Hadronization Models using Machine Learning

AI4EIC, 11/29

Manuel Szewc

University of CINCINNATI

MLHAD

In this talk

I hope to convey how **Hadronization** is a complicated and worthwhile target with a two part talk:

- Brief introduction to Hadronization.
- Efforts to apply ML to Hadronization

This talk is mostly based on arxiv:2203.04983, arxiv:2311.09296 and ongoing preliminary work.

Hadronization at colliders

Image from Pythia 8.3 manual

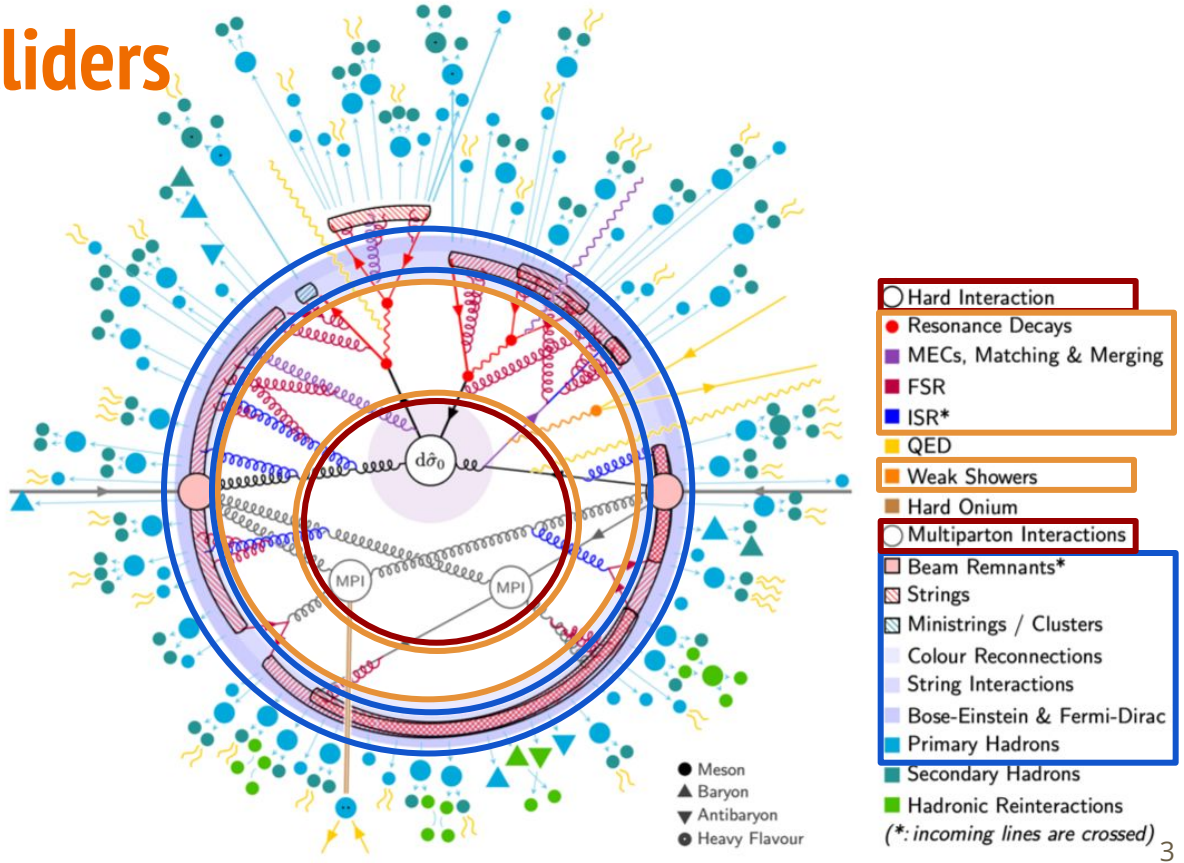
The radial coordinate is time

or 1/energy scale.

Hard process dσ, perturbatively calculated. Partons hidden to experiment.

Perturbative evolution from hard to hadronization scale, also perturbative. Hidden to experiment

Hadronization: combining partons into hadrons. Non perturbative. Hadrons can be measured



Modeling hadronization

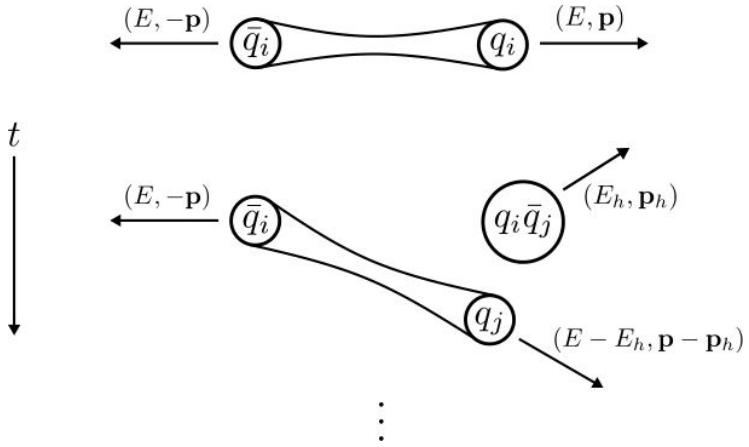
Hadronization is a inherently non-perturbative process → Empirical models for predictions.

Two main models: the **Lund String model (Pythia)** and the Cluster model (Herwig).

Lund String Model: Colored singlets + ~20 parameters → Hadrons

Each hadron is characterized by its four-momenta and its flavour. Translated into three variables of interest: **z, p_T, flavour.**

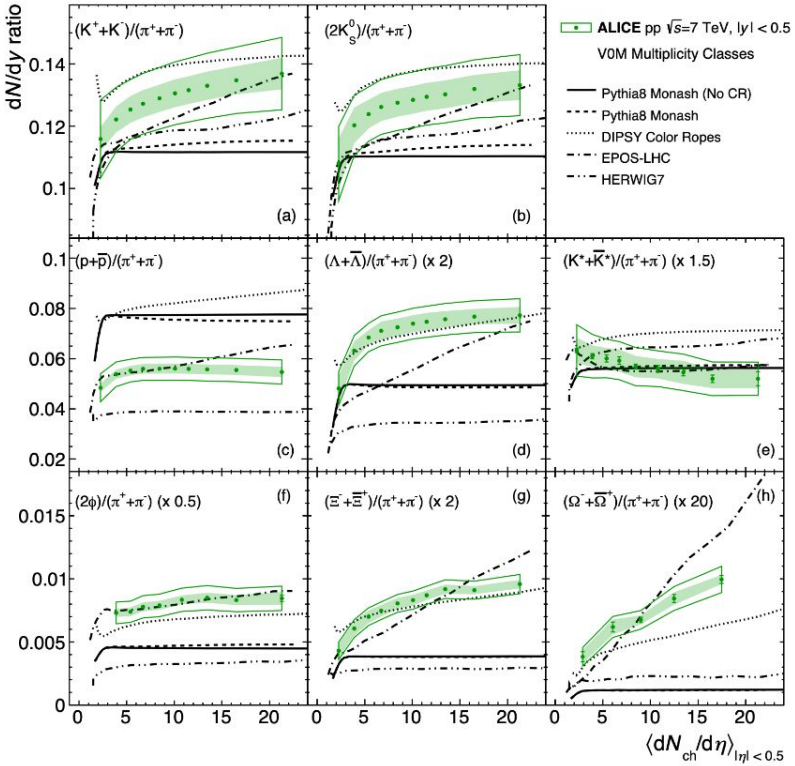
Simplified example from arxiv:2203.04983.



However...

Tuned Pythia is **very** successful. **However**, we are pushing the models to their limits.

Collective effects in general are tricky to recover e.g. heavy baryon production at high event multiplicities as in arxiv:1807.11321.



Machine Learning to the rescue?

Complex problem with **no full model flexible enough** and where **training is expensive**? → Machine Learning should be really useful here!

A lot of possible ways to attack this problem. The richness of the involved physics **forbids the use of any plug-and-play algorithms**.

Two groups have recently tackled the subject: **MLHAD** (arxiv:2203.04983, arxiv:2311.09296) and **HADML** (arxiv:2203.12660, arxiv:2305.17169). Different **generators** (Pythia, Herwig) and different **architectures** (cSWAE,BNF, GAN) with different **degrees of implementation**.

Learning the Lund Fragmentation model

First task: learn the **Lund String first hadronization pdfs** for $e^+e^- \to q\bar{q}$ at various energies. Checks feasibility of the problem.

Introduce **inductive bias**. Improve over the existing empirical model by first mapping it to a learnable model.

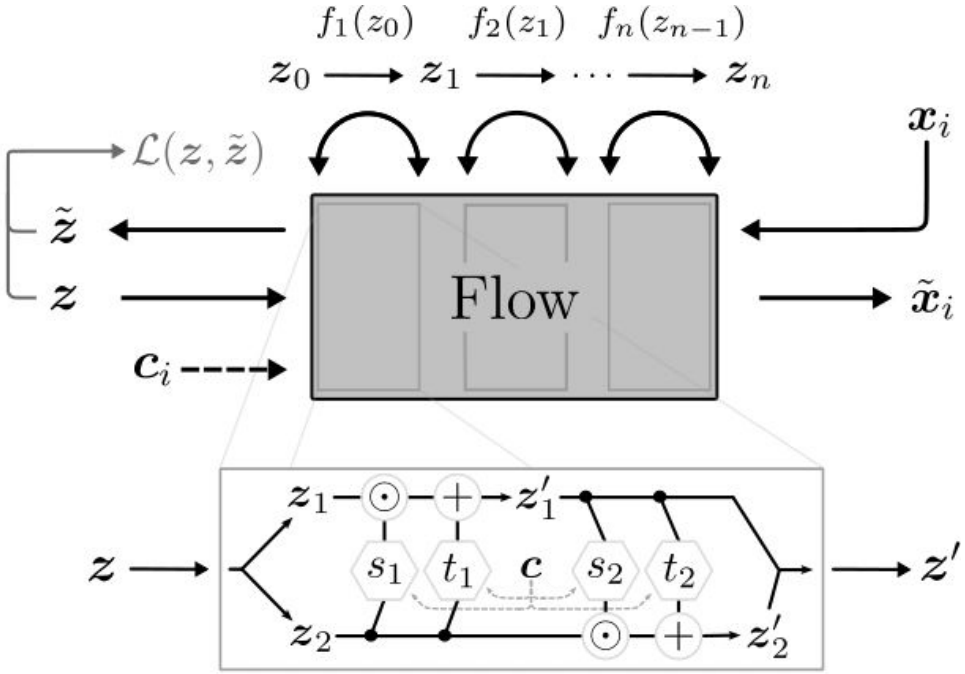
The first hadronization pdf can be iteratively applied to get a full chain.

$$p_{\mathrm{Lund}}(x) \to p_{\mathrm{ML}}(x) \qquad x = \{p_z, p_T\}$$

Normalizing Flows

A normalizing flow relates a variable **z** with known, simple pdf to the observed data **x**. If flexible enough, the flow allows to **sample** and **infer** with **exact likelihood**.

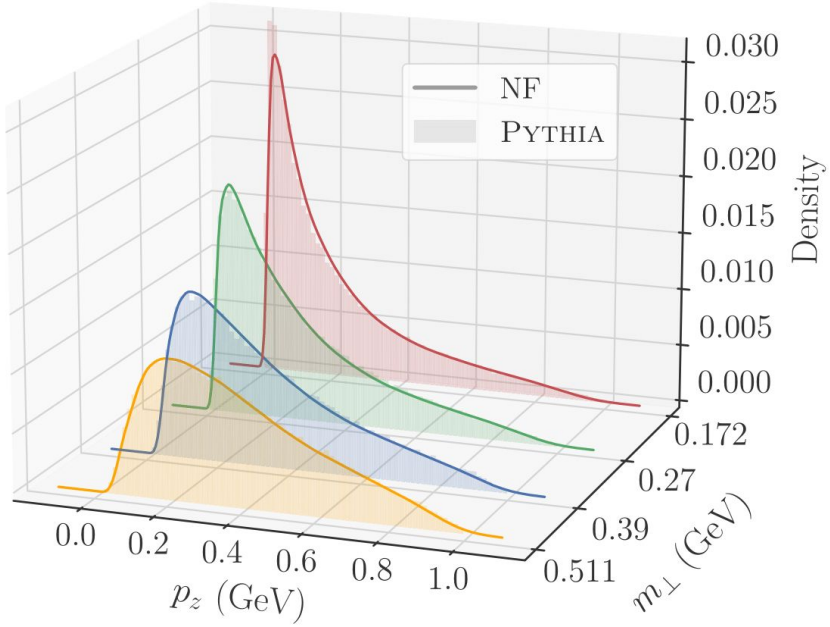
Our baseline choice of NF is a Masked Autoregressive Flow (**MAF)** with Real Non-Volume Preserving (**RealNVP**) transformations.



Normalizing Flows for the Lund fragmentation

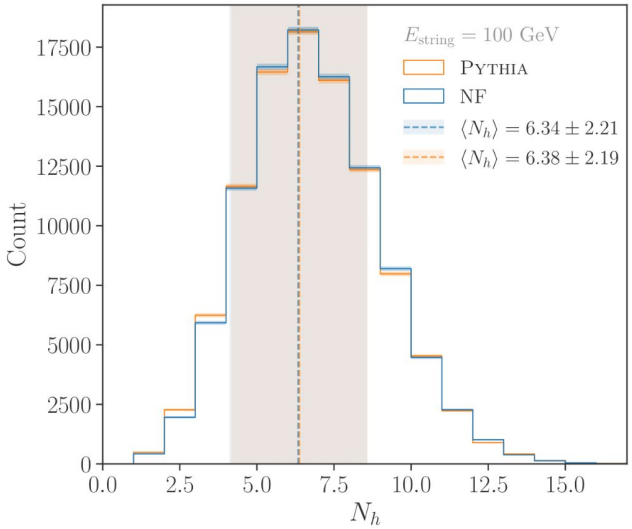
We train a NF on **first hadronization data with only pions** and learn the two-dimensional distribution while conditioning on the transverse mass m_T → adaptable to different flavors

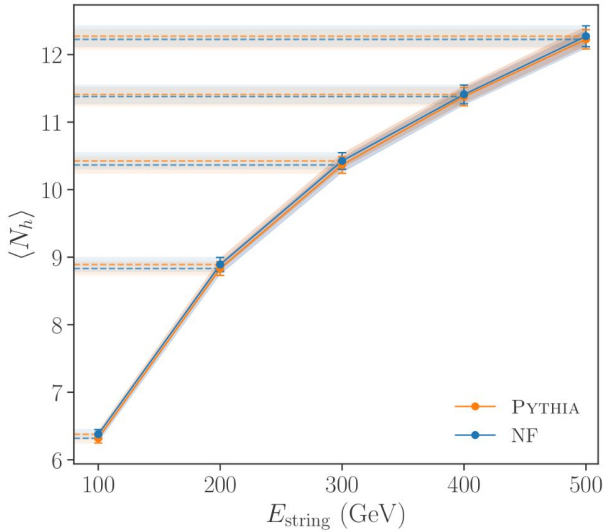
We show here how the **NF learns the appropriate Lund distribution** with its complicated relation between m_T and p_z.



Obtaining a full chain

Fragmentation chain of N_h successive hadronizations. We recover the probabilistic distribution of the chain length conditioned on the initial energy of the string.





Learning with uncertainties

Any hadronization model should be able to deal in different uncertainties:

- **Training** uncertainties due to **finite sample size of the training dataset** and the **bias of the model**
- **Data** uncertainties inherent to the training data, ie **systematic effects**

Even for the well established Lund String model, obtaining the uncertainties on the parameters and translating them to observables is **complicated**

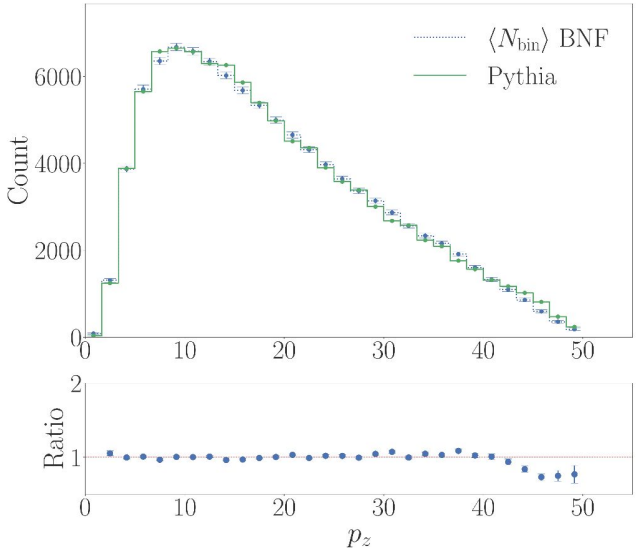
Can ML be a more natural way of working with uncertainties?

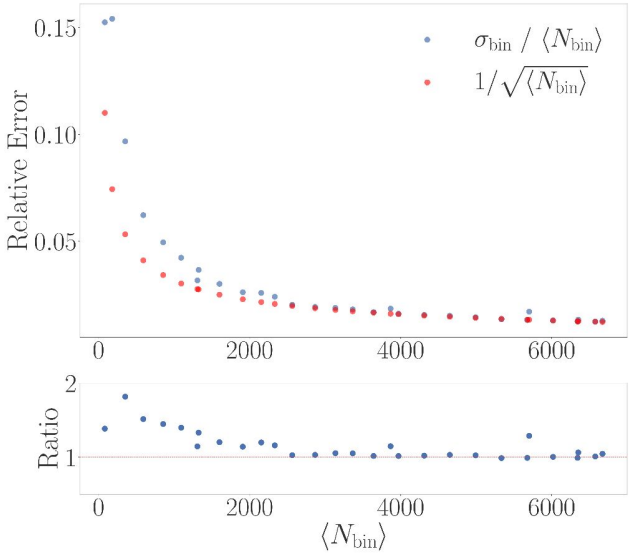
Bayesian Normalizing Flows

We deploy a BNF to capture the training error. We still train on first hadronization but now we capture **a family of distributions**

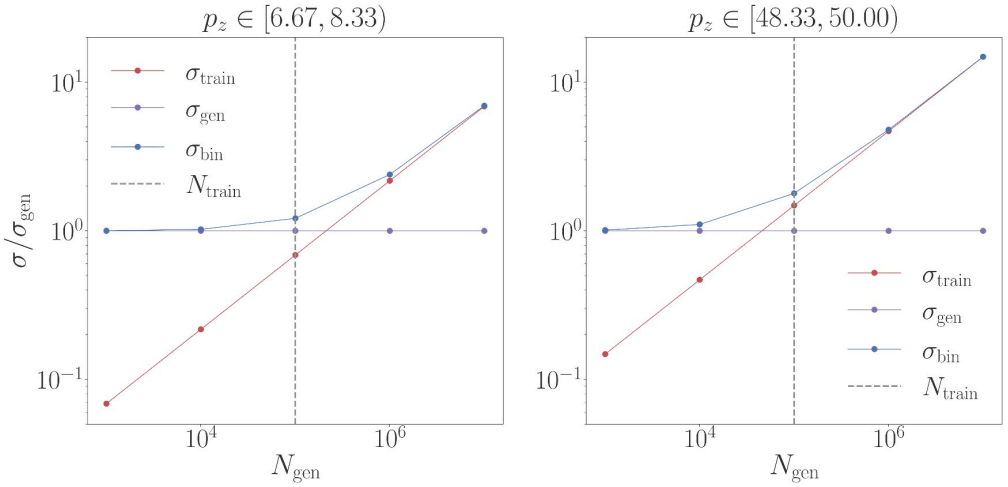
The BNF **captures the underlying distribution**.

The uncertainty is always **higher than the Poisson uncertainty** due to the additional **training error**.





Bayesian Normalizing Flows

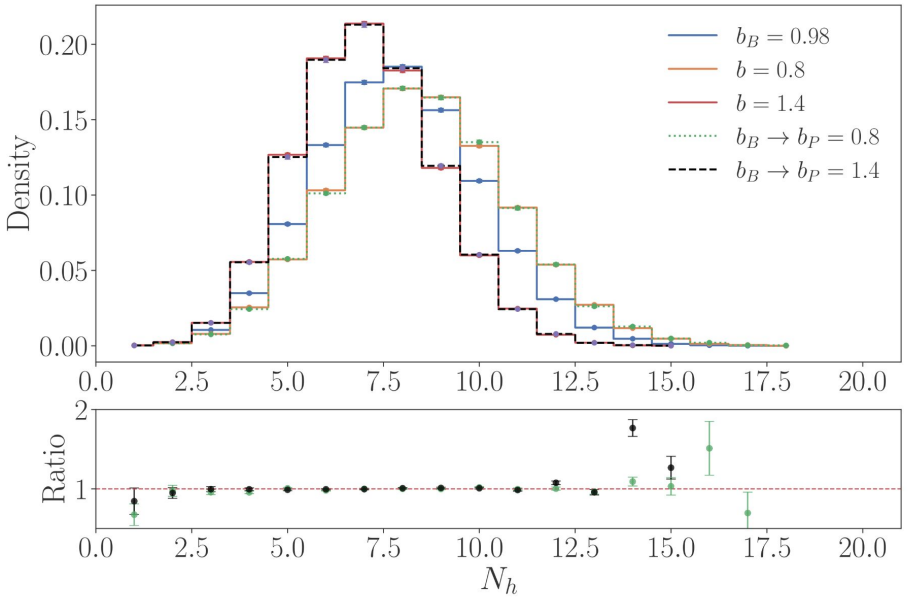


The **training uncertainty becomes the limiting factor** after a certain generated size (which depends on the quality of training).

Systematic Uncertainties

We capture systematic uncertainties by **conditioning during training**.

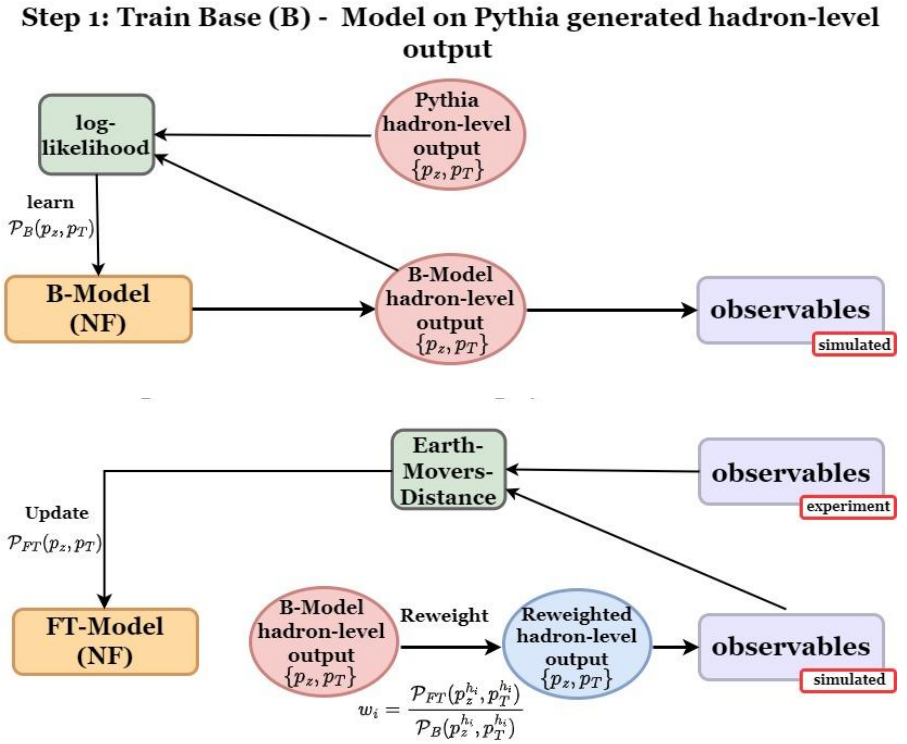
We **propagate** these uncertainties through **reweighting** using the **exact learned likelihood**.



Learning from data: MAGIC

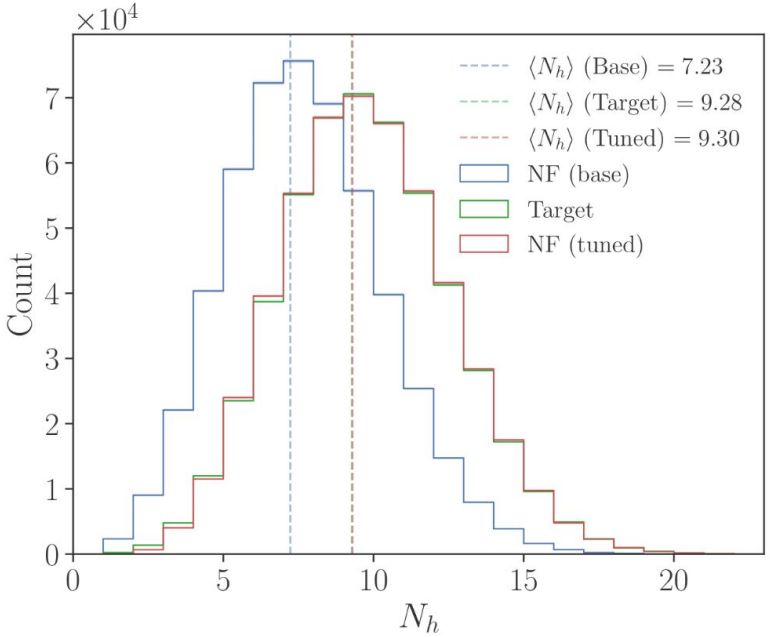
However, the end-goal is to learn a **better model** from data.

We propose a new strategy, called **M**icroscopic **A**lterations **G**enerated from **I**nfrared **C**ollections. It is a two-step procedure



MAGIC

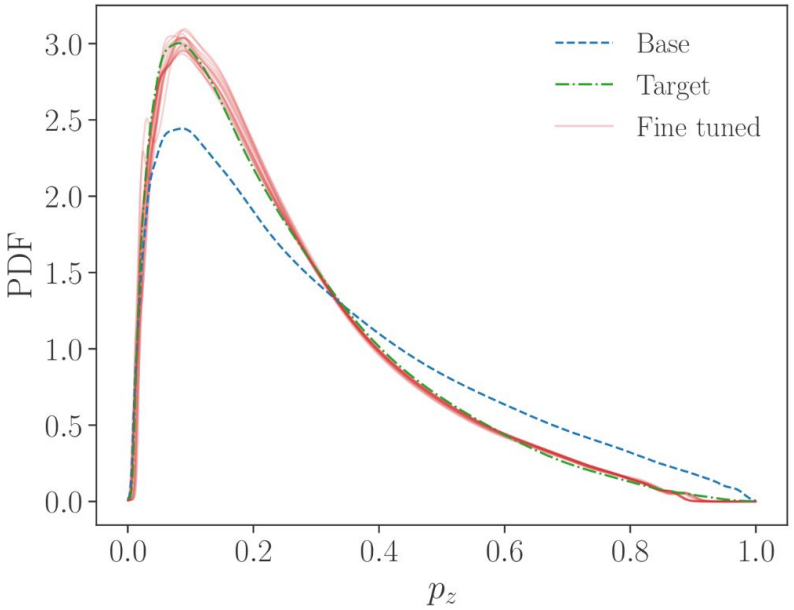
We compare the original learned model (**base**) with the **target** (here pseudo-data from Pythia with different parameters) and obtain a new model (**tuned**)



MAGIC

Because we are reweighting in terms of the underlying hadronizations, we obtain a **new hadronization model**!

Here we worked only with p_z. For $\{p_T, p_z\}$, we need more observables to break degeneracies.



Next steps:

We need to properly characterize how MAGIC scales with the **number of generated events** and the **dimensionality of the high-level observables**.

Also, we still need to think about incorporating **other string topologies and different physical constraints** that the model should obey.

We also need to **incorporate errors into MAGIC**. Hopefully this yields **better models** with **calibrated errors**.

Conclusions

Hadronization is the type of problem you dream of if you want to work in ML for HEP: **physically meaningful** and **complicated enough** that it is not simply a case of plug-and-play with any ML algorithm.

The variety of colour topologies and correlations between hadronizations pose a challenge to represent in an appropriate manner for any learnable algorithm.

Training itself is an issue! Developments in Simulation Based Inference and Nested Sampling could be really useful.

Any developments can also impact existing tuning efforts and uncertainty estimations (see supplemental material for an example)**.**

This is very much an open problem, so feedback is necessary!

The MLHAD team

Christian Bierlich, Phil Ilten, Tony Menzo, Stephen Mrenna, Manuel Szewc, Michael Wilkinson, Ahmed Youssef, and Jure Zupan





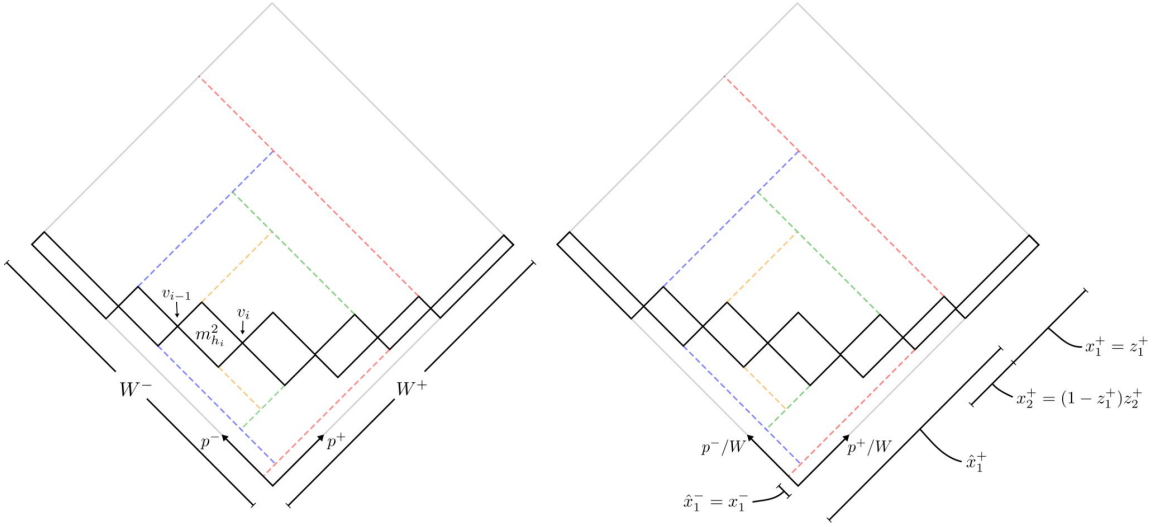




Thanks!
©QUINO

Backup slides

Momentum space for finding next hadronization



Limitations of the Lund String model

O(20%) to O(50%) discrepancies between proton-proton and ion-ion collisions

Heavy particle composition as a function of event multiplicity is mismodelled at high event multiplicities

Mismodelling of the mass dependence of the average transverse momentum

Minimum bias description can be incompatible because of low transverse momentum mismodelling

Ridge in pp collisions missing in Pythia (and in general long range correlations are hard to model)

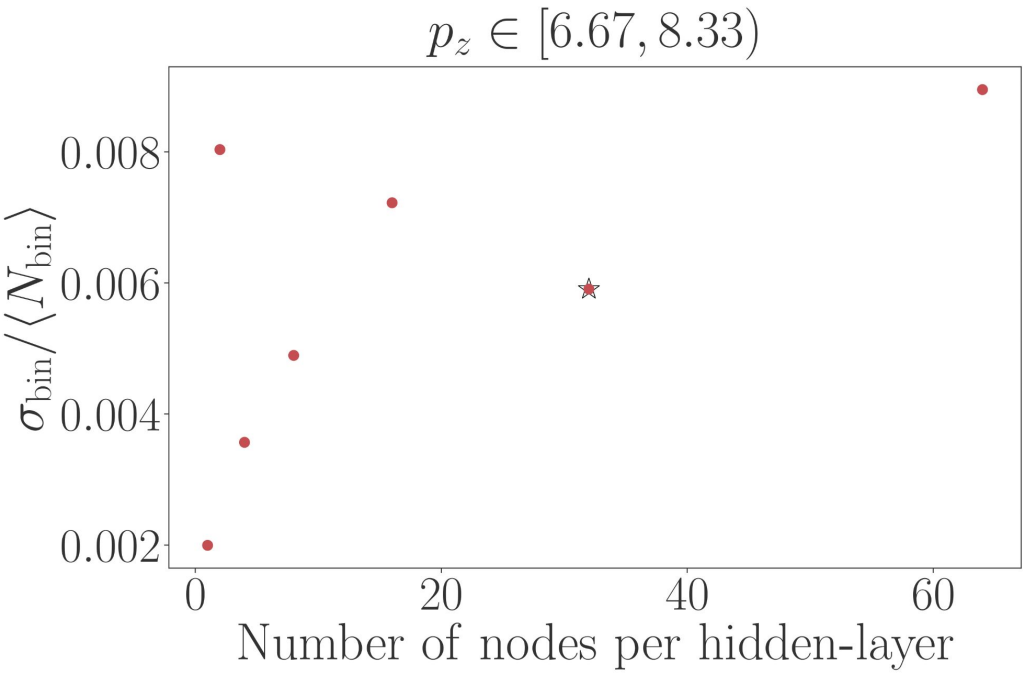
Charged particle multiplicity spectrum is very sensitive to color reconnections and MPI modelling

Impact of model architecture

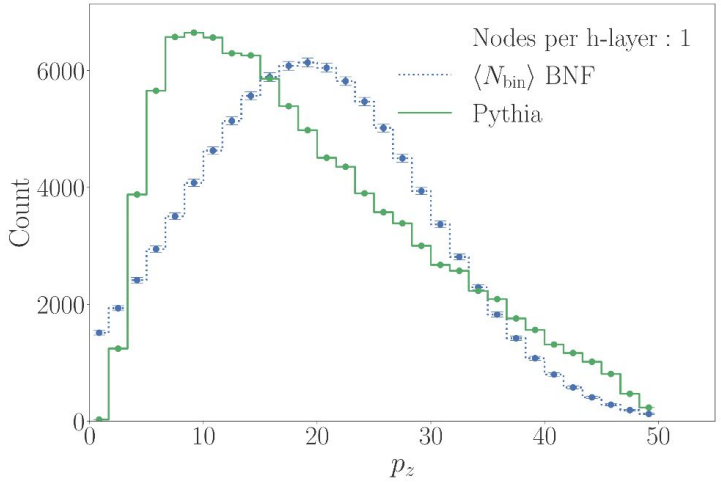
We scan over possible models.

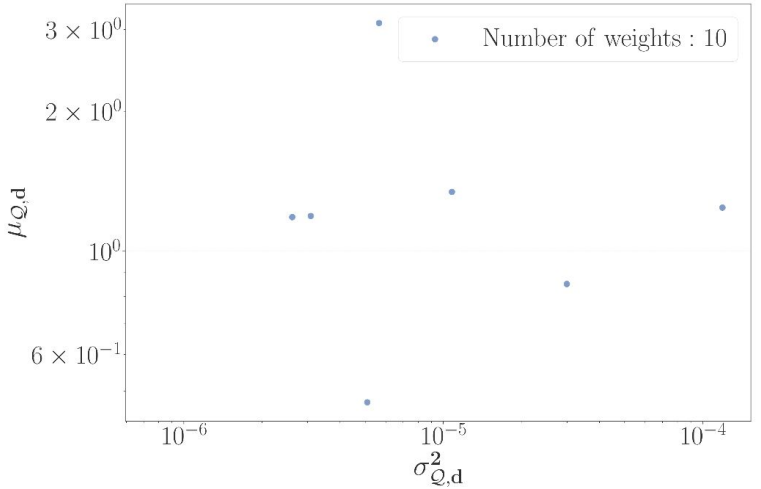
For models that are too simple, the uncertainty is artificially low because the posterior collapses to a delta function.

For models that are too complex, the uncertainty increases because there are not enough constraints.

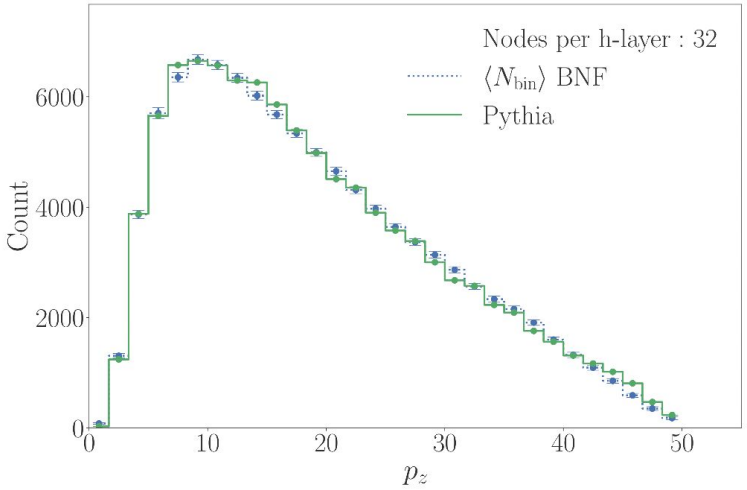


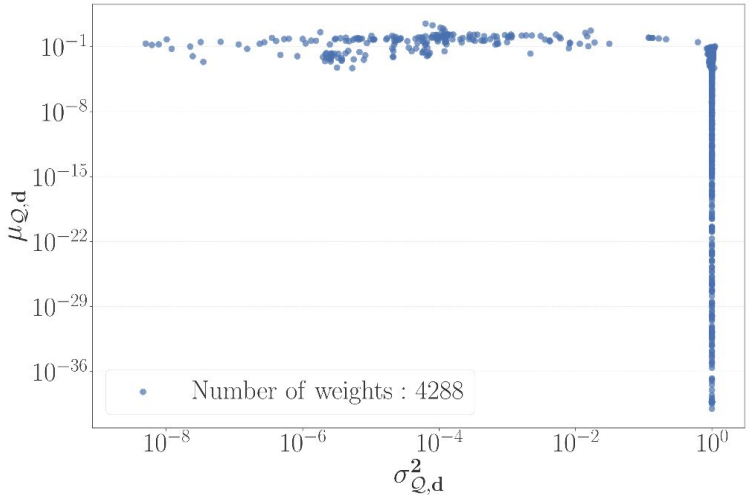
Too simple


Nodes per h-layer : 1
$\langle N_{\text{bin}} \rangle$ BNF
Pythia
Count
p_z


Number of weights : 10
$\mu_{\mathcal{Q},\mathrm{d}}$
$\sigma^2_{\mathcal{Q},\mathrm{d}}$

Complex enough





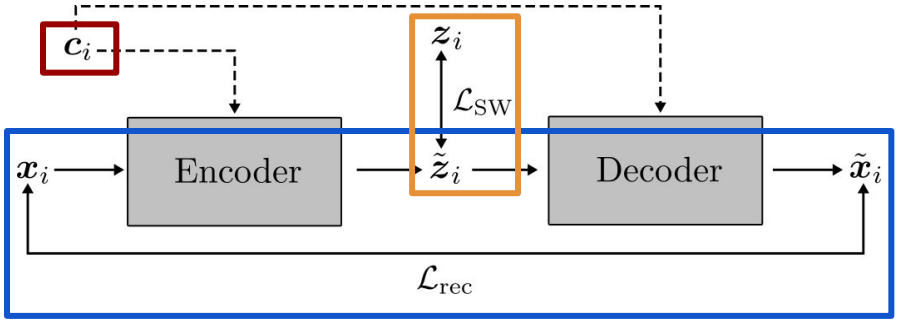
conditional Sliced Wasserstein AE

Probabilistic generative models are known to obtain **convincing physical observables** from **limited datasets** while retaining **flexibility** and **control of the output**.

Traditional Auto-Encoder with a **key difference:** go from Gaussian latent space to more flexible distribution.

Achieved through the Sliced Wasserstein method for pdf distance computation.

The energy of the string enters as a condition vector.



Training settings

N = 4x10^5 events

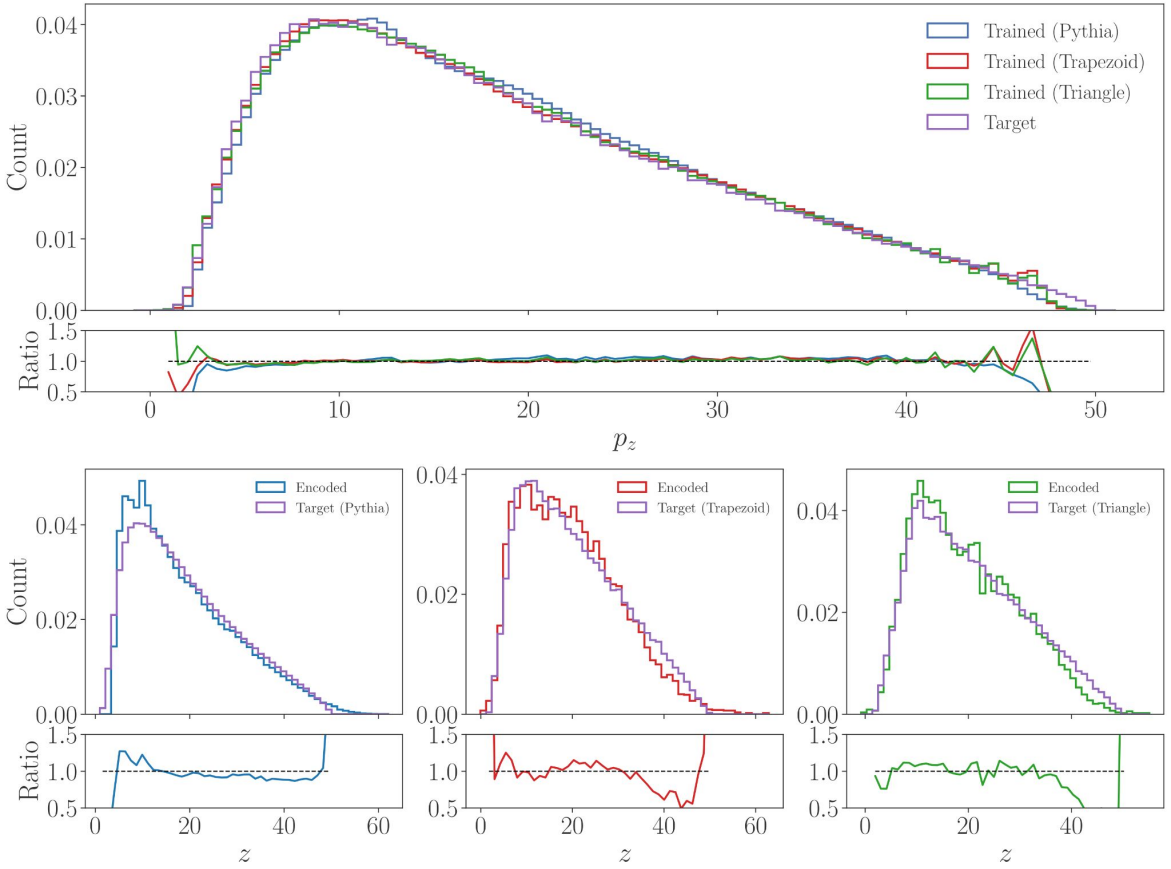
80% training, 20% validation

Saved in lists of a 100 events (→ 4000 "events")

Variable x	Target z	t (epochs)	d_z	λ	L
p_z'	PYTHIA	150	35	35	15
	Trapezoidal	300	2	20	30
	Triangular	150	2	30	25
p_T	PYTHIA	100	20	30	30
	Skew-norm	120	4	20	25
	Triangular	120	4	15	25

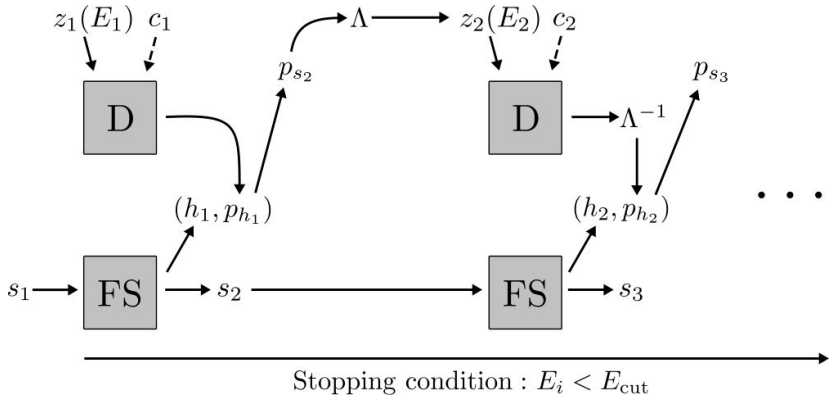
Learning Pythia

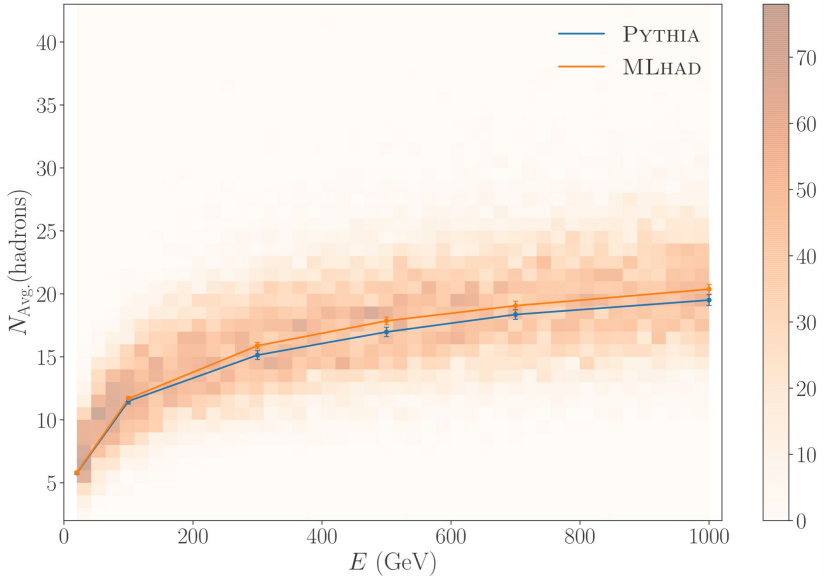
cSWAE learns the p_z and p_T spectrum for different string energies and different pdfs in the latent space.



Obtaining a full chain

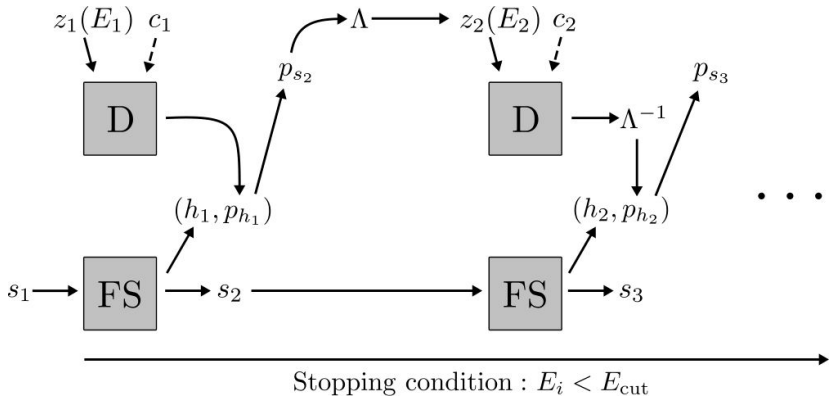
Fragmentation chain of N successive hadronizations. cSWAE recovers the probabilistic distribution of the chain length conditioned on the initial energy of the string.

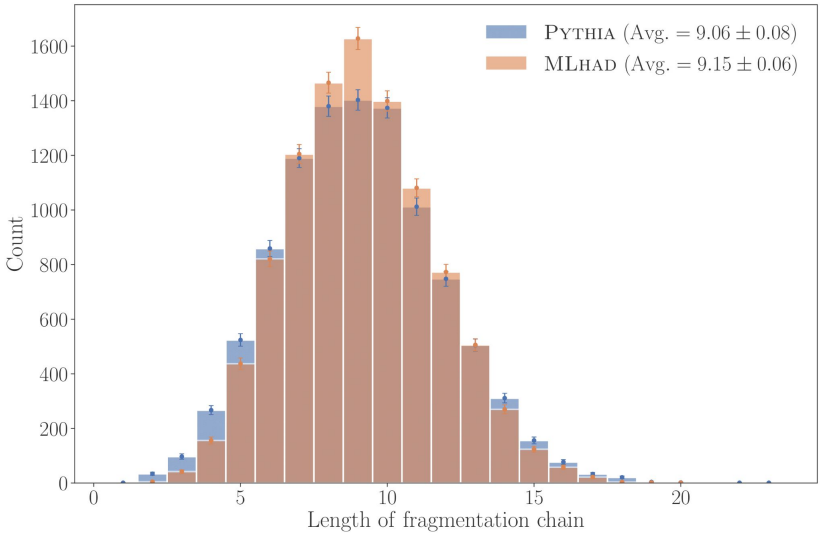




Obtaining a full chain

Fragmentation chain of N successive hadronizations.





Limitations

cSWAE has only been trained in **each variable separately** and only for **simple strings producing pions**.

It assumes no correlation between hadronization steps.

Not a single hadronization generator: generate a batch and have to post-process them for a single chain generation.

Valid until a certain energy E_{cut}

However, keep in mind this is a first crack at a very complicated problem.

Calibrated Simulation-Based Inference with WALDO

Introduced by L. Masserano, T. Dorigo, R. Izbicki, M. Kuusela, A. B. Lee in arxiv:2205.15680

Solve the inverse problem of estimating a posterior or **confidence intervals** on **simulator parameters** θ given an **observed dataset** X

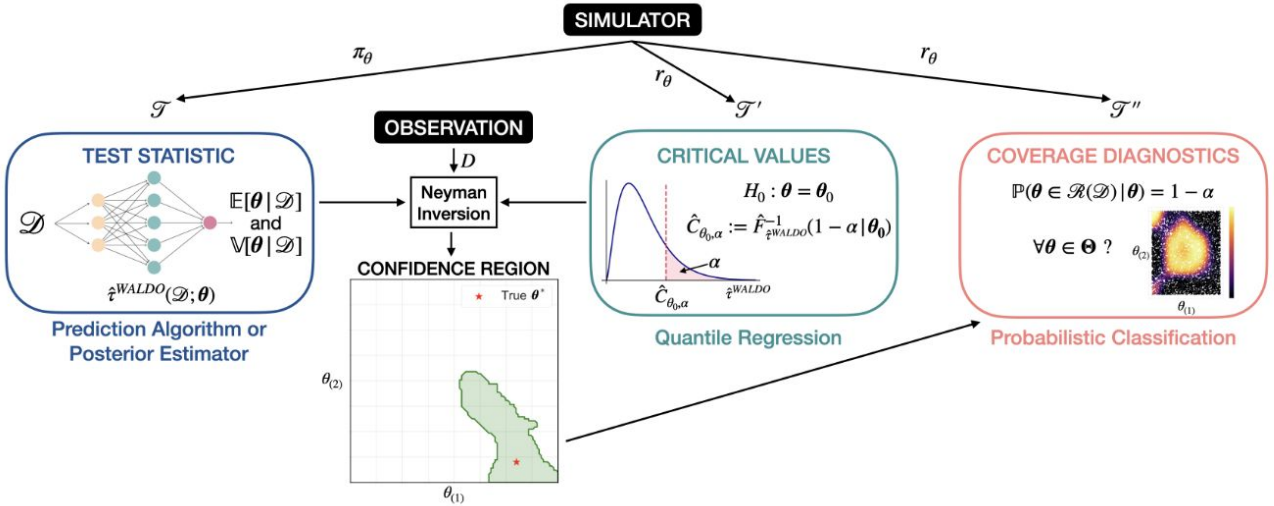
It does so by building a **test statistic** τ which converges to the Wald test statistic. This statistic and its distribution are obtained through simulations + training of three sets of ML estimators.

The main takeaway is it provides a **way of fitting our reweighting function to data with appropriate uncertainties**.

Calibrated Simulation-Based Inference with WALDO

Introduced by L. Masserano, T. Dorigo, R. Izbicki, M. Kuusela, A. B. Lee in arxiv:2205.15680. Fig. from the original paper:

(Pythia / MLHAD)



Calibrated Simulation-Based Inference with WALDO

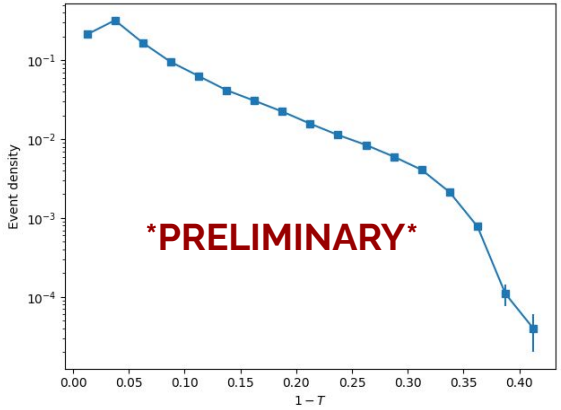
Before going full NN let's implement WALDO as a **Pythia tuner.** Pythia tunes are usually done with Professor (both by itself or with ML-enhancements Apprentice). Bayesian frameworks have also been explored. Usually very daunting task

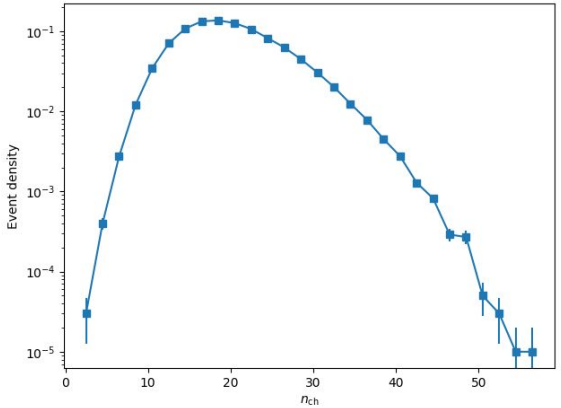
Let's see how we deal with the data used to tune p_T and p_z distributions for light flavors

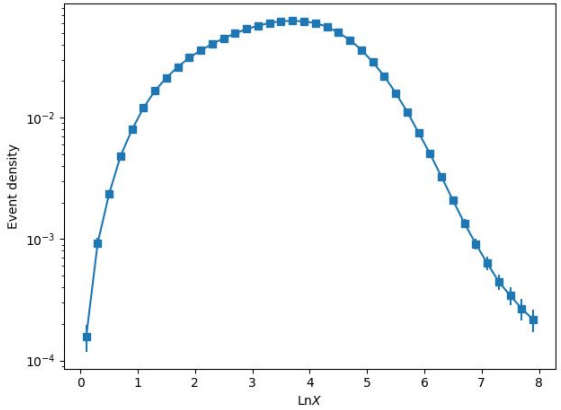
We choose the same set of observables used for the **standard Monash Tune**: **LEP Z→hadrons, event shapes and charged hadron properties**

We want to tune $\boldsymbol{\sigma_Q}$,$\boldsymbol{a_{Lund}}$, $\boldsymbol{b_{Lund}}$ as a proof of concept

Calibrated Simulation-Based Inference with WALDO



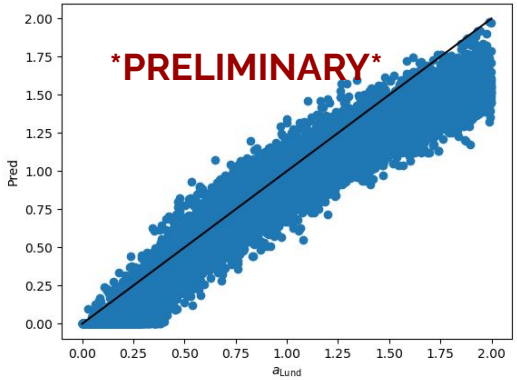


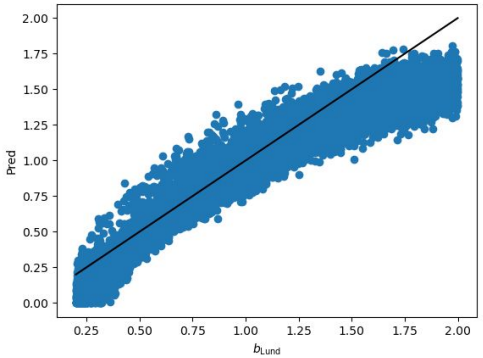


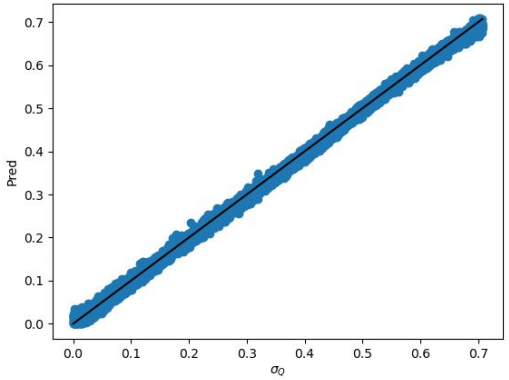
Uncertainties in training data → Already an interesting challenge. We incorporate them with explicit and implicit data augmentation.

Explicit: resample according to uncertainties. **Implicit**: resample for each pass using a probabilistic layer (in some sense similar to Dropout). More memory efficient

Calibrated Simulation-Based Inference with WALDO







Mean Estimation is not perfect. Model can be biased!

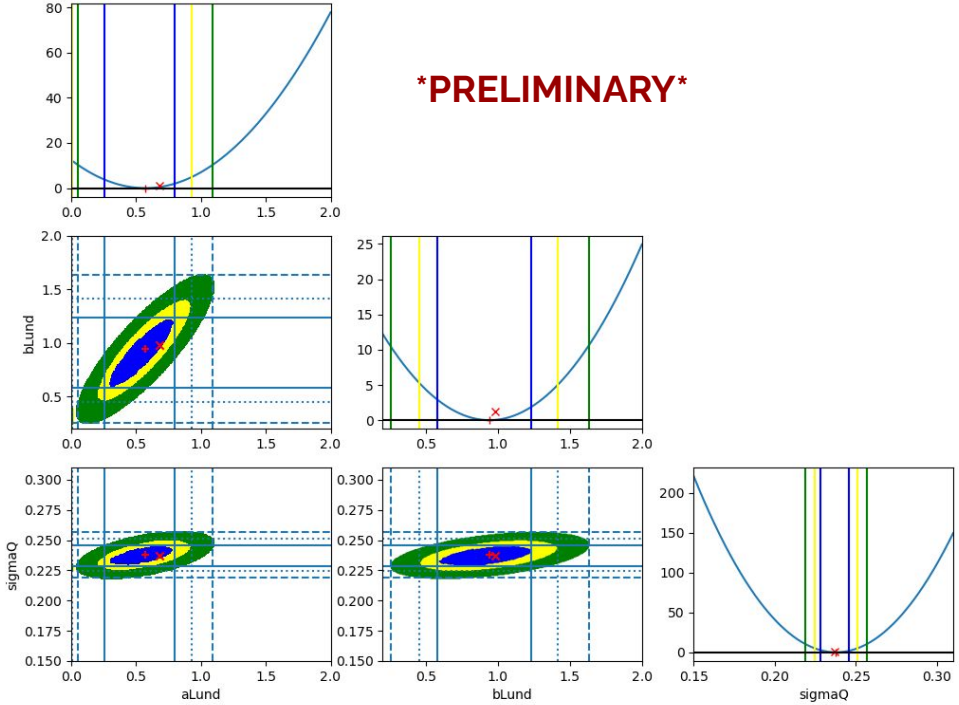
Variance estimation is also very tricky (usually very small) but if done well can overcome the bias by producing larger confidence intervals.

Calibrated Simulation-Based Inference with WALDO

As a self-consistency check, we recover the Monash parameters.

If training is done correctly, WALDO guarantees proper uncertainties with correct coverage.

This is already useful for Pythia itself

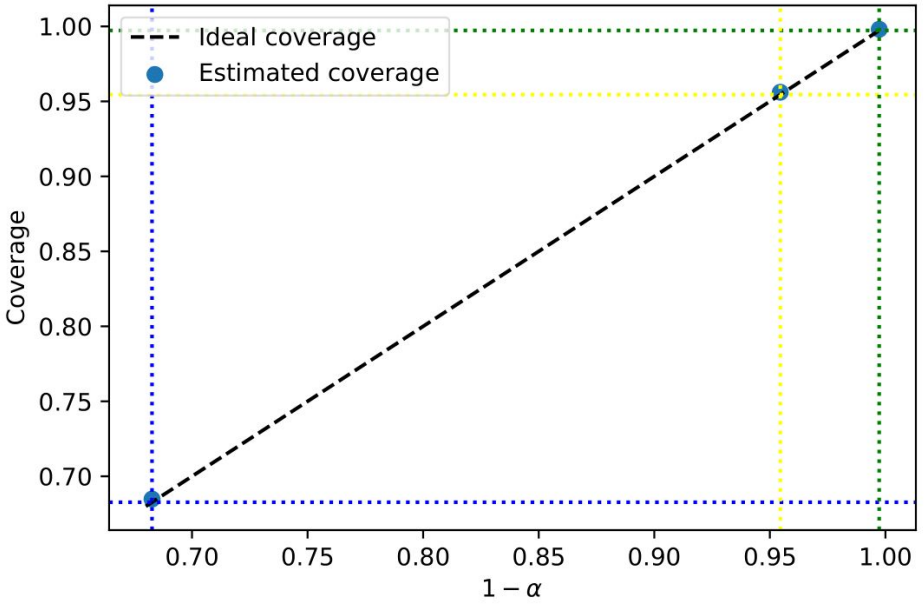


Calibrated Simulation-Based Inference with WALDO

We perform the coverage diagnostics

Appropriate coverage is achieved. This is a very important feature of WALDO → not only can we train, we can obtain meaningful uncertainties

PRELIMINARY



Next steps

- Do full calibration of p_T and p_z sector (4 additional parameters which control heavy quark and diquark cases)
- Replace Lund function with a learnable estimator with manageable set of (hyper)parameters
- Deal with discrete flavor parameters → different approach needed?
- Take advantage of reweighting to get larger datasets
- Do all of this with available **data** (already started for the three parameter case + LEP data)

Next directions: Observable choices

Definition of better observables for training. We need observables sensitive to differences in the hadronization models