Learning uncertainties for LHC-networks

Nina Elmer

MadGraph5 meeting

arXiv: 2412.12069

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

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

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

- **Reliable interpolation** of two-loop amplitudes [2412.09534]
  - Target accuracy $< 10^{-4}$ [2412.09534]
  - Scaling up to $Z + 5j$ [2411.00446]
- Amplitude surrogates part of MadGraph and MadNIS (besides generative integration)

- **Reliable interpolation** of two-loop amplitudes [2412.09534]
  - Target accuracy $< 10^{-4}$ [2412.09534]
  - Scaling up to $Z + 5j$ [2411.00446]
- Amplitude surrogates part of MadGraph and MadNIS (besides generative integration)
- Improve theory predictions for LHC:

- **Reliable interpolation** of two-loop amplitudes [2412.09534]
  - Target accuracy $< 10^{-4}$ [2412.09534]
  - Scaling up to $Z+5j$ [2411.00446]
- Amplitude surrogates part of MadGraph and MadNIS (besides generative integration)
- Improve theory predictions for LHC:

  1. **Faster precision** simulations
  2. **Control** through calibrated uncertainties

Motivation

- How are learned uncertainties linked to the accuracy of predictions?
- Can they be controlled?

Motivation

- How are learned uncertainties linked to the accuracy of predictions?
- Can they be controlled?
- Two types of uncertainties:

Motivation

- How are learned uncertainties linked to the accuracy of predictions?
- Can they be controlled?
- Two types of uncertainties:
  - **Systematic**: Plateaus for perfect training
  - **Statistical**: Vanishes for perfect training

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 using **variational inference**: $A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A\,|\,x) = \int \mathrm{d}\theta\, \underbrace{q(\theta)}_{\text{network distribution}}\, \underbrace{\bar{A}(x,\theta)}_{\text{network output}}$

Motivation

- Fit set of Amplitudes $A(x)$ with training data: $\{x, A(x)\}$
- Prediction using **variational inference**: $A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A\,|\,x) = \int \mathrm{d}\theta\, \underbrace{q(\theta)}_{\text{network distribution}}\, \underbrace{\bar{A}(x,\theta)}_{\text{network output}}$
- Heteroscedastic loss:

$$\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma_\theta(x_i)^2} + \log \sigma_\theta(x_i) + \ldots$$

Motivation

- Fit set of Amplitudes $A(x)$ with training data: $\{x, A(x)\}$
- Prediction using **variational inference**: $A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A\,|\,x) = \int \mathrm{d}\theta\, \underbrace{q(\theta)}_{\text{network distribution}}\, \underbrace{\bar{A}(x,\theta)}_{\text{network output}}$
- Heteroscedastic loss:

$$\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma_\theta(x_i)^2} + \log \sigma_\theta(x_i) + \dots$$

- Additional statistical uncertainty:

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

Motivation

- Fit set of Amplitudes $A(x)$ with training data: $\{x, A(x)\}$
- Prediction using **variational inference**: $A(x) \equiv \langle A \rangle = \int \mathrm{d}A \, A \, p(A \,|\, x) = \int \mathrm{d}\theta \, q(\theta) \, \bar{A}(x, \theta)$

network distribution

network output

- Heteroscedastic loss:

$$\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma_\theta(x_i)^2} + \log \sigma_\theta(x_i) + \ldots$$

- Additional statistical uncertainty:

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

Motivation

- Fit set of Amplitudes $A(x)$ with training data: $\{x, A(x)\}$
- Prediction using **variational inference**: $A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A\,|\,x) = \int \mathrm{d}\theta\, q(\theta)\, \bar{A}(x,\theta)$

network distribution

network output

- Heteroscedastic loss:

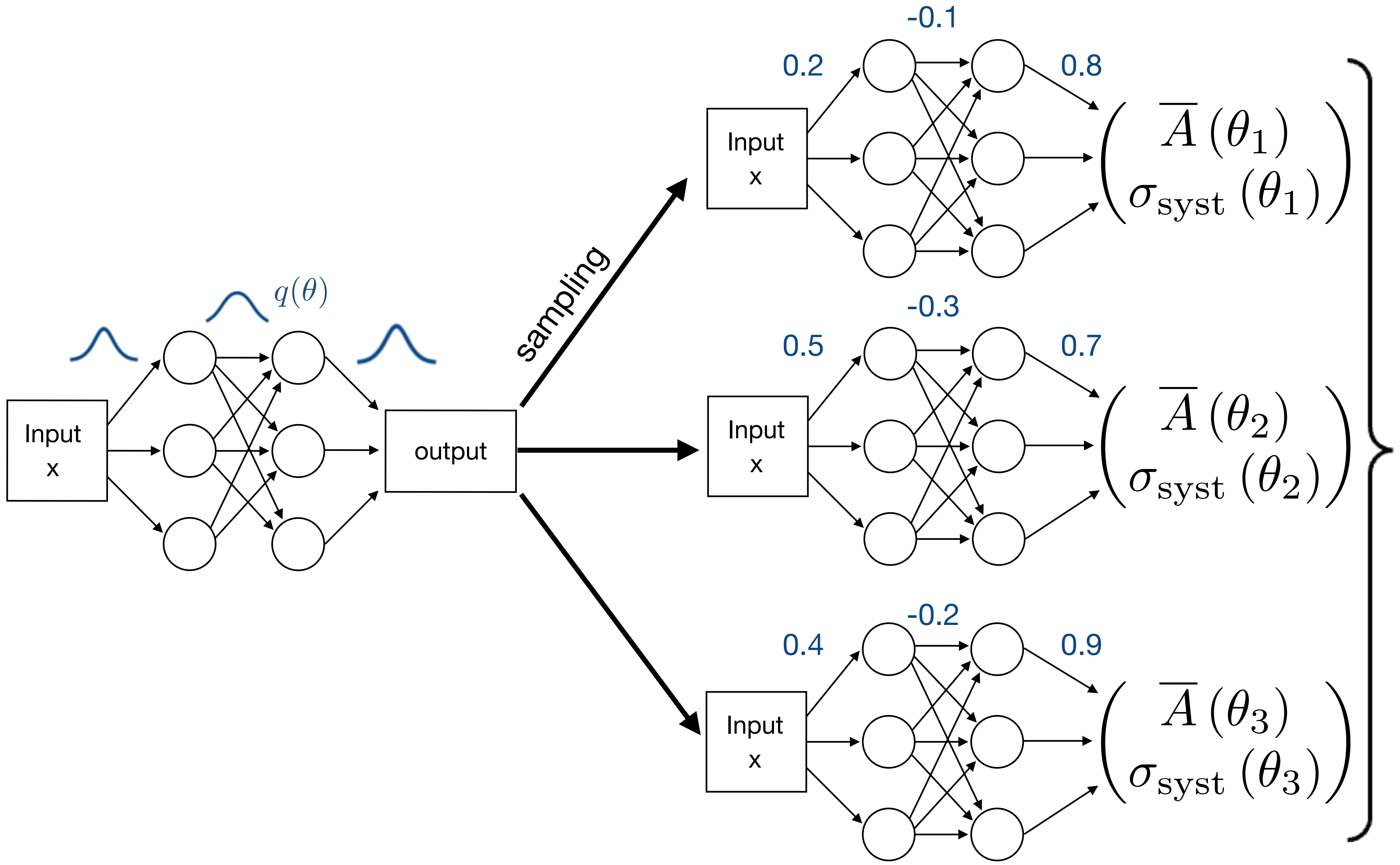
$$\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|\,f(x_i) - f_\theta(x_i)\,|^2}{2\sigma_\theta(x_i)^2} + \log \sigma_\theta(x_i) + \ldots$$

- Additional statistical uncertainty:

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

Bayesian neural networks (BNNs)

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



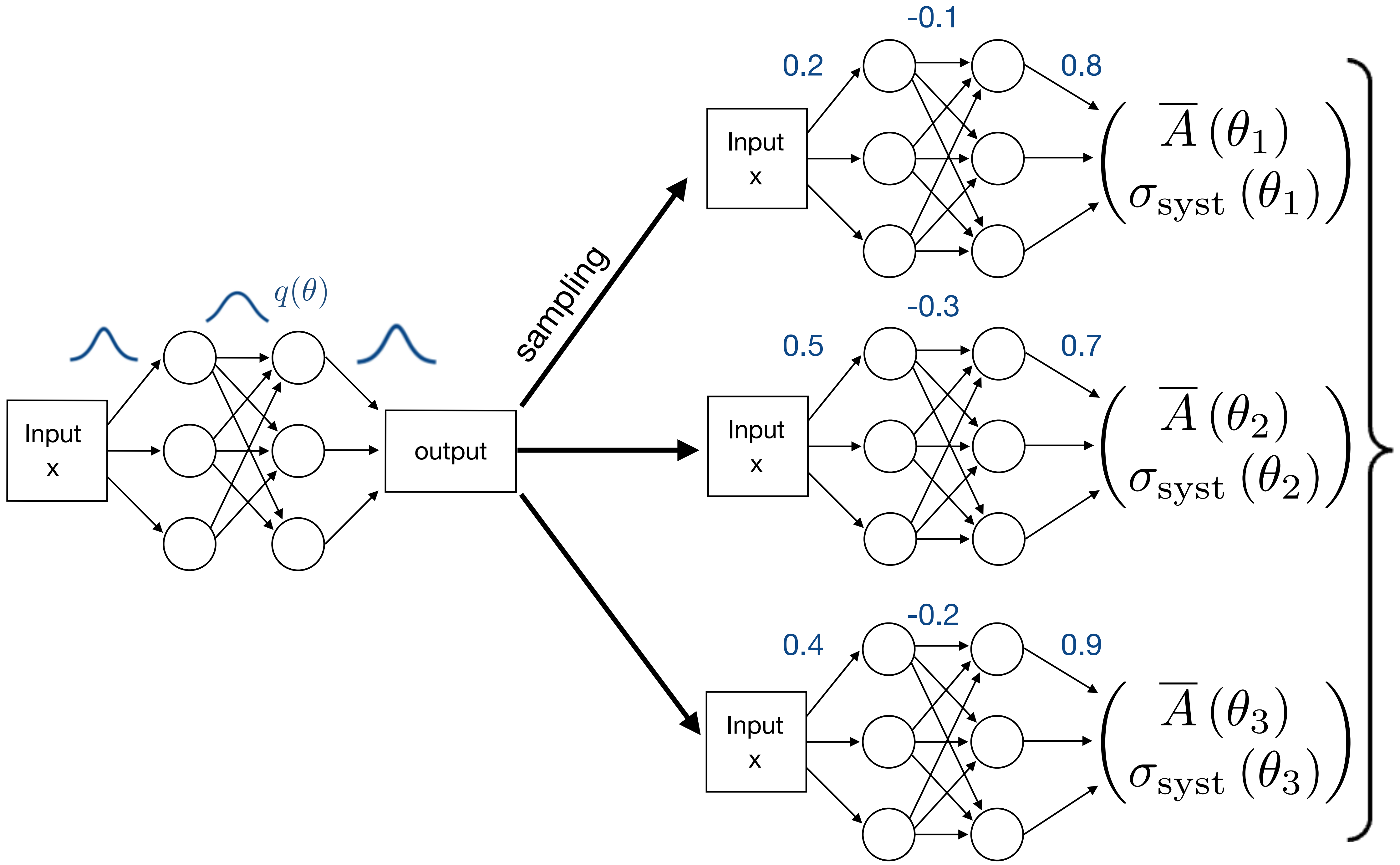
$$\langle A \rangle = \frac{1}{N} \sum_i^N \overline{A}(\theta_i)$$

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

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

Bayesian neural networks (BNNs)

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



$q(\theta)$
Input x
output
sampling
0.2
-0.1
0.8
0.5
-0.3
0.7
0.4
-0.2
0.9
Input x
Input x
Input x
$\begin{pmatrix} \overline{A}(\theta_1) \\ \sigma_{\text{syst}}(\theta_1) \end{pmatrix}$
$\begin{pmatrix} \overline{A}(\theta_2) \\ \sigma_{\text{syst}}(\theta_2) \end{pmatrix}$
$\begin{pmatrix} \overline{A}(\theta_3) \\ \sigma_{\text{syst}}(\theta_3) \end{pmatrix}$

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

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

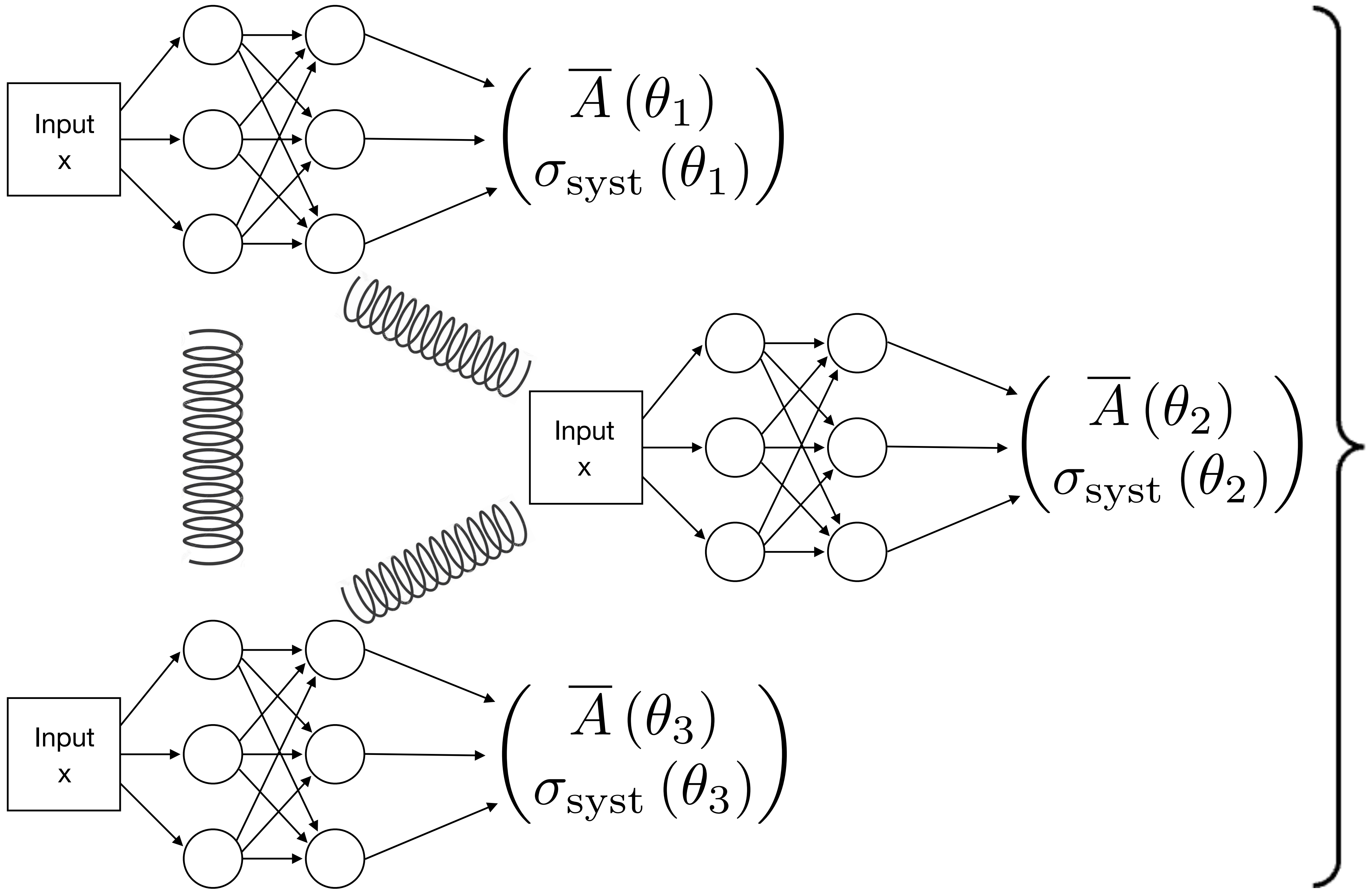
$$\sigma^2_{\text{stat}} = \frac{1}{N} \sum_i^N \left(\langle A \rangle - \overline{A}(\theta_i)\right)^2$$

- Parameters: **Network weights** $q(\theta)$
- $q(\theta)$: Params of a Gaussian
- Ensemble: **Sample** $q(\theta)$
- Used for ATLAS topocluster calibration [2412.04370]

Repulsive ensembles (REs)

Ensemble of networks

Output


Input x
Input x
Input x
$\begin{pmatrix} \overline{A}(\theta_1) \\ \sigma_{\text{syst}}(\theta_1) \end{pmatrix}$
$\begin{pmatrix} \overline{A}(\theta_2) \\ \sigma_{\text{syst}}(\theta_2) \end{pmatrix}$
$\begin{pmatrix} \overline{A}(\theta_3) \\ \sigma_{\text{syst}}(\theta_3) \end{pmatrix}$

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

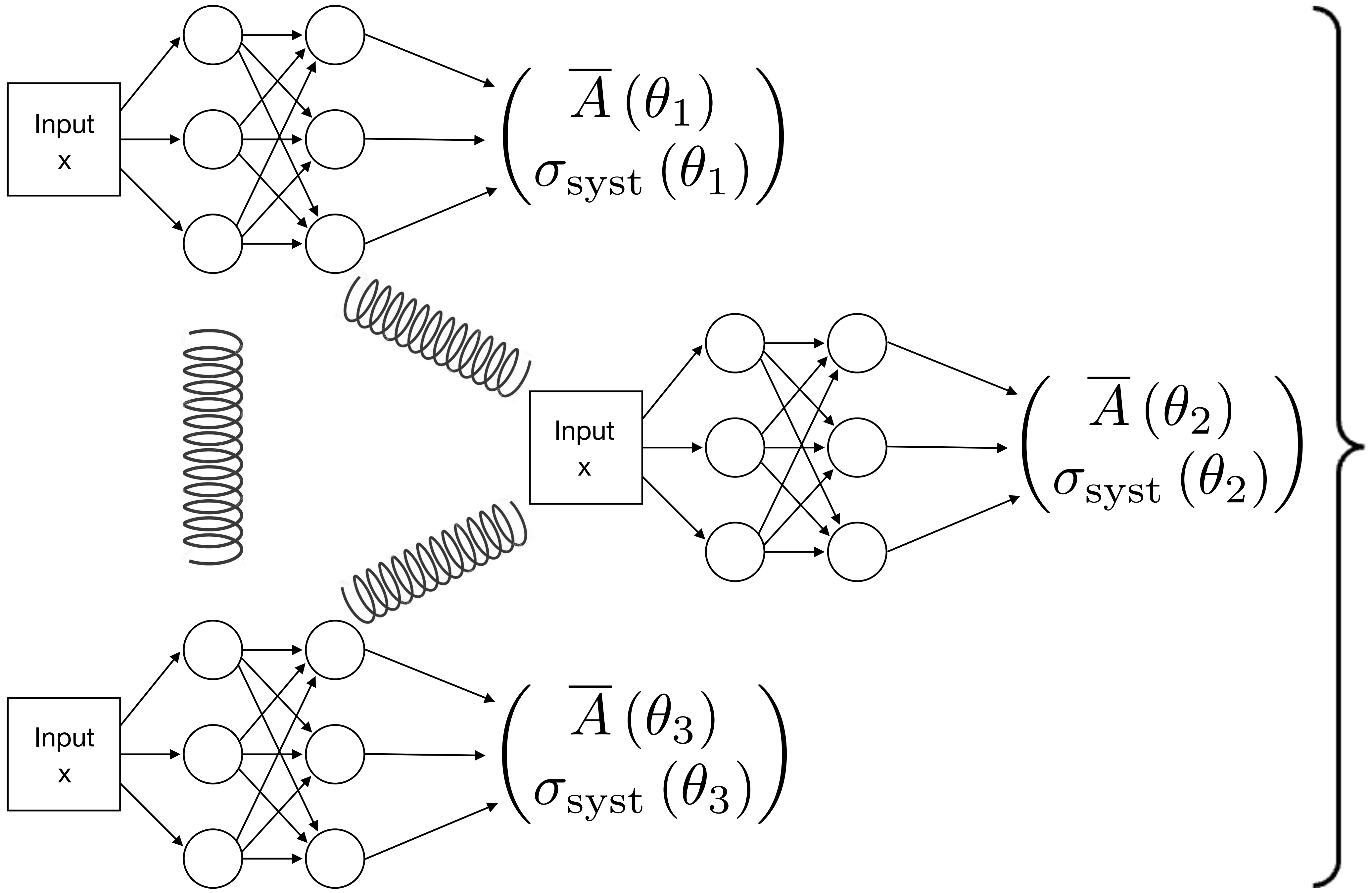
$$\sigma^2_{\text{syst}} = \frac{1}{N} \sum_i^N \sigma^2_{\text{syst}}(\theta_i)$$

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

Repulsive ensembles (REs)

Ensemble of networks

Output


Input x
$\begin{pmatrix} \overline{A}(\theta_1) \\ \sigma_{\text{syst}}(\theta_1) \end{pmatrix}$
Input x
$\begin{pmatrix} \overline{A}(\theta_2) \\ \sigma_{\text{syst}}(\theta_2) \end{pmatrix}$
Input x
$\begin{pmatrix} \overline{A}(\theta_3) \\ \sigma_{\text{syst}}(\theta_3) \end{pmatrix}$

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

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

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

- Repulsive term:
  Cover **full posterior** distribution
- Members **trained simultaneously**

Adding Gaussian noise

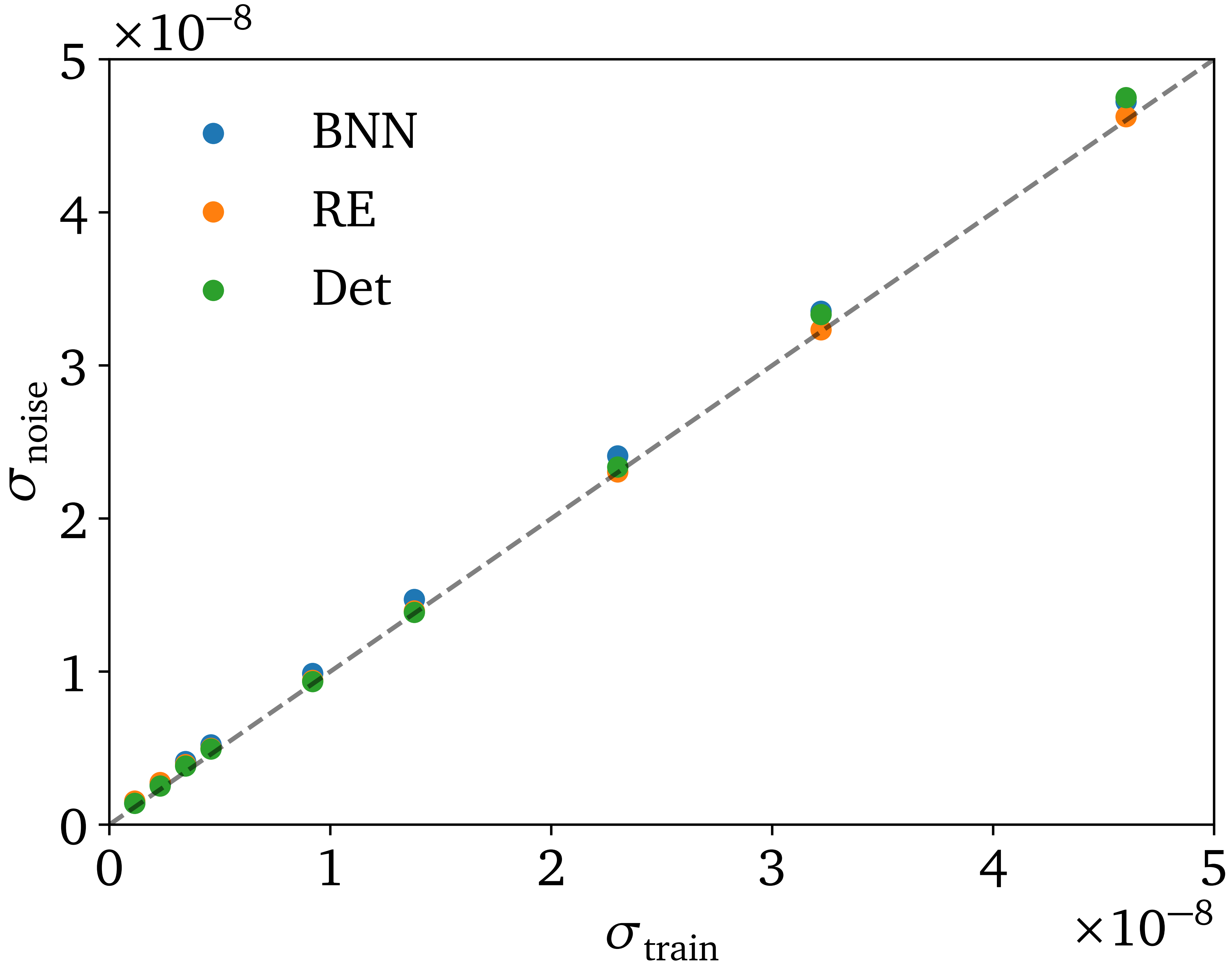
$$\sigma^2_{\mathrm{tot}} = \sigma^2_{\mathrm{syst},0} + \sigma^2_{\mathrm{noise}} + \sigma^2_{\mathrm{stat}}$$

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

Adding Gaussian noise

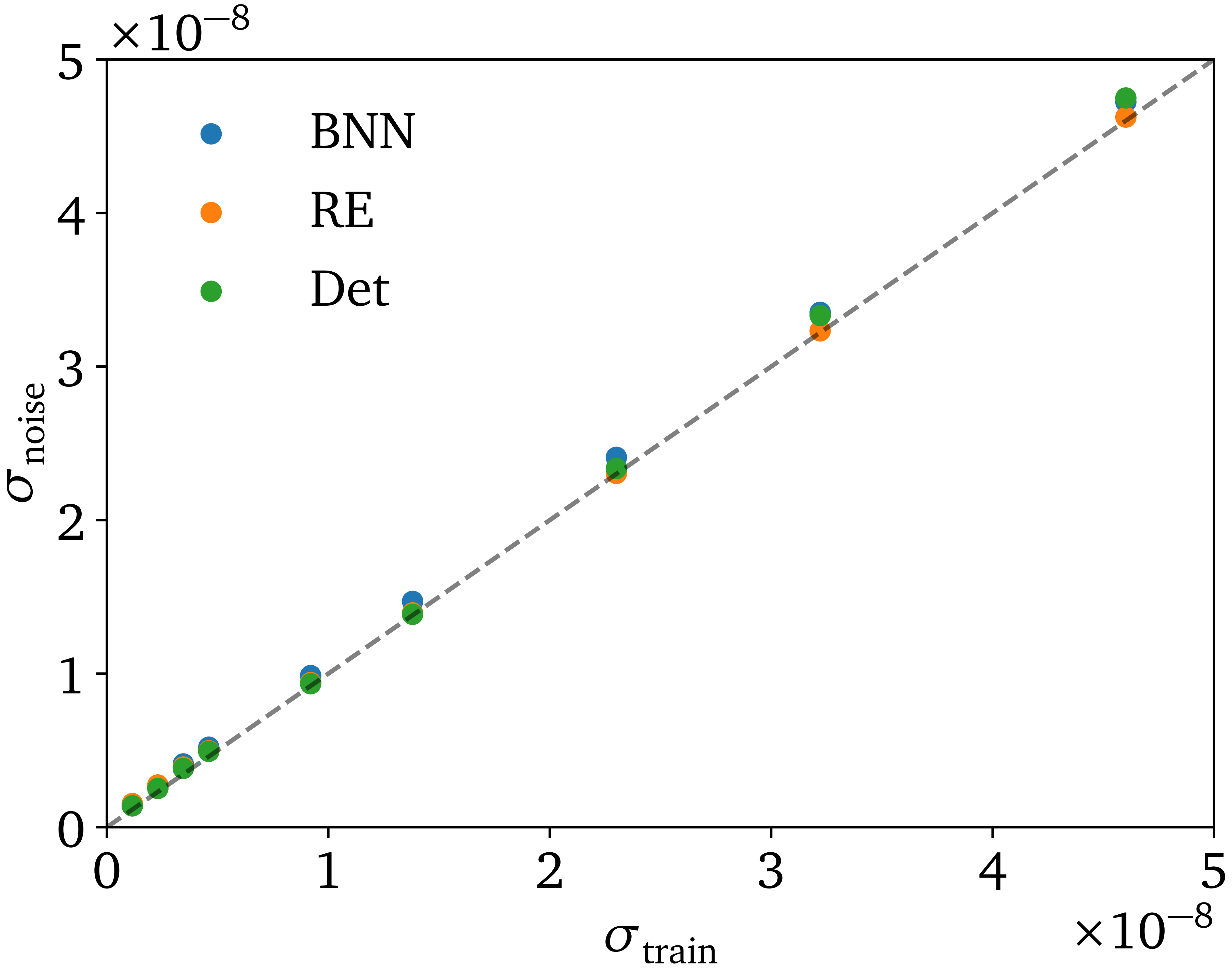
$$\sigma_{\text{tot}}^2 = \sigma_{\text{syst},0}^2 + \sigma_{\text{noise}}^2 + \sigma_{\text{stat}}^2$$

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

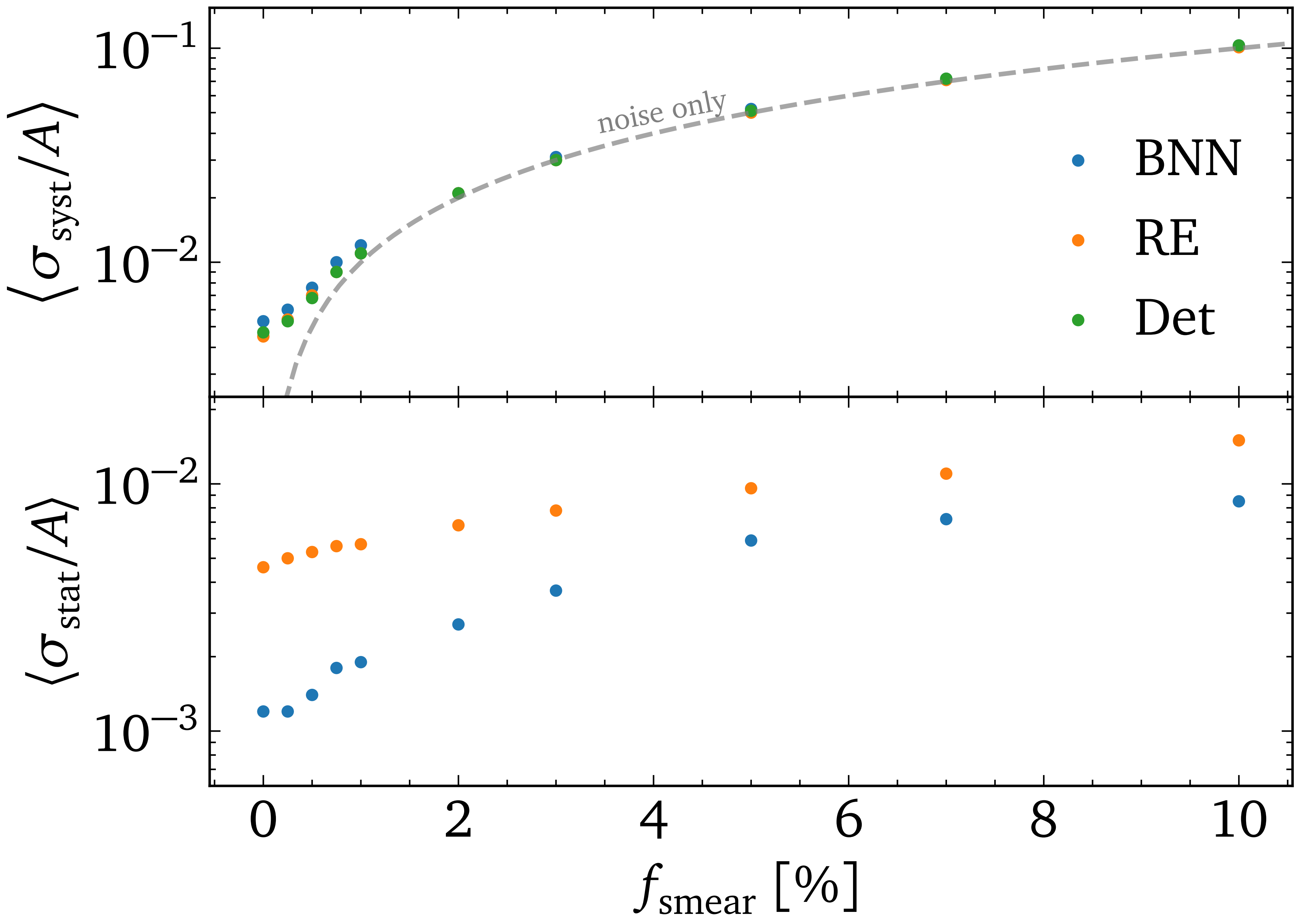


Adding Gaussian noise

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



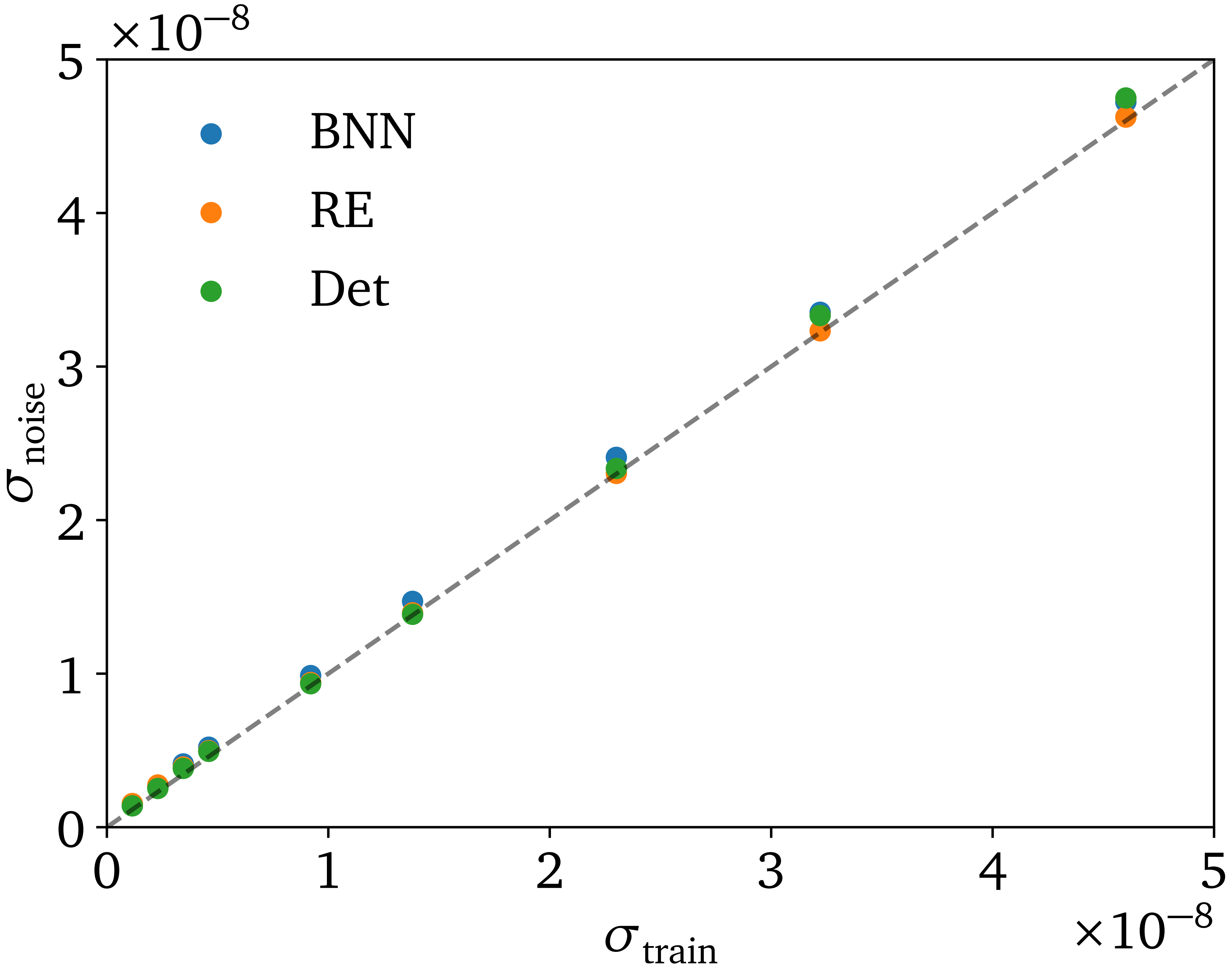
$$\sigma_{\mathrm{train}} = f_{\mathrm{smear}}\, A_{\mathrm{true}}$$

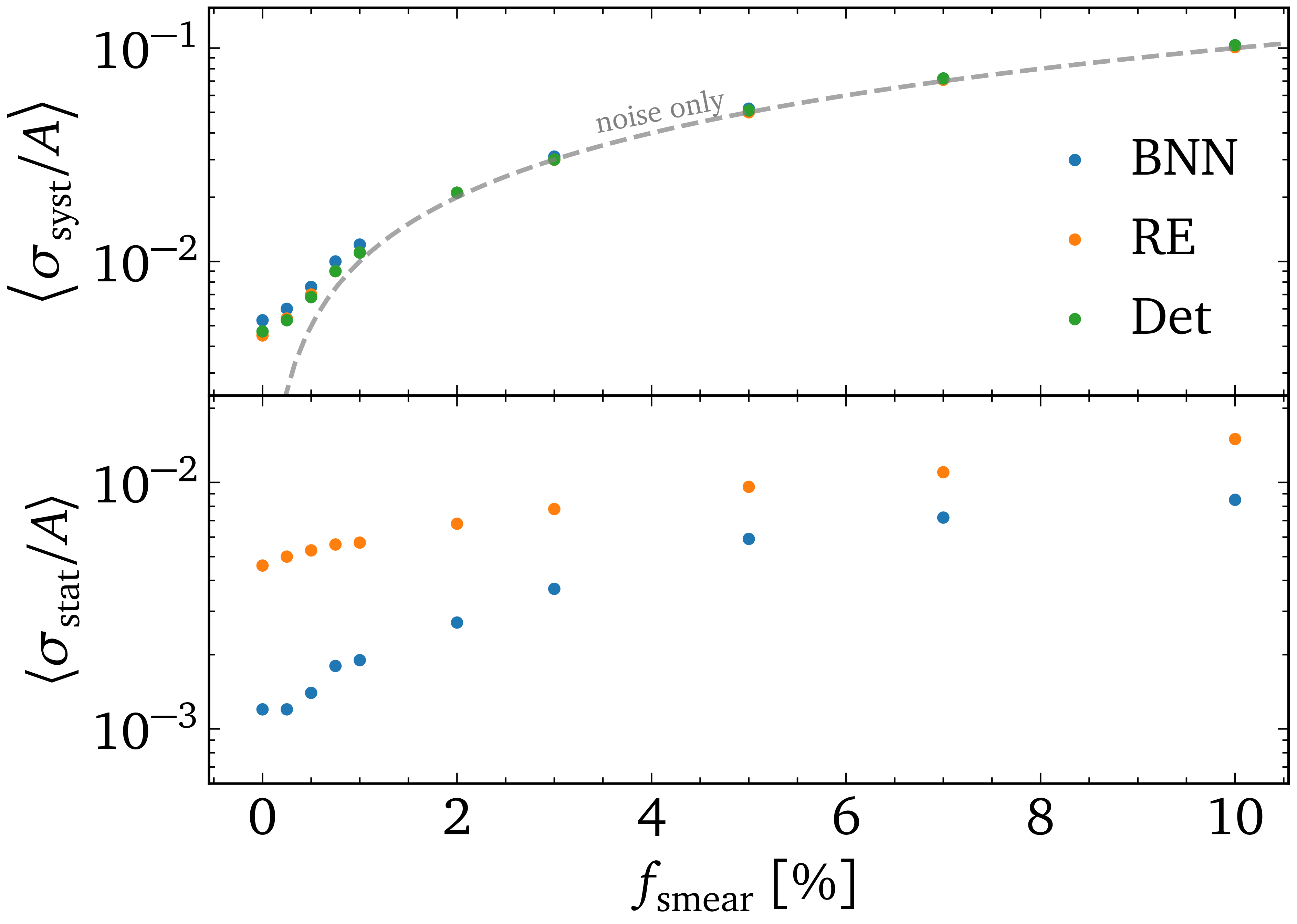


Adding Gaussian noise

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

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

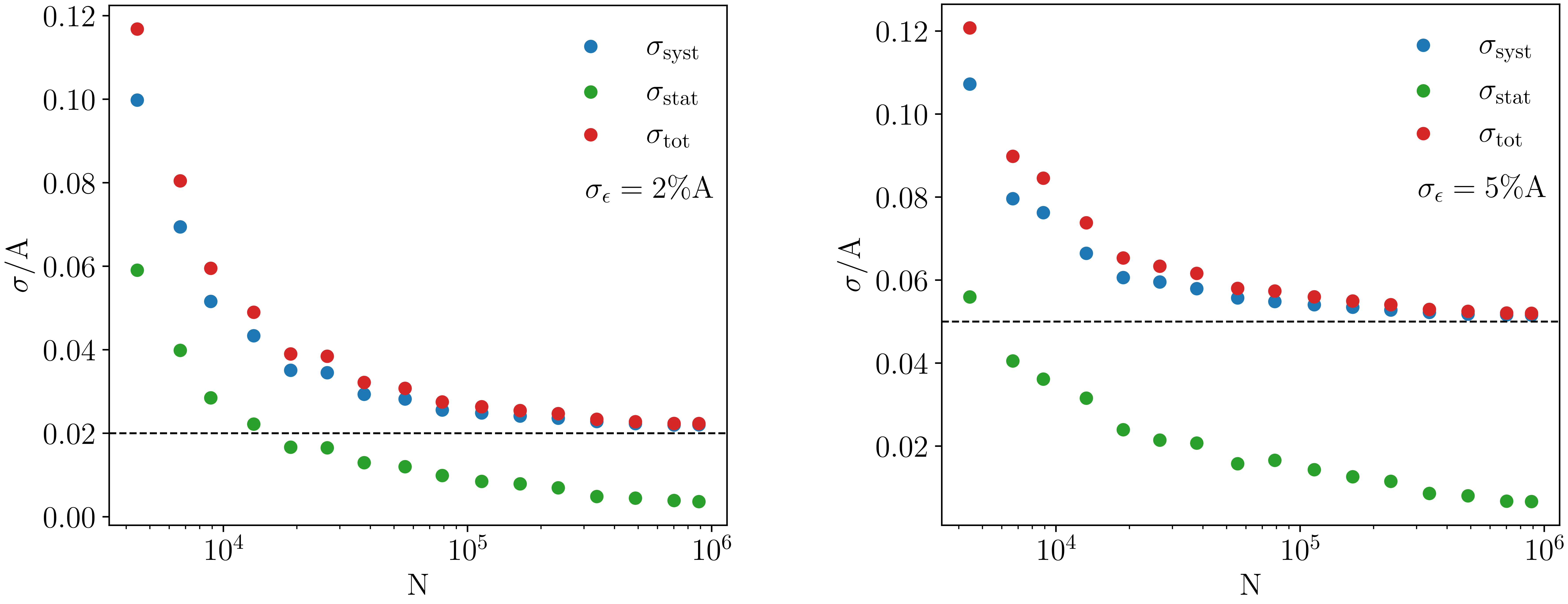




➡ Networks learn noise as **systematic** uncertainty

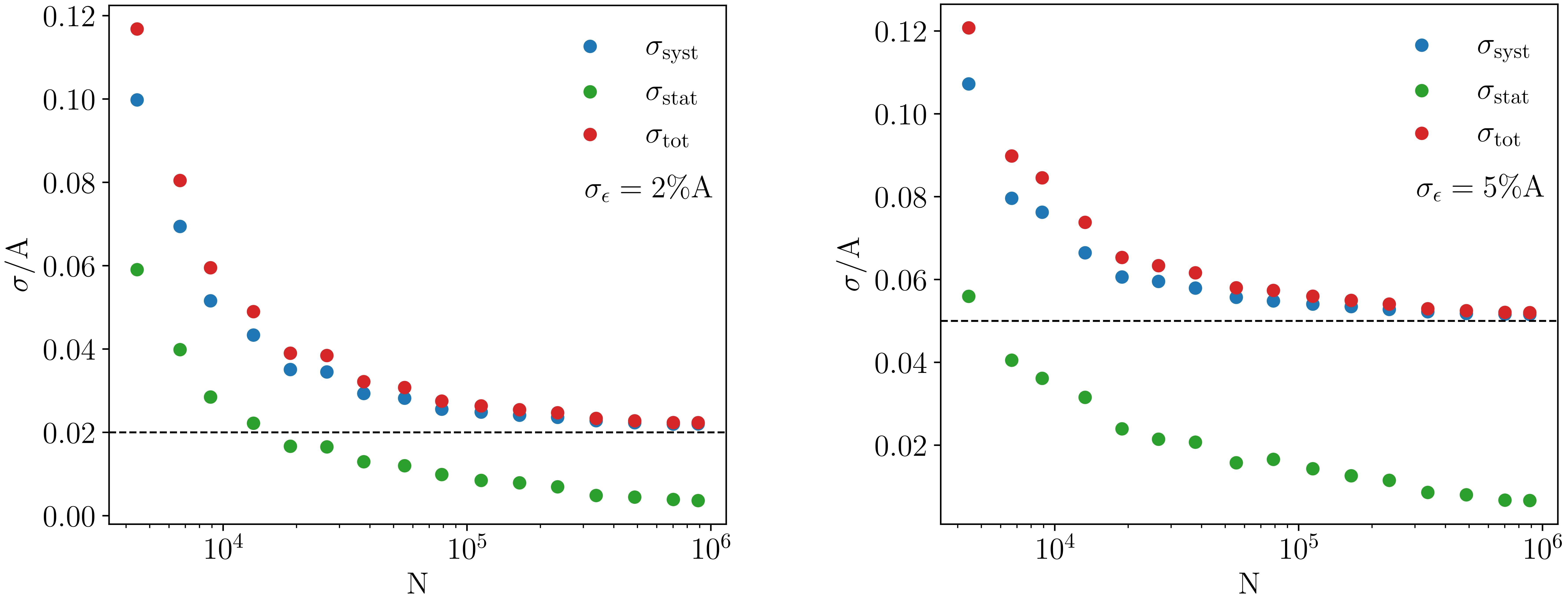
➡**Intrinsic uncertainty** of 0.4%

Uncertainty behavior



BNN only

Uncertainty behavior



BNN only

1. Statistical uncertainty **independent** of noise

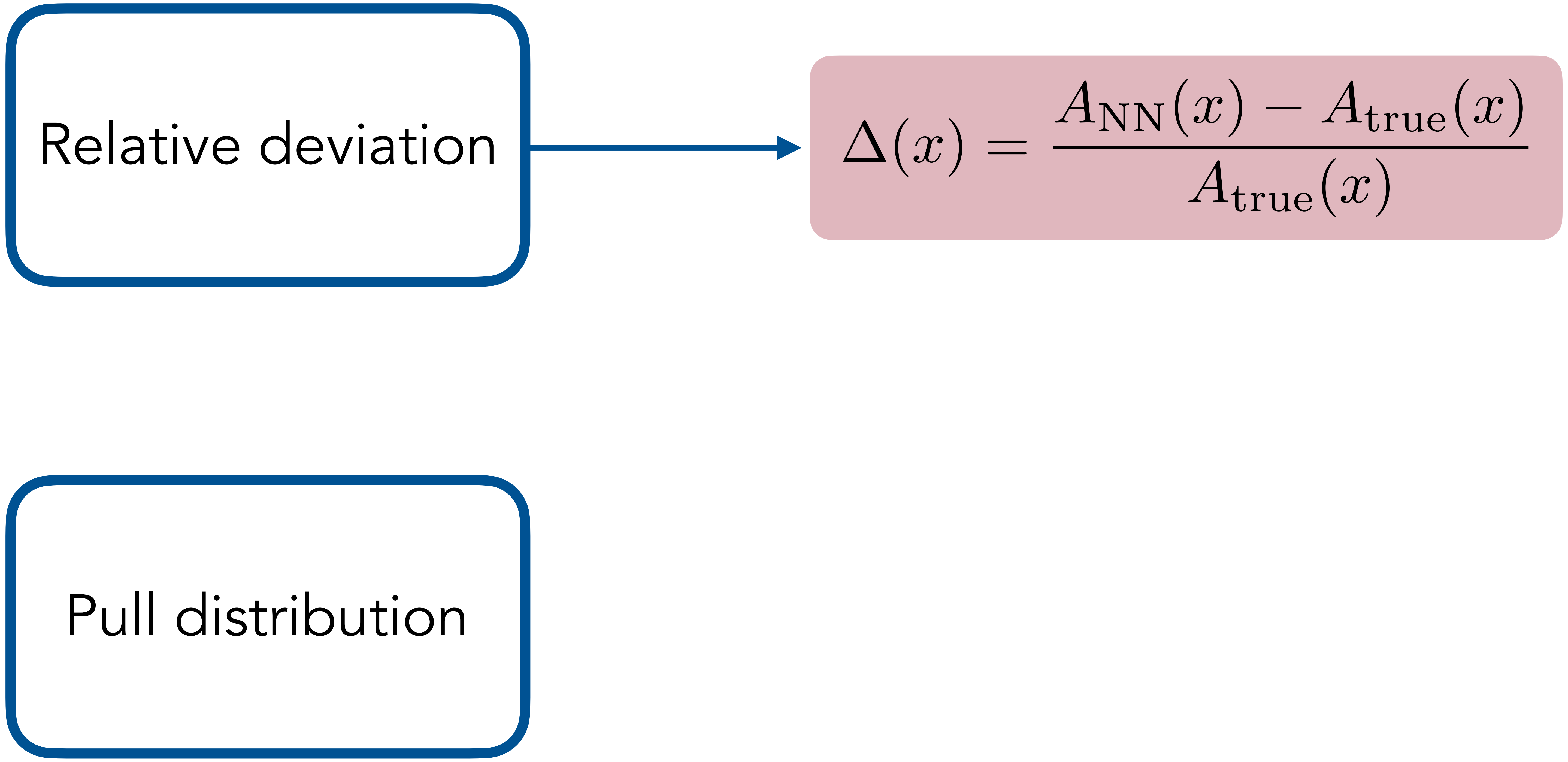
2. Systematic uncertainty **plateaus** on noise level

Calibration of results

Relative deviation

Pull distribution

Calibration of results

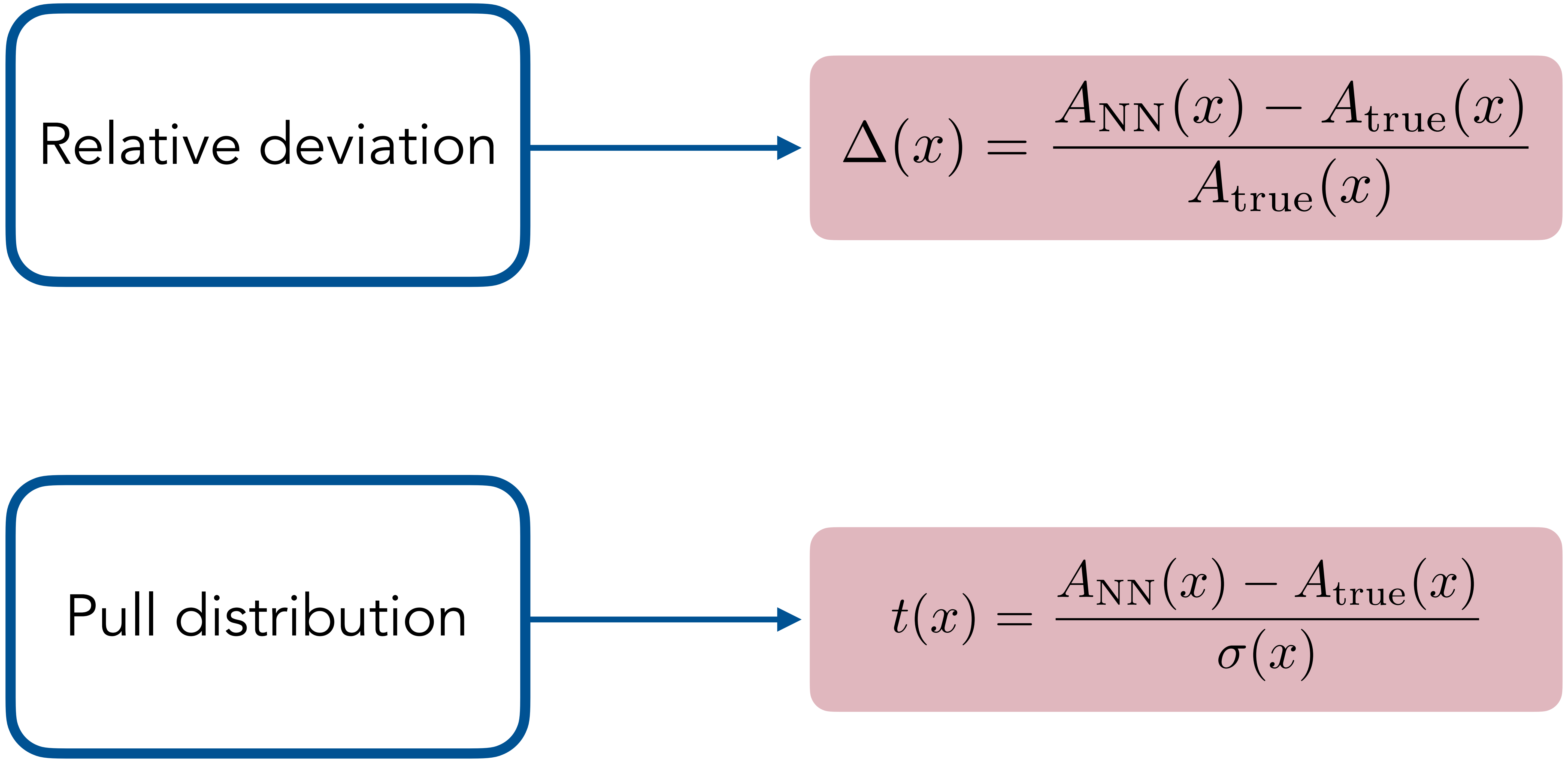


Relative deviation

$$\Delta(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{A_{\text{true}}(x)}$$

Pull distribution

Calibration of results



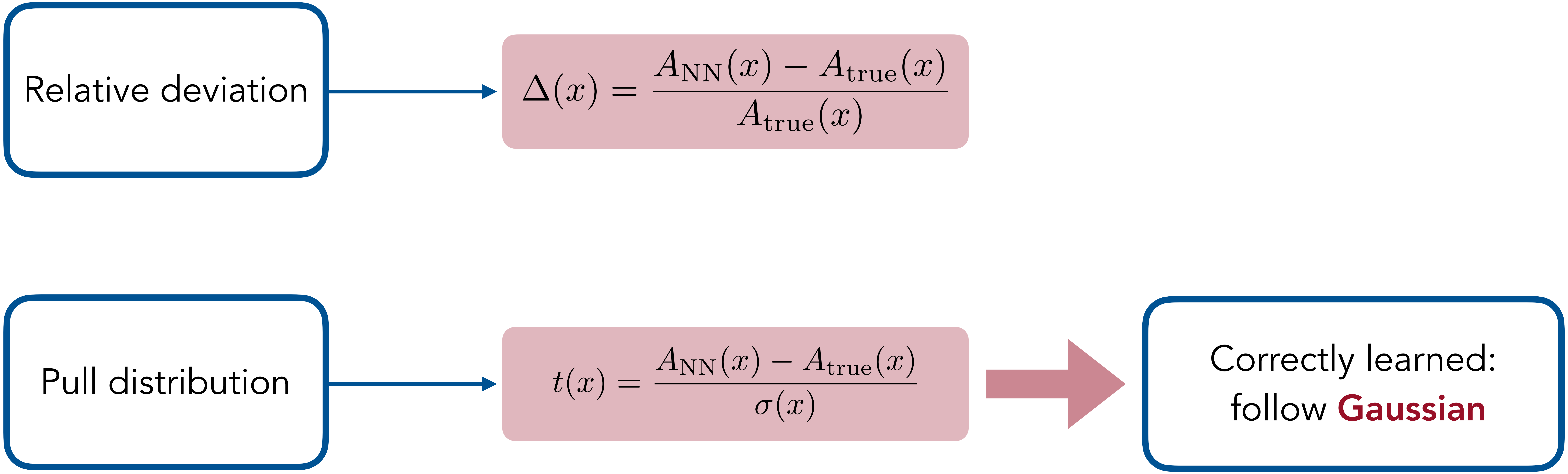
Relative deviation

$$\Delta(x) = \frac{A_{\mathrm{NN}}(x) - A_{\mathrm{true}}(x)}{A_{\mathrm{true}}(x)}$$

Pull distribution

$$t(x) = \frac{A_{\mathrm{NN}}(x) - A_{\mathrm{true}}(x)}{\sigma(x)}$$

Calibration of results



Relative deviation

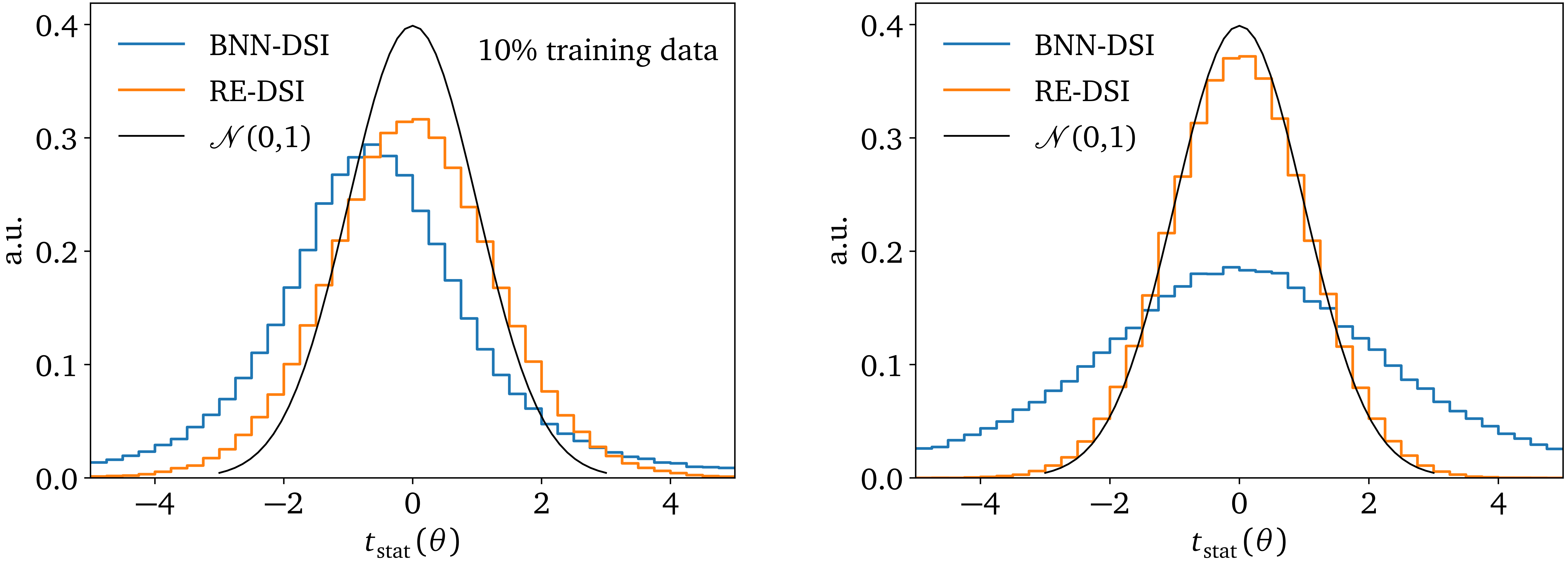
$$\Delta(x) = \frac{A_{\mathrm{NN}}(x) - A_{\mathrm{true}}(x)}{A_{\mathrm{true}}(x)}$$

Pull distribution

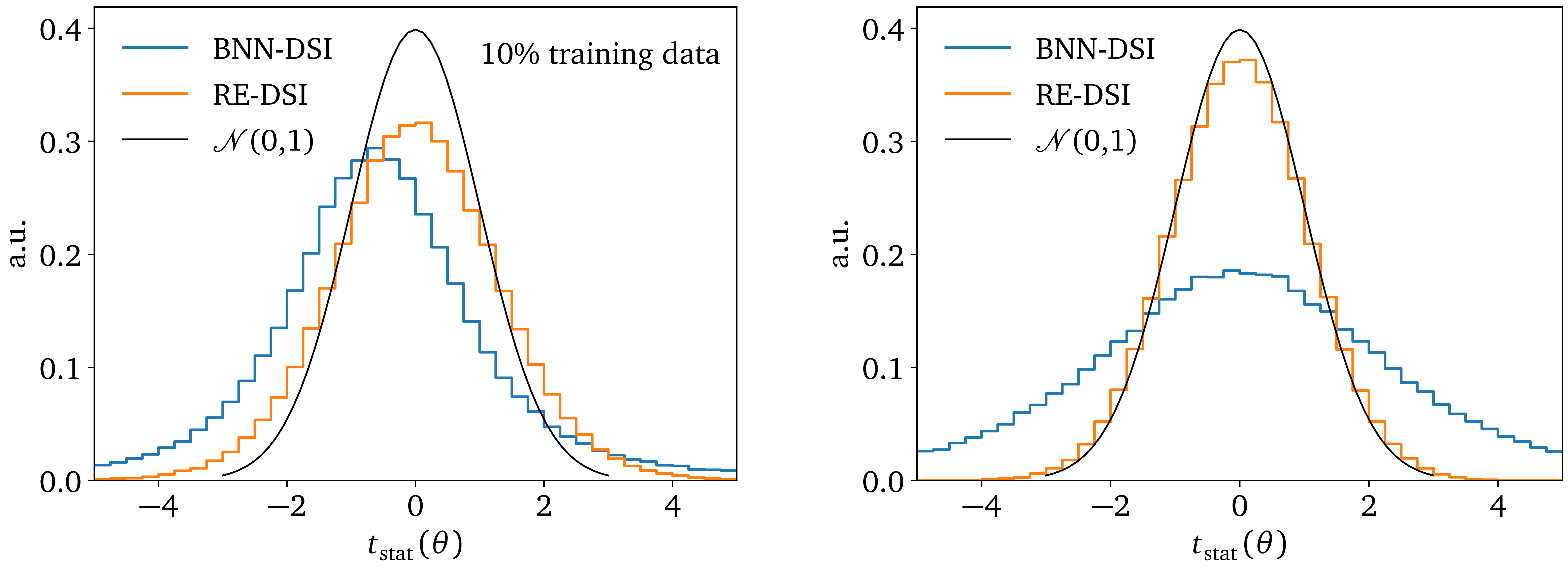
$$t(x) = \frac{A_{\mathrm{NN}}(x) - A_{\mathrm{true}}(x)}{\sigma(x)}$$

Correctly learned: follow **Gaussian**

Statistical pull

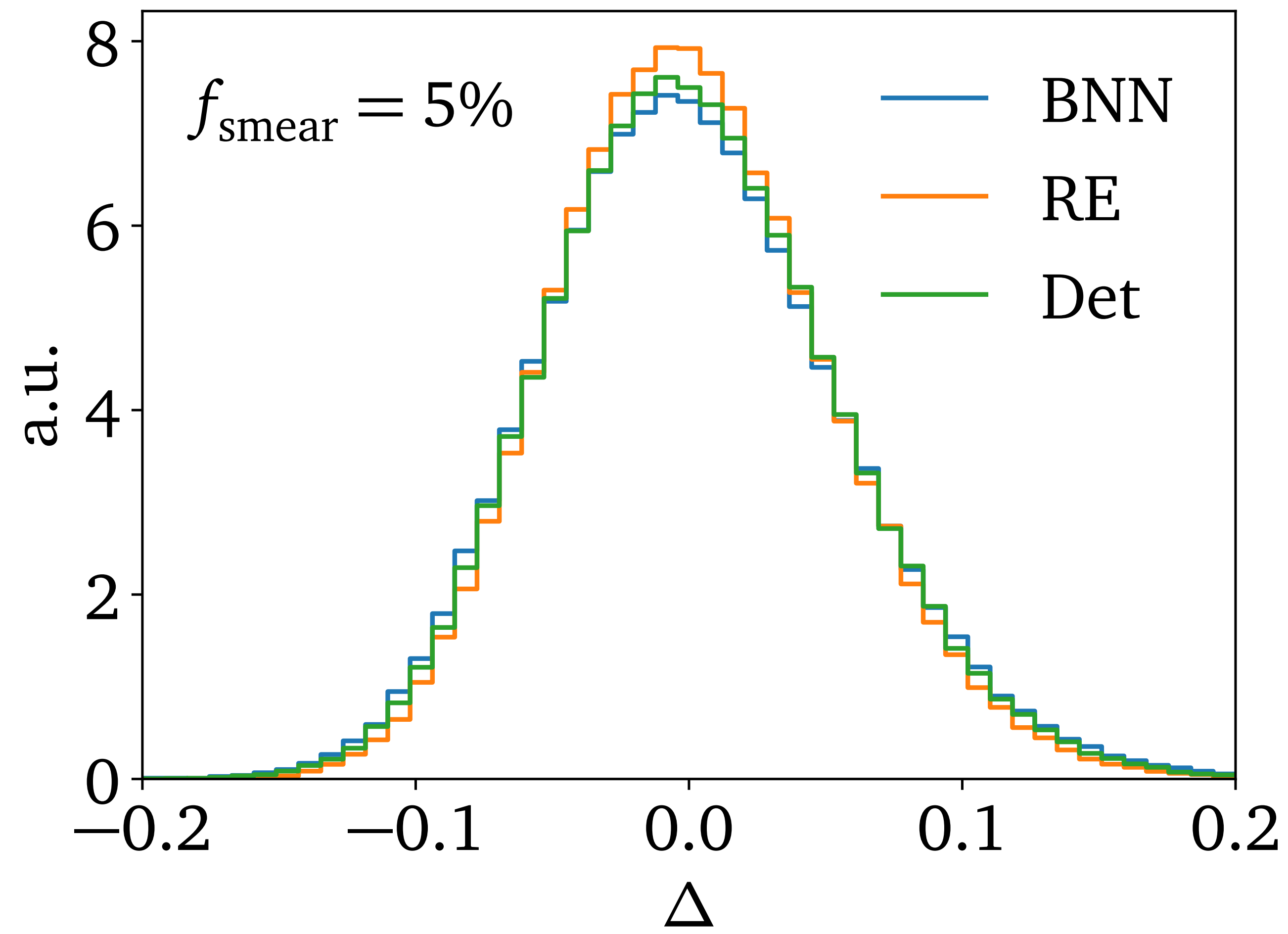


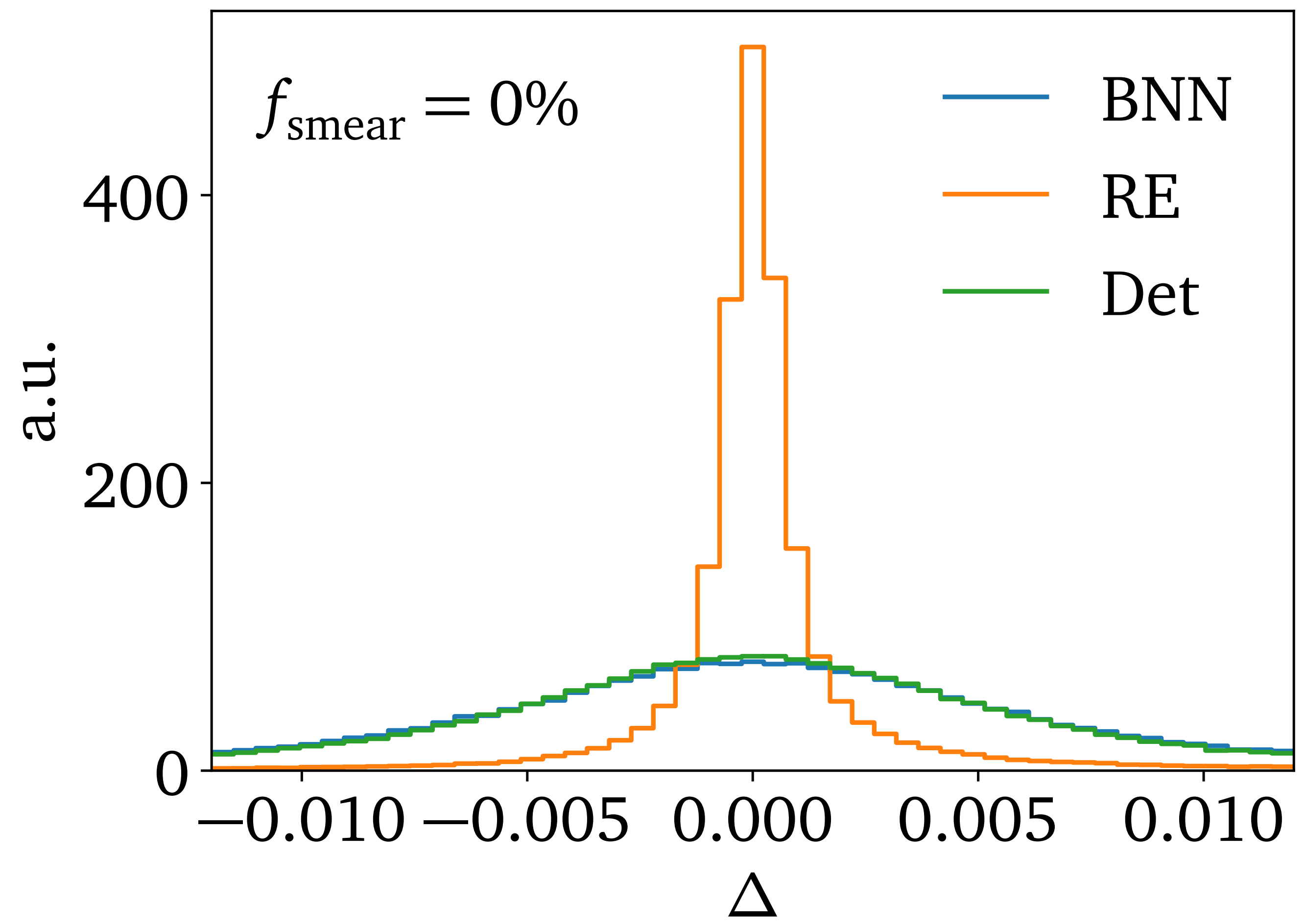
Statistical pull



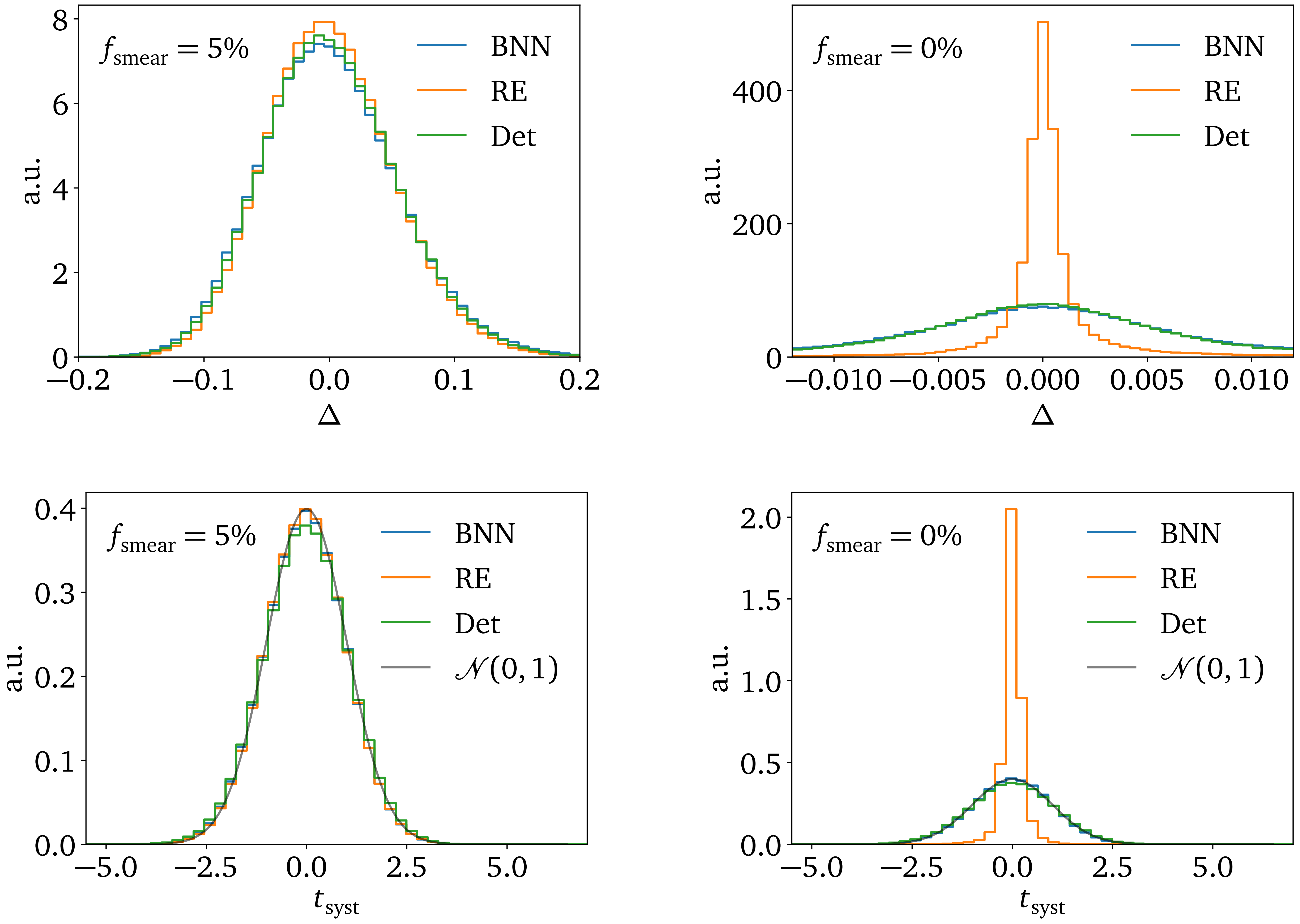
➡Repulsive ensemble: Advantage for **statistical** uncertainty

Systematic pull and accuracy

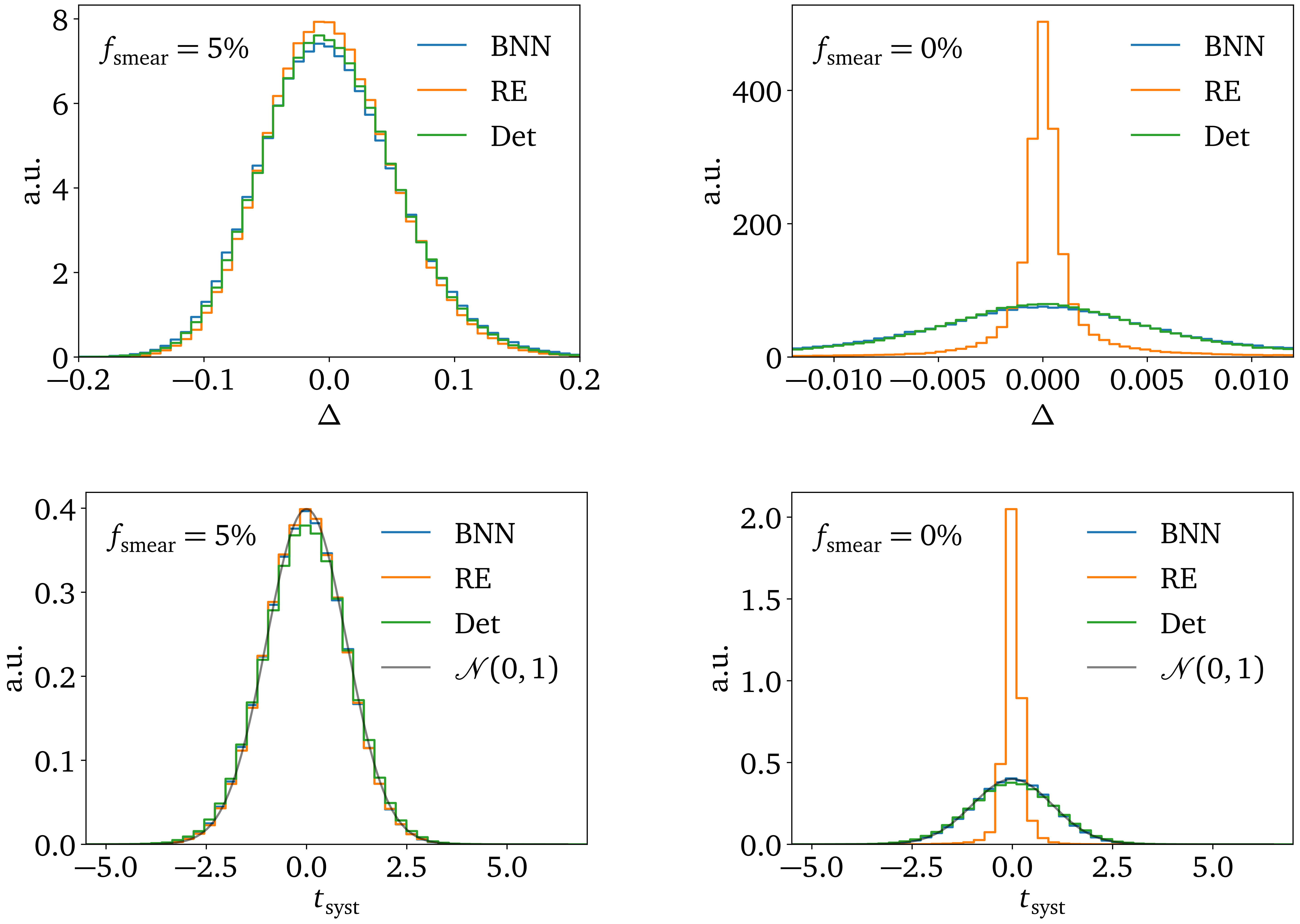




Systematic pull and accuracy



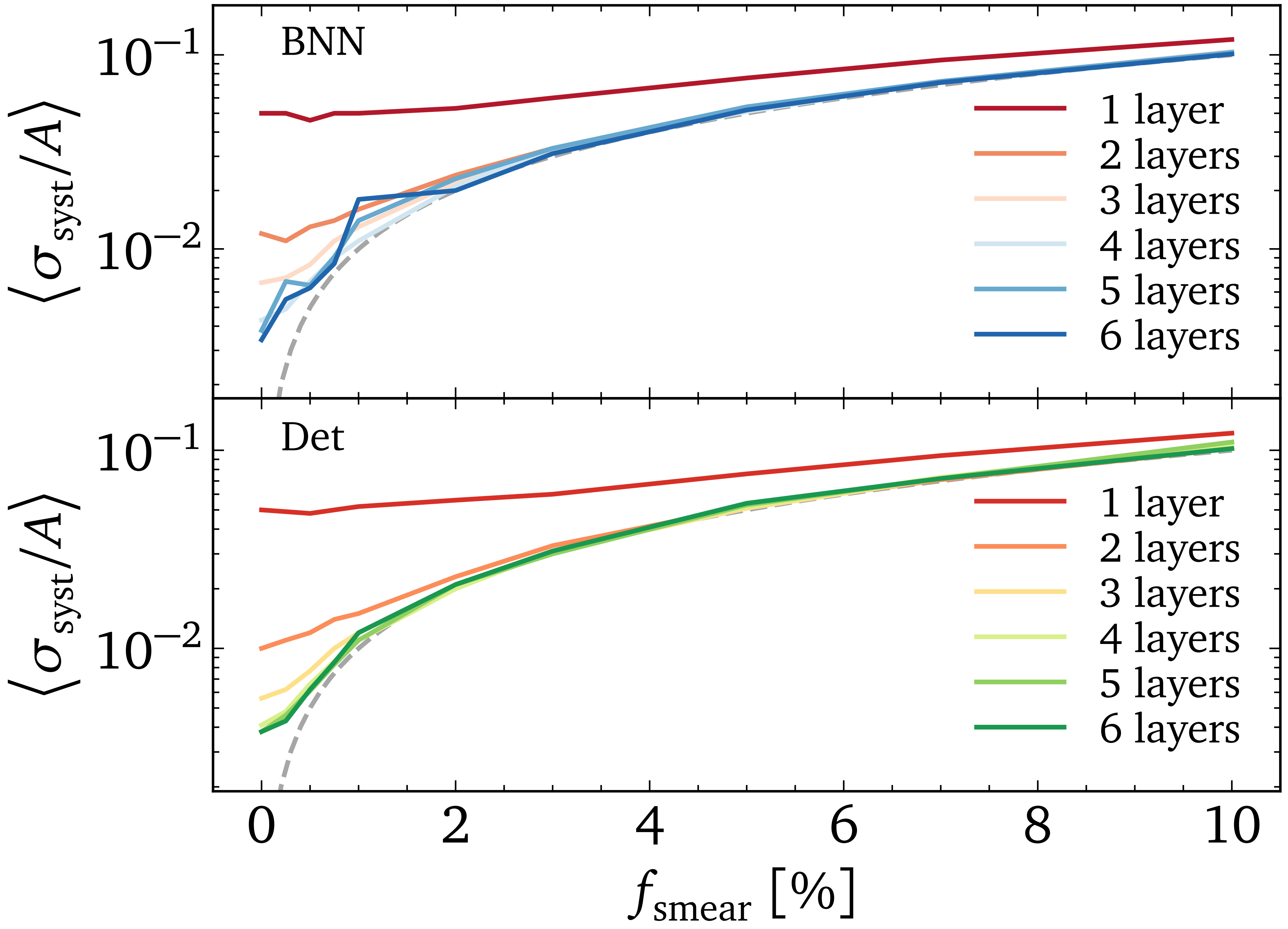
Systematic pull and accuracy



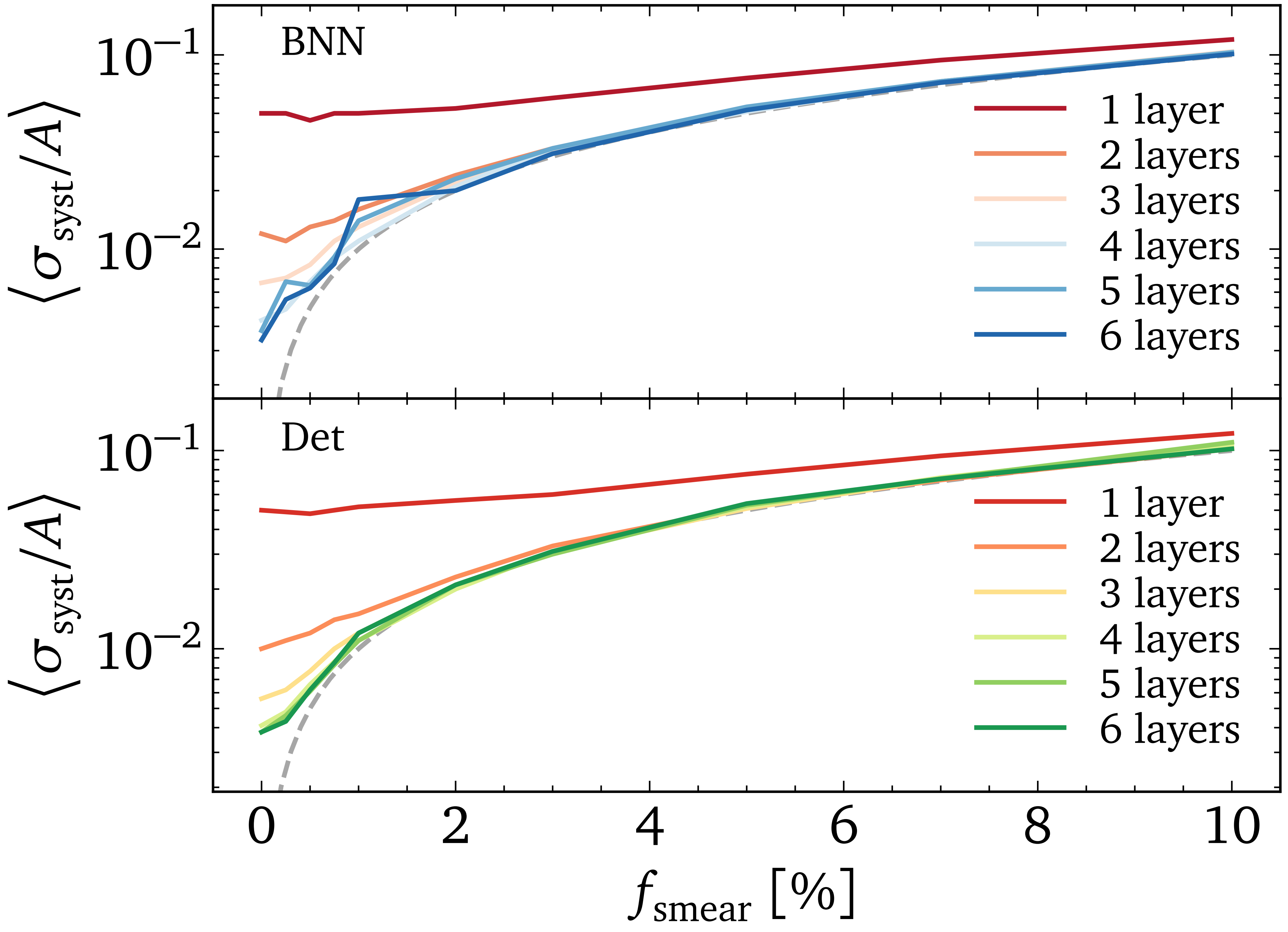
➡ Calibrated results

➡ BNN: Advantage for learning **systematics**

Improve network expressivity

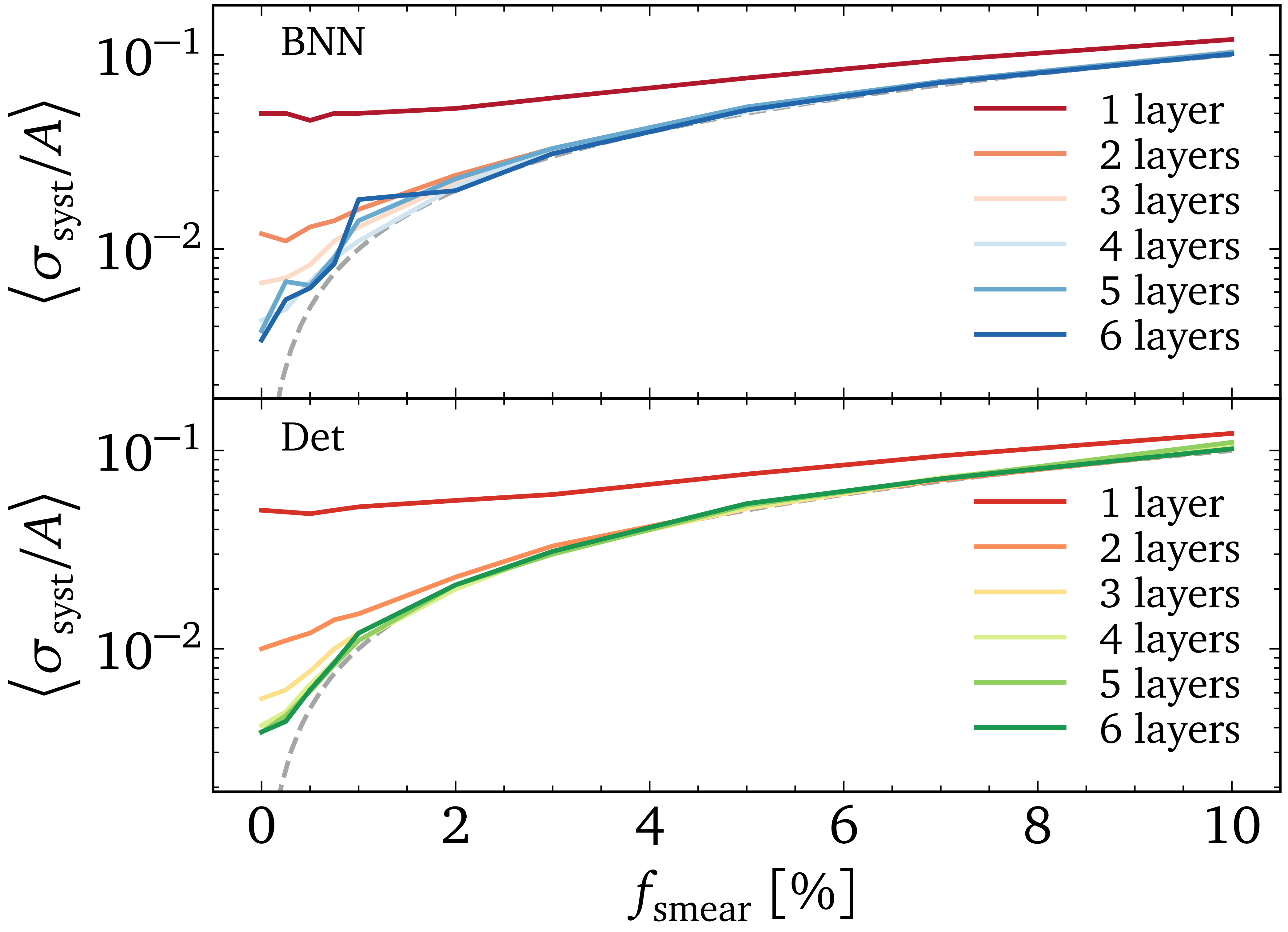


Improve network expressivity



- Source of intrinsic uncertainty?
- BNN: Only last layer Bayesian

Improve network expressivity



- Source of intrinsic uncertainty?
- BNN: Only last layer Bayesian

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

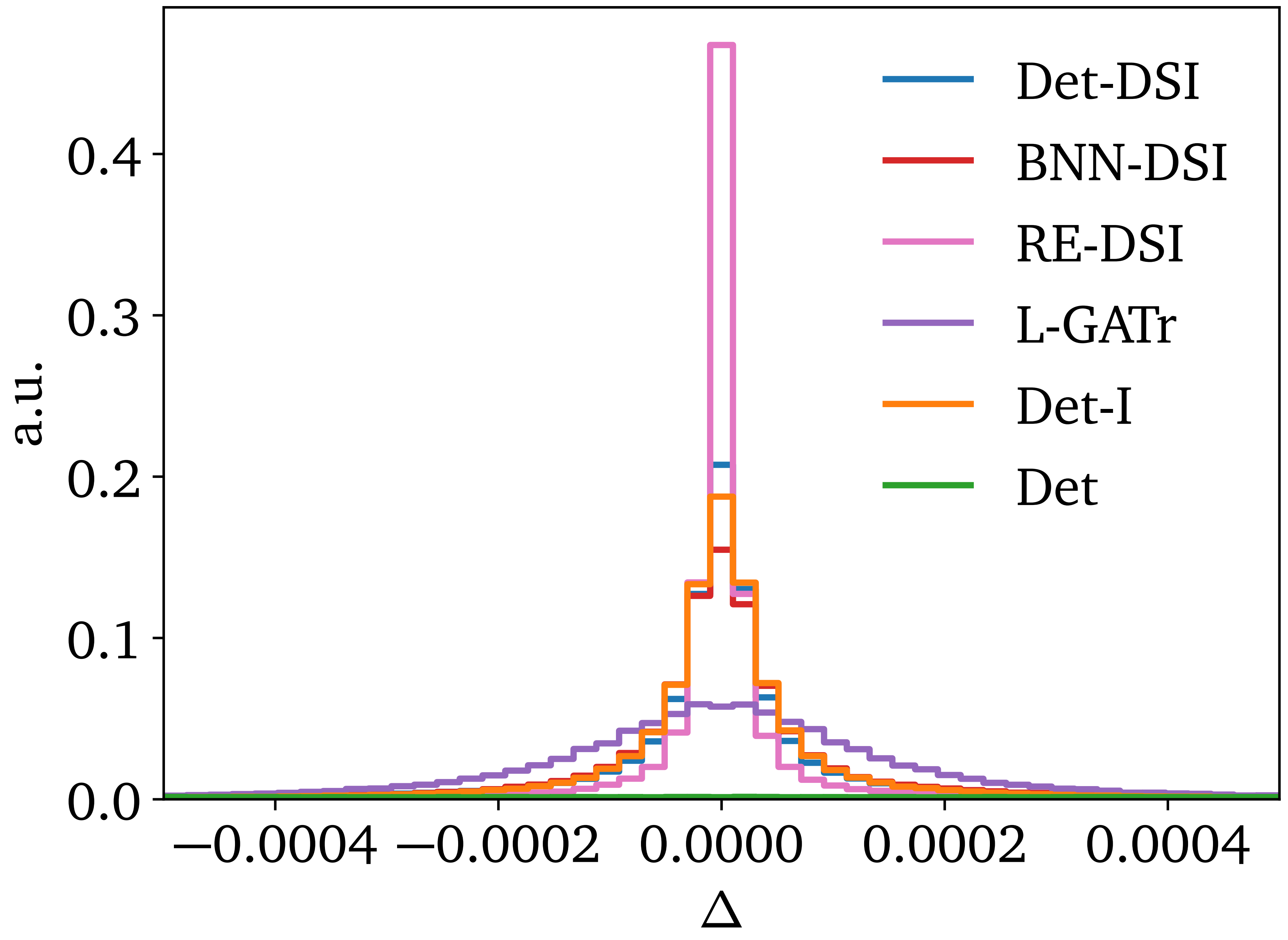
➡ Improvement of intrinsic uncertainty to 0.3%

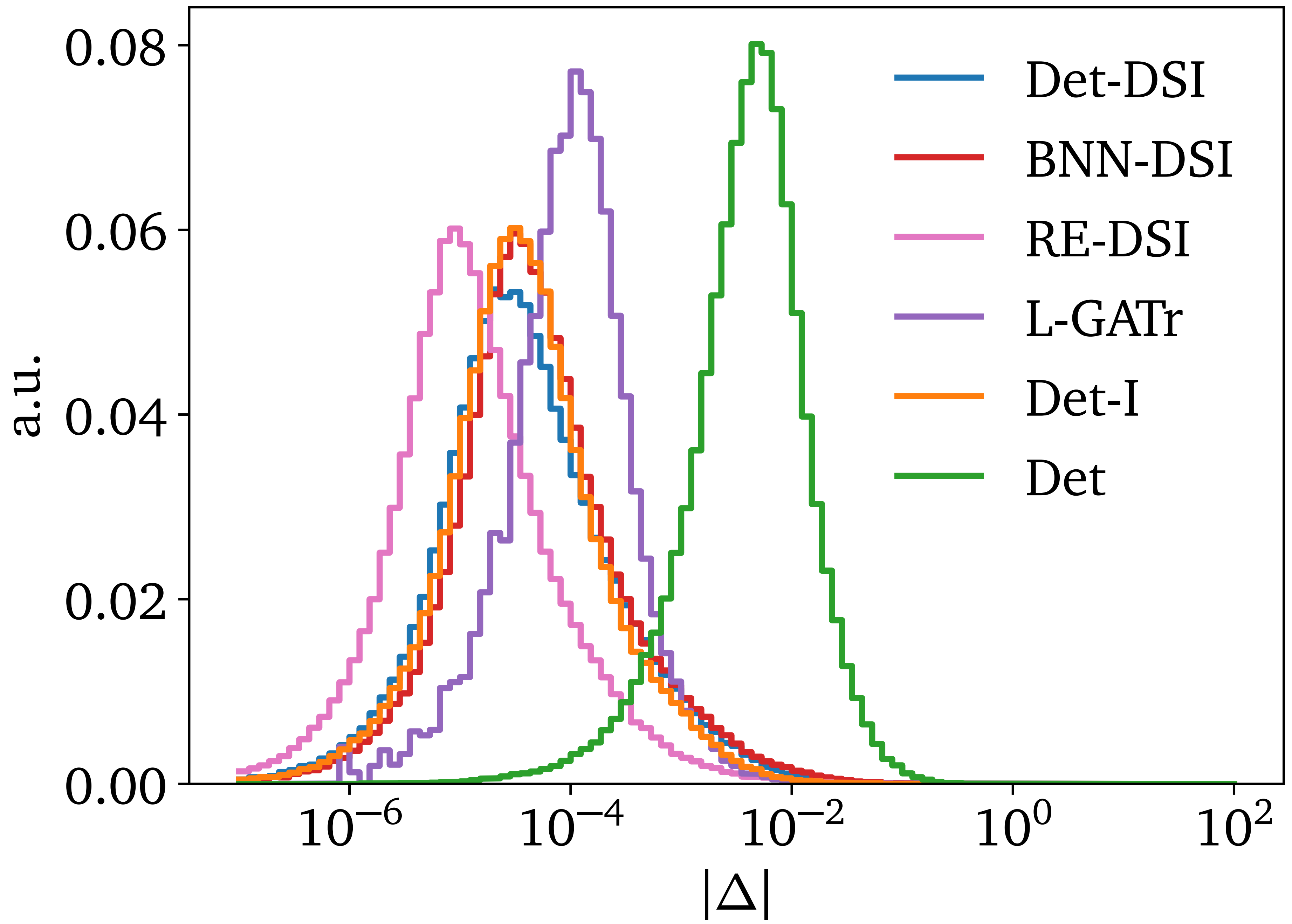
Testing advanced architectures

- Enhance standard networks through representation learning:

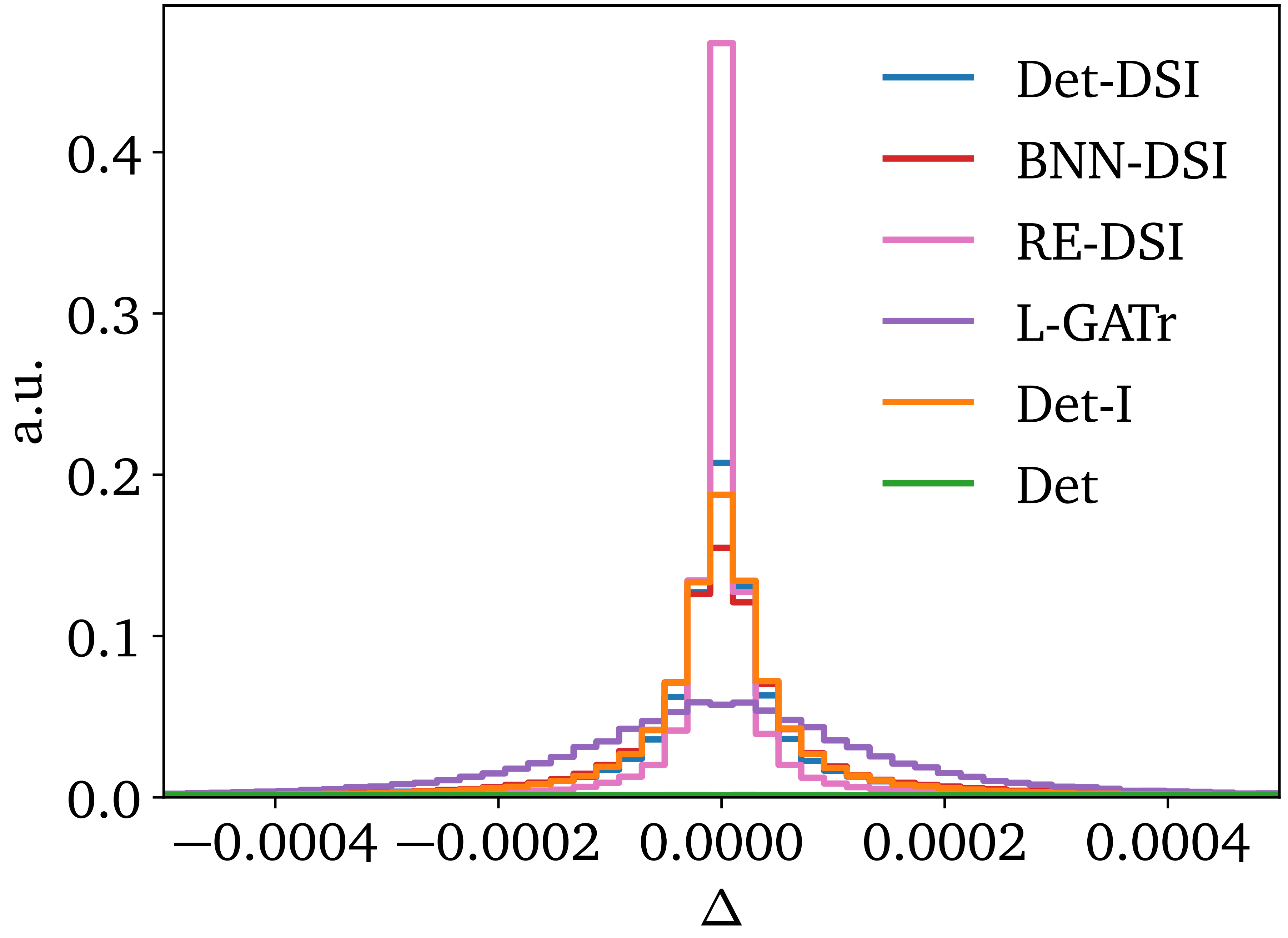
1. **Deep Sets** (DS): learns embedding for each particle type
2. **Deep Sets Invariants** (DSI): DS with Lorentz invariance added as input
3. **L-GATr**: fully Lorentz equivariant network architecture [2411.00446]

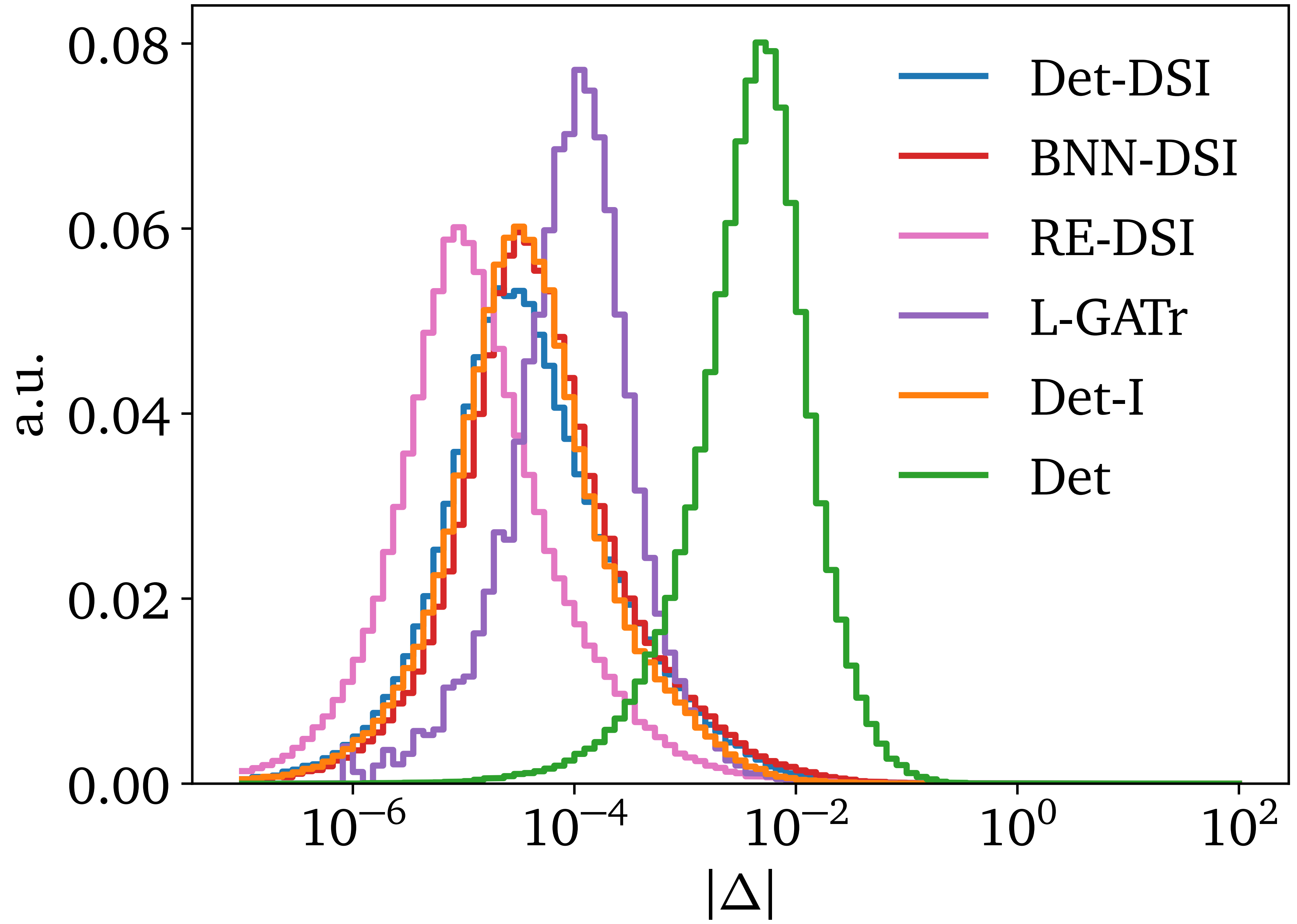
Advanced architectures - Accuracy





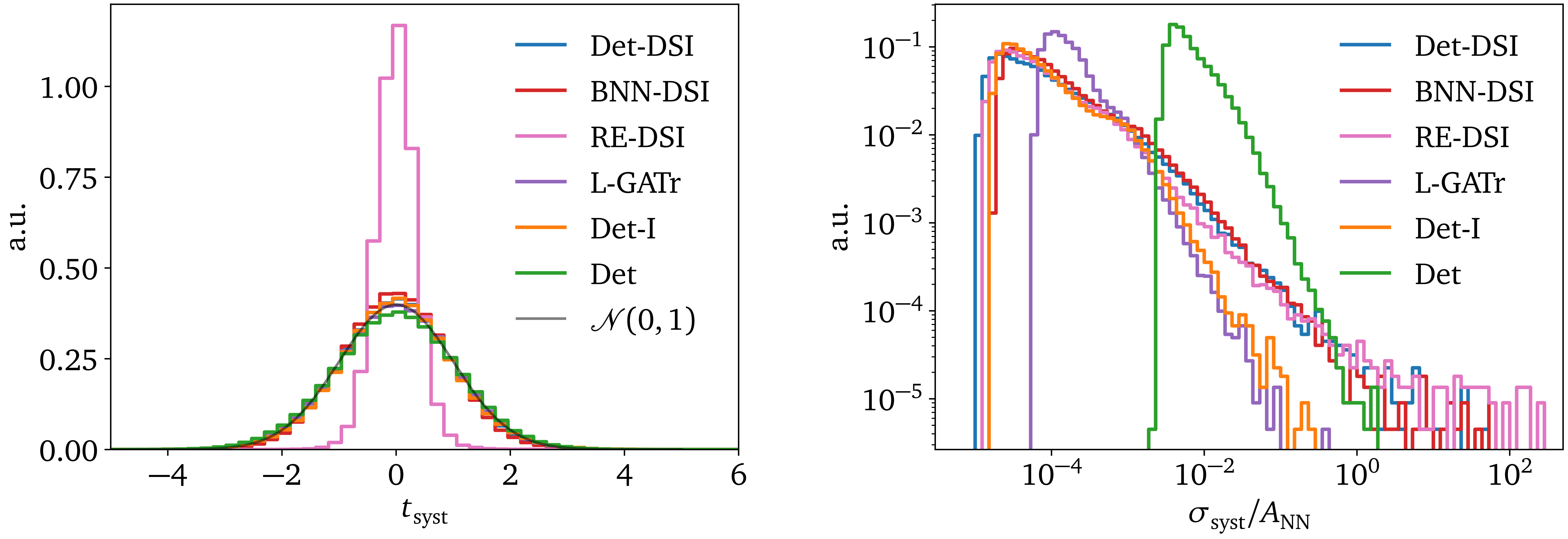
Advanced architectures - Accuracy



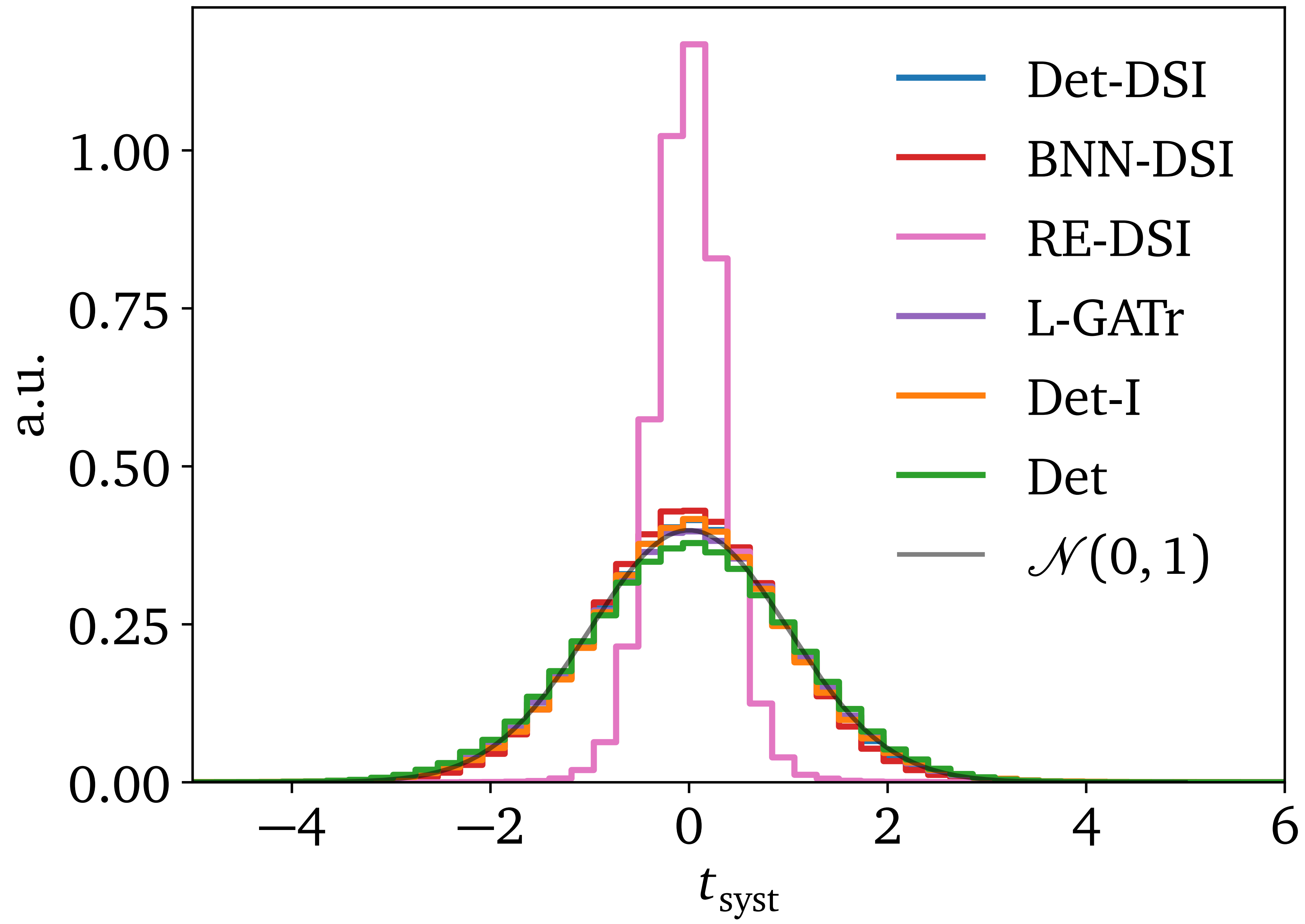


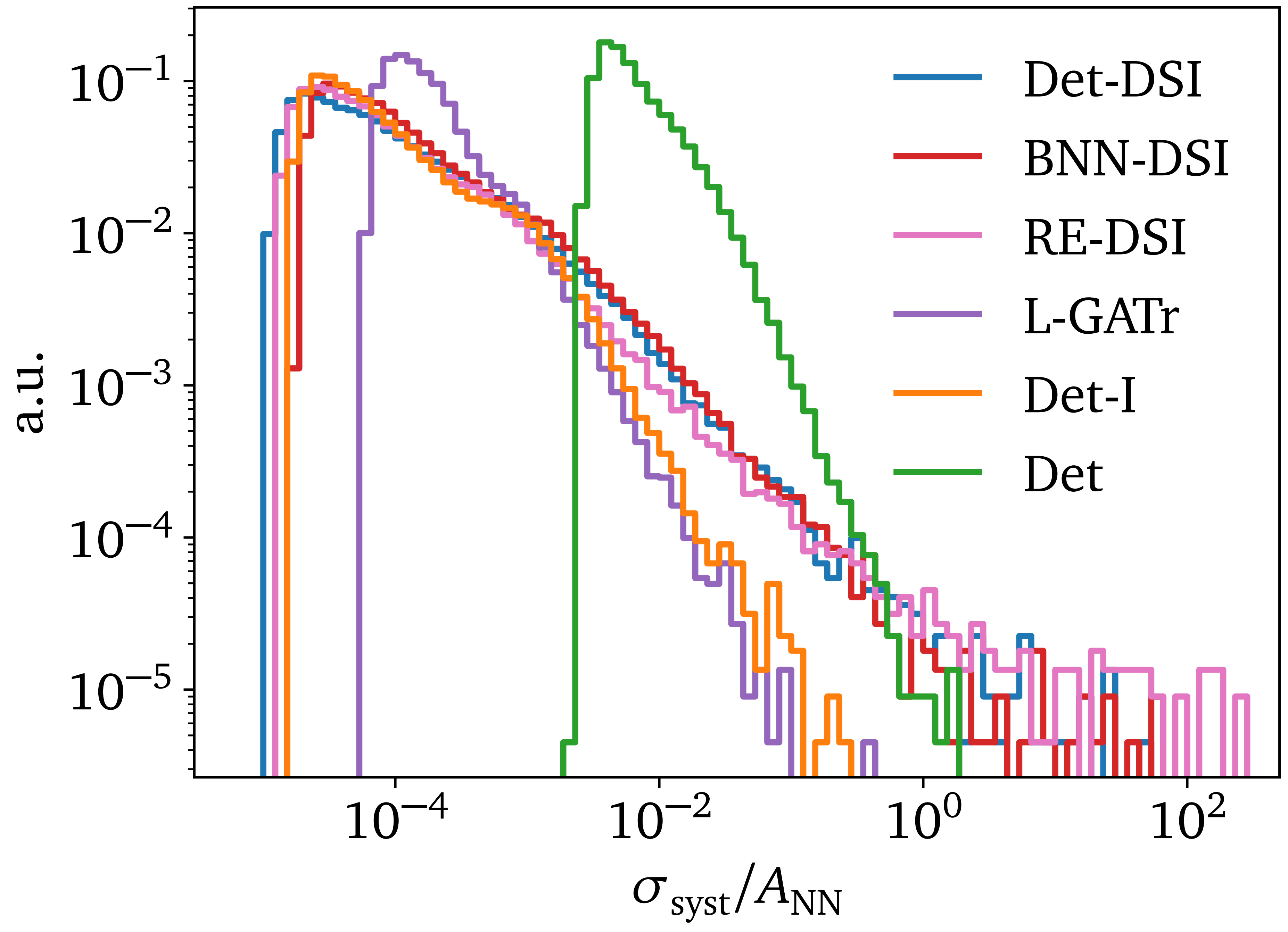
➡**Controlled accuracy** to 10^{-5} level

Advanced architectures - Uncertainties



Advanced architectures - Uncertainties





➡ **Calibrated uncertainty** with **precision** on 10^{-5} level

➡ **Data preprocessing** gain improvement in intrinsic uncertainties

Conclusion

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

Conclusion

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

Thank you for your attention!

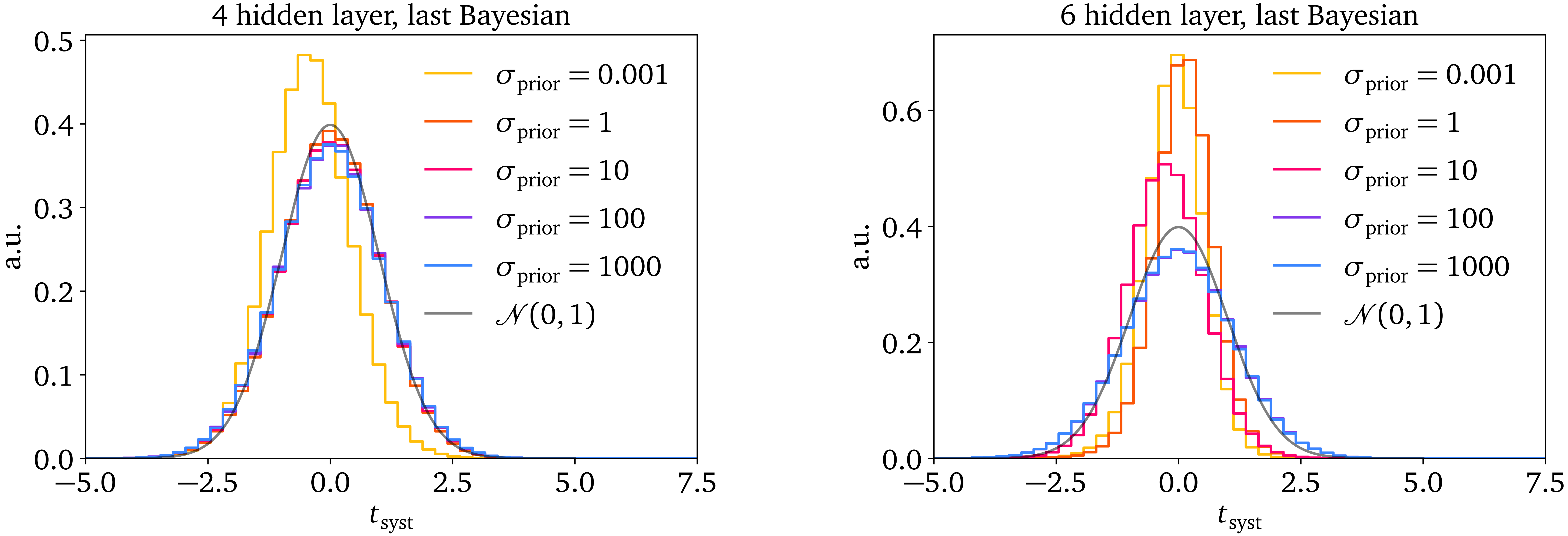
Back up / Additional material

Reducing the training size

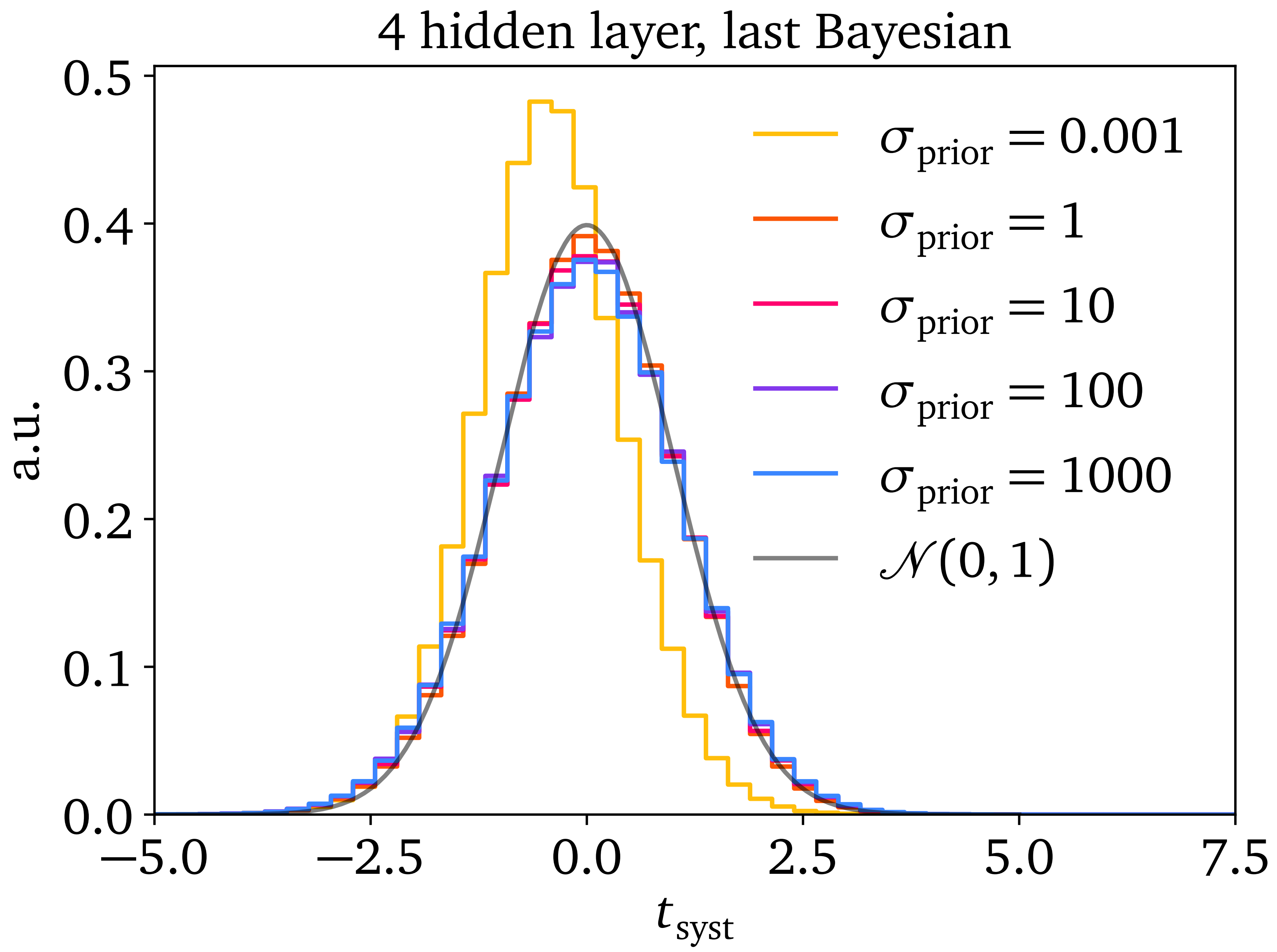
- Systematic uncertainty dominant over statsitical
- Training on 700000 phase space points: $\sigma_{\text{tot}}(x) \approx \sigma_{\text{syst}}(x) \gg \sigma_{\text{stat}}(x)$
- Reducing training data to 100000 phase space points

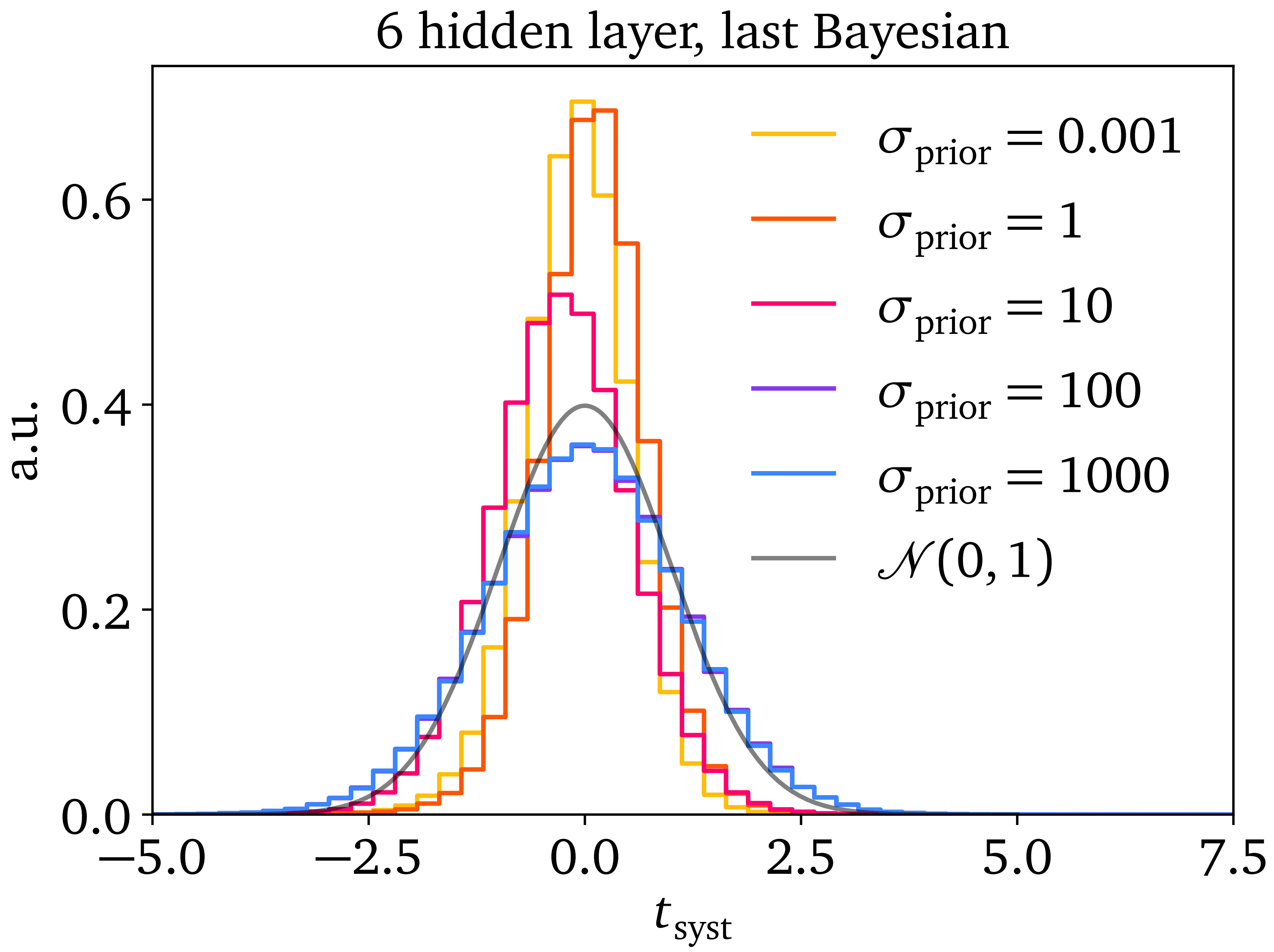
	70%	10%
$\langle \sigma_{\text{syst, BNN-DSI}}/A \rangle$	$8.7 \cdot 10^{-5}$	$2.5 \cdot 10^{-4}$
$\langle \sigma_{\text{stat, BNN-DSI}}/A \rangle$	$3.6 \cdot 10^{-5}$	$1.5 \cdot 10^{-4}$
$\langle \sigma_{\text{syst, RE-DSI}}/A \rangle$	$5.1 \cdot 10^{-5}$	$2.9 \cdot 10^{-4}$
$\langle \sigma_{\text{stat,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

Prior influence in the BNN



Prior influence in the BNN


4 hidden layer, last Bayesian

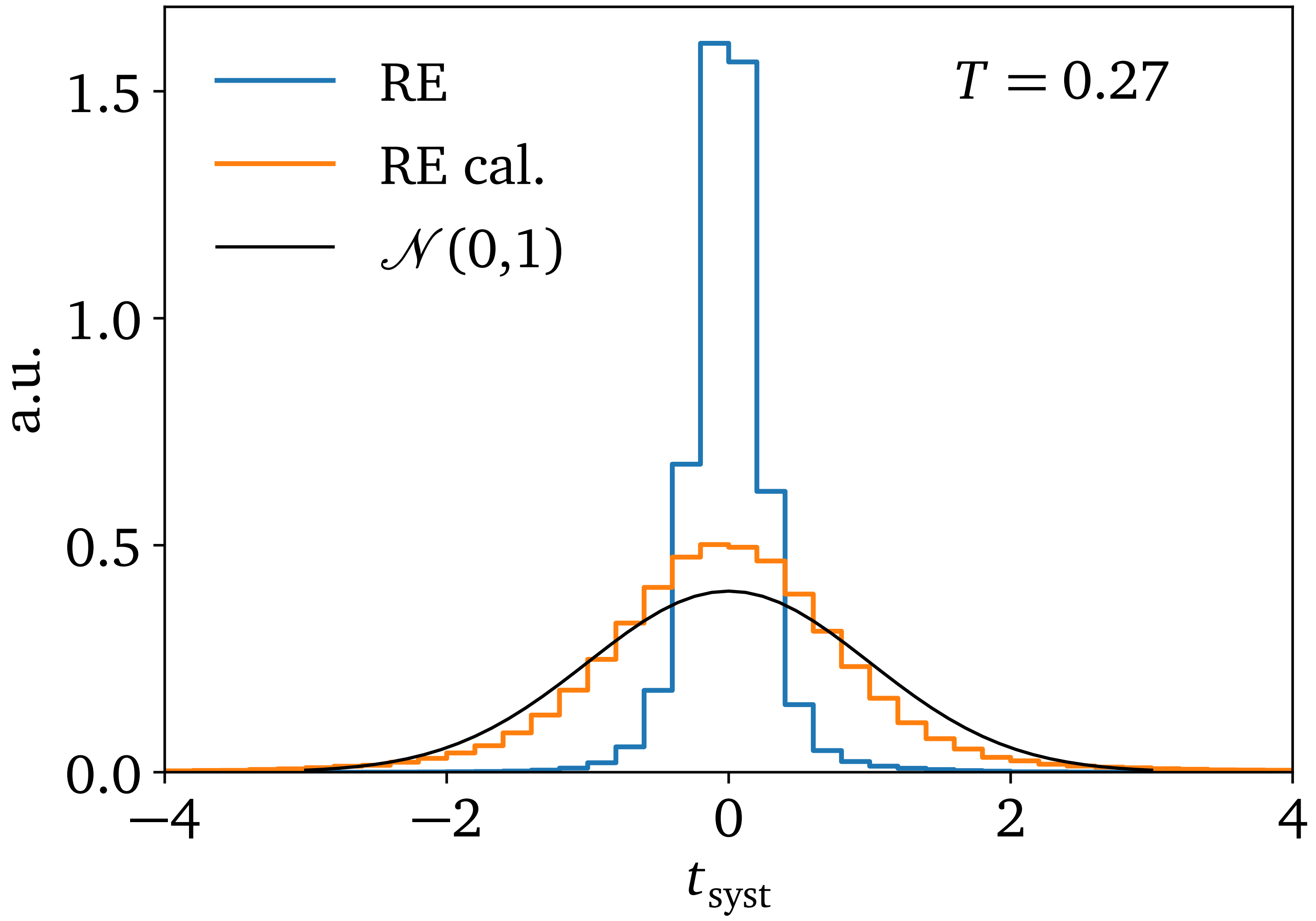

6 hidden layer, last Bayesian

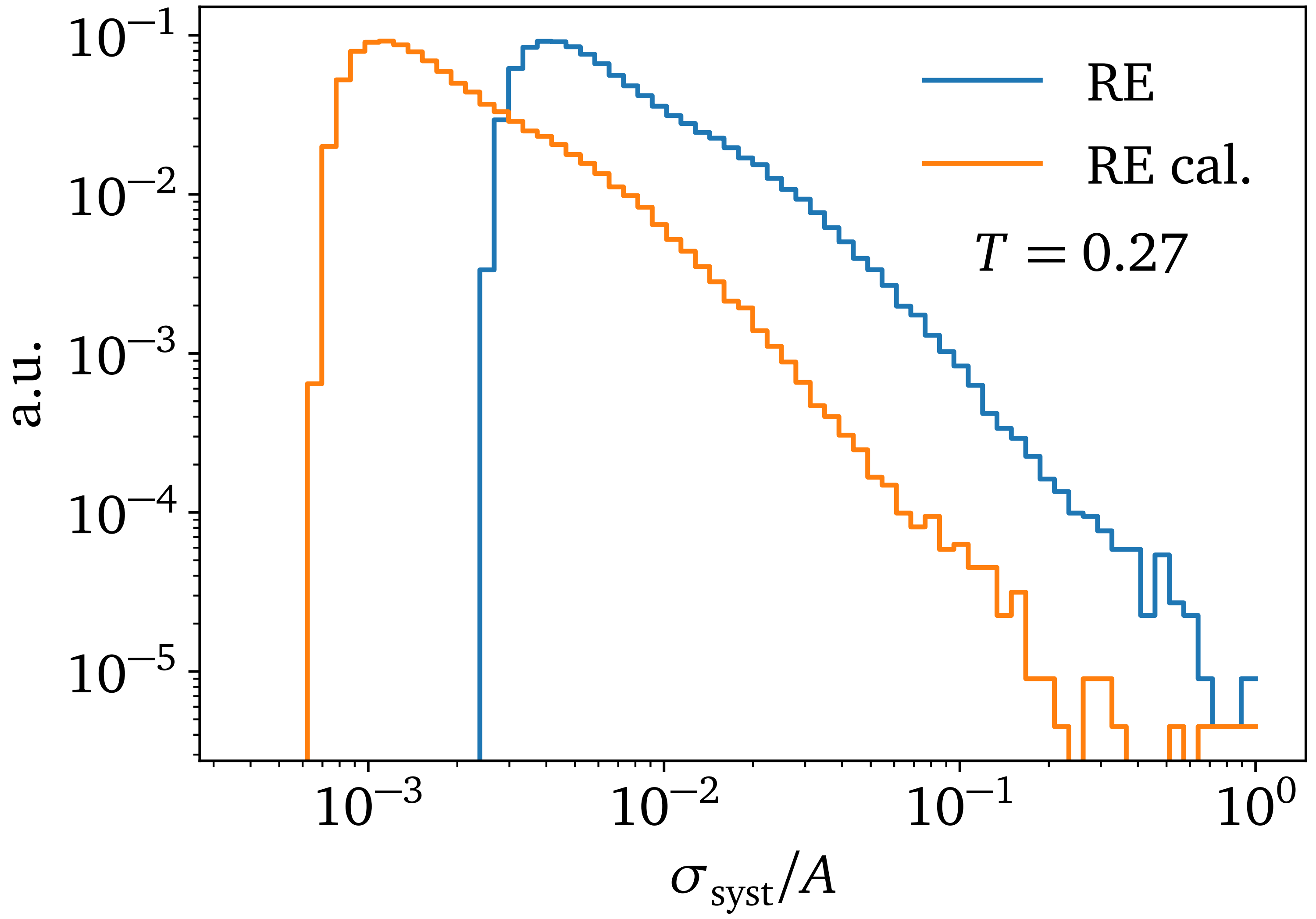
➡ Results don’t depend on prior

➡ For more layers a larger prior is needed

Calibration of networks

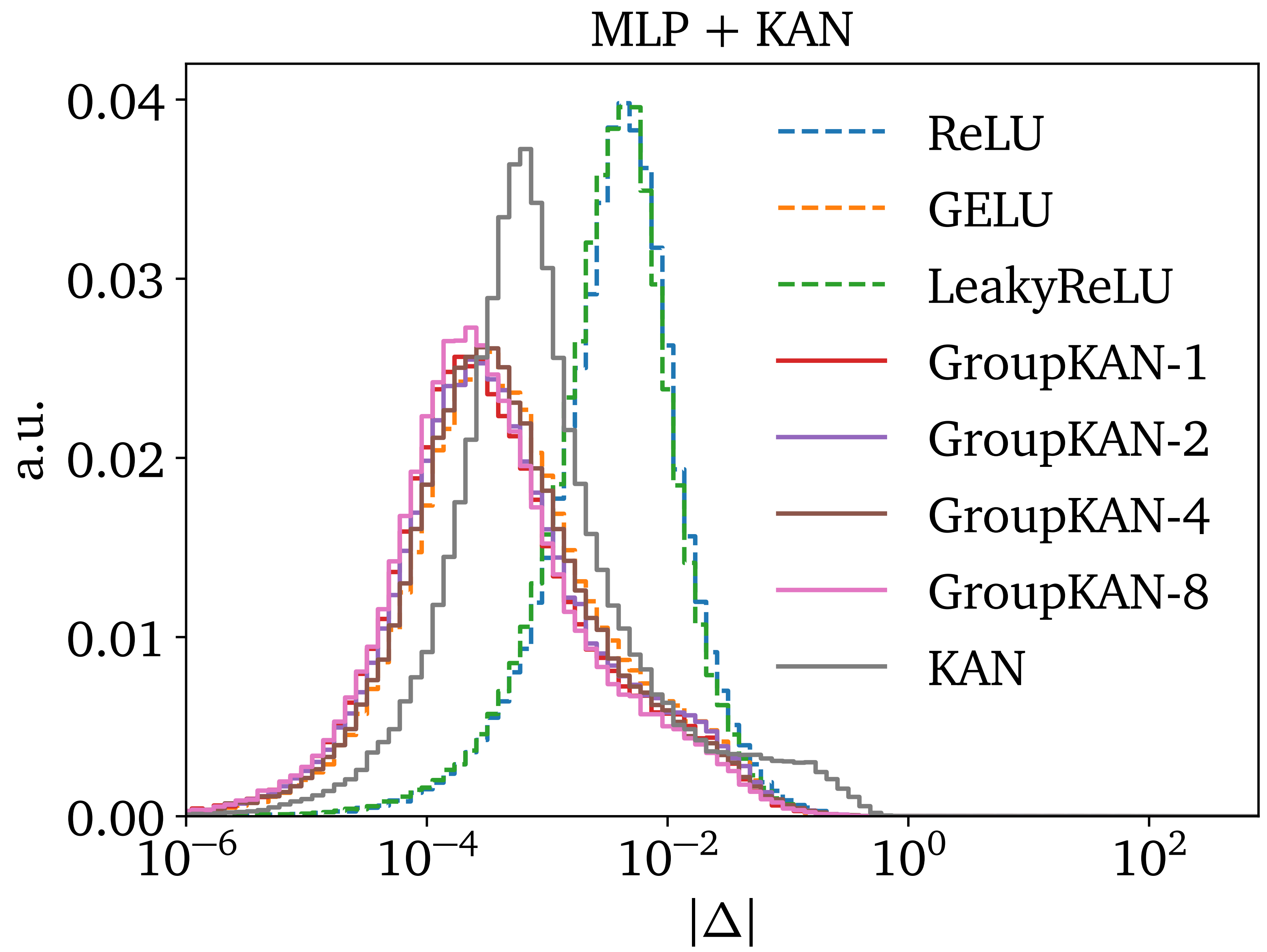
- Calibrate RE by introducing scaling parameter T: $\sigma_{\text{syst}} \to \sigma_{\text{syst}} \times T$
- T estimated by using stochastic gradient descent $\mathcal{L}_T(x) = \left\langle \frac{|A_{\text{true}}(x) - \bar{A}(x)|^2}{2\sigma^2(x)T^2} + \log \sigma(x)T \right\rangle_{x\sim D_{\text{train}}}$

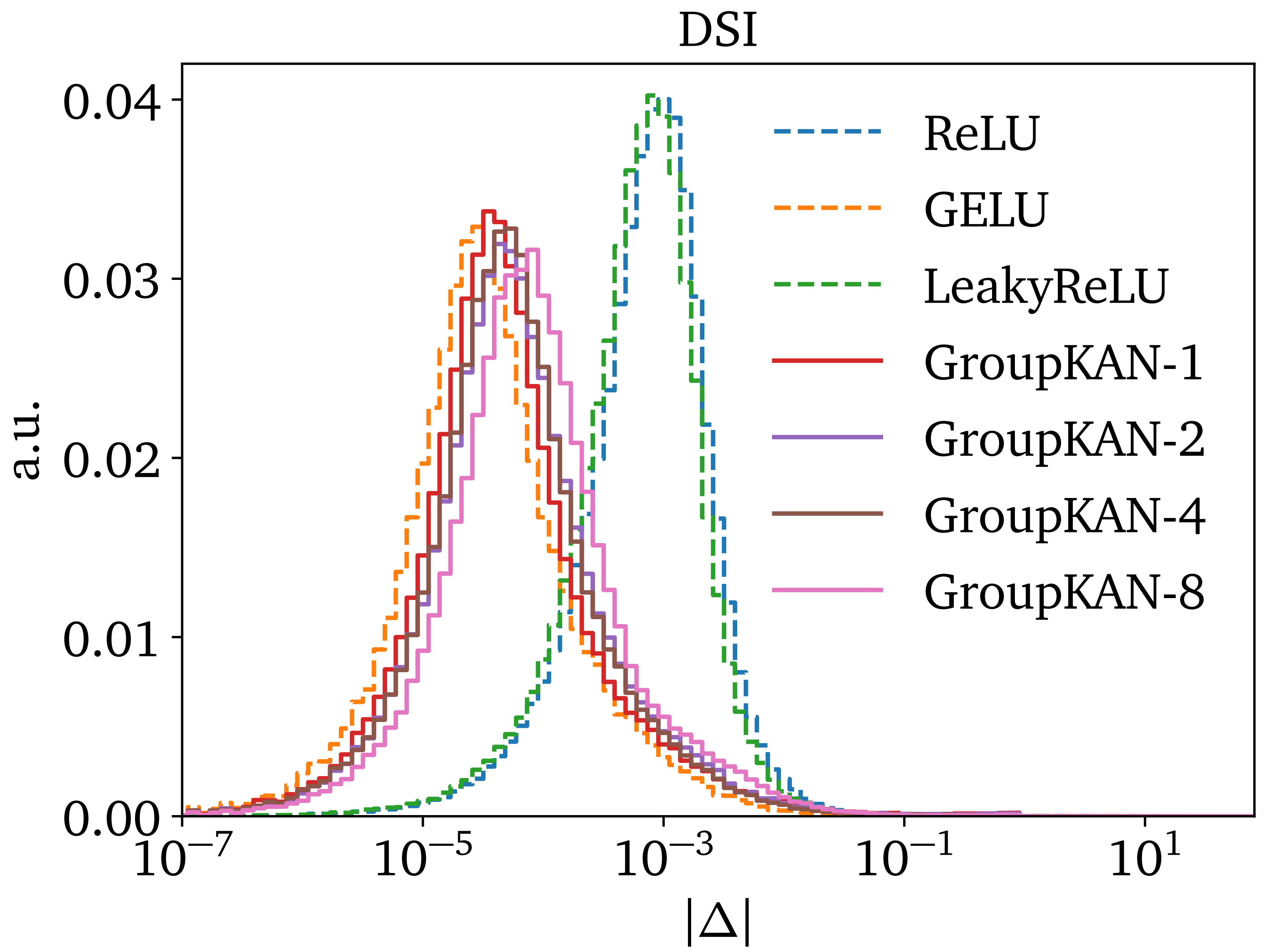




Kolmogorov-Arnold networks (KANs)

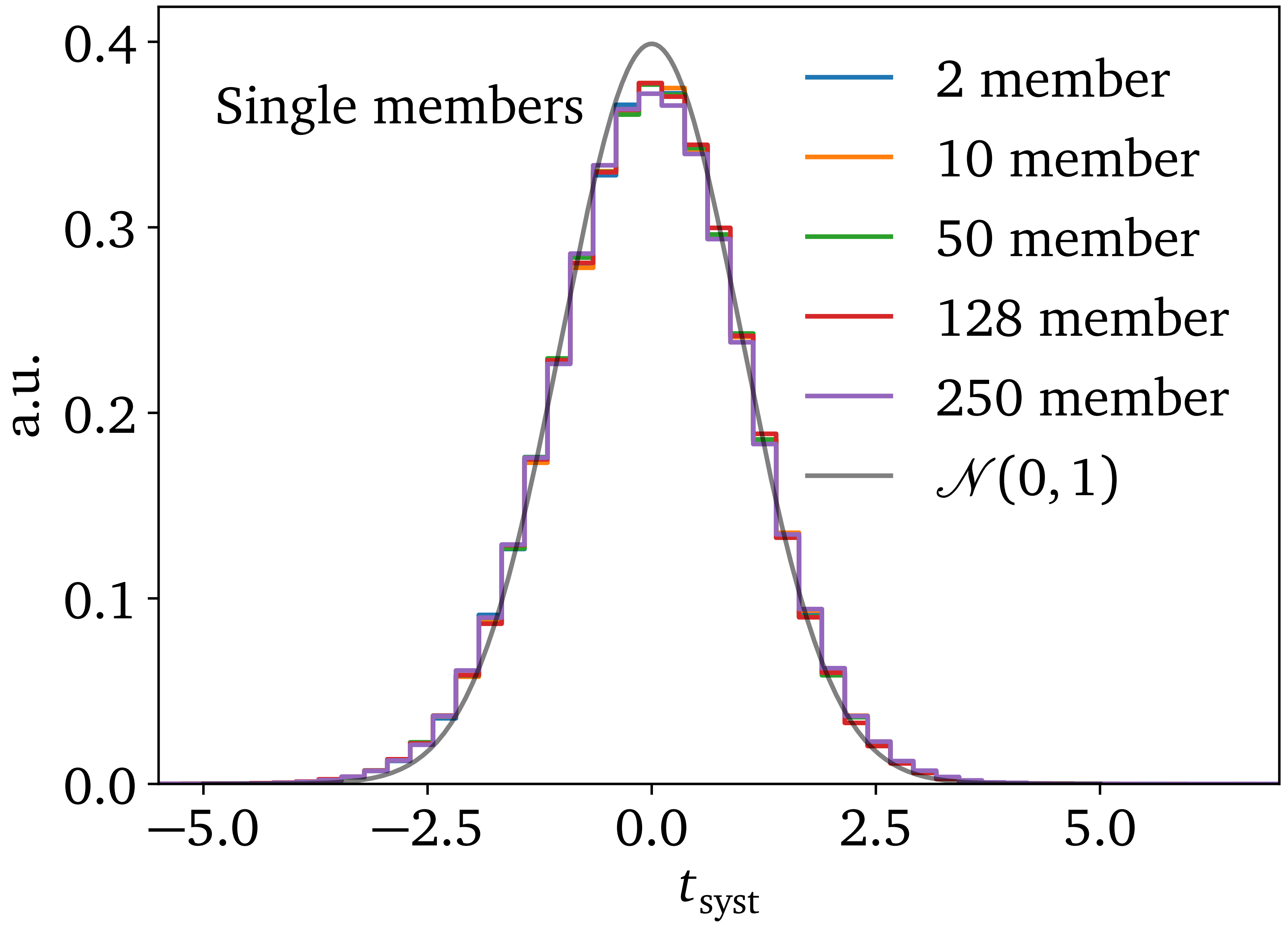
- Use KANs for calibration
- GroupKANs allow for learnable activation functions

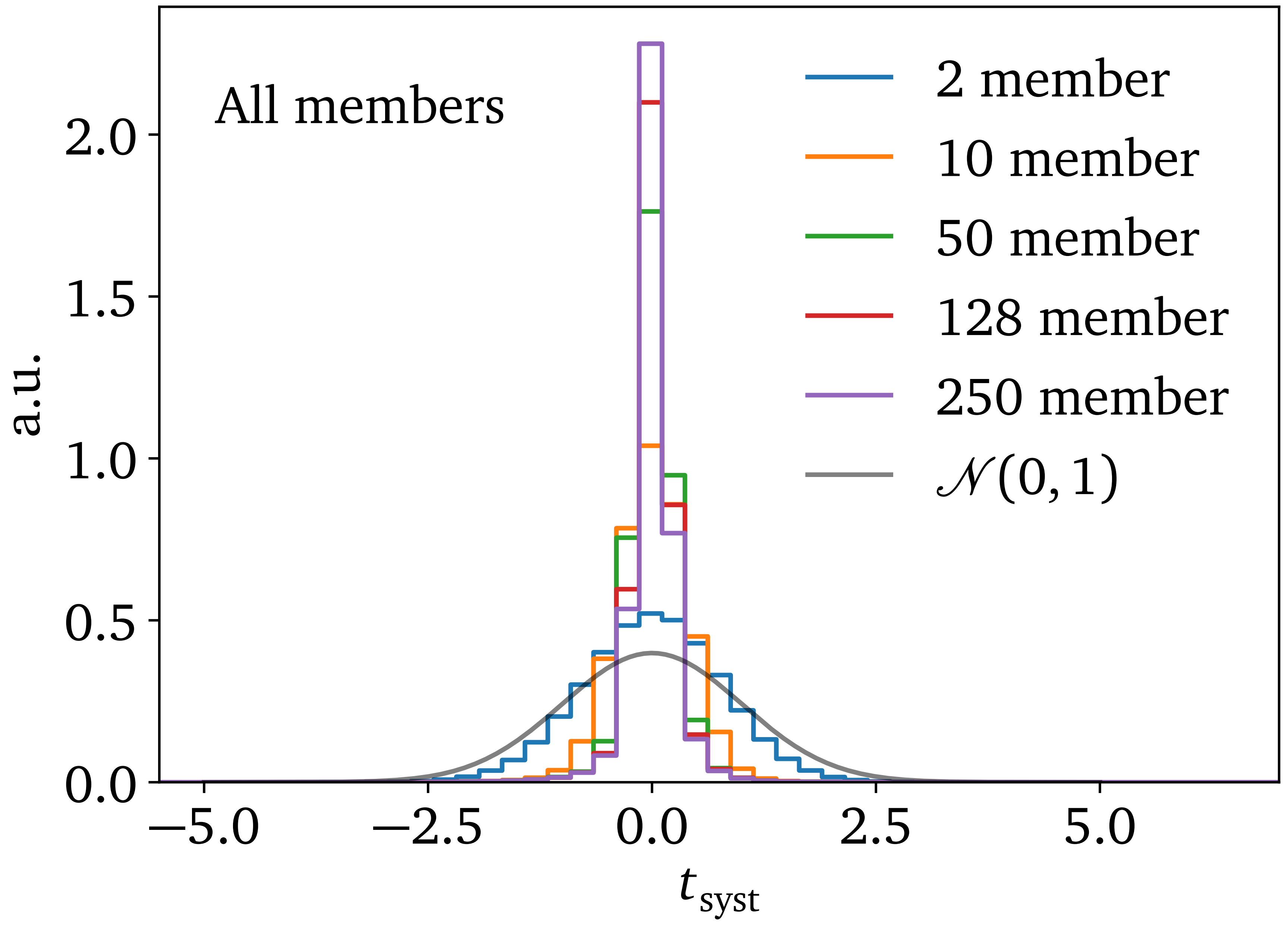

MLP + KAN


DSI

Systematic pull of REs - remove

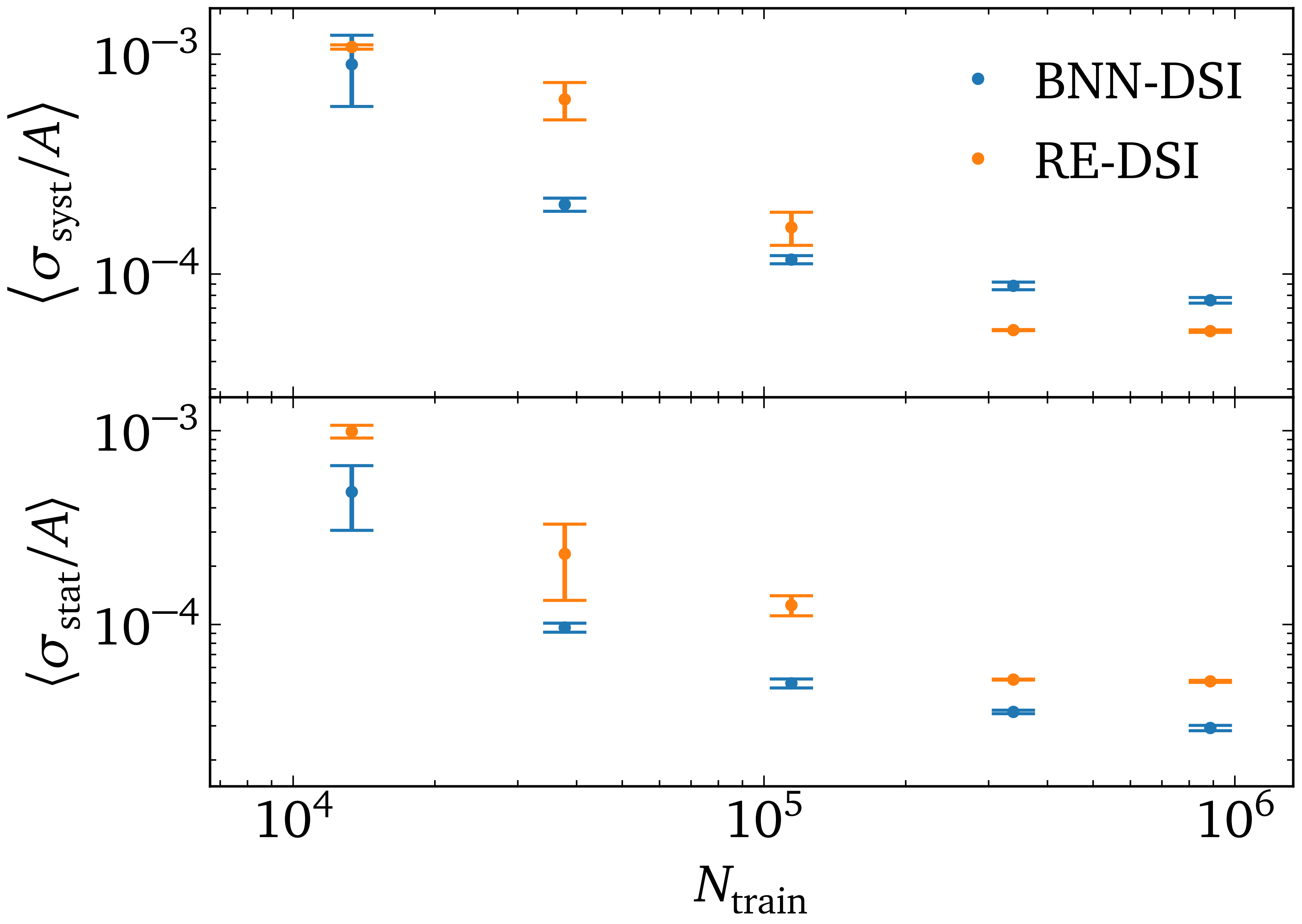
- Learned $\sigma_{syst}(x)$ too conservative

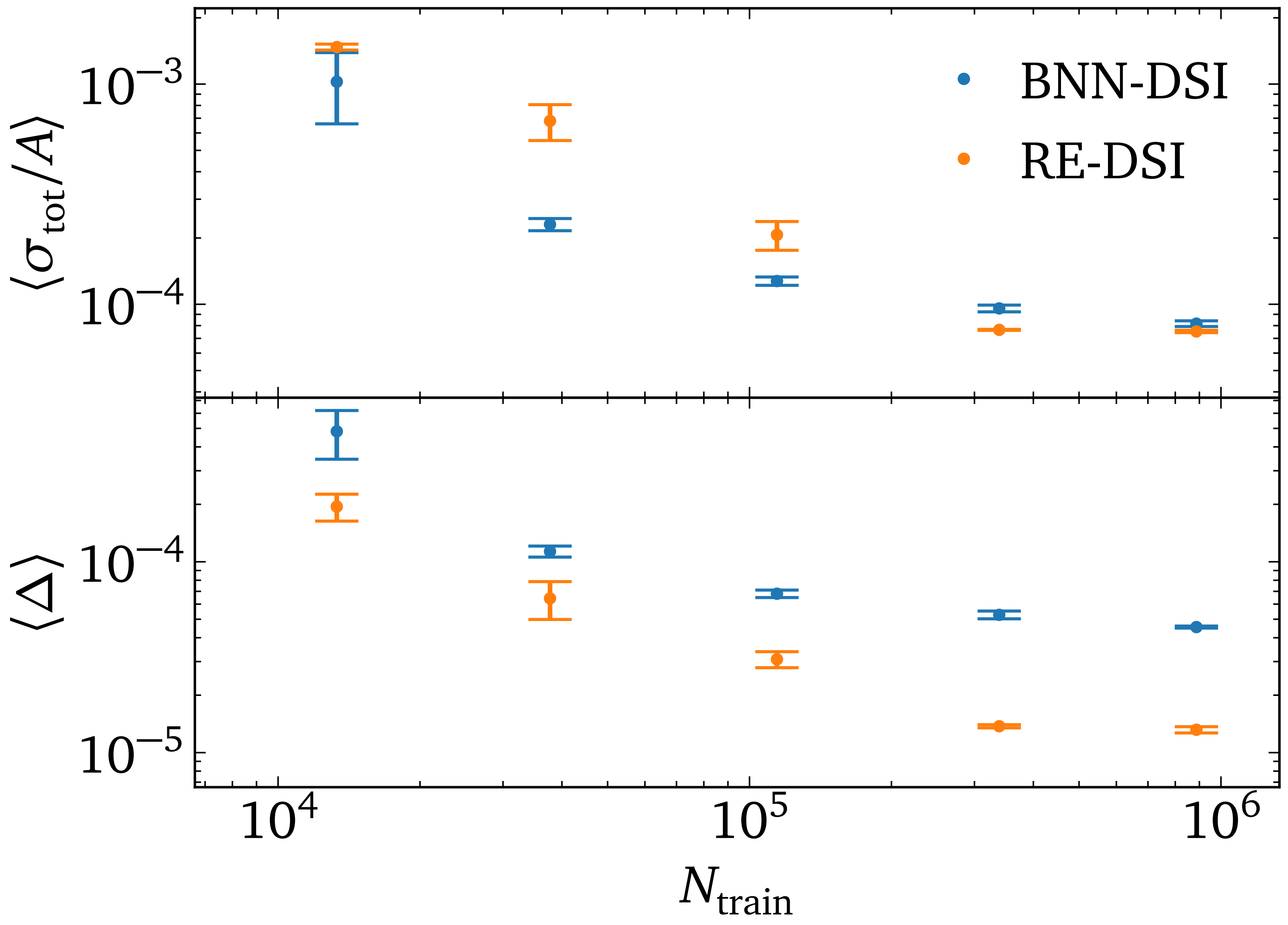




➡Prediction benefits from ensemble nature but not σ_{syst}

Relative uncertainty vs training size





➡ σ_{syst} always larger

➡ σ larger for RE-DSI than BNN-DSI

➡ Difference in σ_{tot} for small data

➡ RE-DSI more accurate in prediction