Accuracy and Uncertainties for ML-Amplitudes

Nina Elmer

Nikhef Theory Seminar
19. December 2024

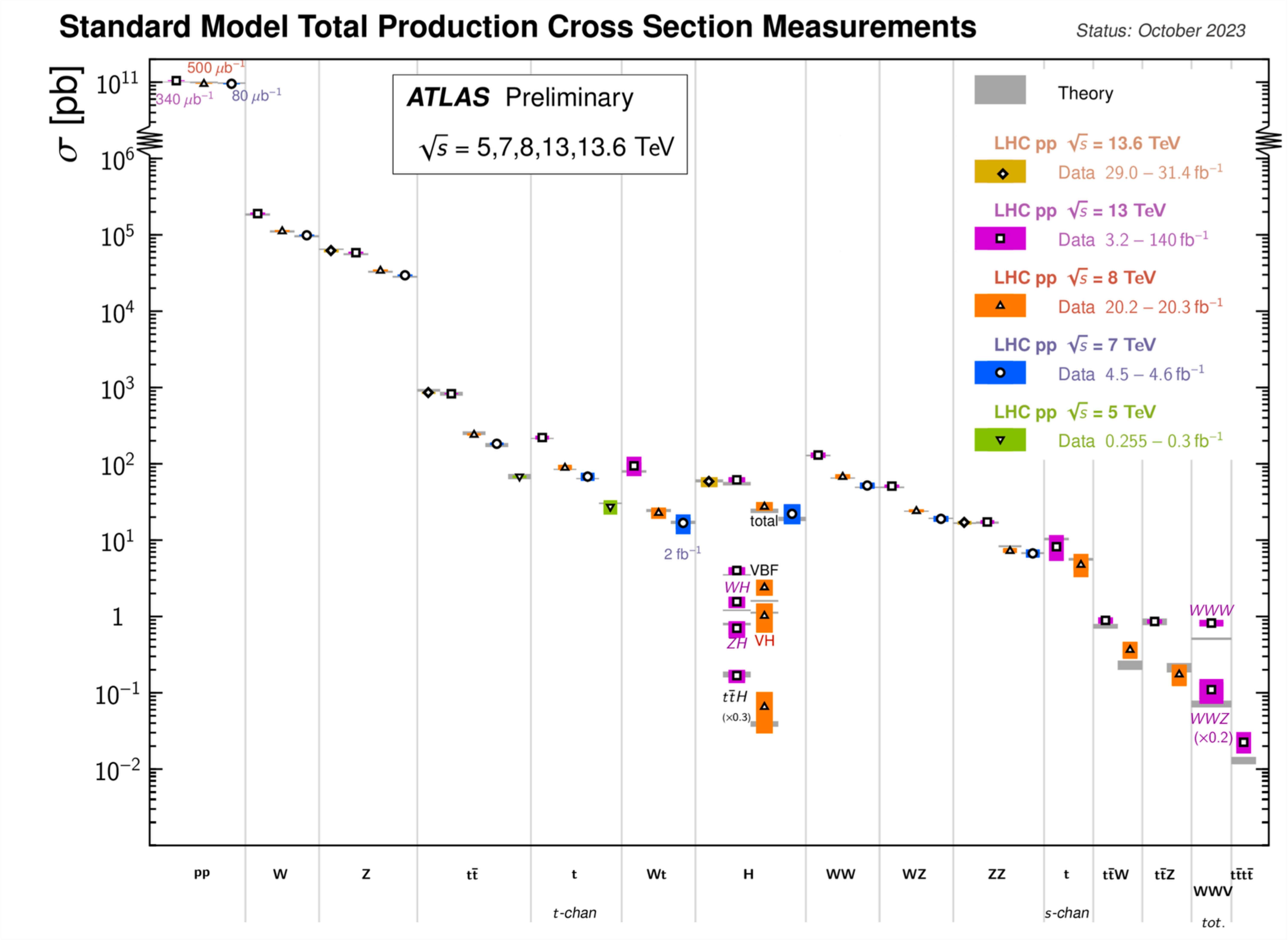
arXiv: 2412.12069

with H. Bahl, L. Favaro, M. Haußmann, T. Plehn, and R. Winterhalder

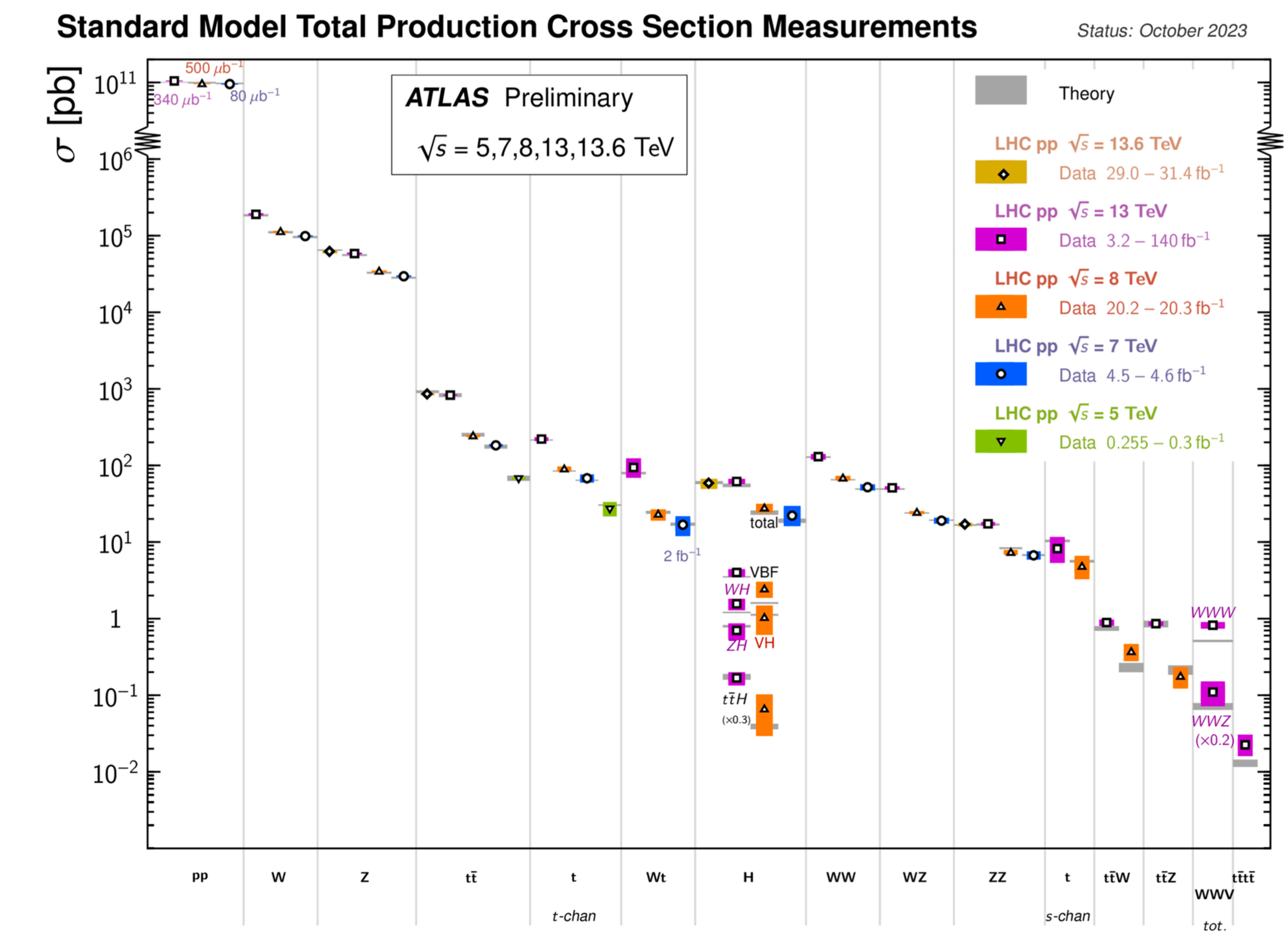
UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

IMPRS
for Precision Tests of Fundamental Symmetries
INTERNATIONAL MAX PLANCK RESEARCH SCHOOL

Entering the precision era

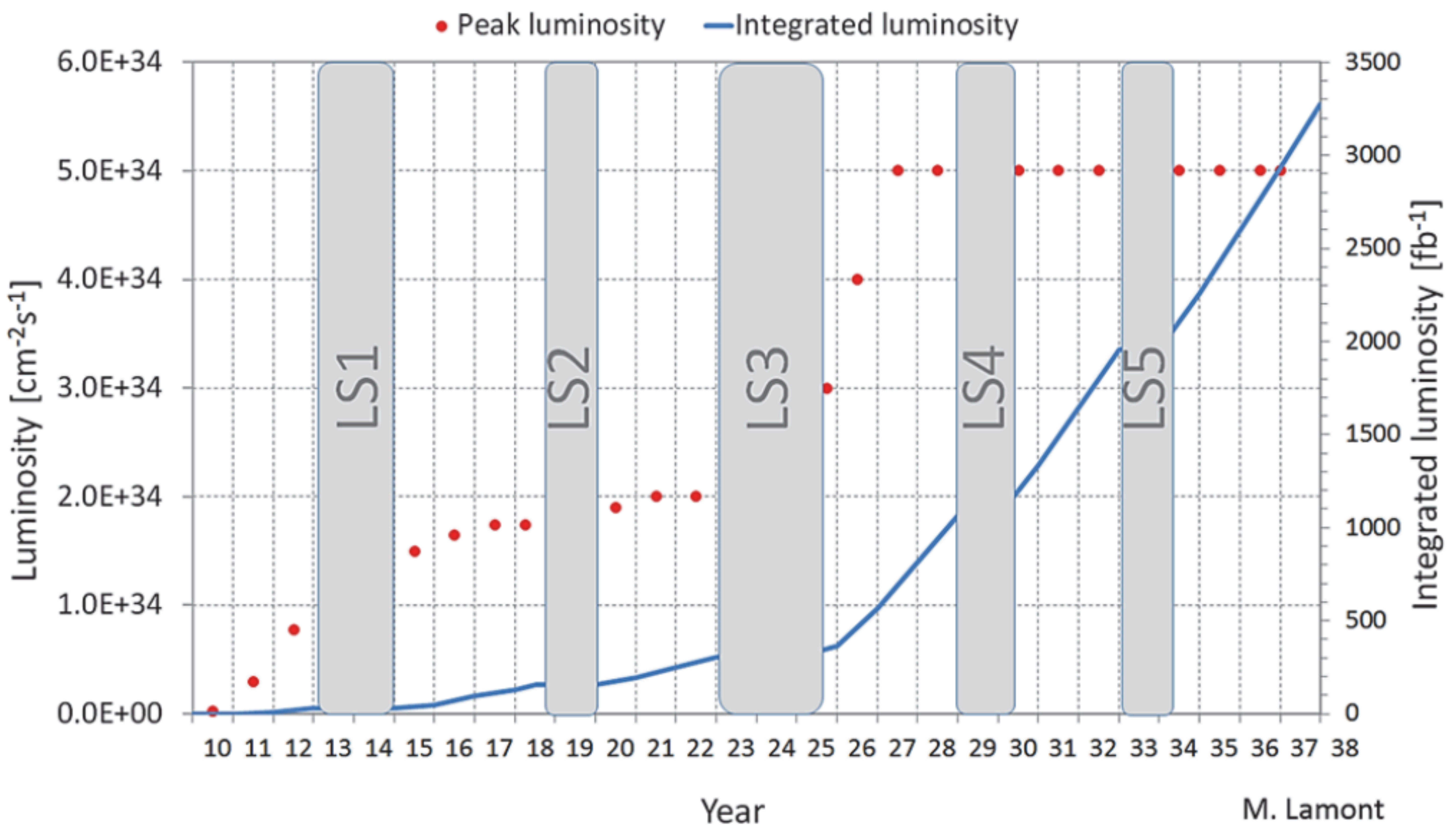


Entering the precision era



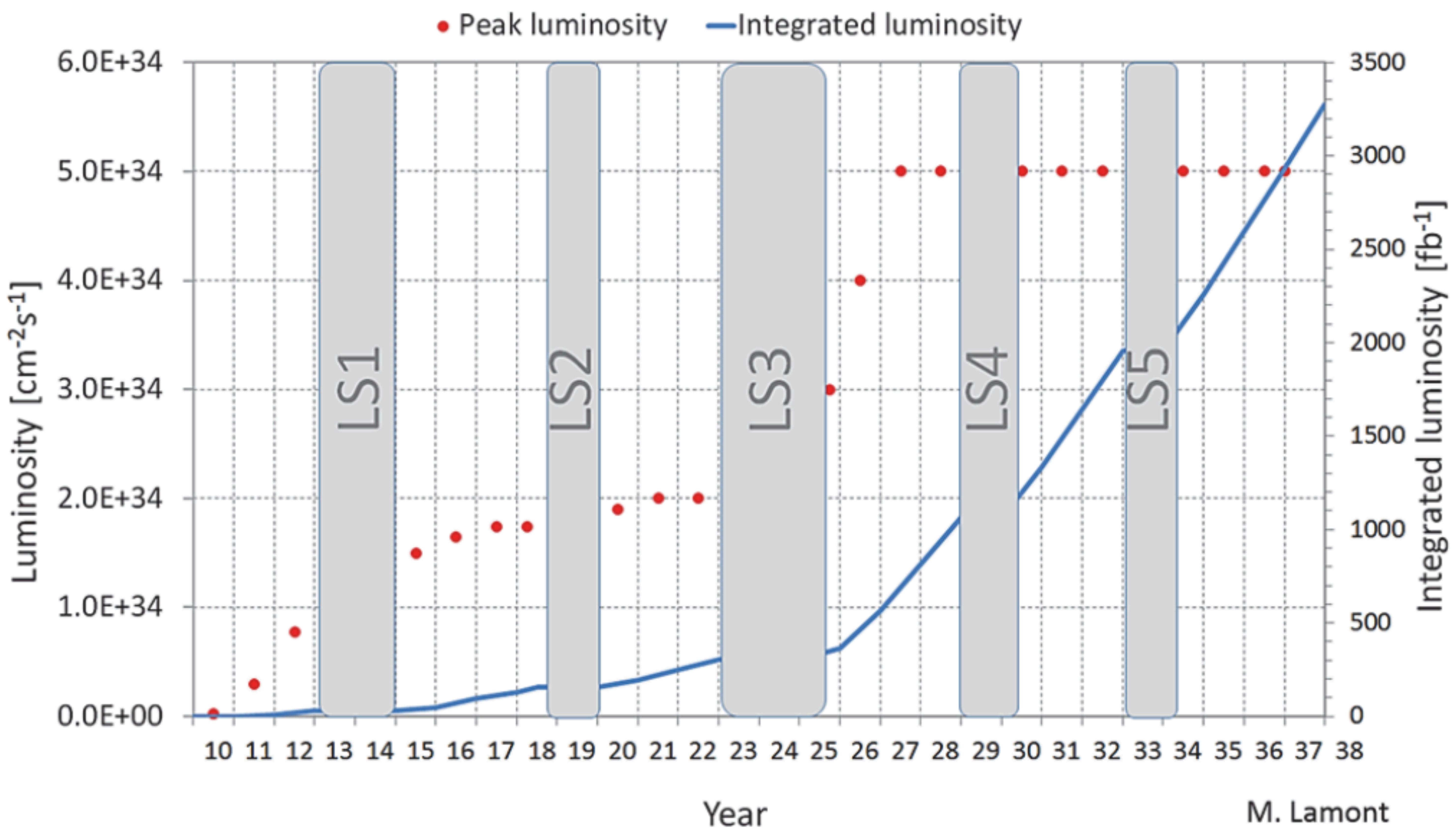
➡ Precision is central aspect of LHC physics

The challenge of precision



Large amouts of data from HL-LHC

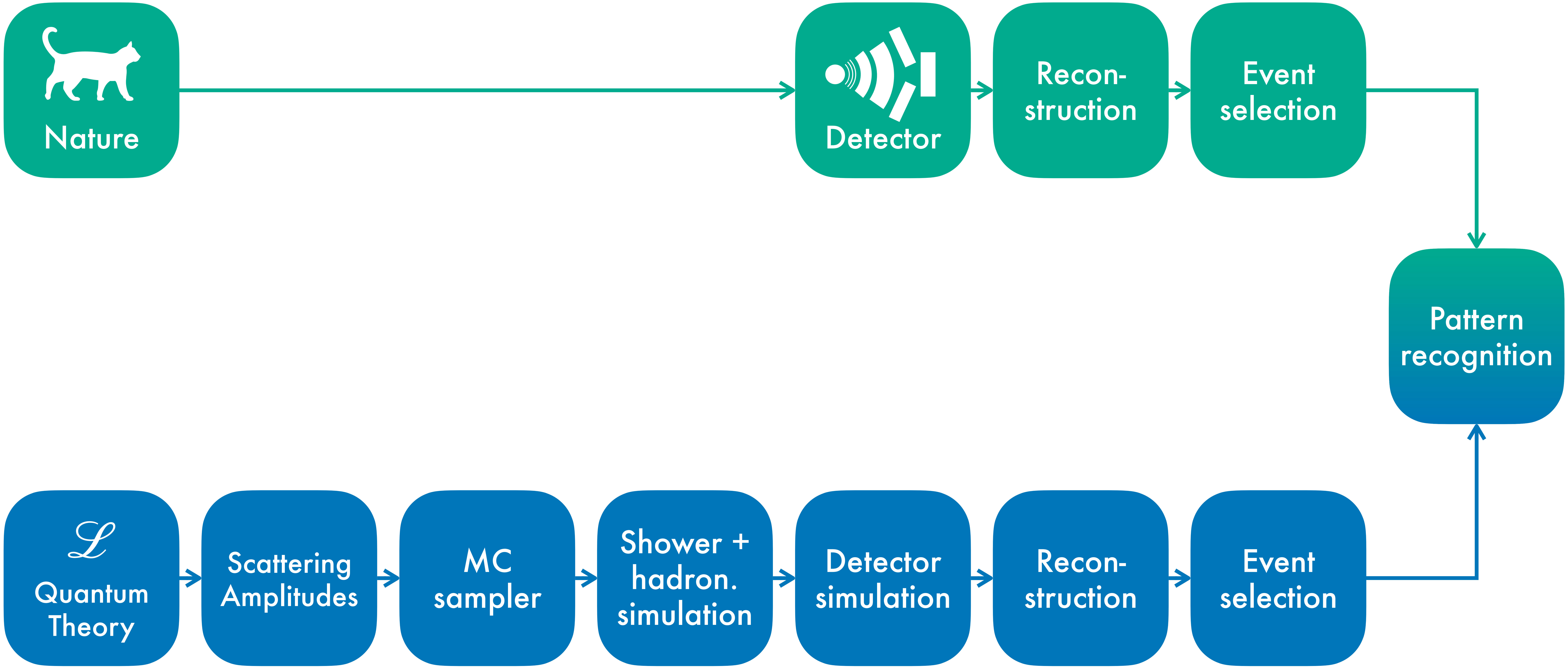
The challenge of precision



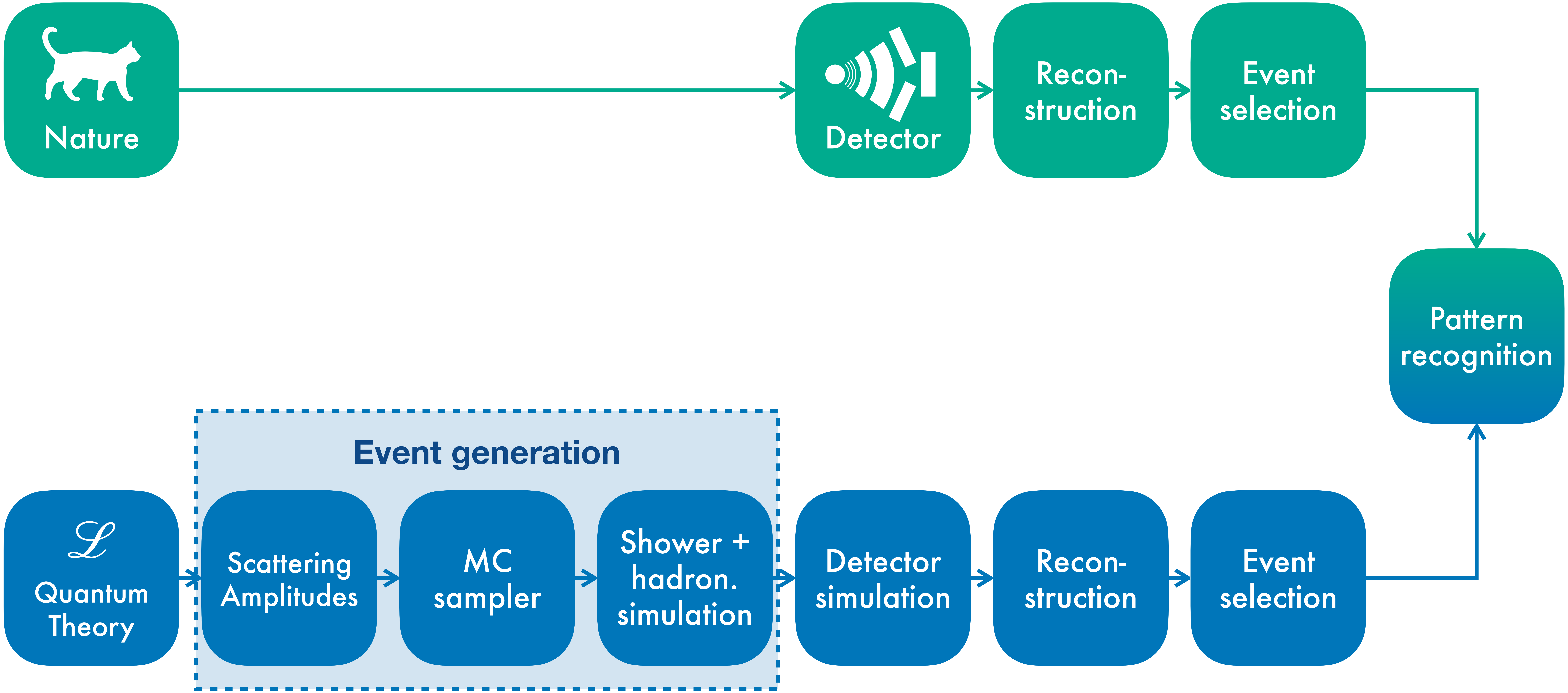
Large amouts of data from HL-LHC

➡ ML tools for processing and evaluating the data

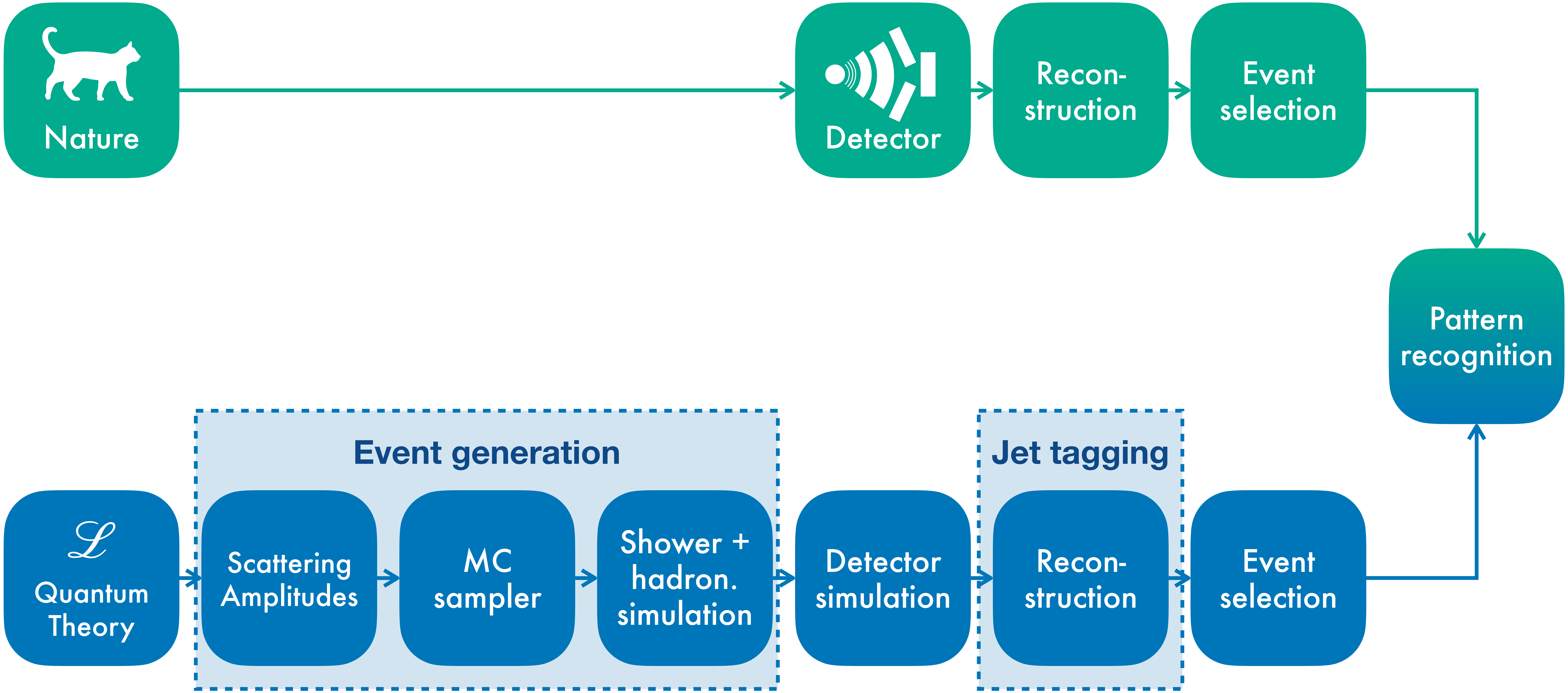
LHC physics and machine learning



LHC physics and machine learning

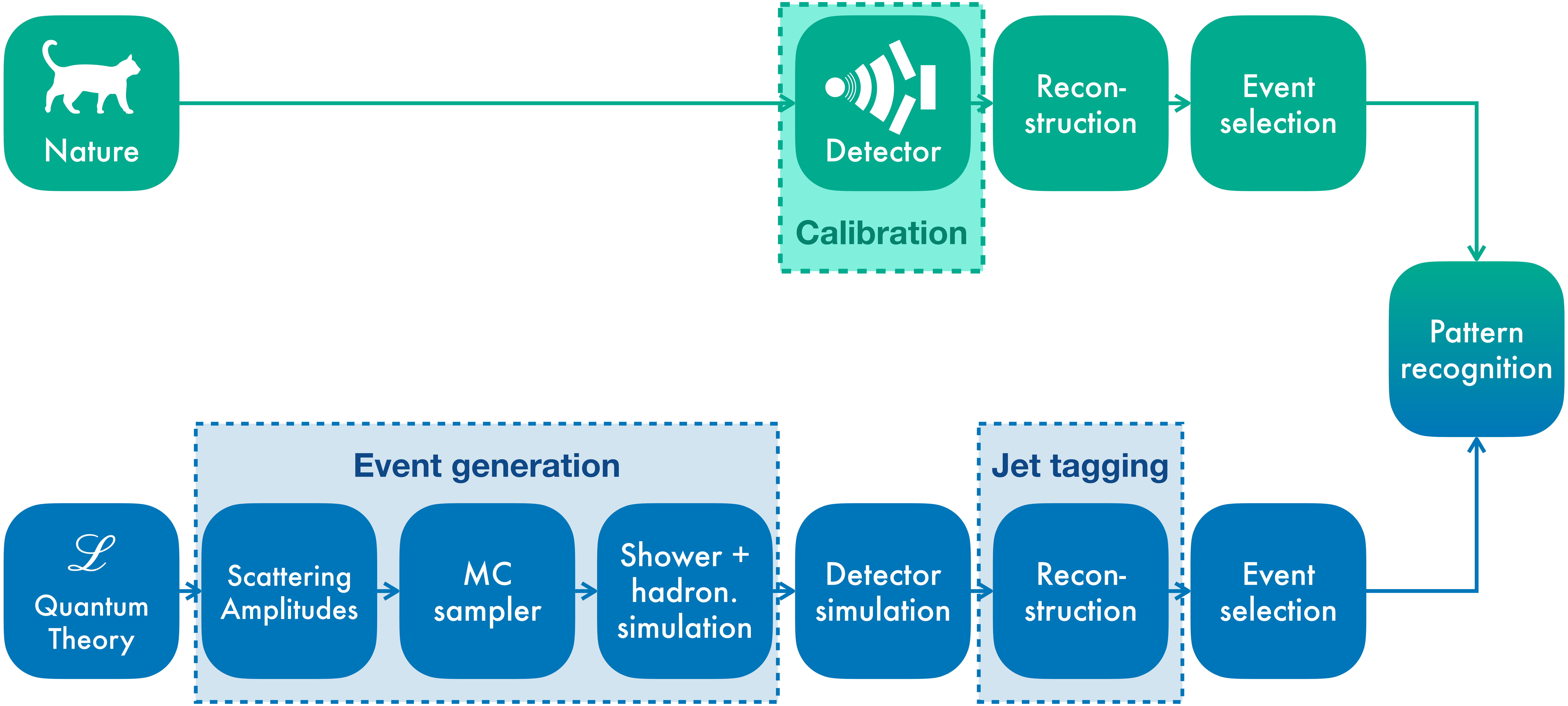


LHC physics and machine learning



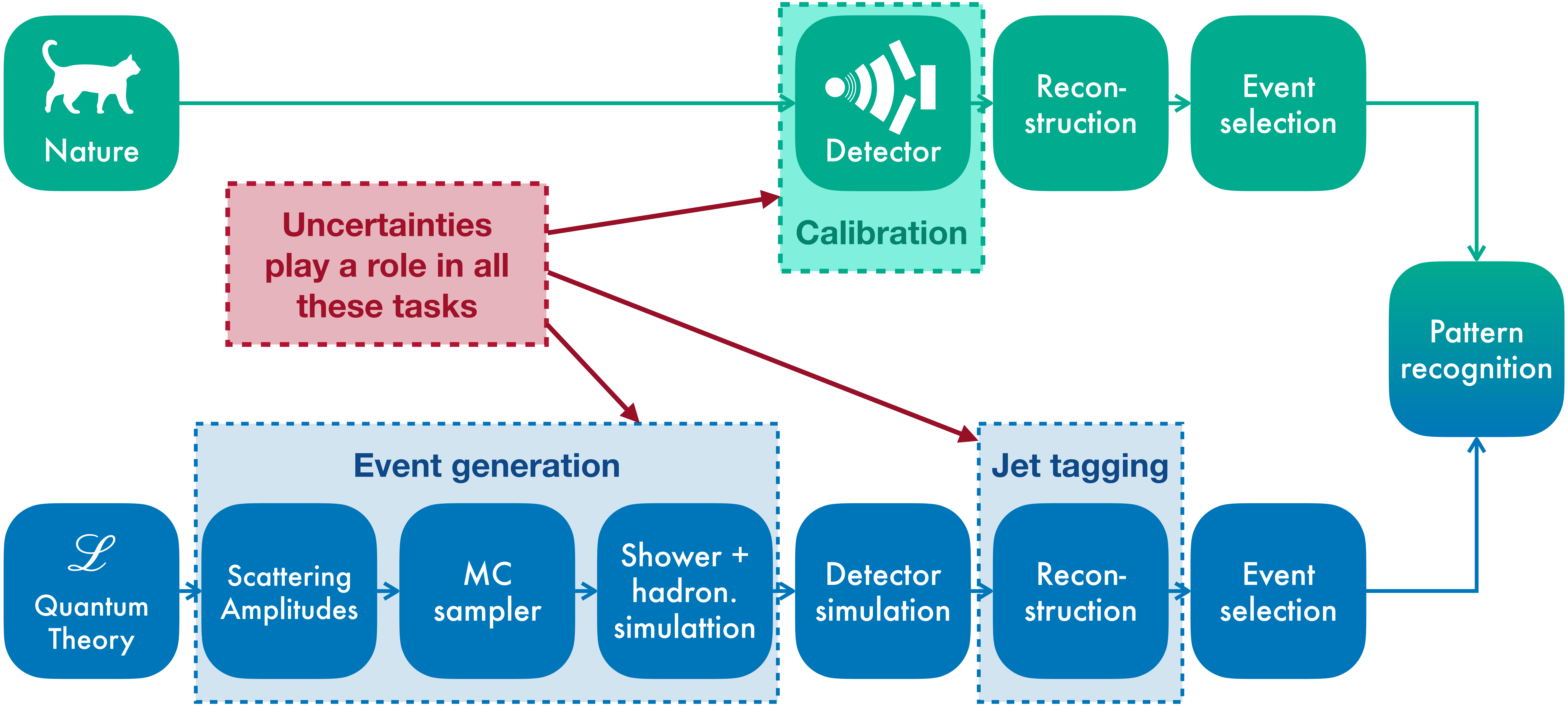
Credits: R. Winterhalder

LHC physics and machine learning



Credits: R. Winterhalder

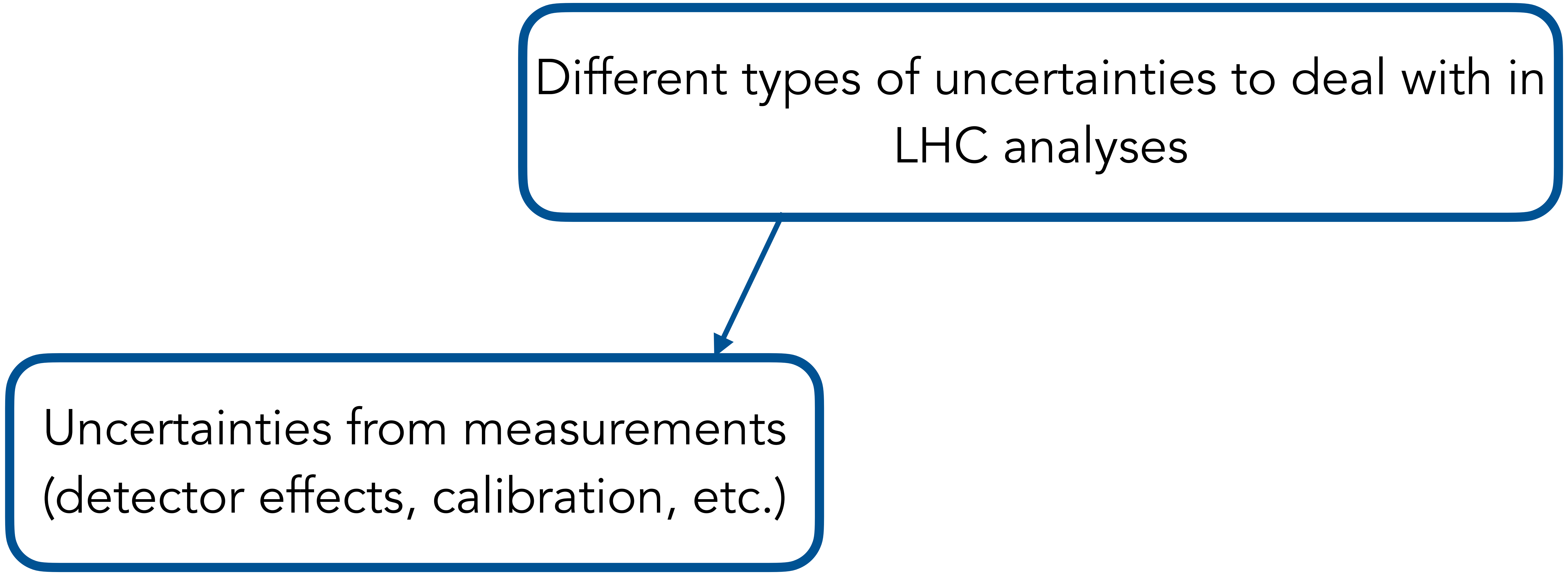
LHC physics and machine learning



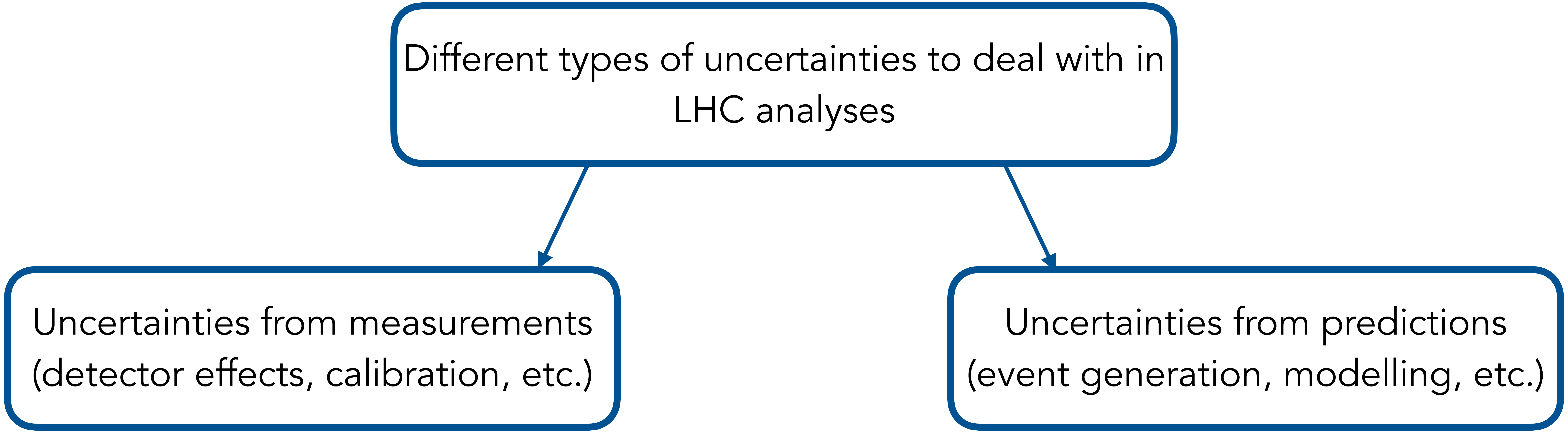
Combining ML with precision measurements

Different types of uncertainties to deal with in LHC analyses

Combining ML with precision measurements



Combining ML with precision measurements



Combining ML with precision measurements

Different types of uncertainties to deal with in LHC analyses

Uncertainties from measurements (detector effects, calibration, etc.)

Uncertainties from predictions (event generation, modelling, etc.)

➡How do ML networks handle these?

➡Intrinsic ML uncertainties?

Outline

Part I: **Different networks and architectures**

Part II: Systematic uncertainties

Part III: Statistical uncertainties

Motivation

- Fit set of Amplitudes $A(x)$, with training data: $\{x, A(x)\}$

Motivation

- Fit set of Amplitudes $A(x)$, with training data: $\{x, A(x)\}$
- Prediction in regression:

Motivation

- Fit set of Amplitudes $A(x)$, with training data: $\{x, A(x)\}$
- Prediction in regression:

prior

$$A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A \,|\, x) = \int \mathrm{d}\theta\, q(\theta)\, \overline{A}(x, \theta) \quad \text{with} \quad p(A \,|\, x) = \int \mathrm{d}\theta\, p(A \,|\, \theta, x)\, p(\theta \,|\, x)$$

network output

Motivation

- Fit set of Amplitudes $A(x)$, with training data: $\{x, A(x)\}$
- Prediction in regression:

prior

$$A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A\,|\,x) = \int \mathrm{d}\theta\, q(\theta)\, \overline{A}(x, \theta) \quad \text{with} \quad p(A\,|\,x) = \int \mathrm{d}\theta\, p(A\,|\,\theta, x)\, p(\theta\,|\,x)$$

network output

$$\sigma^2_{\text{tot}}(x) \equiv \langle (A - \langle A \rangle)^2 \rangle = \int \mathrm{d}A\, \left(A - \langle A \rangle\right)^2 p(A\,|\,x) = \int \mathrm{d}\theta\, q(\theta) \left(\overline{A^2}(x, \theta) - \overline{A}(x, \theta)^2 \right) + \int \mathrm{d}\theta\, q(\theta) \left(\overline{A}(x, \theta) - \langle A \rangle \right)^2$$

Motivation

- Fit set of Amplitudes $A(x)$, with training data: $\{x, A(x)\}$
- Prediction in regression:

prior

$$A(x) \equiv \langle A \rangle = \int dA\, A\, p(A\,|\,x) = \int d\theta\, q(\theta)\, \bar{A}(x,\theta) \quad \text{with} \quad p(A\,|\,x) = \int d\theta\, p(A\,|\,\theta, x)\, p(\theta\,|\,x)$$

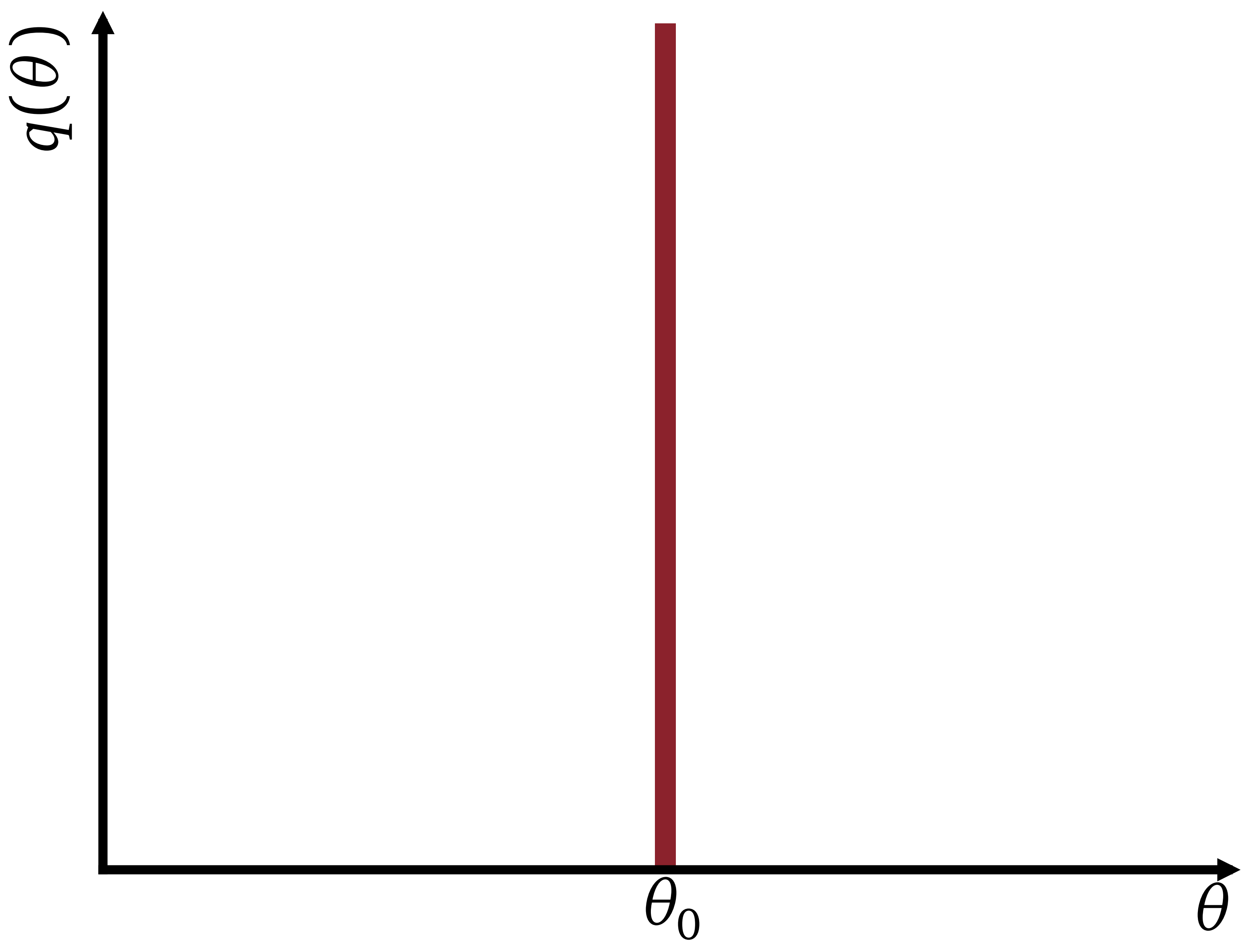
network output

$$\sigma_{\text{tot}}^2(x) \equiv \langle (A - \langle A \rangle)^2 \rangle = \int dA\, \left(A - \langle A \rangle\right)^2 p(A\,|\,x) = \int d\theta\, q(\theta) \left(\overline{A^2}(x,\theta) - \bar{A}(x,\theta)^2 \right) + \int d\theta\, q(\theta) \left(\bar{A}(x,\theta) - \langle A \rangle \right)^2$$

Gaussian uncertainty in heteroscedastic loss: $\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma(x_i)^2} + \log \sigma(x_i) + \ldots$

Bayesian neural networks (BNNs)

Neural network

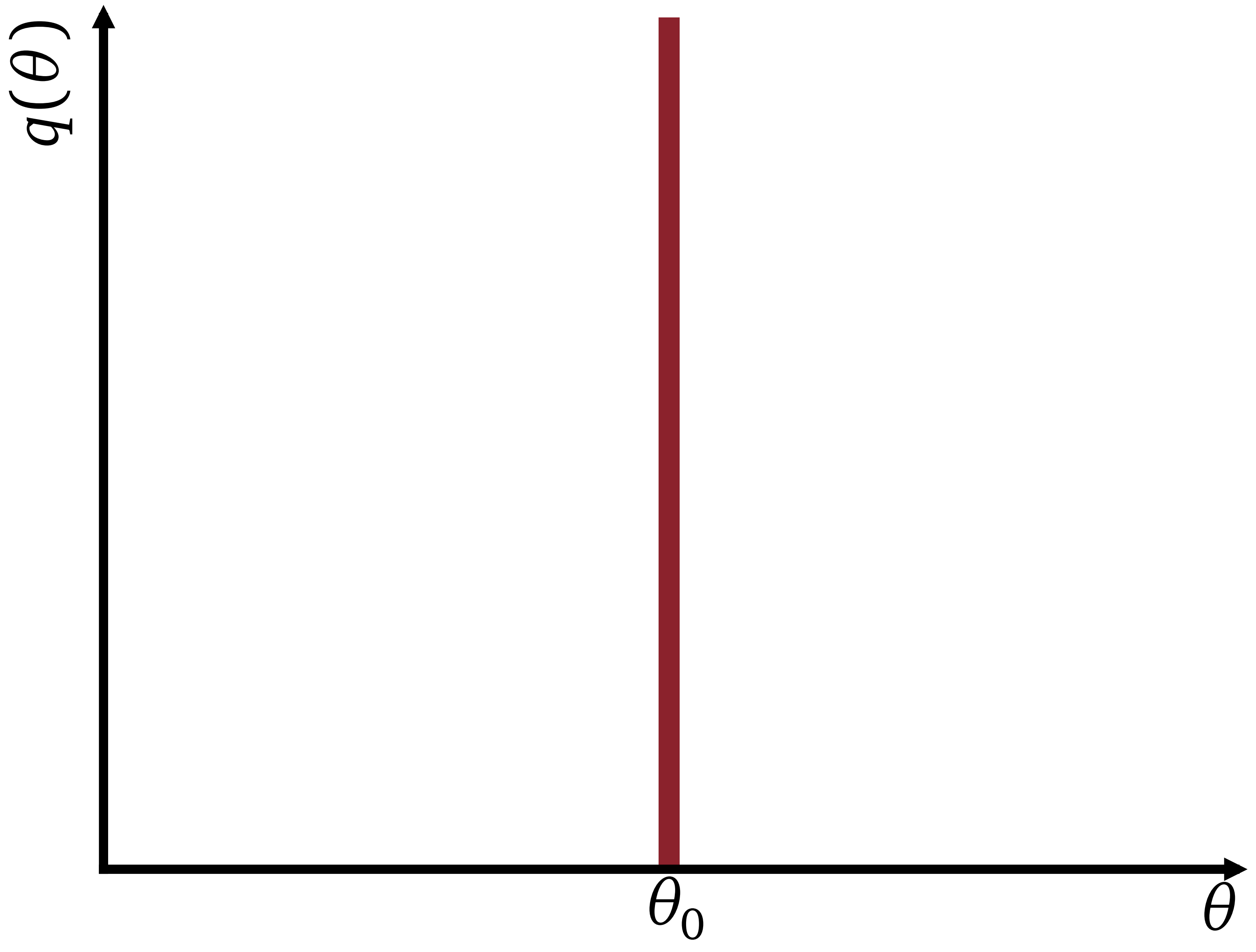


Network weights are deterministic

$$\theta = \theta_0$$

Bayesian neural networks (BNNs)

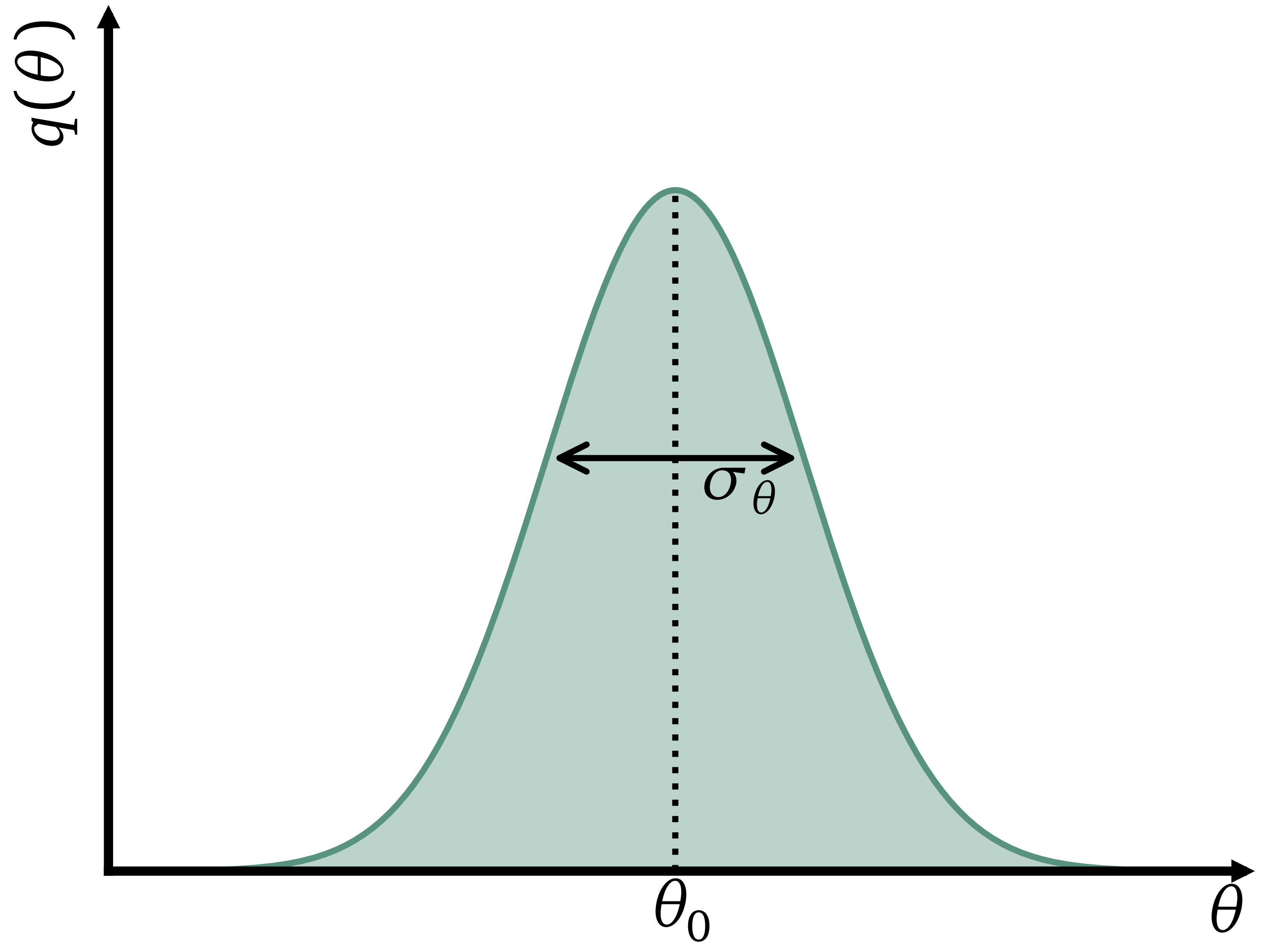
Neural network



Network weights are deterministic

$$\theta = \theta_0$$

Bayesian neural network



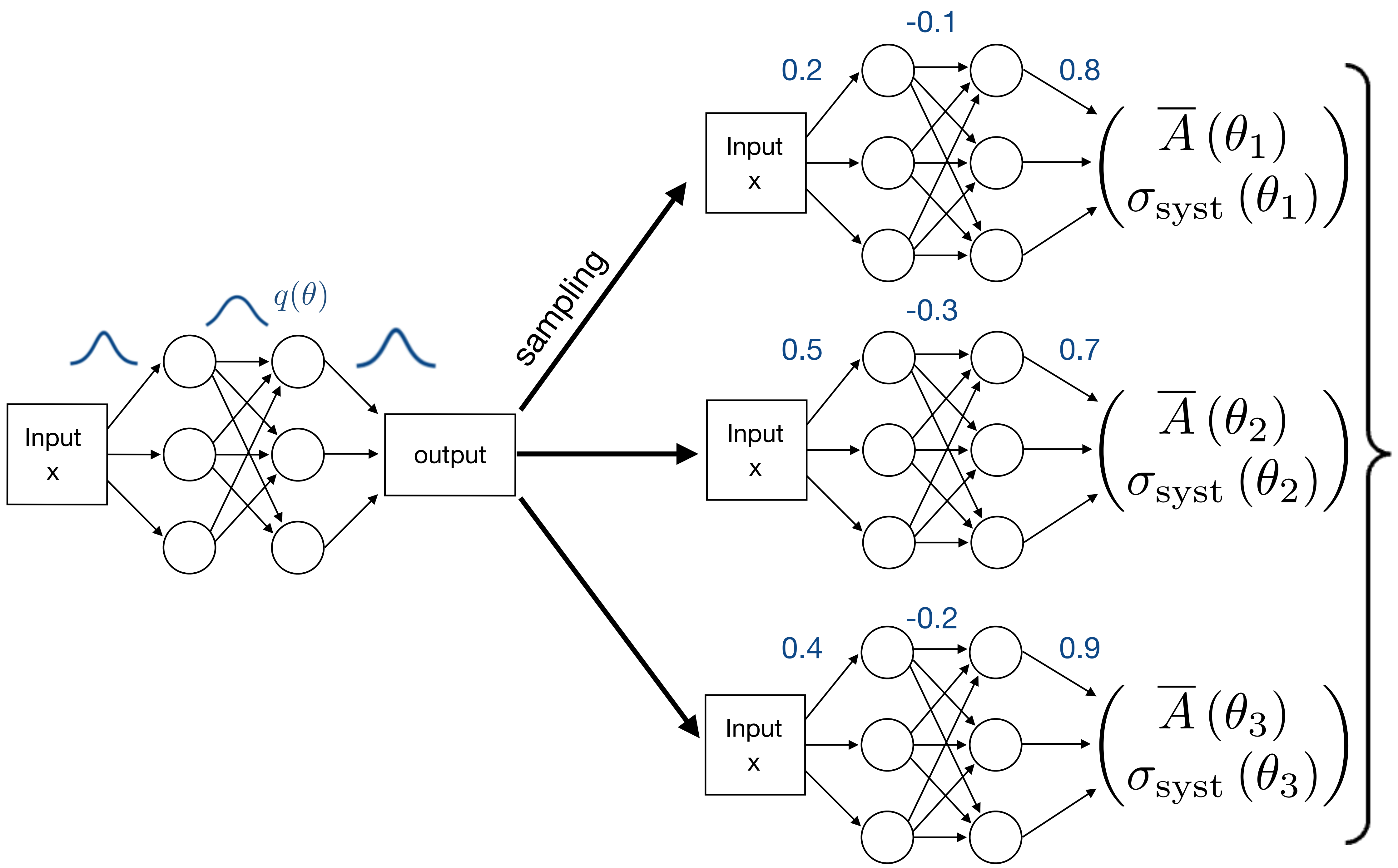
Network weights are drawn from distribution

$$\theta \sim q(\theta)$$

Credits: R. Winterhalder

Bayesian neural networks (BNNs)

BNN **Ensemble of networks** **Output**



$$\langle A \rangle = \frac{1}{N} \sum_i^N \overline{A}(\theta_i)$$

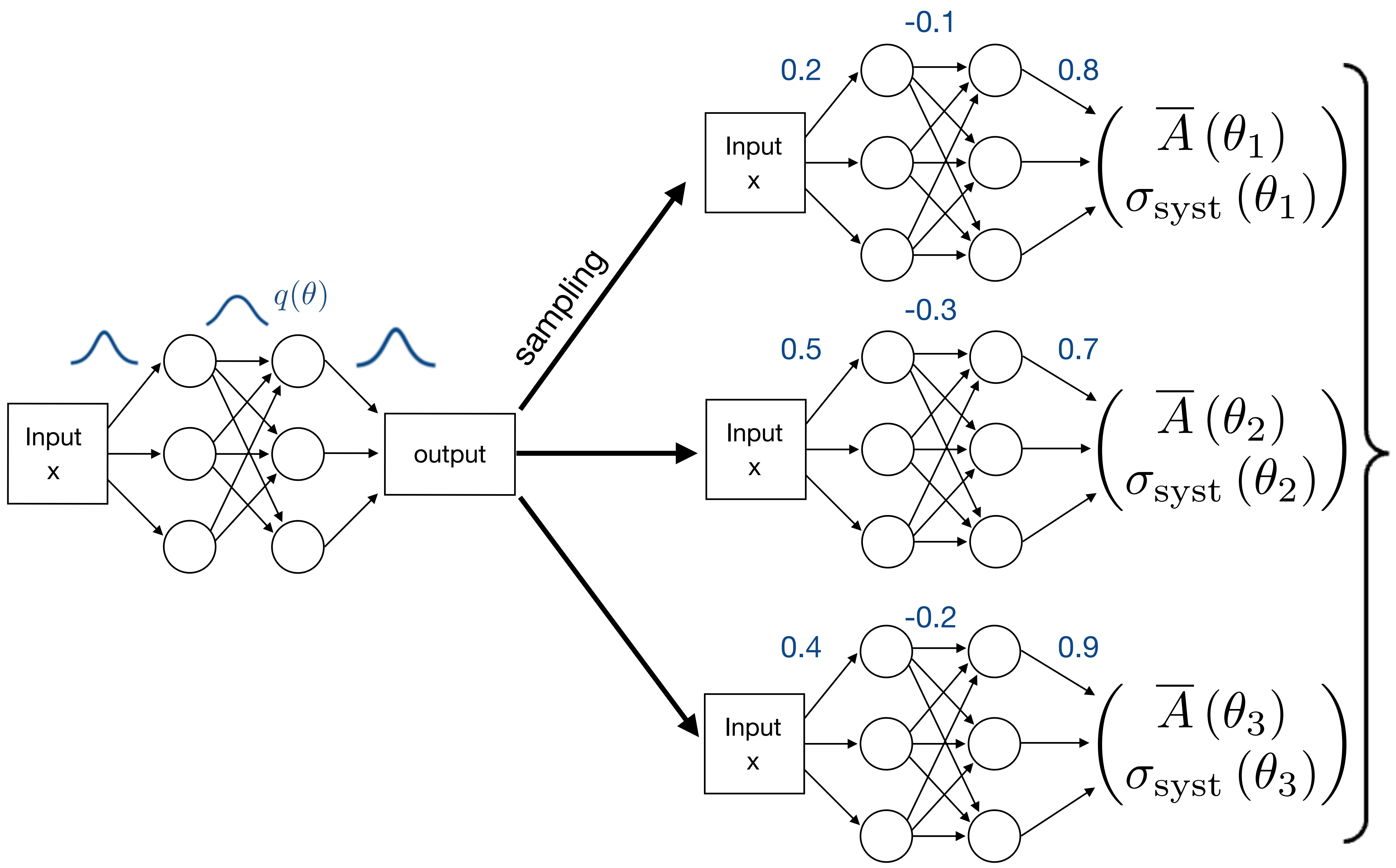
$$\sigma^2_{\text{syst}} = \frac{1}{N} \sum_i^N \sigma^2_{\text{syst}}(\theta_i)$$

$$\sigma^2_{\text{stat}} = \frac{1}{N} \sum_i^N \left(\langle A \rangle - \overline{A}(\theta_i) \right)^2$$

$$\mathcal{L}_{\text{BNN}} = \sum_x \Big[\text{KL}[q(\theta), p(\theta)] - \big\langle \log p(D_{\text{train}}|\theta) \big\rangle_{\theta \sim q(\theta)} \Big]$$

Bayesian neural networks (BNNs)

BNN **Ensemble of networks** **Output**



$$\langle A \rangle = \frac{1}{N} \sum_i^N \overline{A}(\theta_i)$$

$$\sigma^2_{\text{syst}} = \frac{1}{N} \sum_i^N \sigma^2_{\text{syst}}(\theta_i)$$

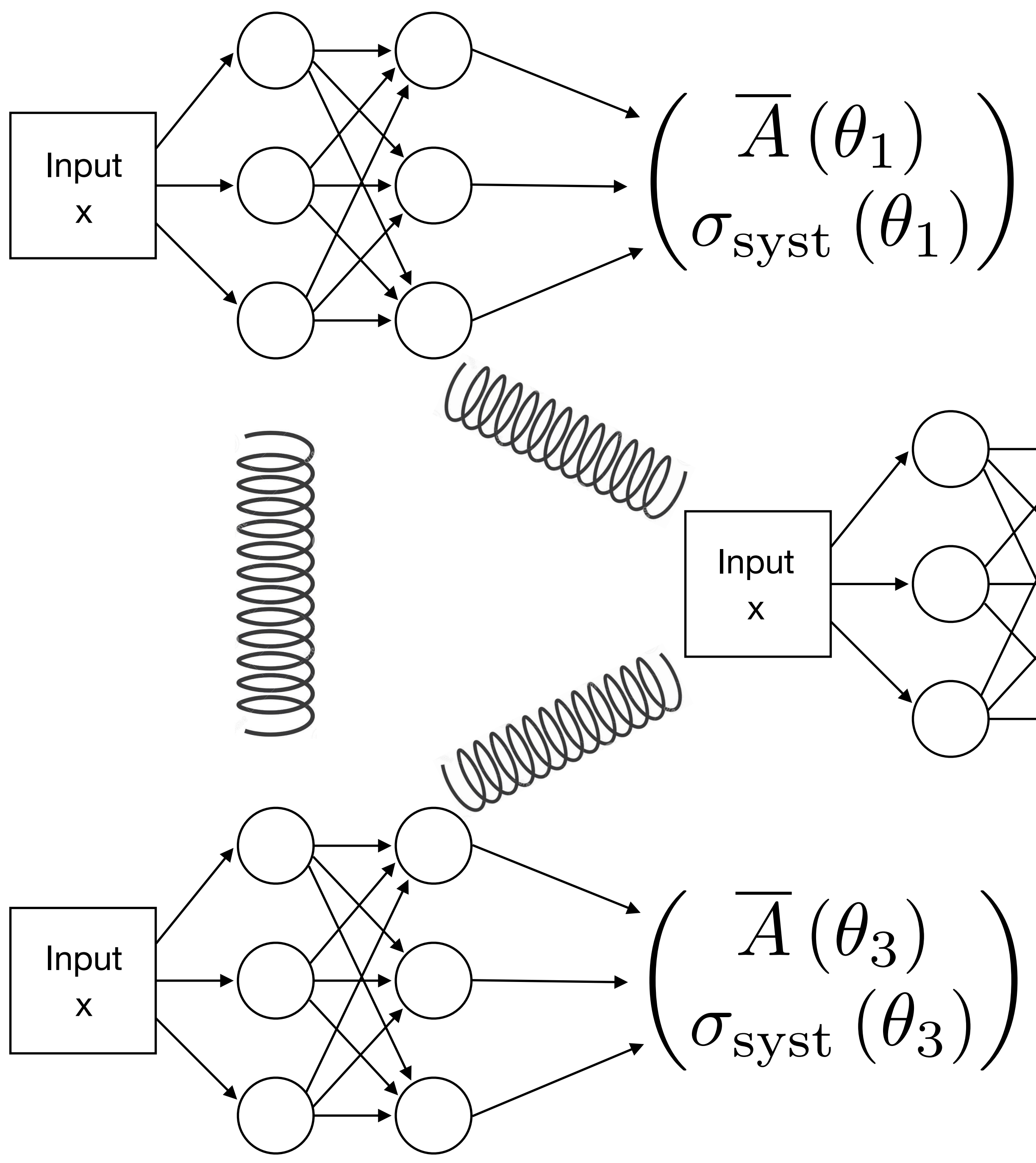
$$\sigma^2_{\text{stat}} = \frac{1}{N} \sum_i^N \left(\langle A \rangle - \overline{A}(\theta_i)\right)^2$$

$$\mathcal{L}_{\text{BNN}} = \sum_x \Big[\text{KL}[q(\theta), p(\theta)] - \big\langle \log p(D_{\text{train}}|\theta) \big\rangle_{\theta \sim q(\theta)} \Big]$$

- Parameters: **Network weights $q(\theta)$**
- $q(\theta)$ params of a Gaussian distribution
- Ensemble: Sample from weight distribution

Repulsive ensembles (REs)

Ensemble of networks **Output**



$$\langle A \rangle = \frac{1}{N} \sum_i^N \overline{A}(\theta_i)$$

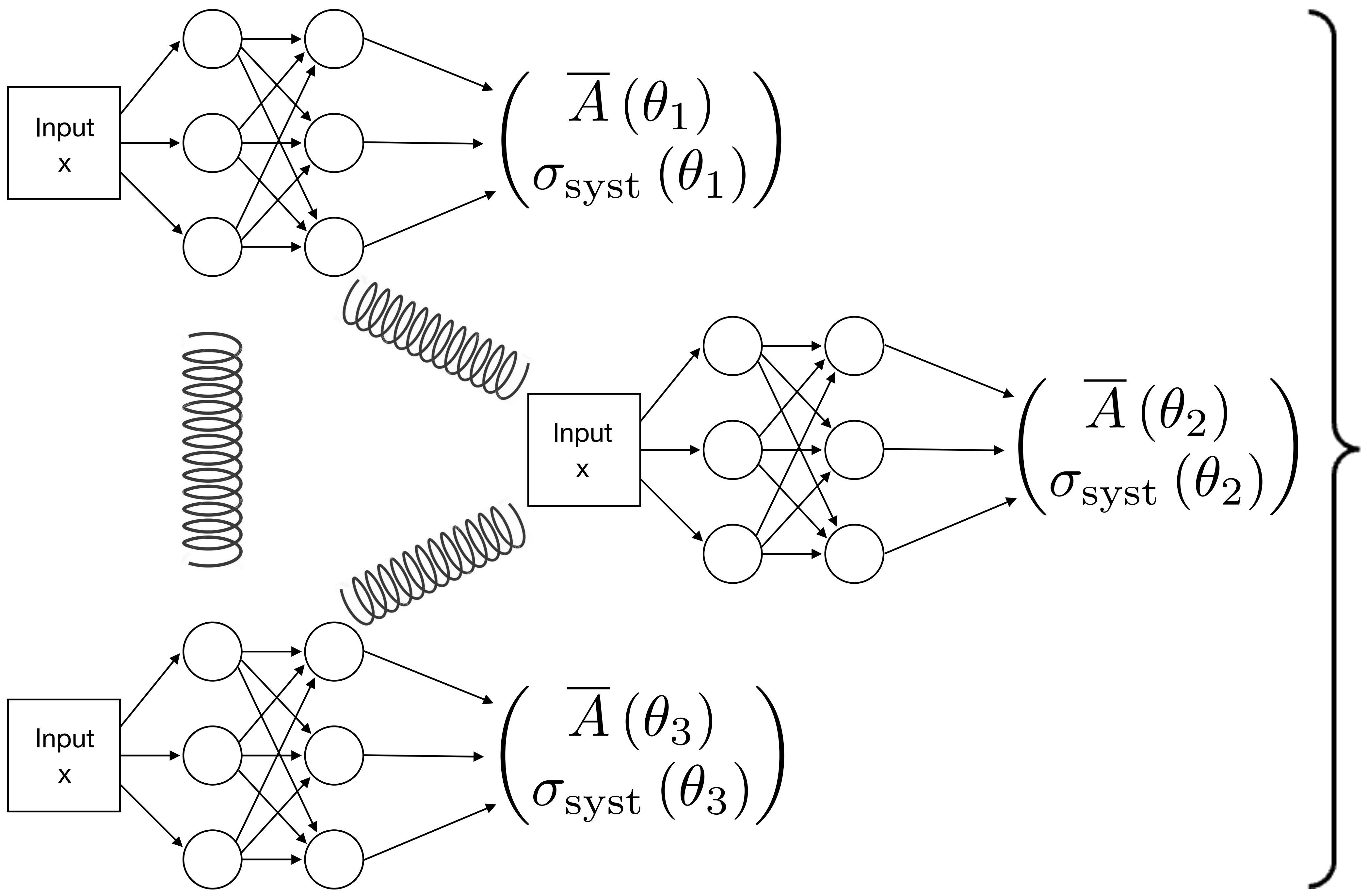
$$\sigma_{\text{syst}}^2 = \frac{1}{N} \sum_i^N \sigma_{\text{syst}}^2(\theta_i)$$

$$\sigma_{\text{stat}}^2 = \frac{1}{N} \sum_i^N \left(\langle A \rangle - \overline{A}(\theta_i)\right)^2$$

$$\mathcal{L}_{\text{RE}} = \sum_{i=1}^{n} \left[-\frac{1}{B} \sum_{b=1}^{B} \log p(x_b|\theta_i) + \frac{\beta}{N} \frac{\sum_{j=1}^{n} k(A_{\theta_i}(x), \overline{A_{\theta_j}(x)})}{\sum_{j=1}^{n} k(\overline{A_{\theta_i}(x)}, \overline{A_{\theta_j}(x)})} + \frac{\theta_i^2}{2N\sigma^2} \right]$$

Repulsive ensembles (REs)

Ensemble of networks | **Output**



$$\langle A \rangle = \frac{1}{N} \sum_i^N \overline{A}(\theta_i)$$

$$\sigma^2_{\text{syst}} = \frac{1}{N} \sum_i^N \sigma^2_{\text{syst}}(\theta_i)$$

$$\sigma^2_{\text{stat}} = \frac{1}{N} \sum_i^N \left(\langle A \rangle - \overline{A}(\theta_i) \right)^2$$

- Repulsive term: Cover full posterior distribution
- Ensemble members **trained simultaneously**

$$\mathcal{L}_{\text{RE}} = \sum_{i=1}^{n} \left[-\frac{1}{B} \sum_{b=1}^{B} \log p(x_b|\theta_i) + \frac{\beta}{N} \frac{\sum_{j=1}^{n} k(A_{\theta_i}(x), \overline{A_{\theta_j}(x)})}{\sum_{j=1}^{n} k(\overline{A_{\theta_i}(x)}, \overline{A_{\theta_j}(x)})} + \frac{\theta_i^2}{2N\sigma^2} \right]$$

Different network architectures

- MLP: fully connected linear layers followed by non-linearities
- Deep Sets (DS): learns embedding for each particle type
- Deep Sets Invariants (DSI): DS with Lorentz invariance added as input
- L-GATr: fully Lorentz equivariant network architecture [2411.00446]

Learning Amplitudes

- Learning scattering amplitudes $|M|^2$
- One-loop partonic process: $gg \rightarrow \gamma\gamma g$

Learning Amplitudes

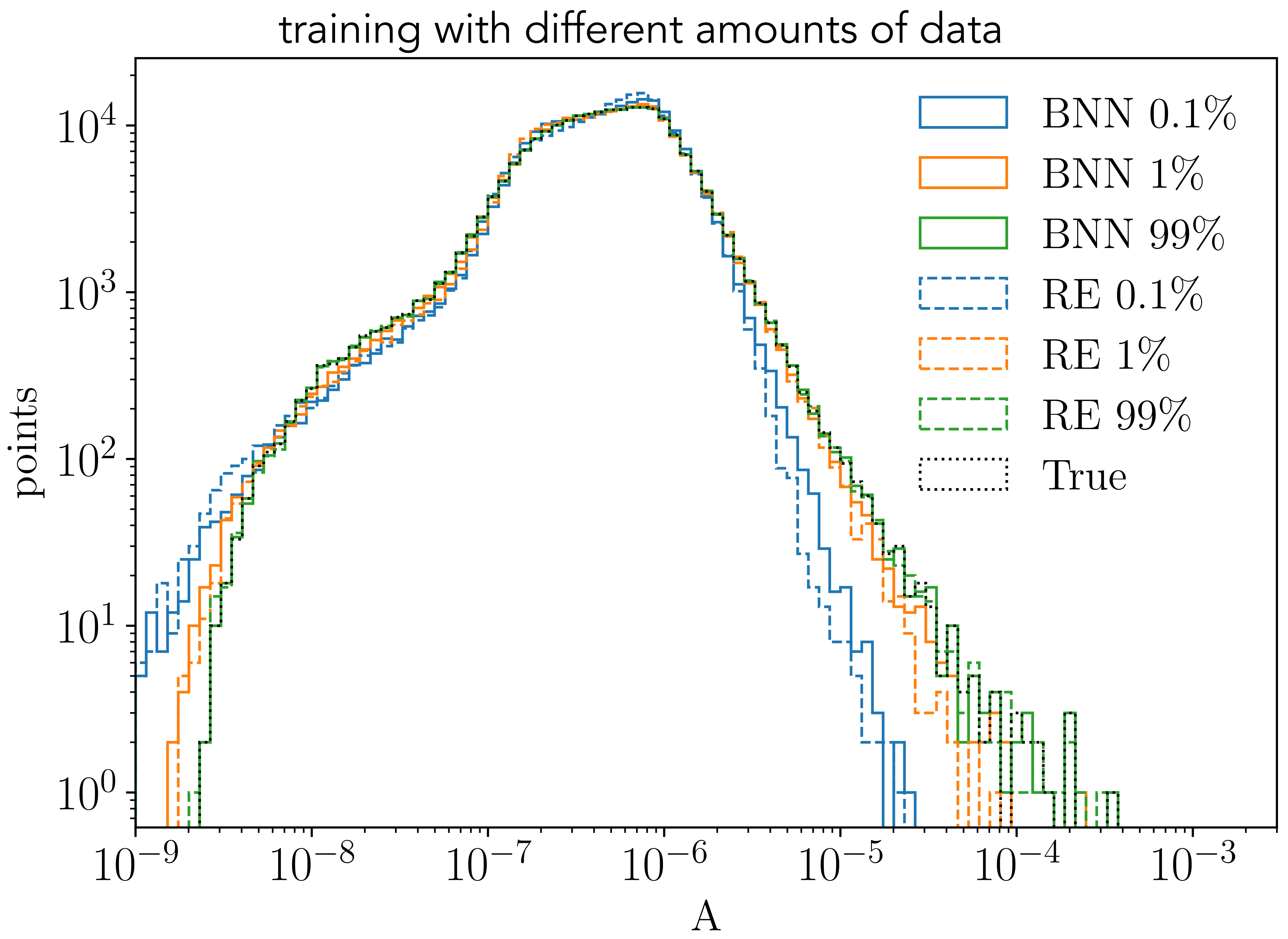
- Learning scattering amplitudes $|M|^2$
- One-loop partonic process: $gg \to \gamma\gamma g$
- Apply basic cuts:

$$p_{T,j} > 20\,\mathrm{GeV}$$
$$p_{T,\gamma} > 40, 30\,\mathrm{GeV}$$
$$|\eta_j| < 5$$
$$|\eta_\gamma| < 2.37$$
$$R_{jj,j\gamma,\gamma\gamma} > 0.4$$

Learning Amplitudes

- Learning scattering amplitudes $|M|^2$
- One-loop partonic process: $gg \to \gamma\gamma g$
- Apply basic cuts:

$p_{T,j} > 20\,\text{GeV}$
$p_{T,\gamma} > 40, 30\,\text{GeV}$
$|\eta_j| < 5$
$|\eta_\gamma| < 2.37$
$R_{jj,j\gamma,\gamma\gamma} > 0.4$

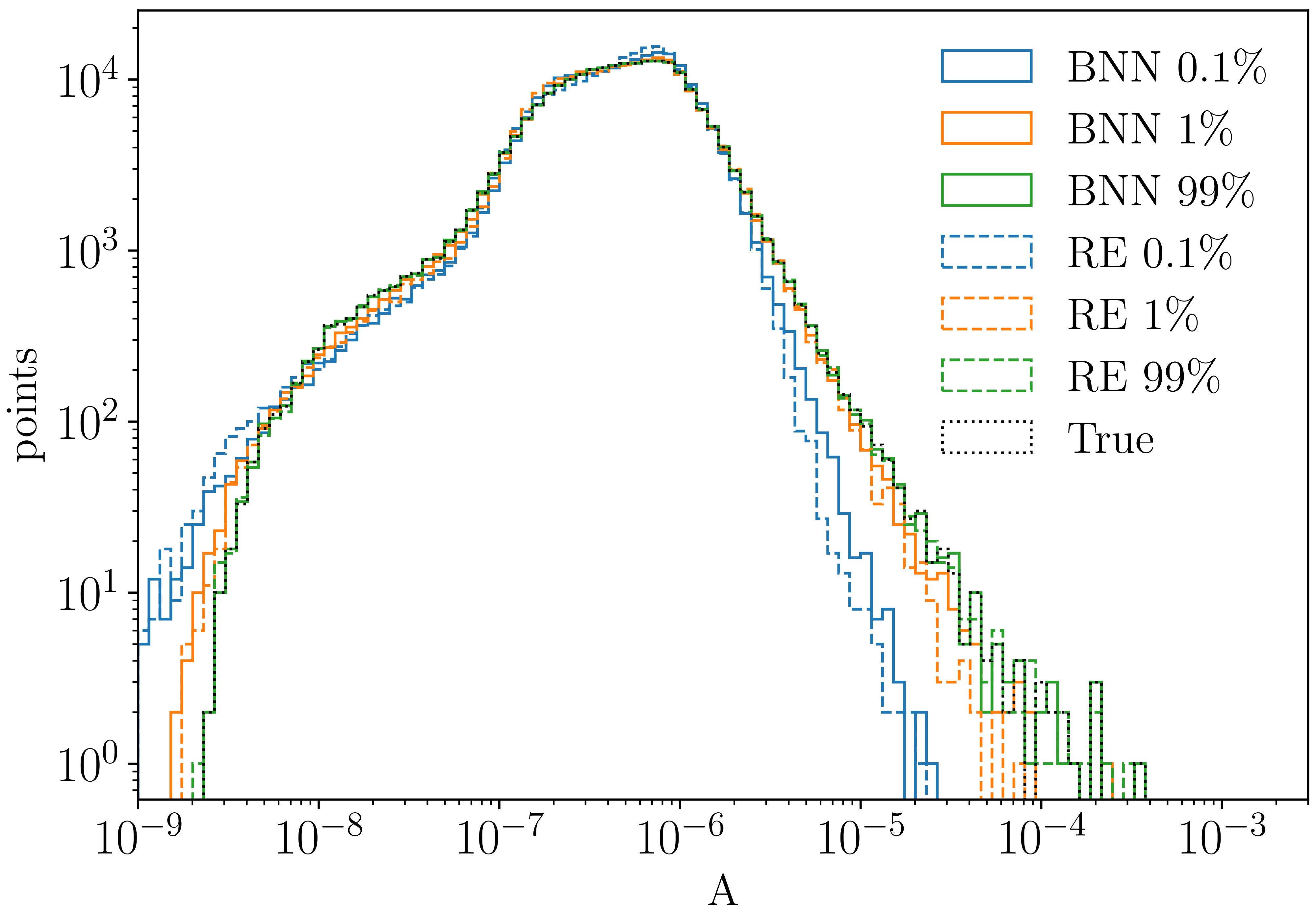


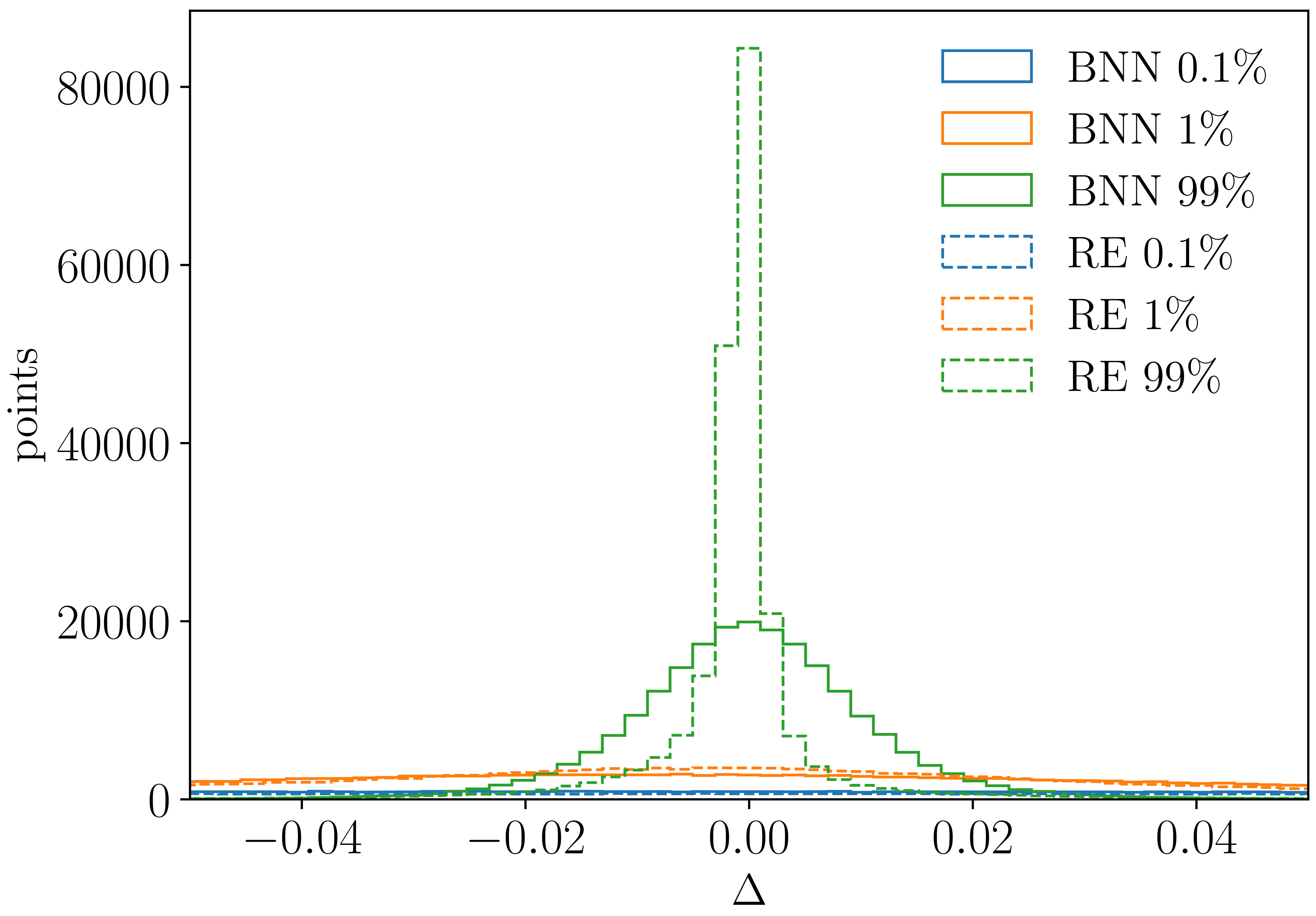
Prediction accuracy

- Full dataset: 1.1M phase space points (unweighted samples)
- Accuracy testing: $\Delta(x) = \frac{A_{\mathrm{NN}}(x) - A_{\mathrm{true}}(x)}{A_{\mathrm{true}}(x)}$

Prediction accuracy

- Full dataset: 1.1M phase space points (unweighted samples)
- Accuracy testing: $\Delta(x) = \dfrac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{A_{\text{true}}(x)}$





Different types of uncertainties

Two uncertainty types:

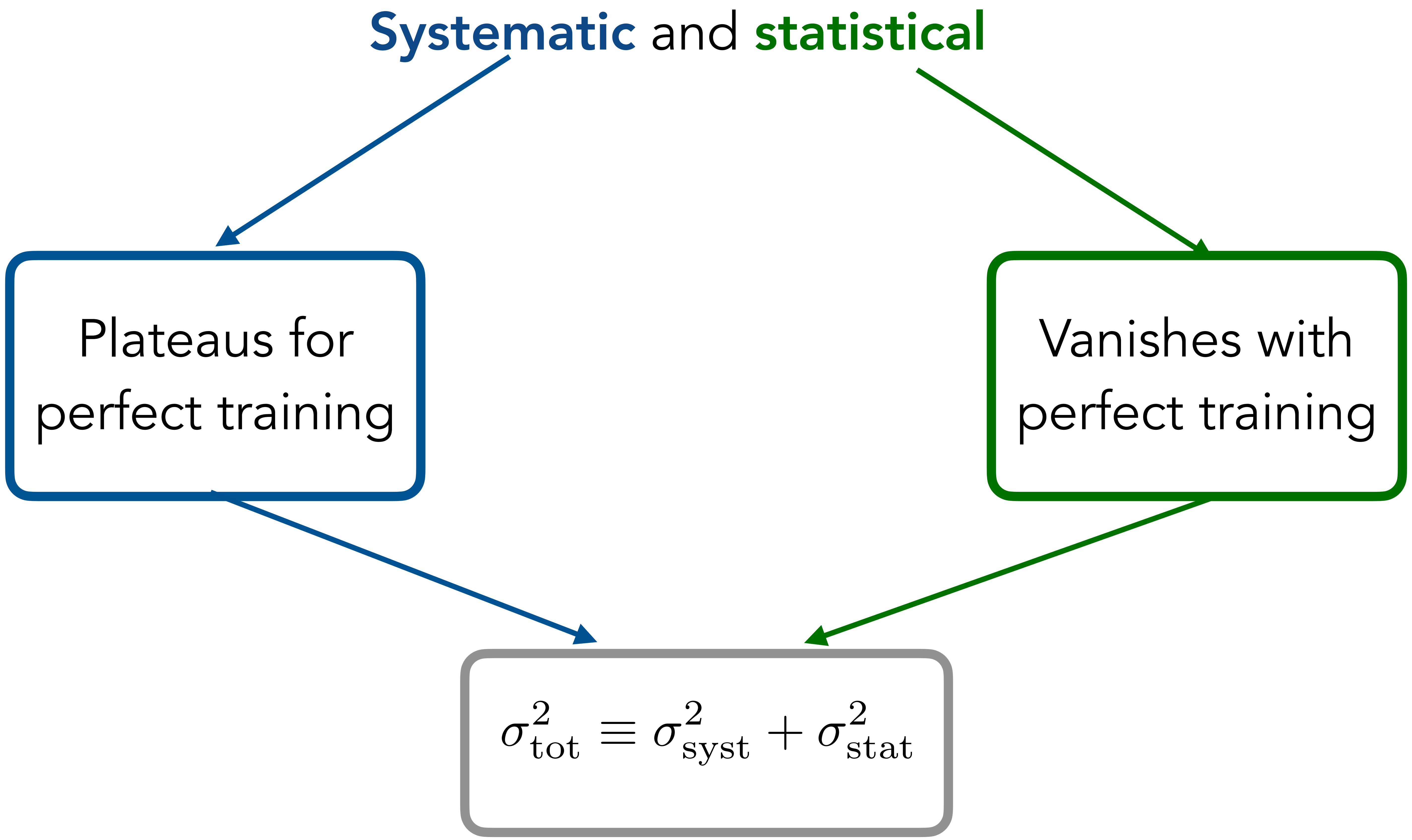
Systematic and **statistical**

Plateaus for perfect training

Vanishes with perfect training

Different types of uncertainties

Two uncertainty types:



Outline

Part I: Different networks and architectures

Part II: **Systematic uncertainties**

Part III: Statistical uncertainties

Adding Gaussian noise

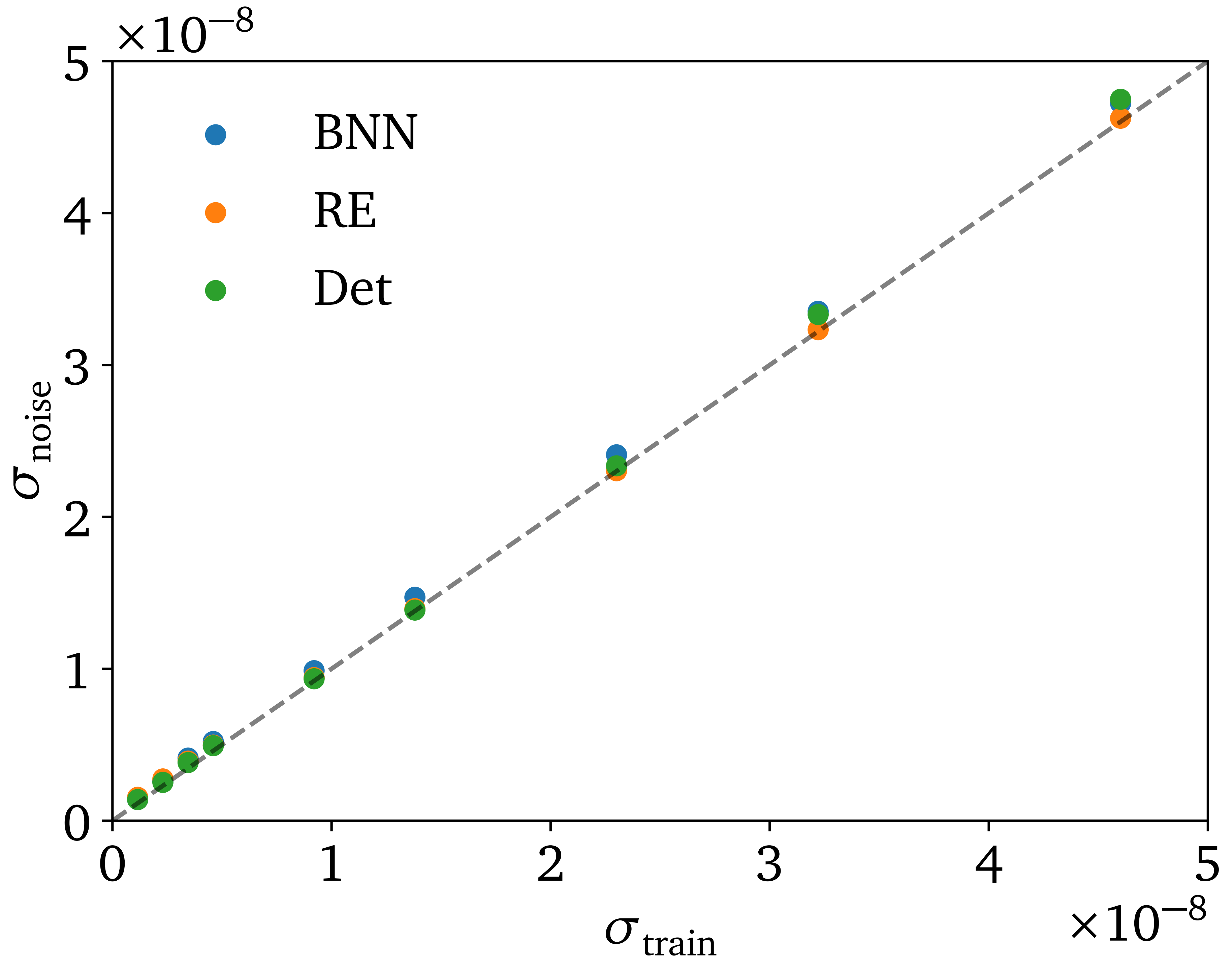
$$\sigma^2_{\text{tot}} = \sigma^2_{\text{syst},0} + \sigma^2_{\text{noise}} + \sigma^2_{\text{stat}}$$

$$\sigma_{\text{train}} = f_{\text{smear}}\, A_{\text{true}}$$

Adding Gaussian noise

$$\sigma^2_{\text{tot}} = \sigma^2_{\text{syst},0} + \sigma^2_{\text{noise}} + \sigma^2_{\text{stat}}$$

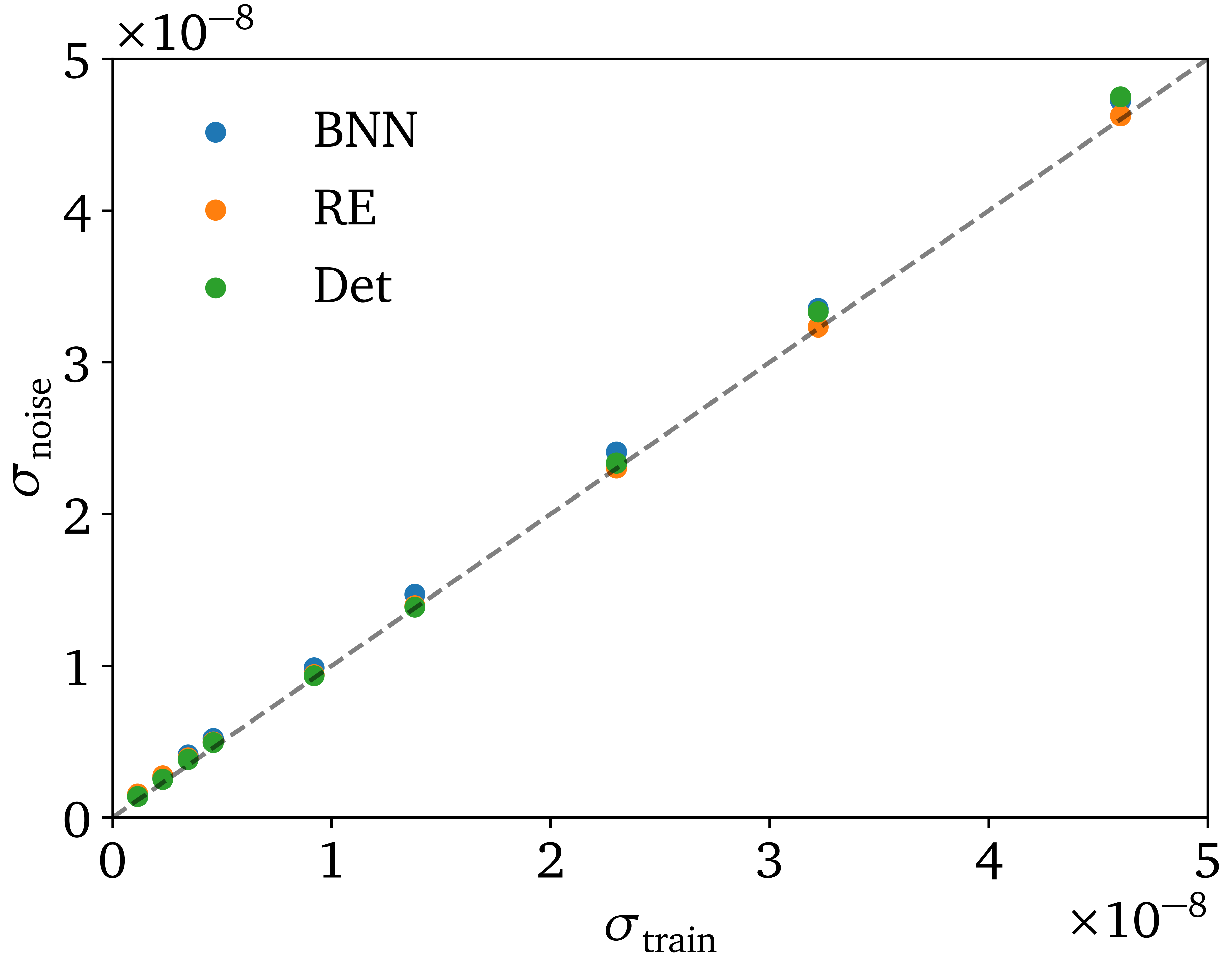
$$\sigma_{\text{train}} = f_{\text{smear}}\, A_{\text{true}}$$

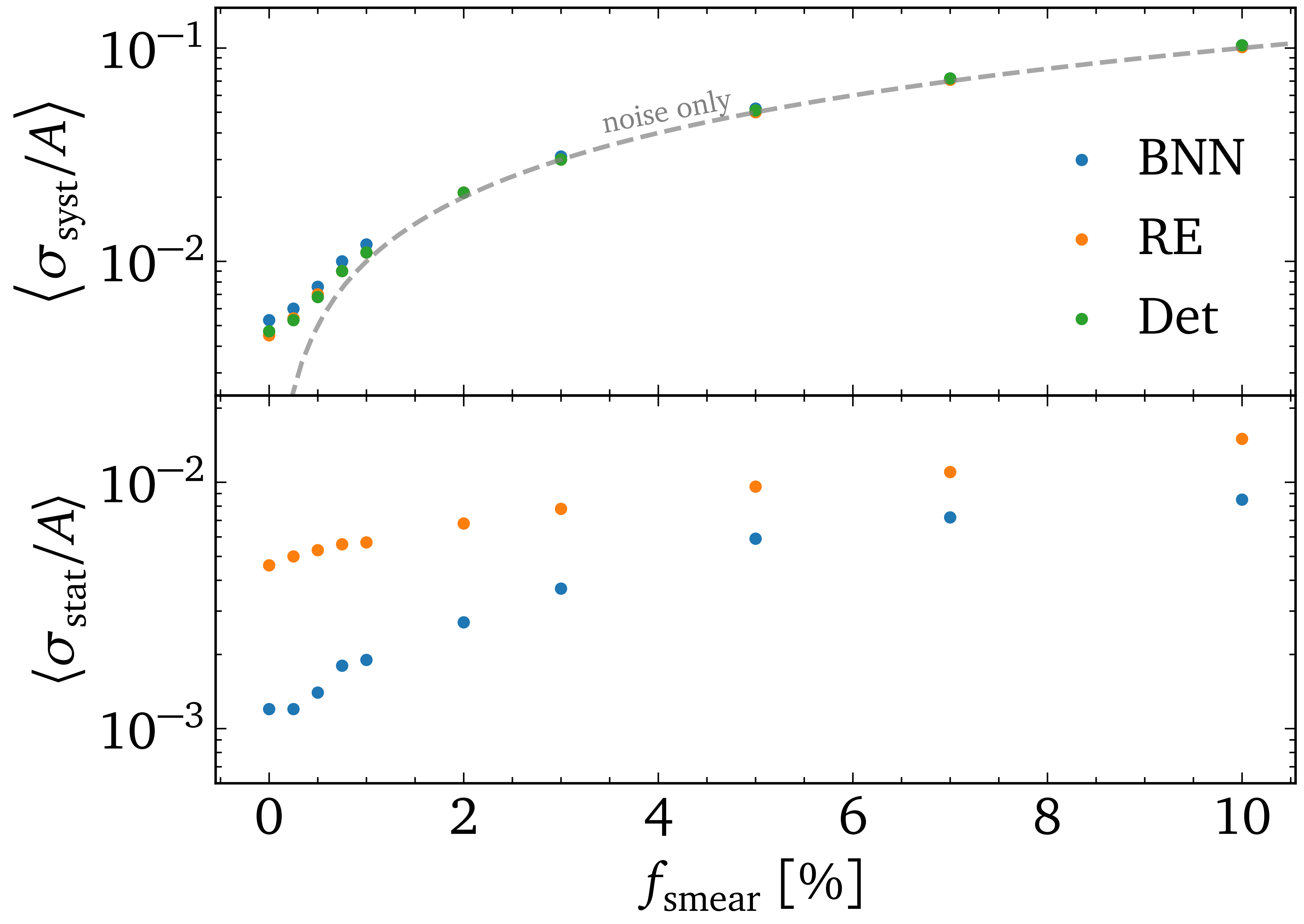


Adding Gaussian noise

$$\sigma_{\text{tot}}^2 = \sigma_{\text{syst},0}^2 + \sigma_{\text{noise}}^2 + \sigma_{\text{stat}}^2$$

$$\sigma_{\text{train}} = f_{\text{smear}} A_{\text{true}}$$

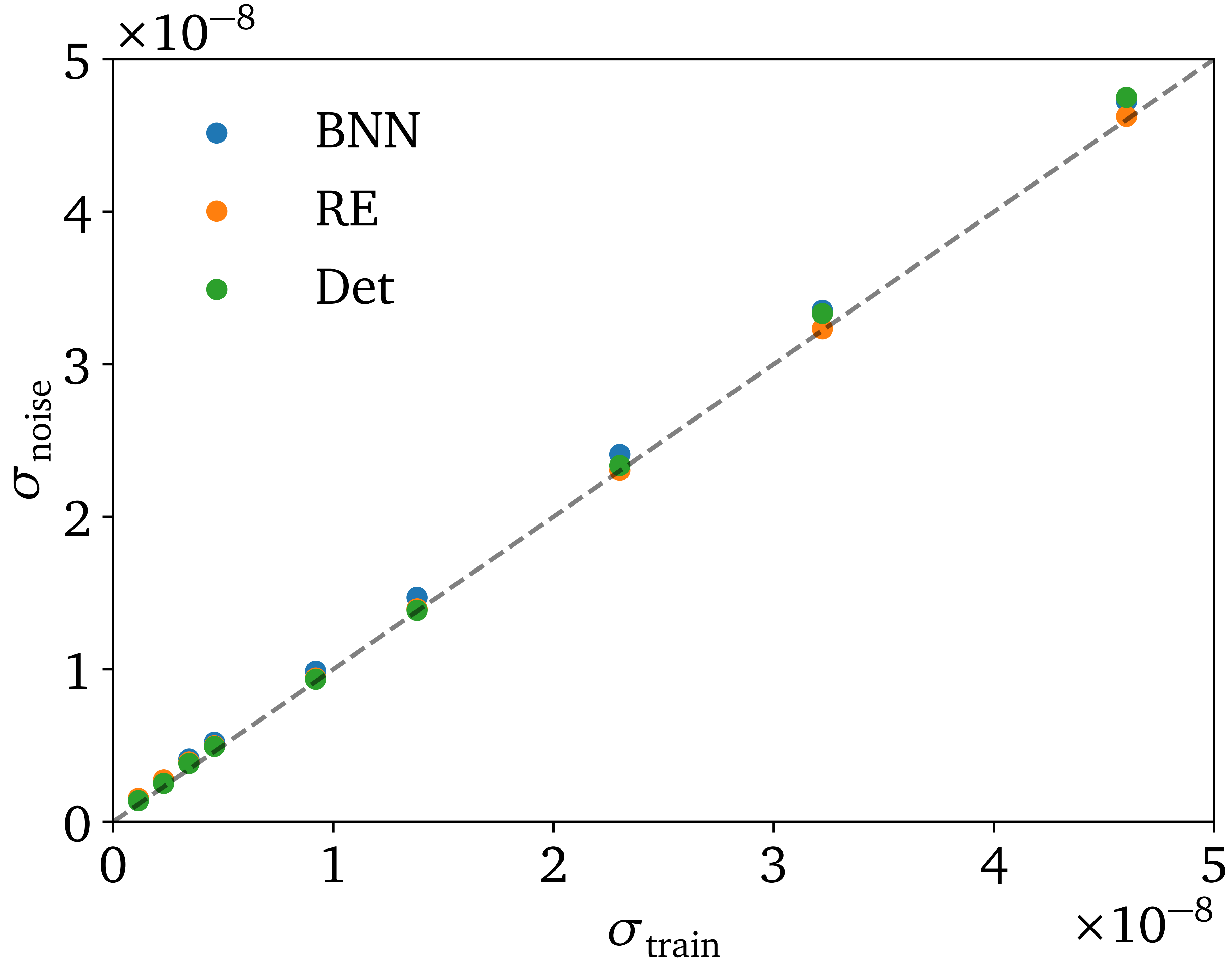


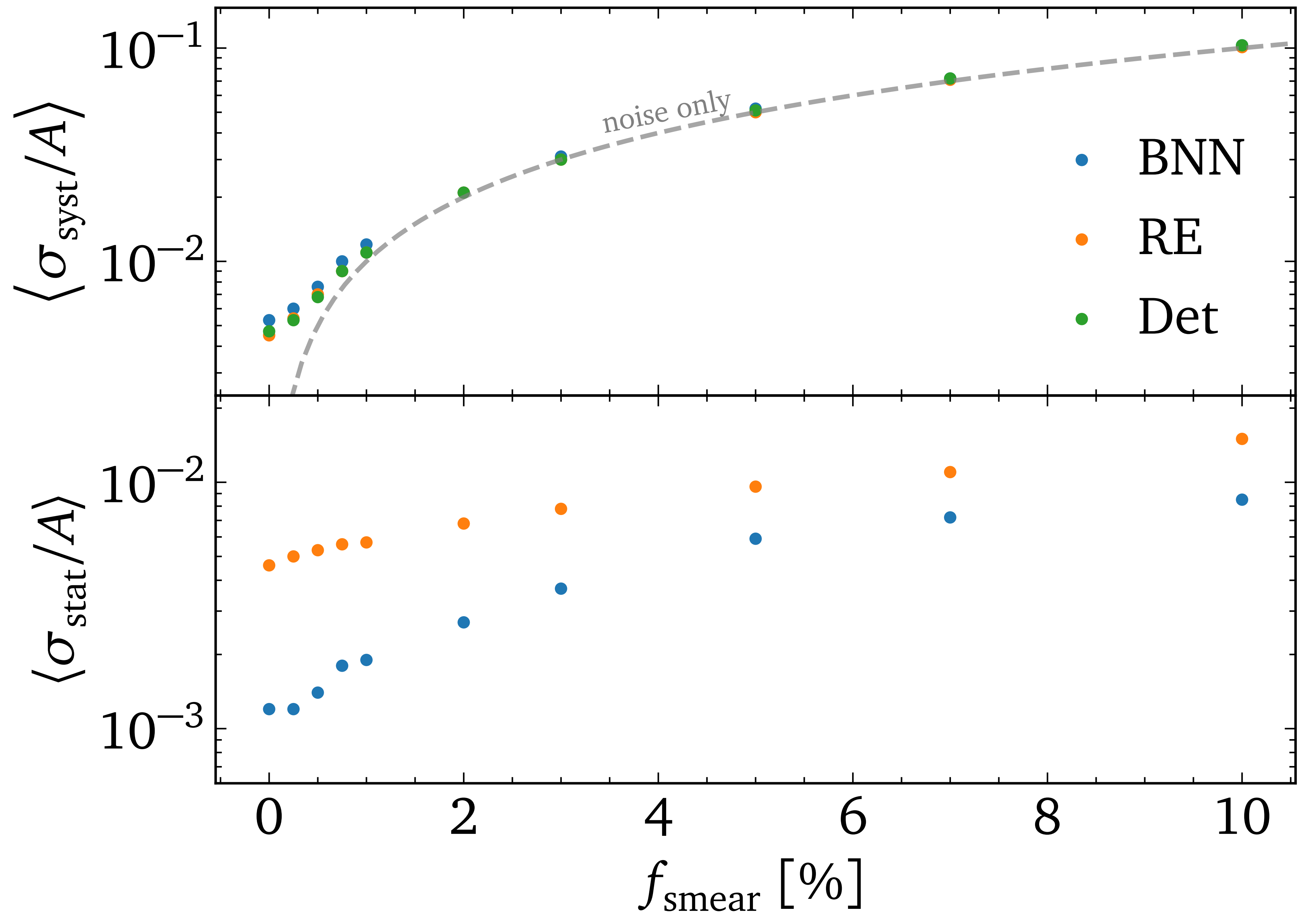


Adding Gaussian noise

$$\sigma_{\text{tot}}^2 = \sigma_{\text{syst},0}^2 + \sigma_{\text{noise}}^2 + \sigma_{\text{stat}}^2$$

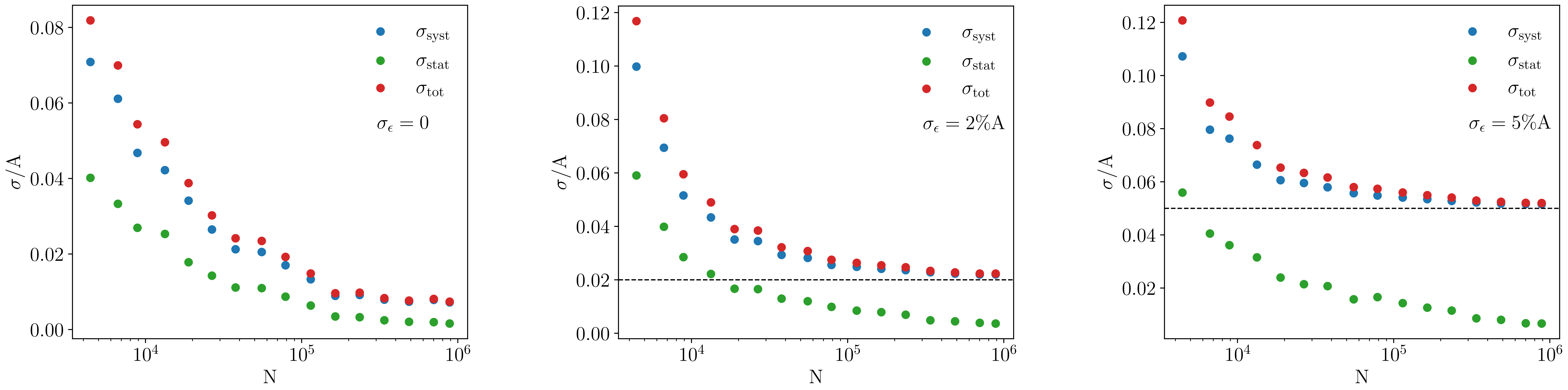
$$\sigma_{\text{train}} = f_{\text{smear}}\, A_{\text{true}}$$





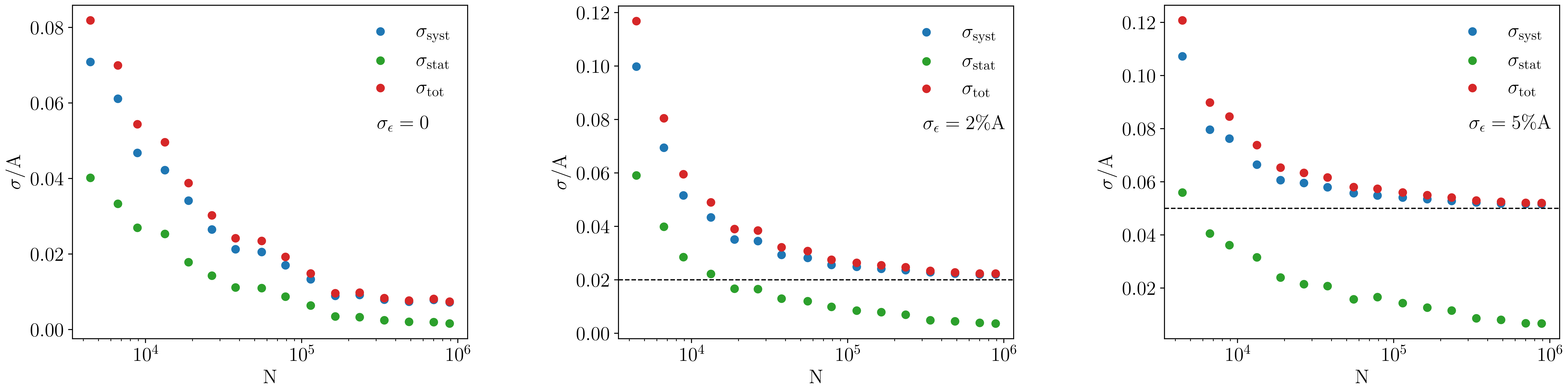
➡ Networks learn noise as systematic uncertainty

Uncertainty behavior



BNN only

Uncertainty behavior



BNN only

1. Statistical uncertainty **independent** of noise

2. Systematic uncertainty **plateaus** on noise level

Pull distribution

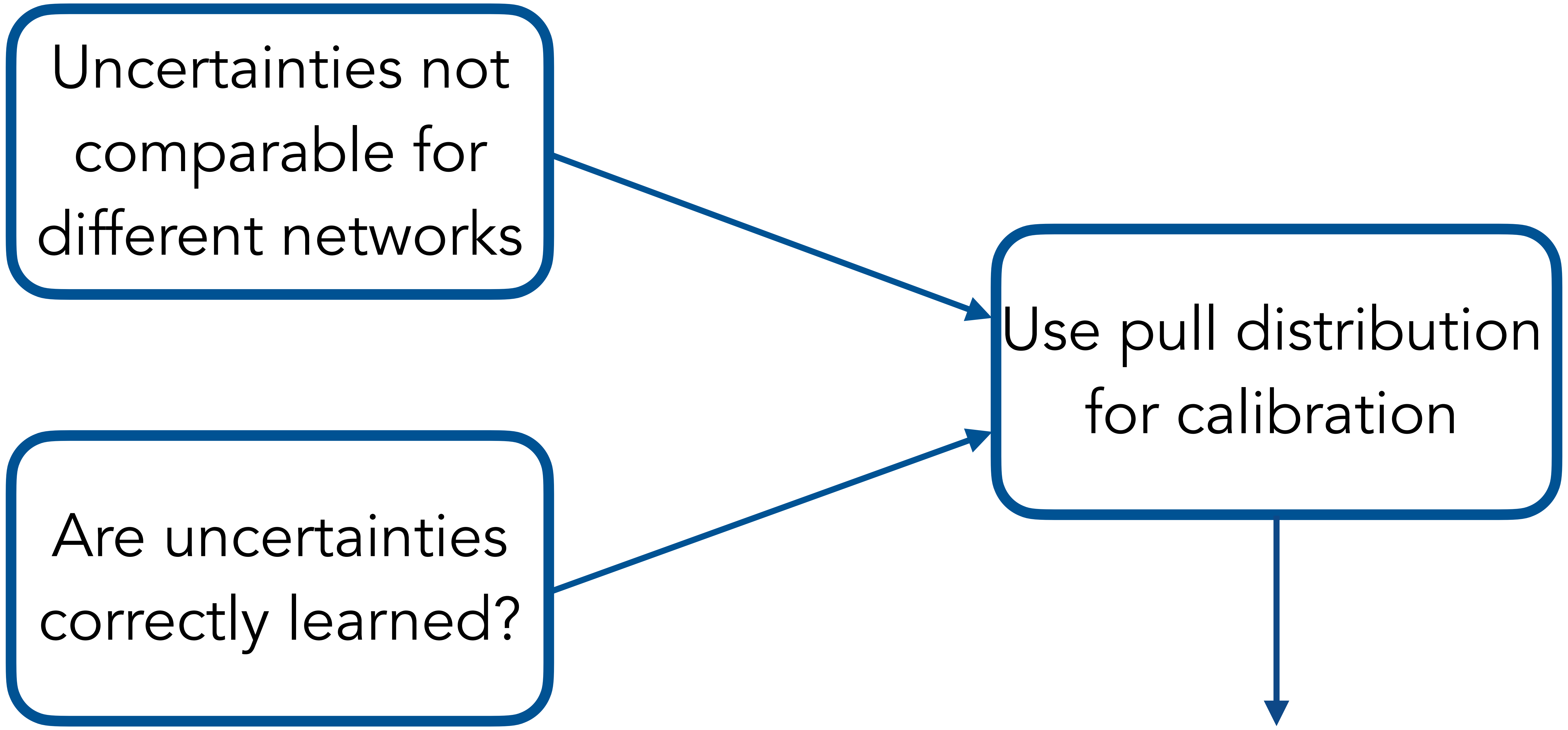
Problems:

Uncertainties not comparable for different networks

Are uncertainties correctly learned?

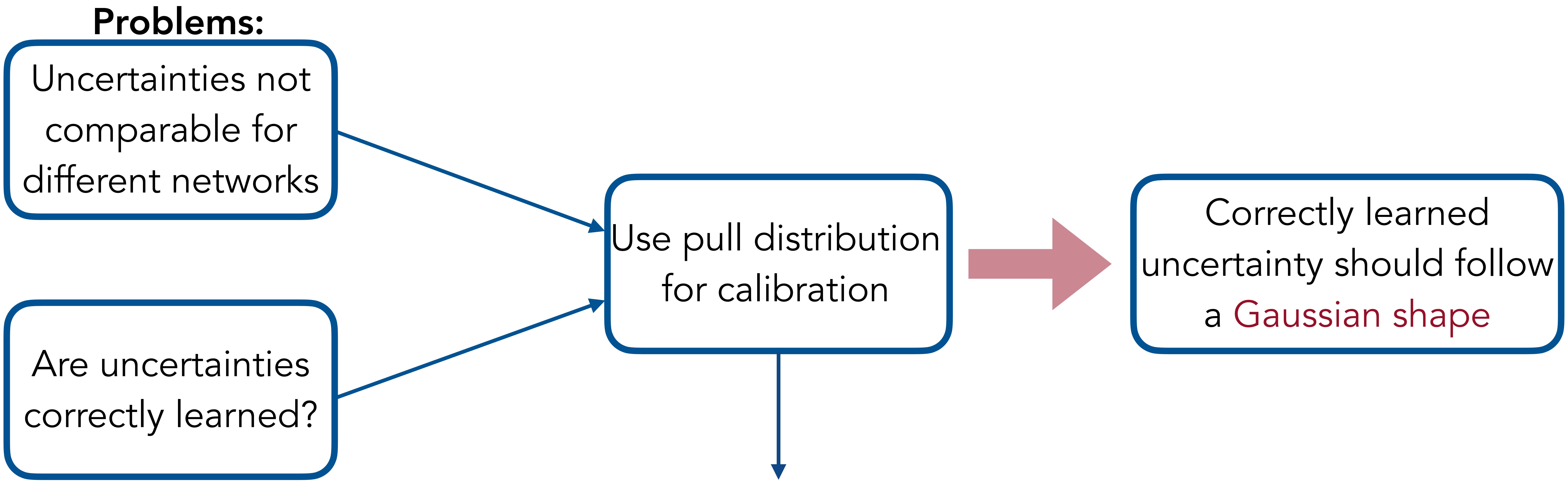
Pull distribution

Problems:



$$t_{\text{het}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{train}}(x)}{\sigma(x)}$$

Pull distribution



$$t_{\text{het}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{train}}(x)}{\sigma(x)}$$

Systematic pull

- Limit of perfect network training: $A_{NN} \to A_{true}$ with $q(\theta) = \delta(\theta - \theta_0)$
- Gaussian: $\langle A \rangle(x) \approx A_{true}(x)$ and $\sigma_{syst}(x) \approx \sigma_{train}(x)$

Systematic pull

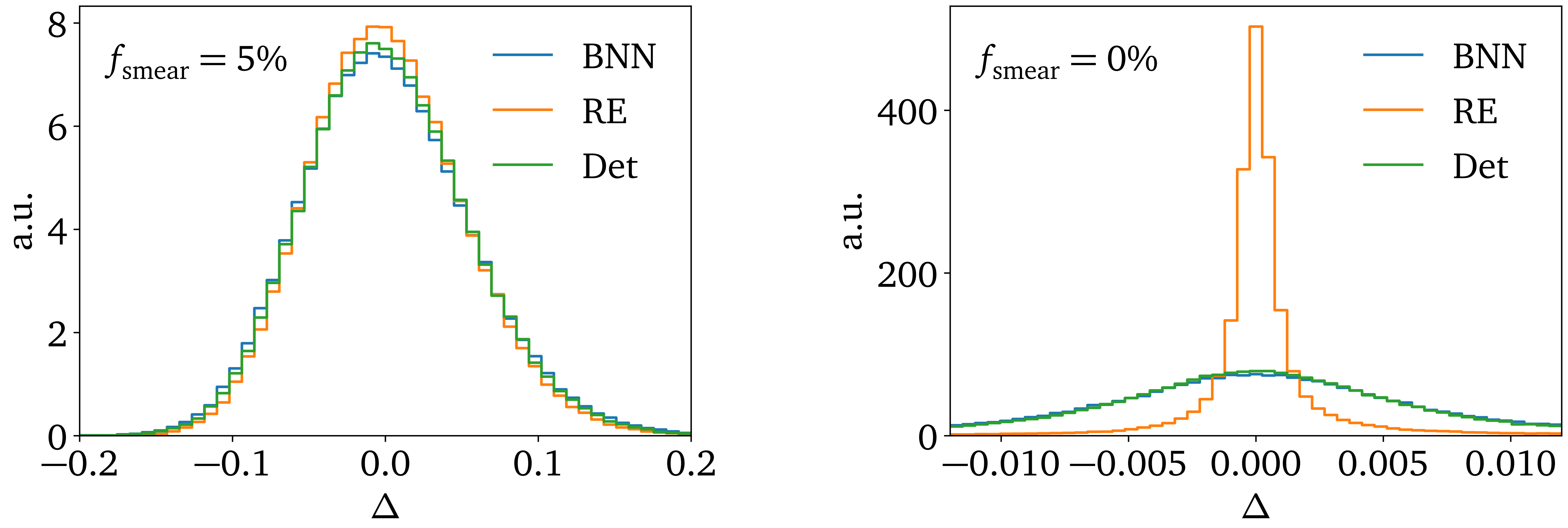
- Limit of perfect network training: $A_{NN} \to A_{true}$ with $q(\theta) = \delta(\theta - \theta_0)$
- Gaussian: $\langle A \rangle(x) \approx A_{true}(x)$ and $\sigma_{syst}(x) \approx \sigma_{train}(x)$
- BNN and RE: relates learned $A(x)$ to actual (noisy) training data $A_{\text{train}}(x)$

Systematic pull

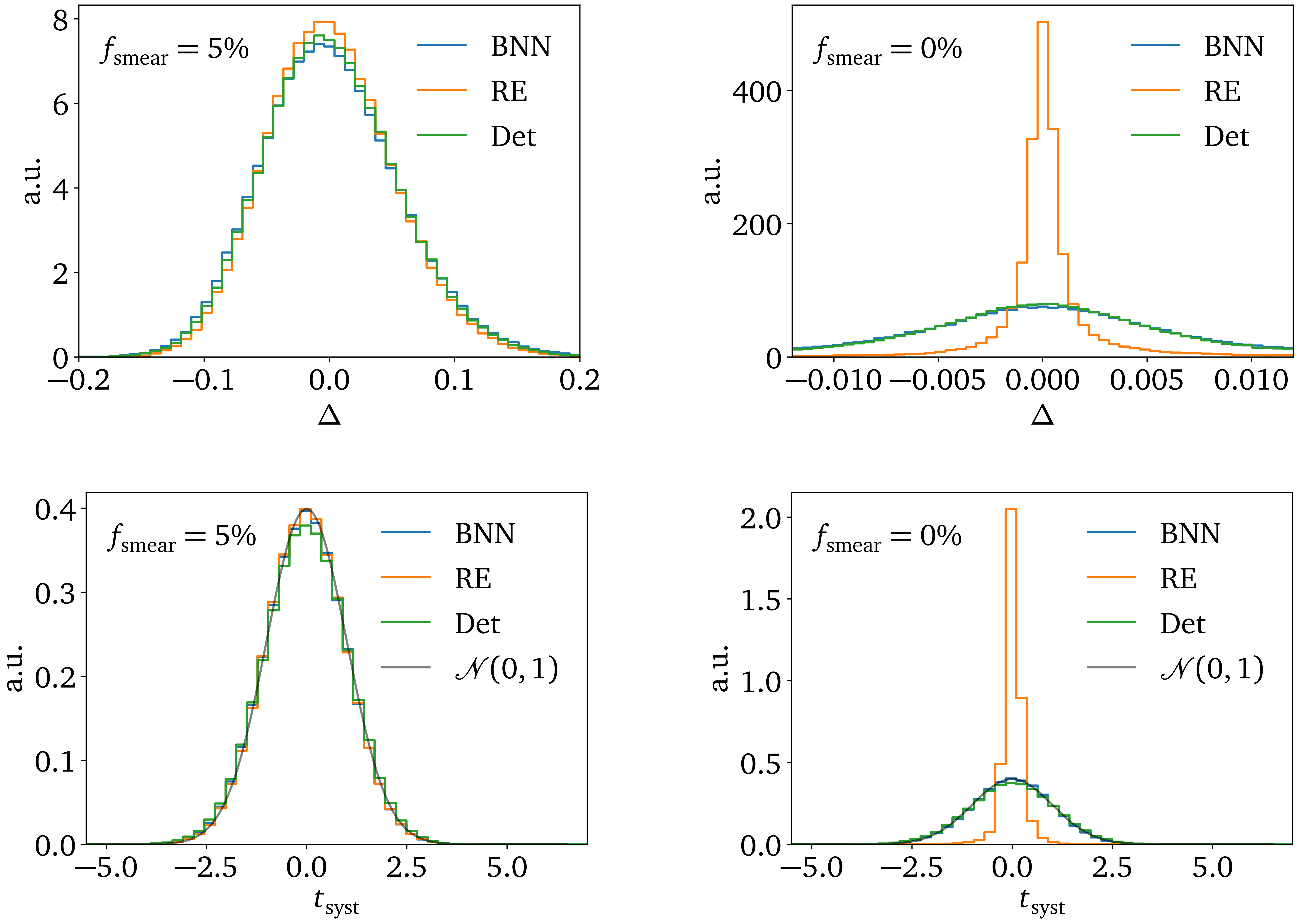
- Limit of perfect network training: $A_{NN} \to A_{true}$ with $q(\theta) = \delta(\theta - \theta_0)$
- Gaussian: $\langle A \rangle(x) \approx A_{true}(x)$ and $\sigma_{syst}(x) \approx \sigma_{train}(x)$
- BNN and RE: relates learned $A(x)$ to actual (noisy) training data $A_\text{train}(x)$

$$t_\text{syst}(x) = \frac{1}{\sigma_\text{syst}(x)} \int dA \; [A - A_\text{train}(x)] \; p(A|x, \theta_0)$$
$$= \frac{\langle A \rangle(x) - A_\text{train}(x)}{\sigma_\text{syst}(x)}$$

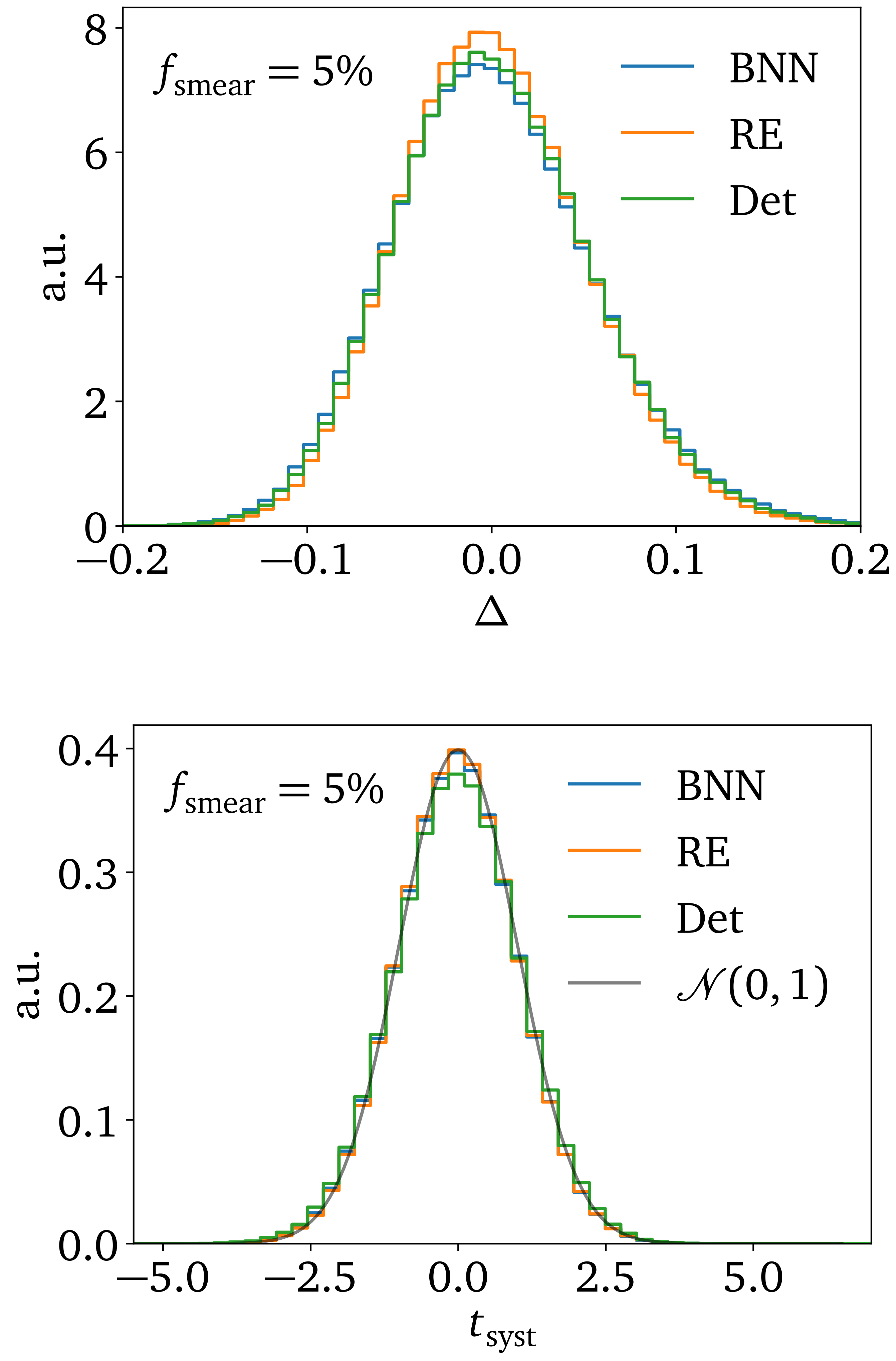
Systematic pull - Results



Systematic pull - Results



Systematic pull - Results



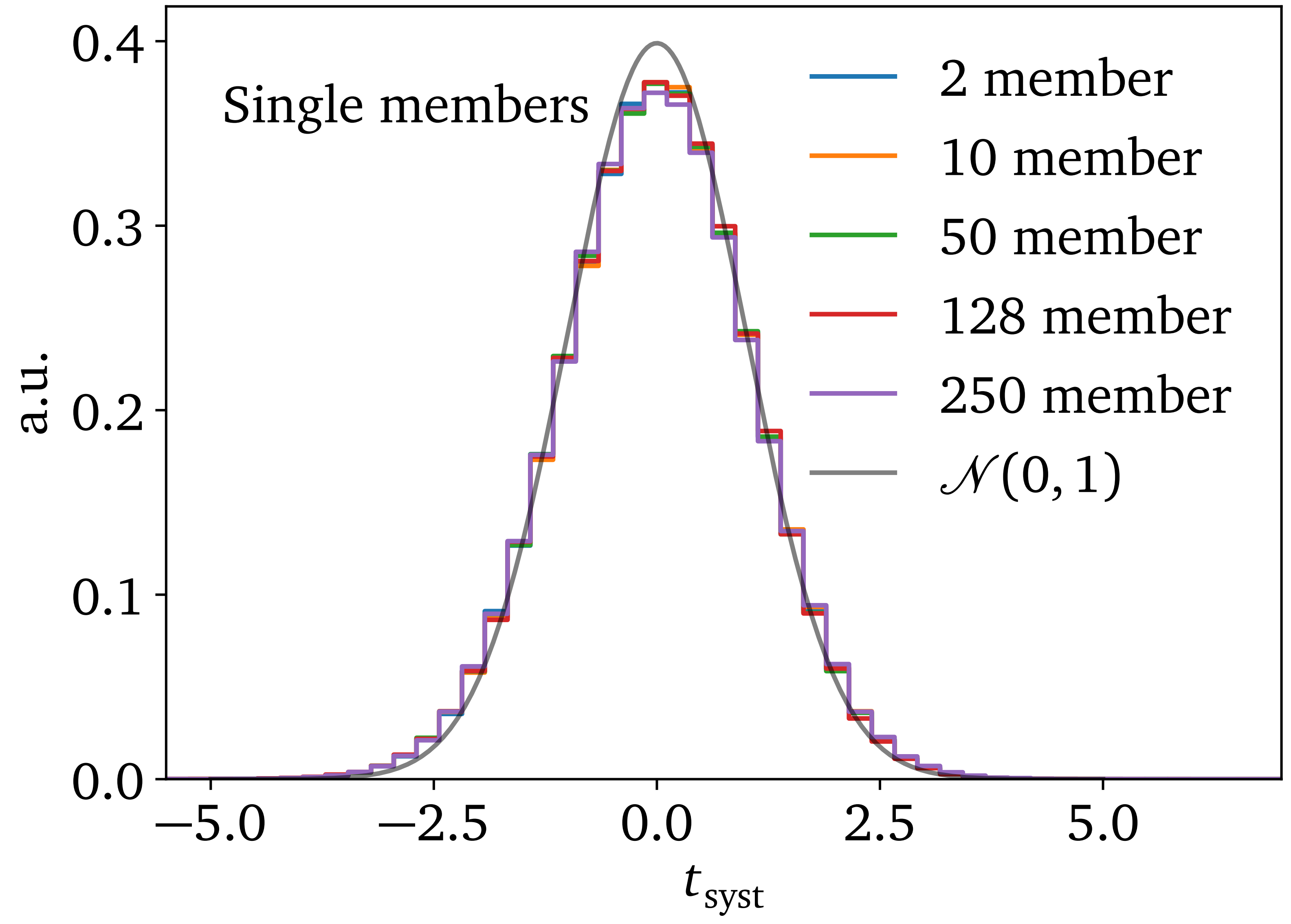
➡ Well calibrated results, except for RE without any noise

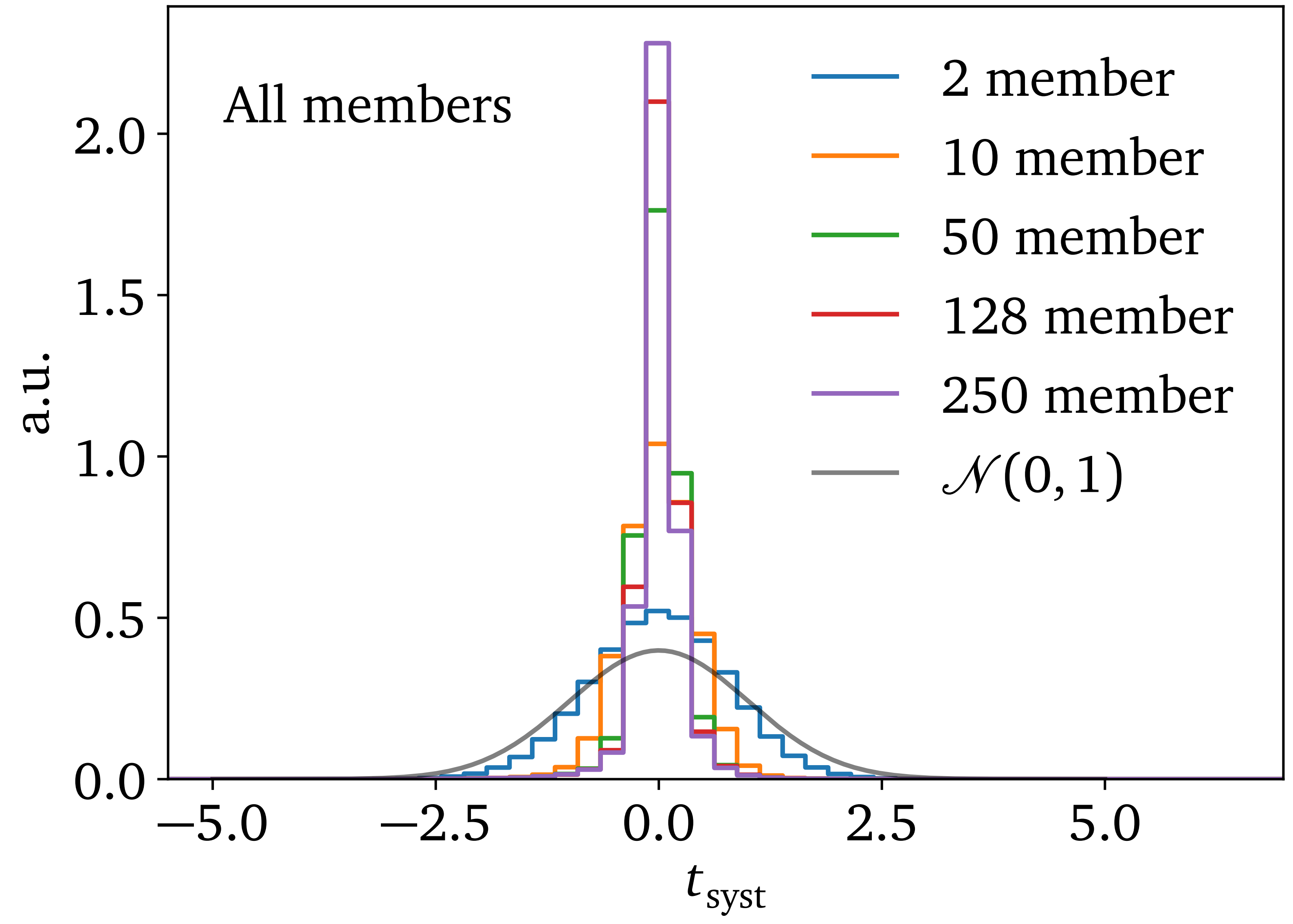
Systematic pull of REs

- Learned $\sigma_{syst}(x)$ too conservative

Systematic pull of REs

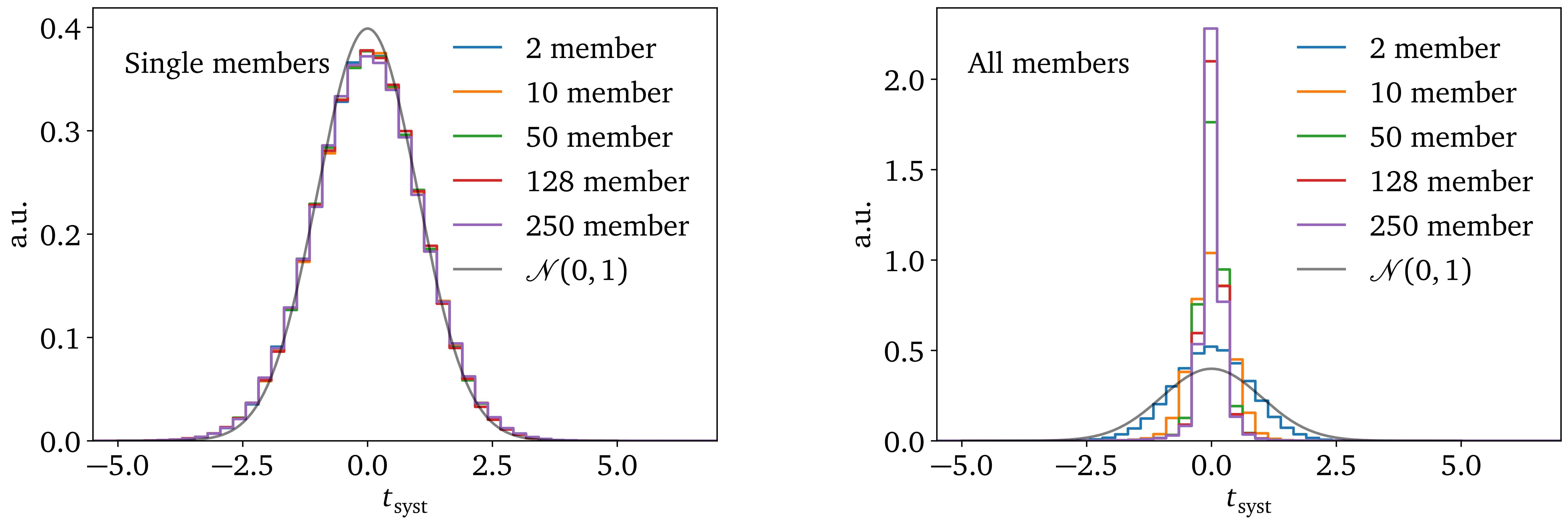
- Learned $\sigma_{syst}(x)$ too conservative





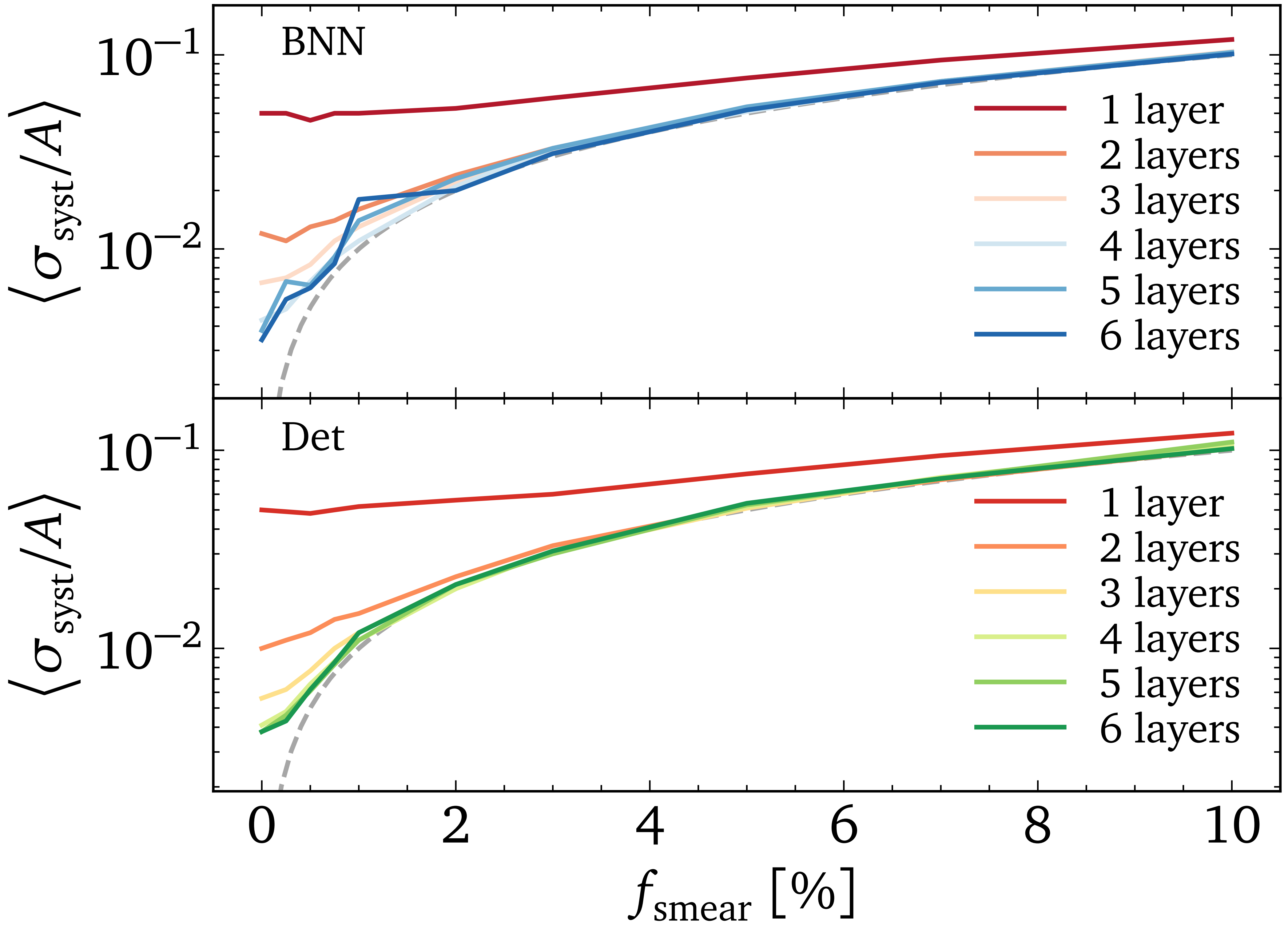
Systematic pull of REs

- Learned $\sigma_{syst}(x)$ too conservative

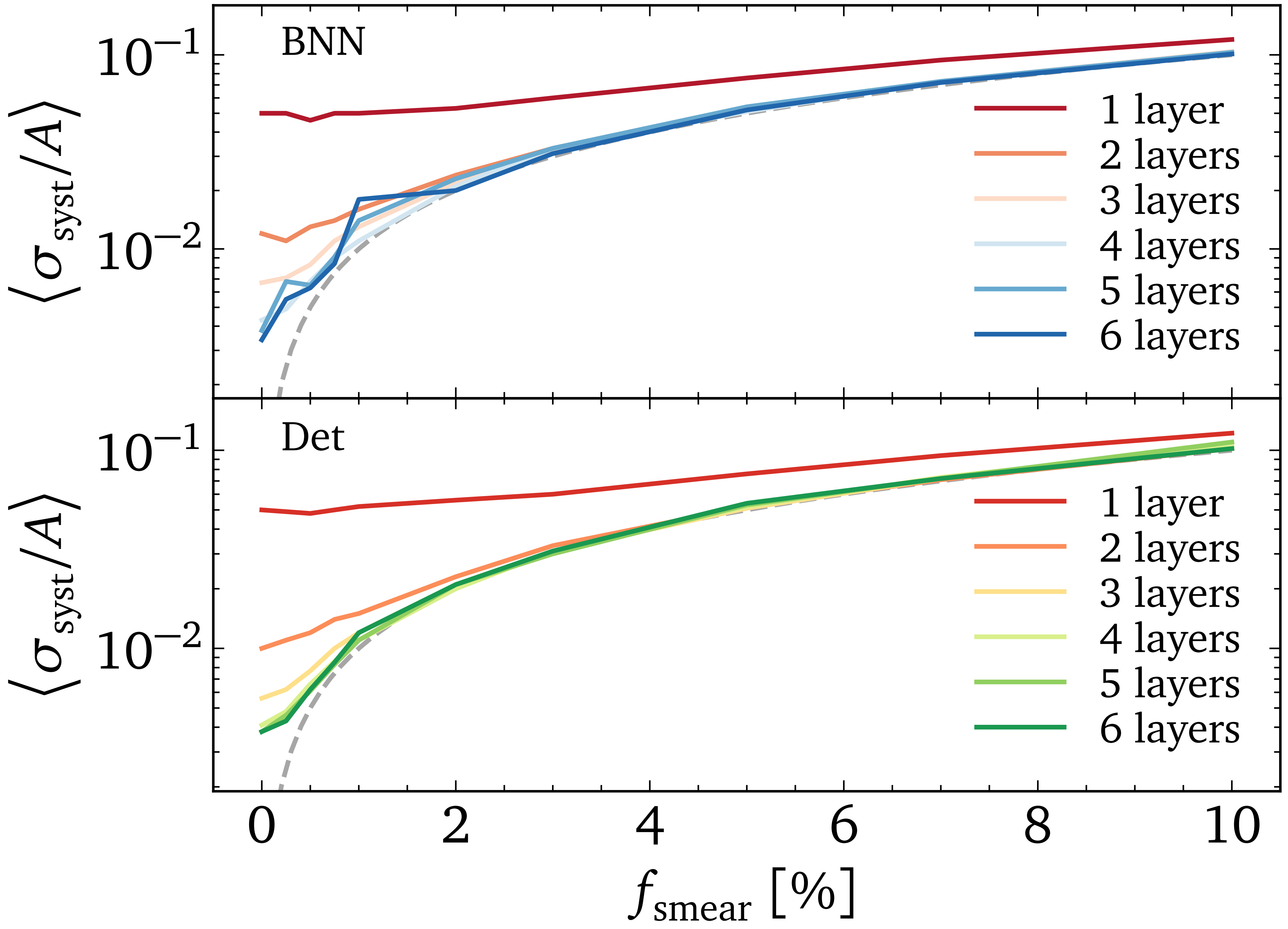


➡Prediction benefits from ensemble nature but not σ_{syst}

Systematics from network expressivity



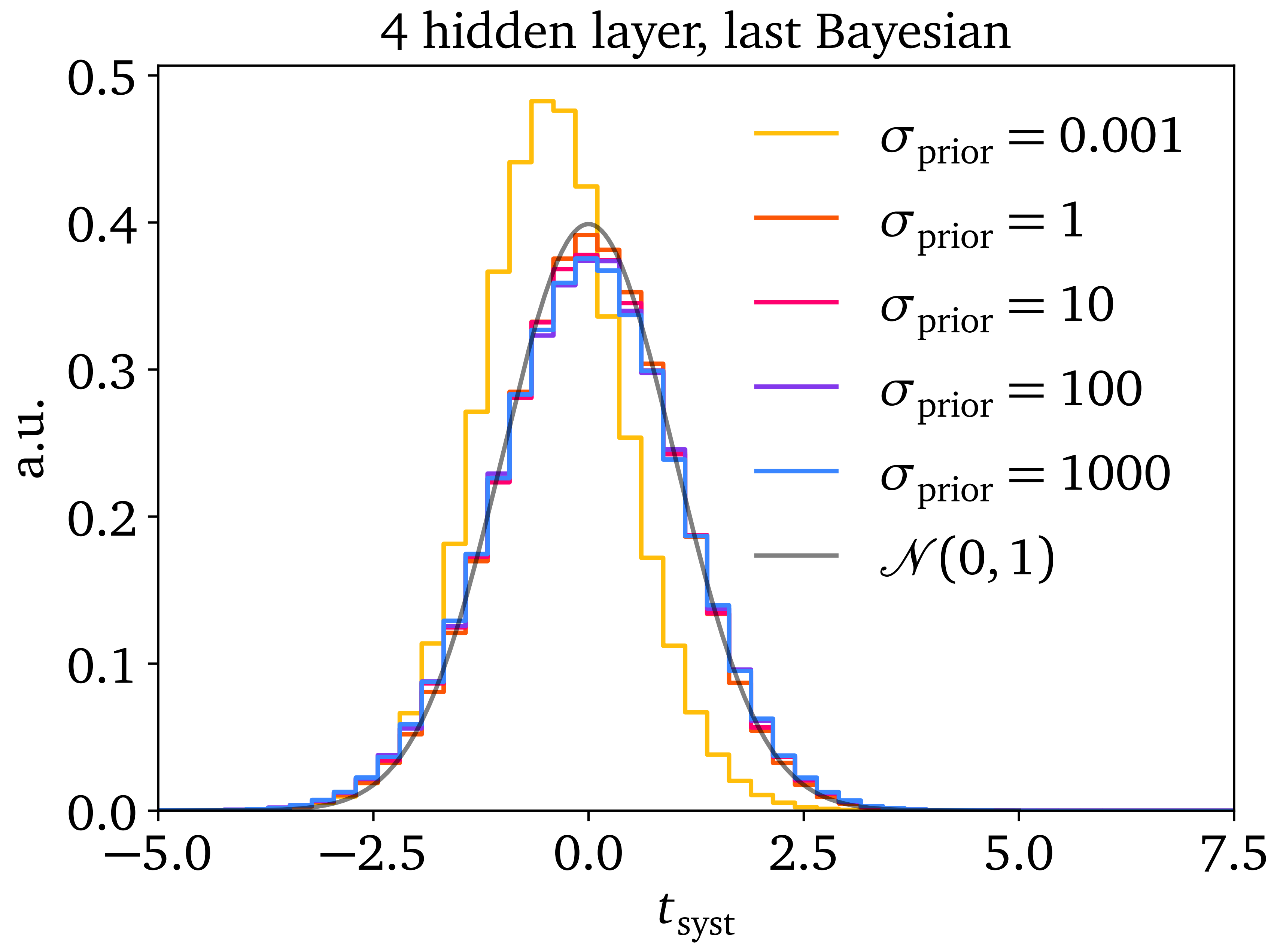
Systematics from network expressivity

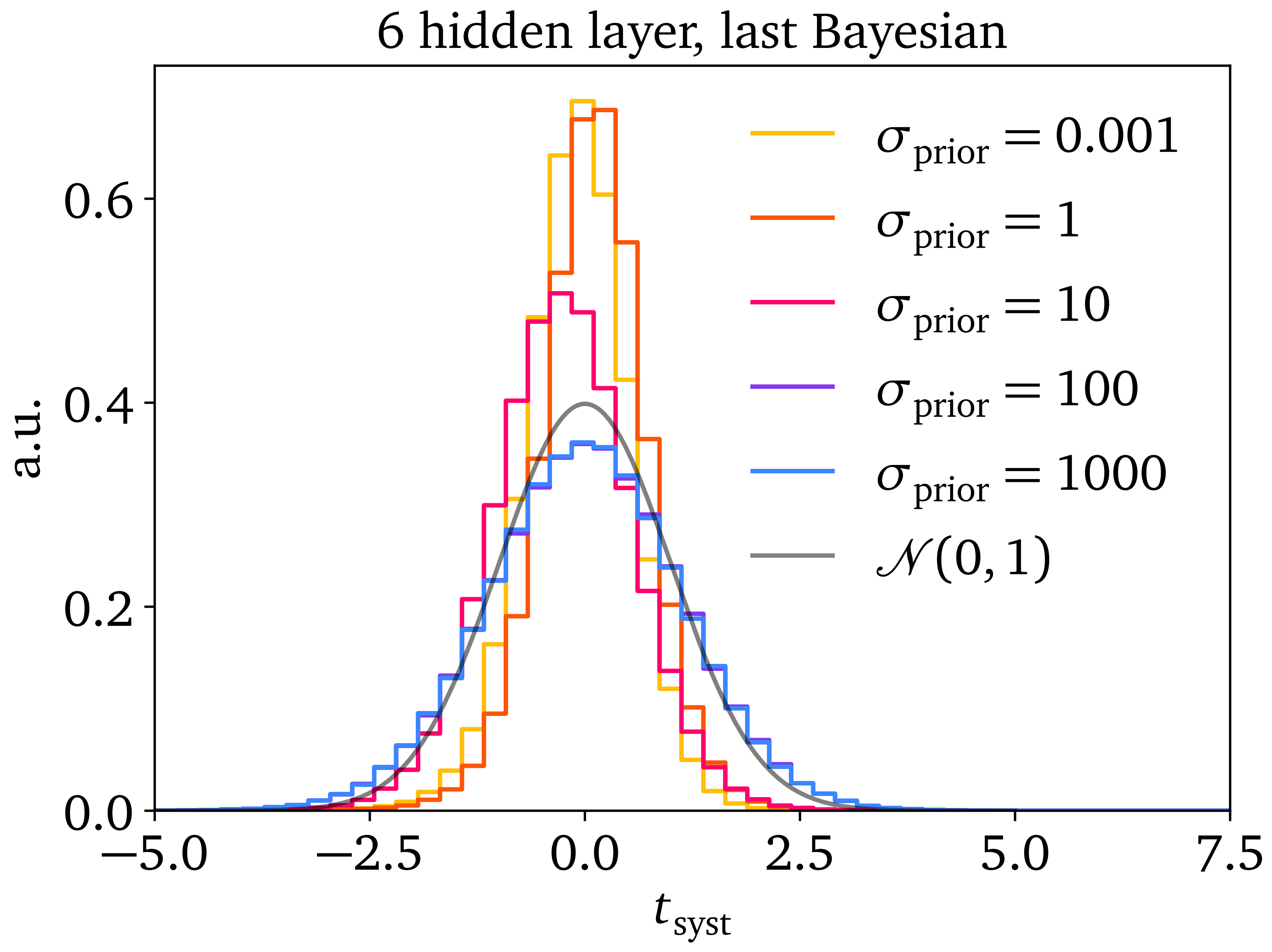


- Need at least three layers
- BNN: only last layer Bayesian
- Six layer: network gets too large

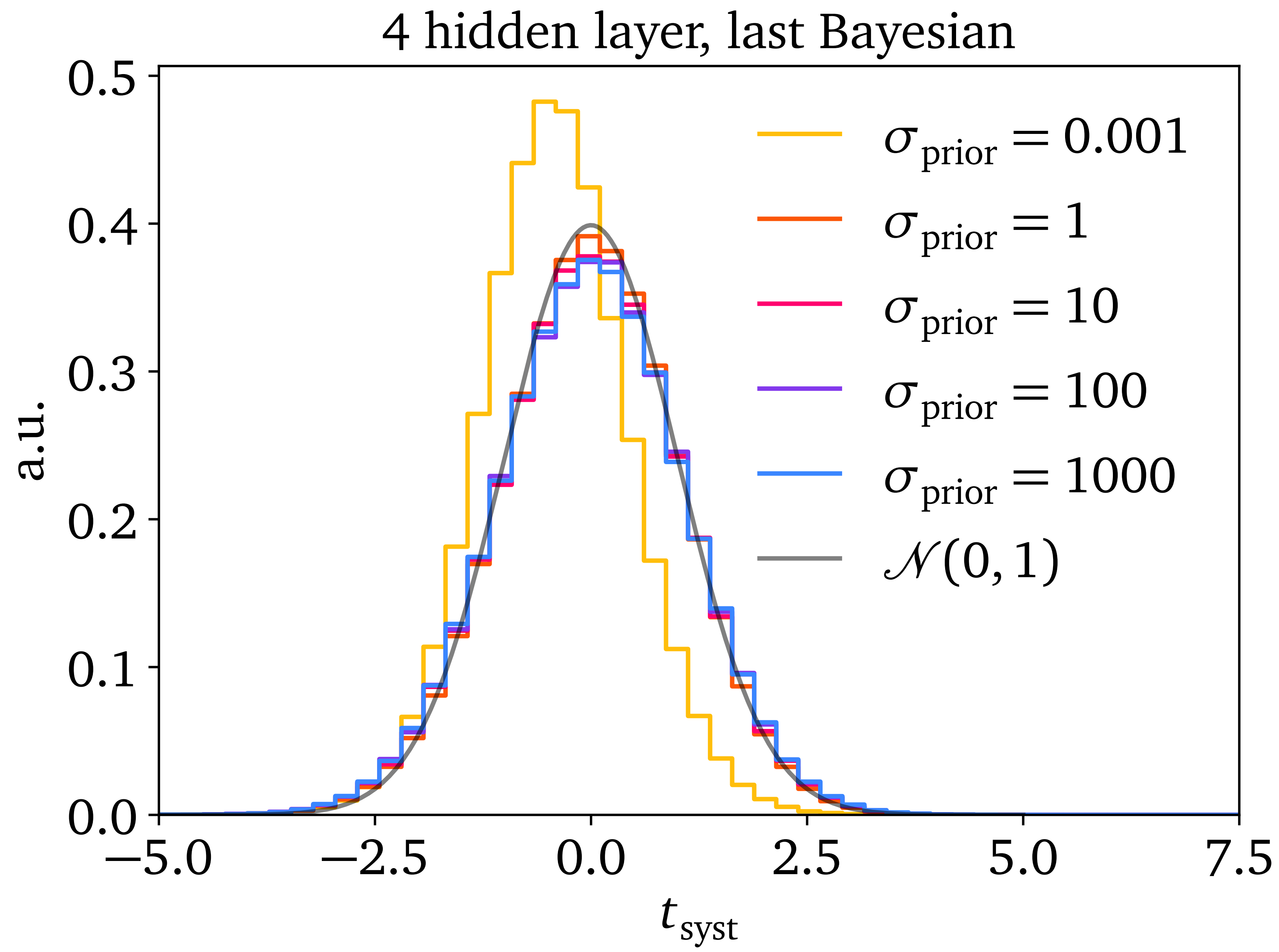
➡ More expressivity and better sensitivity for small noise with more layers

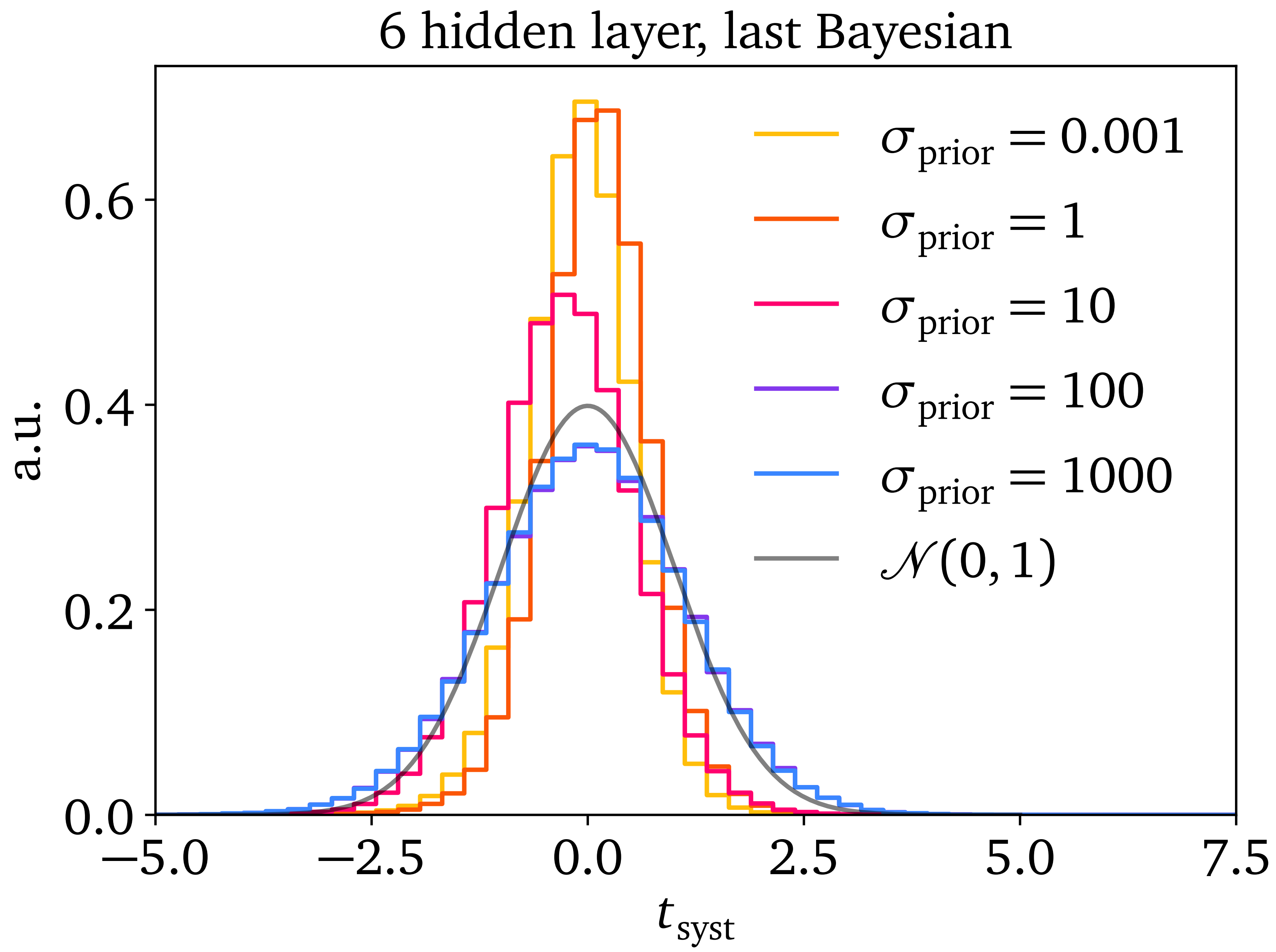
Prior influence in the BNN





Prior influence in the BNN

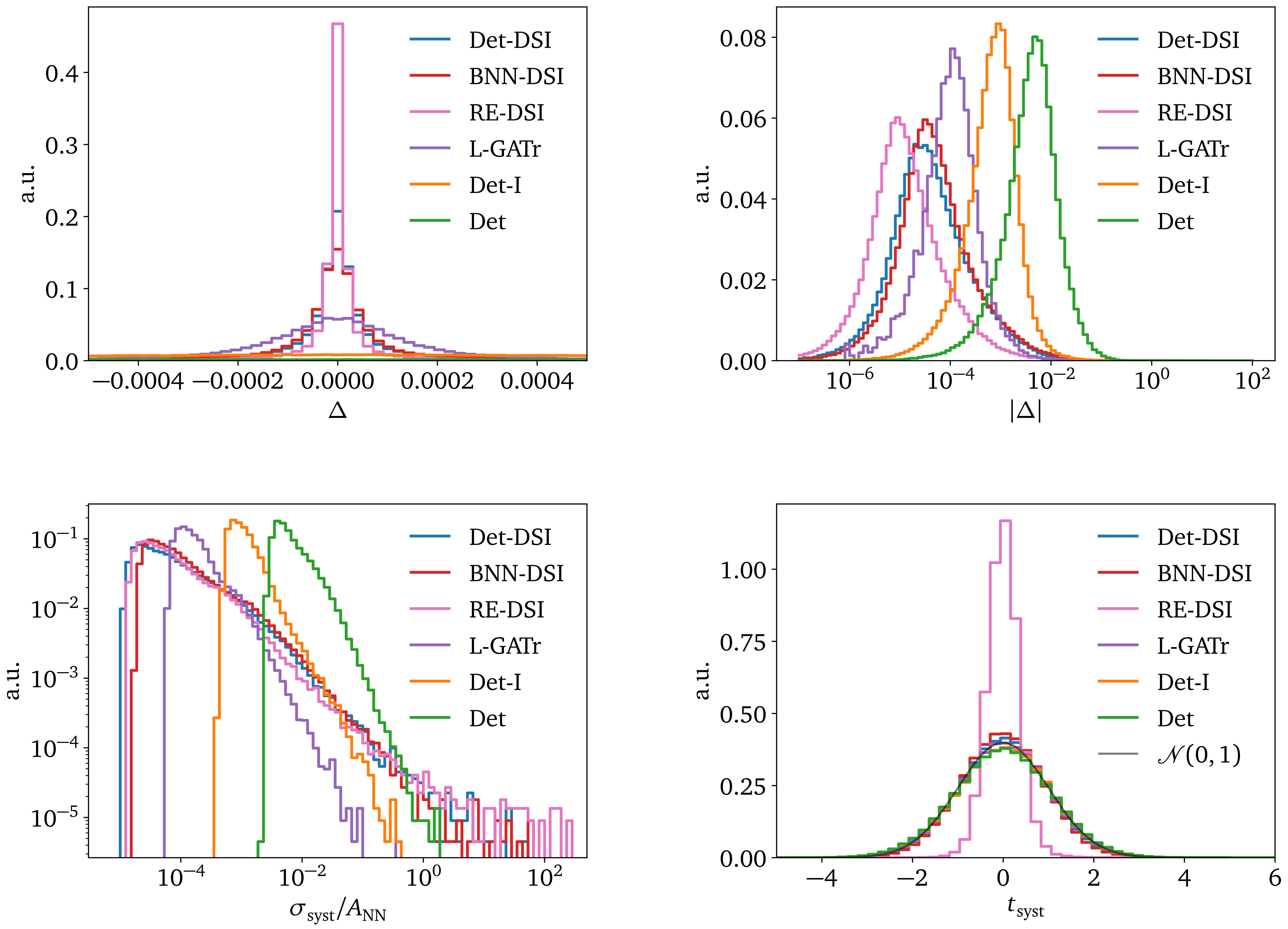




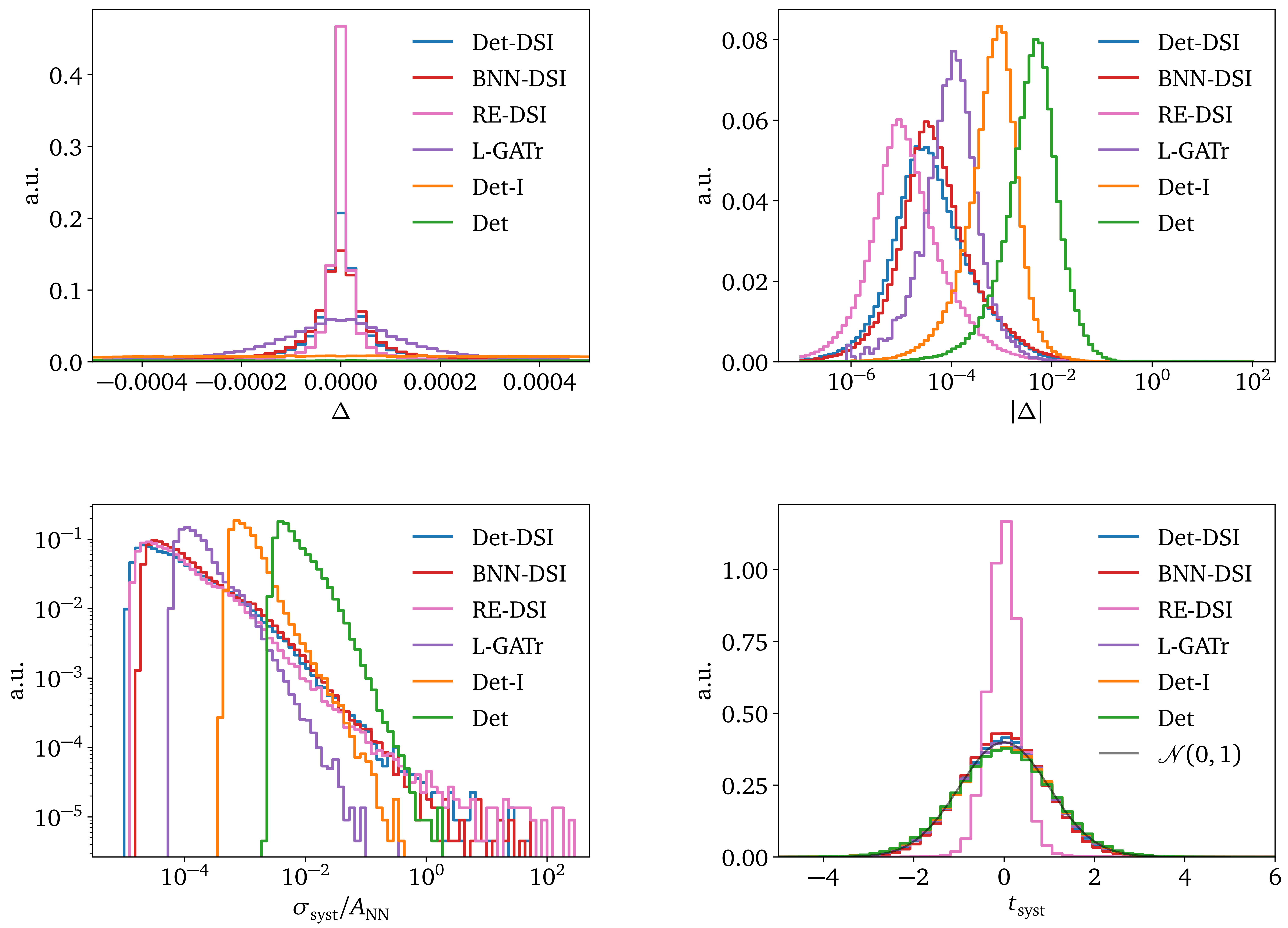
➡ Results don't depend on prior

➡ For more layers a larger prior is needed

Comparing to advanced architectures



Comparing to advanced architectures



- Calibrated σ_{syst} except for RE-variant

➡ Controlled accuracy on 10^{-5} level ($\sim 0.001\ \%$)

➡ Advanced architectures give better results

Outline

Part I: Different networks and architectures

Part II: Systematic uncertainties

Part III: **Statistical uncertainties**

Statistical pull

- Limit of infinite training data: $\langle A \rangle(x) = \bar{A}(x, \theta_0)$ with $q(\theta) = \delta(\theta - \theta_0)$
- Follows Gaussian structure in limit of $N \to \infty$

Statistical pull

- Limit of infinite training data: $\langle A \rangle(x) = \bar{A}(x, \theta_0)$ with $q(\theta) = \delta(\theta - \theta_0)$

network output of single member

- Follows Gaussian structure in limit of $N \to \infty$

Statistical pull

- Limit of infinite training data: $\langle A \rangle(x) = \bar{A}(x, \theta_0)$ with $q(\theta) = \delta(\theta - \theta_0)$

network output of single member

- Follows Gaussian structure in limit of $N \to \infty$
- $\sigma_{\text{stat}}(x)$ calculated from variance of $\bar{A}(x, \theta)$

Statistical pull

- Limit of infinite training data: $\langle A \rangle(x) = \bar{A}(x, \theta_0)$ with $q(\theta) = \delta(\theta - \theta_0)$

network output of single member

- Follows Gaussian structure in limit of $N \to \infty$

- $\sigma_{\text{stat}}(x)$ calculated from variance of $\bar{A}(x, \theta)$

network output of all members

Statistical pull

- Limit of infinite training data: $\langle A \rangle(x) = \bar{A}(x, \theta_0)$ with $q(\theta) = \delta(\theta - \theta_0)$

network output of single member

- Follows Gaussian structure in limit of $N \to \infty$
- $\sigma_{\text{stat}}(x)$ calculated from variance of $\bar{A}(x, \theta)$

network output of all members

$$t_{\text{stat}}(x, \theta) = \frac{\overline{A}(x, \theta) - A_{\text{true}}(x)}{\sigma_{\text{stat}}(x)}$$

Scaled statistical pull

- Statistical uncertainty and pull based on mean and variance
- Scales with number of samples M

Scaled statistical pull

- Statistical uncertainty and pull based on mean and variance
- Scales with number of samples M

$$\sigma_{\mathrm{stat,M}}(x) = \frac{\sigma_{\mathrm{stat}}(x)}{\sqrt{M}}$$

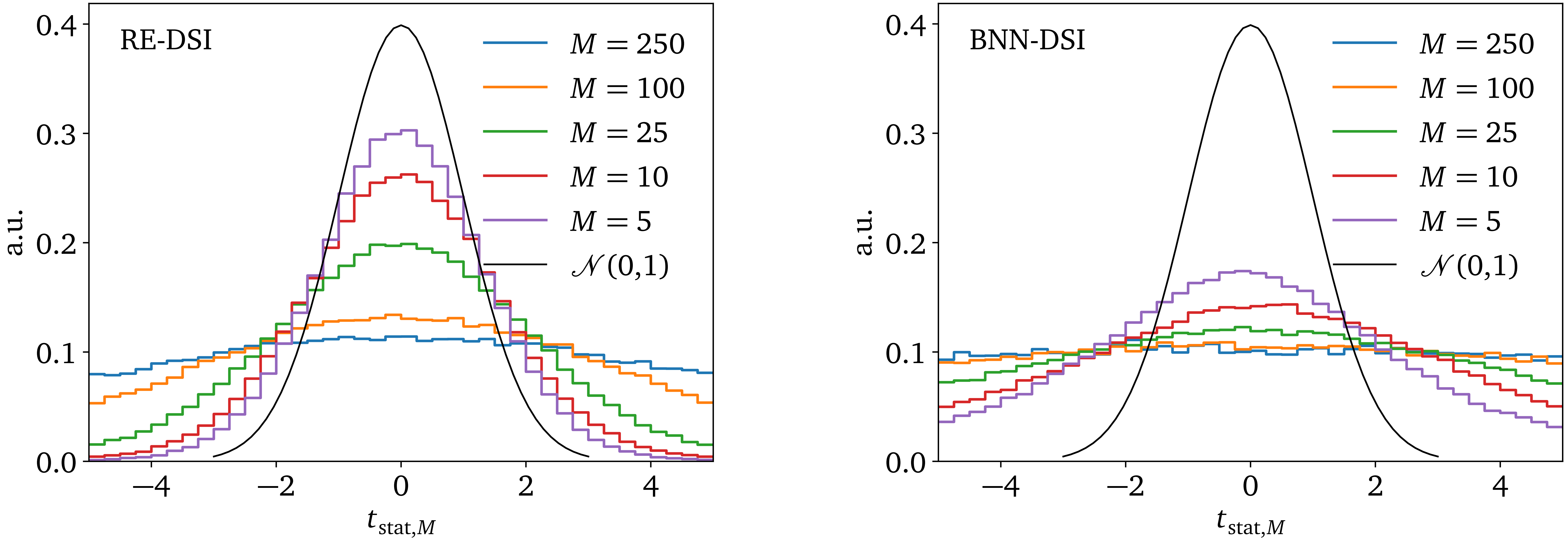
Scaled statistical pull

- Statistical uncertainty and pull based on mean and variance
- Scales with number of samples M

$$\sigma_{\mathrm{stat,M}}(x) = \frac{\sigma_{\mathrm{stat}}(x)}{\sqrt{M}}$$

$$\hat{t}_{\mathrm{stat},M}(x) = \frac{\langle A \rangle_M(x) - \langle A \rangle(x)}{\sigma_{\mathrm{stat,M}}(x)}$$

Scaled statistical pull



- Use N=512 samples for BNN and RE
- Evaluate scaled pull for subset M
- M→N: correlation increases, $\sigma_{\text{stat,M}} \to \sigma_{\text{stat}}$

Reducing the training size

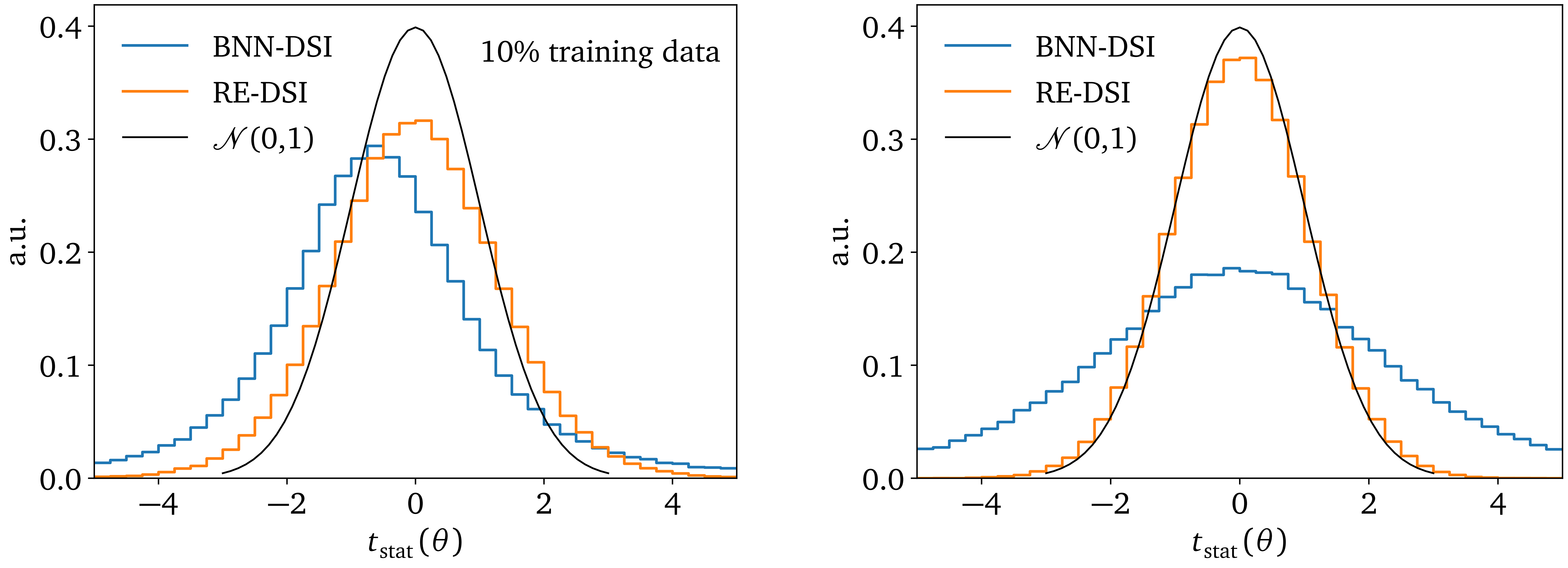
- Systematic uncertainty dominant over statsitical
- Training on 700000 phase space points: $\sigma_{\text{tot}}(x) \approx \sigma_{\text{syst}}(x) \gg \sigma_{\text{stat}}(x)$
- Reducing training data to 100000 phase space points

Reducing the training size

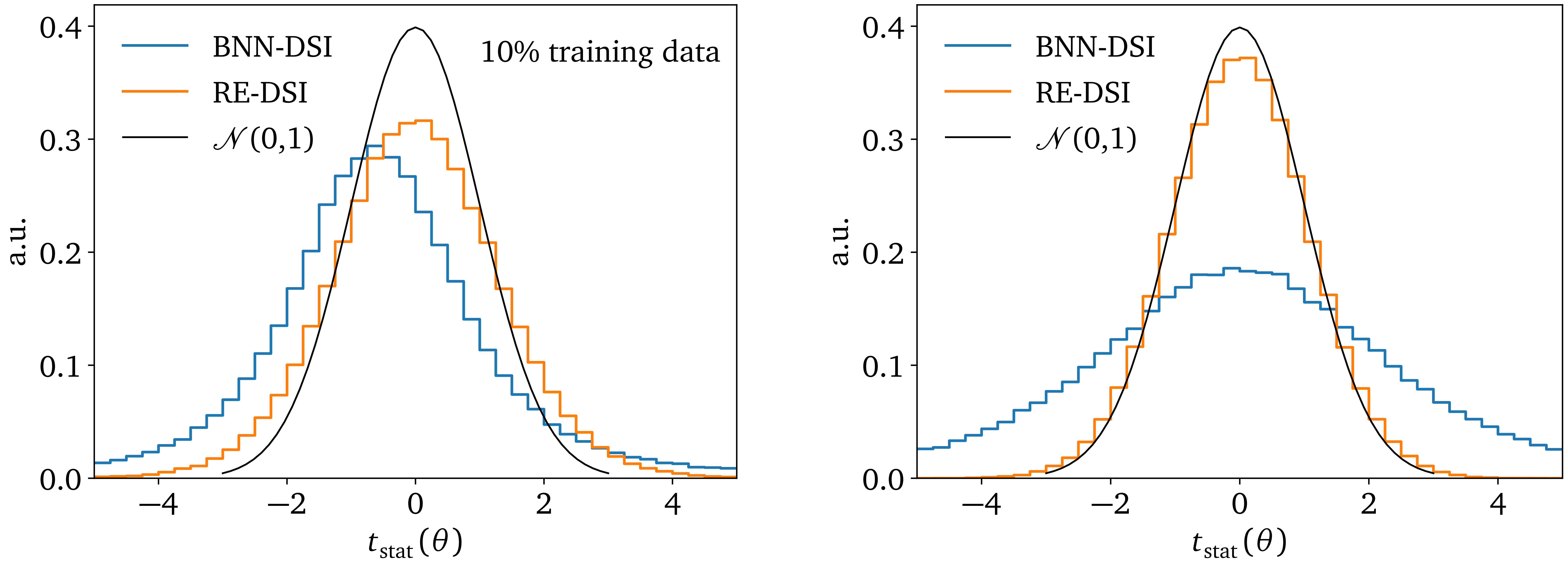
- Systematic uncertainty dominant over statsitical
- Training on 700000 phase space points: $\sigma_{\text{tot}}(x) \approx \sigma_{\text{syst}}(x) \gg \sigma_{\text{stat}}(x)$
- Reducing training data to 100000 phase space points

	70%	10%
$\langle \sigma_{\text{syst, BNN-DSI}}/A \rangle$	$8.7 \cdot 10^{-5}$	$2.5 \cdot 10^{-4}$
$\langle \sigma_{\text{stat, BNN-DSI}}/A \rangle$	$3.6 \cdot 10^{-5}$	$1.5 \cdot 10^{-4}$
$\langle \sigma_{\text{syst, RE-DSI}}/A \rangle$	$5.1 \cdot 10^{-5}$	$2.9 \cdot 10^{-4}$
$\langle \sigma_{\text{stat, RE-DSI}}/A \rangle$	$4.8 \cdot 10^{-5}$	$2.2 \cdot 10^{-4}$

Dependence on training size

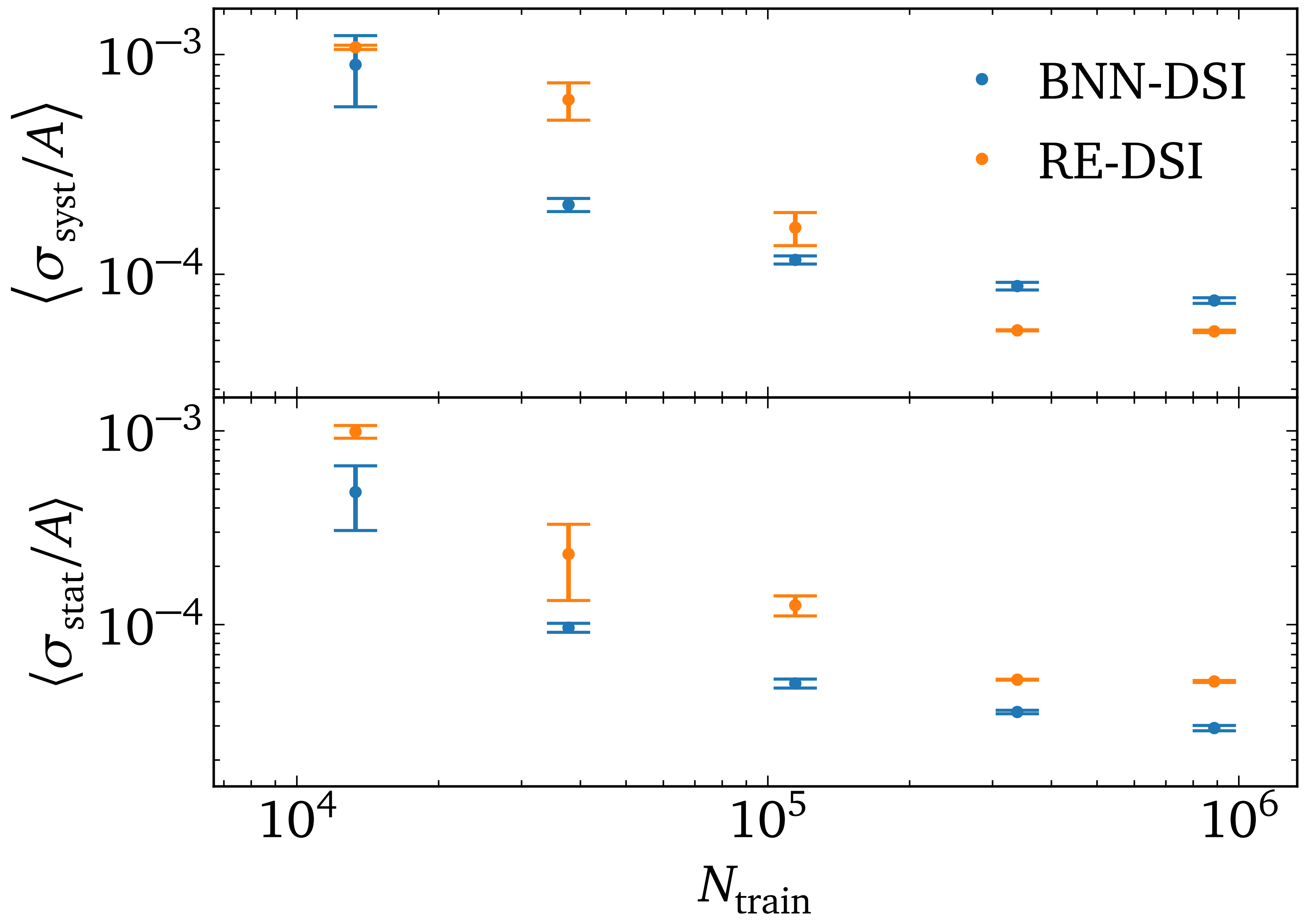


Dependence on training size

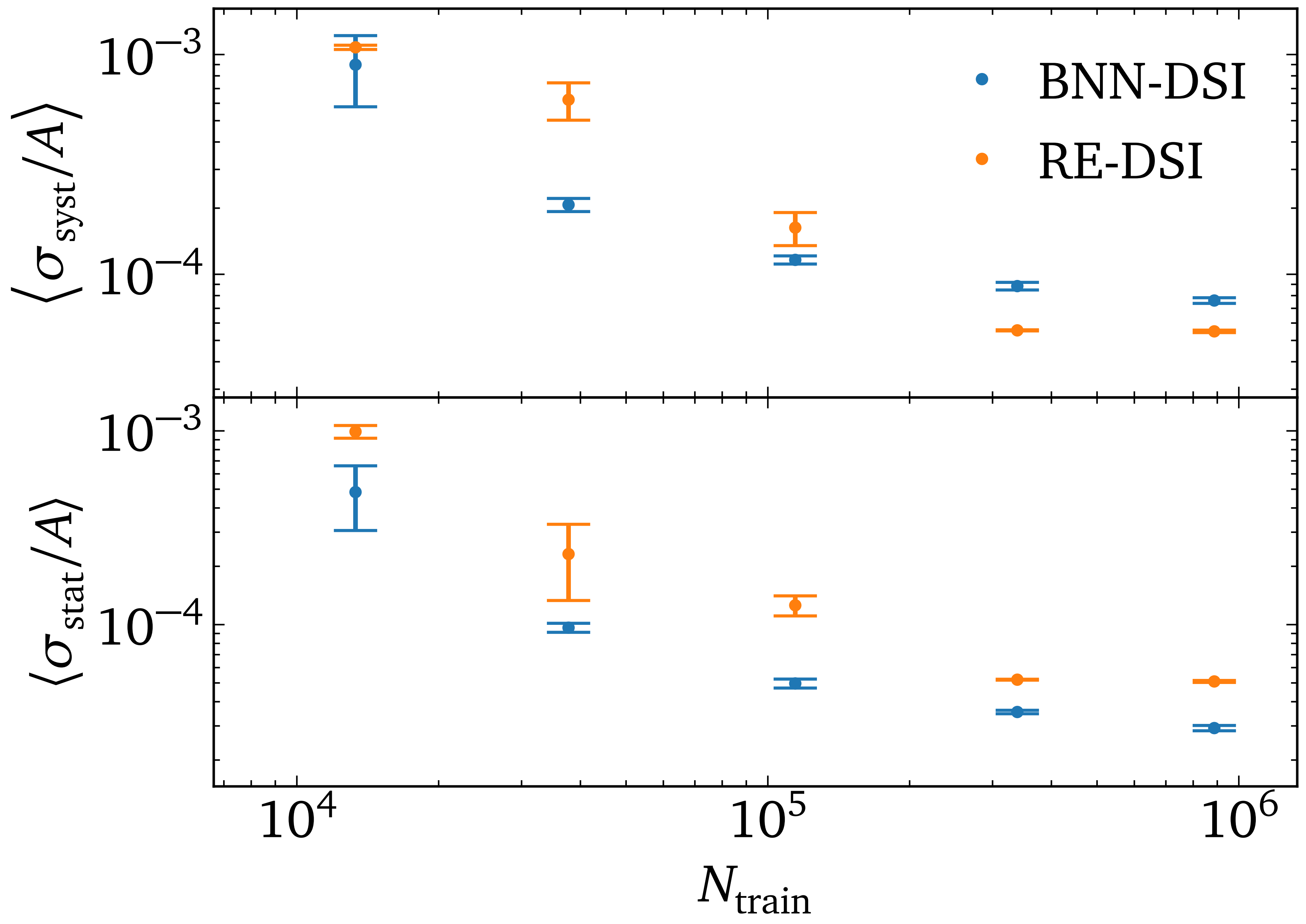


- RE good calibrated due to ensemble nature
- BNN good calibrated for small training, large training: overconfident

Relative uncertainty vs training size



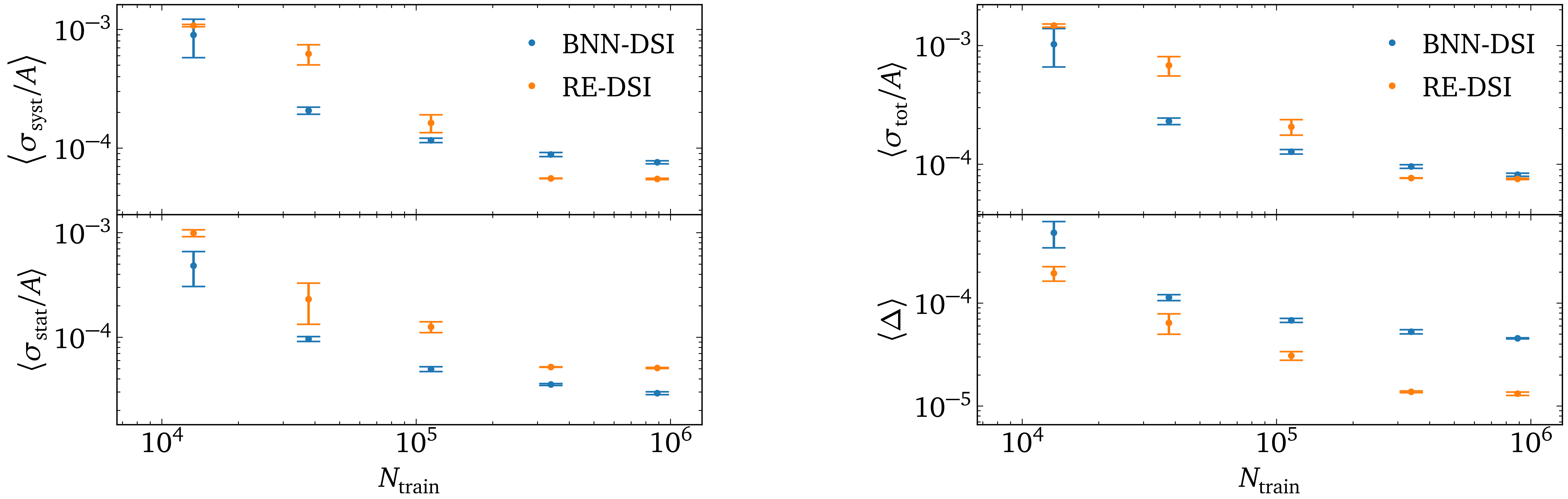
Relative uncertainty vs training size



➡ σ_{syst} always larger

➡ σ larger for RE-DSI than BNN-DSI

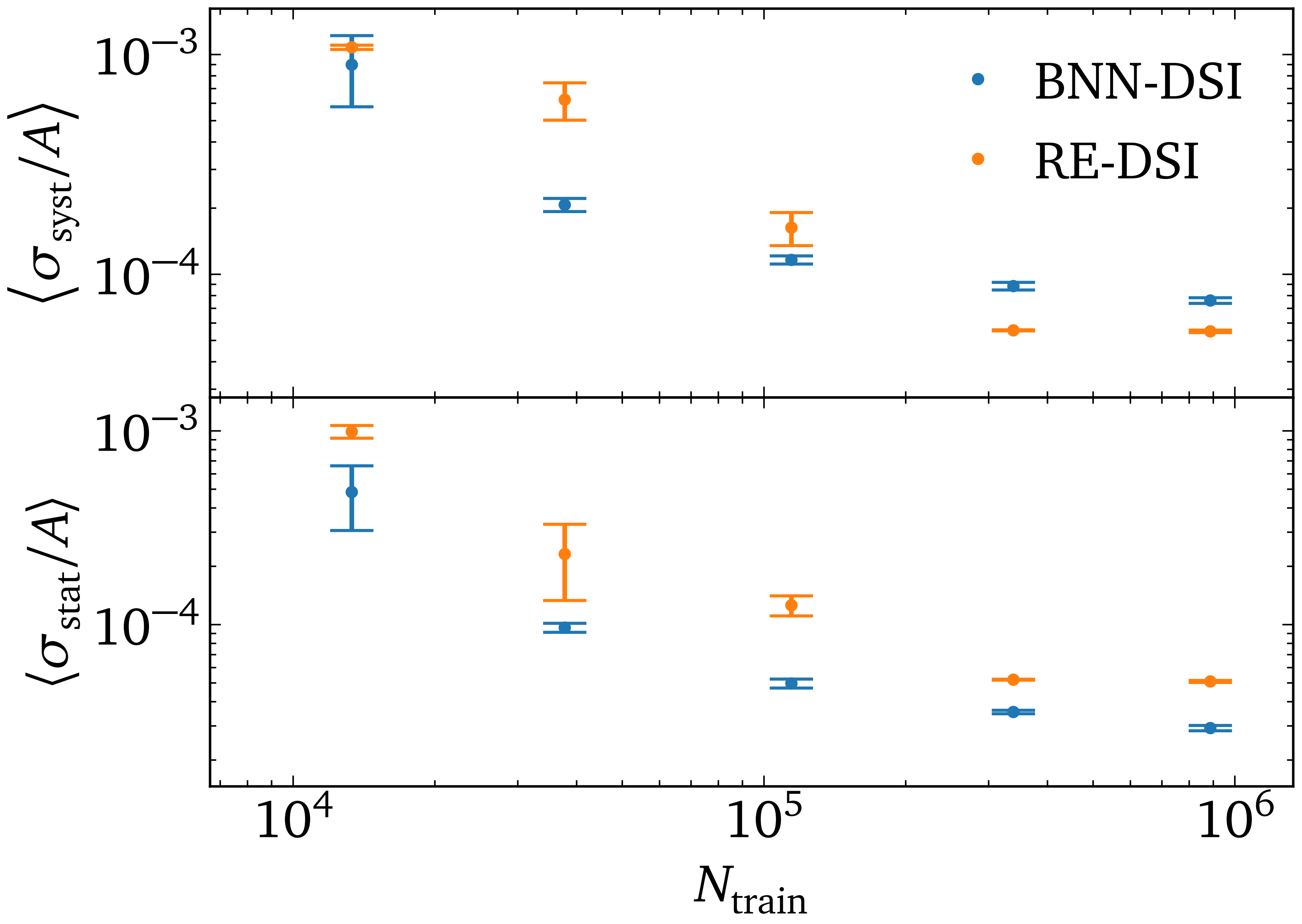
Relative uncertainty vs training size

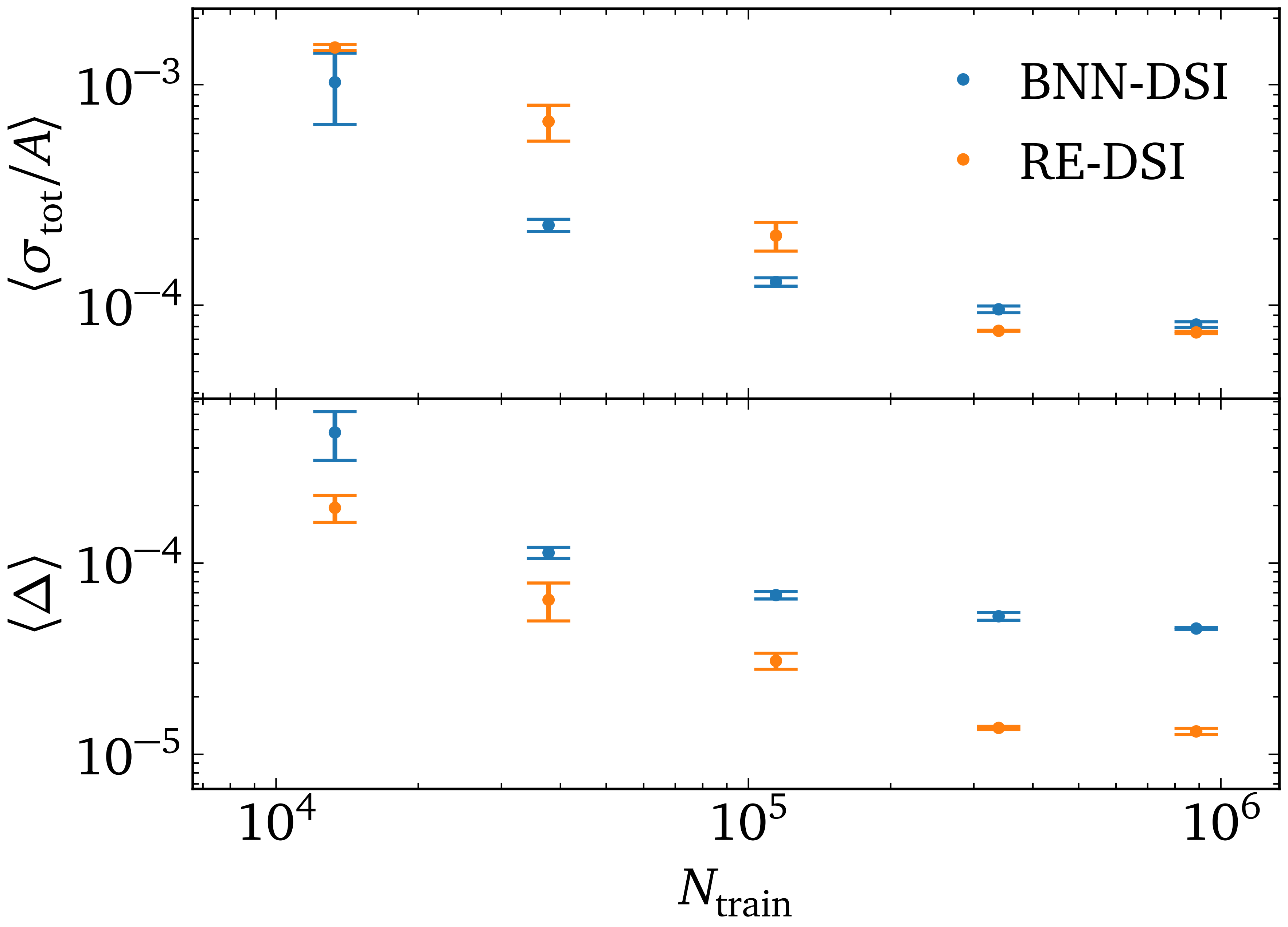


➡ σ_{syst} always larger

➡ σ larger for RE-DSI than BNN-DSI

Relative uncertainty vs training size





➡ σ_{syst} always larger

➡ Difference in σ_{tot} for small data

➡ σ larger for RE-DSI than BNN-DSI

➡ RE-DSI more accurate in prediction

Conclusion

1. Able to track systematic and statistical uncertainties
2. Networks mostly well calibrated (if not: calibration possible)
3. RE benefit from ensemble nature in precision
4. Advanced networks are able to give controlled accuracy on 10^{-5} level

Conclusion

1. Able to track systematic and statistical uncertainties
2. Networks mostly well calibrated (if not: calibration possible)
3. RE benefit from ensemble nature in precision
4. Advanced networks are able to give controlled accuracy on 10^{-5} level

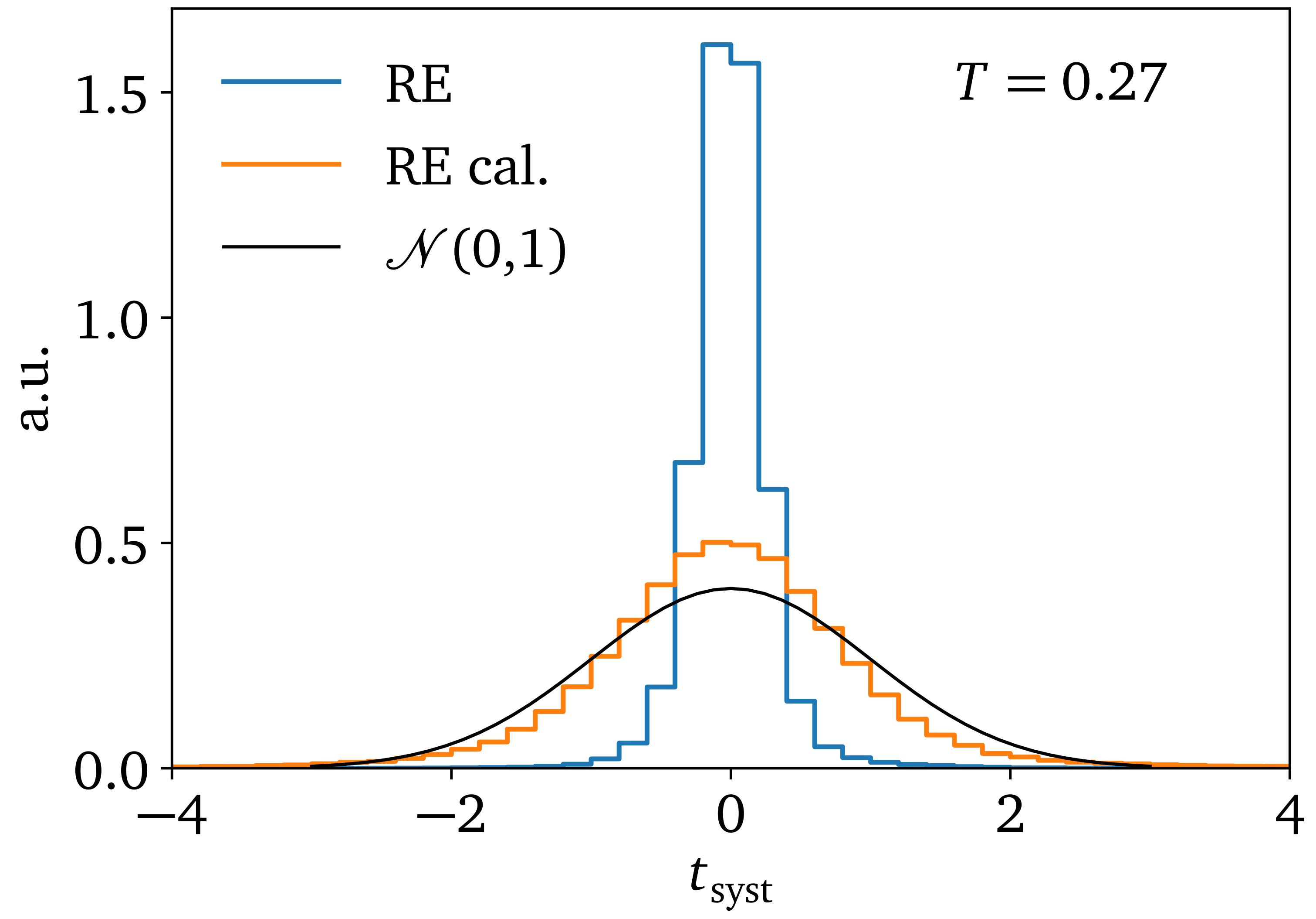
Thank you for your attention!

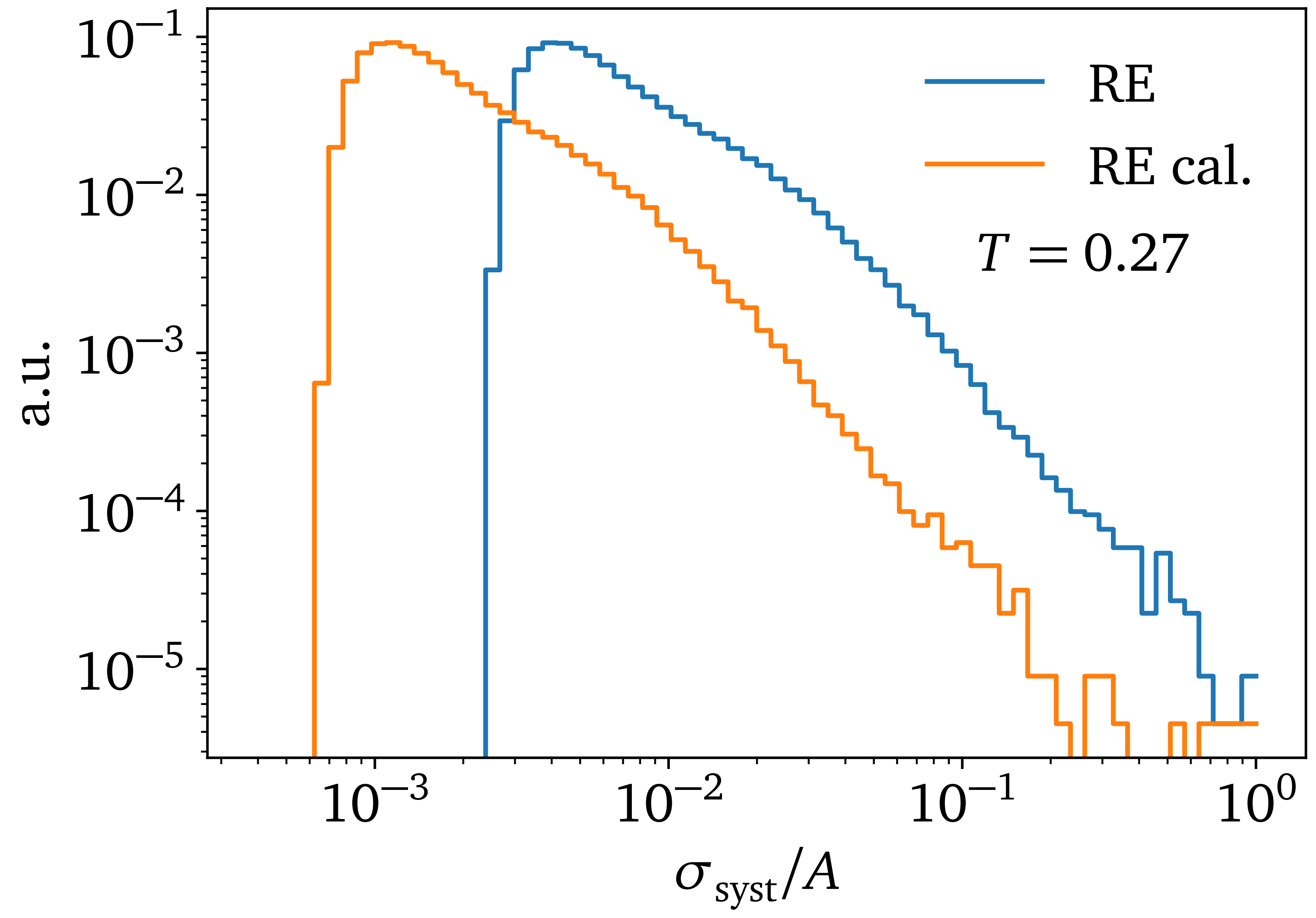
Happy holidays! 🎅🎄

Back up / Additional material

Calibration of networks

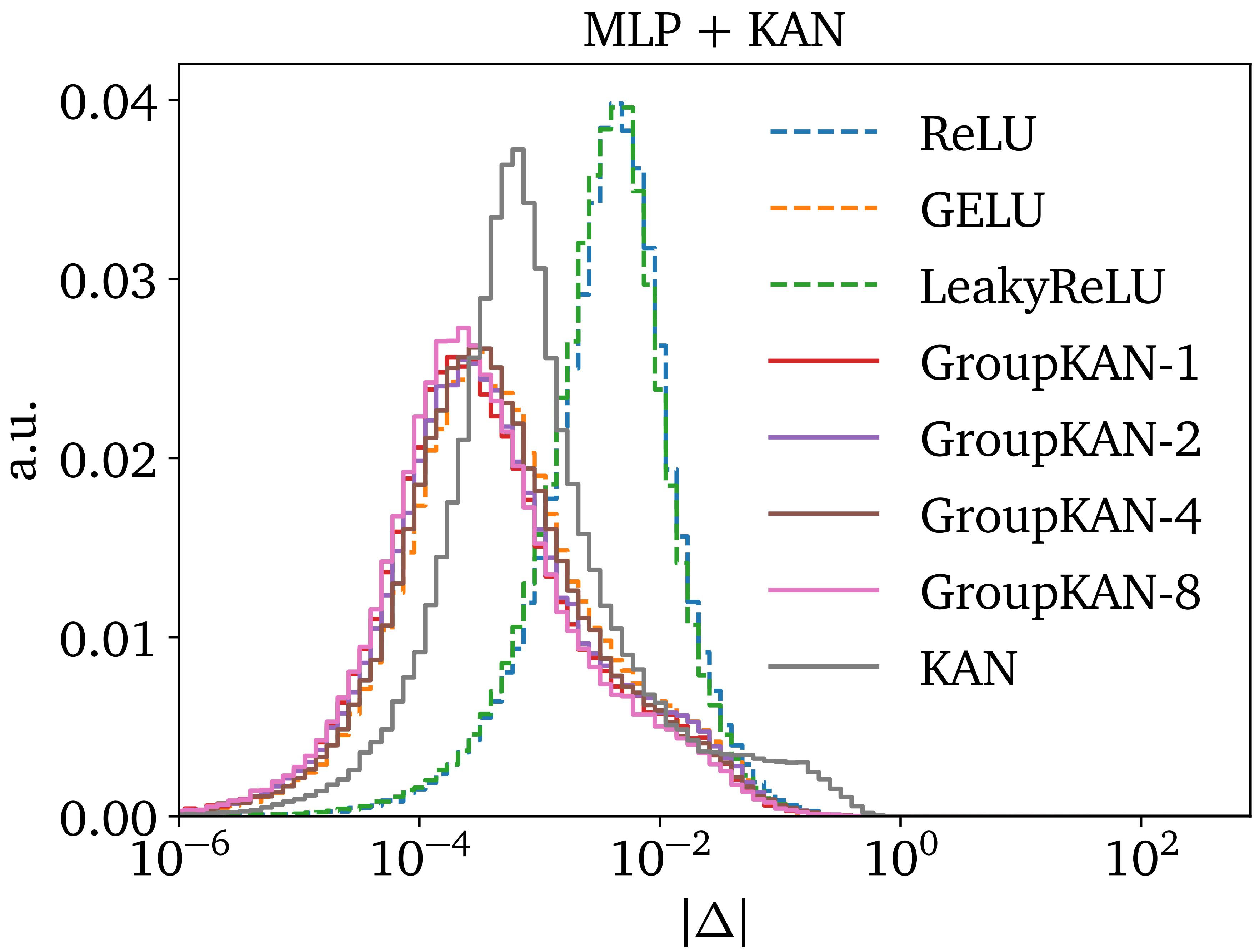
- Calibrate RE by introducing scaling parameter T: $\sigma_\text{syst} \to \sigma_\text{syst} \times T$
- T estimated by using stochastic gradient descent $\mathcal{L}_T(x) = \left\langle \frac{|A_\text{true}(x) - \bar{A}(x)|^2}{2\sigma^2(x)T^2} + \log\sigma(x)T \right\rangle_{x\sim D_\text{train}}$

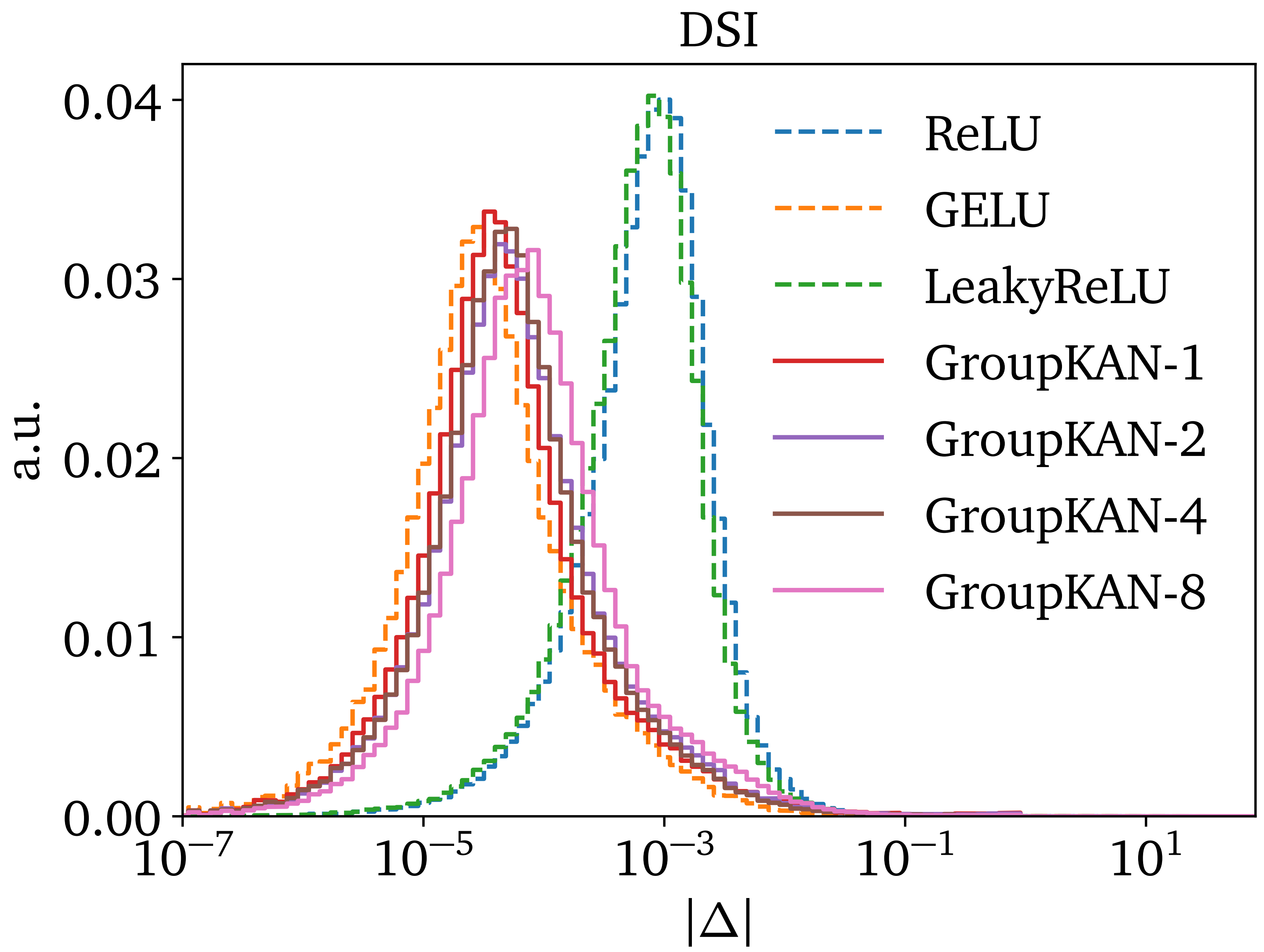




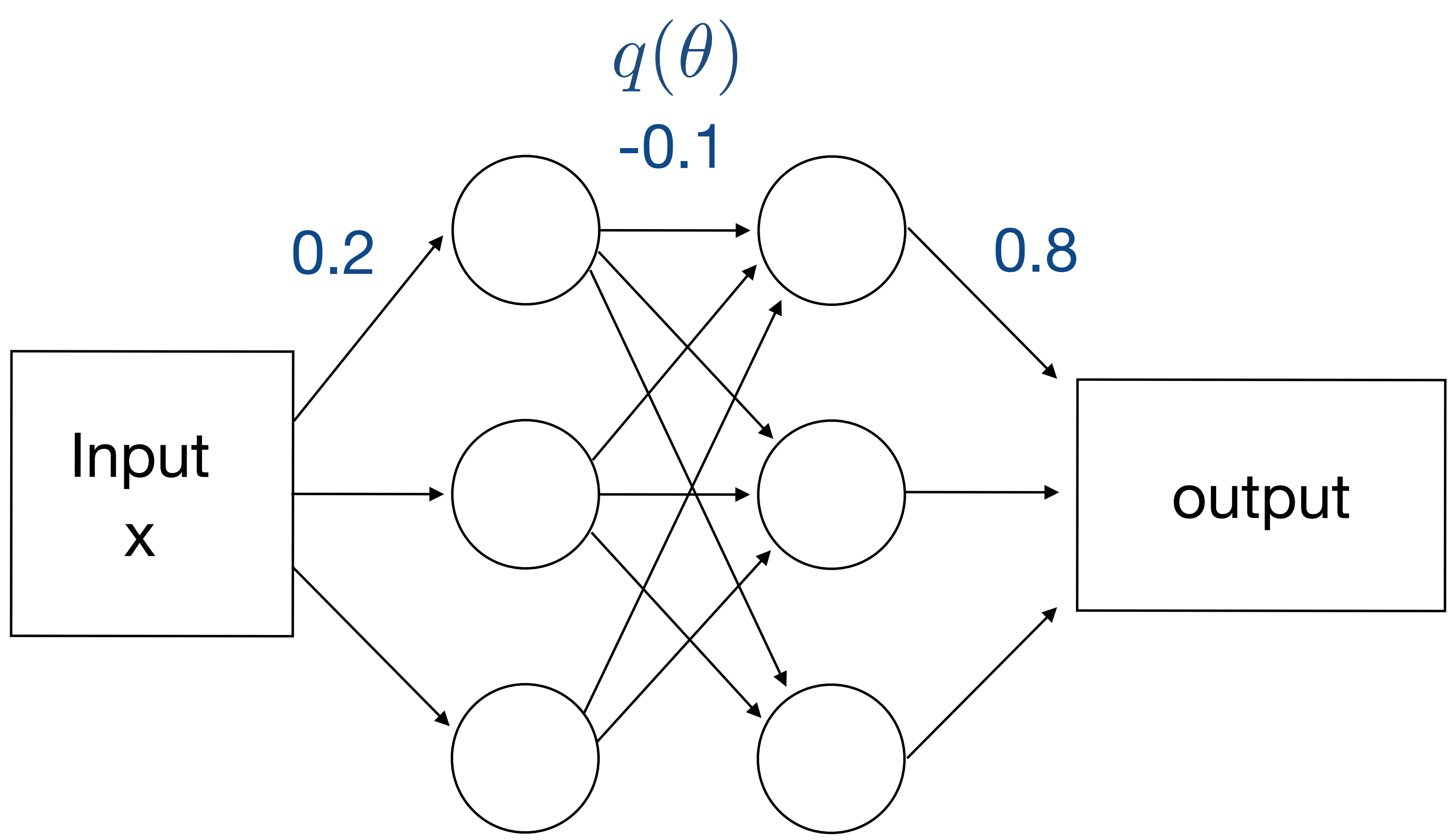
Kolmogorov-Arnold networks (KANs)

- Use KANs for calibration
- GroupKANs allow for learnable activation functions





Deterministic network



- Parameter: network weights $q(\theta)$
- Gaussian uncertainty in loss
- Predicts **mean** and **variance**

$$\mathcal{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma(x_i)^2} + \log \sigma(x_i) + \text{const.}$$