Can You Hear the Shape of a Jet?

An IAIFI Story

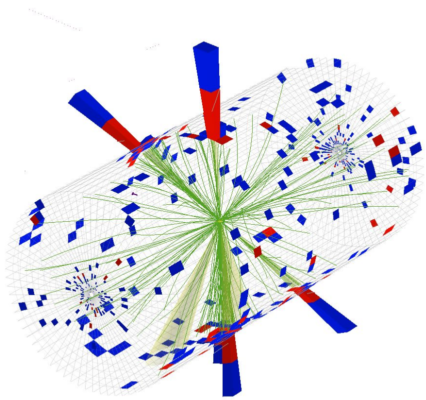
Rikab Gambhir

Email me questions at rikab@mit.edu!

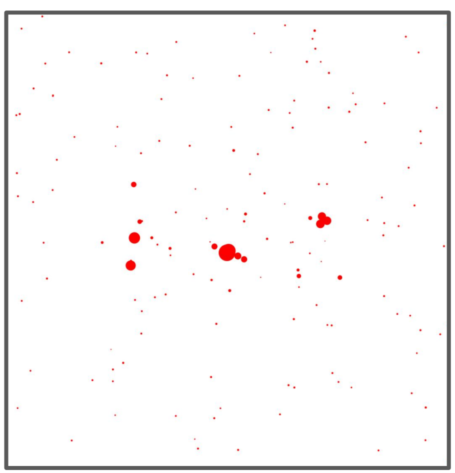
Based on [Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]

Download with `pip install pyshaper`

How do we **characterize** collider data?



Pictured: Several jets in the CMS Detector



A jet, as an energy-weighted 2D point cloud

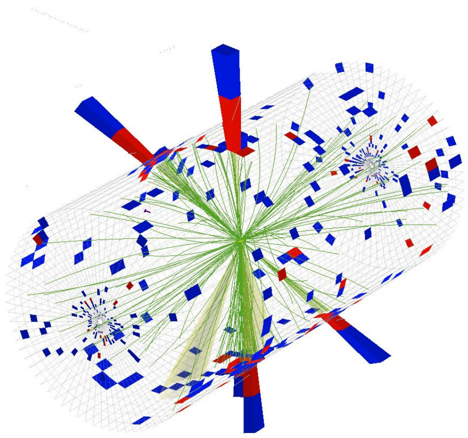
We have **complicated** event data, called **jets**, from collider experiments.

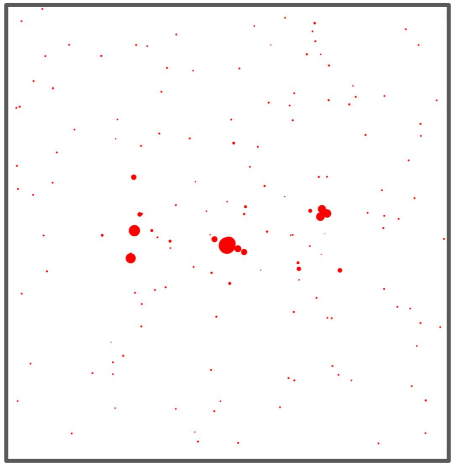
This data is extremely highly dimensional, hard to understand, and very noisy (both experimentally and theoretically).

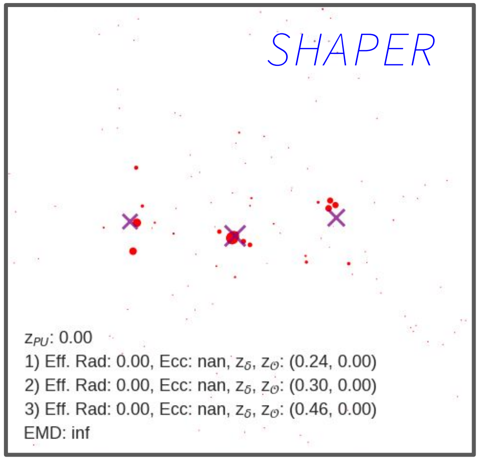
Can we extract the important and salient features of our jets with ML-inspired techniques?

In other words, **can you “hear” the shape of a jet?**

How do we characterize collider data?







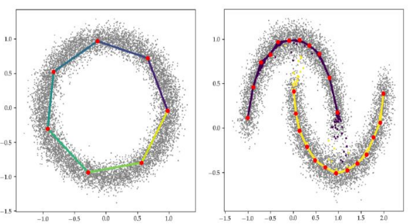
Collider Data → **Energy Flow** → **Shapes**

Main Lessons:

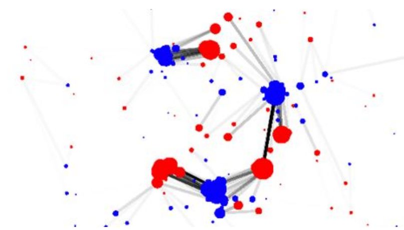
1. Matching jets to idealized jet shapes ⟺ Manifold learning from the ML community.
2. Use a comparison metric that preserves both geometry and physics symmetries.

Yes, you *CAN* hear the shape of a jet!

NSF AI Institute for Artificial Intelligence & Fundamental Interactions

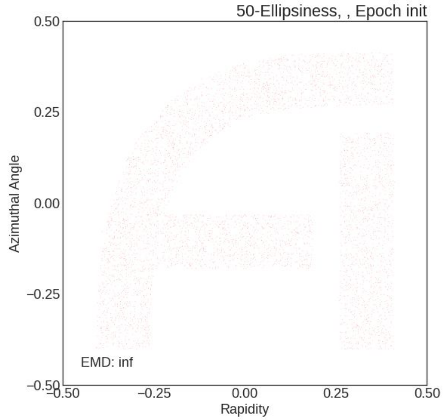


Piecewise-Linear Manifold Approximation with K-Deep Simplices (KDS, 2012.02134)



Well-Defined Metric on Particle Collisions using Energy Mover's Distance (EMD, 2004.04159)

SHAPER: Learning the Shape of Collider Events



$$\mathcal{O}_{\mathcal{M}}(\mathcal{E}) = \min_{\mathcal{E}'_\theta \in \mathcal{M}} \mathrm{EMD}(\mathcal{E}, \mathcal{E}'_\theta)$$

$$\theta = \operatorname*{argmin}_{\mathcal{E}'_\theta \in \mathcal{M}} \mathrm{EMD}(\mathcal{E}, \mathcal{E}'_\theta)$$

Framework for defining and calculating useful observables for collider physics!

Manifold Learning for statistical distributions ⇔ jet physics

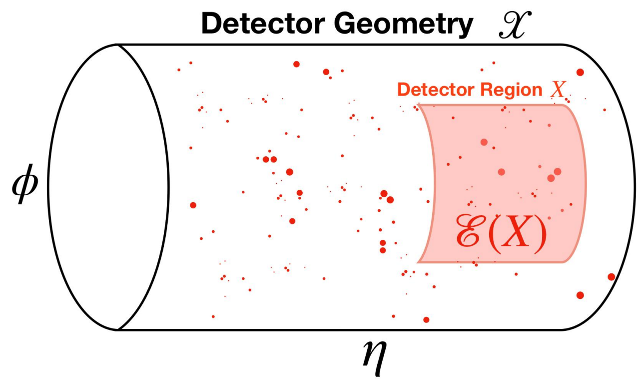
The Energy Flow

We can embed jet data into the **energy flow**:

Detector Coordinate (η, ϕ)

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

Energy Fraction E_i / E_{Tot}



The **energy flow** contains *ALL* IRC-safe (“robust”) information about a jet!

Just a 2D probability distribution →**energy flows** let us translate our physics problem into a pure **ML/statistics** problem!

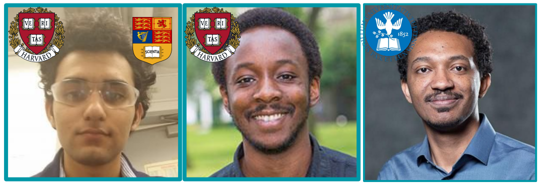
When are two jets similar?

Building off of **K-Deep Simplices**, and adding **physics-inspired structure**, we show that the **Wasserstein Metric (EMD)** is *the* natural structure for probing the geometric structure of statistical distributions

$$\mathcal{L}_R(\mathcal{E},\mathcal{E}') = \min_{\pi_{ij}\geqslant 0}\left[\sum_{i=1}^{M}\sum_{j=1}^{M'}\pi_{ij}\frac{|x_i - x'_j|}{R}\right] + \left|\sum_{i=1}^{M} z_i - \sum_{j=1}^{M'} z'_j\right|,$$

$$\text{where} \sum_{i=1}^{M}\pi_{ij} \leqslant z'_j, \quad \sum_{j=1}^{M'}\pi_{ij} \leqslant z_i, \quad \sum_{i,j}^{M,M'}\pi_{ij} = \min\left(\sum_{i=1}^{M} z_i, \sum_{j=1}^{M'} z'_j\right)$$

Dogra Ba Tasissa



K-Deep Simplices, Dictionary Learning, & Manifold Learning

\+

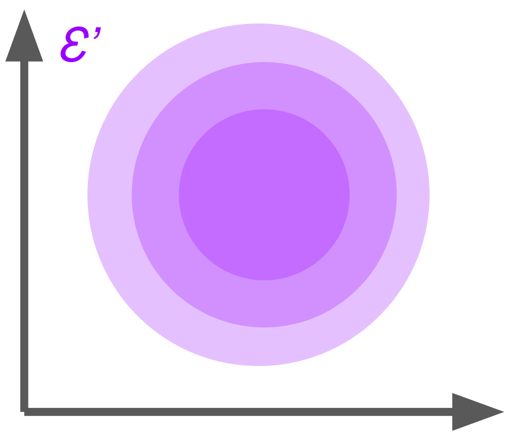
IRC Safety, Unclustered Radiation, & Wasserstein Geometry

Gambhir Thaler

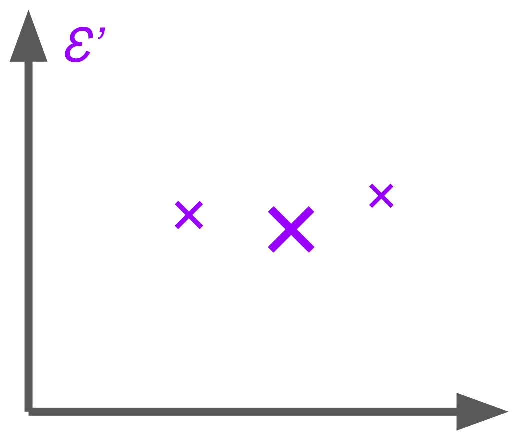
Shapes as Energy Flows

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

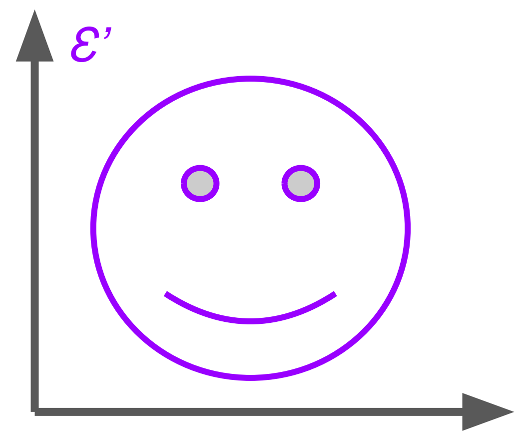
Can make anything you want! Even continuous or complicated shapes.
(Or, something a physicist can predict!)



Shape = 2D Gaussian



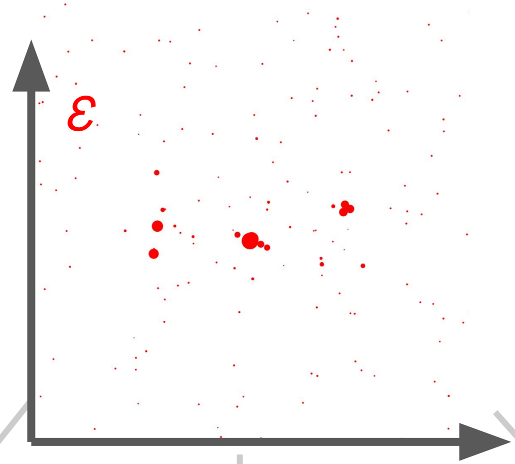
Shape = 3 Points



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!

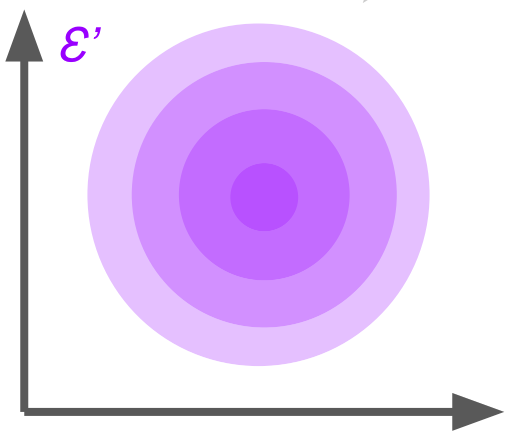


Answers the question: "How much like the shape $\mathcal{E}'$ is my event $\mathcal{E}$?"

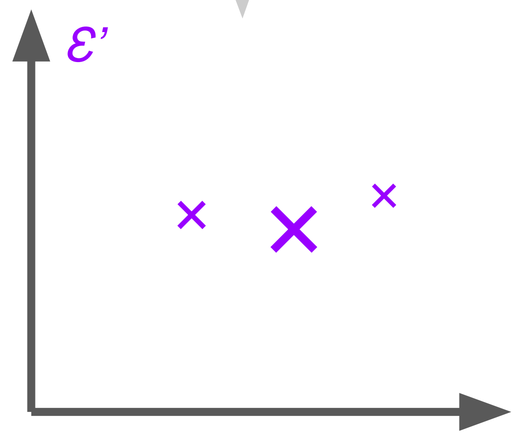
Med EMD($\mathcal{E}$, $\mathcal{E}'$)
"Gausiness"

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

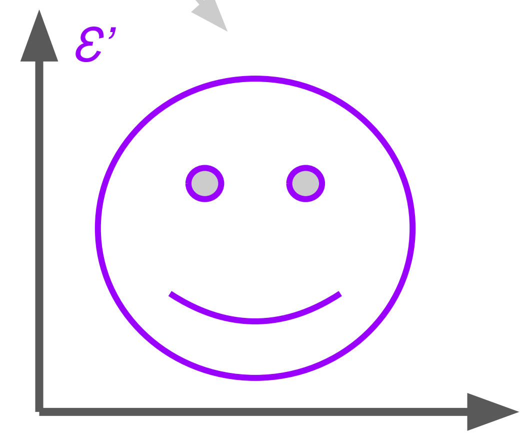
High EMD($\mathcal{E}$, $\mathcal{E}'$)
"Smileyness"



Shape = 2D Gaussian

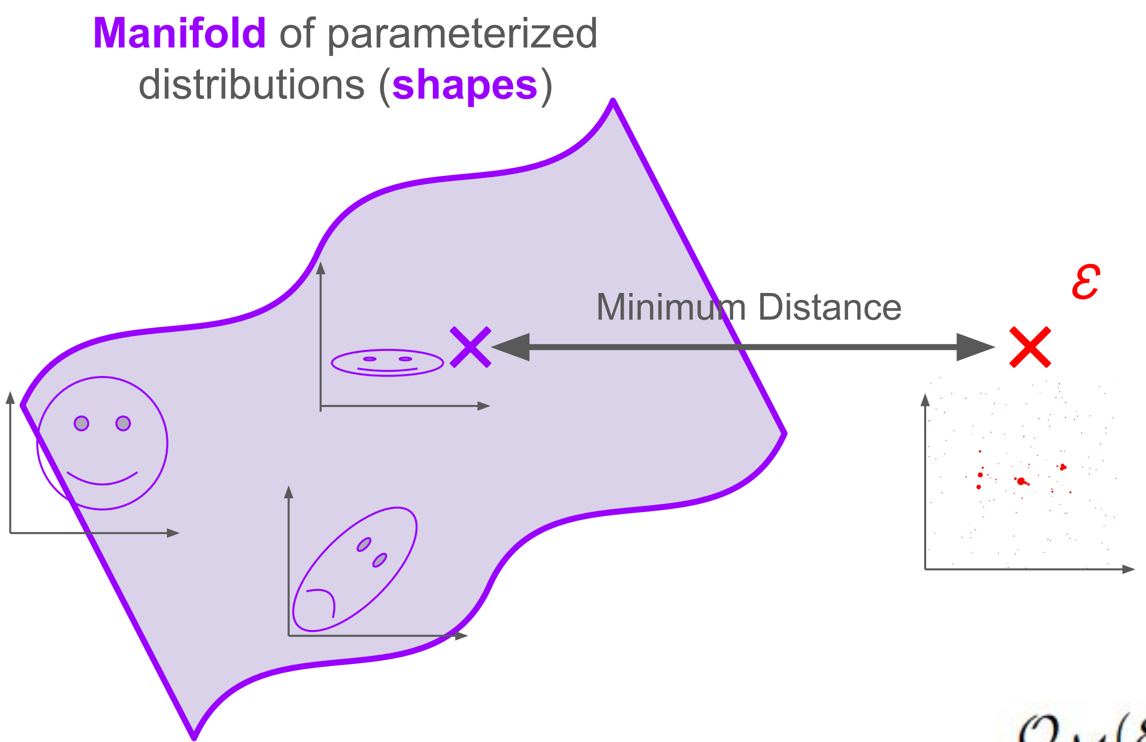


Shape = 3 Points



Shape = Smile

Manifold Learning



Idealizing our jets is a form of dimensionality reduction!

Project our complicated **jet** to an easy-to-understand, low parameter, idealized **shape** – this is manifold learning, ubiquitous in ML.

The only difference – we are considering manifolds of statistical distributions, so use EMD to measure distance!

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

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

The *SHAPER* Framework

Shape-**H**unting **A**lgorithm using **P**arameterized **E**nergy **R**econstruction

- Framework for defining and building IRC-safe observables using parameterized objects
- Easy to programmatically define new observables by specifying parameterization, or by combining shapes
- Returns shapiness and optimal shape parameters

Download with `pip install pyshaper`

```python
# SHAPER
from pyshaper.CommonObservables import buildCommmonObservables
from pyshaper.Observables import Observable
from pyshaper.Shaper import Shaper

# Use Pre-built Observables (N-subjets, rings, disks, ellipses)
observables, pointers = buildCommmonObservables(N = 3, beta = 1, R = 0.8)

# Make new observables by defining energy probability distributions
def uniform_sampler(N, param_dict):
    points = torch.FloatTensor(N, 2).uniform_(-0.8, 0.8).to(device)
    zs = torch.ones((N,)).to(device) / N
    return (points, zs)

observables["Isotropy"] = Observable({}, uniform_sampler, beta = 1, R = 0.8)

# Run SHAPER on data
shaper = Shaper(observables, device = "cpu")
shaper.to(device)
emds, params = shaper.calculate(dataset)

# Done!
```

Example usage of *pySHAPER*, a python implementation of *SHAPER*.

Not just for jet physics – use this to perform *any* statistical manifold learning!

[J. Feydy, tel.archives-ouvertes.fr/tel-02945979;
B. Charlier, J. Feydy, J. Alexis Glaunès F. D. Collin, G. Durif, JMLR:v22:20-275;
J. Feydy, T. Séjourné, F. X. Vialard, S. Amari, A. Trouvé, G. Peyré, 1810.08278;
O. Kitouni, N. Nolte, M Williams, 2209.15624]

Estimating **Wasserstein**

We need a *differentiable, fast* approximation to the EMD for our minimizations

Sinkhorn Divergence: A strictly convex approximation to EMD! Kantorovich potential formalism:

Algorithm 3.4: Symmetric Sinkhorn algorithm, with debiasing

Parameters: Cost function $\mathbf{C} : (x_i, y_j) \in \mathcal{X} \times \mathcal{X} \mapsto \mathbf{C}(x_i, y_j) \in \mathbb{R}$,
Temperature $\varepsilon > 0$.
Input: Positive measures $\alpha = \sum_{i=1}^{N} \alpha_i \delta_{x_i}$ and $\beta = \sum_{j=1}^{M} \beta_j \delta_{y_j}$ with the same mass.

1: $f_i^{\beta\to\alpha},\ g_j^{\alpha\to\beta},\ f_i^{\alpha\leftrightarrow\alpha},\ g_j^{\beta\leftrightarrow\beta} \leftarrow \mathbf{0}_{\mathbb{R}^N},\ \mathbf{0}_{\mathbb{R}^M},\ \mathbf{0}_{\mathbb{R}^N},\ \mathbf{0}_{\mathbb{R}^M}$ ▷ Dual vectors.
2: **repeat** ▷ The four lines below are executed **simultaneously**.
3: $f_i^{\beta\to\alpha} \leftarrow \tfrac{1}{2} f_i^{\beta\to\alpha} + \tfrac{1}{2}\min_{y\sim\beta,\varepsilon}\left[\mathbf{C}(x_i, y) - g^{\alpha\to\beta}(y)\right]$, ▷ $\alpha \leftarrow \beta$
$g_j^{\alpha\to\beta} \leftarrow \tfrac{1}{2} g_j^{\alpha\to\beta} + \tfrac{1}{2}\min_{x\sim\alpha,\varepsilon}\left[\mathbf{C}(x, y_j) - f^{\beta\to\alpha}(x)\right]$, ▷ $\beta \leftarrow \alpha$
$f_i^{\alpha\leftrightarrow\alpha} \leftarrow \tfrac{1}{2} f_i^{\alpha\leftrightarrow\alpha} + \tfrac{1}{2}\min_{x\sim\alpha,\varepsilon}\left[\mathbf{C}(x_i, x) - f^{\alpha\leftrightarrow\alpha}(x)\right]$, ▷ $\alpha \leftarrow \alpha$
$g_j^{\beta\leftrightarrow\beta} \leftarrow \tfrac{1}{2} g_j^{\beta\leftrightarrow\beta} + \tfrac{1}{2}\min_{y\sim\beta,\varepsilon}\left[\mathbf{C}(y, y_j) - g^{\beta\leftrightarrow\beta}(y)\right]$. ▷ $\beta \leftarrow \beta$
4: **until** convergence up to a set tolerance. ▷ Monitor the updates on the potentials.
5: **return** $f_i^{\beta\to\alpha} - f_i^{\alpha\leftrightarrow\alpha},\ g_j^{\alpha\to\beta} - g_j^{\beta\leftrightarrow\beta}$ ▷ Debiased dual potentials $F(x_i)$ and $G(y_j)$.

Implemented using the KerOps+GeomLoss Python Package!

$$\mathcal{O}(\mathcal{E}) = \min_{\mathcal{E}_\theta \in \mathcal{M}} \left[S_\epsilon(\mathcal{E}, \mathcal{E}')\right] \text{ and } \theta(\mathcal{E}) = \operatorname*{argmin}_{\mathcal{E}_\theta \in \mathcal{M}} \left[S_\epsilon(\mathcal{E}, \mathcal{E}')\right], \qquad \text{where}$$

$$S_\epsilon(\mathcal{E}, \mathcal{E}') = \mathrm{OT}_\epsilon(\mathcal{E}, \mathcal{E}_\theta) - \frac{1}{2}\mathrm{OT}_\epsilon(\mathcal{E}, \mathcal{E}) - \frac{1}{2}\mathrm{OT}_\epsilon(\mathcal{E}_\theta, \mathcal{E}_\theta), \qquad \text{and}$$

$$\mathrm{OT}_\epsilon(\mathcal{E}, \mathcal{E}') = \max_{f, g: \mathcal{X} \to \mathbb{R}} \left[\sum_{i=1}^{M} E_i f(x_i) + \sum_{j=1}^{N} E_j' g(y_j) - \epsilon^\beta \log\left(\sum_{ij} E_i E_j' \left(e^{\frac{1}{\epsilon^\beta}\left(f(x_i) + g(y_j) - \frac{d(x_i, y_j)^\beta}{R^\beta}\right)}\right)\right)\right]$$

Nolte

Williams

Kitouni

Can take gradients with respect to the entire event – very useful!

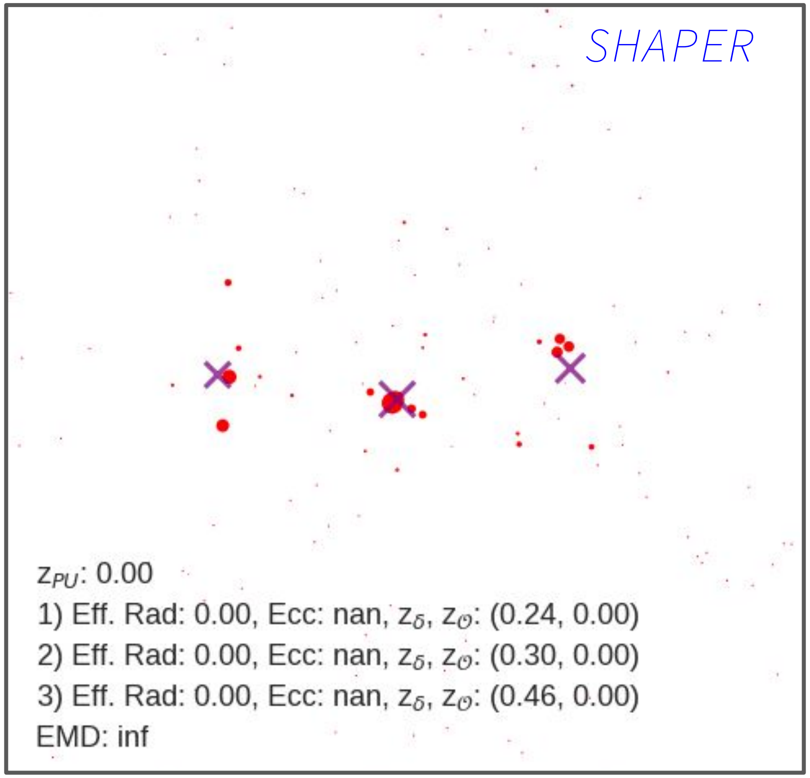
See “Finding NEEMo” for alternatives, also IAIFI!

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!

Think of as a generalization of **k-means clustering**.

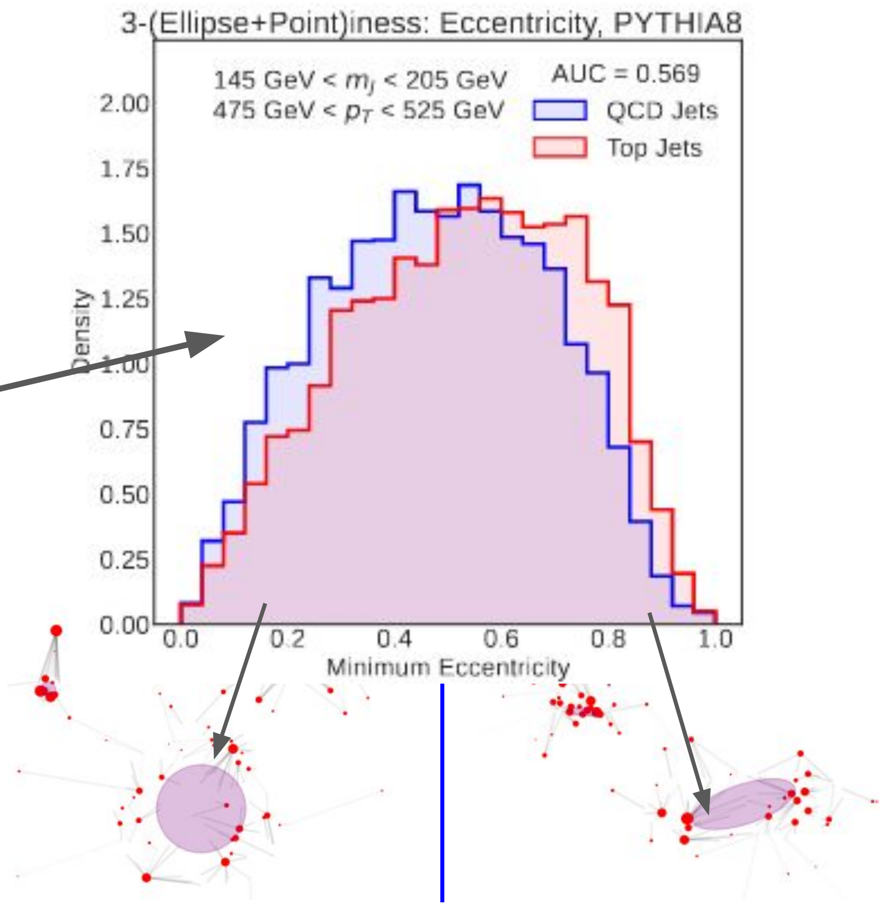
New IRC-Safe Observables

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!



Low Eccentricity (.001) High Eccentricity (.972)

[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

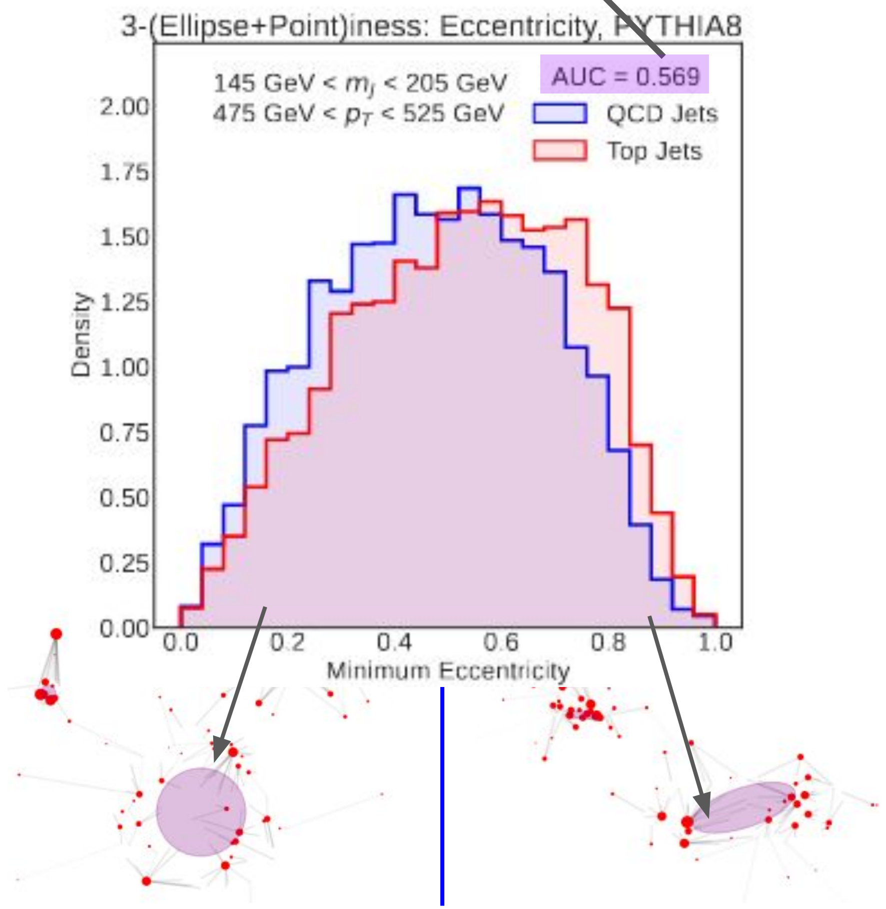
Eccentricity can distinguish between top/QCD jets? Nontrivial result, *could not have been done before*!

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!



Low Eccentricity (.001) High Eccentricity (.972)

Automatic Grooming with Shapes

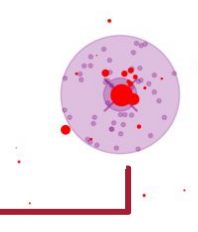
Use shapes to approximate events and extract masses – model pileup with a uniform background with floating weight!

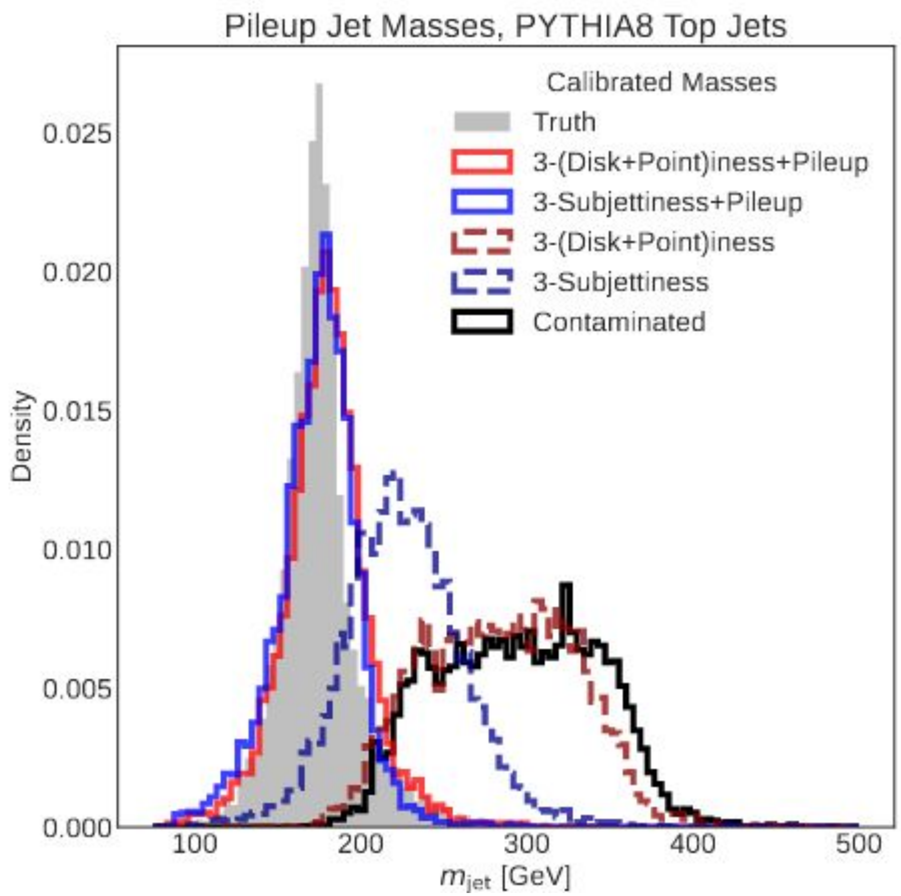
No external hyperparameters, unlike softdrop. Only need to assume pileup is uniform!

Contaminate top jets with 5-30% extra energy spread uniformly in an 0.8x0.8 plane

Consider 4 shapes:

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

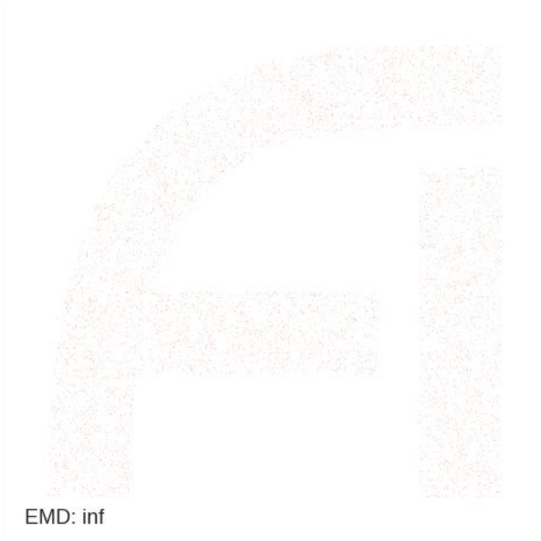




Can also consider ellipses instead of disks – only marginally better performance

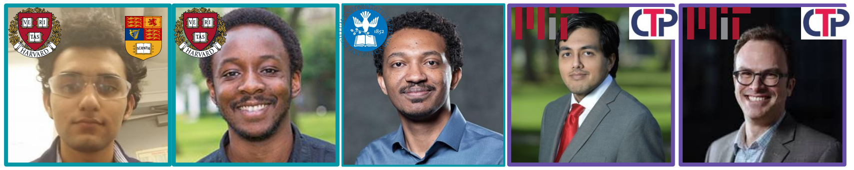
Some Last Fun Animations

The **50-** and **100-Ellipsinesses** of some (extremely unlikely) collider events



Statistical Manifold learning on sophisticated, high dimensional spaces!

… Can you hear the shape of these "jets"?

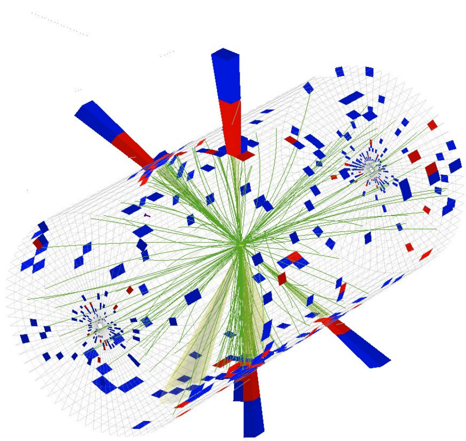


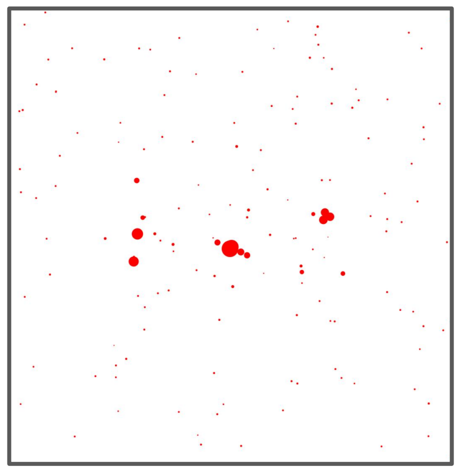
[Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]

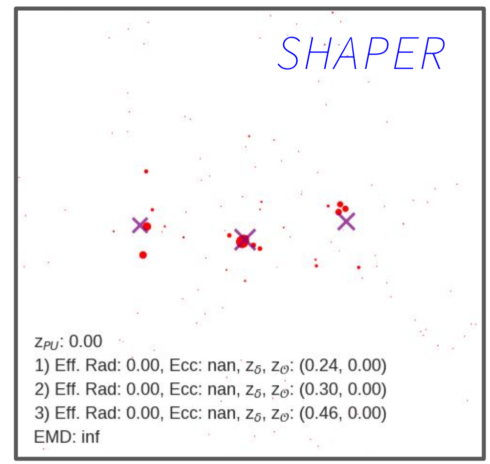
Download with `pip install pyshaper`

Conclusion

- *SHAPER* is a framework for manifold learning on distributions, using EMD inspired by **K-Deep Simplices** plus **physics-inspired structure**
- **Jet physics** maps exactly onto this manifold learning problem, allowing us to build custom observables and jet algorithms!
- Made possible by collaborations across fields and institutions!





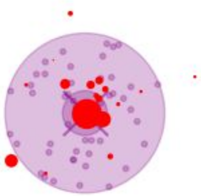


Collider Data → **Energy Flow** → **Shapes**

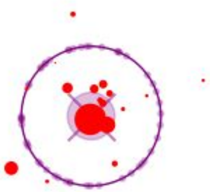
Yes, you *CAN* hear the shape of a jet!

Appendices

Observables on CMS OpenData

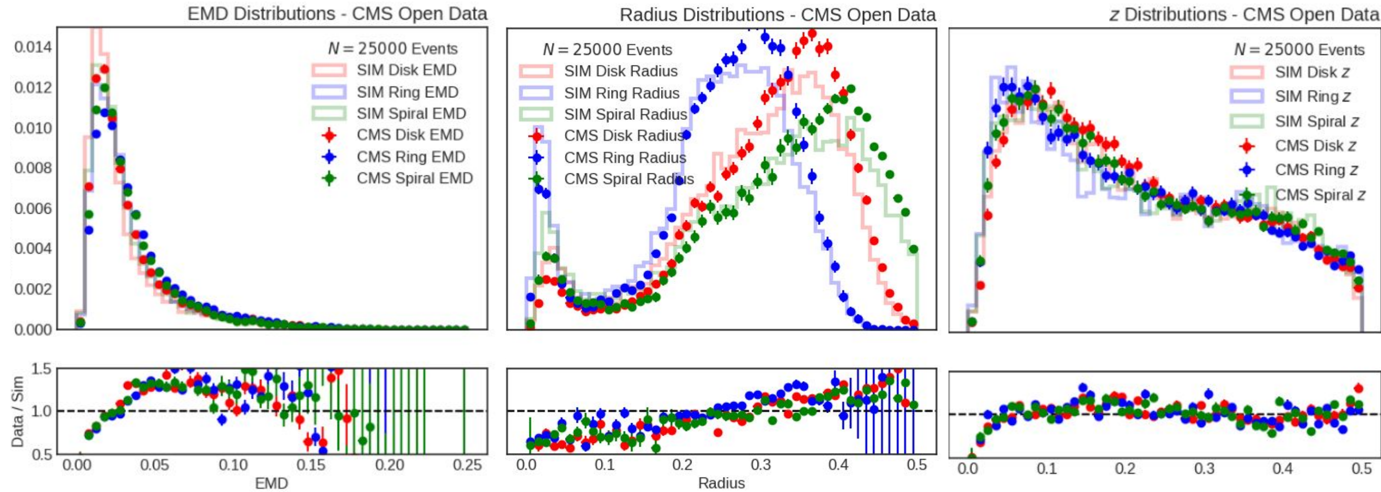


Disk + δ-function



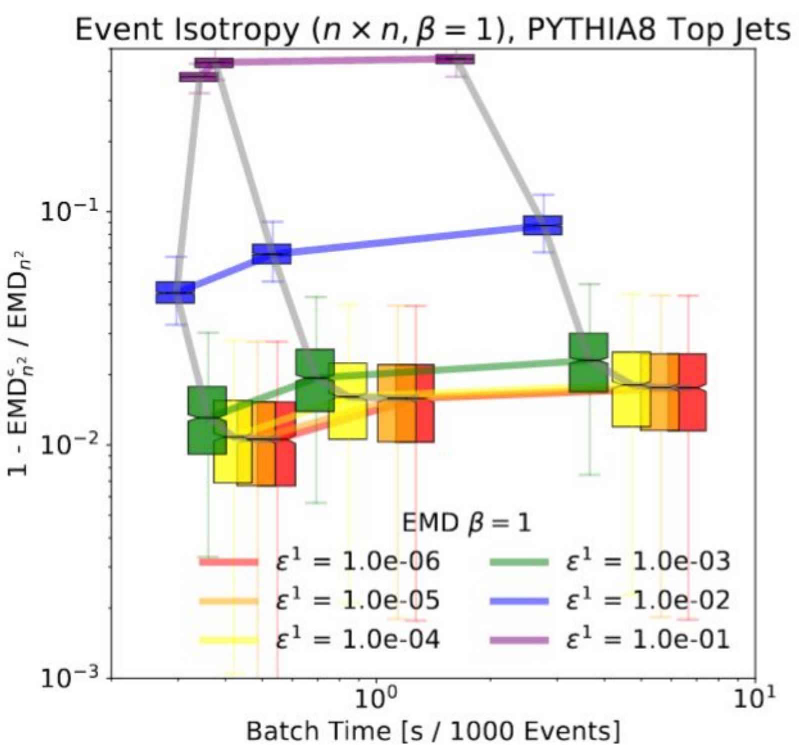
Ring + δ-function

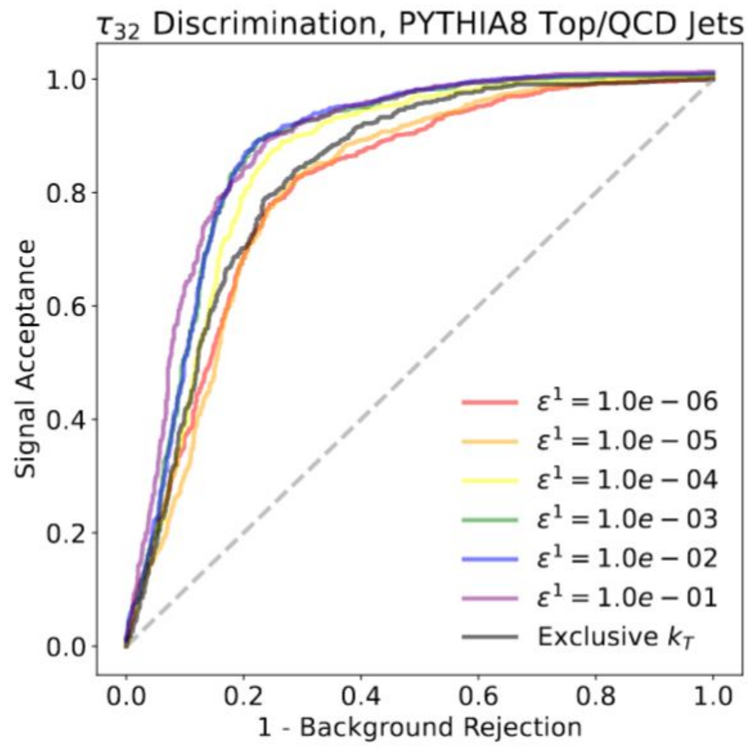
Spiral + δ-function



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

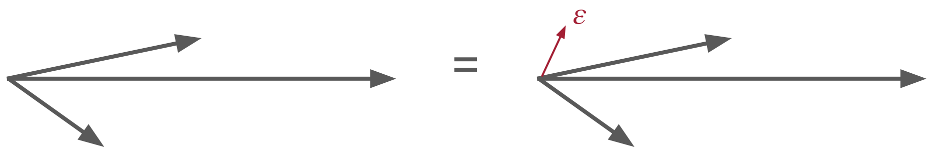
Performance Benchmarks



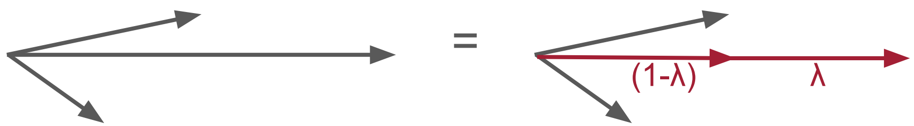


IRC Safety

Infrared Safety: An observable is unchanged under a soft emission



Collinear Safety: An observable is unchanged under a collinear splitting



Observables and Wasserstein

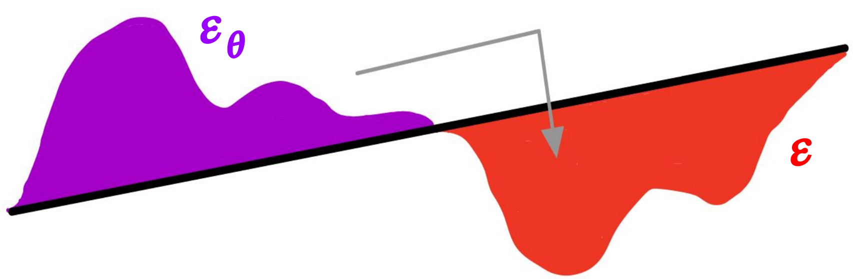
It can be shown that *any* observable on events, that* …

1. … is non-negative and finite
2. … is IRC-safe
3. … is translationally invariant
4. … is invariant to particle labeling
5. … respects the detector metric *faithfully***

… can be written as an optimization of the **Wasserstein Metric (Earth/Energy Mover's Distance)** between the real event and a manifold of idealized energy flows

$$\mathcal{O}_{\mathcal{M}}(\mathcal{E}) = \min_{\mathcal{E}'_\theta \in \mathcal{M}} \text{EMD}(\mathcal{E}, \mathcal{E}'_\theta)$$

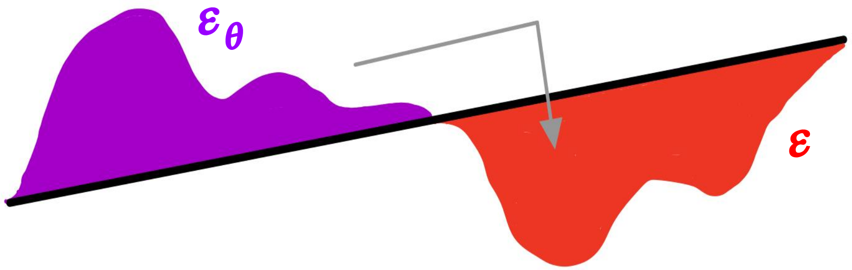
$$\theta = \underset{\mathcal{E}'_\theta \in \mathcal{M}}{\text{argmin}}\, \text{EMD}(\mathcal{E}, \mathcal{E}'_\theta)$$



EMD = Work done to move "dirt" optimally

*Ask me for more details on this offline!
** Preserves distances between *extended* objects, not just points

Observables and Wasserstein



EMD = Work done to move "dirt" optimally

$$\mathcal{O}_{\mathcal{M}}(\mathcal{E}) = \min_{\mathcal{E}'_\theta \in \mathcal{M}} \mathrm{EMD}(\mathcal{E}, \mathcal{E}'_\theta)$$

$$\theta = \underset{\mathcal{E}'_\theta \in \mathcal{M}}{\mathrm{argmin}}\, \mathrm{EMD}(\mathcal{E}, \mathcal{E}'_\theta)$$

4. … is invariant to particle labeling
5. … respects the detector metric *faithfully*

… c

Was

Dist

mar

$$\mathrm{EMD}^{(\beta,R)}(\mathcal{E}, \mathcal{E}') = \min_{\pi_{ij} \geq 0} \left[\frac{1}{\beta R^\beta} \sum_{i=1}^{M} \sum_{j=1}^{N} \pi_{ij} d_{ij}^\beta \right] + |\Delta E_{\mathrm{tot}}|$$

$$\sum_{i=1}^{M} \pi_{ij} \leq E'_j, \; \sum_{j=1}^{N} \pi_{ij} \leq E_i \text{ and } \sum_{j=1}^{N} \pi_{ij} = \min(E_{\mathrm{tot}}, E'_{\mathrm{tot}})$$

Ask me for more details on this after!