PAPER • OPEN ACCESS

Implicit quantile networks for emulation in jet physics

To cite this article: Braden Kronheim *et al* 2024 *Mach. Learn.: Sci. Technol.* **5** 045073

View the article online for updates and enhancements.

MACHINE
LEARNING
Science and Technology

PAPER

Implicit quantile networks for emulation in jet physics

Braden Kronheim[1,*] ⓘ, Ali Al Kadhim[2] ⓘ, Michelle P Kuchera[3,4] ⓘ, Harrison B Prosper[2]
and Raghuram Ramanujan[4] ⓘ

[1] Department of Physics, University of Maryland, College Park, MD 20742, United States of America
[2] Department of Physics, Florida State University, Tallahassee, FL 32306, United States of America
[3] Department of Physics, Davidson College, Davidson, NC 28035, United States of America
[4] Department of Mathematics and Computer Science, Davidson College, Davidson, NC 28035, United States of America
[*] Author to whom any correspondence should be addressed.

E-mail: bkronhei@umd.edu

Abstract

The ability to model and sample from conditional densities is important in many physics applications. Implicit quantile networks (IQN) have been successfully applied to this task in domains outside physics. In this work, we illustrate the potential of IQNs as components of emulators using the simulation of jets as an example. Specifically, we use an IQN to map jets described by their 4-momenta at the generation level to jets at the event reconstruction level. The conditional densities emulated by our model closely match those generated by `Delphes`, while also enabling faster jet simulation.

1. Introduction

High-fidelity simulators are of critical importance in many fields of science as they provide the connection between theoretical models and potential (and actual) observations. In high-energy physics, the simulation pipeline comprises an event generator which encodes the theoretical predictions of particle interactions, a detector simulator, and event reconstruction which transforms low-level data to objects that model final state particles.

Viewed abstractly, a high-energy physics simulator is a procedure that maps an event x comprising one set of particles to another event y comprising a different set of particles. A collection of such multi-level events is a point cloud approximation of expressions of the form

$$o(y) = \int p(y|x)\, u(x)\, \mathrm{d}x, \tag{1}$$

each of which maps a density $u(x)$, typically unobserved, to an observed density $o(y)$ via a conditional density $p(y|x)$, which, in general, is multi-dimensional. In some contexts, $p(y|x)$ is called a *response function*. The mapping $u(x) \to o(y)$ is referred to as *folding*, while the inverse mapping $o(y) \to u(x)$ is referred to as *unfolding*[5].

An example of equation (1) is the mapping of a jet, a collimated collection of particles (see, for example, [1]), prior to its interaction with a particle detector to the jet observed in the detector. Traditionally, the interaction of a jet of particles with a detector is modeled using a Monte Carlo method based on the widely used `GEANT4` toolkit [2]. `GEANT4` provides high-fidelity simulations of particle interactions with matter, but comes with a high computational cost. For this reason, experimental collaborations have devoted considerable effort to building fast simulators (see, for example, [3, 4]) in which the slower parts of the full (`GEANT4`-based) simulator are replaced by hand-coded parameterized approximations to the conditional densities $p(y|x)$.

---

[5] Unlike folding, unfolding is an ill-posed problem due to the information loss from folding. In order to render an unfolding procedure well-posed, information must be injected into the procedure via some form of regularization.

Unfortunately, hand coding of these densities for fast simulators is an error-prone and labor-intensive task, which must be repeated every time the detector changes. Recently, several groups have sought to sidestep this bottleneck by replacing components of the simulation pipeline with machine learning models that emulate the replaced components [5–8]. A significant benefit of models that permit extremely fast sampling from the conditional densities is that they provide a straightforward way to fold theoretical predictions so that these can be compared directly with unfolded generator level observations. This is of particular interest to those who wish to use published statistical models and likelihoods [9], which typically use unfolded observations.

Fast folding is of interest even with unfolded data. In measurements of QCD differential observables such as jet differential cross sections in nuclear and particle physics, the observables are unfolded to remove detector effects (see, for example, [10, 11]). However, the QCD predictions are made at the parton level. Therefore, it is necessary to map the predicted differential observables at the parton level to the level of the unfolded observables. In this context, the ratio $o(\boldsymbol{x})/u(\boldsymbol{x})$ is referred to as the non-perturbative correction to the theoretical prediction.

In this paper, we demonstrate the effectiveness of implicit quantile networks (IQN) in modeling $p(\boldsymbol{y}|\boldsymbol{x})$. Given a large sample of paired simulated jets—one at the particle *generation* level (i.e. before the particles enter the particle detectors) and the other at the *event reconstruction* level (i.e. jets recorded in the detectors)—we show that the *jet response function* $p(\boldsymbol{y}|\boldsymbol{x})$ can be accurately modeled with an IQN. We shall refer to the sampling operation of an IQN for nuclear and particle physics observables as *stochastic folding*.

The paper is organized as follows. In section 2 we briefly describe related work and the work that inspired the current paper. This is followed in section 3 by a description of the IQN model. Section 4 describes the datasets used, the training of our models, and the results. We discuss these results in section 5 and give our conclusions in section 6.

2. Related work

The use of neural networks to approximate $p(\boldsymbol{y}|\boldsymbol{x})$ was first studied by White [12] and Taylor [13], an approach that is of considerable interest in many fields including high-energy physics; see, for example, [14–17] and the references therein. Generative adversarial networks (GAN) [18–21], normalizing flows [22–26] and combinations of flows, GANs, and autoencoders [27] for modeling conditional densities have been explored in a variety of applications. Recent work has also proposed methods for calibrating predictive distributions [28]. Deep neural networks have been trained using the average *quantile loss* (see section 3) to model detector response and to perform jet reconstruction [29, 30]. In the latter works, however, researchers focused on accurately modeling specific quantiles of interest. In contrast, we model the full quantile function using deep neural networks by extending the network architecture to include the quantile τ as one of the inputs—an approach first described by Ostrovski *et al* [31]. A preliminary version of this work was presented at the machine learning and the physical sciences (ML4PS) workshop co-located with the neural information processing systems conference in 2021 [32].

3. Implicit IQNs

Given a one-dimensional conditional cumulative distribution function $\tau = F(y|\boldsymbol{x})$, the *quantile function* is its inverse $y = F^{-1}(\tau|\boldsymbol{x})$. The quantile function thus maps a given cumulative probability (quantile) τ to the value y of the random variable Y such that $\Pr[Y \leqslant y] = \tau$. Given a training dataset comprising samples y conditional on $\boldsymbol{x}$, the *quantile regression* problem is to construct an estimator $f(\tau, \boldsymbol{x}; \boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$ that approximates F^{-1}. This problem can be cast as an optimization problem [33] in which the quantile loss,

$$\mathcal{L}(f, y) = \begin{cases} \tau\,(y - f(\tau, \mathbf{x}; \boldsymbol{\theta})) & y \geqslant f(\tau, \mathbf{x}; \boldsymbol{\theta}) \\ (1 - \tau)\,(f(\tau, \mathbf{x}; \boldsymbol{\theta}) - y) & y < f(\tau, \mathbf{x}; \boldsymbol{\theta}) \end{cases}, \tag{2}$$

averaged over a set of training examples $(\boldsymbol{x}, y)$ is minimized. An *implicit IQN* is a deep neural network that operates as the estimator f.

This method has many applications including the stochastic folding of one set of particles to another. We illustrate the simplicity and efficacy of the approach by applying it to the stochastic folding of jets at the generation level ('gen-jets') to jets at the level of event reconstruction ('reco-jets'). Each gen-jet is defined by the 4-momentum $\boldsymbol{x} = (p_T, \eta, \phi, m)$, where p_T, η, ϕ, m are the jet transverse momentum, pseudo-rapidity, azimuthal angle, and mass, respectively. A reco-jet is analogously defined by $\boldsymbol{y} = (p_T', \eta', \phi', m')$. Following Ostrovski *et al*, we can write the 4-dimensional conditional density $p(\boldsymbol{y}|\boldsymbol{x})$ as

$$p(\boldsymbol{y}|\boldsymbol{x}) = p(p_T'|\boldsymbol{x})\, p(\eta'|\boldsymbol{x},p_T')\, p(\phi'|\boldsymbol{x},p_T',\eta')$$
$$\times p(m'|\boldsymbol{x},p_T',\eta',\phi'). \tag{3}$$

Each of the four densities in equation (3) can be modeled with an independent IQN, as shown in section 4.3. In the following, we refer to this model as `IQNx4`. However, it is also possible to model equation (3) with a *single* IQN by a judicious choice of inputs. Every training example $(p_T,\eta,\phi,m) \to (p_T',\eta',\phi',m')$ is 'unrolled' into the four training examples

$$(p_T,\eta,\phi,m,1,0,0,0,0,0,0) \to p_T',$$
$$(p_T,\eta,\phi,m,0,1,0,0,p_T',0,0) \to \eta',$$
$$(p_T,\eta,\phi,m,0,0,1,0,p_T',\eta',0) \to \phi',$$
$$(p_T,\eta,\phi,m,0,0,0,1,p_T',\eta',\phi') \to m', \tag{4}$$

where the left-hand sides of equation (4) are the possible inputs to the single IQN and the one-hot encoding after p_T,η,ϕ,m in equation (4) specifies which target is associated with the given unrolled example. We call this model `IQNx1`. In addition to the one-hot encoding $\boldsymbol{z}$ and the differing components $\boldsymbol{y}$ of the reconstruction-level 4-momentum from the training sample, the quantile τ is also an input to our model $f(\tau,\boldsymbol{x},\boldsymbol{z},\boldsymbol{y};\boldsymbol{\theta})$. During training, the quantile τ is repeatedly and independently sampled from $U(0,1)$ and an independent value of τ is associated with each unrolled example.

We train a deep neural network on batches of unrolled examples from the training set with randomly sampled τ values. At inference time, the trained model is used autoregressively: the unrolled examples, each with a randomly sampled quantile, are provided in the order shown in equation (4) with the quantities p_T',η',ϕ' now the values *predicted* by the trained model, rather than the values from the training data.

Since the trained model approximates four quantile functions—one quantile function at a time depending on which one-hot encoding is used—the model is an *implicit* approximation of the multi-dimensional conditional density $p(\boldsymbol{y}|\boldsymbol{x})$ in the following sense. The 1-dimensional conditional densities, from which $p(\boldsymbol{y}|\boldsymbol{x})$ is formed using equation (3), can be computed from $p(y^{(n)}|\boldsymbol{x},y^{(1)},\cdots,y^{(n-1)}) = (\partial f_n/\partial\tau)^{-1}$, where f_n is the model specified with the nth one-hot encoding.

One problem that can arise in quantile regression is *quantile crossing*, where the approximation to the quantile function is not monotonic. Prior work has attempted to mitigate this problem by imposing constraints on the model architecture or by optimizing novel loss functions [34, 35]. Tagasovska and Lopez-Paz, however, observed that the problem becomes significantly less pronounced when the full quantile function is approximated [36]. In this paper, we propose using a regularized average loss

$$\mathcal{L}' = \mathcal{L} + \lambda \cdot \mathbb{1}\left[-f'\right]\left(f'\right)^2$$

that further alleviates this problem, where $f' = \partial f/\partial\tau$ and λ is a hyperparameter. By penalizing negative gradients of f with respect to τ, the regularization term favors solutions that are monotonically non-decreasing. This is equivalent to the condition $(f')^{-1} = p(y|\boldsymbol{x}) > 0$.

4. Stochastic folding of Jets

4.1. Dataset generation

As this study is a proof of concept, we limit our investigations to the fast folding of gen-jets to reco-jets. The `Pythia8` [37] event generator (v8.307) is used to simulate proton-proton collisions at 13 TeV at the Large Hadron Collider (LHC)[6], while the `Delphes` [38] detector simulator (v3.5.0) provides an approximate simulation of the interaction of jets with one of the particle detectors at the LHC, namely, the Compact Muon Solenoid (CMS) detector [39]. A particle clustering algorithm, called anti-k_T [40], is used at both the particle generation and event reconstruction levels to cluster particles into jets with a radius parameter $R = 0.5$ using `FastJet` [41].

We simulated 10 million hard QCD jets using the `Pythia8-Delphes` pipeline, of which 1 million were set aside as test data. Of the remaining 9 million jets, we used 8 million as training data, leaving the remaining 1 million jets for model validation and hyperparameter tuning. These jets had a generator level p_T cut of 20 GeV and a reconstruction level cut of 5.0 on the absolute pseudorapidity corresponding to the edge of the CMS Hadronic Forward detector. Both tracker and calorimeter information was used for the particles

---

[6] https://home.cern/science/accelerators/large-hadron-collider.

included in the reconstruction level jets. The complete code to reproduce the results in this paper, released with a GPL3 license, can be found at the linked GitHub repository[7][42].

4.2. Data preprocessing
Prior to training, the simulated 4-momenta are transformed as follows

$$
\begin{aligned}
\mathbb{T}(p_T) &= z(\log p_T), \\
\mathbb{T}(\eta) &= z(\eta), \\
\mathbb{T}(\phi) &= z(\phi), \\
\mathbb{T}(m) &= z(\log(m+2)),
\end{aligned}
\tag{5}
$$

where $\mathbb{T}(.)$ denotes the transformed quantity and $z(.)$ the function that standardizes its argument, i.e. ensures that the transformed quantity has zero mean and unit variance. We take the logarithm of p_T and m prior to standardization to reduce the range of these quantities, which can sometimes vary by orders of magnitude. The replacement of m with $m+2$ in equation (5) avoids potential numerical problems with logarithms of small jet masses. We also transform the quantile τ provided to the IQN as follows

$$
\mathbb{T}(\tau) = 6\tau - 3,
\tag{6}
$$

so that the transformed τ is on roughly the same scale as the other transformed inputs, while taking care to use the original value of τ in the computation of the loss function in equation (2). Finally, the IQN target was chosen to be

$$
z\left(\frac{y_n + 10}{x_n + 10}\right), \qquad n = 1, \cdots, 4,
\tag{7}
$$

where x_n is the nth component of $(\log p_T, \eta, \phi, \log(m+2))$ and y_n the corresponding component of $(\log p'_T, \eta', \phi', \log(m'+2))$. In our experiments, we found this target to be easier to model than $z(y_n)$. The constant 10 in the numerator and denominator of equation (7) ensures that this ratio is always well-defined and positive. These transformations are appropriately inverted before the calculation of any downstream quantities of interest such as the marginal densities.

4.3. Model architecture and training
Each individual network in the `IQNx4` model is a dense, feed-forward neural network. These models are completely independent of each other and can be trained in any order, or in parallel. The values of key hyperparameters—shared by all the networks—are listed in table 1. Each IQN is trained using AMSGrad [43], an optimization algorithm in the stochastic gradient descent family. We implemented a learning schedule, decreasing the learning rate of AMSGrad by a factor of 10 after every 100 epochs of training. When performing this annealing, we found it beneficial to resume training the model from the configuration in which it achieved the lowest validation loss over the past 100 epochs, rather than from its final configuration. In our experiments, we achieved maximal performance with three such learning rate decay steps (i.e. with 400 epochs of overall training). This process took about 24 h using an Intel 14 700KF with no core restrictions and 32G of RAM available. Note that the instantaneous CPU usage is usually around 2 cores and GPU acceleration was not useful for these models given the small network and batch sizes.

As noted in section 3, an alternative approach (`IQNx1`) is to model the 4-dimensional conditional density with a single, autoregressive network. Aside from operating on a different input representation (the unrolled examples presented in equation (4)), the `IQNx1` model is identical to the individual IQNs comprising the `IQNx4` model in every other way, including the network architecture, the training procedure, and duration.

4.4. Results
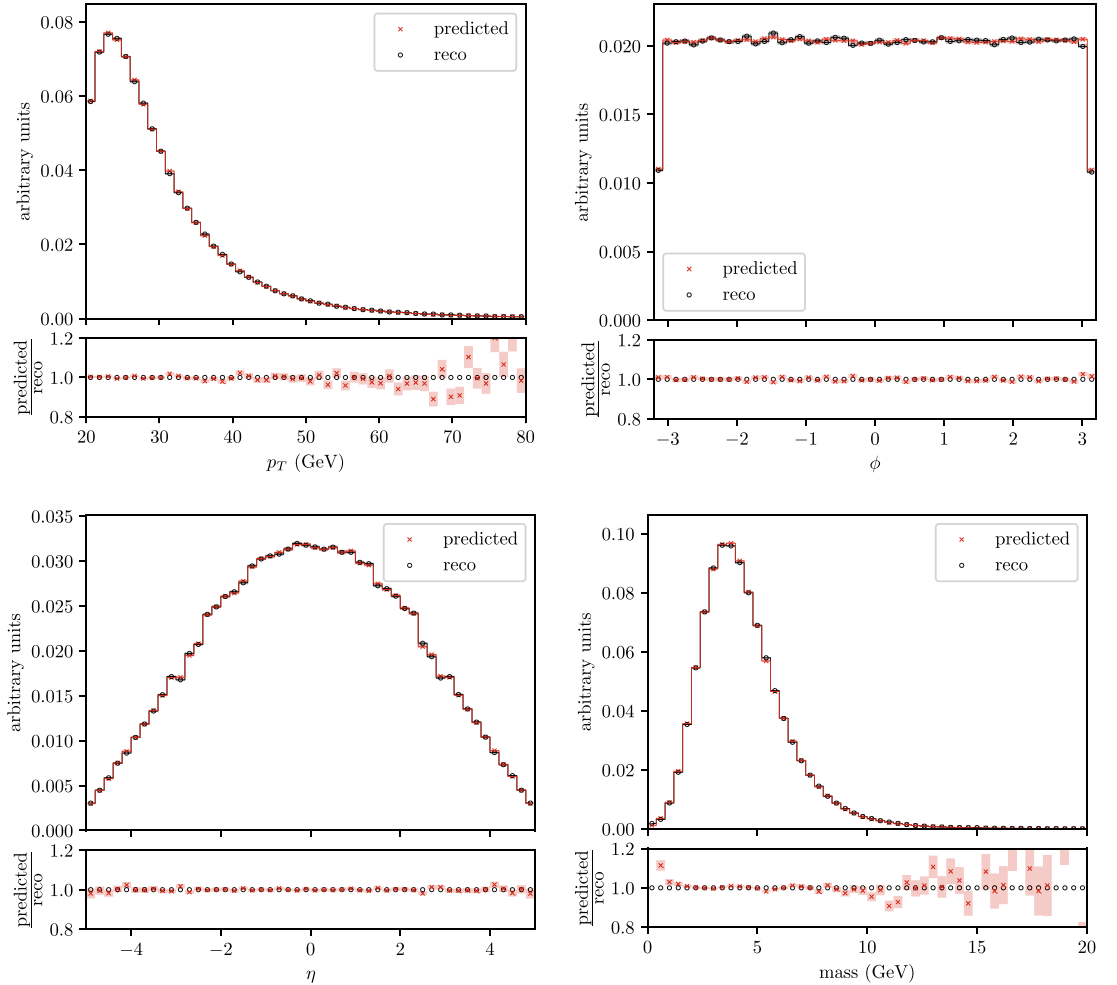We assess the effectiveness of the trained models by comparing the predicted marginal density of each component of the reco-jet 4-momentum (p'_T, η', ϕ', m') with the corresponding reco-jet marginal density from the test set, both integrated over the full gen-jet phase space or over a small subset of the latter.

Using the trained `IQNx4` model, 1000 predictions for the reco-jet 4-momenta are generated for each of the 10^6 gen-jets from the test set. For each reco-jet 4-momentum component, a single prediction is used per gen-jet to construct the reco-jet marginal densities. The reco-jet marginal densities are presented in figure 1 superimposed on the original reco-jet marginal densities computed from the test data. All uncertainties
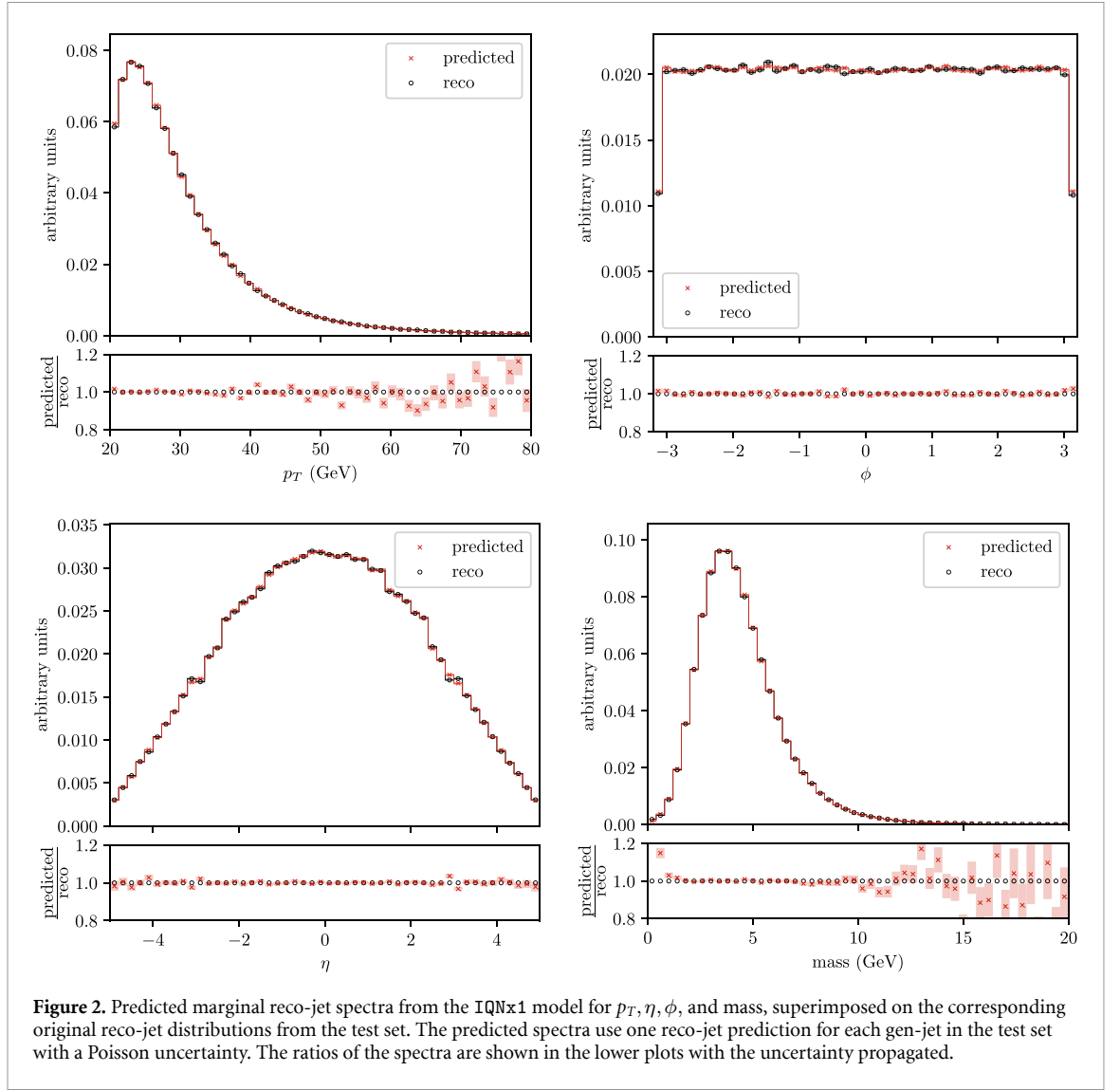
---

[7] https://github.com/alpha-davidson/Jet-IQNs.

Table 1. Hyperparameter values used to configure and train the networks comprising the `IQNx4` and `IQNx1` models.

Hyperparameter	Value
number of layers	5
nodes per layer	50
parameter initialization	Glorot [44]
activation function: LeakyReLU [45]	$\alpha = 0.3$
batch size	512
initial learning rate	10^{-3}
gradient penalty (λ)	100
total training epochs	400

Figure 1. Predicted marginal reco-jet spectra from the `IQNx4` model, one each for p_T, η, ϕ, and mass, superimposed on the corresponding original reco-jet distributions from the test set with a Poisson uncertainty. The ratios of the spectra are shown in the lower plots with the uncertainty propagated.

come from a normal approximation of the Poisson distribution to one standard deviation with a center value and variance equal to the number of counts in the bin. As described in section 3, a single network, `IQNx1`, can be trained in an autogressive manner to accomplish the same task. For comparison, the same predictions are generated for this model in figure 2 as for the `IQNx4` model.
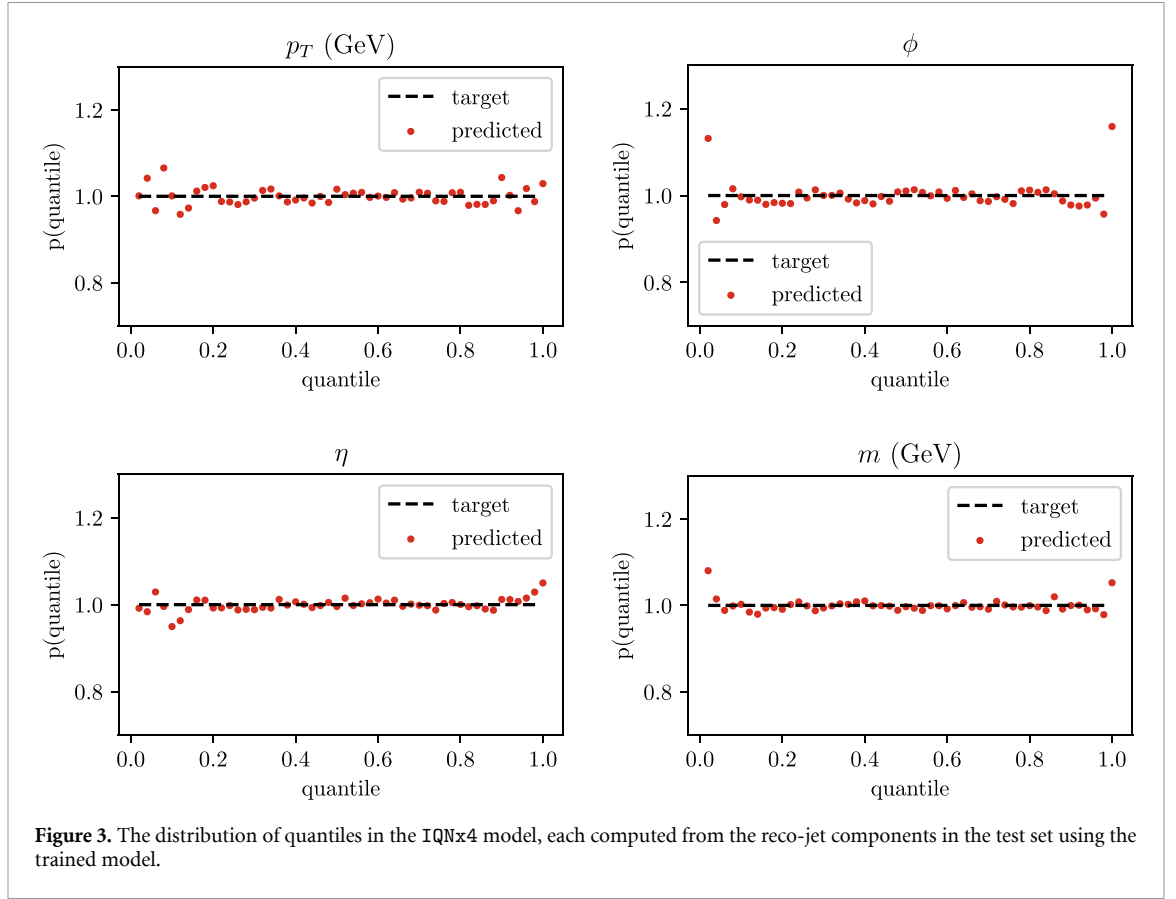
To further validate the `IQNx4`, we perform the following closure test. The cumulative distribution function (cdf) $\tau' = F(y)$ for each component $y \in \{p'_T, \eta', \phi', m'\}$ of the reco-jet 4-momentum is approximated using the 1000 predictions for each of the 10^6 gen-jets in the test set. The approximated cdfs are used to map each 4-momentum component y in the test set to a quantile, τ'. If the IQN's modeling of the quantile function and, therefore, the conditional densities is accurate, then we should expect the distribution

Figure 2. Predicted marginal reco-jet spectra from the `IQNx1` model for p_T, η, ϕ, and mass, superimposed on the corresponding original reco-jet distributions from the test set. The predicted spectra use one reco-jet prediction for each gen-jet in the test set with a Poisson uncertainty. The ratios of the spectra are shown in the lower plots with the uncertainty propagated.

of the quantiles to follow $U(0, 1)$. While there is some structure in the distributions in figure 3, we see that the quantile distributions are indeed approximately uniform.

As a further test we trained a boosted decision tree (BDT) model using XGBoost [46] to distinguish between the true reco-jet distribution and the simulated one. We created a binary classification task, where one category of examples comprised a gen-jet 4-vector concatenated with the original reco-jet 4-vector as generated by `Delphes`. The other category of examples were constructed by concatenating a gen-jet 4-vector with a reco-jet 4-vector as predicted by our model. The BDT was trained with a maximum depth of 6 and an early stopping patience of 1000 and the binary cross entropy loss. The training set was 60% of the main test set, while the validation and test sets were each 20% of the main test set. The resulting BDT had an AUC of 0.53. Thus, it was able to find a small difference between the two datasets. As the binary cross entropy learns the mean of a distribution, this indicates that there is a slight discrepancy in the mean of the learned distributions. Given the agreement in the closure tests, we attribute these small discrepancies to be driven by the tails of the distributions. While small in number these can have significant impacts on the mean, thus impacting it more than the quantiles.

The results presented above integrate over the entire gen-jet phase space. However, our claim is that IQNs are able to model conditional densities. We can investigate this claim by restricting the gen-jet phase space to a small region. The region defined by the criteria $30\,\text{GeV} < p_T < 35\,\text{GeV}$, $|\eta| < 1$, $|\phi| < 1$, and $5\,\text{GeV} < m < 10\,\text{GeV}$ contains approximately 1.2×10^4 gen-jets with roughly the same 4-momenta. The associated distributions of reco-jet 4-momenta are shown in figure 4. Again, we see excellent agreement between the predicted and true distributions.

Figure 3. The distribution of quantiles in the `IQNx4` model, each computed from the reco-jet components in the test set using the trained model.

In order to further assess the effectiveness of our IQN methods, we compare the predicted reco-jet marginal densities to the original reco-jet marginal densities by performing two-sample tests using the Kolmogorov–Smirnov (KS) test statistic

$$d_{\mathrm{KS}}(P, Q) \equiv \sup_{x \in \mathbb{R}} |F_P(x) - F_Q(x)|, \tag{8}$$

where F_P and F_Q denote the cumulative distribution functions of each of the reco-jet variables of the samples P and Q, respectively. The null distribution of the KS statistic is approximated by repeatedly splitting the test set into two sets P_i and Q_i using bootstrap resampling and computing the KS statistic for each pair of bootstrap samples (P_i, Q_i). Similarly, the KS distribution for the `IQNx4` and `IQNx1` models are computed in the same manner as the null distribution, but instead compares bootstrapped datasets from the null distribution and bootstrapped data from generated data from each model. The results are shown in figure 5 for both IQN models and compared with those of a baseline calculation using the simulated reco-jet distributions of the test set. The overlap in distributions of KS test results between the baseline distribution and our models indicate that our IQN models generate distributions similar to the baseline distributions. Recent work [47] proposes robust tests of generative models that will be interesting to apply to the IQN model for further analysis.

5. Discussion

We have demonstrated the efficacy of IQNs for sampling from conditional densities. Moreover, the same model can be used to approximate the densities themselves. We have demonstrated that one can model the conditional probability distributions in the output space $\boldsymbol{y}$ by four independent deep neural networks. We also demonstrated that one can train a single network to accomplish the same task. The `IQNx4` and `IQNx1` models yield excellent results over most of the jet phase space with small deviations where we expect to see them, namely, where there are comparatively fewer training data for p_T and m, or where the modeling of detector shape effects in η is difficult. It is likely, however, that these effects could be reduced by suitable adjustments to the training data, for example, by modeling the residuals ($\boldsymbol{\delta y} = \boldsymbol{y} - \boldsymbol{x}$) rather than the ratios.

The choice of using a model in the style of `IQNx1` or `IQNx4` would depend on the specific use case. Based on the results obtained here the single model version is slightly more performant. This is likely due to
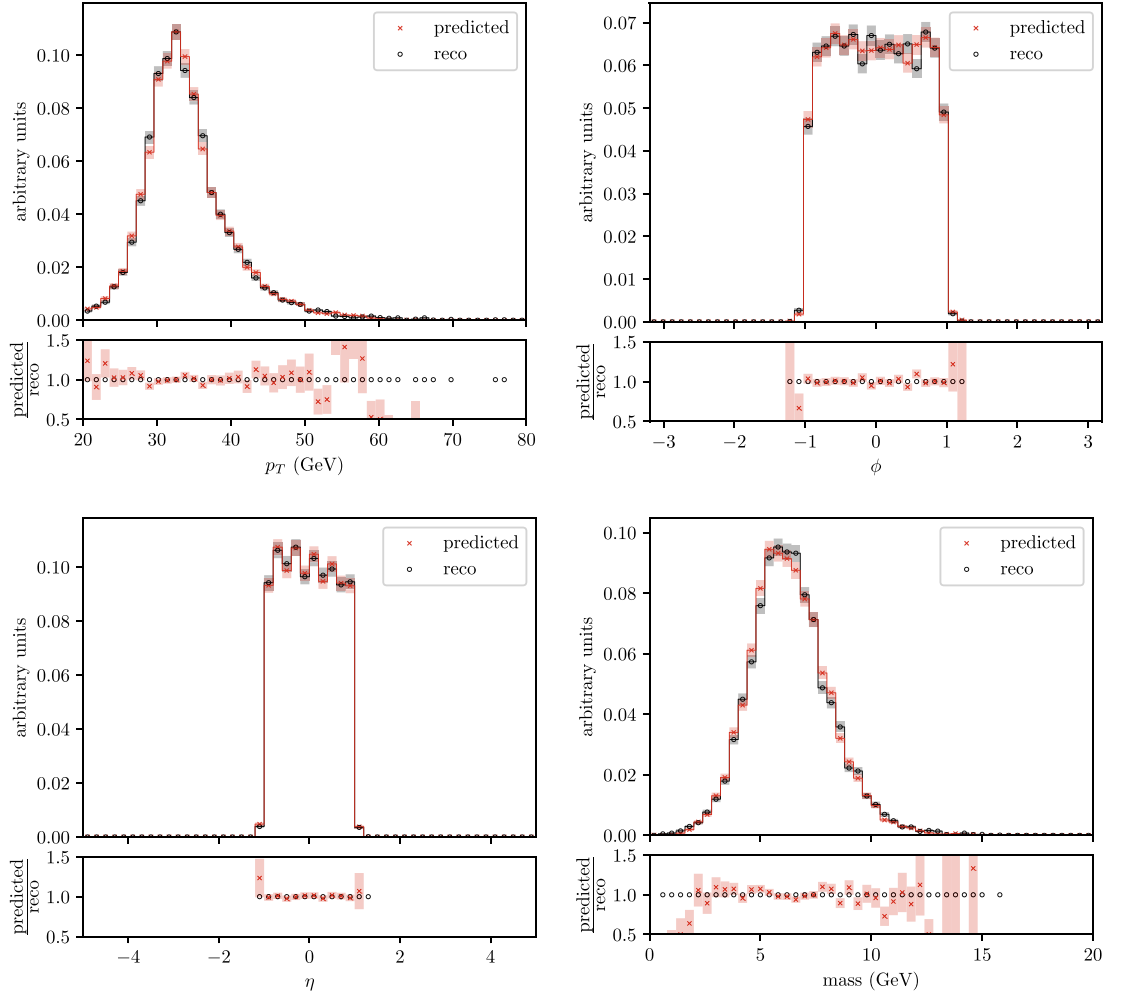
Figure 4. The predicted marginal reco-jet spectra from the `IQNx4` model, conditioned on gen-jets in the phase-space bin $30\,\text{GeV} < p_T < 35\,\text{GeV}$, $|\eta| < 1$, $|\phi| < 1$, and $5\,\text{GeV} < m < 10\,\text{GeV}$, are compared to the reco-jet spectra of the test data. The predicted spectra displays the results of one prediction from the IQN for each gen-jet in the test set with a Poisson uncertainty. The ratios of the spectra are shown in the lower plots with the uncertainty propagated; 83%, 92%, 95%, and 82% of data points are displayed within chosen y-axis limits for p_T, η, ϕ, and mass respectively.

information shared between the four different prediction modes, but this may not always be the case. Indeed, if independence between the different predictions is absolutely required, having separate network is a necessity as there is always the potential for small correlations in the single model method. Thus, the choice of model should be made based on the specific use case and its requirements.

We also introduced a novel regularization term that mitigates quantile crossing without interfering with the convergence of the network training. For the particular tasks considered, we found that this term did not have a measurable impact. It is still in principle useful though, and may be of use in tasks with more complex distributions or with less training data.

Conditional densities are ubiquitous in nuclear and high-energy physics. For example, they appear in statistical models $p(\boldsymbol{d}|\boldsymbol{\mu}, \boldsymbol{\nu})$, where $\boldsymbol{d}$ are observable data and $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are parameters of interest and nuisance parameters, respectively. They also appear in response functions $r(\boldsymbol{y}|\boldsymbol{x})$ that appear in multi-dimensional integrals of the form $o(\boldsymbol{y}) = \int r(\boldsymbol{y}|\boldsymbol{x})\, u(\boldsymbol{x})\, d\boldsymbol{x}$ that map an unobserved spectrum $u(\boldsymbol{x})$ to an observed spectrum $o(\boldsymbol{y})$ of which the jet 4-momenta spectra are a typical example. Normalizing flows have been used for applications similar to the one addressed in our work, such as in [24–26, 48] with [26] being the closest to our task. Preliminary work indicates their usefulness for our stochastic folding task as well. In these tests, more model choices and tuning are required than for the IQN approach, thus more targeted future work will be needed for a fair quantitative comparison.

There is a renewed push in high-energy physics to publish full statistical models [9]. IQNs provide a simple and effective way to both encapsulate statistical models and to compute them quickly, as well as to model the numerous response functions that appear in the analysis of particle physics data at the LHC and other particle physics research facilities.
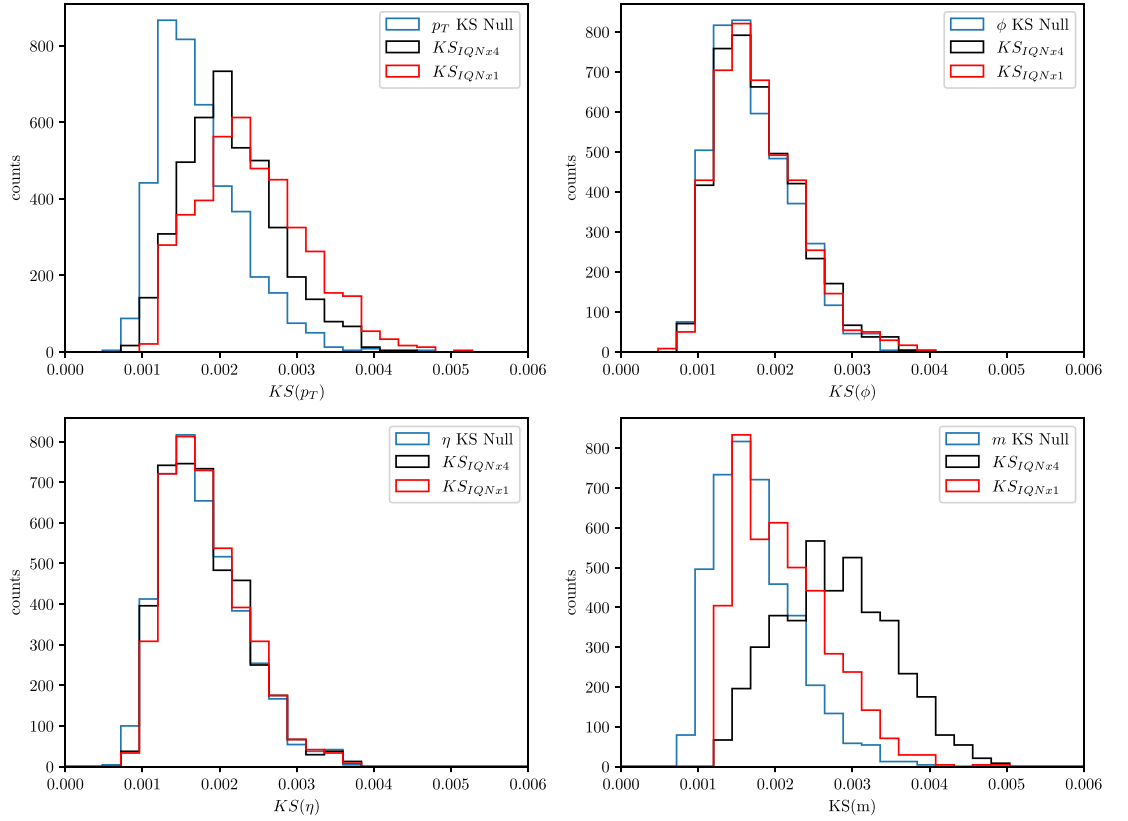
Figure 5. Kolmogorov-Smirnow (KS) tests for each predicted variable computed between 1×10^3 tests of 1) two random sets of 5×10^5 samples each drawn from the test dataset (blue), 2) one random set of 5×10^5 samples from the test dataset and one random set of 5×10^5 samples from the `IQNx4` generated dataset (black), and 3) one random set of 5×10^5 samples from the test dataset and one random set of 5×10^5 samples from the `IQNx1` generated dataset (red).

Finally, IQNs could be the basis of very fast simulators in which the hand-coding of conditional densities is replaced by appropriately trained networks. A high-fidelity simulator, such as the ones based on `GEANT4`, can be regarded as a tree of conditional densities from which one samples. The slower parts of a full `GEANT4`-based simulator could be replaced by fast emulators modeled using IQNs. Indeed, this is precisely the motivation for fast simulators like `Delphes`, which is itself outperformed in speed by our IQN-based approach by a factor of ≈ 500.

6. Conclusions

In this work, we presented an application of IQNs, namely, the stochastic folding of jet observables. Our IQN architecture comprises a feed-forward, fully-connected neural network, which is straightforward to train, particularly when compared to other generative modeling techniques such as GANs that tend to be more unstable and require more careful tuning. Our approach uses one-hot encoding to select which quantity of the multi-dimensional conditional density is computed and is conditioned on the desired quantile. Consequently, by randomly sampling quantiles from $U(0, 1)$, samples from the multi-dimensional density can be readily generated. The trained IQNs approximate the marginal densities of the jet observables accurately across the jet phase space. Furthermore, we provided some evidence that this is also true for conditional densities. But confirming that this is true point-by-point over the jet phase space is a challenging task that is the focus of ongoing study.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://zenodo.org/doi/10.5281/zenodo.13121128.

Acknowledgments

ORCID iDs

Braden Kronheim ● https://orcid.org/0000-0003-0704-0972
Ali Al Kadhim ● https://orcid.org/0000-0003-3490-8407
Michelle P Kuchera ● https://orcid.org/0000-0002-9735-6147
Raghuram Ramanujan ● https://orcid.org/0009-0008-3476-3379

References

[1] Schwartzman A 2015 Jet energy calibration at the LHC *Int. J. Mod. Phys.* A **30** 11
[2] Allison J *et al* 2016 Recent developments in GEANT4 *Nucl. Instrum. Methods Phys. Res.* A **835** 186–225
[3] Abdullin S, Azzi P, Beaudette F, Janot P and Perrotta A The CMS Collaboration 2011 The fast simulation of the CMS detector at LHC *J. Phys.: Conf. Ser.* **331** 032049
[4] The ATLAS Collaboration 2010 The ATLAS simulation infrastructure *Eur. Phys. J.* C **70** 823–74
[5] Chen C, Cerri O, Nguyen T Q, Vlimant J R and Pierini M 2021 Analysis-specific fast simulation at the LHC with deep learning *Comput. Softw. Big Sci.* **5** 15
[6] Carminati F, Khattak G, Loncar V, Nguyen T Q, Pierini M, Brito Da Rocha R, Samaras-Tsakiris K, Vallecorsa S and Vlimant J-R 2020 Generative adversarial networks for fast simulation *J. Phys.: Conf. Ser.* **1525** 012064
[7] Sergeev F, Jain N, Knunyants I, Kostenkov G and Trofimova E 2021 Fast simulation of the LHCb electromagnetic calorimeter response using VAEs and GANs *J. Phys.: Conf. Ser.* **1740** 012028
[8] The ATLAS Collaboration 2024 Deep generative models for fast photon shower simulation in ATLAS *Comput. Softw. Big Sci.* 1525 012077
[9] Cranmer K *et al* 2022 Publishing statistical models: getting the most out of particle physics experiments *SciPost. Phys.* **12** 037
[10] Tumasyan A *et al* 2022 Measurement and QCD analysis of double-differential inclusive jet cross sections in proton-proton collisions at $\sqrt{s} = 13$ TeV *J. High Energy Phys.* JHEP02(2022)142
[11] The ATLAS collaboration 2021 Measurement of hadronic event shapes in high-pt multijet final states at $\sqrt{s} = 13$ TeV with the ATLAS detector *J. High Energy Phys.* JHEP01(2021)188
[12] White H 1992 Nonparametric estimation of conditional quantiles using neural networks *Computing Science and Statistics*, ed C Page and R LePage (Springer) pp 190–9
[13] Taylor J W 2000 A quantile regression neural network approach to estimating the conditional density of multiperiod returns *J. Forecast.* **19** 299–311
[14] Baldi P, Cranmer K, Faucett T, Sadowski P and Whiteson D 2016 Parameterized neural networks for high-energy physics *Eur. Phys. J.* C **76** 235
[15] Cranmer K, Brehmer J and Louppe G 2020 The frontier of simulation-based inference *Proc. Natl Acad. Sci.* **117** 30055–62
[16] Brehmer J and Cranmer K 2020 Simulation-based inference methods for particle physics pp 579–611
[17] Gouttes A, Rasul K, Koren M, Stephan J and Naghibi T 2021 Probabilistic time series forecasting with implicit quantile networks *Int. Conf. on Machine Learning* (*2021*)
[18] Blue J, Kronheim B, Kuchera M, Ramanujan R, Biscarat C, Campana S, Hegner B, Roiser S, Rovelli C I and Stewart G A 2021 Conditional wasserstein generative adversarial networks for fast detector simulation *EPJ Web Conf.* **251** 03055
[19] Bellagente M, Butter A, Kasieczka G, Plehn T and Winterhalder R 2020 How to GAN away detector effects *SciPost Phys.* **8** 70
[20] Erdmann M, Glombitza J and Quast T 2019 Precise simulation of electromagnetic calorimeter showers using a wasserstein generative adversarial network *Comput. Softw. Big Sci.* **3** 579–611
[21] Paganini M, de Oliveira L and Nachman B 2018 CaloGAN: simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks *Phys. Rev.* D **97** 014021
[22] Albergo M S, Kanwar G and Shanahan P0167e 2019 Flow-based generative models for markov chain monte carlo in lattice field theory *Phys. Rev.* D **100** 034515
[23] Gao C, Isaacson J and Krause C 2020 `i-flow`: high-dimensional integration and sampling with normalizing flows *Mach. Learn.: Sci. Technol.* **1** 045023
[24] Gao C, Höche S, Isaacson J, Krause C and Schulz H 2020 Event generation with normalizing flows *Phys. Rev.* D **101** 076002
[25] Bieringer S, Butter A, Heimel T, Höche S, Köthe U, Plehn T and Radev S T 2021 Measuring qcd splittings with invertible networks *SciPost Phys.* **10** 126
[26] Krause C and Shih D 2023 Fast and accurate simulations of calorimeter showers with normalizing flows *Phys. Rev.* D **107** 113003
[27] Brehmer J and Cranmer K 2020 Flows for simultaneous manifold learning and density estimation *Conf. on Neural Information Processing Systems* (*Vancouver, Canada*)
[28] Dey B, Zhao D, Newman J A, Andrews B H, Izbicki R and Lee A B 2022 Calibrated predictive distributions via diagnostics for conditional coverage (https://doi.org/10.48550/arXiv.2205.14568)
[29] Cheong S, Cukierman A, Nachman B, Safdari M and Schwartzman A 2020 Parametrizing the detector response with neural networks *J. Instrum.* **15** 01030
[30] Sirunyan A M *et al* 2020 A deep neural network for simultaneous estimation of B jet energy and resolution *Comput. Softw. Big Sci.* **4** 10
[31] Ostrovski G, Dabney W and Munos R 2018 Autoregressive quantile networks for generative modeling *Proc. 35th Int. Conf. on Machine Learning* (*Proc. of Machine Learning Research* vol 80), ed J Dy and A Krause (PMLR) pp 3936–45

[32] Kronheim B, Kuchera M P, Prosper H B and Ramanujan R 2021 Implicit quantile neural networks for jet simulation and correction *NeurIPS Workshop on Machine Learning and The Physical Sciences*

[33] Koenker R and Bassett G 1978 Regression quantiles *Econometrica* **46** 33

[34] Cannon A J 2018 Non-crossing nonlinear regression quantiles by monotone composite quantile regression neural network, with application to rainfall extremes *Stoch. Environ. Res. Risk Assess.* **32** 3207–25

[35] Jun Moon S, Jeon J-J, Sang Hun Lee J and Kim Y 2021 Learning multiple quantiles with neural networks *J. Comput. Graph. Stat.* **30** 1–11

[36] Tagasovska N and Lopez-Paz D 2019 Single-model uncertainties for deep learning *Advances in Neural Information Processing Systems* vol 32, ed H Wallach, H Larochelle, A Beygelzimer, F d' Alché-Buc, E Fox and R Garnett (Curran Associates, Inc.)

[37] Bierlich C *et al* 2022 A comprehensive guide to the physics and usage of PYTHIA 8.3 **3**

[38] Selvaggi M 2016 Delphes 3: latest developments *J. Phys.: Conf. Ser.* **762** 012051

[39] Bayatian G L *et al* 2007 CMS technical design report, volume II: Physics performance *J. Phys. G: Nucl. Part. Phys.* **34** 995–1579

[40] Cacciari M, Salam G P and Soyez G 2008 The anti-ktjet clustering algorithm *J. High Energy Phys.* **2008** 063–063

[41] Cacciari M, Salam G P and Soyez G 2011 Fastjet user manual

[42] Kronheim B, Al Kadhim A and Kuchera M P 2024 *alpha-davidson/Jet-IQNs: Published Code* (https://doi.org/10.5281/zenodo.13121128)

[43] Reddi S J, Kale S and Kumar S 2018 On the convergence of adam and beyond *Int. Conf. on Learning Representations*

[44] Glorot X and Bengio Y 2010 Understanding the difficulty of training deep feedforward neural networks *Proc. 13th Int. Conf. on Artificial Intelligence and Statistics* (*Proc. Machine Learning Research* vol 9), ed Y W Teh and M Titterington (PMLR) pp 249–56

[45] Xu B, Wang N, Chen T and Li M 2015 Empirical evaluation of rectified activations in convolutional network (arXiv:1505.00853)

[46] Chen T and Guestrin C 2016 Xgboost: a scalable tree boosting system *ACM SIGKDD Int. Conf. on Knowledge Discvoery and Data Mining* pp 785–94

[47] Das R, Favaro L, Heimel T, Krause C, Plehn T and Shih D 2024 How to understand limitations of generative networks *SciPost Phys.* **16** 31

[48] Vaselli F, Cattafesta F, Asenov P and Rizzi A 2024 End-to-end simulation of particle physics events with flow matching and generator oversampling *Mach. Learn.: Sci. Technol.* **5** 035007