Published for SISSA by Springer
Received: June 2, 2023
Revised: July 25, 2023
Accepted: August 7, 2023
Published: September 13, 2023

Fitting a deep generative hadronization model

Jay Chan,[a,b] **Xiangyang Ju,**[c] **Adam Kania,**[f] **Benjamin Nachman,**[b,d] **Vishnu Sangli**[e,b] **and Andrzej Siódmok**[f]

[a] *Department of Physics, University of Wisconsin-Madison, Madison, WI 53706, U.S.A.*
[b] *Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, U.S.A.*
[c] *Scientific Data Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, U.S.A.*
[d] *Berkeley Institute for Data Science, University of California, Berkeley, CA 94720, U.S.A.*
[e] *Department of Physics, University of California, Berkeley, CA 94720, U.S.A.*
[f] *Institute of Applied Computer Science, Jagiellonian University, ul. Łojasiewicza 11, 30-348 Krakow, Poland*

E-mail: jaychan@lbl.gov, xju@lbl.gov, ad.kania@student.uj.edu.pl, bpnachman@lbl.gov, vsangli@berkeley.edu, andrzej.siodmok@uj.edu.pl

Abstract: Hadronization is a critical step in the simulation of high-energy particle and nuclear physics experiments. As there is no first principles understanding of this process, physically-inspired hadronization models have a large number of parameters that are fit to data. Deep generative models are a natural replacement for classical techniques, since they are more flexible and may be able to improve the overall precision. Proof of principle studies have shown how to use neural networks to emulate specific hadronization when trained using the inputs and outputs of classical methods. However, these approaches will not work with data, where we do not have a matching between observed hadrons and partons. In this paper, we develop a protocol for fitting a deep generative hadronization model in a realistic setting, where we only have access to a set of hadrons in data. Our approach uses a variation of a Generative Adversarial Network with a permutation invariant discriminator. We find that this setup is able to match the hadronization model in Herwig with multiple sets of parameters. This work represents a significant step forward in a longer term program to develop, train, and integrate machine learning-based hadronization models into parton shower Monte Carlo programs.

Keywords: Parton Shower, Properties of Hadrons

ArXiv ePrint: 2305.17169

https://doi.org/10.1007/JHEP09(2023)084

JHEP09(2023)084

Contents

JHEP09(2023)084

1 Introduction

Hadronization connects theory and experiment by transforming the fundamental degrees of freedom — quarks and gluons — with observable degrees of freedom — hadrons. However, we do no have a first-principles understanding of hadronization and so existing approaches use physically-inspired, highly flexible models fit to data. Our vision is to replace these hand-crafted models with deep learning, where the additional exprresivity would have the potential to enhance precision, the models would be readily differentiable, and they would be naturally compatible with Graphical Processing Unit (GPUs).

There are currently two hadronization models in wide use: the cluster model [1] and the string model [2, 3]. The former is employed by default in the Herwig [4–7] and Sherpa [8, 9] Parton Shower Monte Carlo (PSMC) programs and the latter is used by default in the Pythia [10, 11] PSMC. Previously, refs. [12] and [13] showed that deep generative models could emulate the string and cluster models, respectively, in a simple setting where the neural network has access to parton-hadron pairs and only pions are produced.[1] Furthermore, these models were integrated into the Pythia and Herwig PSMC programs. These papers marked an important milestone, but represent only the first steps along a multiyear program to achieve a complete, integrated, and tuned machine learning (ML)-based hadronization model.

While previous work has shown that neural networks can emulate the existing hadronization models, we want to eventually fit the models to data. A fundamental challenge with using data directly is that hadronization acts locally on partons while only non-local information about hadrons is observable. In other words, events are measured as a permutation-invariant set of hadrons that have no inherent order or grouping to know which hadrons 'came from' the same partons. This means that we need a model that can learn to generate hadrons from partons based on information from a loss function that acts on the set of observable hadrons.

[1] Hadron type was taken from Herwig.

The two-level challenge of fitting to data rules out most standard implementations of deep generative models. Variational Autoencoders (VAE) [14, 15], Normalizing flows (NF) [16, 17], and diffusion models [18–20] do not directly apply because we need to know the probability density of the partons and we need a permutation invariant reconstruction loss (VAE), probability density (NF), or score function (diffusion). While there has been some progress on these fronts [21–30], Generative Adversarial Networks (GANs) [31, 32] can be naturally applied to this setting. For GANs, the latent space does not require a tractable probability density, the discriminator can be applied on a different level (hadrons) as the generator (partons), and permutation invariance can be enforced by using a set-based classifier for the discriminator. GANs were the first deep generative model applied to particle physics data [33–35] and have since been extensively studied (see e.g. refs. [36–38]). GAN-like setups have also been used for two-level fitting in the context of parameter estimation [39] and unfolding [40]. We propose to use GANs for fitting hadronization models to data.

We embed the GAN-based hadronization model HadML introduced in ref. [13] in a full event-level fitting framework. A fully connected neural network takes as input individual clusters and outputs pairs of hadrons. This network acts in the cluster rest frame. The resulting hadrons are then boosted to the lab frame and the GAN discriminator is based on Deep Sets [41], which is a permutation invariant neural network architecture. We restrict ourselves to the cluster model inputs (clusters created from pre-confined partons) and pion outputs in order to focus on the two-level fitting challenge. These simplifications will be relaxed in future work.

This paper is organized as follows. Section 2 introduces the conceptual and technical details behind our fitting framework. Numerical examples are presented in section 3, including two variations on the cluster model. The paper ends with conclusions and outlook in section 4.

2 Methods

2.1 Statistical approach

Our goal is to learn a conditional generator function $G(z, \lambda; \omega_G)$ which maps cluster kinematic properties onto the kinematic properites of the two[2] hadrons from each cluster decay $\{h_1, h_2\} \in \mathbb{R}^{2N_h}$ with the parameters ω_G. Here, $z \in \mathbb{R}^{N_z}$ is the input noise variable sampled from the prior $p(z)$, and $\lambda \in \mathbb{R}^{N_\lambda}$ is the conditional variable, namely the cluster kinematic properties. Since two hadrons from a cluster decay must be back-to-back in the rest frame of cluster, the generator G can instead output the polar angles θ and ϕ of the "first hadron" in the cluster rest frame. Note that here ϕ is defined in the range of $(-\pi/2,\ \pi/2)$, and the hadron with ϕ in this range is defined to be the first hadron. In the original setup [13], a discriminator function $D(\theta, \phi; \omega_D)$, parametrized with ω_D, is learned to represent the probability that $\{\theta, \phi\}$ came from cluster fragmentation rather than the

[2]The cluster model can produce more than two hadrons, but most of the time, at the energies we consider, there are only two. We restrict to two for this study and will explore more complex decays in future work.

JHEP09(2023)084

generator G. G and D are then trained alternately to maximize and minize the loss function, respectively:

$$L = - \sum_{\lambda \sim \text{HERWIG},\, z \sim p(z)} \left(\log \left(D \left(\tau \left(\lambda \right) \right) \right) + \log \left(1 - D \left(G \left(z, \lambda \right) \right) \right) \right) , \tag{2.1}$$

where τ is the cluster fragmentation.

In the setup above, all hadrons are paired and matched to a cluster. In the actual data, however, the only observables are the kinematic properties of each individual hadron. In order to be able to fit the model to actual data, where the hadron matching and cluster information is not accessible, the discriminator function is modified to be $D_E\left(x\right)$, where D_E takes a set of hadron kinematic properties $x \equiv \{h_1, h_2, \ldots, h_n\}$ in the same event as inputs. Furthermore, we parameterize D_E as a Deep Sets model [41]:

$$D_E\left(x\right) = F\left(\frac{1}{n} \sum_{i=1}^{n} \Phi\left(h_i, \omega_{D_\Phi} \right), \omega_F \right) , \tag{2.2}$$

where Φ embeds a set of hadrons into a fixed-length latent space and F acts on the average of the latent space. Due to the average, D_E can take any length of hadron set and is invariant under permutations of hadrons. The loss function thus becomes:

$$L = - \sum_{x \sim \text{data}} \log \left(D_E \left(x \right) \right) - \sum_{\{G\} \sim \text{HERWIG},\, z \sim p(z)} \log \left(1 - D_E \left(\{ G \left(z, \lambda \right) \} \right) \right) , \tag{2.3}$$

where $\{G\left(z, \lambda\right)\}$ is generated by a set of clusters that came from the same event. The generator acts in the cluster rest frame and then the resulting hadrons are boosted into the lab frame before being passed to the discriminator. A summary of the setup and how it differs from ref. [13] is presented in figure 1.

In our implementation, G is a neural network. However, this approach could also be used to fit (without binning) data to a parametric physics model as well. For that case, G would be e.g. the cluster model and the parameters would not be weights and biases of a neural network, but instead the parameters of the cluster model. This would require making the cluster model differentiable so that gradients could be passed through the model. We leave explorations of this hybrid setup to future work.

2.2 Machine learning implementation

Both the generator and discriminator functions are parametrized as neural networks and implemented using PyTorch [42]. The generator is a fully connected network which consists of two hidden layers with 256 nodes per layer. The noise dimension is set to 10. The discriminator comprises two networks ϕ and F. Both ϕ and F are a fully connected network with two hidden layers of 256 nodes each. Each intermediate layer in these networks uses a batch normalization and a LeakyReLU [43] activation function. The last layer of the generator uses a tanh activation function to restrict the outputs to be in the range of $(-1,\ 1)$. The outputs are then scaled and transformed linearly to match the actual range $(-\pi/2,\ \pi/2)$ for ϕ and $(0,\ \pi)$ for θ. The last layer of F uses a sigmoid activation function and no activation is used for the last layer of Φ.

JHEP09(2023)084

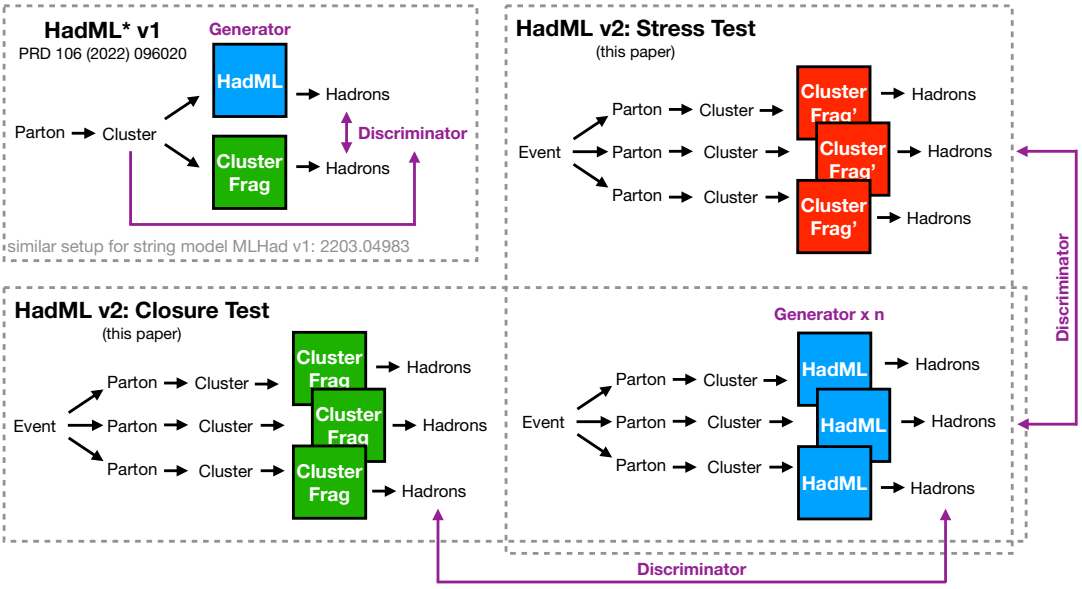

Figure 1. An overview of the model presented in this paper and how it compares to HadML v1 from ref. [13]. Since the clusters are not observable in data, the discriminator in v2 acts on sets of hadrons and does not have access to cluster-hadron-hadron labels. We first study the performance in the same Herwig setup as in ref. [13] ('Closure Test') and then check that it is also able to fit another Herwig setup (Cluster Frag') with variations in the cluster hadronization model ('Stress Test').

All neural network inputs are normalized to the range of $(-1,\ 1)$, whereas the noise prior p is a Gaussian distribution with a mean of 0 and width of 1. The generator and discriminator are optimized alternately (1 discriminator step and 5 generator steps) with Adam [44] with a learning rate of 5×10^{-7} and 10^{-4} for the generator and discriminator, respectively. The training uses a batch size of 10,000 and is performed for 6,000 epochs. The hyperparameters were optimized with Weights and Biases [45].

3 Results

3.1 Datasets

Crucial data for fitting hadronisation models are LEP events collected in e^+e^- collisions at the center-of-mass energy $\sqrt{s} = 91.2\,\text{GeV}$. Therefore, we used such events generated with version 7.2.1 of the Herwig Monte Carlo generator for a training dataset for our Generative Hadronization Model. As mentioned earlier, the cluster model [1] is used for hadronisation in the Herwig generator. Based on the color preconfinement [46], the cluster model groups a partonic final state into a set of colour-singlet clusters (pre-hadrons) with an invariant mass distribution that is independent of the specific hard scattering process or its centre-of-mass energy and that peaks at low masses. Therefore, most clusters decay into two hadrons. However, a small fraction of clusters are too heavy for this approach to be justified. Therefore, these heavy clusters are first split into lighter clusters before decaying. The decay of such massive clusters is not discussed in this publication but will

JHEP09(2023)084

be considered in future work. Each entry in our training data set includes information about the four-momentum of all the light clusters in an event and the four-momenta of their parents (partons) and children (hadrons), along with their flavours. An example of an entry from our data sets is available on Zenodo at ref. [47]. To simplify the training data further, only decays into π mesons were considered.[3] To check whether the model can adapt to different variants of the kinematics of hadron decays, we also prepared two datasets with different, minimal (0) and maximal (2) settings of the **ClSmr** parameter. The **ClSmr** parameter is the main parameter governing the kinematics of cluster hadron decay. Hadrons that contain a parton produced in the perturbative stage of the event retain the direction of the parton in the cluster rest frame with possible Gaussian smearing of the direction. The smearing is controlled by the **ClSmr** parameter through an angle θ_{smear} where

$$\cos\theta_{\text{smear}} = 1 + \textbf{ClSmr}\log\mathcal{R}, \tag{3.1}$$

where $\mathcal{R}$ is a uniform random number chosen from $[0, 1]$. For more details about the parameters of the cluster model implemented in Herwig, see chapter 7 of the generator's manual [5].

In section 3.2 we use the minimal **ClSmr** as our alternative sample and refer to this setup as Herwig Cluster kin^{min}. As would be the case with actual data, we use clusters from the nominal setting when fitting the alternative sample, although changing **ClSmr** does not change the cluster kinematic properties and thus the inputs to the GAN model are statistically correct. When we fit the nominal sample, the cluster inputs to the fit are distinct but statistically identical to those in the dataset we are fitting.

3.2 Fitted models

The training history of the fit is presented in figure 2. As expected, the discriminator loss increases and the generator loss decreases, with a final value near log(2) (classifier outputs 0.5 for all examples). As an independent evaluation of the model performance, we also compute the Wasserstein distance between the true and generated four-momenta in the lab frame that are used by the discriminator to update the generator. The Wasserstein distance is computed as the average over the first Wasserstein distance for each four-vector component with Scipy [48]. Interestingly, the best Wasserstein distance decreases for the first 1000 epochs, then plateaus for the next 3000 epochs, before dropping to the final value around 5500 epochs. There are many possible variations on the GAN training setup that are possible to further improve the performance and we plan to explore these in the future.

The direct inputs and outputs of the model are shown in figure 3. The generator produces two outputs per cluster, corresponding to the angle of one of the pions in the cluster rest frame in spherical coordinates. Histograms corresponding to this model are shown in the top row of figure 3. The marginal distributions looks similar to isotropic

[3] In Herwig, this is achieved by adding the following line: set HadronSelector:Trial 1 into the default LEP.in input card. The only other modification to the default hadronisation settings was the change that the hadrons produced from cluster decays were on the mass shell. This can be achieved by adding the command: set ClusterDecayer:OnShell Yes in the input file.

JHEP09(2023)084

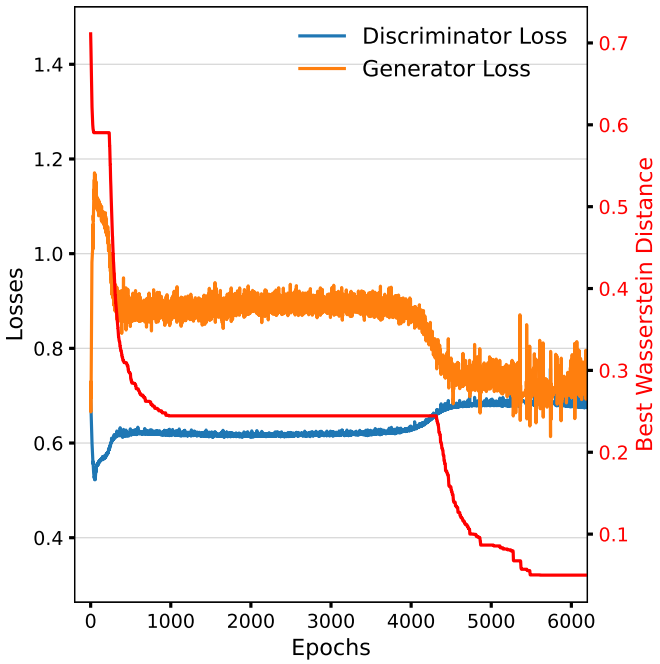

Figure 2. Generator loss, discriminator loss and running best Wasserstein distance as a function of the training epoch. The running best Wasserstein distance is quantified by the y axis on the right side of the plot.

decays. For illustration, we also show what an initialized, untrained GAN looks like in both coordinates. The fact that the initial GAN is so far from the final GAN is a non-trivial demonstration of the learning. Both GAN models match their respective truth Herwig spectra well. The marginal ϕ distribution is uniform, which is difficult for generative models to reproduce exactly. In the future, it may be possible to make this more precise by constructing the model to give a uniform marginal.

After the clusters are decayed, the resulting hadron kinematic properties are Lorentz boosted to the lab frame and then aggregated over all clusters in the event. The second row of figure 3 shows histograms of the resulting hadron four-vectors, which are the inputs to the discriminator. We only show the energy E and the x momentum p_x, but similar trends hold for p_y and p_z. Since hadronization is a small correction for such inclusive observables, the kinematic properties are mostly set by the Herwig parton shower, which is the same for the Herwig and GAN lines in the plots (since the GAN takes the clusters from the parton shower as input). This is the reason why the initial GAN starts so close to Herwig truth. However, the alternative Herwig sample differs significantly from the nominal Herwig sample, in particular in how hadrons split energy, which is most clearly seen in the tails of the energy and momentum distributions. The GAN model is an excellent match to the Herwig events across the full spectra.

Figure 4 goes beyond the direct inputs and outputs by studying derived, but measureable, quantities. The first plot in figure 4 is the number of hadrons. Since we restrict our attention to $1 \to 2$ decays only, the number of hadrons is an even number, with a mode of 12. It is not possible to uniquely pair observed hadrons with their partner from the same cluster

JHEP09(2023)084

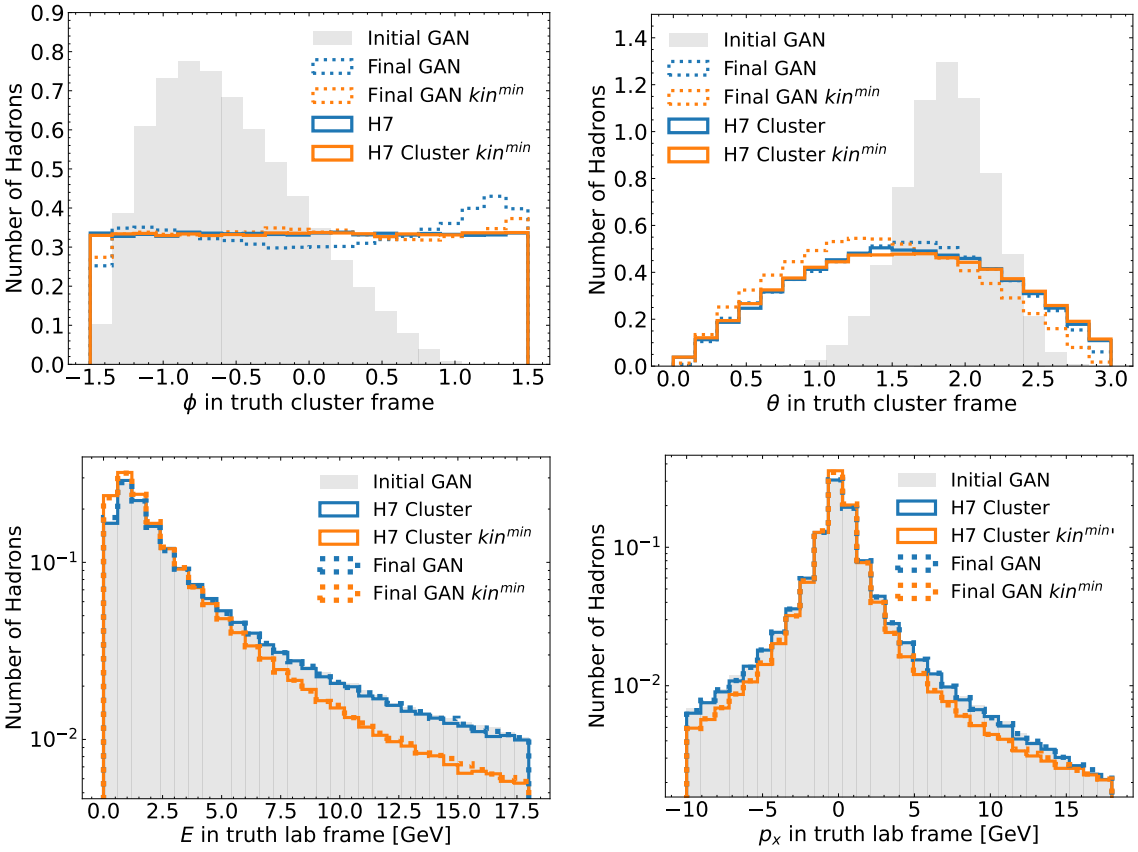

Figure 3. Top: the generative model in the true cluster rest frame. Bottom: two of the four-vector components that are used by the discriminator to update the generator.

decay, but we can approximate the combination using nearest neighbor information. In particular, since the hadron masses are small compared to the typical cluster energy in the lab frame, the two hadrons tend to be close together in phase space. For all hadrons, we assign a hadron neighbor as the particle that minimizes[4] $\Delta R^2 = \Delta\phi^2 + \Delta\eta^2$. A histogram of the resulting ΔR distribution is shown in the middle left plot of figure 4. The peak is at about 0.1, with most hadrons having a neighbor less than 0.1. While there is some difference between models in the ΔR distribution, a most distinguishing observable is the energy sharing between hadrons in the reconstructed cluster (middle right of figure 4). The nominal Herwig has more equal sharing of energy, while the alternative Herwig sample is much more asymmetric. The GAN models are able to match these trends, which both differ significantly from the initialized and untrained GAN model. Future GAN models could be improved by adding in these features to the discriminator directly.

Additionally, we consider properties of the hadrons in the reconstructed cluster frame (bottom row of figure 4). Since the reconstructed clusters are not exactly the true clusters, the ϕ and θ distributions do not exactly match the top row of figure 3, although they are

[4]This metric is most relevant for hadron colliders, but we use it here for simplicity. Similar results hold for $\Delta\theta$ instead of $\Delta\eta$.

JHEP09(2023)084

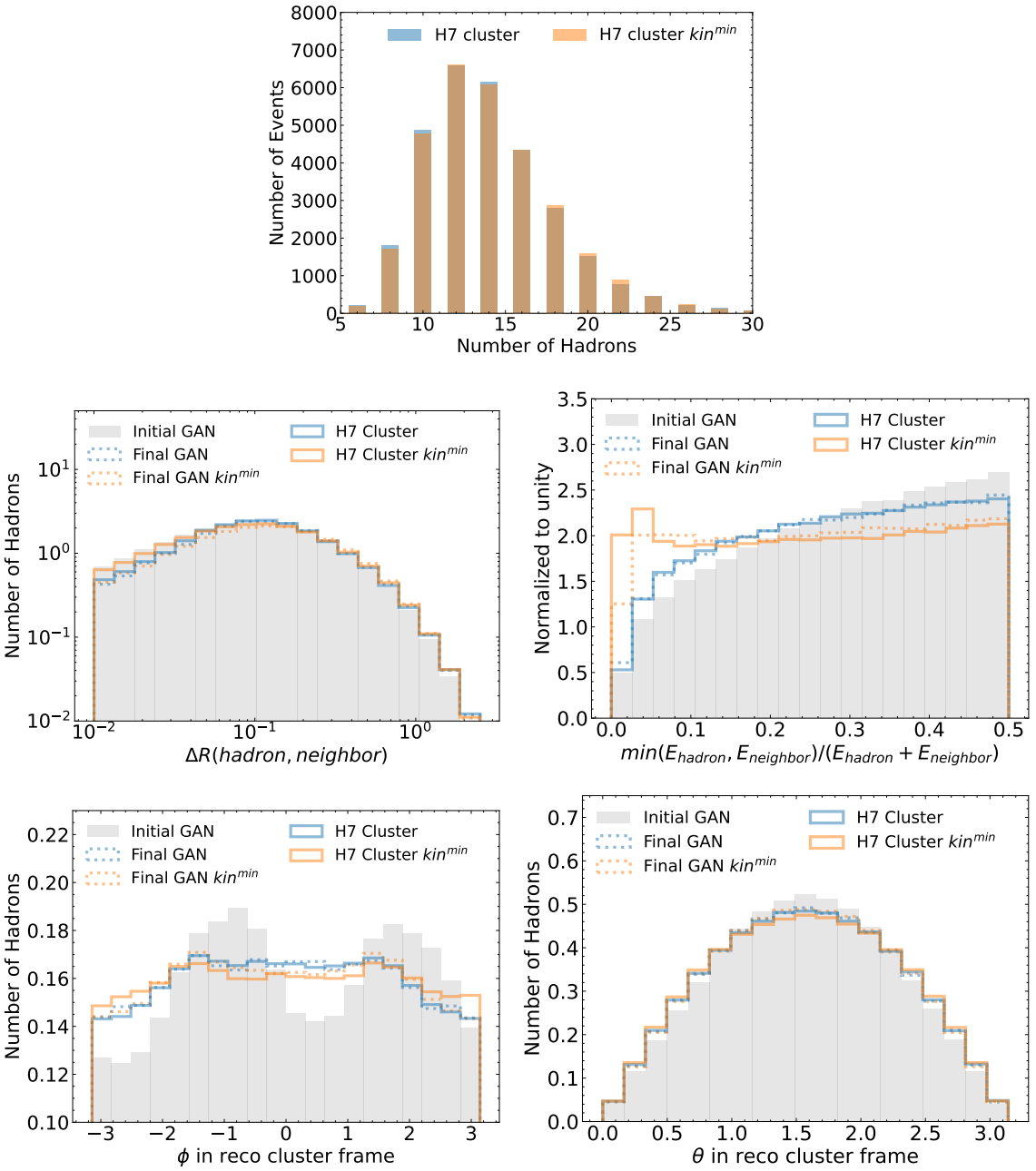

Figure 4. Top: a histogram of the number of hadrons. Since each cluster decays into two pions, the number of hadrons is an even integer. Middle: ΔR between a given hadron and its nearest neighbor in $\phi - \eta$ in the lab frame (left) and the ratio of energies between a given hadron and its neighbor (right). Bottom: the ϕ (left) and θ (right) of the hadrons in the reconstructed cluster frame.

qualitatively similar. The distribution of ϕ is more discriminating between models, where the GAN models perform well, except near the edge of phase space where both GAN model match the nominal Herwig events.

A key advantage of this fitting protocol over other methods is that it can accommodate unbinned and high-dimensional inputs. It would be possible to replace our neural network discriminator (and cross-entropy loss) with a χ^2 fit to binned histograms, like the ones in

JHEP09(2023)084

figure 3 (bottom) and 4, which are all observable in the lab frame. However, this would be a highly non-trivial modification to our setup and would necessarily be less effective. Comparing with standard tools that process low-dimensional and binned inputs would likely be inconclusive because we will not know if the difference in performance is from the tool or from the less information contained in the data.

As a compromise in order to quantify the information gained from using our discriminator setup, we use a set of auxiliary classifiers. Our nominal setup is represented by our discriminator trained on the same inputs as our GAN model and to distinguish the two Herwig cluster model variations. The information content is represented by the area under the Receiver Operating Characteristic (ROC) curve or AUC, which is a standard metric for information content. An AUC of 0.5 means there is no useful information and an AUC of 1 means that the models can be exactly distinguished. For comparison, we compute the AUC also of the single observables in figure 3 (bottom) and figure 4. We do not bin these observables to avoid arbitrary binning choices and assume (which is conservative) that the bins of any actual measurement would be chosen to be maximally effective for this task. Technically, the AUC for single observables is computed by scanning over the observable to determine the true positive rate versus the false positive rate.

Since a threshold cut may not be optimal for all observables, we have also checked how the results change if we train a simple Boosted Decision Tree (BDT) using sklearn [49]. We find that the BDT-based AUCs (including for the neural network as an observable) are consistent with the non-BDT ones. Numerically, the AUCs are as follows: neural network: 0.77, energy ratio (figure 4 upper right): 0.55, ΔR (figure 4 upper left): 0.53, rest frame θ (figure 4 lower right): 0.51, rest frame ϕ (figure 4 lower left): 0.51, p_x (figure 3 lower right): 0.54, E (figure 3 lower left): 0.57. The information content accessible to the neural network far exceeds the information in any of the individual observables.

4 Conclusions and outlook

We have presented a setup for fitting deep generative hadronization models to data. The main challenge we have addressed is the lack of truth labels connecting partons and hadrons, which were used by previous deep generative hadronization models [12, 13]. In order to address this challenge, we used a two-level Generative Adversarial Network (GAN) setup, where the generator acts at parton level and the discriminator acts on hadron level. Since there is no natural order to the hadrons, the discriminator is a classifier based on the Deep Sets architecture that can process variable-length and permutation-invariant inputs. We have shown that we can fit this model to two variations of the Herwig cluster hadronization model. The GAN is able to reproduce Herwig well, with additional refinement and optimization required in the future to improve the prevision further.

While this represents a significant step towards realizing a deep generative hadronization model, there are still other aspects to address. We have restricted our attention to pions, but a complete model will need to generate the full spectrum of hadrons in addition to kinematic information. Additionally, we have started from clusters decaying to two hadrons, while in reality, more complex arrangements are possible. In fact, we ran a test to fit

JHEP09(2023)084

the string model in Pythia using our setup,[5] but the cluster model is not flexible enough. Modifications that allow for more general parton to hadron mappings, including variable-length generation [24–30, 50], will be required in the future. In particular, we would not take pre-confinement as a starting point and instead also model the combination of partons with a neural network (so partons to hadrons instead of clusters to hadrons). Such a model would have the capacity to mimic the cluster or string models as well as go beyond either model. Such an architecture could be swapped out for our generator and use our same GAN setup to do the final fit.

Once we have a full model, there is a question of which data to use for the fit. Traditionally, hadronization models have been fit to histograms (binned differential cross section measurements) from e^+e^- data using tools like Professor [51] and other automated tuning protocols [52–54]. However, these approaches may need to be modified since the parameter space of the models is much bigger. One possibilitiy is to use a variation of Unbinned Profiled Unfolding (UPU) [55], which uses histograms to steer neural networks with a two-level fit for unfolding. The reweighting function in UPU could be replaced with the hadronization model. Another possibility is to start with unbinned data, as is now possible with machine learning-based unfolding methods [21, 56–65]. There are also now first unbinned cross section measurements [66–70], although none are currently published without binning [56]. There are not yet any unbinned measurements from e^+e^-, but results from deep inelastic scattering may be effective, since they share many of the features of e^+e^- that makes them particularly clean with respect to hadron colliders.

While there are still multiple components needed to arrive at a complete ML-based hadronization model, the program ahead is well-motivated. Current models are excellent, but the additional flexibility of neural networks will allow us to improve the precision on hadronization modeling so for precise measurements that are affected by these uncertainties. With improvements in machine learning models, it may also be possible to use these tools to learn more about hadronization itself, which remains a key research topic in nuclear physics.

Software and datasets. The code for this paper can be found at https://github.com/hep-lbdl/hadml/releases/tag/1.0.0 [71]. The data sets are hosted on Zenodo at ref. [47].

Acknowledgments

We thank Aishik Ghosh for many useful discussions. The work of AS is funded by grant no. 2019/34/E/ST2/00457 of the National Science Centre, Poland. AK is founded by the Priority Research Area Digiworld under the program Excellence Initiative – Research University at the Jagiellonian University in Cracow. JC, BN and XJ are supported by the U.S. Department of Energy (DOE), Office of Science under contract number DE-AC02-05CH11231. JC is supported by the DOE, Office of Science under contract DE-SC0017647.

[5] For this, we used exactly the same partons as in the Herwig dataset and ran the string model in Pythia, modified to only produce pions.

JHEP09(2023)084

References

[1] B.R. Webber, *A QCD Model for Jet Fragmentation Including Soft Gluon Interference*, *Nucl. Phys. B* **238** (1984) 492 [INSPIRE].

[2] B. Andersson, G. Gustafson, G. Ingelman and T. Sjöstrand, *Parton Fragmentation and String Dynamics*, *Phys. Rept.* **97** (1983) 31 [INSPIRE].

[3] T. Sjöstrand, *Jet Fragmentation of Nearby Partons*, *Nucl. Phys. B* **248** (1984) 469 [INSPIRE].

[4] G. Corcella et al., *HERWIG 6: An Event generator for hadron emission reactions with interfering gluons (including supersymmetric processes)*, *JHEP* **01** (2001) 010 [hep-ph/0011363] [INSPIRE].

[5] M. Bahr et al., *Herwig++ Physics and Manual*, *Eur. Phys. J. C* **58** (2008) 639 [arXiv:0803.0883] [INSPIRE].

[6] J. Bellm et al., *Herwig 7.0/Herwig++ 3.0 release note*, *Eur. Phys. J. C* **76** (2016) 196 [arXiv:1512.01178] [INSPIRE].

[7] J. Bellm et al., *Herwig 7.2 release note*, *Eur. Phys. J. C* **80** (2020) 452 [arXiv:1912.06509] [INSPIRE].

[8] T. Gleisberg et al., *Event generation with SHERPA 1.1*, *JHEP* **02** (2009) 007 [arXiv:0811.4622] [INSPIRE].

[9] SHERPA collaboration, *Event Generation with Sherpa 2.2*, *SciPost Phys.* **7** (2019) 034 [arXiv:1905.09127] [INSPIRE].

[10] T. Sjöstrand, S. Mrenna and P.Z. Skands, *A Brief Introduction to PYTHIA 8.1*, *Comput. Phys. Commun.* **178** (2008) 852 [arXiv:0710.3820] [INSPIRE].

[11] T. Sjöstrand, S. Mrenna and P.Z. Skands, *PYTHIA 6.4 Physics and Manual*, *JHEP* **05** (2006) 026 [hep-ph/0603175] [INSPIRE].

[12] P. Ilten, T. Menzo, A. Youssef and J. Zupan, *Modeling hadronization using machine learning*, arXiv:2203.04983 [INSPIRE].

[13] A. Ghosh, X. Ju, B. Nachman and A. Siodmok, *Towards a deep learning model for hadronization*, *Phys. Rev. D* **106** (2022) 096020 [arXiv:2203.12660] [INSPIRE].

[14] D.P. Kingma and M. Welling, *Auto-Encoding Variational Bayes*, arXiv:1312.6114 [INSPIRE].

[15] D.P. Kingma and M. Welling, *An Introduction to Variational Autoencoders*, *Found. Trends Mach. Learn.* **12** (2019) 307 [arXiv:1906.02691] [INSPIRE].

[16] C. Gao, J. Isaacson and C. Krause, *i-flow: High-dimensional Integration and Sampling with Normalizing Flows*, *Mach. Learn. Sci. Tech.* **1** (2020) 045023 [arXiv:2001.05486] [INSPIRE].

[17] I. Kobyzev, S.J.D. Prince and M.A. Brubaker, *Normalizing Flows: An Introduction and Review of Current Methods*, *IEEE Trans. Pattern Anal. Machine Intell.* **43** (2021) 3964 [arXiv:1908.09257].

[18] J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan and S. Ganguli, *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*, arXiv:1503.03585.

JHEP09(2023)084

[19] J. Ho, A. Jain and P. Abbeel, *Denoising Diffusion Probabilistic Models*, arXiv:2006.11239.

[20] P. Dhariwal and A. Nichol, *Diffusion Models Beat GANs on Image Synthesis*, arXiv:2105.05233.

[21] J.N. Howard, S. Mandt, D. Whiteson and Y. Yang, *Learning to simulate high energy particle collisions from unlabeled data*, *Sci. Rep.* **12** (2022) 7567 [arXiv:2101.08944] [INSPIRE].

[22] S. Klein, J.A. Raine and T. Golling, *Flows for Flows: Training Normalizing Flows Between Arbitrary Distributions with Maximum Likelihood Estimation*, arXiv:2211.02487.

[23] R. Mastandrea and B. Nachman, *Efficiently Moving Instead of Reweighting Collider Events with Machine Learning*, in the proceedings of the *36th Conference on Neural Information Processing Systems*, New Orleans U.S.A., 28 November–9 December (2022) [arXiv:2212.06155] [INSPIRE].

[24] R. Kansal et al., *Particle Cloud Generation with Message Passing Generative Adversarial Networks*, in the proceedings of the *35th Conference on Neural Information Processing Systems*, Online Conference Canada, 6–14 December (2021) [arXiv:2106.11535] [INSPIRE].

[25] E. Buhmann, G. Kasieczka and J. Thaler, *EPiC-GAN: Equivariant Point Cloud Generation for Particle Jets*, arXiv:2301.08128 [INSPIRE].

[26] B. Käch et al., *JetFlow: Generating Jets with Conditioned and Mass Constrained Normalising Flows*, arXiv:2211.13630 [INSPIRE].

[27] R. Verheyen, *Event Generation and Density Estimation with Surjective Normalizing Flows*, *SciPost Phys.* **13** (2022) 047 [arXiv:2205.01697] [INSPIRE].

[28] M. Leigh et al., *PC-JeDi: Diffusion for Particle Cloud Generation in High Energy Physics*, arXiv:2303.05376 [INSPIRE].

[29] V. Mikuni, B. Nachman and M. Pettee, *Fast Point Cloud Generation with Diffusion Models in High Energy Physics*, arXiv:2304.01266 [INSPIRE].

[30] E. Buhmann et al., *CaloClouds: Fast Geometry-Independent Highly-Granular Calorimeter Simulation*, arXiv:2305.04847 [INSPIRE].

[31] I.J. Goodfellow et al., *Generative Adversarial Nets*, in the proceedings of the *27th International Conference on Neural Information Processing Systems — Volume 2*, Montreal Canada, December 8–13 (2014) [MIT Press, Cambridge, U.S.A. (2014), p. 2672–2680].

[32] A. Creswell et al., *Generative Adversarial Networks: An Overview*, *IEEE Signal Processing Mag.* **35** (2018) 53.

[33] L. de Oliveira, M. Paganini and B. Nachman, *Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis*, *Comput. Softw. Big Sci.* **1** (2017) 4 [arXiv:1701.05927] [INSPIRE].

[34] M. Paganini, L. de Oliveira and B. Nachman, *CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks*, *Phys. Rev. D* **97** (2018) 014021 [arXiv:1712.10321] [INSPIRE].

[35] M. Paganini, L. de Oliveira and B. Nachman, *Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multilayer Calorimeters*, *Phys. Rev. Lett.* **120** (2018) 042003 [arXiv:1705.02355] [INSPIRE].

[36] A. Adelmann et al., *New directions for surrogate models and differentiable programming for High Energy Physics detector simulation*, in the proceedings of the *Snowmass 2021*, Seattle U.S.A., July 17–26 (2022) [arXiv:2203.08806] [INSPIRE].

JHEP09(2023)084

[37] S. Badger et al., *Machine learning and LHC event generation*, *SciPost Phys.* **14** (2023) 079 [arXiv:2203.07460] [INSPIRE].

[38] H.E.P.M.L. Community, *A Living Review of Machine Learning for Particle Physics*, https://iml-wg.github.io/HEPML-LivingReview/.

[39] A. Andreassen et al., *Parameter estimation using neural networks in the presence of detector effects*, *Phys. Rev. D* **103** (2021) 036001 [arXiv:2010.03569] [INSPIRE].

[40] B.N.J.T. K. Desai, *Deconvolving Detector Effects for Distribution Moments*, https://ml4physicalsciences.github.io/2022/files/NeurIPS_ML4PS_2022_43.pdf.

[41] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov and A. Smol, *Deep Sets*, arXiv:1703.06114.

[42] A. Paszke et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, in *Advances in Neural Information Processing Systems 32*, H. Wallach et al. eds., Curran Associates Inc. (2019), p. 8024–8035.

[43] B. Xu, N. Wang, T. Chen and M. Li,, *Empirical Evaluation of Rectified Activations in Convolutional Network*, arXiv:1505.00853.

[44] D.P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, arXiv:1412.6980 [INSPIRE].

[45] L. Biewald, *Experiment Tracking with Weights and Biases*, https://www.wandb.com/.

[46] D. Amati and G. Veneziano, *Preconfinement as a Property of Perturbative QCD*, *Phys. Lett. B* **83** (1979) 87 [INSPIRE].

[47] J. Chan et al., *Herwig dataset for HadML GAN training*, DOI:10.5281/ZENODO.7958362.

[48] P. Virtanen et al., *SciPy 1.0 — Fundamental Algorithms for Scientific Computing in Python*, *Nature Meth.* **17** (2020) 261 [arXiv:1907.10121] [INSPIRE].

[49] F. Pedregosa et al., *Scikit-learn: Machine Learning in Python*, *J. Machine Learning Res.* **12** (2011) 2825 [arXiv:1201.0490] [INSPIRE].

[50] T. Finke, M. Krämer, A. Mück and J. Tönshoff, *Learning the language of QCD jets with transformers*, *JHEP* **06** (2023) 184 [arXiv:2303.07364] [INSPIRE].

[51] A. Buckley et al., *Systematic event generator tuning for the LHC*, *Eur. Phys. J. C* **65** (2010) 331 [arXiv:0907.2973] [INSPIRE].

[52] P. Ilten, M. Williams and Y. Yang, *Event generator tuning using Bayesian optimization*, 2017 *JINST* **12** P04028 [arXiv:1610.08328] [INSPIRE].

[53] A. Andreassen and B. Nachman, *Neural Networks for Full Phase-space Reweighting and Parameter Tuning*, *Phys. Rev. D* **101** (2020) 091901 [arXiv:1907.08209] [INSPIRE].

[54] W. Wang et al., *BROOD: Bilevel and Robust Optimization and Outlier Detection for Efficient Tuning of High-Energy Physics Event Generators*, *SciPost Phys. Core* **5** (2022) 001 [arXiv:2103.05751] [INSPIRE].

[55] J. Chan and B. Nachman, *Unbinned profiled unfolding*, *Phys. Rev. D* **108** (2023) 016002 [arXiv:2302.05390] [INSPIRE].

[56] M. Arratia et al., *Publishing unbinned differential cross section results*, 2022 *JINST* **17** P01024 [arXiv:2109.13243] [INSPIRE].

[57] K. Datta, D. Kar and D. Roy, *Unfolding with Generative Adversarial Networks*, arXiv:1806.00433 [INSPIRE].

JHEP09(2023)084

[58] A. Andreassen et al., *OmniFold: A Method to Simultaneously Unfold All Observables*, *Phys. Rev. Lett.* **124** (2020) 182001 [arXiv:1911.09107] [INSPIRE].

[59] A. Andreassen et al., *Scaffolding Simulations with Deep Learning for High-dimensional Deconvolution*, in the proceedings of the *9th International Conference on Learning Representations*, Online Conference U.S.A., May 3–7 (2021) [arXiv:2105.04448] [INSPIRE].

[60] M. Bunse et al., *Unification of Deconvolution Algorithms for Cherenkov Astronomy*, in the proceedings of the *5th International Conference on Data Science and Advanced Analytics (DSAA)*, Turin Italy, October 1–3 (2018), p. 21–30 [DOI:10.1109/DSAA.2018.00012].

[61] T. Ruhe et al., *Mining for Spectra — The Dortmund Spectrum Estimation Algorithm*, in *Astronomical Data Analysis Software and Systems XXVI*, M. Molinaro, K. Shortridge and F. Pasian eds., Astronomical Society of the Pacific Conference Series **521**, (2019) p. 394,.

[62] M. Bellagente et al., *How to GAN away Detector Effects*, *SciPost Phys.* **8** (2020) 070 [arXiv:1912.00477] [INSPIRE].

[63] M. Bellagente et al., *Invertible Networks or Partons to Detector and Back Again*, *SciPost Phys.* **9** (2020) 074 [arXiv:2006.06685] [INSPIRE].

[64] M. Vandegar, M. Kagan, A. Wehenkel and G. Louppe, *Neural Empirical Bayes: Source Distribution Estimation and its Applications to Simulation-Based Inference*, arXiv:2011.05836 [INSPIRE].

[65] M. Backes, A. Butter, M. Dunford and B. Malaescu, *An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training*, arXiv:2212.08674 [INSPIRE].

[66] H1 collaboration, *Measurement of Lepton-Jet Correlation in Deep-Inelastic Scattering with the H1 Detector Using Machine Learning for Unfolding*, *Phys. Rev. Lett.* **128** (2022) 132002 [arXiv:2108.12376] [INSPIRE].

[67] LHCb collaboration, *Multidifferential study of identified charged hadron distributions in Z-tagged jets in proton-proton collisions at* $\sqrt{s} = 13$ *TeV*, arXiv:2208.11691 [INSPIRE].

[68] H1 collaboration, *Machine learning-assisted measurement of azimuthal angular asymmetries in deep-inelastic scattering with the H1 detector*, H relim-23-031 (2023).

[69] H1 collaboration, *Machine learning-assisted measurement of multi-differential lepton-jet correlations in deep-inelastic scattering with the H1 detector*, H relim-22-031 (2022).

[70] H1 collaboration, *Unbinned Deep Learning Jet Substructure Measurement in High* Q^2 *ep collisions at HERA*, arXiv:2303.13620 [INSPIRE].

[71] J. Chan et al., *Code for HadML GAN training*, DOI:10.5281/ZENODO.7964342.

JHEP09(2023)084