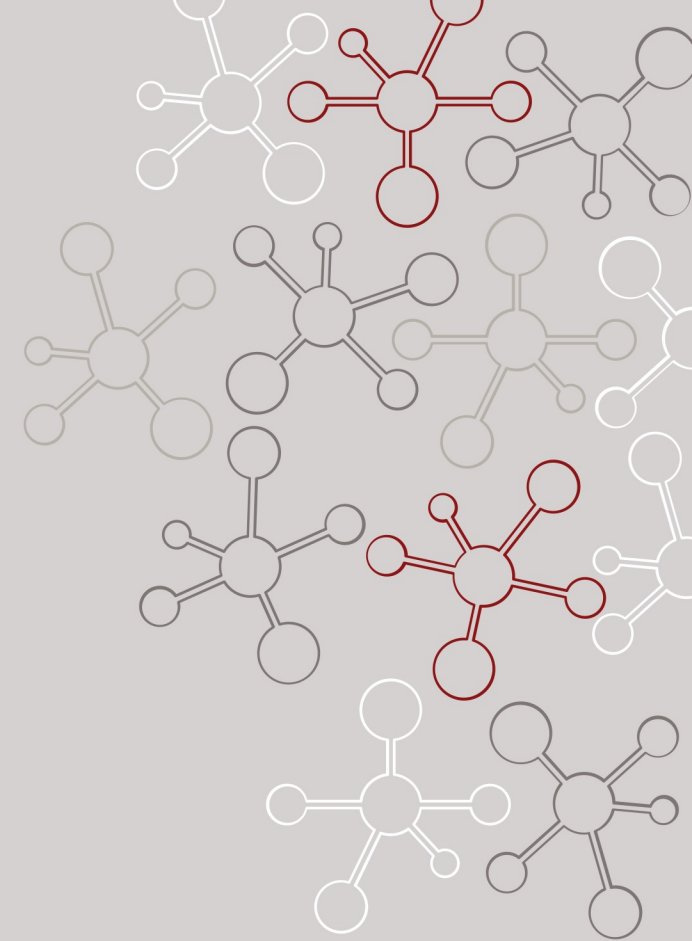


Neural Empirical Bayes

Source Distribution Estimation and its Applications to
Simulation-Based Inference

Maxime Vandegar, Michael Kagan, Antoine Wehenkel & Gilles Louppe

Learning to Discover 2022



Motivation

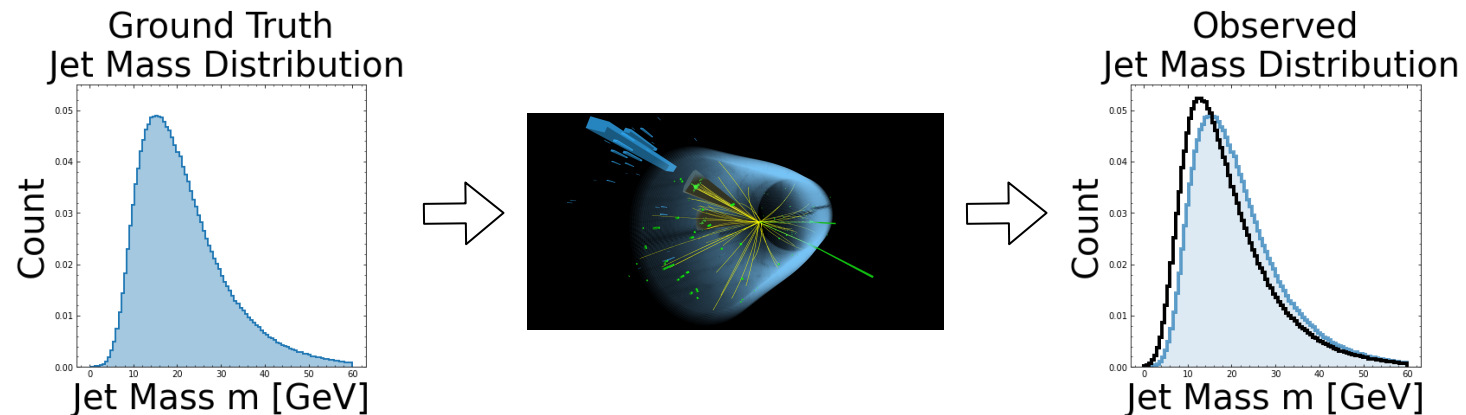
Removing detector effects enables comparison between experiments and theory

- Known as **unfolding** in HEP

Usually done with binned histograms, in 1 or 2 dimensions

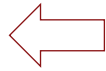
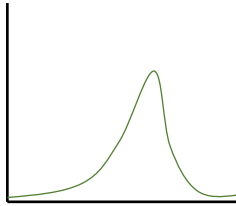
- Loss of information
- Does not capture correlations between variables

Motivates the development of **high-dimensional** and **continuous** methods

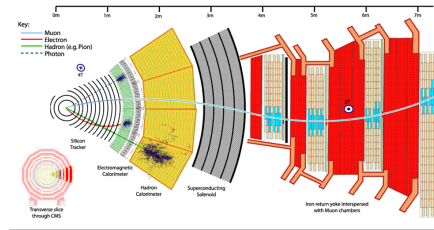


Histogram Approach

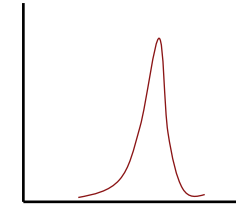
$y \sim p(y)$ = observed distribution



$p(y|x)$ = detector smearing

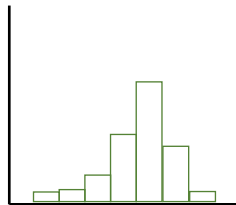


$x \sim p(x)$ = true distribution

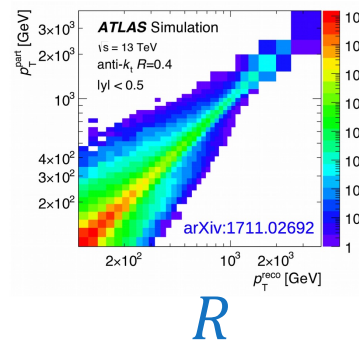


$$p(y) = \int p(y|x)p(x)dx$$

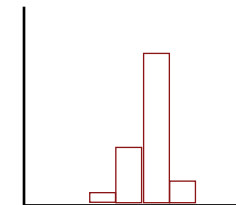
Usually in HEP: solve discrete linear inverse problem: $b = Ra$



b



R



a

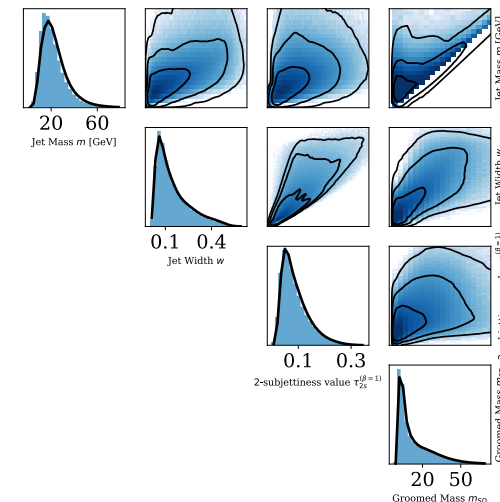
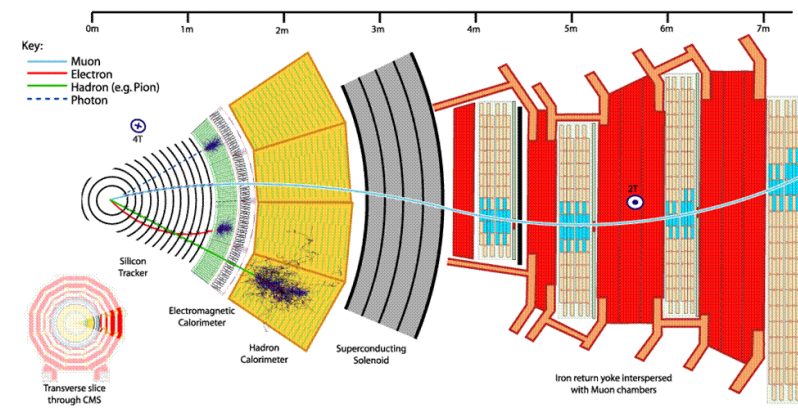
Maximum marginal likelihood

Multidimensional integral

$$\arg \max_{\theta} p(y) = \int p(y|x) p_{\theta}(x) dx$$

Likelihood function (usually not known)

Target distribution



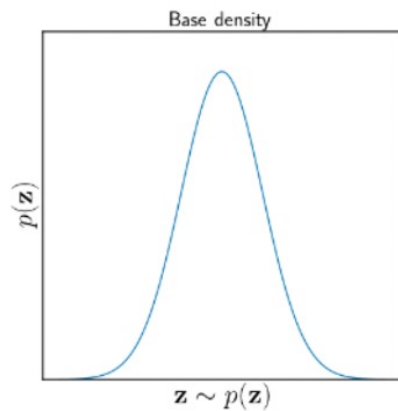
1) Source distribution

Need to model the source distribution with a parametric model $p_{\theta}(x) \in \mathcal{Q}$

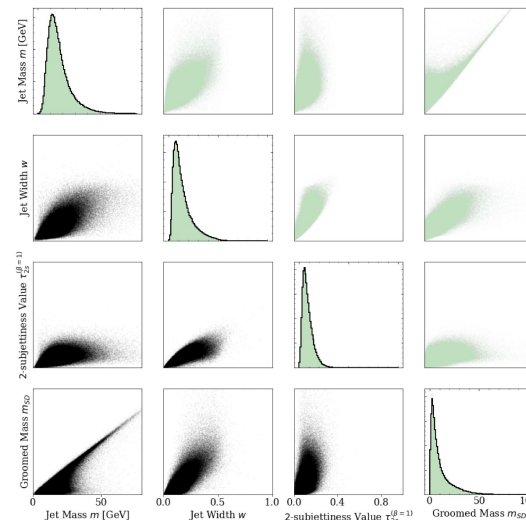
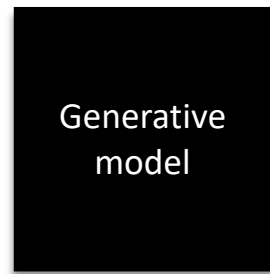
Aim: find θ^* that best explain the data

The distribution family \mathcal{Q} should be flexible enough to model the target distribution

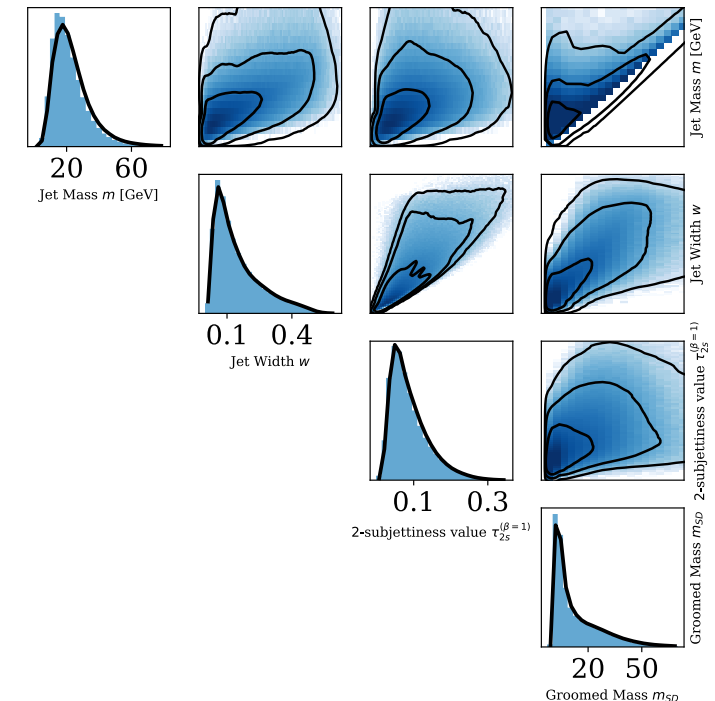
- We use Neural Networks (Deep Generative Models)
 - Efficient at sampling (more on that later)



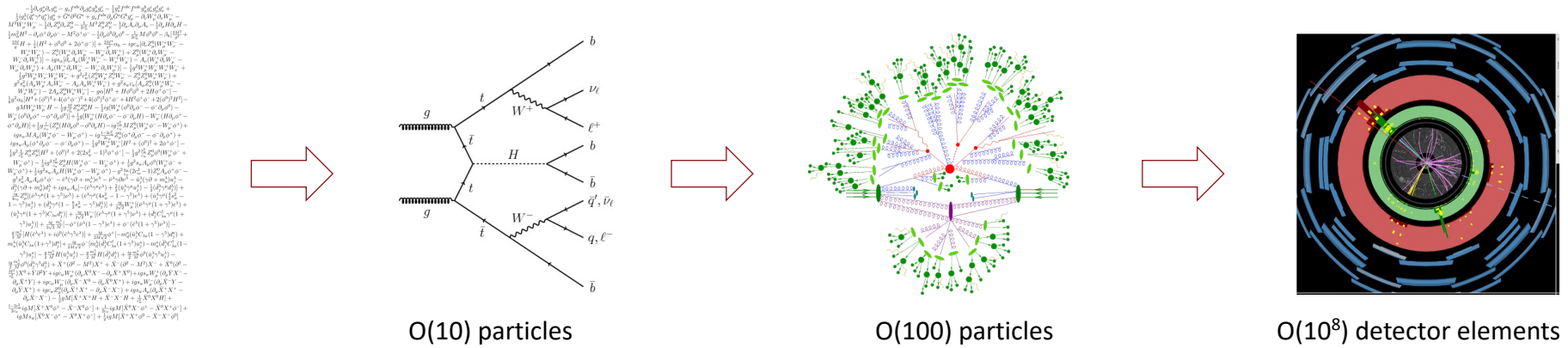
Random noise



Density $p_{\theta}(x)$ defined by the model



2) Likelihood function



Usually in HEP, the likelihood function $p(y|x)$ is unknown

But we can simulate this process

- Mechanistic understanding of interactions, put into code

We use the simulator to learn a **proxy of the likelihood function**

1. Generate a dataset of pairs of parameters and observations $\{x_i, y_i\}_{i=1}^M$
2. Fit a **density estimator** to the generated data (we use normalizing flows)

3) Approximate integral with Monte Carlo Integration

$$\log \int p(y|x)p_{\theta}(x)dx$$
$$\approx \log \sum_i p(y_k|x_i), \quad x_i \sim p_{\theta}(x)$$

Monte Carlo approximation of integrals

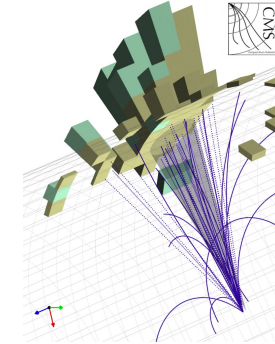
Efficient

- Sampling from learned source $p_{\theta}(x)$ is cheap
- Evaluating learned $p(y|x)$ is cheap
- $p_{\theta}(x)$ and $p(y|x)$ are Neural Networks \rightarrow computations can be parallelized on GPUs

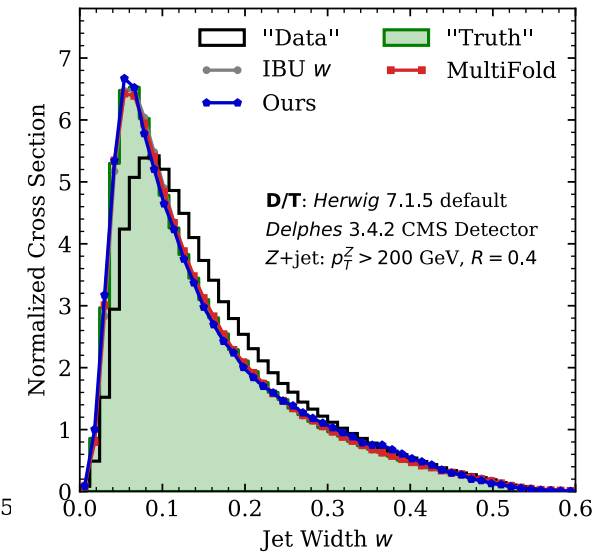
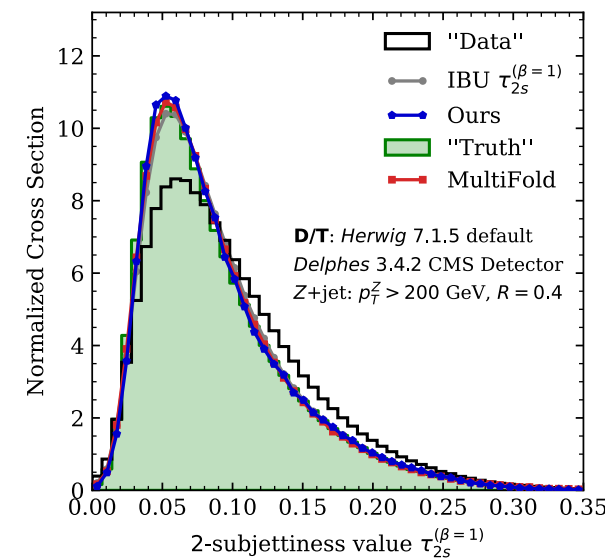
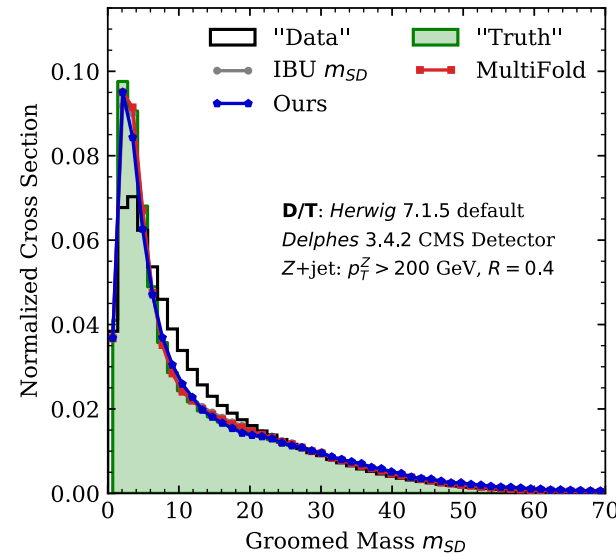
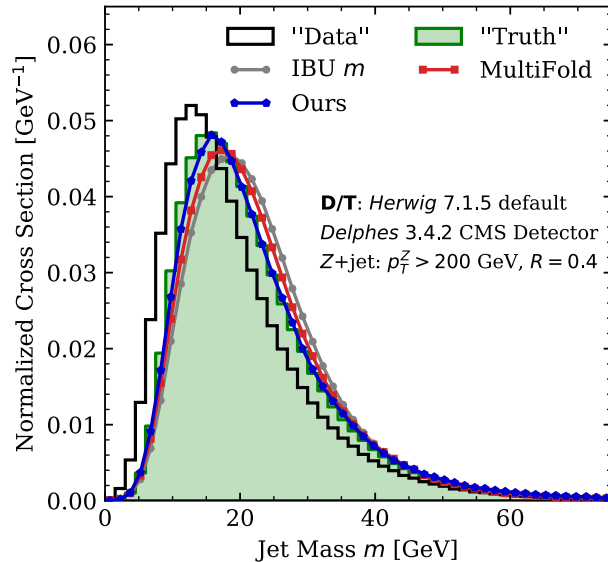
Unfolding Jet Variables in Z+jet Events at the LHC

Source data have been generated with Herwig simulator and then corrupted with Delphes simulator

We use Neural Empirical Bayes (NEB) to retrieve "Truth" jet properties given corrupted samples "Data"



Jet: stream of particles produced by high energy quarks and gluons

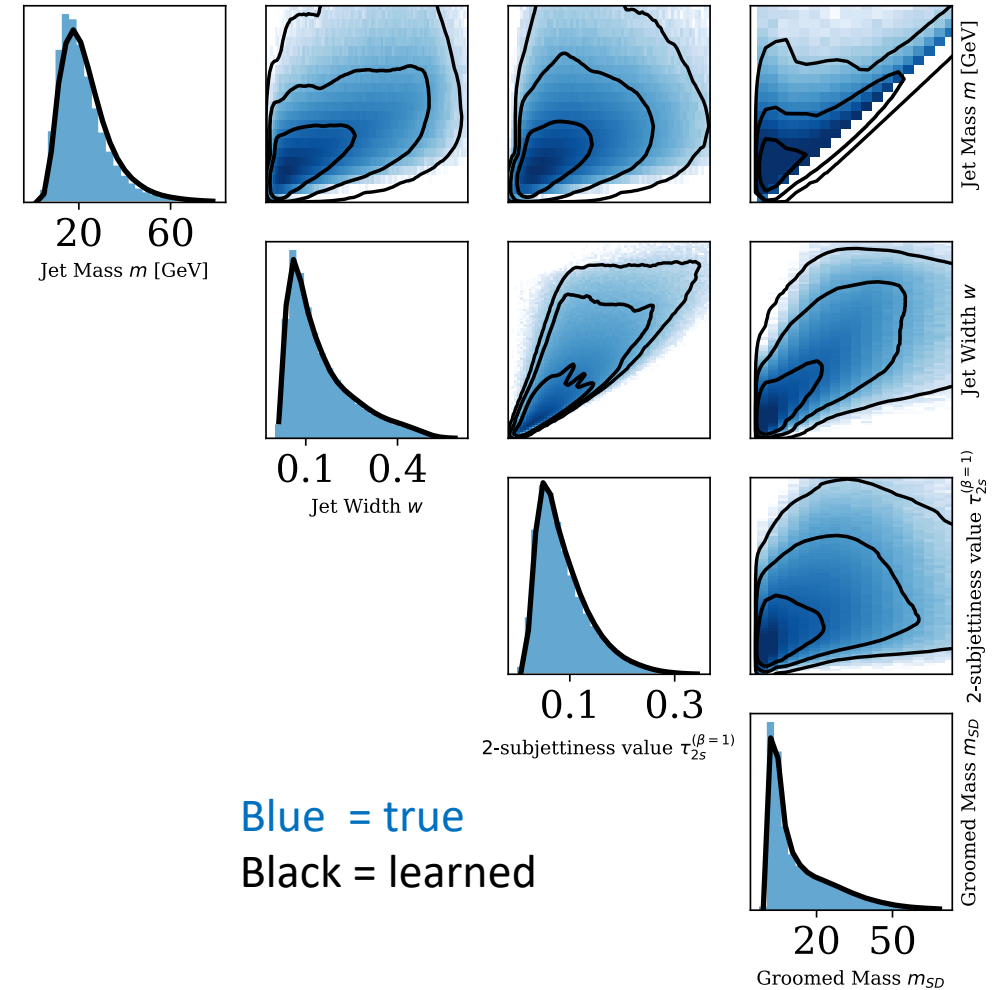


Unfolding Jet Variables in Z+jet Events at the LHC

NEB learns the **multidimensional** distribution

NEB enables **efficient sampling** from the learned distribution

With specific Neural Network architectures, the learned density can be evaluated and differentiated (more on than in the paper)



Posterior Estimation in Z+Jets

Once learned, $p_\theta(x)$ enables reconstruction with estimate of uncertainty

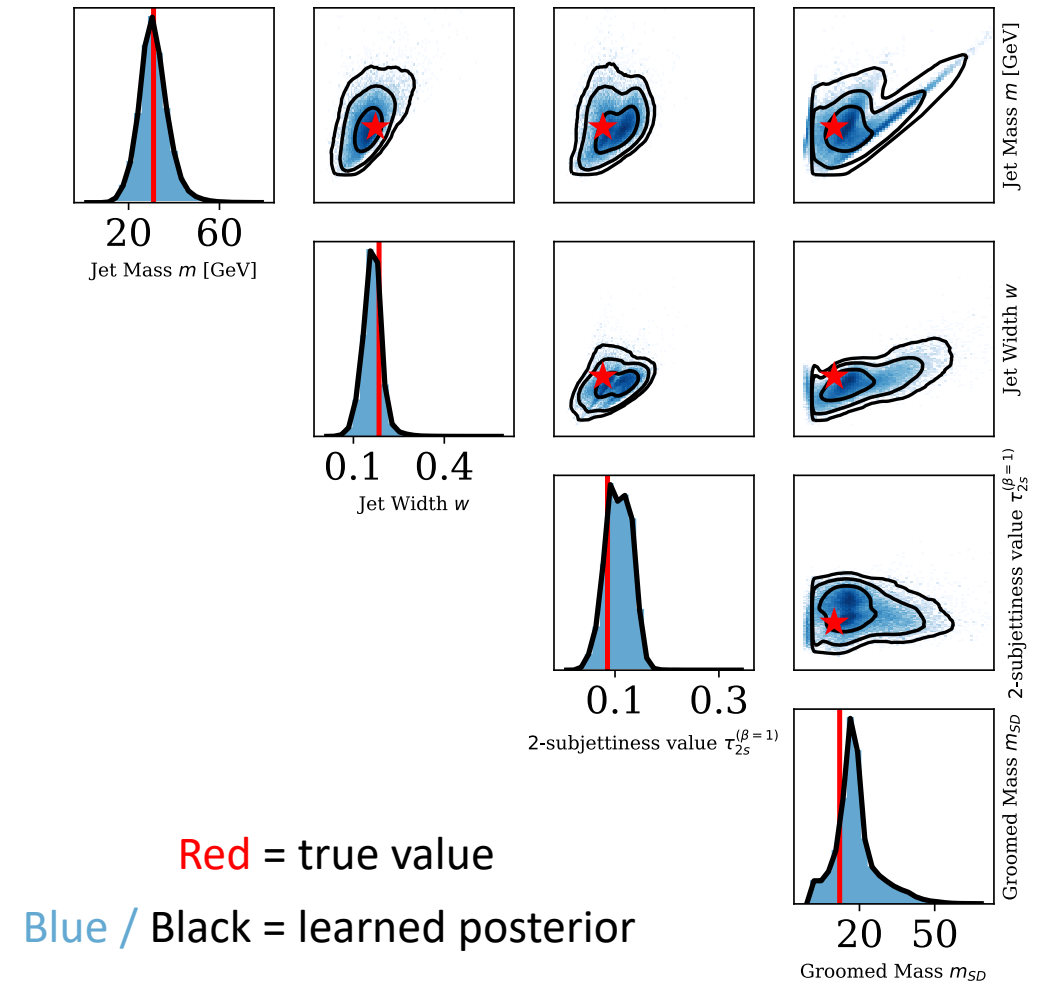
Which value x has generated a new observation y_{obs} ?

Posterior: $p(x|y_{obs}) \propto p(y_{obs}|x)p_\theta(x)$

- Distribution of true values given observation

Method

1. Observation y_{obs}
2. Sample $x \sim p_\theta(x)$
3. Rejection sampling: keep x w/ prob. $\propto p(y|x)$



Conclusion

NEB enables to unfold continuous and multidimensional distributions

After training, the learned model can be heavily sampled from, which is useful for reconstruction

When combined with specific Neural Network architectures, NEB enables density evaluation

Inductive bias helps mitigate the ill-posed nature of problems, and is easily introduced in the models

