Towards a data-driven model of Hadronization using Normalizing Flows

Pheno 2024

Based on arXiv:2311.09296, SciPost Phys. 14, 027 (2023), and 2407.XXXXX

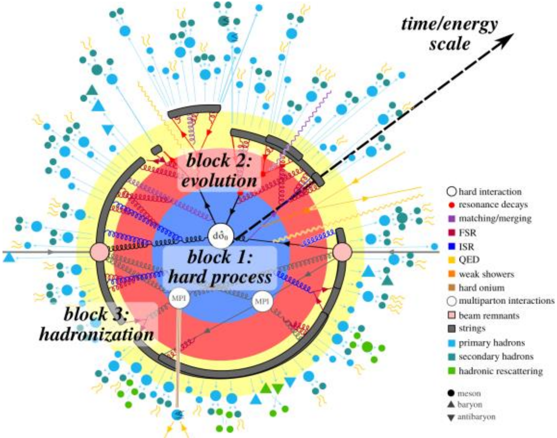
Ahmed Youssef
Ph.D. Candidate, University of Cincinnati
youssead@ucmail.uc.edu

May 15th, 2024

In collaboration with:
C. Bierlich, P. Ilten, T. Menzo, S. Mrenna, M. Szewc, M.K. Wilkinson, and J. Zupan

Motivation

Simulating Collision



- **Hard process:**
  initial high-energy interaction
- **Evolution:**
  parton shower

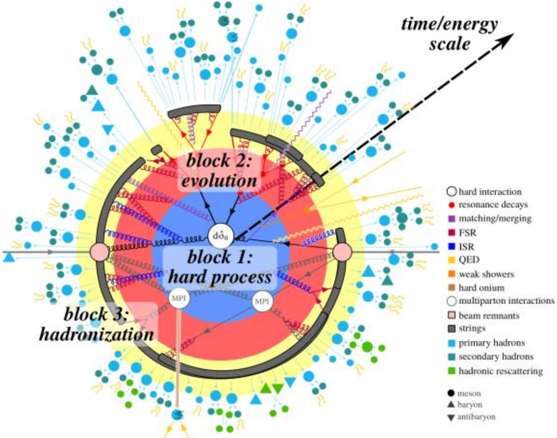
} *perturbative*

- **Hadronization:**
  combine quarks and gluons

} *non-perturbative*

Motivation

Simulating Collision



- **Hard process:**
  initial high-energy interaction
- **Evolution:**
  parton shower

perturbative

- **Hadronization:**
  combine quarks and gluons

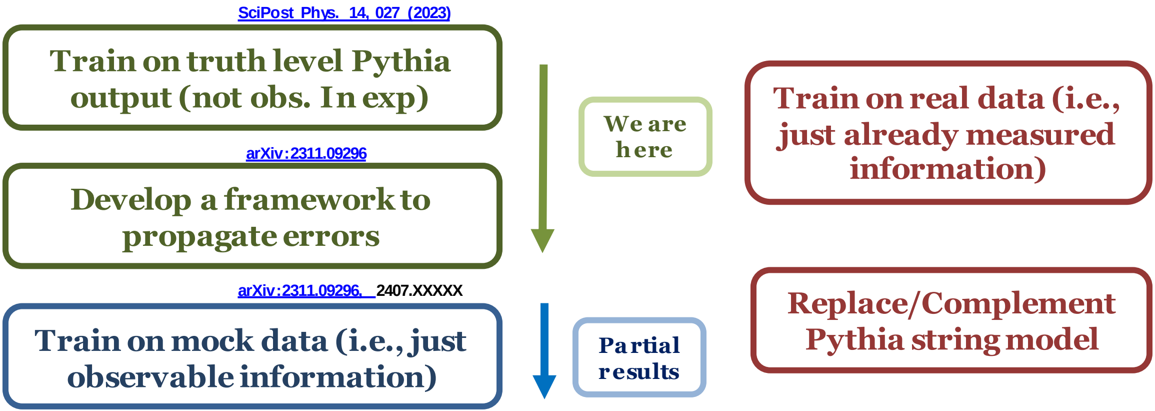
no ve

Use ML!

MLHAD

Big Picture

A series of progressive steps needs to be done before practically useful in Pythia simulations



SciPost Phys. 14, 027 (2023)

Train on truth level Pythia output (not obs. In exp)

arXiv:2311.09296

Develop a framework to propagate errors

arXiv:2311.09296, 2407.XXXXX

Train on mock data (i.e., just observable information)

We are here

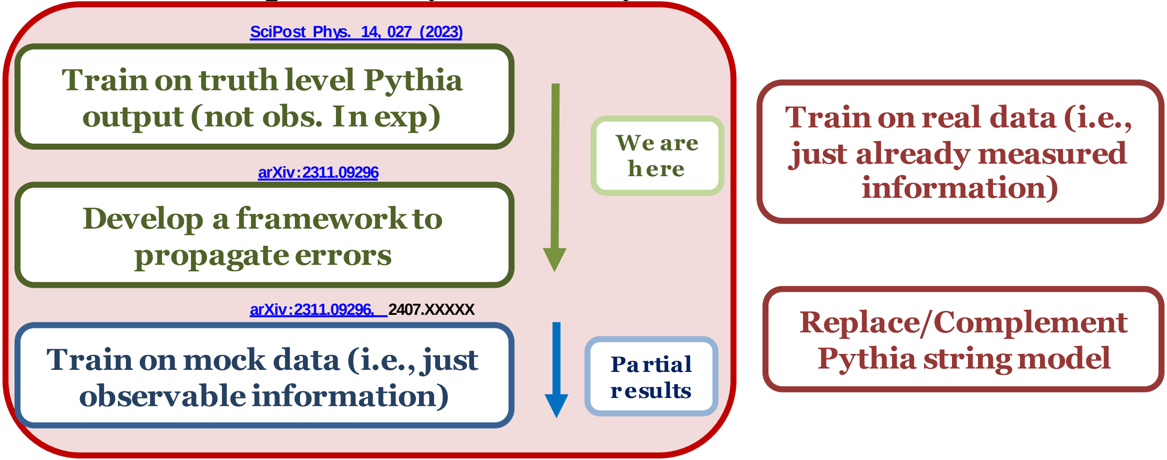
Partial results

Train on real data (i.e., just already measured information)

Replace/Complement Pythia string model

Big Picture

A series of progressive steps needs to be done before practically useful in Pythia simulations



SciPost Phys. 14, 027 (2023)

Train on truth level Pythia output (not obs. In exp)

arXiv:2311.09296

Develop a framework to propagate errors

We are here

arXiv:2311.09296, 2407.XXXXX

Train on mock data (i.e., just observable information)

Partial results

Train on real data (i.e., just already measured information)

Replace/Complement Pythia string model

Hadronization Models

Pythia **String model**

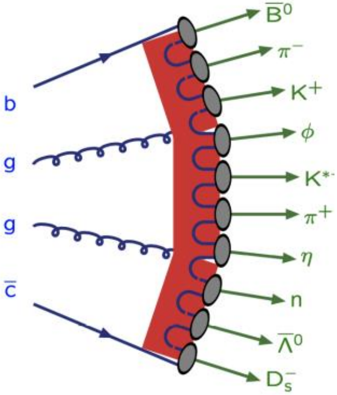


Fig from Vitev, IV YR workshop, 2020

Hadronization Models

Pythia **String model**

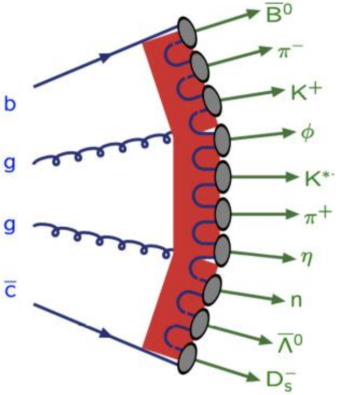
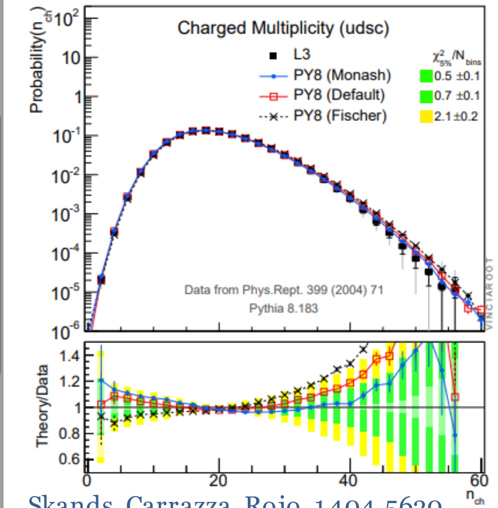


Fig from Vitev, IV YR workshop, 2020

→ Observables



Skands, Carrazza, Rojo, 1404.5630

Hadronization Models

Pythia

String model

Not Observed!

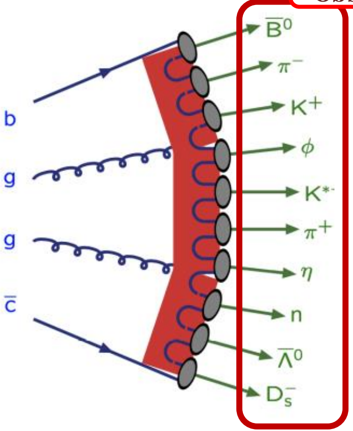
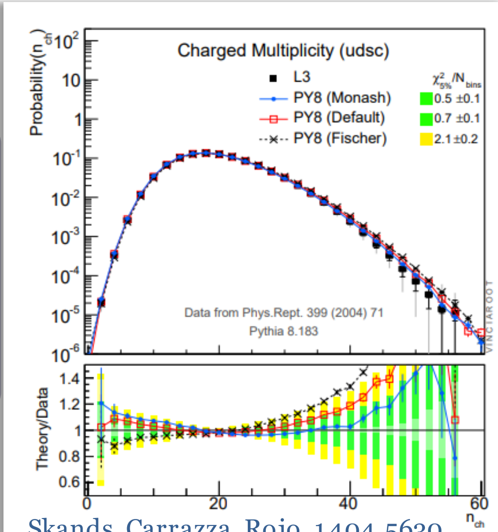


Fig from Vitev, IV YR workshop, 2020

Observables



Data from Phys.Rept. 399 (2004) 71

Pythia 8.183

Skands, Carrazza, Rojo, 1404.5630

MAGIC

(**M**icroscopic **A**lterations **G**enerated from **IR** **C**ollections)

Step 1
Train a Base (B) Model to reproduce Pythia

Step 2
Fine Tune (FT) the B Model on Observables

→ Only access to hadron level information

→ Only access to observables

MAGIC

(Microscopic Alterations Generated from IR Collections)

Step 1
Train a Base (B) Model to reproduce Pythia

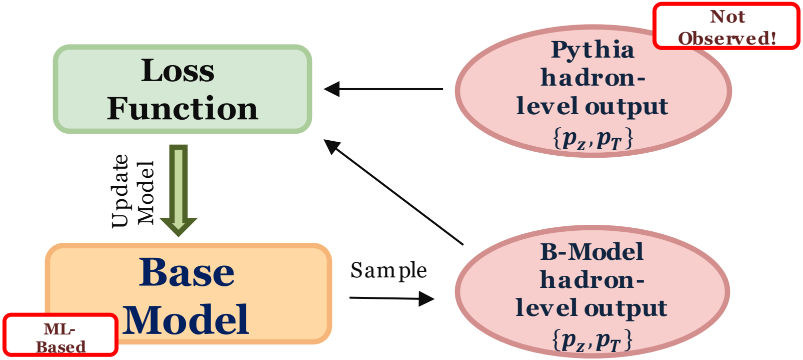
→ Only access to hadron level information

Step 2
Fine Tune (FT) the B Model on Observables

→ Only access to observables

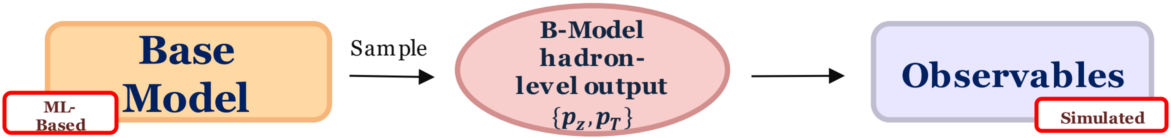
Further Directions

Step 1: Train Base (B) - Model on Pythia generated hadron-level output


Loss Function
Update Model
Base Model
ML-Based
Sample
Pythia hadron-level output $\{p_z, p_T\}$
Not Observed!
B-Model hadron-level output $\{p_z, p_T\}$

Further Directions

Step 1: Train Base (B) - Model on Pythia generated hadron-level output



Base Model

ML-Based

Sample

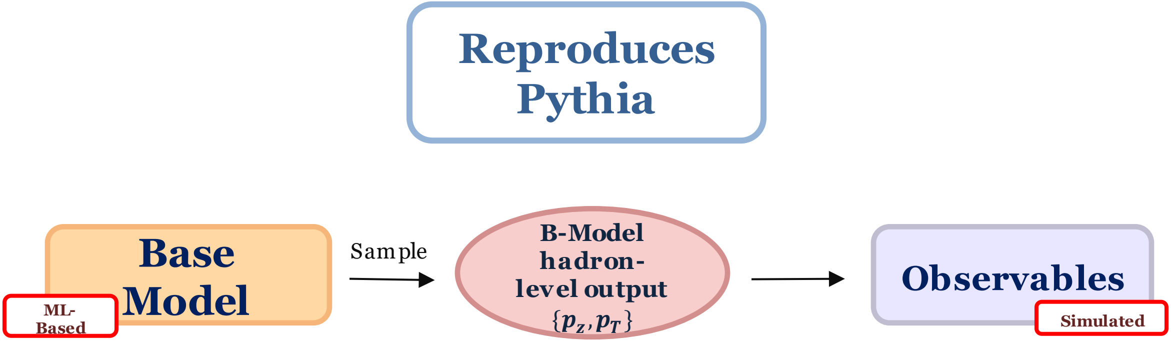
B-Model hadron-level output $\{p_z, p_T\}$

Observables

Simulated

Further Directions

Step 1: Train Base (B) - Model on Pythia generated hadron-level output



Reproduces Pythia

Base Model

ML-Based

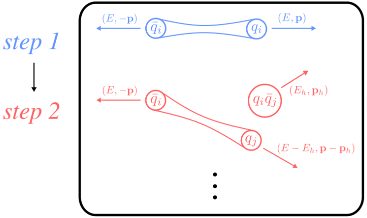
Sample

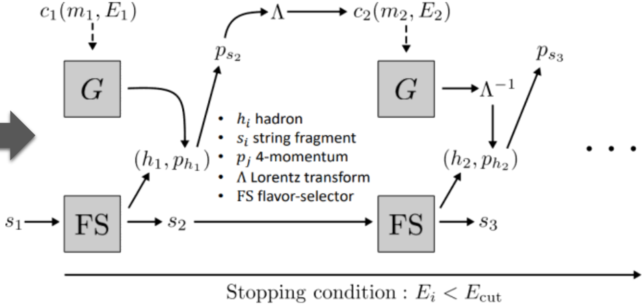
B-Model hadron-level output $\{p_z, p_T\}$

Observables

Simulated

MLHAD Pipeline





- h_i hadron
- s_i string fragment
- p_j 4-momentum
- Λ Lorentz transform
- FS flavor-selector

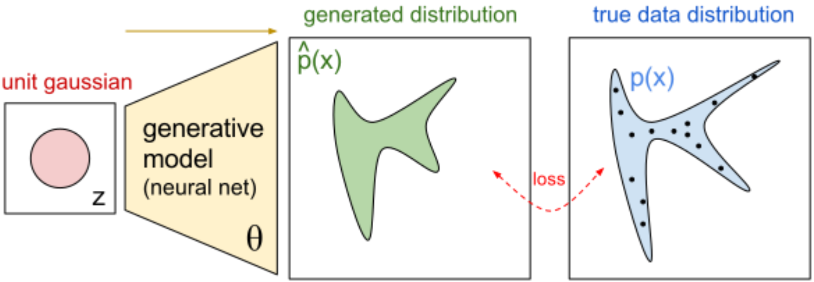
We need a generative model!

Sample hadron kinematics:
Train on $\{p_z, p_T\}$

Emission of different Mesons:
Condition on mass (m) and energy (E)

Generative Models

https://openai.com/research/generative-models



Source: generative models

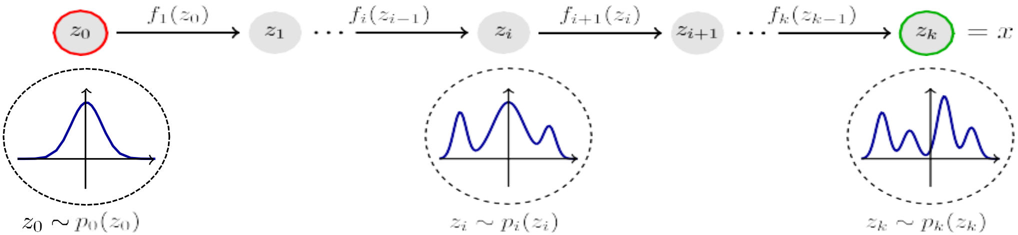
$\Longrightarrow$ Task: Learn the probability distribution $p(x)$ of the data

Which generative model should we choose?

Is it able to learn **complex distributions**?

Do we have access to the **exact probability distribution**?

Normalizing Flows



Zo- random vector sampled from a Gaussian $p_0(z_0)$

Fi – invertible NN that transforms $p_0(z_0)$ to $p_i(z_i)$ by change of variables

Complex target distribution $p_k(z_k)$ is learned

⇒ **Can learn complex distributions!**

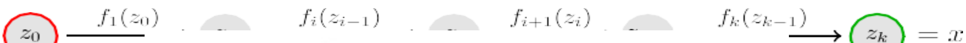
Exact probability distribution is obtained by change of variables

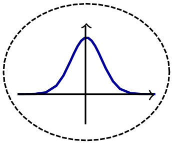
$$p_k(z_k) = p_0(z_0) \prod_{i=1}^{K} \left|\det\left(\frac{\partial f_i(z_{i-1})}{\partial z_{i-1}}\right)\right|^{-1}$$

hHps://github.com/janosh/awesome-normalizing-flows

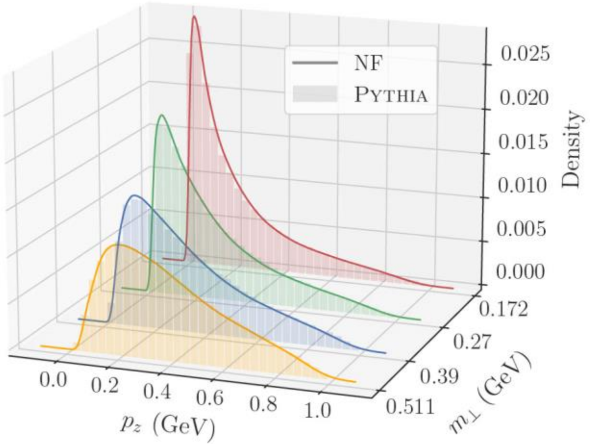
⇒ **Access to the exact probability distribution**

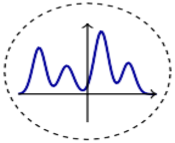
Normalizing Flows





$z_0 \sim p_0(z_0)$





$z_k \sim p_k(z_k)$

Z0- random vector sampled from a Gaussian $p_0(z_0)$

omplex target distribution $p_k(z_k)$ is learned

learn complex distributions!

Exact p obtain

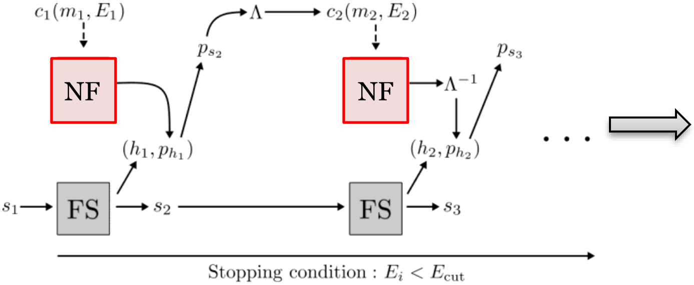
(z_{i-1}) z_{i-1} $\Big) \Big|^{-1}$

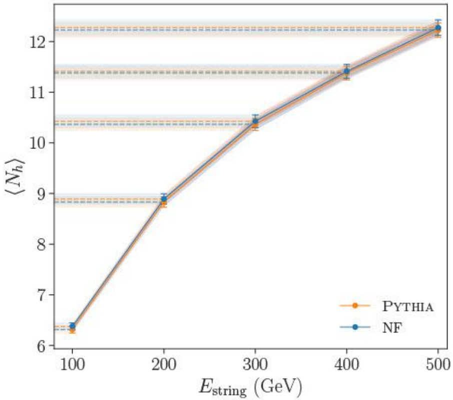
hHps://github.com/janosh/awesome-normalizing-flows

⇒Access to the exact probability distribution

Back to Physics

Implement NF in the fragmentation chain to obtain physical observables





$\Longrightarrow$ ***Multiplicity obtained by MLHad agrees with Pythia!***

MAGIC

(**M**icroscopic **A**lterations **G**enerated from **IR** **C**ollections)

Step 1
Train a Base (B) Model to reproduce Pythia

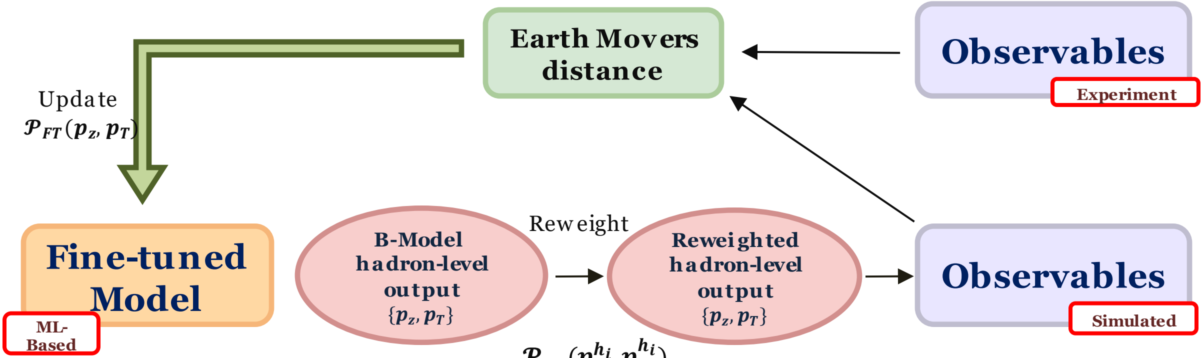
→ Only access to hadron level information

Step 2
Fine Tune (FT) the B Model on Observables

→ Only access to observables

Further Directions

Step 2: Fine-tune B-Model on physical observables



Earth Movers distance

Observables

Experiment

Update $\mathcal{P}_{FT}(p_z, p_T)$

Fine-tuned Model

ML-Based

B-Model hadron-level output $\{p_z, p_T\}$

Reweight

Reweighted hadron-level output $\{p_z, p_T\}$

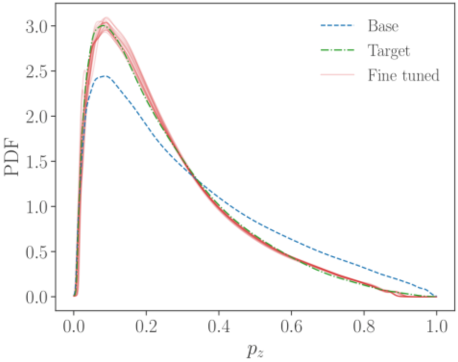
Observables

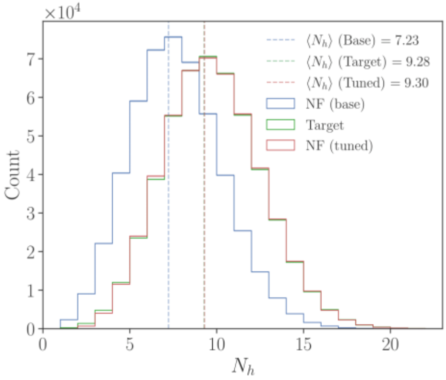
Simulated

$$\omega_i = \frac{\mathcal{P}_{FT}(p_z^{h_i}, p_T^{h_i})}{\mathcal{P}_B(p_z^{h_i}, p_T^{h_i})}$$

Results

Base: Pythia default parameters
Target: Pythia perturbed; aLund=1.5





Conclusion and Outlook

MAGIC is a very promising methods for data-driven hadronization models!

➔ Excellent results by training on only one observable (multiplicity)!

➔ More details on MAGIC and uncertainty quantification in **arXiv: 2311.09296**

More MLHAD work

- Reweighting Monte Carlo Predictions and Automated Fragmentation Variations in Pyhia 8 (arXiv:2308.13459)
- Pythia Flavor Reweigthing (arXiv:24NN.NNNNN)
- Collective Reweighting Method - two part reweighter (arXiv:2407.XXXXX)
- Tuning Hadronization Models

Project Homepage:
https://uchep.gitlab.io/mlhad-docs/

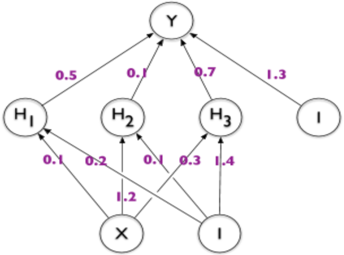
Backup

Uncertainty Quantification

Uncertainty estimation is crucial for event generator predictions!

Statistical (and Training) Uncertainties

„Classical“ Neural Networks

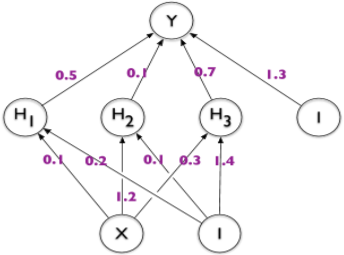


Weights have a fixed value
→ Weight values are updated in each epoch

(Image source: The very Basics of Bayesian Neural Networks)

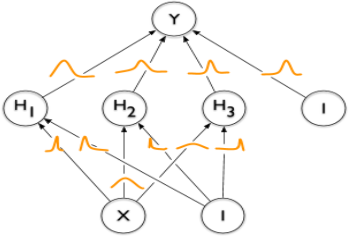
Statistical (and Training) Uncertainties

„Classical“ Neural Networks



Weights have a fixed value
→ Weight values are updated in each epoch

Bayesian Neural Networks (BNN)



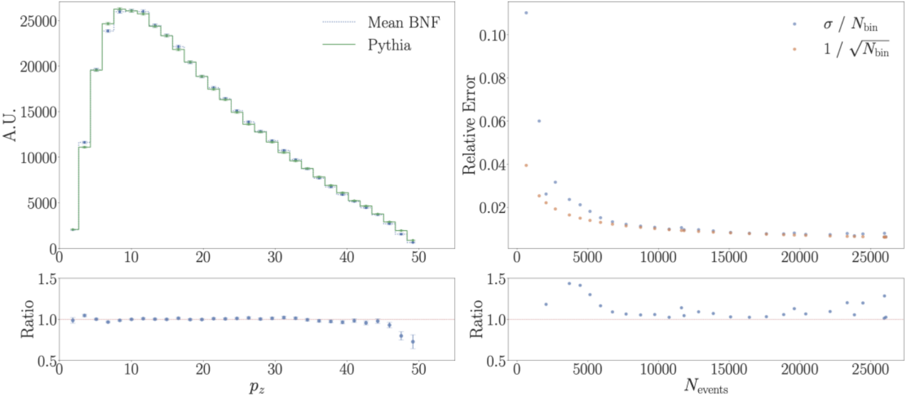
Weights are sampled from a distribution
→ Distribution parameter are updated in each epoch

→ **BNN are easy to implement: Add additional loss function for weight distribution**

→ **Capture statistical and training uncertainties**

(Image source: The very Basics of Bayesian Neural Networks)

Bayesian NF Results

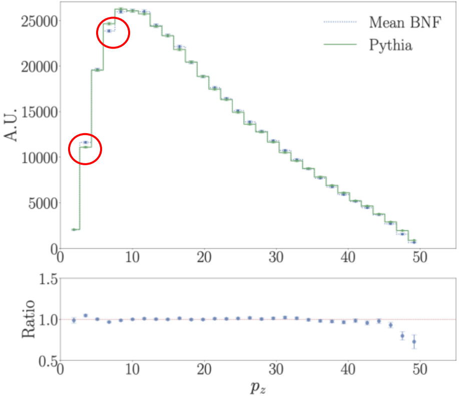


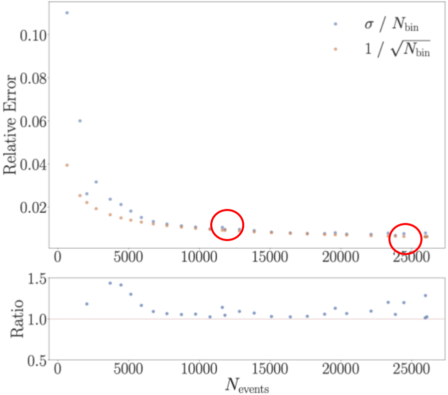
Pythia Sample: One sample with errors corresponding to $\sqrt{N_{bin}}$

Mean BNF: 5×10^5 samples with errors corresponding to the standard deviation

→ **BNF capture the statistical and training uncertainties**

Bayesian NF Results



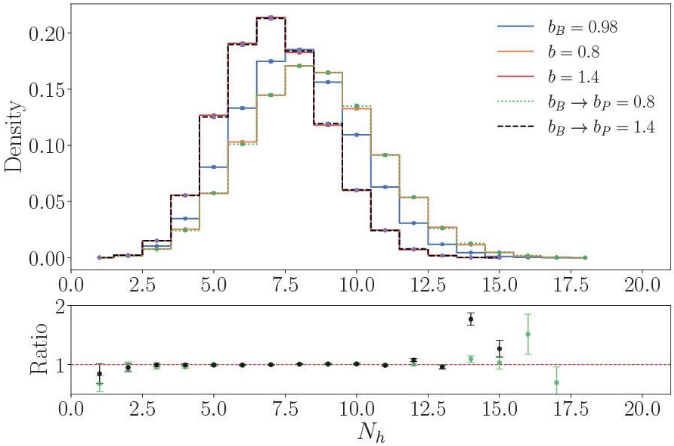


Pythia Sample:
One sample with errors corresponding to $\sqrt{N_{bin}}$

Mean BNF:
5×10^5 samples with errors corresponding to the standard deviation

BNF capture the statistical and training uncertainties

Reweighting with NFs



b is a free parameter in the Lund function used in Pythia: StringZ:bLund

Train nominal NF conditioned on different b
→ Get likelihood

→ Reweight nominal output using ratio of likelihoods:

$$w = \prod_i \frac{p^{(i)}_{nom}(z)}{p^{(i)}_{pert}(z)}$$

Motivation

When is a hadronization model successful?

Motivation

When is a hadronization model successful?

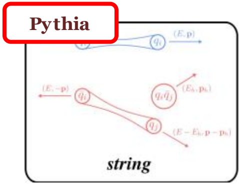
➡ The performance is judged by their description of experimental measurements!

Motivation

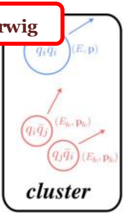
When is a hadronization model successful?

- The performance is judged by their description of experimental measurements!

Pythia



Herwig



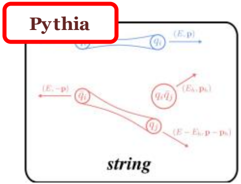
Phenomenological Models (String, Cluster) are currently state of art and are overall very successful, however:

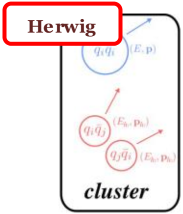
- comparison of data from proton-proton and ion-ion collision with Pythia
  - discrepancies at the level of O(20%) to O(50%) N. Fischer and T. Sj¨ostrand, JHEP 01, 140 (2017), 1610.09818
- recovering collective effects can be challenging, for instance, heavy baryon production at high event multiplicities Alice Collaboration, arXiv: 1807.11321
- no efficient estimation of Uncertainties

Motivation

When is a hadronization model successful?

→ The performance is judged by their description of experimental measurements!


Pythia


Herwig

Phenomenological Models (String, Cluster) are currently state of art and are overall very successful, however:

→ comparison of data from proton-proton and ion-ion collision with Pythia

→ discrepancies at the level of O(20%) to O(50%) N. Fischer and T. Sj¨ostrand, [JHEP 01, 140 (2017), 1610.09818]

→ recovering collective effects can be challenging, for instance, heavy baryon production at high event multiplicities Alice Collaboration, [arXiv: 1807.11321]

→ no efficient estimation of Uncertainties

→ **Both models have a discrepancy in describing experimental measurements!**

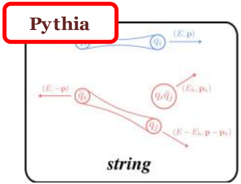
Motivation

When is a hadronization model successful?

- The performance is judged by their description of experimental measurements!

Phenomenological Models (String, Cluster) are currently state of art and are overall very successful, however:

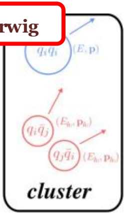
Pythia



- comparison of data from proton-proton and ion-io on with Pythia
  - discrepancies at the level of O(20%) to O er and T. Sj¨ostrand, P 01, 140 (2017), 1610.09818.
- recovering collective effects ca ng, for instance, heavy baryon production at high event Alice Collaboration, arXiv: 1807.11321
- no efficient esti ertainties

We need an innovative approach!

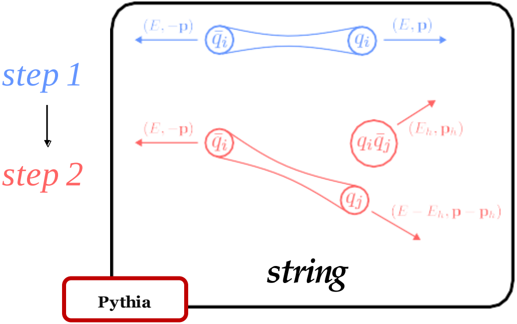
Herwig

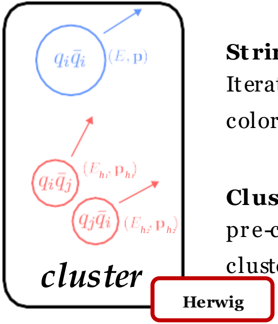


Both models have a discrepancy in describing experimental measurements!

Hadronization Models

Two primary hadronization models are used





String model:
Iteratively split parton connected by QCD color strings with linear potential

Cluster model:
pre-confine partons into proto-clusters, then split by two-body decays

MLhad: Ilten, Menzo,Youssef, Zupan, 2203.04983,
https://gitlab.com/uchep/mlhad

HadML: (Chan, Ghosh,) Ju, (Kania), Nachman, (Sangli,) Siodmok, 2203.12660, 2305.17169

Uncertainty Quantification

Uncertainty estimation is crucial for event generator predictions!

Uncertainty Quantification

Uncertainty estimation is crucial for event generator predictions!

- **Hard matrix element**
- **Parton shower**

Efficient solutions exist!

perturbative calculations depend on choices of scale, values of gauge and other couplings, particle masses, and nonperturbative inputs

Giele et al, [Phys. Rev. D84, 054003 (2011)]

S. Mrenna and P. Skands, [Phys. Rev. D94(7), 074005 (2016)]

Uncertainty Quantification

Uncertainty estimation is crucial for event generator predictions!

→ **Hard matrix element**

→ **Parton shower**

Efficient solutions exist!

perturbative calculations depend on choices of scale, values of gauge and other couplings, particle masses, and nonperturbative inputs

Giele et al, [Phys. Rev. D84, 054003 (2011)]

S. Mrenna and P. Skands, [Phys. Rev. D94(7), 074005 (2016)]

→ **Hadronization**

Efficient solution has remained elusive!

Standard procedure: perform repeated simulations with different sets of values for the model parameters

→ **Computationally very expensive!**

Uncertainty Quantification

Uncertainty estimation is crucial for event generator predictions!

- Hard matrix element
- Parton shower

Efficient solutions exist!

perturbative calculations depend on choices of scale, values of gauge and other couplings, particle masses, and nonperturbative inputs

Giele et al, Phys. Rev. D84, 054003 (2011)
S. Mrenna and P. Skands, Phys. Rev. D94(7), 074005 (2016)

- Hadronization

Efficient solution has rem ve!

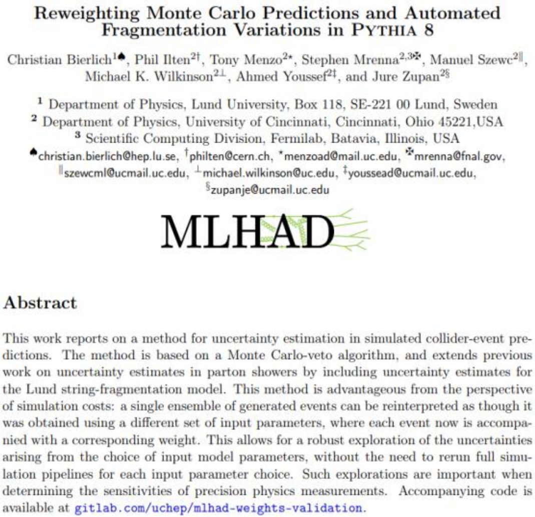
Standard procedure ated simulations with different s or the model parameters

→ putationally very expensive!

Need a more efficient way!

Reweighting Hadronized Pythia Events

**Small Detour:
No ML, only Had**



Reweighting Monte Carlo Predictions and Automated Fragmentation Variations in PYTHIA 8

Christian Bierlich[1♠], Phil Ilten[2†], Tony Menzo[2⋆], Stephen Mrenna[2,3✠], Manuel Szewc[2‖], Michael K. Wilkinson[2⊥], Ahmed Youssef[2‡], and Jure Zupan[2§]

[1] Department of Physics, Lund University, Box 118, SE-221 00 Lund, Sweden
[2] Department of Physics, University of Cincinnati, Cincinnati, Ohio 45221,USA
[3] Scientific Computing Division, Fermilab, Batavia, Illinois, USA

[♠]christian.bierlich@hep.lu.se, [†]philten@cern.ch, [⋆]menzoad@mail.uc.edu, [✠]mrenna@fnal.gov, [‖]szewcml@ucmail.uc.edu, [⊥]michael.wilkinson@uc.edu, [‡]youssead@ucmail.uc.edu, [§]zupanje@ucmail.uc.edu

MLHAD

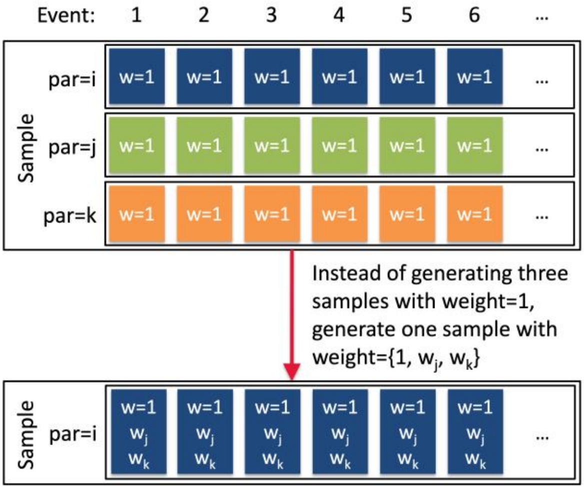
Abstract

This work reports on a method for uncertainty estimation in simulated collider-event predictions. The method is based on a Monte Carlo-veto algorithm, and extends previous work on uncertainty estimates in parton showers by including uncertainty estimates for the Lund string-fragmentation model. This method is advantageous from the perspective of simulation costs: a single ensemble of generated events can be reinterpreted as though it was obtained using a different set of input parameters, where each event now is accompanied with a corresponding weight. This allows for a robust exploration of the uncertainties arising from the choice of input model parameters, without the need to rerun full simulation pipelines for each input parameter choice. Such explorations are important when determining the sensitivities of precision physics measurements. Accompanying code is available at gitlab.com/uchep/mlhad-weights-validation.

Reweighting Hadronized Pythia Events

- **Event generation is time consuming**
  - **We want to reweight events without regenerating**
- **Use a modified veto algorithm**
  - **New event weights for different hadronization param are book kept**

- **We calculate event weights for different hadronization options in a single event generation!**



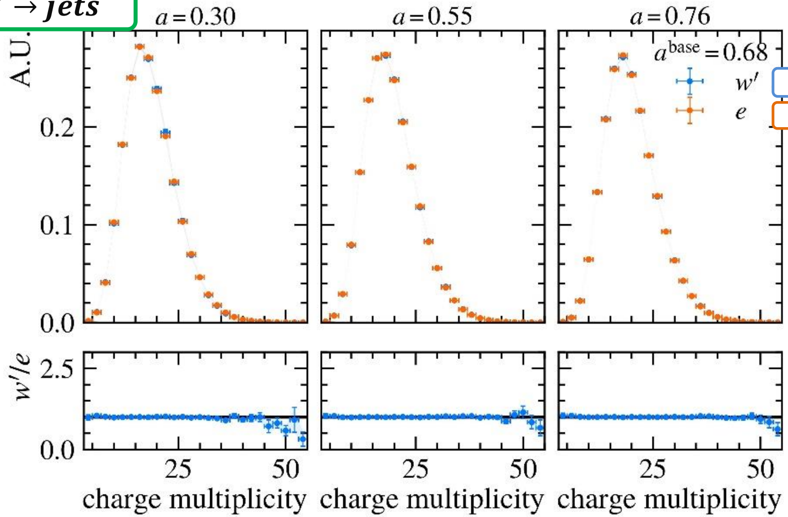
Sample	Event:	1	2	3	4	5	6	...
	par=i	w=1	w=1	w=1	w=1	w=1	w=1	...
	par=j	w=1	w=1	w=1	w=1	w=1	w=1	...
	par=k	w=1	w=1	w=1	w=1	w=1	w=1	...

Instead of generating three samples with weight=1, generate one sample with weight={1, w_j, w_k}

Sample							
par=i	w=1 w_j w_k	w=1 w_j w_k	w=1 w_j w_k	w=1 w_j w_k	w=1 w_j w_k	w=1 w_j w_k	...

Reweighting Hadronized Pythia Events

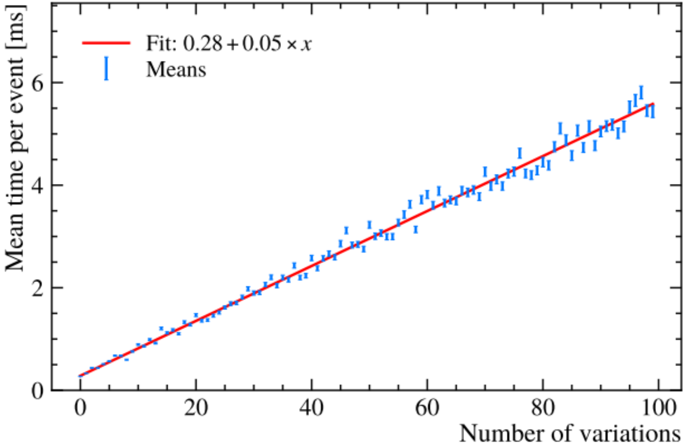
$e^+e^- \to Z \to jets$



reweighted

exact calc.

Reweighting Hadronized Pythia Events

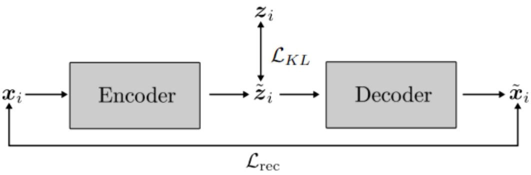


- Generate 100 samples with different variations of aLund
- Each sample has 1000 events
- Cost per additional parameter variation is around 0.05 ms
- We have a speed up by a factor ~3

Generative Models

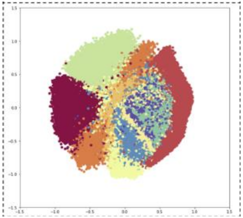
Variational Autoencoder (VAE)

Kingma et al, arXiv:1312.6114



Vanilla VAE

KL-divergence limits the latent space to a simple analytic distribution



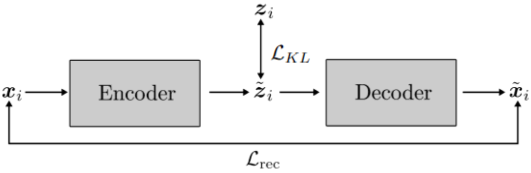
VAE latent space

arXiv: 1804.01947

Generative Models

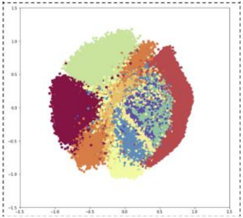
Variational Autoencoder (VAE)

Kingma et al, arXiv:1312.6114



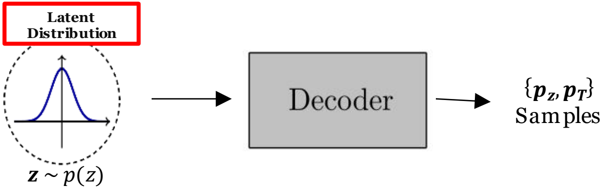
Vanilla VAE

KL-divergence limits the latent space to a simple analytic distribution



VAE latent space
arXiv: 1804.01947

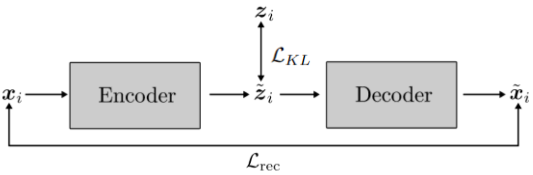
Inference



Generative Models

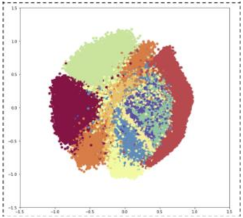
Variational Autoencoder (VAE)

Kingma et al, arXiv:1312.6114



Vanilla VAE

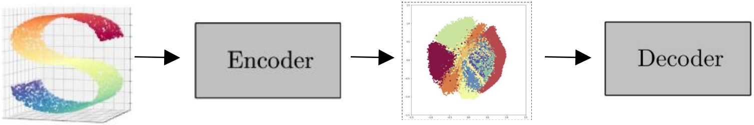
KL-divergence limits the latent space to a simple analytic distribution



VAE latent space
arXiv: 1804.01947

Complex input data

Simple latent space

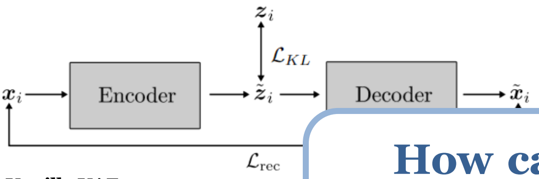


⇒ Complex distribution are hard to learn!

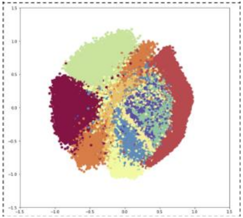
Generative Models

Variational Autoencoder (VAE)

Kingma et al, arXiv:1312.6114



KL-divergence limits the latent space to a simple analytic distribution

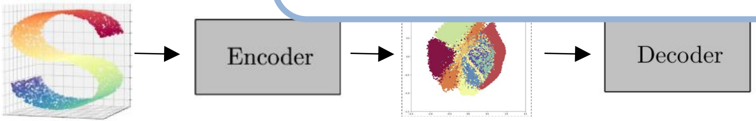


VAE latent space
arXiv: 1804.01947

Vanilla VAE

How can we make VAEs learn more complex distribution?

Complex input data



⇒ Complex distribution are hard to learn!

Generative Models

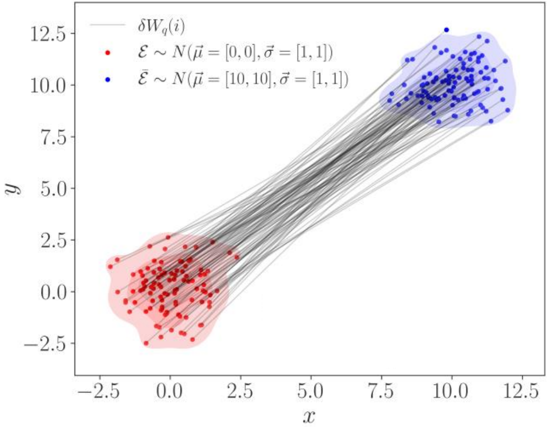
Use Sliced Wasserstein Distance as latent loss function!

Generative Models

Use Sliced Wasserstein Distance as latent loss function!

Wasserstein distance (WD)

$$W_q(\mathcal{E},\bar{\mathcal{E}}) = \left[\min_{\{f_{ij}\geq 0\}} \sum_{i=1}^{N}\sum_{j=1}^{\bar{N}} f_{ij}\left(\hat{d}_{ij}\right)^q\right]^{1/q}$$



Generative Models

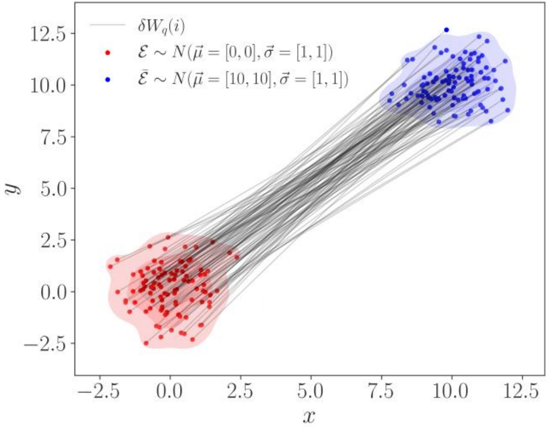
Use Sliced Wasserstein Distance as latent loss function!

Wasserstein distance (WD)

$$W_q(\mathcal{E},\bar{\mathcal{E}}) = \left[\min_{\{f_{ij}\geq 0\}} \sum_{i=1}^{N}\sum_{j=1}^{\bar{N}} f_{ij}\left(\hat{d}_{ij}\right)^q\right]^{1/q}$$

Sliced Wasserstein distance

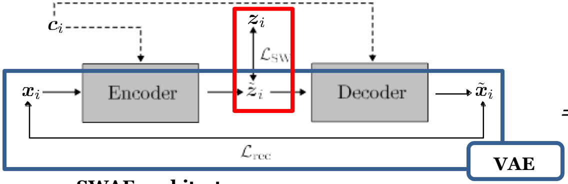
- Projects high dimensional data into one dimensional “slices”
- WD in 1D has a closed form solution
  - Sorted Difference of the two samples



Generative Models

[SciPost Phys. 14, 027 (2023)](#) **Conditional Sliced Wasserstein (SW) Autoencoder (cSWAE)**

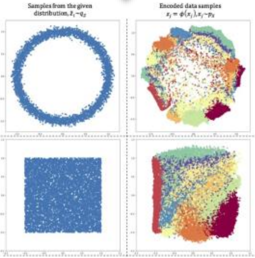
Restricted to Pion emissions



cSWAE architecture
(Architecture used in [SciPost Phys. 14, 027 (2023)](#))

SW distance enables learning any sampleable latent distribution

⇒ **Can learn complex distributions!**

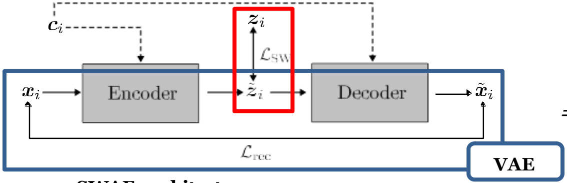


SWAE latent space
(arXiv: 1804.01947)

Generative Models

[SciPost Phys. 14, 027 (2023)] **Conditional Sliced Wasserstein (SW) Autoencoder (cSWAE)**

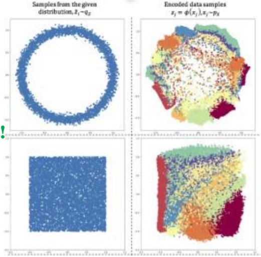
Restricted to Pion emissions



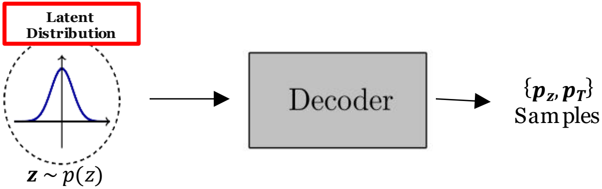
cSWAE architecture
(Architecture used in SciPost Phys. 14, 027 (2023))

SW distance enables learning any sampleable latent distribution

⇒ **Can learn complex distributions!**



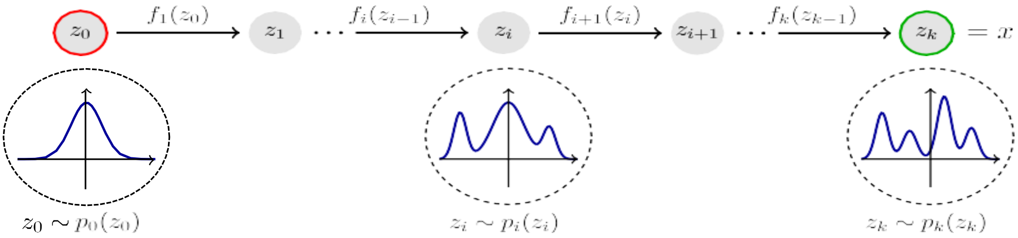
SWAE latent space
(arXiv: 1804.01947)



Decoder "just" generates samples

⇒ **No access to the probability distribution**

Normalizing Flows



Zo- random vector sampled from a Gaussian $p_0(z_0)$

Fi – invertible NN that transforms $p_0(z_0)$ to$p_i(z_i)$ by change of variables

Complex target distribution $p_k(z_k)$ is learned

⇒ **Can learn complex distributions!**

Exact probability distribution is obtained by change of variables

$$p_k(z_k) = p_0(z_0)\prod_{i=1}^{K}\left|\det\left(\frac{\partial f_i(z_{i-1})}{\partial z_{i-1}}\right)\right|^{-1}$$

hHps://github.com/janosh/awesome-normalizing-flows

⇒**Access to the exact probability distribution**

Removed pion emission restriction

Further Directions

- Propagation of errors
  - ML architecture with Bayesian Normalizing Flows (presented in part)
- Train on observables only
  - Two part reweighter (not part of the talk)
  - Train on global observables with Fine tuning (results not shown in this talk)
- To train on experimental data
  - Want fast evaluation of parameter dependency
  - Use reweighting method
  - First implementation in Pythia for Lund string model (to be released soon in Pythia)