

Neural Positive Reweighting

Benjamin Nachman

Lawrence Berkeley National Laboratory

bpnachman.com

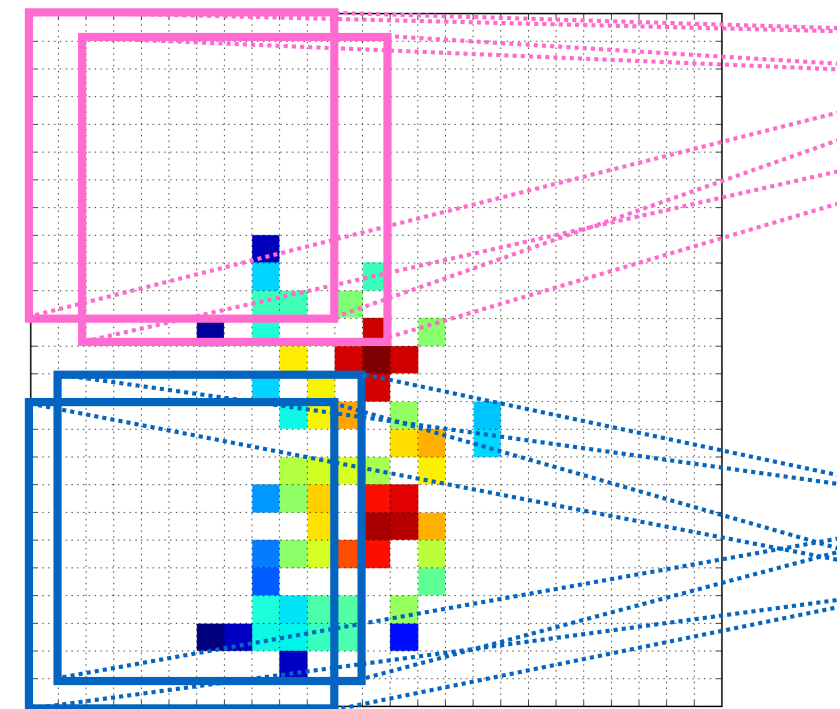
bpnachman@lbl.gov



@bpnachman



bnachman



HSF/WLCG
Virtual Workshop
Nov. 20, 2020

Part I: “**Positive**” Reweighting



Part I: “**Positive**” Reweighting

Suppose that a given phase space point x , there is a non-trivial distribution of weights $p(w|x)$.

Part I: “**Positive**” Reweighting

Suppose that a given phase space point x , there is a non-trivial distribution of weights $p(w|x)$.

Part II: **Neural** Reweighting

Part I: “**Positive**” Reweighting

Suppose that a given phase space point x , there is a non-trivial distribution of weights $p(w|x)$.

Suppose that x is continuous and/or high dimensional

Part II: **Neural Reweighting**

Weighted events

6

Consider the usual expectation value of an observable:

$$\langle \mathcal{O}(x) \rangle \approx \hat{\mathcal{O}}(x) \equiv \sum_{i=1}^N w_i \mathcal{O}(x_i)$$

(for example, could be the content of a histogram bin)

For simplicity, the weights are normalized to 1

Weighted events



Consider the usual expectation value of an observable:

$$\langle \mathcal{O}(x) \rangle \approx \hat{\mathcal{O}}(x) \equiv \sum_{i=1}^N w_i \mathcal{O}(x_i)$$

(for example, could be the content of a histogram bin)

$$= \sum_{j=1}^M \left(\sum_{k=1}^{n_{\mathcal{O}(x_j)}} w_k \right) \mathcal{O}(x_j)$$



sum over distinct observable values

Weighted events

8

Consider the usual expectation value of an observable:

$$\langle \mathcal{O}(x) \rangle \approx \hat{\mathcal{O}}(x) \equiv \sum_{i=1}^N w_i \mathcal{O}(x_i)$$

(for example, could be the content of a histogram bin)

$$= \sum_{j=1}^M \left(\sum_{k=1}^{n_{\mathcal{O}(x_j)}} w_k \right) \mathcal{O}(x_j)$$

↑
sum over distinct observable values

**Only sensitive to
the average weight**

If there is no information in $p(w \mid O(x))$ except the average, we can replace all w 's with $\langle w \mid O(x) \rangle$

$$\{(\mathcal{O}, w_1), (\mathcal{O}, w_2), (\mathcal{O}, w_3)\} \rightarrow \{(\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w})\}$$

If there is no information in $p(w \mid O(x))$ except the average, we can replace all w 's with $\langle w \mid O(x) \rangle$

$$\{(\mathcal{O}, w_1), (\mathcal{O}, w_2), (\mathcal{O}, w_3)\} \rightarrow \{(\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w})\}$$

In fact, you can replace all events with $O(x)=o$ with k events that all have $O(x) = o$ and weight $k \langle w \mid O(x) \rangle$

$$\{(\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w})\} \rightarrow \{(\mathcal{O}, 3\hat{w})\}$$

If there is no information in $p(w | O(x))$ except the average, we can replace all w 's with $\langle w | O(x) \rangle$

$$\{(\mathcal{O}, w_1), (\mathcal{O}, w_2), (\mathcal{O}, w_3)\} \rightarrow \{(\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w})\}$$

In fact, you can replace all events with $O(x)=o$ with k events that all have $O(x) = o$ and weight $k \langle w | O(x) \rangle$

$$\{(\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w}), (\mathcal{O}, \hat{w})\} \rightarrow \{(\mathcal{O}, 3\hat{w})\}$$

Why is this useful? If $k \ll n_o$, then we can pass far fewer events through our detector simulation !!

What if the phase space is not discrete?

$$\hat{\mathcal{O}}_{\text{patch}} \approx \mathcal{O}(x_{\text{patch}}) \sum_{i=1}^{n_{\text{patch}}} w_i$$

Consider a small *patch* around each point in phase space where the observable is approximately constant.

$$\approx \mathcal{O}(x_{\text{patch}}) \sum_{j=1}^{n_{\text{patch}}/k} \left(\frac{k}{n_{\text{patch}}} \sum_{i=1}^{n_{\text{patch}}} w_i \right) \approx \mathcal{O}(x_{\text{patch}}) \sum_{j=1}^{n_{\text{patch}}/k} \boxed{k \langle W \rangle_{\text{patch}}} \\ w_{\text{new,patch}}$$

Not all values of k are equally good.

A good choice of k would be one
that preserves the **uncertainty**
(all values of k preserve the central value)


In the patch, match the sum of the squares of the weights:

$$\sum_{i=1}^{n_{\text{patch}}/k} w_{\text{new,patch}}^2 = \sum_{i=1}^{n_{\text{patch}}} w_i^2$$

Not all values of k are equally good.

A good choice of k would be one
that preserves the **uncertainty**
(all values of k preserve the central value)

In the patch, match the sum of the squares of the weights:

$$\sum_{i=1}^{n_{\text{patch}}/k} w_{\text{new,patch}}^2 = \frac{k}{n_{\text{patch}}} \sum_{i=1}^{n_{\text{patch}}} w_i^2 \approx k \langle W^2 \rangle_{\text{patch}}$$


Local (Resampling + Uncertainties)

15

$$w_{\text{new,patch}} \approx k \langle W \rangle_{\text{patch}}$$

$$w_{\text{new,patch}}^2 \approx k \langle W^2 \rangle_{\text{patch}}$$

\Rightarrow

$$k_{\text{patch}} \approx \frac{\langle W^2 \rangle_{\text{patch}}}{\langle W \rangle_{\text{patch}}^2}$$

$$\begin{aligned} w_{\text{new,patch}} &\approx k \langle W \rangle_{\text{patch}} \\ w_{\text{new,patch}}^2 &\approx k \langle W^2 \rangle_{\text{patch}} \end{aligned} \quad \Rightarrow \quad k_{\text{patch}} \approx \frac{\langle W^2 \rangle_{\text{patch}}}{\langle W \rangle_{\text{patch}}^2}$$

Now, take the continuum limit:

$$K(X) \approx \frac{\langle W^2 | X \rangle}{\langle W | X \rangle^2}$$

N.B. upper case is a random variable

1. Estimate $\widehat{W}(X) \approx \langle W|X \rangle$.
2. Estimate $\widehat{W}^2(X) \approx \langle W^2|X \rangle$.
3. Define $\widehat{K}(X) = \widehat{W}^2(X)/\widehat{W}(X)^2$.
4. For each event i , keep it with probability $1/\widehat{K}(x_i)$; otherwise discard the event. Because $\widehat{K}(x_i) \geq 1$ by construction, no event will be repeated.
5. For each kept event, set the new event weight to be $w_i \mapsto \widetilde{W}(x_i) \equiv \widehat{W}(x_i) \widehat{K}(x_i)$.

1. Estimate $\widehat{W}(X) \approx \langle W|X \rangle$.
2. Estimate $\widehat{W^2}(X) \approx \langle W^2|X \rangle$.
3. Define $\widehat{K}(X) = \widehat{W^2}(X)/\widehat{W}(X)^2$.
4. For each event i , keep it with probability $1/\widehat{K}(x_i)$; otherwise discard the event. Because $\widehat{K}(x_i) \geq 1$ by construction, no event will be repeated.
5. For each kept event, set the new event weight to be $w_i \mapsto \widetilde{W}(x_i) \equiv \widehat{W}(x_i) \widehat{K}(x_i)$.

How?

Fact 1: Neural networks are great function approximators, especially in high dimensions.

Fact 1: Neural networks are great function approximators, especially in high dimensions.

Fact 2: neural networks trained to distinguish two samples learn to approximate (a function monotonic to) the likelihood ratio.

Fact 1: Neural networks are great function approximators, especially in high dimensions.

Fact 2: neural networks trained to distinguish two samples learn to approximate (a function monotonic to) the likelihood ratio.

We need to learn an unbinned likelihood ratio between a weighted sample and an unweighted one.

I do not have time to go into the machine learning, but please ask if you are interested!

We need to learn an unbinned likelihood ratio between a weighted sample and an unweighted one.

If you use this loss function:

$$\mathcal{L}[g] = - \sum_{i=1}^N w_i \log g(x_i) - \sum_{i=1}^N \log (1 - g(x_i))$$

(classifier to distinguish a sample from itself, but weighted)

We need to learn an unbinned likelihood ratio between a weighted sample and an unweighted one.

If you use this loss function:

$$\mathcal{L}[g] = - \sum_{i=1}^N w_i \log g(x_i) - \sum_{i=1}^N \log (1 - g(x_i))$$

(classifier to distinguish a sample from itself, but weighted)

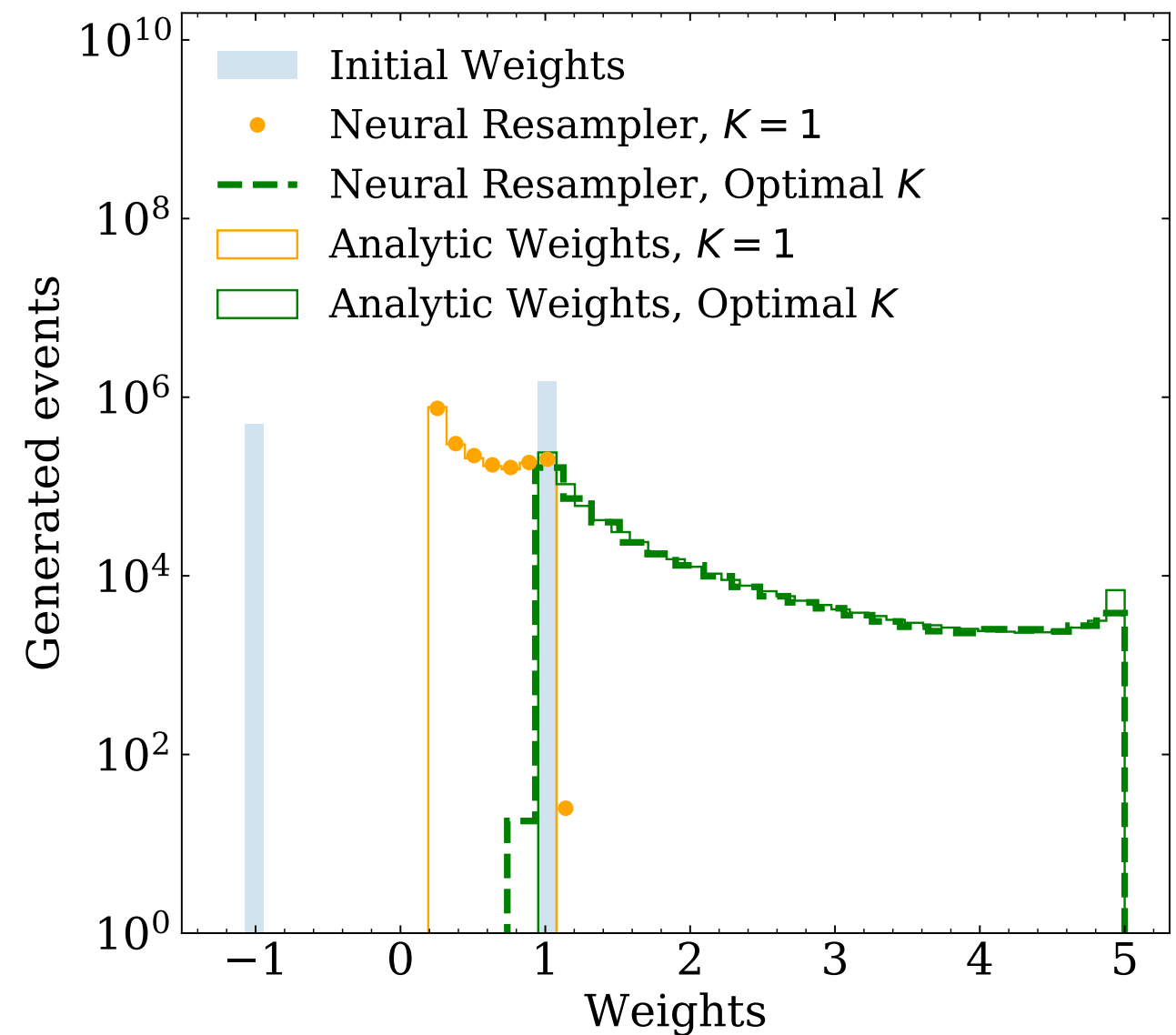
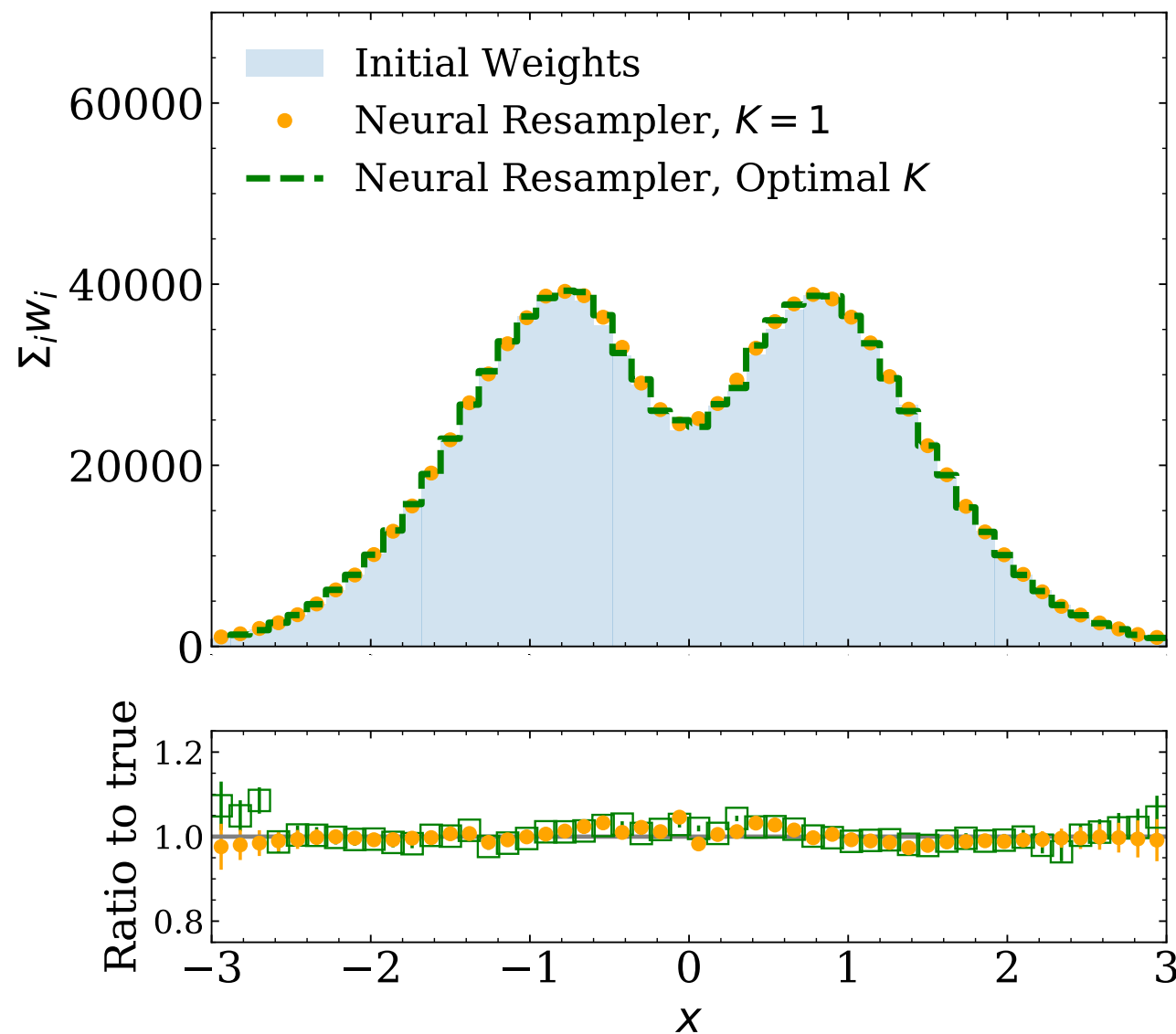
Then, you can estimate the function we need:

$$\Rightarrow \frac{g(x)}{1 - g(x)} = \widehat{W}(x) \approx \langle W | X \rangle \quad \text{(similar story for the weight squared)}$$

Neural Resampling in Action

24

First: Wide Gaussian + (-1) Narrower Gaussian

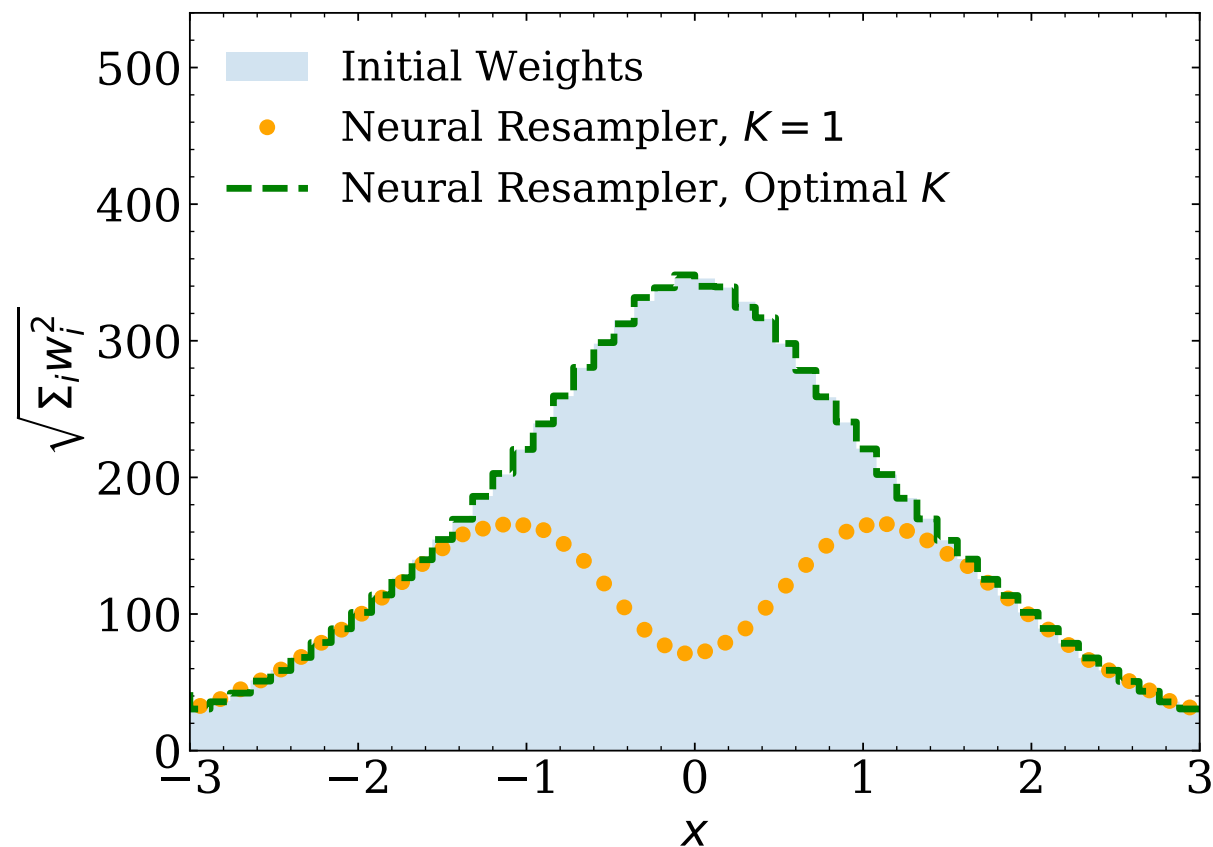


(binning is only for illustration - the resampling is unbinned)

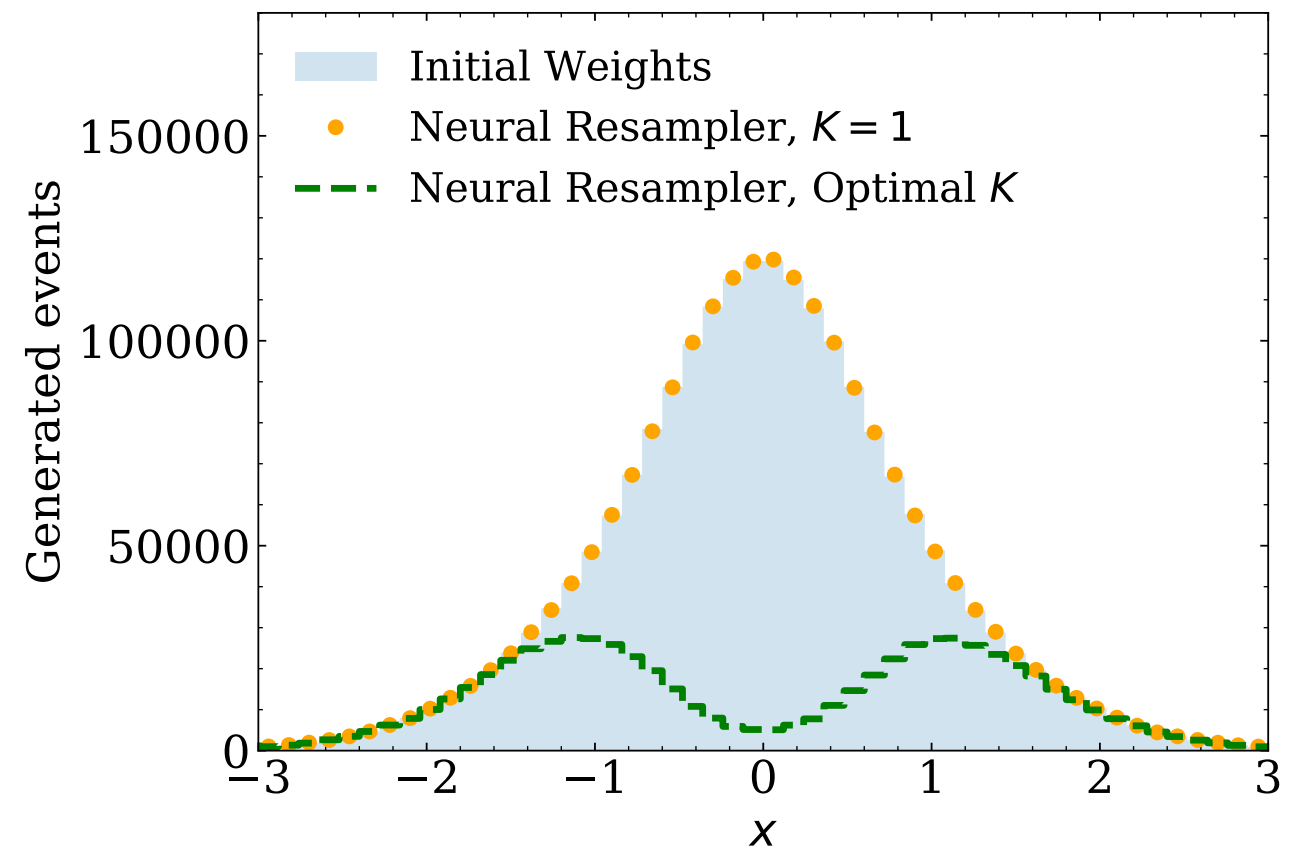
Neural Resampling in Action

25

First: Wide Gaussian + (-1) Narrower Gaussian



Preserves local uncertainty

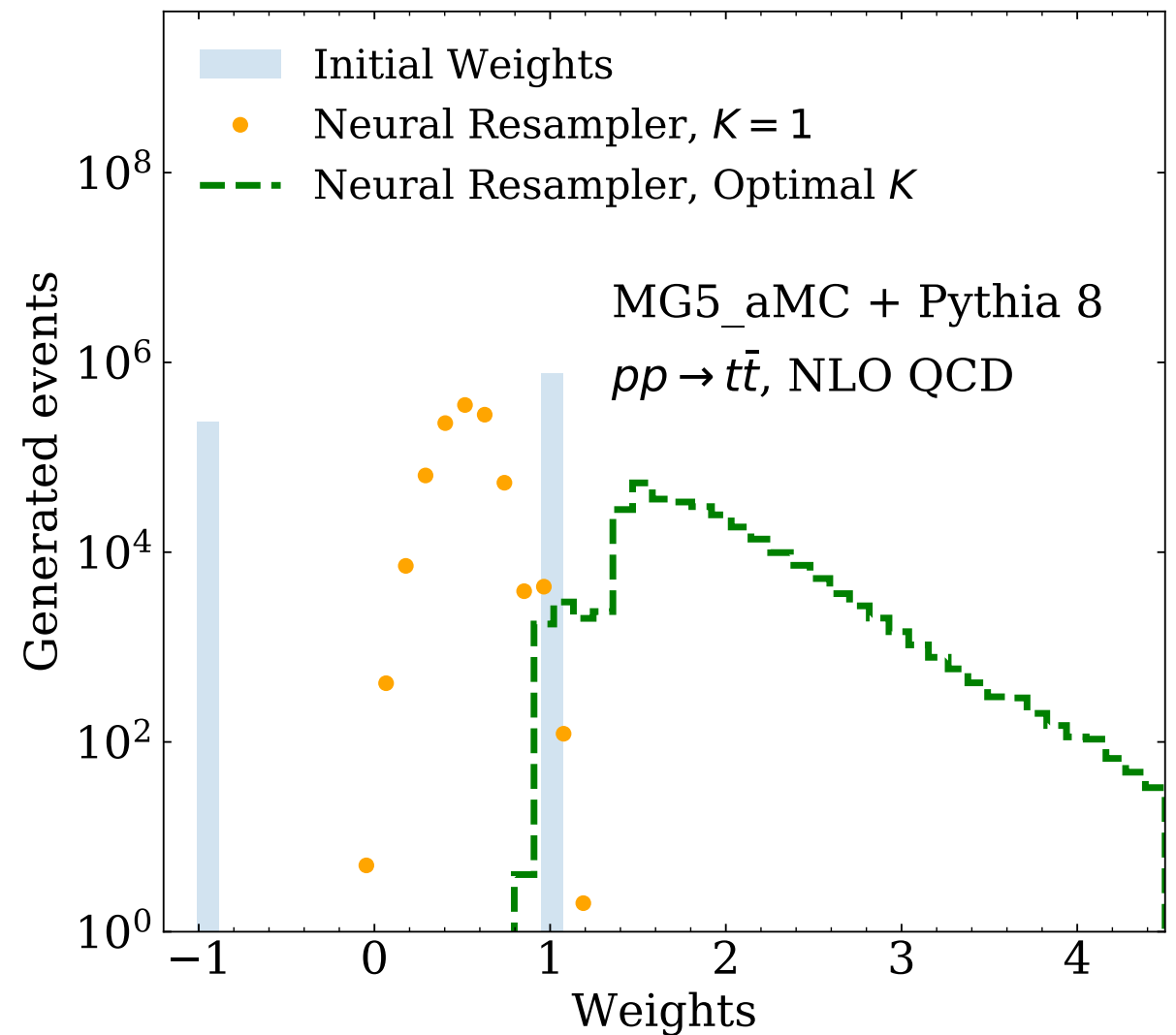
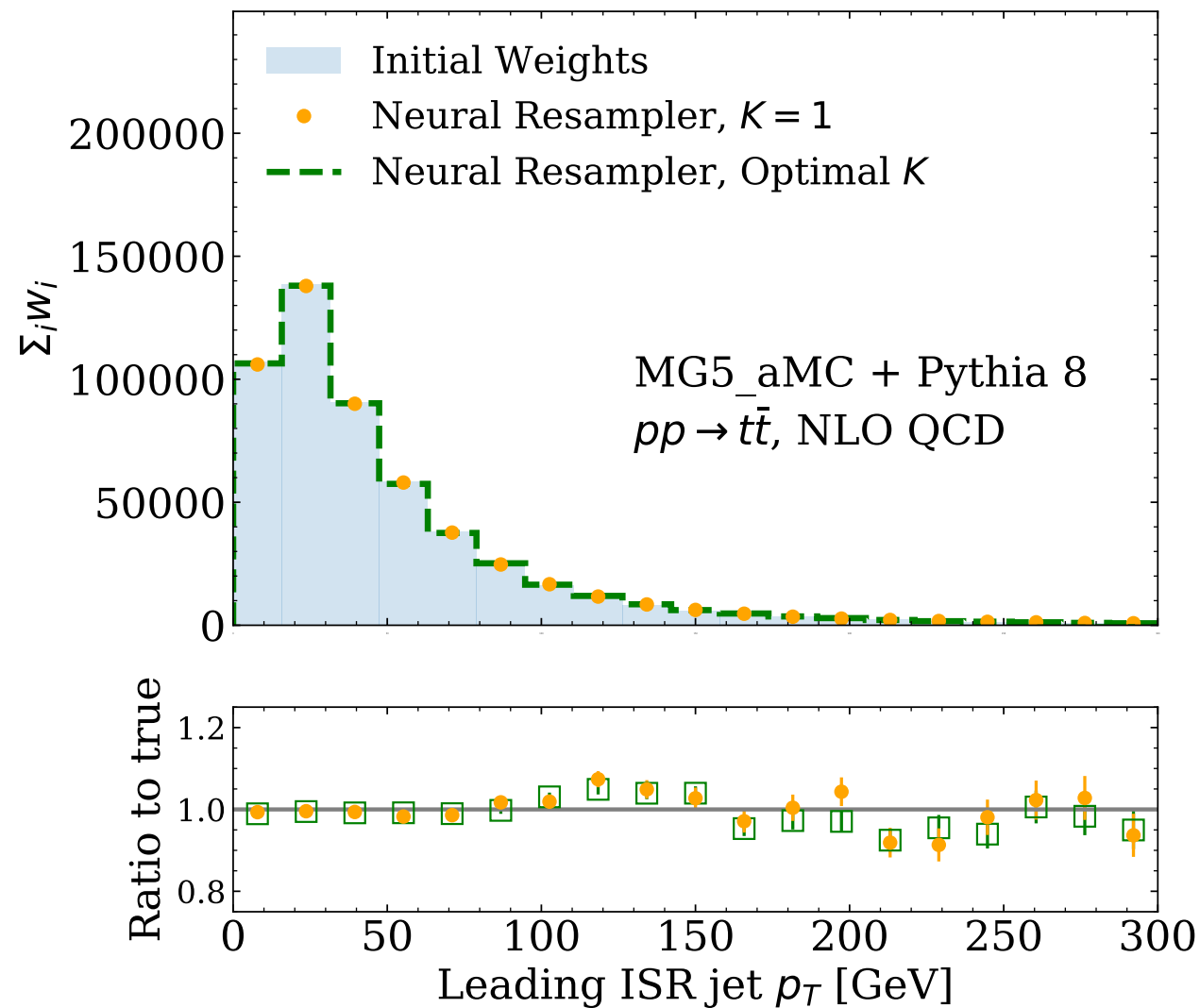


Reduces the
number of events

Neural Resampling in Action

26

Second: Top quark pairs at NLO

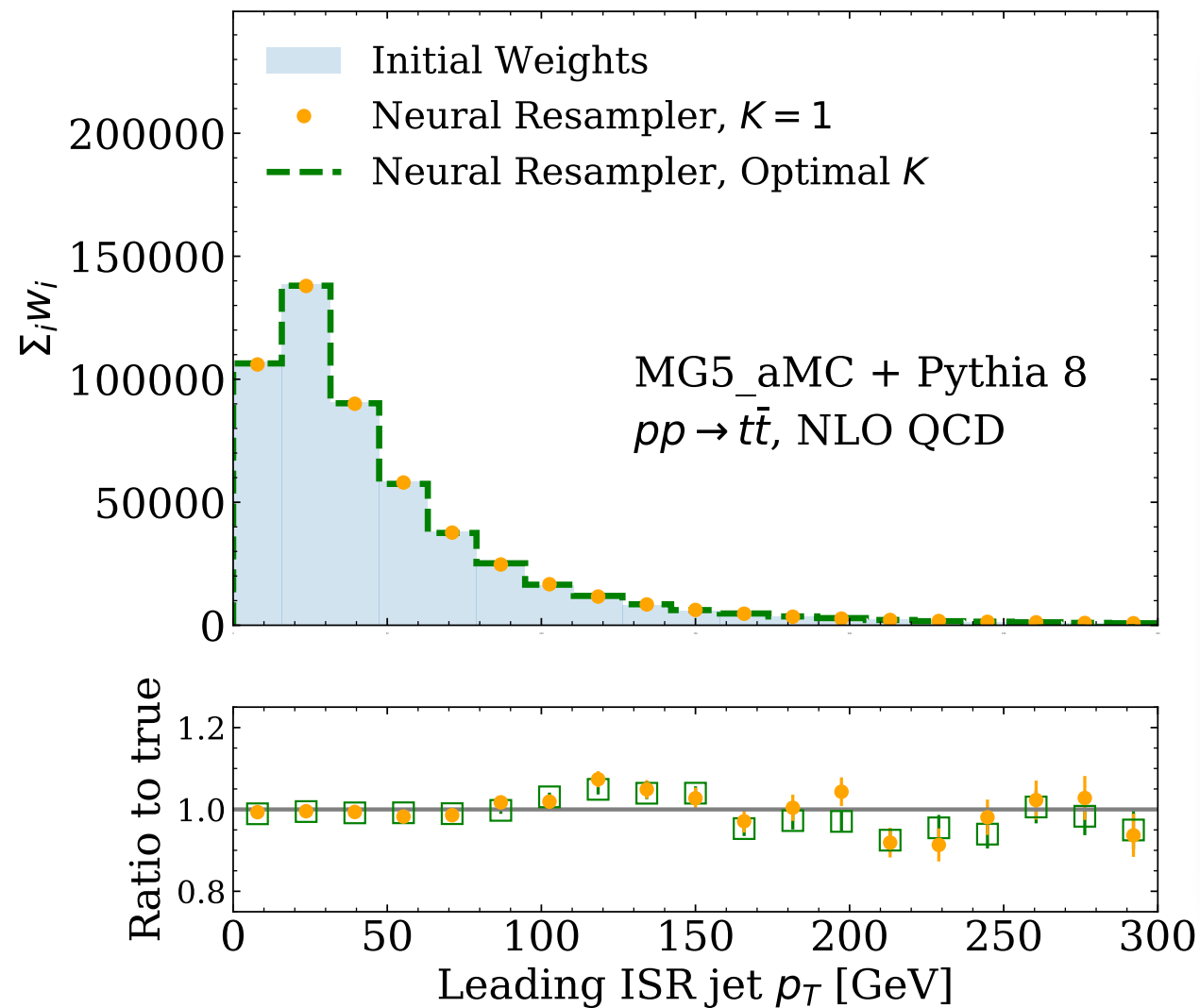


(Uses a set-based neural network called **particle flow networks** acting on jet and lepton 4-vectors)

Neural Resampling in Action

27

Second: Top quark pairs at NLO



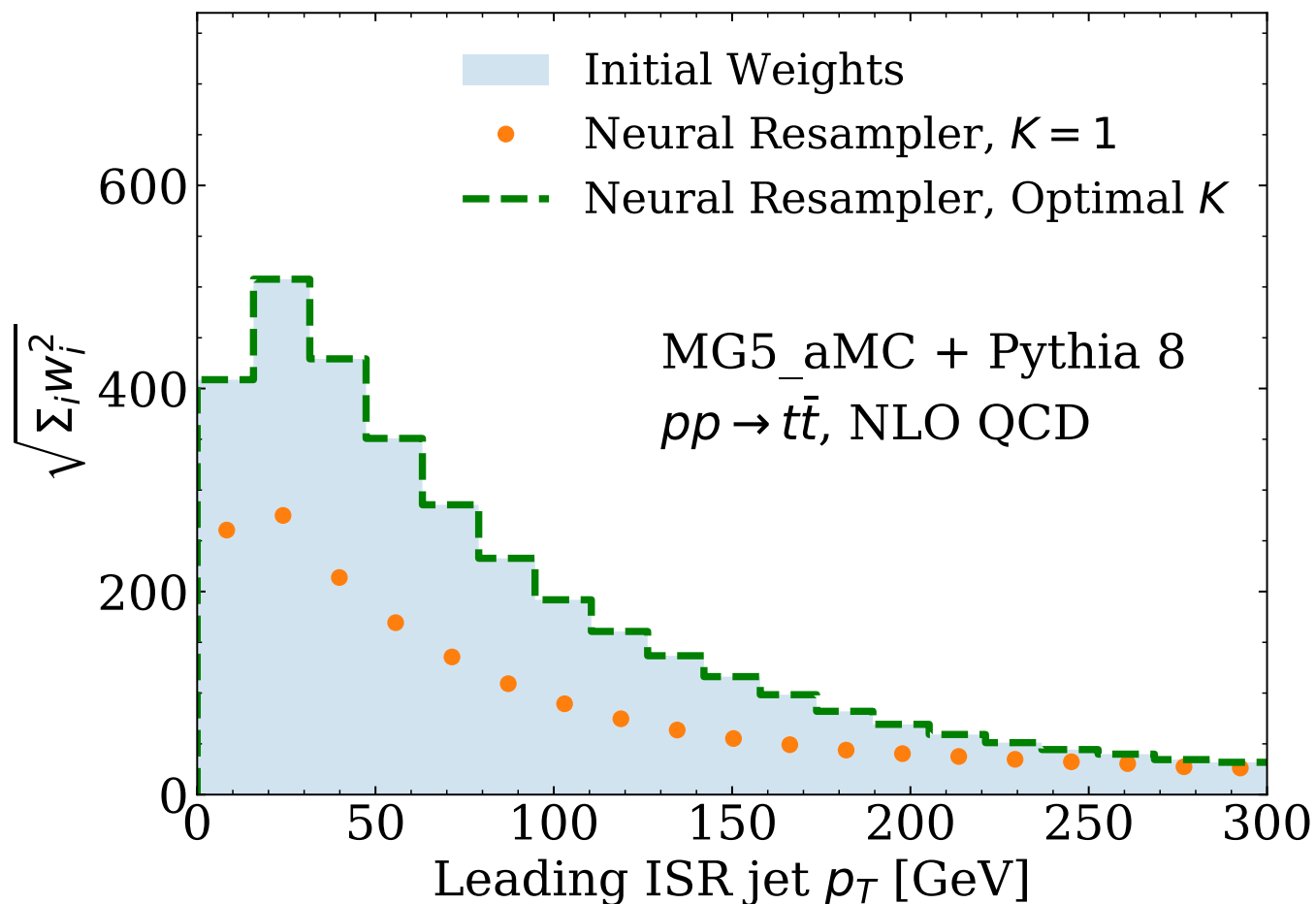
This means you can consider any observable that can be computed from these 4-vectors !

(i.e. reweight full phase space, decide observable later...)

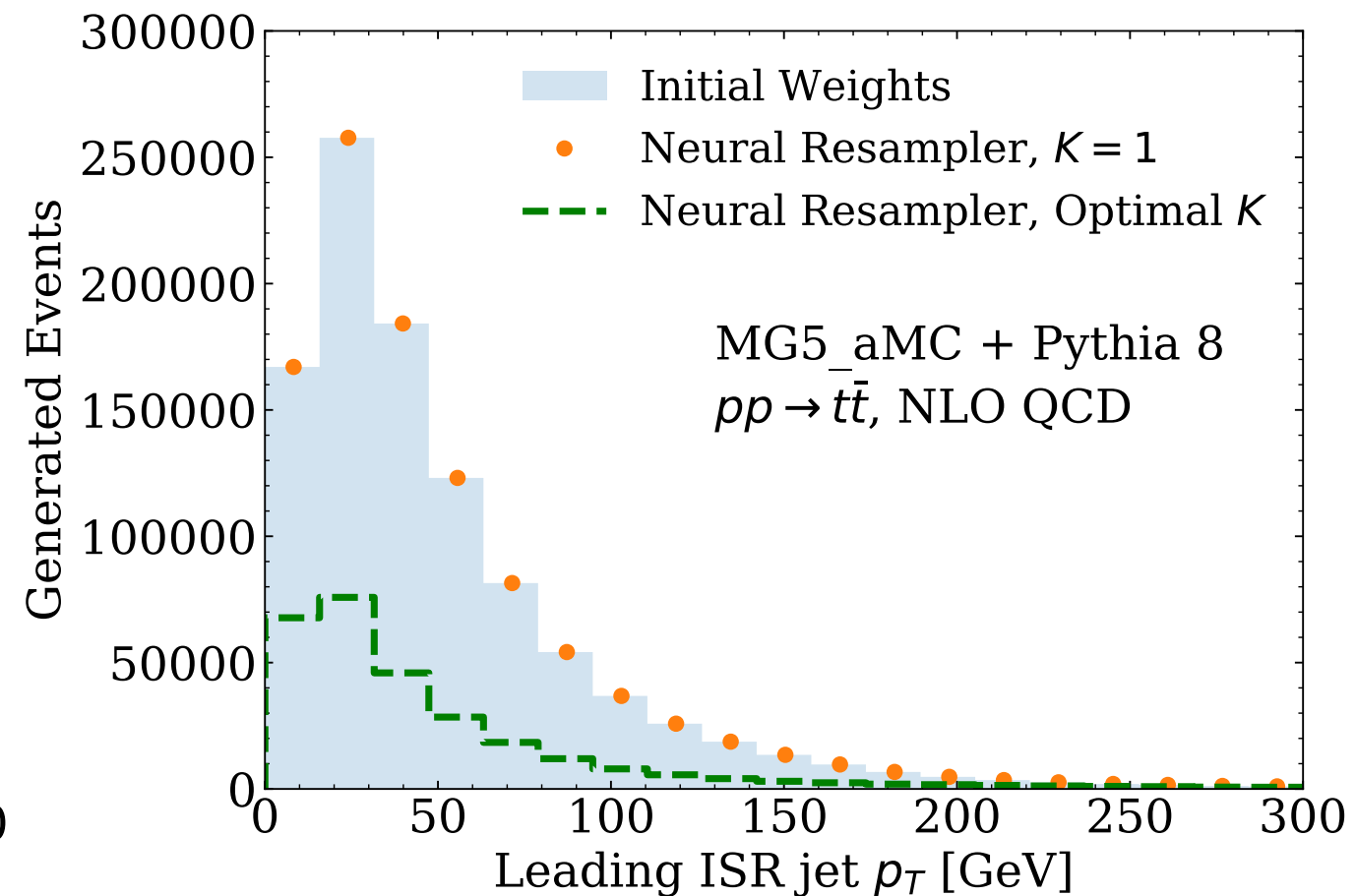
(Uses a set-based neural network called **particle flow networks** acting on jet and lepton 4-vectors)

Neural Resampling in Action

28



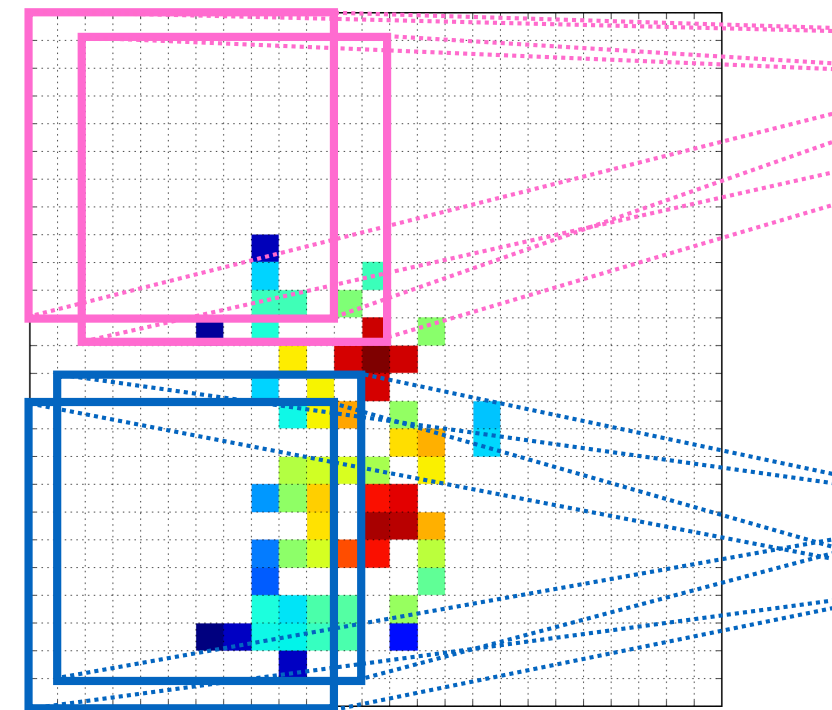
Preserves local uncertainty



Reduces the
number of events

Resampling is a model independent method for reducing the number of events needed to run through detector simulation.

You can preserve the local cross section and the local **uncertainty**.



Neural networks are an effective way of parameterizing the reweighting functions to make the approach **high-dimensional and local**.

Backup

