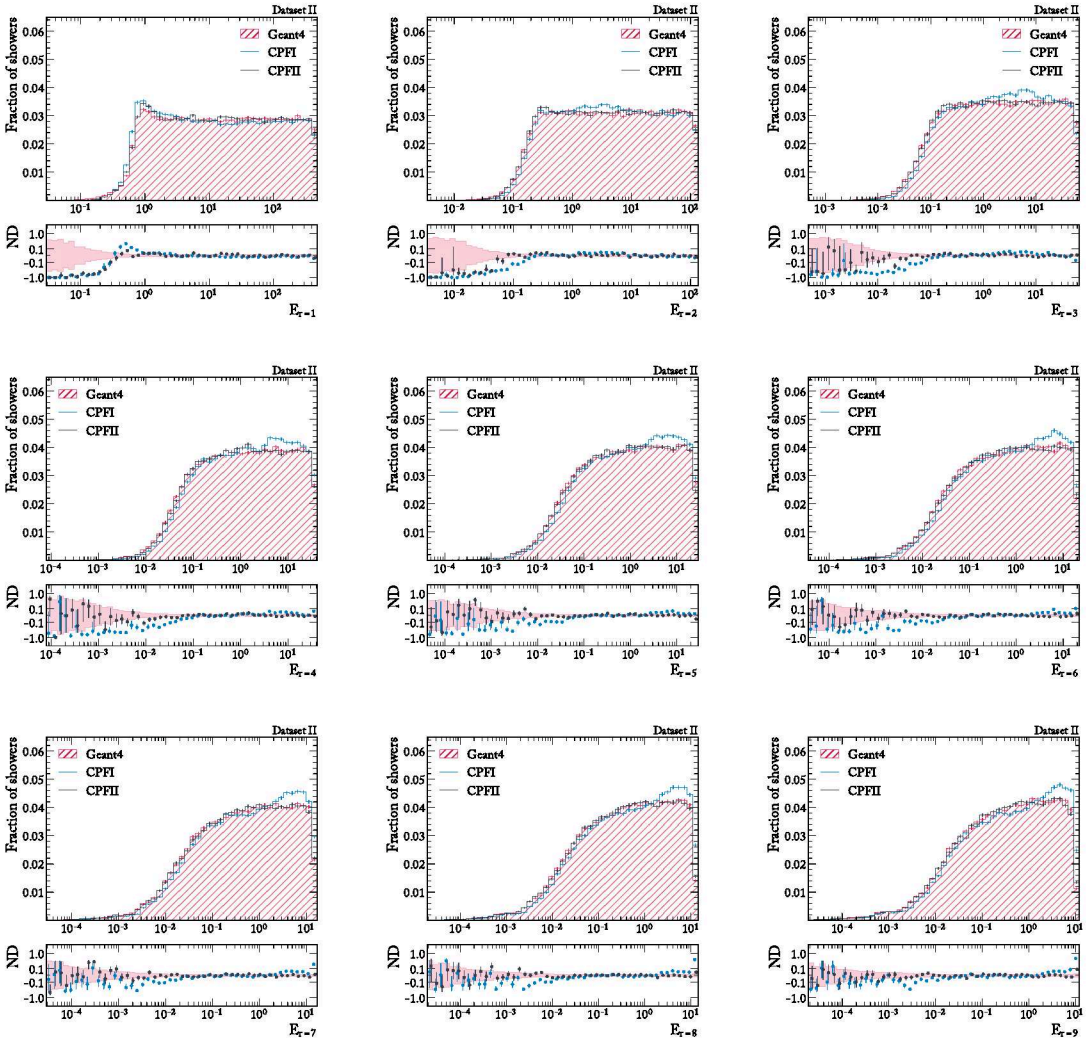

Figure B.3: The marginal energy distributions for all layers in r for Dataset II. See the accompanying text for an explanation of how to read the figures.

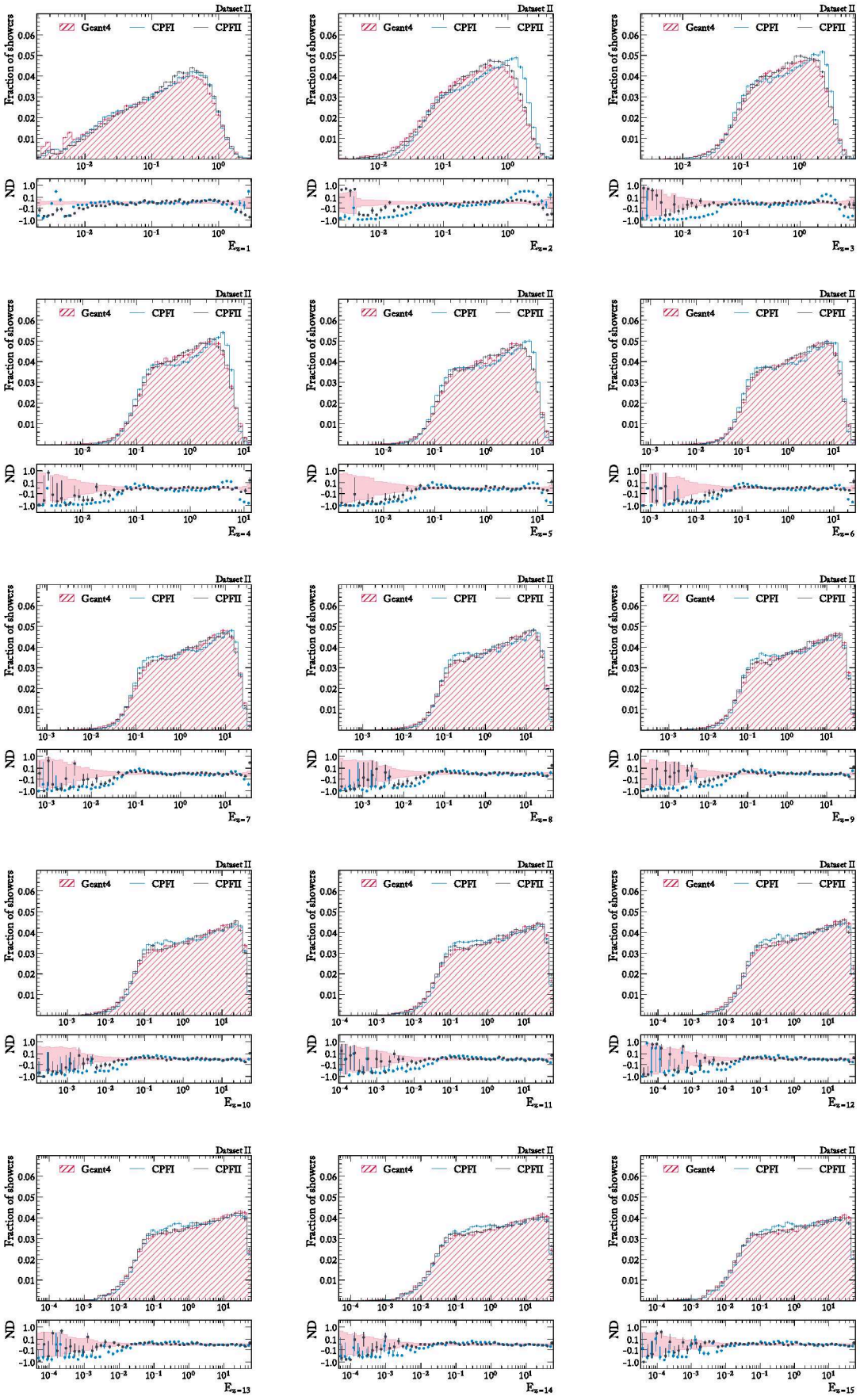

Figure B.4: The marginal energy distributions for the layers in z from 1 to 15 for Dataset II. See the accompanying text for an explanation of how to read the figures.

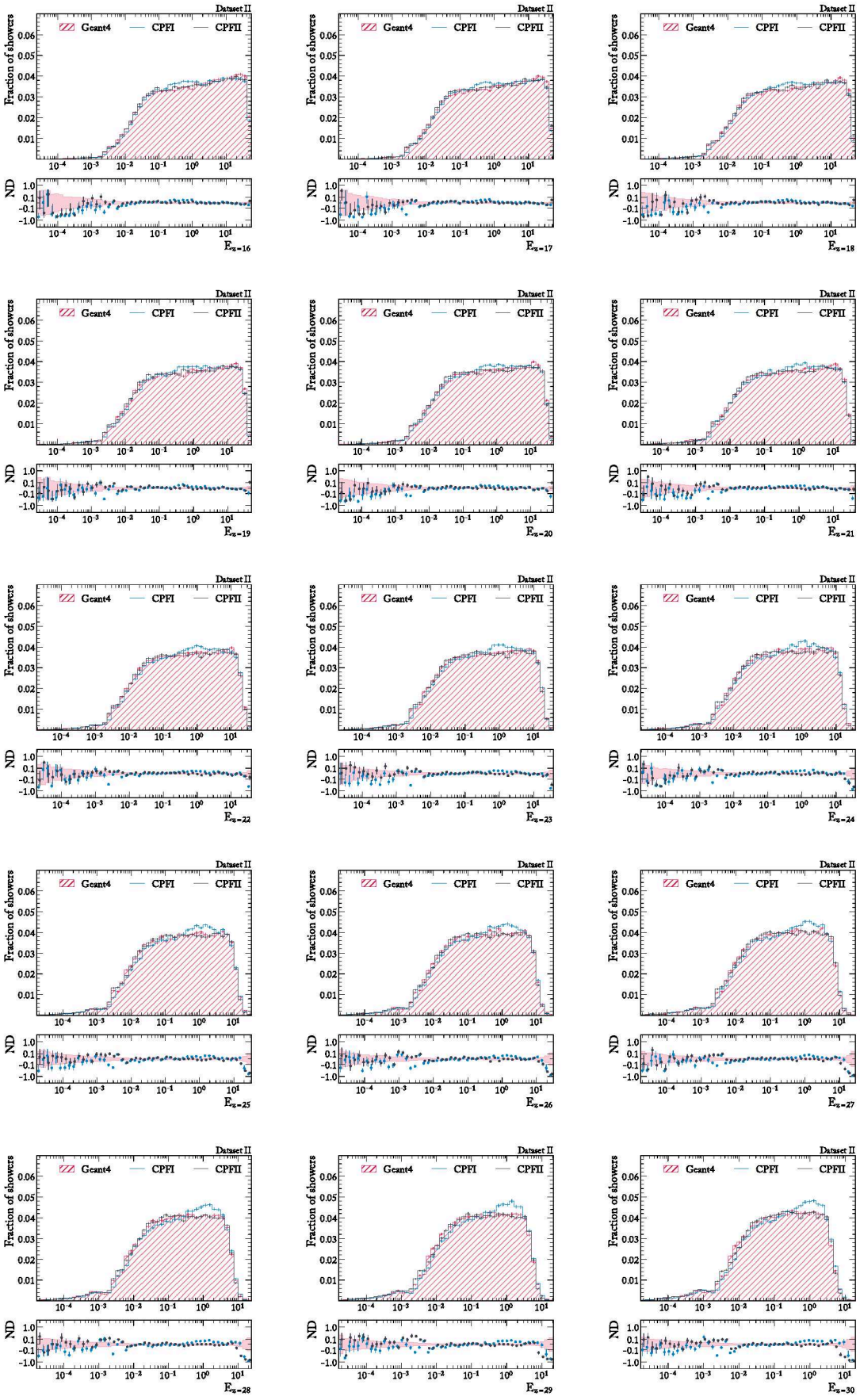

Figure B.5: The marginal energy distributions for the layers in z from 16 to 30 for Dataset II. See the accompanying text for an explanation of how to read the figures.

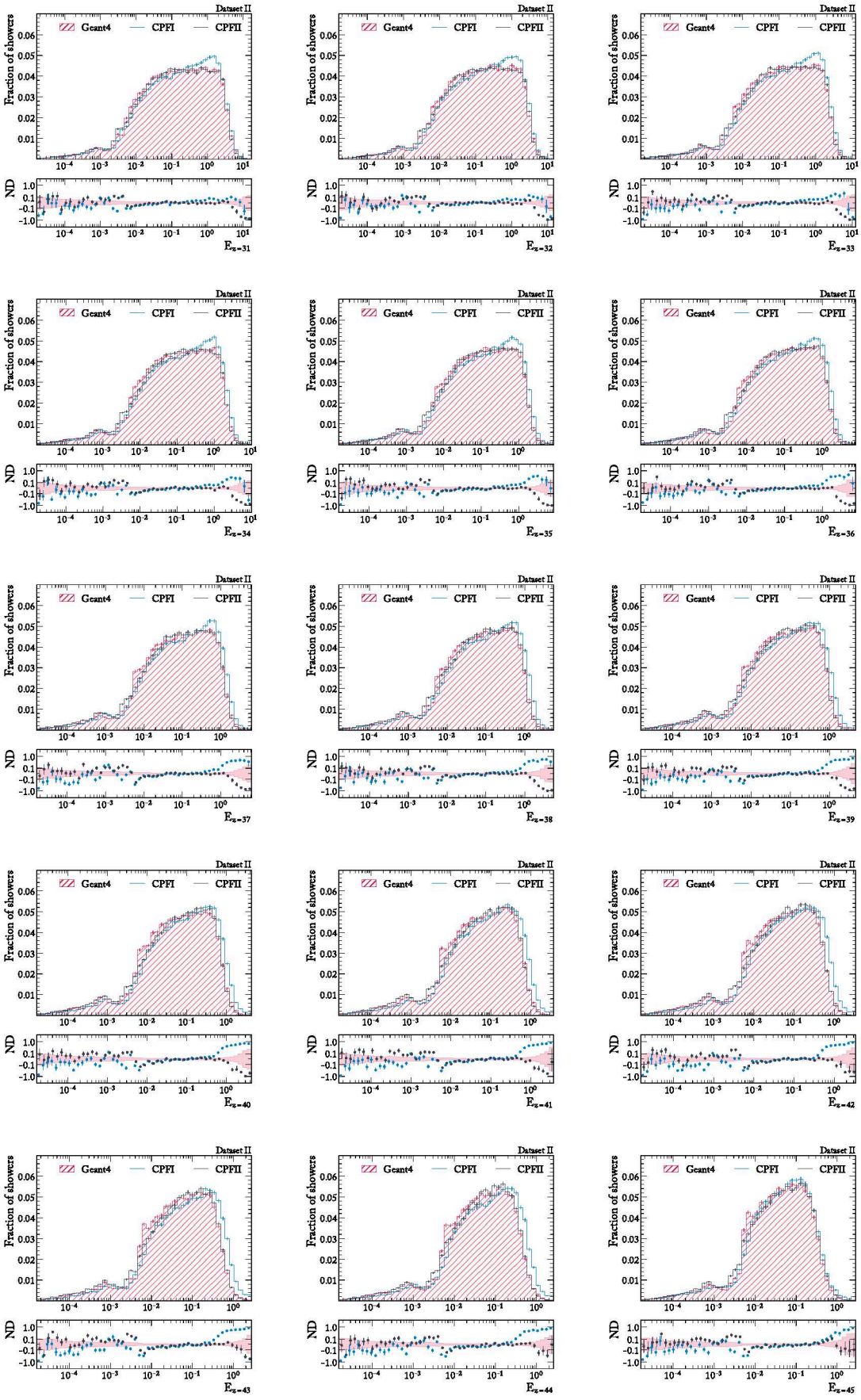

Figure B.6: The marginal energy distributions for the layers in z from 31 to 45 for Dataset II. See the accompanying text for an explanation of how to read the figures.

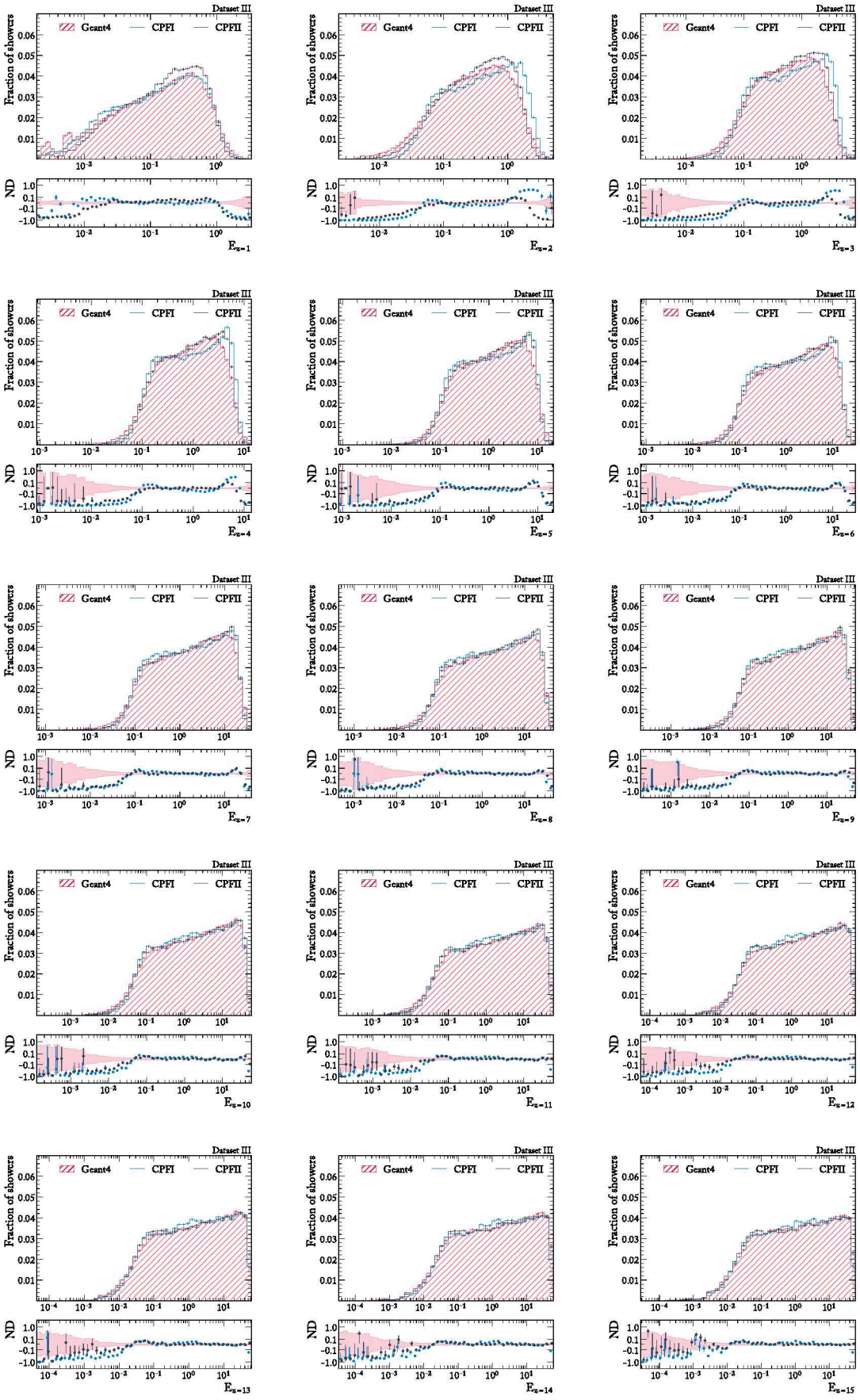

Figure B.7: The marginal energy distributions for the layers in z from 1 to 15 for Dataset III. See the accompanying text for an explanation of how to read the figures.

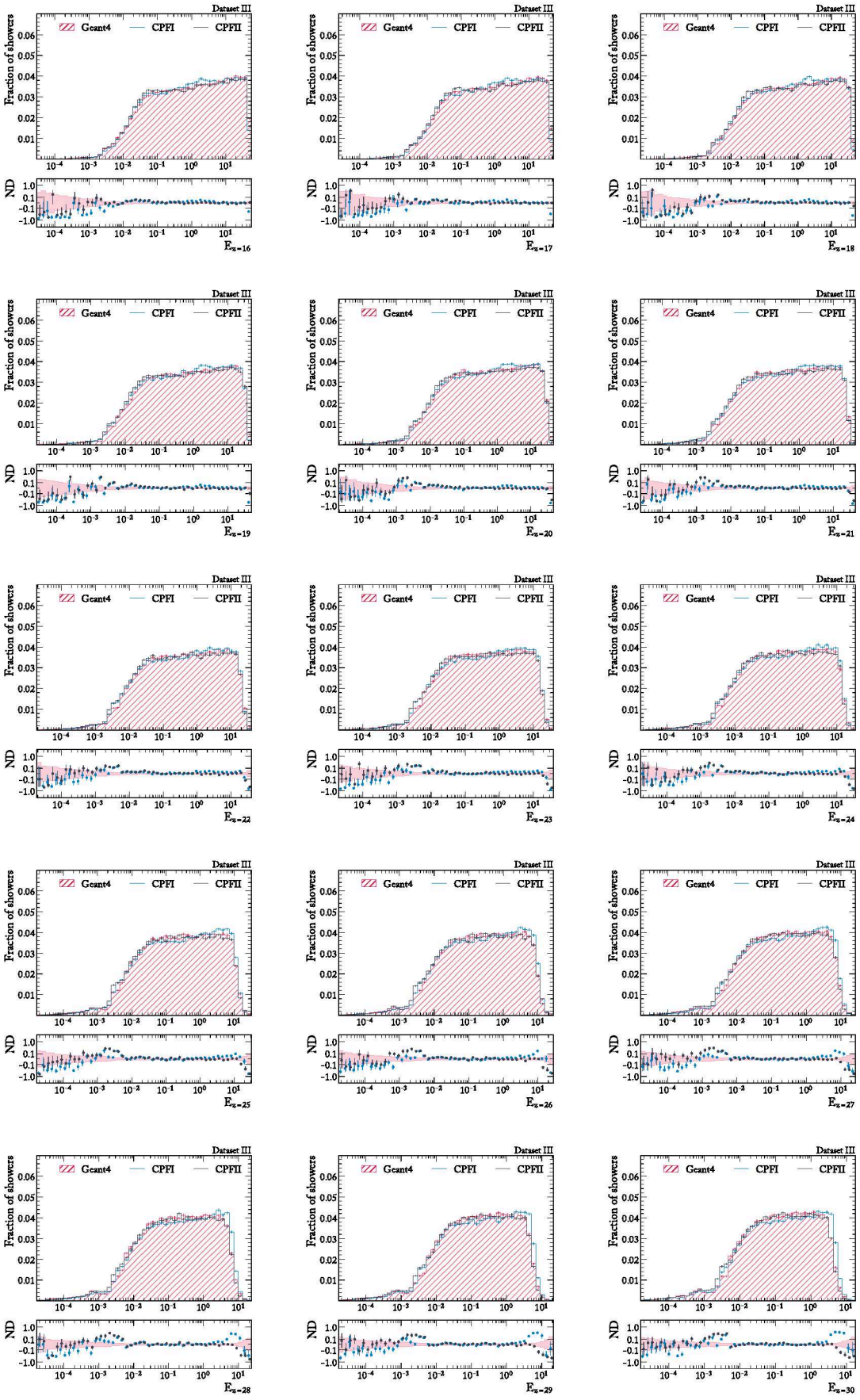

Figure B.8: The marginal energy distributions for the layers in z from 16 to 30 for Dataset III. See the accompanying text for an explanation of how to read the figures.

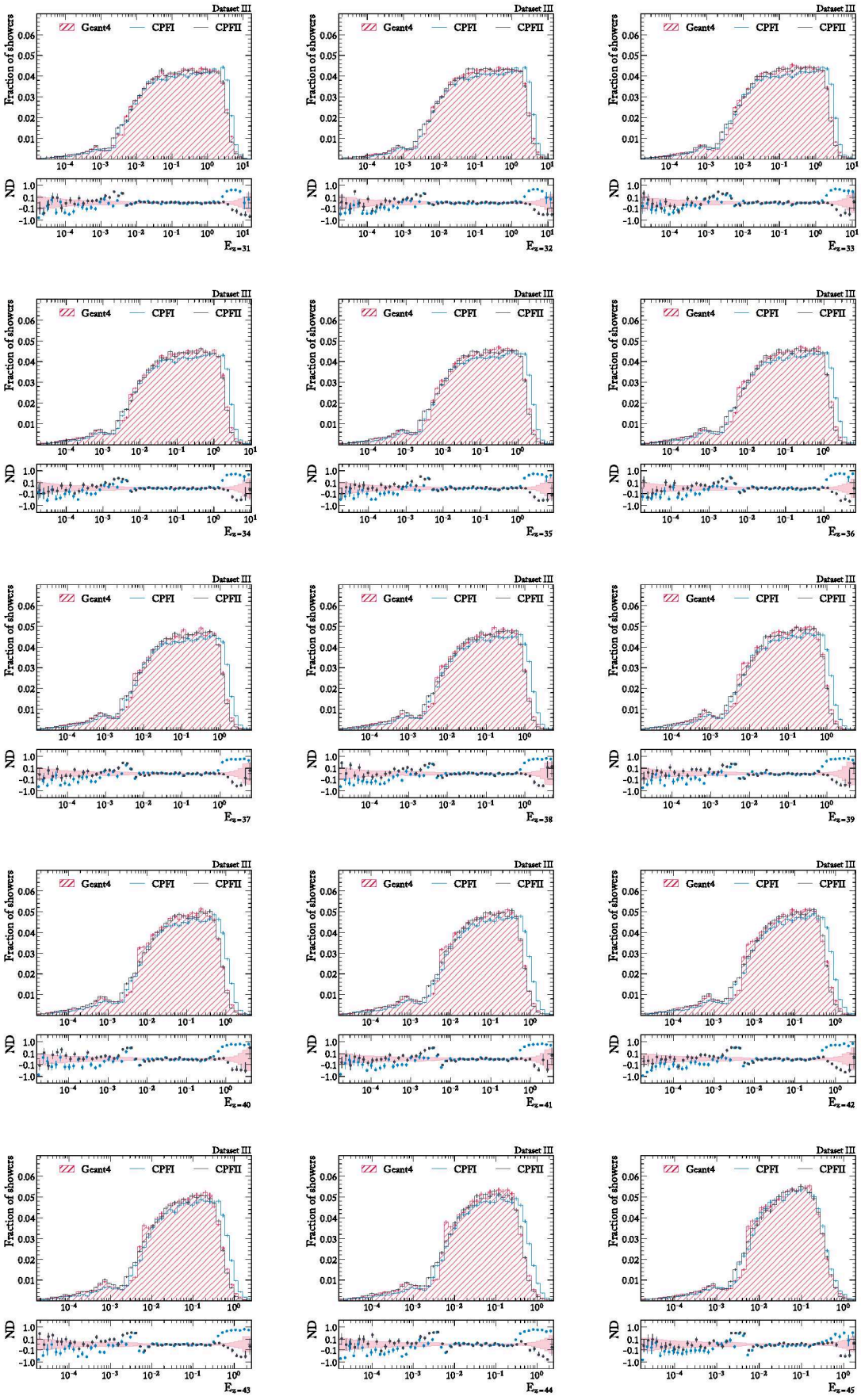

Figure B.9: The marginal energy distributions for the layers in z from 31 to 45 for Dataset III. See the accompanying text for an explanation of how to read the figures.

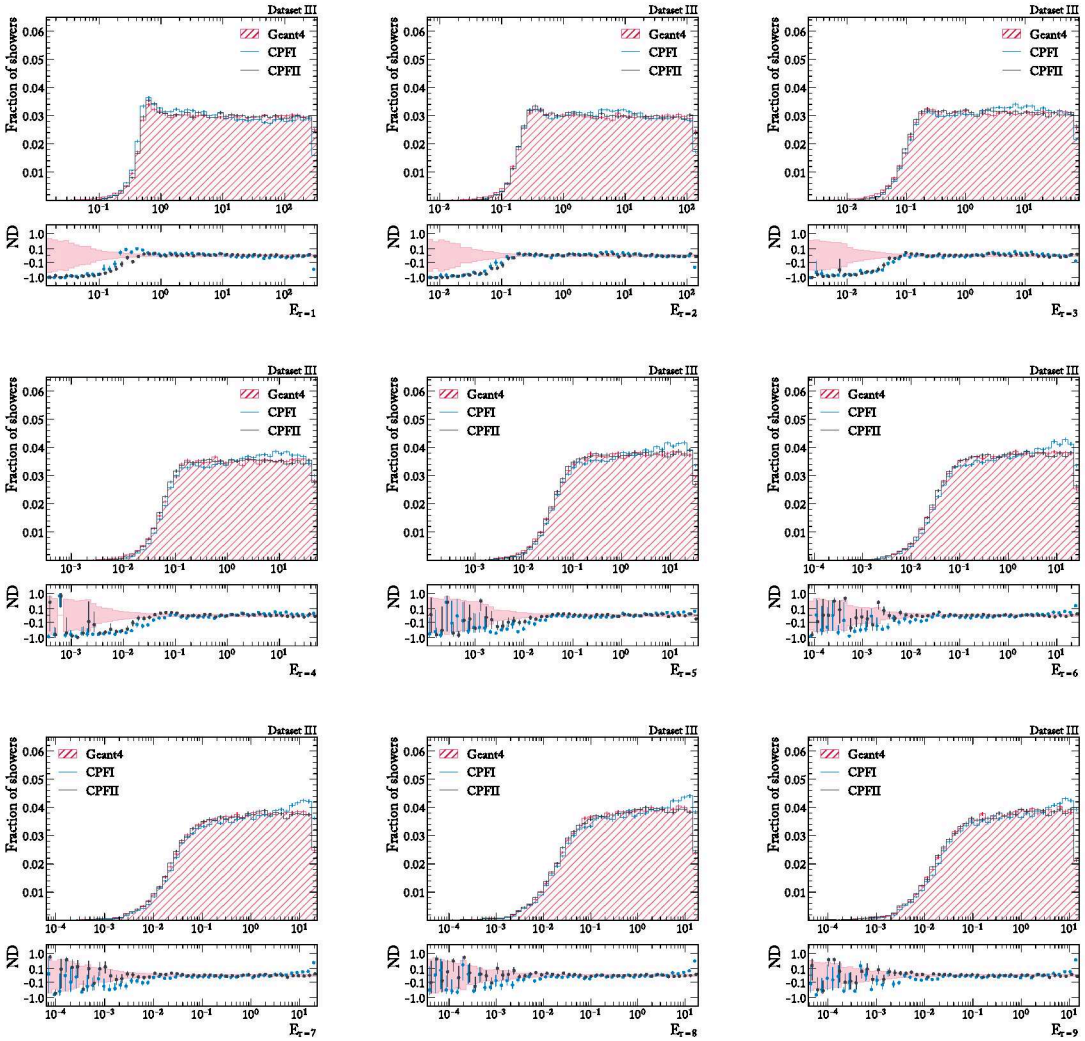

Figure B.10: The marginal energy distributions for the layers in r from 1 to 9 for Dataset III. See the accompanying text for an explanation of how to read the figures.

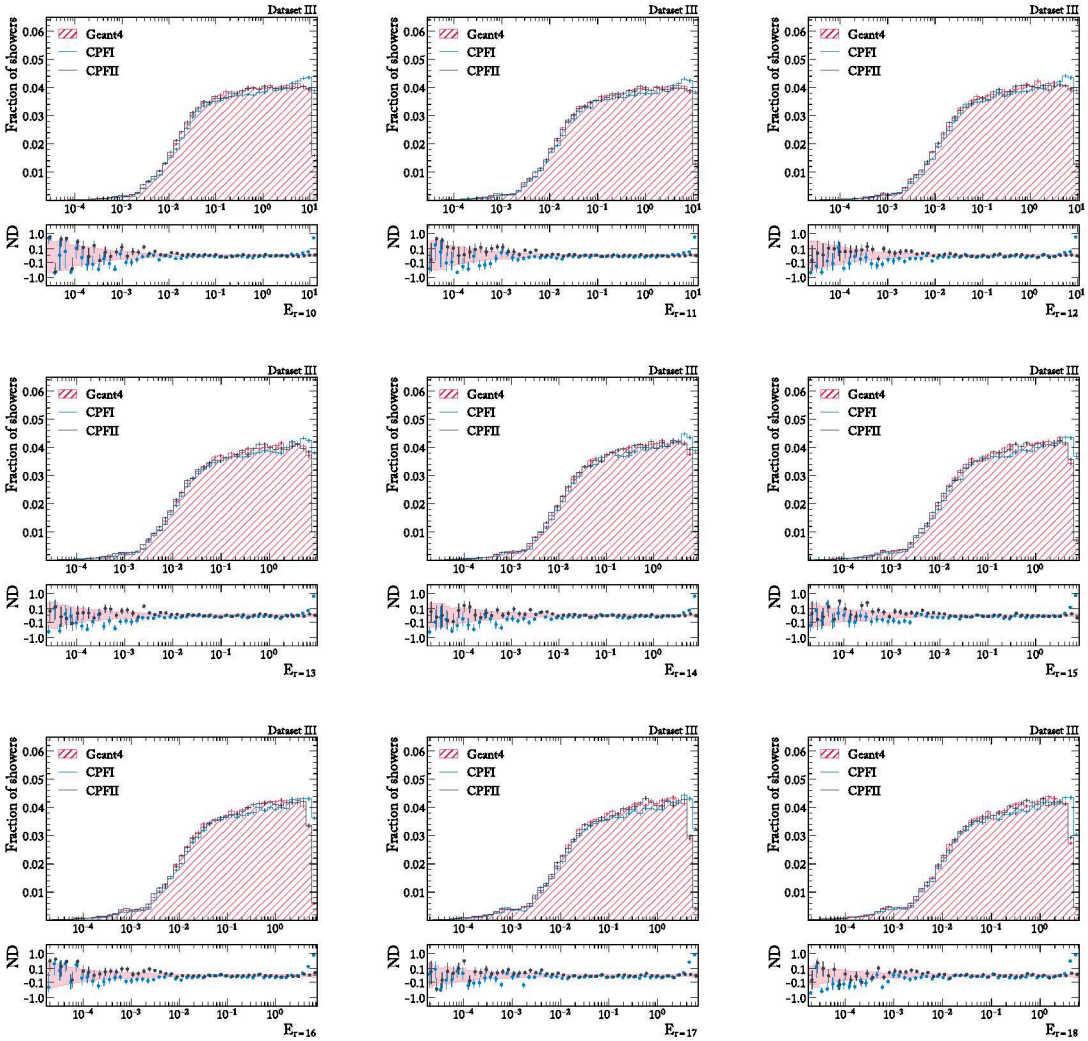

Figure B.11: The marginal energy distributions for the layers in r from 10 to 18 for Dataset III. See the accompanying text for an explanation of how to read the figures.

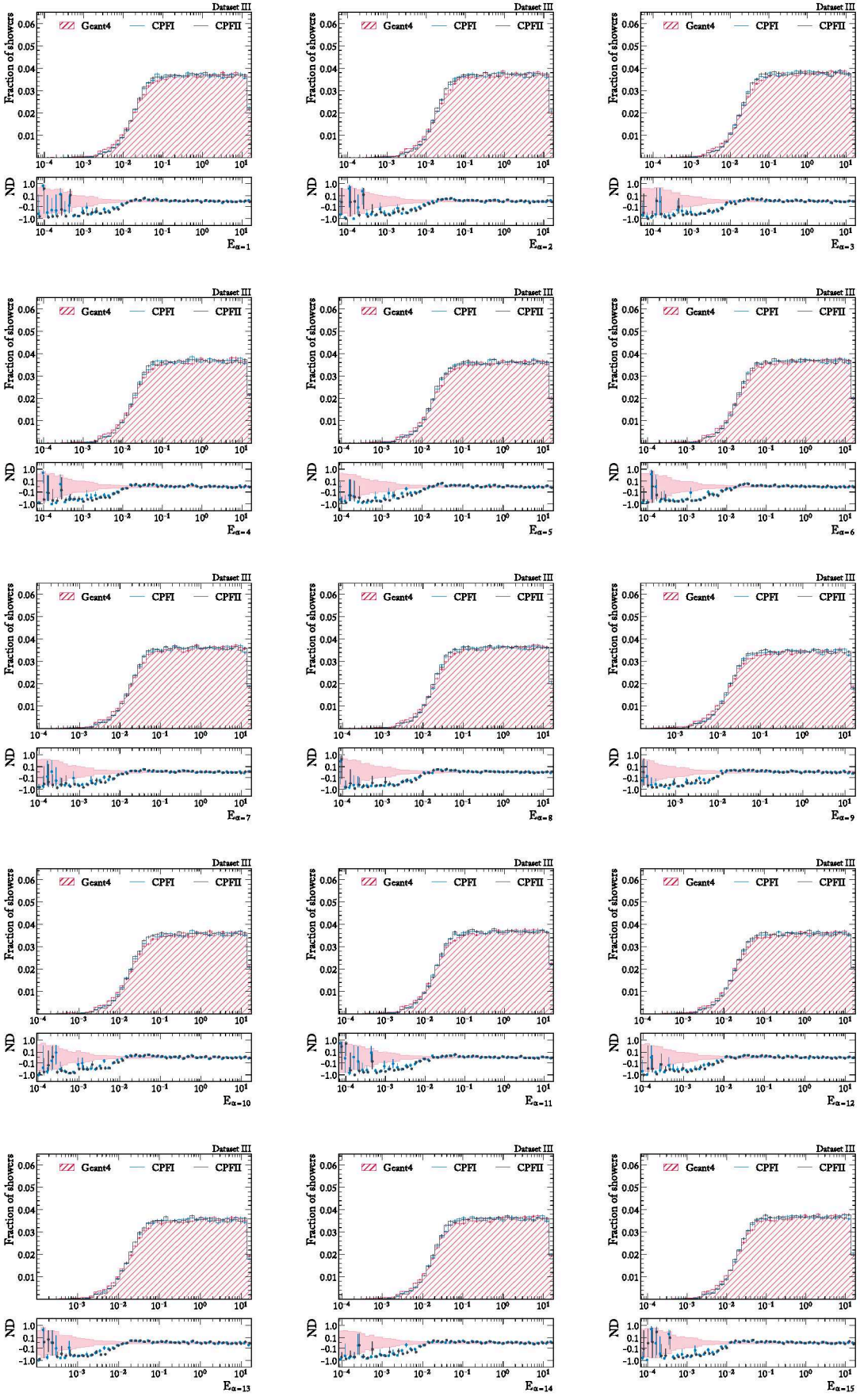

Figure B.12: The marginal energy distributions for the layers in α from 1 to 15 for Dataset III. See the accompanying text for an explanation of how to read the figures.

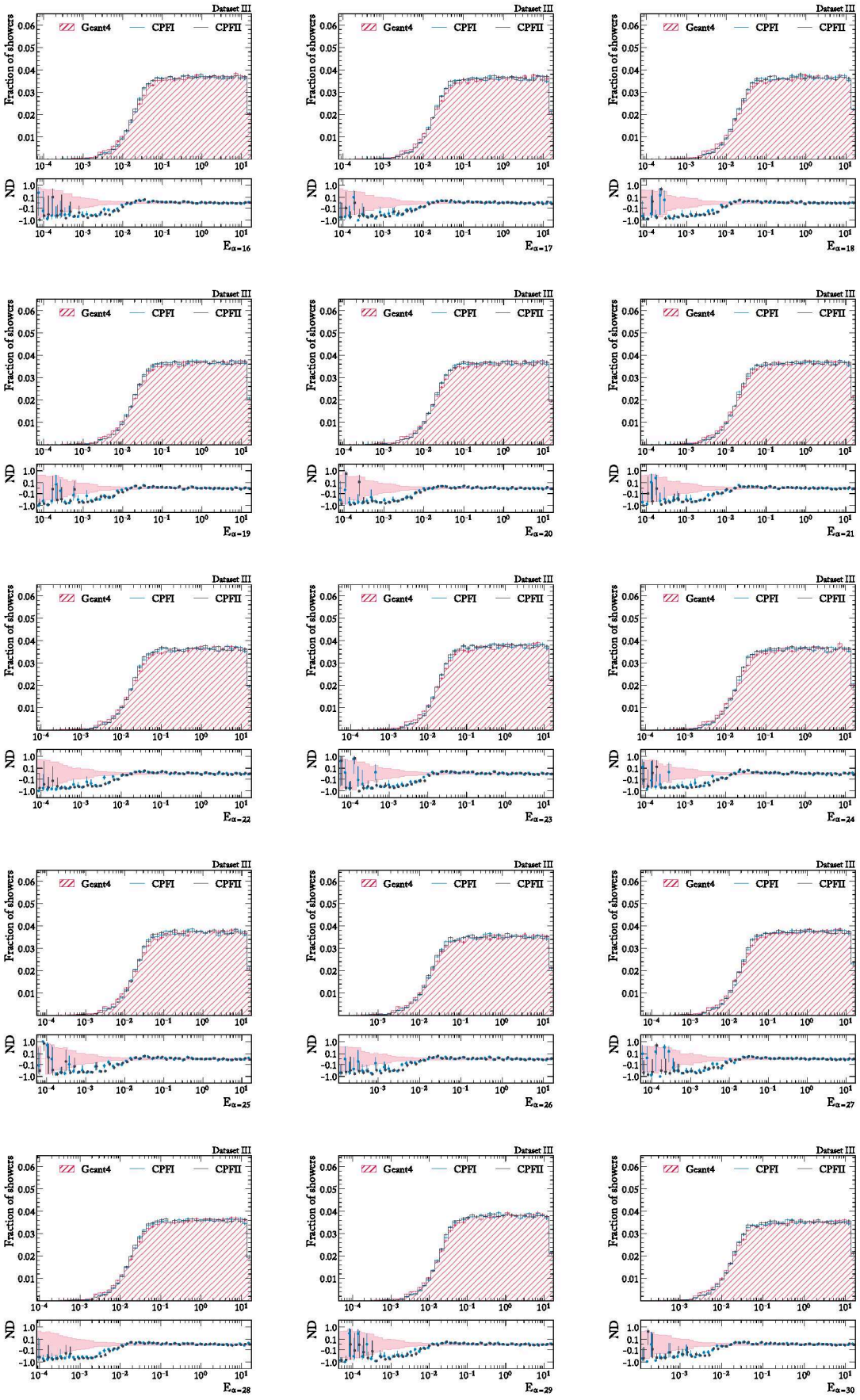

Figure B.13: The marginal energy distributions for the layers in α from 16 to 30 for Dataset III. See the accompanying text for an explanation of how to read the figures.

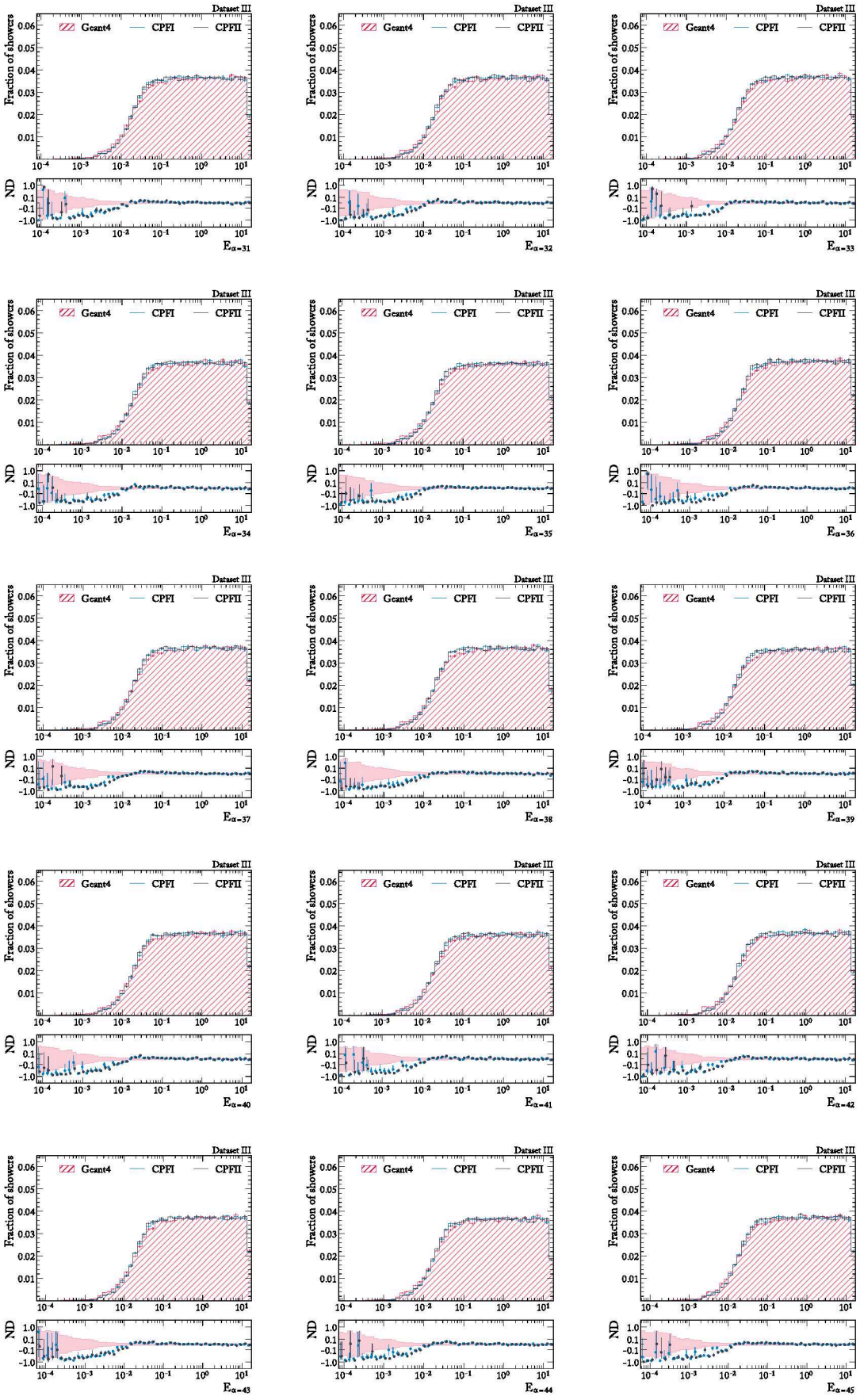

Figure B.14: The marginal energy distributions for the layers in α from 31 to 45 for Dataset III. See the accompanying text for an explanation of how to read the figures.

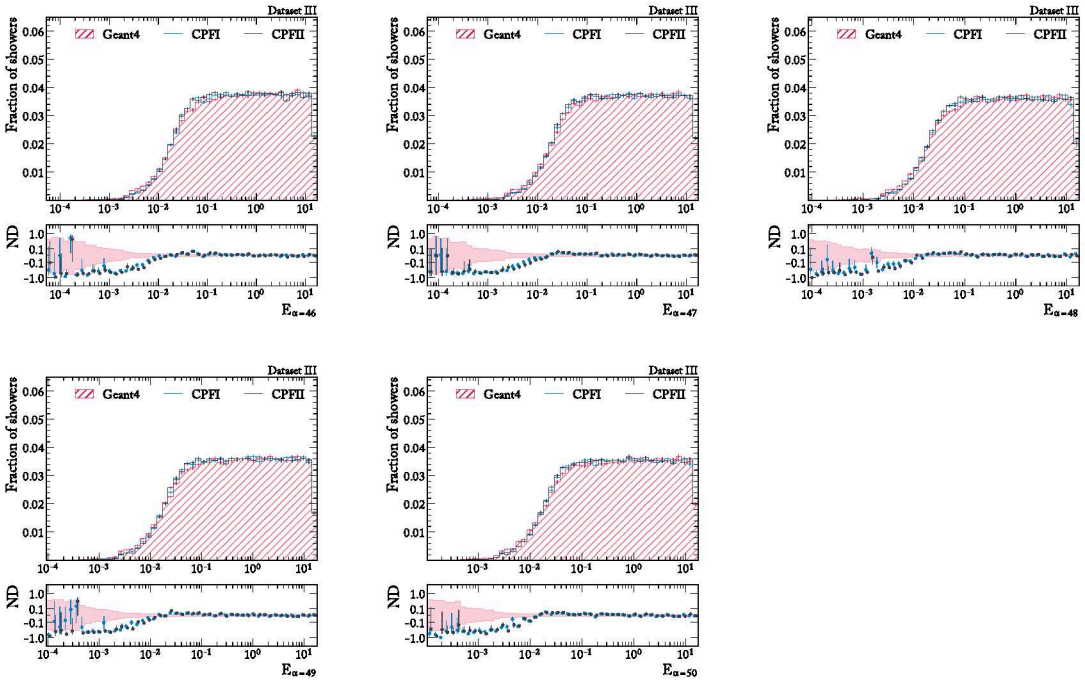

Figure B.15: The marginal energy distributions for the layers in α from 31 to 45 for Dataset III. See the accompanying text for an explanation of how to read the figures.

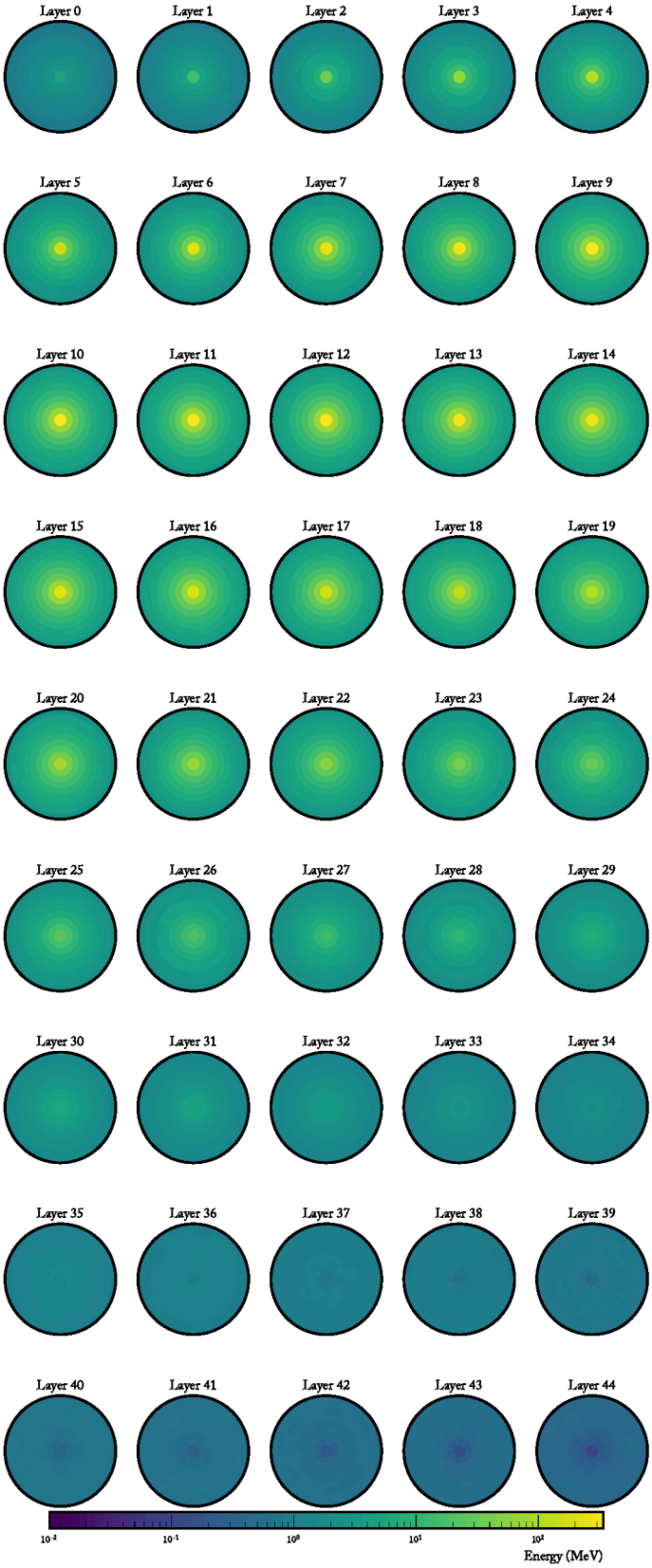

Figure B.16: The average energy deposition of showers in all layers of z for Dataset II is illustrated. The figure has been reproduced with the code provided by the CaloChallenge (Giannelli et al. 2022a).

GLOSSARY

X^2 A measure of homogeneity of two distributions, as defined in Equation (8.18). 86–88, 92, 93, 95–101, 103, 117–120, 124–127

Adam An optimization algorithm that combines the advantages of both AdaGrad and RMSProp. 43

ALICE A Large Ion Collider Experiment 16, 17

Area Under the Curve A metric that quantifies the performance of a binary classifier. 105–107, 128

ATLAS A Toroidal LHC ApparatuS 16, 17

Attention A mechanism that allows neural networks to focus on specific parts of the input data. 48

Bremsstrahlung Electromagnetic radiation emitted by a charged particle when it is decelerated. 23–25, 27, 28

CaloChallenge The challenge that was organized to evaluate the performance of generative models on calorimeter data. 67–69, 71, 83, 105–108, 128, 149

CaloPointFlow The point based normalizing flow model developed in this thesis. 67, 83, 86–89, 91, 100, 103, 104, 107

CaloPointFlow I The first version of the point based normalizing flow model developed in this thesis. 67, 72, 73, 79, 85–106, 127, 133

CaloPointFlow II The second version of the point based normalizing flow model developed in this thesis. 67, 74, 75, 79, 82, 85–106, 127, 133

CERN European Organization for Nuclear Research 16

CMS Compact Muon Solenoid 16–22, 62–65, 151

Convolutional Neural Network A type of neural network that is well-suited for image processing. 44, 45

Coupling Flow A type of normalizing flow that uses a coupling layer to transform the data. 56, 57, 73–75

Cumulative Distribution Function A function that gives the probability that a random variable is less than or equal to a specific value. 51, 79, 80, 111

Deep Sets A neural network architecture that is permutation invariant. 47, 50, 60, 73–75

DeepSetFlow A normalizing flow that uses a deep set architecture to model the transformation. 73, 75, 83, 104, 106

Electromagnetic Calorimeter A detector that measures the energy of electrons and photons. 17, 19, 20, 30, 64

Equivariance is a property of a function whereby the application of a transformation to the function's input commute. 45, 46

Evidence Lower Bound A lower bound on the log-likelihood of the data. 58–60

Fast Simulation CMS's fast simulation software that provides a quick approximation of the detector response. 62, 63, 65

Full Simulation CMS's full simulation software that provides a detailed simulation of the detector response. 62, 64

Geant4 A software toolkit for the simulation of the passage of particles through matter. 32–36, 62–65, 67, 83, 85–92, 94–96, 99, 103–105, 107, 108, 117–121, 123, 124, 126–128

GFlash A fast simulation model that provides a quick approximation of the electromagnetic and hadronic showers. 62–65

Hadron-Electron Ring Accelerator A particle accelerator that operated at DESY from 1992 to 2007. 14, 15

Hadronic Calorimeter A detector that measures the energy of hadrons. 18, 20, 65

High Energy Physics The branch of physics that studies the fundamental particles and forces of nature. 13

Independent and Identically Distributed A sequence of random variables that are independent and have the same probability distribution. 60, 61, 72, 73, 111, 112

Invariance is a property of a function that remains unchanged when the inputs are subjected to a transformations. 45

Kullback-Leibler Divergence A measure of how one probability distribution diverges from a second, expected probability distribution. 38–40, 59

Large Electron-Positron Collider A particle accelerator that operated at CERN from 1989 to 2000. 14, 16

Large Hadron Collider The world's largest and most powerful particle accelerator. 4, 13, 14, 16, 17, 19

LHCb Large Hadron Collider beauty 16, 17

Long Short-Term Memory A type of recurrent neural network that is capable of learning long-term dependencies. 56

Machine Learning A field of computer science that uses statistical techniques to give computers the ability to learn from data. 37, 40, 45

Masked Autoregressive Flow A type of normalizing flow that uses an autoregressive neural network to model the transformation. 56

Mean Squared Error A measure of the average squared difference between the estimated values and the actual values. 39, 40

Multi-Layer Perceptron A type of feedforward neural network. 70, 72–75, 114–117

Natural Language Processing A field of artificial intelligence that focuses on the interaction between computers and humans using natural language. 48

Neural Network A computational model inspired by the human brain. 37, 41, 42, 44–48

Normalized Difference The difference between two values divided by their sum, as defined in Equation (8.16). 84–91, 94, 95, 97–102, 104, 117–127, 133

Normalizing Flow A generative model that learns a sequence of invertible transformations to map a simple distribution to a complex one. 51, 52, 57, 59, 60, 72, 73, 75

Probability Density Function A function that describes the likelihood of a continuous random variable taking on a specific value. 110, 111, 113

Probability Mass Function A function that gives the probability that a discrete random variable is equal to a specific value. 78, 80, 81

Quantum Chromodynamics The quantum field theory that describes the strong nuclear force. 10, 11

Quantum Electrodynamics The quantum field theory that describes the electromagnetic force. 4, 6–8, 13

Quantum Field Theory A theoretical framework that combines quantum mechanics with special relativity. 4, 6, 13

Rational Quadratic Spline A type of spline that uses rational quadratic functions to interpolate the data. 53, 72, 73, 75, 114, 116

Receiver Operating Characteristic A graphical plot that illustrates the diagnostic ability of a binary classifier. 105

RecHit A reconstructed hit positioned in an active detector element. 65

SimHit A simulated hit positioned at a track in the Geant4 simulation. 62, 65

Standard Model The theory that describes the electromagnetic, weak, and strong nuclear interactions 4–6, 11, 12

Stochastic Gradient Descent An optimization algorithm that updates the model's parameters iteratively. 43

Symmetry A system's invariance under a group of transformations. 45

Transformer A deep learning model that uses self-attention mechanisms to process sequential data. 48

Variational Autoencoder A generative model that learns a latent representation of the data. 58–60

BIBLIOGRAPHY

Abramowicz, H., et al. (2015). Combination of measurements of inclusive deep inelastic $e^{\pm}p$ scattering cross sections and QCD analysis of HERA data. *Eur. Phys. J. C*, *75*(12), 580. https://doi.org/10.1140/epjc/s10052-015-3710-4

Acosta, F. T., Mikuni, V., Nachman, B., et al. (2023). Comparison of Point Cloud and Image-based Models for Calorimeter Fast Simulation. *arXiv preprint arXiv:2307.04780*.

Albrecht, J., et al. (2019). A Roadmap for HEP Software and Computing R&D for the 2020s. *Comput. Softw. Big Sci.*, *3*(1), 7. https://doi.org/10.1007/s41781-018-0018-8

ALICE Collaboration. (2008). The ALICE experiment at the CERN LHC. *JINST*, *3*, S08002. https://doi.org/10.1088/1748-0221/3/08/S08002

Allison, J., et al. (2003). Geant4 - a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, *506*(3), 250–303. https://doi.org/10.1016/s0168-9002(03)01368-8

Allison, J., et al. (2006). Geant4 developments and applications. *IEEE Trans. Nucl. Sci.*, *53*, 270. https://doi.org/10.1109/TNS.2006.869826

Allison, J., et al. (2016). Recent developments in Geant4. *Nucl. Instrum. Meth. A*, *835*, 186–225. https://doi.org/10.1016/j.nima.2016.06.125

Amaldi, U. (1981). Fluctuations in calorimetry measurements. *Physica Scripta*, *23*(4A), 409–424. https://doi.org/10.1088/0031-8949/23/4a/012

Amari, S. (1967). A theory of adaptive pattern classifiers. *IEEE Trans. EC*, *16*(3), 299–307.

Amram, O., & Pedro, K. (2023). CaloDiffusion with GLaM for High Fidelity Calorimeter Simulation. *Phys.Rev.D*, *108*, 072014. https://doi.org/10.1103/PhysRevD.108.072014

ATLAS Collaboration. (2008). The ATLAS Experiment at the CERN Large Hadron Collider. *JINST*, *3*, S08003. https://doi.org/10.1088/1748-0221/3/08/S08003

ATLAS Collaboration. (2018). Deep generative models for fast shower simulation in ATLAS. *ATL-SOFT-PUB-2018-001*. http://cds.cern.ch/record/2630433

ATLAS Collaboration. (2022). AtlFast3: the next generation of fast simulation in ATLAS. *Comput. Softw. Big Sci.*, *6*, 7. https://doi.org/10.1007/s41781-021-00079-7

ATLAS Collaboration. (2024). Deep Generative Models for Fast Photon Shower Simulation in ATLAS. *Comput. Softw. Big Sci.*, *8*(1), 7. https://doi.org/10.1007/s41781-023-00106-9

Baldi, P., & Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, *2*(1), 53–58. https://doi.org/https://doi.org/10.1016/0893-6080(89)90014-2

Ball, R. D., et al. (2021). The Path to Proton Structure at One-Percent Accuracy. *Eur.Phys.J.C*, *82*(5), 428. https://doi.org/10.1140/epjc/s10052-022-10328-7

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, *112*(518), 859–877.

Boehnlein, A., Simon, F., Hernandez, F., et al. (2022). *Hl-lhc software and computing review panel, 2nd report* (tech. rep.). CERN.

Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern.*, *59*(4–5), 291–294. https://doi.org/10.1007/BF00332918

Bronstein, M. M., Bruna, J., Cohen, T., & Veli cković, P. (2021). Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges.

Bryson, A. E. (1961). A gradient method for optimizing multi-stage allocation processes.

Bryson, A., & Ho, Y. (1969). *Applied optimal control: Optimization, estimation, and control*. Blaisdell Pub. Co. http://books.google.ch/books?id=k%5C%5FFQAAAAMAAJ

Bryson, A. E., Jr., & Denham, W. F. (1961). A steepest-ascent method for solving optimum programming problems. (BR-1303).

Buckley, M. R., Krause, C., Pang, I., et al. (2023). Inductive CaloFlow. *arXiv preprint arXiv:2305.11934*.

Buhmann, E., Diefenbacher, S., Eren, E., et al. (2020). Getting High: High Fidelity Simulation of High Granularity Calorimeters with High Speed. *Comput.Softw.Big Sci.*, *5*, 13. https://doi.org/10.1007/s41781-021-00056-0

Buhmann, E., Diefenbacher, S., Eren, E., et al. (2023). CaloClouds: Fast Geometry-Independent Highly-Granular Calorimeter Simulation. *JINST*, *18*, P11025. https://doi.org/10.1088/1748-0221/18/11/P11025

Buhmann, E., Gaede, F., Kasieczka, G., et al. (2023). CaloClouds II: Ultra-Fast Geometry-Independent Highly-Granular Calorimeter Simulation. *arXiv preprint arXiv:2309.05704*.

Buhmann, E., Kasieczka, G., & Thaler, J. (2023). EPiC-GAN: Equivariant Point Cloud Generation for Particle Jets. *SciPost Phys.*, *15*, 130. https://doi.org/10.21468/SciPostPhys.15.4.130

Cao, T. Y. (1998). *Conceptual developments of 20th century field theories* (1st.paperback). Cambridge University Press.

Cauchy, A.-L. (1847). *Analyse mathématique. - méthodc générale pour la résolution des systèmes d'équations simultanées*. Cambridge University Press. https://doi.org/10.1017/cbo9780511702396.063

CERN. (1984). LEP Design Report: Vol.2. The LEP Main Ring.

CMS Collaboration. (2008). The CMS Experiment at the CERN LHC. *JINST*, *3*, S08004. https://doi.org/10.1088/1748-0221/3/08/S08004

CMS Collaboration. (2015). Measurement of the inclusive 3-jet production differential cross section in proton–proton collisions at 7 TeV and determination of the strong coupling constant in the TeV range. *Eur. Phys. J. C*, *75*(5), 186. https://doi.org/10.1140/epjc/s10052-015-3376-y

CMS Collaboration. (2017). *The Phase-2 Upgrade of the CMS Endcap Calorimeter* (tech. rep.). CERN. https://doi.org/10.17181/CERN.IV8M.1JY2

CMS Collaboration. (2019). Sketchupcms gallery [Accessed: 02-02-2024].

CMS Collaboration. (2020). A measurement of the Higgs boson mass in the diphoton decay channel. *Phys. Lett. B*, *805*, 135425. https://doi.org/10.1016/j.physletb.2020.135425

Cohen, T. S., & Welling, M. (2016). Group Equivariant Convolutional Networks.

Collaboration, T. A. (2012). Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc. *Physics Letters B*, *716*(1), 1–29. https://doi.org/10.1016/j.physletb.2012.08.020

Collaboration, T. C. (2012). Observation of a new boson at a mass of 125 gev with the cms experiment at the lhc. *Physics Letters B*, *716*(1), 30–61. https://doi.org/10.1016/j.physletb.2012.08.021

Collins, J. (2023, July). *Foundations of perturbative qcd*. Cambridge University Press. https://doi.org/10.1017/9781009401845

Cramér, H. (1946). *Mathematical Methods of Statistics*. Princeton University Press.

Cresswell, J. C., Ross, B. L., Loaiza-Ganem, G., et al. (2022). CaloMan: Fast generation of calorimeter showers with density estimation on learned manifolds. *36th Conference on Neural Information Processing Systems*.

Das, R., Favaro, L., Heimel, T., et al. (2023). How to Understand Limitations of Generative Networks.

de Broglie, L. (1923). Waves and quanta. *Nature*, *112*(2815), 540–540. https://doi.org/10.1038/112540a0

Diefenbacher, S., Eren, E., Kasieczka, G., Korol, A., Nachman, B., & Shih, D. (2020). DCTRGAN: Improving the Precision of Generative Models with Reweighting. *JINST*, *15*(11), P11004. https://doi.org/10.1088/1748-0221/15/11/P11004

Dinh, L., Krueger, D., & Bengio, Y. (2014). NICE: Non-linear Independent Components Estimation.

Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2017). Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*.

Dreyfus, S. E. (1962). The numerical solution of variational problems. *Journal of Mathematical Analysis and Applications*, 30–45.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*, 2121–2159.

Durkan, C., Bekasov, A., Murray, I., et al. (2019). Neural Spline Flows. *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. *Annals Statist.*, *7*(1), 1–26. https://doi.org/10.1214/aos/1176344552

Einstein, A. (1905). Ist die trägheit eines körpers von seinem energieinhalt abhängig? *Annalen der Physik*, *323*(13), 639–641. https://doi.org/10.1002/andp.19053231314

Englert, F., & Brout, R. (1964). Broken Symmetry and the Mass of Gauge Vector Mesons (J. C. Taylor, Ed.). *Phys. Rev. Lett.*, *13*, 321–323. https://doi.org/10.1103/PhysRevLett.13.321

Ernst, F., Favaro, L., Krause, C., et al. (2023). Normalizing Flows for High-Dimensional Detector Simulations. *arXiv preprint arXiv:2312.09290*.

Ezhela, V. V., Lugovsky, S. B., & Zenin, O. V. (2003). Hadronic part of the muon g-2 estimated on the sigma**2003(tot)(e+ e- —> hadrons) evaluated data compilation.

Fabjan, C. W., & Gianotti, F. (2003). Calorimetry for particle physics. *Reviews of Modern Physics*, *75*(4), 1243–1286. https://doi.org/10.1103/revmodphys.75.1243

Feynman, R. P. (1950). Mathematical formulation of the quantum theory of electromagnetic interaction (L. M. Brown, Ed.). *Phys. Rev.*, *80*, 440–457. https://doi.org/10.1103/PhysRev.80.440

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*(4), 193–202. https://doi.org/10.1007/bf00344251

Gagunashvili, N. D. (2007). Comparison of weighted and unweighted histograms. *PoS*, 054.

Galbraith, D. (2013, December). Ux: Standard model of the standard model. https://davidgalbraith.org/portfolio/ux-standard-model-of-the-standard-model/

Gell-Mann, M. (1961). The Eightfold Way: A Theory of strong interaction symmetry. https://doi.org/10.2172/4008239

Gell-Mann, M. (1964). A Schematic Model of Baryons and Mesons. *Phys. Lett.*, *8*, 214–215. https://doi.org/10.1016/S0031-9163(64)92001-3

Germain, M., Gregor, K., Murray, I., & Larochelle, H. (2015). MADE: Masked Autoencoder for Distribution Estimation.

Giammanco, A. (2014). The Fast Simulation of the CMS Experiment (D. L. Groep & D. Bonacorsi, Eds.). *J. Phys. Conf. Ser.*, *513*, 022012. https://doi.org/10.1088/1742-6596/513/2/022012

Giannelli, M. F., Kasieczka, G., Krause, C., et al. (2022a). Fast Calorimeter Simulation Challenge [https://calochallenge.github.io/homepage/]. https://calochallenge.github.io/homepage/

Giannelli, M. F., Kasieczka, G., Krause, C., et al. (2022b, March). Fast Calorimeter Simulation Challenge 2022 - Dataset 2. https://doi.org/10.5281/zenodo.6366271

Giannelli, M. F., Kasieczka, G., Krause, C., et al. (2022c, March). Fast Calorimeter Simulation Challenge 2022 - Dataset 3. https://doi.org/10.5281/zenodo.6366324

Glashow, S. L. (1961). Partial Symmetries of Weak Interactions. *Nucl. Phys.*, *22*, 579–588. https://doi.org/10.1016/0029-5582(61)90469-2

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., et al. (2014). Generative Adversarial Networks.

Grindhammer, G., & Peters, S. (1993). The Parameterized simulation of electromagnetic showers in homogeneous and sampling calorimeters. *International Conference on Monte Carlo Simulation in High-Energy and Nuclear Physics - MC 93*.

Gumbel, E. J. (1935). Les valeurs extrêmes des distributions statistiques. *Annales de l'institut Henri Poincaré*, *5*(2), 115–158.

Gumbel, E. J. (1941). The return period of flood flows. *The annals of mathematical statistics*, *12*(2), 163–190.

Guralnik, G. S., Hagen, C. R., & Kibble, T. W. B. (1964). Global Conservation Laws and Massless Particles (J. C. Taylor, Ed.). *Phys. Rev. Lett.*, *13*, 585–587. https://doi.org/10.1103/PhysRevLett.13.585

Hagiwara, K., et al. (2002). Review of Particle Physics. *Physical Review D*, *66*, 010001+. http://pdg.lbl.gov

Hampshire & Waibel. (1989). A novel objective function for improved phoneme recognition using time delay neural networks. *International Joint Conference on Neural Networks*. https://doi.org/10.1109/ijcnn.1989.118586

Hasert, F. J., et al. (1973). Search for Elastic ν_μ Electron Scattering. *Phys. Lett. B*, *46*, 121–124. https://doi.org/10.1016/0370-2693(73)90494-2

Hashemi, H., & Krause, C. (2023). Deep Generative Models for Detector Signature Simulation: An Analytical Taxonomy. *arXiv preprint arXiv:2312.09597*.

Higgs, P. W. (1964). Broken Symmetries and the Masses of Gauge Bosons (J. C. Taylor, Ed.). *Phys. Rev. Lett.*, *13*, 508–509. https://doi.org/10.1103/PhysRevLett.13.508

Hinton, G., Srivastava, N., & Swersky, K. (2014). Neural networks for machine learning: Lecture 6a - overview of mini-batch gradient descent.

Ho, J., Chen, X., Srinivas, A., et al. (2019). Flow++: Improving flow-based generative models with variational dequantization and architecture design. *arXiv preprint arXiv:1902.00275*.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Comput.*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8

Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, *148*(3), 574–591. https://doi.org/10.1113/jphysiol.1959.sp006308

Huijben, I. A. M., Kool, W., Paulus, M. B., et al. (2023). A review of the gumbel-max trick and its extensions for discrete stochasticity in machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(2), 1353–1371. https://doi.org/10.1109/TPAMI.2022.3157042

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., & Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, *37*, 183–233.

Käch, B., & Melzer-Pellmann, I. (2023). Attention to Mean-Fields for Particle Cloud Generation. *arXiv preprint arXiv:2305.15254*.

Käch, B., Melzer-Pellmann, I., & Krücker, D. (2023). Pay attention to mean-fields for point cloud generation. https://ml4physicalsciences.github.io/2023/files/NeurIPS%5C%5FML4PS%5C%5F2023%5C%5F10.pdf

Kelley, H. J. (1960). Gradient theory of optimal flight paths. *ARS Journal*, *30*(10), 947–954.

Kim, S., Lee, S.-g., Song, J., Kim, J., & Yoon, S. (2018). Flowavenet: A generative flow for raw audio. *arXiv preprint arXiv:1811.02155*.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization.

Kingma, D. P., & Dhariwal, P. (2018). Glow: Generative Flow with Invertible 1x1 Convolutions.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., & Welling, M. (2016). Improving Variational Inference with Inverse Autoregressive Flow.

Kingma, D. P., & Welling, M. (2013a). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kingma, D. P., & Welling, M. (2013b). Auto-Encoding Variational Bayes.

Klokov, R., Boyer, E., & Verbeek, J. J. (2020). Discrete point flow networks for efficient point cloud generation. *European Conference on Computer Vision*.

Kobyzev, I., Prince, S. J. D., & Brubaker, M. A. (2021). Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Trans. Pattern Anal. Machine Intell.*, *43*(11), 3964–3979. https://doi.org/10.1109/tpami.2020.2992934

Kolanoski, H., & Wermes, N. (2016). *15 kalorimeter*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-45350-6_15

Kool, W., van Hoof, H., & Welling, M. (2019). Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:76662039

Krause, C., Pang, I., & Shih, D. (2022). CaloFlow for CaloChallenge Dataset 1. *arXiv preprint arXiv:2210.14245*.

Krause, C., & Shih, D. (2021). CaloFlow: Fast and Accurate Generation of Calorimeter Showers with Normalizing Flows. *Phys.Rev.D*, *107*, 113003. https://doi.org/10.1103/PhysRevD.107.113003

Langacker, P. (1986). The standard model and beyond (D. F. Geesaman, Ed.). *AIP Conf. Proc.*, *150*, 142–164. https://doi.org/10.1063/1.36182

LeCun, Y., Boser, B., Denker, J. S., et al. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551. https://doi.org/10.1162/neco.1989.1.4.541

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*, 436–444. https://doi.org/10.1038/nature14539

Lee, J., Lee, Y., Kim, J., Kosiorek, A. R., Choi, S., & Teh, Y. W. (2019). Set transformer: A framework for attention-based permutation-invariant neural networks. *Proceedings of the 36th International Conference on Machine Learning*, PMLR 97.

Lee, S., Hauptmann, J., & Wigmans, R. (2018). Where the energy goes [Accessed: 02-02-2024].

LHCb Collaboration. (2008). The LHCb Detector at the LHC. *JINST*, *3*, S08005. https://doi.org/10.1088/1748-0221/3/08/S08005

Linnainmaa, S. (1970). *The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors* [Master's thesis, Univ. Helsinki].

Linnainmaa, S. (1976). Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics*, *16*(2), 146–160.

Liu, J., Kumar, A., Ba, J., et al. (2019). Graph normalizing flows. *arXiv preprint arXiv:1905.13177*.

Lopez-Paz, D., & Oquab, M. (2016). Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*.

Maidment, J. R. (1986). *DESY III: Design Report* (tech. rep. No. DESY-HERA-86-11). https://doi.org/10.3204/PUBDB-2017-01008

Mikuni, V., & Nachman, B. (2022). Score-based Generative Models for Calorimeter Shower Simulation. *Phys.Rev.D*, *106*, 092009. https://doi.org/10.1103/PhysRevD.106.092009

Mikuni, V., & Nachman, B. (2024). CaloScore v2: single-shot calorimeter shower simulation with diffusion models. *JINST*, *19*(02), P02001. https://doi.org/10.1088/1748-0221/19/02/P02001

Mikuni, V., Nachman, B., & Pettee, M. (2023). Fast Point Cloud Generation with Diffusion Models in High Energy Physics. *Phys.Rev.D*, *108*, 036025. https://doi.org/10.1103/PhysRevD.108.036025

Miyamoto, S., & Horikawa, K. (2008). New subaru gamma-ray beam source and application research. *The Review of Laser Engineering*, *36*, 798–805. https://doi.org/10.2184/lsj.36.798

Nachman, B. (2016). *Investigating the Quantum Properties of Jets and the Search for a Supersymmetric Top Quark Partner with the ATLAS Detector* [Doctoral dissertation, Stanford U., Phys. Dept.].

Neyman, J., & Pearson, E. S. (1933). On the Problem of the Most Efficient Tests of Statistical Hypotheses. *Phil. Trans. Roy. Soc. Lond. A*, *231*(694-706), 289–337. https://doi.org/10.1098/rsta.1933.0009

Nielsen, D., Jaini, P., Hoogeboom, E., et al. (2020). Survae flows: Surjections to bridge the gap between vaes and flows. *arXiv preprint arXiv:2007.02731*.

Noether, E. (1918). Invariante variationsprobleme. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, *1918*, 235–257. http://eudml.org/doc/59024

Oliva, J., Dubey, A., Zaheer, M., Poczos, B., Salakhutdinov, R., Xing, E., & Schneider, J. (2018). Transformation autoregressive networks. *International Conference on Machine Learning*, 3898–3907.

Paganini, M., de Oliveira, L., & Nachman, B. (2018a). Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multilayer Calorimeters. *Phys. Rev. Lett.*, *120*(4), 042003. https://doi.org/10.1103/PhysRevLett.120.042003

Paganini, M., de Oliveira, L., & Nachman, B. (2018b). CaloGAN : Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks. *Phys. Rev.*, (1), 014021. https://doi.org/10.1103/PhysRevD.97.014021

Pais, A., & Treiman, S. (1975). How many charm quantum numbers are there? *Physical Review Letters*, *35*(23), 1556.

Pang, I., Raine, J. A., & Shih, D. (2023). SuperCalo: Calorimeter shower super-resolution. *arXiv preprint arXiv:2308.11700*.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2021). Normalizing Flows for Probabilistic Modeling and Inference. *J. Machine Learning Res.*, *22*(1), 2617–2680. https://doi.org/10.5555/3546258.3546315

Papamakarios, G., Pavlakou, T., & Murray, I. (2017). Masked Autoregressive Flow for Density Estimation.

Pearson, K. (1904). *On the theory of contingency and its relation to association and normal correlation*. Cambridge University Press.

Pontryagin, L. S., Boltyanskii, V. G., Gamrelidze, R. V., et al. (1961). *The mathematical theory of optimal processes*.
The Pontryagin Maximization Principle.

Prenger, R., Valle, R., & Catanzaro, B. (2019). Waveglow: A flow-based generative network for speech synthesis. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3617–3621.

Rezende, D., & Mohamed, S. (2015). Variational inference with normalizing flows (F. Bach & D. Blei, Eds.). *Proceedings of the 32nd International Conference on Machine Learning*, *37*, 1530–1538. http://proceedings.mlr.press/v37/rezende15.html

Rezende, D., Mohamed, S., & Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models.

Riebesell, J. (2022, April). Higgs potential. https://tikz.net/higgs-potential/

Robinson, N., Allred, B., Jones, M., Moreno, A., Kimball, J., Naugle, D., Erickson, T., & Richardson, A. (2017). A dynamic landsat derived normalized difference vegetation index (ndvi) product for the conterminous united states. *Remote Sensing*, *9*(8), 863. https://doi.org/10.3390/rs9080863

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Salam, A. (1968). Weak and Electromagnetic Interactions. *Conf. Proc. C*, *680519*, 367–377. https://doi.org/10.1142/9789812795915_0034

Salimans, T., Karpathy, A., Chen, X., & Kingma, D. P. (2017). Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Schnake, S., Krücker, D., & Borras, K. (2022). Generating calorimeter showers as point clouds. https://ml4physicalsciences.github.io/2022/files/NeurIPS%5C%5FML4PS%5C%5F2022%5C%5F77.pdf

Schnake, S., Krücker, D., & Borras, K. (2024). CaloPointFlow II Generating Calorimeter Showers as Point Clouds.

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., & Ganguli, S. (2015). Deep Unsupervised Learning using Nonequilibrium Thermodynamics.

Tabak, E. G., & Turner, C. V. (2013). A Family of Nonparametric Density Estimation Algorithms. *Commun. Pure Appl. Math.*, *66*(2), 145–164. https://doi.org/10.1002/cpa.21423

Tabak, E. G., & Vanden-Eijnden, E. (2010). Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, *8*(1), 217–233. https://doi.org/10.4310/cms.2010.v8.n1.a11

Topsoe, F. (2000). Some inequalities for information divergence and related measures of discrimination. *IEEE Transactions on Information Theory*, *46*(4), 1602–1609. https://doi.org/10.1109/18.850703

Uria, B., Murray, I., & Larochelle, H. (2014). Rnade: The real-valued neural autoregressive density-estimator. *arXiv preprint arXiv:1306.0186*.

Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention Is All You Need. *31st International Conference on Neural Information Processing Systems*.

Vieira, T. (2014). *Gumbel-max trick and weighted reservoir sampling*. Retrieved September 17, 2024, from https://timvieira.github.io/blog/post/2014/08/01/gumbel-max-trick-and-weighted-reservoir-sampling/

Waibel, A., Hanazawa, T., Hinton, G., et al. (1990). Phoneme recognition using time-delay neural networks. *Readings in Speech Recognition*, 393–404. https://doi.org/10.1016/b978-0-08-051584-7.50037-1

Wainwright, M. J., Jordan, M. I., et al. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, *1*(1–2), 1–305.

Weinberg, S. (1967). A Model of Leptons. *Phys. Rev. Lett.*, *19*, 1264–1266. https://doi.org/10.1103/PhysRevLett.19.1264

Werbos, P. J. (1981). Applications of advances in nonlinear sensitivity analysis. *Proceedings of the 10th IFIP Conference, 31.8 - 4.9, NYC*, 762–770.

Wigmans, R. (2018). Calorimetry - energy measurements in particle physics, 2nd edition. *Contemporary Physics*, *59*(2), 226–227. https://doi.org/10.1080/00107514.2018.1450300

Wilkinson, J. H. (Ed.). (1965). *The algebraic eigenvalue problem*. Oxford University Press, Inc.

Workman, R. L., et al. (2022). Review of Particle Physics. *PTEP*, *2022*, 083C01. https://doi.org/10.1093/ptep/ptac097

Yang, C. N., & Mills, R. L. (1954). Conservation of isotopic spin and isotopic gauge invariance. *Phys. Rev.*, *96*, 191–195. https://doi.org/10.1103/PhysRev.96.191

Yang, G., Huang, X., Hao, Z., et al. (2019). PointFlow: 3D Point Cloud Generation With Continuous Normalizing Flows. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2019.00464

Zaheer, M., Kottur, S., Ravanbakhsh, S., et al. (2017). Deep Sets. *CoRR*. http://arxiv.org/abs/1703.06114v3

Zurbano Fernandez, I., et al. (2020). High-Luminosity Large Hadron Collider (HL-LHC): Technical design report (I. Béjar Alonso, O. Brüning, P. Fessia, L. Rossi, L. Tavian, & M. Zerlauth, Eds.). *CERN Yellow Reports: Monographs*. https://doi.org/10.23731/CYRM-2020-0010

Zweig, G. (1964, February). An SU(3) model for strong interaction symmetry and its breaking. Version 2. In D. B. Lichtenberg & S. P. Rosen (Eds.), *Delevopments in the Quark Theory of Hadrons. VOL. 1. 1964 - 1978* (pp. 22–101).

Eidesstattliche Erklärung

Simon Patrik Schnake erklärt hiermit, dass diese Dissertation und die darin dargelegten Inhalte die eigenen sind und selbstständig, als Ergebnis der eigenen originären Forschung, generiert wurden.

Hiermit erkläre ich an Eides statt:

1. Diese Arbeit wurde vollständig oder größtenteils in der Phase als Doktorand dieser Fakultät und Universität angefertigt;
2. Sofern irgendein Bestandteil dieser Dissertation zuvor für einen akademischen Abschluss oder eine andere Qualifikation an dieser oder einer anderen Institution verwendet wurde, wurde dies klar angezeigt;
3. Wenn immer andere eigene- oder Veröffentlichungen Dritter herangezogen wurden, wurden diese klar benannt;
4. Wenn aus anderen eigenen- oder Veröffentlichungen Dritter zitiert wurde, wurde stets die Quelle hierfür angegeben. Diese Dissertation ist vollständig meine eigene Arbeit, mit der Ausnahme solcher Zitate;
5. Alle wesentlichen Quellen von Unterstützung wurden benannt;
6. Wenn immer ein Teil dieser Dissertation auf der Zusammenarbeit mit anderen basiert, wurde von mir klar gekennzeichnet, was von anderen und was von mir selbst erarbeitet wurde;
7. Teile dieser Arbeit sind bereits in den folgenden Referenzen veröffentlicht worden:
    - Schnake, S., Krücker, D., & Borras, K. (2022). Generating Calorimeter Showers as Point Clouds.
    - Schnake, S., Krücker, D., & Borras, K. (2024). CaloPointFlow II Generating Calorimeter Showers as Point Clouds.
    - Claudius Krause et al. CaloChallenge 2022: A Community Challenge for Fast Calorimeter Simulation.

Hamburg, den 10.06.2025 Simon Patrik Schnake