Accuracy and uncertainty control for ML-Amplitudes

Nina Elmer

KISS Annual meeting
12.03.2025

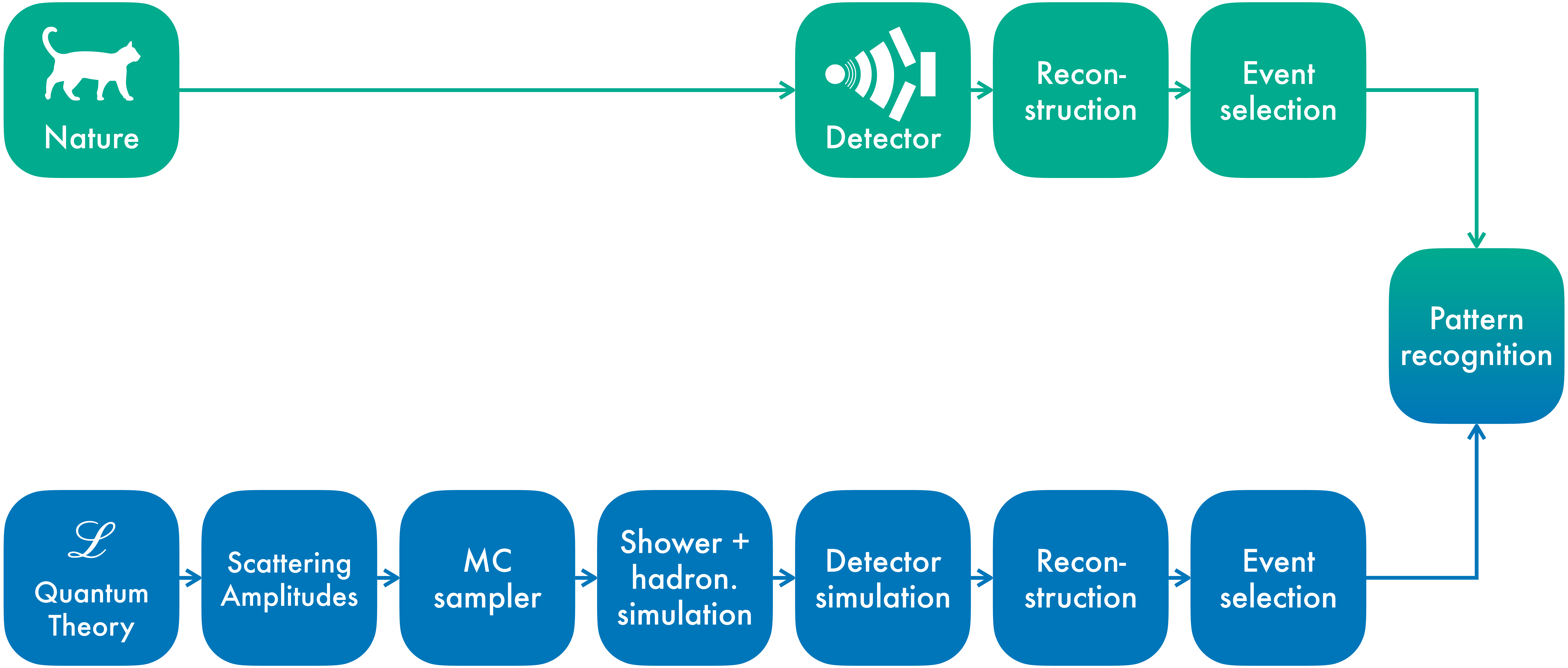
arXiv: 2412.12069

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

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

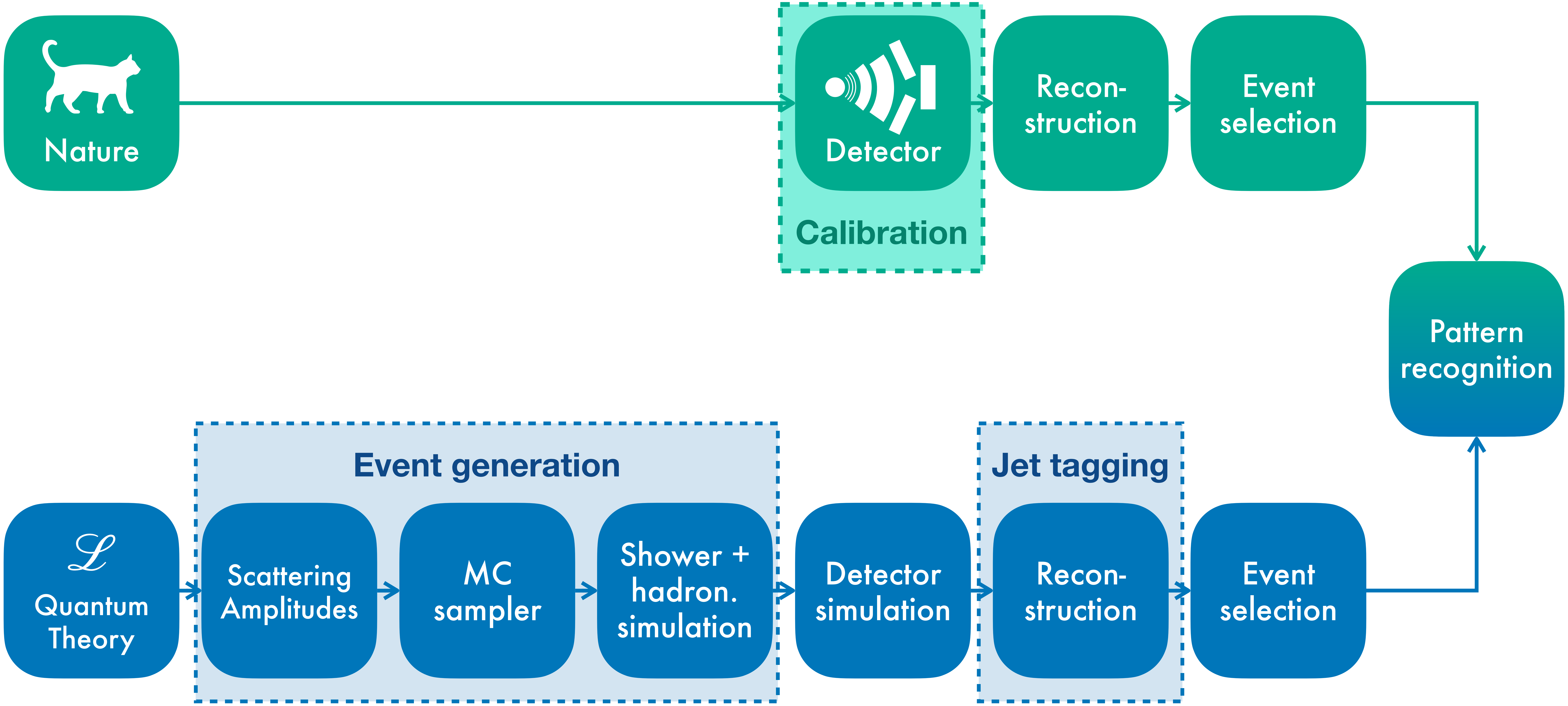
IMPRS
for Precision Tests of Fundamental Symmetries
INTERNATIONAL MAX PLANCK RESEARCH SCHOOL

LHC physics and machine learning



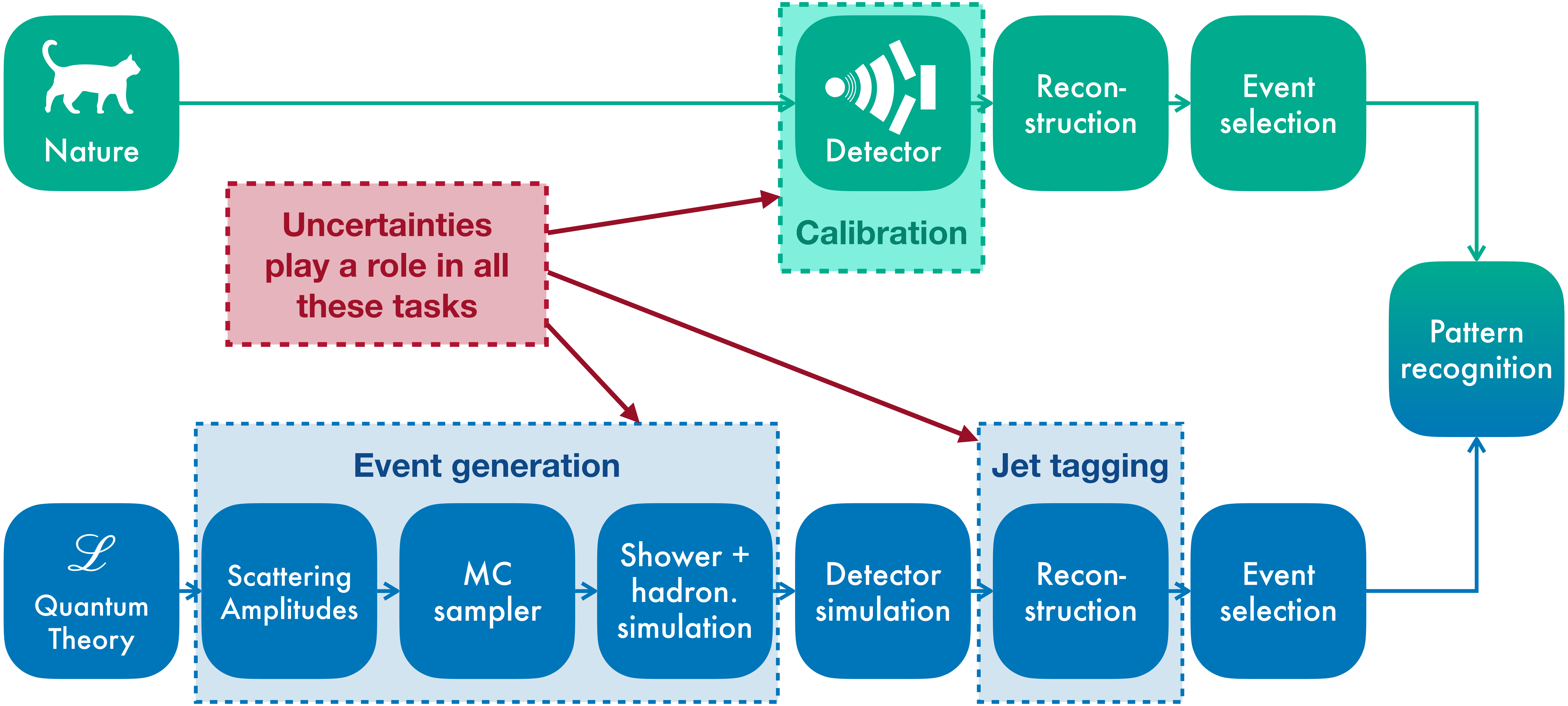
Credits: R. Winterhalder

LHC physics and machine learning



Credits: R. Winterhalder

LHC physics and machine learning



Credits: R. Winterhalder

Motivation

- **Reliable interpolation** of two-loop amplitudes [2412.09534]
  - Scaling up to $Z + 5j$ [2411.00446]
- Amplitude surrogates part of MadGraph and MadNIS (besides generative integration)
  → Theo's talk

Motivation

- **Reliable interpolation** of two-loop amplitudes [2412.09534]
  - Scaling up to $Z + 5j$ [2411.00446]
- Amplitude surrogates part of MadGraph and MadNIS (besides generative integration)
  → Theo's talk
- Improve theory predictions for LHC:

Motivation

- **Reliable interpolation** of two-loop amplitudes [2412.09534]
  - Scaling up to $Z+5j$ [2411.00446]
- Amplitude surrogates part of MadGraph and MadNIS (besides generative integration)
  → Theo's talk
- 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 (epistemic uncertainty)
  - **Statistical**: Vanishes for perfect training (aleatoric uncertainty)

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

Motivation

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

network distribution network output

- Heteroscedastic loss:

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

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:

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

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:

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

- 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**: $A(x) \equiv \langle A \rangle = \int \mathrm{d}A\, A\, p(A\,|\,x) = \int \mathrm{d}\theta\, q(\theta)\, \overline{A}(x,\theta)$

network distribution network output

- Heteroscedastic loss:

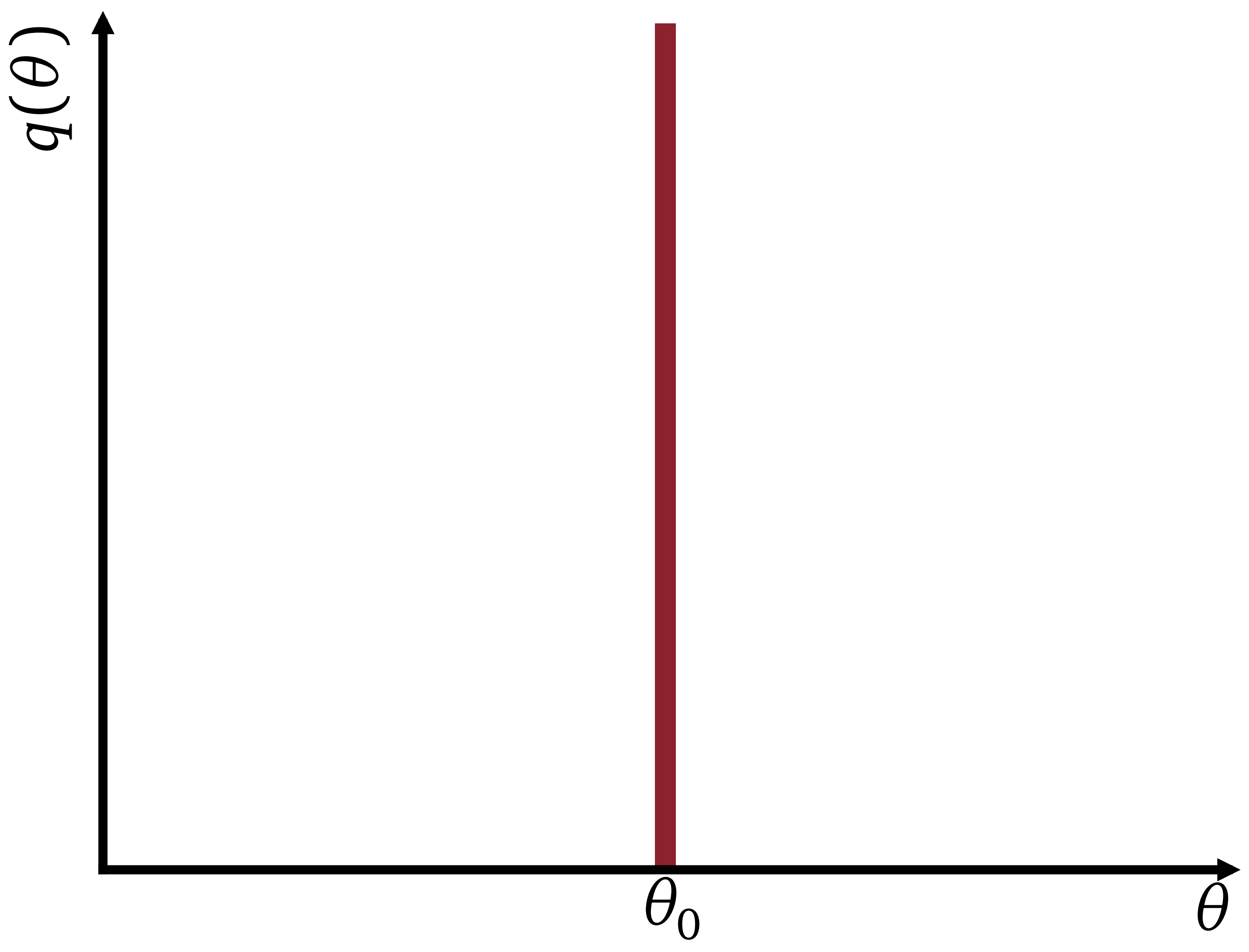
$$\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma_\theta(x_i)^2} + \log \sigma_\theta(x_i) + \ldots$$

- Additional statistical uncertainty:

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

Bayesian neural networks (BNNs)

Neural network

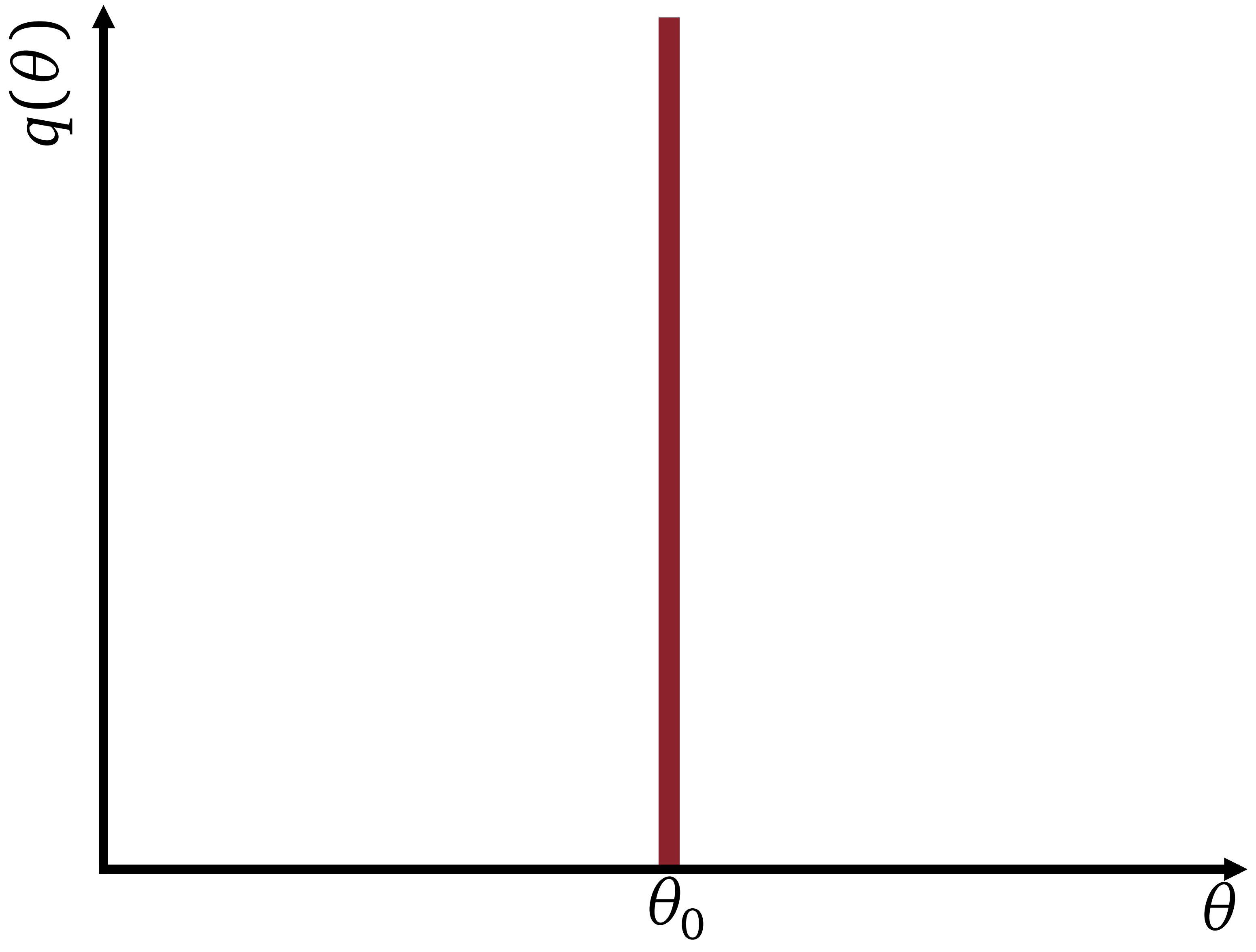


Network weights are deterministic

$$\theta = \theta_0$$

Bayesian neural networks (BNNs)

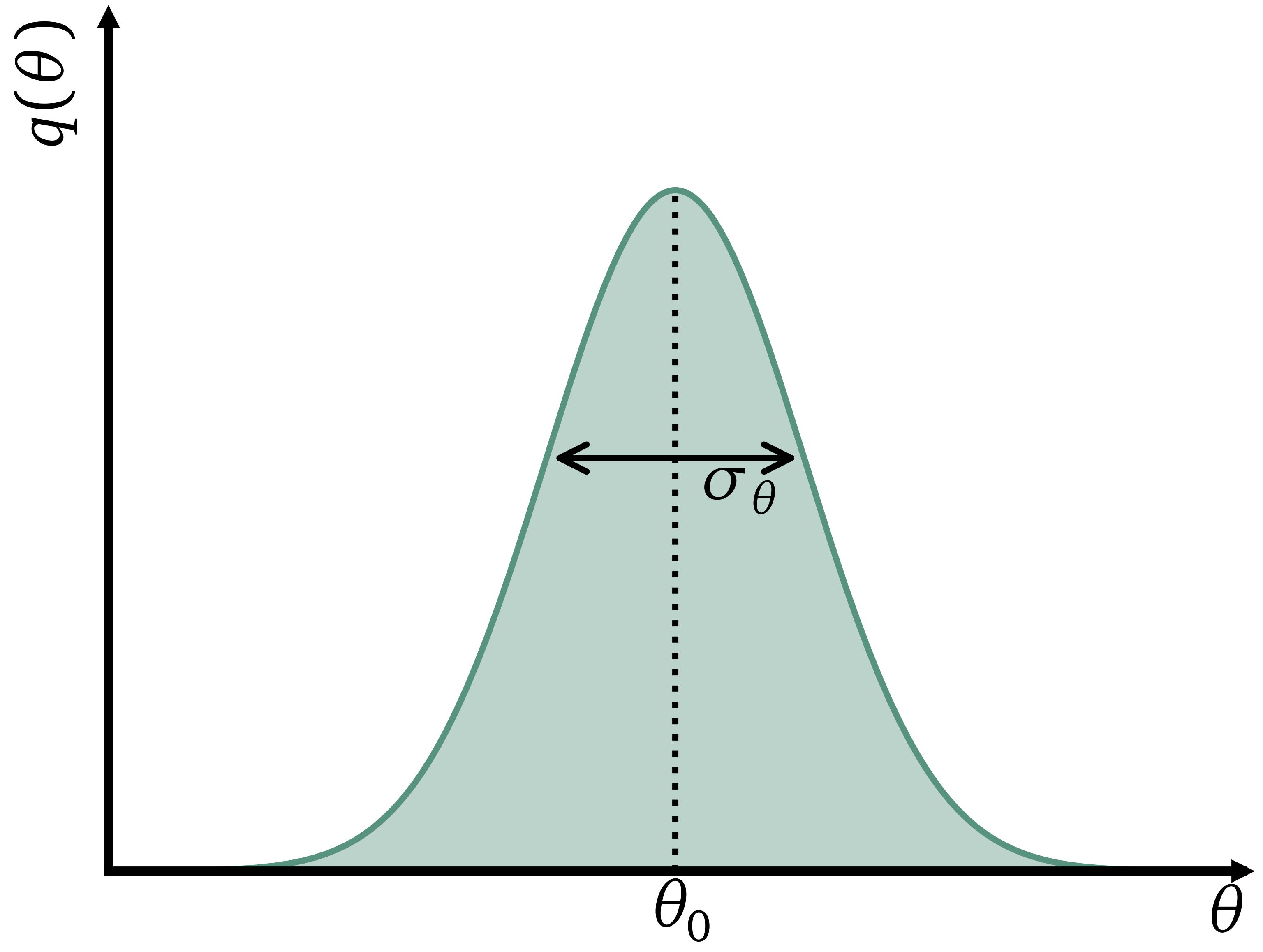
Neural network



Network weights are deterministic

$$\theta = \theta_0$$

Bayesian neural network

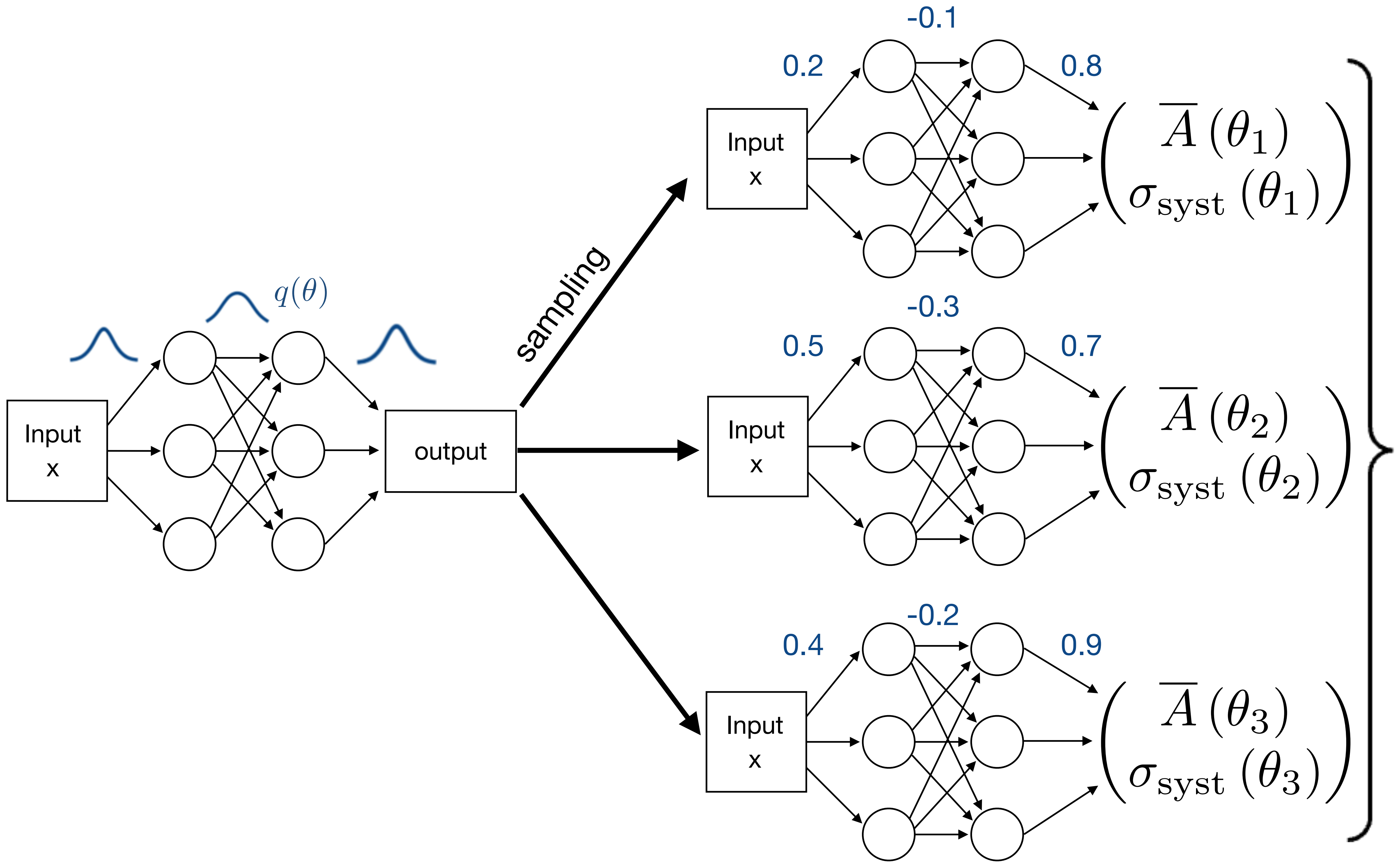


Network weights are drawn from distribution

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

Bayesian neural networks (BNNs)

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



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

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

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

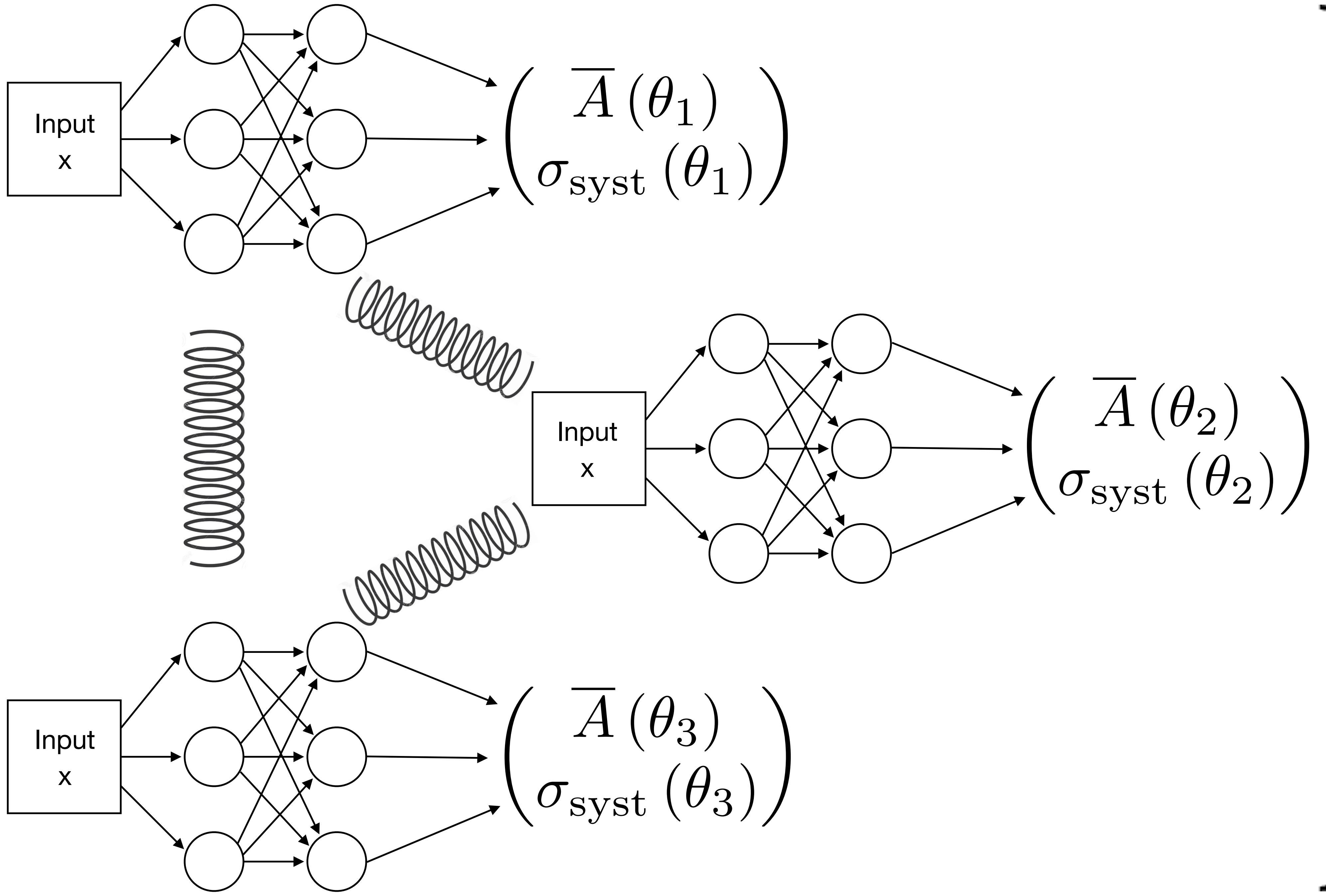
- Parameters: **Network weights** $q(\theta)$
- $q(\theta)$: Params of a Gaussian
- Ensemble: **Sample** $q(\theta)$

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

Repulsive ensembles (REs)

Ensemble of networks

Output



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

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

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

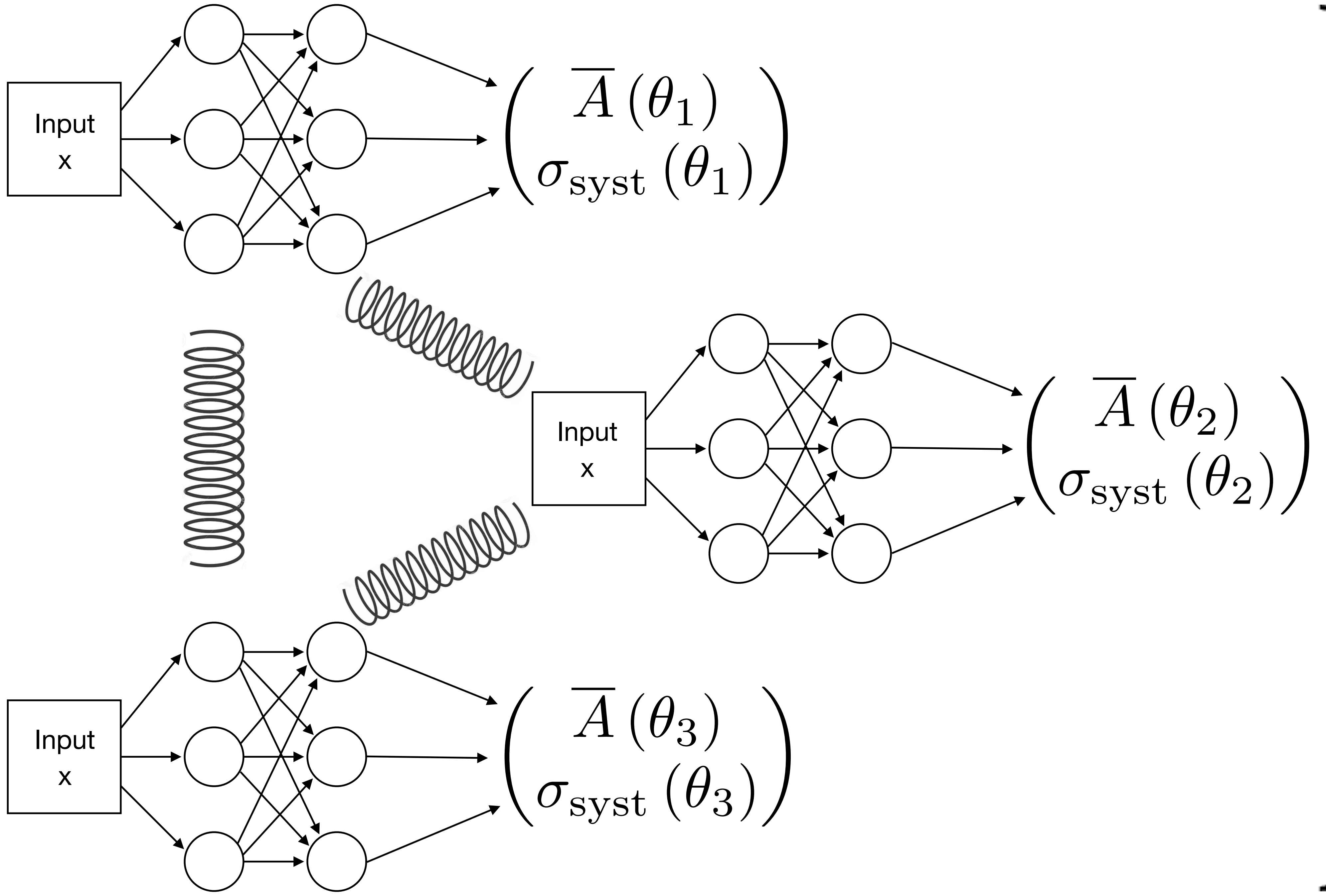
- Repulsive term:
  Cover **full posterior** distribution
- Members **trained simultaneously**

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

Repulsive ensembles (REs)

Ensemble of networks

Output



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

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

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

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

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

Learning Amplitudes

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

Learning Amplitudes

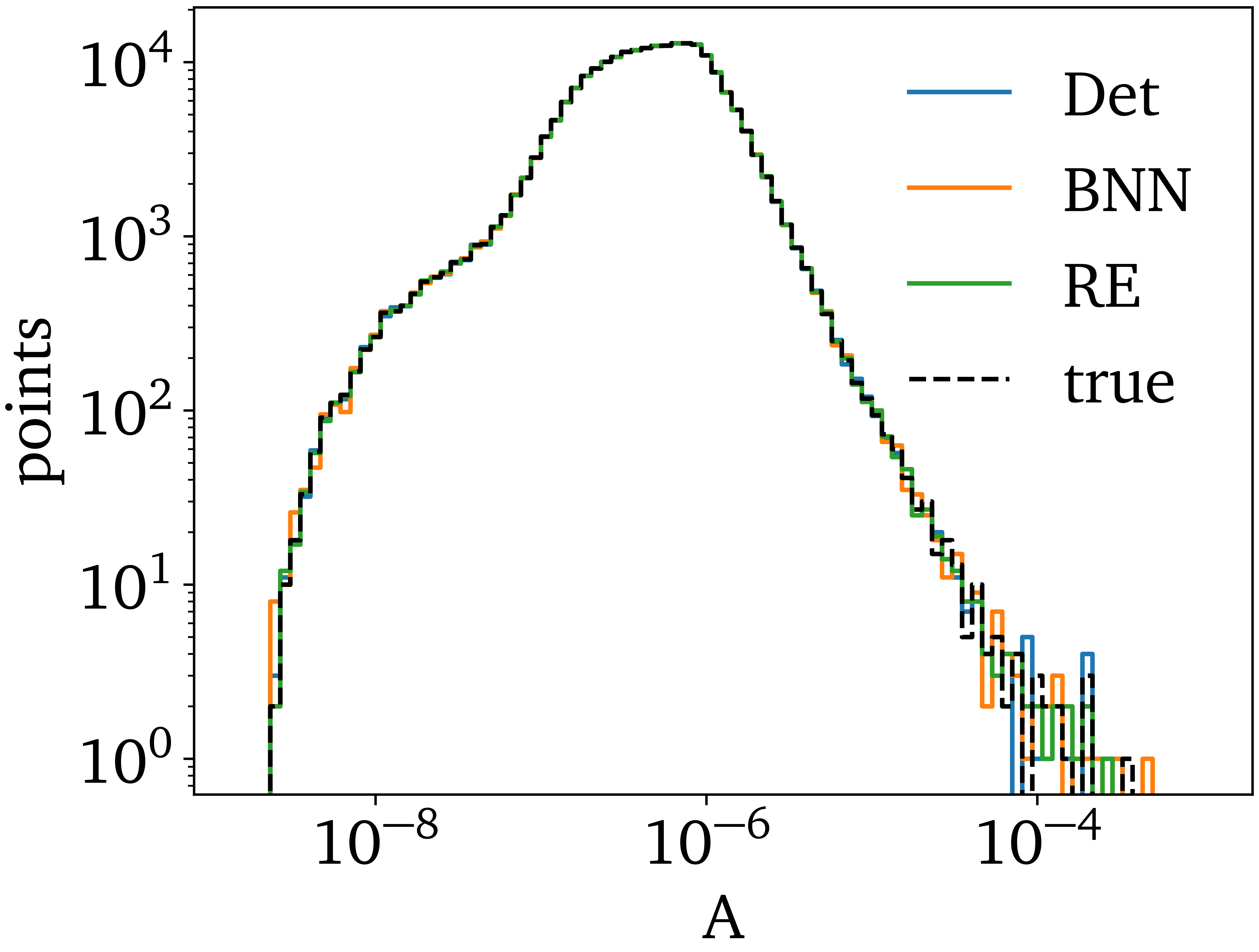
- Learning scattering amplitudes $|M|^2$
- One-loop partonic process: $gg \to \gamma\gamma g$
- 1.1 M phase space points
- Apply basic cuts:

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

Learning Amplitudes

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

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



Adding Gaussian noise

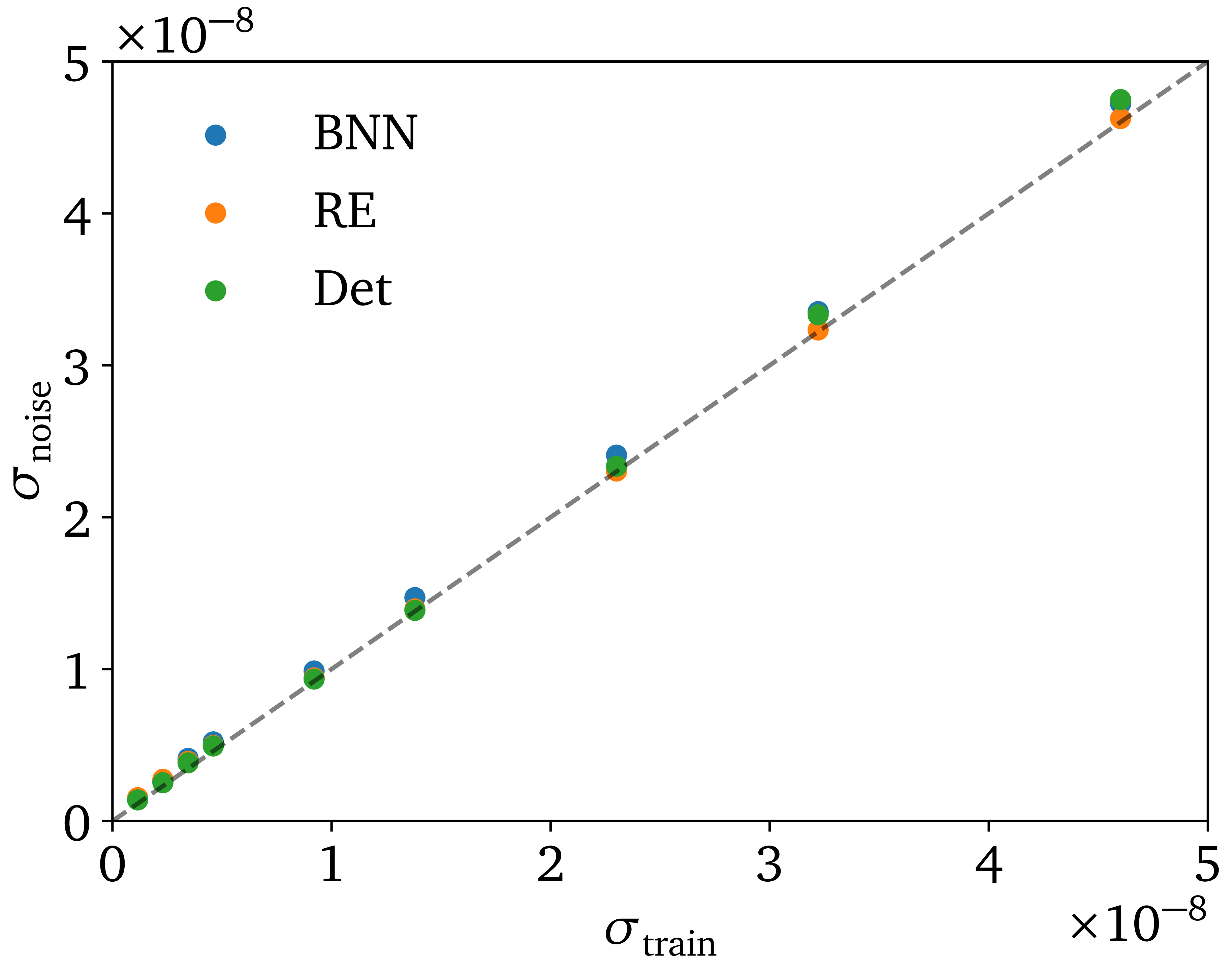
$$\sigma^2_{\text{tot}} = \sigma^2_{\text{syst},0} + \sigma^2_{\text{noise}} + \sigma^2_{\text{stat}}$$

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

Adding Gaussian noise

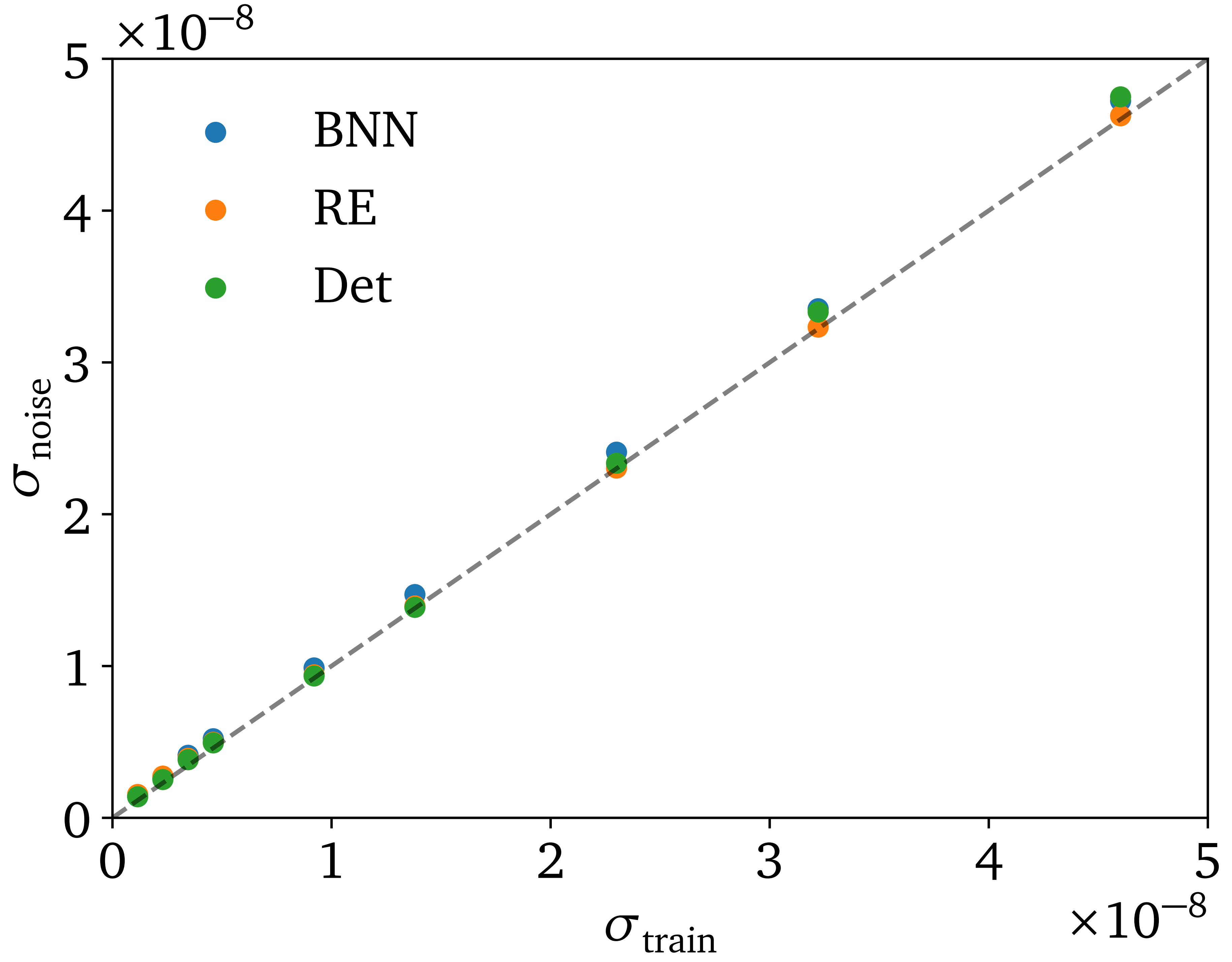
$$\sigma_{\text{tot}}^2 = \sigma_{\text{syst},0}^2 + \sigma_{\text{noise}}^2 + \sigma_{\text{stat}}^2$$

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

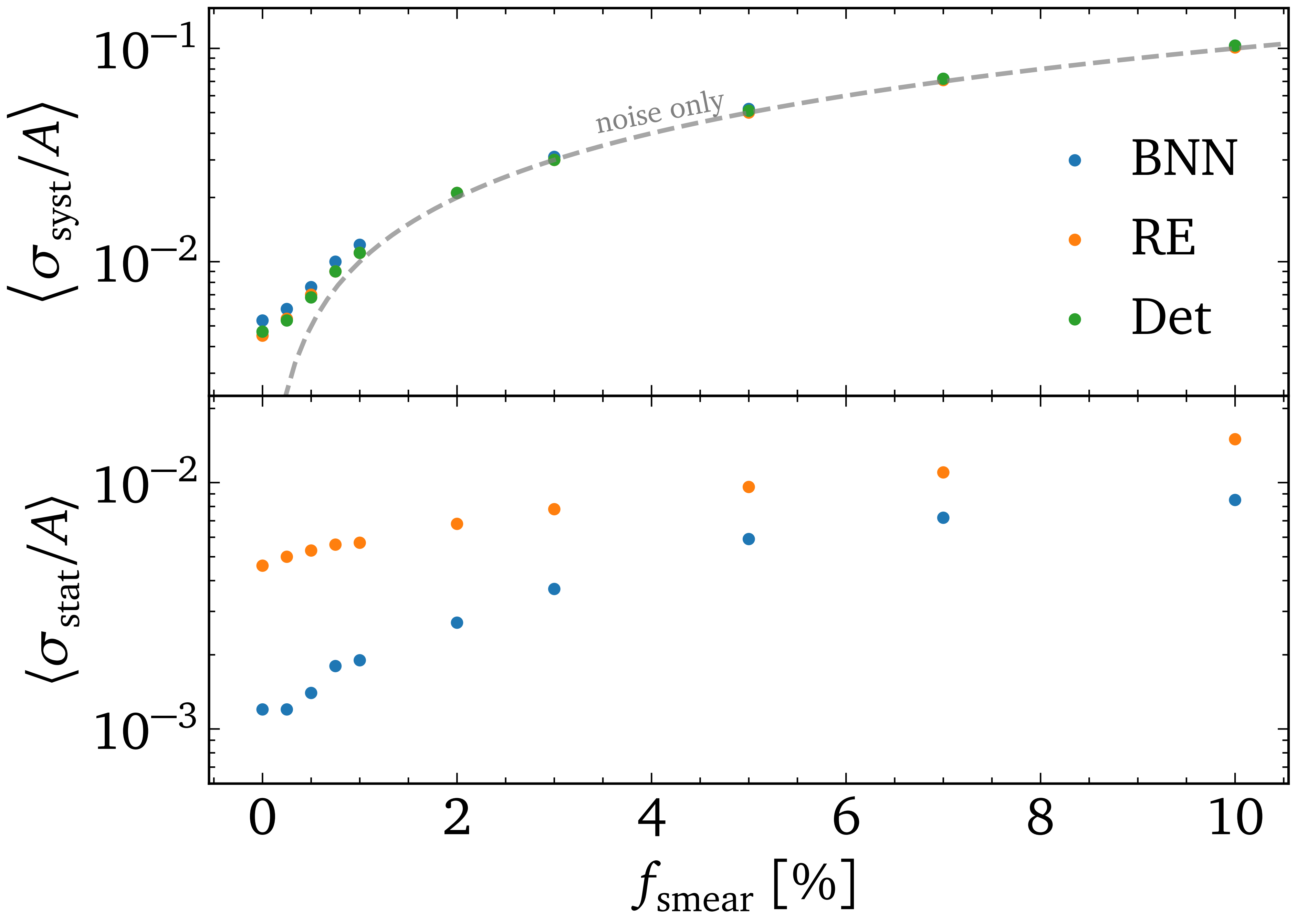


Adding Gaussian noise

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



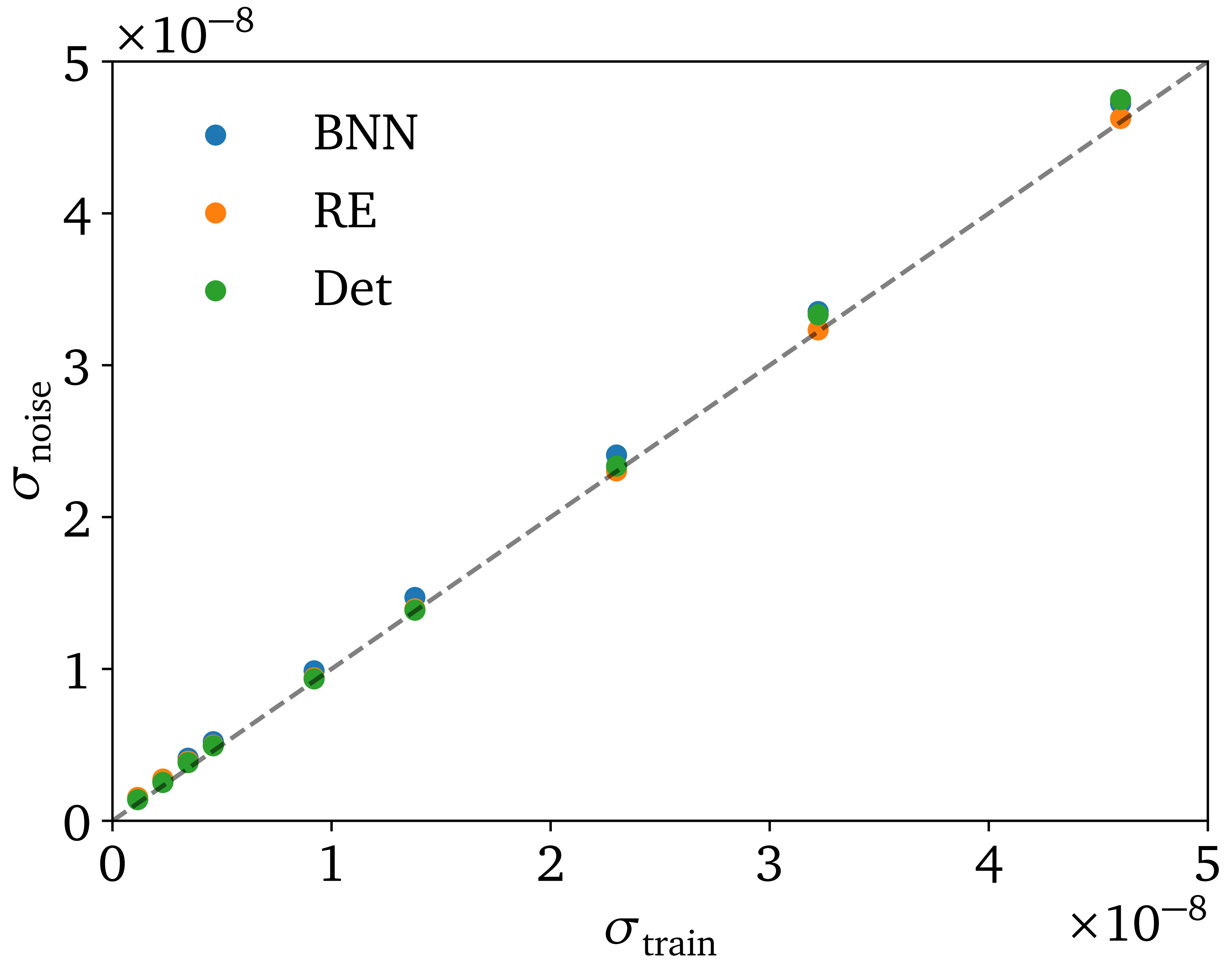
$$\sigma_{\text{train}} = f_{\text{smear}}\, A_{\text{true}}$$

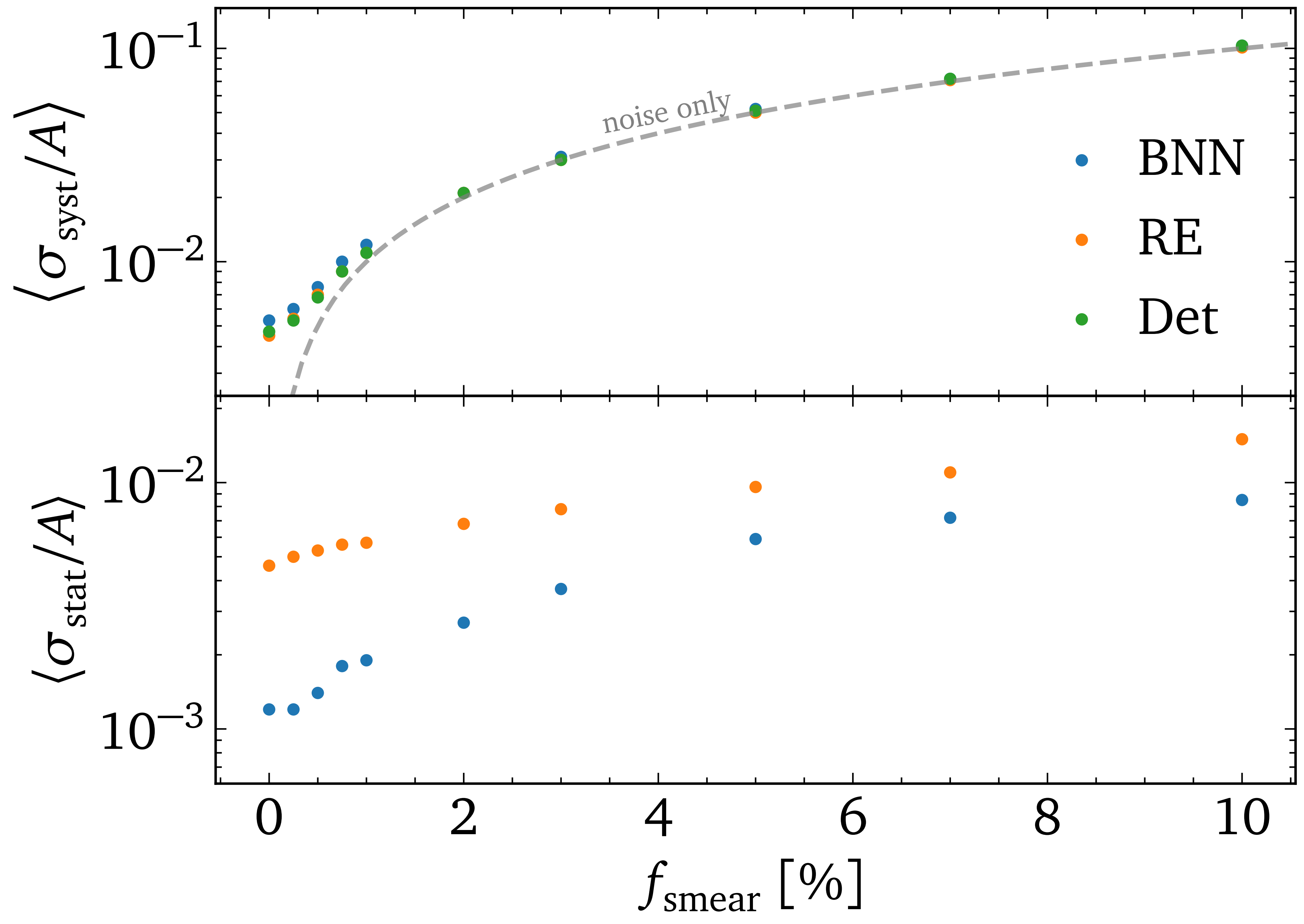


Adding Gaussian noise

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

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

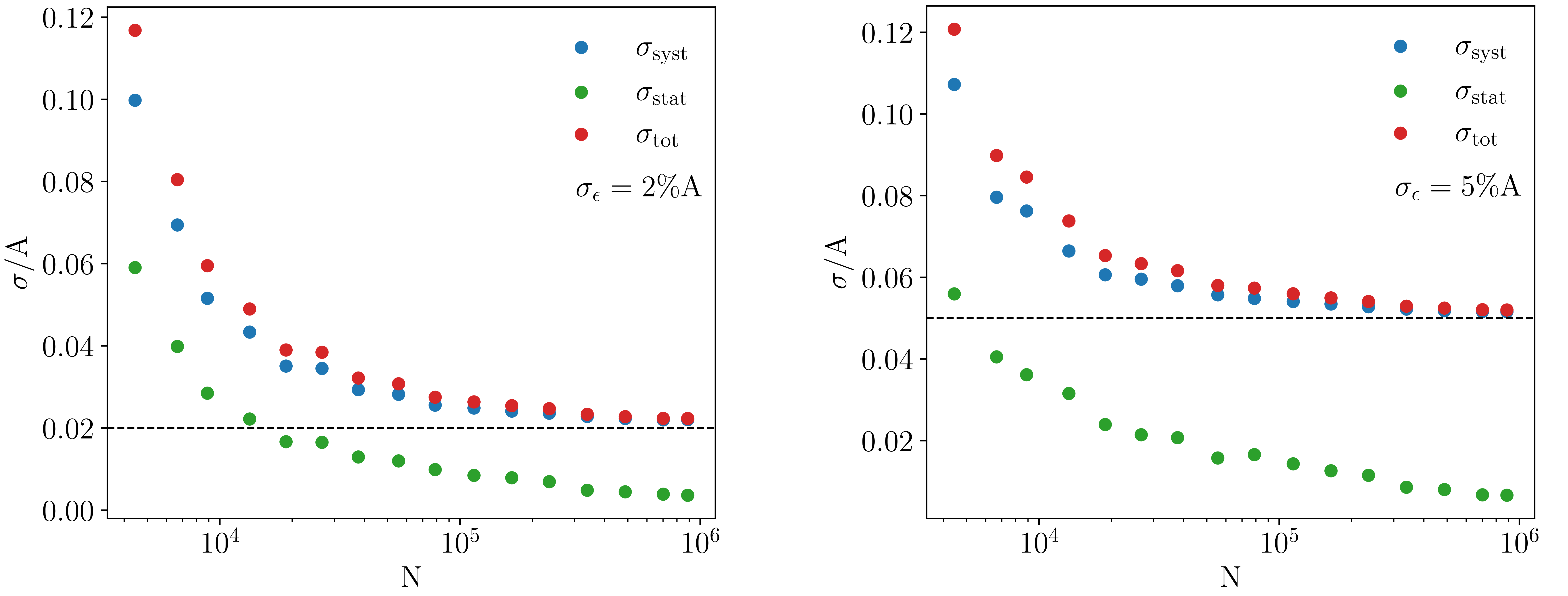




➡ Networks learn noise as **systematic** uncertainty

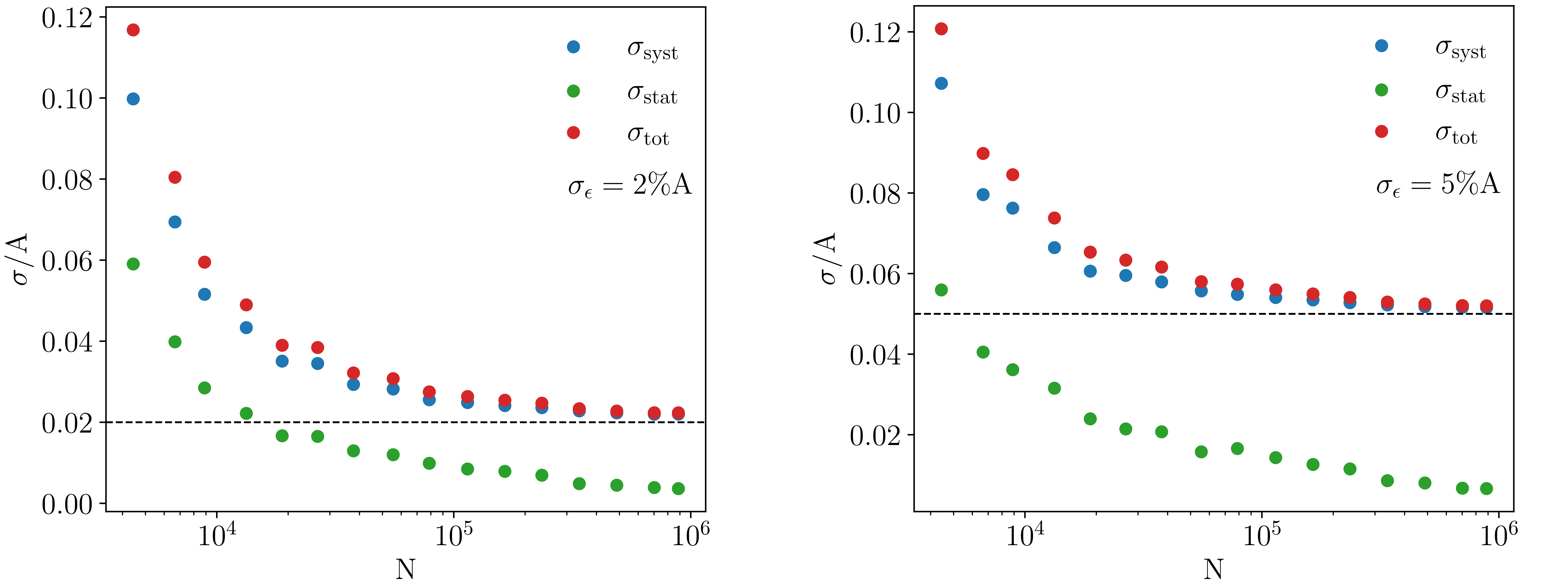
➡**Intrinsic uncertainty** of 0.4%

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

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

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

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

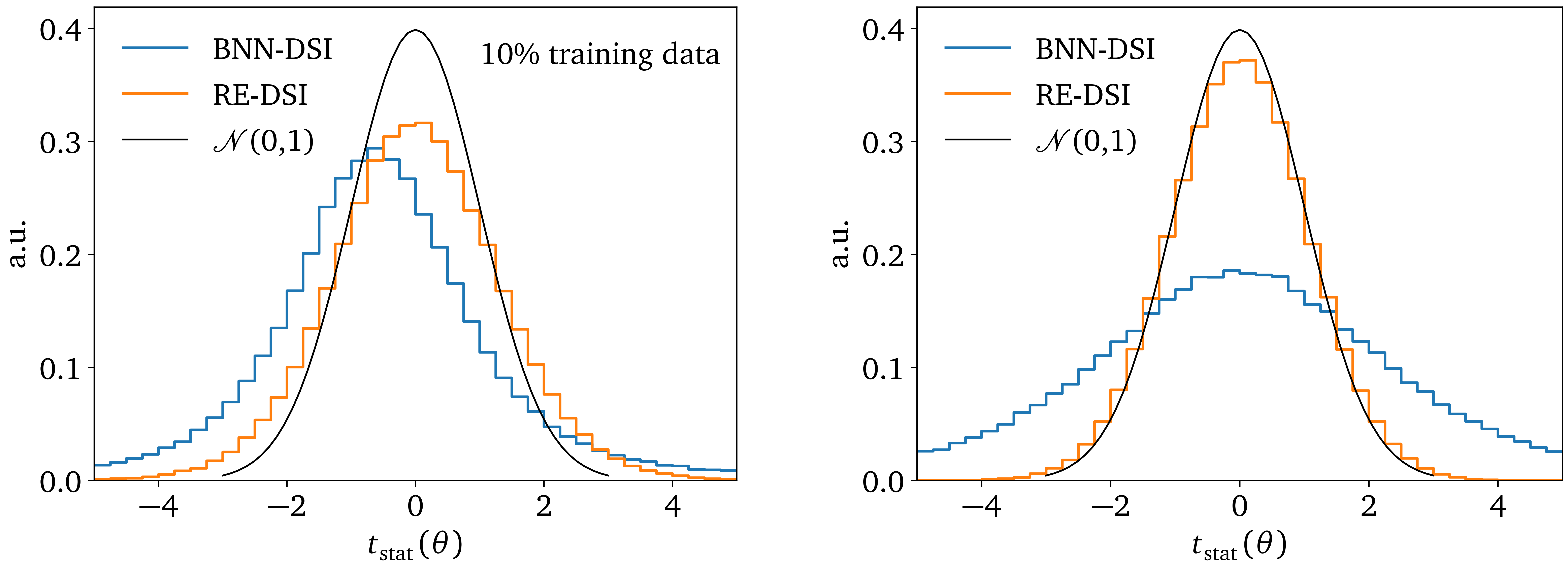
Correctly learned: follow **Gaussian**

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

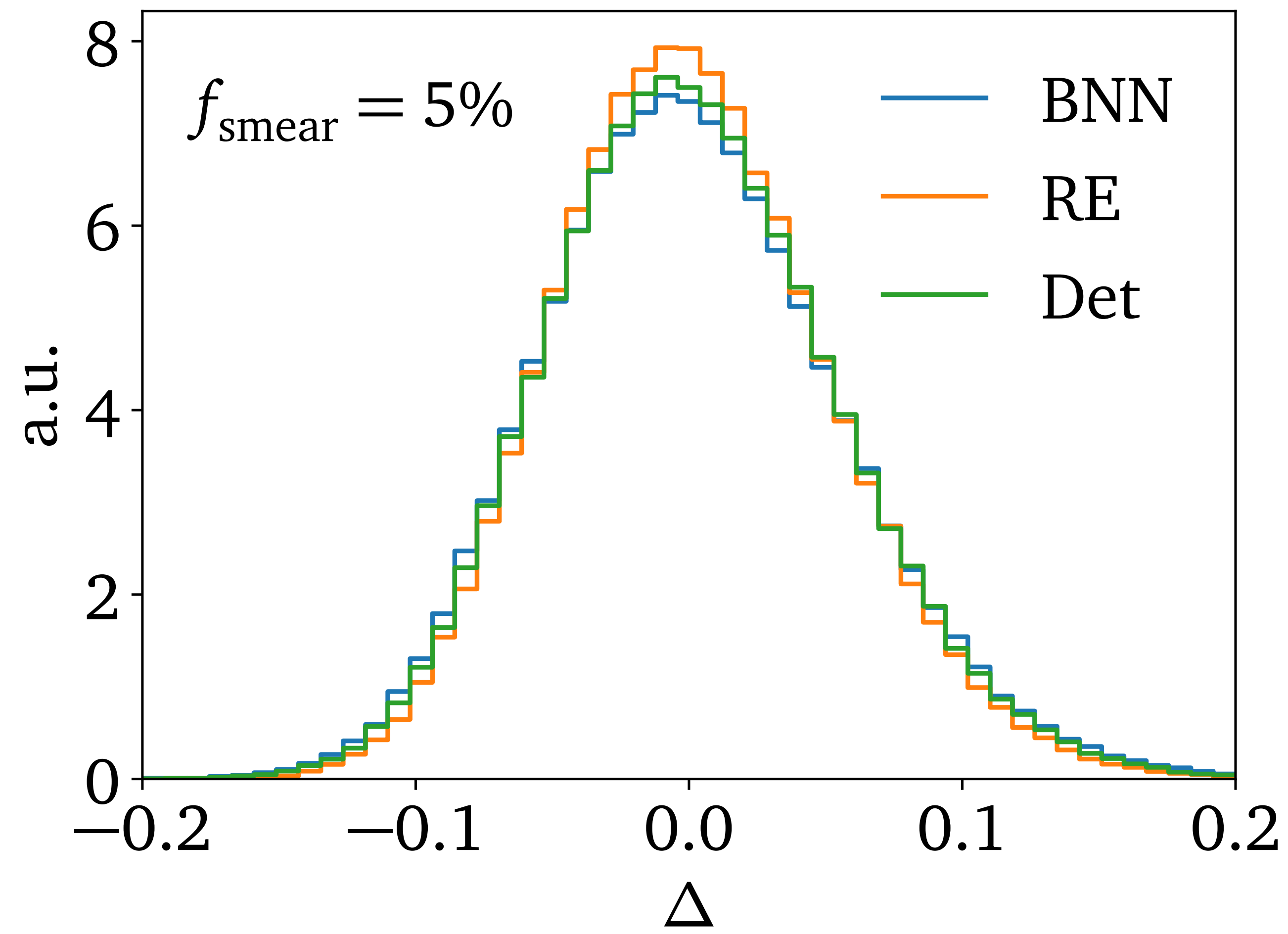
	70%	10%
$\langle \sigma_{\text{syst, BNN-DSI}}/A \rangle$	$8.7 \cdot 10^{-5}$	$2.5 \cdot 10^{-4}$
$\langle \sigma_{\text{stat, BNN-DSI}}/A \rangle$	$3.6 \cdot 10^{-5}$	$1.5 \cdot 10^{-4}$
$\langle \sigma_{\text{syst, RE-DSI}}/A \rangle$	$5.1 \cdot 10^{-5}$	$2.9 \cdot 10^{-4}$
$\langle \sigma_{\text{stat,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

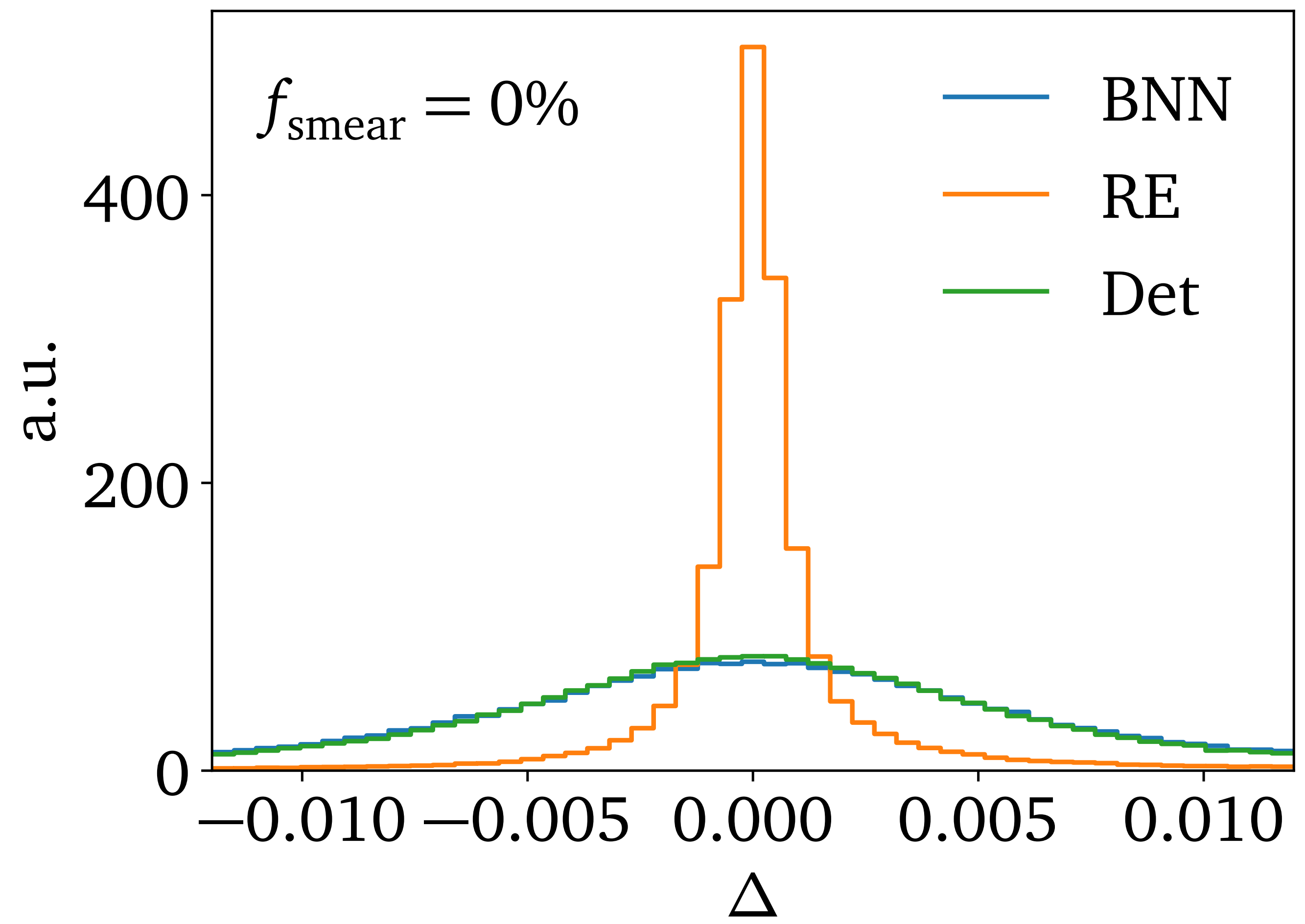
Statistical pull



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

Systematic pull and accuracy

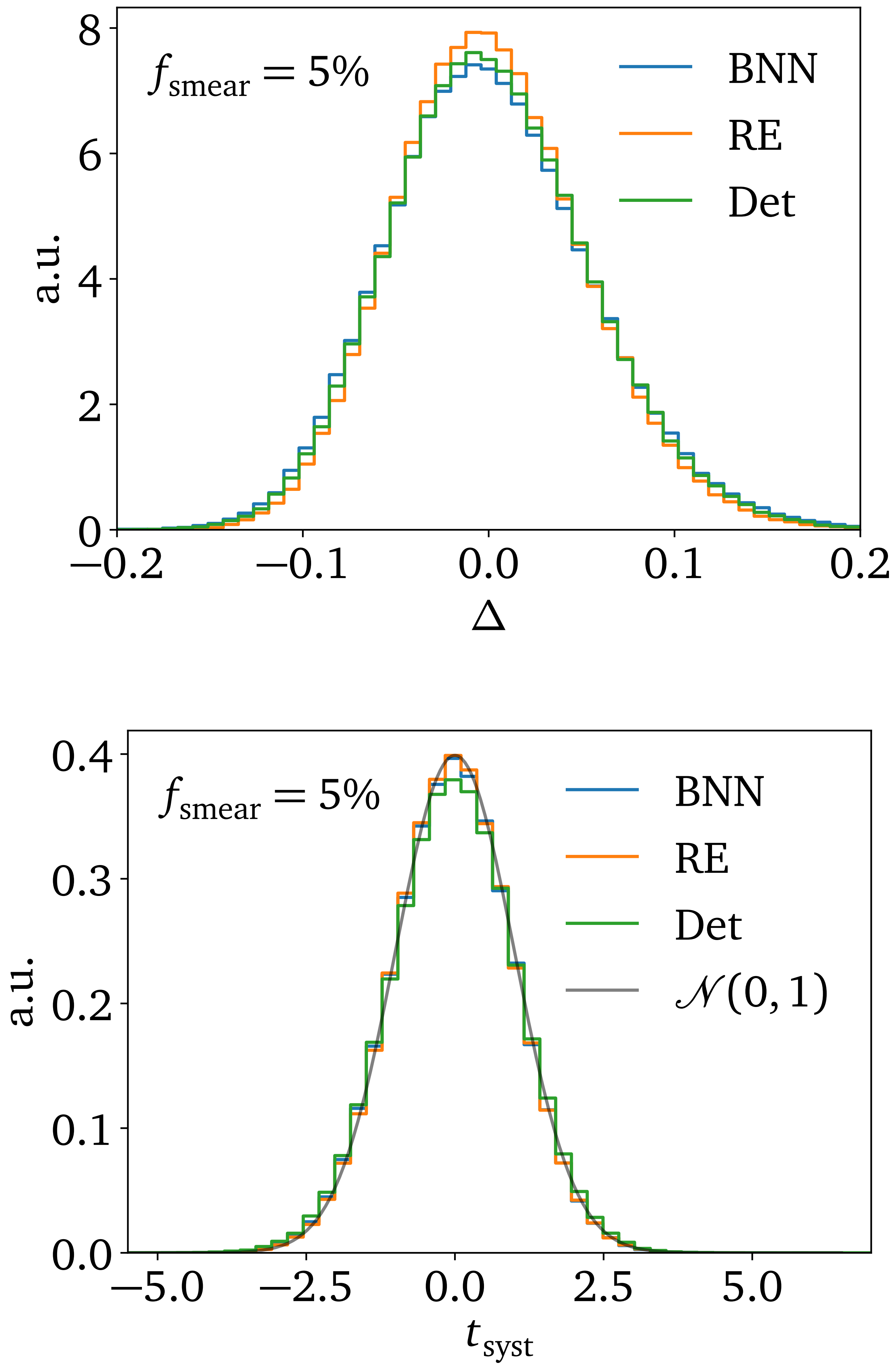




➡Calibrated results

➡BNN: Advantage for learning **systematics**

Systematic pull and accuracy

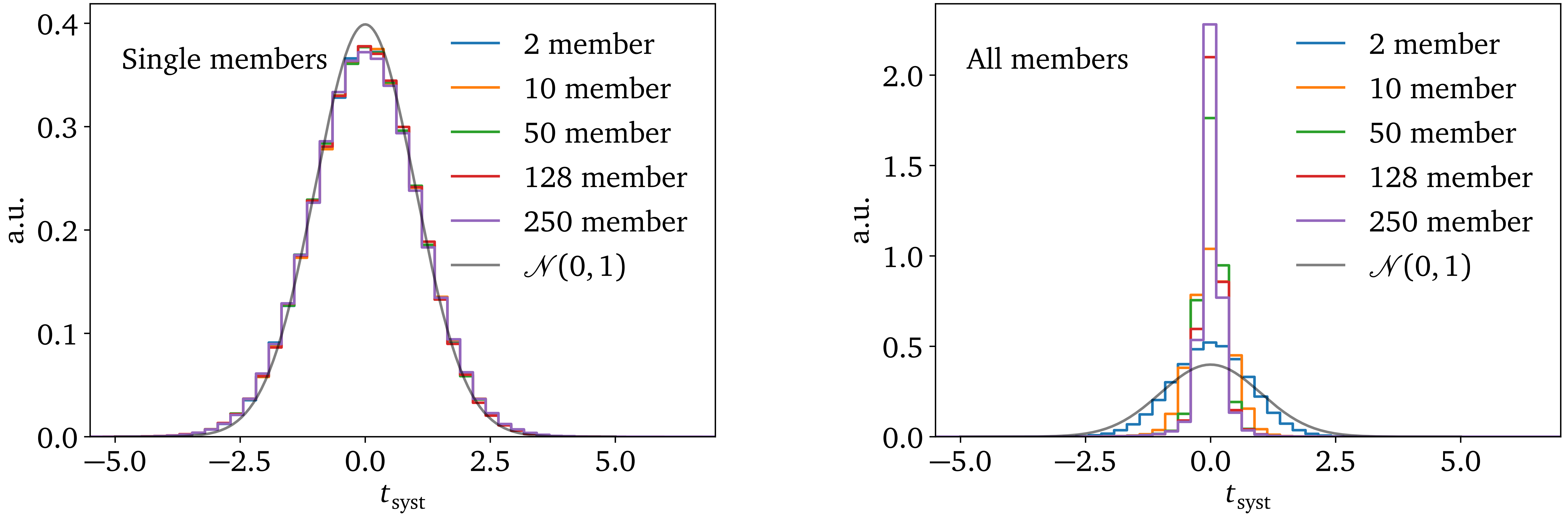


➡Calibrated results

➡BNN: Advantage for learning **systematics**

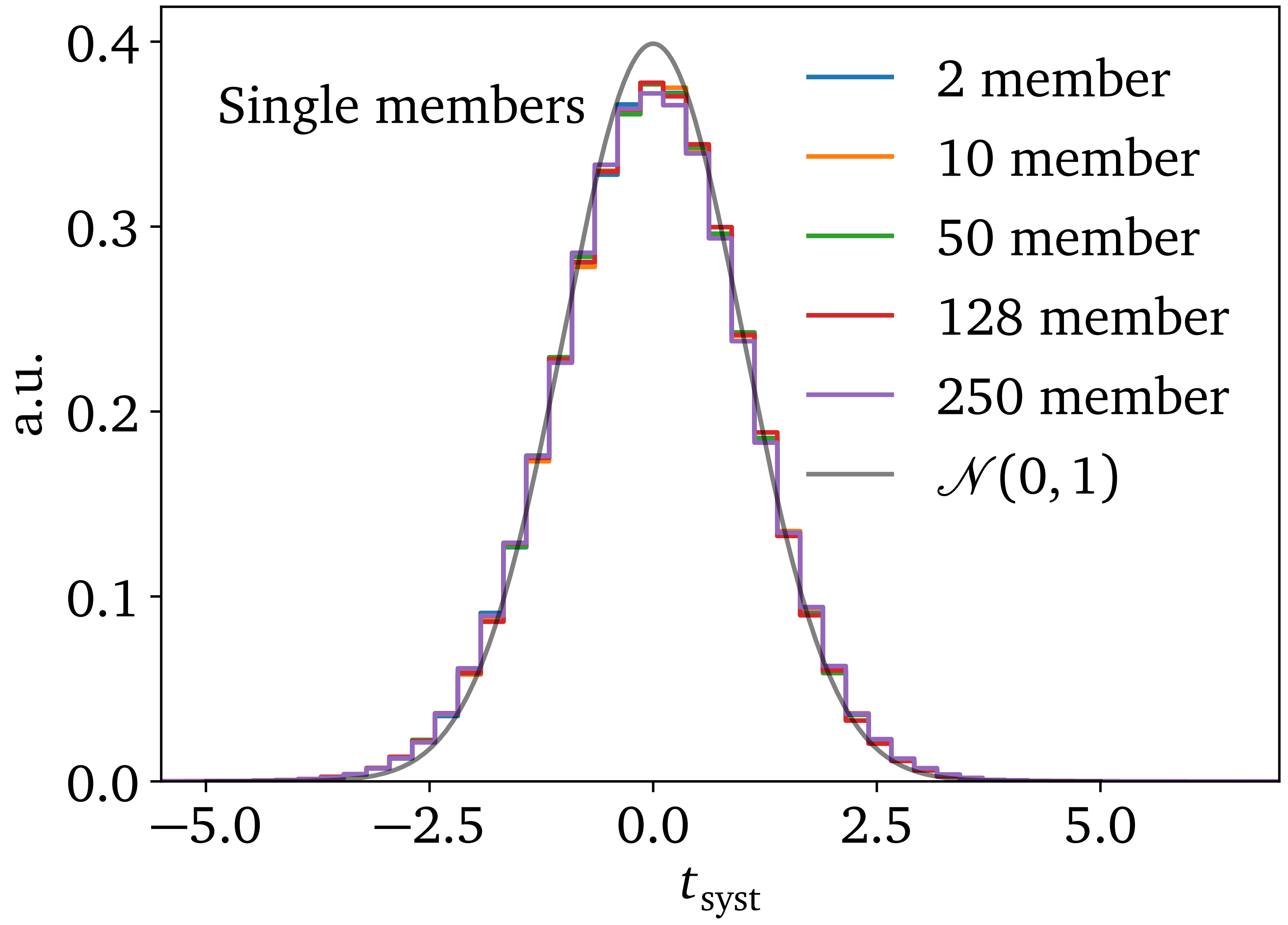
Systematic pull for REs

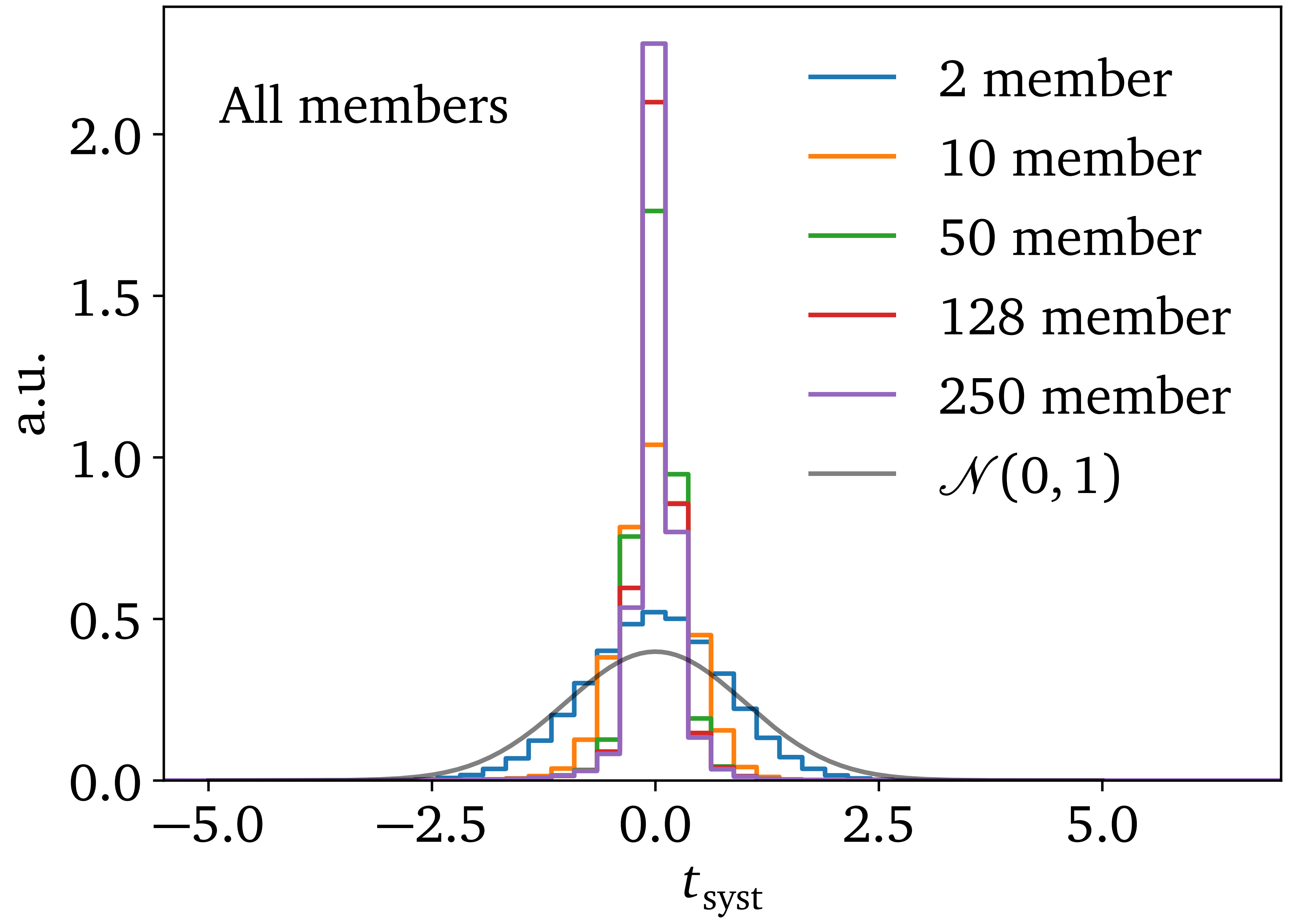
- Learned $\sigma_{syst}(x)$ too conservative



Systematic pull for REs

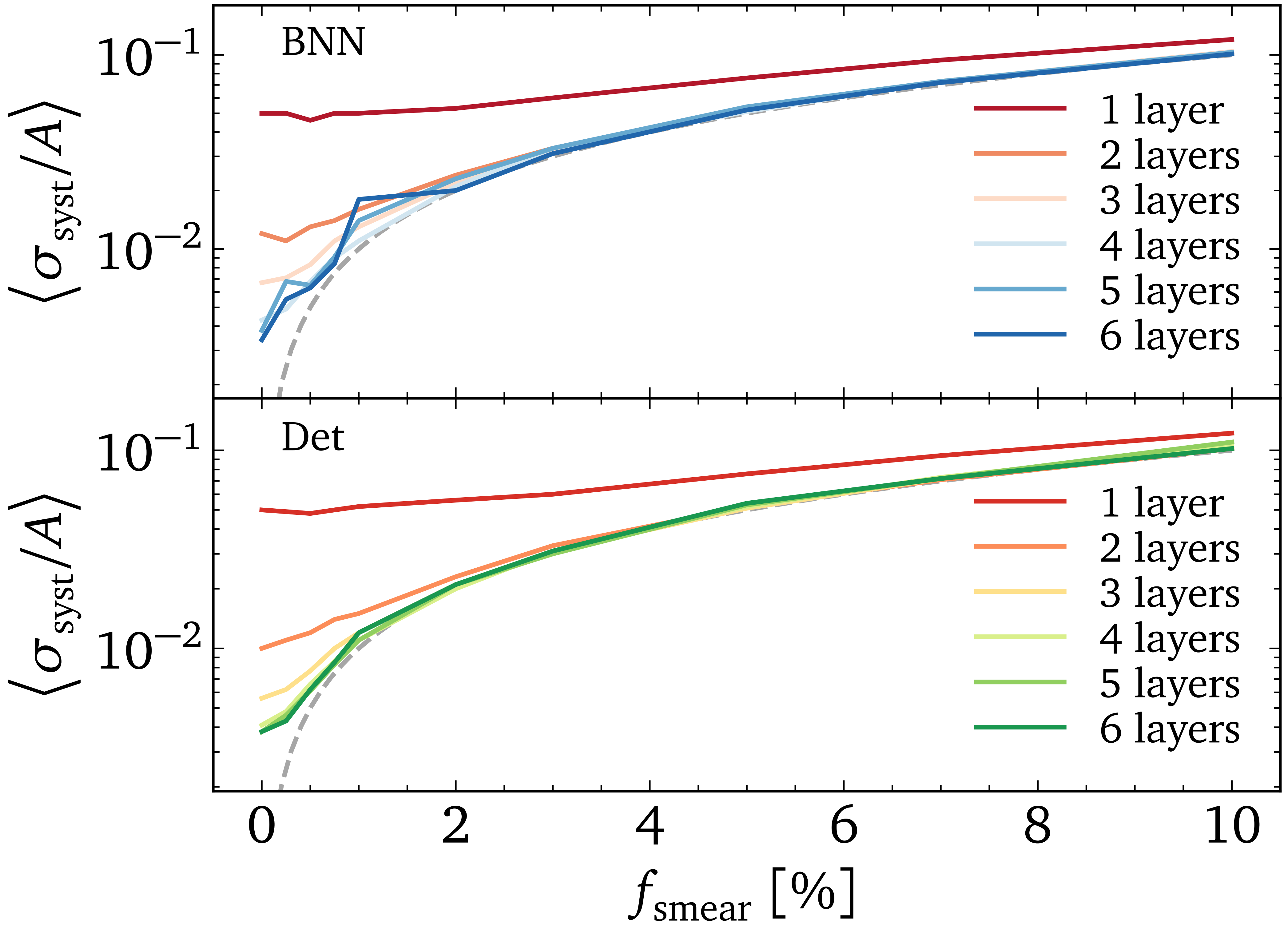
- Learned $\sigma_{syst}(x)$ too conservative





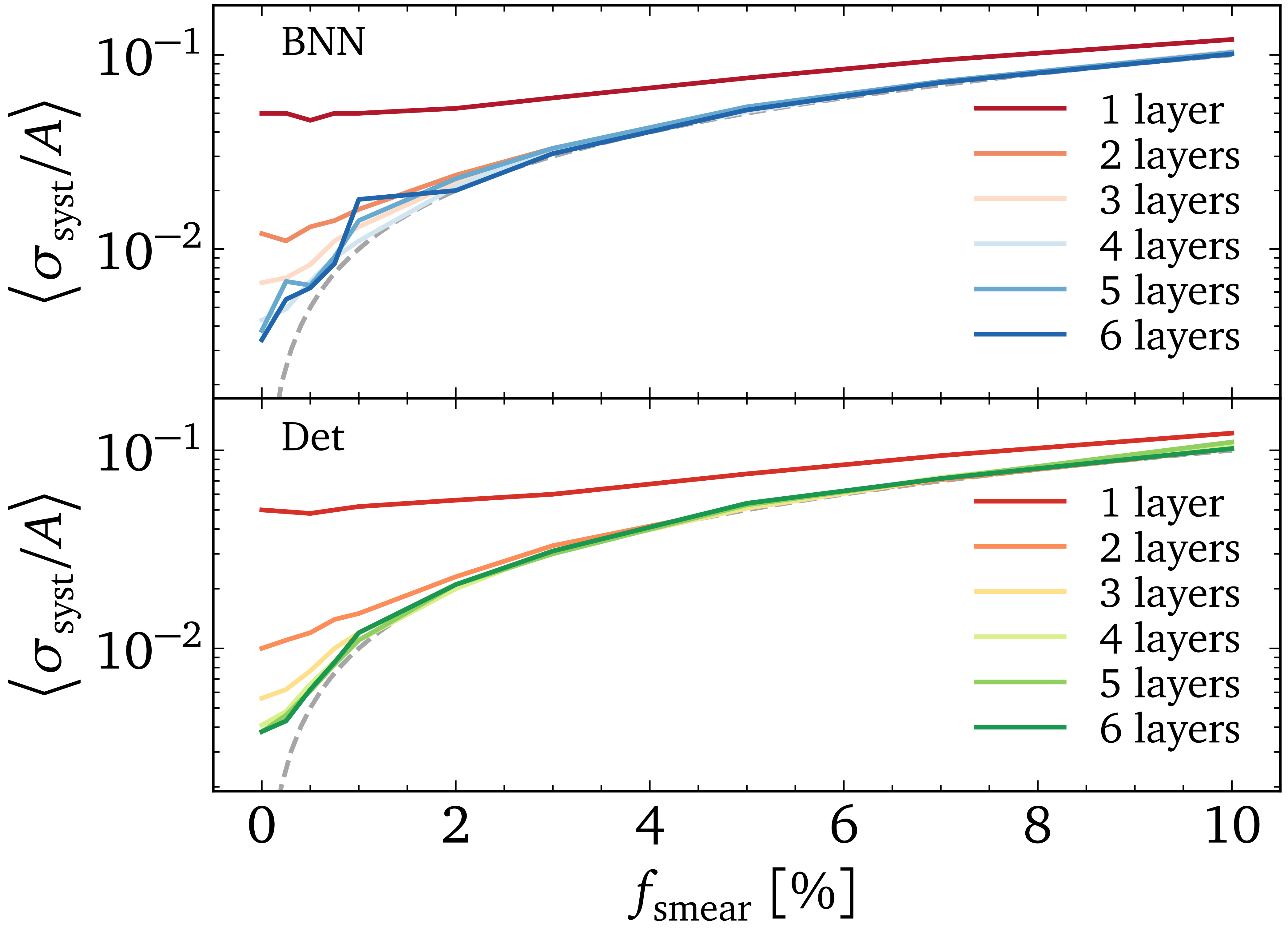
➡ Prediction benefits from ensemble nature but not σ_{syst}

Improve network expressivity



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

Improve network expressivity

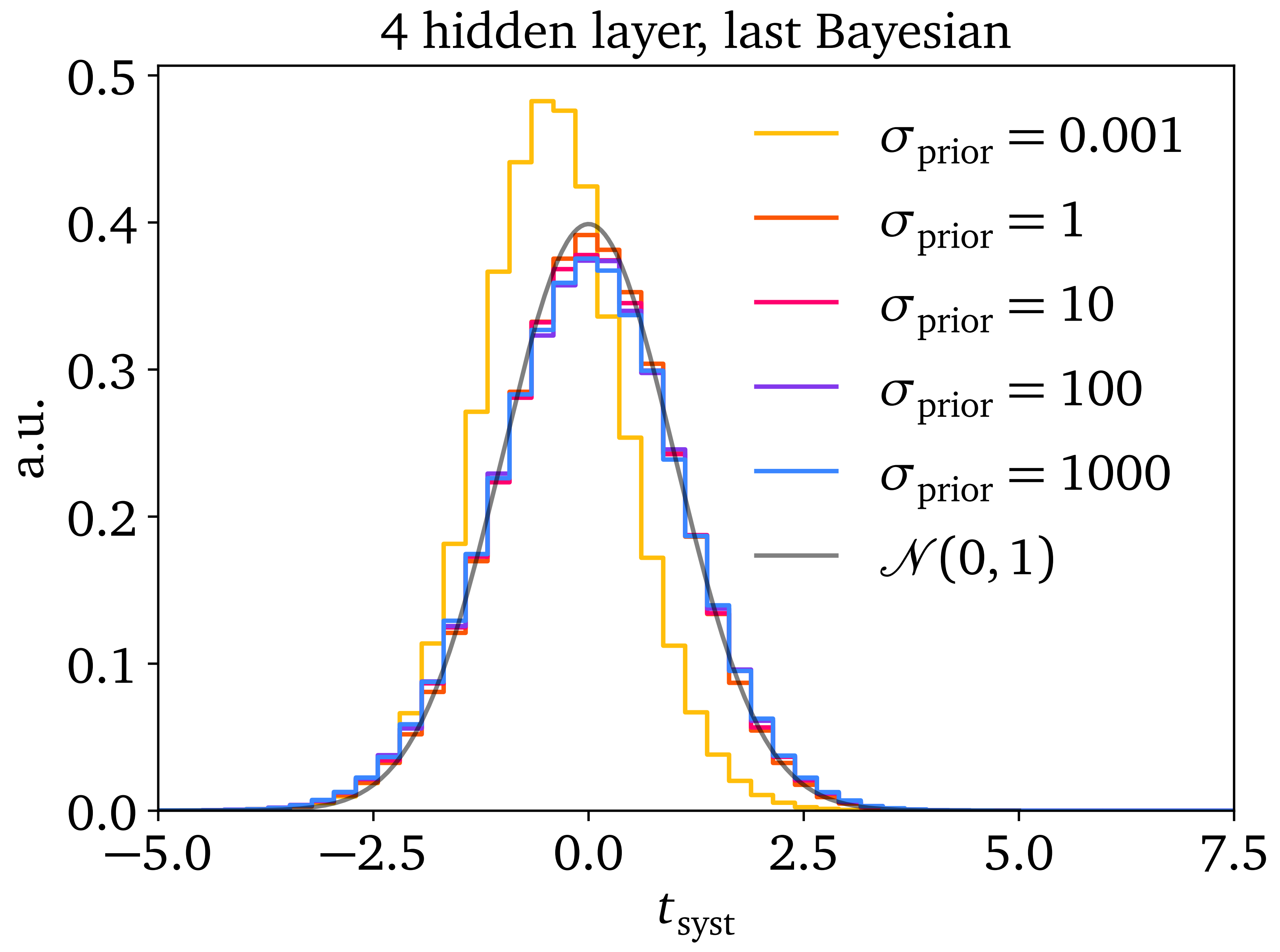


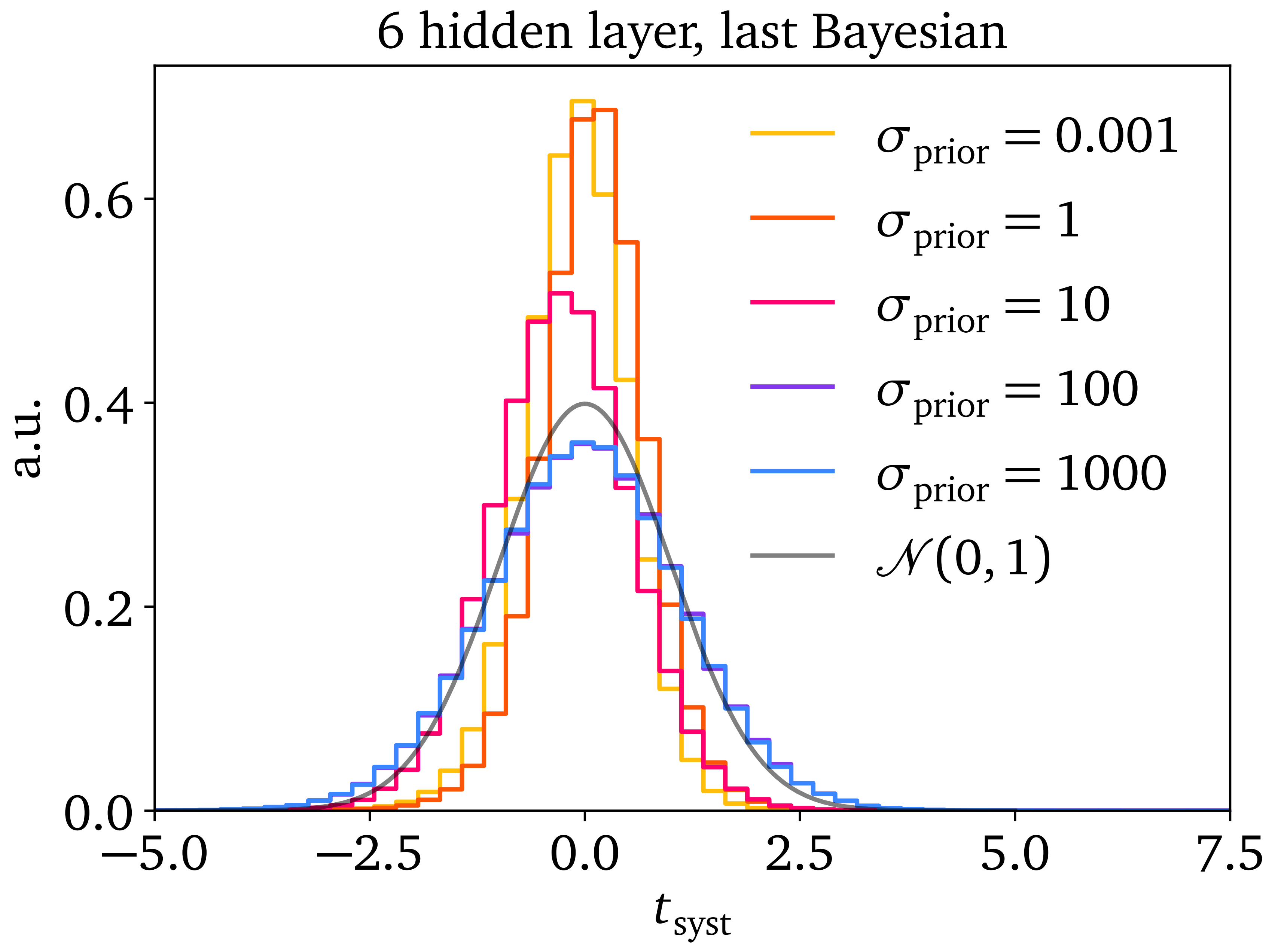
- Source of intrinsic uncertainty?
- BNN: Only last layer Bayesian

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

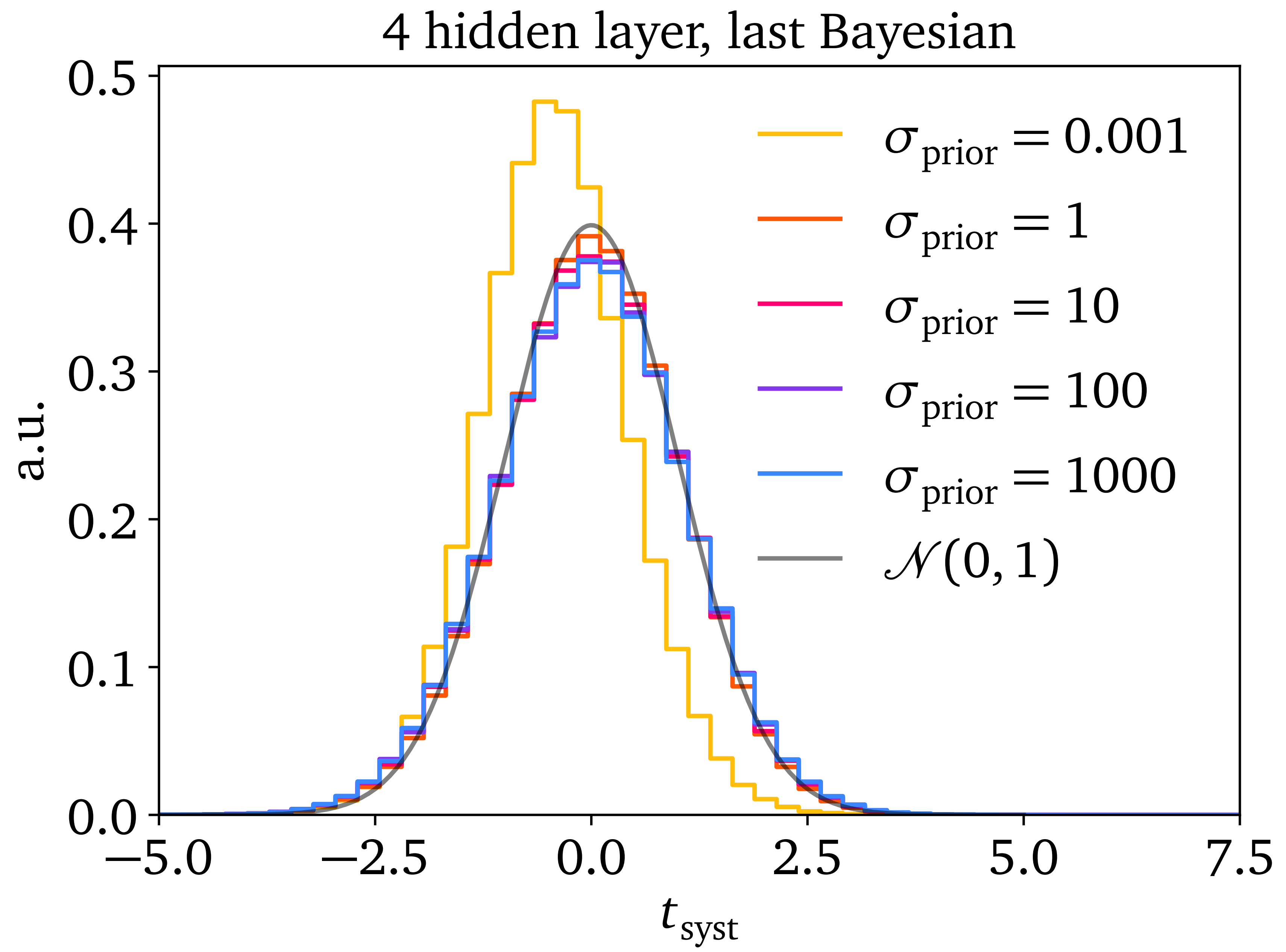
➡ Improvement of intrinsic uncertainty to 0.3%

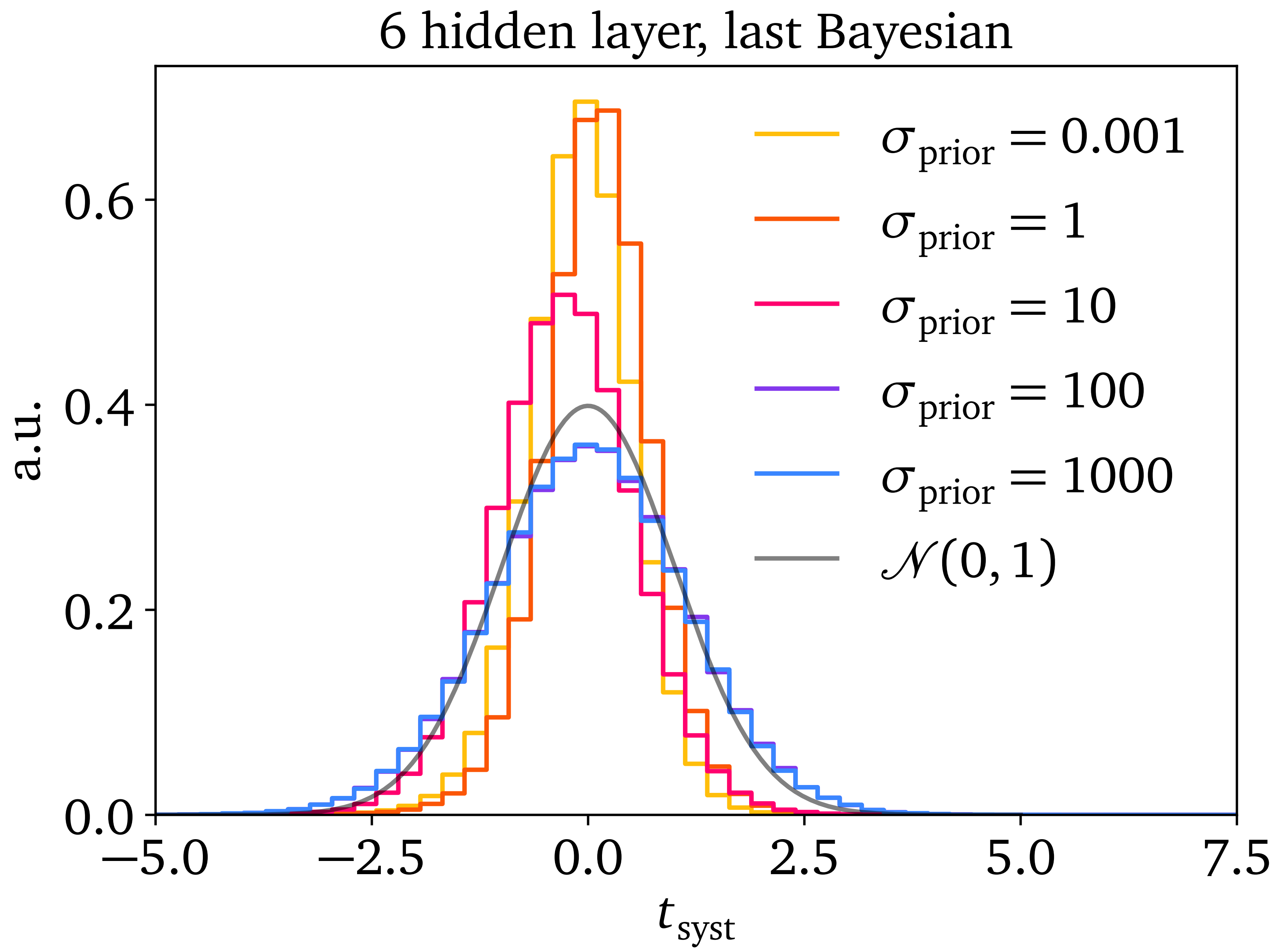
Prior influence in the BNN





Prior influence in the BNN





➡ Results don't depend on prior

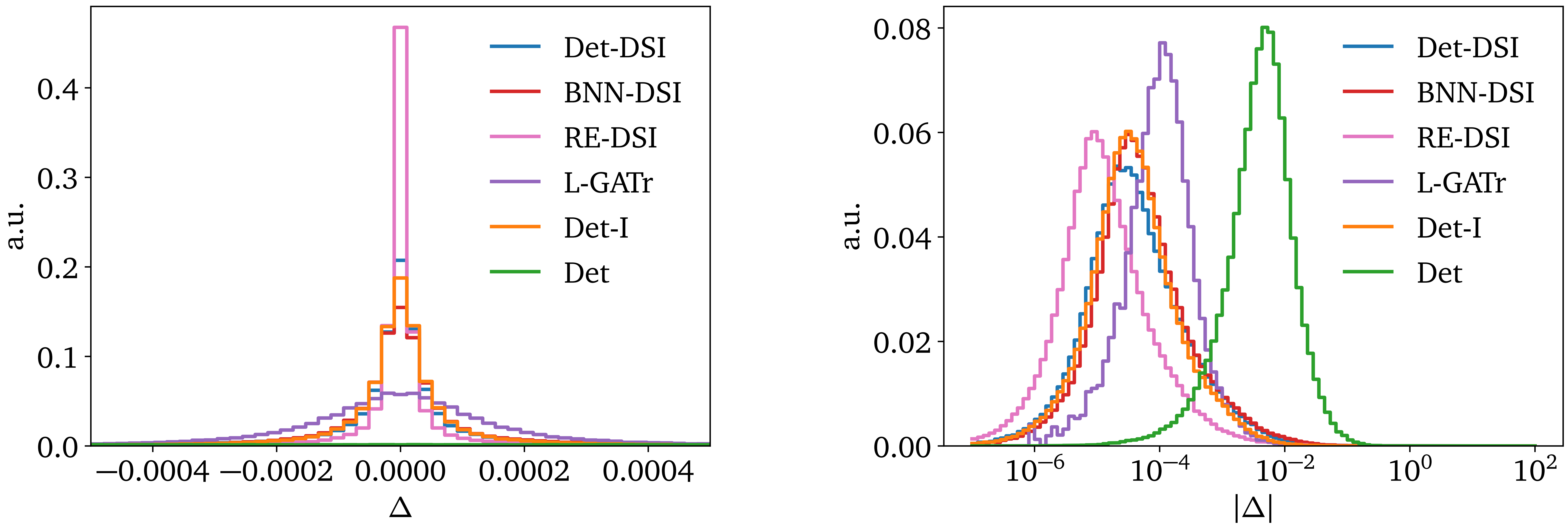
➡ For more layers a larger prior is needed

Testing advanced architectures

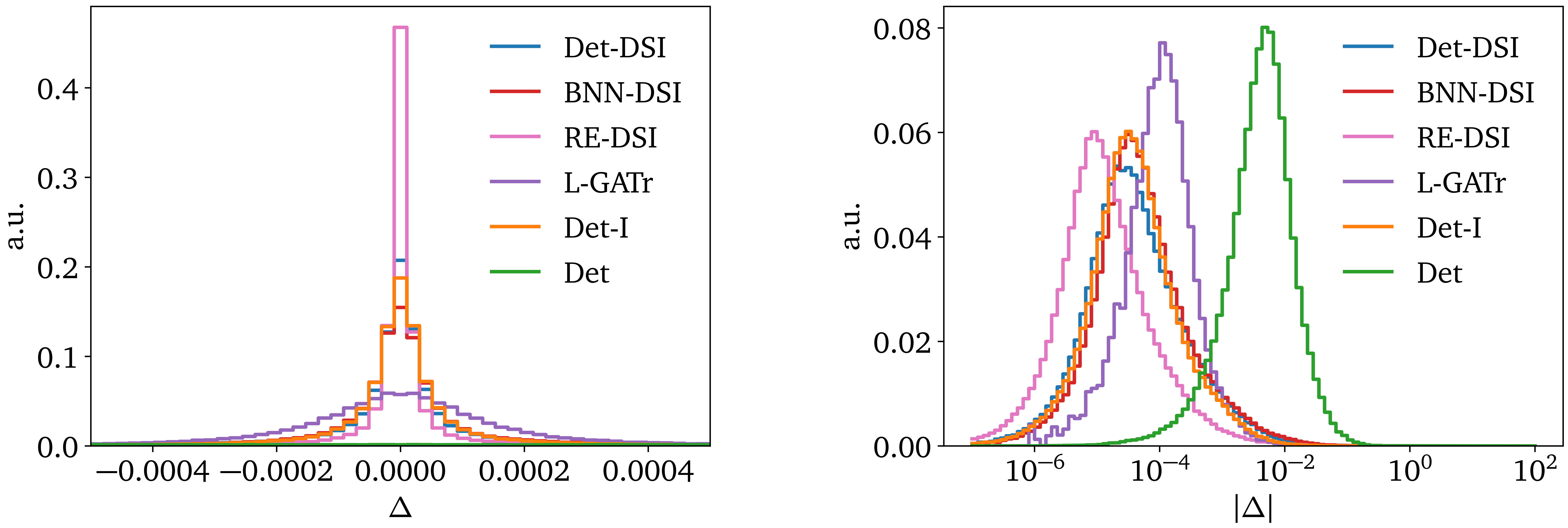
- Enhance standard networks through representation learning:

  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

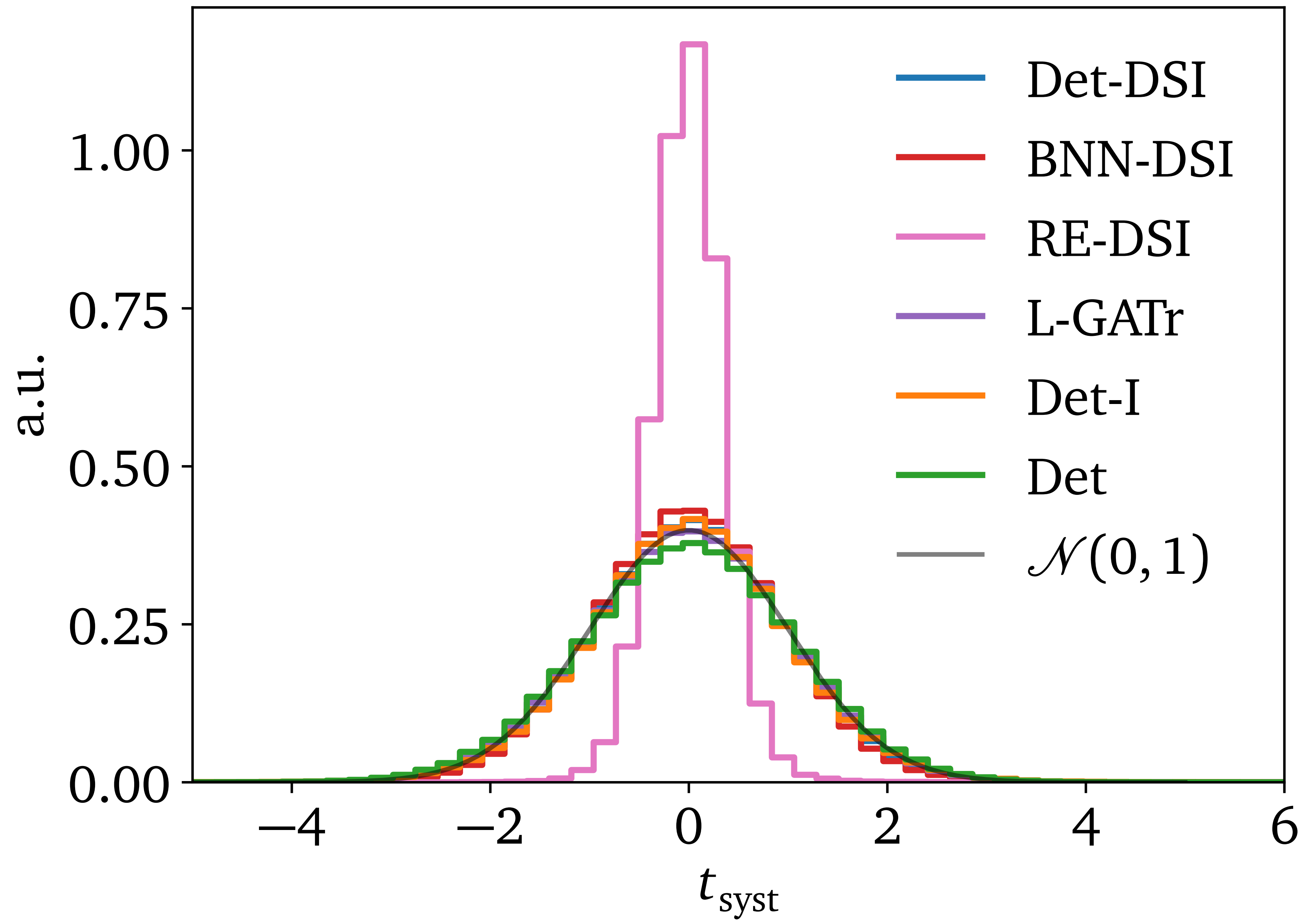


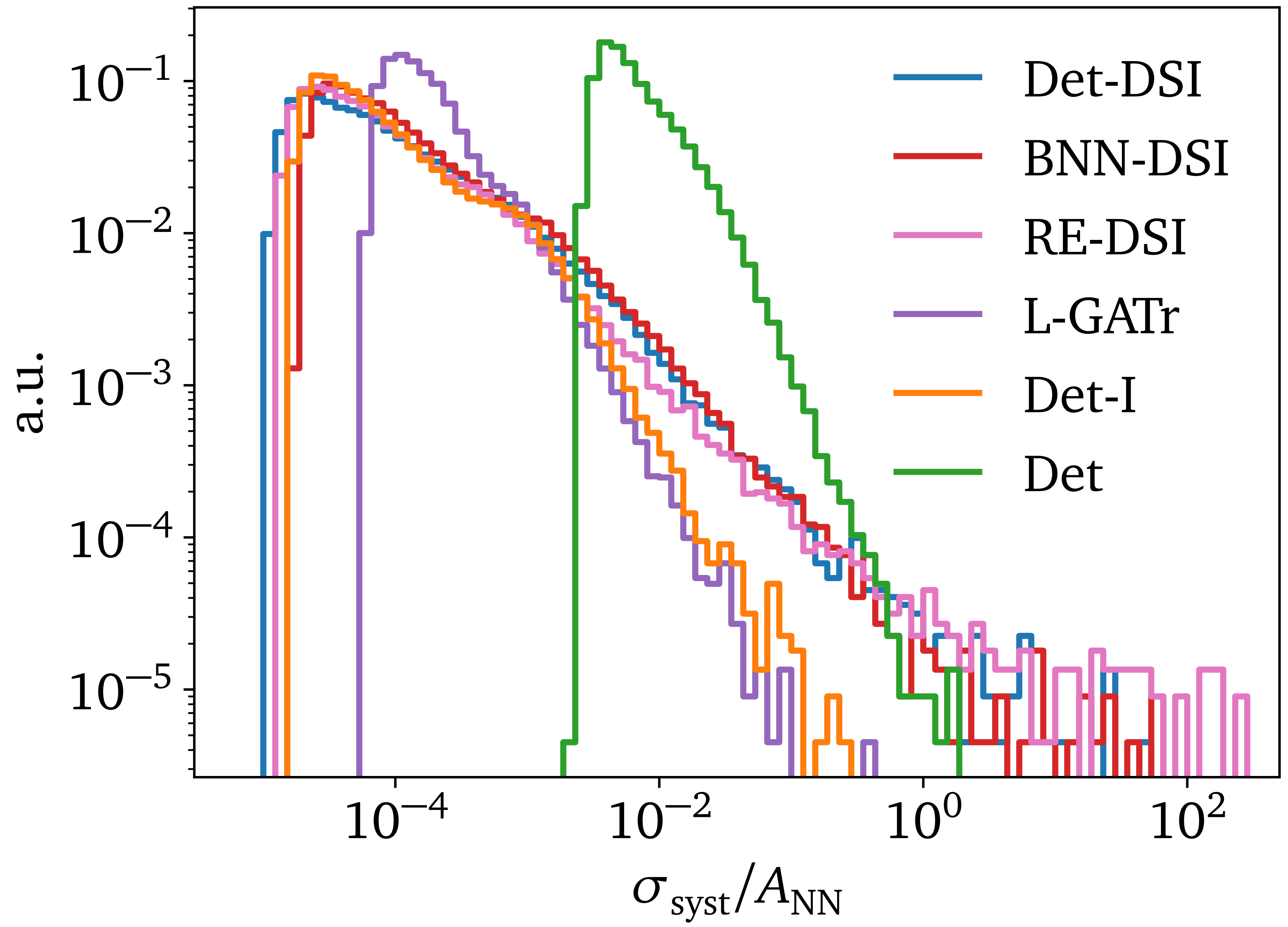
Advanced architectures - Accuracy



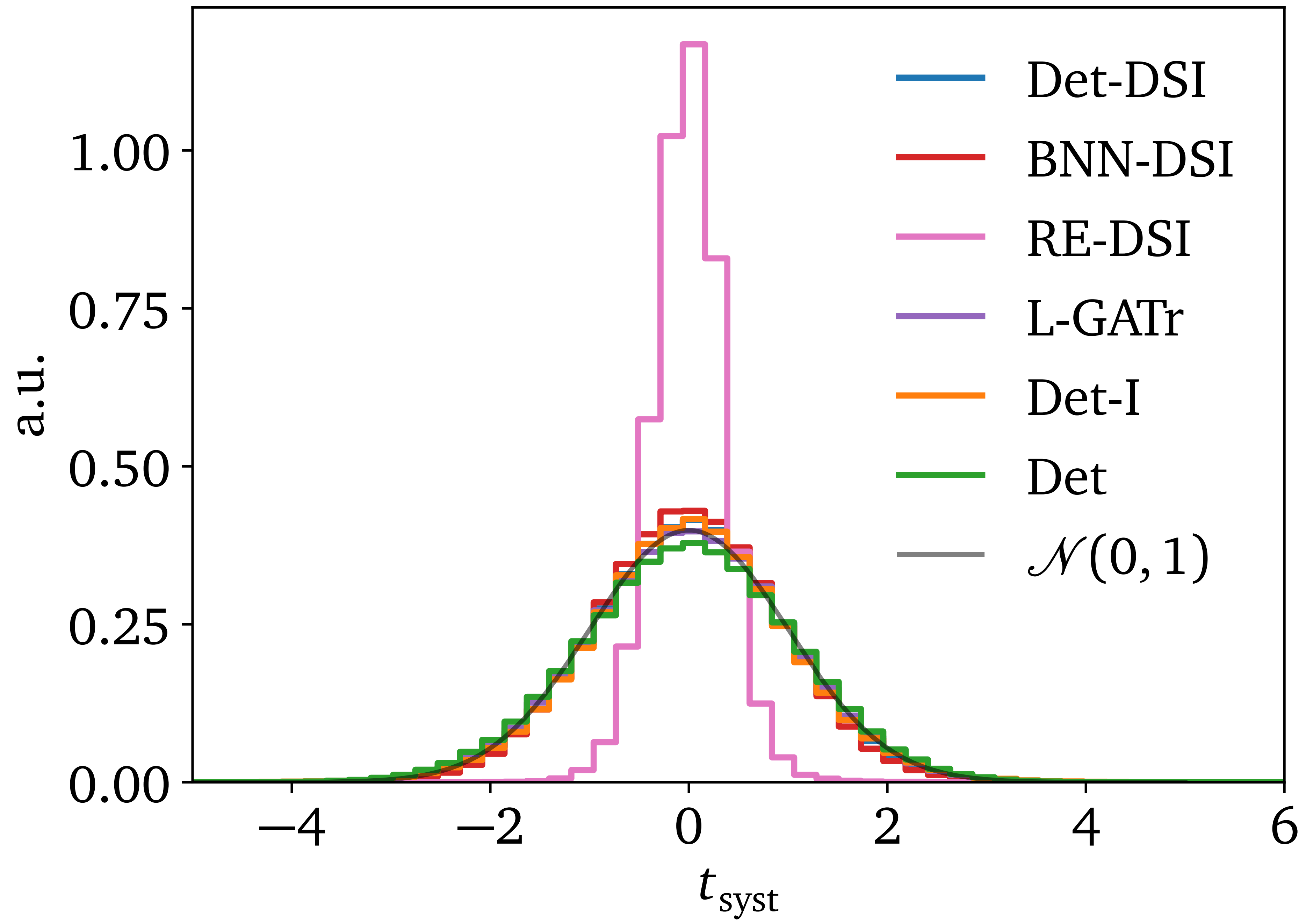
➡**Controlled accuracy** to 10^{-5} level

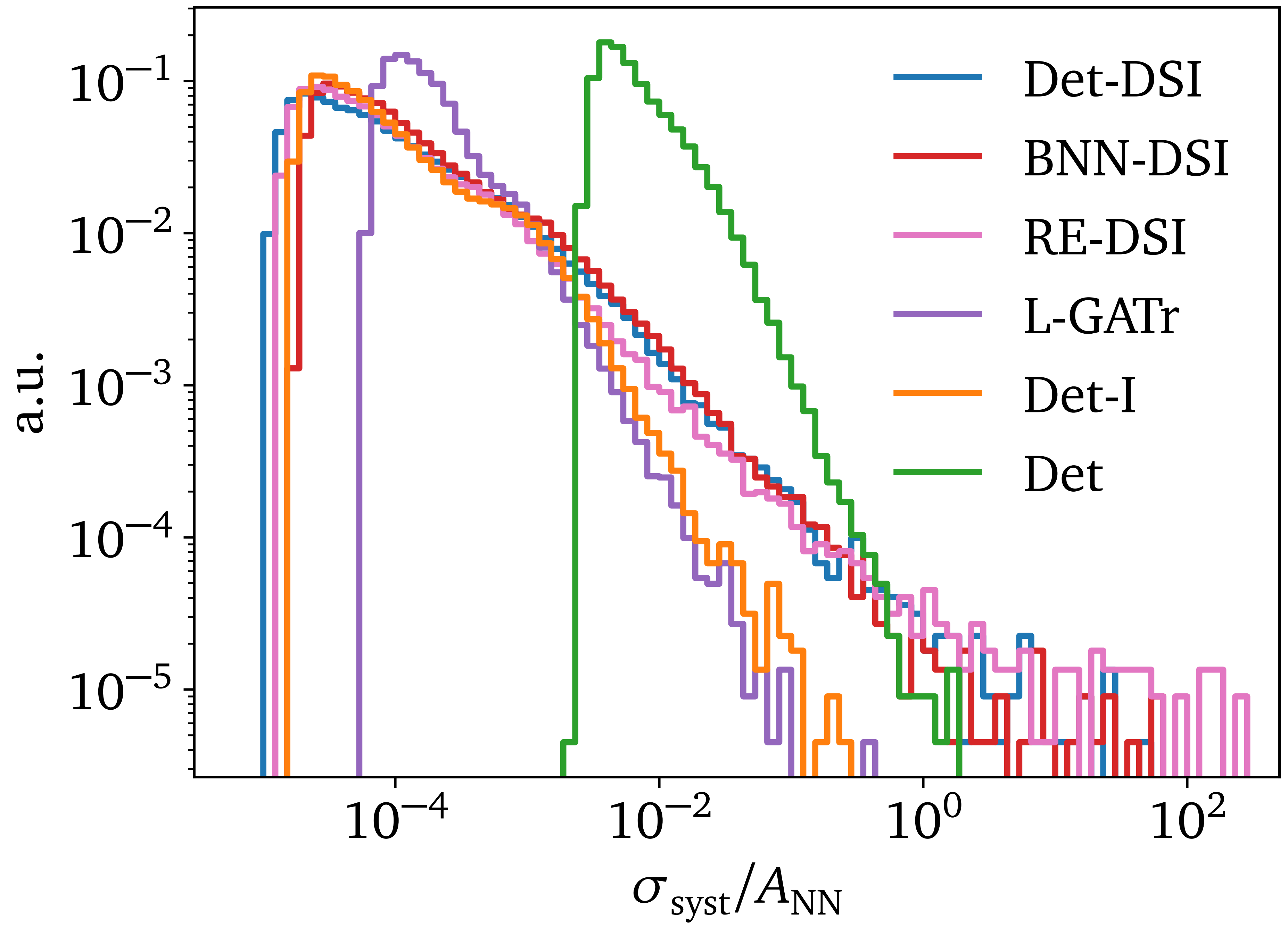
Advanced architectures - Uncertainties





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

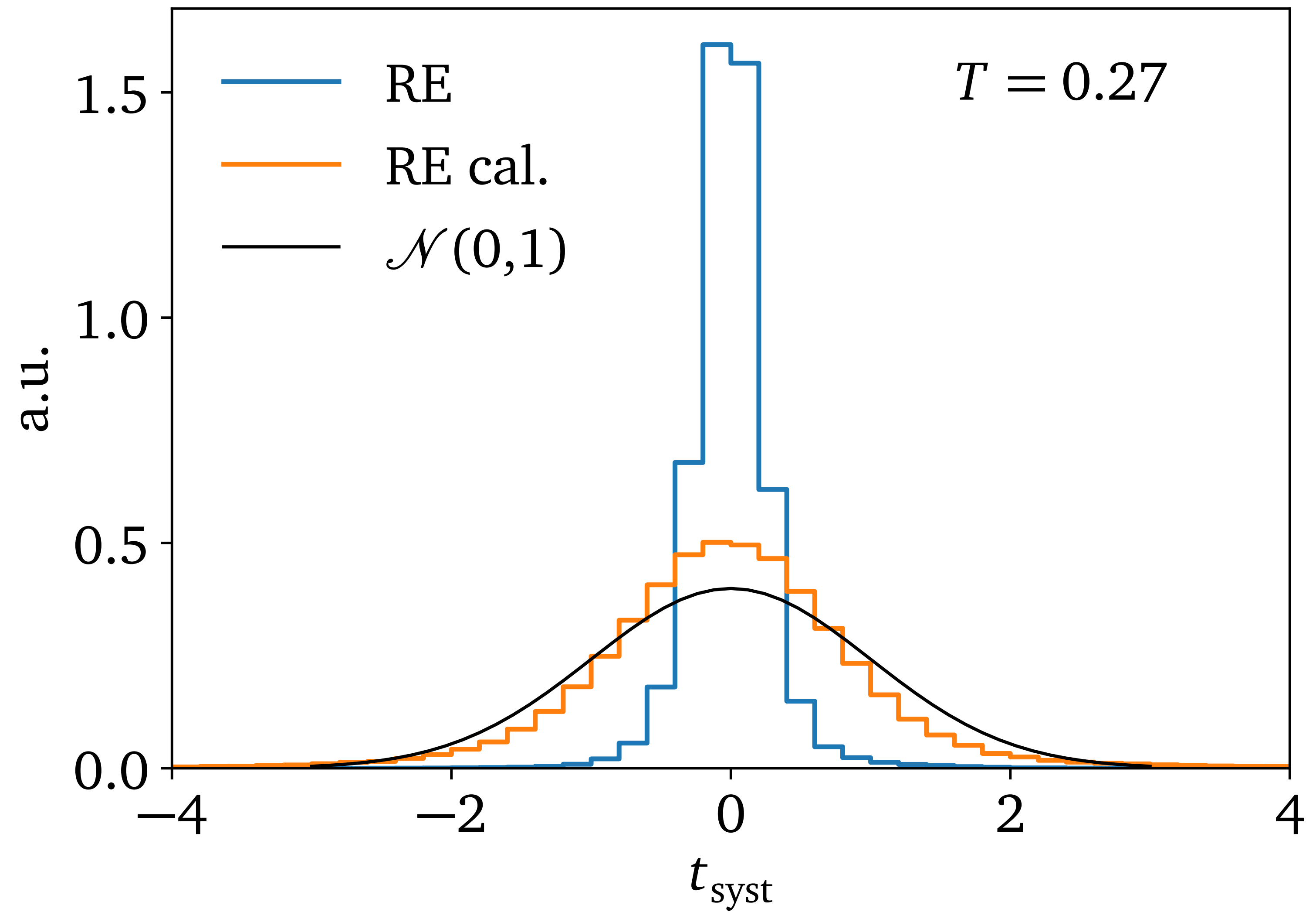
1. Able to track systematic and statistical uncertainties
2. Networks are calibrated (if not: calibration possible)
3. RE benefits from ensemble nature in precision
4. Networks are able to give controlled accuracy on 10^{-5} level

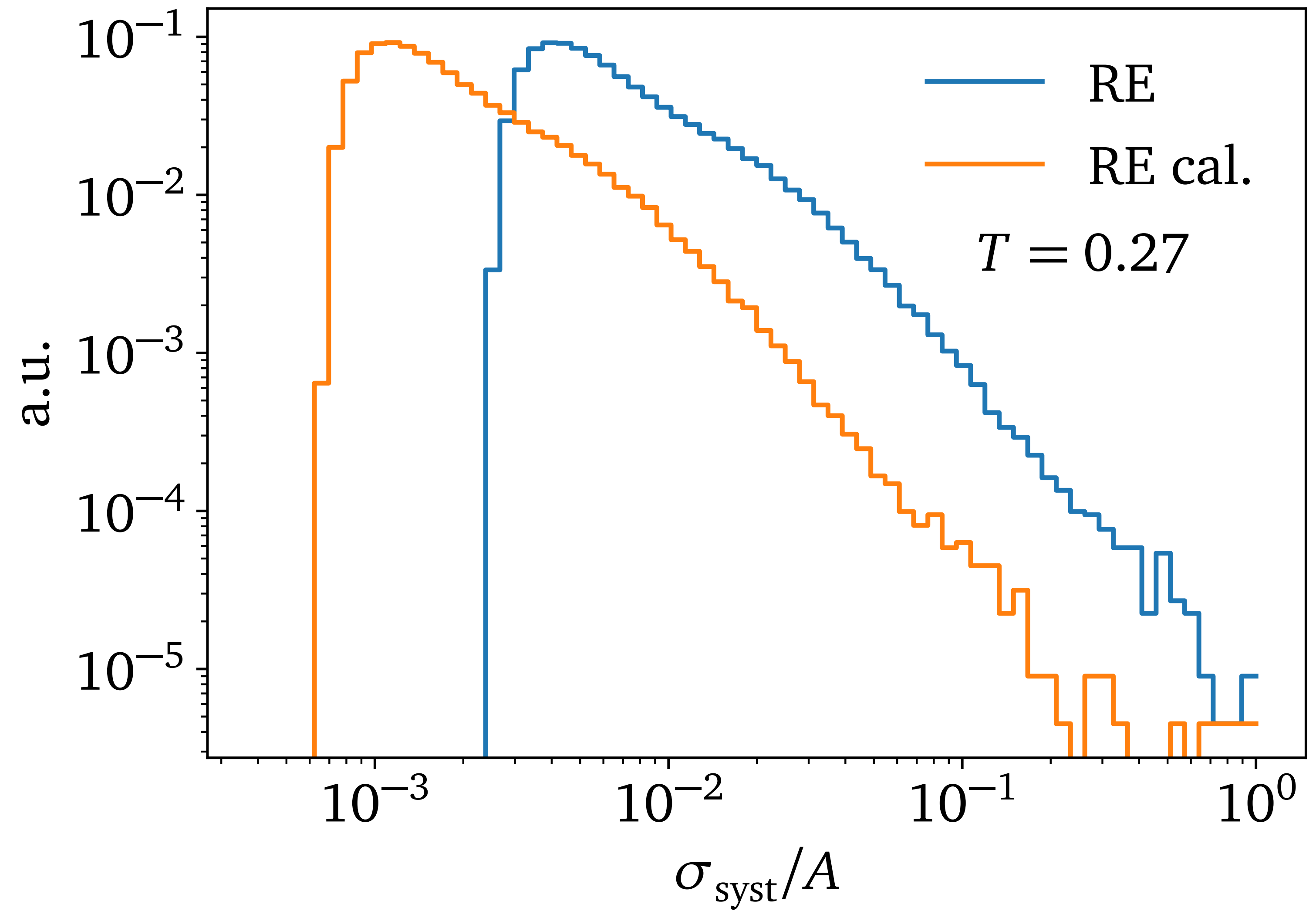
Thank you for your attention!

Back up / Additional material

Calibration of networks

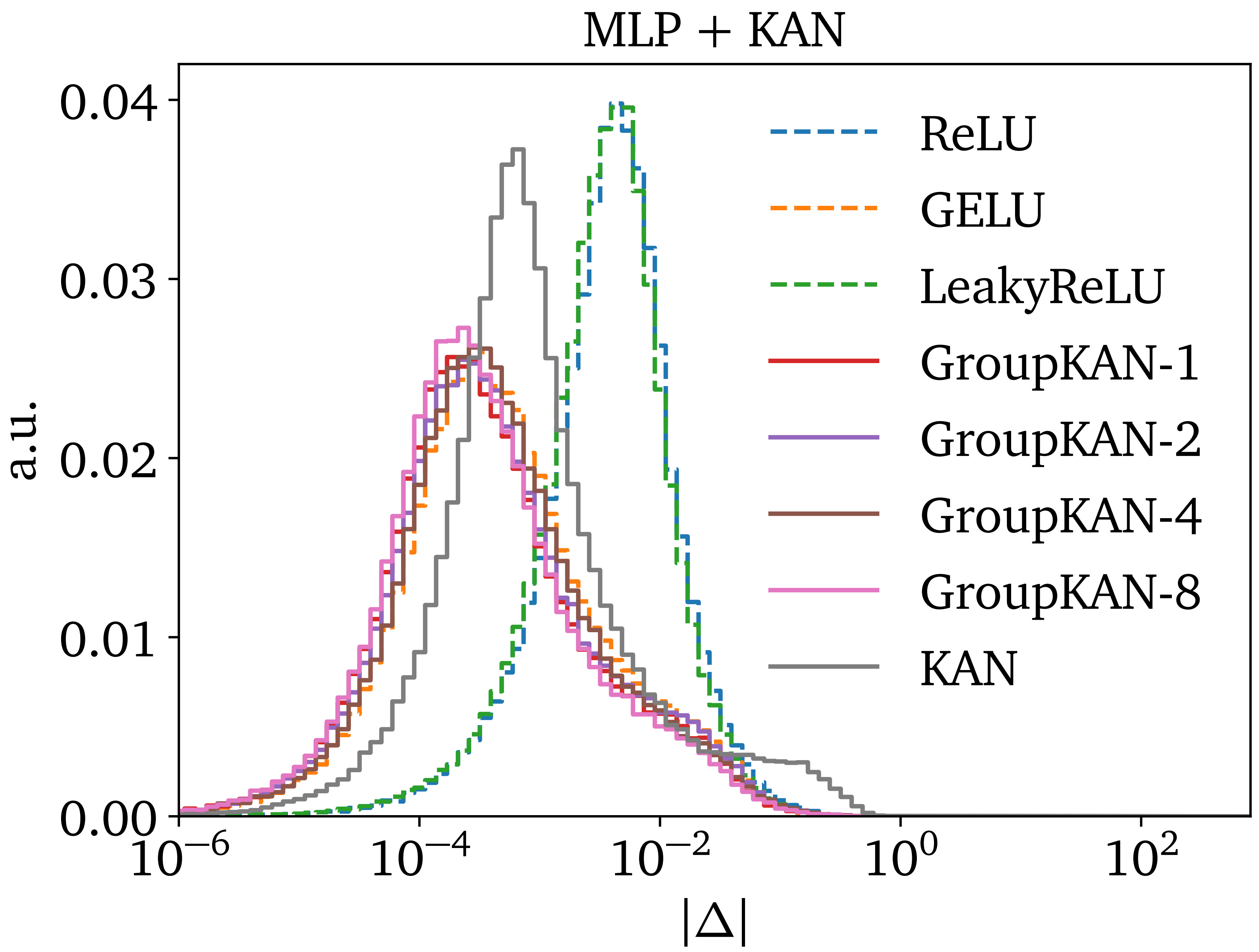
- Calibrate RE by introducing scaling parameter T: $\sigma_{\text{syst}} \to \sigma_{\text{syst}} \times T$
- T estimated by using stochastic gradient descent $\mathcal{L}_T(x) = \left\langle \frac{|A_{\text{true}}(x) - \bar{A}(x)|^2}{2\sigma^2(x)T^2} + \log \sigma(x) T \right\rangle_{x \sim D_{\text{train}}}$

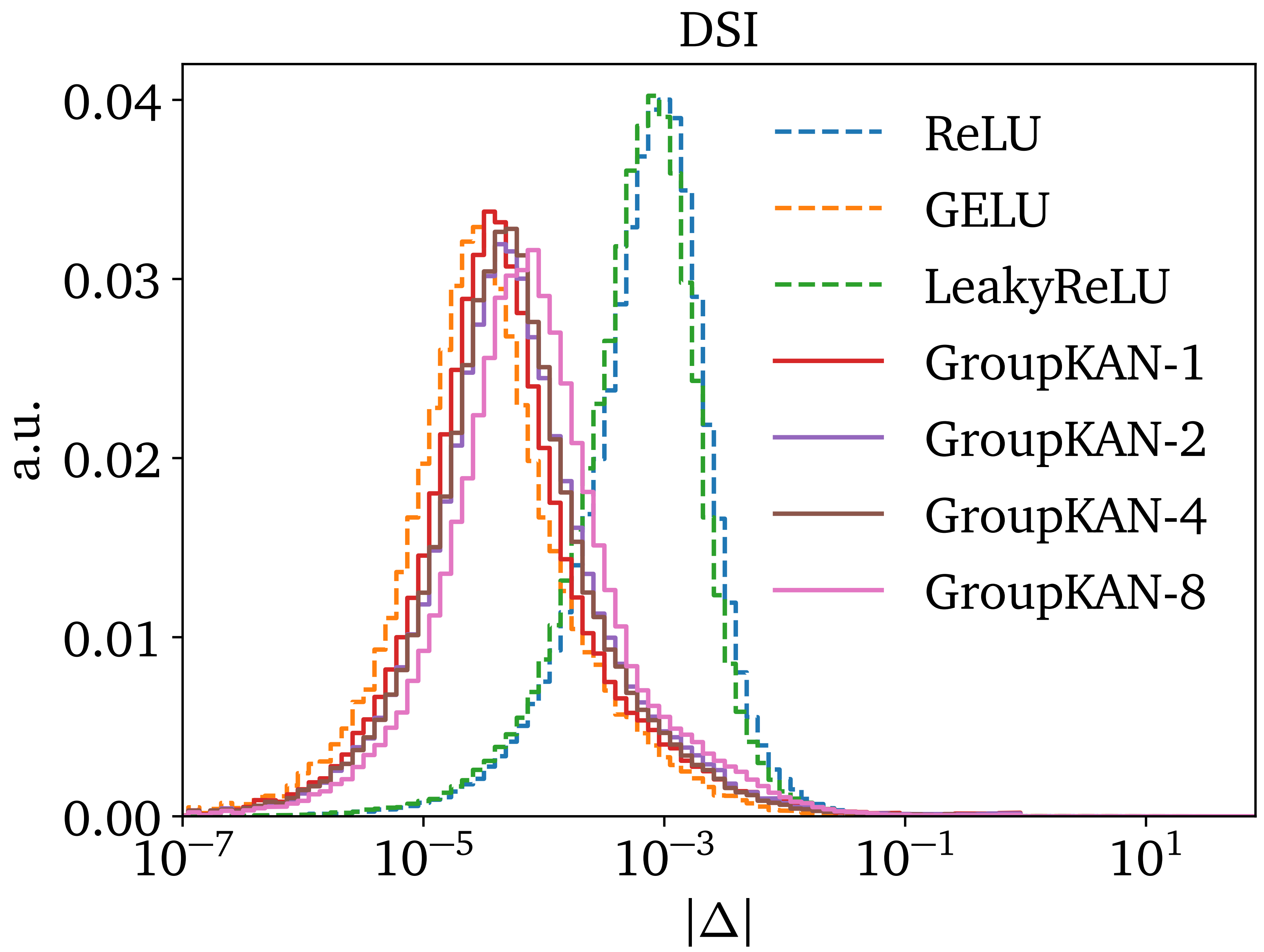




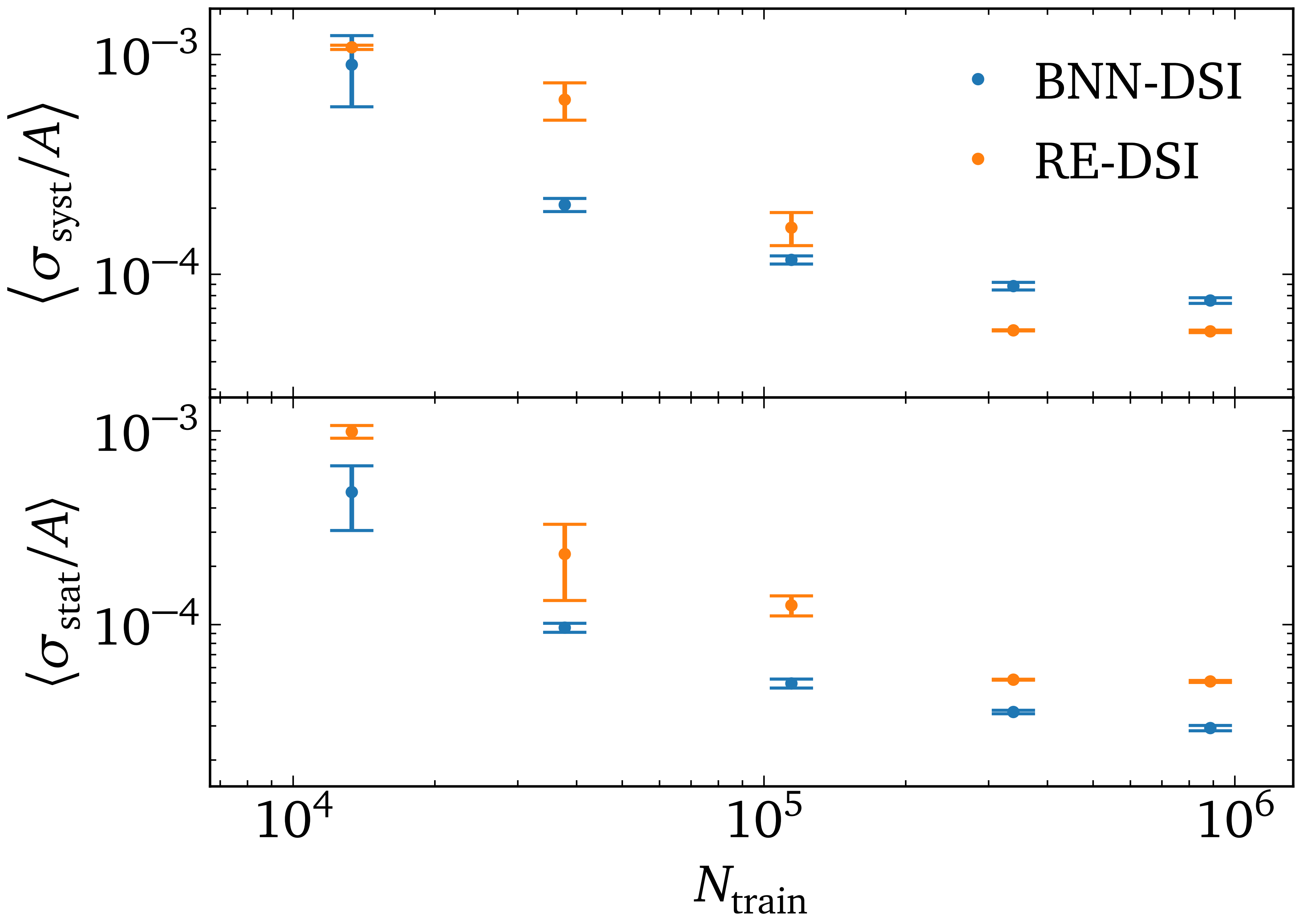
Kolmogorov-Arnold networks (KANs)

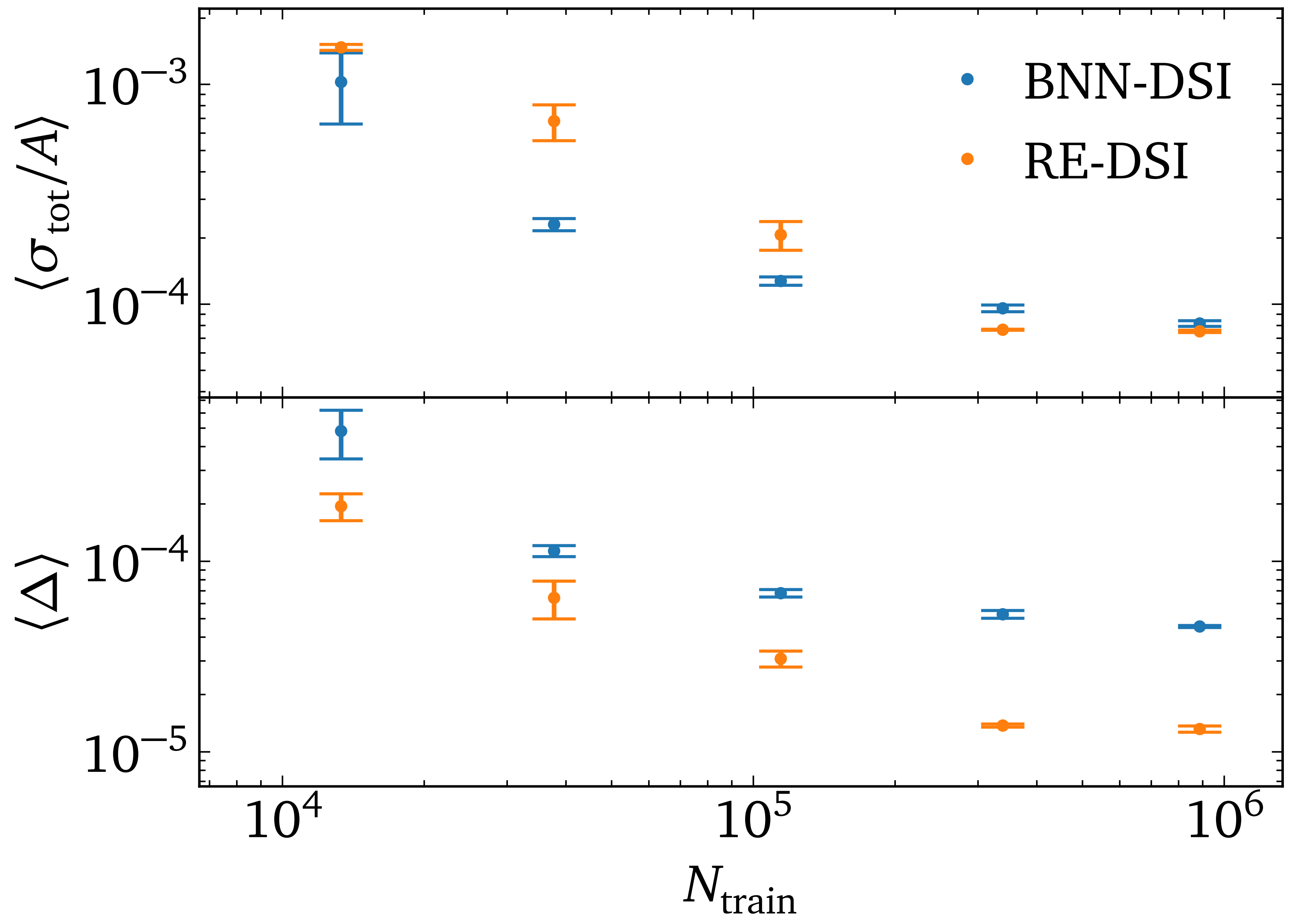
- Use KANs for calibration
- GroupKANs allow for learnable activation functions





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