Point Cloud Strategies for Boosted Tops

Patrick T. Komiske III
Massachusetts Institute of Technology
Center for Theoretical Physics

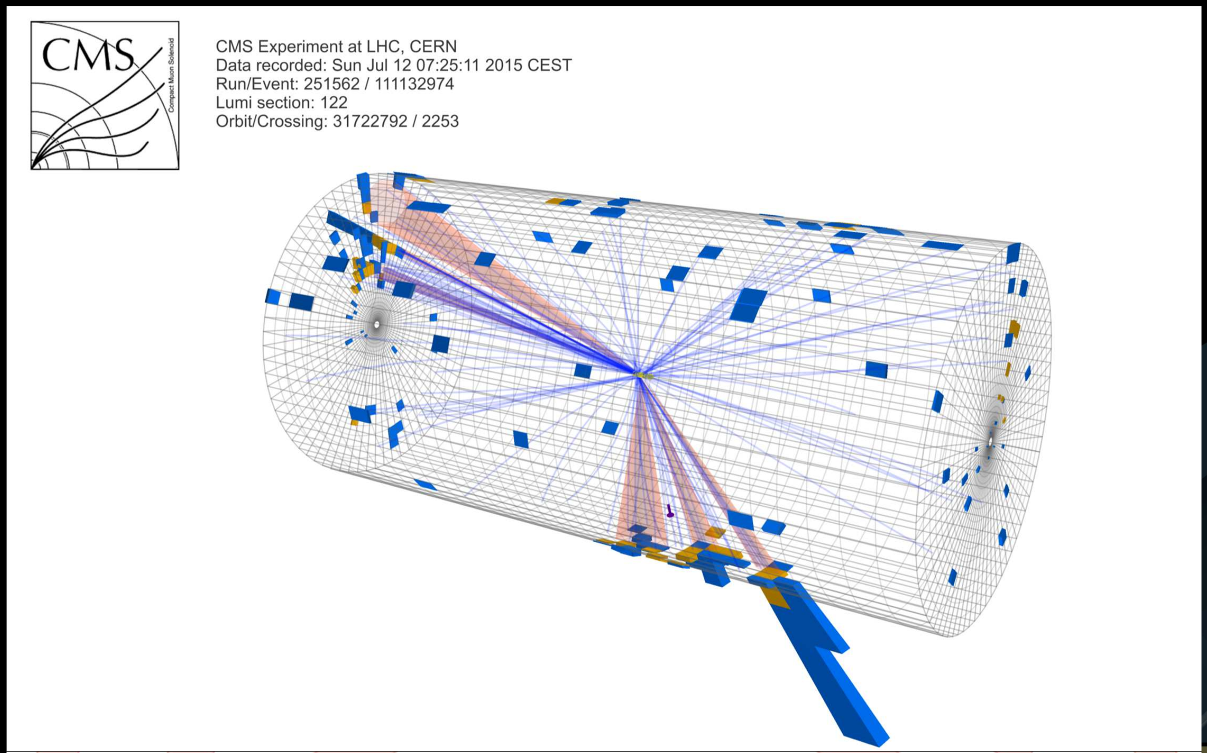
Boosted Objects for New Physics Searches
Fermilab, Illinois – 11/13/2018

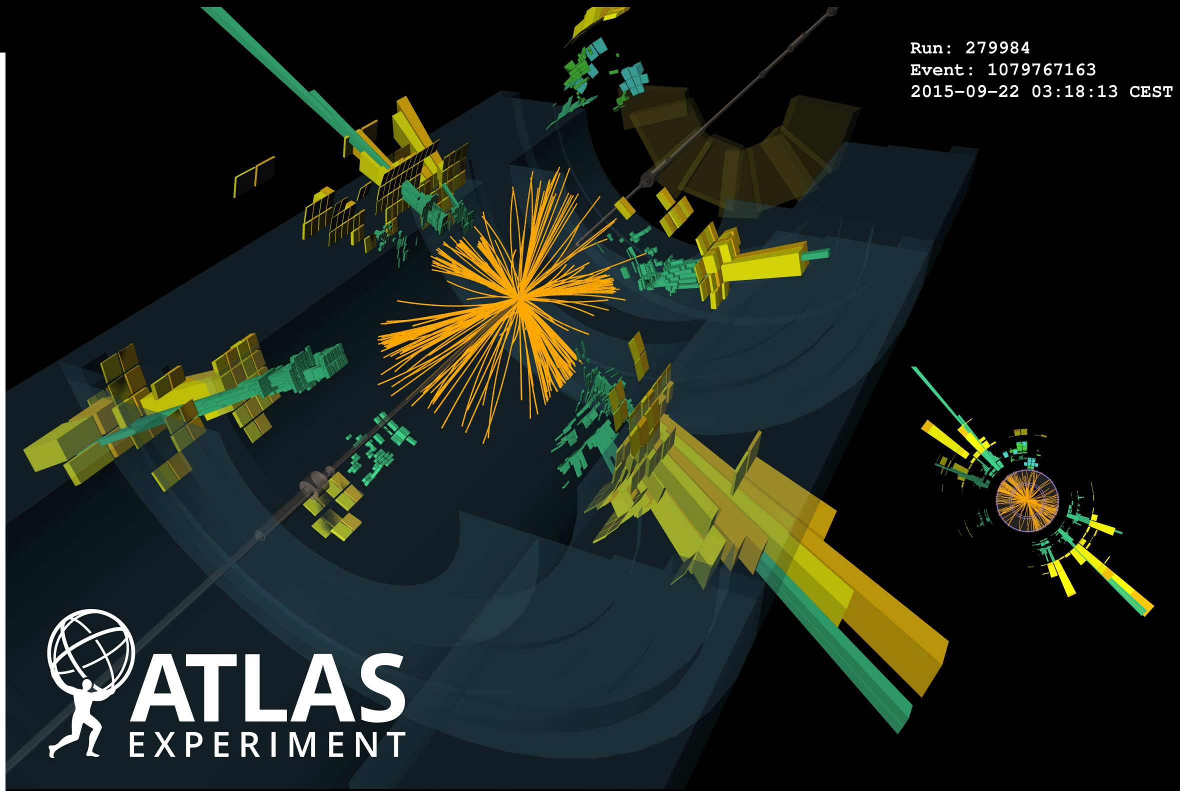
Based on work with Eric Metodiev and Jesse Thaler

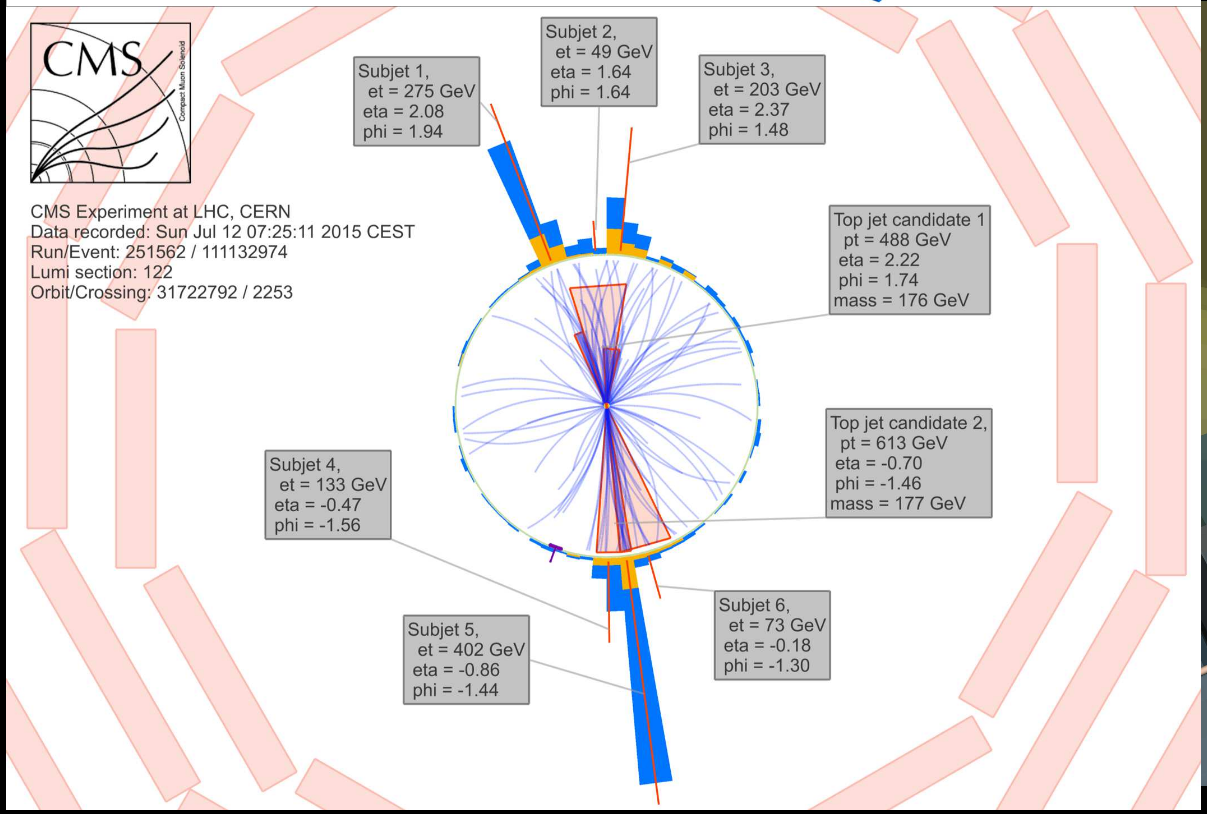
1712.07124   1810.05165

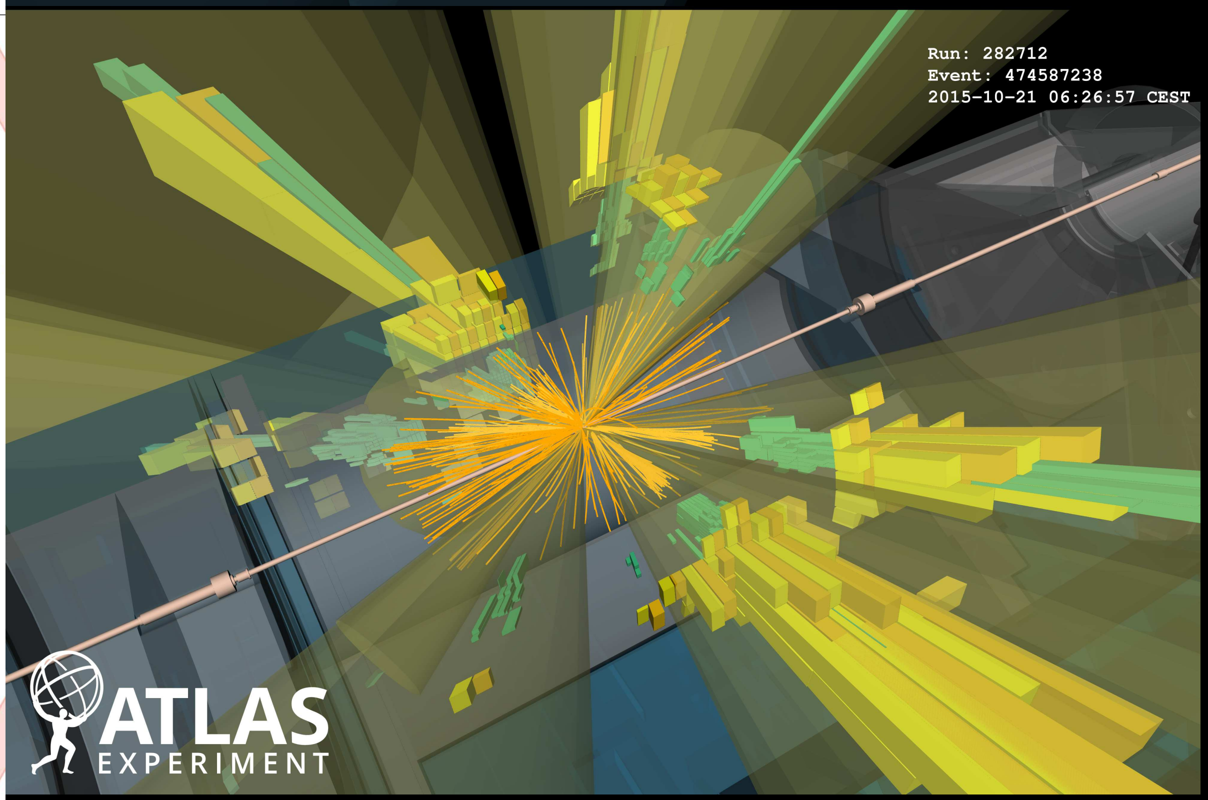
https://energyflow.network

Boosted Event Topologies at the LHC





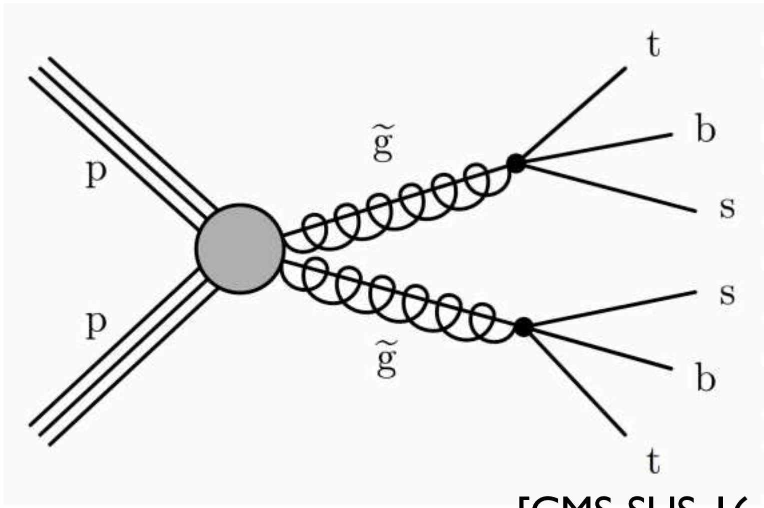




Why Boosted Tops?

Many models of new physics contain boosted Standard Model final states

e.g. Z' ⟶ $t\bar{t}$, cascade decays, various SUSY scenarios



[CMS-SUS-16-040]

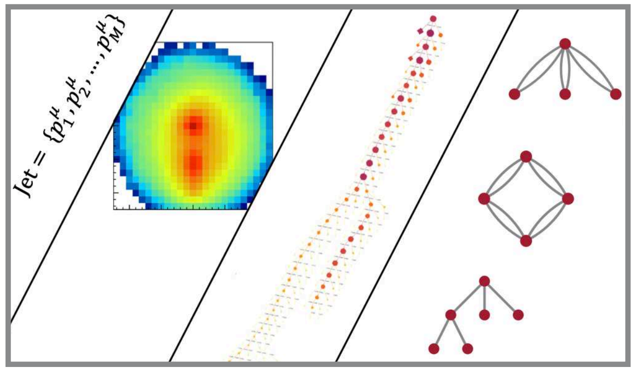
Boosted tops provide a way of testing and benchmarking multi-prong substructure techniques

Modern boosted top tagging is extremely effective!

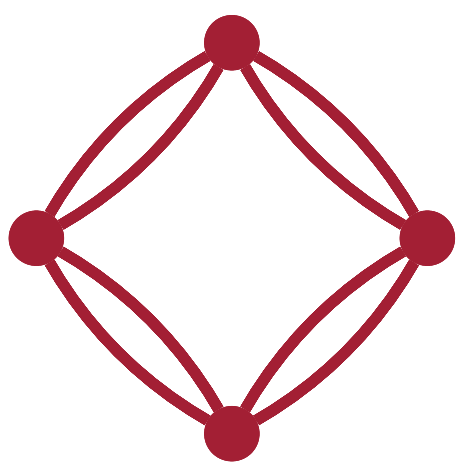
Current CMS default – AK8 PUPPI jets, b tagged subjets, Soft Drop mass cuts, τ_{32} cut

[CMS-B2G-17-017, 1810.05905]

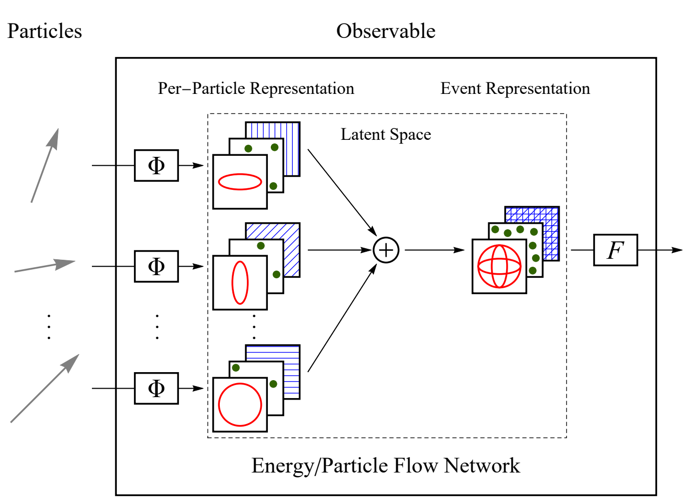
Goal of this talk: *Demonstrate alternative, bottom up approaches to top tagging that go back to the basics and attempt to harness the power of the ML revolution*



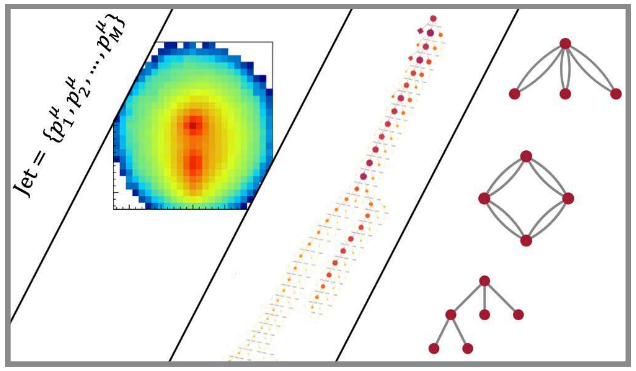
Jets as Point Clouds



Energy Flow Polynomials



Energy Flow Networks



Jets as Point Clouds

Energy Flow Polynomials

Energy Flow Networks

What is a Jet?

An **unordered**, **variable length** collection of particles

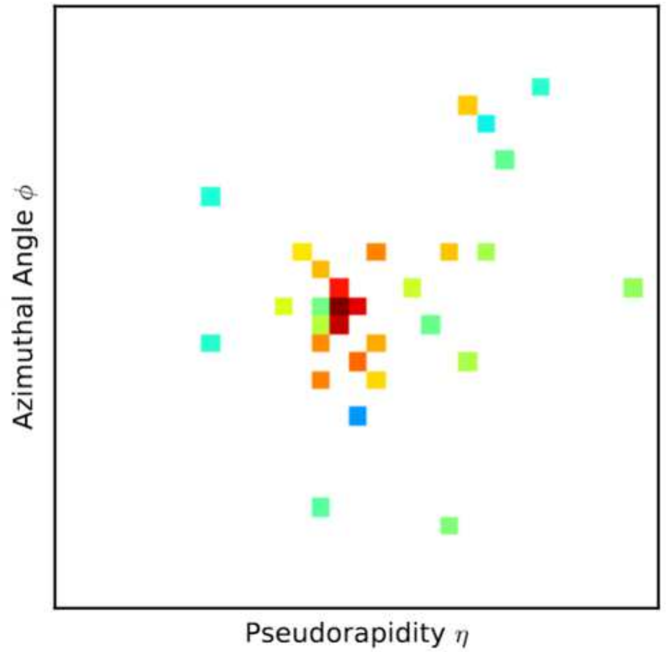
Due to quantum-mechanical indistinguishability
Due to probabilistic nature of jet formation

$$J\left(\{p_1^\mu,\ldots,p_M^\mu\}\right) = J(\{p_{\pi(1)}^\mu,\ldots,p_{\pi(M)}^\mu\}), \qquad \underbrace{M \geq 1}_{\text{Multiplicity}}\ , \quad \underbrace{\forall\,\pi \in S_M}_{\text{Permutations}}$$

p_i^μ represents *all* the particle properties:

- Four-momentum – $(E, p_x, p_y, p_z)_i^\mu$
- Other quantum numbers (e.g. particle id, charge)
- Experimental information (e.g. vertex info, quality criteria, PUPPI weights)

Contrast with jet images
d dimensional particles, *N x N* pixels
dN^2 jet image inputs, *dM* point cloud inputs



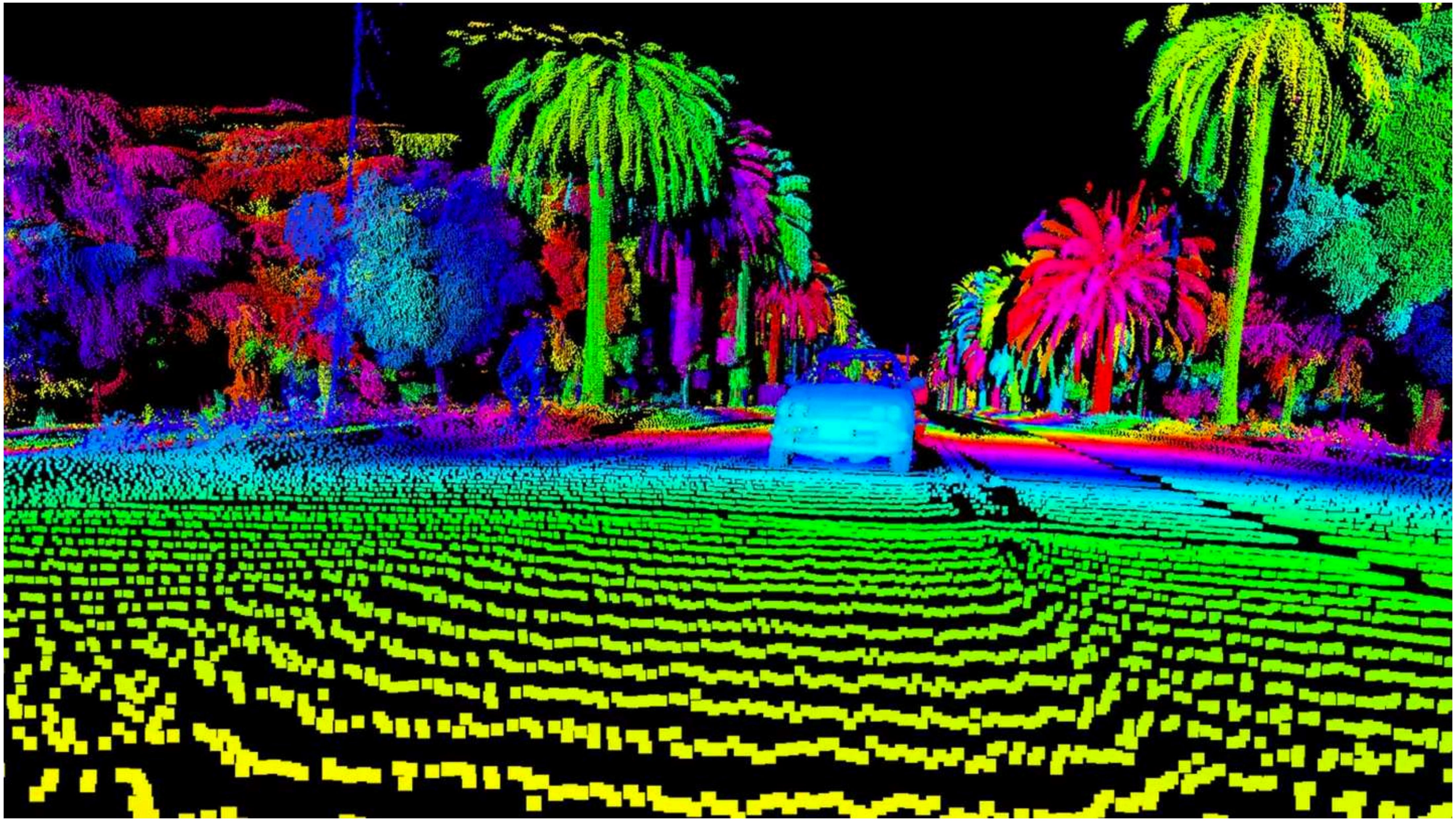
Particles are the medium in which theory and experiment meet

Success of CMS Particle Flow validates particles as fundamental objects in particle physics

Point Clouds

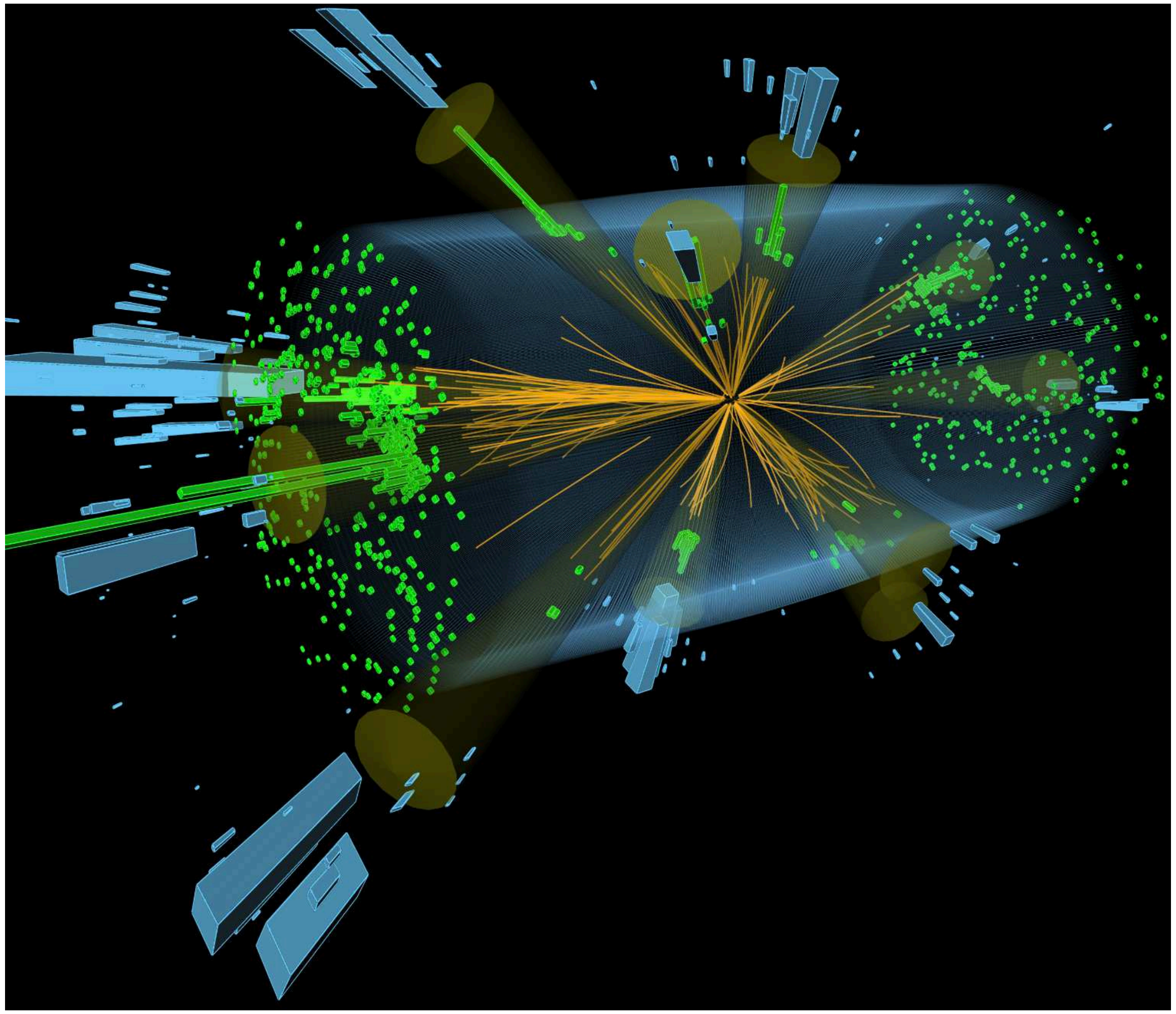
Point cloud: "A set of data points in space" –Wikipedia

LIDAR data from self-driving car sensor



Particle Collision Events as Point Clouds

Point cloud: "A set of data points in space" –Wikipedia

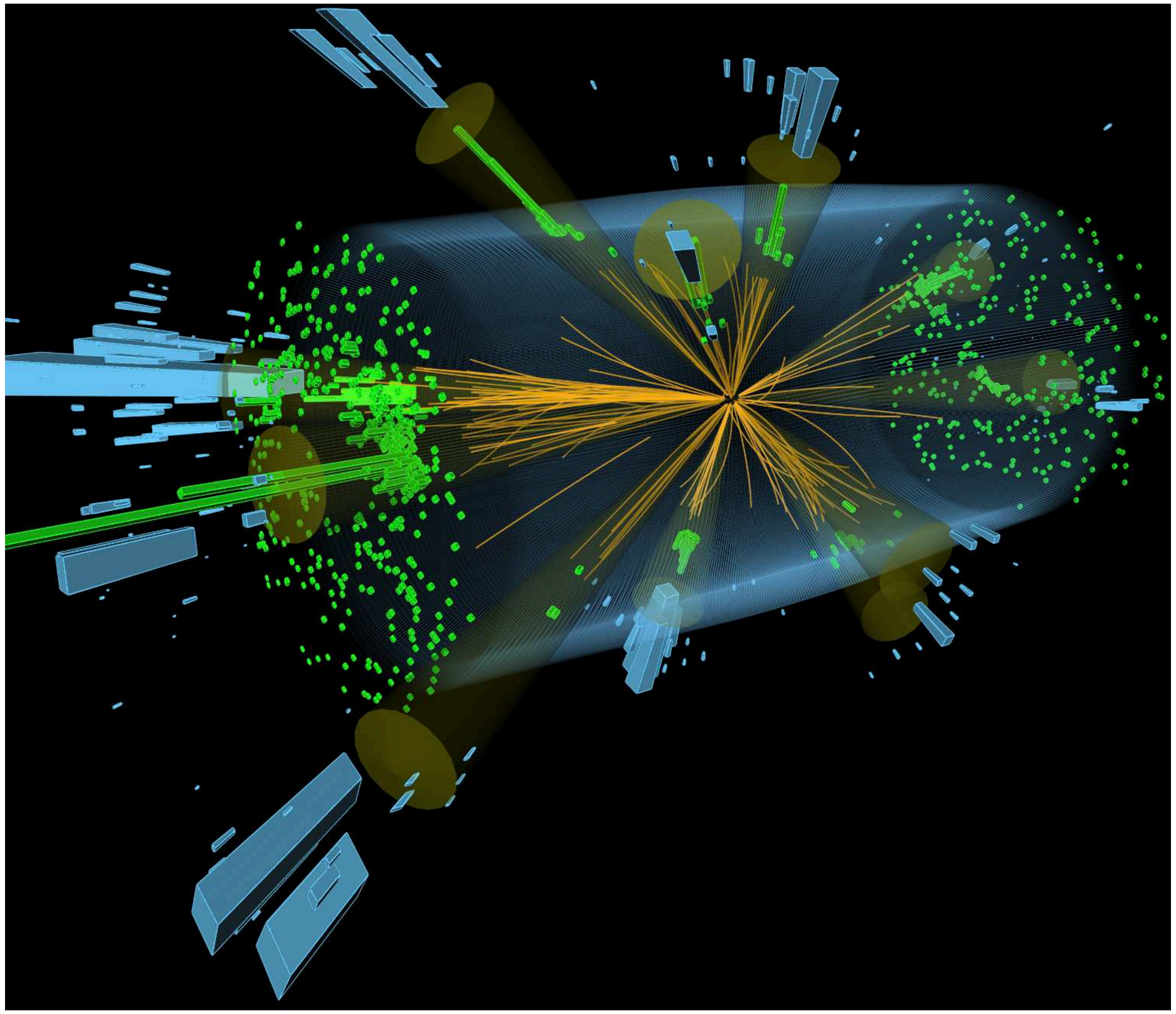


Multi-jet event at CMS

Particle Collision Events as Point Clouds

Point cloud: "A set of data points in space" –Wikipedia

Jet/event — Particles — Feature space



Multi-jet event at CMS

Processing Point Clouds

Methods for processing point clouds/jets should respect the appropriate symmetries

Variable constituent multiplicity requires at least one of:

Preprocessing to another representation (jet images, *N*-subjettiness, etc.)

Truncation to an (arbitrary) fixed size

Recurrent NN structure

Particle permutation symmetry requires:

Permutation symmetric observables

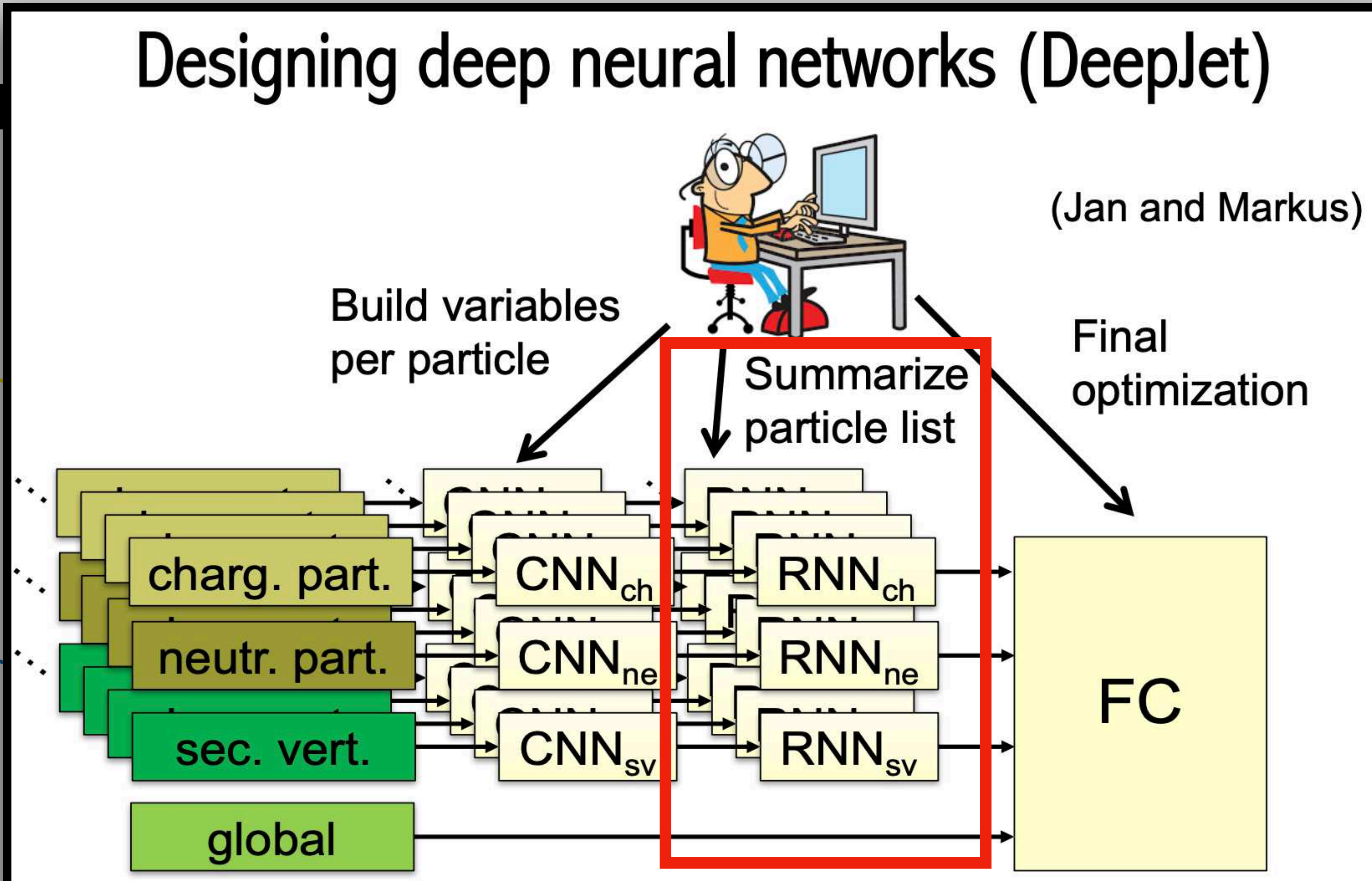
Permutation symmetric architectures

Processing Point Clouds

M *he*

Var

Par



Slide from [Markus Stoye's talk]

Jet Representations ⟷ Analysis Tools

Two key choices when analyzing jets

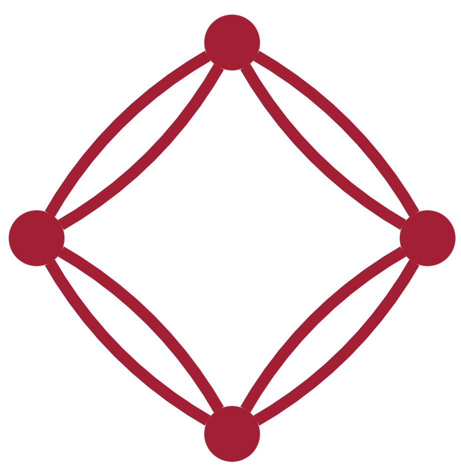
How to represent the jet	⟺	How to analyze that representation
• Single expert observable • A few expert observables • Many expert observables • Jet images • *N*-subjettiness basis • Energy flow polynomials	**Fixed Processing**	• Threshold cut • Multidimensional likelihood • Boosted decision tree (BDT), shallow neural network (NN) • Convolutional NN (CNN) • Dense neural network (DNN) • Linear classification
• List of particles • Clustering tree • Set of particles	**Flexible Processing**	• Recurrent NN (RNN) • Recursive NN • Energy flow network

Jet Representations ⟷ Analysis Tools

Two key choices when analyzing jets

How to represent the jet	⟺	How to analyze that representation
• Single expert observable		• Threshold cut
• A few expert observables		• Multidimensional likelihood
• Many expert observables		• Boosted decision tree (BDT), shallow neural network (NN)
	Fixed Processing	
• Jet images		• Convolutional NN (CNN)
• *N*-subjettiness basis		• Dense neural network (DNN)
• Energy flow polynomials		• Linear classification
• List of particles		• Recurrent NN (RNN)
• Clustering tree	**Flexible Processing**	• Recursive NN
• Set of particles		• Energy flow network

Jets as Point Clouds



Energy Flow Polynomials

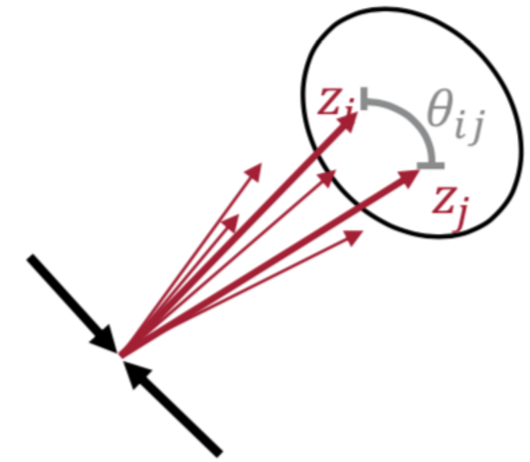
Fixed point cloud processing

Energy Flow Networks

Energy Flow Polynomials (EFPs)

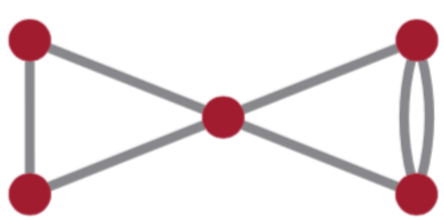
[PTK, Metodiev, Thaler, 1712.07124]

$$\text{EFP}_G = \underbrace{\sum_{i_1=1}^{M} \cdots \sum_{i_N=1}^{M}}_{\text{Correlator}} \text{ of } \underbrace{z_{i_1} \cdots z_{i_N}}_{\text{Energies}} \text{ and } \underbrace{\prod_{(k,\ell)\in G} \theta_{i_k i_\ell}}_{\text{Angles}}$$



Generalizes many well-known and studied classes of energy correlators observables

A family of energy correlators with angular structures determined by multigraphs



$$= \sum_{i_1=1}^{M} \sum_{i_2=1}^{M} \sum_{i_3=1}^{M} \sum_{i_4=1}^{M} \sum_{i_5=1}^{M} z_{i_1} z_{i_2} z_{i_3} z_{i_4} z_{i_5} \theta_{i_1 i_2} \theta_{i_2 i_3} \theta_{i_1 i_3} \theta_{i_1 i_4} \theta_{i_1 i_5} \theta_{i_4 i_5}^2$$

<u>Multigraph correspondence</u>

$\longleftrightarrow$ z_{i_j}

$\longleftrightarrow$ $\theta_{i_k i_l}$

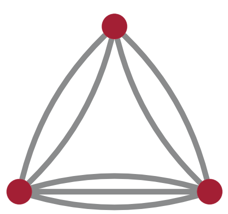
Energy and Angle Measure

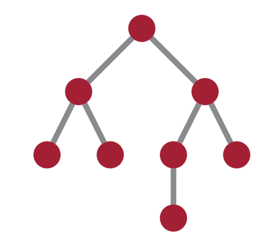
Hadronic : $z_i = \dfrac{p_{Ti}}{\sum_j p_{Tj}}, \quad \theta_{ij} = \left(\Delta y_{ij}^2 + \Delta\phi_{ij}^2\right)^{\beta/2}$

Linear Basis of IRC-Safe Observables

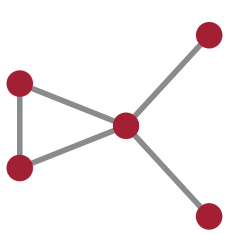
One can show via the Stone-Weierstrass approximation theorem that any IRC-safe observable is a linear combination of EFPs

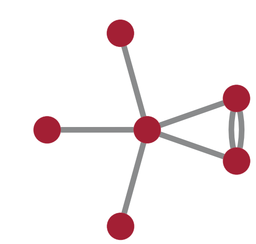
$$\mathcal{S} \simeq \sum_{G \in \mathcal{G}} s_G \mathrm{EFP}_G, \quad \mathcal{G} \text{ a set of multigraphs}$$





Multivariate combinations of EFPs only require <u>linear methods</u> to achieve full generality

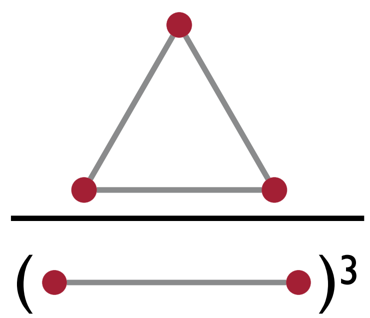




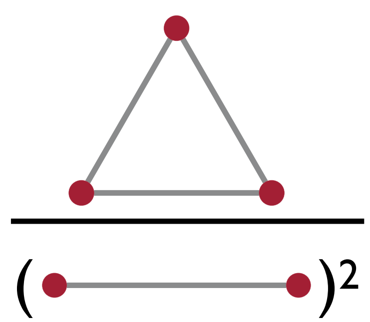
Strategy: Learn coefficients s_G via linear regression or classification

Familiar Observables as EFPs

$m_J^2 =$ 

$D_2 =$ 

[Larkoski, Moult, Neill, 2014]

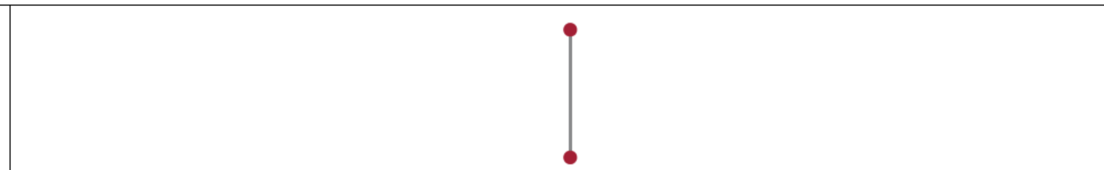
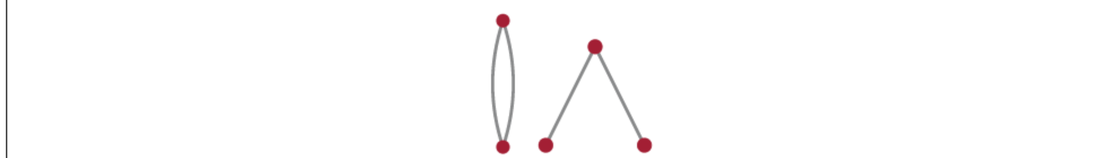
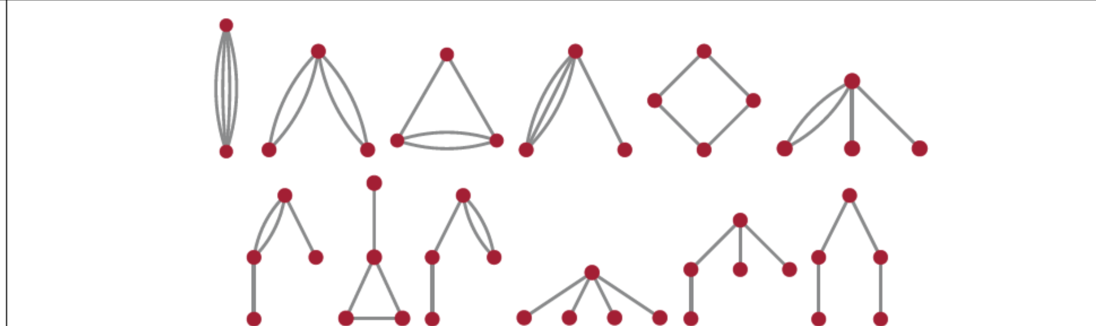
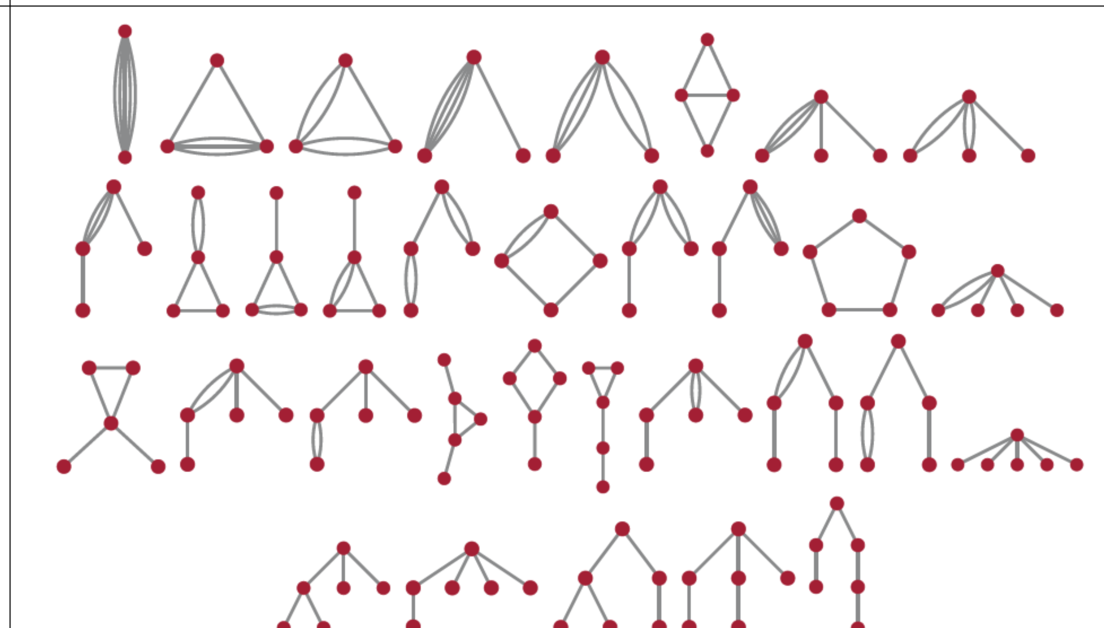
$C_2 =$ 

[Larkoski, Salam, Thaler, 2013]

Energy correlation functions are complete graphs

Even angularities are exact linear combinations of EFPs

EFPs organized by degree d – number of edges

Degree	Connected Multigraphs
$d=1$	
$d=2$	
$d=3$	
$d=4$	
$d=5$	

Computation Complexity of EFPs – Variable Elimination

Naive computation complexity of an energy correlator is $\mathcal{O}(M^N)$

For ~100 particles this becomes intractable for $N > 4$

Computation Complexity of EFPs – Variable Elimination

Naive computation complexity of an energy correlator is $\mathcal{O}(M^N)$

For ~100 particles this becomes intractable for $N > 4$

⇨ EnergyCorrelator `fjcontrib` package gives up in this case

```
// if N > 5, then throw error
if (_N > 5) {
    throw Error("EnergyCorrelator is only hard coded for N = 0,1,2,3,4,5");
}
```

Computation Complexity of EFPs – Variable Elimination

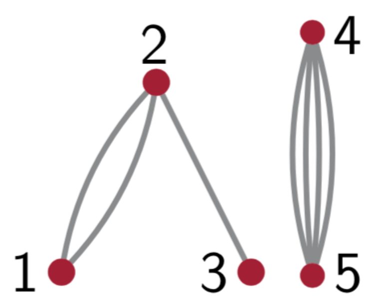
Naive computation complexity of an energy correlator is $\mathcal{O}(M^N)$

For ~100 particles this becomes intractable for $N > 4$

⇨ EnergyCorrelator `fjcontrib` package gives up in this case

```
// if N > 5, then throw error
if (_N > 5) {
    throw Error("EnergyCorrelator is only hard coded for N = 0,1,2,3,4,5");
}
```

Variable elimination (VE) algorithm can speedup EFPs by finding efficient elimination ordering



$$= \left(\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\sum_{i_3=1}^{M} z_{i_1} z_{i_2} z_{i_3} \theta_{i_1 i_2}^2 \theta_{i_2 i_3}\right)\left(\sum_{i_4=1}^{M}\sum_{i_5=1}^{M} z_{i_4} z_{i_5} \theta_{i_4 i_5}^4\right)$$

Disconnected is product of connected

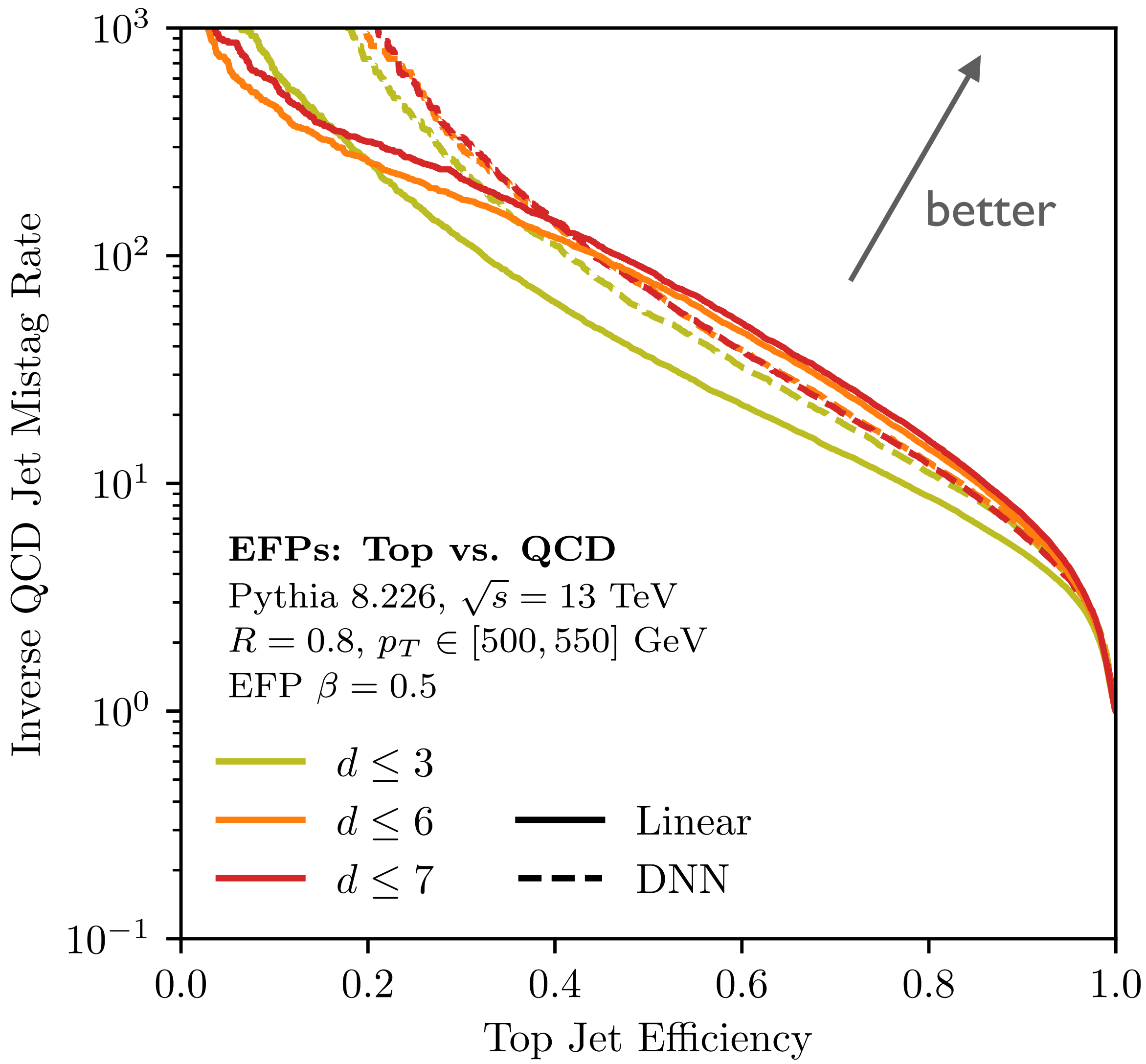
$$= \underbrace{\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\sum_{i_3=1}^{M}\sum_{i_4=1}^{M}\sum_{i_5=1}^{M}\sum_{i_6=1}^{M}\sum_{i_7=1}^{M}\sum_{i_8=1}^{M} z_{i_1} z_{i_2} z_{i_3} z_{i_4} z_{i_5} z_{i_6} z_{i_7} z_{i_8} \prod_{j=2}^{7} \theta_{i_1 i_j}}_{\mathcal{O}(M^8)}$$

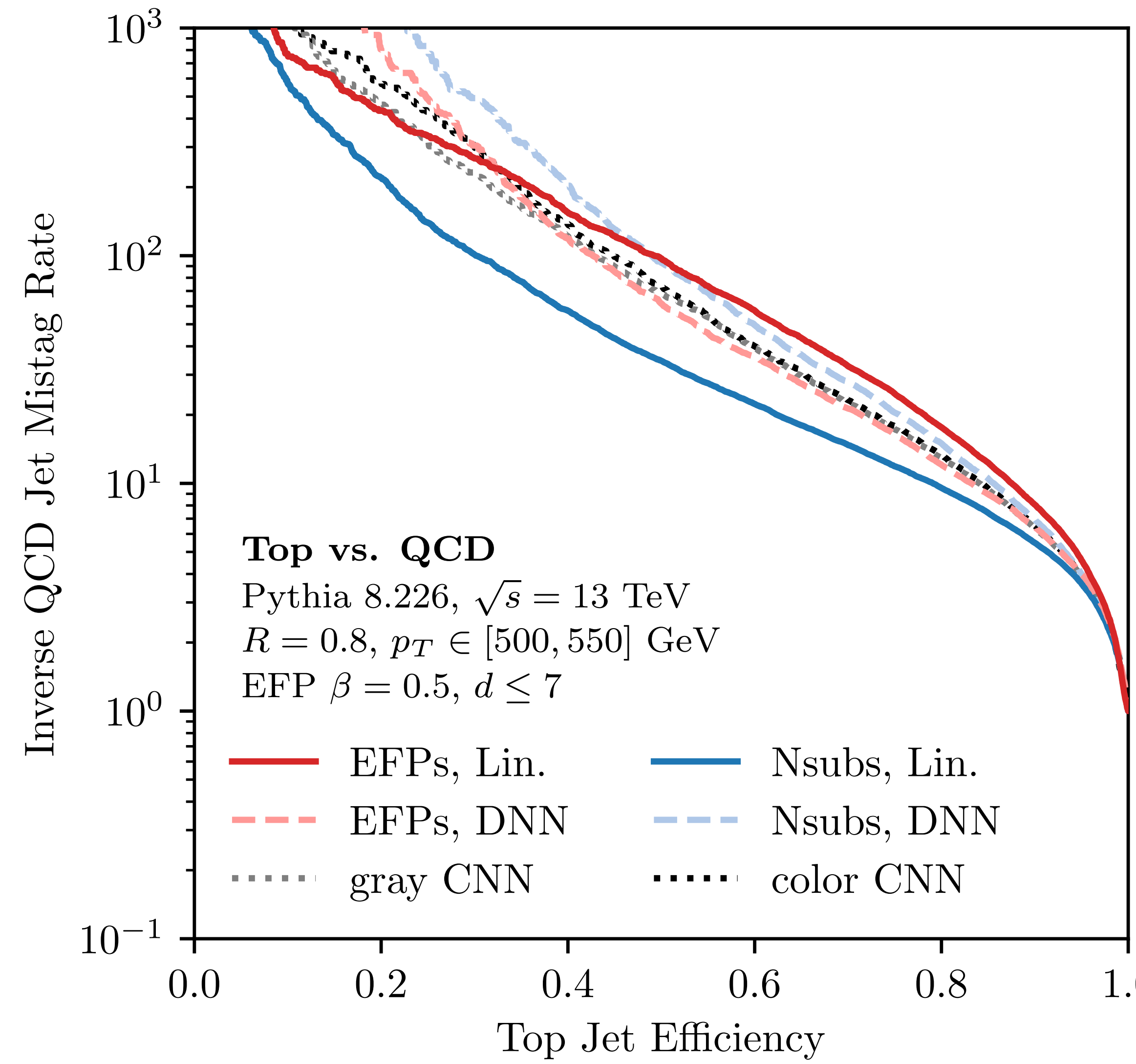
$$= \underbrace{\sum_{i_1=1}^{M} z_{i_1} \left(\sum_{i_2=1}^{M} z_{i_2} \theta_{i_1 i_2}\right)^7}_{\mathcal{O}(M^2)}$$

Clever parentheses placement corresponds to good elimination ordering

All tree graphs become $\mathcal{O}(M^2)$

EFPs for Boosted Tops





Saturation observed with more EFPs

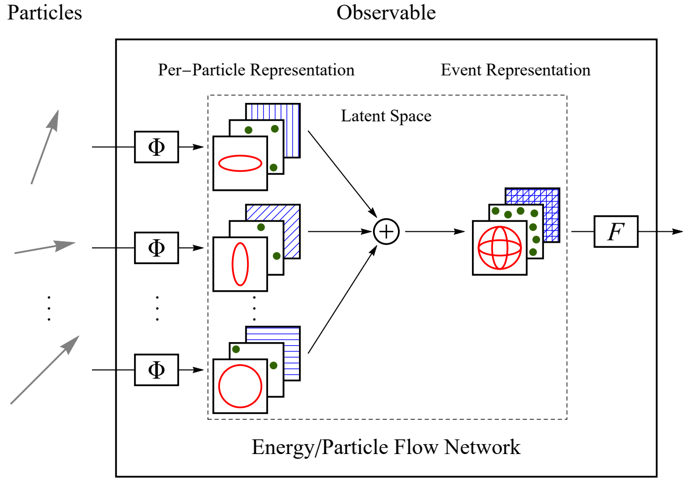
Linear EFPs excel at high efficiency

DNN gets there faster but linear suffices

[de Oliviera, Kagan, Mackey, Nachman, Schwartzman, 2015]
[PTK, Metodiev, Schwartz, 2016]
[Datta, Larkoski, 2017]

Jets as Point Clouds

Energy Flow Polynomials



Energy Flow Networks

Flexible/learnable point cloud processing

(EFNs for Q/G talk on Thursday @ ML4Jets!)

Symmetric Function Parametrization

A general permutation-symmetric function is *additive* in a latent space

Deep Sets: Namespace for additive symmetric function parametrization

Deep Sets

[1703.06114]

Manzil Zaheer[1,2], **Satwik Kottur**[1], **Siamak Ravanbhakhsh**[1],
Barnabás Póczos[1], **Ruslan Salakhutdinov**[1], **Alexander J Smola**[1,2]
[1] Carnegie Mellon University [2] Amazon Web Services

Deep Sets Theorem [63]. *Let $\mathfrak{X} \subset \mathbb{R}^d$ be compact, $X \subset 2^{\mathfrak{X}}$ be the space of sets with bounded cardinality of elements in $\mathfrak{X}$, and $Y \subset \mathbb{R}$ be a bounded interval. Consider a continuous function $f : X \to Y$ that is invariant under permutations of its inputs, i.e. $f(x_1, \ldots, x_M) = f(x_{\pi(1)}, \ldots, x_{\pi(M)})$ for all $x_i \in \mathfrak{X}$ and $\pi \in S_M$. Then there exists a sufficiently large integer ℓ and continuous functions $\Phi : \mathfrak{X} \to \mathbb{R}^\ell$, $F : \mathbb{R}^\ell \to Y$ such that the following holds to an arbitrarily good approximation:*[1]

$$f(\{x_1, \ldots, x_M\}) = F\left(\sum_{i=1}^{M} \Phi(x_i)\right). \tag{2.1}$$

Symmetric Function Parametrization

A general permutation-symmetric function is *additive* in a latent space

Deep Sets: Namespace for additive symmetric function parametrization

Deep Sets

[1703.06114]

Manzil Zaheer[1,2], Satwik Kottur[1], Siamak Ravanbhakhsh[1], Barnabás Póczos[1], Ruslan Salakhutdinov[1], Alexander J Smola[1,2]
[1] Carnegie Mellon University [2] Amazon Web Services

Variable length

Feature space

Deep Sets Theorem [63]. *Let $\mathfrak{X} \subset \mathbb{R}^d$ be compact, $X \subset 2^{\mathfrak{X}}$ be the space of sets with bounded cardinality of elements in $\mathfrak{X}$, and $Y \subset \mathbb{R}$ be a bounded interval. Consider a continuous function $f : X \to Y$ that is invariant under permutations of its inputs, i.e. $f(x_1, \ldots, x_M) = f(x_{\pi(1)}, \ldots, x_{\pi(M)})$ for all $x_i \in \mathfrak{X}$ and $\pi \in S_M$. Then there exists a sufficiently large integer ℓ and continuous functions $\Phi : \mathfrak{X} \to \mathbb{R}^\ell$, $F : \mathbb{R}^\ell \to Y$ such that the following holds to an arbitrarily good approximation:*[1]

Permutation invariance

Latent space

$$f(\{x_1, \ldots, x_M\}) = F\left(\sum_{i=1}^{M} \Phi(x_i)\right). \tag{2.1}$$

General parametrization for a function of sets

Deep Sets for Particle Jets

[PTK, Metodiev, Thaler, 1810.05165]

Particle Flow Network (PFN)

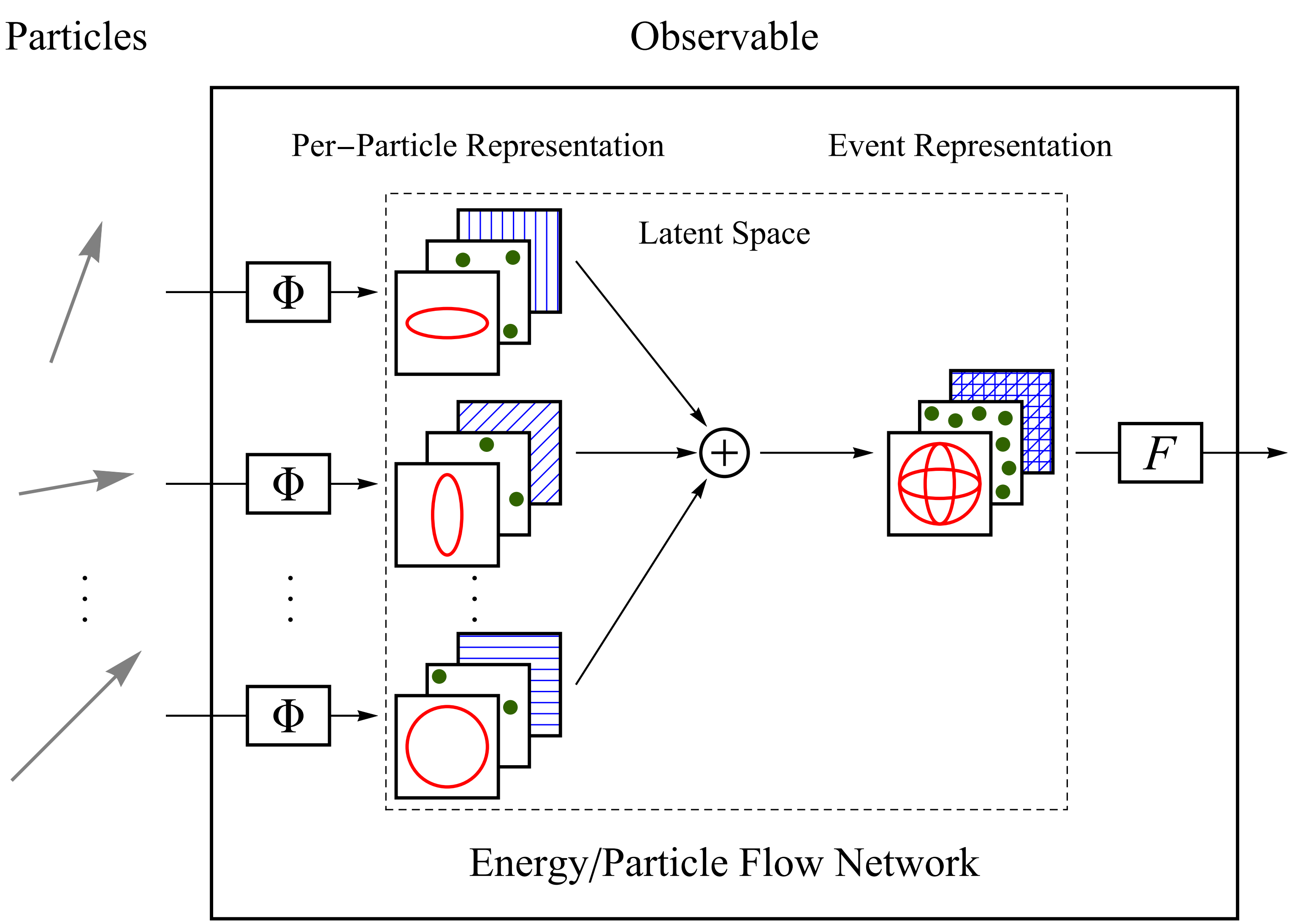
$$\text{PFN}(\{p_1^\mu,\ldots,p_M^\mu\}) = F\left(\sum_{i=1}^{M}\Phi(p_i^\mu)\right)$$

Fully general latent space

Energy Flow Network (EFN)

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

IRC-safe latent space

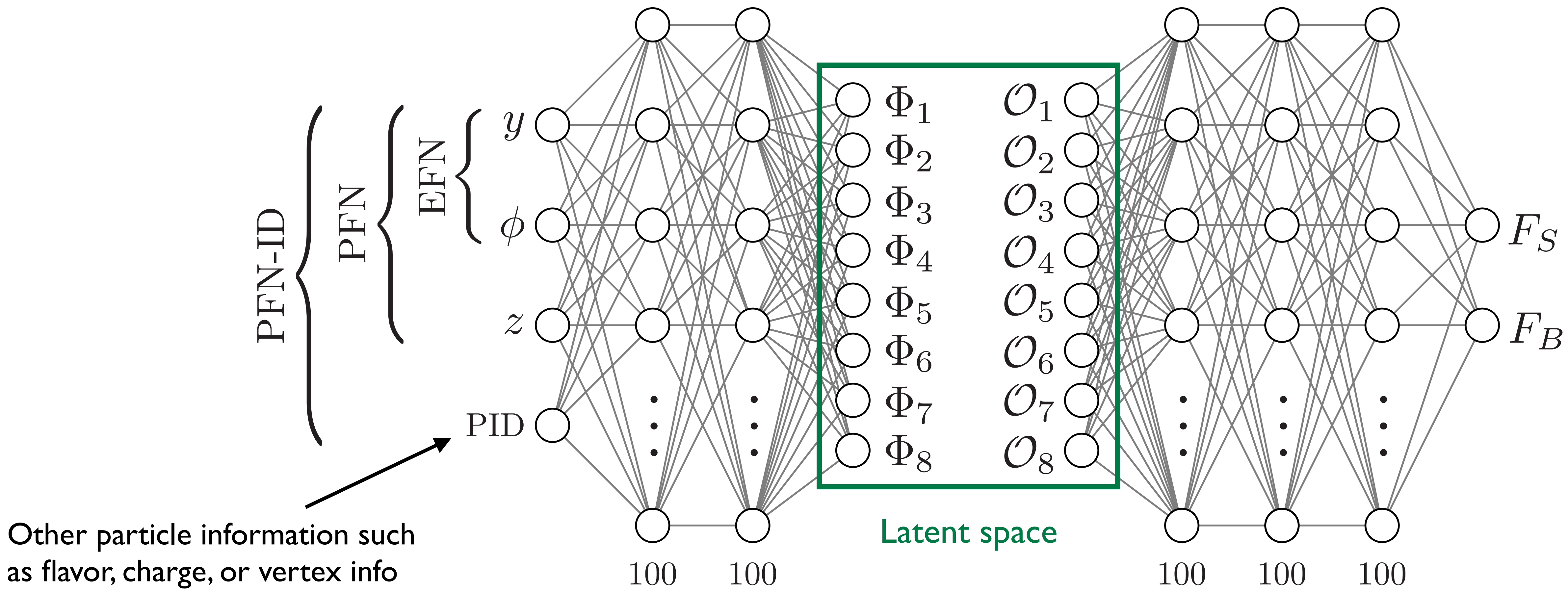


Approximating Φ and *F* with Neural Networks

Employ neural networks as arbitrary function approximators

Use fully-connected networks for simplicity

Default sizes – Φ: (100, 100, ℓ), *F*: (100, 100, 100)



$$\text{PFN}: \quad \mathcal{O}_a = \sum_{i=1}^{M} \Phi_a(z_i, y_i, \phi_i, [\text{PID}_i]) \qquad \text{EFN}: \quad \mathcal{O}_a = \sum_{i=1}^{M} z_i \Phi_a(y_i, \phi_i)$$

Top Jet Samples and Other Methods

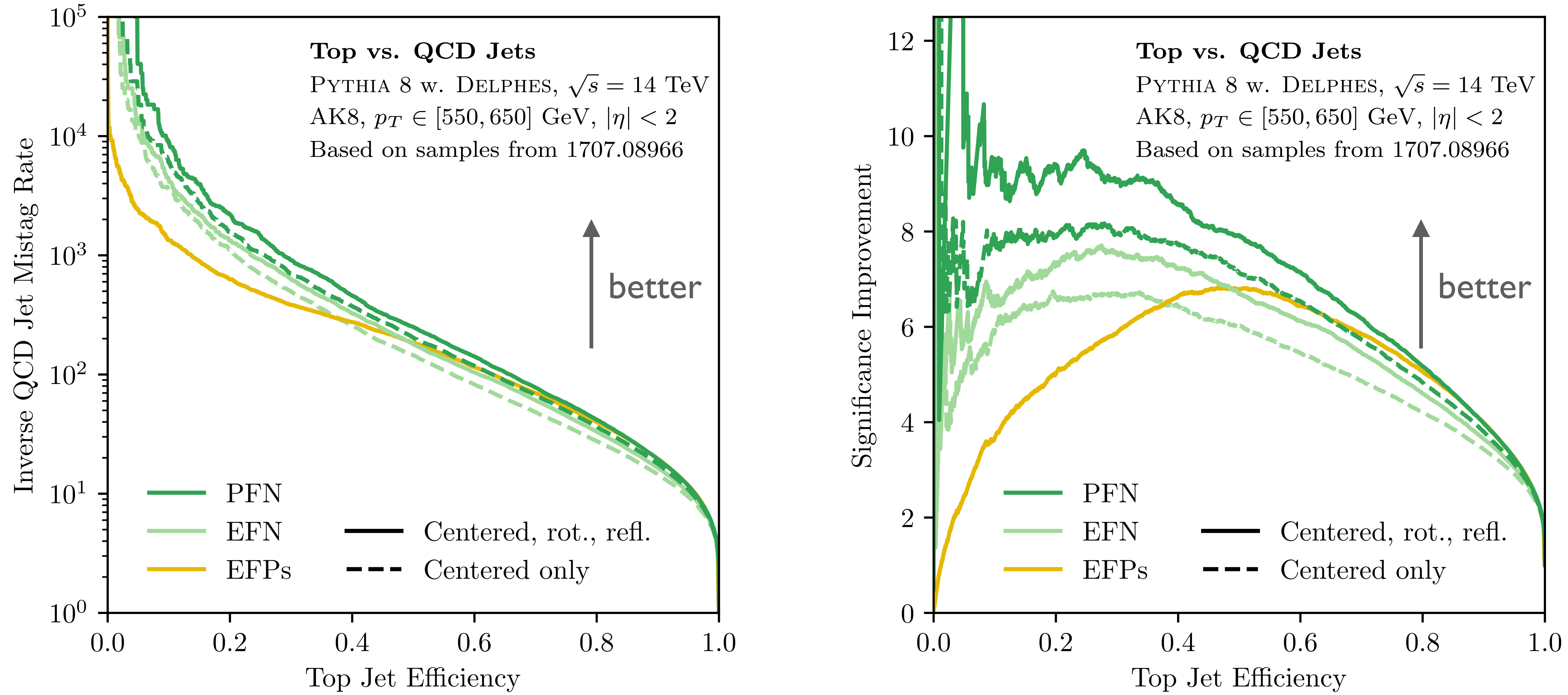
[Butter, Kasieczka, Plehn, Russell, 2017]

Common top and QCD dijet samples for standardized benchmarking

$p_T \in [550, 650]$ GeV, AK8 jets, fully-merged, Delphes simulation, 2m jets total

Approach	AUC	Acc.	1/eB @ (eS=0.3)	Contact	Comments
LoLa	0.979	0.928		G. Kasieczka S. Leiss	Preliminary number, based on LoLa
LBN	0.981	0.931	863	M. Rieger	Preliminary number
CNN	0.981	0.93	780	D. Shih	Model from *(1803.00107)*
P-CNN (1D CNN)	0.980	0.930	782	H. Qu, L. Gouskos	Preliminary, use kinematic info only
6-body N-subs. (+mass and pT) NN	0.979	0.922	856	K. Nordstrom	Based on 1807.04769
8-body N-subs. (+mass and pT) NN	0.980	0.928	795	K. Nordstrom	Based on 1807.04769
Linear EFPs	0.980	0.932	380	PTK, E. Metodiev	d<= 7, chi <= 3 EFPs with FLD. Based on 1712.07124
Particle Flow Network (PFN)	0.982	0.932	888	PTK, E. Metodiev	Median over ten trainings. Based on Table 5 in 1810.05165
Energy Flow Network (EFN)	0.979	0.927	619	PTK, E. Metodiev	Median over ten trainings. Based on Table 5 in 1810.05165

Classification Performance

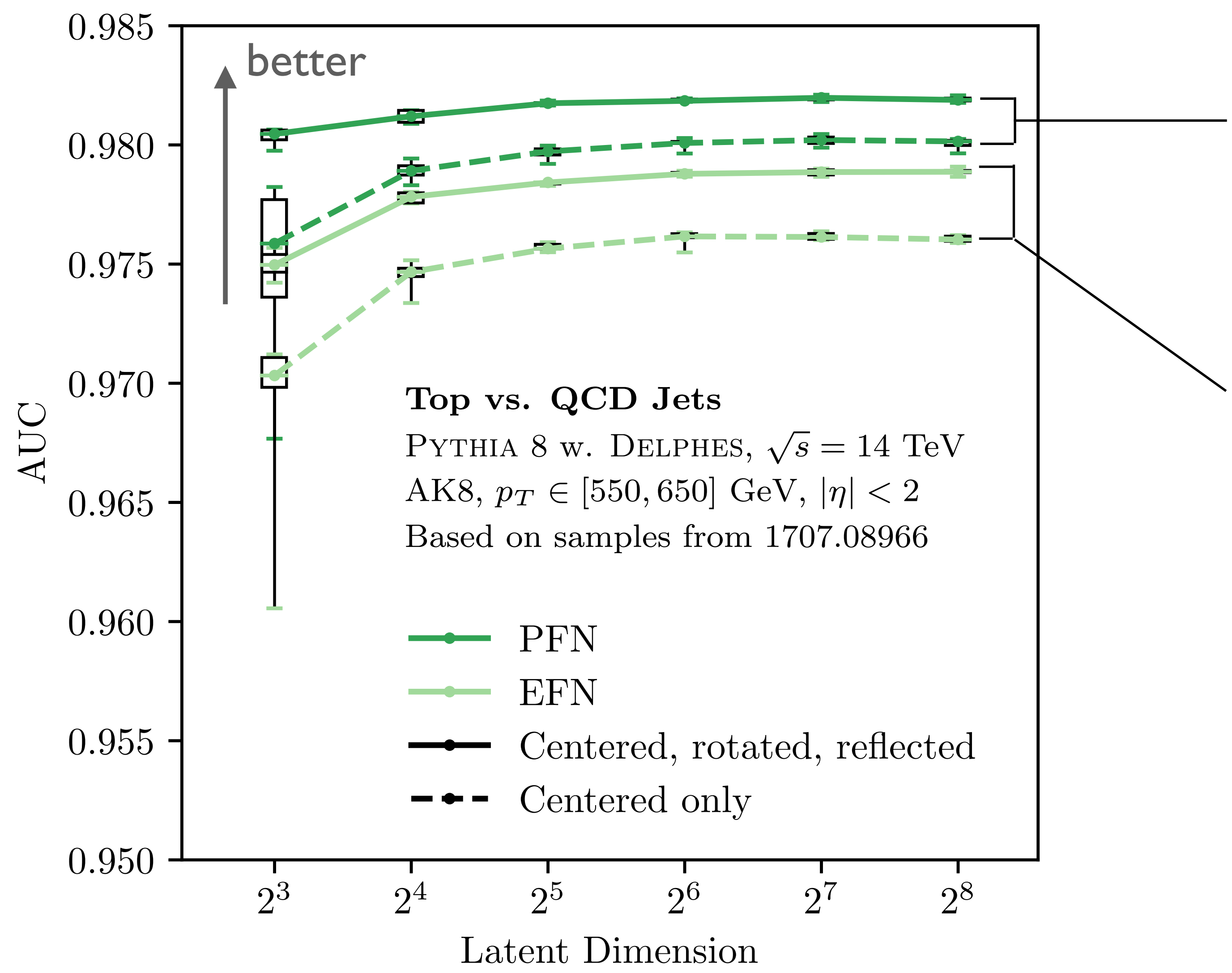


Latent space dimension $\ell = 256$

EFN/PFN rotation and reflection preprocessing helpful

EFPs are comparable to EFN and even better at high signal efficiency

EFN Latent Dimension Sweep



Preprocessing clearly helpful

IRC unsafe information clearly helpful

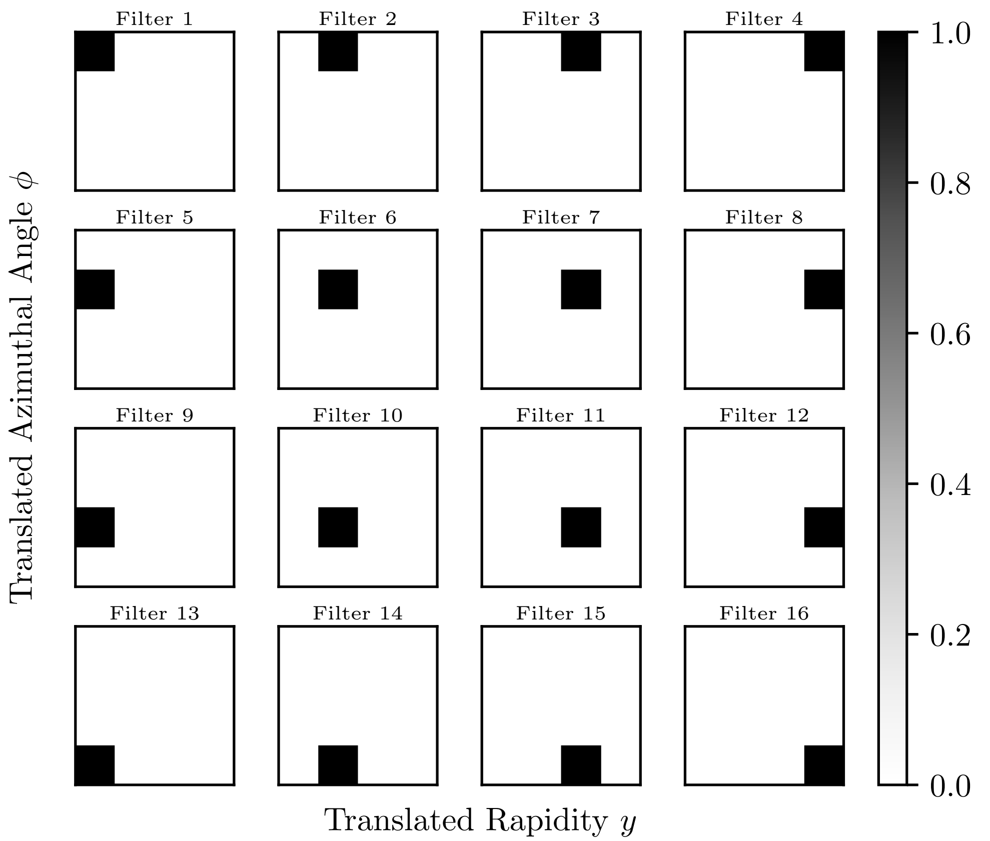
Performance quickly saturates as latent dimension increases

Energy Flow Network Visualization

EFN observables are two-dimensional geometric functions

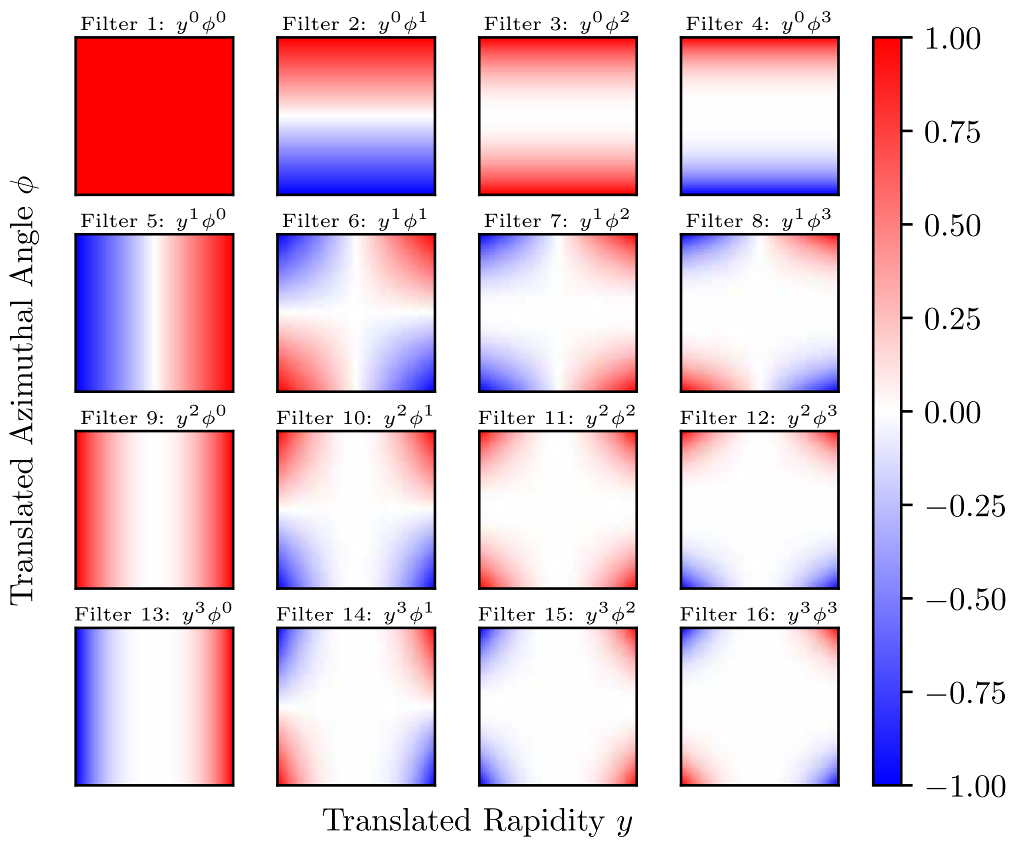
Visualize EFN observables as *filters* in the translated rapidity-azimuth plane

Jet images as EFN filters



[Cogan, Kagan, Strauss, Schwartzman, 2014]
[de Oliviera, Kagan, Mackey, Nachman, Schwartzman, 2015]

Moments as EFN filters

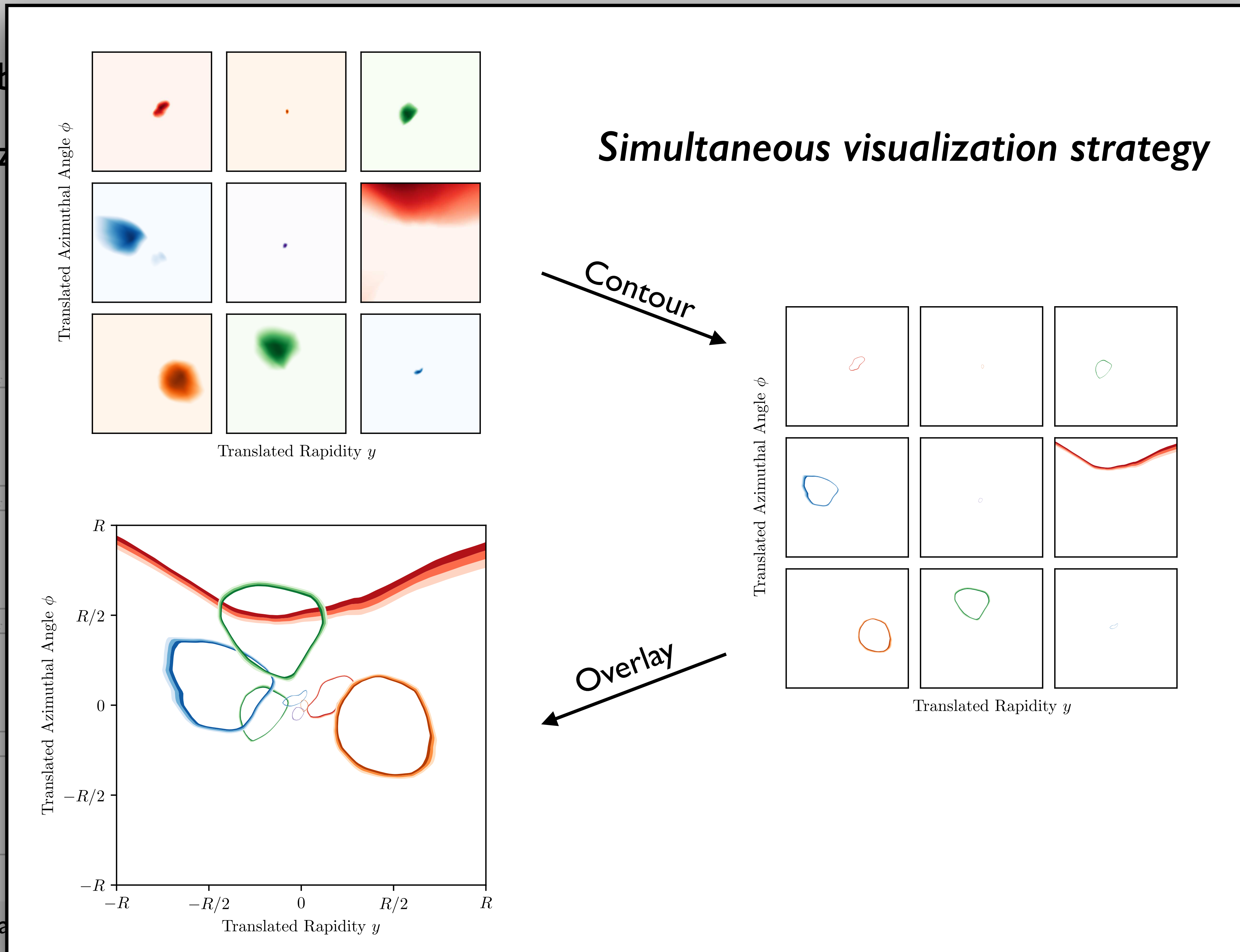


[Donoghue, Low, Pi, 1979]
[Gur-Ari, Papucci, Perez, 2011]

Energy Flow Network Visualization

EFN ob

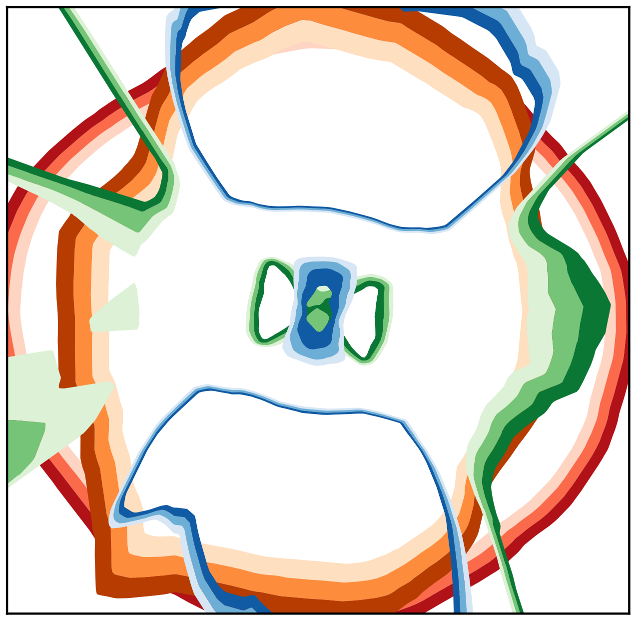
Visualiz



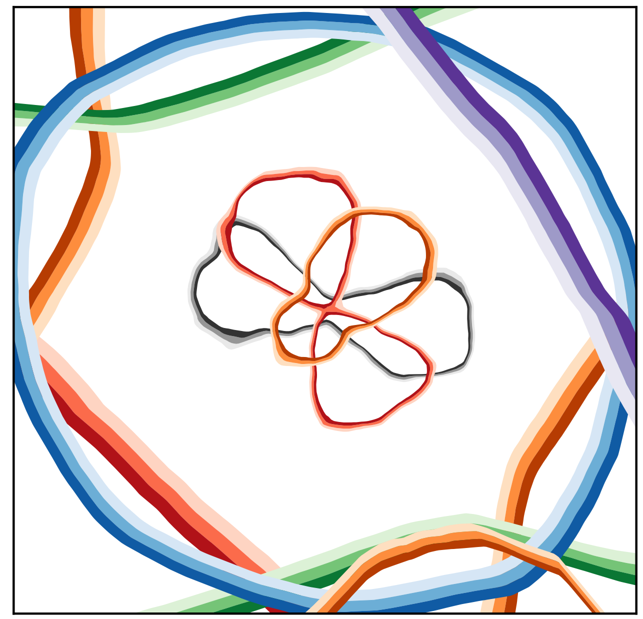
[Cogan, Ka … v, Pi, 1979]
[de Olivier … erez, 2011]

Visualizing EFN Filters

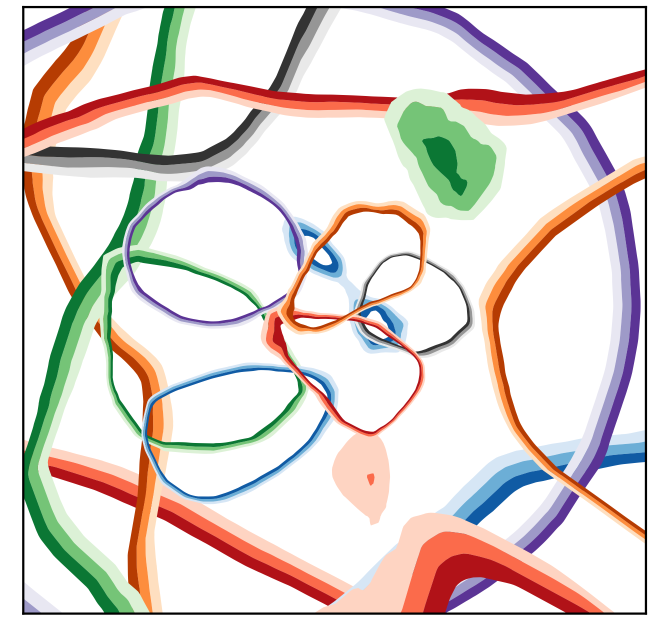
Without rotation/reflection preprocessing



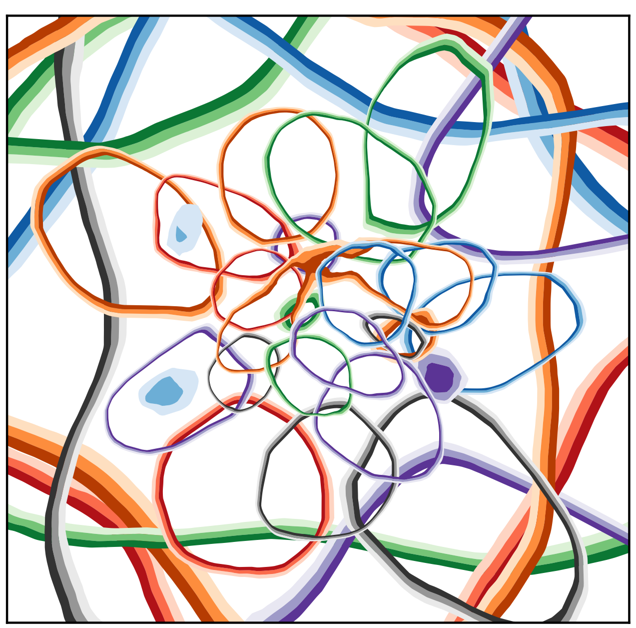
$\ell = 4$



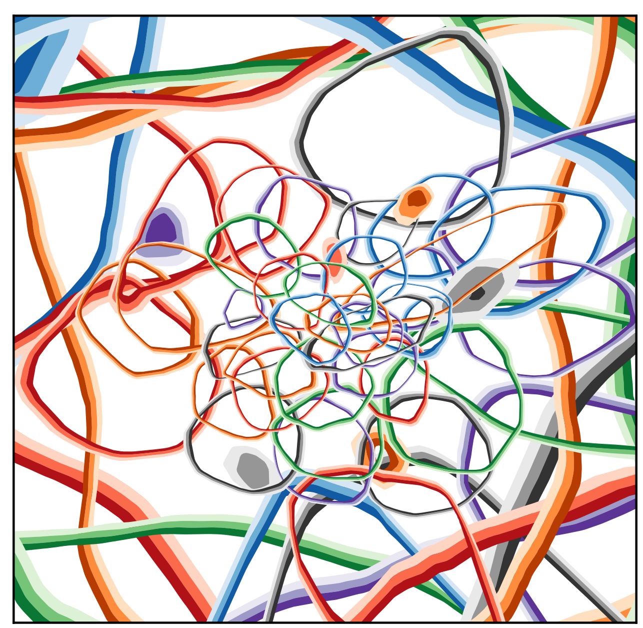
$\ell = 8$



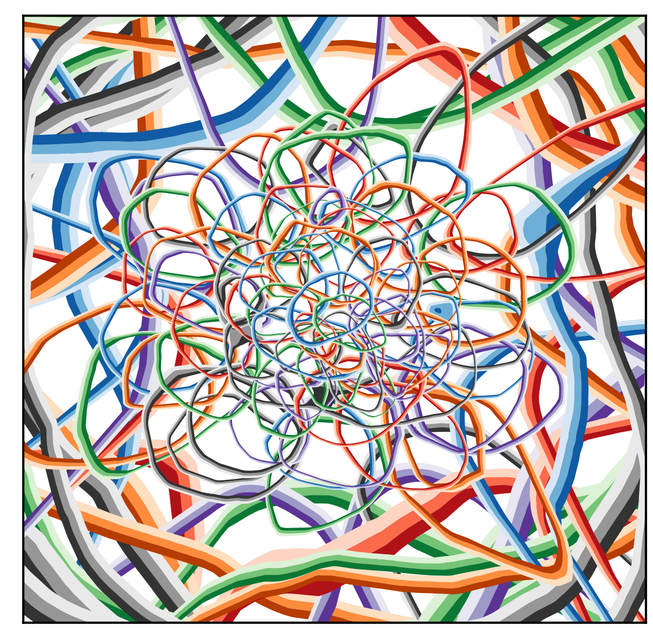
$\ell = 16$



$\ell = 32$



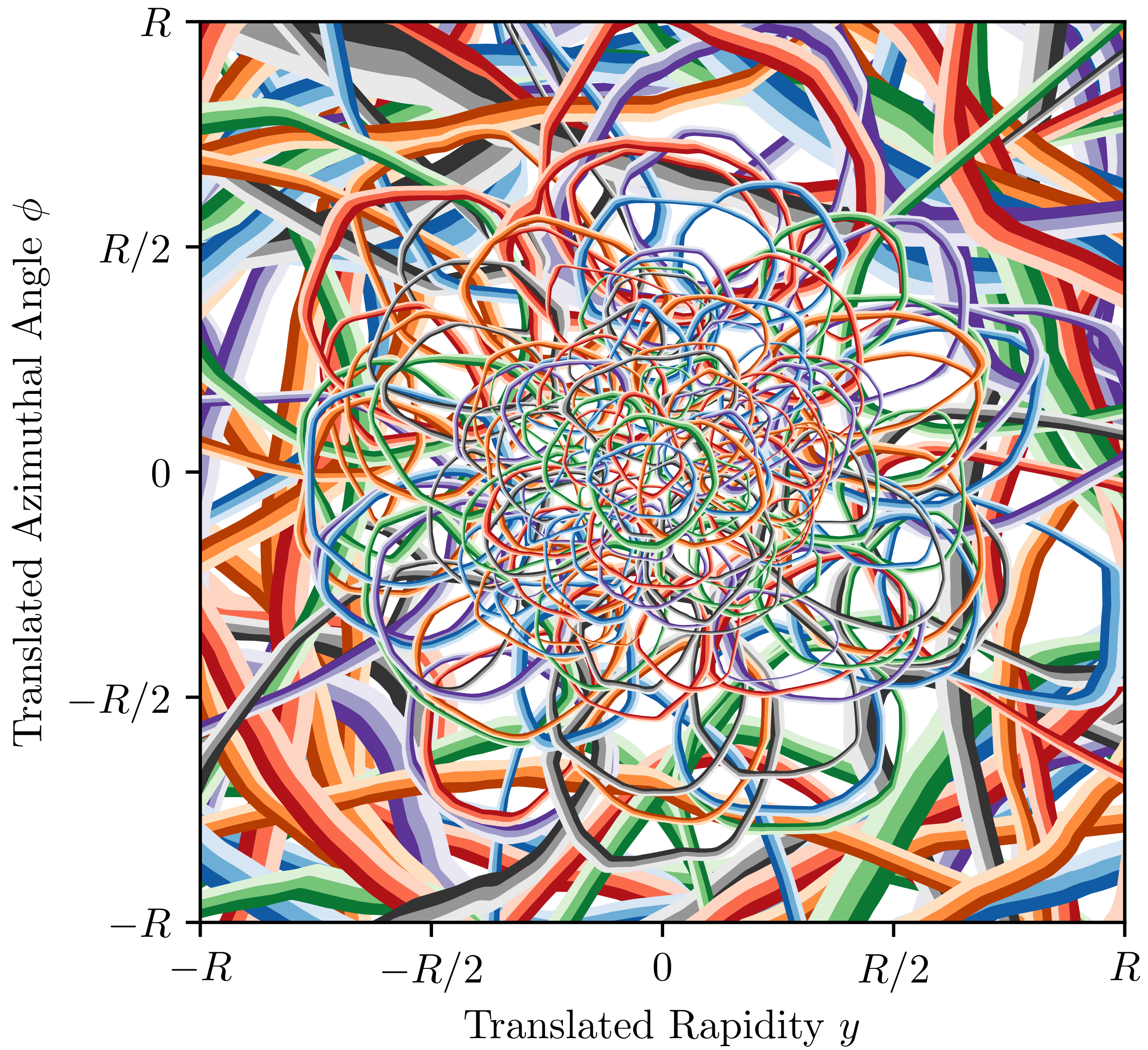
$\ell = 64$



$\ell = 128$

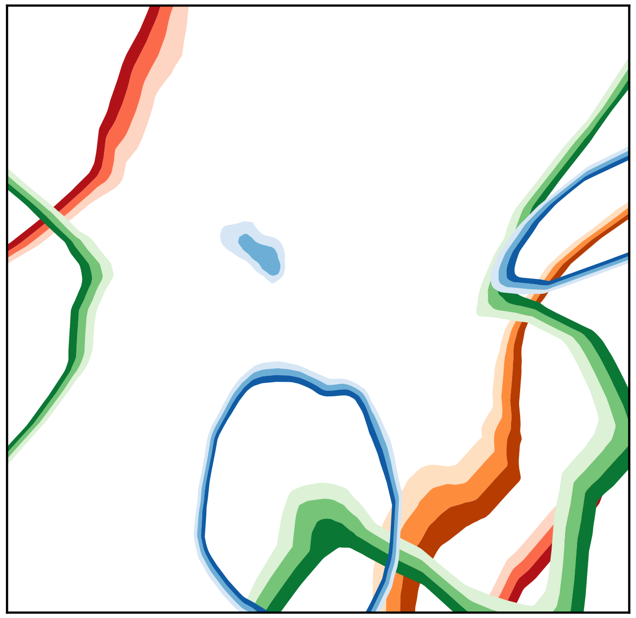
Visualizing EFN Filters

Without rotation/reflection preprocessing

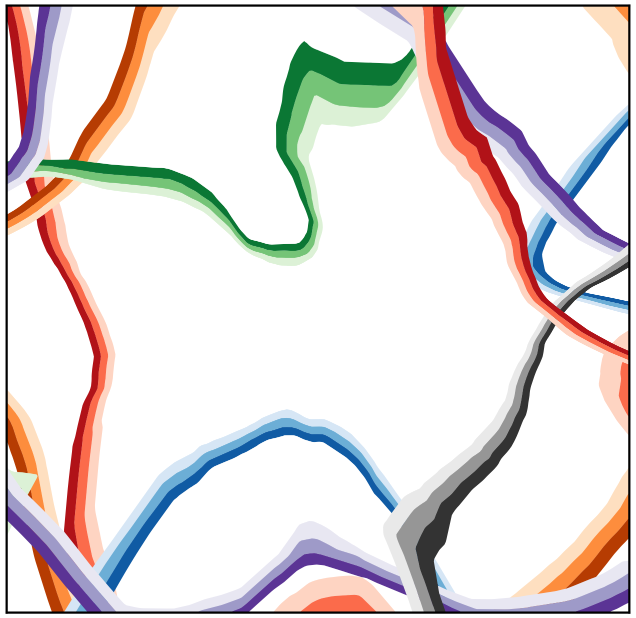


Visualizing EFN Filters

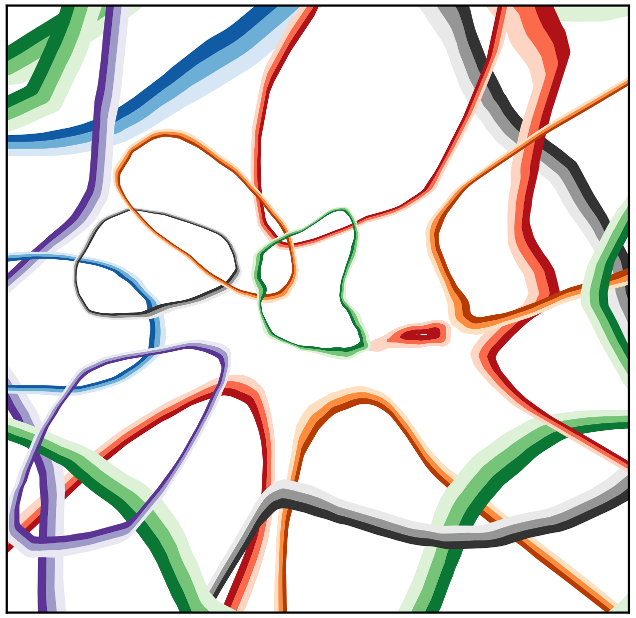
With rotation/reflection preprocessing



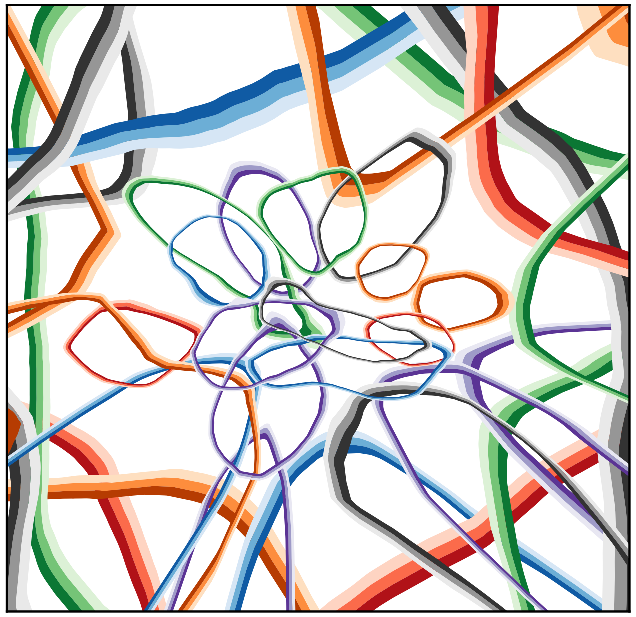
$\ell = 4$



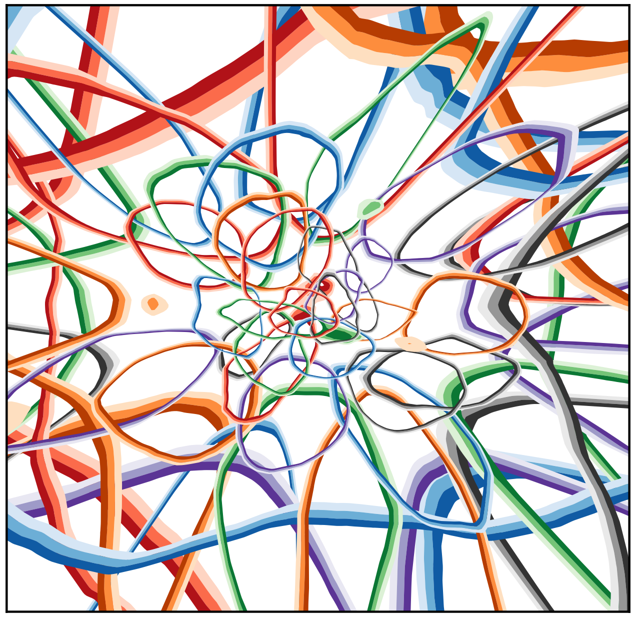
$\ell = 8$



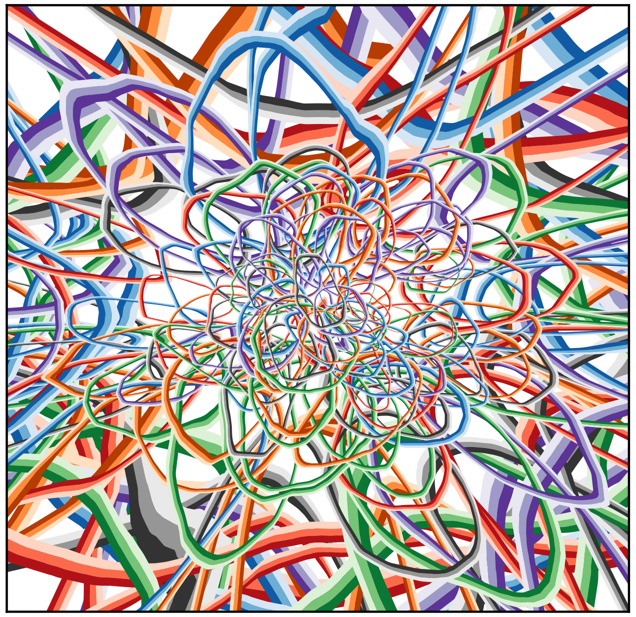
$\ell = 16$



$\ell = 32$



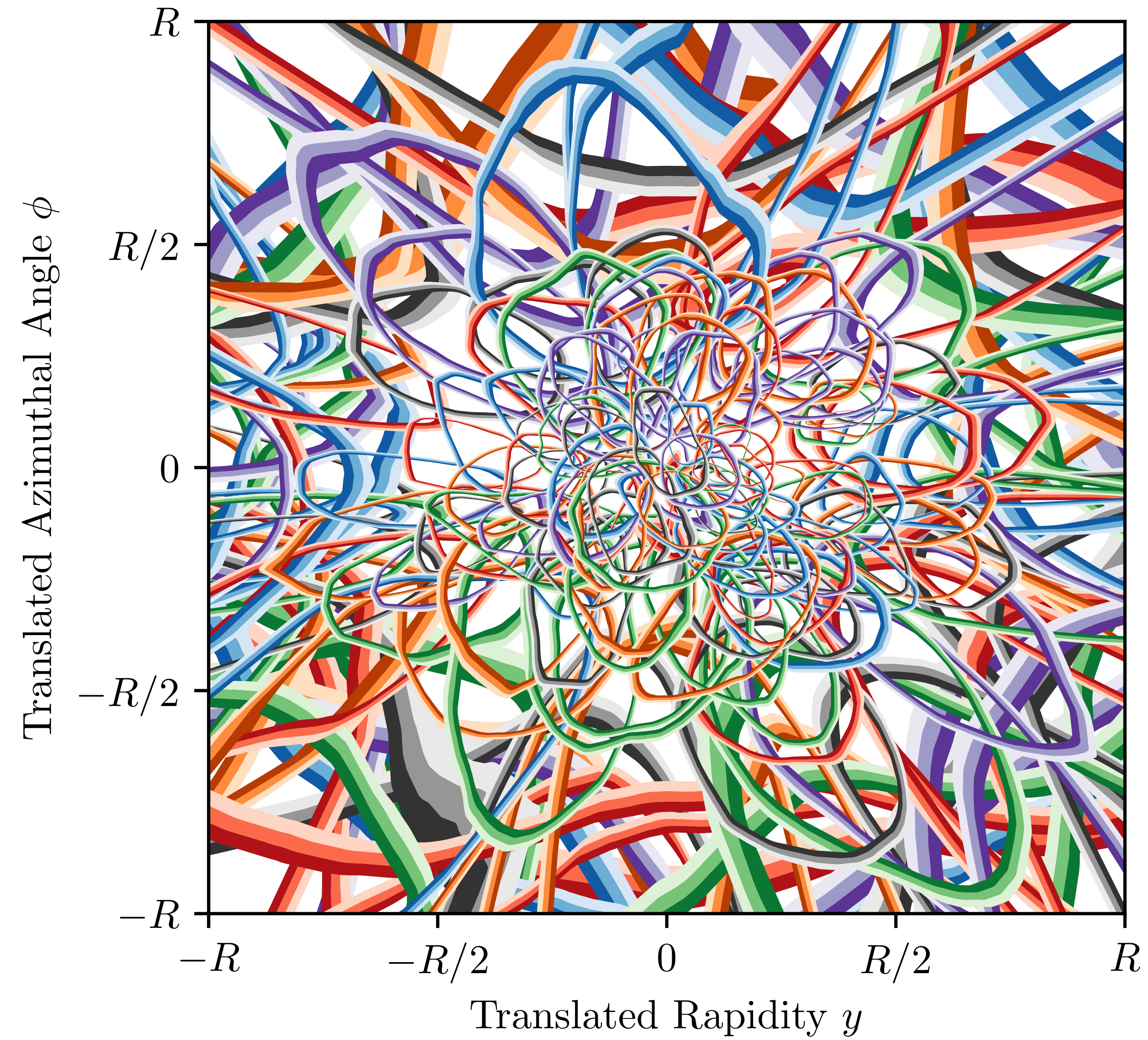
$\ell = 64$

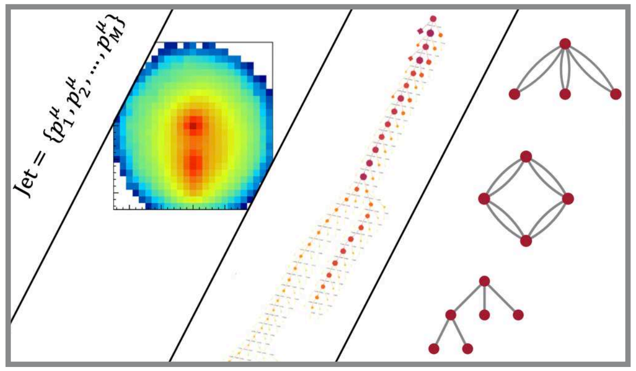


$\ell = 128$

Visualizing EFN Filters

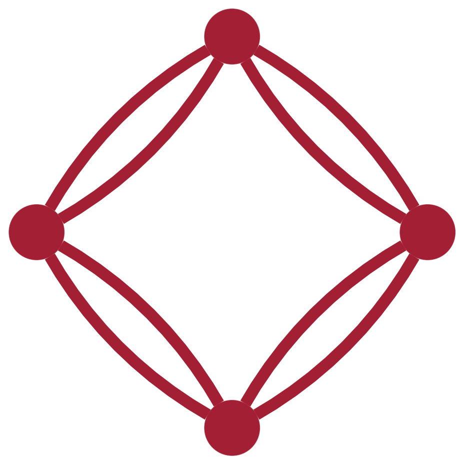
Without rotation/reflection preprocessing





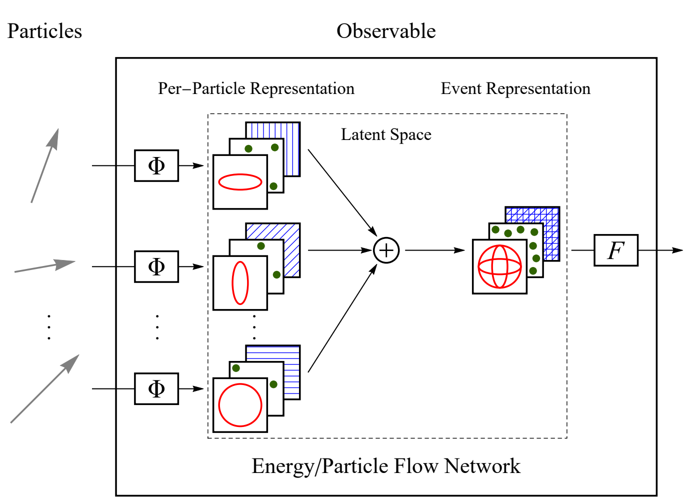
Jets as Point Clouds

Jets have the same symmetries as point clouds
Respecting symmetries key for maximal performance



Energy Flow Polynomials

Linear basis of IRC-safe observables
Incredibly simple architecture competes with modern ML



Energy Flow Networks

Excellent performance, fascinating visualizations via IRC safety

(EFNs for Q/G talk on Thursday @ ML4Jets!)

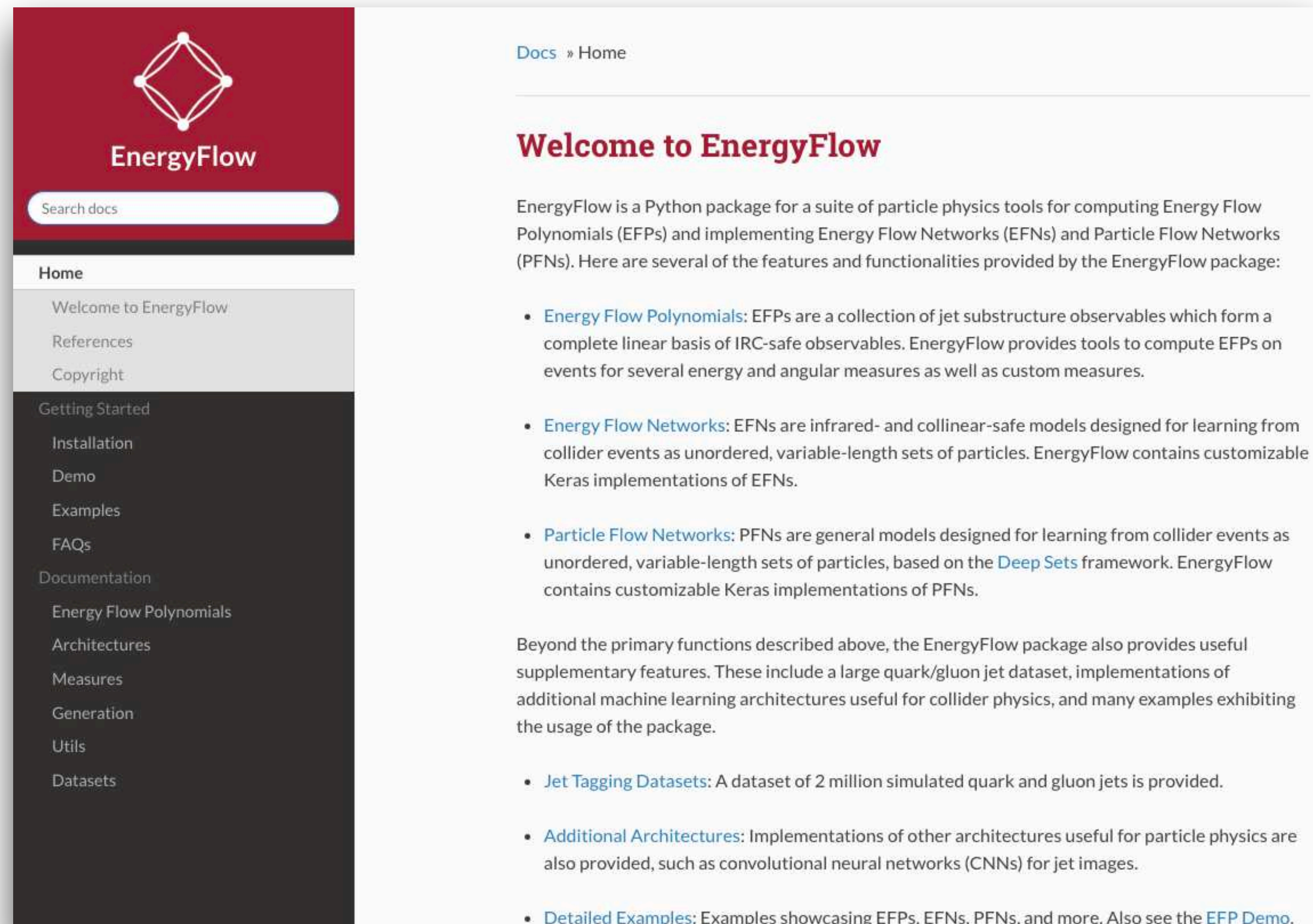
EnergyFlow Python Package

Implements variable elimination for efficient EFP computation

Contains EFN and PFN implementations in Keras

CNN, DNN architectures included for easy model comparison

Several detailed examples demonstrating how to train models and make visualizations



EnergyFlow

Search docs

Home

Welcome to EnergyFlow

References

Copyright

Getting Started

Installation

Demo

Examples

FAQs

Documentation

Energy Flow Polynomials

Architectures

Measures

Generation

Utils

Datasets

Docs » Home

Welcome to EnergyFlow

EnergyFlow is a Python package for a suite of particle physics tools for computing Energy Flow Polynomials (EFPs) and implementing Energy Flow Networks (EFNs) and Particle Flow Networks (PFNs). Here are several of the features and functionalities provided by the EnergyFlow package:

- Energy Flow Polynomials: EFPs are a collection of jet substructure observables which form a complete linear basis of IRC-safe observables. EnergyFlow provides tools to compute EFPs on events for several energy and angular measures as well as custom measures.
- Energy Flow Networks: EFNs are infrared- and collinear-safe models designed for learning from collider events as unordered, variable-length sets of particles. EnergyFlow contains customizable Keras implementations of EFNs.
- Particle Flow Networks: PFNs are general models designed for learning from collider events as unordered, variable-length sets of particles, based on the Deep Sets framework. EnergyFlow contains customizable Keras implementations of PFNs.

Beyond the primary functions described above, the EnergyFlow package also provides useful supplementary features. These include a large quark/gluon jet dataset, implementations of additional machine learning architectures useful for collider physics, and many examples exhibiting the usage of the package.

- Jet Tagging Datasets: A dataset of 2 million simulated quark and gluon jets is provided.
- Additional Architectures: Implementations of other architectures useful for particle physics are also provided, such as convolutional neural networks (CNNs) for jet images.
- Detailed Examples: Examples showcasing EFPs, EFNs, PFNs, and more. Also see the EFP Demo.

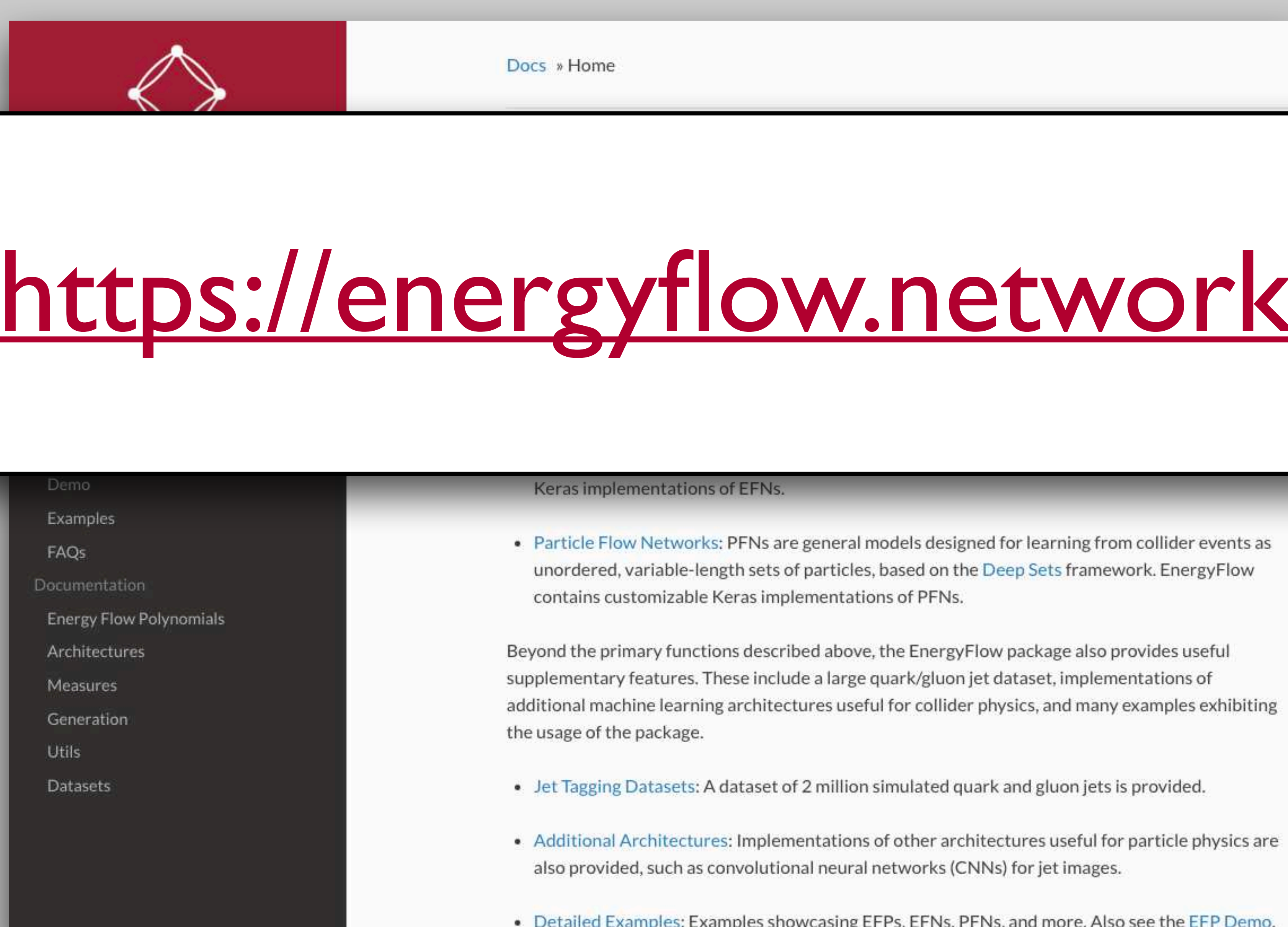
EnergyFlow Python Package

Implements variable elimination for efficient EFP computation

Contains EFN and PFN implementations in Keras

CNN, DNN architectures included for easy model comparison

Several detailed examples demonstrating how to train models and make visualizations


Docs » Home

Demo
Examples
FAQs
Documentation
Energy Flow Polynomials
Architectures
Measures
Generation
Utils
Datasets

Keras implementations of EFNs.

- Particle Flow Networks: PFNs are general models designed for learning from collider events as unordered, variable-length sets of particles, based on the Deep Sets framework. EnergyFlow contains customizable Keras implementations of PFNs.

Beyond the primary functions described above, the EnergyFlow package also provides useful supplementary features. These include a large quark/gluon jet dataset, implementations of additional machine learning architectures useful for collider physics, and many examples exhibiting the usage of the package.

- Jet Tagging Datasets: A dataset of 2 million simulated quark and gluon jets is provided.
- Additional Architectures: Implementations of other architectures useful for particle physics are also provided, such as convolutional neural networks (CNNs) for jet images.
- Detailed Examples: Examples showcasing EFPs, EFNs, PFNs, and more. Also see the EFP Demo.

https://energyflow.network

Thank You!

Classification Performance