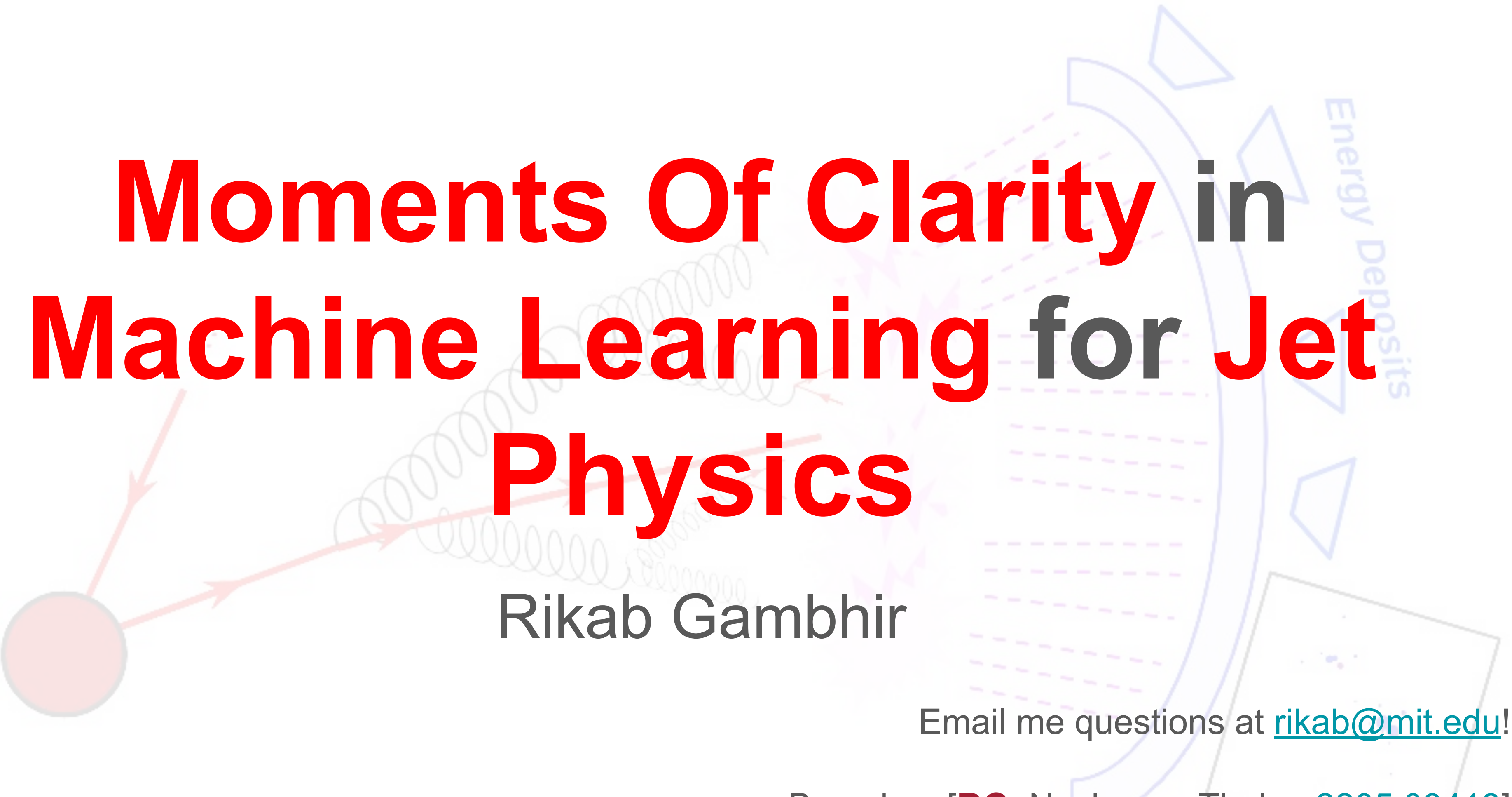
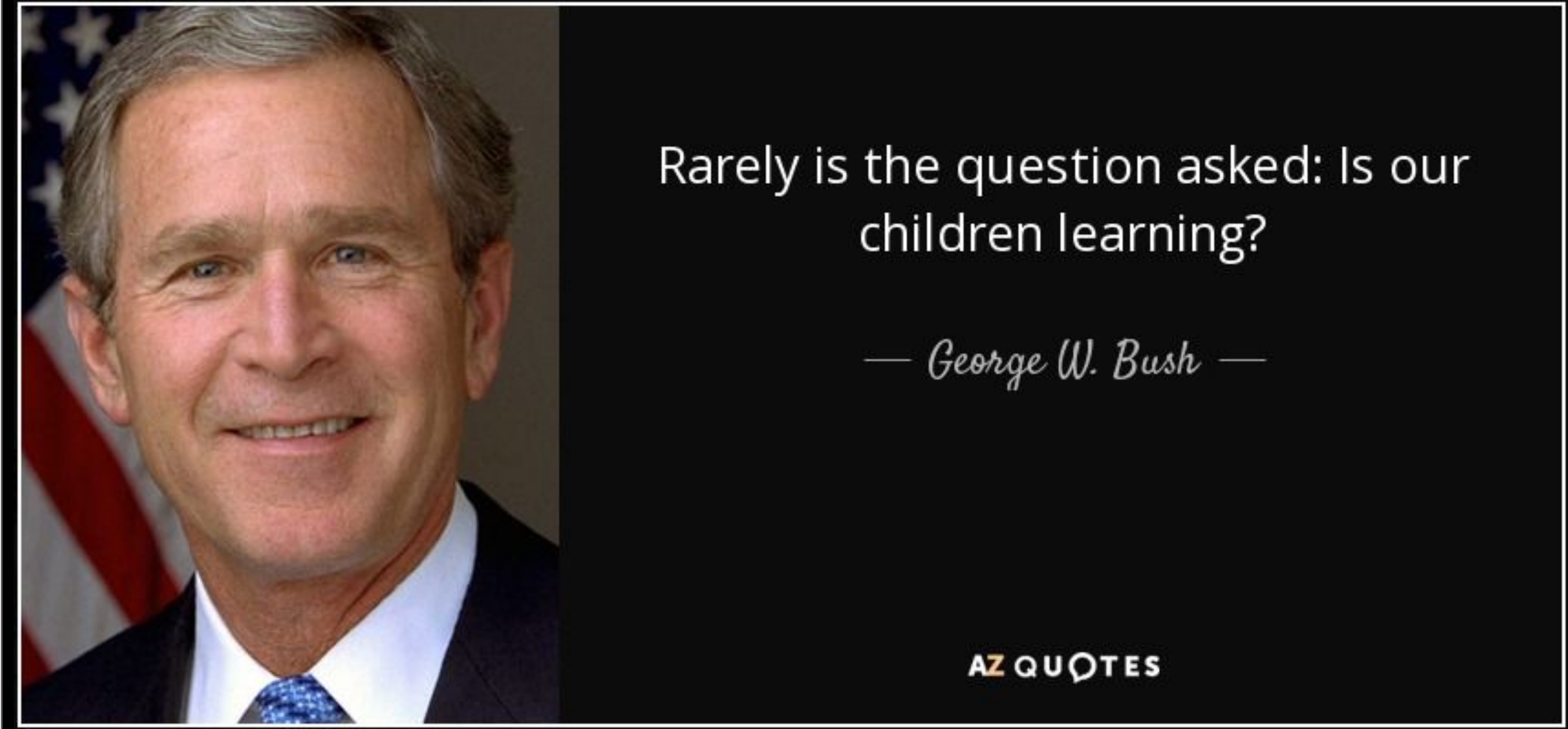
Moments Of Clarity in Machine Learning for Jet Physics



Rikab Gambhir

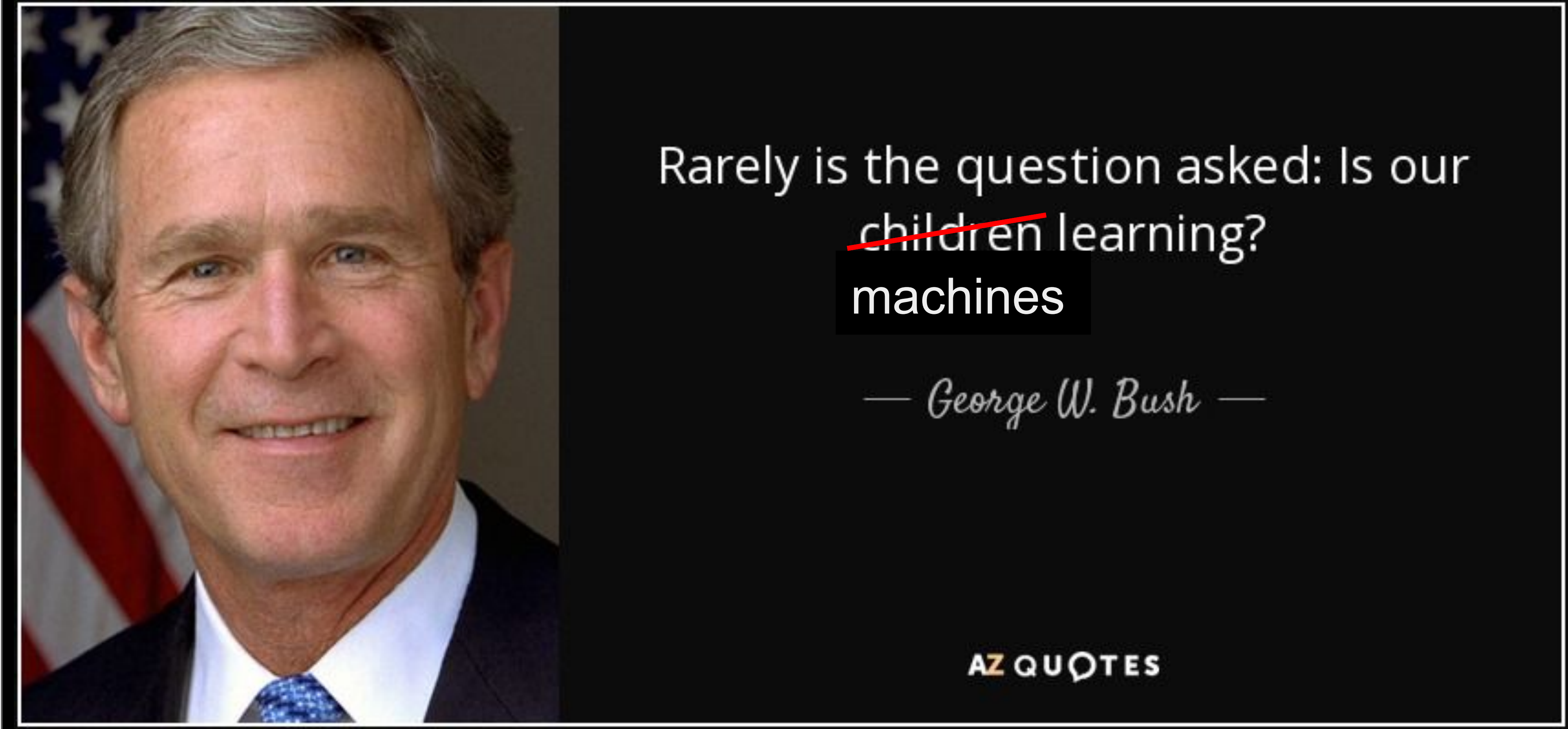
Email me questions at rikab@mit.edu!

Based on [**RG**, Nachman, Thaler, 2205.03413]
[**RG**, Nachman, Thaler, 2205.05084]
[Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]
[**RG**, Thaler, Wu, WIP]
[**RG**, Osathapan, Tasissa, Thaler, 2403.08854]



Rarely is the question asked: Is our children learning?

— *George W. Bush* —

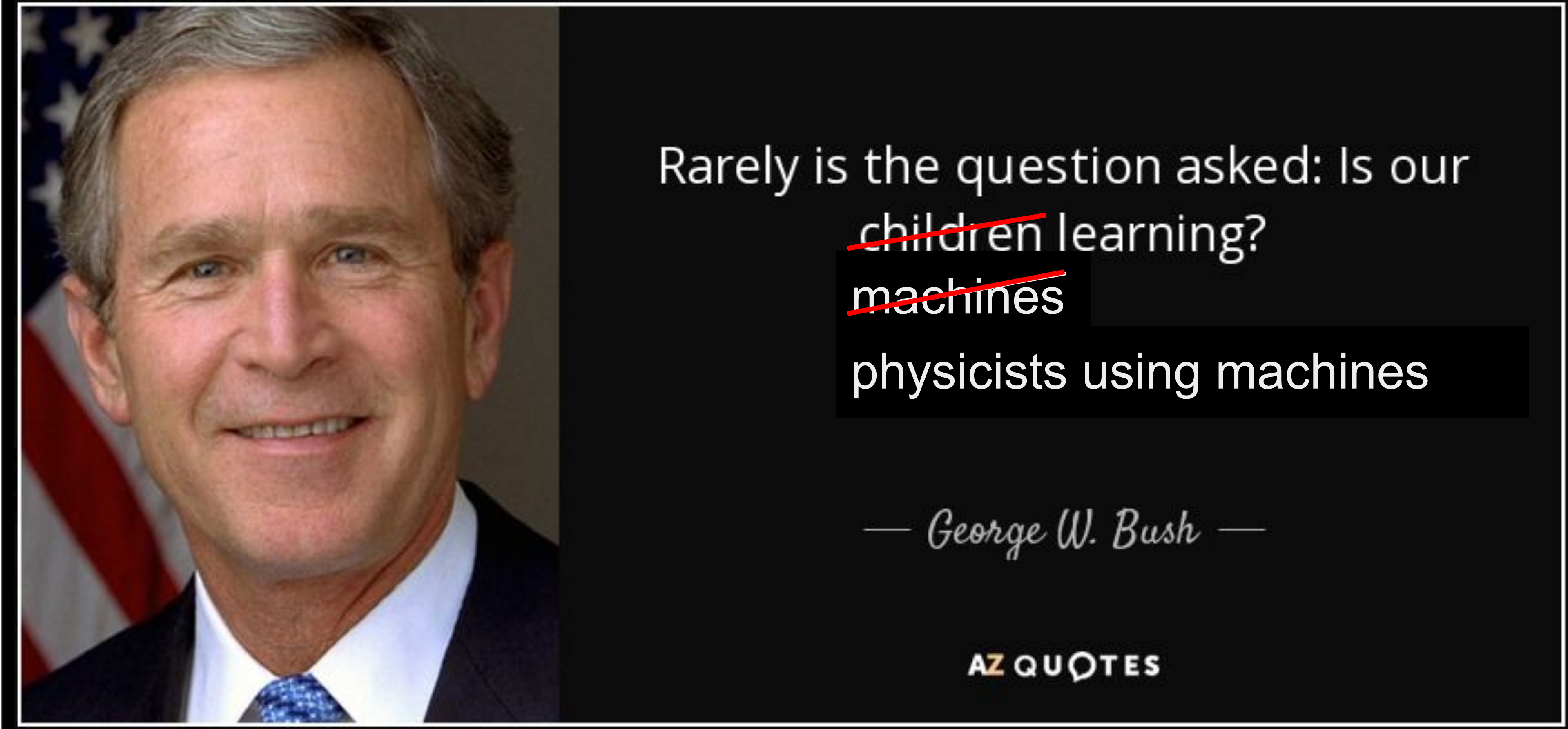


Rarely is the question asked: Is our ~~children~~ learning?

machines

— George W. Bush —

AZ QUOTES



Rarely is the question asked: Is our ~~children~~ learning?

~~machines~~

physicists using machines

— George W. Bush —

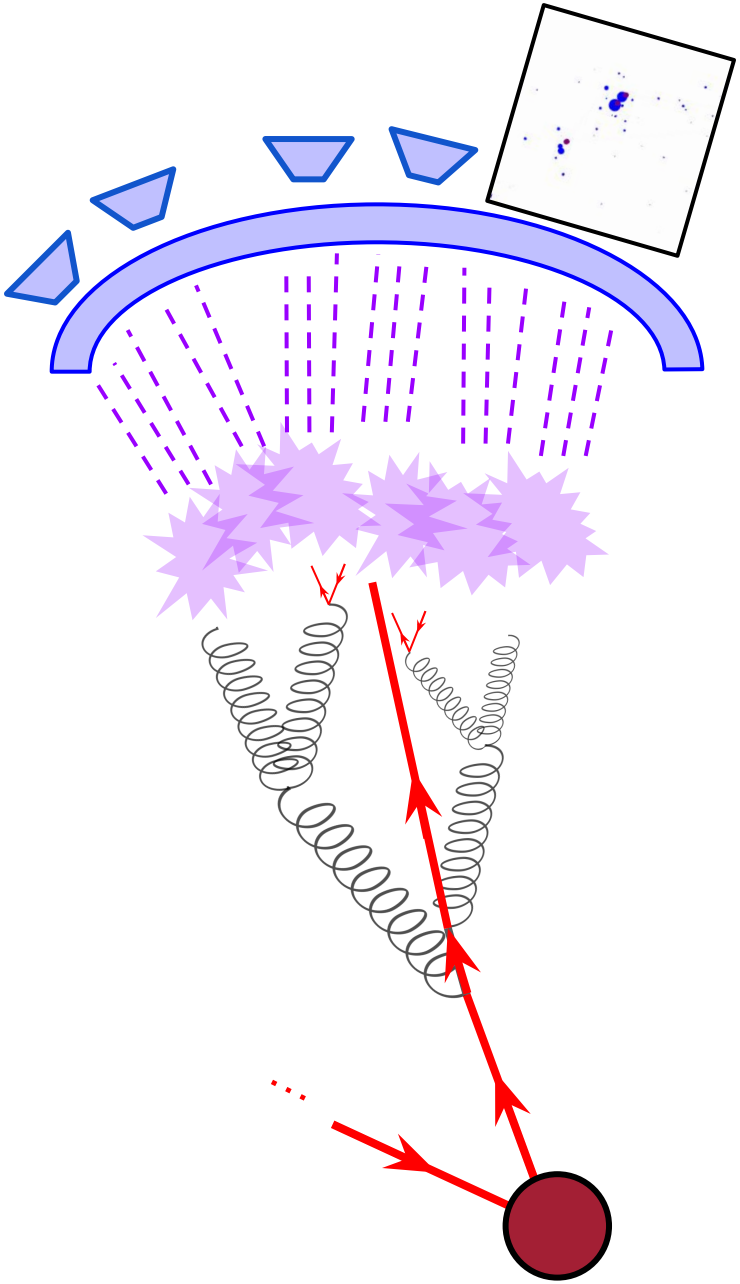
AZ QUOTES

Introduction

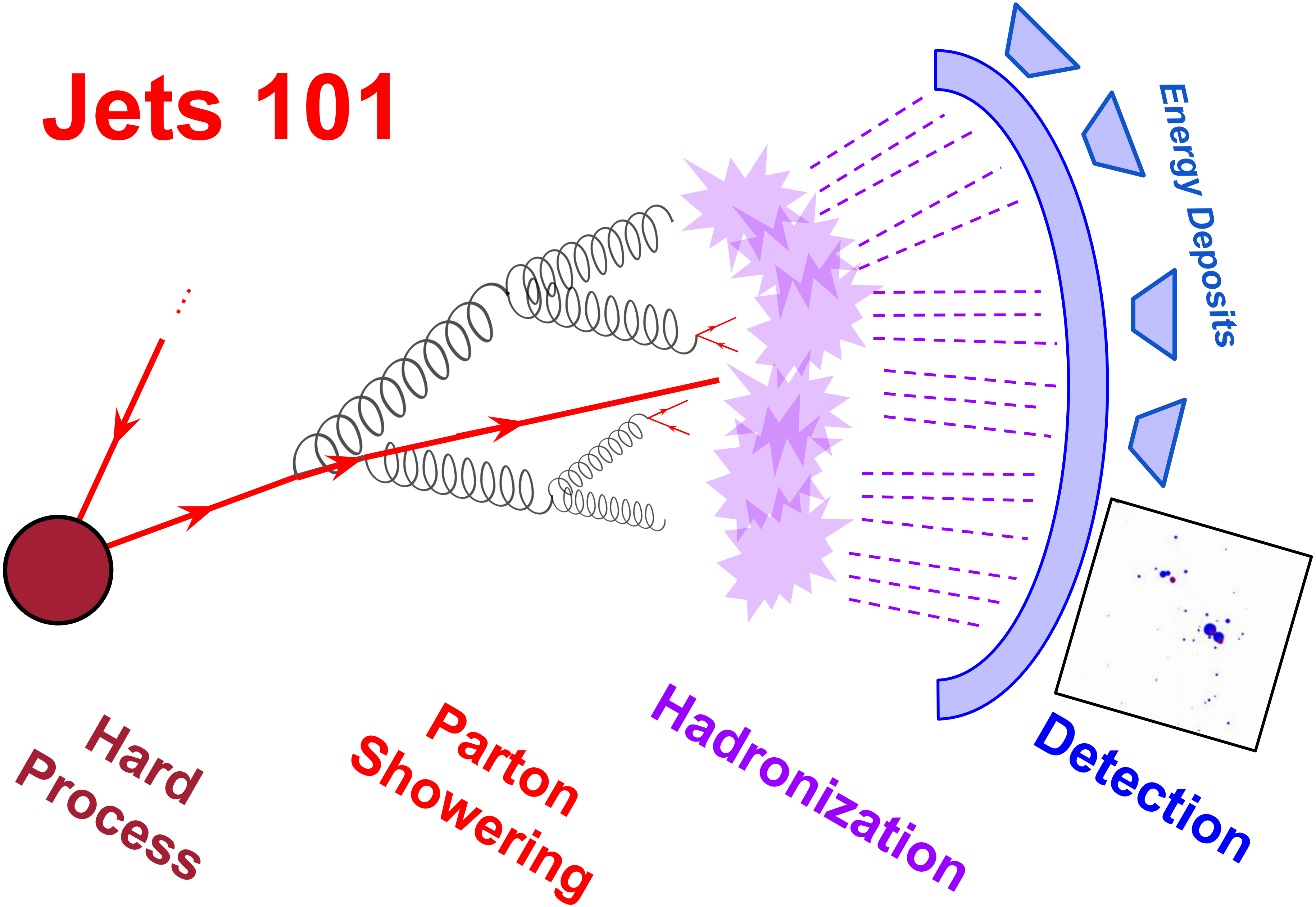
I want to study **jets** at the **Large Hadron Collider** (LHC). Jets are complicated!

I want to use **machine learning** (ML) to make my job easier, but with safeguards in place to make sure the physics I learn makes some sense!

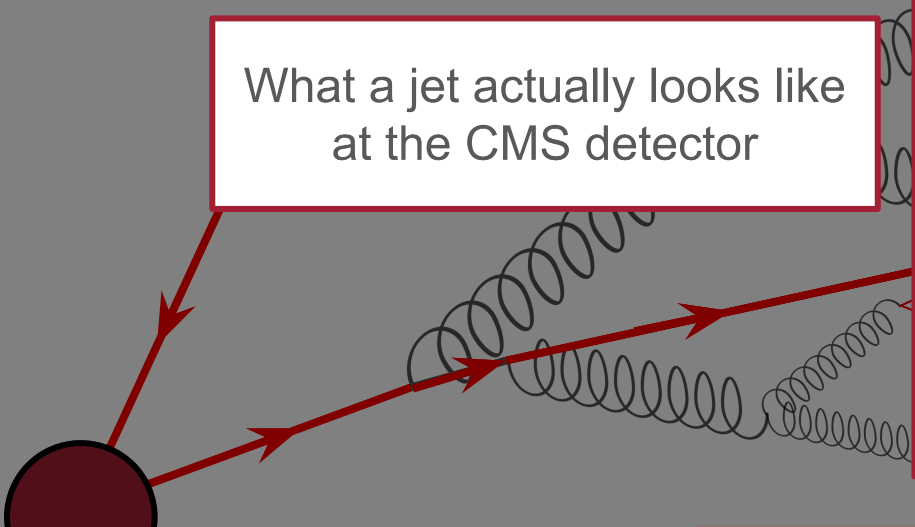
This talk: 3 vignettes from my work in jet physics designing ML algorithms to give me exactly what I want.



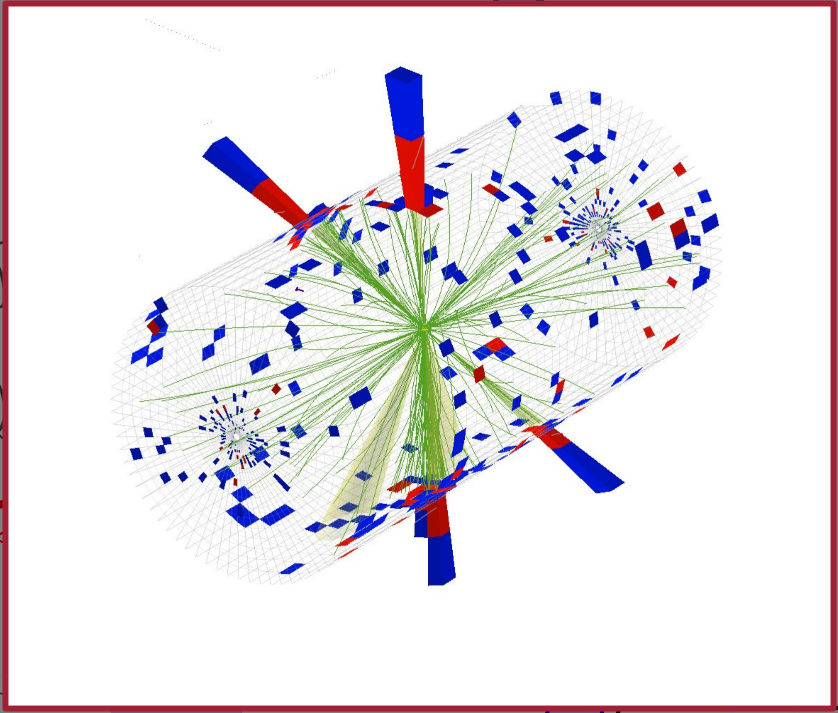
Jets 101


Energy Deposits
Hard Process
Parton Showering
Hadronization
Detection

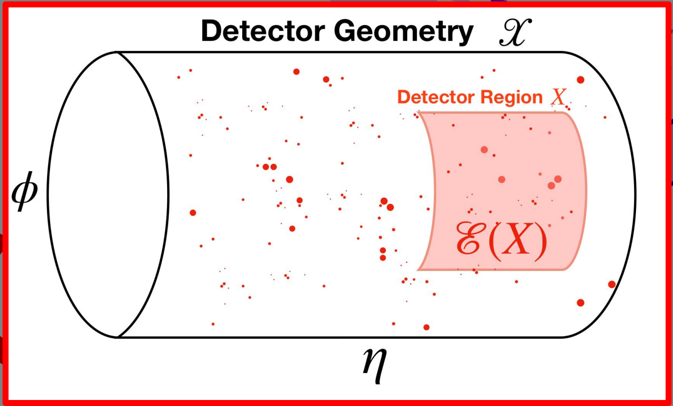
Jets 101

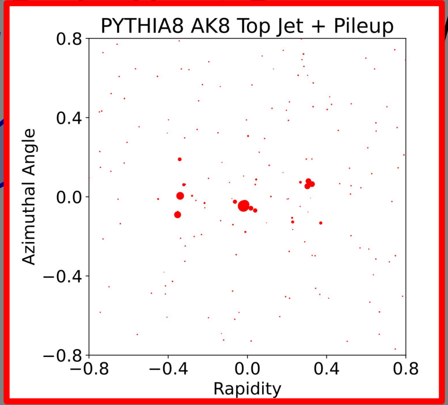


What a jet actually looks like at the CMS detector

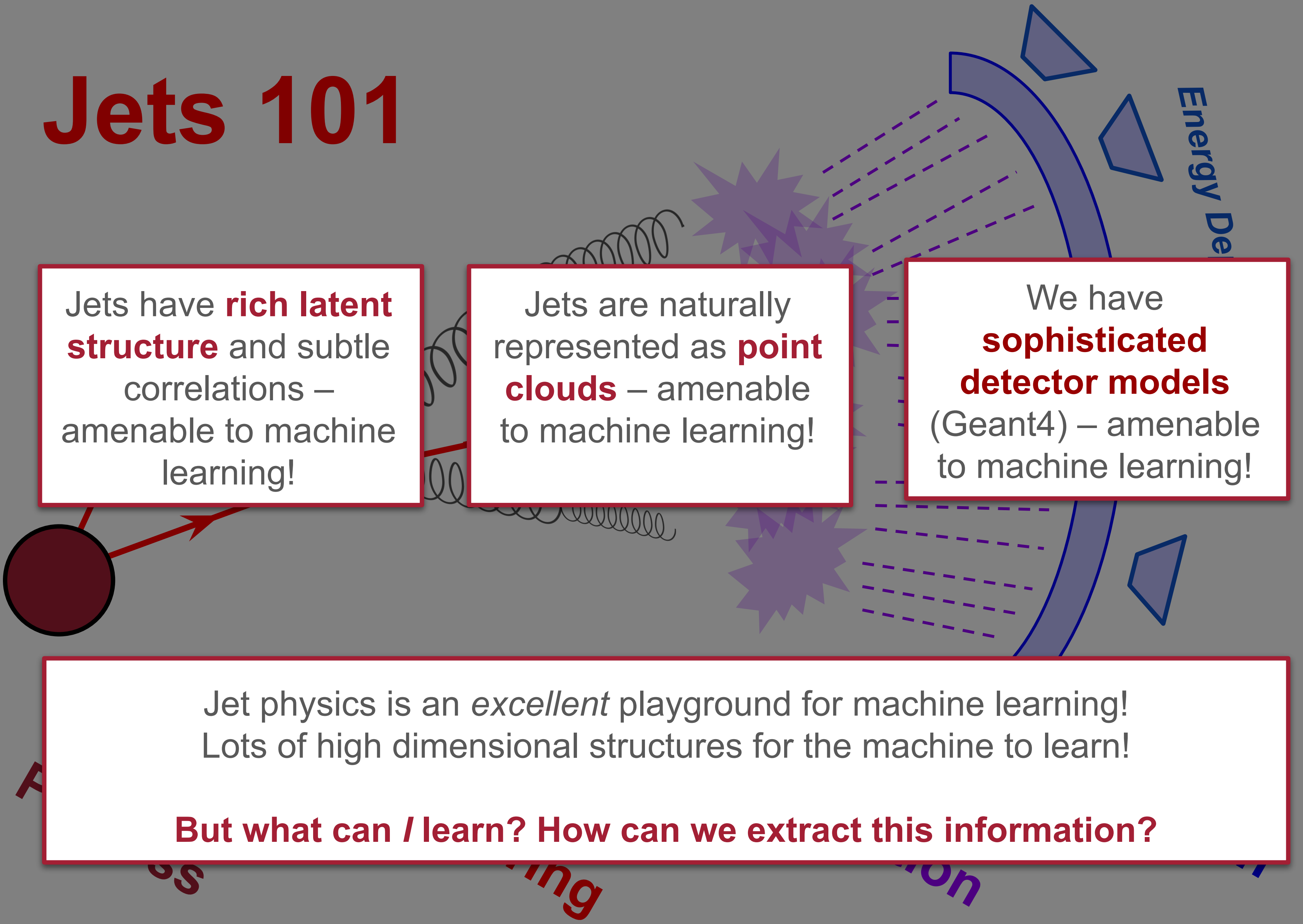


My theorist's picture of a jet




PYTHIA8 AK8 Top Jet + Pileup

Jets 101



Jets have **rich latent structure** and subtle correlations – amenable to machine learning!

Jets are naturally represented as **point clouds** – amenable to machine learning!

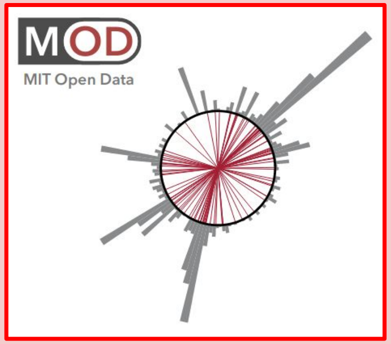
We have **sophisticated detector models** (Geant4) – amenable to machine learning!

Jet physics is an *excellent* playground for machine learning!
Lots of high dimensional structures for the machine to learn!

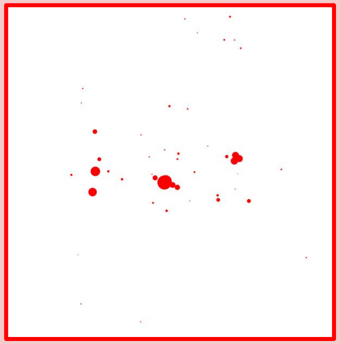
But what can *I* learn? How can we extract this information?

Things a **machine** can understand

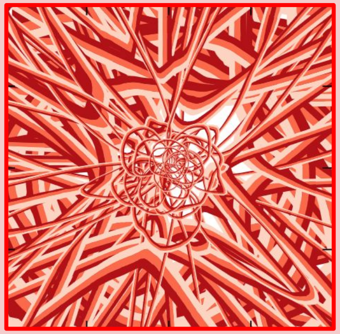
Sophisticated Detector Models



Complicated Point Clouds

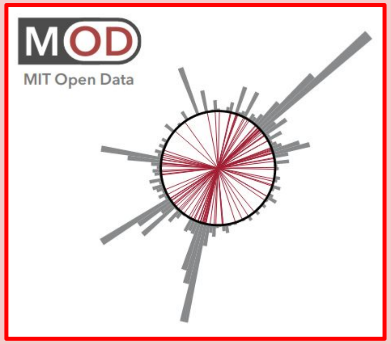


Huge Latent Spaces

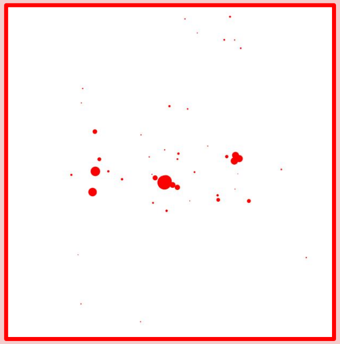


Things a **machine** can understand

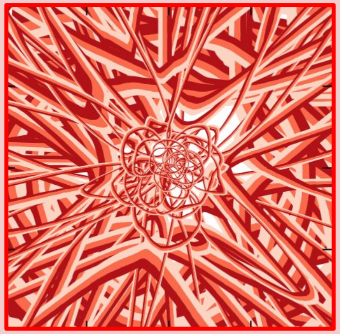
Sophisticated Detector Models

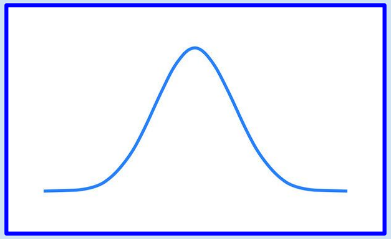


Complicated Point Clouds

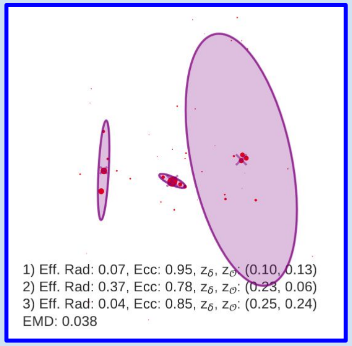


Huge Latent Spaces

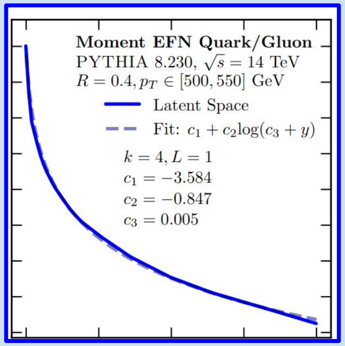




Gaussians



Basic Kindergarten Shapes

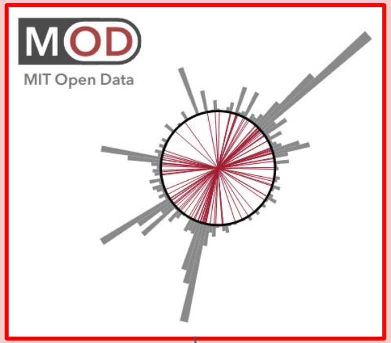


Addition, multiplication, and a few elementary functions

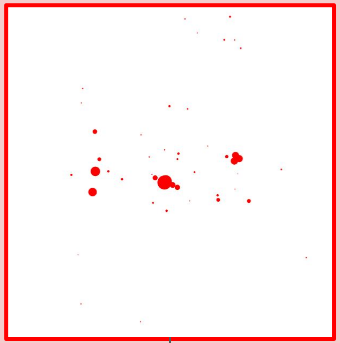
Things my **mortal physicist brain** can understand

Things a **machine** can understand

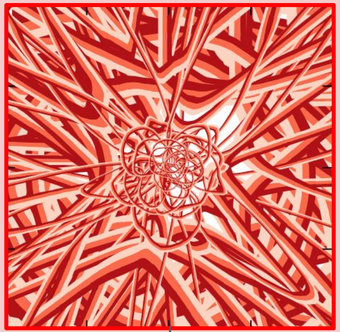
Sophisticated Detector Models



Complicated Point Clouds

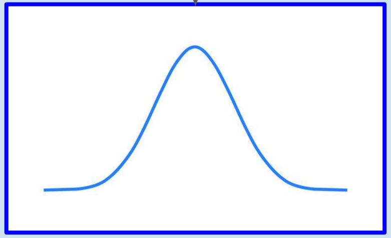


Huge Latent Spaces



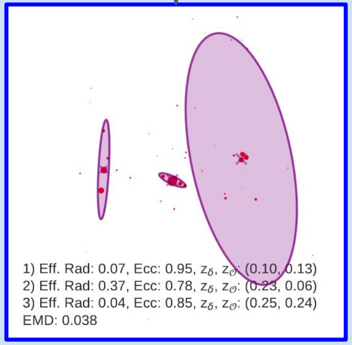
Interface: *Targeted* and *goal-motivated* network architectures and loss functions!

Using the Gaussian Ansatz and DV Loss



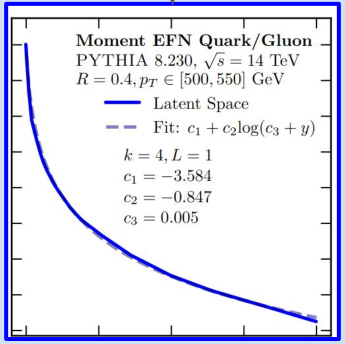
Gaussians

Using faithful optimal transport



Basic Kindergarten Shapes

Using Moment Pooling



Addition, multiplication, and a few elementary functions

Things my **mortal physicist brain** can understand

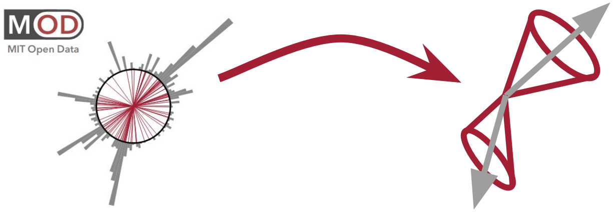
Things a **machine** can understand

Sophisticated Detector Models

Complicated point clouds

Huge Latent Spaces

Learning **Uncertainties** the **Frequentist** Way

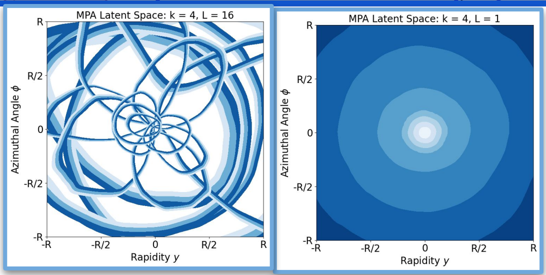


[**RG**, Nachman, Thaler, 2205.05084
[**RG**, Nachman, Thaler, 2205.05084]

Can you Hear the **Shape** of a Jet?

[Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]

The **Moments** of Clarity: Streamlining Latent Spaces



[**RG**, Osathapan, Thaler, 2403.08854]

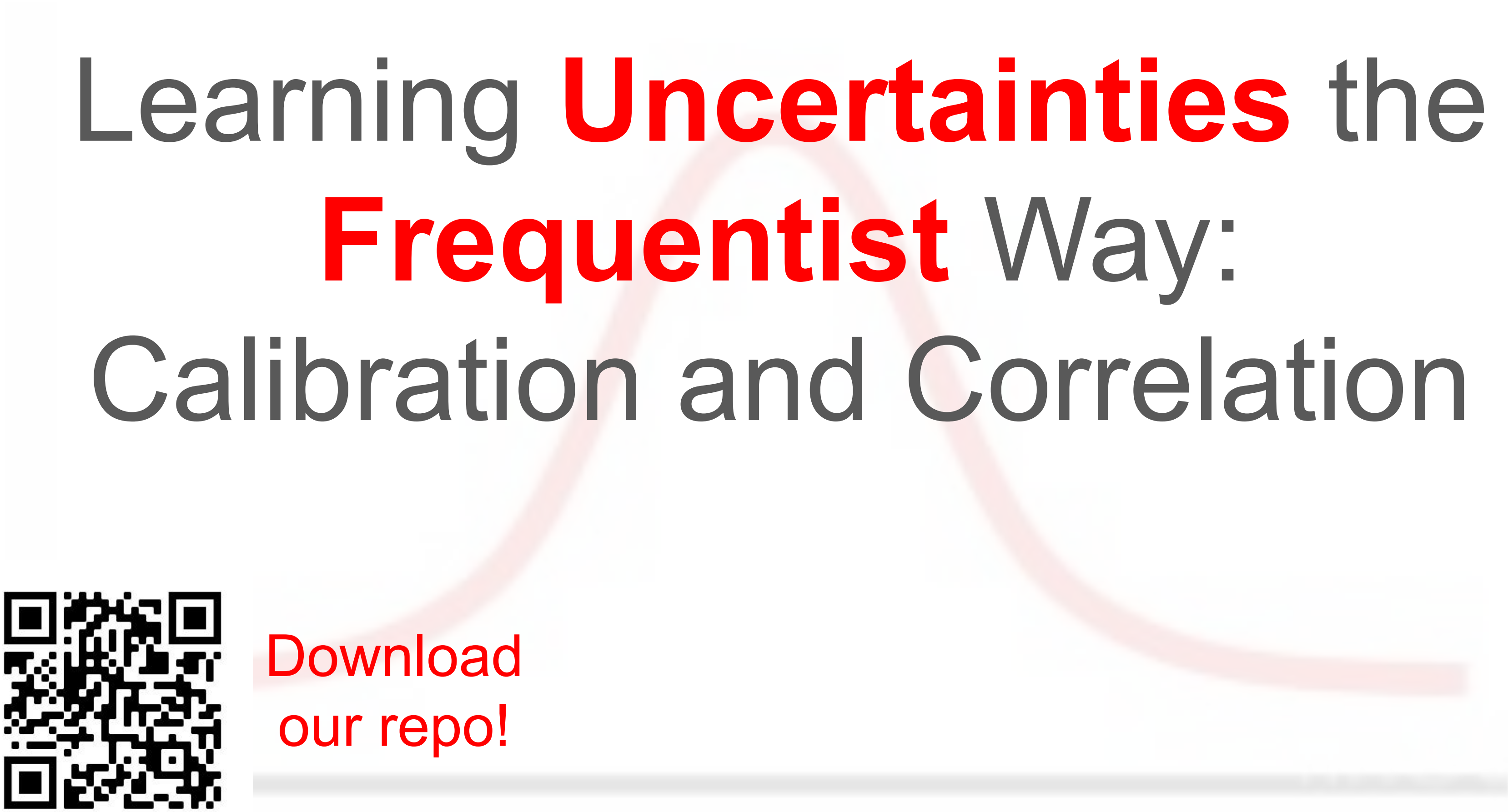
a

Things my **mortal physicist brain** can understand

Based on: [RG, Nachman, Thaler, 2205.03413]
[RG, Nachman, Thaler, 2205.05084]

Part I

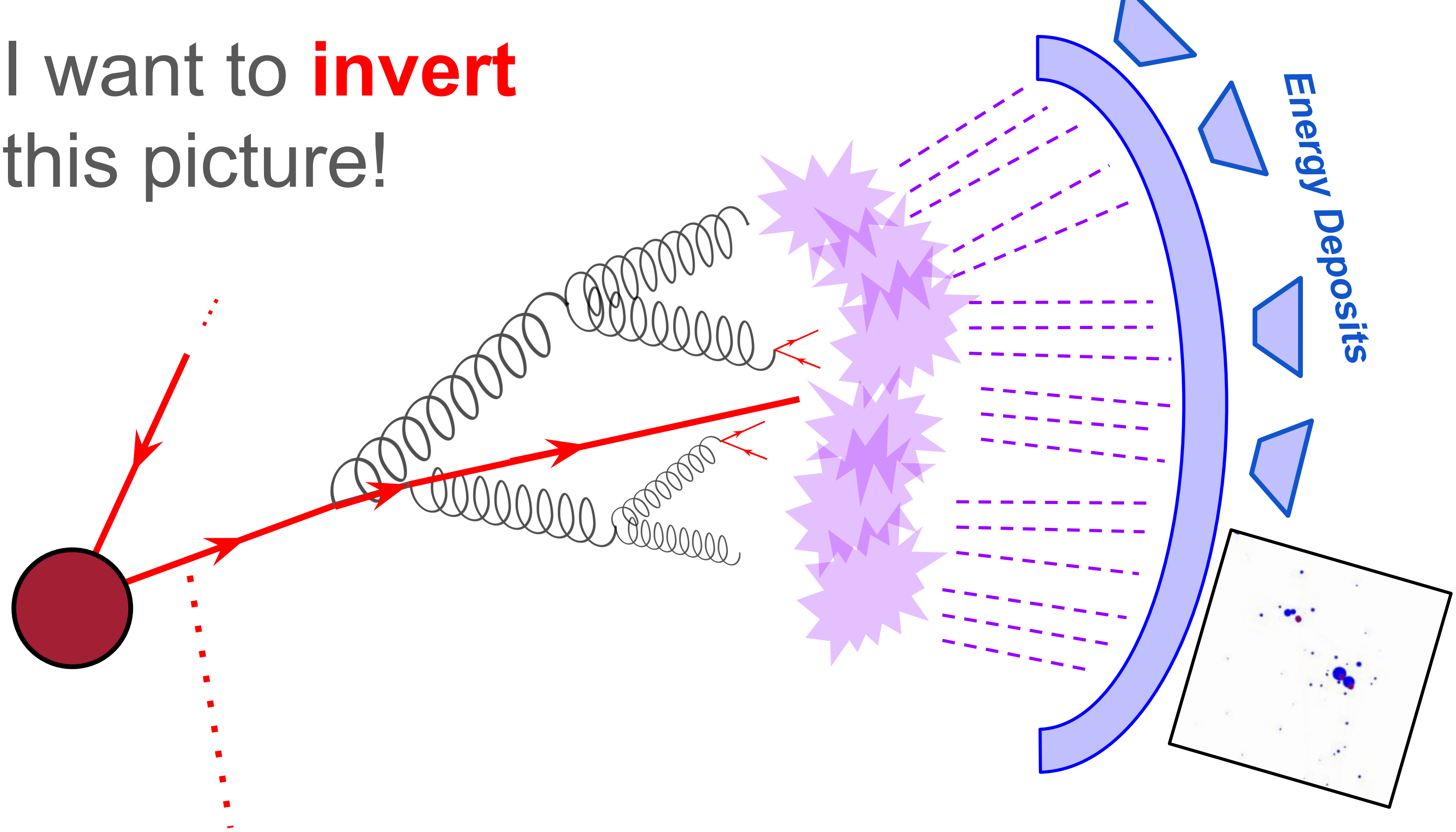
Learning Uncertainties the Frequentist Way: Calibration and Correlation



Download our repo!

Try *pip install GaussianAnsatz*

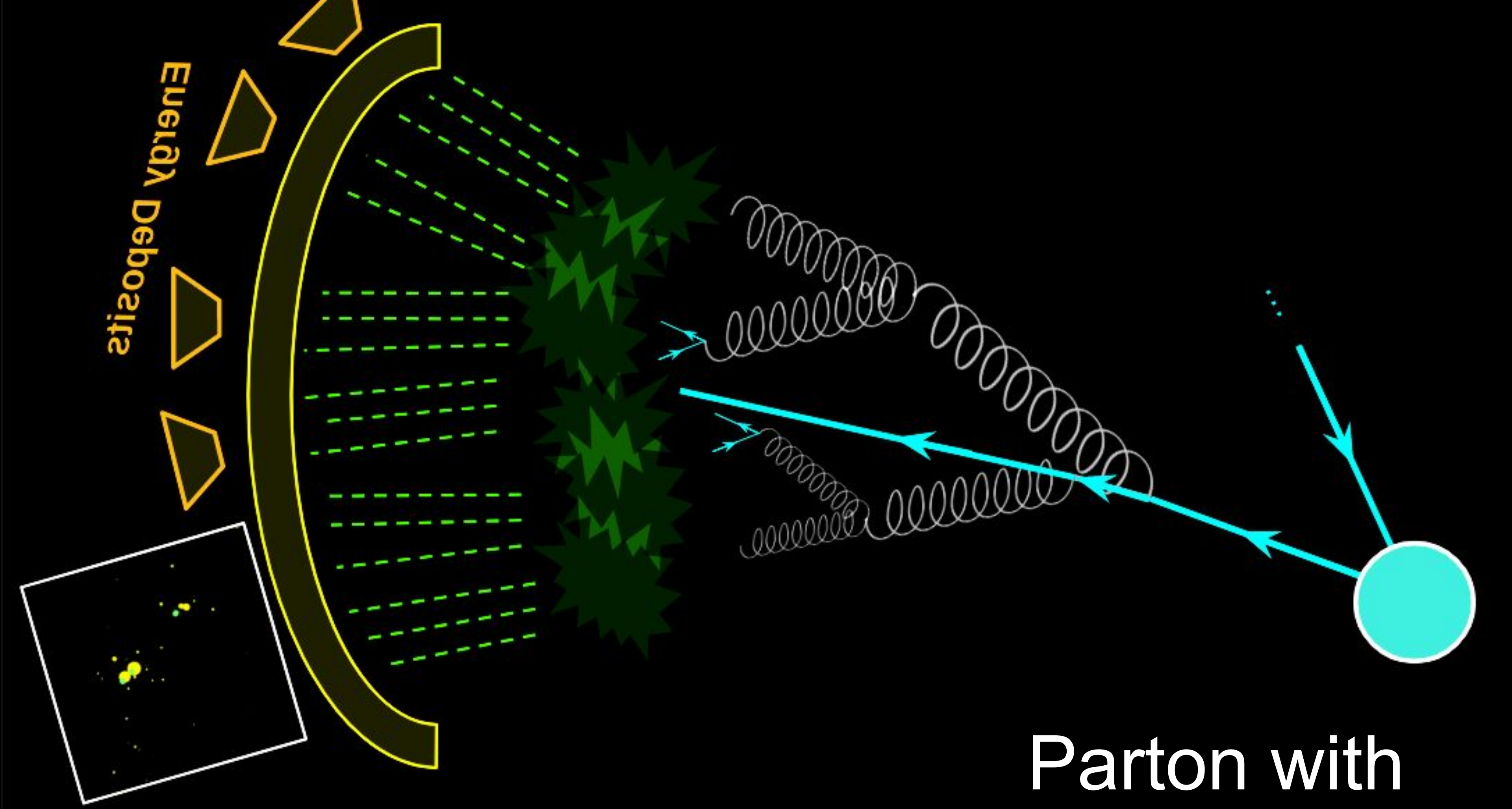
I want to **invert** this picture!



Parton with **true** energy ***z***

Model: Pythia + Geant4

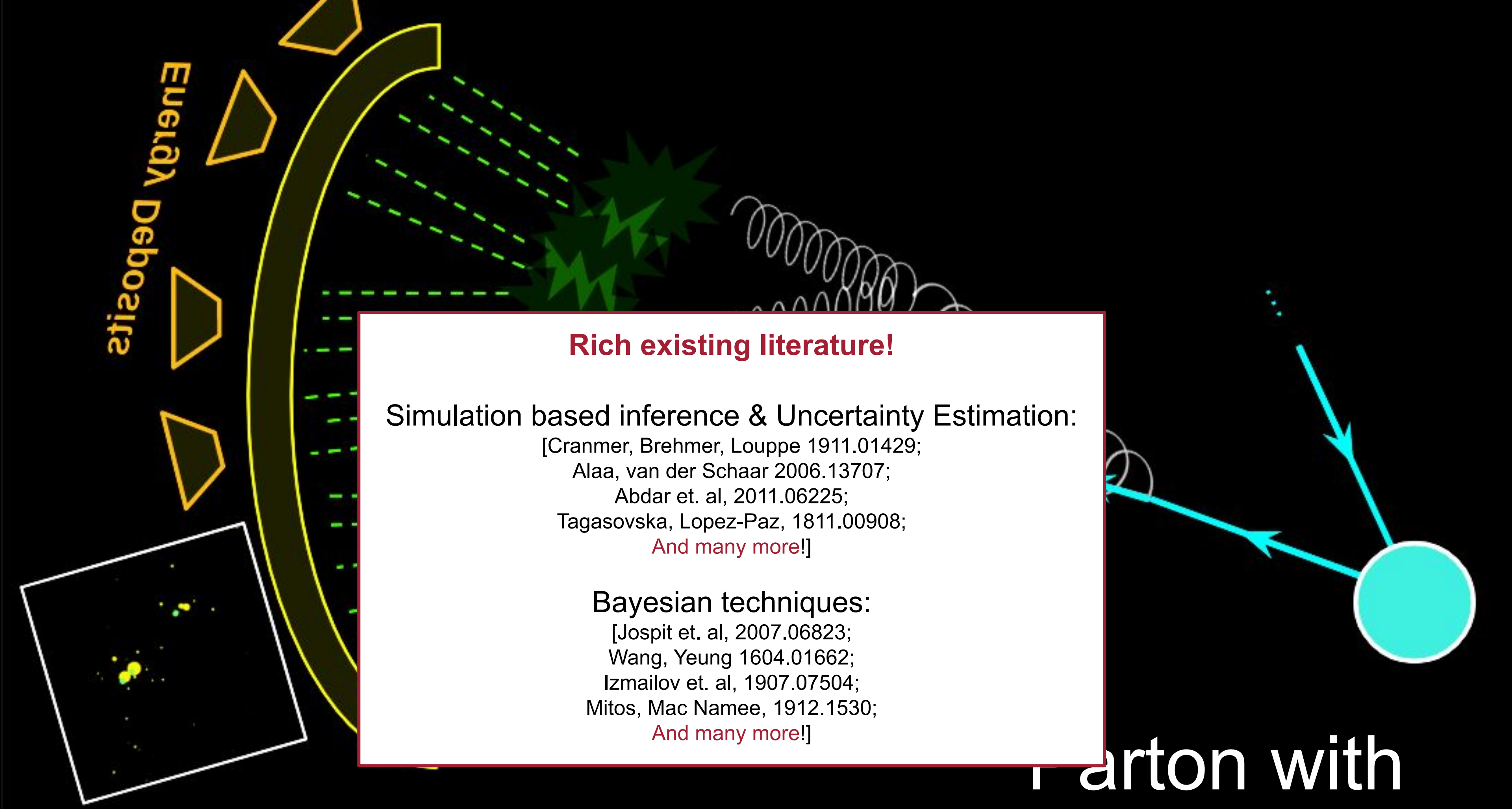
Measured particles ***x***



Parton with **estimated** energy $\hat{z}$

Measured particles x

What function does this?
Machine learn it!



Rich existing literature!

Simulation based inference & Uncertainty Estimation:
[Cranmer, Brehmer, Louppe 1911.01429;
Alaa, van der Schaar 2006.13707;
Abdar et. al, 2011.06225;
Tagasovska, Lopez-Paz, 1811.00908;
And many more!]

Bayesian techniques:
[Jospit et. al, 2007.06823;
Wang, Yeung 1604.01662;
Izmailov et. al, 1907.07504;
Mitos, Mac Namee, 1912.1530;
And many more!]

Measured particles x

What function does this? Machine learn it!

Parton with **estimated** energy $\hat{z}$

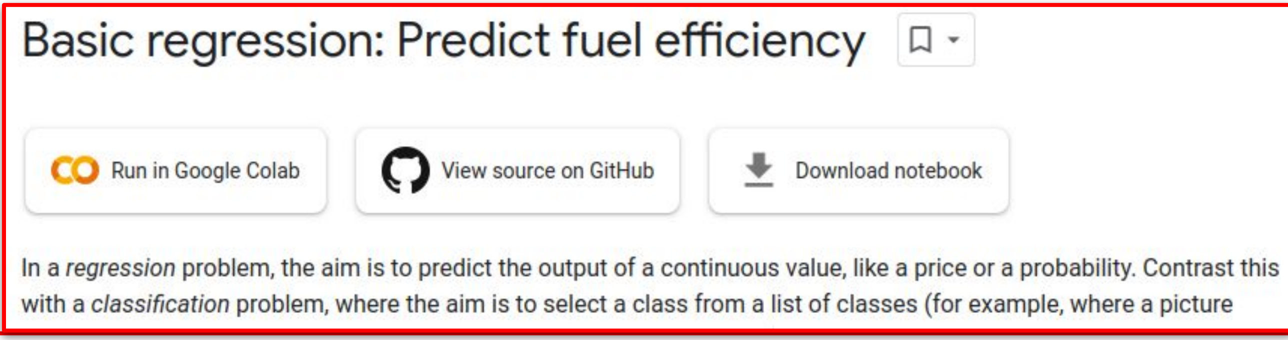
Problem Statement

Given a training set of (*x*, *z*) pairs, can we machine learn a function *f* such that *f(x)* = *z*? With uncertainties?

Yes. Extremely easily. This is just bread-and-butter least-squares regression with a neural network *g*:

$$\underset{g}{\operatorname{argmin}}\, \mathbb{E}_{\text{train}}[(g(X)-Z)^2]$$

It's the first thing you learn how to do !



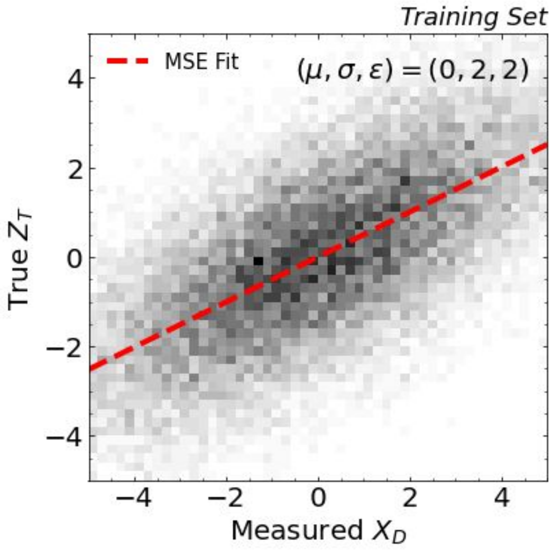
Basic regression: Predict fuel efficiency

In a *regression* problem, the aim is to predict the output of a continuous value, like a price or a probability. Contrast this with a *classification* problem, where the aim is to select a class from a list of classes (for example, where a picture

Problem Statement

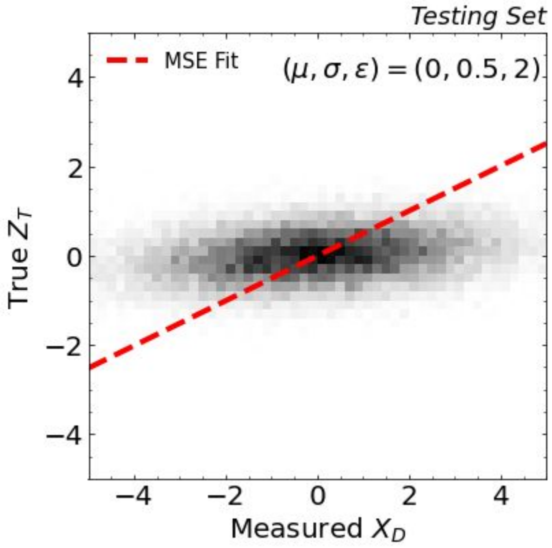
Given a training set of … e learn a function *f* such that *f(x)* = *z*? Wit…

Not so fast!



Same "detector" sim *p(x|z)*, only different priors *p(z)*!

This doesn't make a good, universal calibration.



It's the first thing

Upon closer inspection, MSE gives us prior dependent neural regressions! Is this what we really want? Back to the drawing board!

In a regression problem, the aim is to predict the output of a continuous value, like a price or a probability. Contrast this with a *classification* problem, where the aim is to select a class from a list of classes (for example, where a picture

Calibration

Let’s agree on what makes a good calibration, then design a loss for it!

1. **Closure**: On average, *f(x)* should be correct for each *x*! That is, *f* is **unbiased**.

$$\begin{aligned} b(z) &= \mathbb{E}_{\text{test}}[f(X) - z | Z = z] \\ &= 0 \end{aligned}$$

2. **Universality**: *f(x)* should not depend on the choice of sampling for *z*. That is, *f* is **prior-independent**.

$$f \text{ depends only on } \underline{p(x|z)}, \text{ and not } p(z)$$

Likelihood: Detector simulation, noise model, etc

What if the detector simulation is imperfect? Ask me later!

Calibration

Our network should make it easy to extract unbiased estimates! (e.g. via maximum likelihood)

n design a loss for it!

1. **Closure**: On average, *f(x)* should be correct for each *x*! That is, *f* is **unbiased**.

$$b(z) = \mathbb{E}_{\text{test}}[f(X) - z | Z = z] \\ = 0$$

Our loss should give us this likelihood!

2. **Universality**: *f(x)* should not depend on the choice of sampling for *z*. That is, *f* is **prior-independent**.

f depends only on $\underline{p(x|z)}$, and not $p(z)$

Likelihood: Detector simulation, noise model, etc

What if the detector simulation is imperfect? Ask me later!

[Belghazi, Baratin, Rajeswar, Ozair, Bengio, Courville, Hjelm, 1801.04062;
Le, Nguyen, Phung, 1711.01744
Nowozin; Cseke, 1606.00709]

Learning the likelihood

The **Donsker-Varadhan Representation (DVR)** of the KL divergence has been used in the statistics literature for mutual information estimation

$$\mathcal{L}_{\mathrm{DVR}}[T] = -\left(\mathbb{E}_{P_{XZ}}[T] - \log\left(\mathbb{E}_{P_X \otimes P_Z}\left[e^T\right]\right)\right)$$

Interestingly, a nonlocal loss!

Strict bound on *I(X;Z)*

What we want!

Minimized when $$T(x,z) = \log \frac{p(x|z)}{p(x)} + c$$

Unimportant

Lots of other losses also work, but DVR has very nice convergence properties - ask me later!

Inference using *T*

We can use a neural net to parameterize *T(x,z)*, and use standard gradient descent techniques to minimize the DVR loss. Then we can do …

$$\hat{z}(x) = \underset{z}{\text{argmax}}\, T(x,z)$$

Inference

$$\left[\hat{\sigma}_z^2(x)\right]_{ij} = -\left[\frac{\partial^2 T(x,z)}{\partial z_i\, \partial z_j}\right]^{-1}\Bigg|_{z=\hat{z}}$$

Gaussian Uncertainty Estimation

BUT!

- Maxima are hard to estimate – even *more* gradient descent!
- Second derivatives are sensitive to the choice of activations in *T* – ReLU spoils everything!

We solve both problems with the **Gaussian Ansatz**

The Gaussian Ansatz

Parameterize *T(x,y)* in the following way (the **Gaussian Ansatz**):

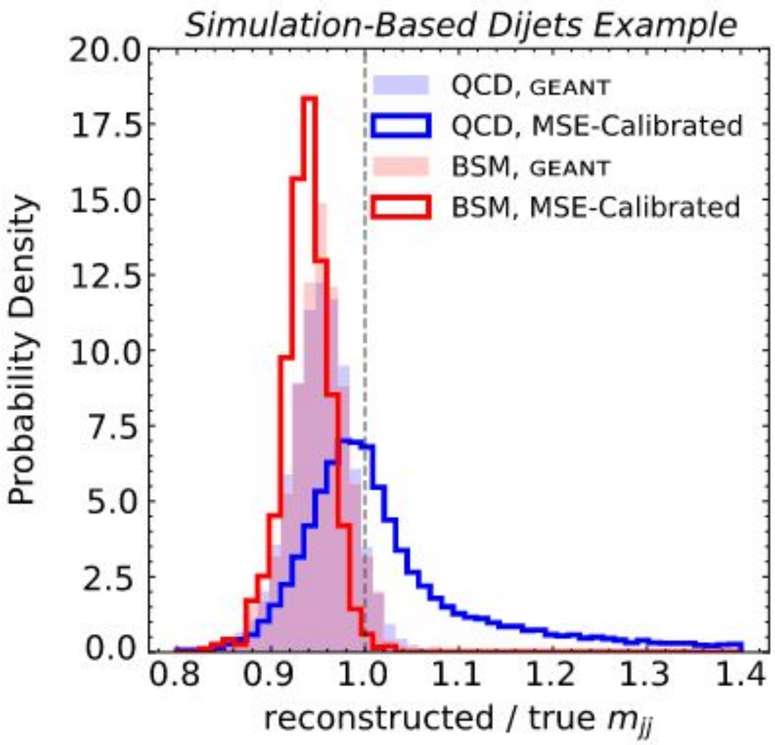
$$\begin{aligned} T(x,z) = A(x) \\ &+ \big(z - B(x)\big) \cdot D(x) \\ &+ \frac{1}{2}\big(z - B(x)\big)^T \cdot C(x,z) \cdot \big(z - B(x)\big) \end{aligned}$$

Where *A(x), B(x), C(x,z), and D(x)* are learned functions. Then, if *D→0*, our inference and (Gaussian) uncertainties are given by …

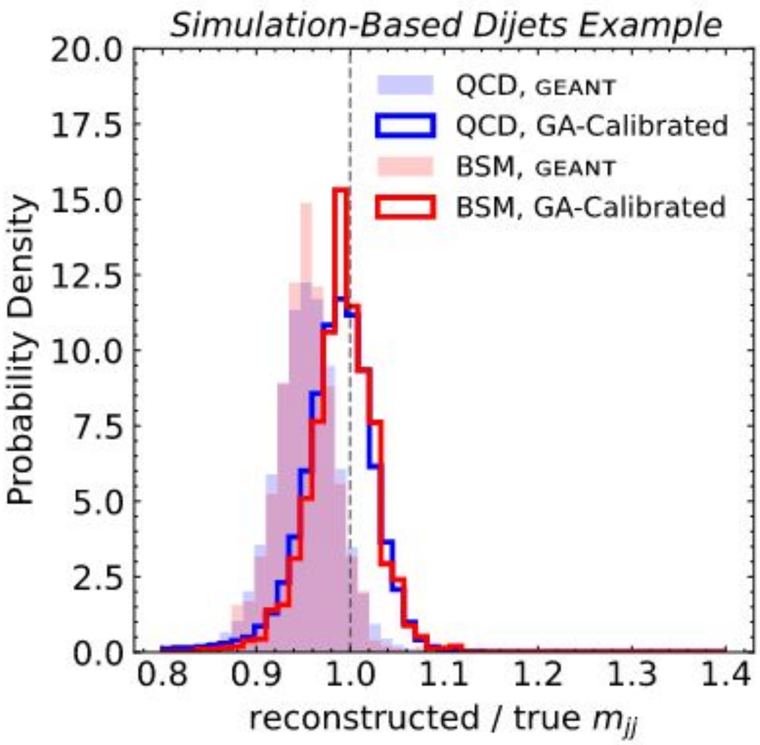
$$\hat{z}(x) = B(x) \qquad \hat{\sigma}_z^2(x) = -\big[C(x, B(x))\big]^{-1}$$

No additional post processing or numerical estimates required!

Example 1: QCD and BSM Dijets



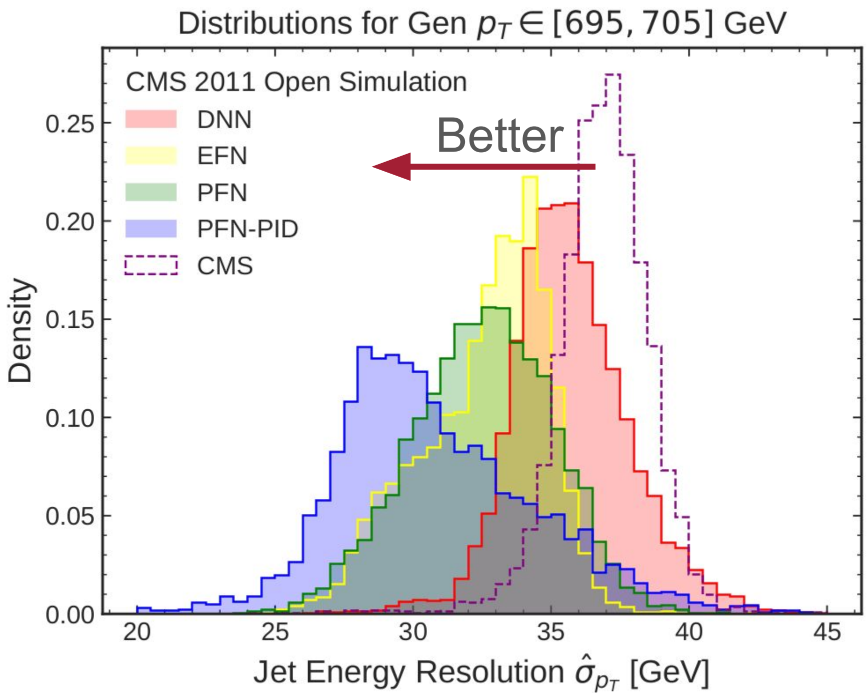
(Left) MSE-fitted network.



(Right) Gaussian Ansatz-fitted network

Clever loss function choice: *Prior dependence is built-in!*

Example 2: Jet Energy Resolution



Best of both worlds: Using ML to extract more information* than hand-crafted features, while still being able to extract resolutions in a prior-independent and unbiased way!

*I mean information literally. Ask me later about the cool information theory of the DV loss!

Part I Summary

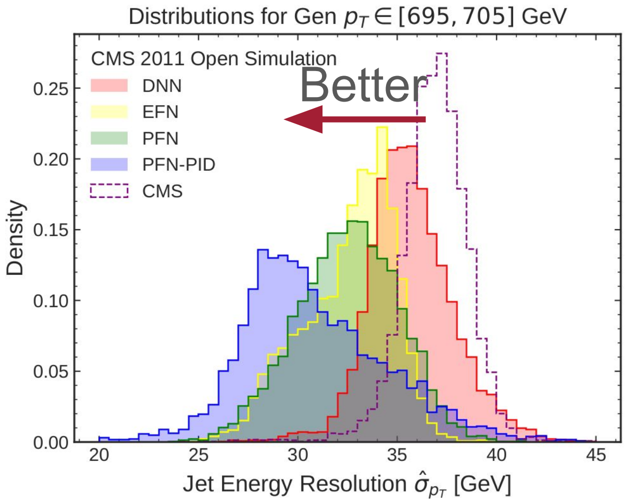
By choosing the following loss:

$$\mathcal{L}_{\mathrm{DVR}}[T] = -\left(\mathbb{E}_{P_{XZ}}[T] - \log\left(\mathbb{E}_{P_X \otimes P_Z}\left[e^T\right]\right)\right)$$

With the following network parameterization:

$$\begin{aligned} T(x,z) = A(x) \\ &+ (z - B(x)) \cdot D(x) \\ &+ \frac{1}{2}(z - B(x))^T \cdot C(x,z) \cdot (z - B(x)) \end{aligned}$$

We can get **unbiased**, **prior-independent** inference, and easily extract **maximum likelihood** and **resolution** estimates!



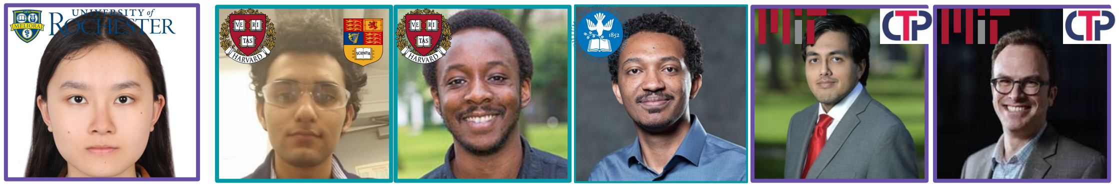
Based on: [Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]
[**RG**, Thaler, Wu, WIP]

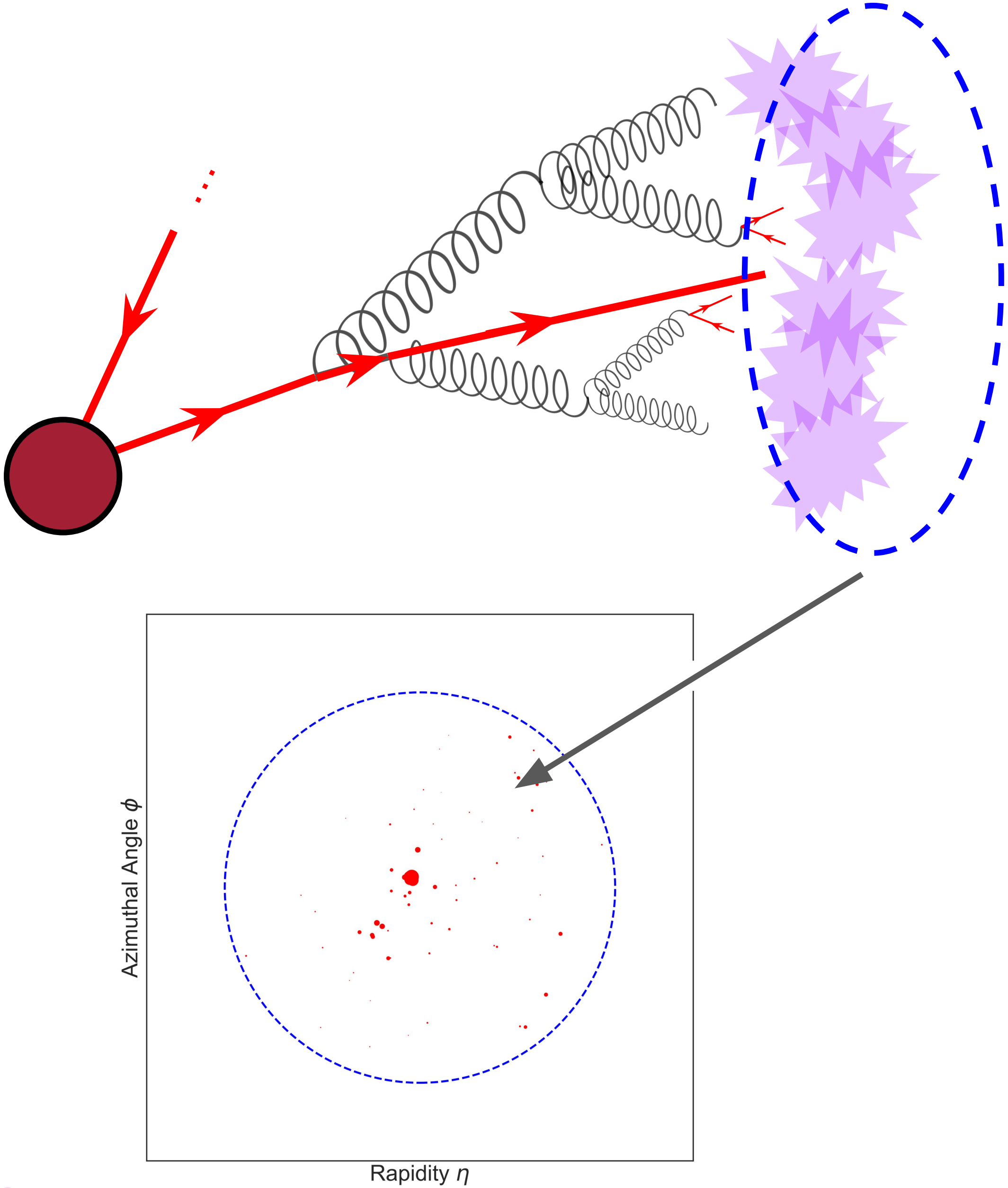
Part II

Can you Hear the **Shape** of a Jet?

Download our repo!

Try *`pip install pyshaper`*

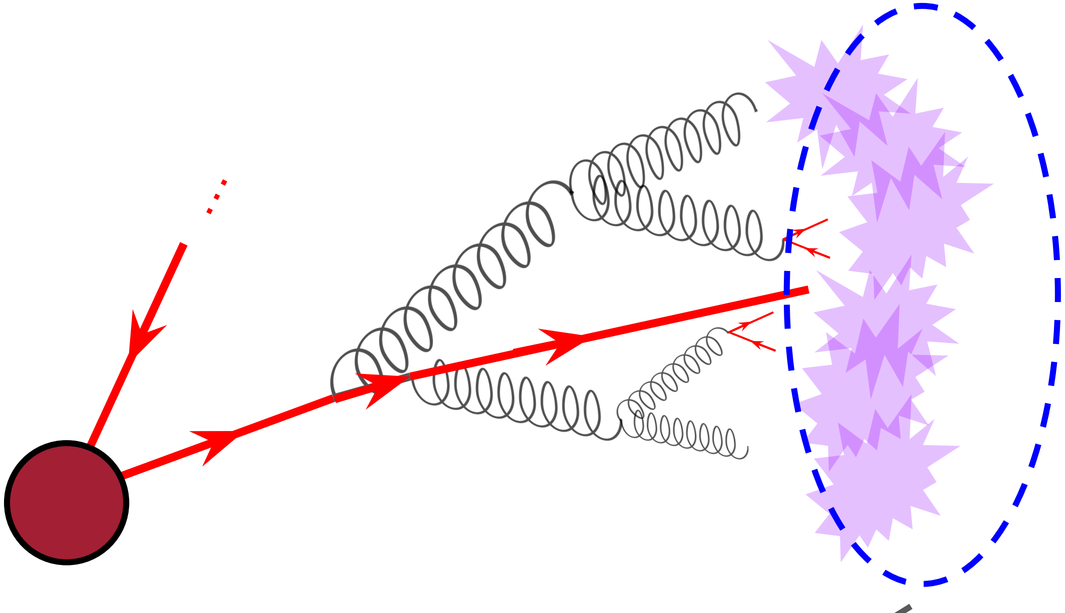




How "*big*" is this jet?

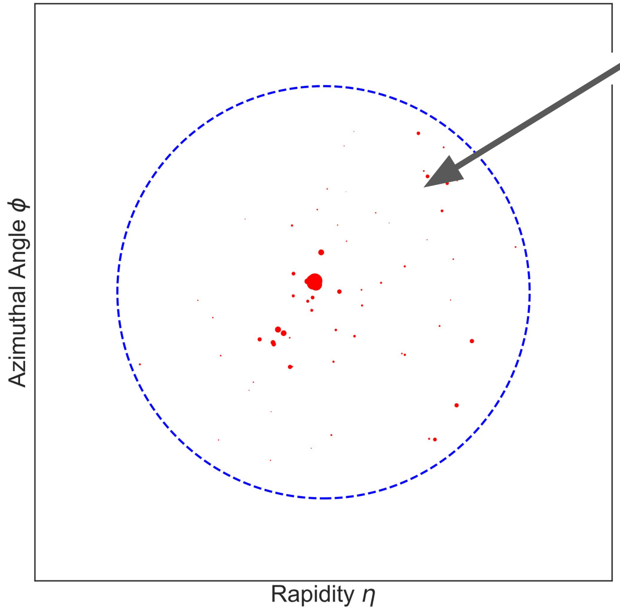
… What do we even mean by the "size" of a jet?

I don't mean the fixed size of a jet algorithm, e.g. AK4, but rather a dynamical notion of a jet size!



How “*big*” is this jet?

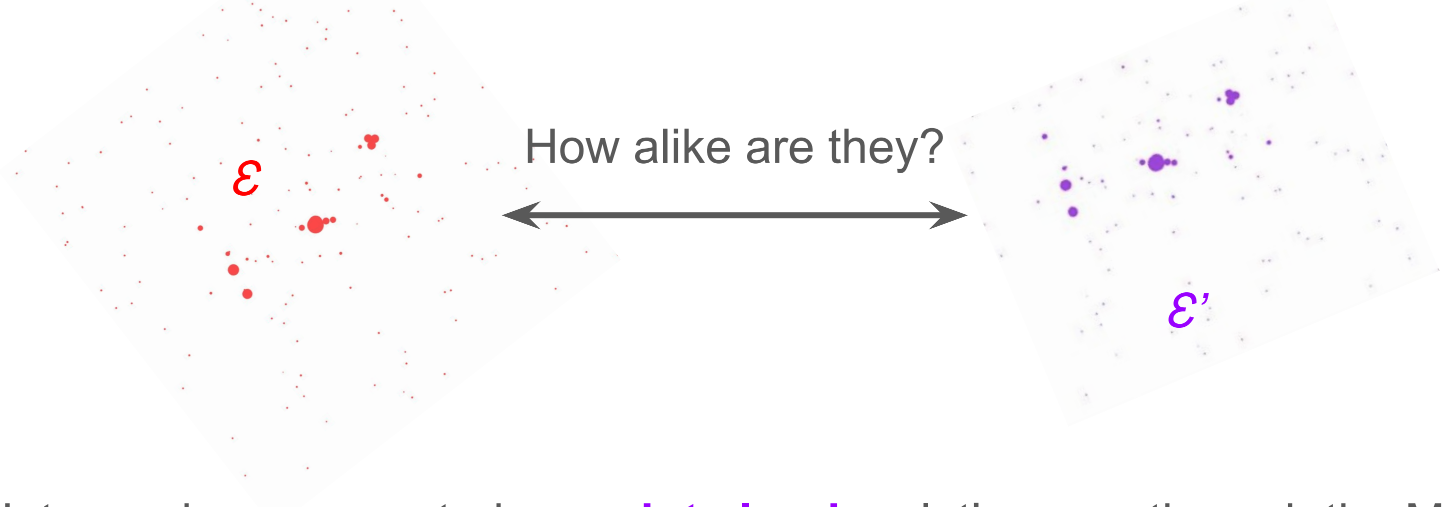
… What do we even mean by the “size” of a jet?



By following this line of thinking, we’ll be able to come up with a new class of QCD observables!

I don’t mean the fixed size of a jet algorithm, e.g. AK4, but rather a dynamical notion of a jet size!

More generally: When are jets similar?

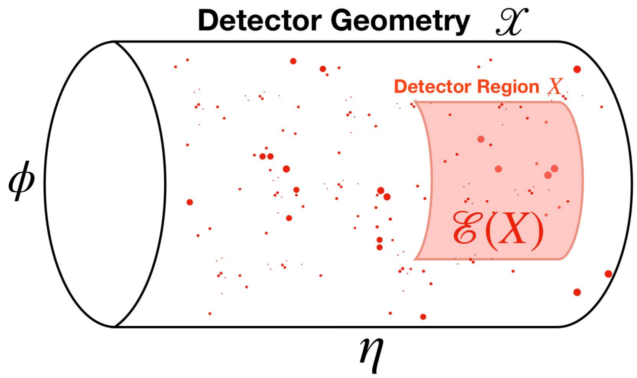


Jets can be represented as **point clouds** – let's scour through the ML and computer vision literature for a metric on point clouds we like![*]

Detector Coordinate (η, ϕ)

$$\mathcal{E}(y) = \sum_i z_i \delta(y - y_i)$$

Energy Fraction E_i / E_{Tot}



[*] Or better yet, construct one!

The Wasserstein Metric

Let’s demand the following reasonable properties of our metric on point clouds:

1. … is nonnegative and finite
2. … is **IRC-safe**[*] (Calculable and robust)
3. … is translationally invariant
4. … is invariant to particle labeling
5. respects the detector metric ***faithfully***[**]

[*] Ask me for more details on this offline!

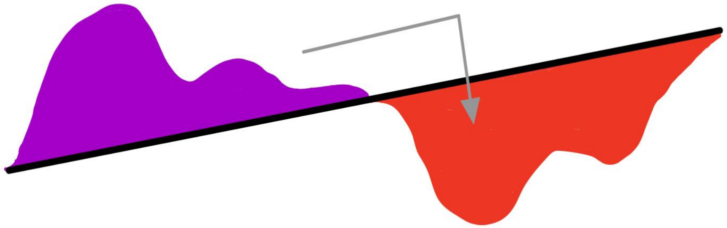
[**] Preserves distances between *extended* objects, not just points

Proved in [Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]
Ask me later for proof details!
See also: [Komiske, Metodiev, Thaler, 1902.02346,
Komiske, Metodiev, Thaler, 2004.04159]

The Wasserstein Metric

Let's demand the following reasonable properties of our metric on point clouds:

1. … is nonnegative and finite
2. … is **IRC-safe*** (Calculable and robust)
3. … is translationally invariant
4. … is invariant to particle labeling
5. respects the detector metric ***faithfully*****



EMD = Work done to move "dirt" optimally

It turns out that the *only* metric that satisfies this is the **Wasserstein Metric / EMD**!

$$\text{EMD}^{(\beta,R)}(\mathcal{E},\mathcal{E}') = \min_{\pi\in\mathcal{M}(\mathcal{X}\times\mathcal{X})}\left[\frac{1}{\beta R^\beta}\left\langle \pi, d(x,y)^\beta\right\rangle\right] + |\Delta E_{\text{tot}}|$$

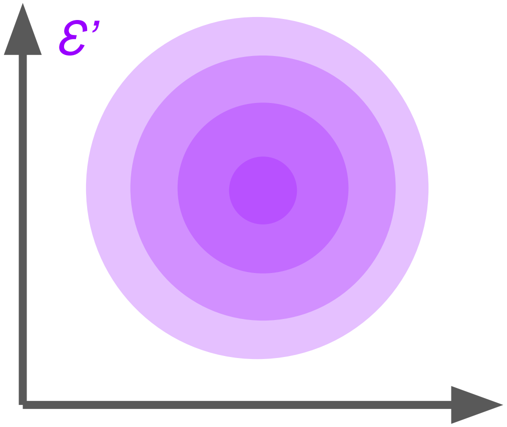
$$\pi(\mathcal{X},Y)\leq\mathcal{E}'(Y),\quad \pi(X,\mathcal{X})\leq\mathcal{E}(X),\quad \pi(\mathcal{X},\mathcal{X}) = \min(E_{\text{tot}},E'_{\text{tot}})$$

A staple of computer vision ML is also useful for jets!

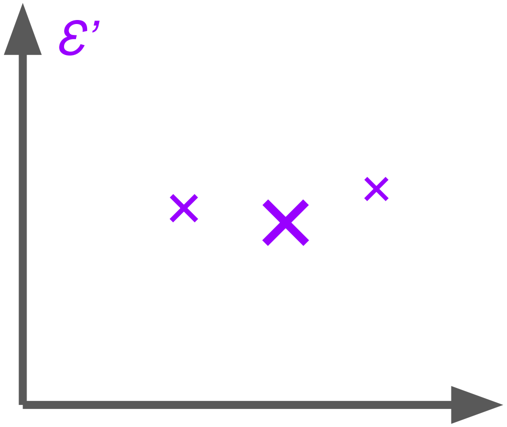
Shapes as Energy Flows

Energy flows don’t have to be real events – they can be *any* energy distribution in detector space, or **shape**.

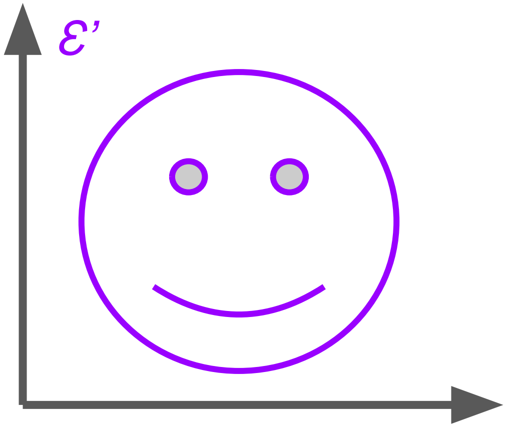
Can make anything you want! Even continuous or complicated shapes.
(Or, something you calculate in perturbative QCD)



Shape = 2D Gaussian



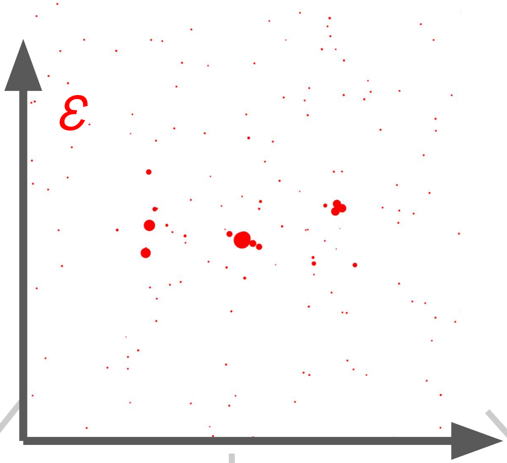
Shape = 3 Points



Shape = Smile

Shapiness

The EMD between a real event or jet $\mathcal{E}$ and idealized shape $\mathcal{E}'$ is the [shape]iness of $\mathcal{E}$ – a well defined IRC-safe observable!

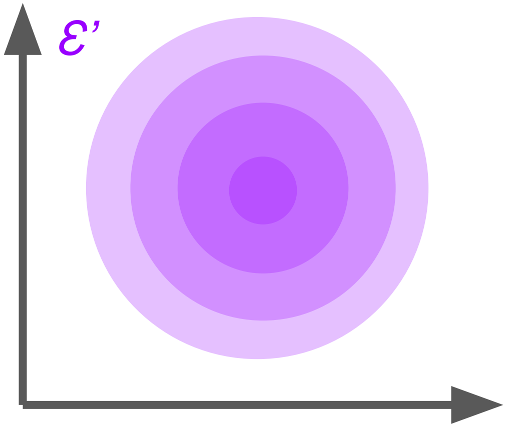


Answers the question: “How much like the shape $\mathcal{E}'$ is my event $\mathcal{E}$?”

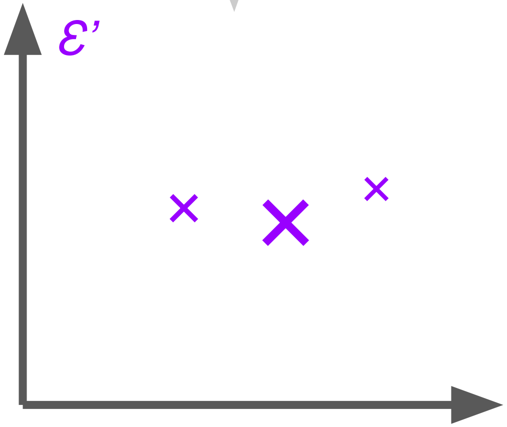
Med EMD($\mathcal{E}$, $\mathcal{E}'$)
“Gaussiness”

Low EMD($\mathcal{E}$, $\mathcal{E}'$)
“3-Pointiness”
AKA “3-Subjettiness”

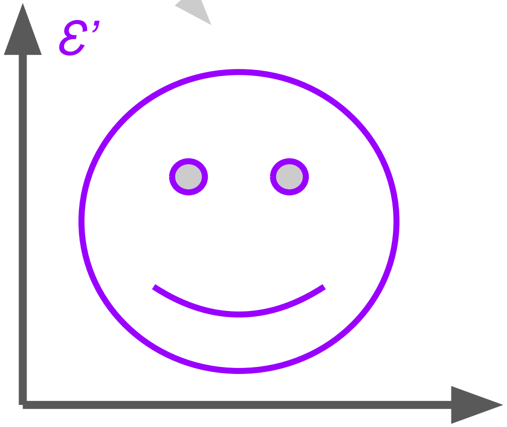
High EMD($\mathcal{E}$, $\mathcal{E}'$)
“Smileyness”



Shape = 2D Gaussian



Shape = 3 Points



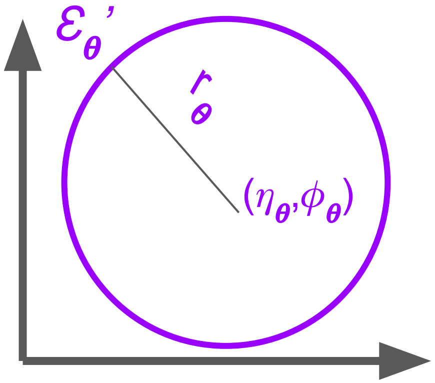
Shape = Smile

Mathematical Details - **Shapiness**

Rather than a single shape, consider a **parameterized manifold $\mathcal{M}$** of **energy flows.**

e.g. The manifold of uniform circle energy flows:

$$\mathcal{E}_{\boldsymbol{\theta}}'(y) = \begin{cases} \frac{1}{2\pi r_\theta} & |\vec{y} - \vec{y}_\theta| = r_\theta \\ 0 & |\vec{y} - \vec{y}_\theta| \neq r_\theta \end{cases}$$


$\mathcal{E}_{\boldsymbol{\theta}}$'
$r_{\boldsymbol{\theta}}$
$(\eta_{\boldsymbol{\theta}},\phi_{\boldsymbol{\theta}})$

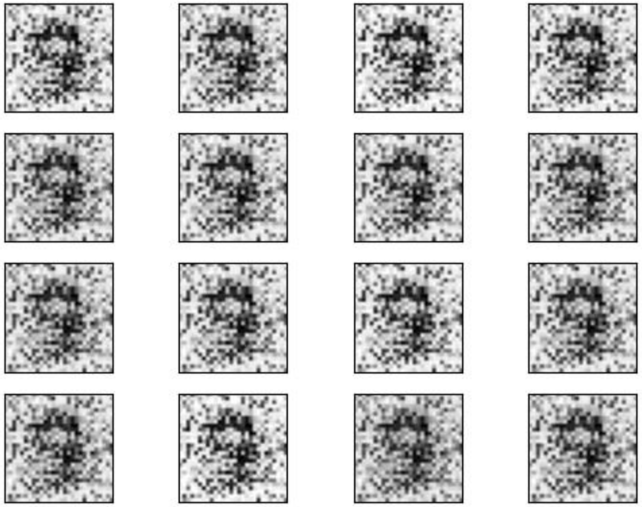
Then, for an event $\mathcal{E}$, define the **shapiness $\mathcal{O}_{\mathcal{M}}$** and **shape parameters $\theta_{\mathcal{M}}$**, given by:

$$\mathcal{O}_{\mathcal{M}}(\mathcal{E}) \equiv \min_{\mathcal{E}_\theta \in \mathcal{M}} \text{EMD}^{(\beta,R)}(\mathcal{E}, \mathcal{E}_\theta)$$

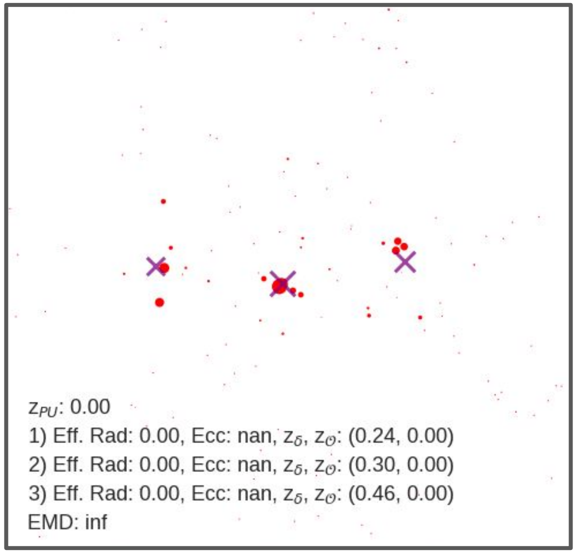
$$\theta_{\mathcal{M}}(\mathcal{E}) \equiv \underset{\mathcal{E}_\theta \in \mathcal{M}}{\text{argmin}}\, \text{EMD}^{(\beta,R)}(\mathcal{E}, \mathcal{E}_\theta)$$

This is basically just a W-GAN!

Fitting a 2D image distribution with a fully flexible generative neural network



Fitting a 2D point cloud with a small parameterized generator



Rather than using a fully-flexible neural network to fit our distributions with the Wasserstein metric, as in a W-GAN, we craft specific parameters of interest!

[P. Komiske, E. Metodiev, and J. Thaler, 2004.04159;
J. Thaler, and K. Van Tilburg, 1011.2268;
I. W. Stewart, F. J. Tackmann, and W. J. Waalewijn, 1004.2489.;
S. Brandt, C. Peyrou, R. Sosnowski and A. Wroblewski, PRL 12 (1964) 57-61;
C. Cesarotti, and J. Thaler, 2004.06125]

Observables ⇔ Manifolds of Shapes

Observables can be specified solely by defining a **manifold of shapes**:

$$\mathcal{O}_{\mathcal{M}}(\mathcal{E}) \equiv \min_{\mathcal{E}_\theta \in \mathcal{M}} \text{EMD}^{(\beta,R)}(\mathcal{E}, \mathcal{E}_\theta),$$

$$\theta_{\mathcal{M}}(\mathcal{E}) \equiv \operatorname*{argmin}_{\mathcal{E}_\theta \in \mathcal{M}} \text{EMD}^{(\beta,R)}(\mathcal{E}, \mathcal{E}_\theta),$$

Many well-known observables* already have this form!

Observable	Manifold of Shapes
N-Subjettiness	Manifold of *N*-point events
N-Jettiness	Manifold of *N*-point events with floating total energy
Thrust	Manifold of back-to-back point events
Event / Jet Isotropy	Manifold of the single uniform event

… and more!

All of the form “How much like **[shape]** does my **event** look like?”
Generalize to *any* shape.

*These observables are usually called event shapes or jet shapes in the literature – we are making this literal!

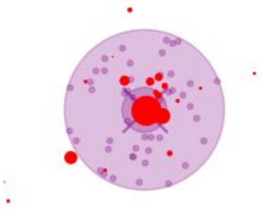
Hearing Jets Ring

(and Disk, and Ellipse)

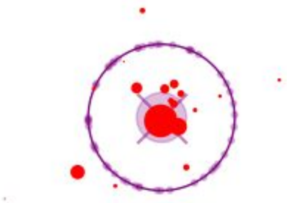
$\mathcal{O}_\mathcal{M}(\mathcal{E})$ answers: “How much like a shape in $\mathcal{M}$ does my event $\boldsymbol{\mathcal{E}}$ look like?”

$\theta_\mathcal{M}(\mathcal{E})$ answers: “Which shape in $\mathcal{M}$ does my event $\boldsymbol{\mathcal{E}}$ look like?”

Can define complex manifolds to probe increasingly subtle geometric structure!

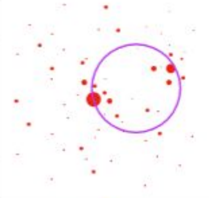
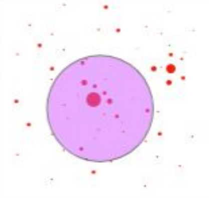
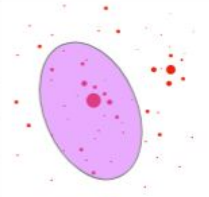
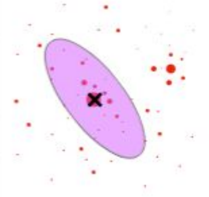
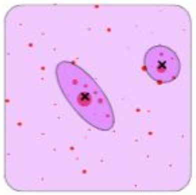


Disk + δ-function



Ring + δ-function

Probing **collinear** (δ-function) and **soft** (shape) structure!

Shape	Specification	Illustration
Ringiness $\mathcal{O}_R$	**Manifold of Rings** $\mathcal{E}_{x_0,R_0}(x) = \frac{1}{2\pi R_0}$ for $\lvert x - x_0\rvert = R_0$ $x_0 =$ Center, $R_0 =$ Radius	
Diskiness $\mathcal{O}_D$	**Manifold of Disks** $\mathcal{E}_{x_0,R_0}(x) = \frac{1}{\pi R_0^2}$ for $\lvert x - x_0\rvert \leq R_0$ $x_0 =$ Center, $R_0 =$ Radius	
Ellipsiness $\mathcal{O}_E$	**Manifold of Ellipses** $\mathcal{E}_{x_0,a,b,\varphi}(x) = \frac{1}{\pi ab}$ for $x \in \text{Ellipse}_{x_0,a,b,\varphi}$ $x_0 =$ Center, $a, b =$ Semi-axes, $\varphi =$ Tilt	
(Ellipse Plus Point)iness	**Composite Shape** $\mathcal{O}_E \oplus \tau_1$ Fixed to same center x_0	
N-(Ellipse Plus Point)iness Plus Pileup	**Composite Shape** $N \times (\mathcal{O}_E \oplus \tau_1) \oplus \mathcal{I}$	

Some examples of new shapes you can define!

[see also L., B. Nachman, A. Schwartzman, C. Stansbury, 1509.02216;
see also B. Nachman, P. Nef, A. Schwartzman, M. Swiatlowski, C. Wanotayaroj, 1407.2922;
see also M. Cacciari, G. Salam, 0707.1378]

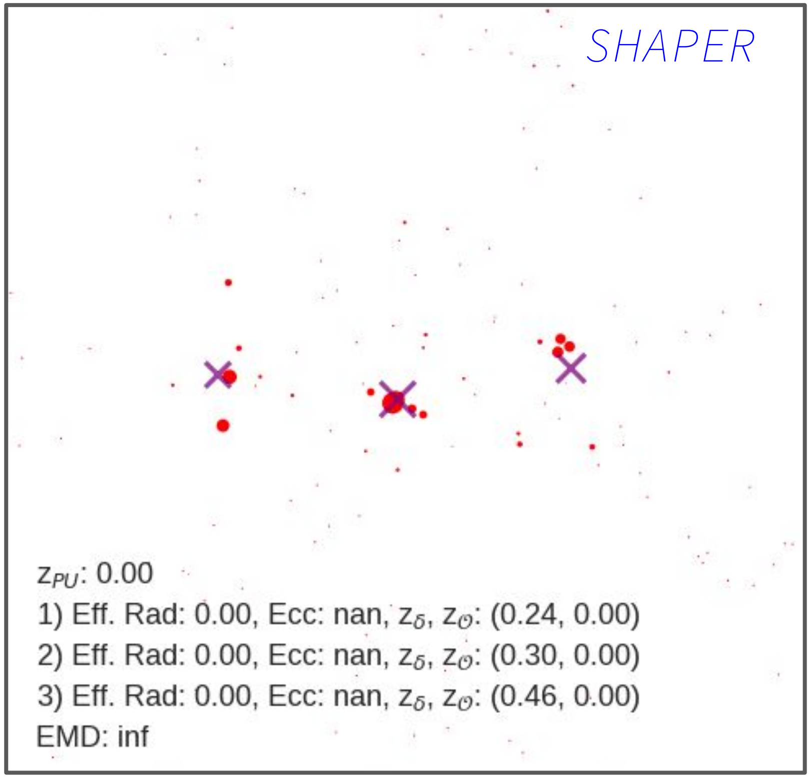
New IRC-Safe Observables

The *SHAPER* framework makes it easy to algorithmically invent new jet observables!

e.g. ***N-(*Ellipse+Point)iness+Pileup** as a jet algorithm:

- Learn jet centers + collinear radiation
- Dynamic jet radii (no *R* parameter!)
- Dynamic eccentricities and angles
- Dynamic jet energies
- Dynamic pileup (no z_{cut} parameter!!!)
- Learned parameters for discrimination

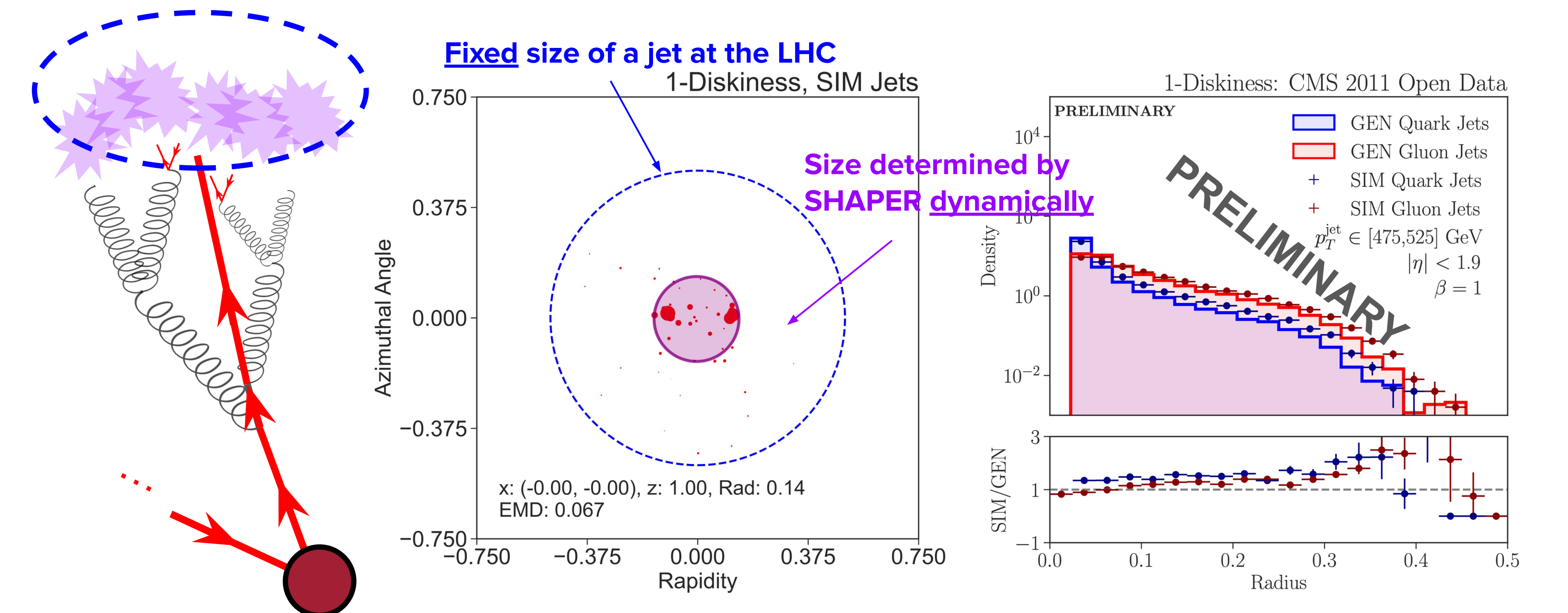
Can design custom specialized jet algorithms to learn jet substructure!



z_{PU}: 0.00
1) Eff. Rad: 0.00, Ecc: nan, z_δ, z_Θ: (0.24, 0.00)
2) Eff. Rad: 0.00, Ecc: nan, z_δ, z_Θ: (0.30, 0.00)
3) Eff. Rad: 0.00, Ecc: nan, z_δ, z_Θ: (0.46, 0.00)
EMD: inf

[RG, Thaler, Wu; WIP]

Back to our question: How Big are Jets?



Fixed size of a jet at the LHC

Size determined by SHAPER dynamically

Slide images from [Wu; APS 2023]

We can now answer this question in a precise way!

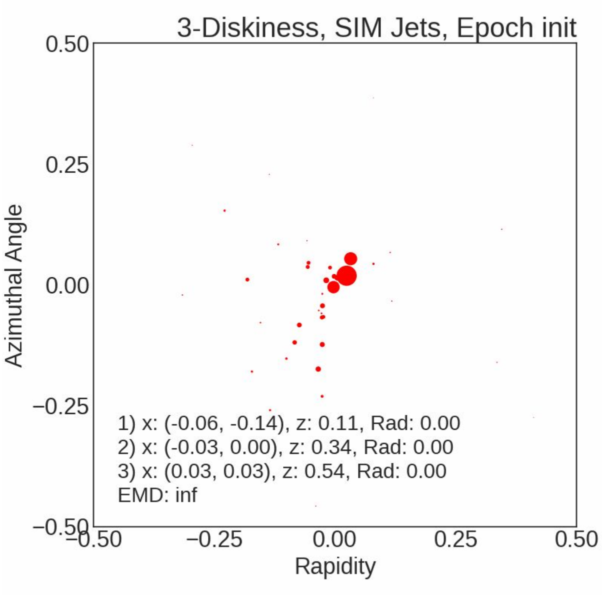
Part II Summary

By choosing the following loss function:

$$\mathrm{EMD}^{(\beta,R)}(\mathcal{E},\mathcal{E}') = \min_{\pi\in\mathcal{M}(\mathcal{X}\times\mathcal{X})}\left[\frac{1}{\beta R^\beta}\left\langle \pi, d(x,y)^\beta\right\rangle\right] + |\Delta E_{\mathrm{tot}}|$$

$$\pi(\mathcal{X},Y)\leq\mathcal{E}'(Y),\quad \pi(X,\mathcal{X})\leq\mathcal{E}(X),\quad \pi(\mathcal{X},\mathcal{X}) = \min(E_{\mathrm{tot}},E'_{\mathrm{tot}})$$

Based on IRC-safety and geometric faithfulness, we can build well-defined and robust observables that quantify *targeted* geometric properties of point clouds!



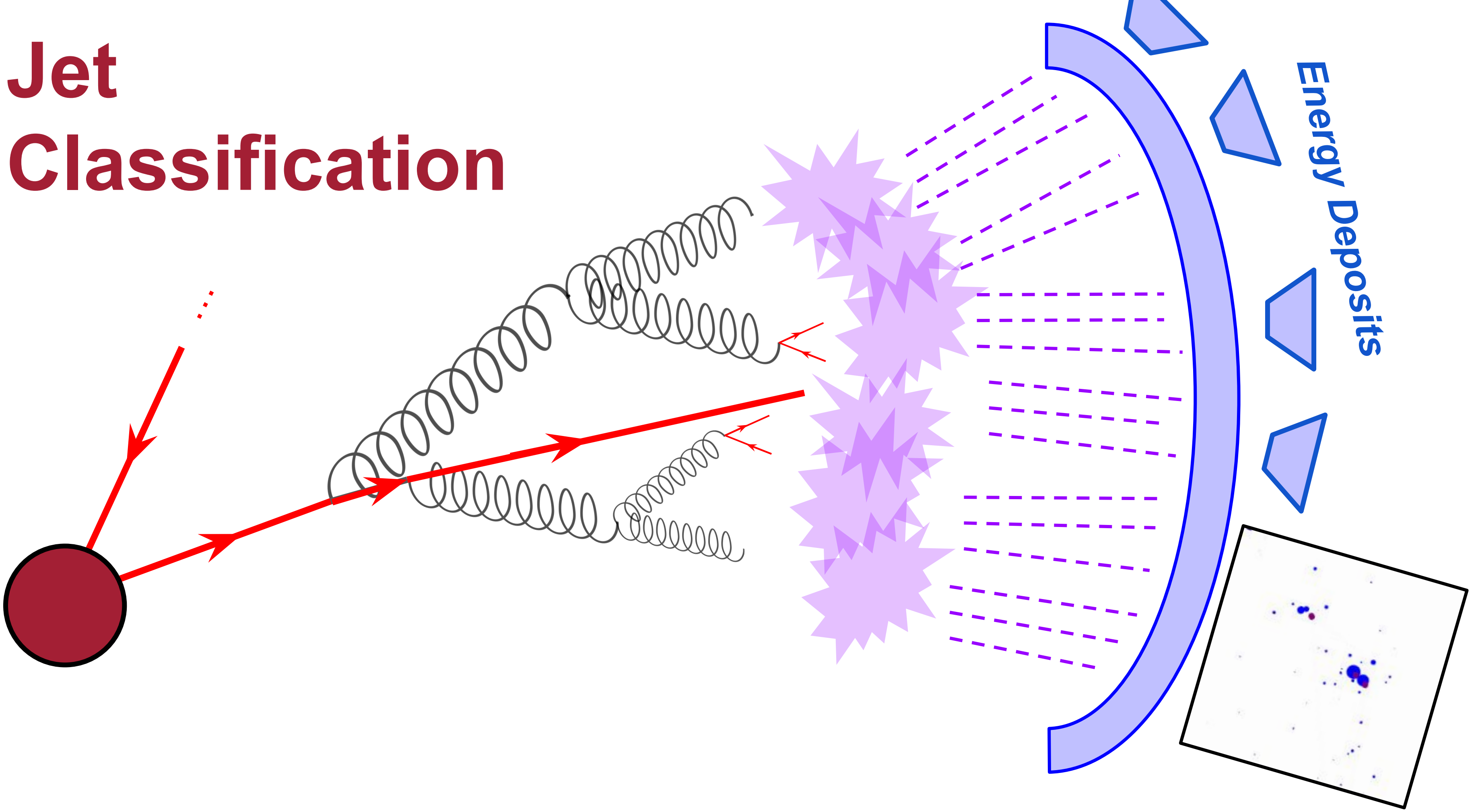
We can make jet shapes well defined!

Based on: [RG, Osathapan, Tasissa, Thaler, 2403.08854]

Part III

Moments of Clarity: Streamlining Latent Spaces

Jet Classification

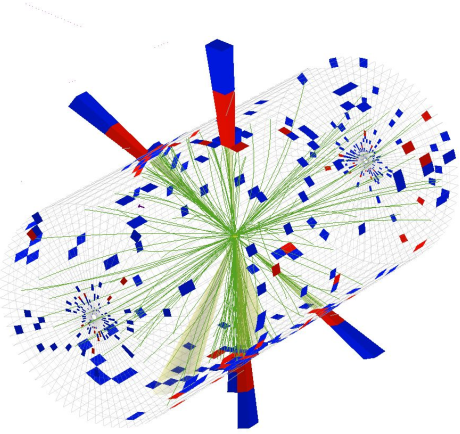


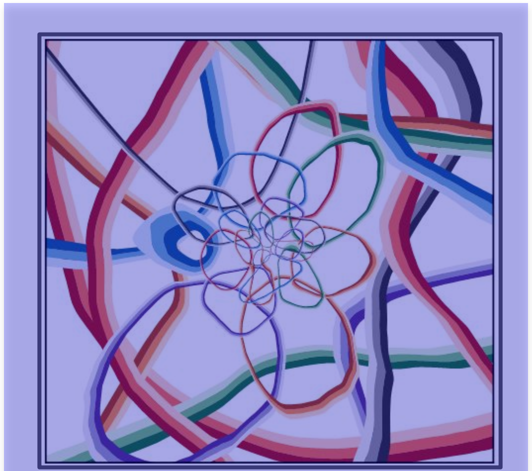
Given the final measured **point cloud**, was the **initiating parton** a **quark** or a **gluon**?

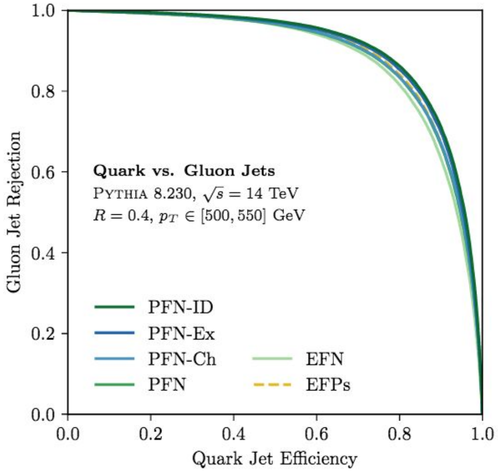
A staple machine learning task!

[Komiske, Metodiev, Thaler, 1810.05165]

Typical Machine Learning Setup







Collider Data → Latent Space → Observables

~1000 Dimensional

~10-100 Dimensional

~1-10 Dimensional

Pictured: An Energy Flow Network (**EFN**):

$$\mathcal{O}(\{p_1,\ldots,p_M\}) = F\left(\sum_{i=1}^{M} z_i \Phi(\hat{p}_i)\right)$$

(How) can we understand and constrain this?

(How) can we be more efficient?

Deep Learning on Point Clouds

The **Deep Sets Theorem** tells us how to parameterize functions on point clouds*:

Set of momenta

a = 1 … *L*, the *Latent Dimension* index

$$\mathcal{O}(\mathcal{P}) = F\left(\langle \phi^a \rangle_{\mathcal{P}}\right)$$

$$\langle \phi \rangle_{\mathcal{P}} \equiv \sum_i z_i \phi(\hat{p}_i)$$

The Deep Sets Theorem guarantees that any function on sets $\mathscr{P}$ can be written this way, for "sufficiently complex" *F* and ϕ and large enough *L.*

For this talk, I will focus primarily on Energy Flow Networks (EFNs) where the sums are energy-weighted.

*More sophisticated architectures still reduce to Deep Sets in some limit, e.g. graph networks

Deep Learning on Point Clouds

The **Deep Sets Theorem** tells us how to parameterize functions on point clouds*:

Set of momenta

a = 1 … *L*, the *Latent Dimension* index

$$\mathcal{O}(\mathcal{P}) = F\left(\langle\phi^a\rangle_\mathcal{P}\right)$$

$$\langle\phi\rangle_\mathcal{P} \equiv \sum_i z_i \phi(\hat{p}_i)$$

The Deep Sets Theorem guarantees that any function on sets $\mathscr{P}$ can be written this way, for “sufficiently complex” *F* and ϕ and large enough *L.*

But how complex do *F* and ϕ need to be? Can I reduce the number of latent dimensions?

For this talk, I will focus primarily on Energy Flow Networks (EFNs) where the sums are energy-weighted.

*More sophisticated architectures still reduce to Deep Sets in some limit, e.g. graph networks

Moment Pooling

$$\mathcal{O}(\mathcal{P}) = F\left(\langle\phi^a\rangle_\mathcal{P}\right)$$

Generalize to more moments! “**Moment Pooling**”

k = Highest order moment considered

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\underbrace{\langle\phi^a\rangle}_{\text{1st Moment}}{}_\mathcal{P}, \underbrace{\langle\phi^{a_1}\phi^{a_2}\rangle}_{\text{2nd Moment}}{}_\mathcal{P}, \ldots, \underbrace{\langle\phi^{a_1}\ldots\phi^{a_k}\rangle}_{k\text{th Moment}}{}_\mathcal{P}\right)$$

$$L_\text{eff} = \binom{L+k}{k}$$

Example: 2nd Moment $\langle\Phi^{a_1}\Phi^{a_2}\rangle_\mathcal{P} = \sum_{i\in\mathcal{P}} z_i\Phi^{a_1}(x_i)\Phi^{a_2}(x_i)$

Moment Pooling – Why?

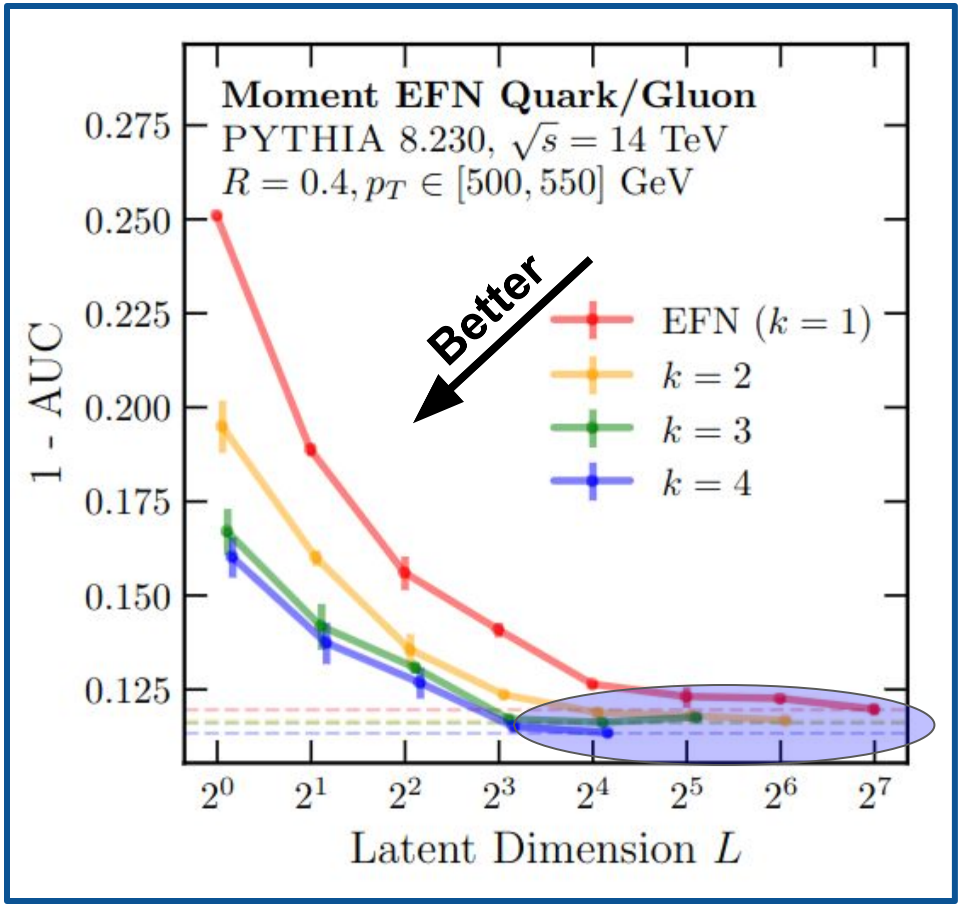
k = Highest order moment considered

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\underbrace{\langle\phi^{a}\rangle}_{1^{\text{st}}\text{ Moment}}{}_{\mathcal{P}}, \underbrace{\langle\phi^{a_1}\phi^{a_2}\rangle}_{2^{\text{nd}}\text{ Moment}}{}_{\mathcal{P}}, \dots, \underbrace{\langle\phi^{a_1}\dots\phi^{a_k}\rangle}_{k^{\text{th}}\text{ Moment}}{}_{\mathcal{P}}\right)$$

$$L_{\text{eff}} = \binom{L+k}{k}$$

- **Explicit Multiplication**: Neural nets are mostly piecewise linear! But most functions we learn aren’t. Moments are just multiplication, but for distributions!
- **Latent Space Compression**: For large *L*, there are effectively L^k latent dimensions due to combinatorics, but still made of only *L* functions! Fewer latent dimensions means easier analysis!

Moment Pooling Results

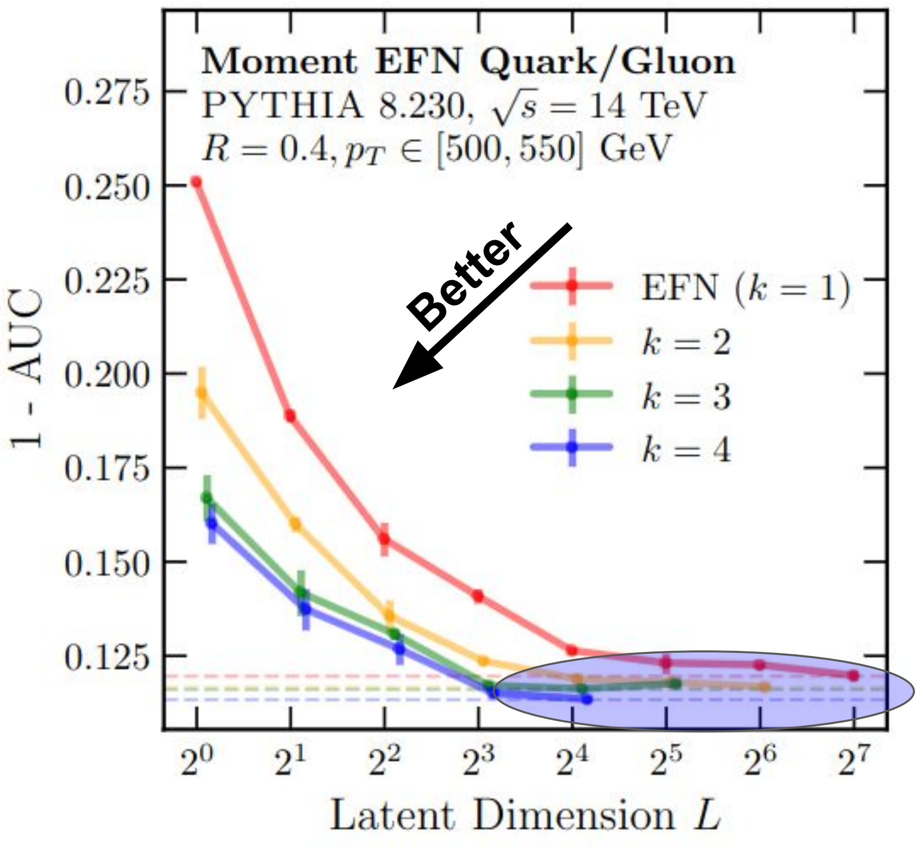


A high order **Moment EFN** can achieve the same top performance on quark/gluon classification as an **ordinary EFN**, but with far fewer latent dimensions!*

A Moment EFN is a more efficient encoding of the same data!

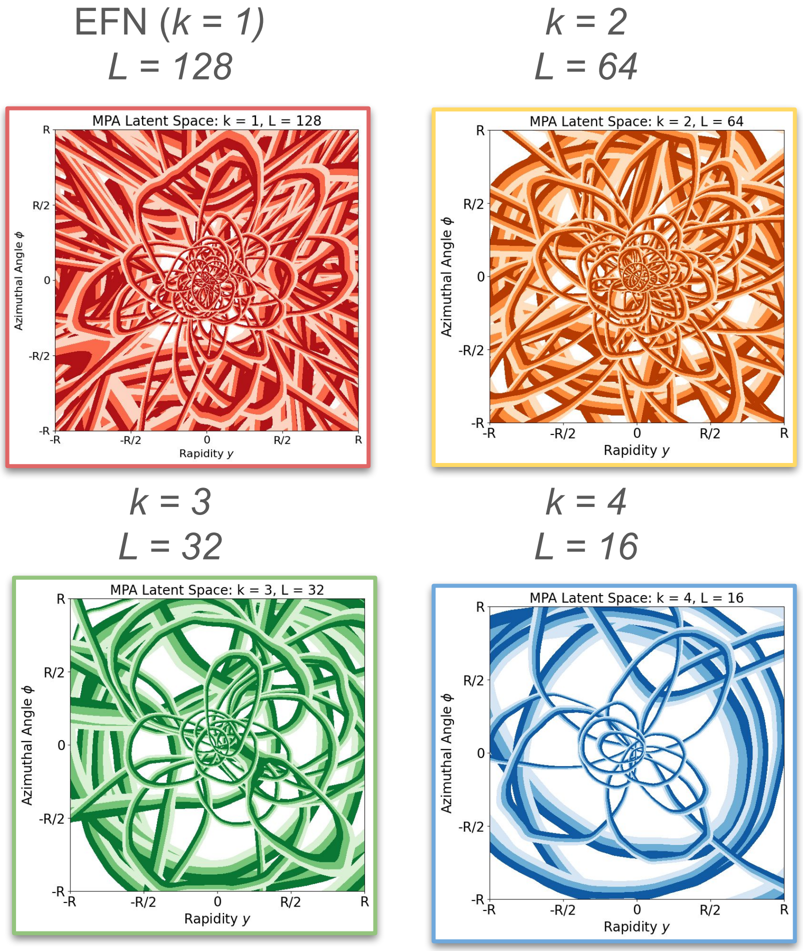
*Interestingly, the performance is constant in the *effective* latent dimensions and number of parameters. Ask me about this!

The Black Box



Better

Latent Spaces



EFN (k = 1)
L = 128

k = 2
L = 64

k = 3
L = 32

k = 4
L = 16

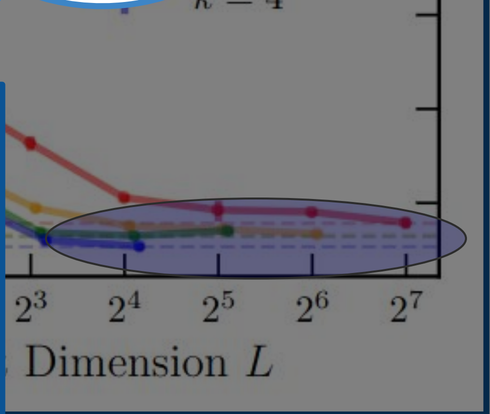
The 45% - 55% contours of each of the L different ϕ functions

*Interestingly, the performance is constant in the *effective* latent dimensions and number of parameters. Ask me about this!

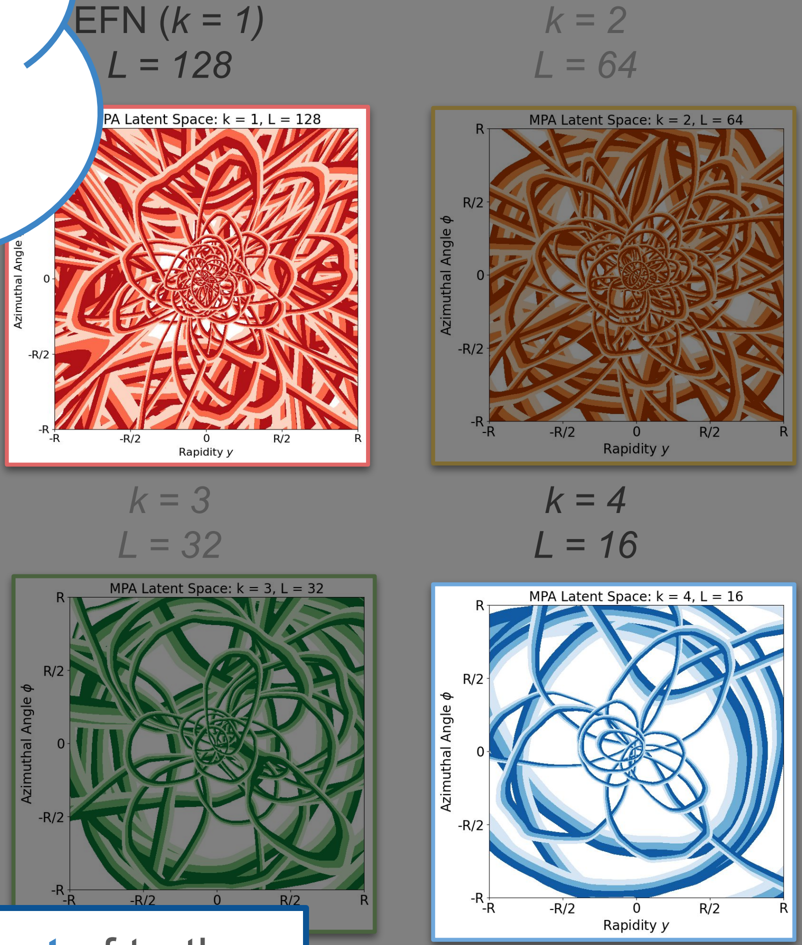
The Bl

So what? The information is just being shuffled around. Is 16 latent dimensions any more interpretable than 128?

The machine probably learned something, but I didn't!



Latent Spaces

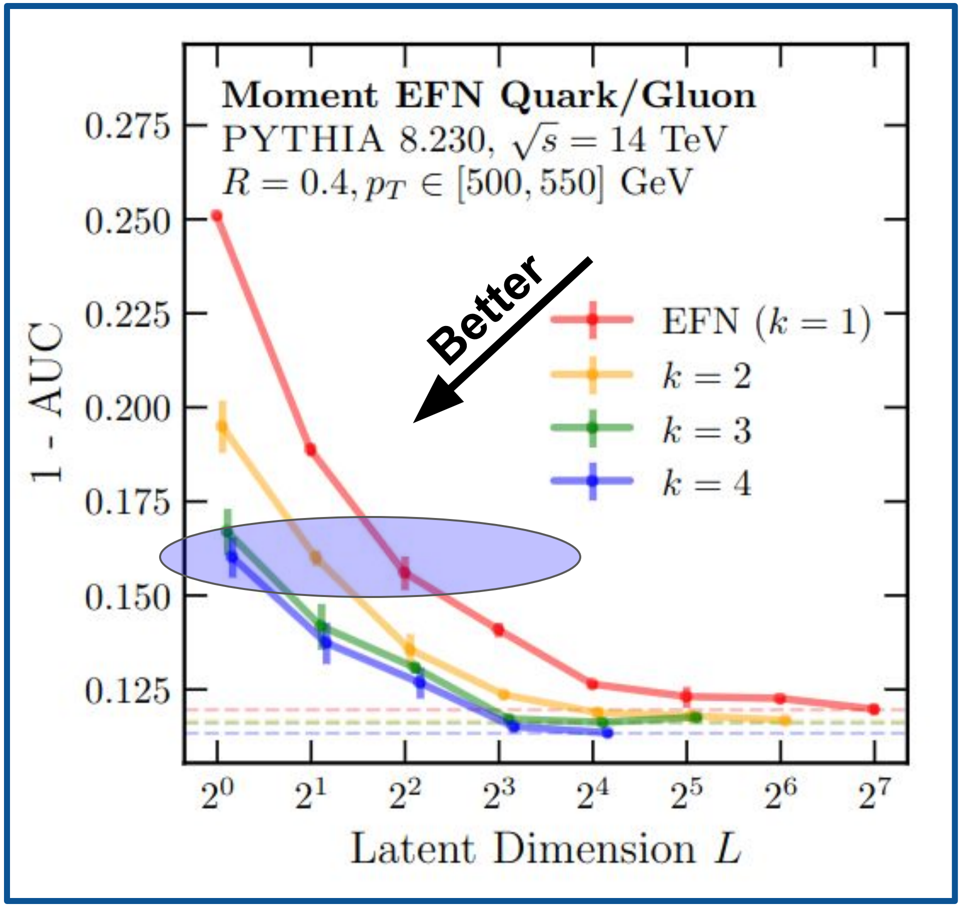


rs of each of the L different ϕ functions

Me, just before the **moment** of truth.

*Interestingly, the performance is constant in the *effective* latent dimensions and number of parameters. Ask me about this!

Moment Pooling Results (Cont'd)



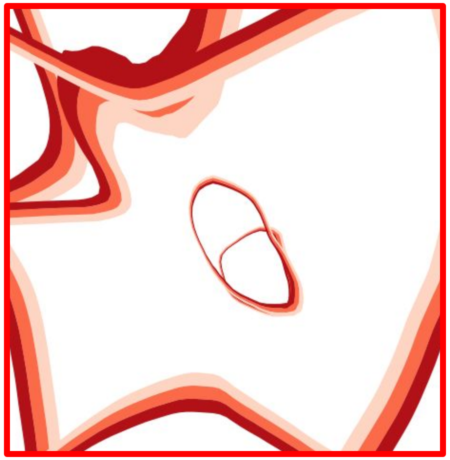
Let's instead look at networks with just a *single* latent dimension.

A ***k* = *4* Moment EFN** with *just one* latent dimension does just as good as an **ordinary EFN** with *four* latent dimensions!

Going down to *L = 1*

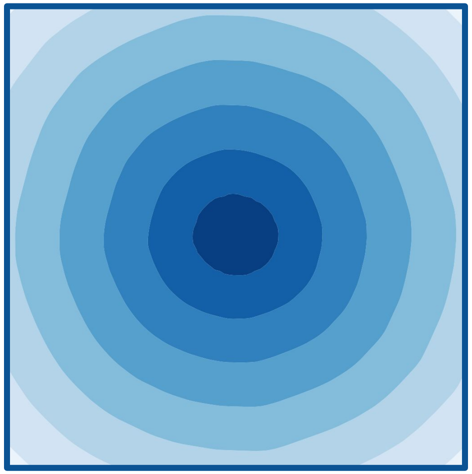
The more efficient encoding is extremely useful when we can get to *L = 1*!
Look how much simpler allowing neural networks to multiply makes things!

EFN (*k = 1*)
L = 4



Same information!

Order *k = 4*
L = 1



- 4 Different Functions
- Combined in a complicated way
- No symmetry
- ???

- Single function
- Expected to be, and is, radially symmetric
- We can interpret this!

The **Moment** of Truth

We can extract physics from this!

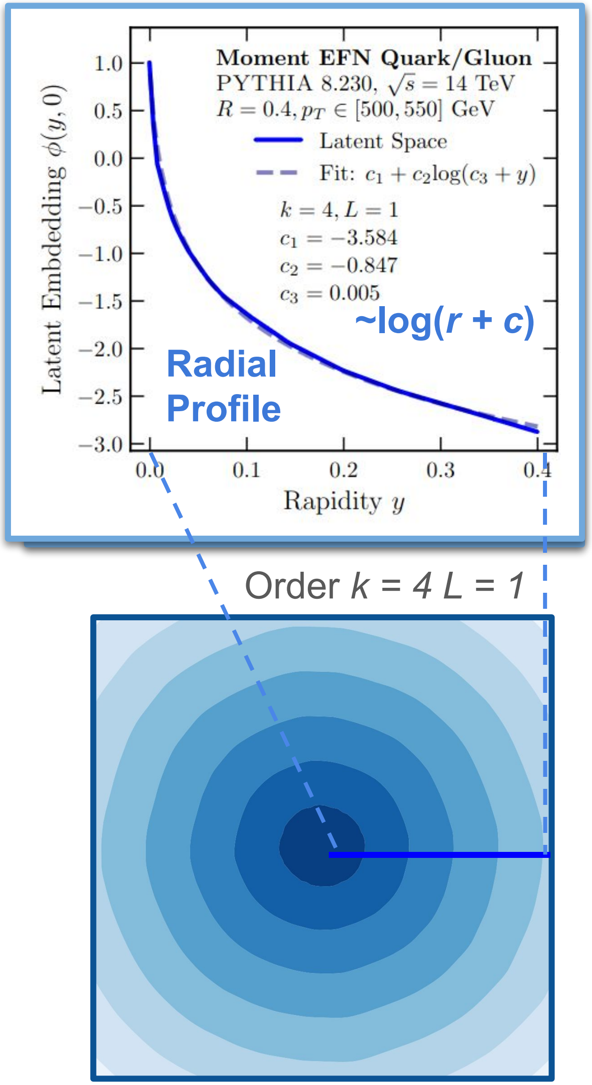
$$\Phi_{\mathcal{L}}(r) = c_1 + c_2 \log(c_3 + r)$$

This defines a **log angularity** observable, related to the □→0 limit of ordinary angularities

We can even see non perturbative physics!

$$c_3 \sim \frac{\Lambda_{\text{QCD}}}{p_T R} \sim 0.001$$

Just this alone is enough to get an IRC-safe quark-gluon classifier with an AUC ~ 0.83!



~log(*r* + *c*)

Radial Profile

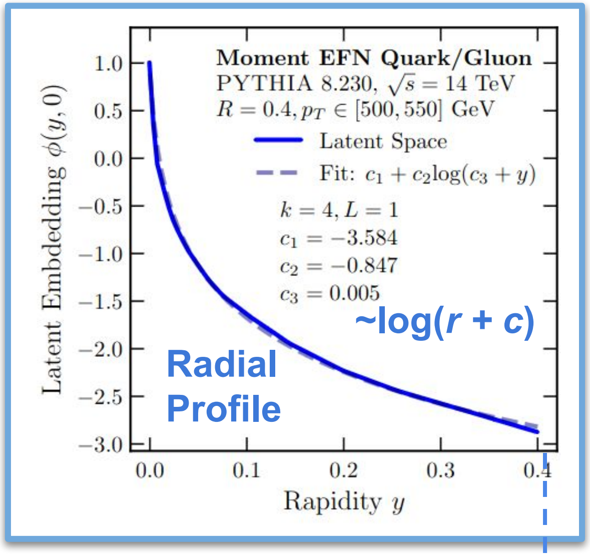
Order *k* = 4 *L* = 1

Part III Summary

By choosing the following new architecture for Deep Sets:

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\langle\phi^a\rangle_{\mathcal{P}}, \langle\phi^{a_1}\phi^{a_2}\rangle_{\mathcal{P}}, ..., \langle\phi^{a_1}...\phi^{a_k}\rangle_{\mathcal{P}}\right)$$

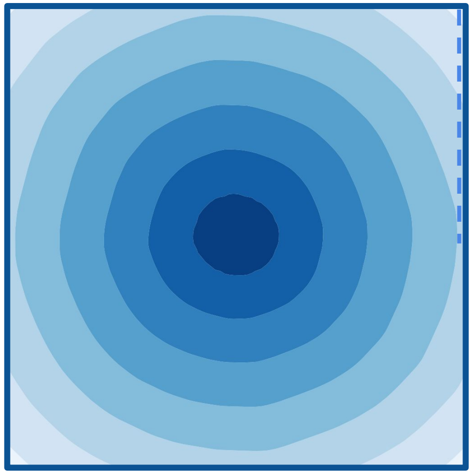
We expand the basis of primitive operations to include multiplication, allowing for **streamlined latent spaces** that are easier to extract physics information from!



~log(*r* + *c*)

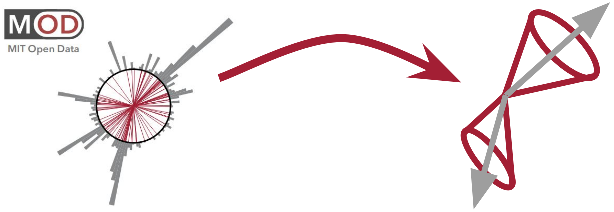
Radial Profile

Order *k* = *4* *L* = *1*



Part IV: Conclusion

Learning **Uncertainties** the **Frequentist** Way

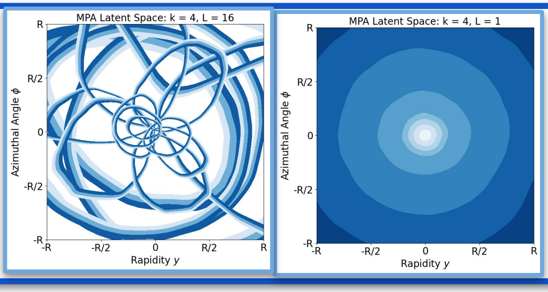


[**RG**, Nachman, Thaler, 2205.05084
[**RG**, Nachman, Thaler, 2205.05084]

Can you Hear the **Shape** of a Jet?

[Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]

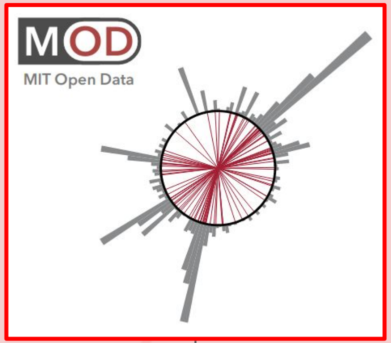
The **Moments** of Clarity: Streamlining Latent Spaces



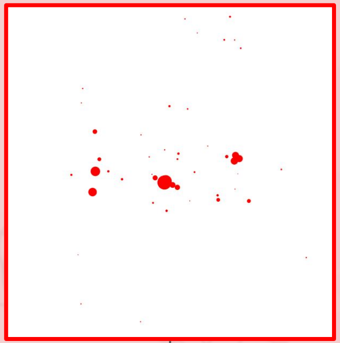
[**RG**, Osathapan, Thaler, 2403.08854]

Things a **machine** can understand

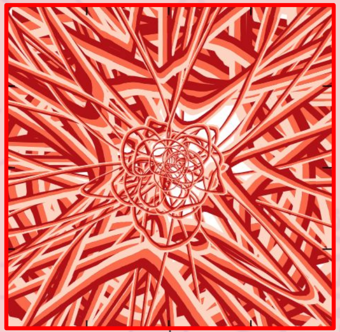
Sophisticated Detector Models



Complicated Point Clouds



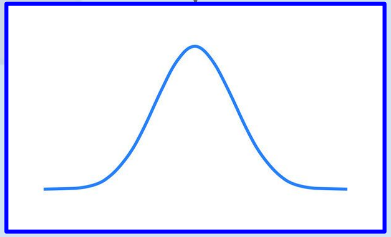
Huge Latent Spaces

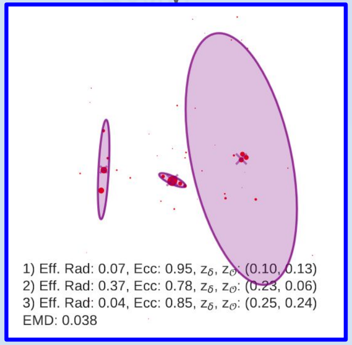


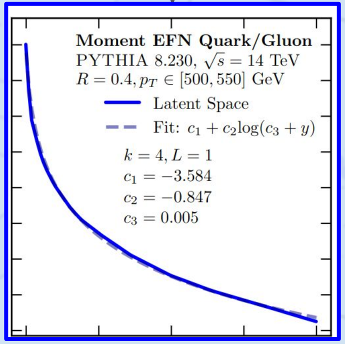
Using the Gaussian Ansatz and DV Loss

Using faithful optimal transport

Using Moment Pooling







Gaussians

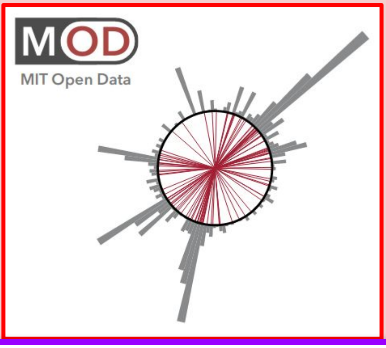
Basic Kindergarten Shapes

Addition, multiplication, and a few elementary functions

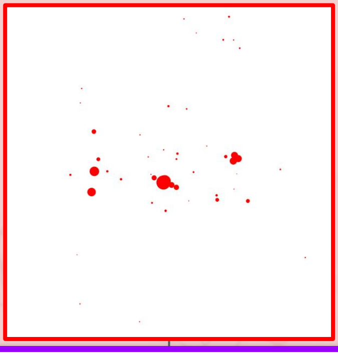
Things my **mortal physicist brain** can understand

Things a **machine** can understand

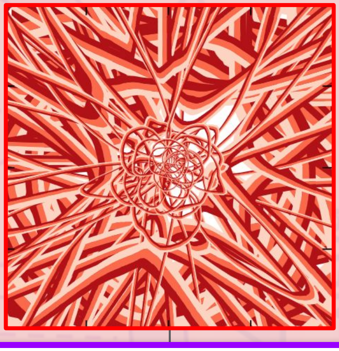
Sophisticated Detector Models



Complicated Point Clouds



Huge Latent Spaces

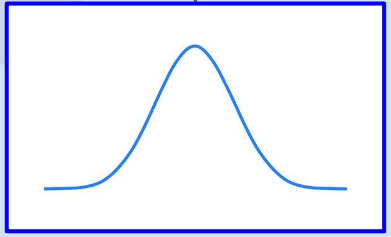


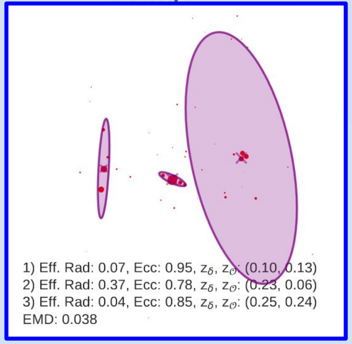
Main Takeaway: By using purpose-built machine learning architectures and losses, we can make sure we extract the **physics we want** from our machines!

Using the Gaussian Ansatz and DV Loss

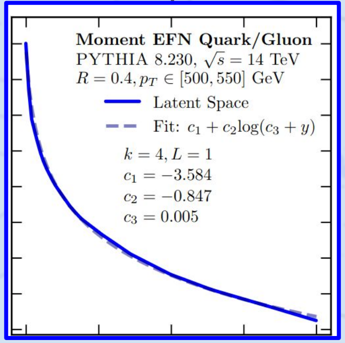
Using faithful optimal transport

Using Moment Pooling





1) Eff. Rad: 0.07, Ecc: 0.95, z_δ, z_φ: (0.10, 0.13)
2) Eff. Rad: 0.37, Ecc: 0.78, z_δ, z_φ: (0.23, 0.06)
3) Eff. Rad: 0.04, Ecc: 0.85, z_δ, z_φ: (0.25, 0.24)
EMD: 0.038



Gaussians

Basic Kindergarten Shapes

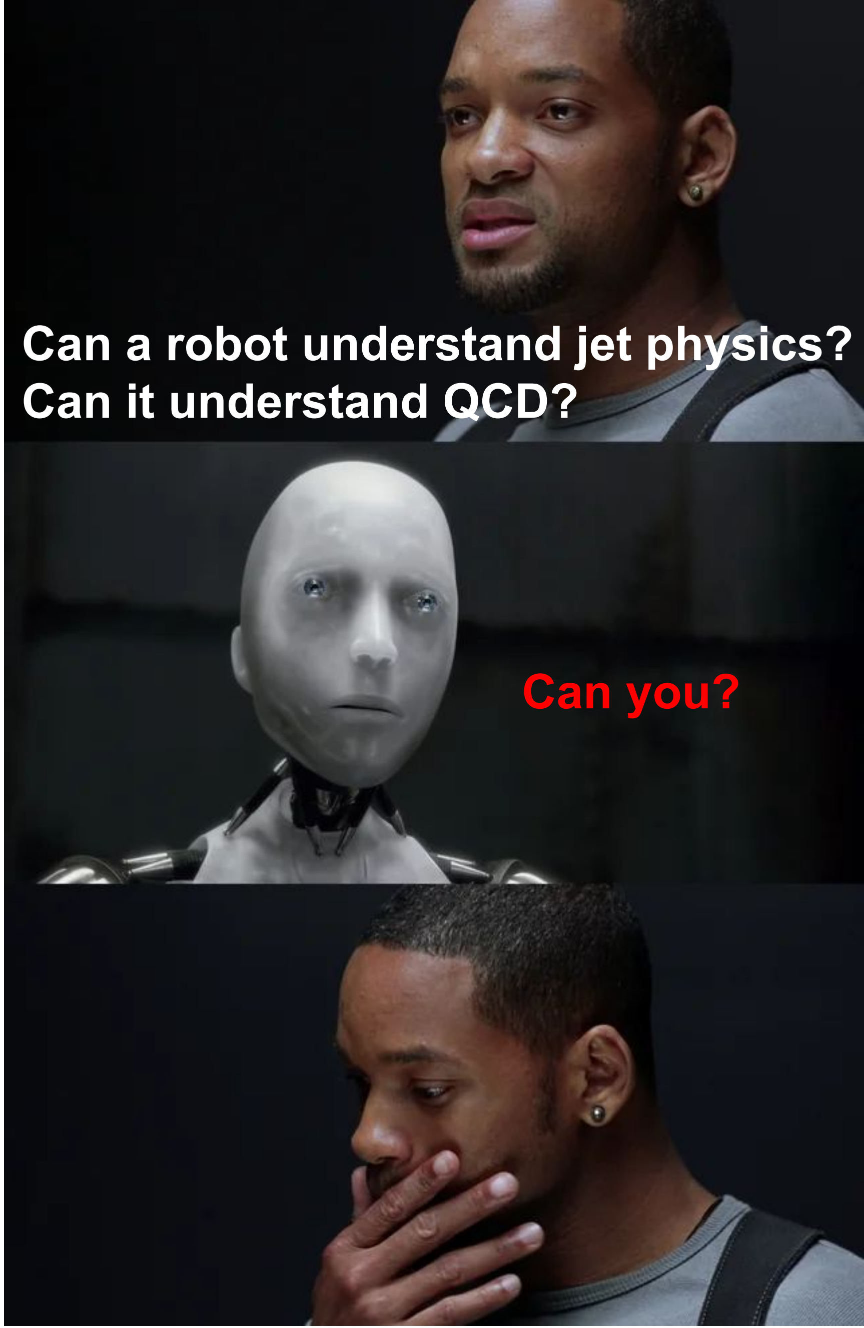
Addition, multiplication, and a few elementary functions

Things my **mortal physicist brain** can understand

Email me questions at rikab@mit.edu!

Based on [**RG**, Nachman, Thaler, 2205.03413]
[**RG**, Nachman, Thaler, 2205.05084]
[Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]
[**RG**, Thaler, Wu, WIP]
[**RG**, Osathapan, Tasissa, Thaler, 2403.08854]

Any Questions?



Backup

Learning MLC

How do we calculate *f*?

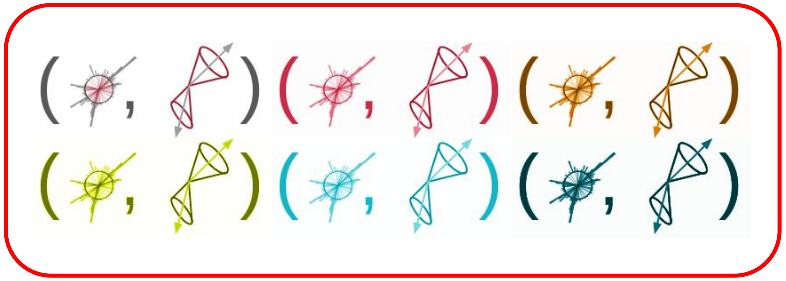
$$f_{\mathrm{MLC}}(x) = \operatorname*{argmax}_z p_{\mathrm{train}}(x|z)$$
$$= \operatorname*{argmax}_z \log \underbrace{\frac{p_{\mathrm{train}}(x,z)}{p_{\mathrm{train}}(x)p_{\mathrm{train}}(z)}}_{T(x,z)}$$

The function *T* is the likelihood ratio between *p(x,z)* and *p(x)p(z)*.

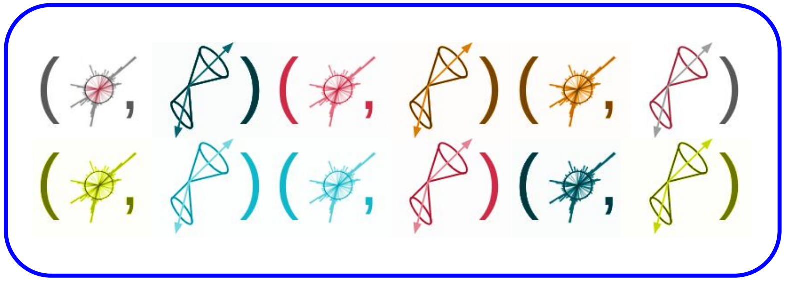
↓ Neyman–Pearson

T is the **optimal classifier** between (*x*,*z*) pairs and shuffled (*x*,*z*) pairs!

Class P



Class Q



Classify between ***P*** and ***Q***!

Aside: Mutual information

A measure for non-linear interdependence is the **Mutual Information**:

$$I(X;Z) = \int \mathrm{d}x\,\mathrm{d}z\,p(x,z)\log\frac{p(x,z)}{p(x)\,p(z)}$$
$$= \mathbb{E}_{\mathrm{train}}T(X,Z)$$

Answers the question: How much information, in terms of bits, do you learn about *Z* when you measure *X* (or vice versa)?

When doing calibration this way, we get a measure of the **correlation** between *X* and *Z*, *for free*.

Example 1: Gaussian Calibration Problem

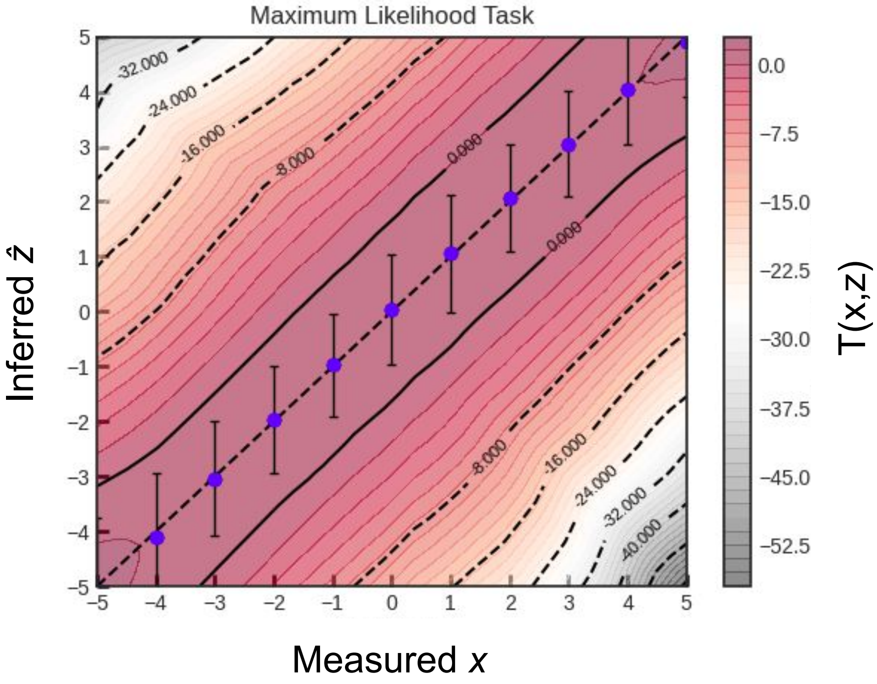
Gaussian noise model: p(x|z) ~ N(z, 1)

Model:

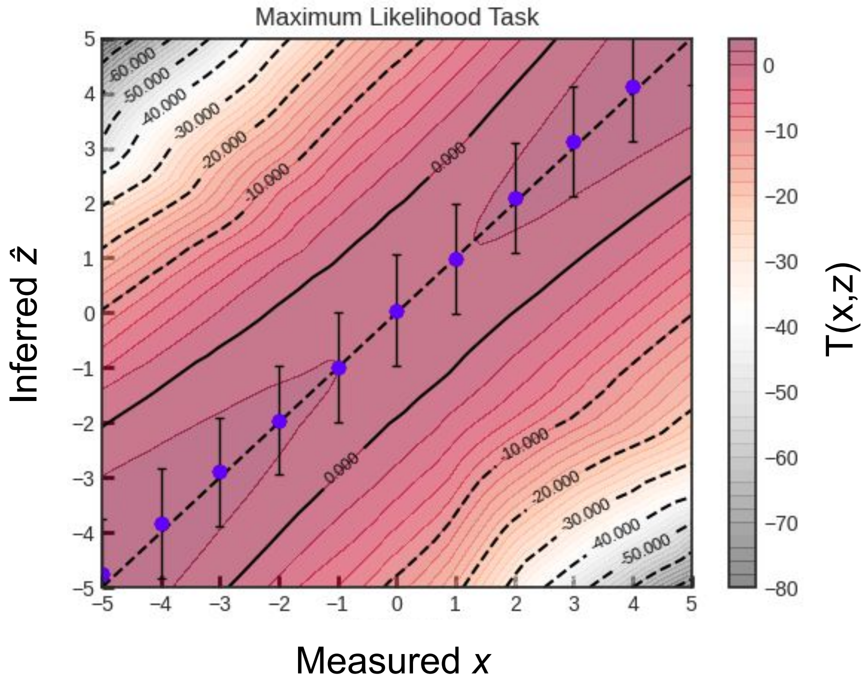
- The *A*, *B*, *C*, and *D* networks are each Dense networks with 4 layers of size 32
- ReLU activations
- All parameters have an L2 regularization (λ = 1e-6)
- The D network regularization slowly increased to (λ_D = 1e-4)

Learned mutual information of 1.05 natural bits

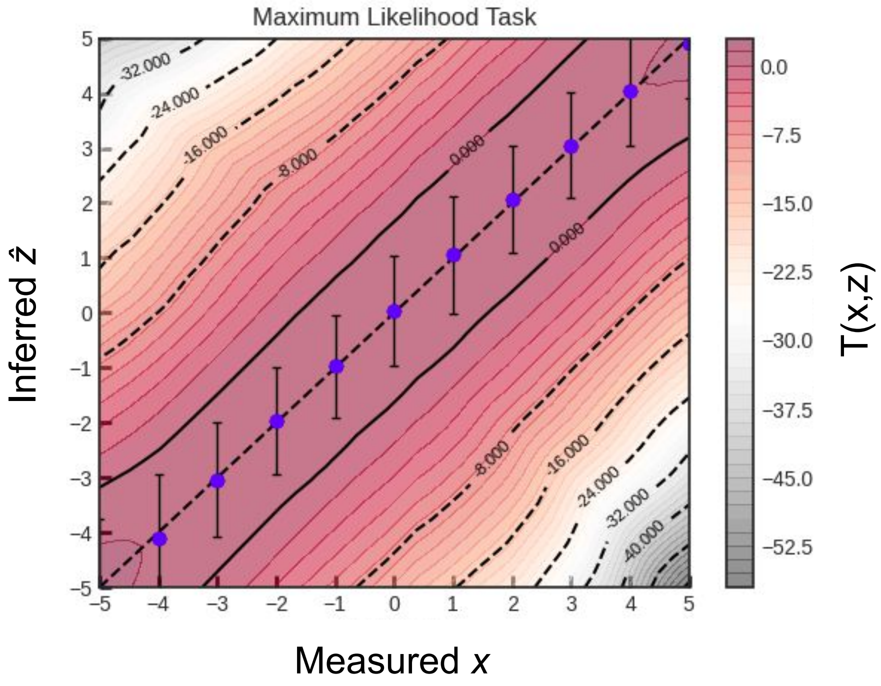
Reproduces the expected maximum likelihood outcome and the correct resolution!



Example 1 - Prior Independence



P(z) ~ N(0, 2.5)



P(z) ~ U(-5, 5)

[**RG**, Nachman, Thaler, PRL 129 (2022) 082001]

[CMS, 1607.03663]

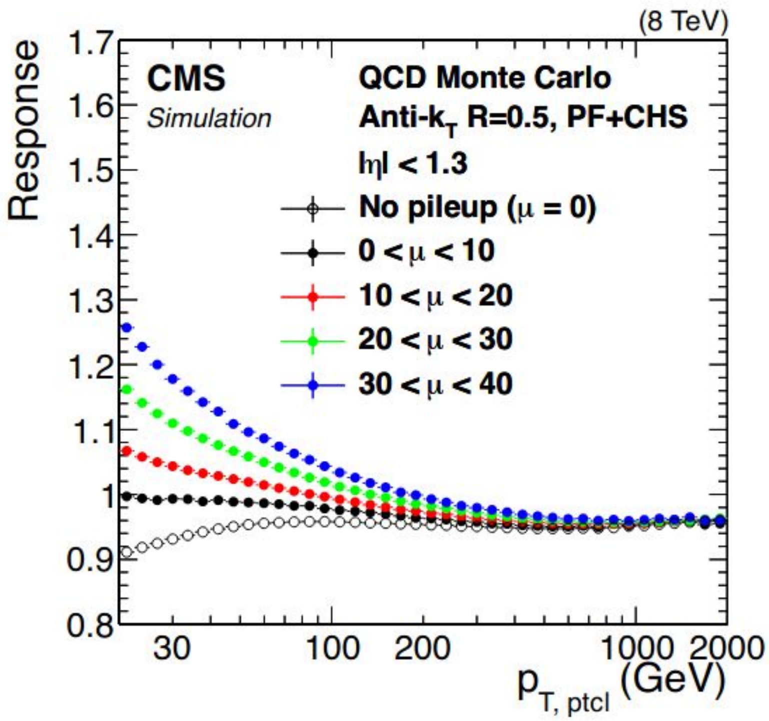
Example 3: Jet Energy Calibrations

Measure a set particle flow candidates *x* in the detector. What is the underlying jet p_T, *x*, and its uncertainty?

Define the **jet energy scale (JES)** and **jet energy resolution (JER)** as the ratio of the underlying (GEN) jet p_T (resolution) to the measured total (SIM) jet p_T

$$\hat{p}_T \equiv \mathrm{JEC} \times p_{T,\mathrm{SIM}} \approx p_{T,\mathrm{GEN}}$$

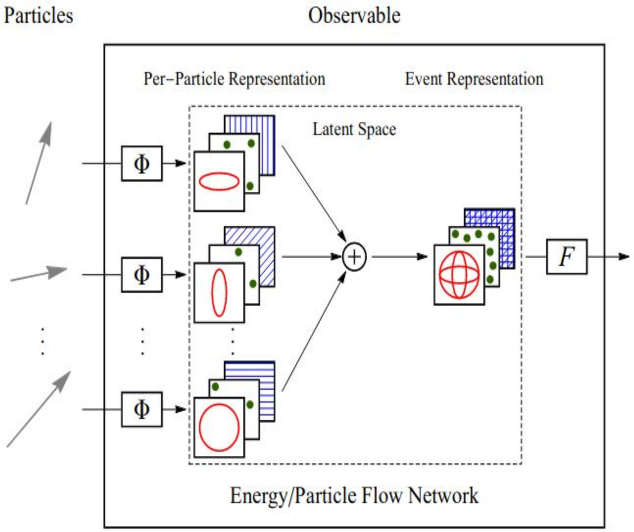
$$\hat{\sigma}_{p_T} = \mathrm{JER} \times p_{T,\mathrm{SIM}}$$



Example 3: Models

- **DNN**: X = (Jet p_T, Jet η, Jet φ), Dense Neural Network
- **EFN**: X = {(PFC p_T, PFC η, PFC φ)}, Energy Flow Network
- **PFN**: X = {(PFC p_T, PFC η, PFC φ)}, Particle Flow Network
- **PFN-PID**: X = {(PFC p_T, PFC η, PFC φ, PFC PID)}, Particle Flow Network

For each model, *A(x)*, *B(x)*, *C(x,z)*, and *D(x)* are all of the same type.



Permutation-invariant function of point clouds
For EFN's, manifest IRC Safety

Details on hyperparameters can be found in [**RG**, Nachman, Thaler, PRL 129 (2022) 082001]

[Komiske, Mastandrea, Metodiev, Naik, Thaler, PRD 2020;
Larkoski, Marzani, Thaler, Tripathee, Xue, 1704.05066;
Cacciari, Salam, Soyez, 0802.1189;
http://opendata.cern.ch/]

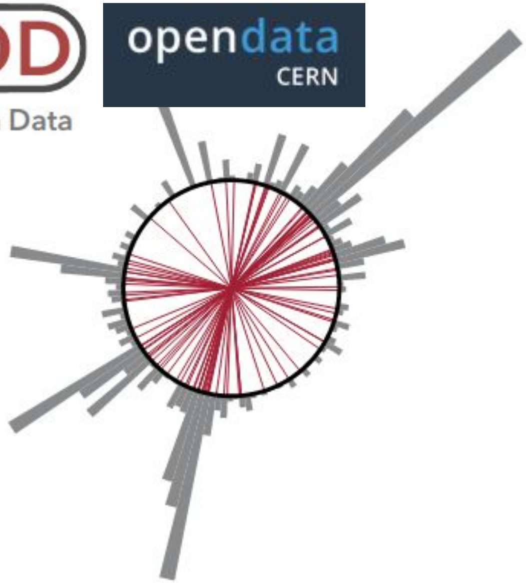
Example 3: Jet Dataset

Using CMS Open Data:

- *CMS2011AJets* Collection, SIM/GEN QCD Jets (AK 0.5)
- Select for jets with 500 GeV < Gen p_T < 1000 GeV, $|\eta| < 2.4$, quality ≥ 2
- Select for jets with ≤ 150 particles
- Jets are rotated such that jet axis is centered at (0,0)
- Train on 100k jets

MOD MIT Open Data

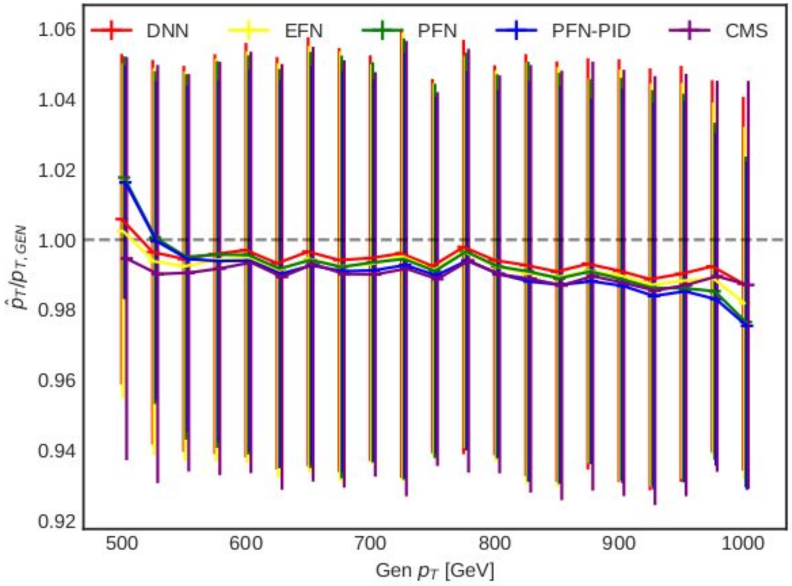
opendata CERN



Jet Energy Scales

For jets with a true p_T between 695-705 GeV, we should expect well-trained models to predict 700 GeV on average!

Model	Gaussian Fit [GeV]
DNN	695 ± 38.2
EFN	692 ± 37.7
PFN	702 ± 37.4
PFN-PID	693 ± 35.9
CMS Open Data	695 ± 37.4



Close to 1.00 – unbiased estimates!

[**RG**, Nachman, Thaler, PRD 106 (2022) 036011]

Data Based Calibration

“What if my detector simulation *p(x|z)* is imperfect”?

Given a *bad* simulator $p_{SIM}(x|z)$, we can correct it by matching it to data:

$$\hat{p}(x_D|z_T) = p_{\mathrm{sim}}(h(x_D)|z_T)|h'(x_D)|$$

Where

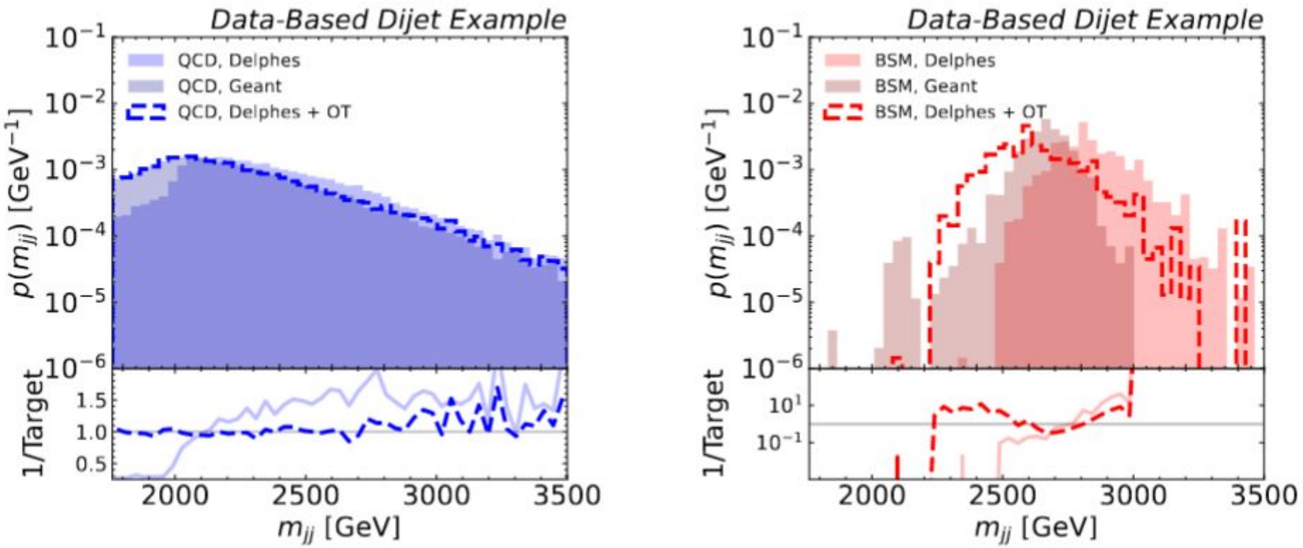
$$h(x_D) = P_{\mathrm{data}}^{-1}(P_{\mathrm{sim}}(x_D))$$

The function *h* “optimally transports” points to where they belong and reweights them.

[**RG**, Nachman, Thaler, PRD 106 (2022) 036011]

Data Based Calibration

BUT! There is a cost. We have to give up prior independence.



“Fixing” the Delphes simulation to match Geant4 works when trained on **Prior 1 (QCD)**, but fails miserably when applied to **Prior 2 (BSM)**, despite being the same detector simulation!

No (known) method of prior independent DBC, but no proof it is impossible!

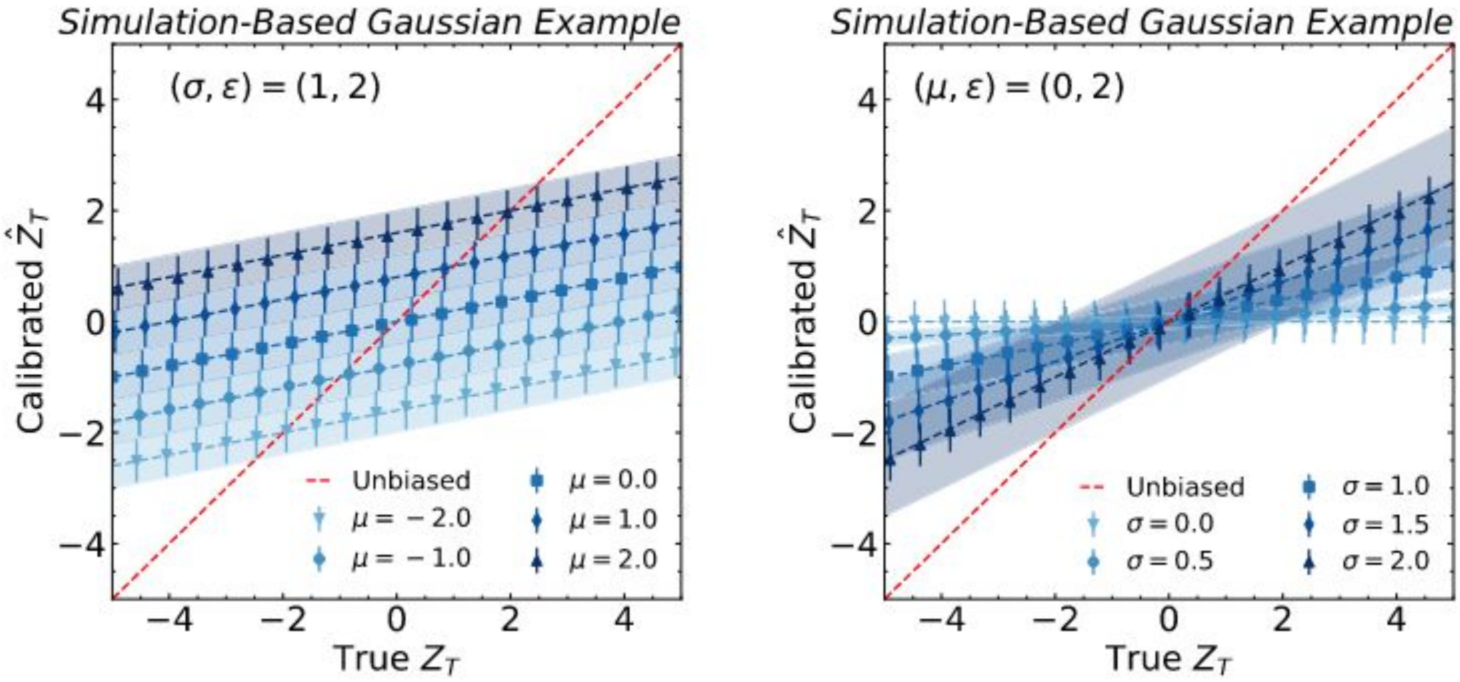
[**RG**, Nachman, Thaler, PRD 106 (2022) 036011]

Prior dependence of MSE

MSE fits for a gaussian noise model, for different choices of *z* prior.

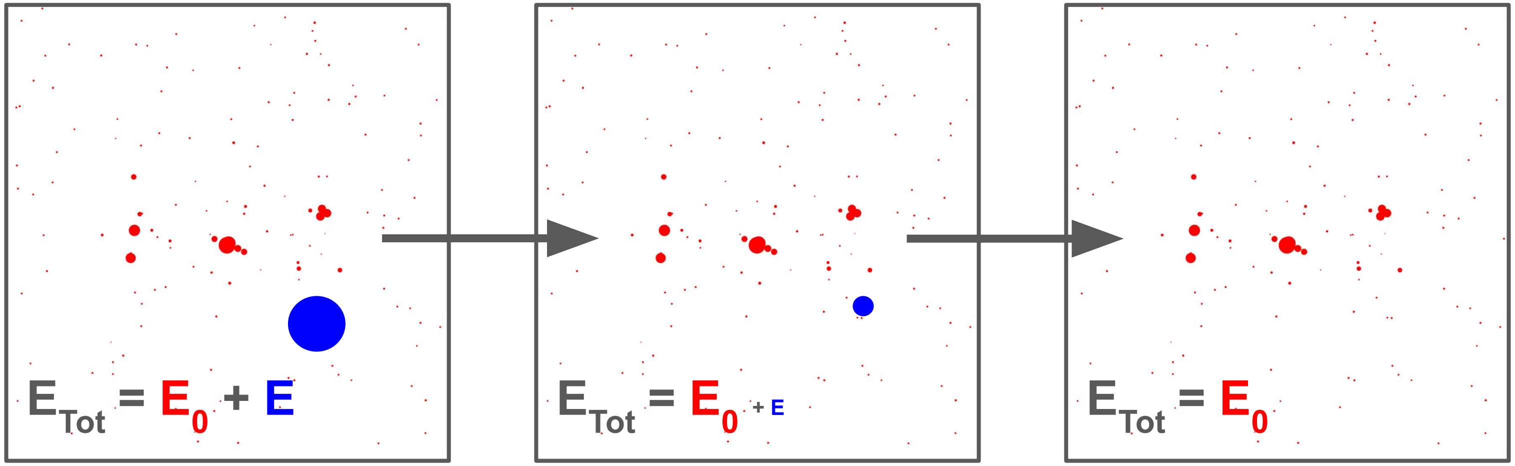
Left: Different choices of mean

Right: Different choices of width



Mathematical Details – Topology

Definition (The **Weak* Topology**): A sequence of measures converges if all of their expectation values converge, as real numbers.

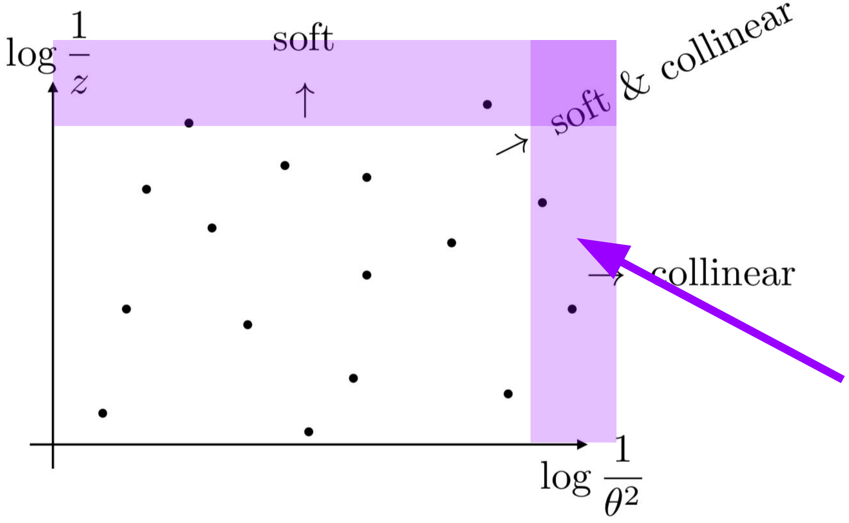


Definition (**Weak* Continuity**): An observable $\mathcal{O}(\mathcal{E})$ is continuous with respect to energy flows if, for any sequence of measures $\mathcal{E}_n$ that converges to $\mathcal{E}$, the sequence of real numbers $\mathcal{O}(\mathcal{E}_n)$ converges to $\mathcal{O}(\mathcal{E})$.

Topology ⇔ IRC-Safety

Two ways to change the expectation values of an energy flow:

1. Change a particle's energy slightly, or add a low-energy particle - **IR**
2. Move a particle's position slightly, or split particles in two - **C**



An observable $\mathcal{O}$ is continuous if it changes only slightly under the above perturbations.

The regions of phase space causing IRC divergences is suppressed — $\mathcal{O}$ is **IRC-Safe**!

Proved in [Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]
Ask me later for proof details!

Mathematical Details - Geometry

When are two events similar? We need a metric to compare!

Properties we want:

1. … is non-negative, non-degenerate, symmetric and finite
2. … is weak* continuous (IRC-safe)
3. … lifts the detector metric **faithfully**

Explained shortly!

The only* metric on distributions satisfying the above is the **Wasserstein Metric**:

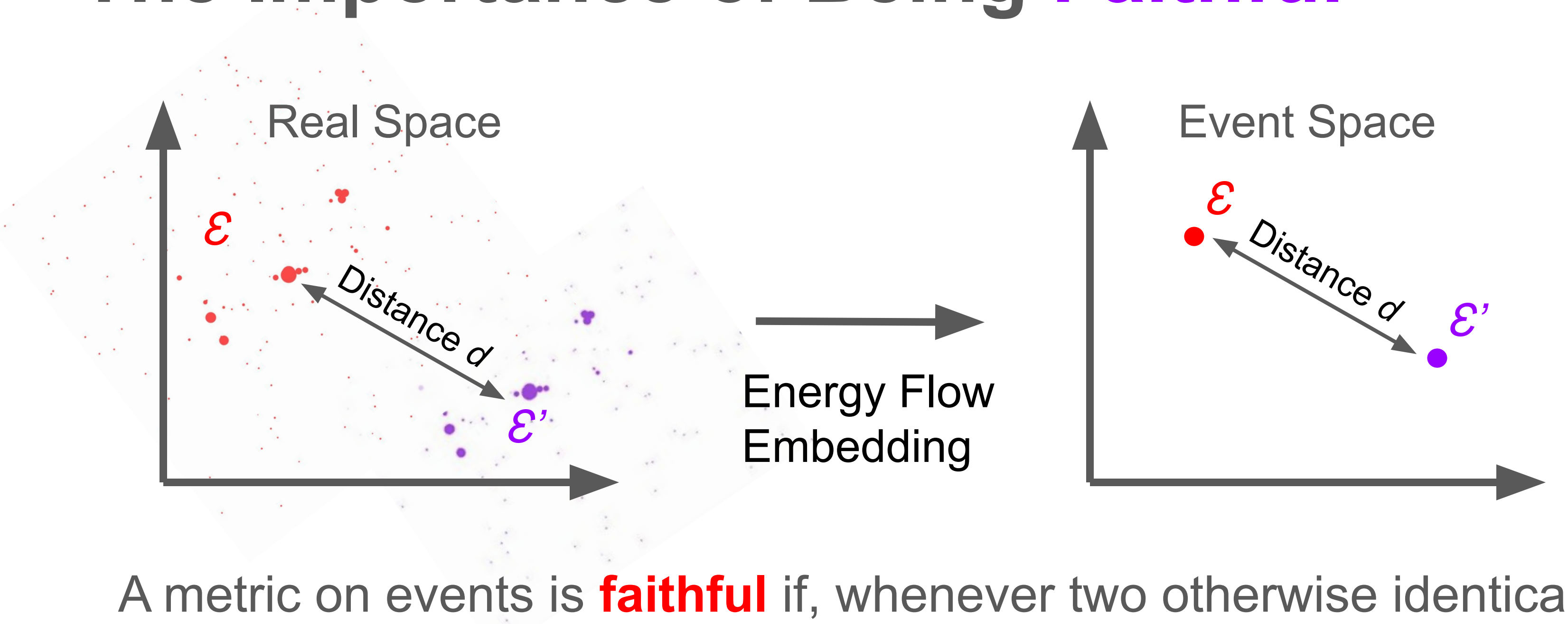
$$\text{EMD}^{(\beta,R)}(\mathcal{E},\mathcal{E}') = \min_{\pi\in\mathcal{M}(\mathcal{X}\times\mathcal{X})}\left[\frac{1}{\beta R^\beta}\left\langle \pi, d(x,y)^\beta\right\rangle\right] + |\Delta E_{\text{tot}}|$$

$$\pi(\mathcal{X},Y)\leq\mathcal{E}'(Y),\quad \pi(X,\mathcal{X})\leq\mathcal{E}(X),\quad \pi(\mathcal{X},\mathcal{X}) = \min(E_{\text{tot}},E'_{\text{tot}})$$

*There exist other metrics on distributions that are faithful only for very specific real-space distance norms, but we want them all!

Proved in [Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]
*Thanks to Cari Cesarotti for helping come up with this name.

The Importance of Being **Faithful***

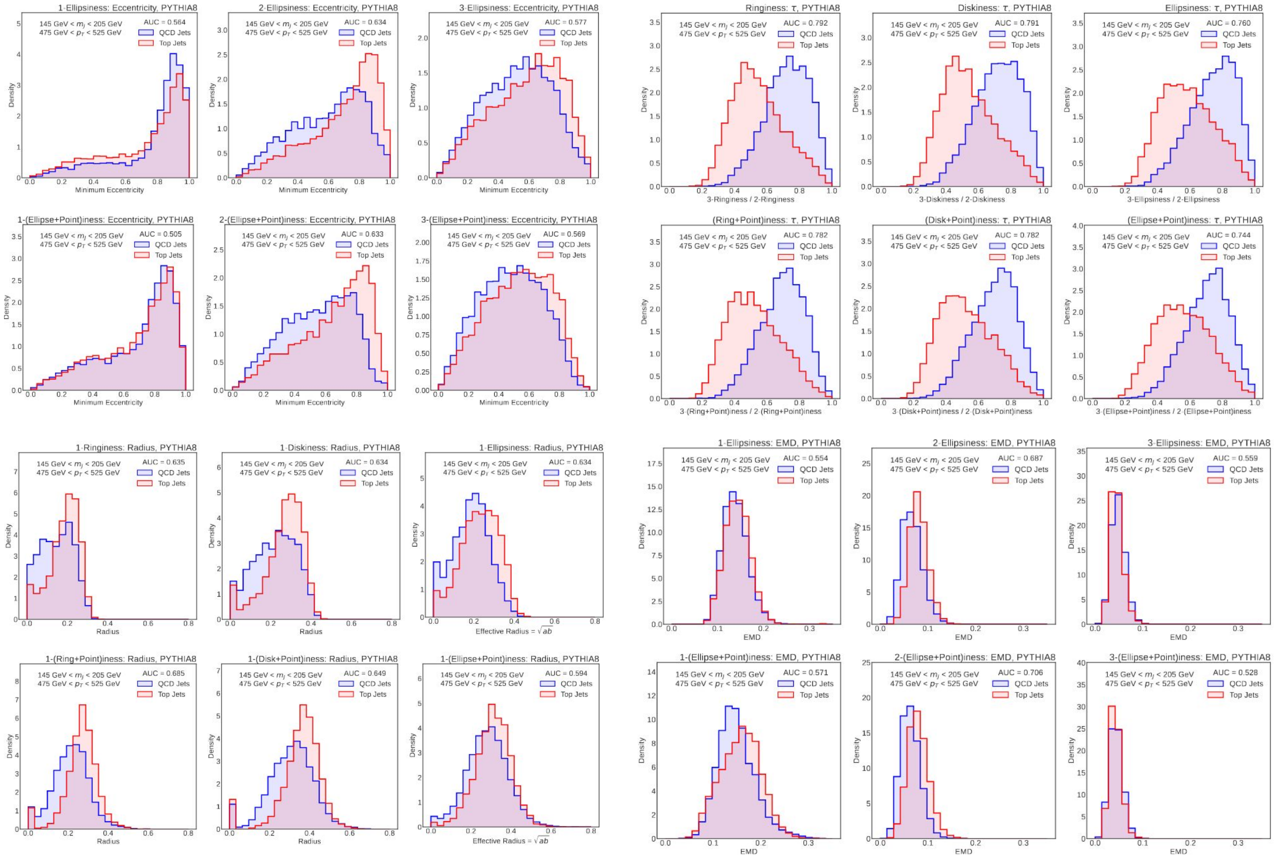


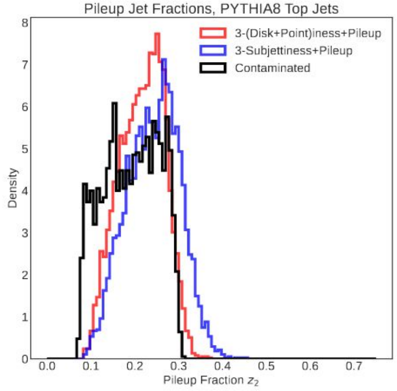
A metric on events is **faithful** if, whenever two otherwise identical events $\mathcal{E}$ and $\mathcal{E}'$ are separated in real space by a distance *d*, the distance between the events is also *d*. Or any predetermined invertible function of *d*

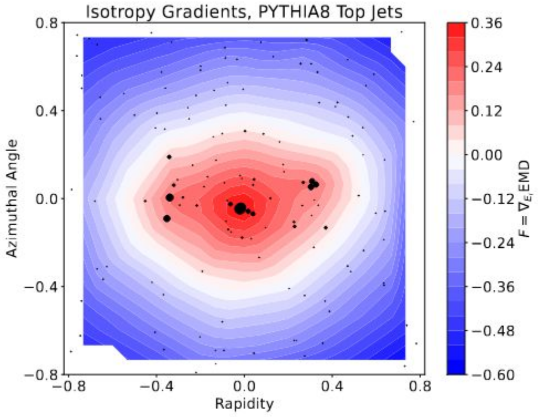
Only the **Wasserstein Metric** does this! → Can use to build **event** and **jet shapes** (old and new)!

Faithfulness also ensures very nice **numerical properties**, including no vanishing or exploding gradients.

New IRC-Safe Observables







… **Lots** of extractable information!

Automatic Grooming with Shapes

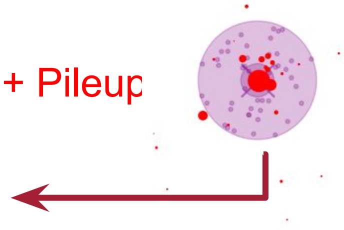
Use shapes to approximate events and extract m background with floating weight!

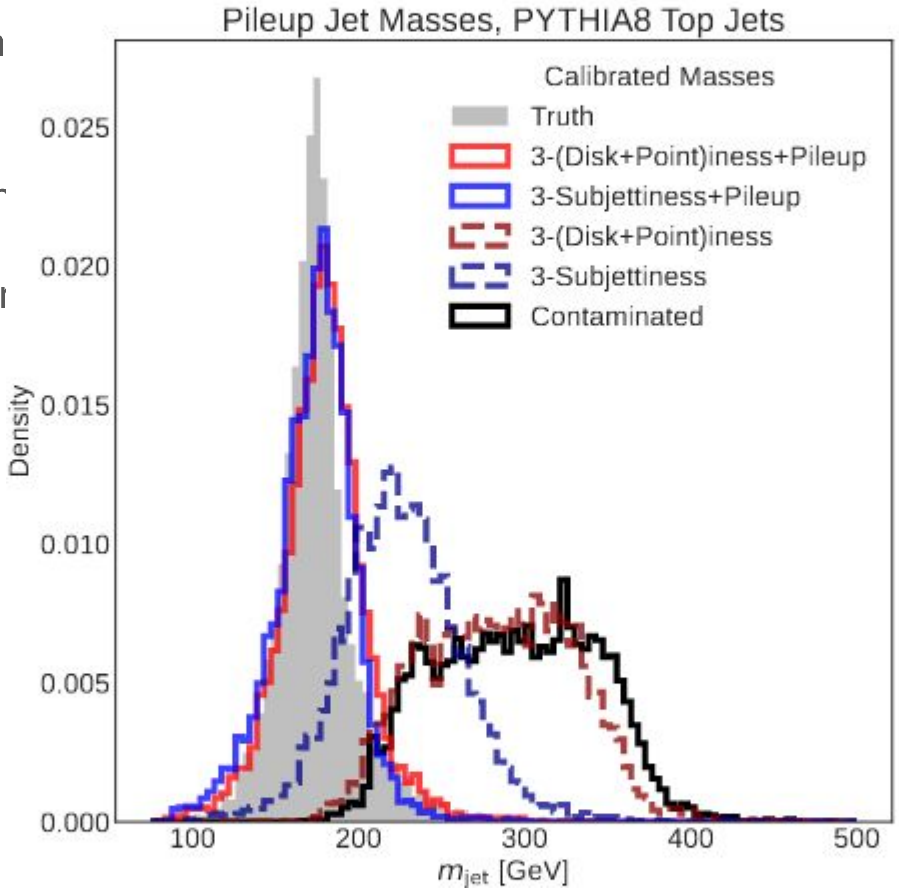
No external hyperparameters, unlike softdrop. Or

Contaminate top jets with 5-30% extra energy spr

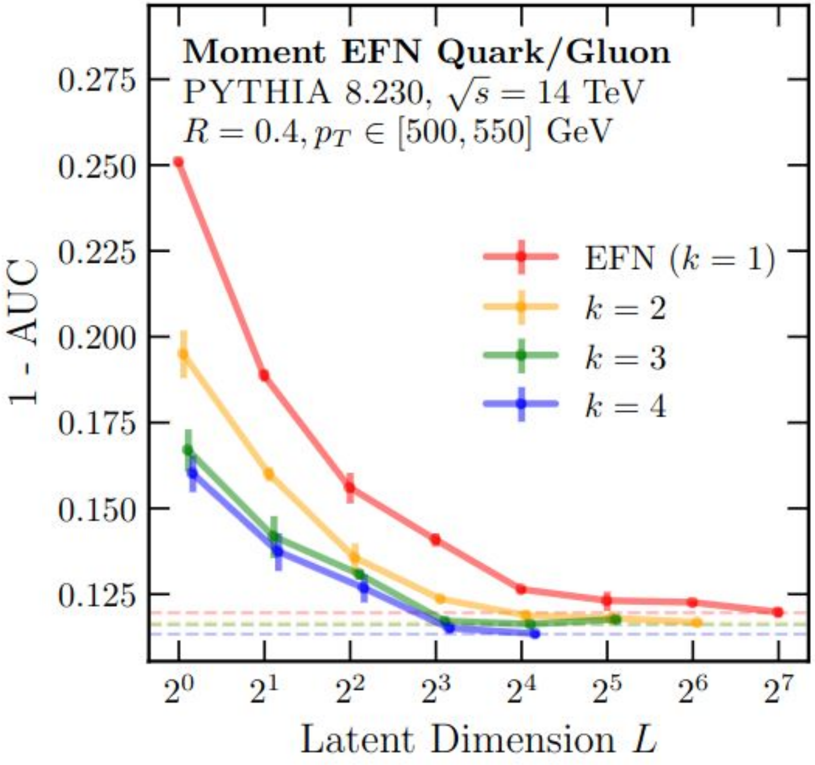
Consider 4 shapes:

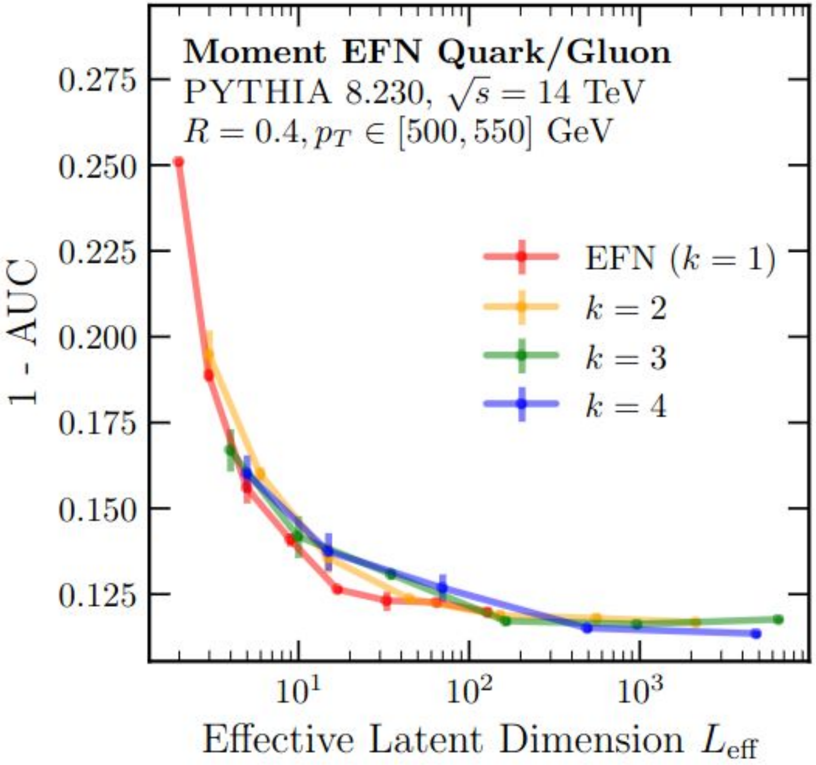
- 3-Subjettiness
- 3-Subjettiness + Pileup
- 3-(Disk+Point)iness
- 3-(Disk+Point)iness + Pileup

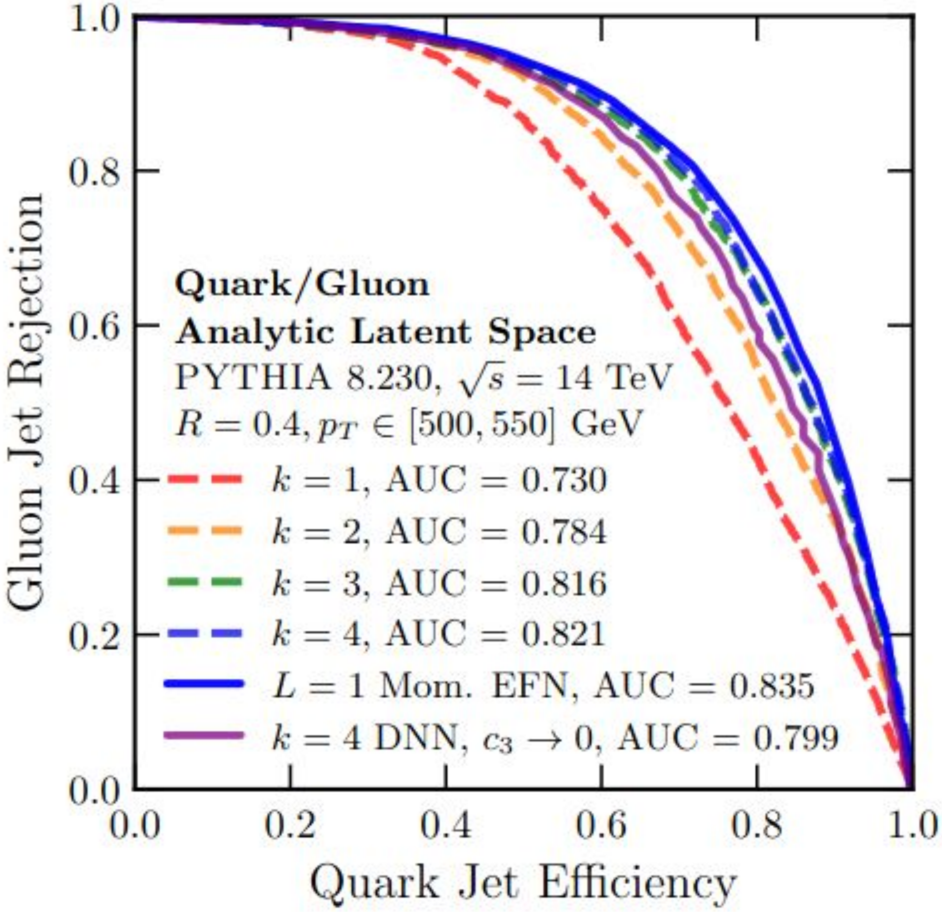




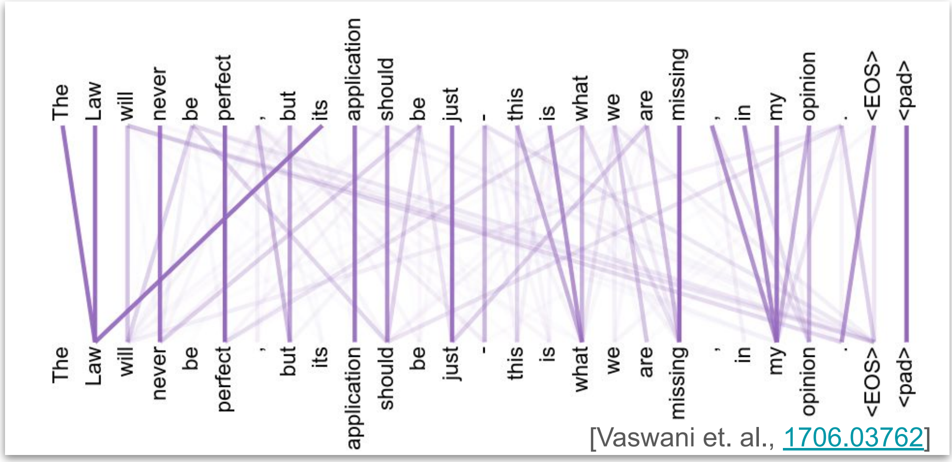
Can also consider ellipses instead of disks – only marginally better performance


Moment EFN Quark/Gluon
PYTHIA 8.230, $\sqrt{s} = 14$ TeV
$R = 0.4, p_T \in [500, 550]$ GeV
1 - AUC
0.275
0.250
0.225
0.200
0.175
0.150
0.125
EFN ($k = 1$)
$k = 2$
$k = 3$
$k = 4$
2^0
2^1
2^2
2^3
2^4
2^5
2^6
2^7
Latent Dimension L


Moment EFN Quark/Gluon
PYTHIA 8.230, $\sqrt{s} = 14$ TeV
$R = 0.4, p_T \in [500, 550]$ GeV
1 - AUC
0.275
0.250
0.225
0.200
0.175
0.150
0.125
EFN ($k = 1$)
$k = 2$
$k = 3$
$k = 4$
10^1
10^2
10^3
Effective Latent Dimension L_{eff}


Quark/Gluon
Analytic Latent Space
PYTHIA 8.230, $\sqrt{s} = 14$ TeV
$R = 0.4, p_T \in [500, 550]$ GeV
$k = 1$, AUC = 0.730
$k = 2$, AUC = 0.784
$k = 3$, AUC = 0.816
$k = 4$, AUC = 0.821
$L = 1$ Mom. EFN, AUC = 0.835
$k = 4$ DNN, $c_3 \to 0$, AUC = 0.799
Gluon Jet Rejection
Quark Jet Efficiency
0.0
0.2
0.4
0.6
0.8
1.0

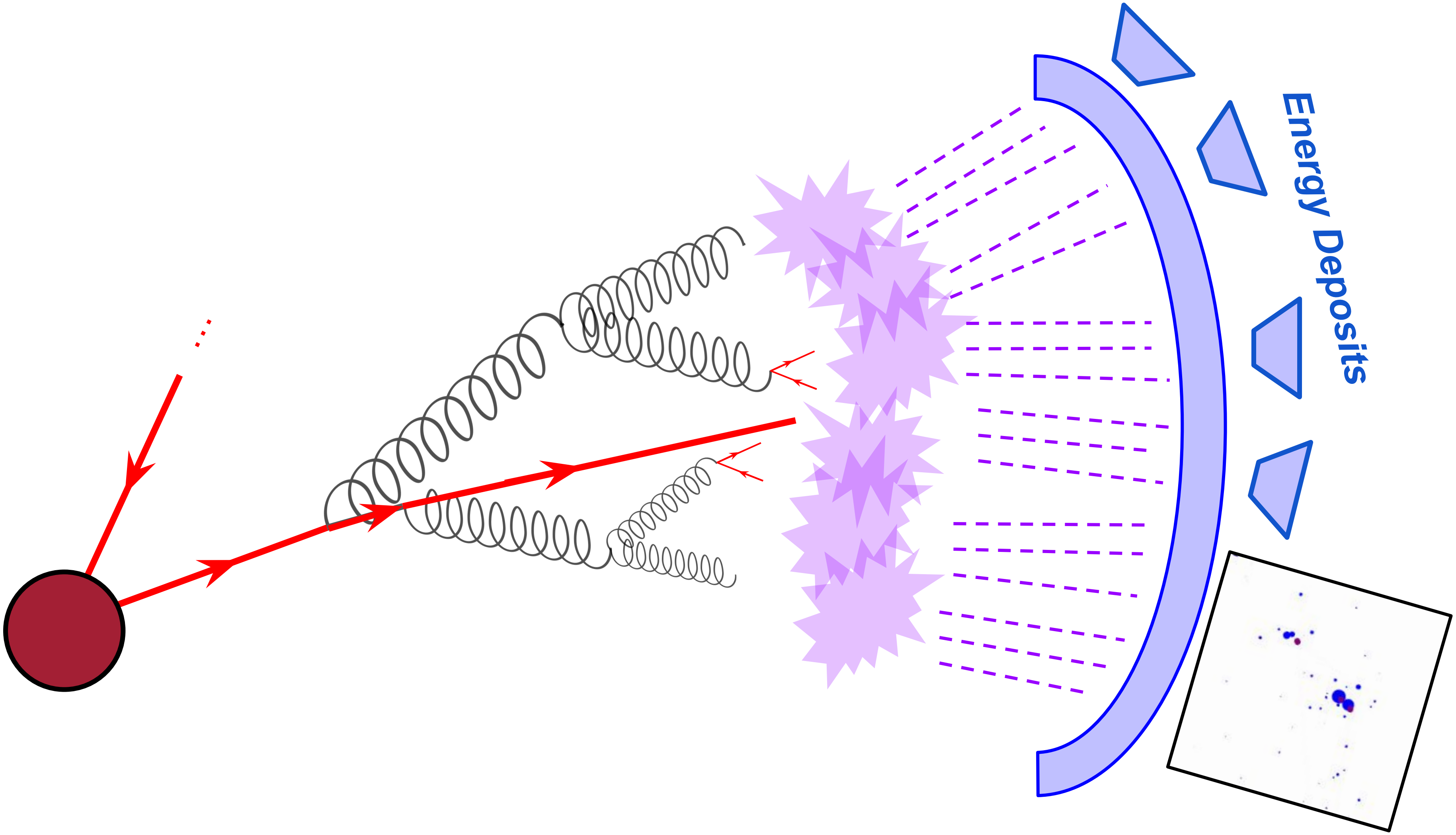
Attention is all you need


The Law will never be perfect , but its application should be just - this is what we are missing , in my opinion . <EOS> <pad>
The Law will never be perfect , but its application should be just - this is what we are missing , in my opinion . <EOS> <pad>

[Vaswani et. al., 1706.03762]

$$\sim \quad \langle \phi^{a_1} \phi^{a_2} \rangle_{\mathcal{P}}$$

Can view moment pooling as a multi-headed self-attention-like mechanism
Each latent variable weights each other latent variable


Energy Deposits