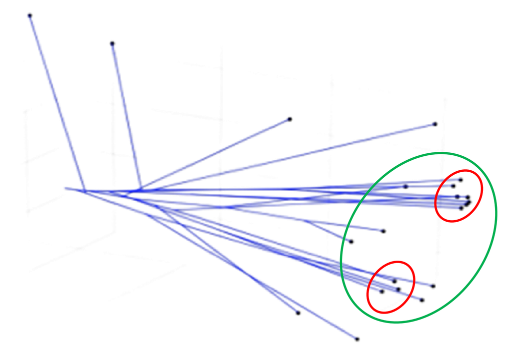
Reconstructing Jet History With Machine Learning

A. Sengupta [1,2], R. J. Fries [1,2], M. Kordell [1,2]
[1] Cyclotron Institute, Texas A&M University, College Station TX 77843.
[2] Department of Physics and Astronomy, Texas A&M University, College Station TX 77843.

Jets

"Jet" → narrow cone of partons and hadrons; only occur in high-energy (≫1 GeV) collisions.

- Can be used as probes to understand QGP properties



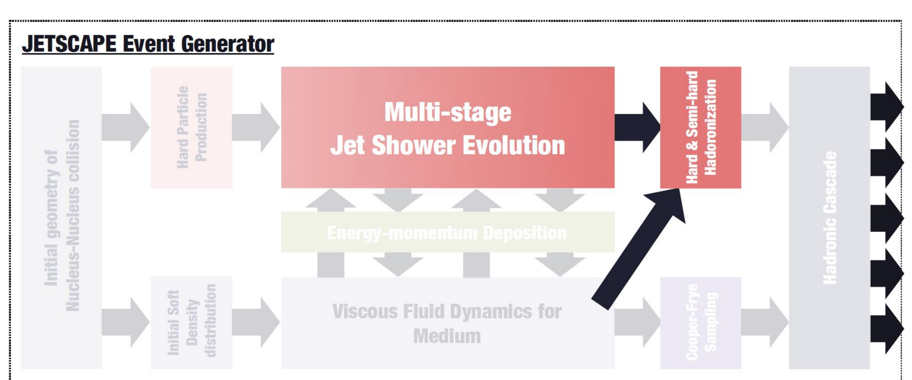
Why Reconstruct Jet History?

- We would like to understand the local properties of the surrounding medium, which is directly linked to the space-time evolution of a jet shower.
- Extracting jet space time evolution poses a challenge – experiments cannot directly measure jet history, and only have access to final state hadron momenta.
- Can we use **machine learning (ML)** to reconstruct the jet history?
- Ultimately, we want to reconstruct jet history in real collisions, using final state data – but for now, we want to test the **feasibility** of this approach using data from a Monte Carlo simulation.

JETSCAPE Framework[1,2,3]

Monte Carlo simulation of relativistic heavy-ion collisions.

Many modules for soft and hard physics.



Y. Tachibana (2022)

We use 4 modules that deal with jets in quark-gluon plasma (QGP):

1. **Parton Gun** – fires a parton with a given energy and flavor in the x-direction.
2. **MATTER** – propagates and splits the initial jet parton, until its virtuality falls below a threshold Q_0.
3. **LBT** – propagates low-virtuality and real partons through the QGP medium (not used in this study).
4. **Hybrid Hadronization** – hadronizes partons through recombination (short distances) and string fragmentation (long distances).

Neural Networks

A **Neural Network (NN)** is a type of machine learning (ML) architecture.

The NN takes a set of features, $\vec{x}$, as its input (e.g. momentum vectors of particles).

Passes the features through "hidden layers" consisting of "neurons". Each neuron takes an input x_j in terms of the outputs y_i of the neurons that feed it

$$x_j = \sum_i w_{ij}\, y_i + b_j$$

where w_{ij} and b_j are the weights and biases, which can be adjusted.

Neurons apply nonlinear **activation functions** to their inputs (e.g. tanh): $y_j = f_{\text{activation}}(x_j)$

Get a predicted value, $y_{\text{predicted}}$, as the output layer (see figure below).

A **supervised NN** is trained using known outputs. Predicted values are compared to known/actual values using a loss function e.g. Mean Squared Error (MSE), given by

$$E = \frac{1}{2}\left(y_{\text{predicted}} - y_{\text{actual}}\right)^2$$

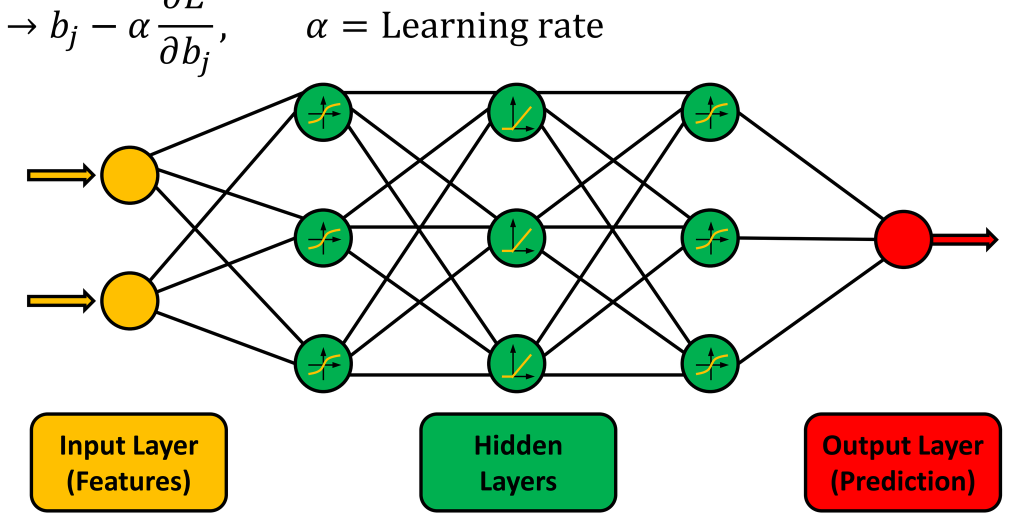
The model is then tuned to reduce the loss function, commonly using **gradient descent**,

$$w_{ij} \to w_{ij} - \alpha\frac{\partial E}{\partial w_{ij}}, \qquad b_j \to b_j - \alpha\frac{\partial E}{\partial b_j}, \qquad \alpha = \text{Learning rate}$$

Repeated until the loss function is optimized for best predictions.

Unlike linear regression, NNs can find complex, **nonlinear patterns**.

In this study, we use the JETSCAPE framework to generate training data for a NN.



Training Data

Note: We are sticking to 100 GeV **vacuum jets** in this study.

JETSCAPE provides two sets of data: (η, ϕ, E) of partons, and (η, ϕ, E) of hadrons.

Ultimately, we want good predictions from the **hadron** data, since that is what experiments can measure.

- Features: (η, ϕ, E) of the partons/hadrons/reclustered jets.
- 2-3 hidden layers, each with 10-200 nodes (neurons).
- Activation functions: ReLU, tanh and/or sigmoid.
- Size of training and test sets: ~140,000 events each.

Longitudinal & Transverse Observables

We want to predict two observables, R_g and z_g, related to transverse and longitudinal jet structure respectively. They are defined as

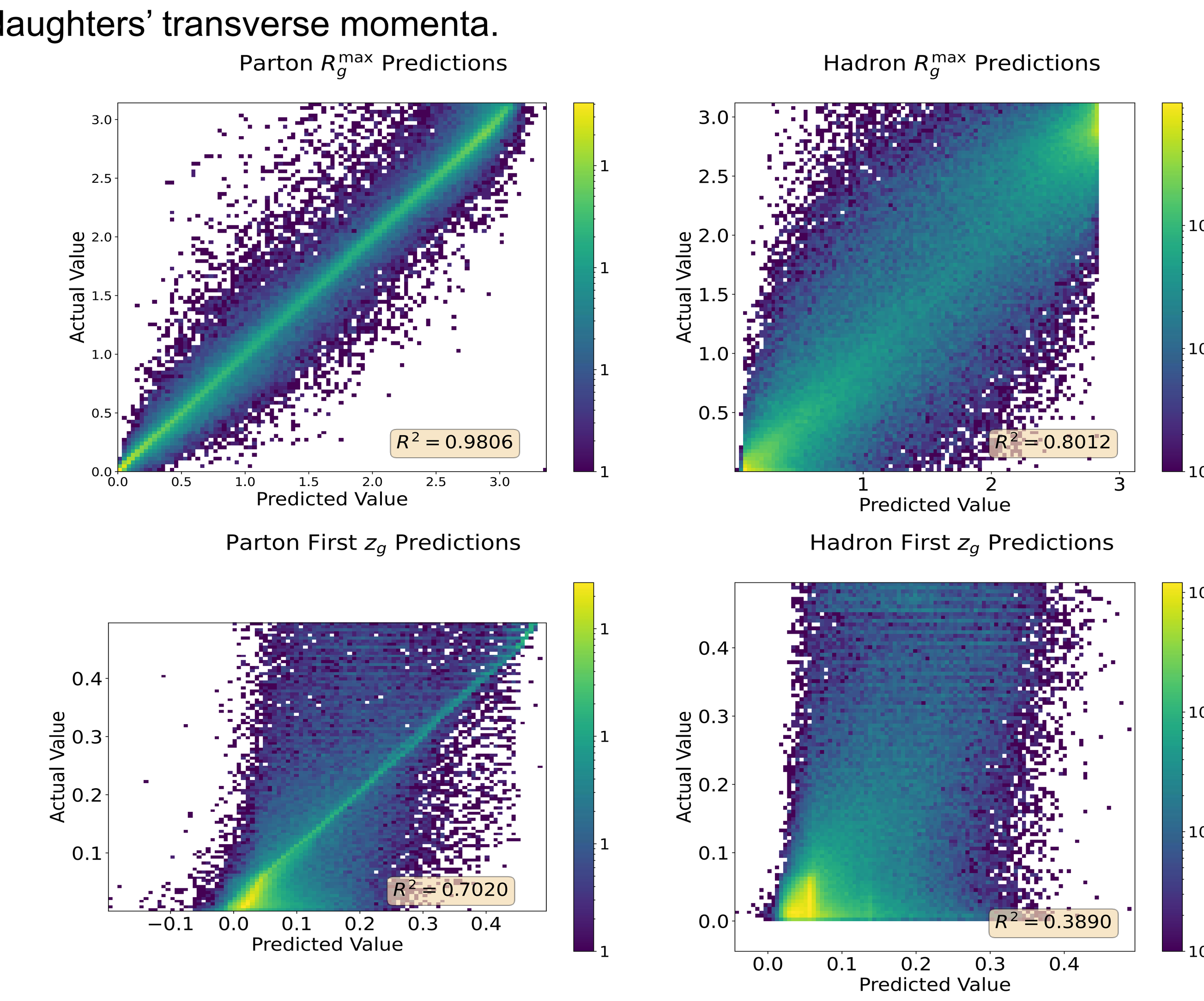
$$R_g^{\max} = \text{Max}\left[\sqrt{(\eta_1-\eta_2)^2 + (\phi_1-\phi_2)^2}\right]$$
$$\propto \text{Angle of } \textit{widest} \text{ splitting}$$

$$z_g^{(1)} = \frac{\min(p_{T1}, p_{T2})}{p_{T1}+p_{T2}} \text{ for the } \textit{first}\ 1 \to 2 \text{ split}$$

where p_{T1} and p_{T2} are the daughters' transverse momenta.

$R_g^{\max}$ is **well-predicted** by both parton data (left) and hadron data (right).

$z_g^{(1)}$ is still well-predicted by parton data (left) but is **poorly-predicted** by hadron data (right).



Preliminary finding: **Longitudinal structure information is lost in hadronization**.

Two Challenges

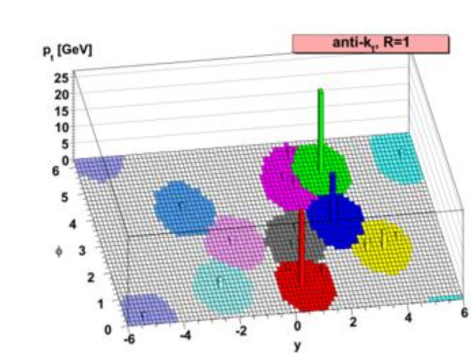
1. Want to align with experimental procedure – using **jet clustering of hadrons** as input. Unfortunately, this makes the ML predictions worse, as final state information is lost.
2. Want to ask the right question. What observable might be easier to predict from final state data? We can **apply cuts** to data and observables, as seen below.

Clustering Algorithms[4]

We use Fastjet to cluster the hadrons into jets of varying radii R. Clusters particles within distance d_{ij} of each other, where

$$d_{ij} = \min\left(p_{Ti}^{2k}, p_{Tj}^{2k}\right)\frac{\left(\eta_i-\eta_j\right)^2 + \left(\phi_i-\phi_j\right)^2}{R^2}$$

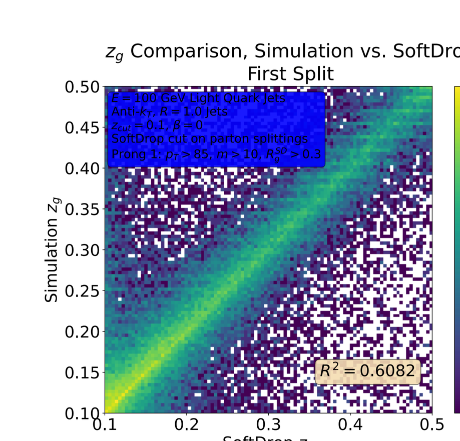
$k=-1 \to$ anti-k_T algorithm $\qquad k = 0 \to$ Cambridge-Aachen algorithm



SoftDrop[5]-Inspired Cuts

Turn to **SoftDrop** for inspiration – algorithm designed to trim soft, wide-angle radiation.

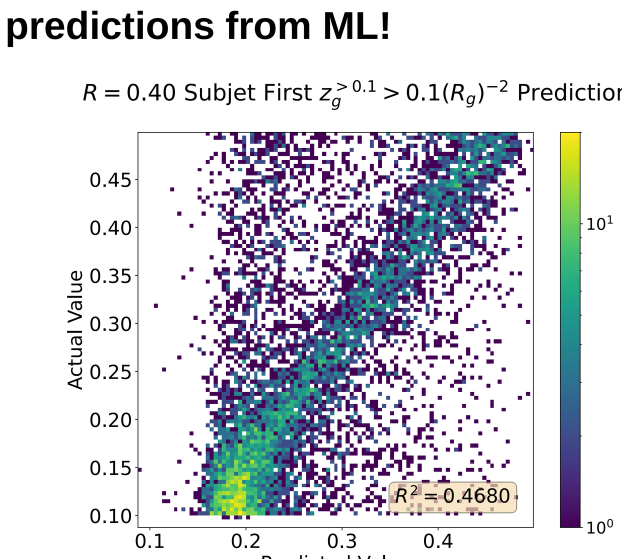
1. Anti-k_T jet finding on final state hadron data.
2. Cambridge-Aachen (C/A) reclustering.
3. Decluster C/A jet (call it j_0), producing two prongs. Call them j_1 and j_2. Test for the SoftDrop condition:
$$z_g > z_{cut}\left(\frac{\Delta R_{12}}{R_0}\right)^\beta$$
$$z_{cut}, R_0, \beta \text{ variable parameters}$$
4. If condition is met, return j_0, else repeat step 3 on the harder prong.



With well-chosen cuts on both the prongs and the splitting data, SoftDrop can recover the first z_g (that satisfies all cuts) with a good correlation ($R^2 \approx 0.6$).

→ We can use SoftDrop-like cuts on our data to improve our z_g predictions from ML!

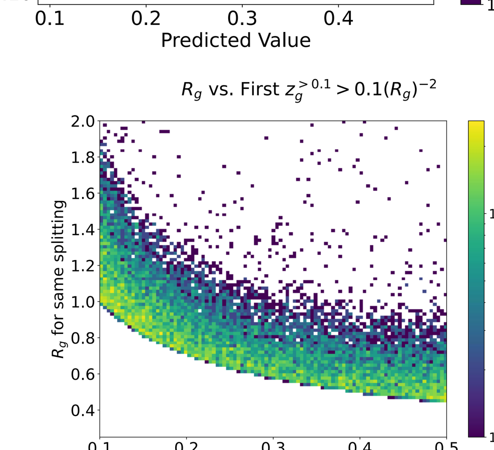
- $R = 1.0$ anti-k_T jet finding, $R = 0.4$ C/A reclustering.
- SoftDrop-like condition on parton splittings:
$$z_g > z_{cut}\left(R_g\right)^\beta$$
- Additional cut: $z_g > 0.1$.



Predict first z_g (if any) that satisfies both criteria. With $z_{cut} = 0.1$, $\beta = -2$, we get **markedly improved predictions** from reclustered jet data (top right).

- Softdrop-like condition favors **prominent** splittings (large z_g or large R_g).
- Cut on z_g removes remaining soft, wide-angle splittings (bottom right).

Resulting splittings are easier to resolve → better predictions!



Future Work

1. Further refine cuts, algorithms and procedures.
2. Apply to e^+e^- and $A + A$ simulations.
3. Apply to experimental data.

[1] J.H. Putschke et. al., arXiv:1903.07706, [2] A. Kumar et. al., Phys. Rev. C **102**, 054906 (2020), [3] A. Kumar et. al., arXiv:2204.01163
[4] M. Cacciari , G.P. Salam, G. Soyez , JHEP **04** (2008), 063
[5] A.J. Larkoski, S. Marzani, G. Soyez, J. Thaler, JHEP **1405** (2014), 146

Acknowledgments: This work was funded in part by NSF Grant Numbers PHY-2111568 and PHY-2004571.