



PAPER • OPEN ACCESS

A differentiable surrogate model for the generation of radio pulses from in-ice neutrino interactions

To cite this article: Philipp Pilar *et al* 2026 *Mach. Learn.: Sci. Technol.* **7** 025017

View the [article online](#) for updates and enhancements.

You may also like

- [Scalable data-driven basis selection for linear machine learning interatomic potentials](#)
Tina Torabi, Matthias Miltzer, Michael P Friedlander et al.
- [Interpretable visualizations of data spaces for classification problems](#)
Christian Jorgensen, Arthur Y Lin, Rhushil Vasavada et al.
- [Molecular representations of quantum circuits for quantum machine learning](#)
Elham Torabian and Roman V Krems



PAPER

OPEN ACCESS

RECEIVED

12 September 2025

REVISED

27 November 2025

ACCEPTED FOR PUBLICATION

23 February 2026

PUBLISHED

4 March 2026

Original content from this work may be used under the terms of the [Creative Commons Attribution 4.0 licence](#).

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.



A differentiable surrogate model for the generation of radio pulses from in-ice neutrino interactions

Philipp Pilar^{1,*} , Martin Ravn², Christian Glaser^{2,3} and Niklas Wahlström¹¹ Department of Information Technology, Uppsala University, Uppsala, Sweden² Department of Physics and Astronomy, Uppsala University, Uppsala, Sweden³ Department of Physics, TU Dortmund University, Dortmund, Germany

* Author to whom any correspondence should be addressed.

E-mail: philipp.pilar@it.uu.se

Keywords: surrogate model, differentiable programming, radio pulses, physics-informed machine learning, modularized architecture, diffusion model, in-ice neutrino interactions

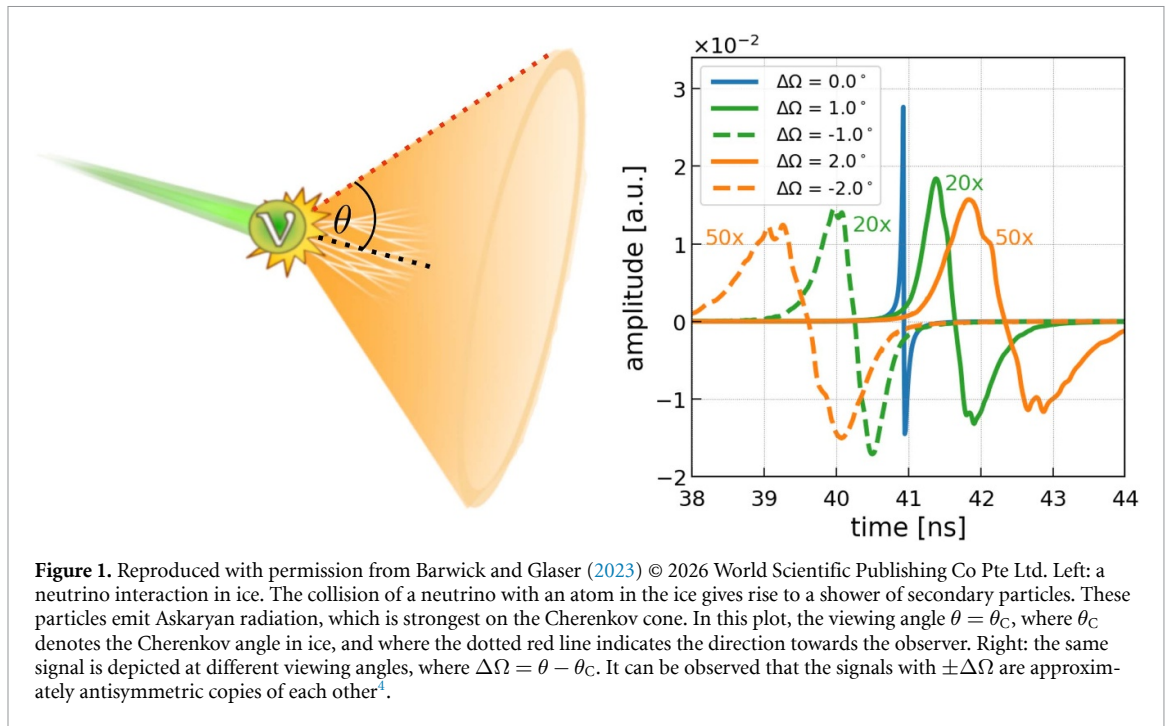
Abstract

The planned IceCube-Gen2 radio neutrino detector at the South Pole will enhance the detection of cosmic ultra-high-energy neutrinos. It is crucial to utilize the available time until construction to optimize the detector design. A fully differentiable pipeline, from signal generation to detector response, would allow for the application of gradient descent techniques to explore the parameter space of the detector. In our work, we focus on the aspect of signal generation, and propose a modularized deep learning architecture to generate radio signals from in-ice neutrino interactions conditioned on the shower energy and viewing angle. The model is capable of generating differentiable signals with amplitudes spanning multiple orders of magnitude, as well as consistently producing signals corresponding to the same underlying event for different viewing angles. The modularized approach ensures physical consistency of the samples and leads to advantageous computational properties when using the model as part of a bigger optimization pipeline.

1. Introduction

The detection of high-energy neutrinos opens up a unique way of perceiving the depths of the cosmos (Mészáros *et al* 2019). Due to their close-to-vanishing interaction with matter, they can traverse vast distances essentially undisturbed. Hence, as messengers, they carry the potential to unambiguously identify the location of cosmological events. However, their weak interaction also makes them extraordinarily difficult to detect, and the instrumentation of huge volumes is required to obtain acceptable detection rates. A cost-efficient way to achieve this is to instrument vast ice shelves in Greenland or Antarctica with an array of compact radio detector stations (Barwick and Glaser 2023). The big volume of detector material, combined with the high transparency of ice, allows for the occasional detection of high-energy neutrinos.

We focus our work on the radio detector of the IceCube-Gen2 project (Aartsen *et al* 2021, IceCube-Gen2 Collaboration 2023), which is currently in development for construction at the South Pole and whose design can still be optimized. The potential improvement is large. In Glaser *et al* (2023), under optimistic assumptions, it was estimated that an improved station layout, together with additional trigger improvements, can lead to an up to three-times better detector at negligible additional costs. To maximize the efficacy of the detector design, the precise locations and orientations of the radio antennas need to be optimized. The differentiable programming paradigm (Baydin *et al* 2018, Blondel and Roulet 2024) constitutes an intriguing option to address this problem. Constructing a fully differentiable pipeline, from neutrino event to detector response, would allow for the use of gradient descent methods for optimization. In this work, we consider the aspect of radio pulse generation and design a differentiable surrogate model to replace (parts of) the existing Monte Carlo simulations.



The radio pulses originate from neutrino interactions in ice (Barwick and Glaser 2023). A cascade of secondary particles forms a particle shower with time-varying excess negative charge, which gives rise to Askaryan radiation (see section 2.1 for details). A schematic of the process is depicted in the left part of figure 1. The radiation is strongest when observed on the Cherenkov cone with viewing angle $\theta = \theta_C$, which is 56° in deep ice. In the right plot of figure 1, signals from the same event at different viewing angles are depicted. The signals exhibit an approximate antisymmetry around θ_C , and the amplitude decreases quickly when θ diverges from it. The neutrino energy E determines the underlying signal shapes that are possible. While the distance between the event and the observer also affects the signal, we do not consider this aspect in this work and instead use a constant distance of 1 km.

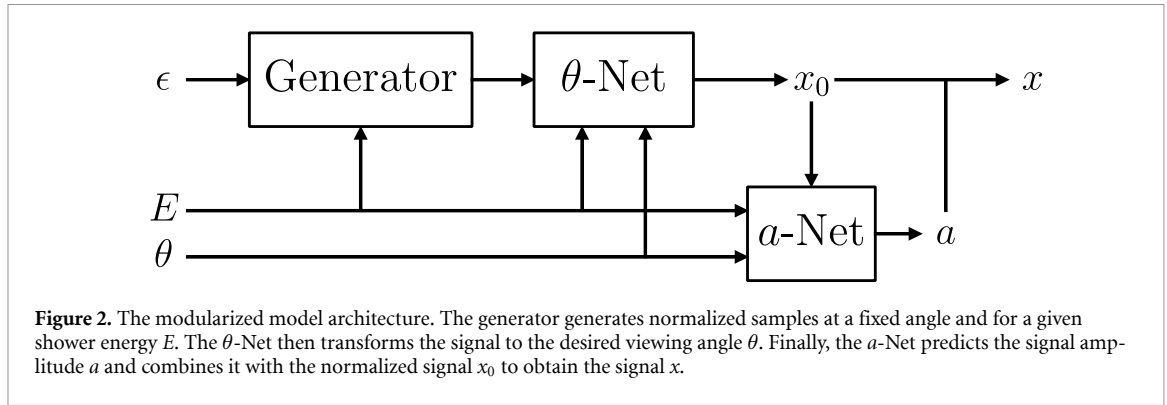
Additional requirements are placed on the generative surrogate model due to its usage as a component in the detector optimization pipeline. It needs to generate signals together with their respective amplitudes, which range over many orders of magnitude. Furthermore, it needs to be able to generate multiple correlated signals, corresponding to the same neutrino interaction as observed by antennas at different positions. To fulfill these requirements, we develop a modularized model architecture, which is illustrated in figure 2.

A generative model (the generator) producing normalized signals lies at the core of this architecture. The type of model used for the generator can be freely chosen, as long as realistic samples are generated. Even non-differentiable simulations, such as the Monte Carlo simulation itself, are possible; it is one of the advantages of the modularized architecture to enable (approximate) differentiability for the event parameters also in such cases. In most of our experiments, we utilized the MC simulation as the generator. To illustrate certain aspects of the model architecture, we also employed a diffusion model (Ho et al 2020) as an alternative. The normalized signals are subsequently transformed to the different viewing angles of interest and combined with their final amplitudes. Two additional networks are employed for these steps.

2. Background

In this section, we give background information on the dataset and the physical processes involved in giving rise to the radio signals under consideration.

⁴ Radio Detection of High Energy Neutrinos in Ice, Steven Barwick and Christian Glaser, Edited by Floyd W. Stecker, The Encyclopedia of Cosmology, Vol 2 Chapter 6, Copyright @ 2023 World Scientific Publishing.



2.1. Askaryan radiation

Askaryan radiation was first described in Askar'yan (1965). Neutrinos can occasionally interact with atoms in the ice and cause a cascade of secondary particles, called a particle shower. As the particle shower develops, electrons outnumber positrons, and a negative charge excess builds up along its trajectory, resulting in the generation of radio emission. Since the particle shower is traveling faster than the speed of light in ice, the emitted radiation exhibits Cherenkov-like characteristics and is strongest on the Cherenkov cone, with viewing angle $\theta = \theta_C$, where $\theta_C = 55.82^\circ$ is the Cherenkov angle in deep ice. See figure 1 for an illustration of this process.

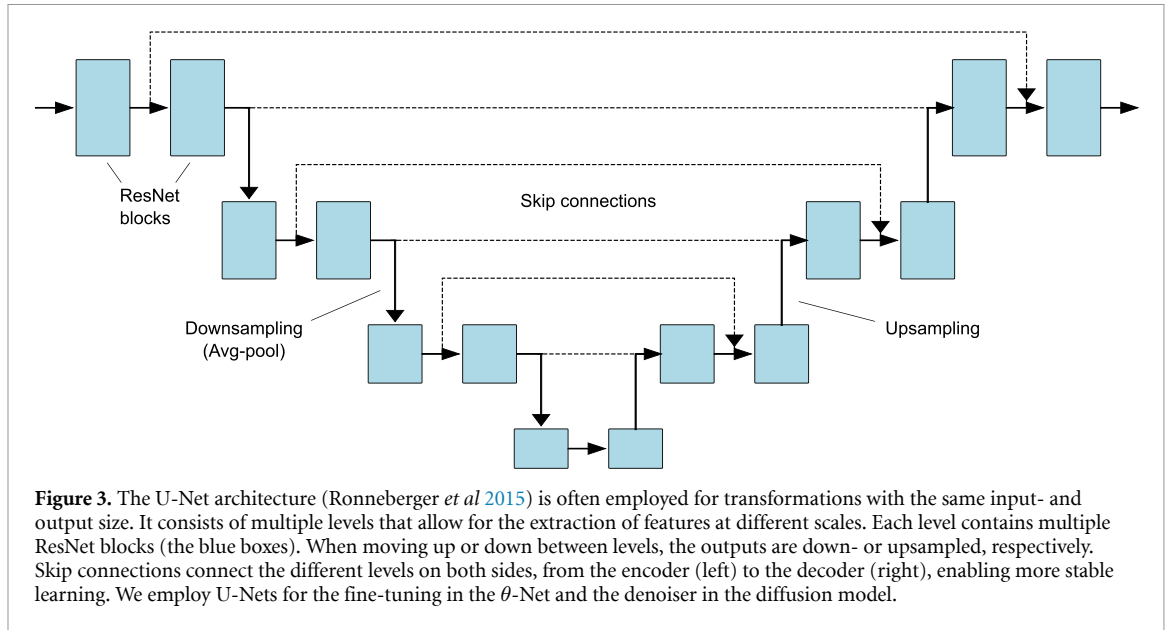
While the development of the particle shower is stochastic, its main characteristics depend on the neutrino energy and the type of interaction. The higher the neutrino energy, the more secondary particles can be created, and the larger the negative charge excess becomes. In hadronic showers, the first few interactions are hadronic, before electrons and positrons are created via pair production. Here, the charge-excess profile exhibits a single peak and does not differ significantly between different energies, apart from the amplitude. In electromagnetic showers, the neutrino interaction immediately gives rise to an electron, and subsequently electron-positron pairs. Due to the Landau-Pomeranchuk-Migdal effect (Landau and Pomeranchuk 1953), electrons can travel further between interactions at higher energies. Multiple spatially separated showers can emerge, which result in charge-excess profiles with multiple peaks, and, in turn, more complicated radio signals. A more detailed description of the processes giving rise to Askaryan radiation can be found in Barwick and Glaser (2023).

2.2. Dataset

The data under consideration in this project consists of 1D waveforms generated with the NuRadioMC library (Glaser *et al* 2020) using the ARZ2020 model (Alvarez-Muniz *et al* 2011). We consider electron-neutrino charged current interactions. Energies lie in the range $\log_{10} E(\text{eV}) \in [15, 19]$, and viewing angles in $\theta \in [35.82^\circ, 75.57^\circ]$. Due to the aforementioned stochasticity in the shower development, different shower profiles are possible at the same energy. In each $0.1 \log_{10} E(\text{eV})$ bin, the NuRadioMC library provides 10 different shower profiles. The viewing angle θ can be freely chosen in the stated range (for values outside that range, the signal amplitude becomes vanishingly small) and we generated data on a grid covering the range stated above with 160 values. Hence, our dataset consists of 65 600 samples, in total, but there are only 410 truly independent signals. Exemplary samples are depicted, e.g. in figures 1 and 8.

For each of these samples, the following data is available: the waveform x on a time grid of size 1024 corresponding to a sampling rate of 5×10^{-10} (s), together with the event energy E , and the viewing angle θ . We denote the amplitude as a , and the normalized sample with amplitude unity as x_0 . When used as inputs to the neural networks, the values of E and θ are normalized to lie in the range $[0, 1]$ via $E \rightarrow \frac{\log_{10} E[\text{eV}] - 15}{4}$ and $\theta \rightarrow \frac{\theta[^\circ] - 35.82}{39.75}$.

When training our models, we split the data into train and test data. To make the test data truly independent, when a signal was added to the test data, all the corresponding signals with the same energy and shower profile at different angles were added to the test data as well. In total, 90% of the data was used for training, and 10% was held out for testing. This proportion holds at all energies.



3. Model architecture⁵

A schematic of the architecture of our generative model is given in figure 2. Machine learning models generally perform best when inputs and outputs lie in a narrow range of values (Huang *et al* 2023). For the E and θ values, this can be straightforwardly achieved by normalizing the data, as described in the previous section. The large range in which the amplitudes of the radio pulses lie (compare section 2.2), however, poses a challenge. Due to the zero-crossings of the samples, taking the logarithm as we did for E to reduce the range of the numerical values before normalizing is not possible. For this reason, we have introduced the amplitude net (a -Net) to predict the signal amplitudes based on their shape, as well as the viewing angle θ and the neutrino energy E . This allows us to use normalized data with amplitude unity for each sample when training the different model components.

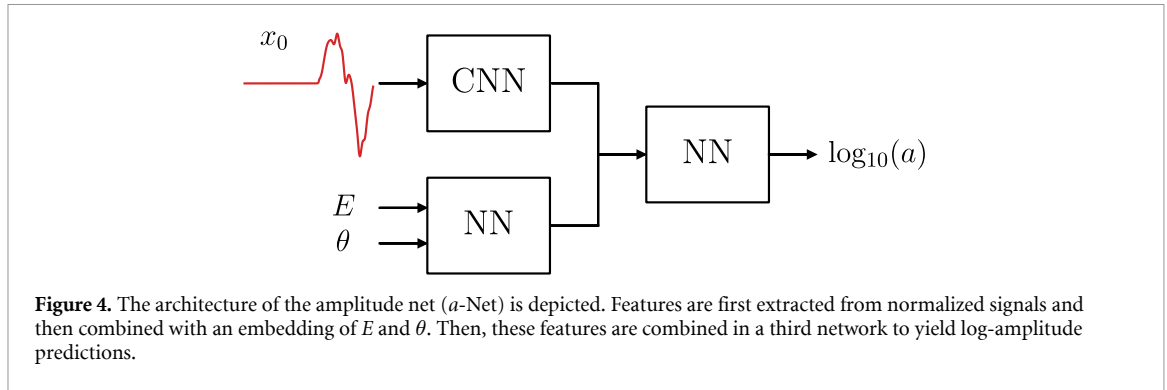
Another requirement for the model is the ability to generate the same signal when viewed from different angles. This is relevant when considering the case where multiple antennas located at different positions detect the same neutrino interaction. While the generative model may automatically learn the correct relationship from the data, this is not certain, and we want to add a stronger inductive bias to ensure correct model behavior. We achieve this by further splitting the generative model into the generator, which now generates normalized signals at a fixed reference angle θ_r , and the θ -Net, which transforms the thus-generated signals to the desired angle.

Apart from ensuring more accurate sample generation, the modularized architecture has additional advantages when viewed from the perspective of a differentiable optimization pipeline. This becomes apparent when considering the computational path that the inputs take in figure 2. The viewing angle θ does not pass through the generator, and the computational graph only needs to be created for the θ -Net and the a -Net when computing gradients. While the generative model still takes the energy as input, the strongest contributions to the energy derivatives are expected to originate from the amplitude. Hence, approximate gradients can be obtained efficiently also for E when neglecting the contribution from the generator. This approximation enables the use of expensive generative models like diffusion models with many denoising steps. It also enables the use of non-differentiable models for the generator, such as Monte Carlo simulations.

3.1. A diffusion model as generator

While existing MC simulations can be used for the generator, we also evaluate the option of using a generative machine learning model in their stead. Denoising diffusion probabilistic models (DDPMs, Ho *et al* (2020)) generate samples from Gaussian noise ϵ by removing the noise stepwise, as outlined in appendix C. The denoiser performing this task is its centerpiece, and the U-Net architecture (Ronneberger *et al* 2015) depicted in figure 3 is usually employed for it. Due to their constituent convolutional layers, U-Nets are well-suited for spatiotemporal data. The different levels of the U-Net enable

⁵ The code for the project is available on GitHub: https://github.com/ppilar/IC_Surrogate.



the extraction of features at different scales, with skip connections helping to retain information from extracted features at higher levels of the U-Net. In our implementation, we also chose the U-Net architecture and based it on the variant that comes with the python package by Lucidrains (2020).

Each of the U-Net layers contains multiple ResNet blocks. An average-pooling layer is then utilized to downsample to the layer below the current one; in the upward pass, an upsampling layer is employed instead. Skip connections connect to the opposite side of the U-Net, both from the middle and the end of each U-Net level. The ResNet blocks contain multiple convolutional layers, supplemented with skip connections, as well as a rescaling operation. Scale and shift for the rescaling operation are calculated via a fully-connected layer from the embedding of the diffusion step, which is used to condition on the current step k in the denoising process.

We have chosen 6 levels for the U-Net, with [8, 16, 32, 64, 128, 256] channels in the convolutional layers, respectively. The Adam optimizer with learning rate 8×10^{-5} was employed, and the network was trained until convergence for 5×10^5 iterations with batch size 128. For the diffusion and denoising process, 500 steps were chosen. When sampling signals, the effective number of steps taken can be reduced (Salimans and Ho 2022), and we use 50 steps when sampling from the trained DDPM.

In our model, we have modified the step embedding to also condition on the neutrino energy E and, optionally, the viewing angle θ . This is done by first using three different nets to find separate embeddings for each of these quantities, followed by another network to combine the embeddings. The individual embedding networks each consist of a positional encoding layer of dimension 16 followed by two linear layers with outputs of size 32. The GELU activation function is applied between the linear layers. Then, the embeddings are combined via another neural network consisting of two linear layers with the GELU activation function in between.

3.2. The *a*-Net

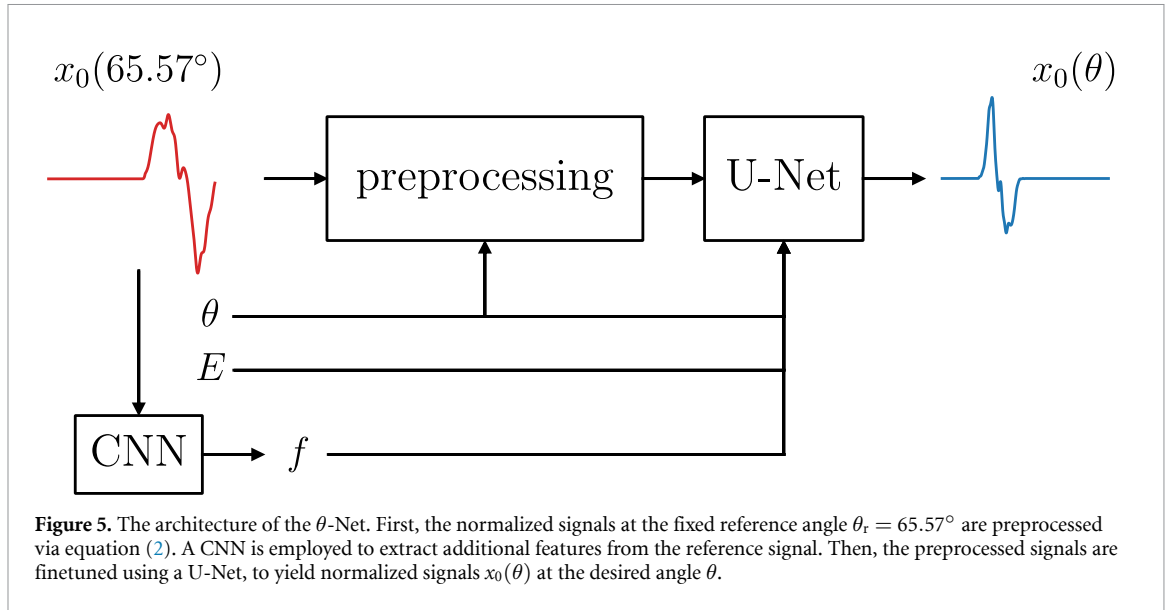
Figure 4 depicts the architecture of the *a*-Net, which consists of three separate subcomponents. First, the normalized waveforms x_0 are fed through a convolutional network to extract features. These features are then concatenated with a learned embedding of the neutrino energy E and the viewing angle θ . Together, they are subsequently passed through another fully-connected neural network that predicts the logarithm of the amplitude.

The convolutional network consists of 3 layers with 32, 64, and 128 channels, respectively. The corresponding kernels have size 3, stride 1 and zero-padding enabled. After each convolutional layer, an average-pooling layer with kernel size 2 and stride 2 is applied, effectively halving the size of the layer output. After the final convolutional layer, the output is flattened into an array of size 16384. The embedding network contains 3 layers of width 40. The third network contains 3 layers of width 256, 128, and 1, respectively. The ReLU activation function is applied after each layer, apart from average-pooling layers.

The mean squared error loss,

$$\mathcal{L}_a = \frac{1}{N_{\text{bs}}} \sum_{i=1}^{N_{\text{bs}}} (\log_{10}(\hat{a}_i) - \log_{10}(a_i))^2, \quad (1)$$

was used for training the *a*-Net. The Adam optimizer was used with learning rate 0.0001 and batch size $N_{\text{bs}} = 128$. The training lasted 150000 iterations, until convergence.



3.3. The θ -Net

The θ -Net transforms reference signals at the fixed angle $\theta_r = 65.57^\circ$ to the desired viewing angle θ . The precise choice for the angle θ_r is essentially arbitrary, the main condition being that all relevant details of the signals must be resolved well at this angle (compare, e.g. figure 8). The architecture of the θ -Net is depicted in figure 5, and consists of two main blocks, one for preprocessing and one for fine-tuning the signals. Furthermore, a small convolutional network is employed to extract additional features from the reference signals. In appendix A, an ablation study is conducted to show the impact of these components.

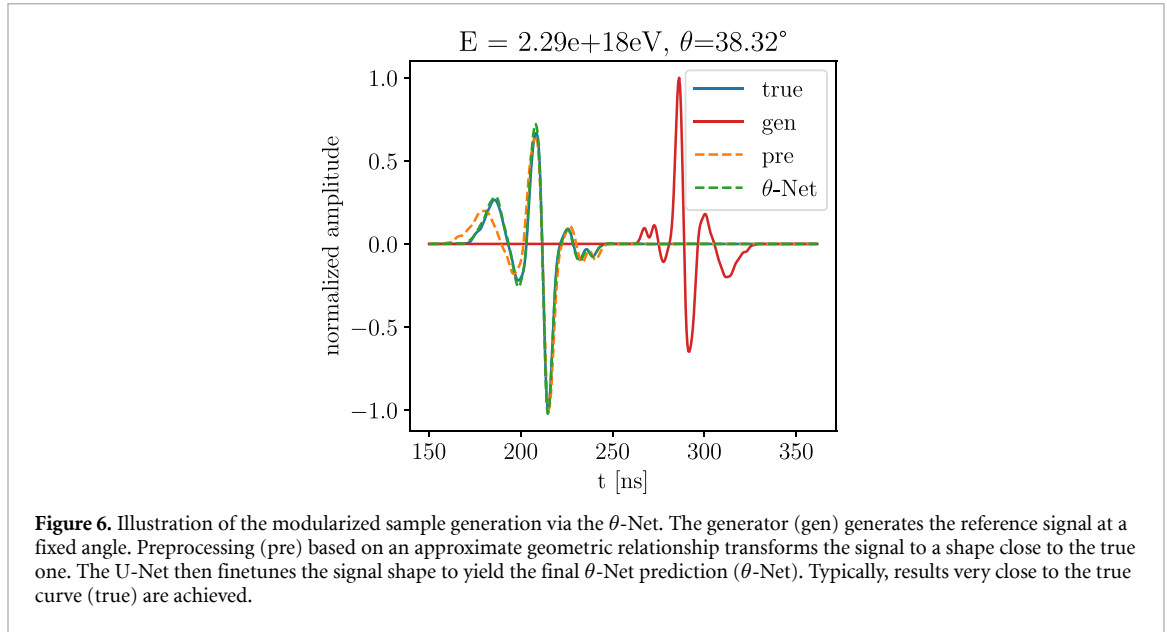
The main purpose of the preprocessing block is to simplify the task of the subsequent U-Net. The normalized reference signals are transformed by using a rough geometric relationship obeyed by the signals. As can be seen in figure 8, when starting at a large viewing angle, the signals first get squeezed when approaching the Cherenkov angle $\theta_C = 55.82^\circ$, then flipped upside down after crossing it, and finally unsqueezed again when moving on to lower values θ . The following expression captures this behavior well:

$$\tilde{x}_0(t, \theta) = \begin{cases} x_0(f(t - t_0) + t_0, \theta_r) & \text{if } \theta \geq \theta_C, \\ 1 - x_0(0.75f(t_0 - t) + t_0, \theta_r) & \text{if } \theta < \theta_C, \end{cases} \quad (2)$$

where $t_0 = 256\text{ns}$ is the middle point of the chosen time grid (approximately, where signals with $\theta = \theta_C$ have their maximum), and where $f = \max(|\frac{\theta - \theta_C}{\theta_r - \theta_C}|, 0.1)$ is an empirically determined squeezing factor. The minimum value of 0.1 prevents signals from being squeezed to almost zero width close to θ_C , thereby losing the information inherent in the sample. The additional factor 0.75 for $\theta < \theta_C$ reflects the fact that the signals appear to be squeezed more strongly in this regime. Both factors were determined empirically.

The convolutional network is employed to extract features from the reference signals, which may get lost when transforming the signal. These features are then used together with E and θ to condition the U-Net. Since the reference signals are zero on the left half of the time axis, only the right half is passed as input to the CNN. The CNN consists of three convolutional layers, followed by a flattening operation and 3 fully-connected layers. The convolutional layers have [16, 32, 64] channels, respectively, a kernel size of 4, and a stride of 4. After the flattening operation, the outputs contain 512 features. The linear layers then map to [100, 50, 32] hidden units, respectively. The ReLU activation function is applied after each of these layers, except the last one.

Subsequently, the U-Net architecture depicted in figure 3 is employed for fine-tuning the signals. The U-Net is very similar to the one used in the diffusion model, which is a slightly modified version of the one from Lucidrains (2020). For the θ -Net, we chose to use 5 U-Net levels with [8, 16, 32, 64, 128] channels, respectively. The ResNet layers are conditioned on the parameters E and θ in the same way as before in the diffusion model. In contrast to the diffusion model, the U-Net is only applied once here. Since there is no denoising step involved in this case, a constant value is used instead. Figure 6 illustrates the signal generation via the θ -Net.



The loss function

$$\mathcal{L}_\theta = \frac{1}{N_{\text{bs}}} \sum_{i=1}^{N_{\text{bs}}} \int (\hat{x}_{0i}(t, \theta_i) - x_{0i}(t, \theta_i))^2 dt \quad (3)$$

was employed for the θ -Net. The Adam optimizer was used with learning rate 0.0003 and batch size 128. The model was trained for 300000 iterations until convergence.

4. Results

In this section, we investigate the performance of the individual model components. We verify that the modularized architecture accurately reproduces the physical dependence of the signal shape on the viewing angle, together with realistic amplitudes. In a simple optimization experiment, we demonstrate the usability of the model architecture in future optimization pipelines. Additional results for the DDPM as generator are given in appendix C.

4.1. Amplitude predictions

In figure 7, the distributions of amplitudes in the training and test data are depicted, together with the error distributions of the a -Net predictions. To evaluate the accuracy of the a -Net, we chose the distance of true and predicted values of a in log-space, i.e. $\Delta \log_{10}(a) = \log_{10}(a) - \log_{10}(\hat{a})$. The 0.95-quantile of the absolute error in log space, $|\Delta \log_{10}(a)|$, lies at 0.042 for the test data. This means that in 95% of the cases, the error in the a -Net predictions of a is below roughly 10% of the corresponding signal amplitudes. The full evaluation metrics are shown in table 1.

4.2. θ -dependence

In figure 8, the same signal from the test data when viewed at different angles is depicted, once after the preprocessing step and once when using the full θ -Net. It can be observed that the preprocessing succeeds at getting the signal reasonably close to the final signal, but is not yet accurate enough without the fine-tuning via the U-Net.

In figure 9, an equivalent plot is given for the case where a DDPM is used as the generator, and where no separate θ -Net is employed. It is apparent that the DDPM alone does not automatically learn the correct θ -dependence. It roughly reproduces the locations of the peaks, but fails to accurately reproduce the finer features of the signals. This shows that the θ -Net is also required to ensure sample consistency when using a generator other than the MC simulation, in addition to bestowing properties advantageous for the model differentiability.

We quantify the accuracy of the θ -Net via the normalized absolute difference between the true and predicted curve, integrated over the time domain, $\Delta x_0 = \frac{\int_{-\infty}^{\infty} |x_0(t, \theta) - \hat{x}_0(t, \theta)| dt}{\int_{-\infty}^{\infty} |x_0(t, \theta)| dt}$, as well as the difference along the t -axis between the positions of the true and predicted maximum of the signal, $\Delta \max x_0 =$

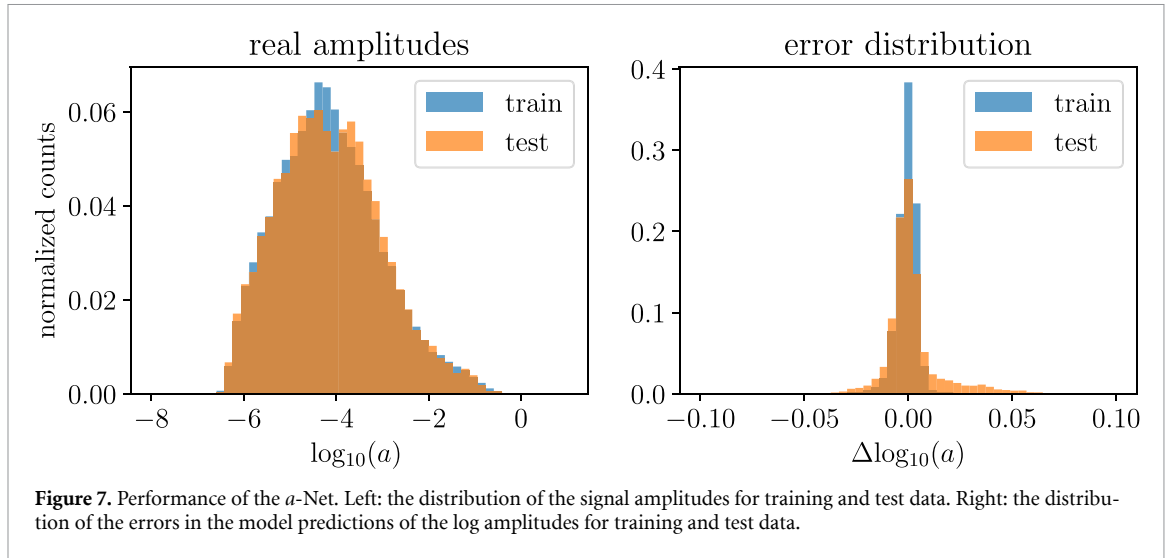


Table 1. Evaluation metrics for the a -Net. For $\Delta \log_{10}(a)$, the mean plus-or-minus one standard deviation is given. For $|\Delta \log_{10}(a)|$, the 0.95-quantile is given.

| a -Net | $\Delta \log_{10}(a)$ | $ \Delta \log_{10}(a) $ |
|----------|-----------------------|-------------------------|
| Train | 0.002 ± 0.004 | 0.008 |
| Test | 0.007 ± 0.017 | 0.042 |

$\operatorname{argmax}_t \hat{x}_0(t, \theta) - \operatorname{argmax}_t x_0(t, \theta)$. We also consider the analogous metrics in Fourier space, where \tilde{x}_0 denotes the Fourier transform of x_0 : $\Delta \tilde{x}_0 = \frac{\int_{-\infty}^{\infty} |\tilde{x}_0(f, \theta) - \hat{\tilde{x}}_0(f, \theta)| df}{\int_{-\infty}^{\infty} |\tilde{x}_0(f, \theta)| df}$, and $\Delta \max \tilde{x}_0 = \operatorname{argmax}_f \hat{\tilde{x}}_0(f, \theta) - \operatorname{argmax}_f \tilde{x}_0(f, \theta)$.

The distributions of these quantities are depicted in figure 10, for training and test data. It is apparent that the model performs almost as well on the test data as on the training data, showing that there is no overfit. The full evaluation metrics are shown in table 2. The 0.95-quantile of 0.006 for Δx_0 shows that most signals are reproduced with an error of less than 1%. Furthermore, the peaks of the signals are at the correct spot, or deviate by only one bin in the time domain in about 95% of the cases, as evidenced by the results for $\Delta \max x_0$. Hence, the overall performance of the θ -Net is excellent.

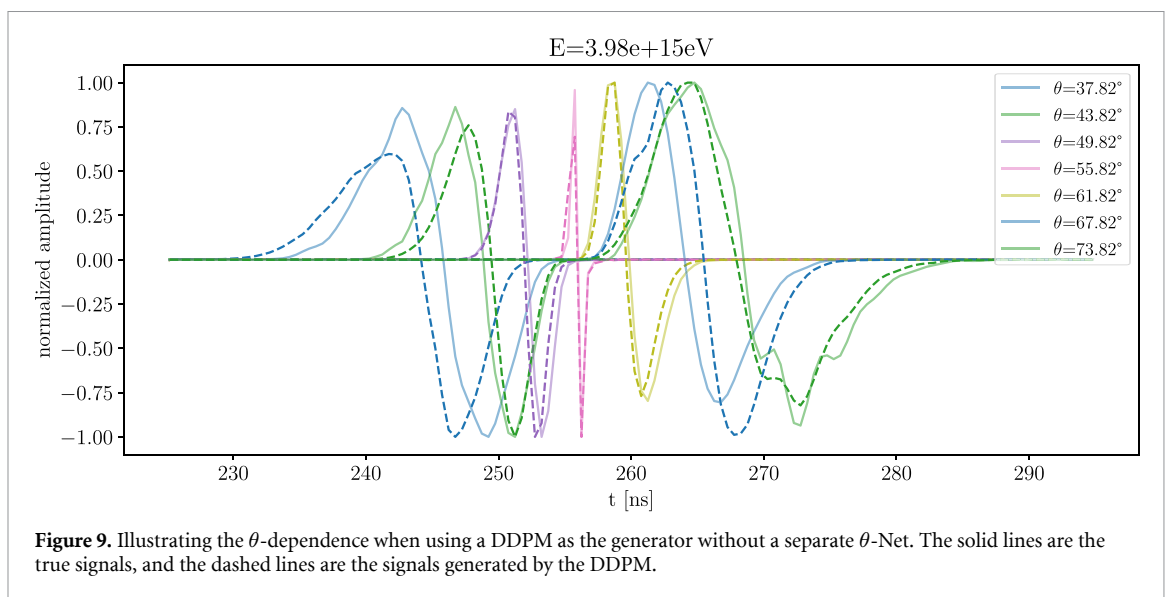
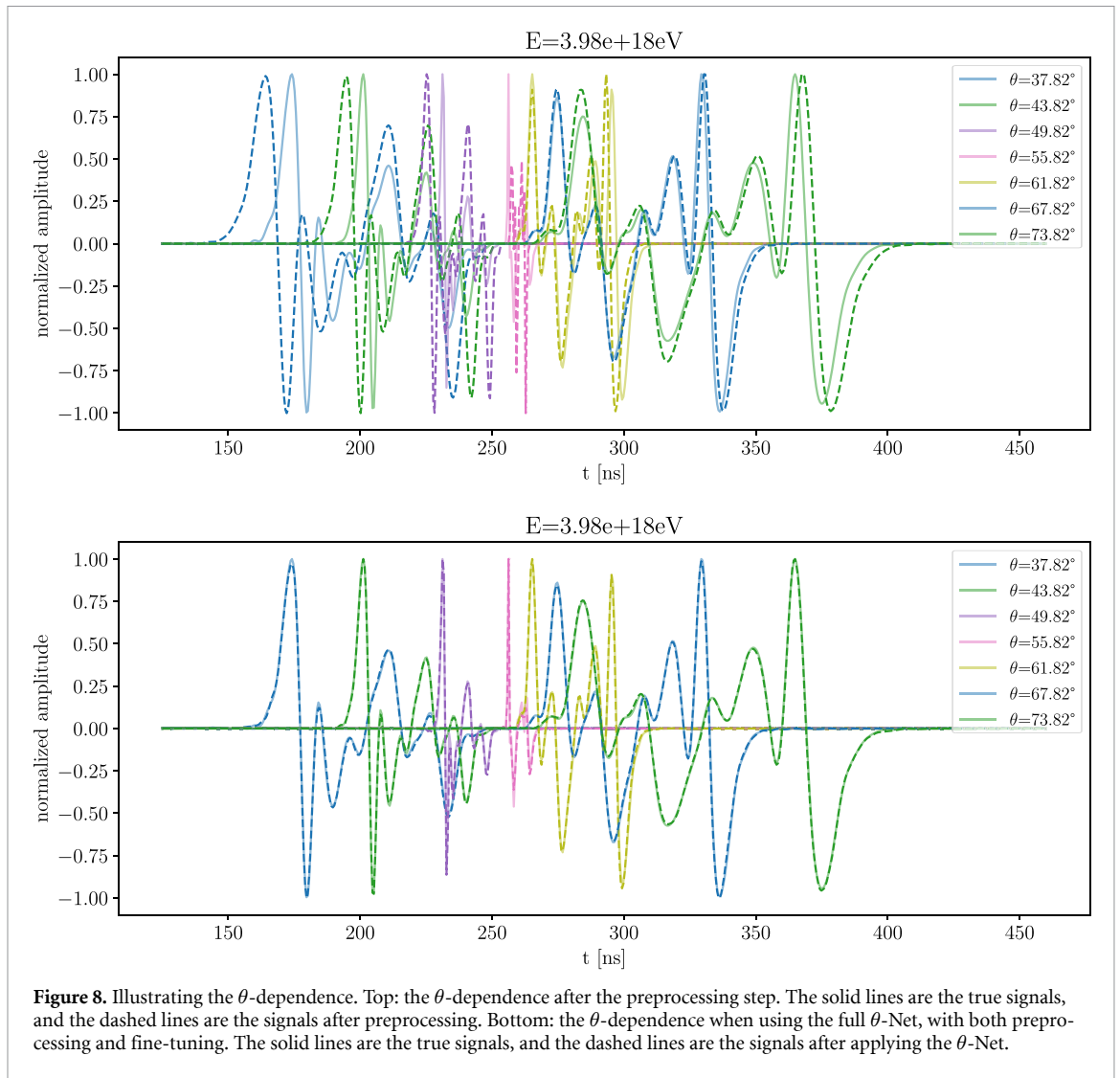
4.3. Optimization experiment

To evaluate the suitability of our model for usage as a constituent in a larger optimization pipeline, we also conducted a simple optimization experiment. Here, the objective of the optimization is to minimize the area under the normalized signals, by optimizing θ . For this purpose, we employ the loss function

$$A(\theta) = \frac{1}{N_{\text{bs}}} \sum_{i=1}^{N_{\text{bs}}} \int_{-\infty}^{\infty} |\hat{x}_{i0}(t, \theta)| dt, \quad (4)$$

where N_{bs} denotes the batch size. The results are depicted in figure 11. We conduct this experiment for an architecture with θ -Net acting on the training data (i.e. using the MC simulation as generator), and an architecture with DDPM as generator, but without θ -Net.

During the optimization, the viewing angle goes towards values close to the Cherenkov angle, as expected (compare figure 8). When considering the pure DDPM variant without θ -Net, the model can take considerably longer to converge at high energies. Another advantage of the θ -Net architecture is that the VRAM requirements during optimization are significantly lower, with about 0.8GB, as opposed to around 3.6GB when using the DDPM (with $N_{\text{bs}} = 10$). The execution speed is also much faster, with 0.06 s per iteration for the variant with θ -Net vs 2.13 s per iteration for the variant with DDPM. When combining the DDPM with θ -Net, the VRAM advantage would persist since the computational graph would not need to be created for the DDPM, as explained in section 3. The accuracy of the optimization results is limited by the resolution of the time grid. The Adam optimizer with learning rate 0.01 was used for this experiment, together with a learning rate scheduler halving the learning rate every 150 iterations.



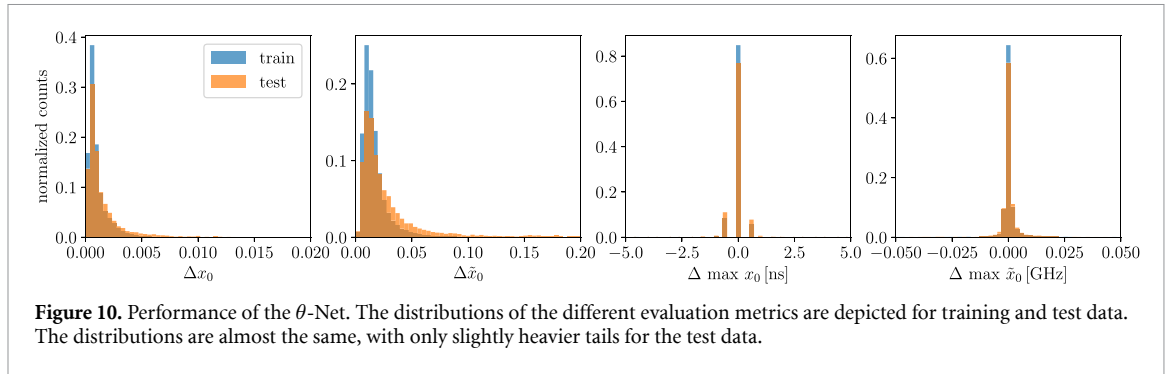
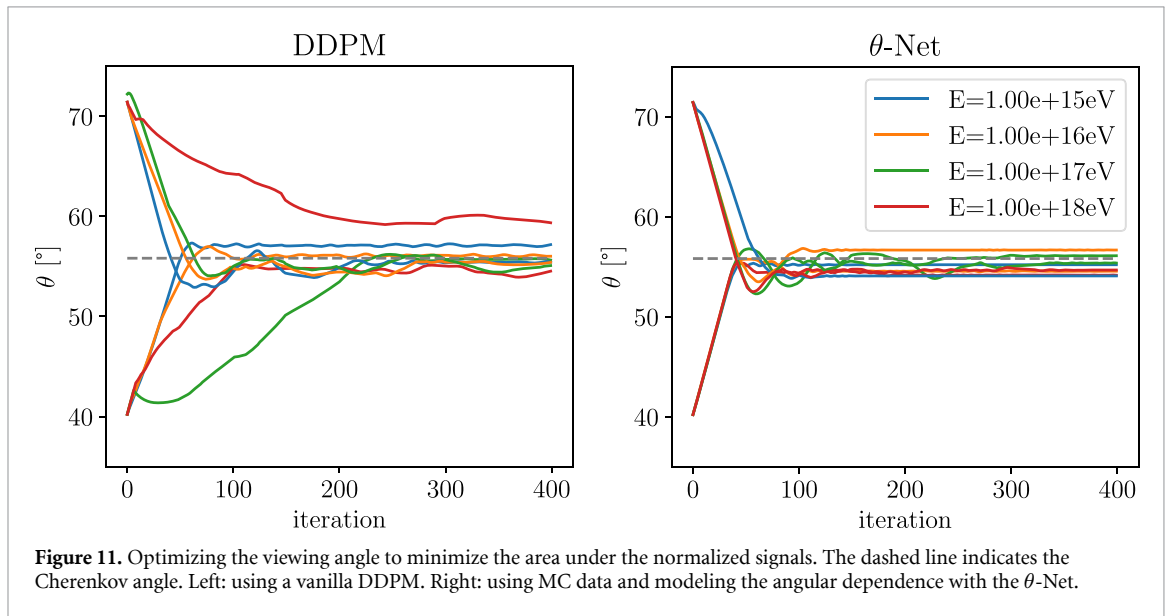


Table 2. Evaluation metrics for the θ -Net. For positive quantities, the 0.95-quantile is given. Otherwise, the 0.05- and 0.95-quantiles are given.

| θ -Net | Δx_0 | $\Delta \tilde{x}_0$ | $\Delta \max x_0$ (ns) | $\Delta \max \tilde{x}_0$ (GHz) |
|---------------|--------------|----------------------|------------------------|---------------------------------|
| Train | 0.003 | 0.039 | [−0.500, 0.500] | [−0.008, 0.008] |
| Test | 0.006 | 0.106 | [−0.500, 0.500] | [−0.023, 0.010] |



5. Conclusions and outlook

We have presented a differentiable surrogate model for the generation of radio pulses from in-ice neutrino interactions. The model consists of three main components: a generative model, a network to adjust the viewing angle of the signal, and a network to predict the signal amplitude. This modularized model architecture allows for realistic and consistent sample generation across a wide range of energies and viewing angles. In particular, it allows for the creation of related signals as measured by different antennas that stem from the same underlying event. Moreover, the model remains differentiable even when using a non-differentiable generator, such as Monte Carlo simulations.

We have evaluated the different model components separately and demonstrated good model performance. As of now, the best choice is to use existing Monte Carlo simulations for the generator. While the diffusion model succeeds at reproducing the training data, more data would be required to enable the generation of truly novel samples. Both the θ -Net and the a -Net are highly accurate at their respective tasks. Most importantly, the modularized model architecture has proven effective at enabling differentiability of the signals with regard to the parameters of interest in an accurate and hardware-efficient manner.

In the future, we aim to utilize the model developed in this work in a larger optimization pipeline to optimize the design of the radio detector in IceCube-Gen2. We have already taken first steps in this direction for a simplified detector setup (Ravn *et al* 2025). Similar model architectures may also prove effective for enabling differentiability in unrelated applications, where the effect of the parameters of interest on the signal shape can be separated from the fundamental generative process. A separate amplitude net may be useful for generative models in general, which deal with data that covers an extensive range of amplitudes.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://github.com/ppilar/IC_Surrogate (Pilar *et al* 2025).

Funding statement

The work is financially supported by the Swedish Research Council (VR) via the project *Physics-informed machine learning* (registration number: 2021-04 321). CG is supported by the European Union (ERC, NuRadioOpt, 10 111 6890). This work used resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06 725.

Conflict of interest statement

There are no conflicts of interest.

Ethics statement

There are no ethical concerns.

Appendix A. Ablation study for the θ -Net

In this appendix, we conduct an ablation study to demonstrate the impact that the different components of the θ -Net depicted in figure 5 have. The evaluation metrics introduced in section 4.2 are given in table 3 for the θ -Net with different components missing.

The preprocessing without any further fine-tuning performs clearly worse than the full θ -Net. When omitting the preprocessing, the resulting model also performs poorly. This demonstrates that the U-Net benefits from being given a simpler task to learn. A likely reason for this is that long-range dependencies, which are relevant when transforming signals from large to small angles, can be difficult to learn for convolutional architectures. Omitting the CNN for extracting additional features only slightly degrades the performance. Its inclusion is therefore not essential, but still beneficial.

Table 3. Ablation study for the θ -Net. For positive evaluation metrics, the 0.95-quantile is given. Otherwise, the 0.05- and 0.95-quantiles are given. Bold font highlights best performance.

| | Δx_0 | $\Delta \tilde{x}_0$ | $\Delta \max x_0$ (ns) | $\Delta \max \tilde{x}_0$ (GHz) |
|--------------------|--------------|----------------------|----------------------------------|---------------------------------|
| Only preprocessing | 0.020 | 0.237 | [−1.500, 2.000] | [−0.045, 0.113] |
| No preprocessing | 0.028 | 0.176 | [−1.500, 1.000] | [−0.039, 0.020] |
| No CNN | 0.010 | 0.107 | [− 0.500 , 1.000] | [− 0.016 , 0.012] |
| Full θ -Net | 0.006 | 0.106 | [− 0.500 , 0.500] | [−0.023, 0.010] |

Appendix B. Hyperparameter selection

The final choices for the details of the model architecture stated in section 3 were guided by the requirements of computational efficiency and accuracy. In general, the number of network layers and their respective numbers of channels and widths were chosen as small as possible without negatively impacting the validation accuracy. For the model components involving a U-Net in particular, the number of levels, the depth of the ResNet blocks, as well as the number of channels were tuned. Omitting the self-attention layer, which is sometimes applied after the final ResNet block of each level in U-Net architectures led to significant improvements in both the network size and the model performance. The presented model outperformed variants of the architecture with both roughly half as many and twice as many parameters.

The training hyperparameters were chosen to ensure effective and efficient training. We manually tuned the learning rates to train the different model components fast, without negatively affecting their validation accuracy. Increasing the learning rates by an order of magnitude over the chosen values typically led to decreased performance. The number of iterations was chosen to stop the training after convergence.

Appendix C. Background on diffusion models

Denoising diffusion probabilistic models (DDPMs) (Ho *et al* 2020) have gained a large amount of popularity since they were first introduced. They have been successfully used in a large variety of applications (Yang *et al* 2023), most prominently in computer vision (Croitoru *et al* 2023). A major advantage of DDPMs over other generative models, such as generative adversarial models, is that the training is more robust due to their likelihood-based objective function. They are also less prone to suffering from issues such as mode collapse.

When training DDPMs, two processes are involved. In the forward or diffusion process, noise is added step-by-step to samples x_0 from the training data:

$$q(x_k | x_{k-1}) = \mathcal{N}(x_k; \sqrt{1 - \beta_k} x_{k-1}, \beta_k \mathbf{I}), \quad (5)$$

$$q(x_k | x_0) = \mathcal{N}(x_k; \sqrt{\bar{\alpha}_k} x_0, (1 - \bar{\alpha}_k) \mathbf{I}), \quad (6)$$

where the β_k define the noise schedule, x_k denotes the sample after adding noise k times, and where $\alpha_k = 1 - \beta_k$, $\bar{\alpha}_k = \prod_{s=1}^k \alpha_s$. In the reverse or denoising process, the diffusion model reverts this process and removes the noise step-by-step:

$$p_\theta(x_{k-1} | x_k) = \mathcal{N}(x_{k-1}; \mu_\theta(x_k, k), \beta_k \mathbf{I}), \quad (7)$$

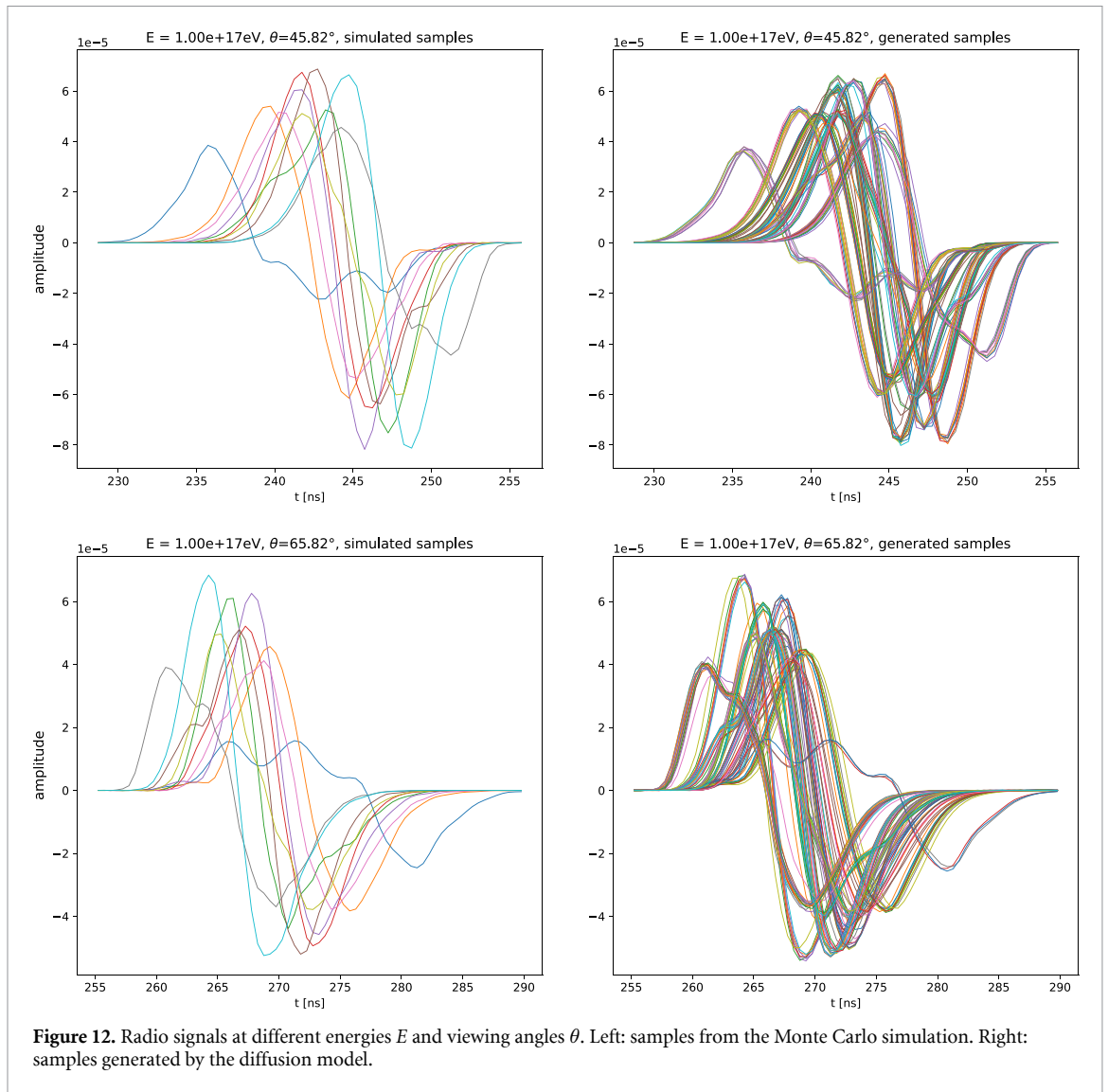
where $x_k = \sqrt{\bar{\alpha}_k} x_0 + \sqrt{1 - \bar{\alpha}_k} \epsilon$ and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, and where μ_θ denotes the predicted mean. In practice, however, it is more effective to let the diffusion model predict the noise directly. The model is then trained with the loss function

$$\mathcal{L} = \mathbb{E}_{x_0, \epsilon, k} \left[\left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_k} x_0 + \sqrt{1 - \bar{\alpha}_k} \epsilon, k) \right\|^2 \right], \quad (8)$$

where ϵ_θ denotes the noise prediction of the denoiser, which constitutes the main component of the DDPM that needs to be learned. Once fully trained, the DDPM can generate samples from random Gaussian noise.

C.1. Generated samples

In figure 12, generated samples from the diffusion model at fixed E and θ values are compared to corresponding samples from the training data. It is apparent that the DDPM accurately reproduces the training data. This shows that the DDPM is well-suited for modeling this type of data. However, while the generated samples exhibit a certain variation around the training samples, the DDPM does not yet produce truly novel signals. This is presumably due to the limited size of the training data, which contains



only 410 fundamentally different samples. There are efforts underway to generate a larger dataset, which may enable the DDPM to go beyond merely reproducing the training data in the future.

In figure 13, the distribution of the existing data and the generated data is depicted around different energies E and viewing angles θ . It is apparent that, on the whole, the distributions agree well, indicating that the DDPM reproduces the entire training data and does not suffer from mode collapse.

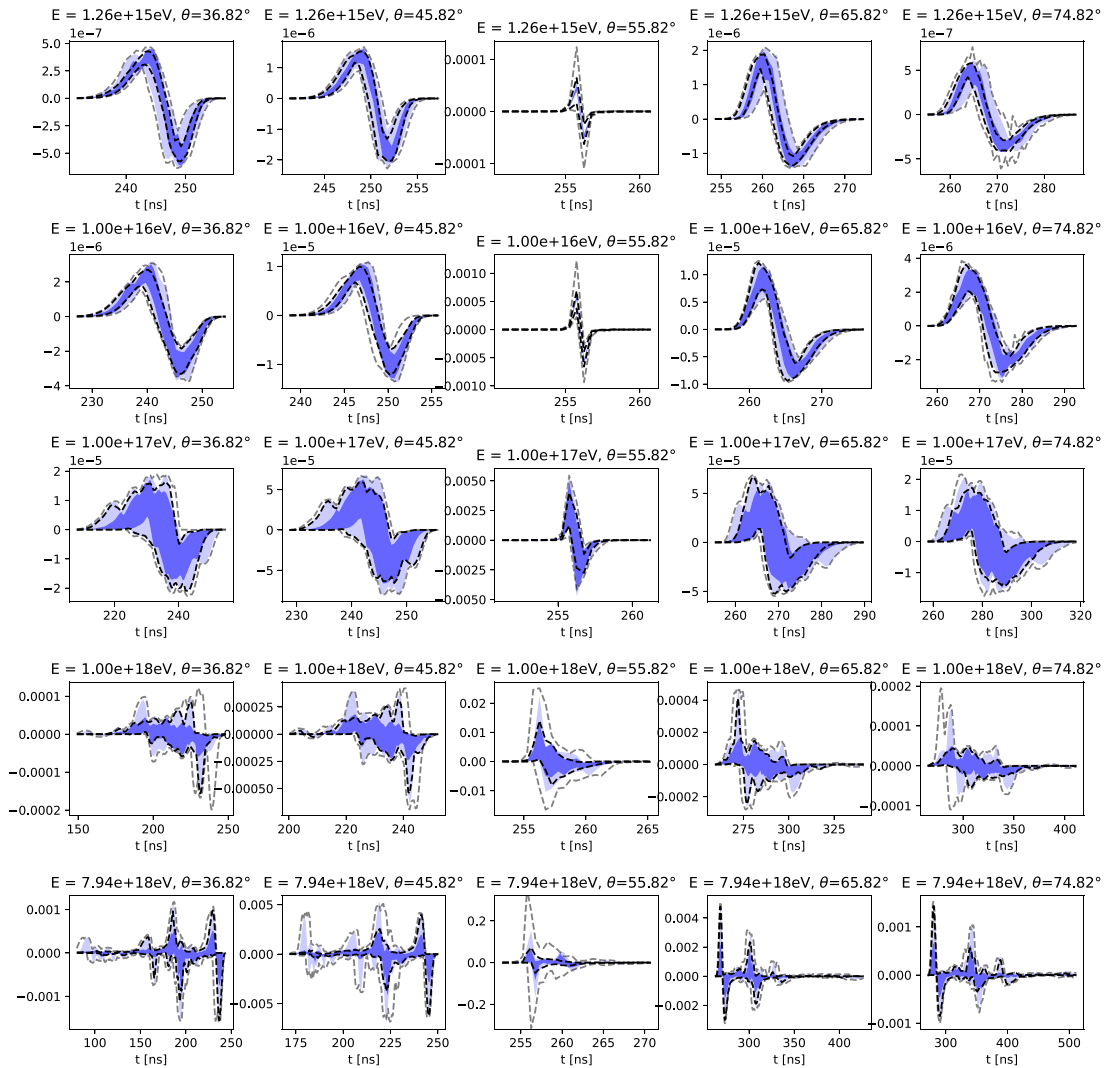


Figure 13. The distribution of real and generated signals, conditional on the energy E and angle θ (within $\pm 0.025\Delta_{E/\theta}$ of the corresponding values in logspace). The values on the y-axis correspond to the signal amplitude. The shaded blue area corresponds to the real distribution, with the light blue area corresponding to the total range of signal values at that point and the dark blue area to the range from the 0.1-quantile to the 0.9-quantile. The dashed lines depict the corresponding quantiles for the generated data.

ORCID iDs

Philipp Pilar  0000-0002-9935-3716

Christian Glaser  0000-0001-5998-2553

References

- Aartsen M G *et al* 2021 IceCube-Gen2: the window to the extreme Universe *J. Phys. G: Nucl. Part. Phys.* **48** 060501
- Alvarez-Muniz J, Romero-Wolf A and Zas E 2011 Practical and accurate calculations of Askaryan radiation *Phys. Rev. D* **84** 103003
- Askaryan G A 1965 Coherent radio emission from cosmic showers in air and in dense media *Sov. J. Exp. Theor. Phys.* **21** 658
- Barwick S and Glaser C 2023 Radio detection of high energy neutrinos in ice *Encyclopedia Cosmol.* **2** 237–302
- Baydin A G, Pearlmutter B A, Radul A A and Siskind J M 2018 Automatic differentiation in machine learning: a survey *J. Mach. Learn. Res.* **18** 1–43
- Blondel M and Roulet V 2024 The elements of differentiable programming (arXiv:2403.14606)
- Croitoru F-A, Hondru V, Ionescu R T and Shah M 2023 Diffusion models in vision: a survey *IEEE Trans. Pattern Anal. Mach. Intell.* **45** 10850–69
- Glaser C *et al* 2020 NuRadioMC: simulating the radio emission of neutrinos from interaction to detector *Eur. Phys. J. C* **80** 1–35
- Glaser C, Coleman A and Glusenkamp T 2023 NuRadioOpt: optimization of radio detectors of ultra-high energy neutrinos through deep learning and differential programming *PoS ICRC2023* 1114
- Ho J, Jain A and Abbeel P 2020 Denoising diffusion probabilistic models *Advances in Neural Information Processing Systems* vol 33 pp 6840–51
- Huang L, Qin J, Zhou Y, Zhu F, Liu Li and Shao L 2023 Normalization techniques in training DNNs: methodology, analysis and application *IEEE Trans. Pattern Anal. Mach. Intell.* **45** 10173–96

- IceCube-Gen2 Collaboration *IceCube-Gen2 Technical Design Report* IceCube-Gen2 website 2023
- Landau L D and Pomeranchuk I Y 1953 Limits of applicability of the theory of Bremsstrahlung electrons and pair production at high-energies *Dokl. Akad. Nauk Ser. Fiz.* **92** 535–6
- Langgård Ravn M, Pilar P, Glaser C, Wahlström N and Glüsenkamp T 2025 Optimization pipeline for in-ice radio neutrino detectors *PoS MODE2025* 031
- Lucidrains 2020 Denoising diffusion Pytorch (available at: <https://github.com/lucidrains/denoising-diffusion-pytorch>)
- Mészáros P, Fox D B, Hanna C and Murase K 2019 Multi-messenger astrophysics *Nat. Rev. Phys.* **1** 585–99
- Pilar P, Ravn M, Glaser C and Wahlström N 2025 Ice Cube Surrogate *GitHub* (available at: https://github.com/ppilar/IC_Surrogate)
- Ronneberger O, Fischer P and Brox T 2015 U-Net: convolutional networks for biomedical image segmentation *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18* (Springer) pp 234–41
- Salimans T and Ho J 2022 Progressive distillation for fast sampling of diffusion models *Int. Conf. on Learning Representations*
- Yang L, Zhang Z, Song Y, Hong S, Xu R, Zhao Y, Zhang W, Cui B and Yang M-H 2023 Diffusion models: a comprehensive survey of methods and applications *ACM Comput. Surv.* **56** 1–39