

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/391047636>

End-to-End Detector Optimization with Diffusion Models: A Case Study in Sampling Calorimeters

Article in *Particles* · April 2025

DOI: 10.3390/particles8020047

CITATIONS

0

READS

2

19 authors, including:



[Muhammad Awais](#)

University of Padua

12 PUBLICATIONS 33 CITATIONS

[SEE PROFILE](#)



[Xuan Tung Nguyen](#)




















INFN - Istituto Nazionale di Fisica Nucleare

5 PUBLICATIONS 0 CITATIONS

[SEE PROFILE](#)

Article

End-to-End Detector Optimization with Diffusion Models: A Case Study in Sampling Calorimeters

Kylian Schmidt ^{1,*} , Krishna Nikhil Kota ¹ , Jan Kieseler ^{1,†} , Andrea De Vita ^{2,6} , Markus Klute ¹ , Abhishek ⁴ , Max Aehle ^{5,†} , Muhammad Awais ^{2,3} , Alessandro Breccia ² , Riccardo Carroccio ² , Long Chen ^{5,†} , Tommaso Dorigo ^{3,6,†,‡} , Nicolas R. Gauger ^{5,†} , Enrico Lupi ^{2,6} , Federico Nardi ^{2,7,†} , Xuan Tung Nguyen ^{5,6} , Fredrik Sandin ^{3,†} , Joseph Willmore ²  and Pietro Vischia ^{8,†} 

¹ Institute for Experimental Particle Physics, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany; ep21btech11016@iit.ac.in (K.N.K.); jan.kieseler@cern.ch (J.K.); markus.klute@kit.edu (M.K.)

² Dipartimento di Fisica e Astronomia, Università di Padova, Via F. Marzolo 8, 35131 Padova, Italy; andrea.devita.1@studenti.unipd.it (A.D.V.); awaisafzal628@gmail.com (M.A.); alessandro.breccia@studenti.unipd.it (A.B.); riccardo.carroccio@studenti.unipd.it (R.C.); enrico.lupi@studenti.unipd.it (E.L.); federico.nardi.4@studenti.unipd.it (F.N.); josephwillmore@icloud.com (J.W.)

³ Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, 971 87 Luleå, Sweden; tommaso.dorigo@gmail.com (T.D.); fredrik.sandin@ltu.se (F.S.)

⁴ National Institute of Science Education and Research, Jatni 752050, India; abhishek.2019@niser.ac.in

⁵ Scientific Computing Group, University of Kaiserslautern-Landau (RPTU), Gottlieb-Daimler-Straße, 67663 Kaiserslautern, Germany; max.aehle@scicomp.uni-kl.de (M.A.); long.chen@scicomp.uni-kl.de (L.C.); nicolas.gauger@scicomp.uni-kl.de (N.R.G.); xuantung.nguyen@pd.infn.it (X.T.N.)

⁶ INFN, Sezione di Padova-Via F. Marzolo 8, 35131 Padova, Italy

⁷ Laboratoire de Physique Clermont Auvergne, 63170 Aubière, France

⁸ ICTEA, Universidad de Oviedo, 33004 Oviedo, Spain; pietro.vischia@cern.ch

* Correspondence: kylian.schmidt@kit.edu

† MODE Collaboration: <https://mode-collaboration.github.io>.

‡ Universal Scientific Education and Research Network, Italy.

Abstract: Recent advances in machine learning have opened new avenues for optimizing detector designs in high-energy physics, where the complex interplay of geometry, materials, and physics processes has traditionally posed a significant challenge. In this work, we introduce the end-to-end. AI Detector Optimization framework (AIDO), which leverages a diffusion model as a surrogate for the full simulation and reconstruction chain, enabling gradient-based design exploration in both continuous and discrete parameter spaces. Although this framework is applicable to a broad range of detectors, we illustrate its power using the specific example of a sampling calorimeter, focusing on charged pions and photons as representative incident particles. Our results demonstrate that the diffusion model effectively captures critical performance metrics for calorimeter design, guiding the automatic search for a layer arrangement and material composition that align with known calorimeter principles. The success of this proof-of-concept study provides a foundation for the future applications of end-to-end optimization to more complex detector systems, offering a promising path toward systematically exploring the vast design space in next-generation experiments.

Keywords: computational modeling; machine learning; diffusion model; calorimeter; particle detector; holistic optimization



Academic Editor: Armen Sedrakian

Received: 3 February 2025

Revised: 27 February 2025

Accepted: 2 April 2025

Published: 23 April 2025

Citation: Schmidt, K.; Kota, K.N.; Kieseler, J.; De Vita, A.; Klute, M.; Abhishek; Aehle, M.; Awais, M.; Breccia, A.; Carroccio, R.; et al. End-to-End Detector Optimization with Diffusion Models: A Case Study in Sampling Calorimeters. *Particles* **2025**, *8*, 47. <https://doi.org/10.3390/particles8020047>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Designing a high-performance detector for particle physics is inherently a high-dimensional optimization challenge, requiring the reconciliation of multiple objectives, such as energy or

momentum resolution, timing accuracy, and cost. The design space may include both continuous parameters (e.g., layer thickness) and discrete parameters (e.g., material choices), as well as the reconstruction algorithms that translate raw detector signals into physics observables. This complexity is further compounded by the stochastic nature of particle interactions, which complicates the application of gradient-based methods [1]. While sufficiently generic and differentiable reconstruction algorithms exist (e.g., Refs. [2–5]), the high-fidelity simulations describing particle interactions with matter are typically not differentiable, preventing the straightforward use of gradient descent. Ongoing efforts to overcome this limitation include making established simulation packages directly differentiable [6–8], developing custom differentiable simulation pipelines from the ground-up (e.g., TomOpt [9]), or employing local surrogate models that are valid within a limited trust region and retraining them once that region is exited [10]. Each of these approaches faces specific challenges, especially when dealing with detectors containing hundreds of thousands of readout channels.

To address these difficulties, an *end-to-end* approach can be employed, whereby only one global performance quantity is predicted by a surrogate neural model for each sample in the dataset (event). This stands in contrast with approaches that use generative models to recreate the statistic patterns of low-level information in the detector, such as [11–15]. The end-to-end strategy abstracts the problem away from the microscopic details of the simulation and reconstruction, permitting the integration of the model into a comprehensive software pipeline. Such an end-to-end surrogate has the advantage of capturing the essential mapping from the design parameters to physics-relevant performance metrics without having to explicitly model each individual hit in a high-granularity detector. It also enables more direct optimization loops, as the gradients obtained from the end-to-end model can directly steer the design choices. Notably, discrete parameters such as the presence or absence of specific sub-detectors or the choice of different materials can be accommodated by neural networks through careful encoding, or with techniques that effectively smooth the discrete design space [9].

Among the various detector subsystems in modern particle physics experiments, calorimeters stand out both for their critical role in measuring particle energies and for their inherently large design space. Calorimeters have been widely used since the 1950s to determine the energy of incident particles by sandwiching high-density passive layers and active materials that produce measurable signals. In collider-based experiments, the performance demands on calorimeters have rapidly evolved to address new challenges: for instance, highly segmented calorimeters in both transverse and longitudinal directions can identify the hadronic decays of boosted heavy particles inside wide jets [16–19], and the availability of a large magnetic field combined with high segmentation permits efficient particle flow algorithms [20,21] to significantly improve event reconstruction. Meanwhile, new fabrication and sensor technologies, such as 3D-printed scintillators or the use of highly granular silicon as the active medium [22], further expand the design space and underscore the importance of systematic optimization.

In this work, we illustrate how the end-to-end framework AIDO can be deployed to optimize both continuous and discrete parameters of a sampling calorimeter. We focus on a simplified setup involving only photons and charged pions as incident particles, and we seek to identify the arrangement of passive and active layers that provides the best energy reconstruction performance. Far from claiming to discover new, cutting-edge calorimeter designs, our primary goal is to demonstrate that the pipeline can identify designs that align with well-known calorimeter principles. This validation paves the way for future studies involving more complex objectives and design choices, where human intuition alone may fall short of identifying optimal solutions.

This paper is organized as follows. In Section 2, we motivate the comparison of different detector designs based on a single metric computed from simulations. Section 3

introduces the concept of a digital twin and explains how it boosts gradient computation, making end-to-end optimization feasible. Section 4 details the workflow structure for the iterative simulation-reconstruction-optimization loop, while Section 5 describes how discrete detector parameters can be encoded and learned. In Section 6, we apply this end-to-end optimization approach to a sampling calorimeter. Finally, Section 7 summarizes our findings and outlines potential applications and improvements for future work.

2. Mapping of Detector Parameters to Performance

Geant4 [23–25] is a fast simulation framework that accurately reproduces the interaction of particles with matter and is widely used in High-Energy Physics (HEP) to understand the behavior of detectors. The framework provides a common core of functionalities, on top of which developers can build a simulation software for their specific use-case. The flexibility provided by Geant4 allows physicists and engineers to quickly test and refine the design of new detectors by sharing some tunable parameters with other programs (macro files, python bindings, or Command Line arguments). While these interfaces are convenient for human users, they also open up another potential application if connected with a hyper-network, as discussed in Section 3.

Typically, the outputs of a Geant4 simulation are divided into events, each representing the full initial-particles-to-final-readout chain, including, for example, the energy deposited in a single cell, the number of particles that passed through a given layer, or the direction of flight of a particle. This low-level information is fed into custom algorithms, which compute further physical quantities of interest, such as the momentum, the total recorded energy, or the flight time. One considerable advantage of simulations in general is that, in contrast to real HEP experiments, the initial conditions of an event are precisely known and reproducible. Consequently, reconstruction algorithms that produce high-level variables, such as the energy resolution of a calorimeter or the accuracy of a tracking component, can be evaluated in an unbiased way. It is this observation that motivates the automatic optimization of detectors, which can be achieved by tuning the parameters and computing the resulting performance with a reconstruction algorithm.

Given a detector simulation using a set of adjustable parameters θ , we denote the configuration space of all possible detectors as $\Theta = \{\theta^0 \times \theta^1 \cdots \times \theta^P\}$, where each parameter θ^p describes a certain physical property of the detector. This includes continuous parameters, such as the position or thickness of a component, and categorical parameters, such as a material type. For continuous parameters, one must ensure that the physical meaning is properly encoded by setting boundaries, e.g., forbidding negative thicknesses. Further information about the translation of discrete parameters into a suitable format for Machine Learning (ML) is detailed in Section 5.

As a first step, we consider the simulation to be equivalent with a mapping $f : \theta, \theta_{\text{ext}} \mapsto F_{\text{sim}}$ that takes as inputs θ and a second set of external variables θ_{ext} , which are potentially pseudo-random, making f a stochastic process. We categorize the outputs into three terms, $F_{\text{sim}} = \{x, T, C\}$, where x is the set of all low-level, hit-based features (the energy deposited in each cell, the position of an individual hit, etc.). The targets T , or true information, describe the known initial conditions of the simulation and are used for the evaluation of the reconstruction algorithm (such as the true initial energy). Lastly, C denotes all additional context information for each event (such as the initial particle type), which is used as input and not for evaluation. This distinction is important to inform the reconstruction algorithm of the differences between events that are not encoded in the hit-based features x .

In the second step, the reconstruction is a second function $g : \theta, F_{\text{sim}} \mapsto E_{\text{reco}}$, which produces the reconstructed targets E_{reco} containing all the high-level variables of interest based on the outputs of the simulation. This is usually the end of the reconstruction chain, but in the

present work we are interested in comparing the performance of different detector designs. For this purpose, we compare E_{true} and E_{reco} using a Loss function, yielding a single scalar Loss term \mathcal{L} that describes the goodness of each detector design. In essence, we can combine the effects of simulation and reconstruction on θ into a single function $V : \theta \mapsto \mathcal{L}$ that yields a unique Loss for each set of parameters θ , excluding stochastic effects. The function V , and by extension the Loss \mathcal{L} , include stochastic noise added by the simulation and potentially also by the reconstruction itself, e.g., if it is computed using a Deep Neural Network (DNN). The stated goal of this work is therefore to efficiently explore the performance-space of $V : \theta \mapsto \mathcal{L}$ and find

$$\theta_{\text{opt}} = \underset{\theta}{\operatorname{argmin}}(\mathcal{L}, V). \quad (1)$$

By definition, θ_{opt} is the set of (optimal) parameters that minimizes \mathcal{L} . This minimum can be found using conventional gradient descent methods [26], as long as the stochastic noise of \mathcal{L} is not too large. In practice however, this noise tends to dominate the local Loss landscape, and methods such as the one described next are required to obtain meaningful gradients.

3. Digital Twin Approach and Optimization

In the previous Section, we introduced an example of a digital twin, namely the Geant4 simulation software that reproduces the behavior of a real-world detector. This Section proposes a second digital twin to emulate the behavior of the combined simulation and reconstruction system. This second digital twin is motivated by the fact that the loss \mathcal{L} is both highly noisy and computationally expensive. Indeed, even though a sequential pipeline that computes $V(\theta) = \mathcal{L}$ is sufficient for the task of optimization, it would require a full simulation and reconstruction from scratch for each gradient step and be highly inefficient. We demonstrate a significant improvement in computational efficiency by learning the expected detector performance with a Deep Neural Network, which addresses both challenges: first, once learned, its evaluation is inexpensive, and second, it partially smooths out irregularities in the gradient.

This approach starts by sampling a subset $\Theta_{\text{sample}} \subseteq \Theta = \{\theta_0, \dots, \theta_N\}$ from a confined region within the larger parameter space. We sample using a multi-variate normal distribution $\theta_{\text{sample}} = \mathcal{N}(\theta, K_{\theta})$ for a total of N simulations, where K_{θ} is the sampling covariance matrix. A generative DNN, named the surrogate model in the following, learns function S , which locally approximates the effects of $V(\theta)$. In this way, the performance of a new parameter $\theta_{\text{new}} \notin \Theta_{\text{sample}}$ is approximated by evaluating $S(\theta_{\text{new}}) = \mathcal{L}'_{\text{new}}$, provided that θ_{new} is also within the training region Θ_{sample} . Figure 1 shows this equivalence between the simulation–reconstruction system and the surrogate model.

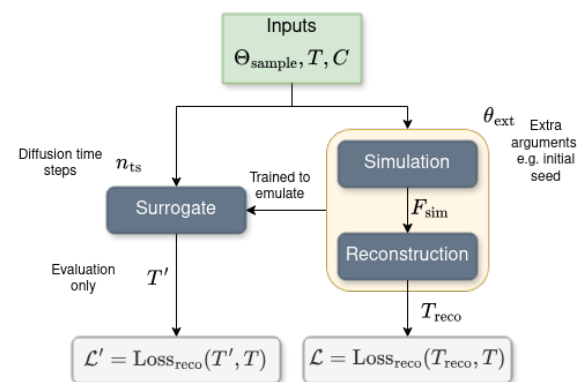


Figure 1. Illustration of the surrogate model as a digital twin of the combined reconstruction and simulation system. By avoiding the internal low-level information F_{sim} , the surrogate is able to efficiently produce similar outputs $T' \approx T_{\text{reco}}$.

The minimization problem stated in Equation (1) is performed locally by minimizing \mathcal{L}' instead, which only requires the evaluation of $S(\theta)$. This approach enables the simulation to run in parallel and the reconstruction algorithm to train on several detector designs at the same time, greatly reducing computation time. To ensure that the learned approximation of $\mathcal{L}' \approx \mathcal{L}$ holds, it is necessary to validate the predictions of the surrogate model, as inaccuracies would lead to a misrepresentation of the physical detector.

3.1. Surrogate Model

The surrogate model is a DNN tasked with reproducing the performance \mathcal{L}' for any θ_{new} within the sampled region. There are two main ways for the surrogate to learn \mathcal{L}' , either directly using $\mathcal{L}_{\text{reco}}$ or indirectly by learning T_{reco} and evaluating $\mathcal{L}' = \text{Loss}_{\text{reco}}(T', T)$. For this application, we train a conditional De-noising Diffusion Probabilistic model (DDPM) [27] on the targets T_{reco} . Diffusion models are able to generate new data by training on progressively noisier samples and reversing the noise during evaluation. The advantages of diffusion models include its stable training, high-fidelity samples and potentially easier transfer learning due to their probabilistic nature.

For our purposes, the internal training model is a simple feed-forward network composed of four linear layers using the architecture detailed in Table 1. The training is carried out on the dataset $\{\theta, C, T_{\text{reco}}\}$, where the goal of the model is to predict the added noise. We train the model using the Adam optimizer [28] and a Mean Squared Error (MSE) loss between the *predicted* and the *true* added noise. The Adam optimizer is momentum-based and enhances the transfer learning between iterations. During evaluation, the surrogate predicts new targets $T' = S(\theta', C)$ used to approximate the reconstruction Loss $\mathcal{L}' = \text{Loss}_{\text{reco}}(T', T)$.

Table 1. The architecture of the DNN used by the surrogate model. The input nodes of the first layer are the detector parameters θ , the context information C , the target quantities T , and the internal time step variable n_{ts} , which accounts for the +1 term.

Layer	Input Nodes	Output Nodes	Activation Function
1	$\theta + C + T_{\text{reco}} + 1$	200	ELU
2	200	100	ELU
3	100	T'	Linear

3.2. Optimizer

The optimizer object is a wrapper around the Adam optimizer that adjusts the parameters θ in order to minimize

$$\mathcal{L}_{\text{opt}} = \mathcal{L}' + \mathcal{L}_{\text{penalties}} + \mathcal{L}_{\text{boundaries}} \quad (2)$$

where \mathcal{L}' is the Loss predicted by the surrogate, $\mathcal{L}_{\text{penalties}}$ is a user-defined additional term that takes into account effects such as total cost, maximum size or other physical constraints, and $\mathcal{L}_{\text{boundaries}}$ is a penalty term that ensures that the parameters remain within the evaluation space of the surrogate. This boundary loss is defined as

$$\mathcal{L}_{\text{boundaries}} = \text{mean} \left(\frac{1}{2} \left[\text{Relu} \left(\theta - \frac{\theta_{\text{max}}}{1.1} \right) \right]^2 + \frac{1}{2} \left[\text{Relu} \left(\frac{\theta_{\text{min}}}{1.1} - \theta \right) \right]^2 \right). \quad (3)$$

The scaling 1/1.1 ensures that the penalty loss activates slightly before the actual limit is reached, allowing the optimizer to adjust to the penalty sooner. After each parameter

update, we check that the parameters are still within the well-defined region of the surrogate evaluation space spanned by K_θ with the binary decision

$$(\theta_{n+1} - \theta_n) K_\theta^{-1} (\theta_{n+1} - \theta_n) < b, \quad (4)$$

where θ_{n+1} are the new parameters after the parameter update, θ_n are the central parameters of the trust region of the surrogate, and factor $b \leq 1$ is the scaling of the valid region for the surrogate. We chose the conservative value $b = 0.8$ to ensure that the optimized parameters θ_{n+1} remain well within the valid region. As soon as this condition does not hold anymore, an early stopping activates, and the surrogate model is retrained on the new region. This stopping defines the end of a single iteration of the simulation–reconstruction–surrogate–optimizer (SRSO) pipeline. In order to encourage exploration in the next iteration, we adjust the sampling covariance K_θ given by

$$K_\theta = \text{diag}(\sigma_0, \dots, \sigma_m)^2 + (s - 1) \frac{(\theta_{n+1} - \theta_n) \otimes (\theta_{n+1} - \theta_n)}{|\theta_{n+1} - \theta_n|^2}, \quad (5)$$

where $(\theta_{n+1} - \theta_n)$ is the distance between the current and previous parameters and $s = 0.8 \cdot \max(1, 4|\theta_{n+1} - \theta_n|)$ is an appropriate scaling factor if each $\sigma_m \approx \mathcal{O}(1)$. Concretely, after each iteration where the optimizer reaches the boundaries, the sampling region expands in the direction of change. The resulting modified parameters θ_{n+1} become the new central values for the next iteration.

4. Workflow of the AIDO Package

The AIDO software is a dedicated python package that optimizes the arbitrary detector hyperparameters, as described in the previous Section. The package is composed of two layers: the first is a scheduler that coordinates the execution of the individual sub-components using b2luigi [29]; second is the implementation of the surrogate and optimizer models using pytorch [30]. The AIDO scheduler subdivides the SRSO chain into independent Tasks that can be run on HPC using job schedulers such as HTCondor [31], as shown in Figure 2.

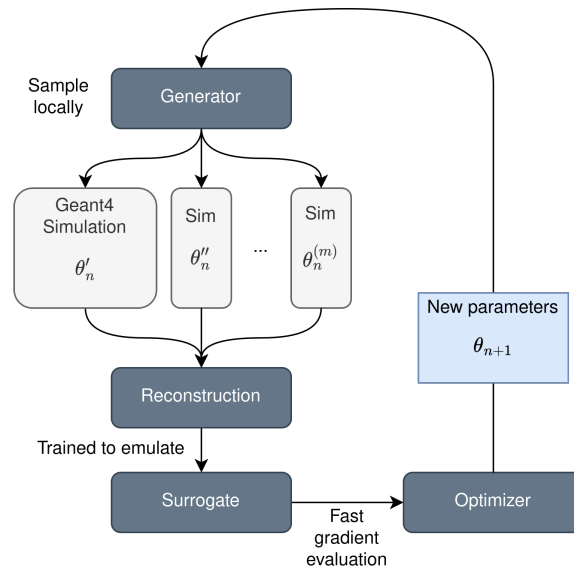


Figure 2. Workflow of a single SRSO iteration using the AIDO package. The top-level generator samples a new set of detector parameters, which are simulated in parallel with Geant4. The outputs of the simulations are used to produce a single reconstruction \mathcal{L} loss. Based on these, the surrogate and optimizer system predicts a new set of parameters for the next iteration.

4.1. Generation

We start with an initial set of parameters $\theta_{n=0}$, which can be chosen manually or randomly. For a set of N parallel simulations, we sample N times from the multivariate normal distribution $\mathcal{N}(\theta, K_\theta)$, where the sampling covariance matrix K_θ encapsulates the correlations between each parameters. This covariance is initialized as $K_\theta = \text{diag}(\sigma_0^2, \dots, \sigma_m^2)$, where σ_m^2 is the variance of each parameter. The choice of the initial variance for each parameter is free, but is learned by the optimizer for all further iterations $n > 0$ according to Equation (5). In addition, sampled parameters that lie outside the allowed boundaries are resampled until valid. Each sampled θ^{sample} is then the configuration of a different detector, which is simulated in the next step.

4.2. Simulation

The simulations are executed in parallel, each with a unique set of detector parameters and a designated output file path. In practice, the propagation of the parameters to Geant4 depends on the available interface. By default, programs and end-users can communicate with a Geant4 executable by passing a macro file that lists a series of Geant4 commands. The values of the detector hyperparameters can then be written with a single script to this file. Some Geant4 programs are linked with a python interface (using C++ to python binding, such as pybind [32]). This allows for a simple propagation of parameters to the individual simulation software. Due to the external Geant4 libraries that are often required by individual Geant4 programs, the approach of containerization is highly recommended. While the output of Geant4 programs is, in most cases, a ROOT file [33], a specific file format is not required by the AIDO software.

4.3. Reconstruction

Once all the simulations are run, the reconstruction computes the goodness of each design. Since regular reconstruction algorithms (especially ML-based) are usually applied to a single large dataset, the AIDO package provides a hook for merging and converting the output files of the simulations into a single dataset and writing it to file. This step ensures that the reconstruction runs on a homogeneous dataset, with the correct format and within its own design conditions, providing maximal flexibility. It is then the responsibility of the user to ensure that the reconstruction algorithm outputs a meaningful performance metric \mathcal{L} for each design.

5. Optimization of Discrete Parameters

Categorical parameters are, as such, not suitable for optimization due to their inherent discontinuity. To overcome this, instead of tuning a single discrete parameter, we assign a probability to each category, reflecting the model's confidence in that category. Equivalently, we can sample from these probability distributions to produce a representative dataset. The sampled categorical values are encoded as one-hot vectors in the training dataset of the surrogate. A parameter ϕ composed of i distinct categories $\phi = \{\phi^0, \dots, \phi^i\}$ is represented by the $i \times i$ matrix M_ϕ , defined as follows:

$$M_\phi = \hat{\mathbb{I}} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}, \quad (6)$$

where category ϕ^i is the i -th row vector. The one-hot encoding scheme is paramount to ensure proper communication between the surrogate and the optimizer. While the

surrogate is trained on a dataset composed of one-hot vectors, it is still able to interpret the query of the optimizer, which is displayed as fractional values. This is due to the inherent ability of DNNs to interpolate between values within their training space. To facilitate training, instead of learning the probabilities directly, the optimizer adjusts the log-probabilities, or logits, $\{z_n\}$, which are defined as $P(\phi_n) = \text{softmax}(z_n)$.

6. Application to a Sampling Calorimeter

As an illustration of possible applications for the AIDO framework, we showcase a sampling calorimeter composed of an absorber and active scintillator layers. Absorber material refers to any passive component that is not part of the readout. Scintillators, on the other hand, are materials that emit optical photons when they are traversed by high-energy particles, which allows for an electronic readout. The layers themselves are shaped as rectangular cuboids with a side-length of 50 cm and are stacked longitudinally along the beam axis, alternating between absorber and scintillator. We set the thickness along the beam axis and the material of the six layers is used as the optimizable parameters. We bound the thickness of all layers to be strictly positive in order to avoid unphysical layouts. The material is a binary choice between either lead or iron for each one of the absorber layers and between lead tungsten PbWO_4 and polystyrene $(\text{C}_8\text{H}_8)_n$ for each scintillator layer.

The task of the optimizer is then to improve the energy resolution by adjusting the thickness of each layer and by choosing the suitable materials. From a physical standpoint, a good calorimeter should maximize the amount of energy that is deposited by the particles as they travel through the material. This can be achieved by increasing the volume of active material while balancing the volume of the absorber, as passive material helps to slow down highly energetic particles, making them more prone to interactions with the active material. Based on the much shorter radiation length of PbWO_4 (1.27 cm) compared to polystyrene (41.31 cm), we expect the optimizer to prefer the use of the former over the latter for the scintillator layers, according to the values found in [34]. The choice of absorber is not as decisive, as the radiation lengths of Fe (1.76 cm) and Pb (0.56 cm) are similar. In the absence of any other constraints, the optimal calorimeter for these conditions would be a single block of PbWO_4 . To address practical considerations, we introduce the cost and size constraints.

6.1. Additional Constraints

In order to recreate a realistic engineering problem, we constrain the total length of the detector to $d_{\max} = 200$ cm. This is enforced with a penalty term of the form

$$\mathcal{L}_{\text{length}} = 10(\text{Relu}(d - d_{\max}))^2. \quad (7)$$

We set the maximal total cost for the whole detector at $\text{cost}_{\max} = 200\text{k EUR}$, which is calculated based on the weighted cost for each layer

$$\text{cost} = \sum_{l=0}^6 d_l c_l P(\phi_l), \quad (8)$$

where c_l is a vector of the same length as ϕ_l , describing the cost for each category. For the three absorber layers, the cost is 25 EUR/cm for lead and 4.16 EUR/cm for iron, while for the scintillator layers the cost is 2.5k EUR/cm for PbWO_4 and 0.01 EUR/cm for polystyrene. We chose these values based on the approximate market price per volume, focusing on the large discrepancy between PbWO_4 and polystyrene. In this way, we ask the optimizer to balance the performance gained by choosing the better material against the increase in total cost. The penalty scaling for exceeding the allowed cost is

$$\mathcal{L}_{\text{cost}} = \text{Relu}\left(\frac{\text{cost}}{\text{cost}_{\text{max}}} - 1\right)^2. \quad (9)$$

Finally, Figure 3 shows the starting and the optimized configuration of the detector with the above cost and size constraints.

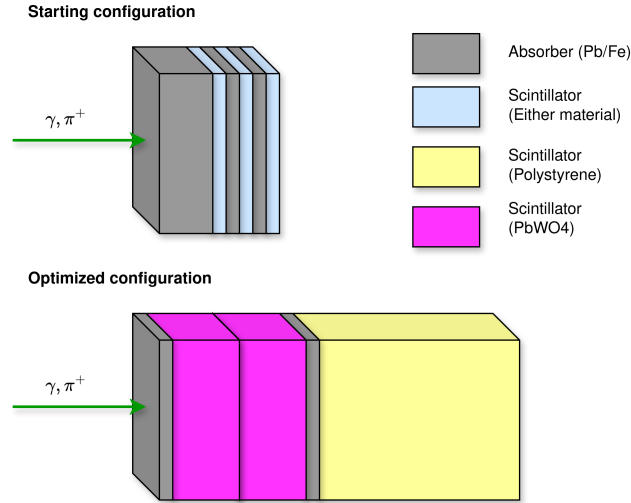


Figure 3. Detector configuration before optimization (**upper**) and after (**lower**). The incoming particles impinge perpendicularly on the detector. In the starting configuration, most particles are stopped by the absorber before they can be recorded by the scintillators. In the optimized configuration, photons are recorded by the large PbWO_4 block, while the longer pion showers are contained by the polystyrene section.

6.2. Simulation and Reconstruction

For each iteration, we simulate 20 distinct detector configurations, as described in Section 4.1. The simulations are run in parallel automatically by the AIDO scheduler, and each simulation runs in single-threaded mode with 400 events. In each event, we shoot a single initial particle, with a random energy that is uniformly sampled between 1 and 20 GeV; in half of the events, the particle is a photon, while in the other half it is a positively charged pion. The goal of this example is to optimize the twelve detector parameters (layer thickness and material composition) such that we minimize the energy resolution of the total calorimeter. For this purpose, we record the idealized total deposited energy in each of the three scintillator layers, without any readout effects.

The reconstruction algorithm is a feed-forward neural network implemented in pytorch, composed of two distinct blocks: a pre-processing block and a main block, summarized in Table 2 and Table 3, respectively. The pre-processing block consists of three fully connected layers, designed to process the initial input parameters ($\#\theta = 12$). The main block processes is composed of four fully connected layers with a single output node that is the reconstructed energy. The learning is performed using the MSE loss between the reconstructed energy E_{rec} and the true initial energy E_{true} of the particle in each event:

$$\mathcal{L} = \text{mean}\left(\frac{(E_{\text{rec}} - E_{\text{true}})^2}{E_{\text{true}} + 1}\right), \quad (10)$$

where E_{rec} is the energy predicted by the reconstruction model and E_{true} is the true MC initial energy. We chose a scaling of $1/(E_{\text{true}} + 1)$ to regulate the importance of events with a higher initial energy. In Section 6.4, we will discuss the impact of this Loss on the final detector configuration. During the forward pass, the output of the pre-processing block is multiplied element-wise with the input vector F_{sim} . The result is concatenated with the

detector parameters θ and passed into the main block. For the surrogate and optimizer, we use the configuration detailed in Section 3, with a learning rate of $lr_S = 0.001$ for the surrogate and $lr_O = 0.01$ for the optimizer. All computations were performed on a system equipped with two Intel Xeon E5-2630 v4 processors and one NVIDIA Titan X GPU.

Table 2. Reconstruction model: pre-processing block architecture.

Layer	Input Nodes	Output Nodes	Activation Function
1	θ	100	ELU
2	100	100	ELU
3	100	F_{sim}	Relu

Table 3. Reconstruction model: main block architecture. We have one output node ($T = E_{\text{rec}}$): the reconstructed energy per event.

Layer	Input Nodes	Output Nodes	Activation Function
1	$\theta + F_{\text{sim}}$	100	ELU
2	100	100	ELU
3	100	100	Relu
4	100	$T = 1$	Linear

6.3. Training Validation

To evaluate the performance of the reconstruction and surrogate models on unseen data, we generate a validation dataset under the same conditions as the training dataset; see Section 4.1. Before describing the validation results, it is important to emphasize that the goal of AIDO is to provide an algorithm capable of optimizing the detector design. The use of a high-performing reconstruction algorithm is not the primary objective as long as it performs adequately and consistently for all detector configurations. Building on this premise, the validation of the reconstruction model is presented in addition to the surrogate model, which serves as a key step of the AIDO framework.

Figure 4 compares the distribution of the simulated energy with the energy predicted by the reconstruction and the surrogate models. The reconstruction model reproduces the overall distribution of the targets and the surrogate model generates a similarly distributed sample over the range of targets. As already pointed out in Ref. [10], the surrogate only needs to provide a reasonable gradient estimate and does not need to model specific details in most cases.

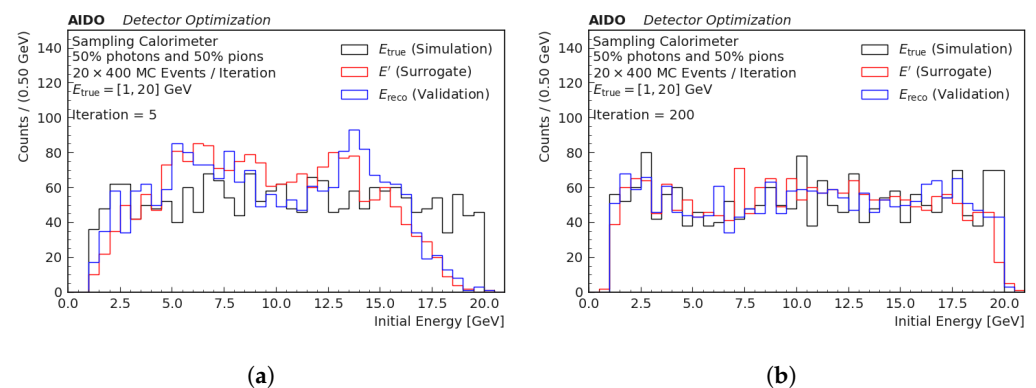


Figure 4. Distribution of the initial energy of the primary particle as predicted by the reconstruction model (E_{reco}) and surrogate model (E'), along with the true simulated energy (E_{true}). The training was performed on 20×400 Monte Carlo events and the validation on 5×400 MC events. (a) shows the validation before and (b) after optimization. In both cases, the surrogate successfully emulates the distribution of samples produced by the reconstruction model.

6.4. Results

The evolution of the detector composition is shown in Figure 5 with the corresponding optimizer loss displayed in Figure 6. From a purely machine learning standpoint, the demonstration is successful: the target metric decreases in a reliable way, confirming the ability of the model to solve the optimization task. However, we can gain further insights into the decisions taken by the model with some task-specific knowledge.

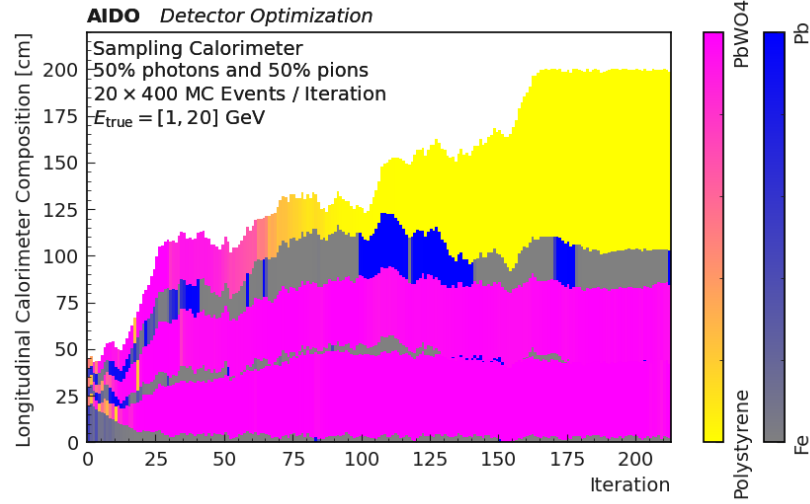


Figure 5. Sliced view of the longitudinal detector composition, with the initial setup at iteration = 0 and the optimized setup at iteration = 220. Each slice on the x -axis represents the currently learned detector configuration (with the side at $y = 0$ facing in the direction of the incoming particle beam, as shown in Figure 3). During training, the large block of the first absorber is gradually thinned while the scintillator layers grow in size.

The initial setup has a large absorber block at the front which traps most of the particles before they can reach the first scintillator layer. Only a few particles are recorded by the subsequent scintillator, so only a small fraction of the training dataset carries meaningful information. In turn, the reconstruction performance is poor, the energy resolution is large, and the optimizer loss is high.

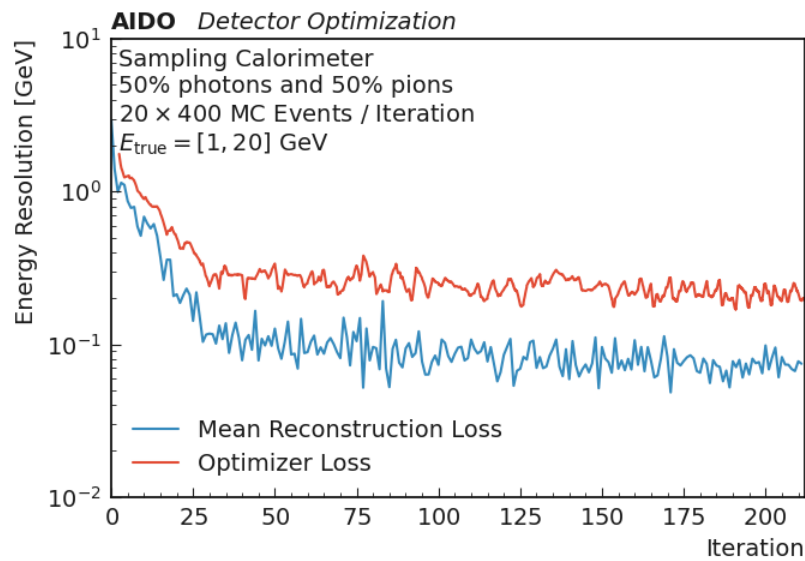


Figure 6. Evolution of the mean reconstruction loss \mathcal{L} for the nominal design during training, as provided by Equation (10), and the synthetic reconstruction loss \mathcal{L}' used as the optimizer loss.

During training, the undesirable absorber layer is gradually removed, which has a net positive impact on the target energy resolution, as shown in Figure 7. At the same time, the scintillator layers increase in length, and since the energy resolution of a calorimeter scales with $1/\sqrt{E}$ [34], a larger active volume leads directly to more deposited energy and better energy resolution. As expected, the optimizer chooses PbWO_4 over polystyrene for the first layer. This makes sense when measuring the energy of photons, as electromagnetic showers benefit from a short but dense active material.

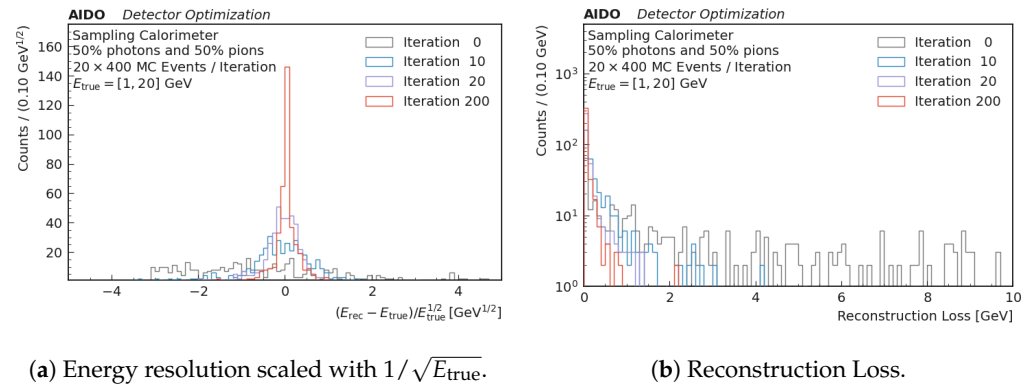


Figure 7. Scaled energy resolution (a) and reconstruction loss (b) of the nominal training dataset (400 MC events) for selected iterations. The narrowing of both distributions indicate the gradual improvement achieved by the network.

After this electromagnetic section, the calorimeter displays a long heterogeneous part. Given the radiation length of PbWO_4 , electromagnetic showers are not expected to have a longitudinal profile that significantly extends beyond 25 cm of material. However, for early showering charged pions, PbWO_4 has excellent properties. Therefore, as can be seen in Figure 8, the available budget is fully exploited, with a focus on the earlier layers and very little or no absorber material.

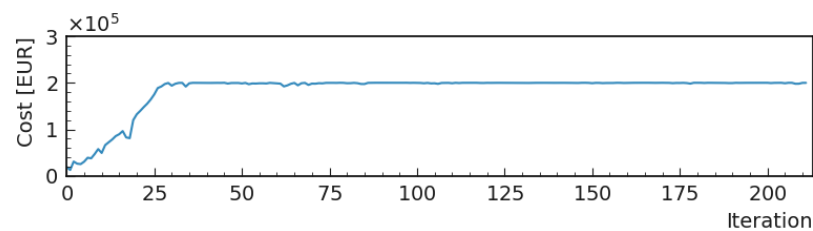


Figure 8. Evolution of the cost constraints during training. The initial growth is mainly due to the increase in the thickness of the scintillator layers made out of the more expensive PbWO_4 .

Once the current composition exhausts the total available budget, the last layer changes to a less expensive material with large radiation length, and therefore requires an absorber to consistently shower the particles. The large iron absorber and the polystyrene scintillator build the last layer, with the shift from PbWO_4 to polystyrene clearly shown in Figure 9. This last section of the detector helps by measuring high-energy pions that shower in the iron block, leaving a meaningful signal in the polystyrene downstream. It can be concluded that the method is able to reconcile multi-target optimization, taking the physics performance into account, as well as cost and length constraints. In particular, the system is capable of optimizing discrete material choices with vastly different costs.

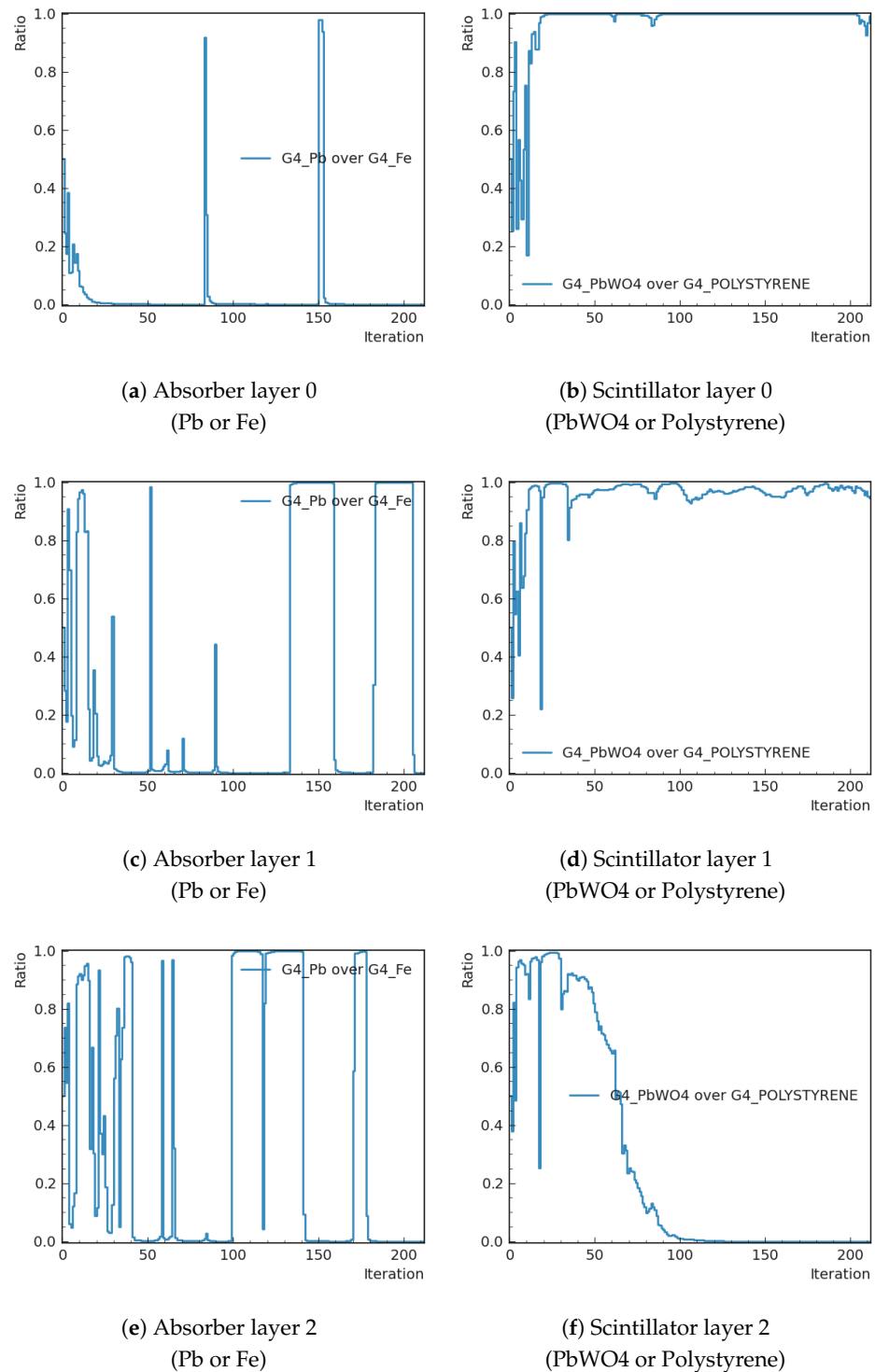


Figure 9. Evolution of the confidence of the optimizer in each material choice. The figures show the fraction of material proposed by the optimizer, with high fractions being considered more desirable by the optimizer. Crucially, the optimizer correctly chooses PbWO₄ as the material for the first and second scintillator layers (b,d). The third layer (f) also starts with PbWO₄ but revert to polystyrene once the cost constraints apply, in order to save costs.

7. Conclusions

In this work, we demonstrate a novel method for investigating the configuration phase space of modern detectors. By using a diffusion-based surrogate model that locally interpolates the behavior of the detector, we are able to efficiently explore and optimize

this configuration space. The method also extends nicely to categorical parameters such as the material choice by adjusting the log-probabilities associated with each category. The AIDO python package allows for large-scale deployment on HPC facilities of the optimization pipeline by atomizing it into separate Luigi Tasks. We confirm the capabilities of the surrogate approach by optimizing a layered calorimeter, which converges towards a familiar design with an electromagnetic section, a section for early showering pions, and a simple sampling calorimeter capturing the remaining hadrons. In the future, our aim will be to further investigate the performance of this model for more complex tasks, including larger parameter spaces and other performance metrics, such as the accuracy of tracking modules. In principle, this method and framework are not limited to calorimeters, such that other tasks, including tracking, can be considered in the future.

Author Contributions: Conceptualization, J.K.; Methodology, J.K. and K.S.; Software, K.S., J.K., K.N.K. and A.D.V.; Validation, A.D.V. and K.S.; Writing—original draft preparation, K.S., A.D.V. and T.D.; Writing—review and editing, K.S., K.N.K., J.K., A.D.V., M.K., A., M.A. (Max Aehle), M.A. (Muhammad Awais), A.B., R.C., L.C., T.D., N.R.G., E.L., F.N., X.T.N., F.S., J.W. and P.V.; Visualization, K.S., K.N.K., J.K., A.D.V., M.K., A., M.A. (Max Aehle), M.A. (Muhammad Awais), A.B., R.C., L.C., T.D., N.R.G., E.L., F.N., X.T.N., F.S., J.W. and P.V.; Supervision, M.K., J.K., T.D., N.R.G., P.V., J.W., L.C., F.S. and N.R.G.; Project administration, M.K., J.K. and T.D.; Funding acquisition, M.K., T.D. and P.V. All authors have read and agreed to the published version of the manuscript.

Funding: The work by T.D. and F.S. was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The work by M.A. and F.S. was partially supported by the Jubilee Fund at the Luleå a University of Technology. The work by P.V. was supported by the “Ramón y Cajal” program under Project No. RYC2021-033305-I funded by the MCIN MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. J.K. is supported by the Alexander-von-Humboldt foundation.

Data Availability Statement: The resources used for the analysis can be found on the following GitHub page (accessed 3 February 2025) <https://github.com/KylianSchmidt/aido>.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SRSO	Simulation–Reconstruction–Surrogate–Optimizer
DNN	Deep Neural Network
DDPM	De-noising Diffusion Probabilistic model
HPC	High-Performance Computing
HEP	High-Energy Physics
MSE	Mean Square Error
ELU	Exponential Linear Unit

References

1. Dorigo, T.; Giammanco, A.; Vischia, P.; Aehle, M.; Bawaj, M.; Boldyrev, A.; de Castro Manzano, P.; Derkach, D.; Donini, J.; Edelen, A.; et al. Toward the end-to-end optimization of particle physics instruments with differentiable programming. *Rev. Phys.* **2023**, *10*, 100085. [[CrossRef](#)]
2. Qasim, S.; Kieseler, J.; Iiyama, Y.; Pierini, M. Learning representations of irregular particle-detector geometry with distance-weighted graph networks. *Eur. Phys. J. C* **2019**, *79*, 608.
3. Kieseler, J. Object condensation: One-stage grid-free multi-object reconstruction in physics detectors, graph and image data. *Eur. Phys. J. C* **2020**, *80*, 886.
4. Kortus, T.; Keidel, R.; Gauger, N.; Kieseler, J. Constrained Optimization of Charged Particle Tracking with Multi-Agent Reinforcement Learning. *arXiv* **2025**, arXiv:2501.05113.

5. Reuter, L.; De Pietro, G.; Stefkova, S.; Ferber, T.; Bertacchi, V.; Casarosa, G.; Corona, L.; Ecker, P.; Glazov, A.; Han, Y.; et al. End-to-End Multi-Track Reconstruction using Graph Neural Networks at Belle II. *arXiv* **2024**, arXiv:2411.13596.
6. Aehle, M.; Blühdorn, J.; Sagebaum, M.; Gauger, N. Forward-Mode Automatic Differentiation of Compiled Programs. *arXiv* **2022**, arXiv:2209.01895. [[CrossRef](#)]
7. Aehle, M.; Blühdorn, J.; Sagebaum, M.; Gauger, N. Reverse-Mode Automatic Differentiation of Compiled Programs. *arXiv* **2022**, arXiv:2212.13760.
8. Aehle, M.; Novák, M.; Vassilev, V.; Gauger, N.; Heinrich, L.; Kagan, M.; Lange, D. Optimization Using Pathwise Algorithmic Derivatives of Electromagnetic Shower Simulations. *arXiv* **2024**, arXiv:2405.07944. [[CrossRef](#)]
9. Strong, G.C.; Lagrange, M.; Orio, A.; Bordignon, A.; Bury, F.; Dorigo, T.; Giammanco, A.; Heikal, M.; Kieseler, J.; Lamparth, M.; et al. TomOpt: Differential optimisation for task- and constraint-aware design of particle detectors in the context of muon tomography. *Mach. Learn. Sci. Tech.* **2024**, *5*, 035002.
10. Shirobokov, S.; Belavin, V.; Kagan, M.; Ustyuzhanin, A.; Baydin, A. Black-Box Optimization with Local Generative Surrogates. In Proceedings of the Advances in Neural Information Processing Systems, Online, 6–12 December 2020; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, p. 146502.
11. Favaro, L.; Ore, A.; Schweitzer, S.P.; Plehn, T. CaloDREAM—Detector Response Emulation via Attentive flow Matching. *arXiv* **2024**, arXiv:2405.09629. [[CrossRef](#)]
12. Mikuni, V.; Nachman, B. CaloScore v2: Single-shot Calorimeter Shower Simulation with Diffusion Models. *J. Instrum.* **2024**, *19*, P02001.
13. Amram, O.; Pedro, K. Denoising diffusion models with geometry adaptation for high fidelity calorimeter simulation. *arXiv* **2023**, arXiv:2308.03876. [[CrossRef](#)]
14. Hashemi, B.; Krause, C. Deep Generative Models for Detector Signature Simulation: A Taxonomic Review. *Rev. Phys.* **2024**, *12*, 100092.
15. Krause, C.; Giannelli, M.F.; Kasieczka, G.; Nachman, B.; Salamani, D.; Shih, D.; Zaborowska, A.; Amram, O.; Borrás, K.; Buckley, M.R.; et al. CaloChallenge 2022: A Community Challenge for Fast Calorimeter Simulation. *arXiv* **2024**, arXiv:2410.21611.
16. Qu, H.; Gouskos, L. Jet tagging via particle clouds. *Phys. Rev. D* **2020**, *101*, 056019. [[CrossRef](#)]
17. Dreyer, F.; Qu, H. Jet tagging in the Lund plane with graph networks. *J. High Energ. Phys.* **2020**, *2021*, 52. [[CrossRef](#)]
18. The ATLAS Collaboration. *Transformer Neural Networks for Identifying Boosted Higgs Bosons Decaying into $b\bar{b}$ and $c\bar{c}$ in ATLAS*; Technical Report; CERN: Geneva, Switzerland, 2023.
19. Kasieczka, G. Boosted Top Tagging Method Overview. *arXiv* **2018**, arXiv:1801.04180.
20. The CMS Collaboration. Particle-flow reconstruction and global event description with the CMS detector. *JINST* **2017**, *12*, P10003.
21. The ATLAS Collaboration. Jet reconstruction and performance using particle flow with the ATLAS Detector. Jet reconstruction and performance using particle flow with the ATLAS Detector. *Eur. Phys. J. C* **2017**, *77*, 466.
22. The CMS Collaboration. *The Phase-2 Upgrade of the CMS Endcap Calorimeter*; Technical Report CERN-LHCC-2017-023. CMS-TDR-019; CERN: Geneva, Switzerland, 2017.
23. Agostinelli, S.; Allison, J.; Amako, K.A.; Apostolakis, J.; Araujo, H.; Arce, P.; Asai, M.; Axen, D.; Banerjee, S.; Barrand, G.J.N.I.; et al. GEANT4-A Simulation Toolkit. *Nucl. Instrum. Meth. A* **2003**, *506*, 250–303. [[CrossRef](#)]
24. Allison, J.; Amako, K.; Apostolakis, J.E.A.; Araujo, H.A.A.H.; Dubois, P.A.; Asai, M.A.A.M.; Barrand, G.A.B.G.; Capra, R.A.C.R.; Chauvie, S.A.C.S.; Chytrcek, R.A.C.R.; et al. Geant4 developments and applications. *IEEE Trans. Nucl. Sci.* **2006**, *53*, 270–278. [[CrossRef](#)]
25. Allison, J.; Amako, K.; Apostolakis, J.; Arce, P.; Asai, M.; Aso, T.; Bagli, E.; Bagulya, A.; Banerjee, S.; Barrand, G.J.N.I.; et al. Recent developments in Geant4. *Nucl. Instrum. Meth. A* **2016**, *835*, 186–225. [[CrossRef](#)]
26. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
27. Ho, J.; Jain, A.; Abbeel, P. Denoising Diffusion Probabilistic Models. *arXiv* **2006**, arXiv:2006.11239.
28. Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR, San Diego, CA, USA, 7–9 May 2015.
29. Eliachevitch, M.; Braun, N.; Heidelberg, A.; Eppelt, J.; De Pietro, G.; Cleora Völker, M.; Baur, A.; Metzner, F.; Bauer, M.; Welsch, M.; et al. belle2/b2luigi: V1.1.2. 2025. Available online: <https://zenodo.org/records/14710805> (accessed on 3 February 2025).
30. Paszke, A. PyTorch: An imperative style, high-performance deep learning library. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019.
31. The HTCondor Team. HTCondor. Available online: <https://zenodo.org/records/14238973> (accessed on 3 February 2025).
32. Jakob, W.; Schreiner, H.; Rhineland, J.; Moldovan, D.; Smirnov, I.; Gokaslan, A.; Jadoul, Y.; Huebl, A.; Staletic, B.; Izmailov S. pybind/pybind11: Version 2.13.6. Available online: <https://zenodo.org/records/13761364> (accessed on 3 February 2025).

33. Brun, R.; Rademakers, F.; Canal, P.; Naumann, A.; Couet, O.; Moneta, L.; Vassilev, V.; Linev, S.; Piparo, D.; Ganis, G. root-project/root: V6.18/02. Available online: <https://zenodo.org/records/3895860> (accessed on 3 February 2025).
34. Particle Data Group. Review of Particle Physics. *Prog. Theor. Exp. Phys.* **2022**, 2022, 083C01. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.