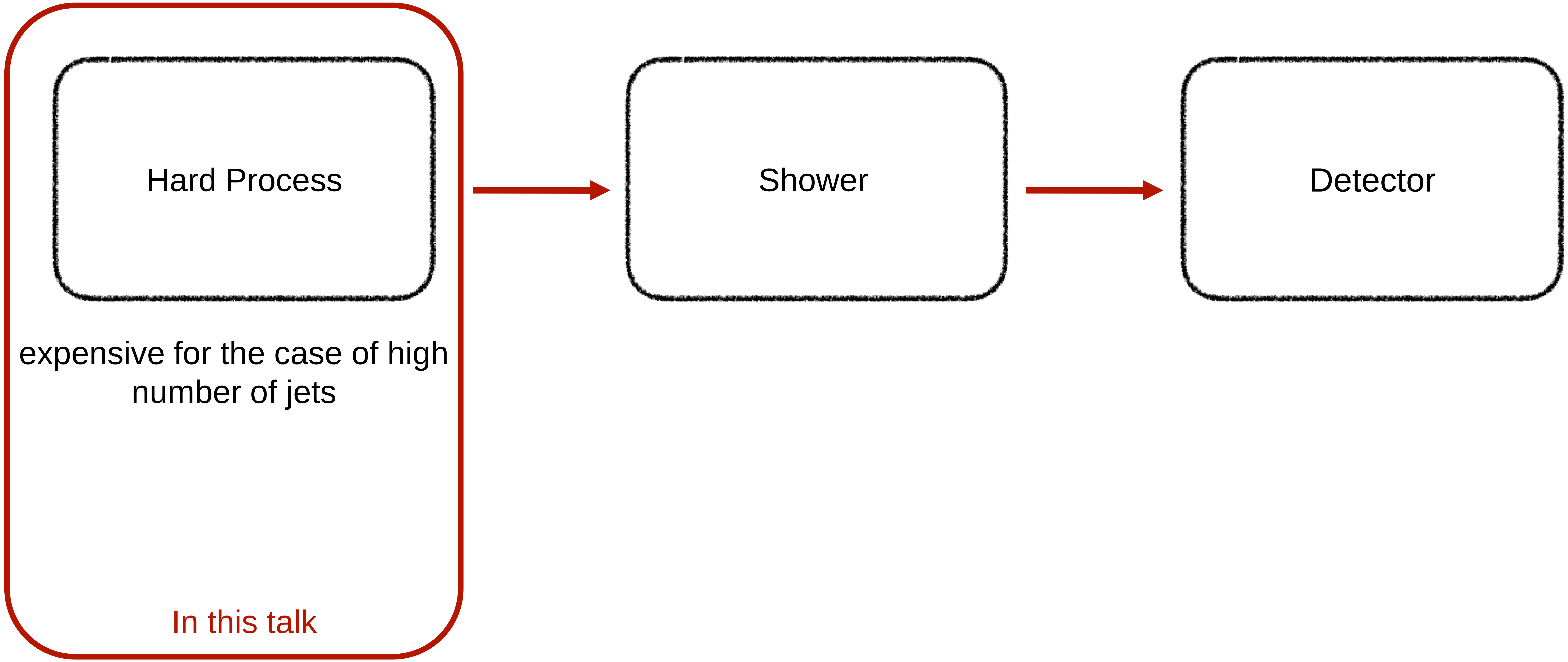
Predicting scattering amplitudes with Bayesian neural networks

By Michel Luchmann, Tilman Plehn, Simon Badger, Sebastian Pitz

Motivation

- Simulations for the LHC are very expensive. The amount of compute needed for the HL-LHC will exceed its budget
- In need of faster event generation! -> Replace parts of the simulation chain with ML methods!



Learning Amplitudes

Simon Badger, Joseph Bullock, arXiv:2002.07516v2 [hep-ph]

Joseph Aylett-Bullock, Simon Badger, Ryan Moodie, arXiv:2106.09474 [hep-ph]

K. Danziger, T. Janßen, S. Schumann, F. Siegert, arXiv:2109.11964 [hep-ph]

Fady Bishara, Marc Montull, arXiv:1912.11055 [hep-ph]

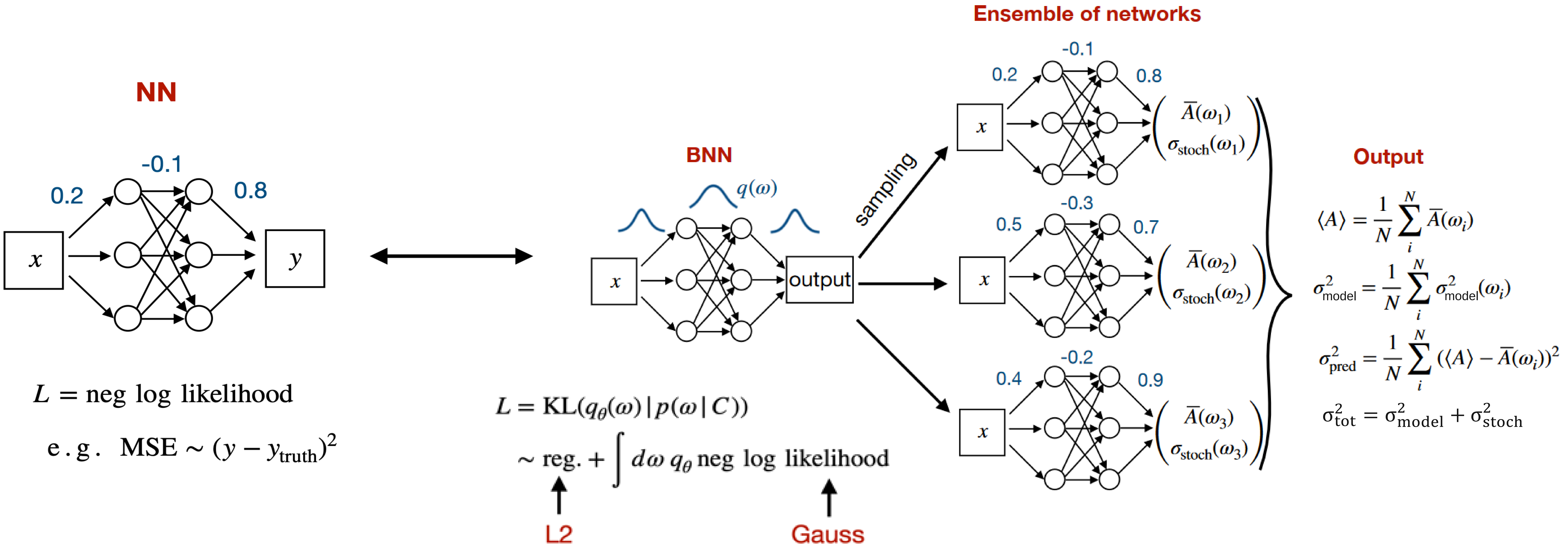
Daniel Maître, Henry Truong, arXiv:2107.06625 [hep-ph]

(Not complete list!)

- **Idea: Replace amplitudes by learnt amplitudes**

    1. Start event generation tool and generate small initial data set
    2. Train (Bayesian) neural network on this dataset
    3. Use trained network to compute amplitudes quickly and generate more data

Bayesian neural networks



IRN Terascale @ LPC-Clermont: "Uncertainty Estimation for LHC Event Generation"

to learn more, for instance:

Y. Gal, Uncertainty in Deep Learning, ph.D. Thesis

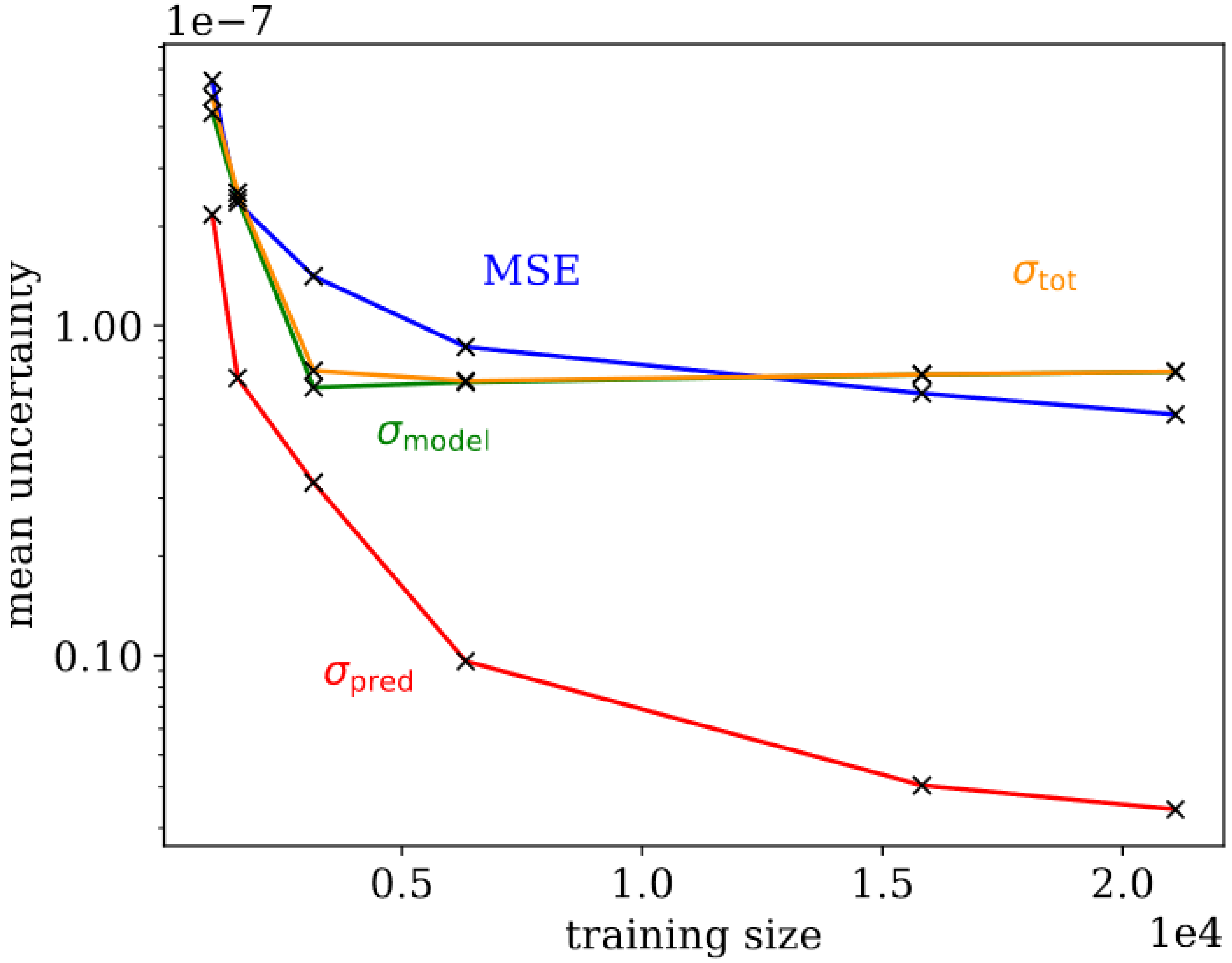
Sven Bollweg, Manuel Haussmann, Gregor Kasieczka, Michel Luchmann, Tilman Plehn et al, arXiv:1904.10004 [hep-ph]

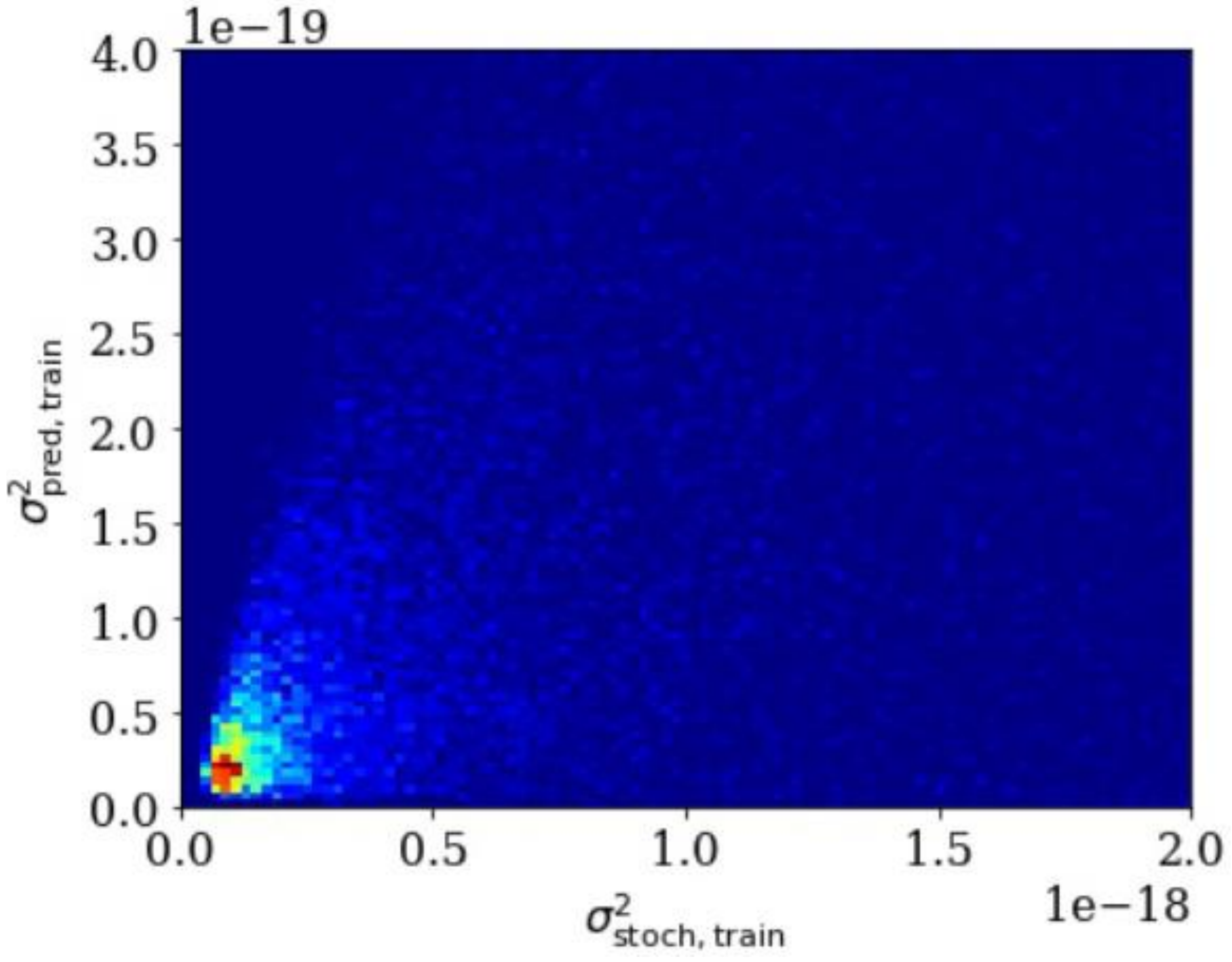
G. Kasieczka, M. L., F. Otterpohl and T. Plehn, arXiv:2003.11099 [hep-ph]

Marco Bellagente, Manuel Haußmann, Michel Luchmann, Tilman Plehn, arXiv:2104.04543 [hep-ph]

Anja Butter, Theo Heimel, Sander Hummerich, Tobias Krebs, Tilman Plehn, Armand Rousselot, Sophia Vent, arXiv:2110.13632 [hep-ph]

Uncertainties





$$\sigma^2_{tot} = \sigma^2_{model} + \sigma^2_{stoch}$$

$$\sigma^2_{pred} = \sum(y - \bar{y})^2$$

$$\sigma^2_{model} = \sum \sigma^2$$

- **2 different uncertainties**
- **For infinitely large training size**
  - $\sigma_{pred} \rightarrow 0$
  - $\sigma_{model} \rightarrow$ const.

More uncertainty analysis: arXiv:2003.11099 [hep-ph]

Setup and Process

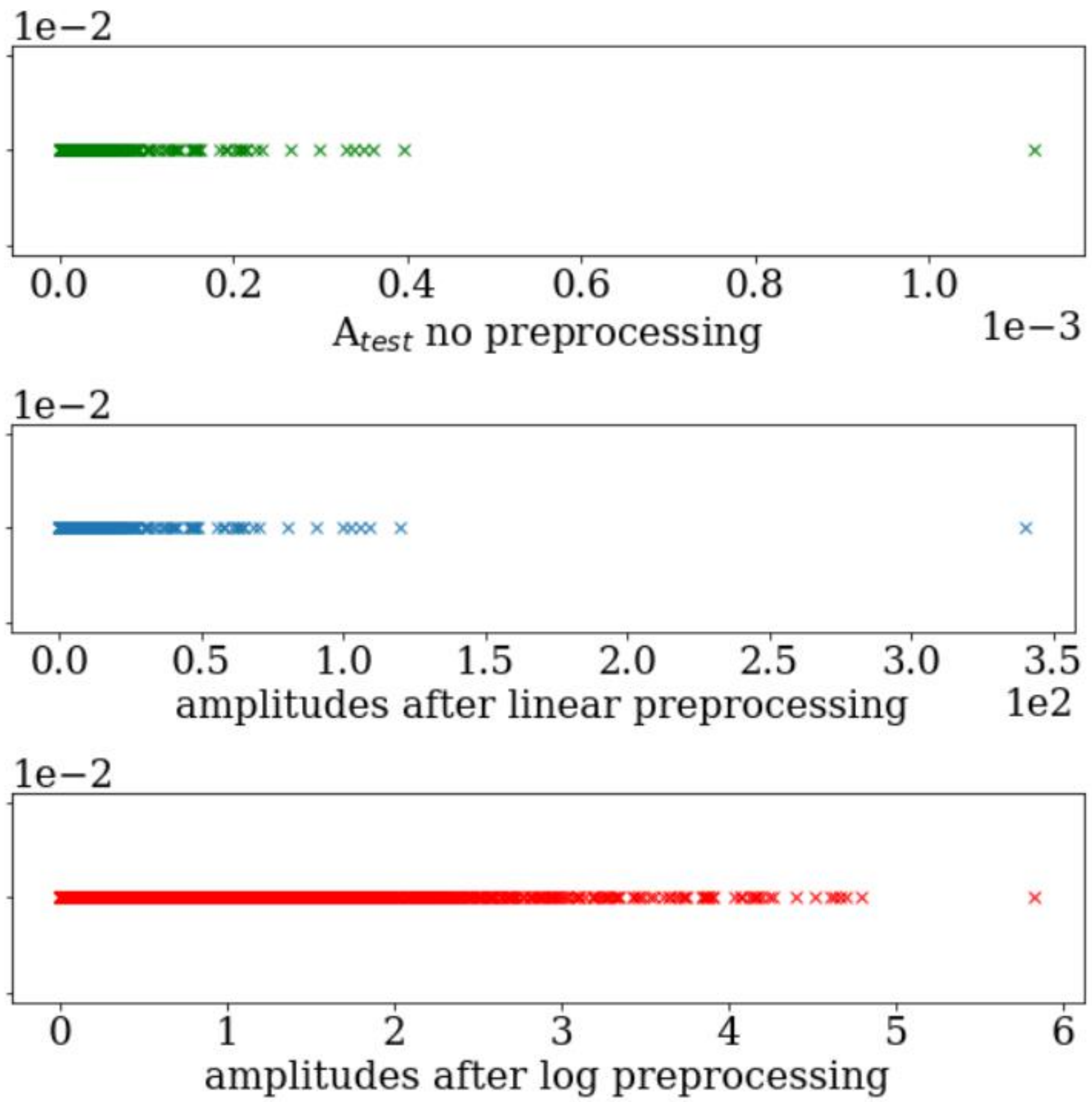
- **Processes:**
  - $gg \rightarrow \gamma\gamma + jets$
  - 1 gluon jet process → analytic solutions knows
  - 2 gluon jet process → only numeric solution
- **Setup:**
  - fully connected dense network with 4 inner layers and ~30 units per layer
  - optimiser: Adam
  - training data: ~30k, test data: ~300k
  - preprocessing: $A \rightarrow \log(A/\sigma_A + 1)$
  - phase space sampling: RAMBO

Preprocessing

- **Preprocessing reduces amount of outliers & normalizes data**

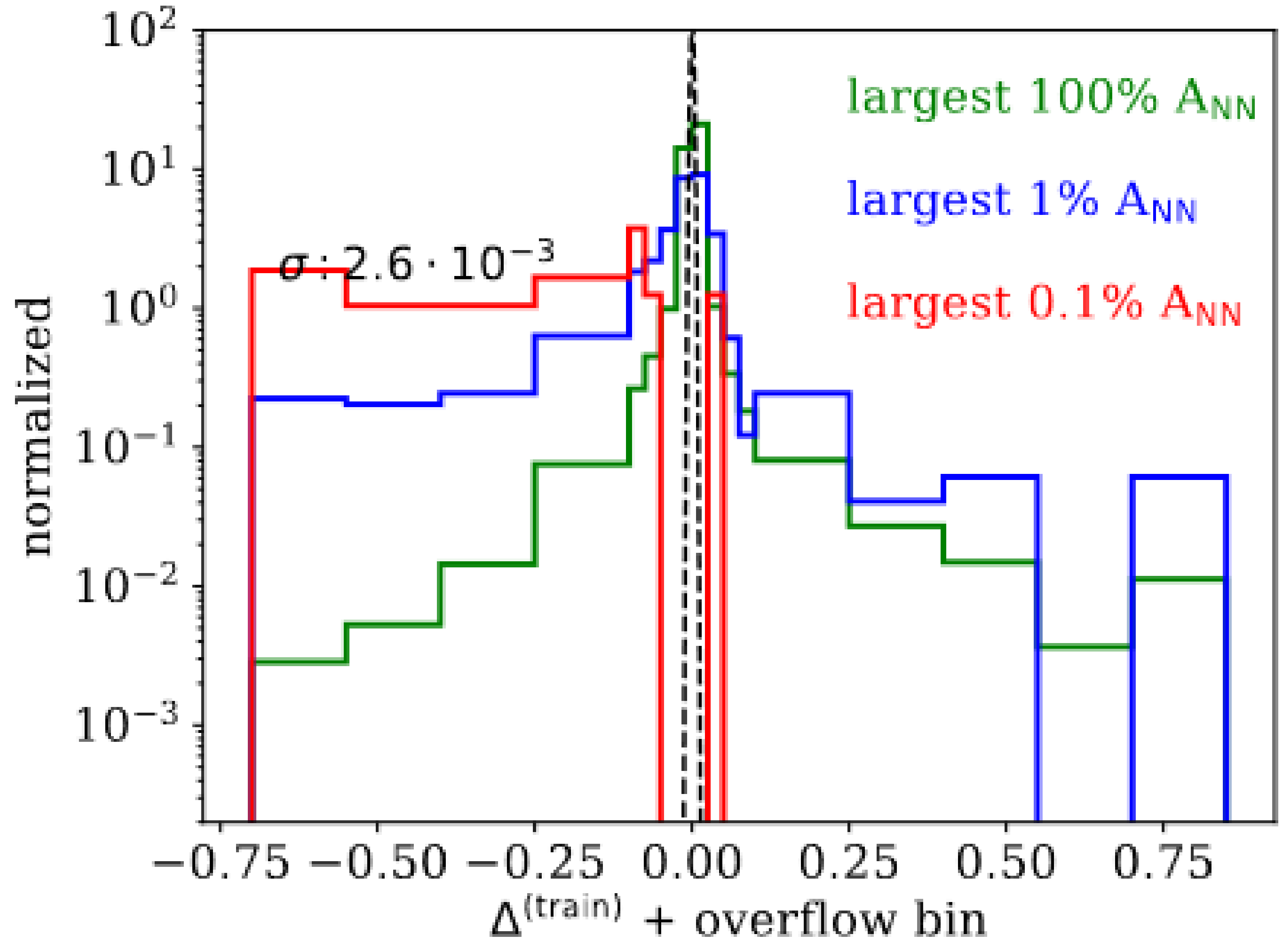
- **Linear:** $A \rightarrow \frac{A - \bar{A}}{\sigma_A}$

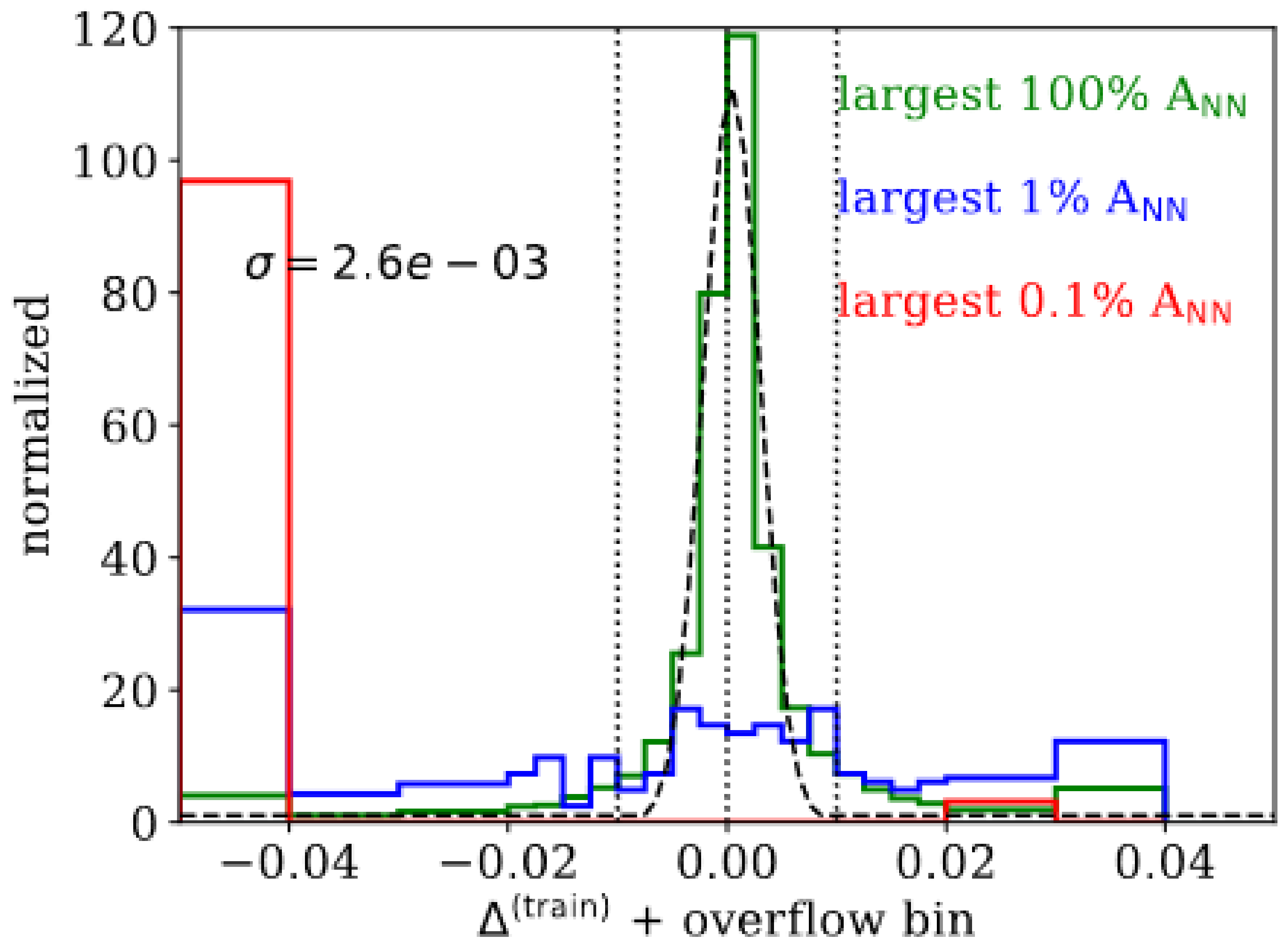
- **Log:** $A \rightarrow \log(A/\sigma_A + 1)$



Results

Performance and Uncertainties





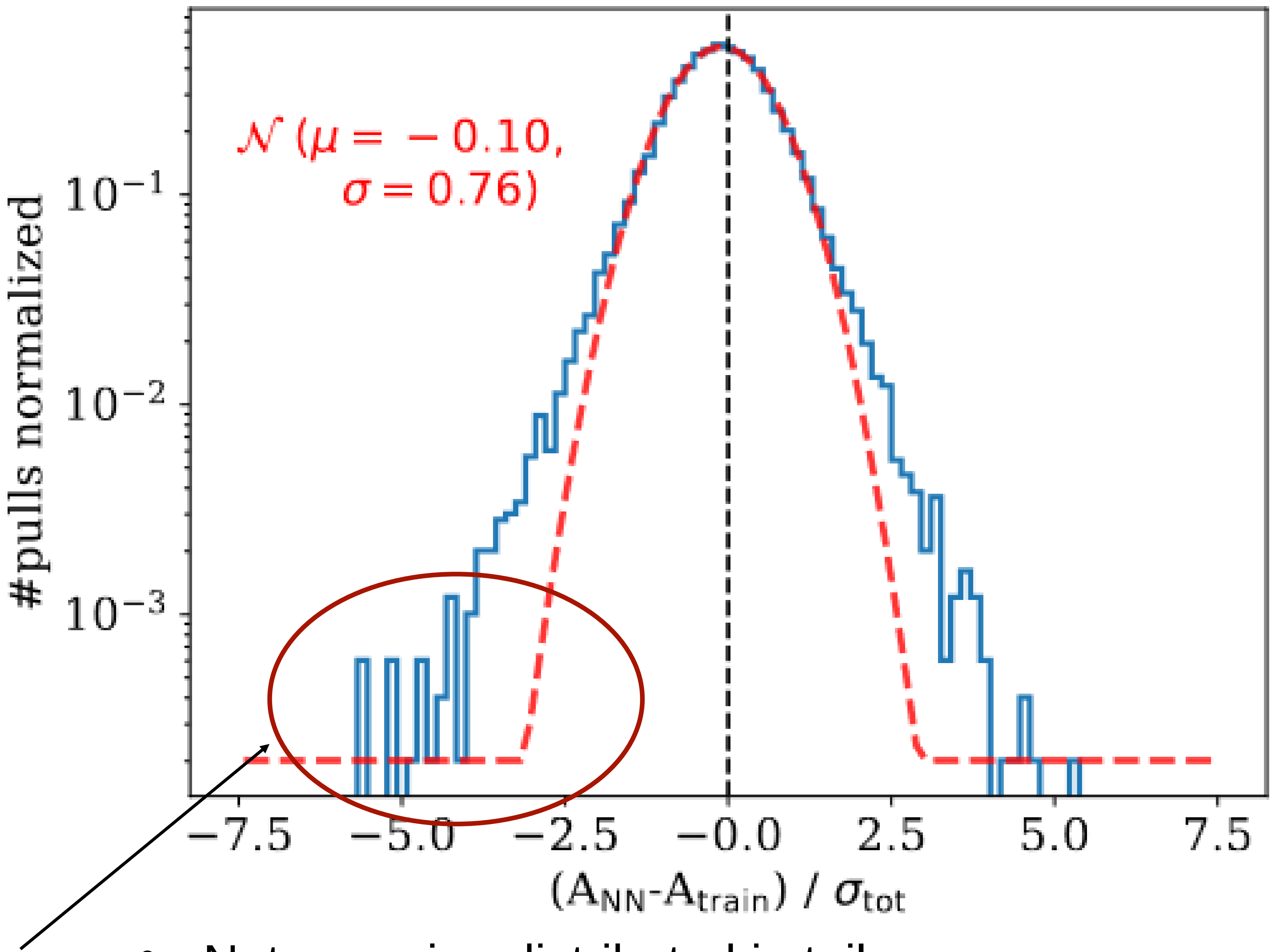
$$\Delta = \frac{A_{j,\mathrm{NN}}}{A_j} - 1$$

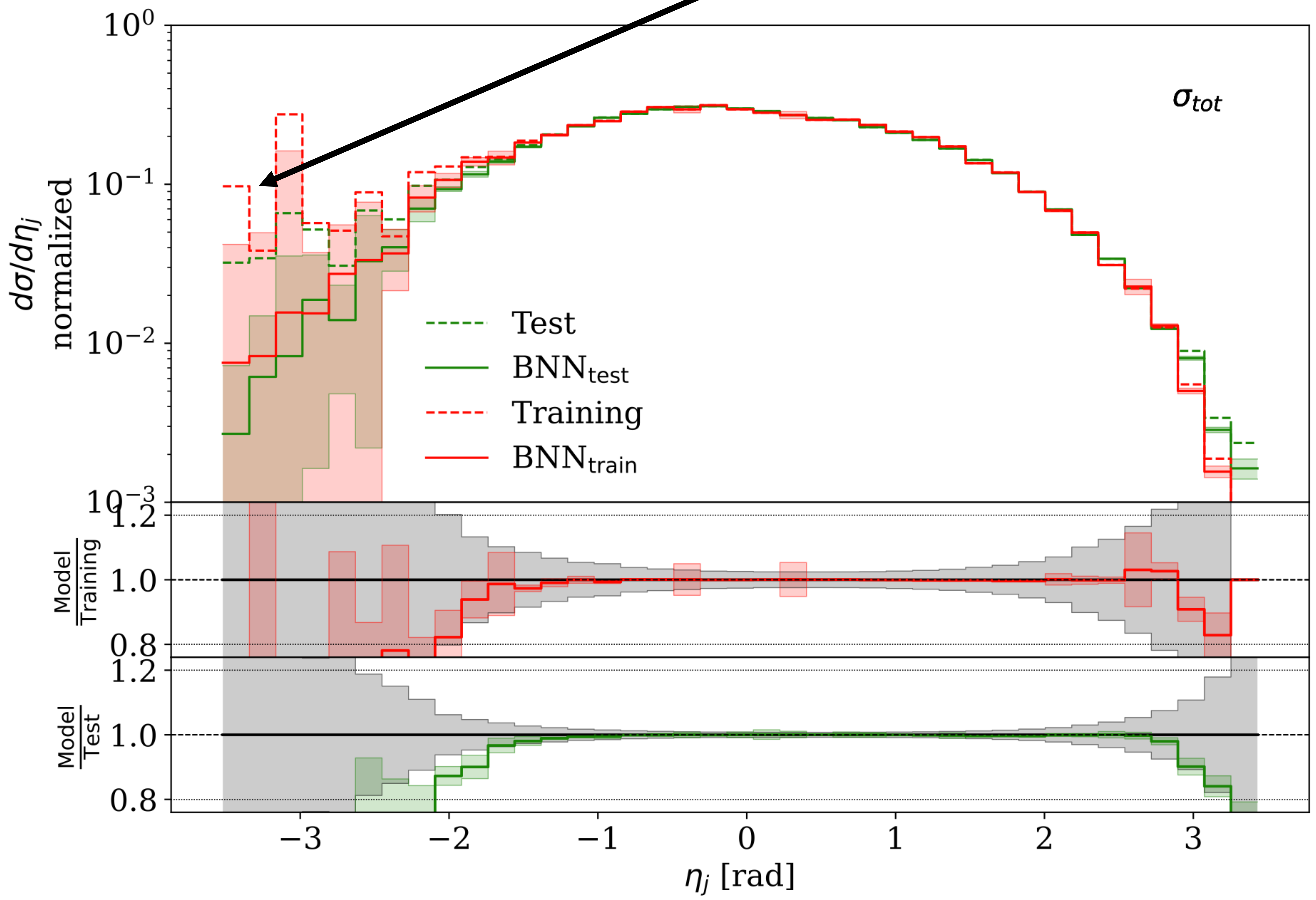
- Large amplitudes are not learnt well → events correspond to areas of IR divergencies
- Performance is good with typical deviation of ~0.3%

Results

“Pulls”/relative deviation & kinematic distributions

Dominated by large amplitudes (IR divergencies)



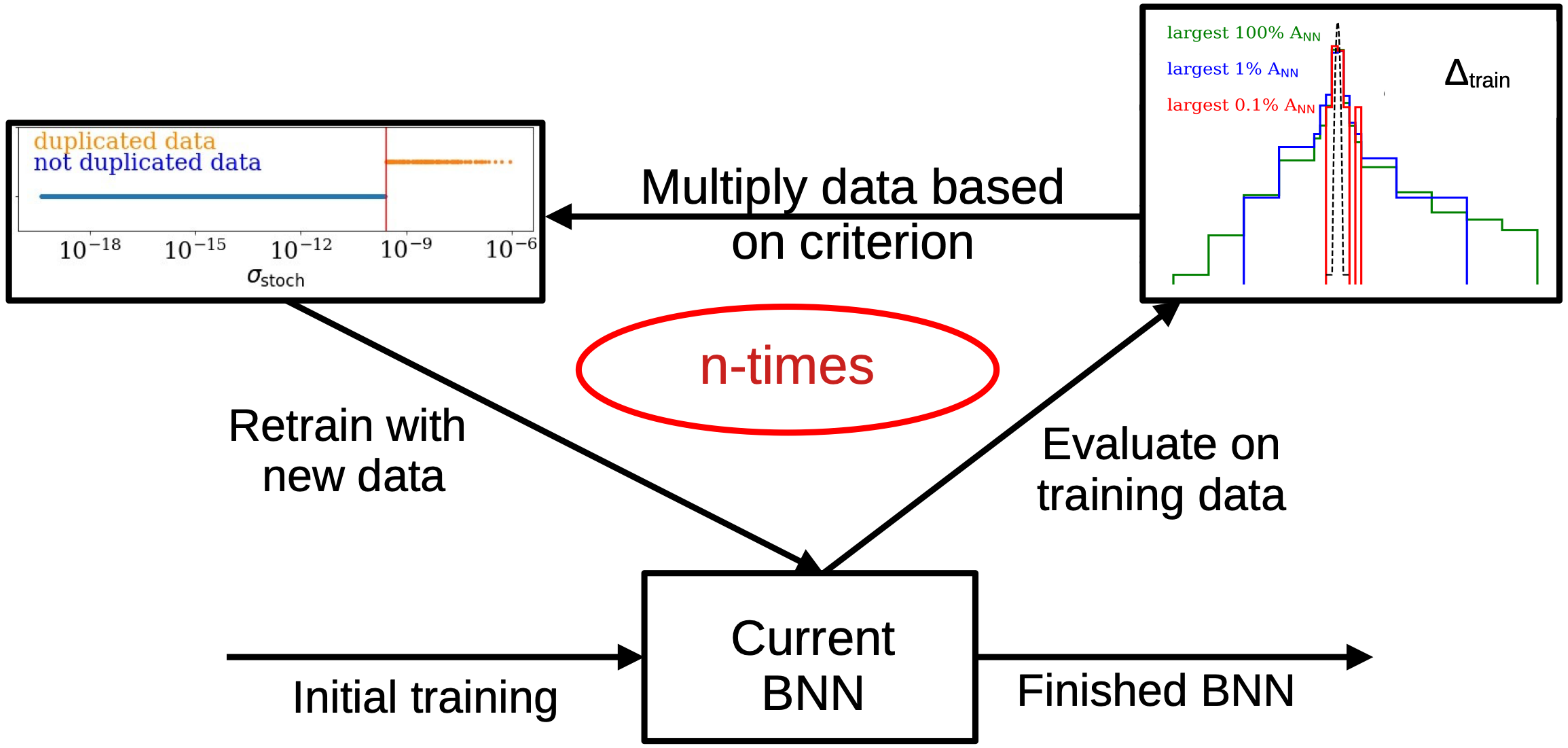


- Not gaussian distributed in tails
- Conservative estimate by total uncertainty
- Large amplitudes are present in regions of low statistics!

Freund, Yoav; Schapire, Robert E: Journal of Computer and System Sciences. 55: 119–139. CiteSeerX 10.1.1.32.8918. doi:10.1006/jcss.1997.1504

Boosting / feedback training

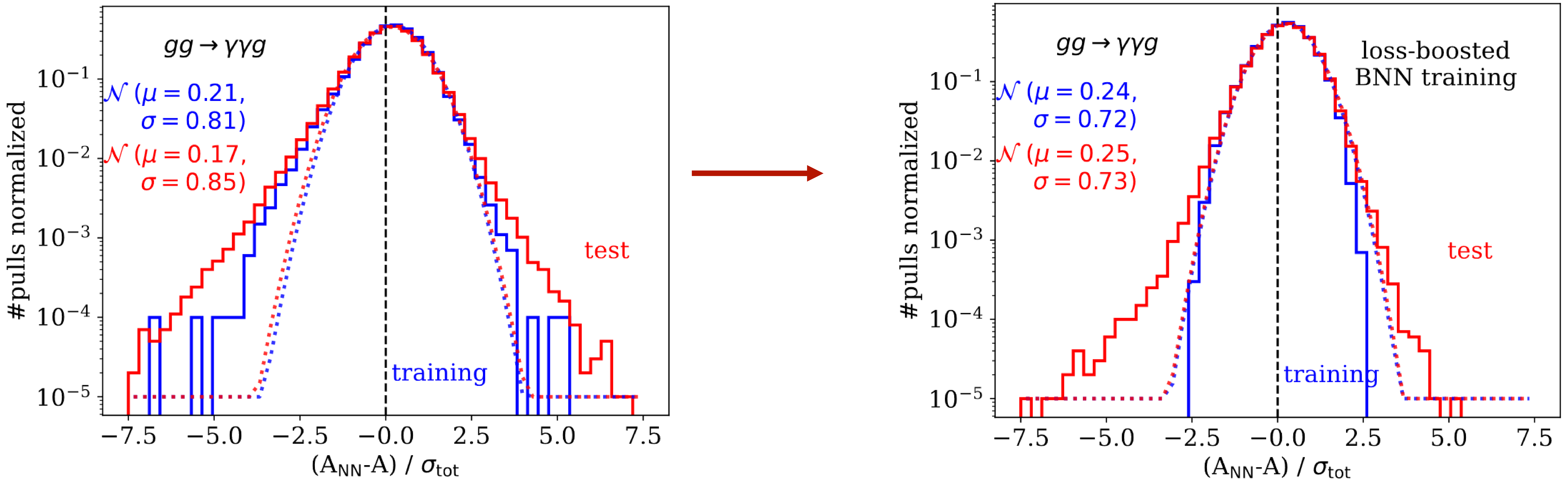
- Let's use the BNN uncertainties to improve the performance
- Similar idea as AdaGraph algorithm for BDT changing weight of events based on performance
- Criteria: can be chosen depending on required improvement



Loss loss / feedback training

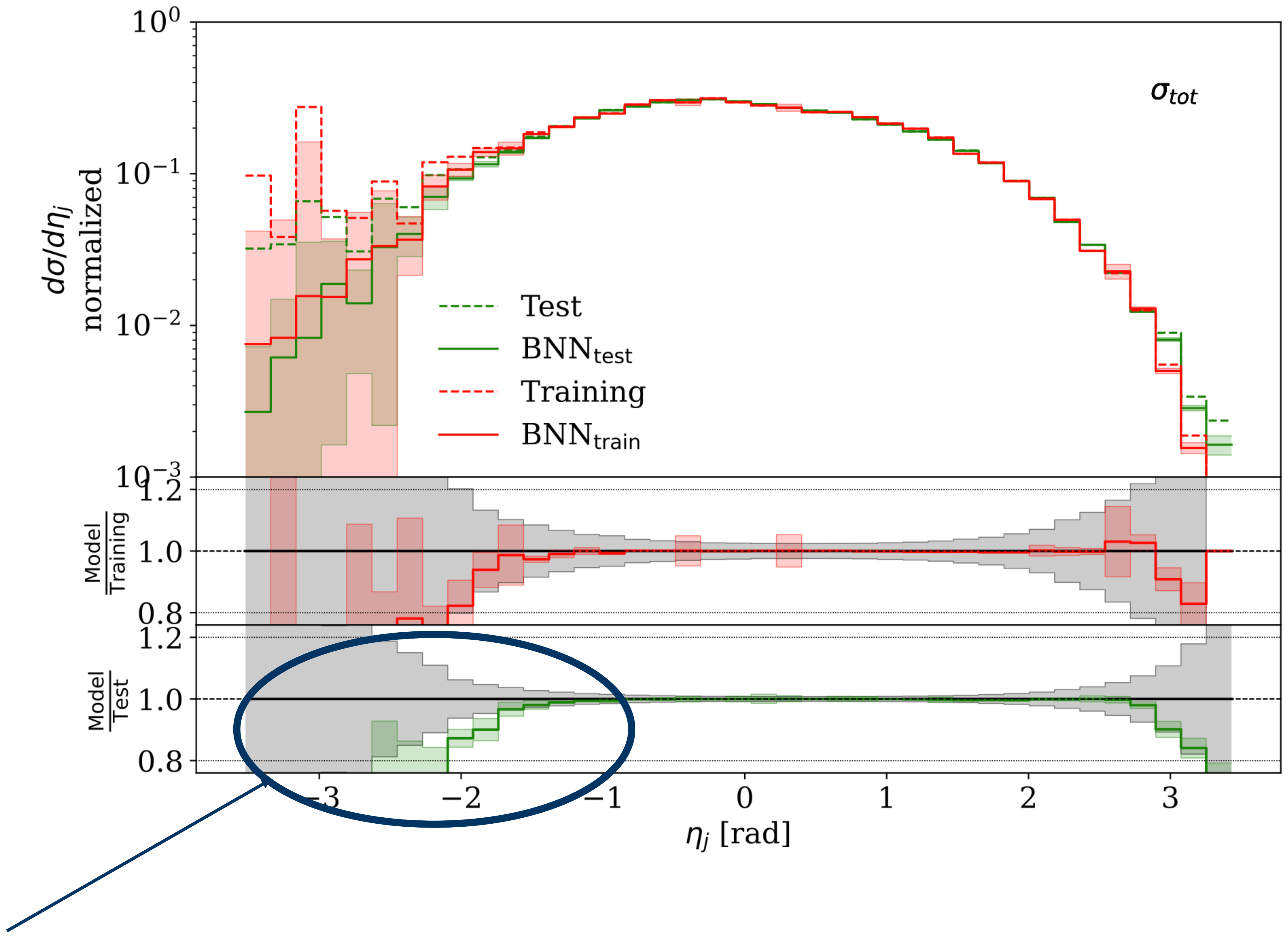
Using the BNN uncertainties to improve the "Pulls"

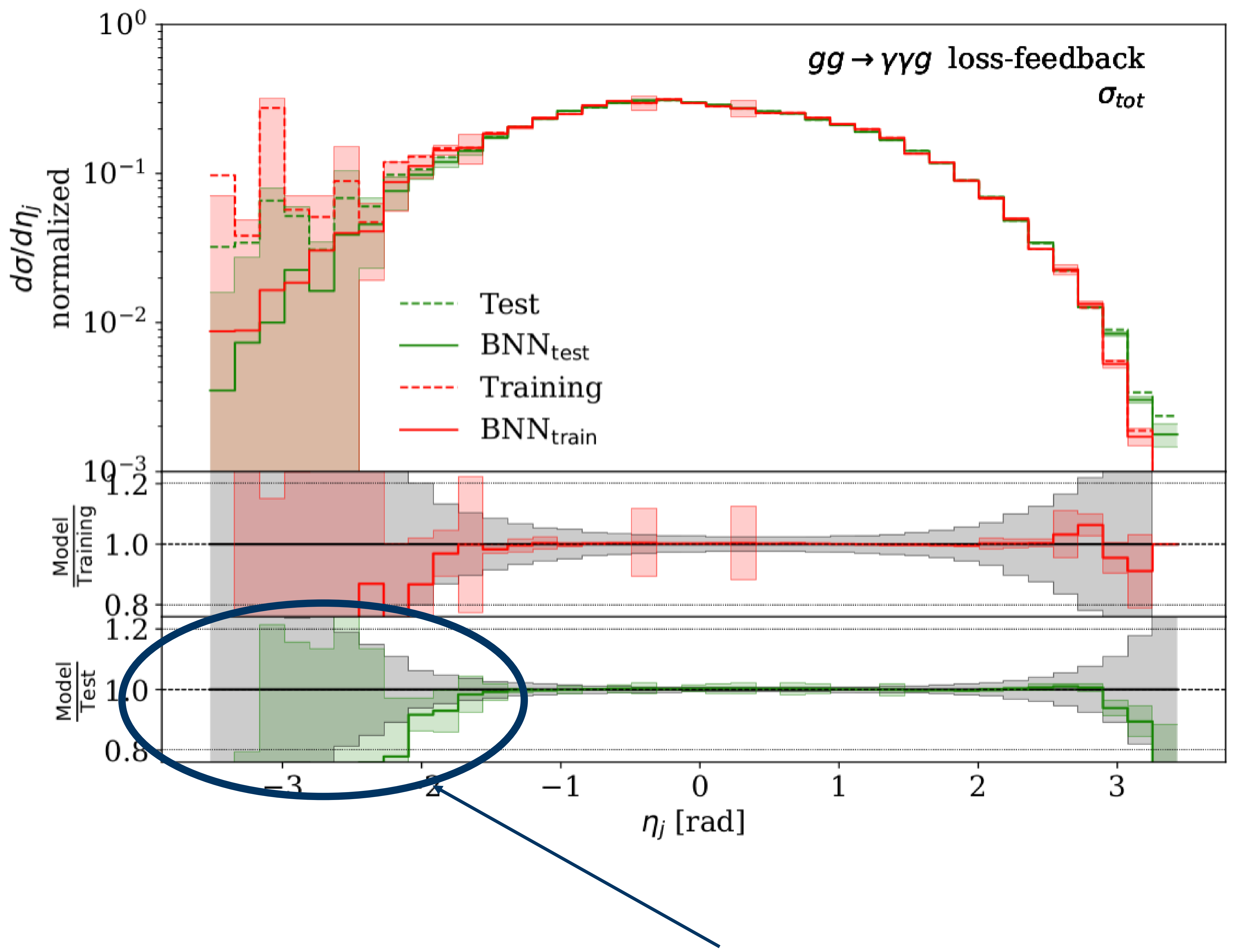
- Let's use the BNN uncertainties to improve the performance of the relative deviation
- Criteria: Select problematic training data in the tails of the "pull" distributions
- No visible change to performance



Loss boosting

Using the uncertainty estimates to improve the performance



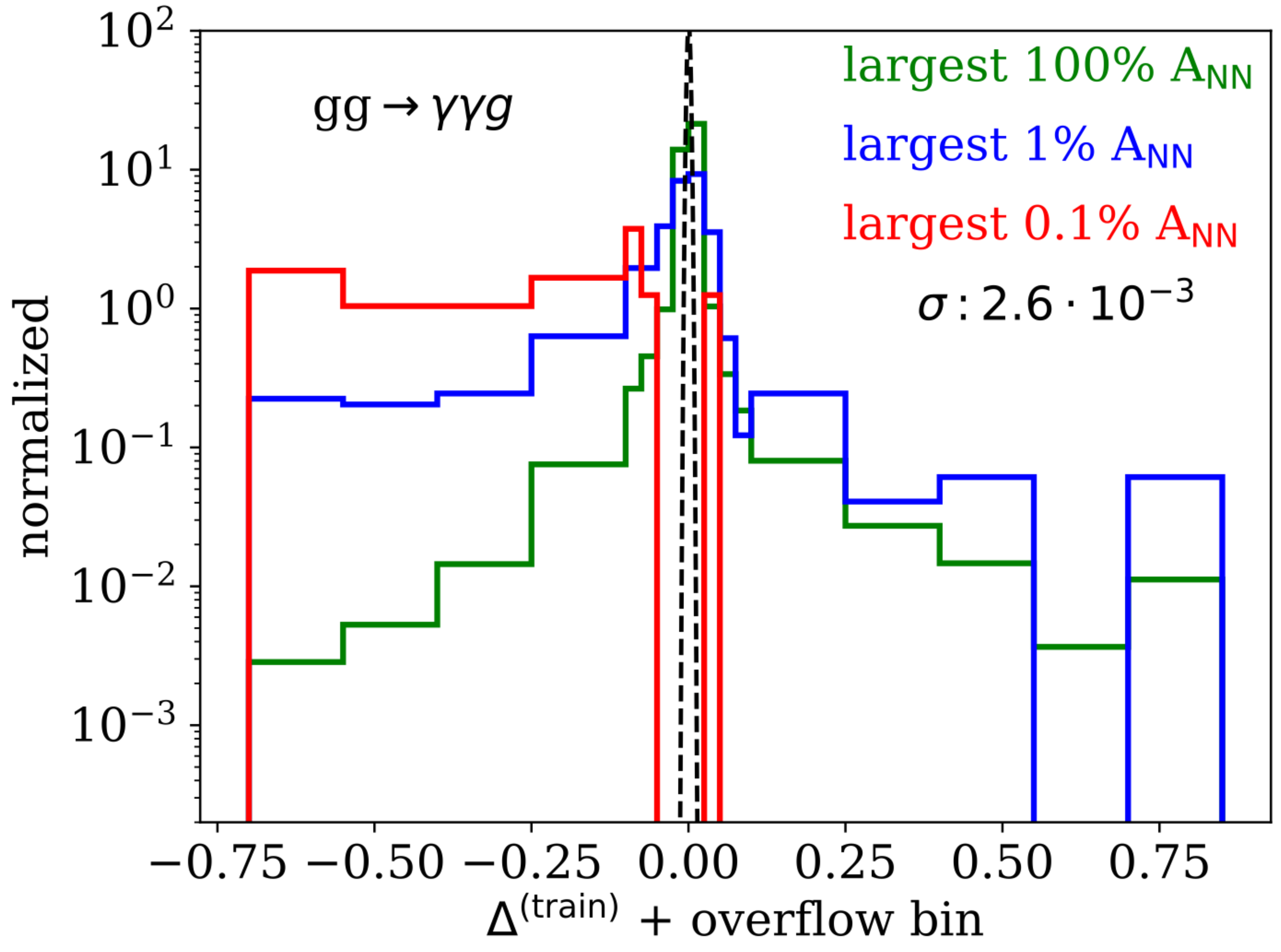


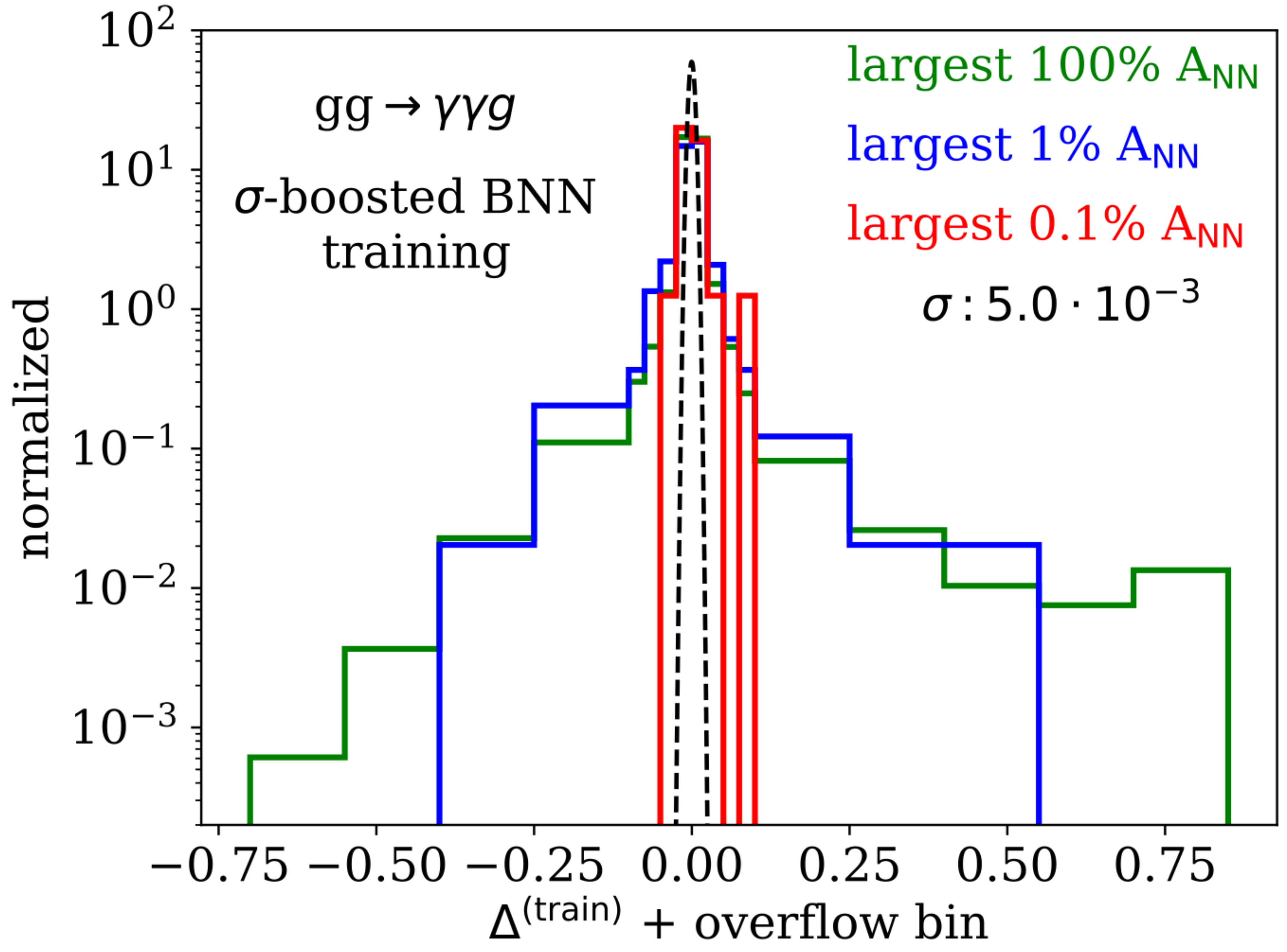
- Significant improvement of the uncertainties of test and training data

Process boosting / feedback training

Using the BNN uncertainties to improve the performance

- Let’s use the BNN uncertainties to also improve the performance
- Criteria: Select training examples with large uncertainties and emphasise them in a follow-up training

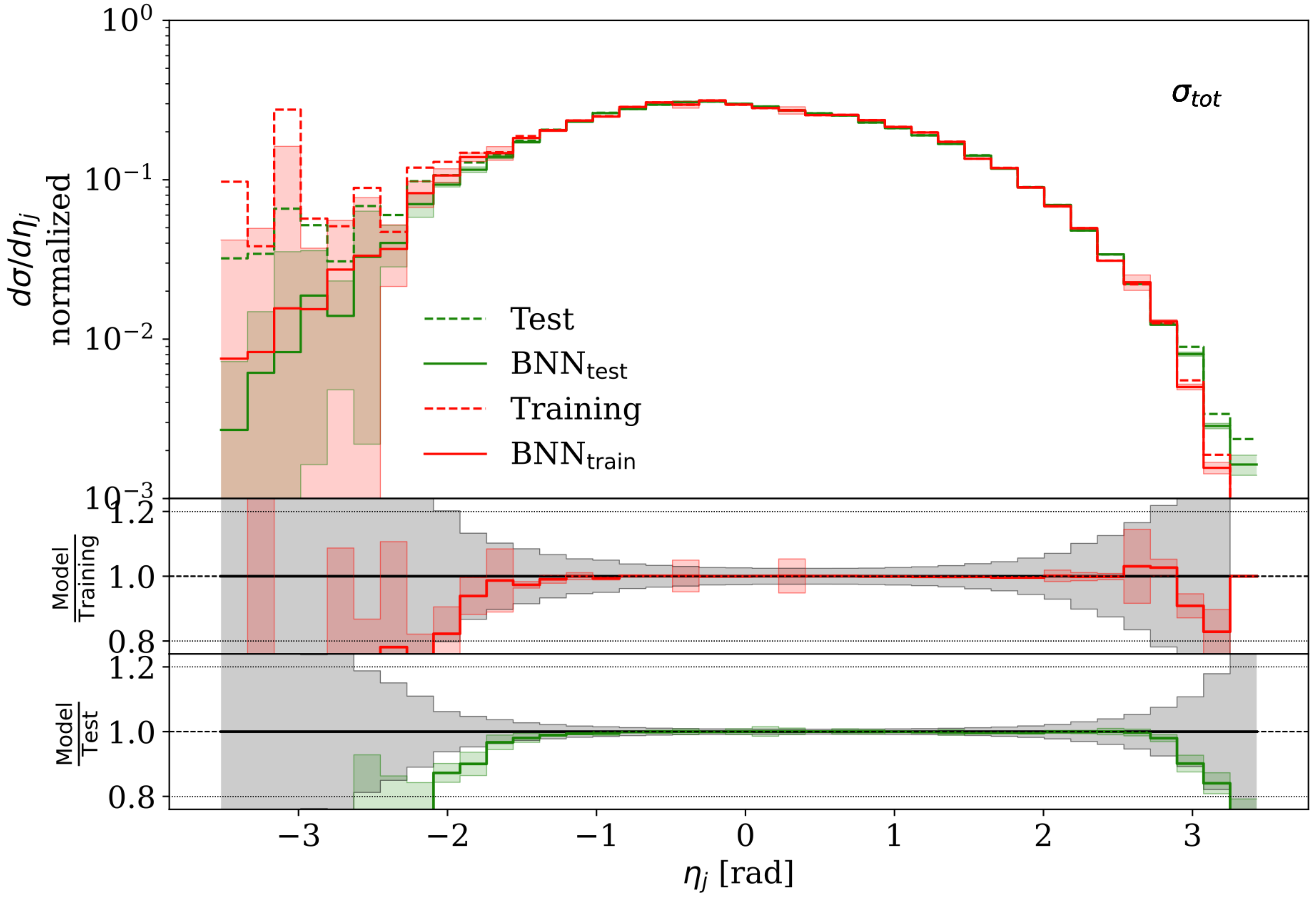


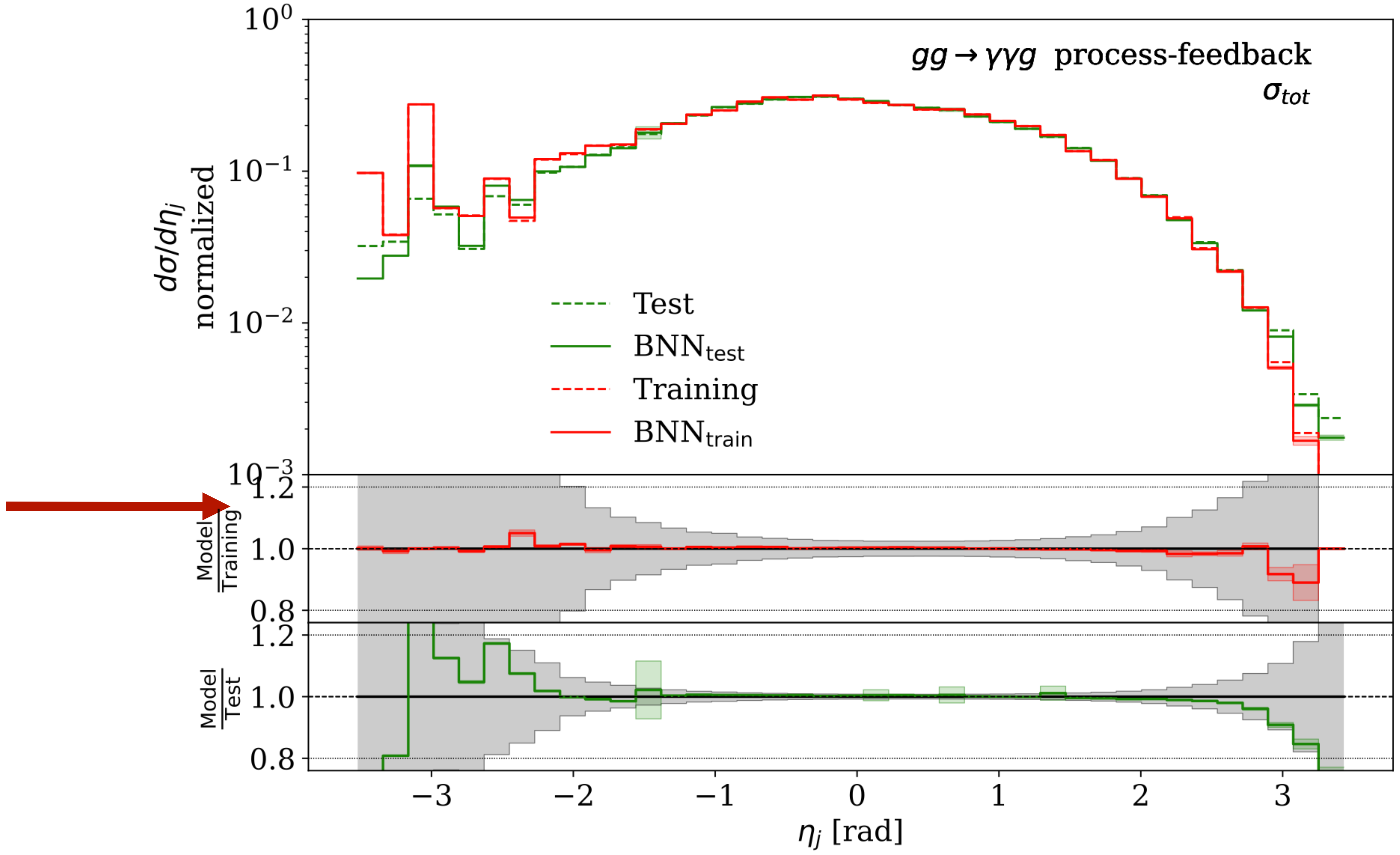


Predictions of large amplitudes improved significantly

Process boosting / feedback training

Using the BNN uncertainties to improve the performance





- Improvement is visible in kinematic distributions for training and test data
- Not really scared of overtraining because amplitudes are noise-free (interpolation vs. fit)
- For test data: To get even better we would have to generate more training data

Conclusion and outlook

- BNNs provide uncertainty estimates needed to integrate ML tools in simulation chain
- We can use uncertainties to improve performance and uncertainties further
- **Next step**: optimize set-up on events with higher multiplicity
- Improve feedback training further: Generate new training data in regions with large uncertainties?

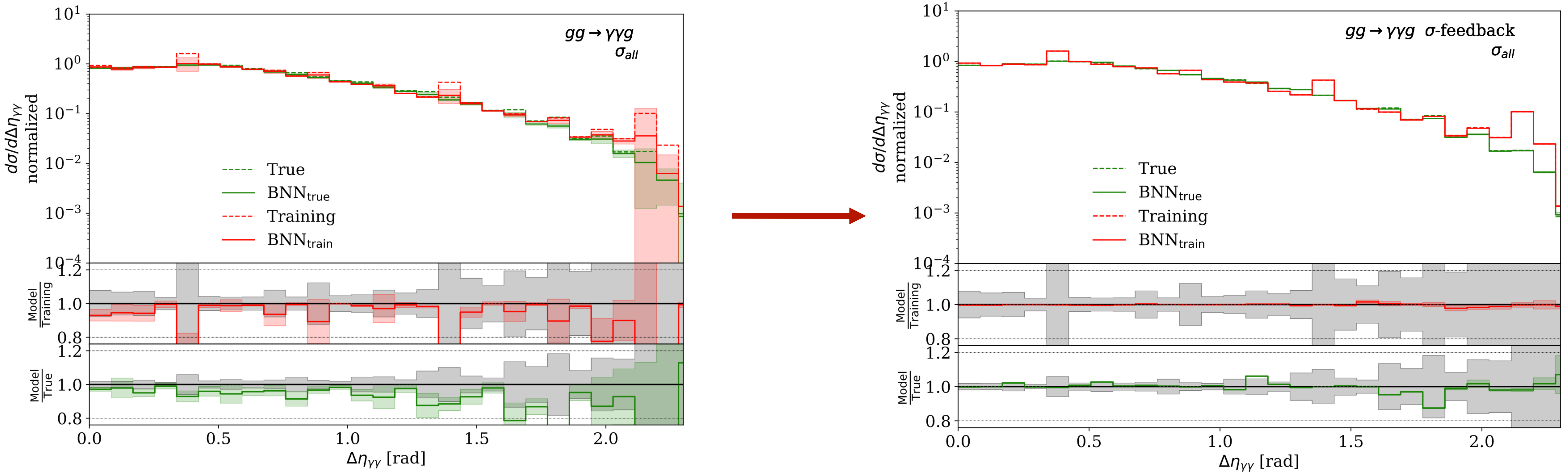
Backup: Uncertainties

Uncertainties detailed

$$
\begin{aligned}
\sigma^2_{\rm tot} &= \langle (A - \langle A \rangle)^2 \rangle \\
&= \int dA\ (A - \langle A \rangle)^2\ p(A|T) \\
&= \int dA d\omega\ (A - \langle A \rangle)^2\ p(A|\omega, T)\ q(\omega) \\
&= \int dA d\omega\ \left(A^2 - 2A\langle A \rangle + \langle A \rangle^2\right)\ p(A|\omega, T)\ q(\omega) \\
&= \int d\omega\ q(\omega) \left[\int dA\ A^2\ p(A|\omega, T) - 2 \int dA\ A\langle A \rangle\ p(A|\omega, T) + \int dA\ \langle A \rangle^2\ p(A|\omega, T) \right] \\
&= \int d\omega\ q(\omega) \left[\overline{A^2}(\omega) - 2\langle A \rangle \overline{A}(\omega) + \langle A \rangle^2 \right] \\
&= \int d\omega\ q(\omega) \left[\overline{A^2}(\omega) - \overline{A}(\omega)^2 + \overline{A}(\omega)^2 - 2\langle A \rangle \overline{A}(\omega) + \langle A \rangle^2 \right] \\
&= \int d\omega\ q(\omega) \left[\overline{A^2}(\omega) - \overline{A}(\omega)^2 + \left(\overline{A}(\omega) - \langle A \rangle\right)^2 \right] \equiv \sigma^2_{\rm stoch} + \sigma^2_{\rm pred}\ .
\end{aligned}
$$

Backup: Process boosting

Using the BNN uncertainties to improve the performance



Backup: Bayesian neural networks

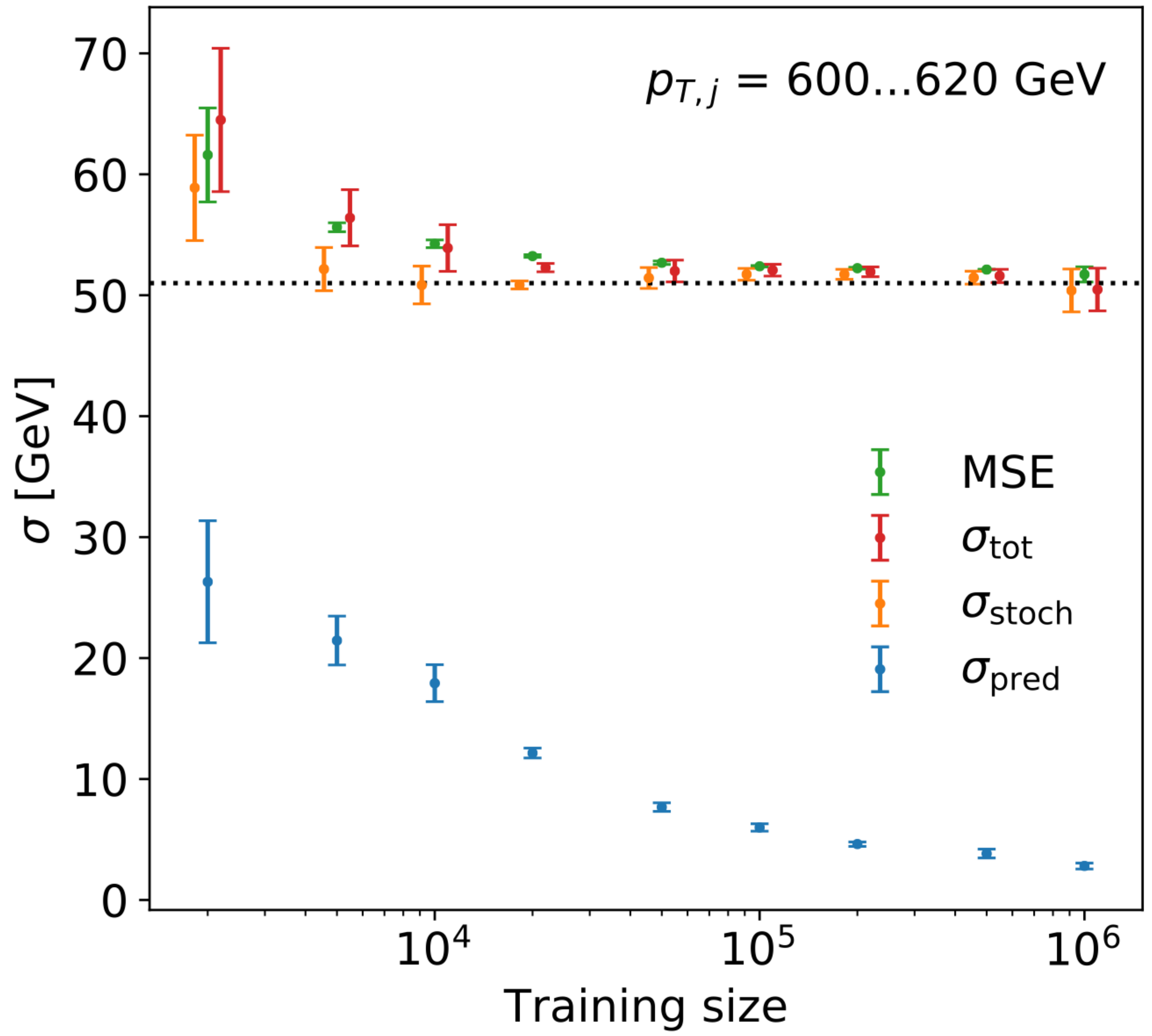
Uncertainties for regression

- BNN: Can split total BNN uncertainty into two uncertainties.

$$\sigma_{\text{tot}}^2 = \sigma_{\text{stoch/model}}^2 + \sigma_{\text{pred}}^2 \sim \text{MSE}$$

stochasticity of data, limited expressivity of model

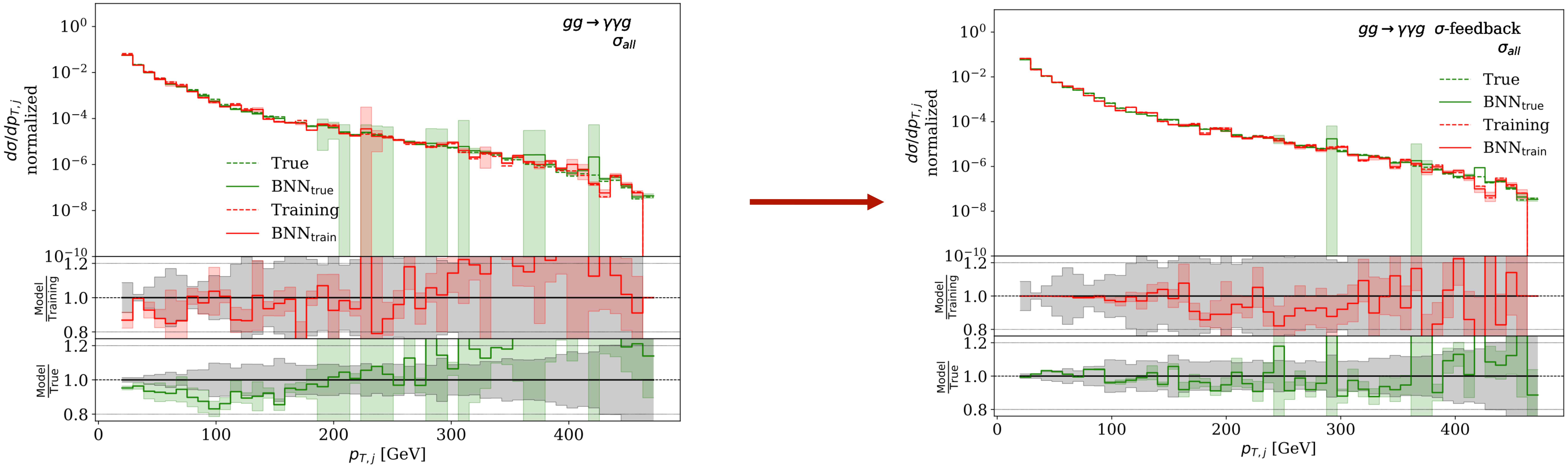
Goes to zero for large training-size



G. Kasieczka, M. L., F. Otterpohl and T. arXiv:2003.11099 [hep-ph]

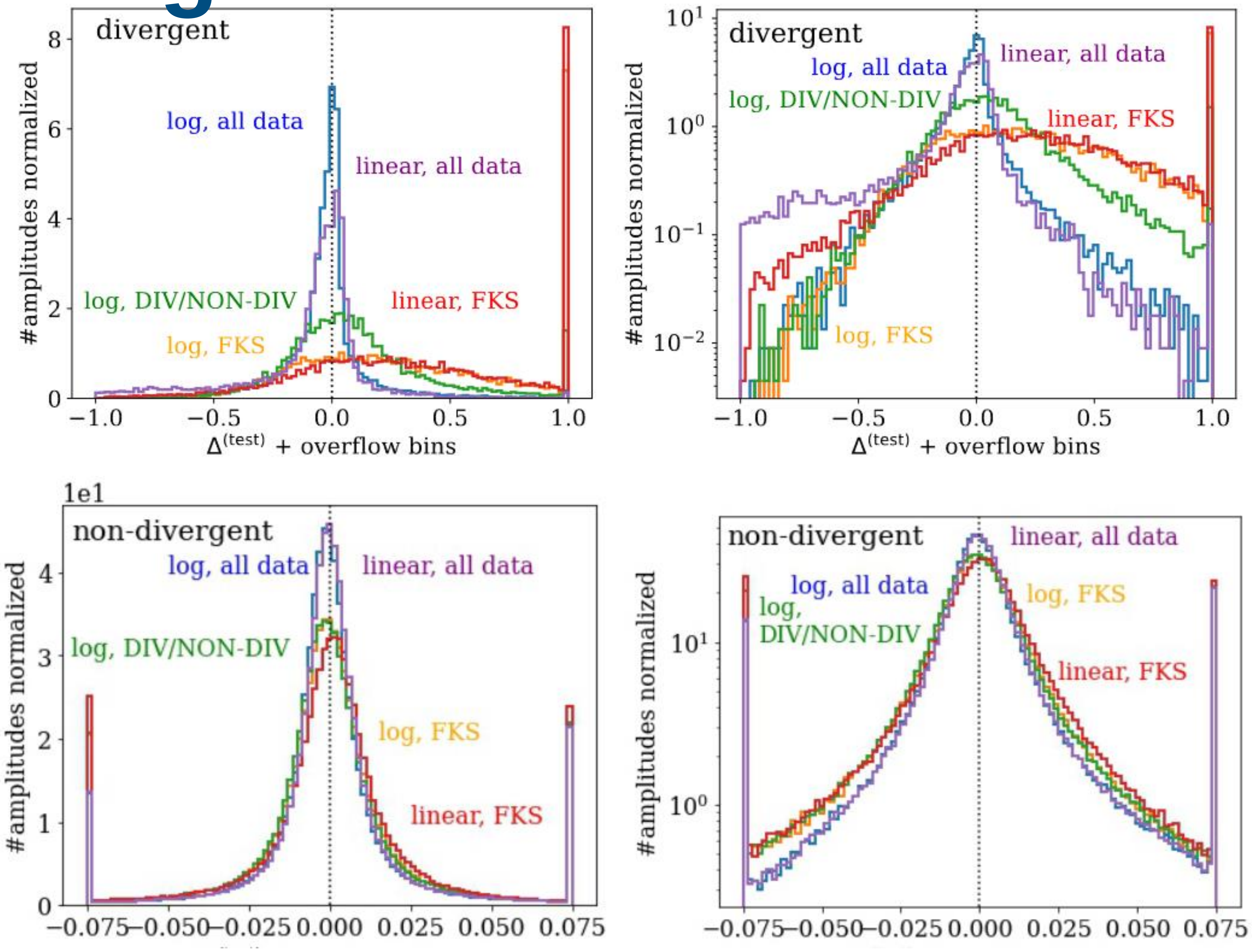
Backup: Process boosting

Using the uncertainty estimates to improve the performance



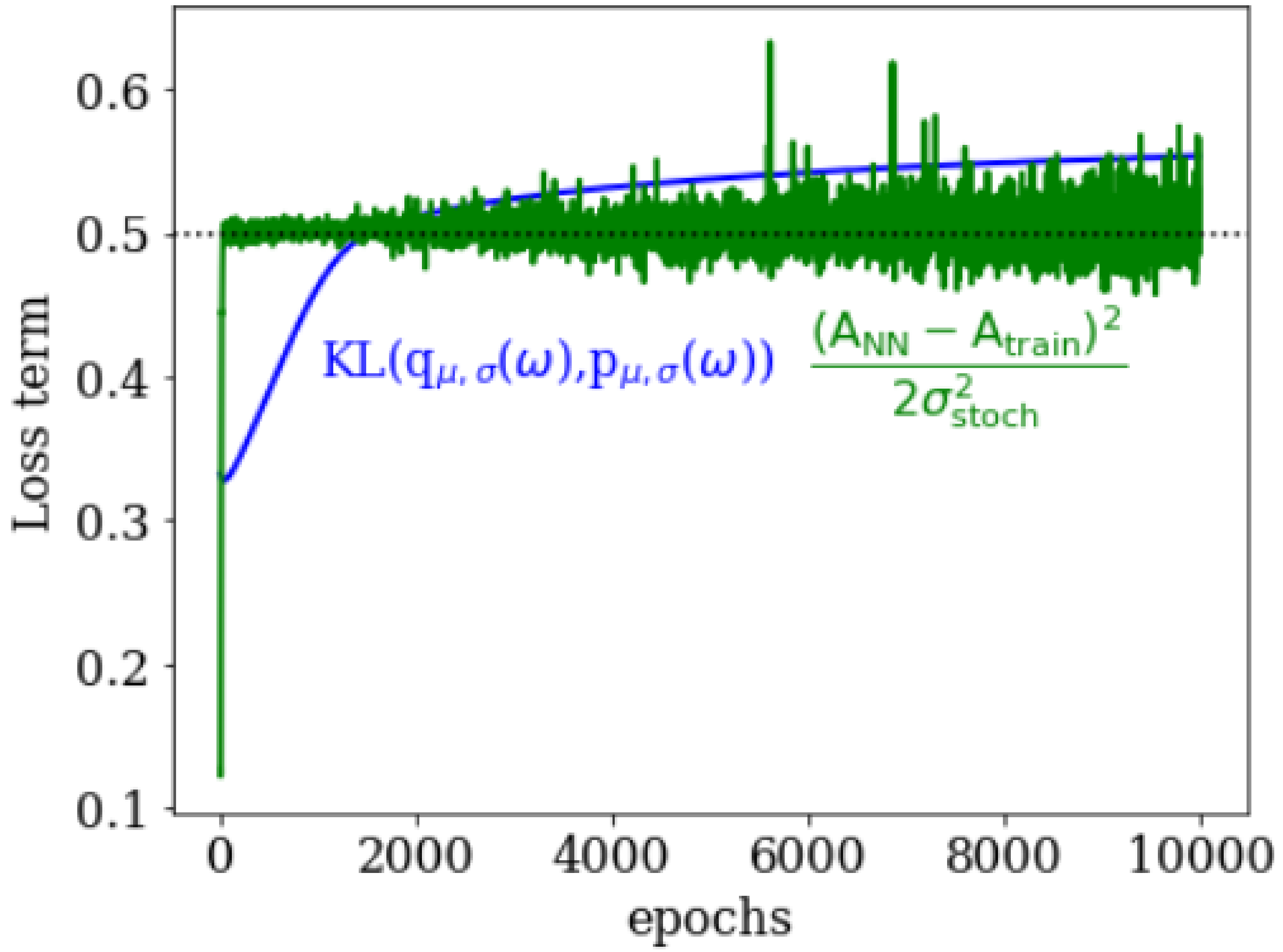
Backup: Preprocessing

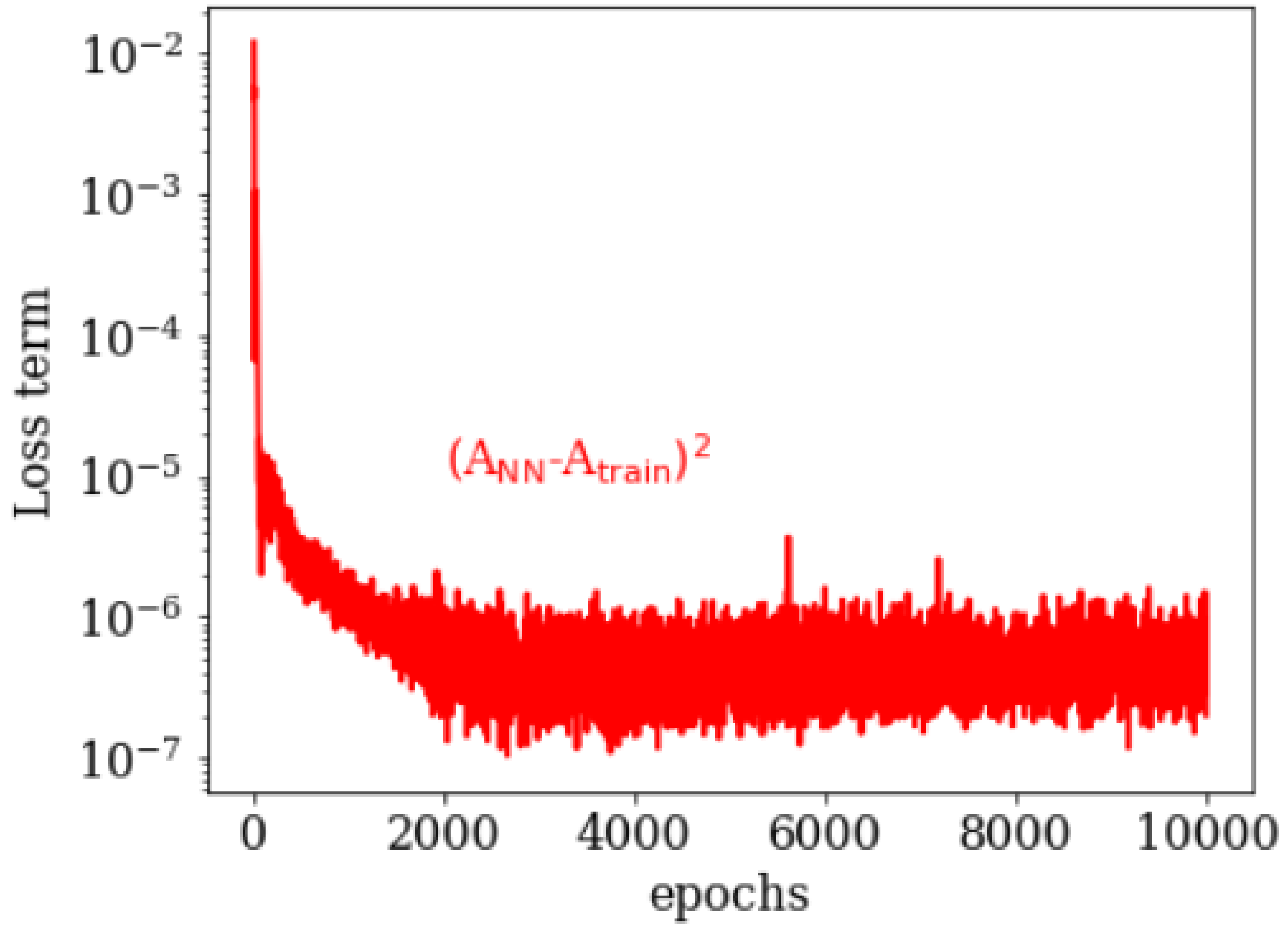
- **Preprocessing reduces amount of outliers & normalizes data**
- **Different preprocessing:**
  - **Split training of divergent and non-divergent data**
  - **Split data using FKS-Subtraction method**
- **Performance better on all data**



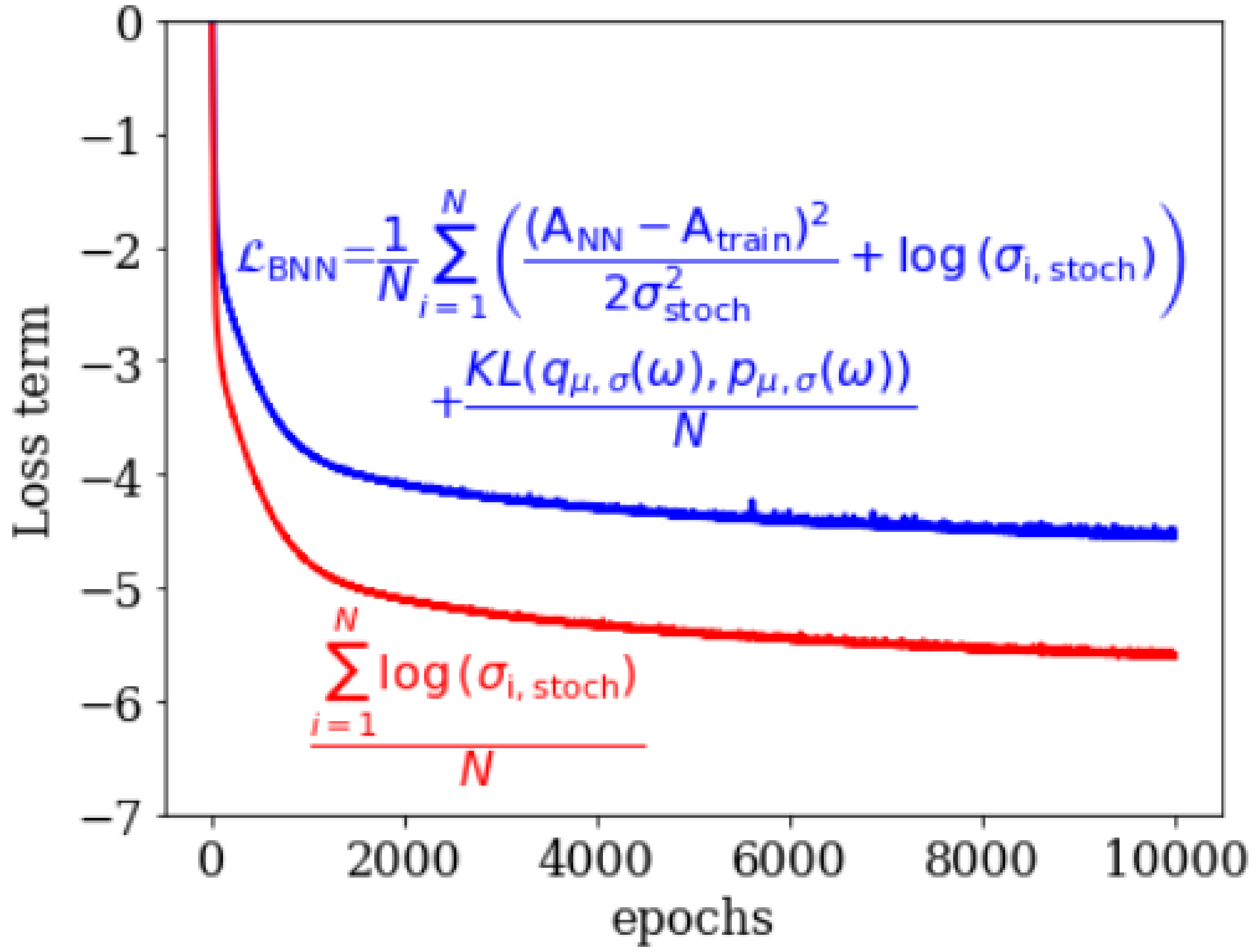
FKS: Frixione, Kunszt, Signer (1969)
More: Rikkert Frederix *et al* JHEP10(2009)003

Backup: Loss of specific terms





- Most amplitudes trained quickly
- Weight distributions (σ_{pred}) still changes



Backup: FKS

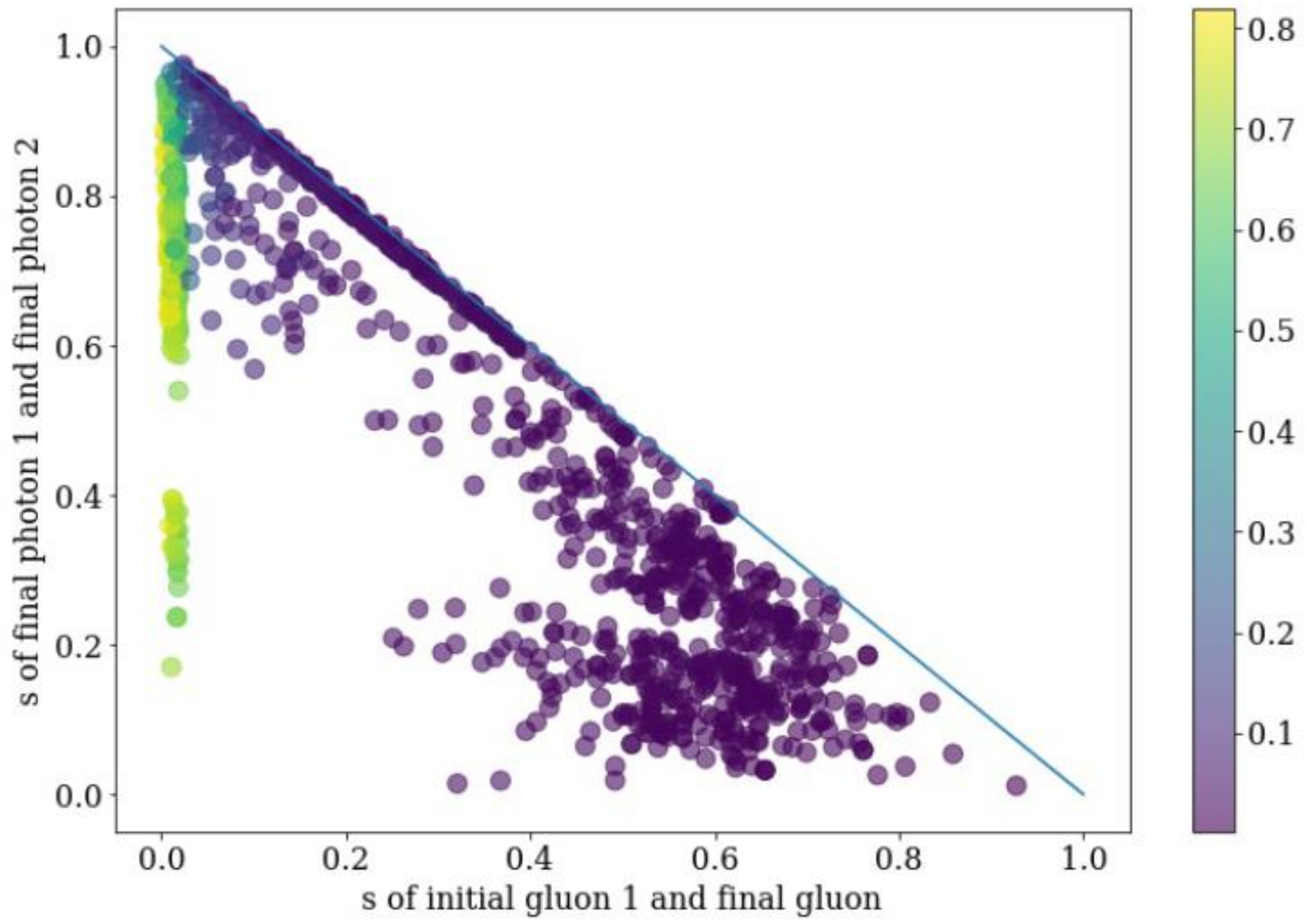
$$\mathcal{P}_{\text{FKS}} = \{(i,j)|1 \le i \le n, 2 \le j \le n, i \ne j, \\ \mathcal{M}^{(\text{n},0)} \text{ or } \mathcal{M}^{(\text{n},1)} \to \infty \text{ if } p_i^0 \to 0 or p_j^0 \to 0 \text{ or } \vec{p}_i || \vec{p}_j\} \tag{28}$$

With the partition sums:

$$S_{i,j} = \frac{1}{D_1 s_{ij}}, D_1 = \sum_{i,j \in \mathcal{P}_{FKS}} \frac{1}{s_{ij}} \tag{29}$$

such that

$$d\sigma = \sum_{i,j} S_{i,j} d\sigma \tag{30}$$



Examplatory partition/weighing function $S_{g,g}$ for first initial gluon and final gluon