Enhancing The Efficiency of Event Generation with MCMC and Machine Learning Techniques

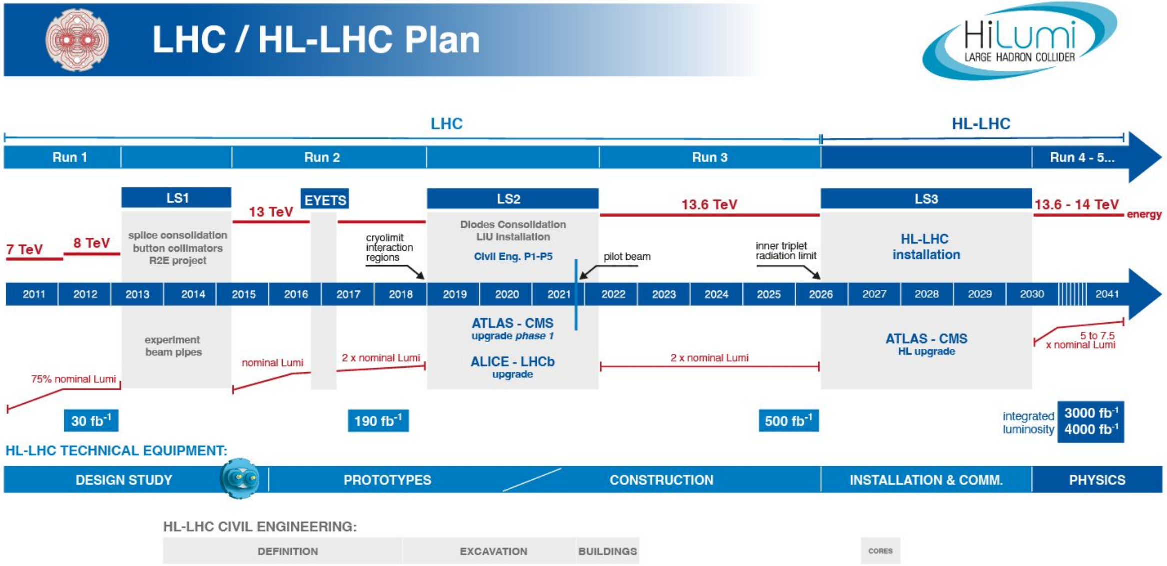
By **Niccolò Tonin**
supervised by prof. **Cornelius Grunwald**
and reviewed by prof. **Kevin Kröninger**

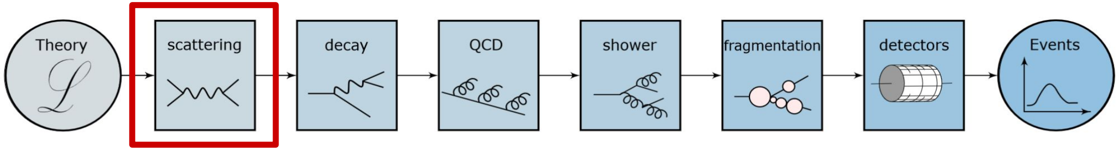
technische universität dortmund

UNIVERSITÉ Clermont Auvergne

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

- It is estimated that the High Luminosity LHC will produce an order of magnitude more data than LHC
- Simulated data will need to be generated at a similar quantity
- Simulated data will be more complex and multimodal

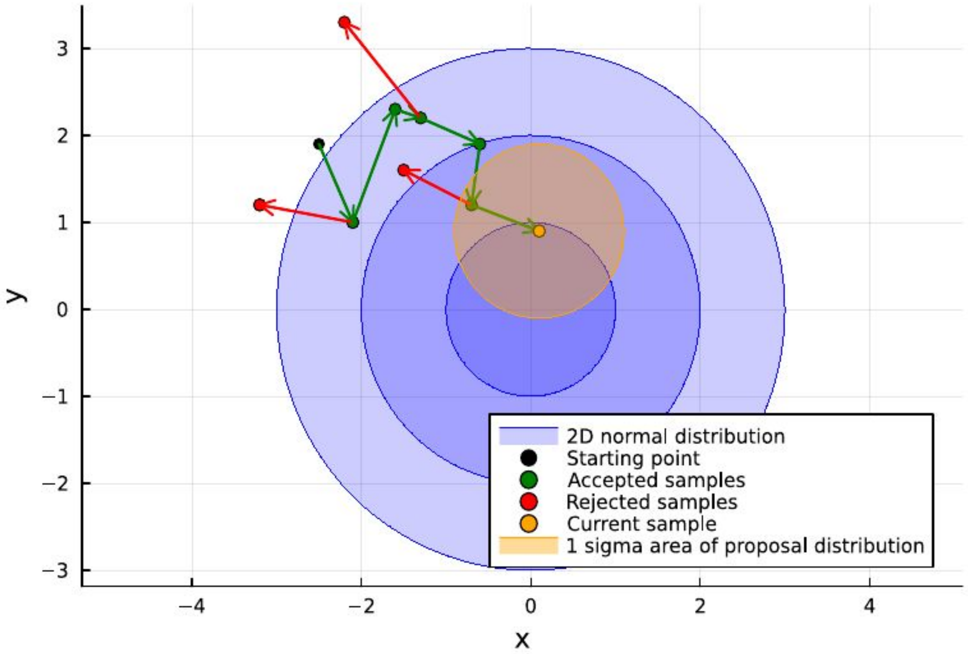




$$\sigma_{pp \to X_n} = \sum_{ab} \int \mathrm{d}x_a \mathrm{d}x_b \, \mathrm{d}\Phi_n \; f_a(x_a, \mu_F^2) f_b(x_b, \mu_F^2) \, |\mathcal{M}_{ab \to X_n}|^2 \, \Theta_n(p_1, \ldots, p_n)$$

2

Markov Chain Monte Carlo



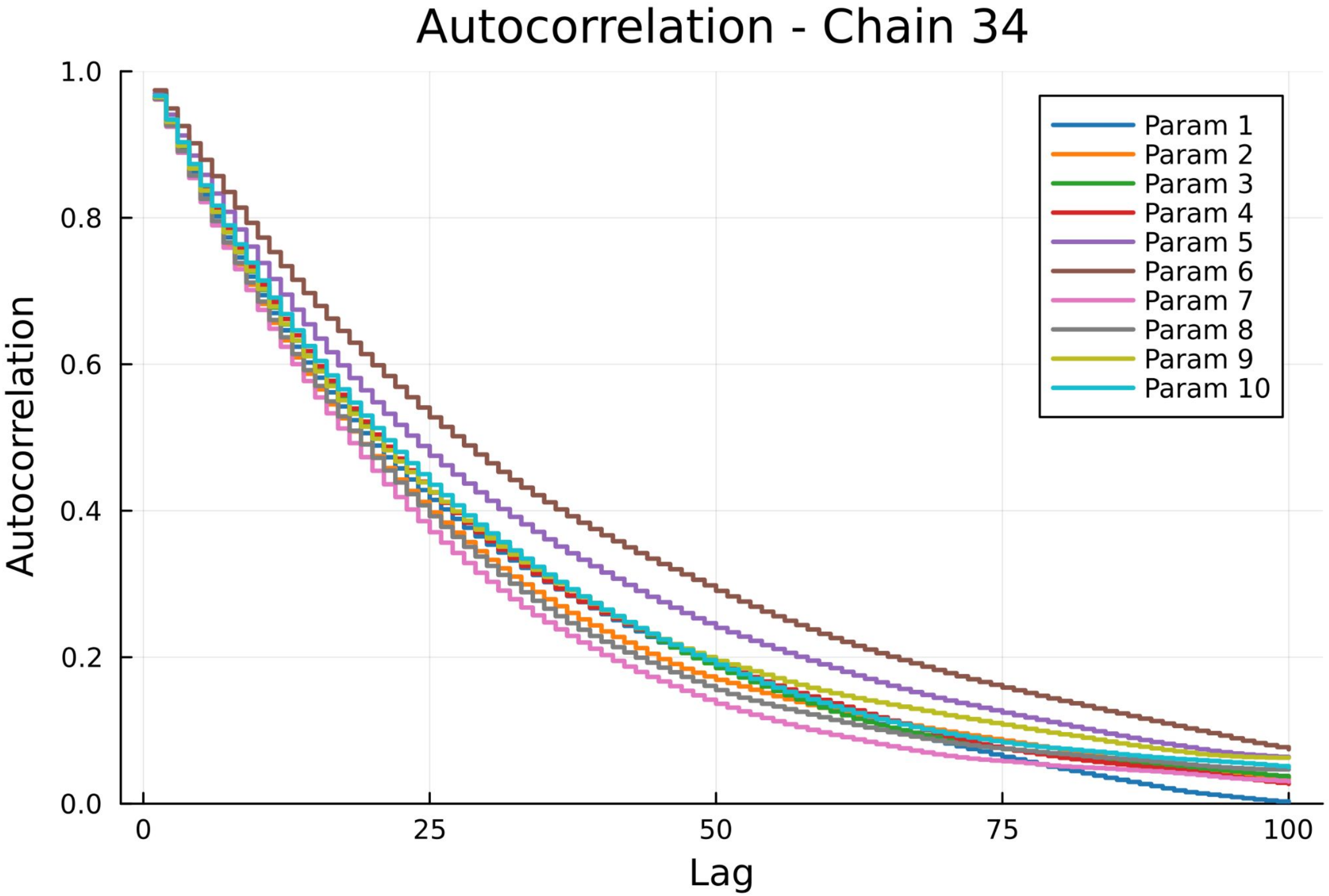
Markov chains are defined by

$$P(x_n|x_{n-1}, x_{x-2}, \cdots, x_1) = P(x_n|x_{n-1})$$

Distributions can be sampled using the Metropolis Hastings algorithm

$$P_{\text{accept}} = \min\left(1, \frac{\pi(x')}{\pi(x_i)} \frac{g(x_i|x')}{g(x'|x_i)}\right)$$

Image from: Willy Herbert Karl Weber. “Walking with the Flow: Normalizing Flow enhanced MCMC Sampling in BAT.Journal”. Master’s thesis.


Autocorrelation - Chain 34
Autocorrelation
Lag
1.0
0.8
0.6
0.4
0.2
0.0
0
25
50
75
100
Param 1
Param 2
Param 3
Param 4
Param 5
Param 6
Param 7
Param 8
Param 9
Param 10

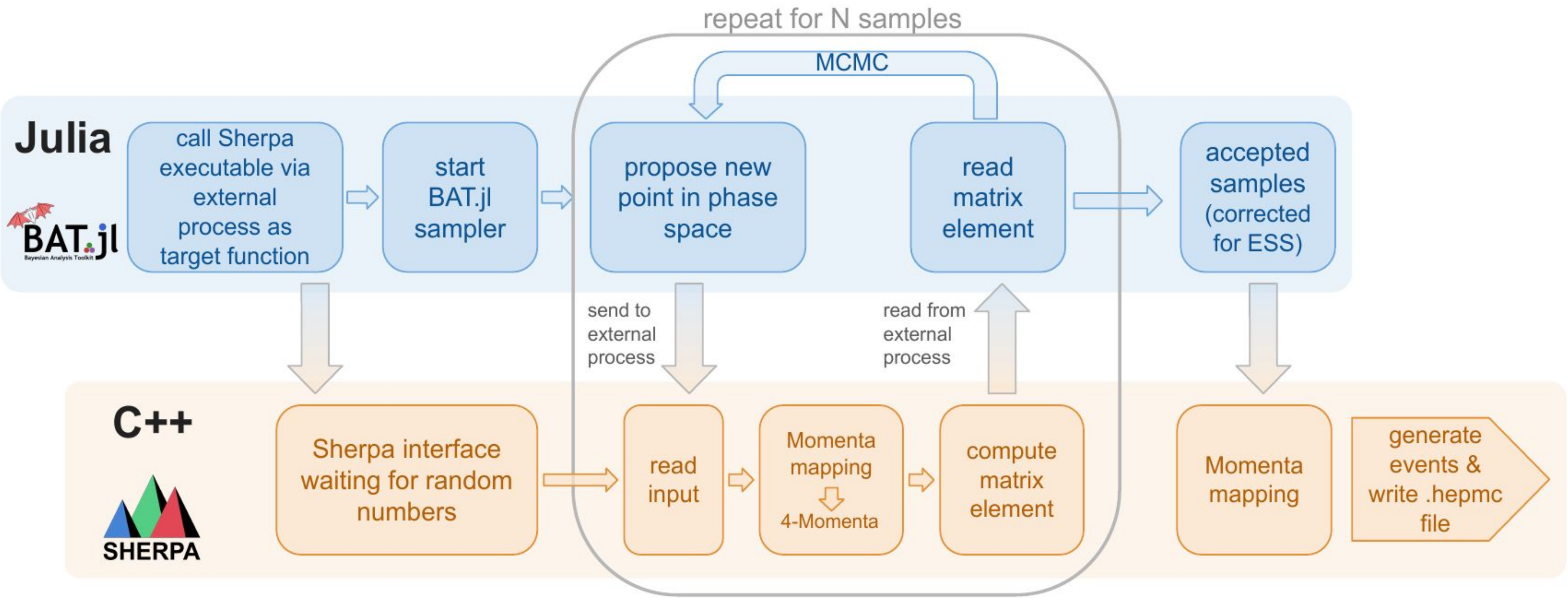
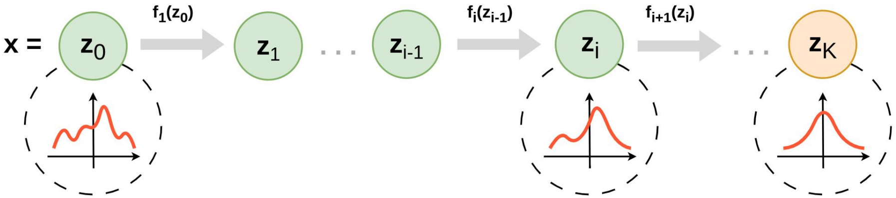
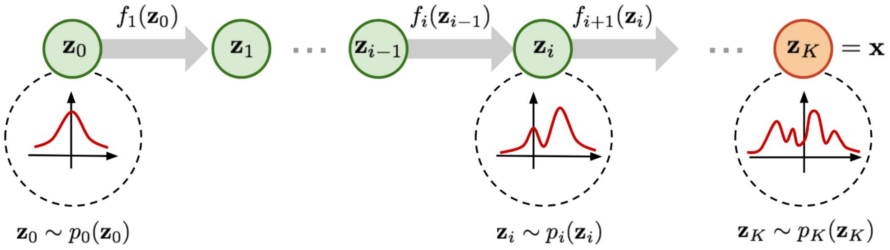


Image from: Salvatore La Cagnina et al. Phase space sampling with Markov Chain Monte Carlo methods

Normalizing flows





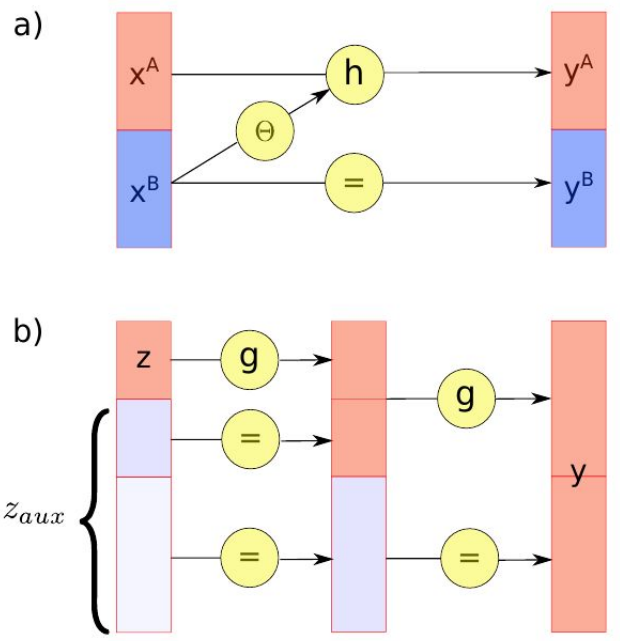
A **normalizing flow** is an *invertible* and *tractable* function that transforms a distribution into a simpler one, usually a normal distribution

$$
\begin{aligned}
p_{\mathbf{Y}}(\mathbf{y}) &= p_{\mathbf{Z}}(\mathbf{f}(\mathbf{y}))\,|\det \mathrm{D}\mathbf{f}(\mathbf{y})| \\
&= p_{\mathbf{Z}}(\mathbf{f}(\mathbf{y}))\,|\det \mathrm{D}\mathbf{g}(\mathbf{f}(\mathbf{y}))|^{-1}
\end{aligned}
$$

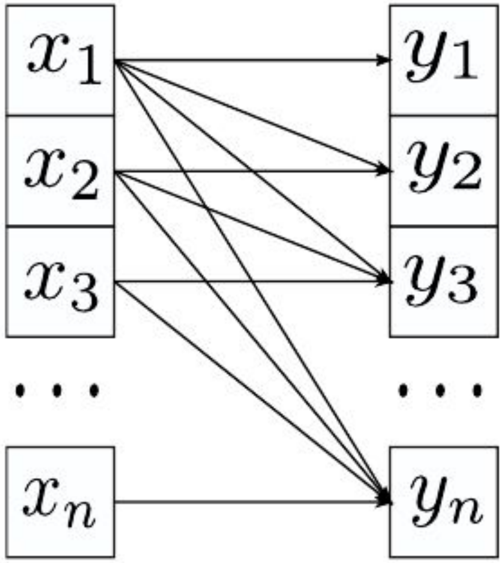
$$
\begin{aligned}
\log p(\mathcal{D}|\Theta) &= \sum_{i=1}^{M} \log p_{\mathbf{Y}}(\mathbf{y}^{(i)}|\Theta) \\
&= \sum_{i=1}^{M} \log p_{\mathbf{Z}}(\mathbf{f}(\mathbf{y}^{(i)}|\theta)|\phi) + \log \left|\det \mathrm{D}\mathbf{f}(\mathbf{y}^{(i)}|\theta)\right|
\end{aligned}
$$

The two most common architectures for normalizing flows are

Coupling

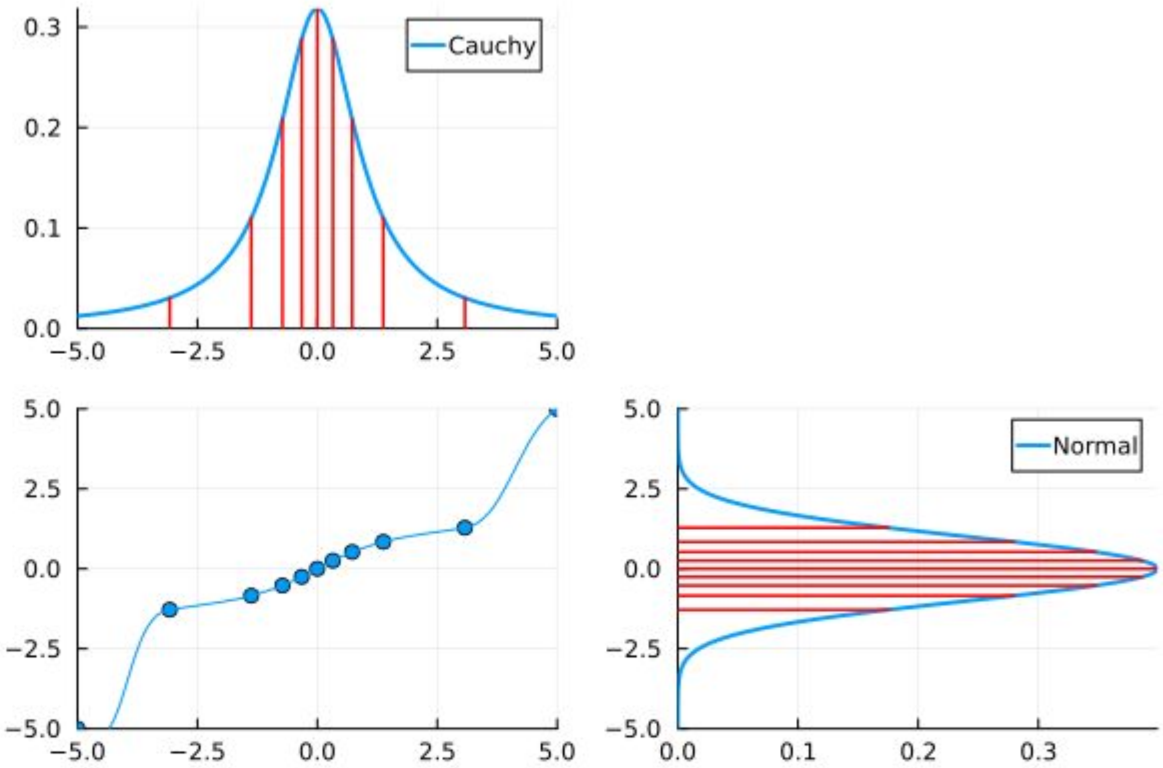


Autoregressive



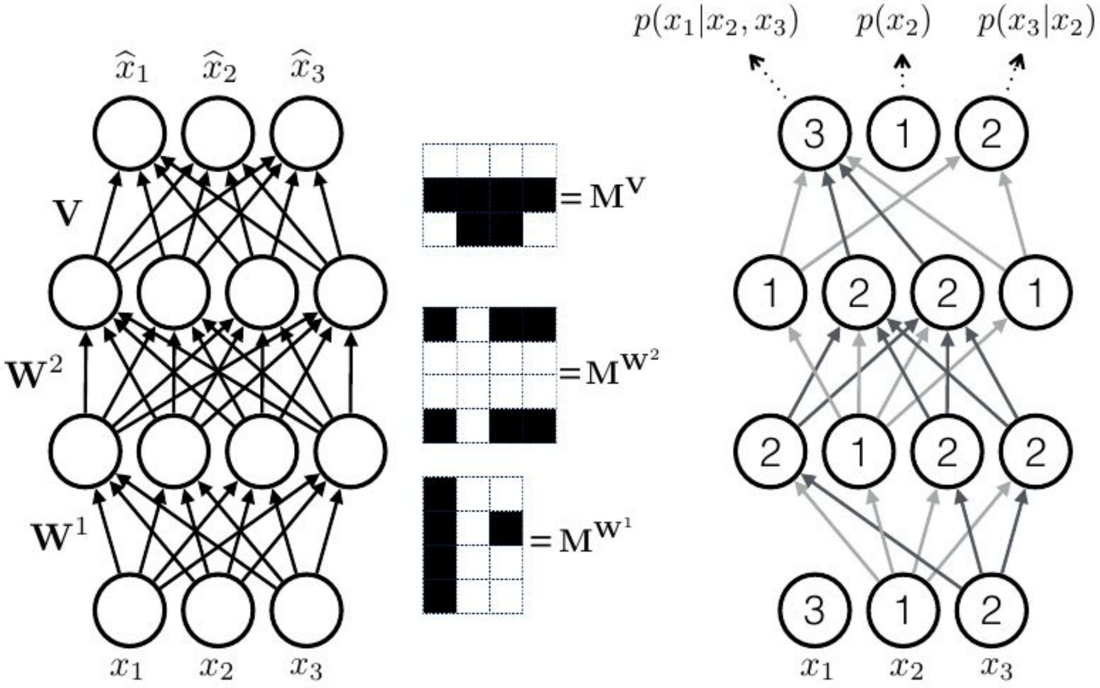
Images from: Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. “Normalizing Flows: An Introduction and Review of Current Methods”

Quantile Mapping Flows



Monotonic Rational Quadratic Splines are used to map the quantiles of each marginal distribution

Masked Autoregressive Flows

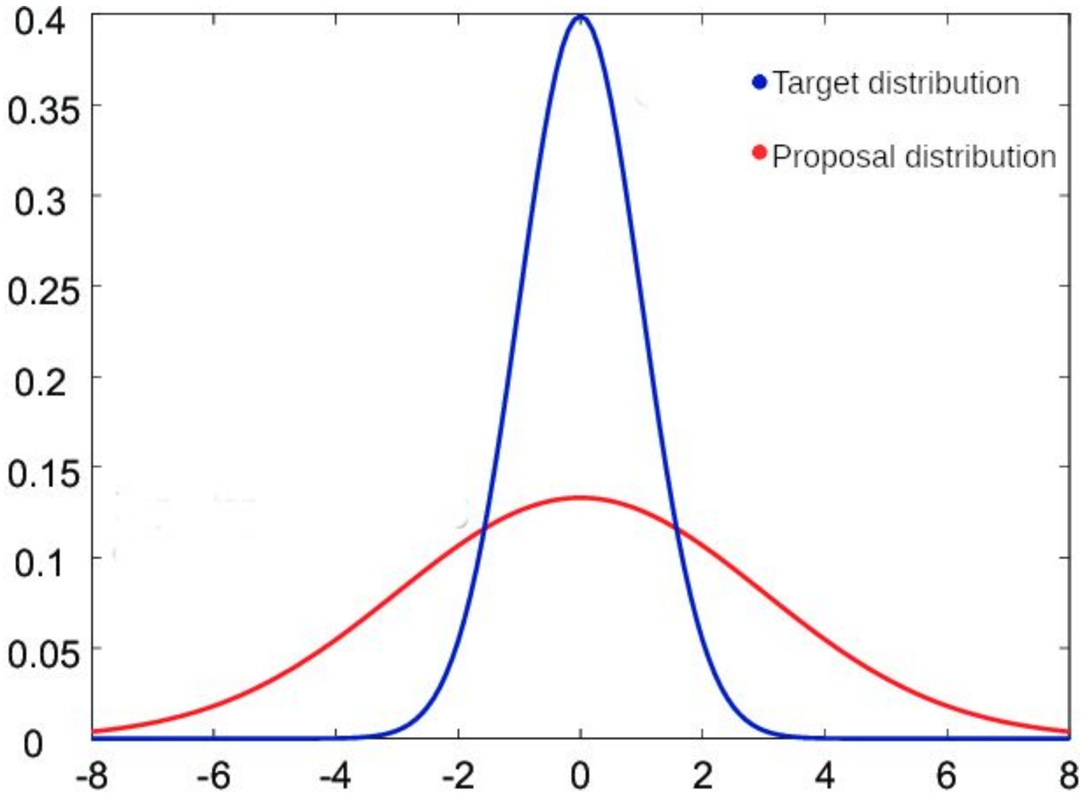


A mask is created to enforce the autoregressive architecture on a neural network

Image from: George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation

Importance sampling

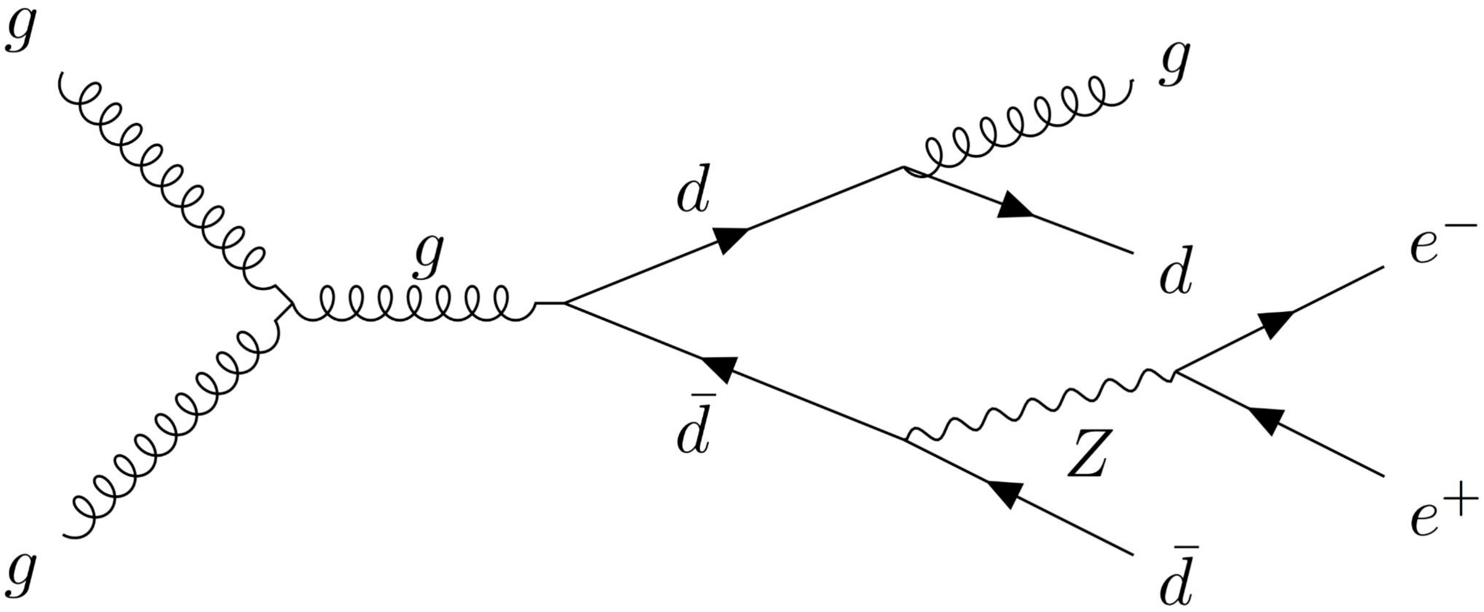
$$\int p(x)dx = \int q(x)\left[\frac{p(x)}{q(x)}\right]dx = \int q(x)w(x)dx$$



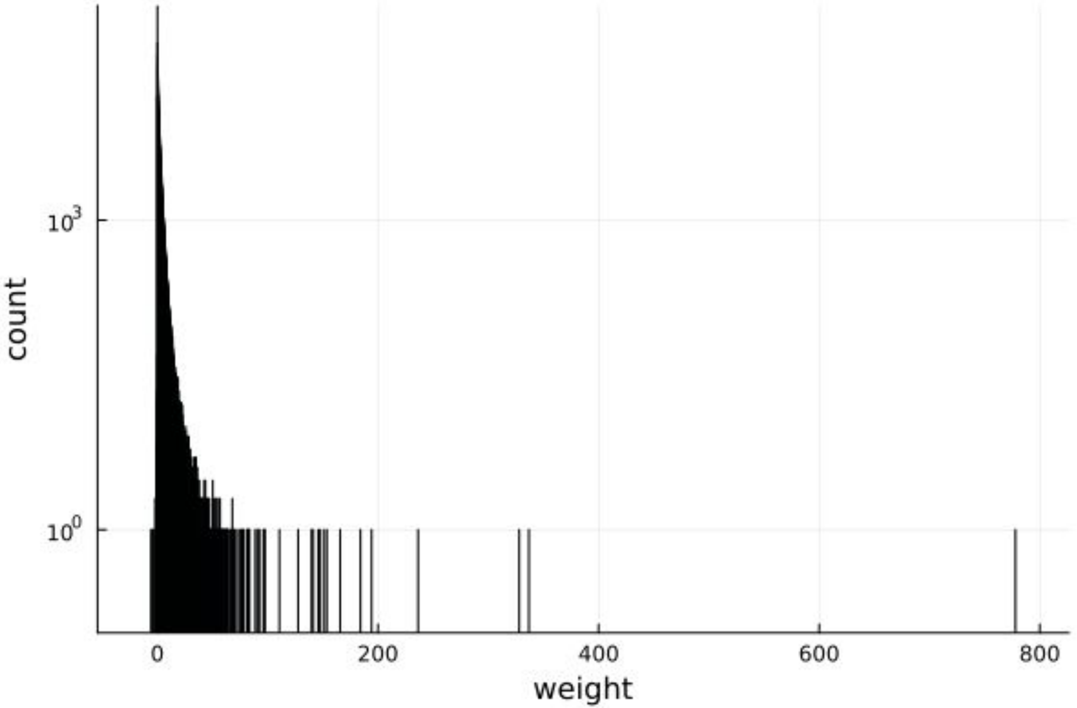
A weight on each sample based on the ratio of the target and proposal probability distributions

The acceptance efficiency can sink very low due to large weight outliers

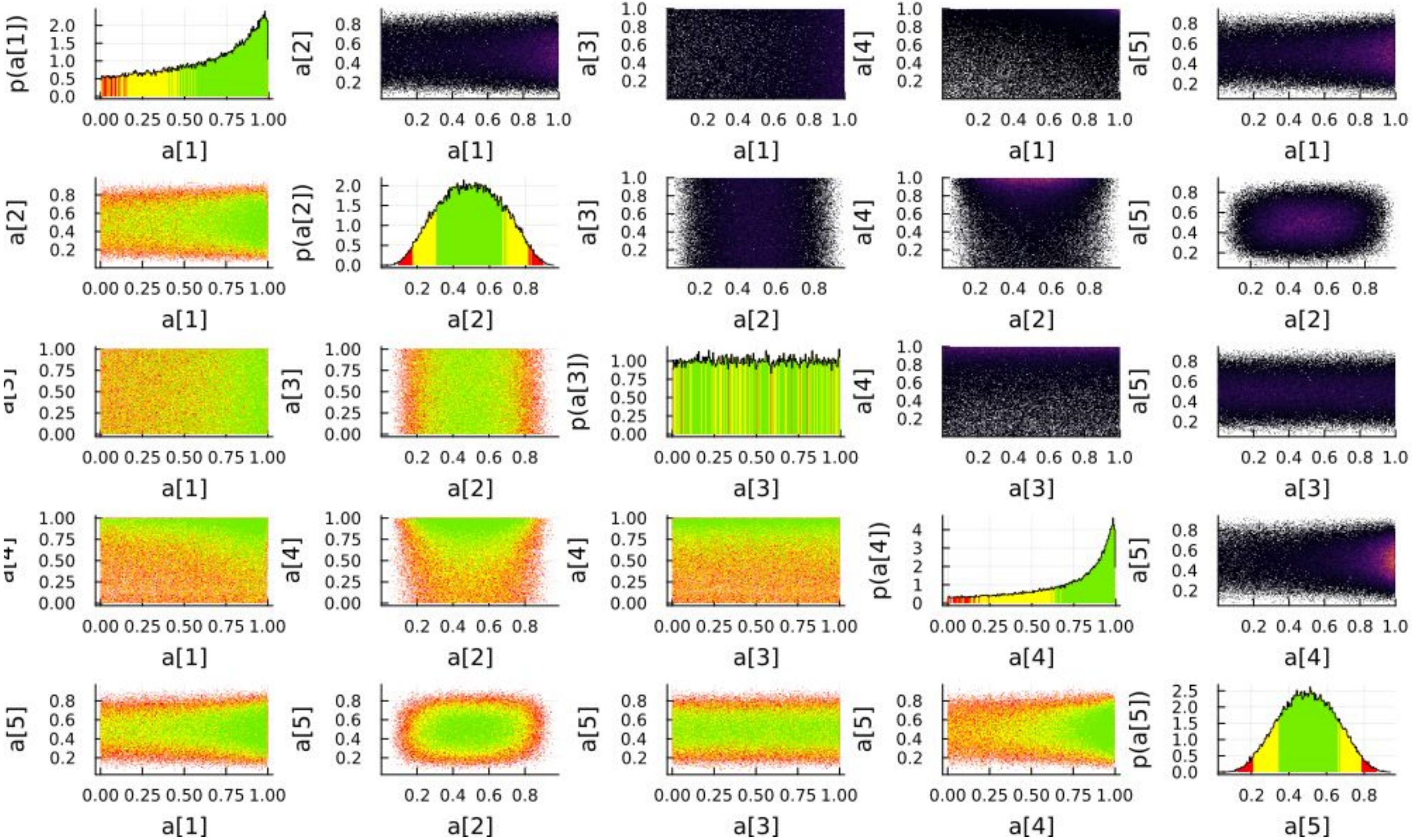
Z + 3 Jets scattering process

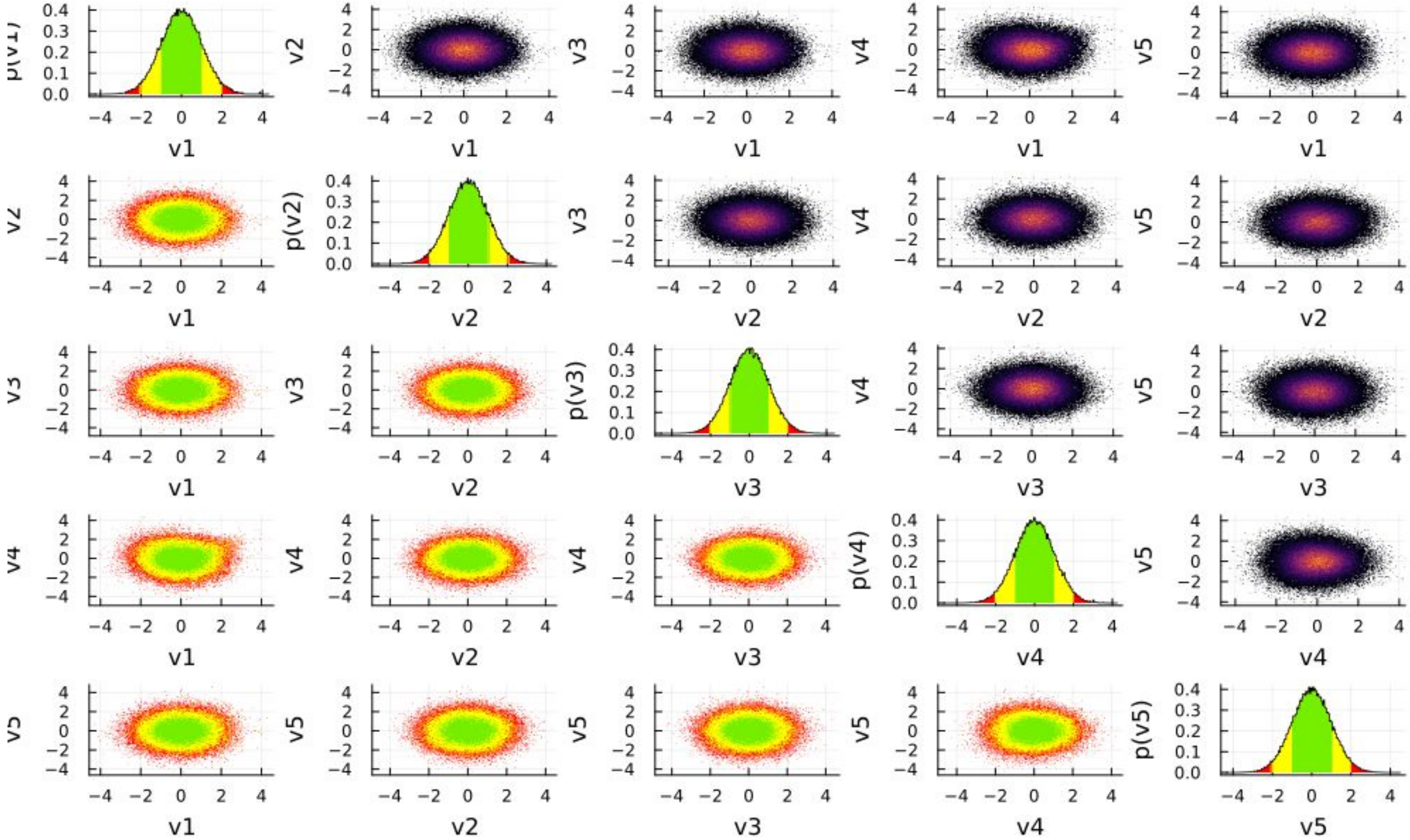


$$g\ g \rightarrow d\ d\ g\ Z \rightarrow d\ d\ g\ e^{+}\ e^{-}$$

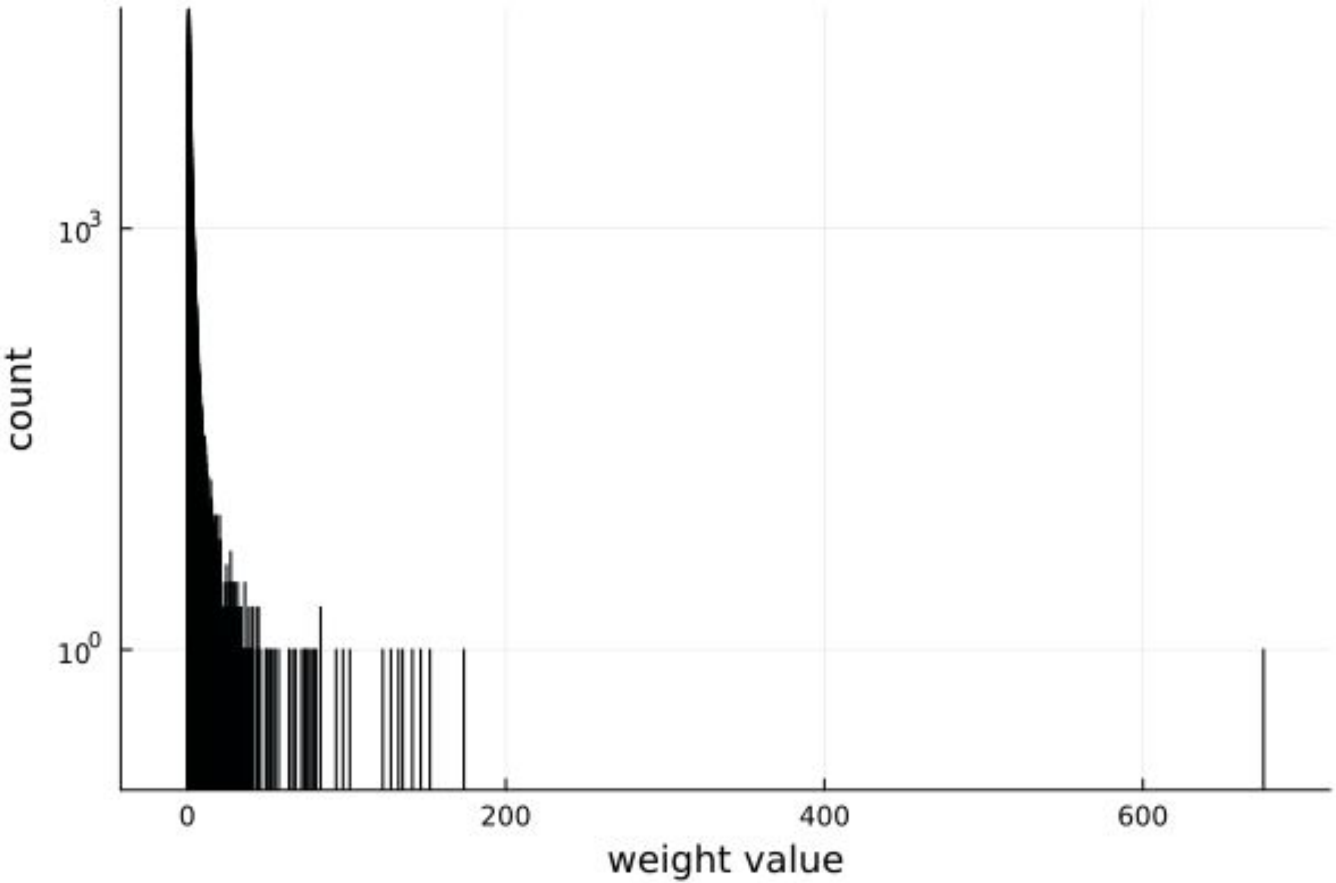


Using pepper native methods one has an unweighting efficiency of about 0.6%

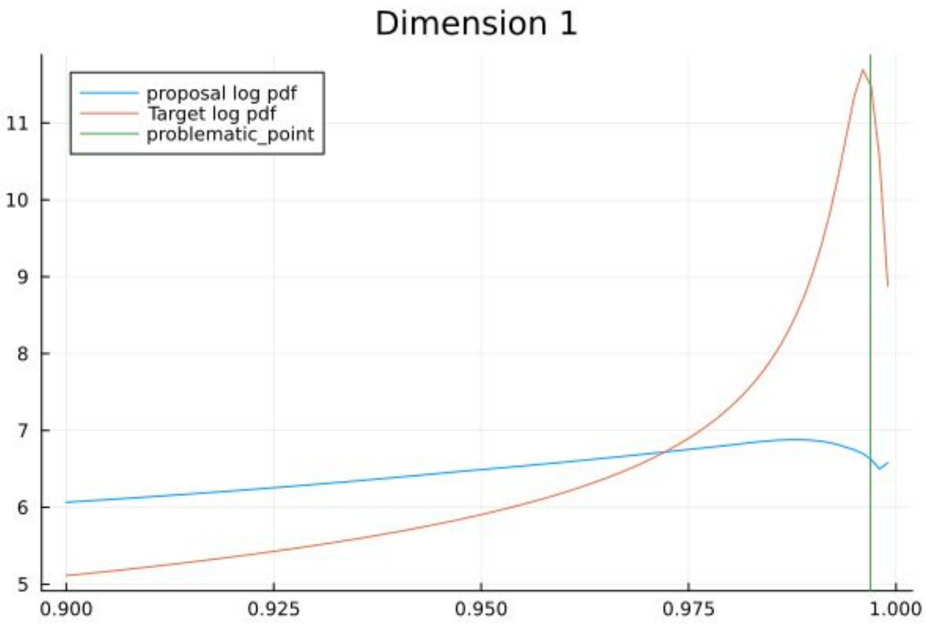

p(a[1])
p(a[2])
p(a[3])
p(a[4])
p(a[5])
a[1]
a[2]
a[3]
a[4]
a[5]

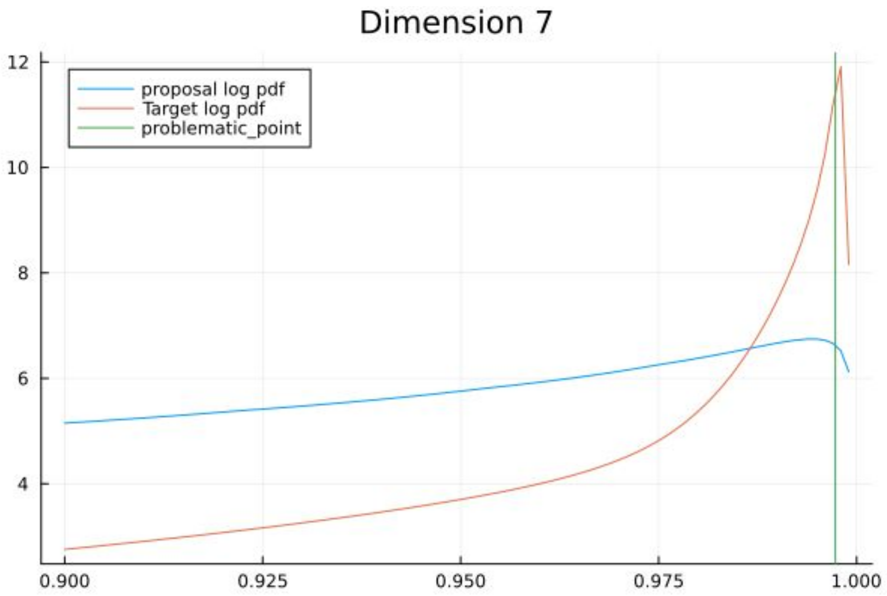


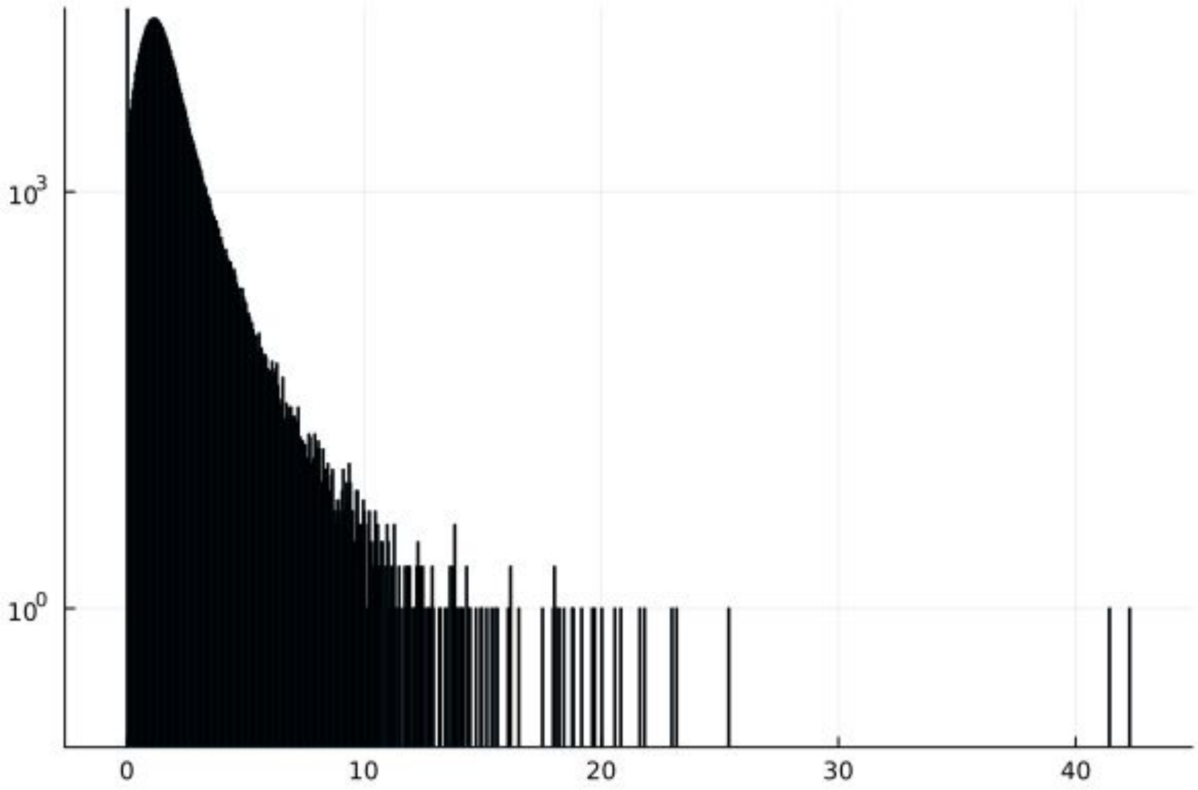
p(v1)
p(v2)
p(v3)
p(v4)
p(v5)
v1
v2
v3
v4
v5



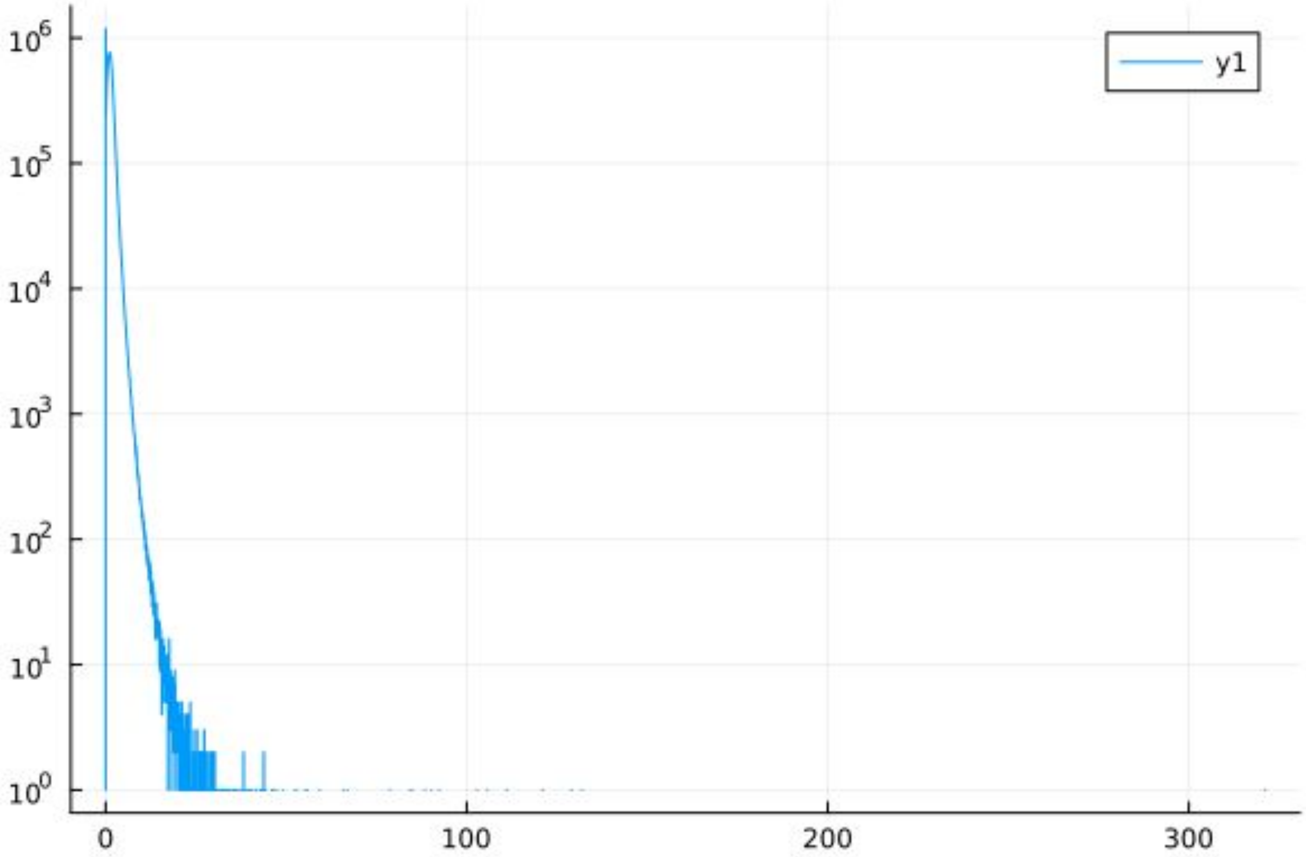
With no MAF component


Dimension 1
proposal log pdf
Target log pdf
problematic_point
5
6
7
8
9
10
11
0.900
0.925
0.950
0.975
1.000


Dimension 7
proposal log pdf
Target log pdf
problematic_point
4
6
8
10
12
0.900
0.925
0.950
0.975
1.000

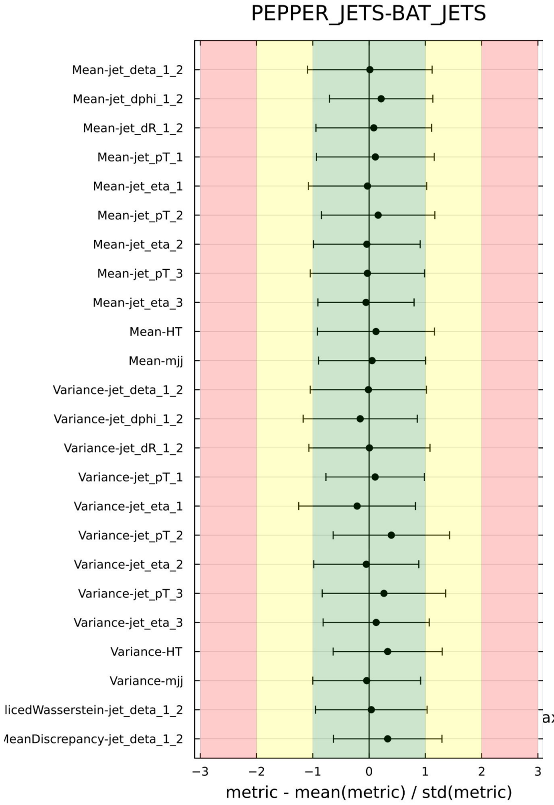


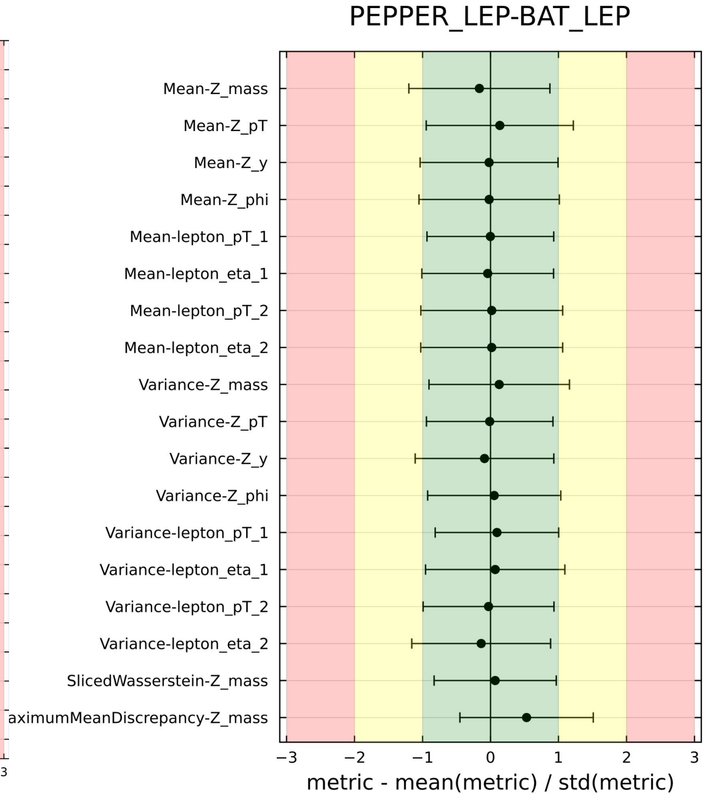
With the MAF component

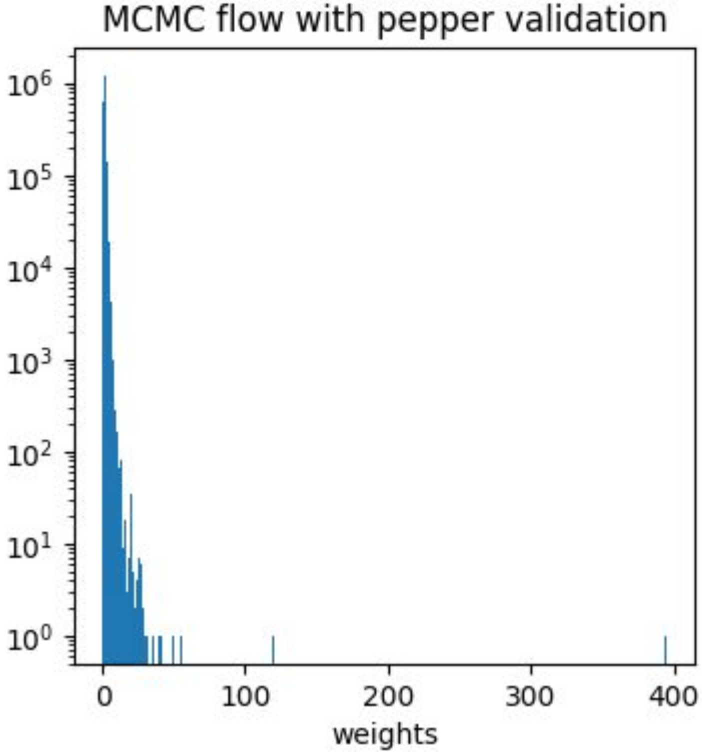


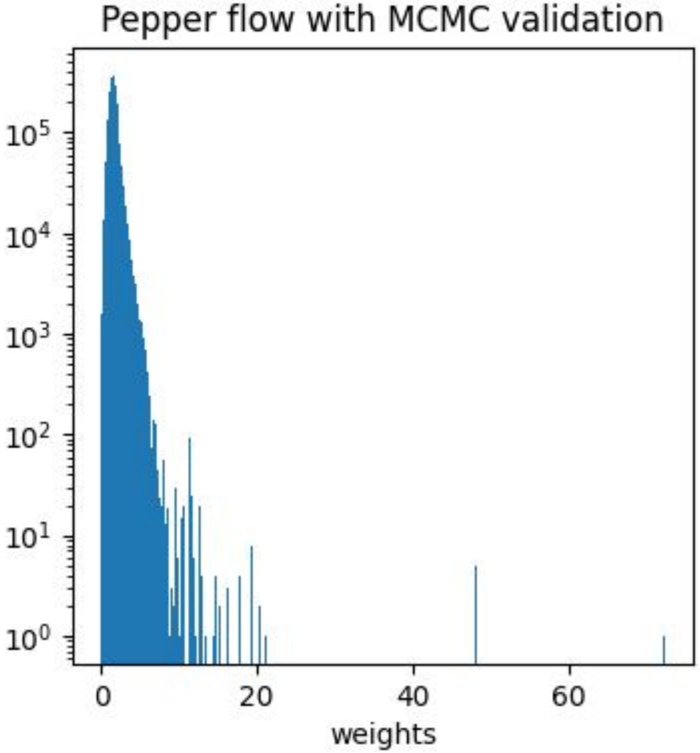
Producing many samples can still lead to outliers

After weight clipping the accepted samples are compatible with the jets and leptonic observables given by pepper samples

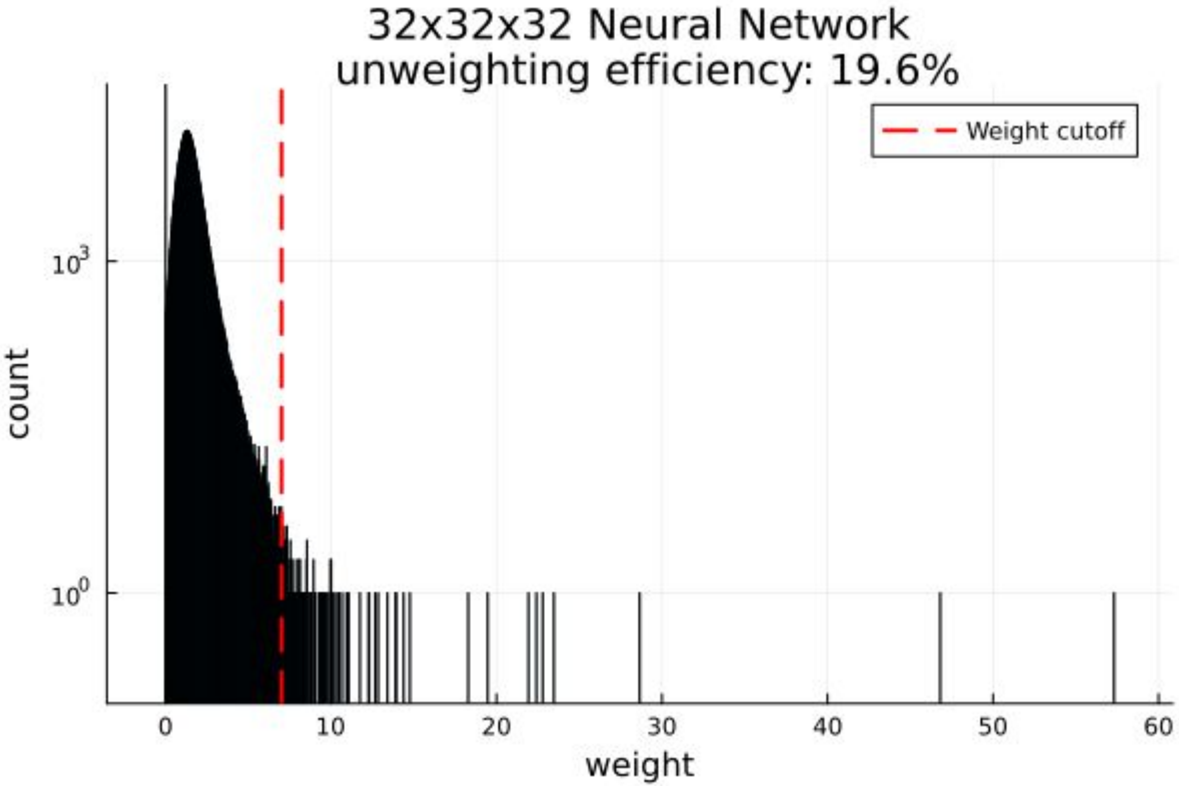








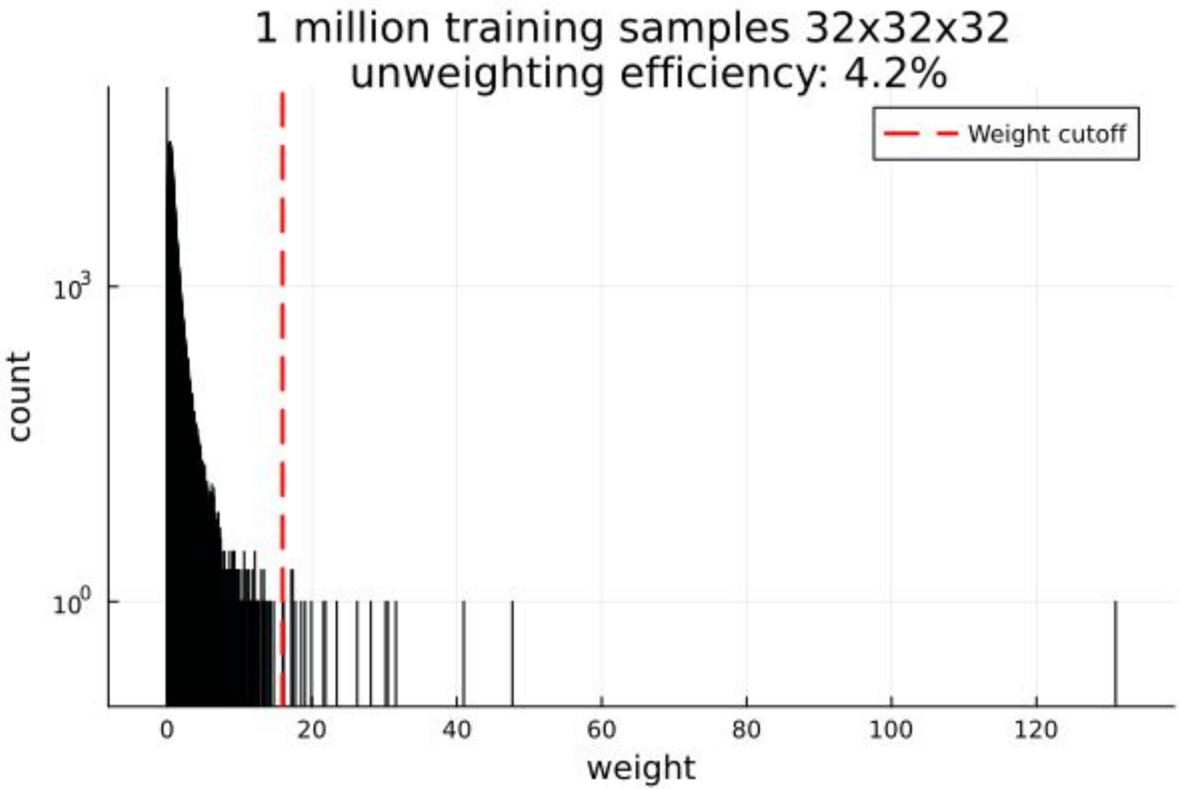
After weight clipping the accepted samples are compatible with the jets and leptonic observables given by pepper samples

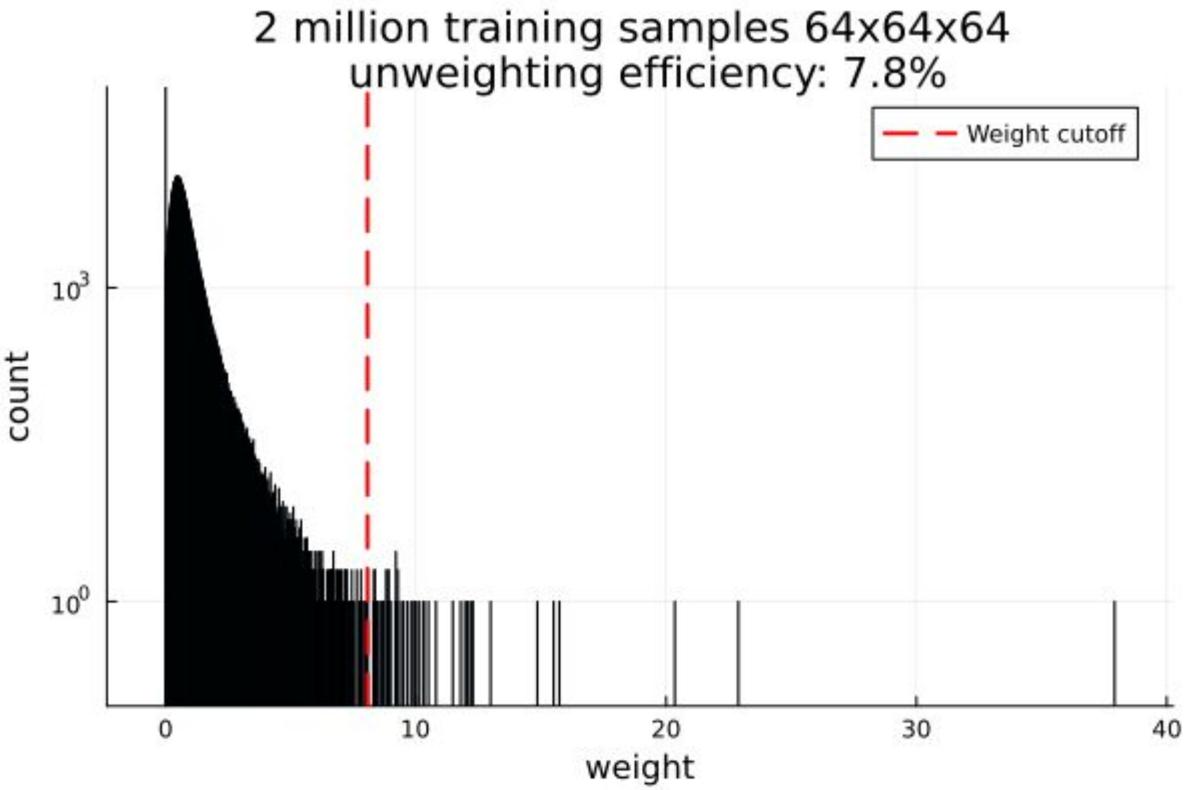


The normalizing flow appears to best perform when it uses a 32x32x32 neural network, with a batch size of 500 and using an earlystopper

$$g\ g \to d\ d\ g\ g\ Z \to d\ d\ g\ g\ e^+\ e^-$$

In the Z + 4 jets case the efficiency drops significantly





Increasing the training dataset gives major gains in performance but it is expensive

Thank you for your attention