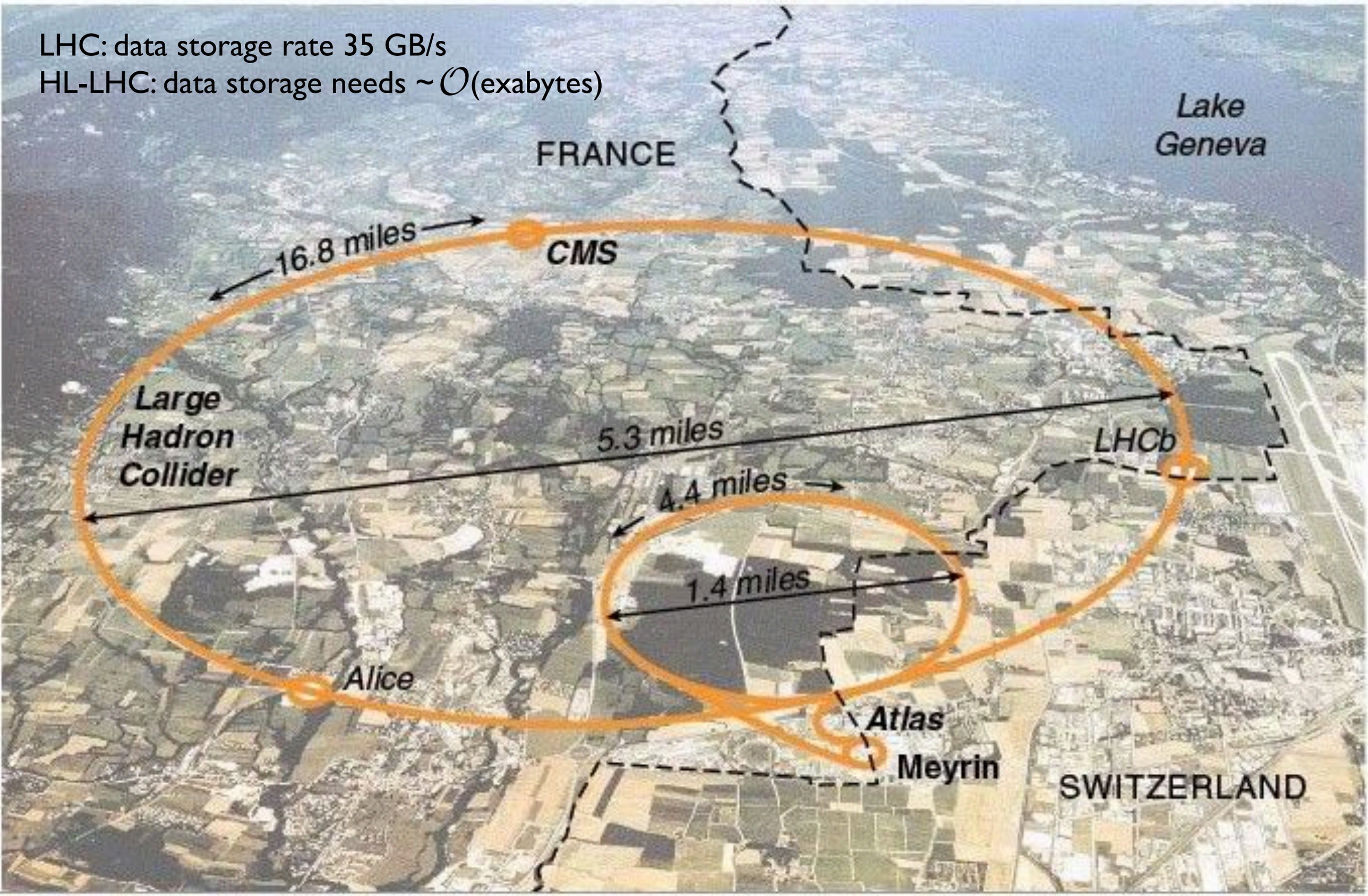
New Computational Techniques and Machine Learning for Jet Physics

Sebastian Macaluso
New York University

Instituto de Fisica La Plata
UNLP
September 21, 2020

The Large Hadron Collider

LHC: data storage rate 35 GB/s
HL-LHC: data storage needs ~ $\mathcal{O}$(exabytes)



JETS

Collimated spray of energetic charged and neutral hadrons

ATLAS EXPERIMENT

Run: 282712
Event: 474587238
2015-10-21 06:26:57 CEST

Jet Classification: Top vs QCD jets

Top quark Phenomenology

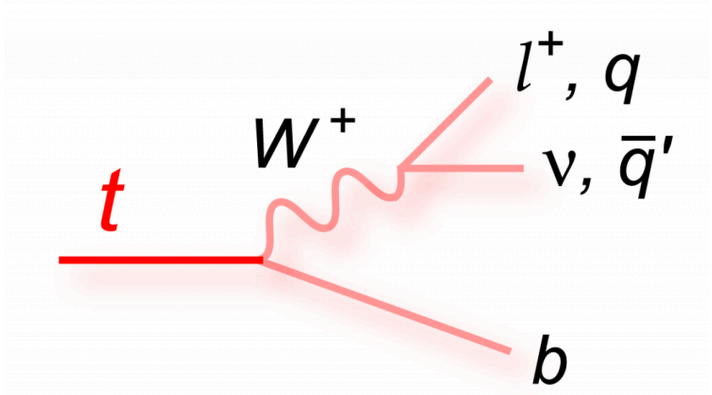


Fig. from www.quantumdiaries.org

- Leptonic decay → *b* quark jet+isolated lepton+MET.
- Hadronic decay → 3 jets.

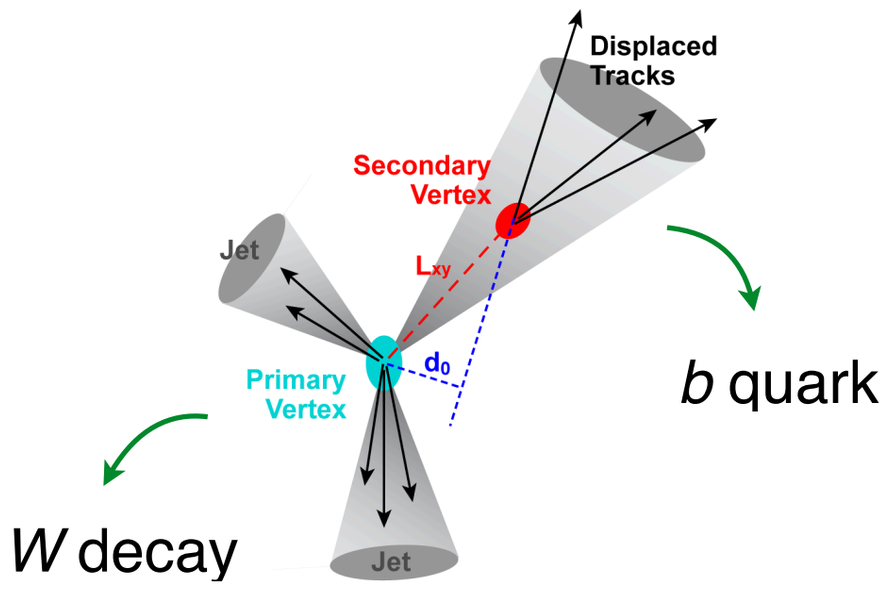


Fig. from www.amva4newphysics.wordpress.com

Hadronic tops are reconstructed as 3 jets at the CMS/ATLAS detectors.

Boosted top and QCD jets

- Collimated decay products of boosted particles are reconstructed as a “fat jet”.

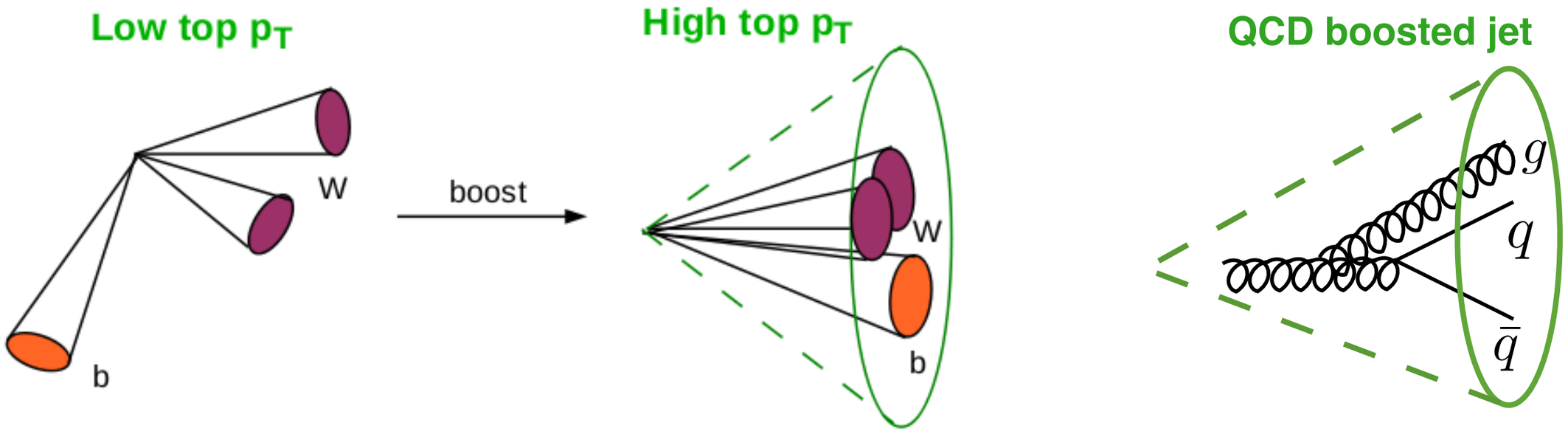


Fig. from www.quantumdiaries.org

- Typical classifiers use physical observables, e.g. jet mass, b-tagging, etc

- Apply them on a cut and count analysis or as high-level inputs (e.g. jet mass) of multivariate machine learning algorithms (BDTs).

Neural Networks

NN's can be seen as a very flexible family of functions parameterized by the weights ω and biases b.

Universality theorem: NNs can be used to approximate any continuous function to any desired precision.

Vanilla MLP

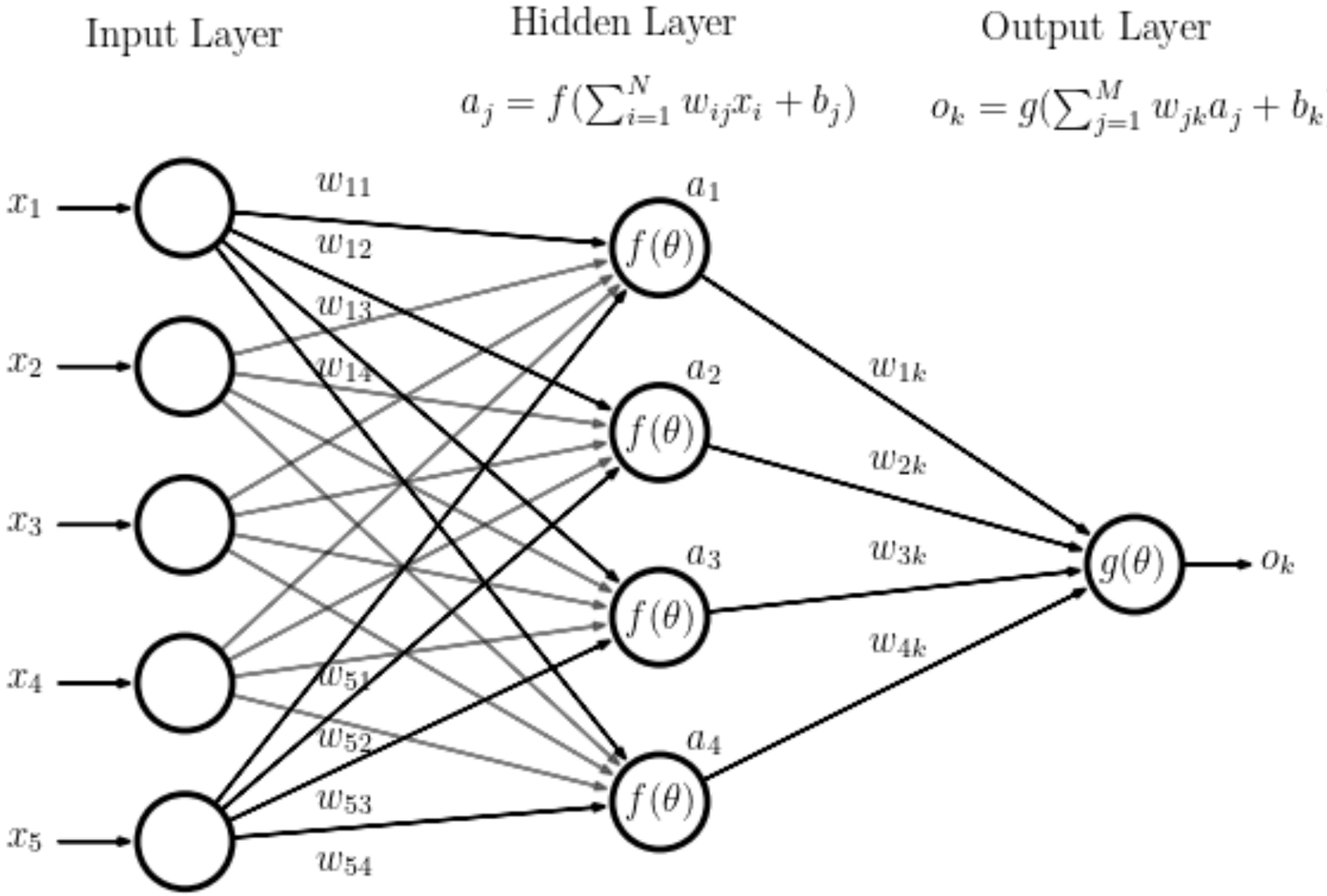
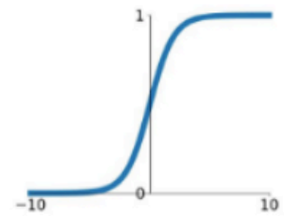


Fig. from www.astroml.org

Activation Functions

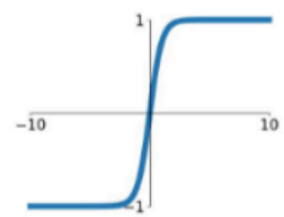
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



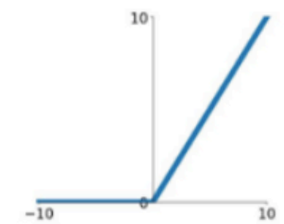
tanh

$\tanh(x)$



ReLU

$\max(0, x)$



Softmax (final layer)

$$g(\theta)_j = \frac{e^{\theta_j}}{\sum_k e^{\theta_k}}$$

Loss Function

$$L = \sum_i f(a(\omega, x_i), y_i)$$

Mean-squared-error (MSE)

$$f(a, y) = (a - y)^2$$

Cross-entropy

$$f(a, y) = -y \log a - (1 - y) \log(1 - a)$$

Learning Process

- Pass inputs x_i to first layer.
- Forward pass: calculate the outputs a_i until the final layer.
- Evaluate the loss function L .
- Backpropagation: find the gradient of L with respect to the weights ω_i .
- Update weights:

$$\omega_{t+1} = \omega_t - \eta \frac{\partial L}{\partial \omega_t}$$

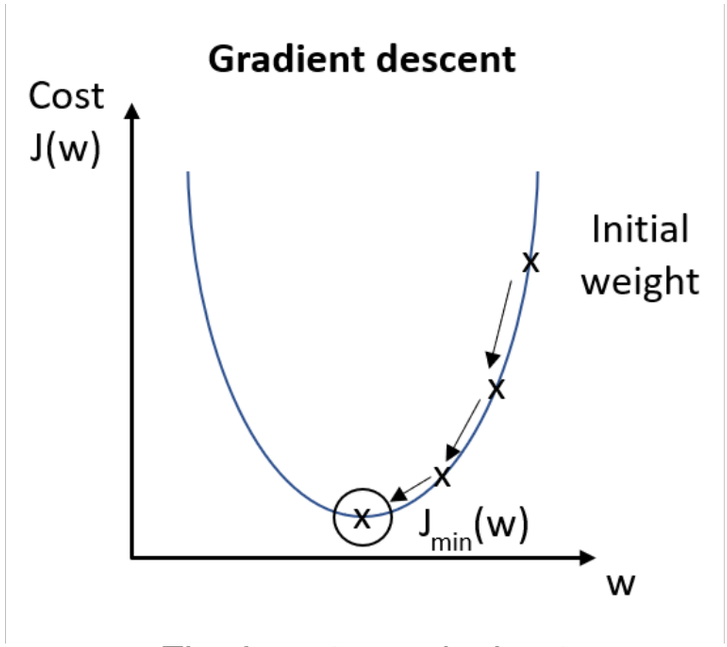


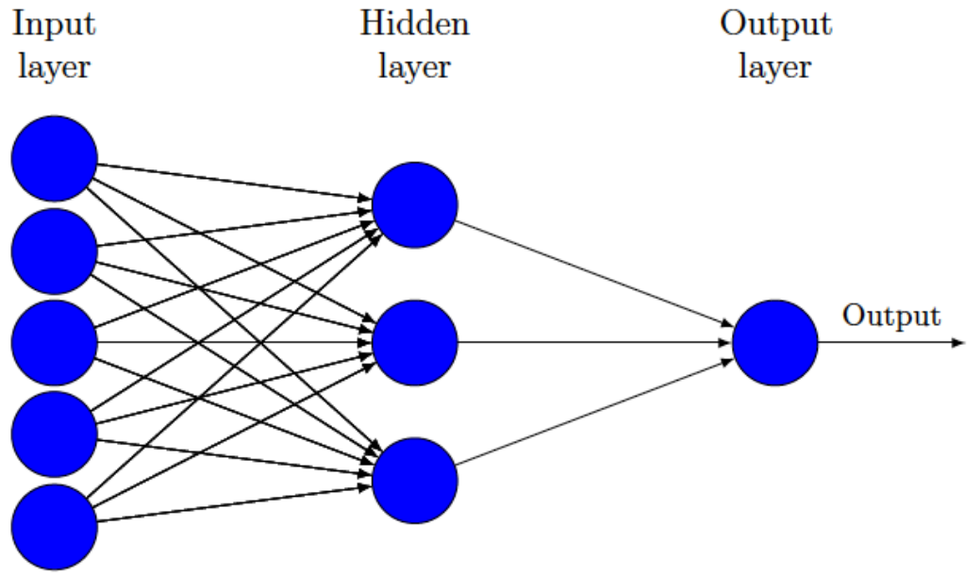
Fig. from towardsai.net

(New) methods applying deep neural networks for jet classification

- Low level inputs (e.g. momentum 4-vectors of jet constituents):
  - ⋆ NN “creates” most useful physical observables.
  - ⋆ NN approximates the optimal classifier.

Neural Networks learn real world notions by:

- Organizing its features in a hierarchical way.
- Building connections among them.
- Going from simple features to more complex and abstract ones.



Convolutional Neural Networks (CNN): Jets as Images

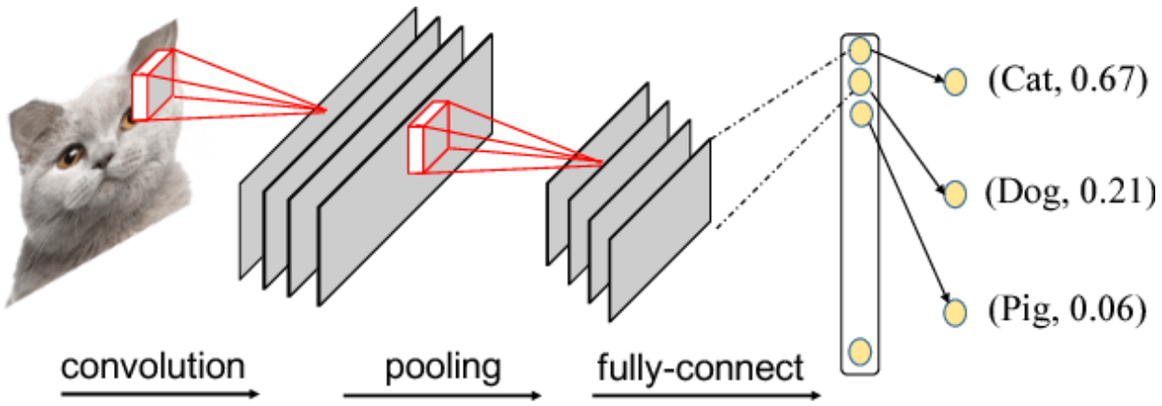


Fig. from arXiv:1712.01670

- Implement locality.
- Led to breakthroughs in computer vision:
  - Pixel level labeling of images for autonomous vehicles.
  - Google's automatic captioning of images.
  - Facebook's DeepFace project.
  - Microsoft surpassing human-level performance on ImageNet classification.

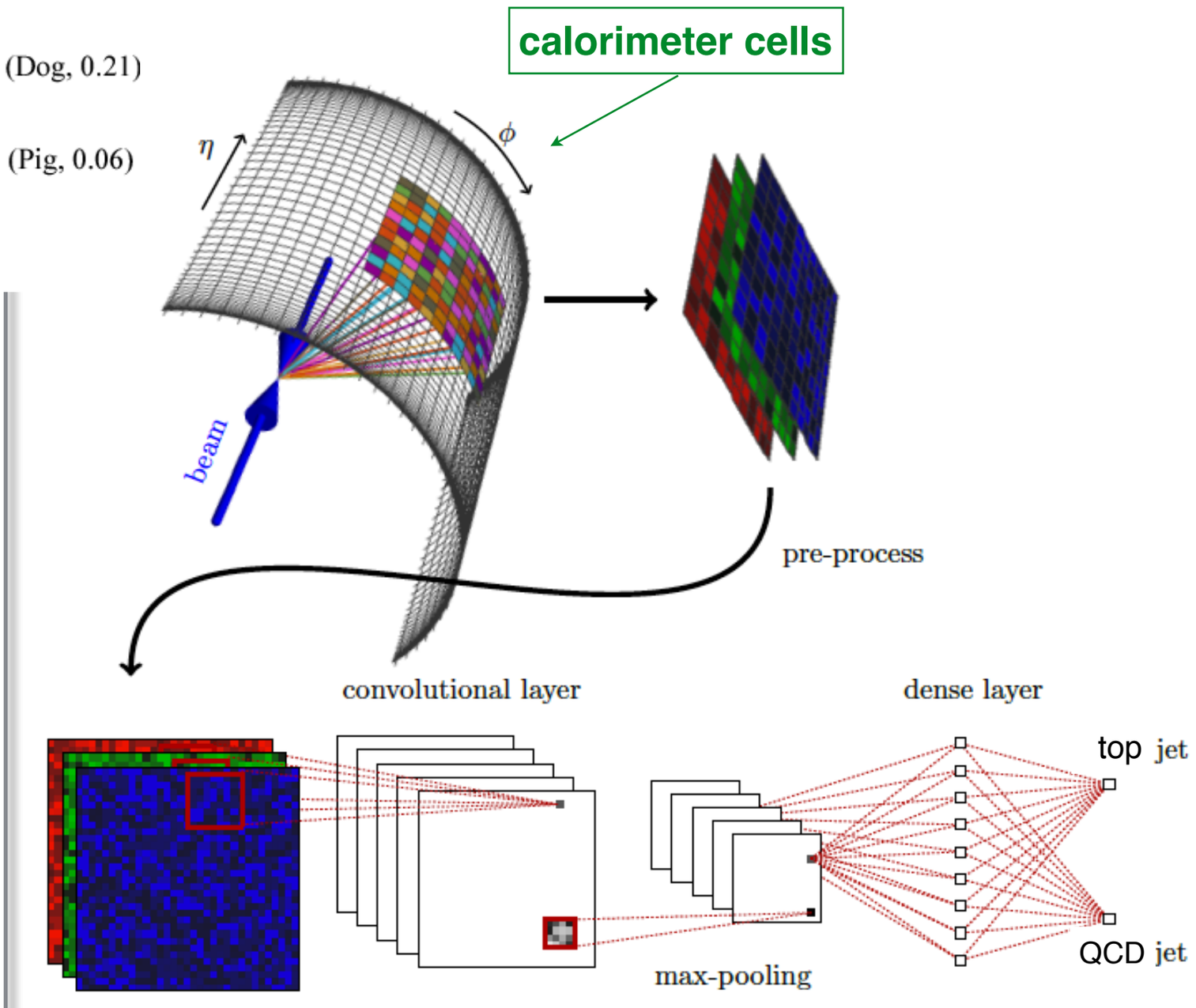
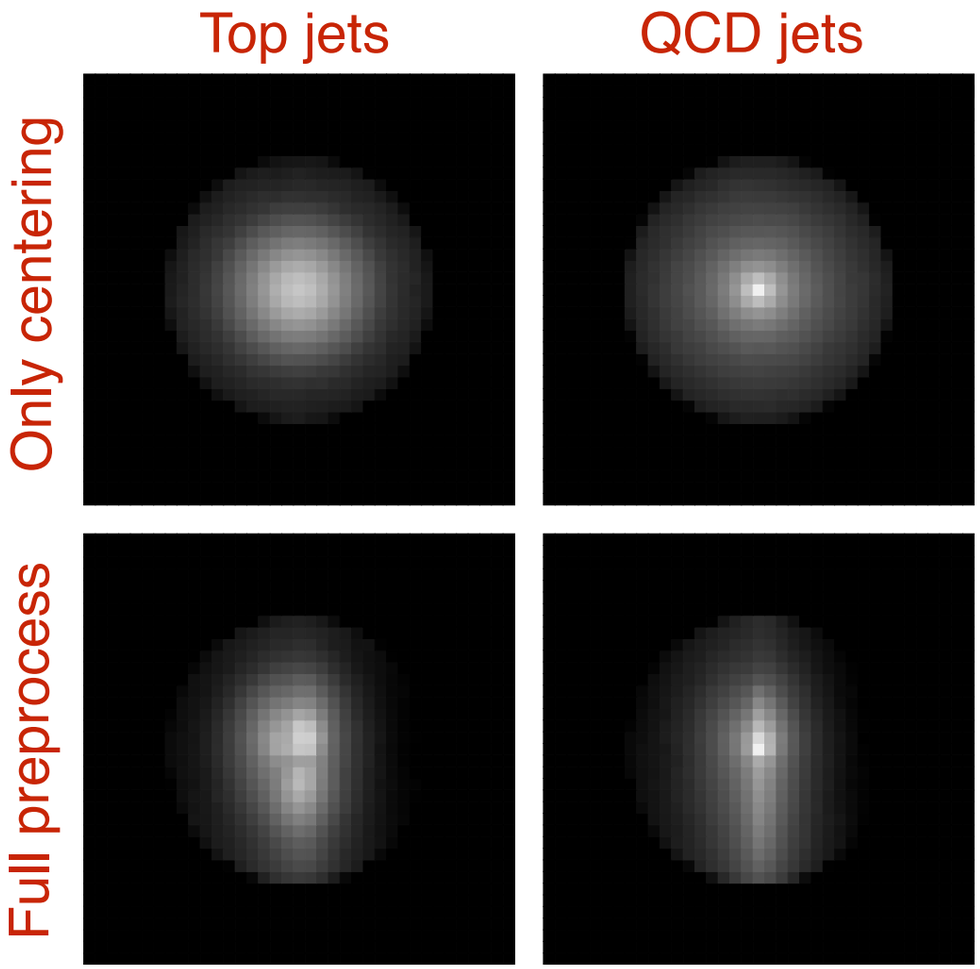


Fig. from Komiske, Metodiev & Schwartz ‘16

Jet images

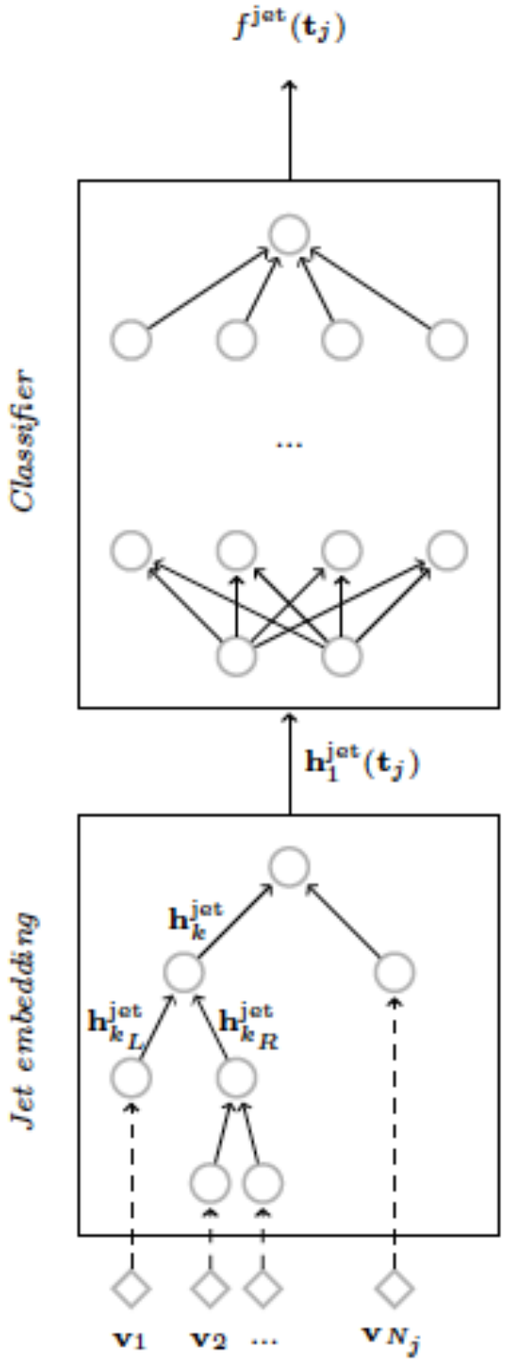
- Image preprocessing: center → rotate → flip → normalize → pixelate.



Average of 100k grayscale jets (total p_T in each pixel).

Fig. from arXiv:1803.00107

Recursive neural networks: jets as binary trees



$$\mathbf{h}_k^{\text{jet}} = \begin{cases} \mathbf{u}_k & \text{if } k \text{ is a leaf} \\ \sigma\left(W_h \begin{bmatrix} \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{u}_k \end{bmatrix} + b_h\right) & \text{otherwise} \end{cases}$$

$$\mathbf{u}_k = \sigma\left(W_u g(\mathbf{o}_k) + b_u\right)$$

- RecNN much fewer trainable parameters: ~10000 vs ~1 M
- Low level input features:

$$|\vec{p}\,|, \eta, \phi, E, \frac{E}{E_{\text{jet}}}, p_{\text{T}}, \theta$$

[Louppe, Cho, Becot, Cranmer ’17]

Tree Network in Network

[Macaluso, Cranmer ’19]

https://github.com/diana-hep/TreeNiN

Other NN architectures

Recurrent NNs: jets as a sequence.

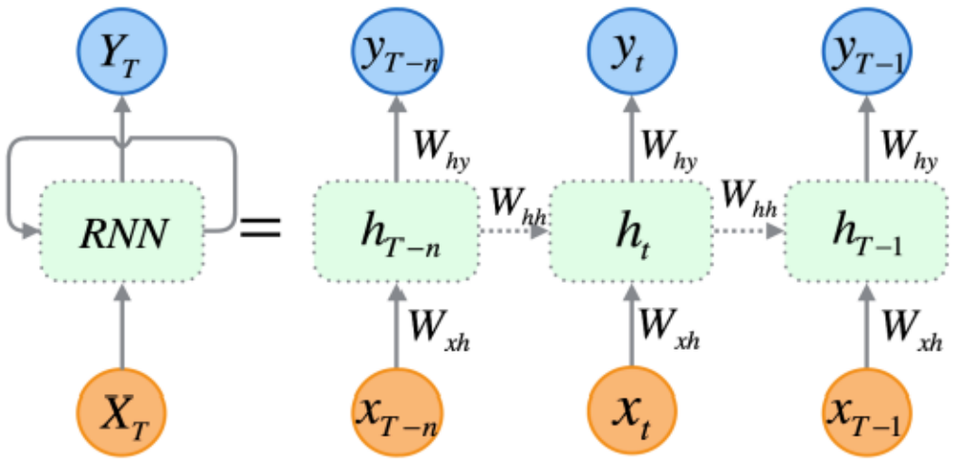
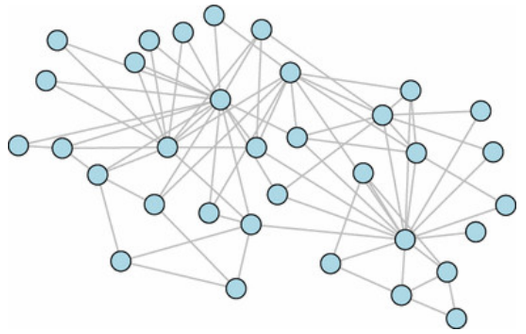


Fig. from www.i2tutorials.com

Sequential ordering of jet 4-vectors

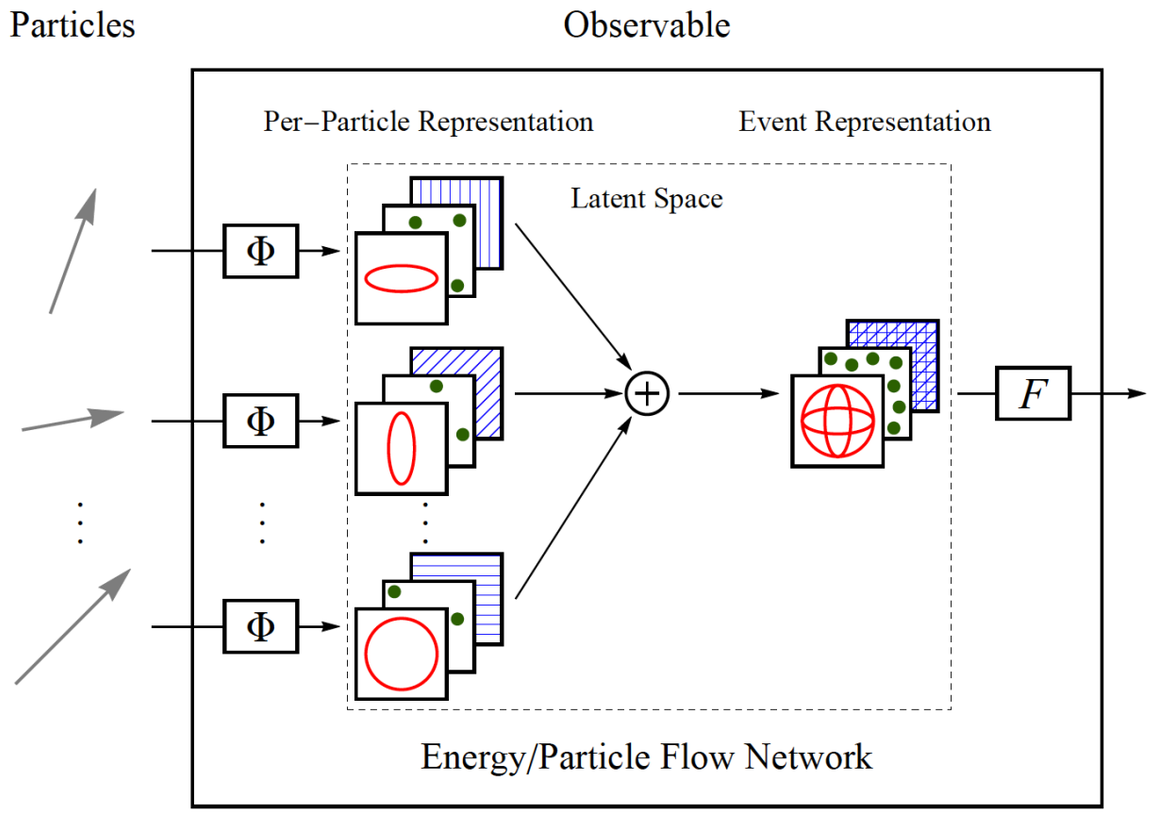
Graph NNs: Jets as point clouds.



ParticleNet [1902.08570]

Fig. from girlstalkmath.com

DeepSets [1810.05165, 1810.05165]



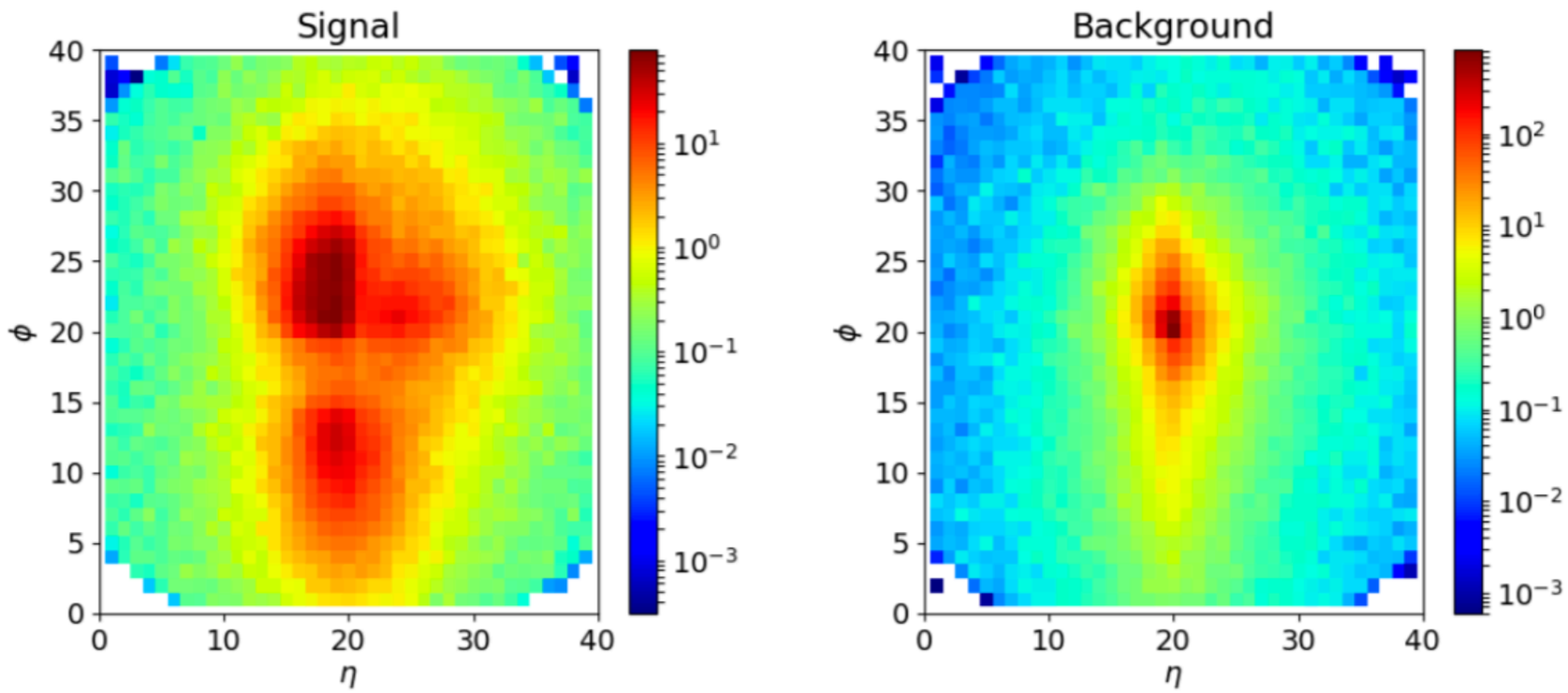
Top Tagging Reference Dataset

- **Top vs QCD jets** produced with Pythia8 @ 14TeV.
- Anti-kt R=0.8 Delphes jets (energy-flow algorithm), pT=[550,650] GeV, match and merge requirements.
- No uncertainties, pile-up.
- Only 4-vectors up to 200 constituents. (No charge or displaced vertex information)
- (1.2M, 400k, 400k) jets as (train, val, test) sets.

Contact

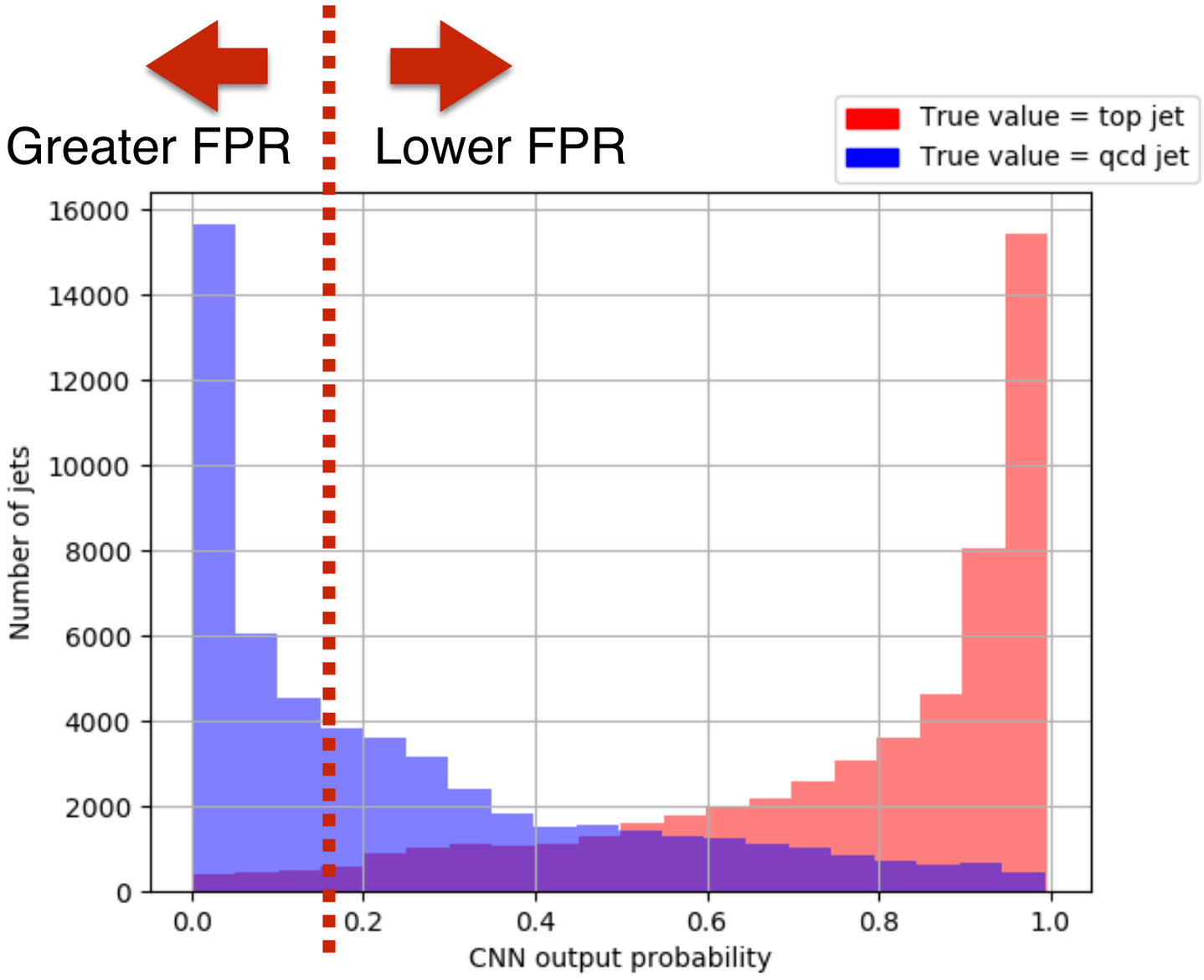
Gregor Kasieczka (gregor.kasieczka@cern.ch)
Michael Russel (russell@thphys.uni-heidelberg.de)
Tilman Plehn (plehn@uni-heidelberg.de)

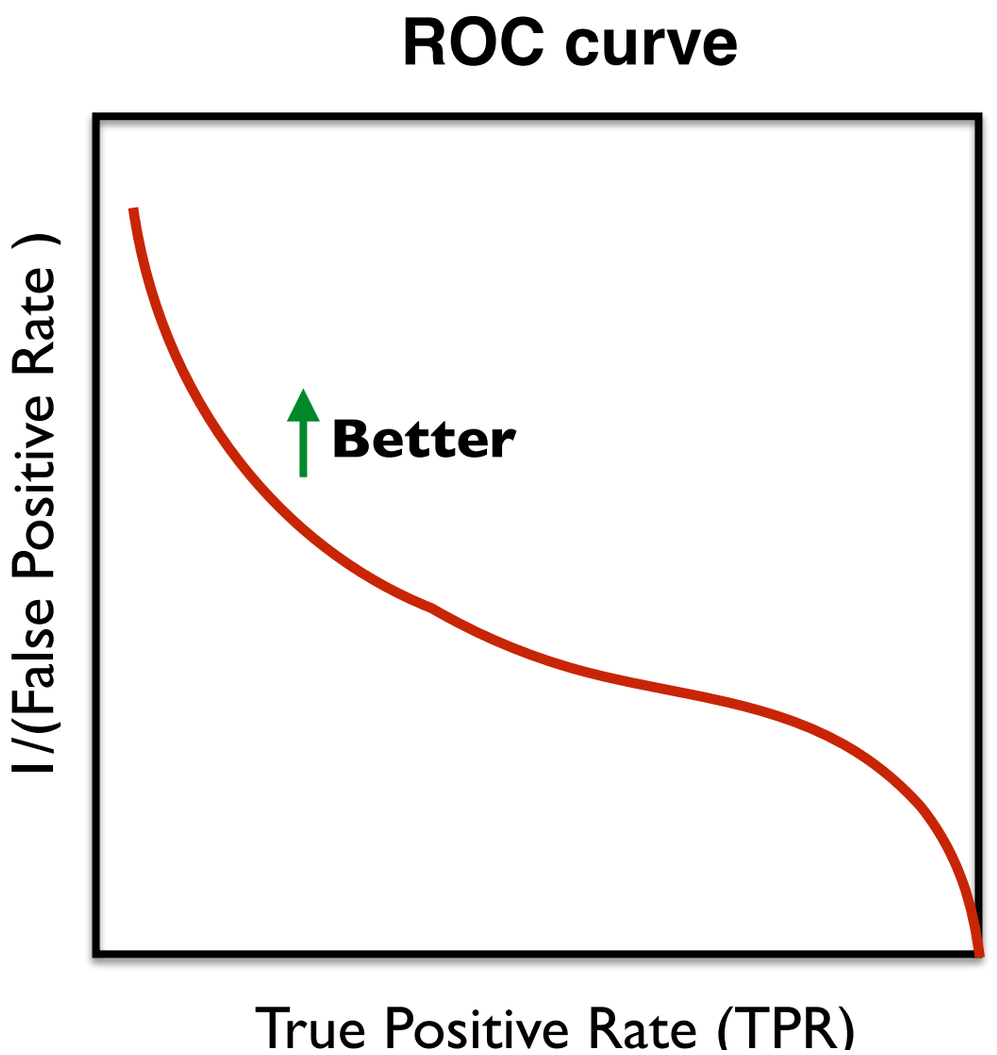
Get dataset here



Average of 10k jet images

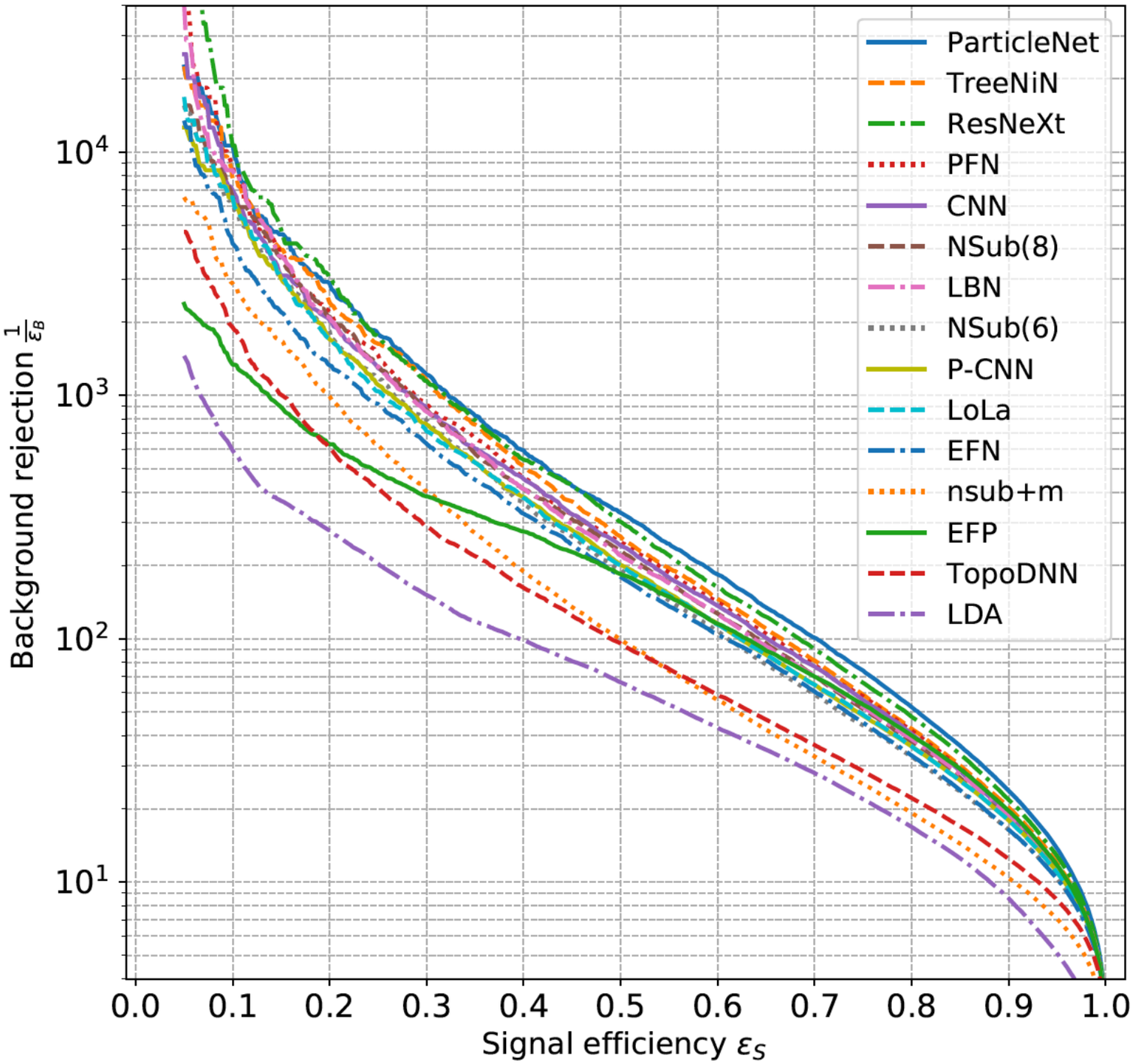
ROC curves





Goal : maximize the split between signal and background output probabilities

The Machine Learning Landscape of Top Taggers

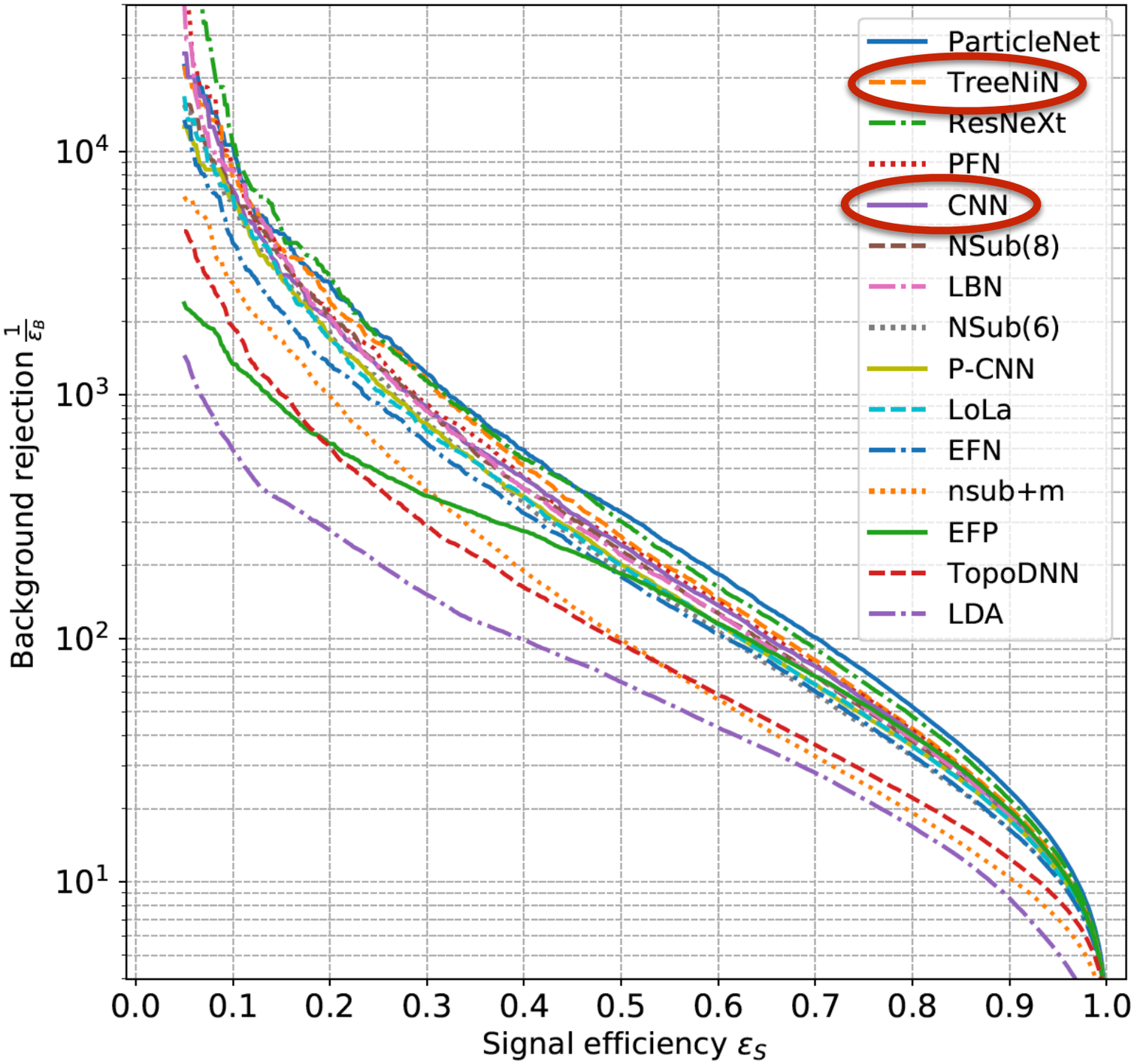


- 9 models trained for each classifier.
- ROC curve for the model that gives the median AUC.

The Machine Learning Landscape of Top Taggers

TreeNiN
[Cranmer, Macaluso '19]

CNN
[Macaluso, Shih '18]

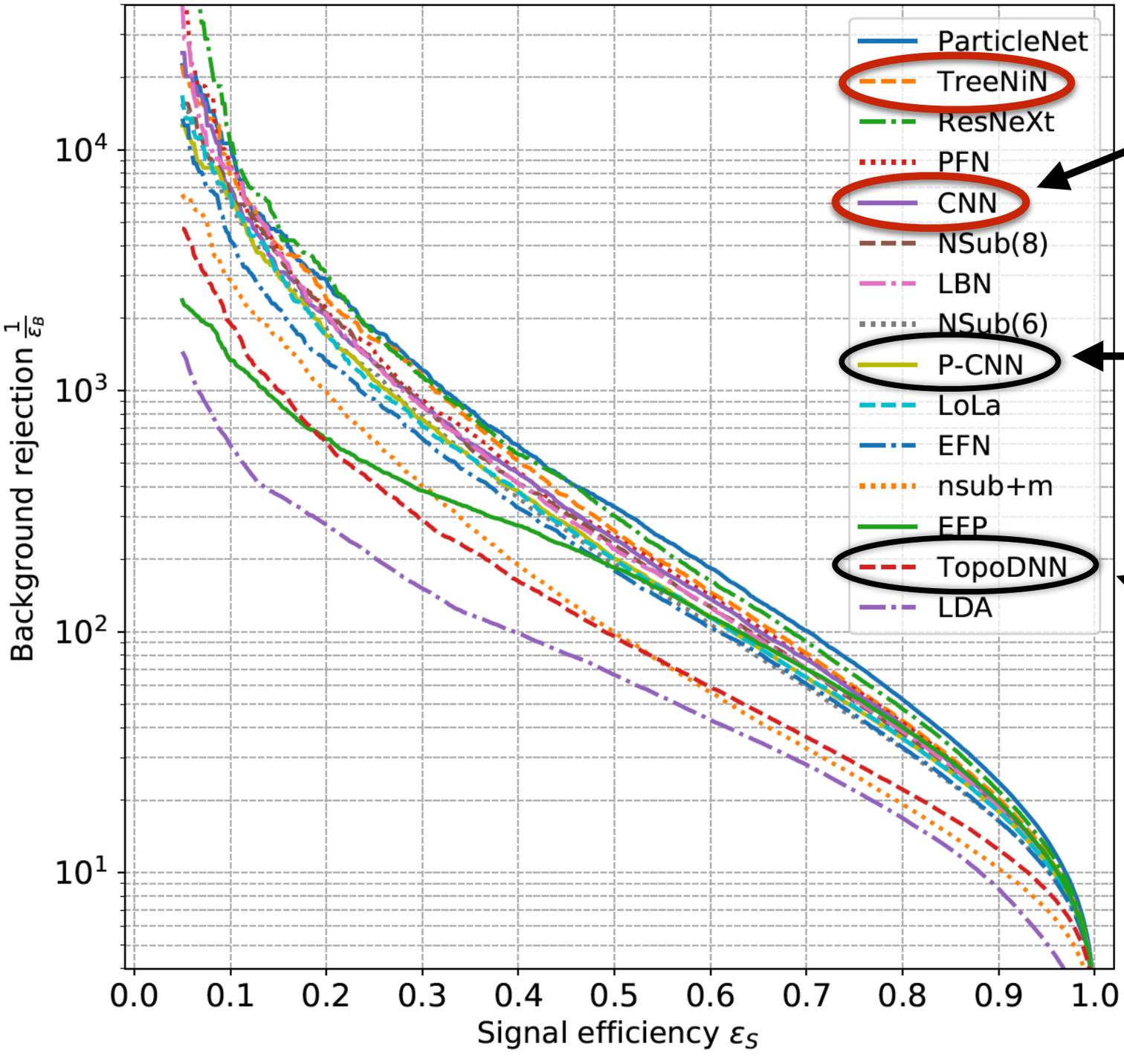


- 9 models trained for each classifier.
- ROC curve for the model that gives the median AUC.

The Machine Learning Landscape of Top Taggers

TreeNiN
[Macaluso, Cranmer '19]

CNN
[Macaluso, Shih '18]



CMS ImageTop architecture initially based on CNN

Similar technology to CMS DeepAK8

Similar technology to ATLAS TopoDNN

- 9 models trained for each classifier.
- ROC curve for the model that gives the median AUC.

Performance metrics

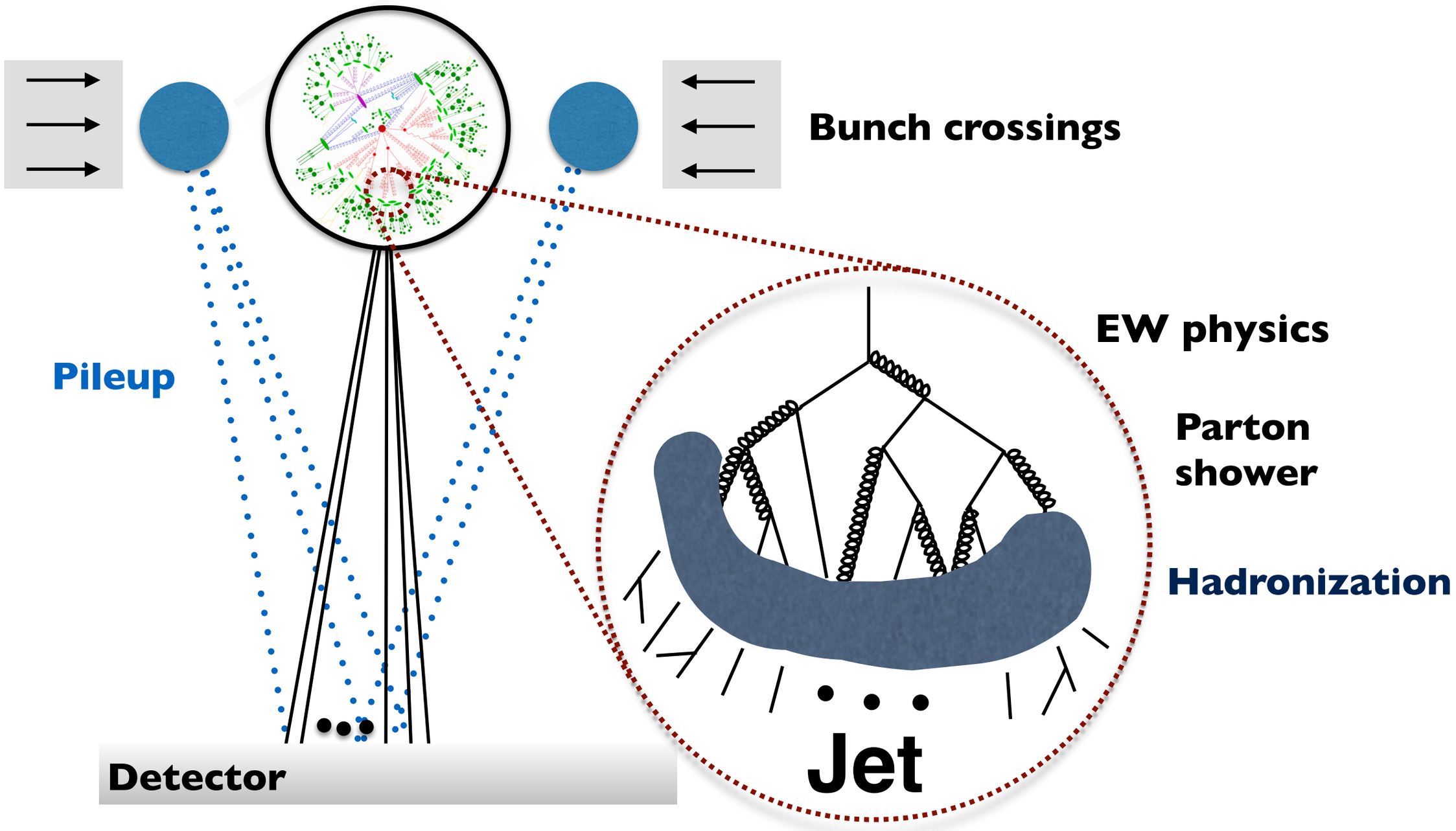
	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$) single	mean	median	#Param
CNN [16]	0.981	0.930	914±14	995±15	975±18	610k
ResNeXt [31]	0.984	0.936	1122±47	1270±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	382± 5	378 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	798±12	808±13	57k
Multi-body N-subjettiness 8 [24]	0.981	0.929	867±15	918±20	926±18	58k
TreeNiN [43]	0.982	0.933	1025±11	1202±23	1188±24	34k
P-CNN	0.980	0.930	732±24	845±13	834±14	348k
ParticleNet [47]	0.985	0.938	1298±46	1412±45	1393±41	498k
LBN [19]	0.981	0.931	836±17	859±67	966±20	705k
LoLa [22]	0.980	0.929	722±17	768±11	765±11	127k
LDA [54]	0.955	0.892	151±0.4	151.5±0.5	151.7±0.4	184k
Energy Flow Polynomials [21]	0.980	0.932	384			1k
Energy Flow Network [23]	0.979	0.927	633±31	729±13	726±11	82k
Particle Flow Network [23]	0.982	0.932	891±18	1063±21	1052±29	82k
GoaT	0.985	0.939	1368±140		1549±208	35k

New results from ParticleNet group at Boost '19 => ParticleNet: $1/\epsilon_B(\epsilon_S = 0.3) = 1615 \pm 93$

ParticleNet-Lite $1/\epsilon_B(\epsilon_S = 0.3) = 1262 \pm 49$

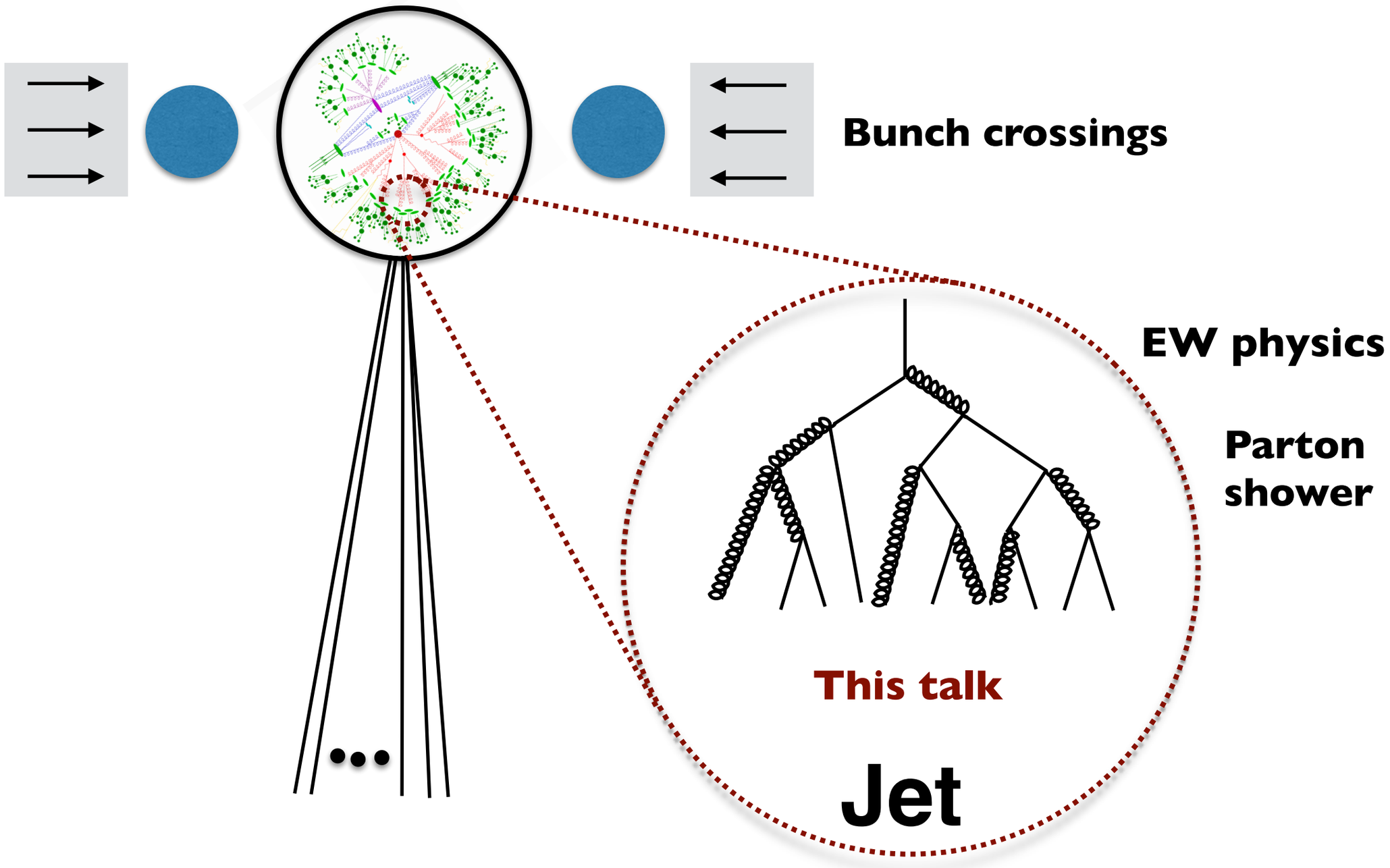
Reframing Jet Physics

[Stefan Hoche 1411.4085]

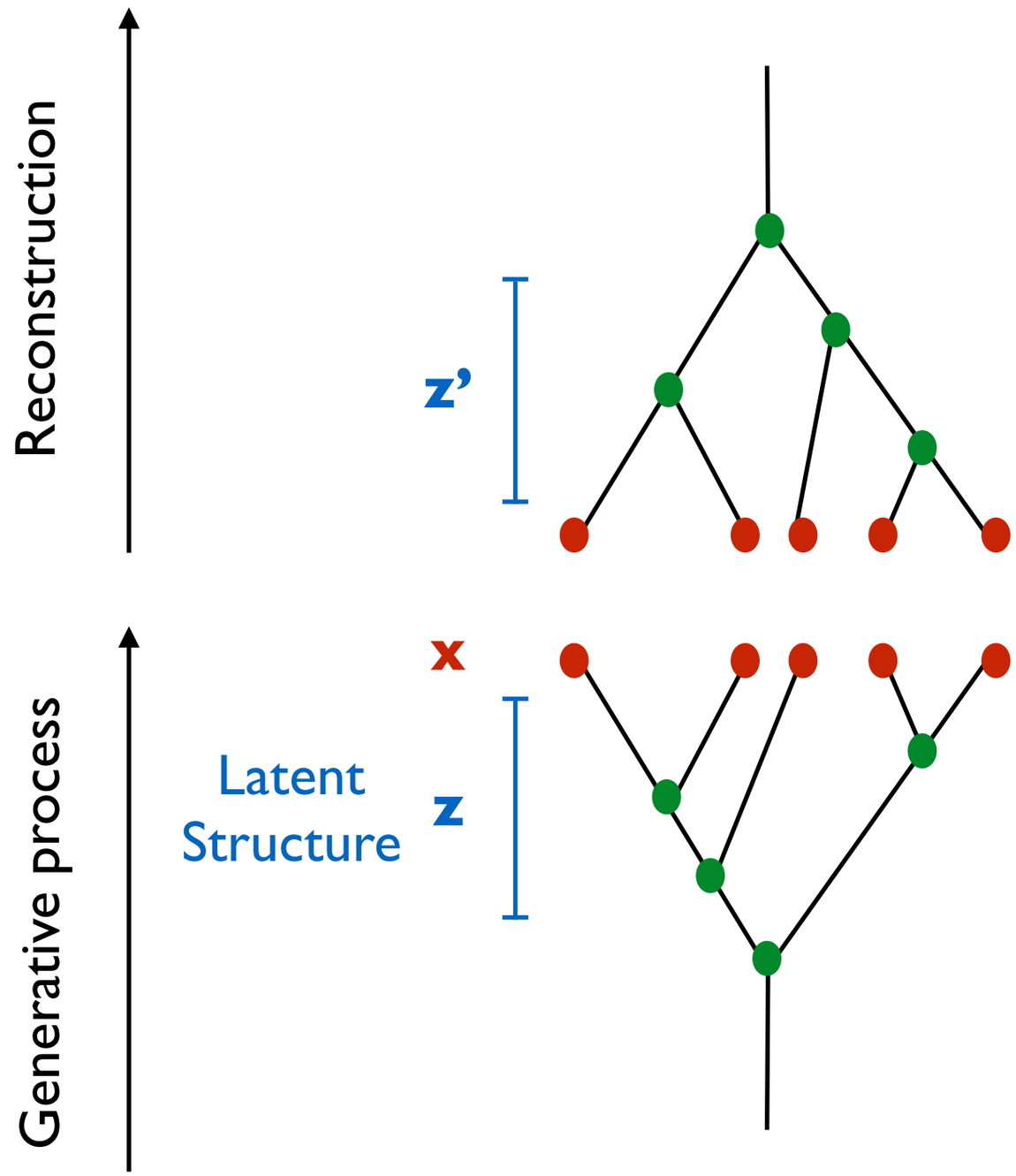


Reframing Jet Physics

[Stefan Hoche 1411.4085]



Jet clustering



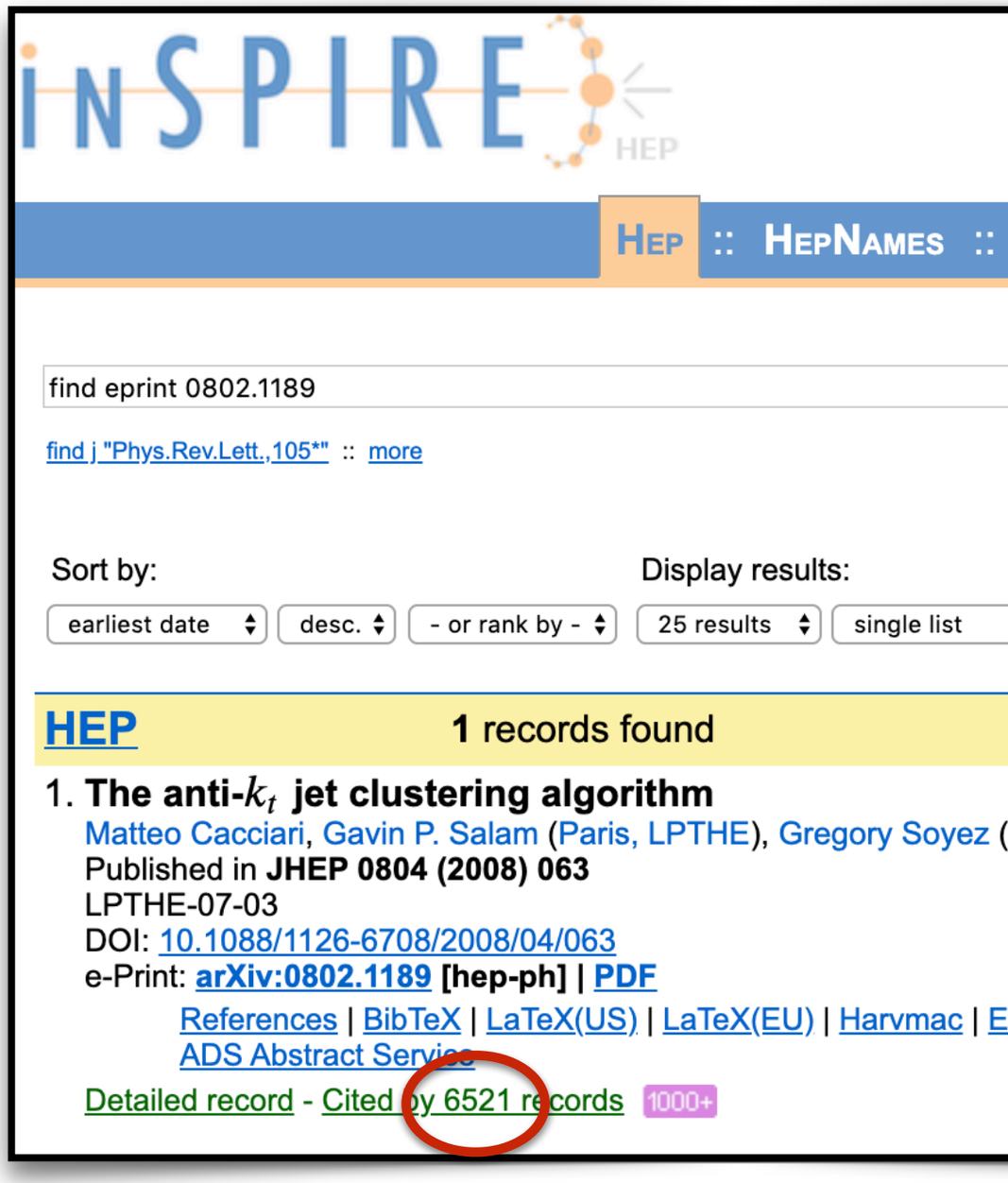
Generalized kt clustering algorithms

- "Heuristic" domain-specific clustering algorithms that do not use the generative model.
- Idea: sequentially cluster jet constituents.
- Permutation invariance: add 4-momenta of each pair that is clustered.
- Merge closest pair based on the distance measure:

$$d_{ij}^{(\alpha)} = \min(p_{ti}^{2\alpha}, p_{tj}^{2\alpha}) \frac{\Delta R_{ij}^2}{R^2}$$

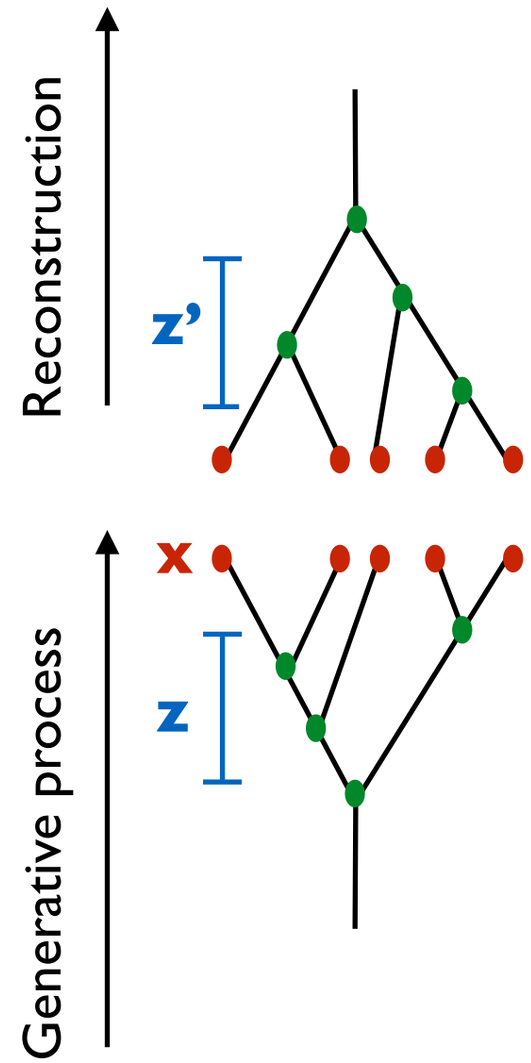
$$\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$$

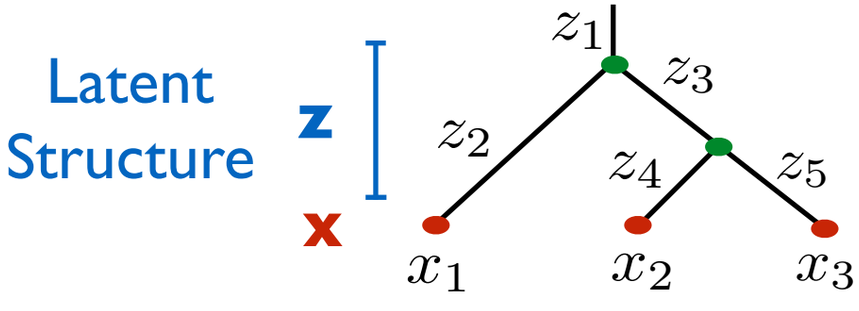
$\alpha = \{1, 0, -1\}$ specifies the the kt, Cambridge/ Aachen and anti-kt algorithms respectively.



Problem settings / probabilistic quantities:

- Joint likelihood $p(x, z|\theta)$
- Maximum likelihood history: $\mathrm{ArgMax}_z\ p(x, z|\theta)$
- Marginal likelihood $p(x|\theta) = \int dz\ p(x, z|\theta)$
- Maximum likelihood parameter: $\mathrm{ArgMax}_\theta\ p(x|\theta)$
- Posterior distribution on histories: $p(z|x, \theta)$
- Posterior distribution on θ : $p(\theta|x)$





Joint likelihood:

$$p(x, z|\theta) = \prod_j p(x_j|z_{\mathrm{parent}(x_j)}, \theta) \prod_i p(z_i|z_{\mathrm{parent}(z_i)}, \theta)$$

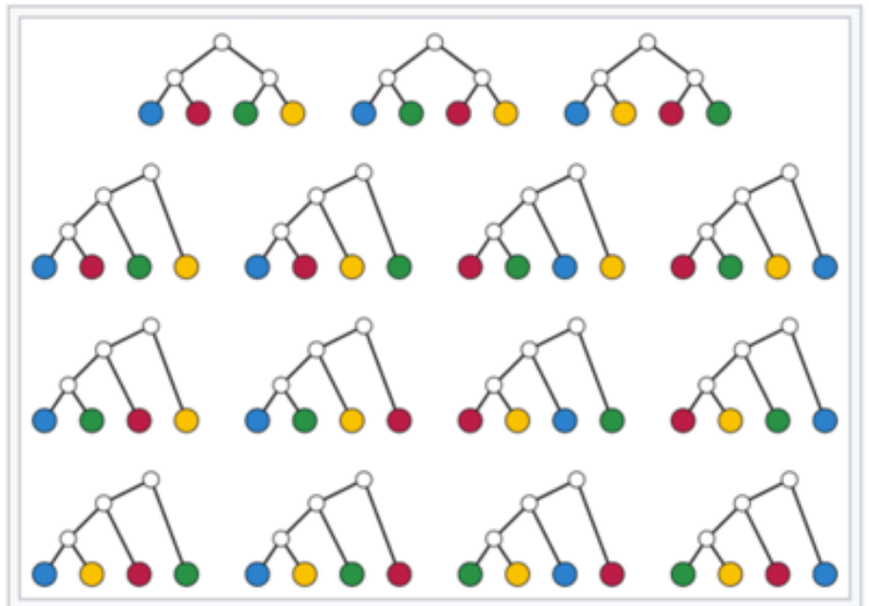
Generation and inference unification

Aim to invert the generative model

Most likely splitting history

- Reconstructing the splitting history is like inverting the generative model.
- **Hard**! The number of histories grows like (2N-3)!!, with N the number of jet constituents.
- Usually jet reconstruction algorithms don't use the probability model directly.
- We use the likelihood for the history as the optimization **objective.**

# of leaves	Approx. # of trees
4	15
5	100
7	10 k
9	2 M
11	600 M



The fifteen different rooted binary trees (with unordered children) on a set of four labeled leaves, illustrating $15 = (2 \times 4 - 3)!!$ (see article text).

https://en.wikipedia.org/wiki/Double_factorial

Parton Showers

PYTHIA

Sherpa

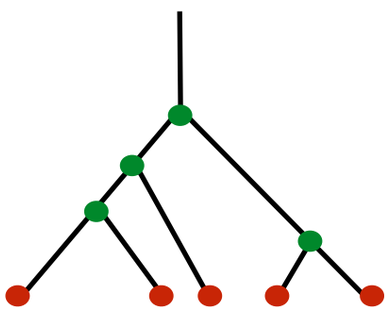
H7

Herwig 7

- It is hard to access the joint likelihood of a showering process.

- Typical implementations of parton showers involve kinematic reshuffling which destroys analytic control of the joint likelihood.

(Analytic control could be restored [Bauer & Tackmann '08])

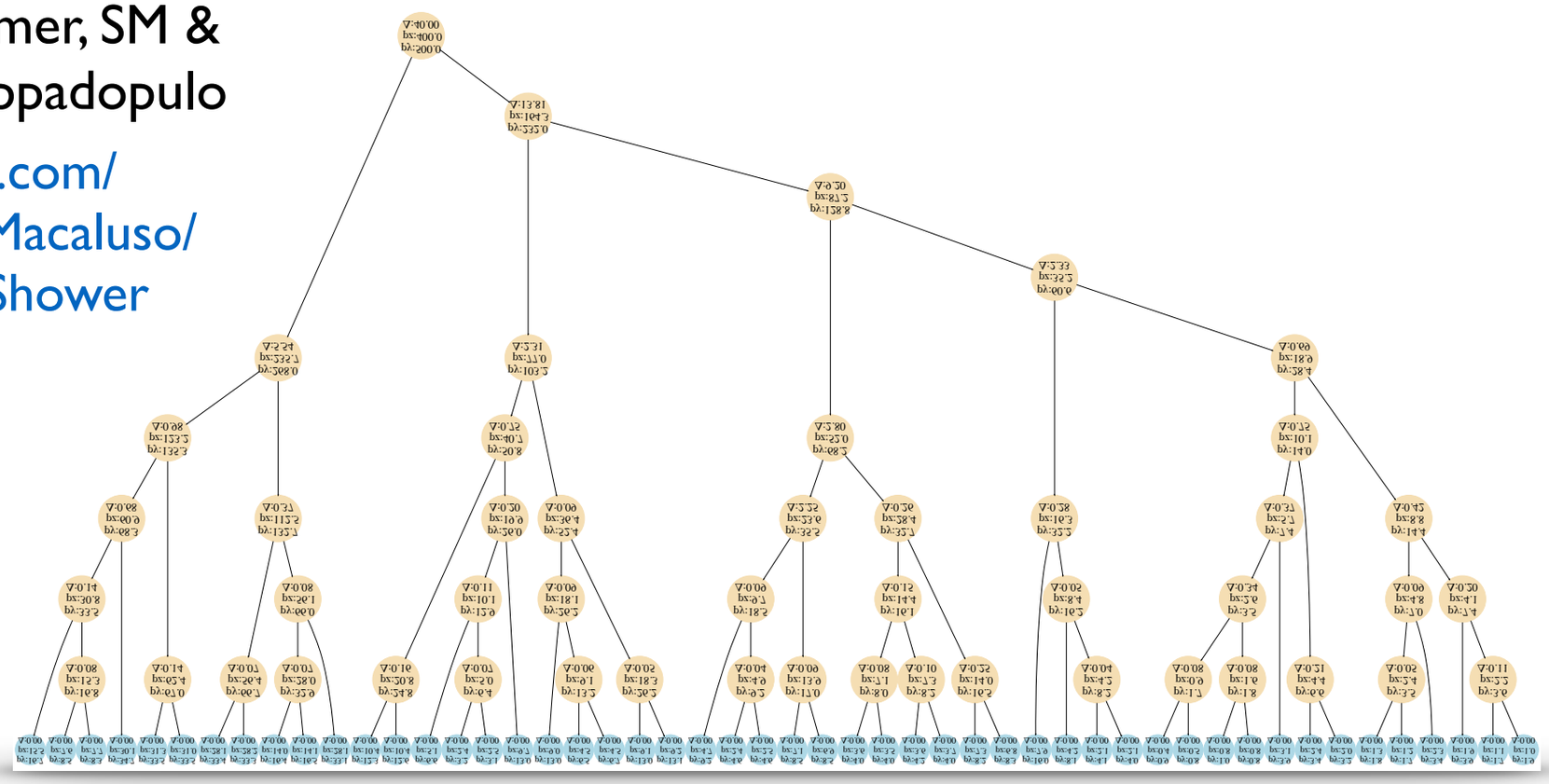


To prototype we built our own toy model!

Ginkgo: Toy Generative Model for Jets

Kyle Cranmer, SM &
Duccio Pappadopulo

github.com/
SebastianMacaluso/
ToyJetsShower



Motivation

- Build a model to aid in ML research for jet physics.

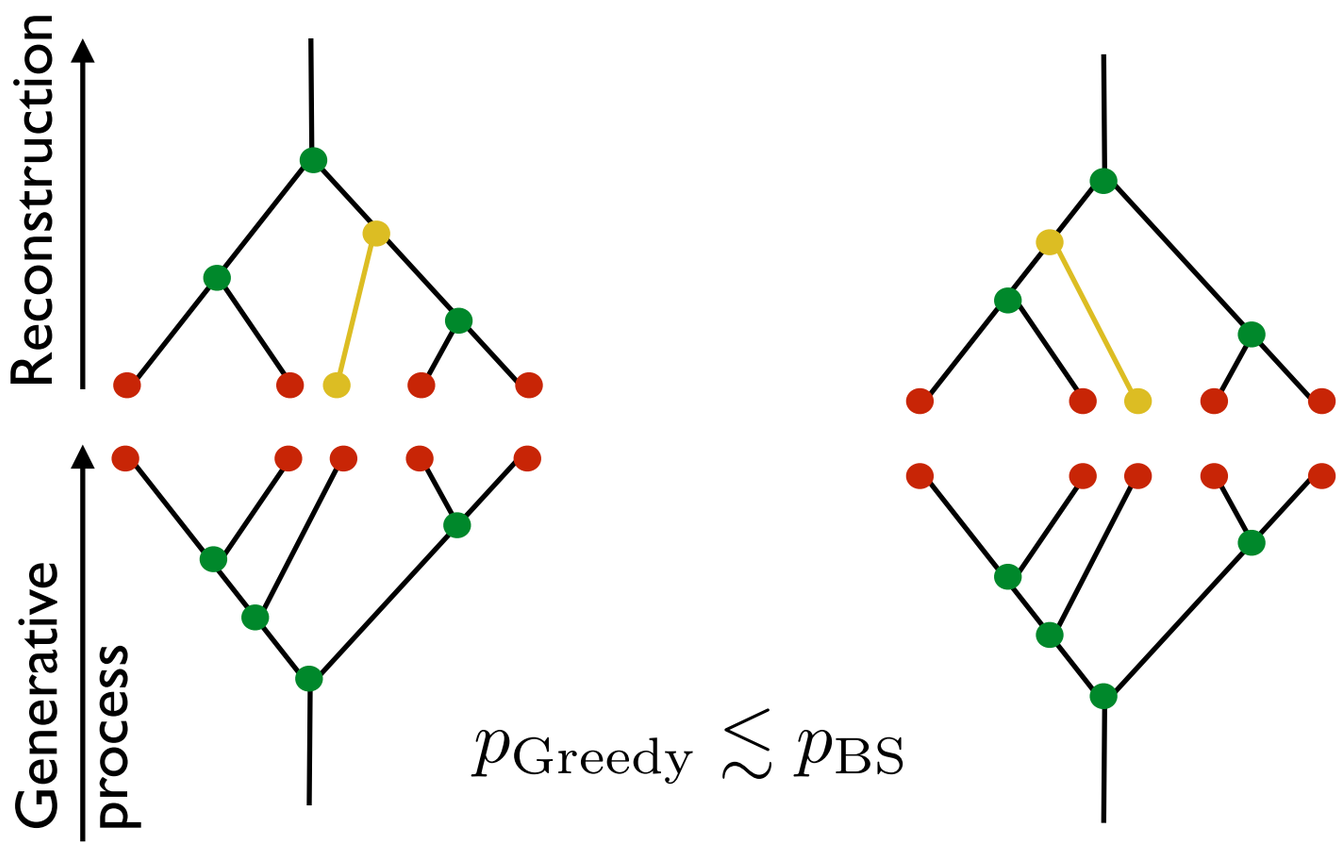
Generation

- Tractable joint likelihood.
- Captures essential ingredients of parton shower generators in full physics simulations.
- Implements an analogue to a parton shower (no hadronization effects).
- Python implementation with few software dependencies.

Likelihood-based jet clustering

Greedy
Joint likelihood from locally maximizing the likelihood at each step.

Beam Search
Maximize the likelihood of multiple steps before choosing the latent path. Maybe we could fix mistakes?



Joint likelihood:

$$p(x, z|\theta) = \prod_i p(\mathrm{L}_i, \mathrm{R}_i|\mathrm{P}_i, \theta)$$

Goal: find the latent structure that maximizes the jet likelihood (MLE).

Given an algorithm, z is fixed.

$\hat{z}_{\text{MLE}} = \text{ArgMax}_z \; p(x, z|\theta)$
Viterbi algorithm
Computationally "infeasible" for jets with ~ 100 constituents

$\hat{z}_{\text{Greedy}}$

$\hat{z}_{\text{BS}}$

Computational bottlenecks in Jet Physics

- **Tuning the parameters of the shower model**
  - Ideally we would tune the parameters θ of PYTHIA, Sherpa or Herwig to find the maximum likelihood $p(\hat{\theta}|x)$. This typically involves a high dimensional parameter space and intractable quantities.

- **Event Generation for events with large jet multiplicity**
  - Matching algorithms, e.g. CKKW-L [arXiv:0109231, 0112284], need to consider every possible clustering history and reweigh them. This becomes infeasible for more than 6 jet configurations.

- **Simulation efficiency for jet backgrounds in signal-rich regions of phase space**
  - Enormous rate of multijets events: how to populate the tails of complicated phase space regions?

Emergent computational techniques that could address/ameliorate these problems:

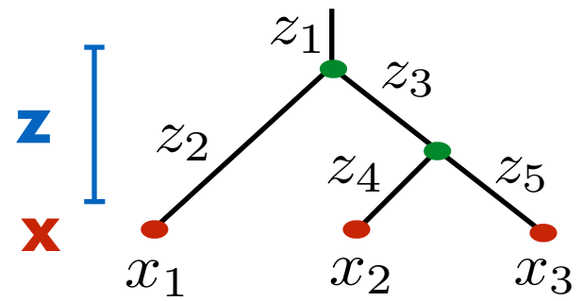
- Likelihood-free Inference of model parameters.
- Hierarchical Cluster Trellis.
- Probabilistic Programming.

Likelihood-free Inference of model parameters

[Brehmer, Louppe, Pavez & Cranmer '18]
[Brehmer, Cranmer, Louppe & Pavez '18]
[Andreassen & Nachman '19]

Posterior distribution on model parameters:

$$p(\theta|x) = \frac{p(x|\theta)\ p(\theta)}{\int d\theta' p(x|\theta')p(\theta')}$$



Marginal likelihood:

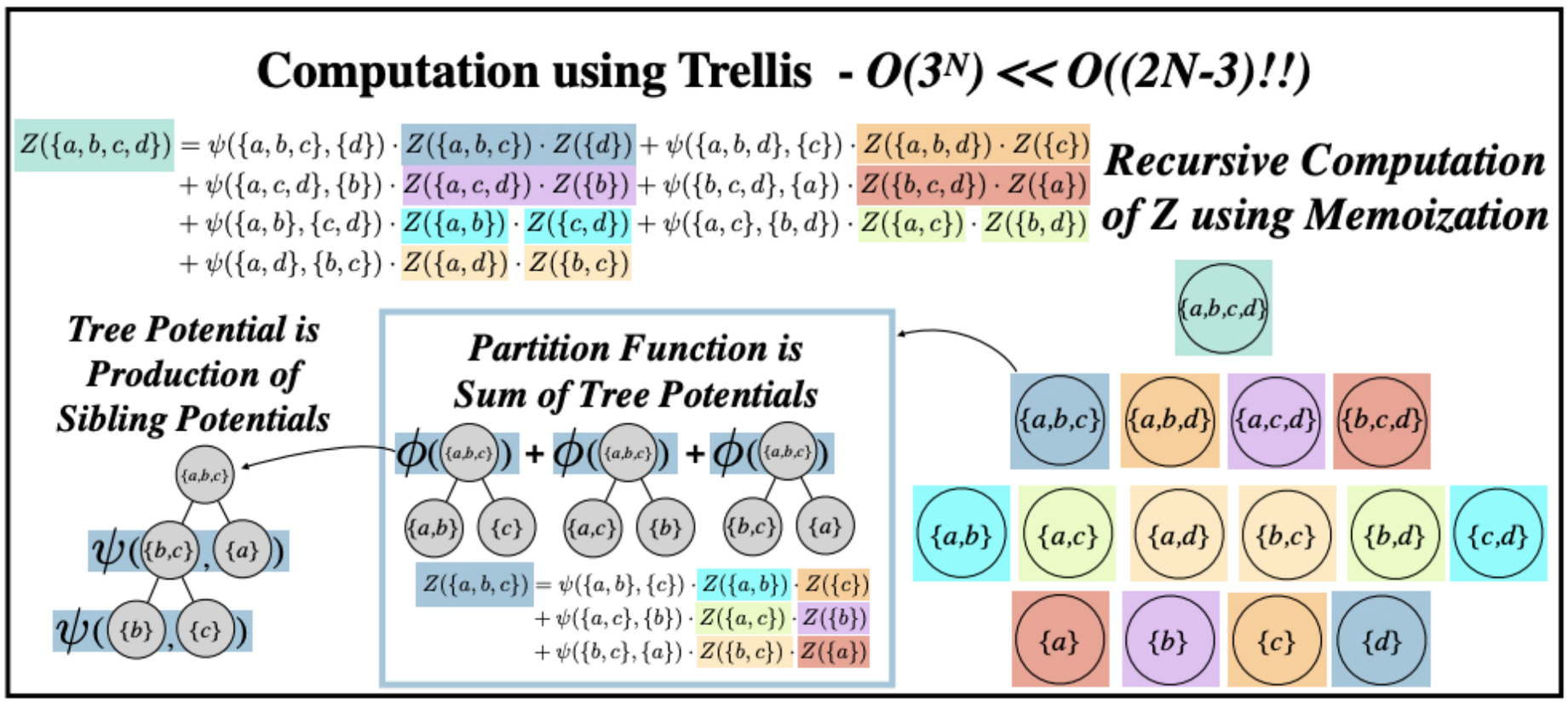
$$p(x|\theta) = \int dz\ p(x|z,\theta)\ p(z,\theta) \quad \longleftarrow \boxed{\text{Intractable}}$$

Approximate $p(x|\theta)$ using machine learning without explicitly marginalizing over all latent structures z.

Hierarchical Cluster Trellis

[Greenberg, Macaluso, Monath, Cranmer, et al '20, arXiv: 2002.11661]

- Hierarchical clustering: recursive splitting of a dataset into subsets until reaching singletons, e.g. jet constituents. (Also applications in cancer genomics).
- Parton showers Markov chain process reframed in terms of hierarchical clusterings.
- New data structure to efficiently consider every possible clustering history. Finds:
  - Exact Maximum likelihood showering history $\mathrm{ArgMax}_z\ p(x, z|\theta)$. Alternative to generalized k_t clustering algorithms.
  - Marginal likelihood $p(x|\theta)$ (sum of the likelihood of all the clustering histories).



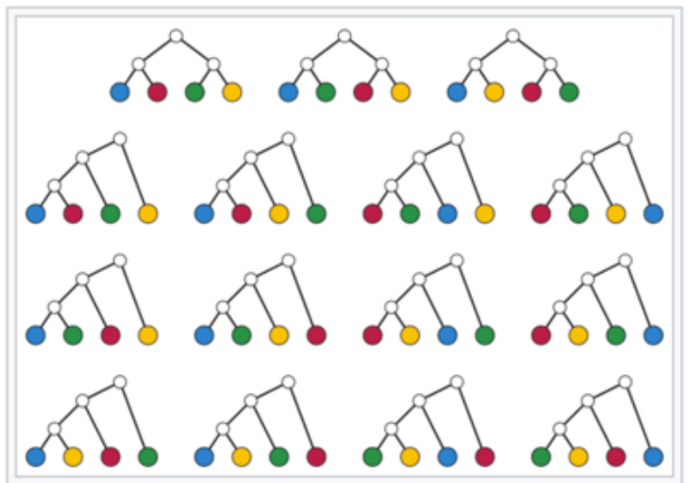
Hierarchical Cluster Trellis

Posterior distribution on histories: Cluster Trellis sampling procedure

- Trellis enables to sample histories from the posterior distribution without enumerating all possible clusterings.

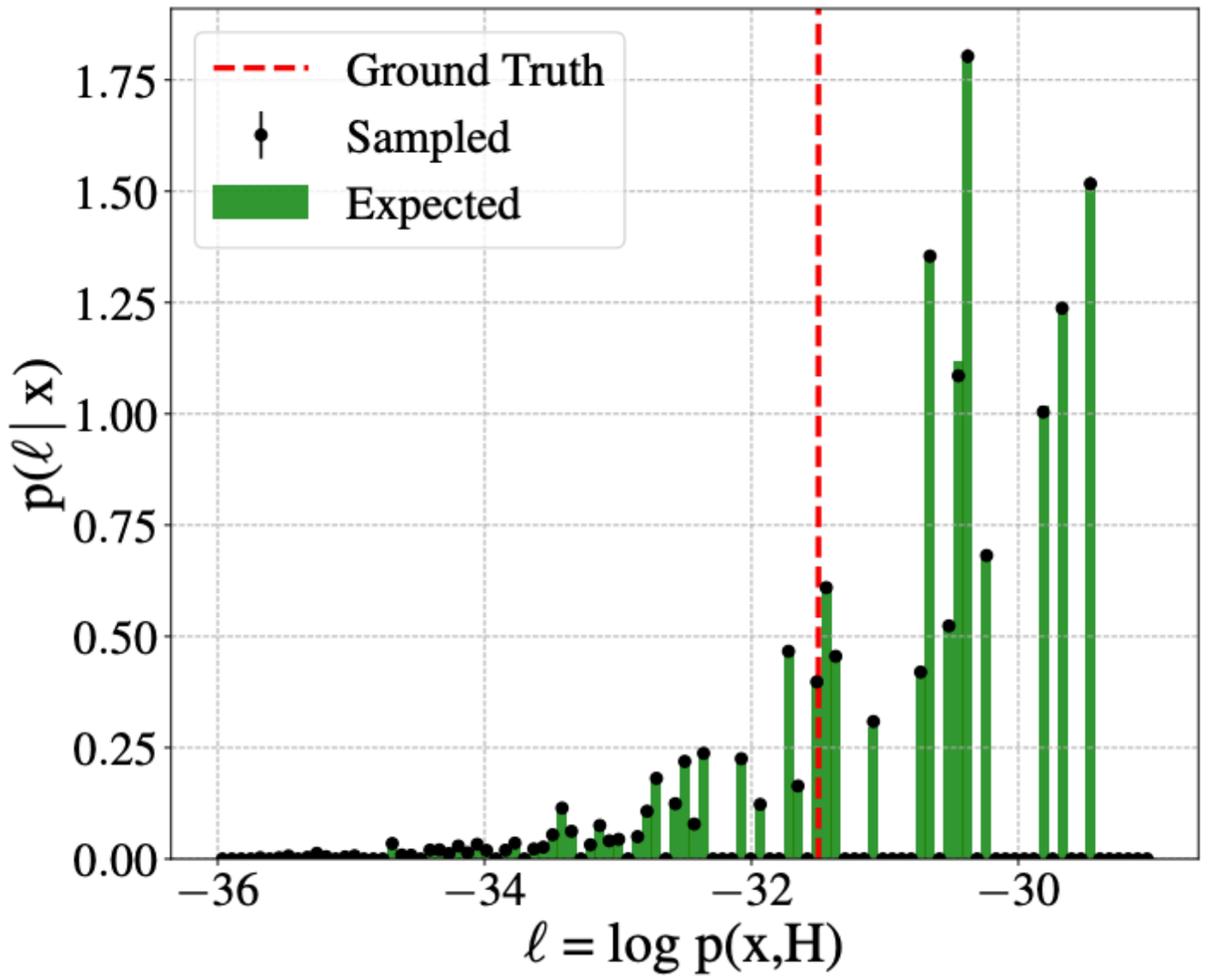
$$p(z|x,\theta) = \frac{p(x|z,\theta)\ p(z|\theta)}{p(x|\theta)}$$

# of leaves	Approx. # of trees
4	15
5	100
7	10 k
9	2 M
11	600 M



The fifteen different rooted binary trees (with unordered children) on a set of four labeled leaves, illustrating $15 = (2 \times 4 - 3)!!$ (see article text).

https://en.wikipedia.org/wiki/Double_factorial



Posterior distribution for a jet with 5 constituents from sampling 10^5 histories.

Hierarchical Cluster Trellis

Event Generation for events with large jet multiplicity.

During simulations, when implementing the CKKW-L matching algorithm, parton final states need to be reweighted with the corresponding Sudakov form factors of each history $p(x, z|\theta)$.

Trellis could be extended to consider 2 → 3 splittings (currently based on binary trees, 1 → 2 splittings); could make feasible the implementation of CKKW-L to a higher jet multiplicity.

Shower deconstruction
[Soper & Spannowsky '11] [Soper & Spannowsky '12]

- Simplified shower model.
- Add likelihood of each clustering history over *microjets* conditioned on S or B.
- Define the likelihood ratio:

$$\chi(p_N) = \frac{P(p_N|S)}{P(p_N|B)}$$

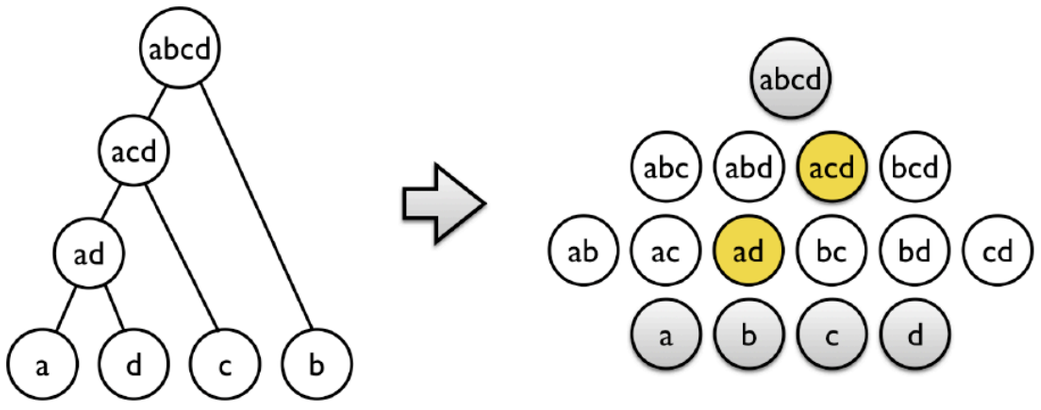
Trellis could provide a more efficient way to consider every history over microjets and/or extend the algorithm to more microjets.

Sparse Hierarchical Cluster Trellis

[Greenberg, Macaluso, Monath, Cranmer, et al '20, work in progress]

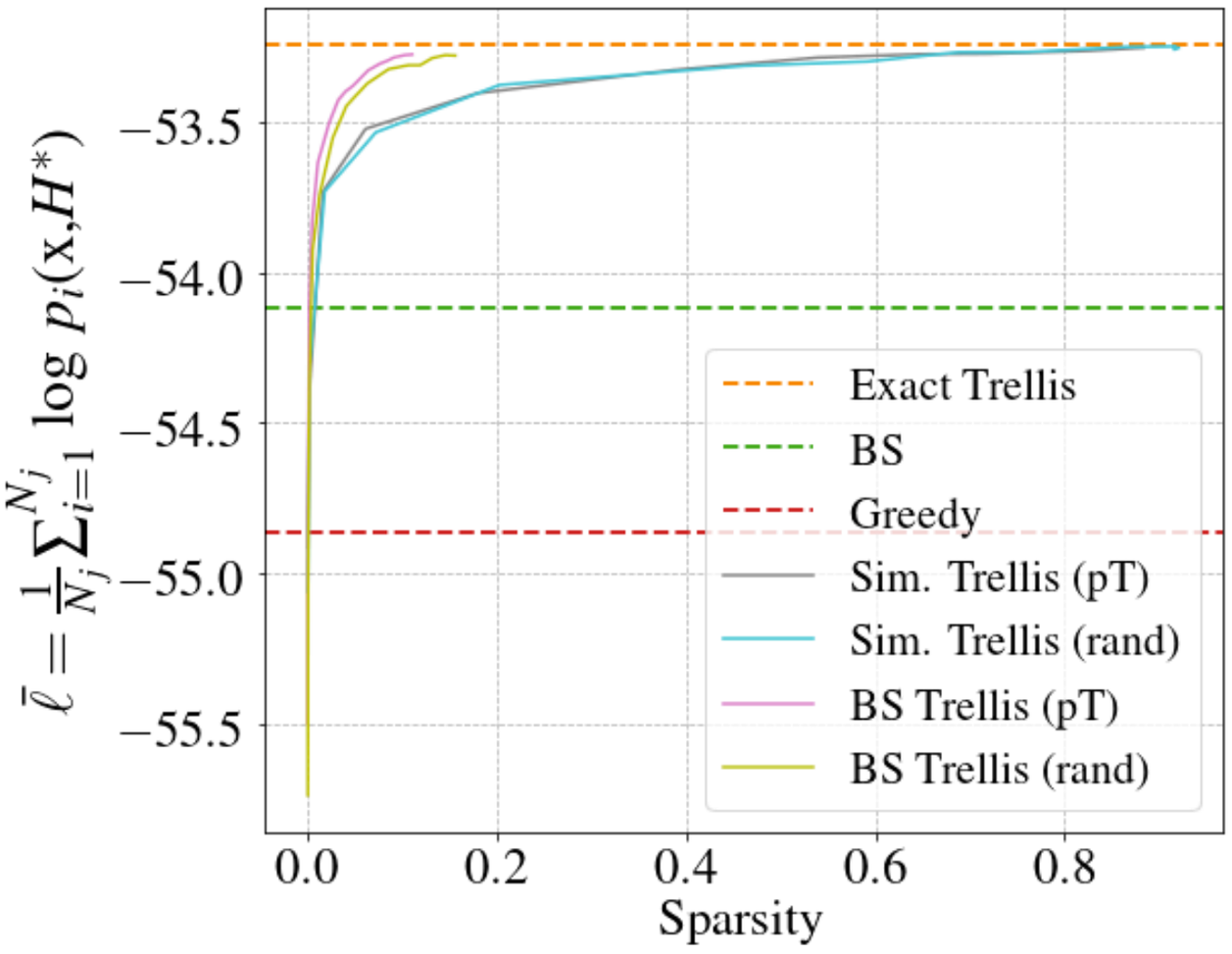
- Scales to much larger datasets by controlling the sparsity index, i.e. the fraction of hierarchies we consider from the total of (2 N - 3)!!.
- Many histories likelihood are negligible compared to the maximum likelihood (MLE) history.
- Finds approximate solutions for the MLE and marginal likelihood when implementing the exact trellis is not feasible.

Building strategies



- Simulator trellis: input binary trees from running a generative model.

- Beam search trellis: input trees from running beam search over a dataset of jet constituents.



Mean MLE history over 100 jets with 9 constituents.

Probabilistic Programming

[Drnevich, Cranmer, Baydin, Macaluso '20, work in progress]

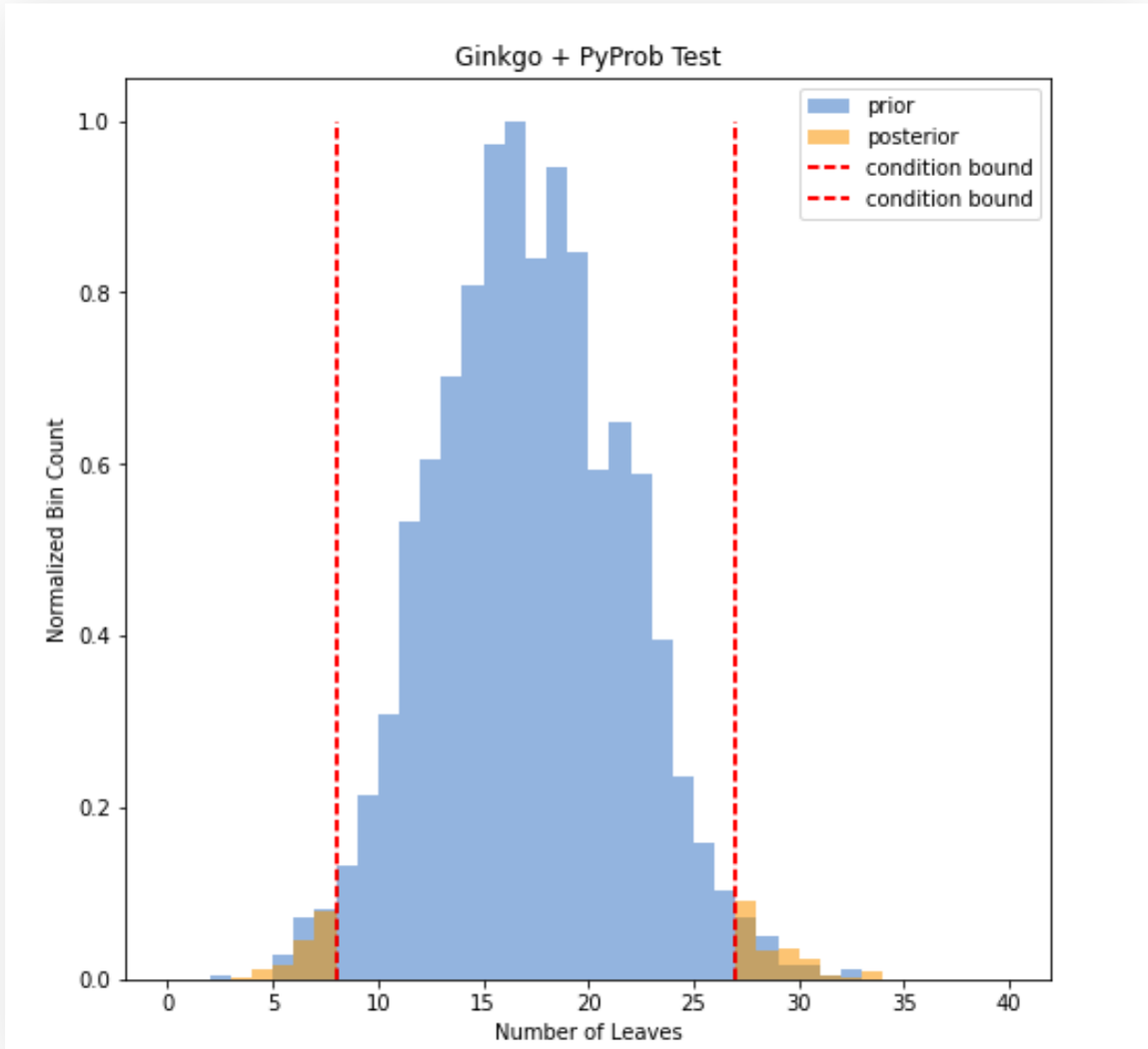
Monte Carlo event generators sample from the distribution $p(x, z|\theta)$ over histories and jet constituents through random number generators.

Probabilistic programming allows to condition the values of random variables x or z by hijacking the random number generators.

Can bias the simulator towards the desired output, e.g. importance sampling.

- Applied to Sherpa
  [Baydin, Shao, Cranmer et al '19].

- Allows to efficiently sample the tails of backgrounds in signal-rich regions of phase space.



Importance sampling on Ginkgo jets using PyProb and conditioning on the number of constituents.

Final remarks and future directions

- Many highly efficient and competitive top taggers with comparable performance!

- Reframe jet physics in probabilistic terms.

- Goal: Unification of generation and inference.
  - e.g. Reframe jet clustering as finding maximum likelihood clustering.
  - e.g. Reframe parameter tuning as maximum likelihood after marginalizing over the (latent) clustering histories.

- New likelihood-based clustering algorithms: greedy, beam search and hierarchical trellis. Beam search is strictly better than greedy algorithm (e.g. kt) in terms of estimating the most likely tree.

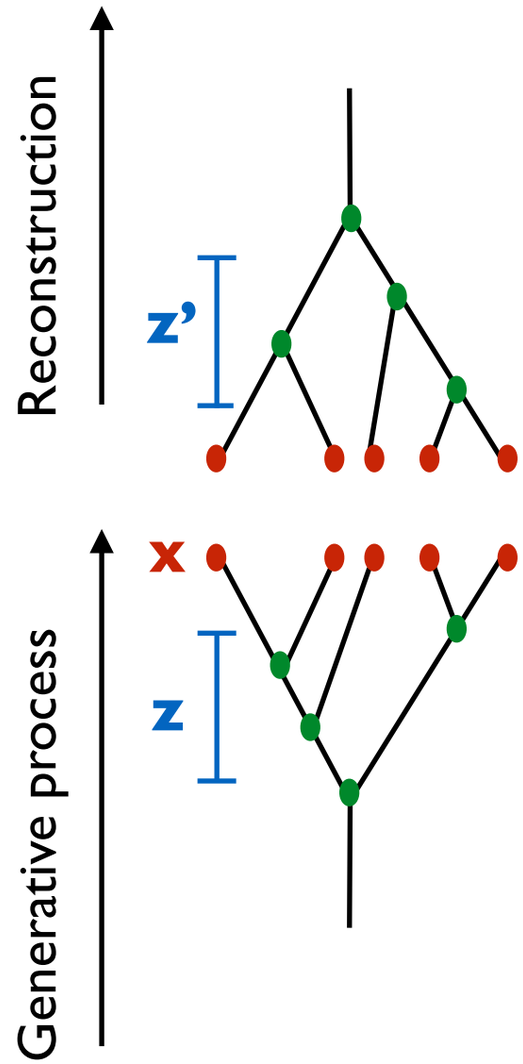
- Emerging computational techniques to address bottlenecks in jet physics:
  - Likelihood-free inference for model parameters tuning.
  - Hierarchical cluster trellis that enables to exactly obtain the maximum likelihood history and consider every possible clustering history.
  - Probabilistic programing to efficiently sample the tails of complicated phase space regions.

Thanks for your attention!

Problem settings

Maximum likelihood history

$$\mathrm{ArgMax}_z \; p(x, z|\theta)$$



Generation and inference unification

- **Most likely splitting history**

Number of clustering histories for N leaves grows as:

$$a(N) = (2N-3)!!$$

# of leaves	Approx. # of trees
4	15
5	100
7	10 k
9	2 M
11	600 M

The fifteen different rooted binary trees (with unordered children) on a set of four labeled leaves, illustrating $15 = (2 \times 4 - 3)!!$ (see article text).

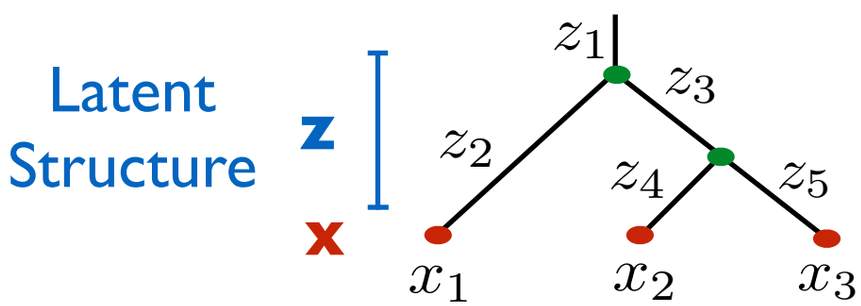
https://en.wikipedia.org/wiki/Double_factorial

- **Posterior distribution on histories**

(Distribution over generated jet)

$$p(z|x,\theta) = \frac{p(x|z,\theta)\ p(z|\theta)}{p(x|\theta)}$$

Sampling with MCMC or probabilistic programing techniques ?



Generation and inference unification

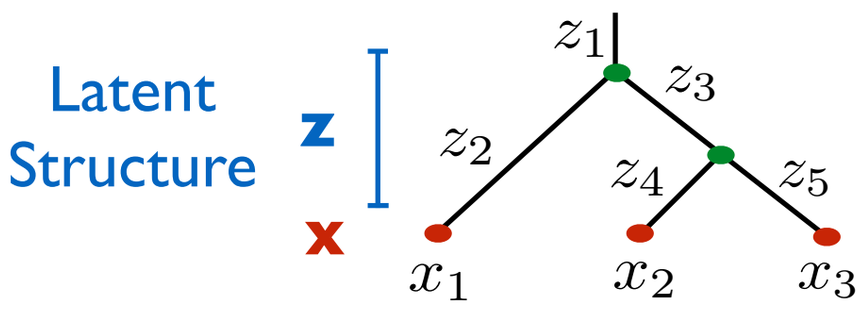
Aim to invert the generative model

- **Posterior distribution on histories**

(Distribution over generated jet)

$$p(z|x,\theta) = \frac{p(x|z,\theta)\ p(z|\theta)}{p(x|\theta)}$$

Sampling with MCMC or probabilistic programing techniques ?



- **Related work: Q-jets**
[Ellis, Hornig, Roy, Krohn & Schwartz '12]

(Distribution over reconstructed jet)

Assemble trees by sequentially choosing a pair with probability $\Omega_{ij} = \omega_{ij}/N$

$$\omega_{ij}^{(\alpha)} \equiv \exp\left(-\alpha \frac{(d_{ij} - d^{\min})}{d^{\min}}\right)$$

- **Related work: Shower deconstruction**
[Soper & Spannowsky '11] [Soper & Spannowsky '12]

  - Simplified shower model.
  - Add likelihood of each clustering history over *microjets* conditioned on S or B.
  - Define the likelihood ratio: $\chi(p_N) = \frac{P(p_N|S)}{P(p_N|B)}$

Generation and inference unification

Aim to invert the generative model

- **Posterior distribution on histories**

(Distribution over generated jet)

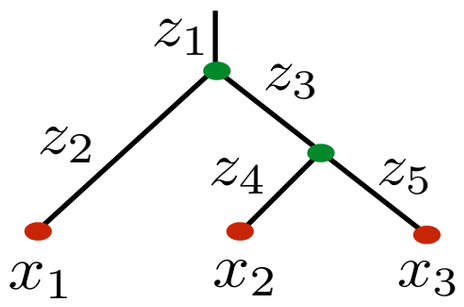
$$p(z|x,\theta) = \frac{p(x|z,\theta)\ p(z|\theta)}{p(x|\theta)}$$

Sampling with MCMC or probabilistic programing techniques ?

Latent Structure

z

x



- **Related work: Q-jets**
[Ellis, Hornig, Roy, Krohn & Schwartz '12]

(Distribution over reconstructed jet)

Assemble trees by sequentially choosing a pair with probability $\Omega_{ij} = \omega_{ij}/N$

$$\omega_{ij}^{(\alpha)} \equiv \exp\left(-\alpha \frac{(d_{ij} - d^{\min})}{d^{\min}}\right)$$

- **Related work: Shower deconstruction**
[Soper & Spannowsky '11] [Soper & Spannowsky '12]

  - Simplified shower model.
  - Add likelihood of each clustering history over *microjets* conditioned on S or B.
  - Define the likelihood ratio: $\chi(p_N) = \frac{P(p_N|S)}{P(p_N|B)}$

Greedy algorithms

Choose the optimal decision locally at each step.

Generalized kt clustering algorithms

- Locally minimize the distance measure d_{ij}
- An analogue of the generalized k_t clustering algorithms can be defined for Ginkgo jets.

Greedy likelihood-based algorithm

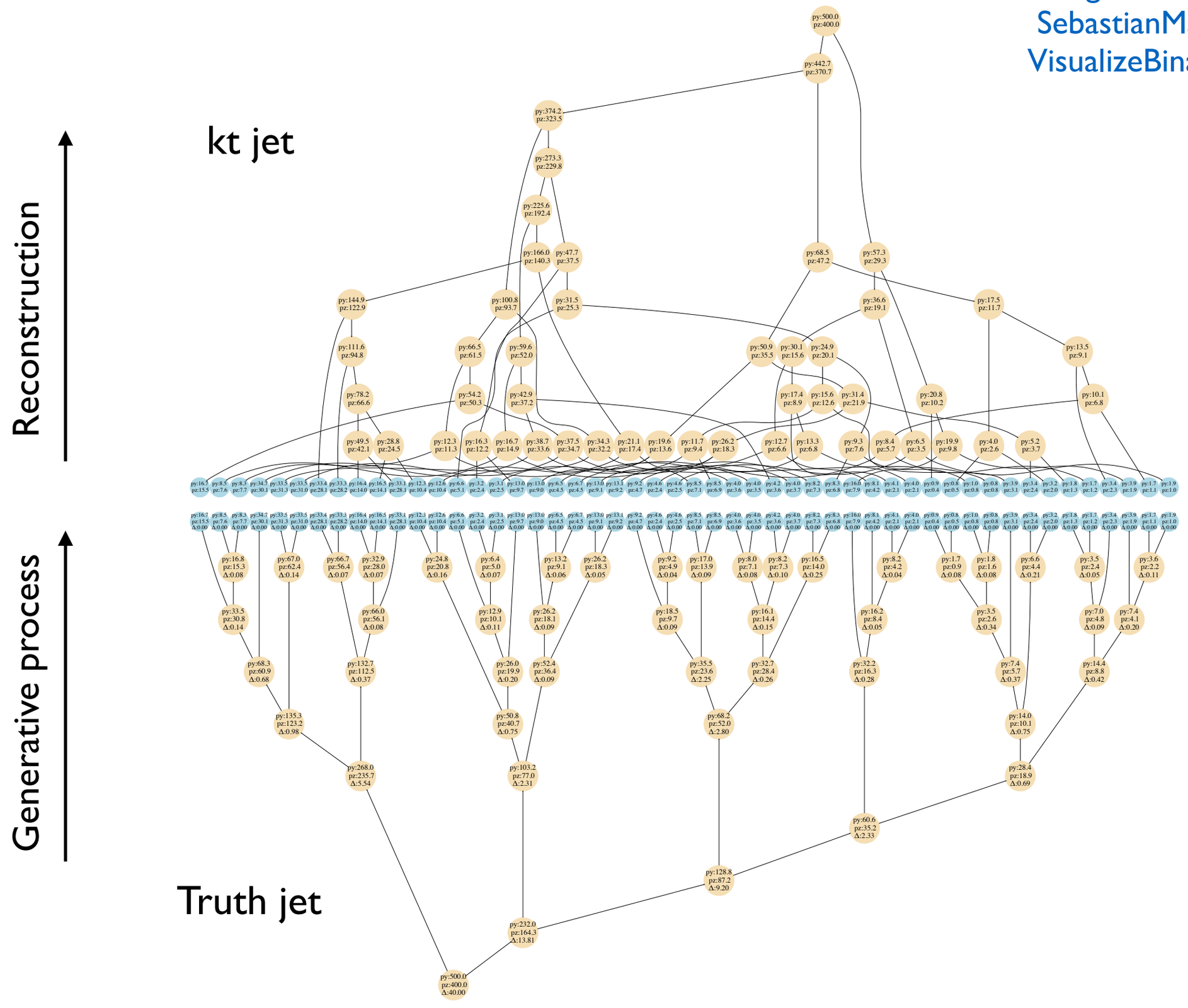
- Choose the pairing of nodes that locally maximizes the likelihood at each step.

Our implementation improves the N^3 brute force time complexity to N^2.
Approach based on nearest neighbors introduced in FastJet
[hep-ph/0512210, G. Salam and M. Cacciari LPTHE-06-02].

1-D binary tree visualizations

Kyle Cranmer, SM &
Duccio Pappadopulo

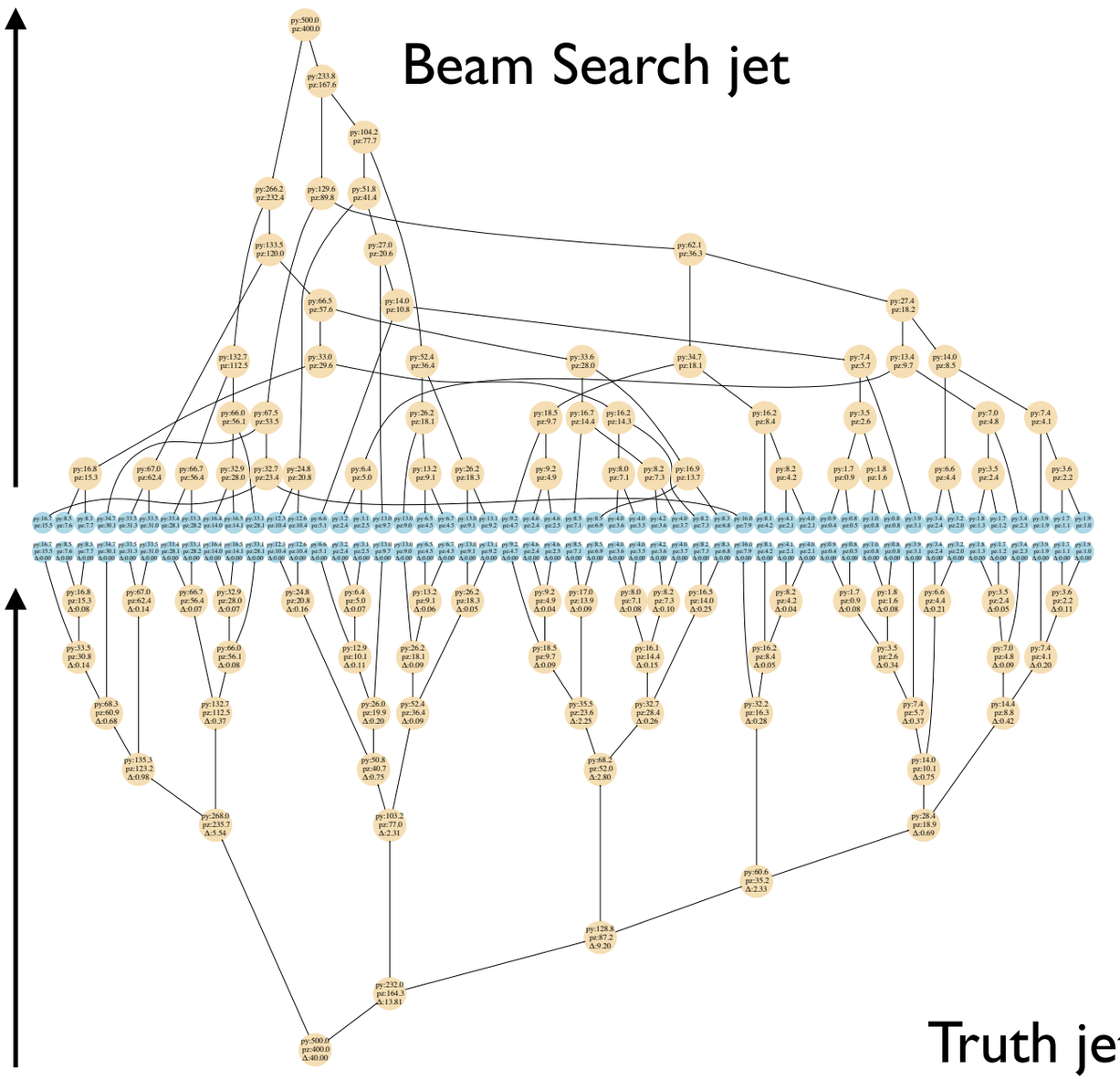
github.com/
SebastianMacaluso/
VisualizeBinaryTrees

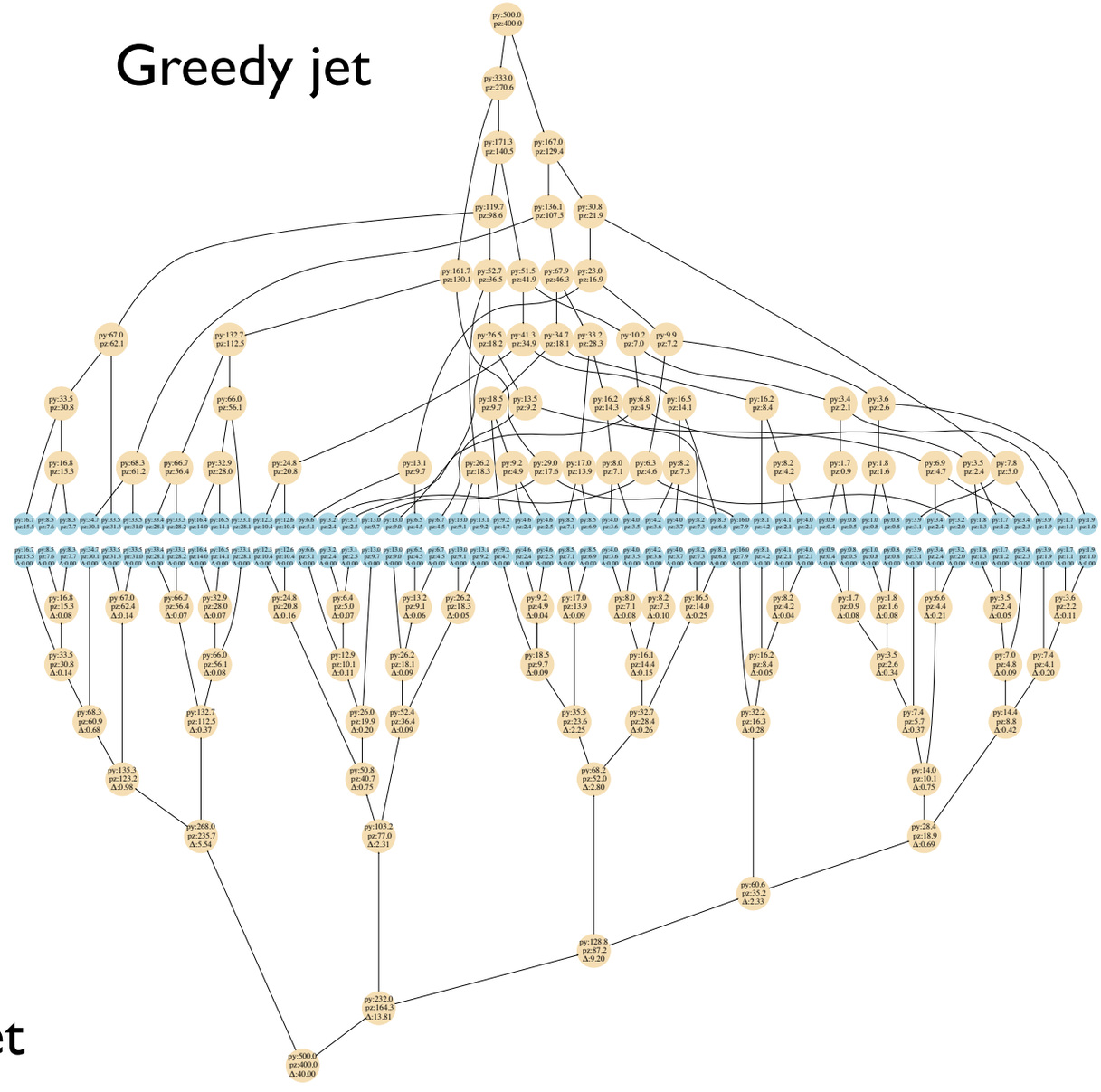


1-D binary tree visualizations

Reconstruction

Generative process



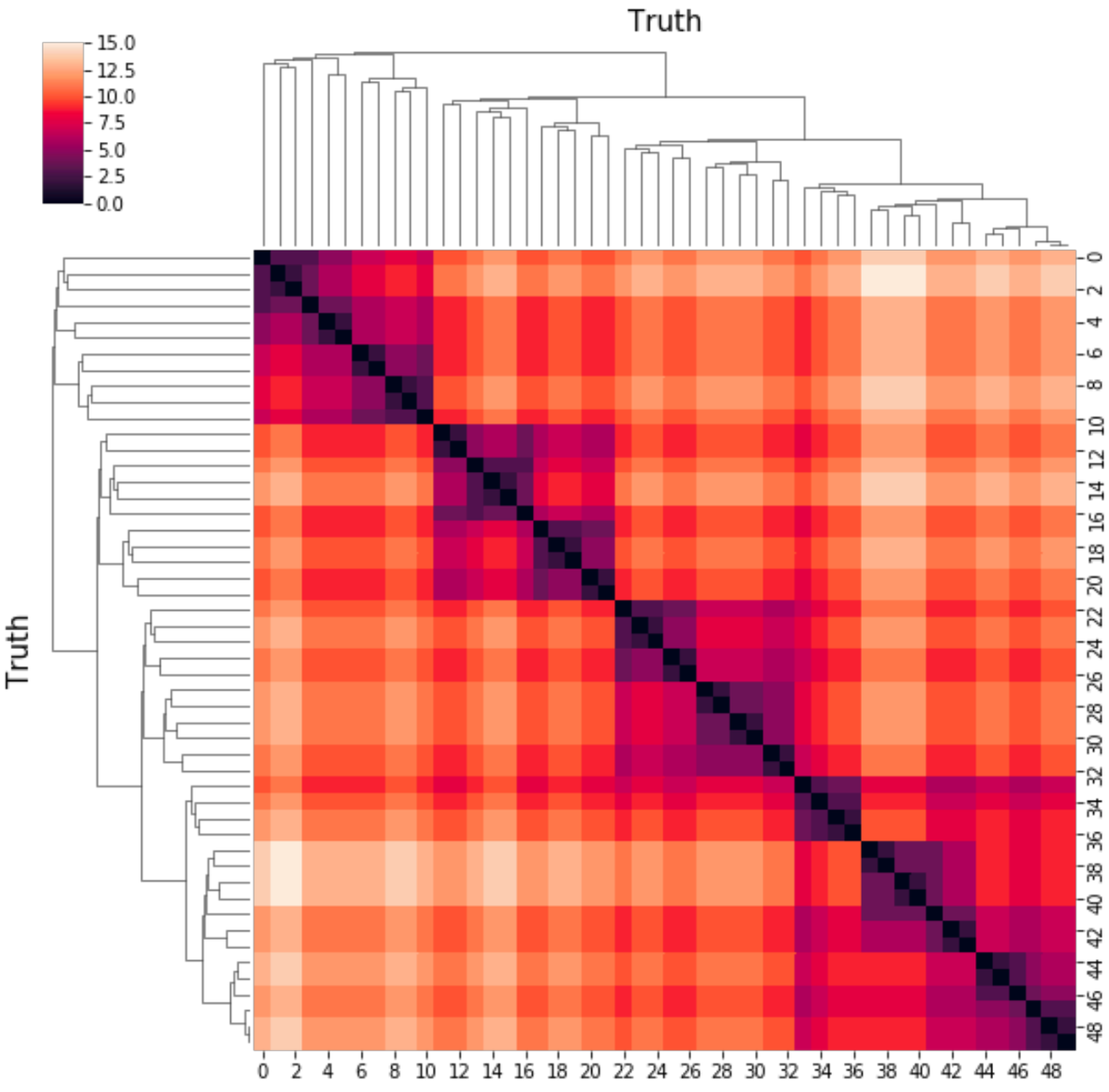


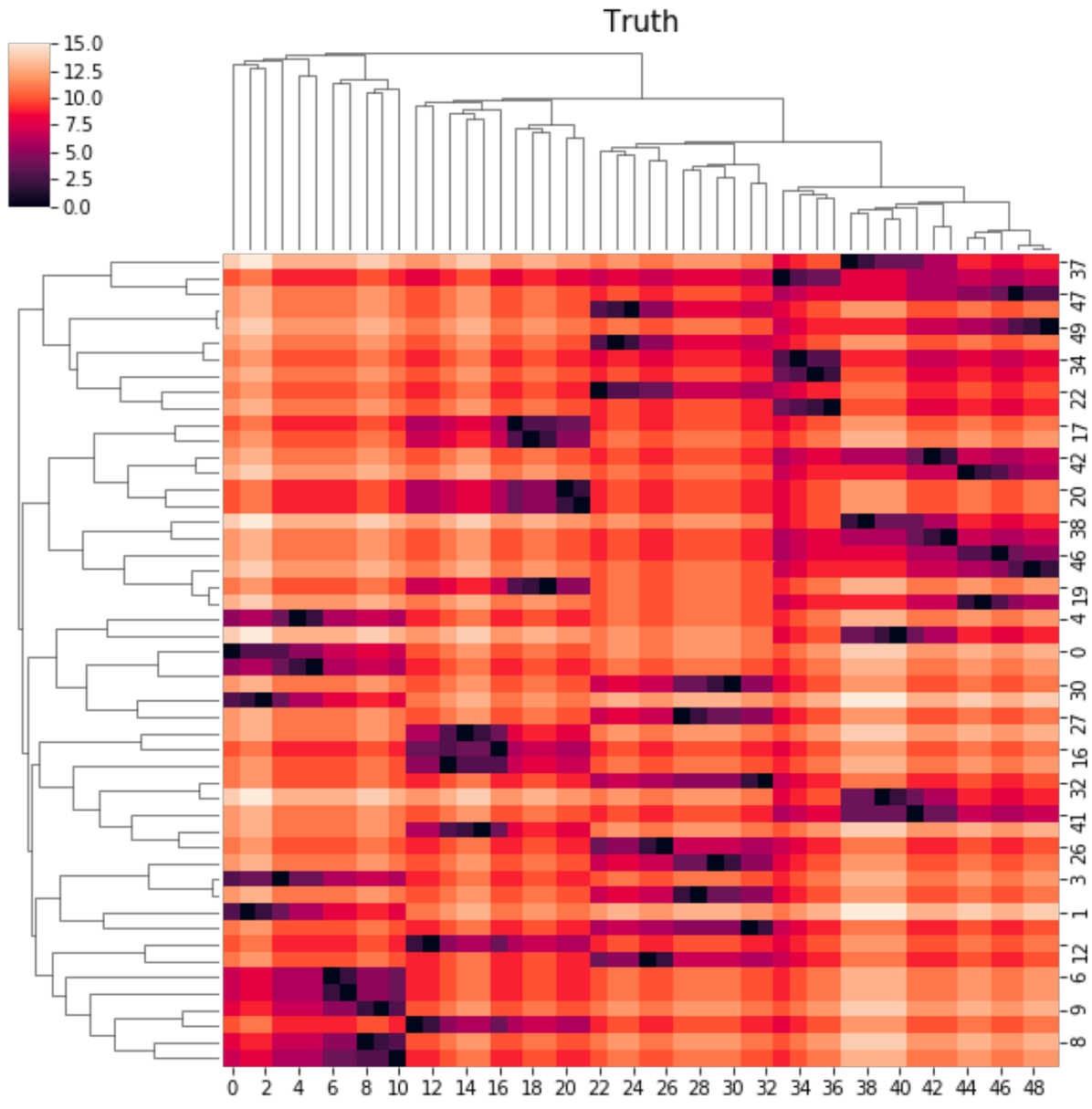
Truth jet

Heat cluster maps

Kyle Cranmer, SM &
Duccio Pappadopulo

github.com/SebastianMacaluso/VisualizeBinaryTrees

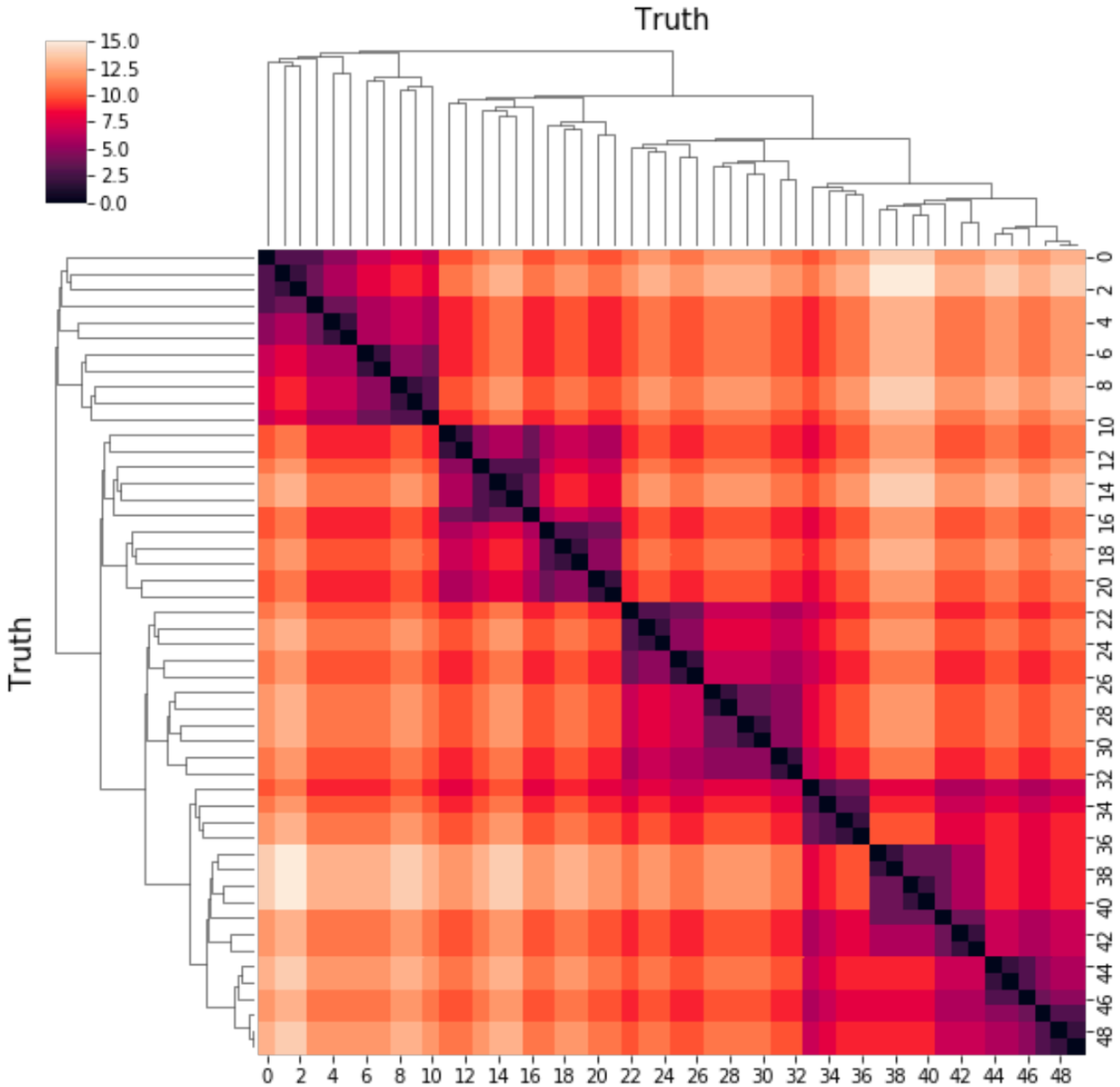


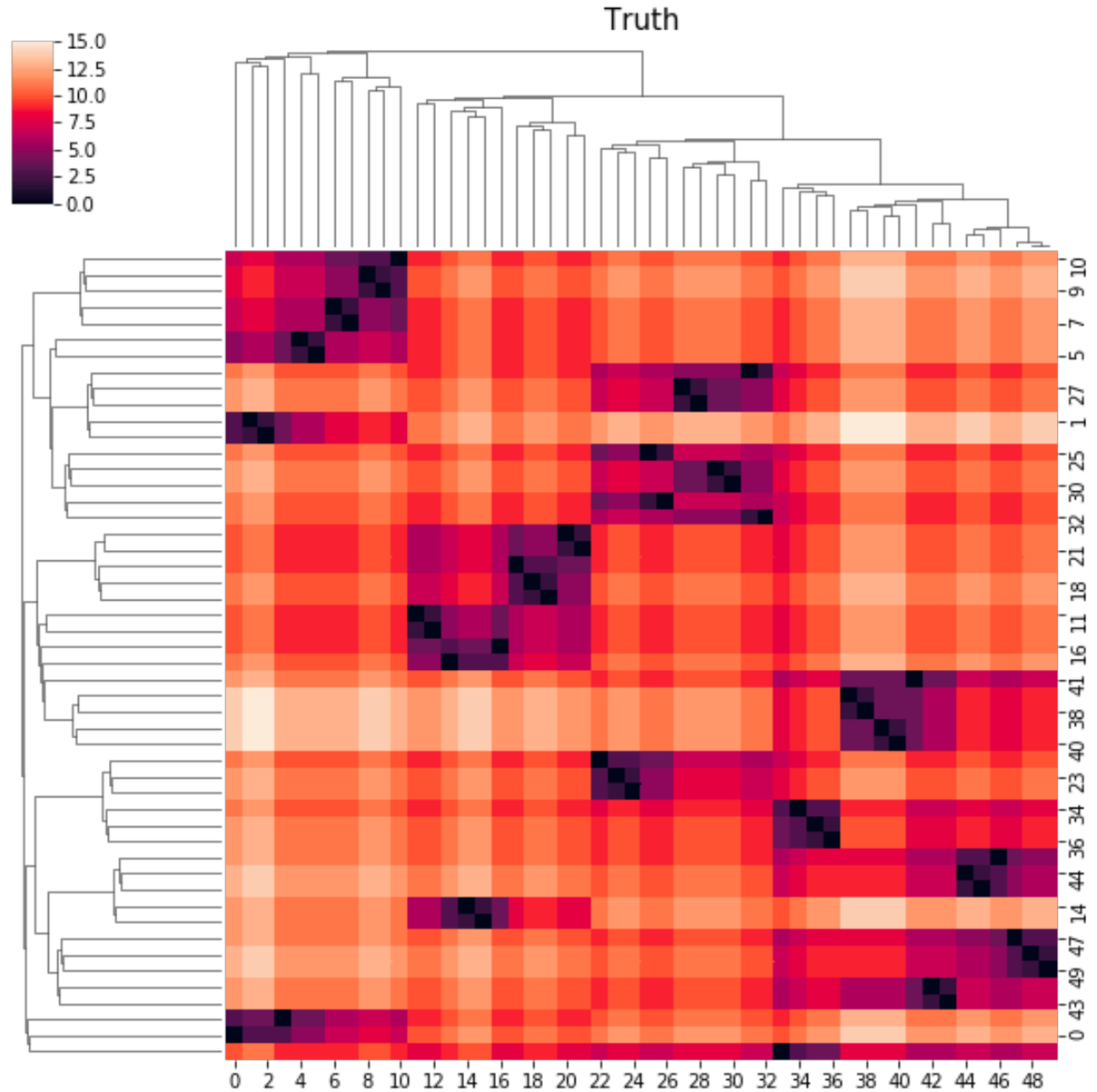


Heat cluster maps

Kyle Cranmer, SM &
Duccio Pappadopulo

github.com/SebastianMacaluso/VisualizeBinaryTrees





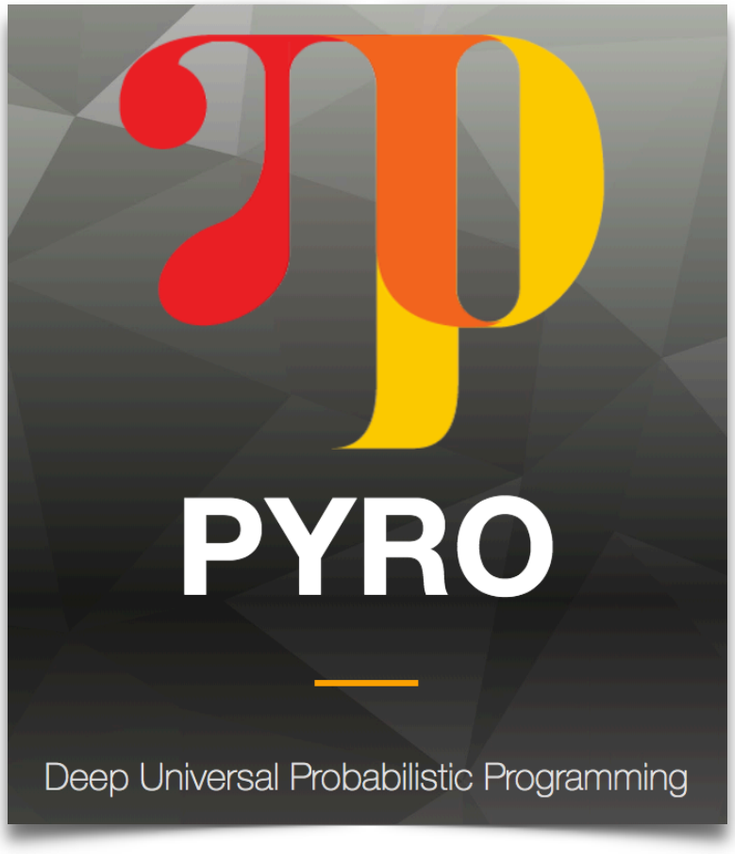
Augmented data

Additional information that characterizes the latent process can be extracted in Ginkgo:

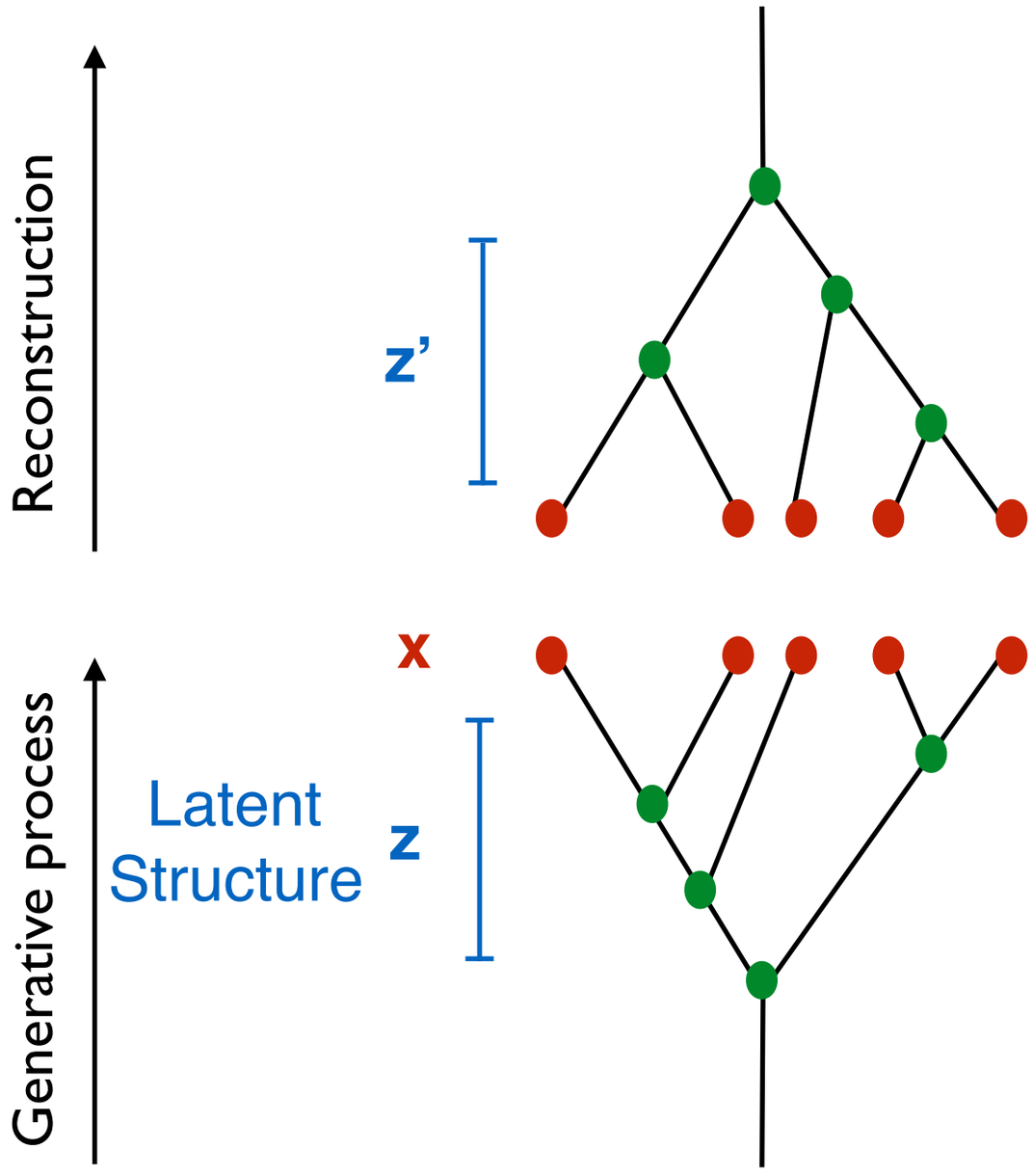
- Joint likelihood
- Joint likelihood ratio
- Joint score

The model keeps track of the augmented data based on a **PYRO** implementation.

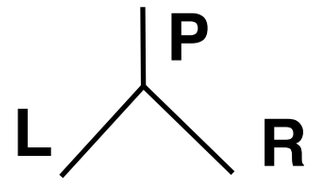
New techniques for likelihood-free inference can be applied.
[J. Brehmer, G. Louppe, J. Pavez & K. Cranmer ’18]



Jet clustering



Can we cluster jets based on the likelihood of each splitting?



Joint likelihood:

$$p(x, z|\theta) = \prod_i p(\mathrm{L}_i, \mathrm{R}_i|\mathrm{P}_i, \theta)$$

Split likelihood:

$$p(\mathrm{L}_i, \mathrm{R}_i|\mathrm{P}_i) \equiv p(\vec{p}_\mathrm{L}, \vec{p}_\mathrm{R}|\vec{p}_\mathrm{P})$$

---

k_t distance measure

$$d(\mathrm{L}_i, \mathrm{R}_i) \equiv d(\vec{p}_\mathrm{L}, \vec{p}_\mathrm{R})$$