Amplitudes for Accuracy and Uncertainty Estimation in ML

Nina Elmer

COMETA - Uncertainty Quantification in Machine Learning
27. January 2025

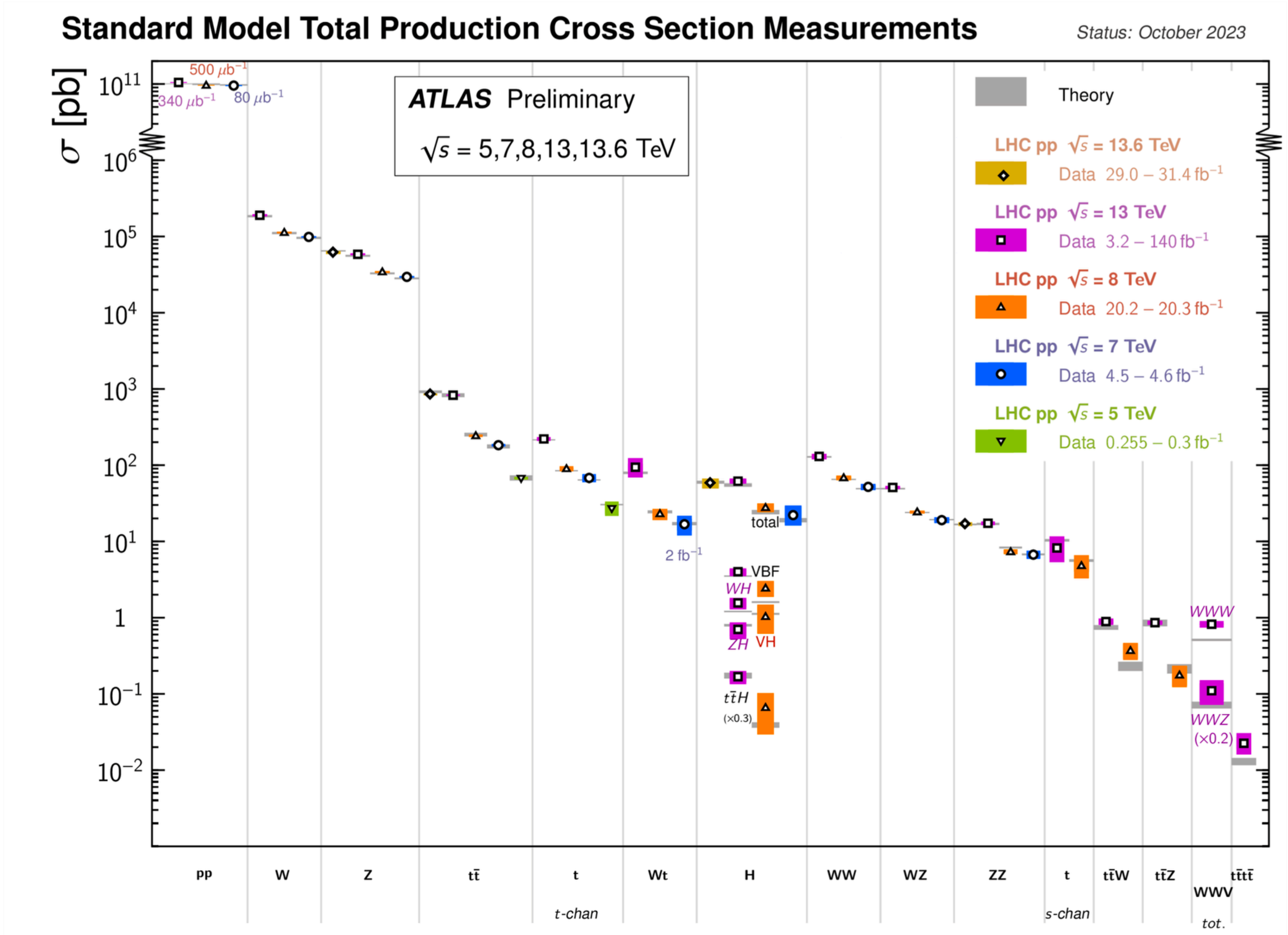
arXiv: 2412.12069

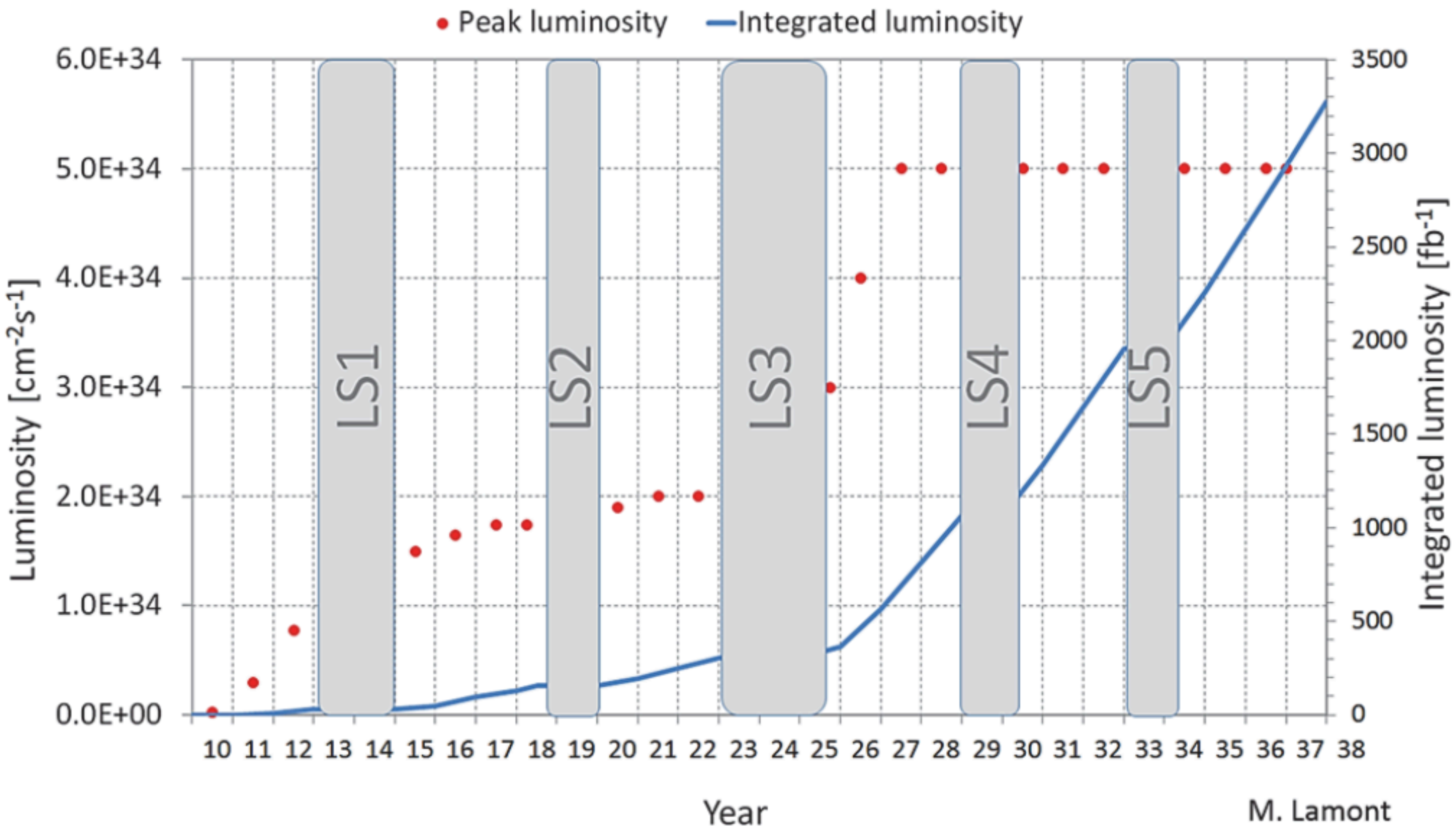
with H. Bahl, L. Favaro, M. Haußmann, T. Plehn, and R. Winterhalder

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

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

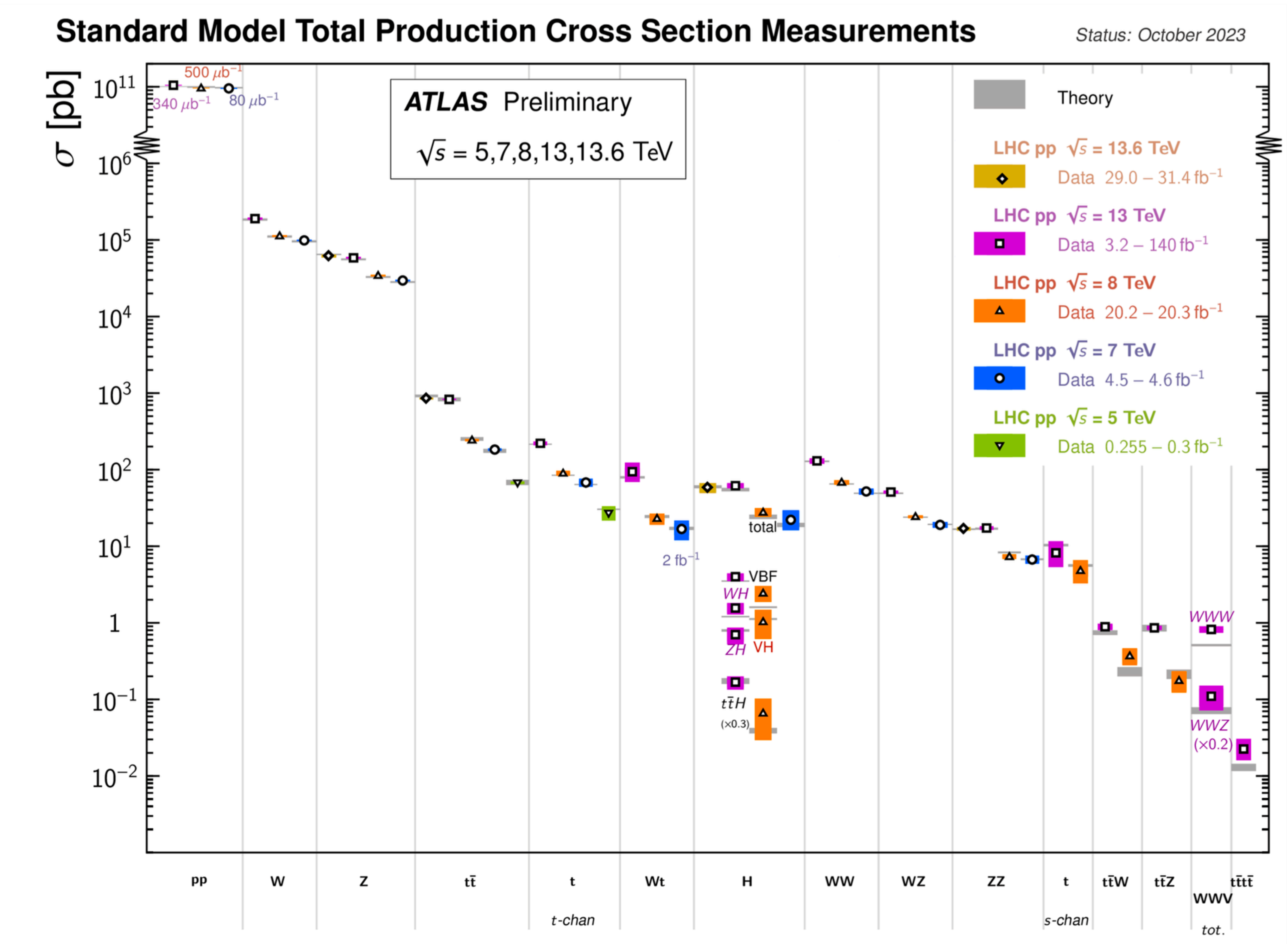
Entering the precision era

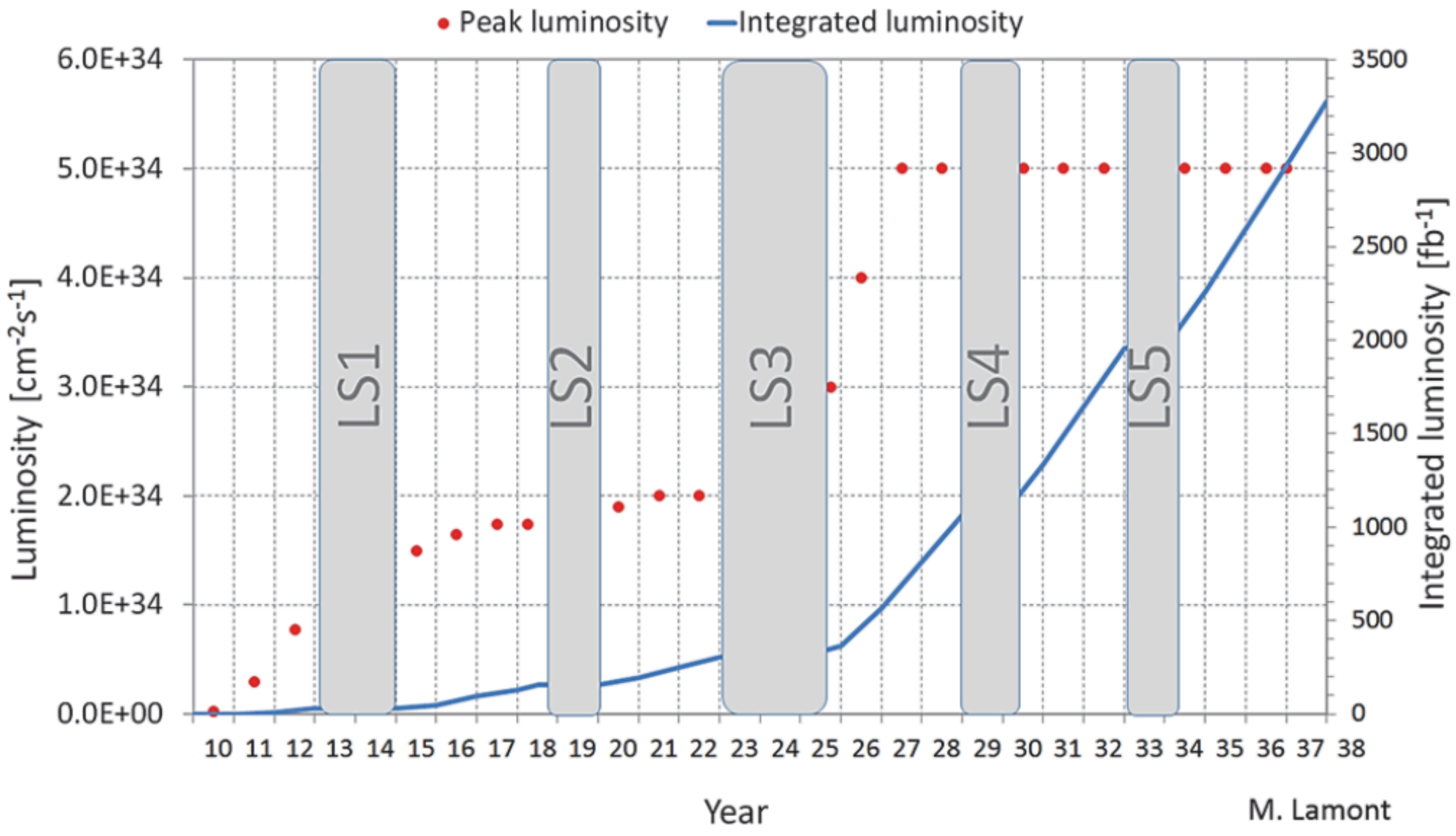




Entering the precision era

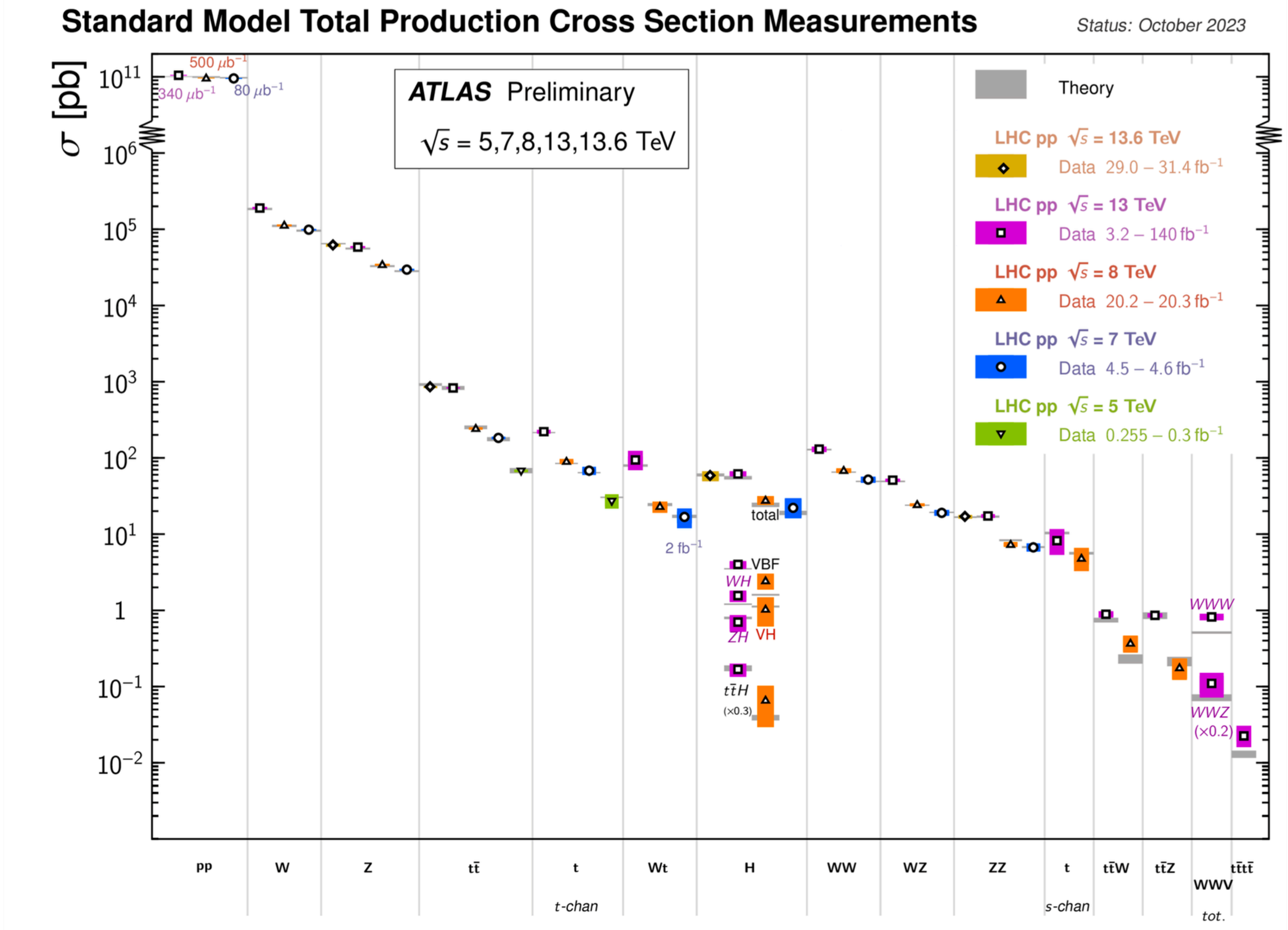
➡ Precision is central aspect of LHC physics

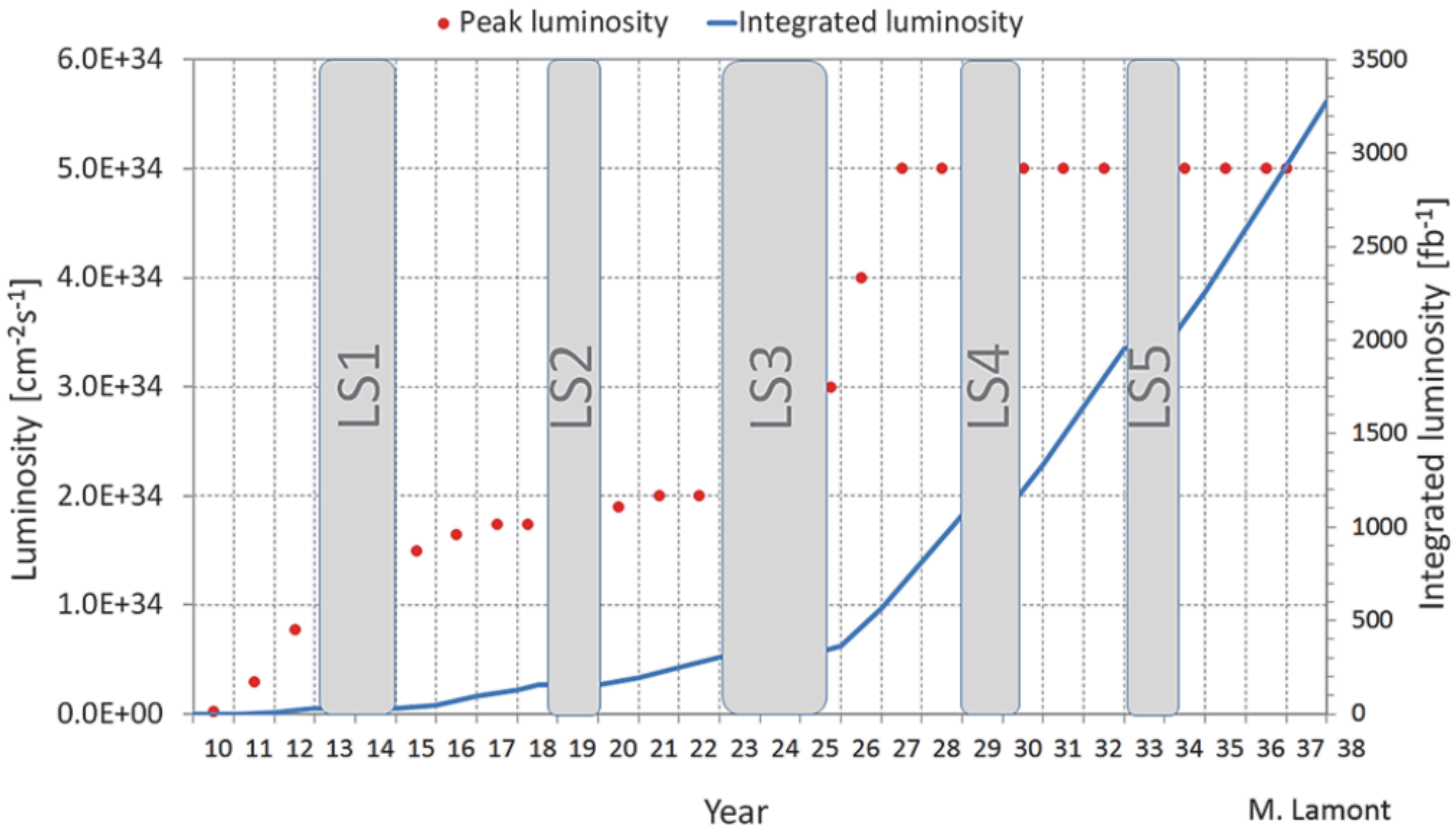




Entering the precision era

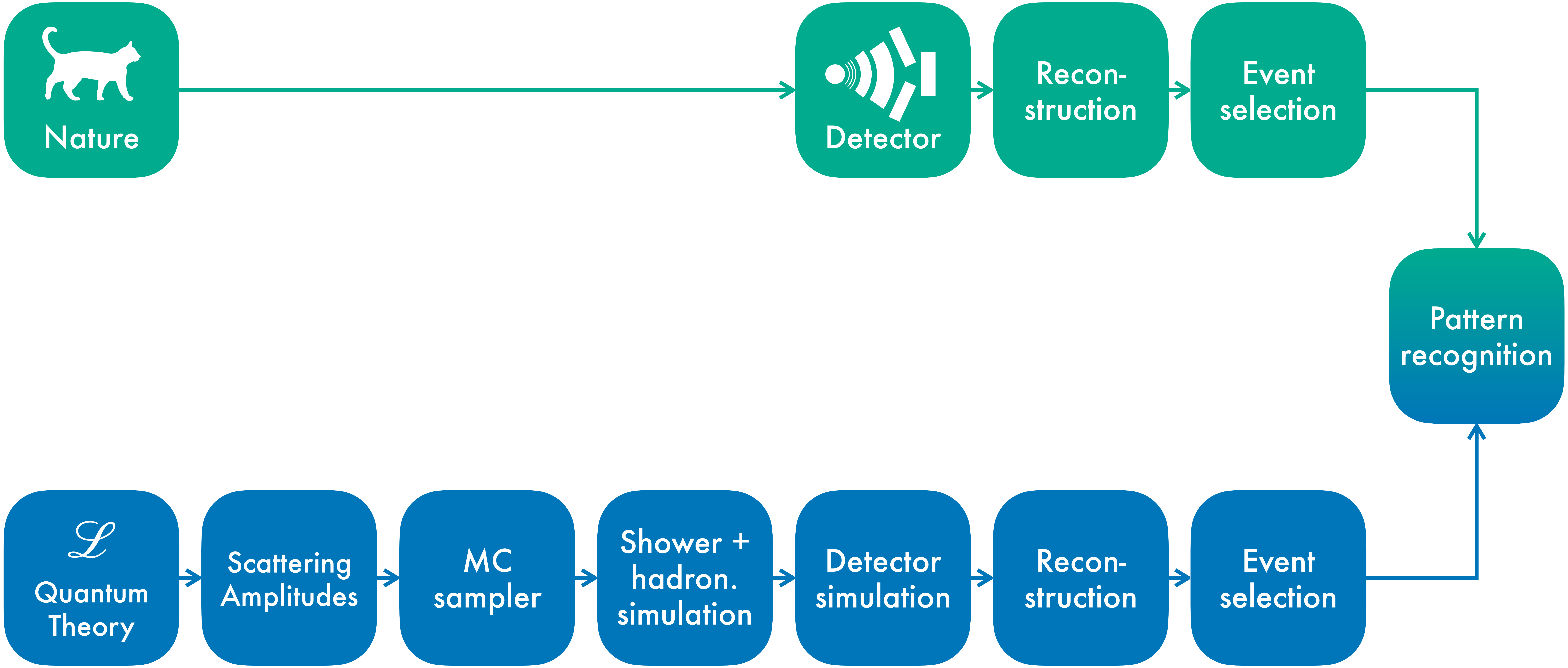
- Precision is central aspect of LHC physics





- ML tools for processing and evaluating the data

LHC physics and machine learning



Nature

Detector

Recon-struction

Event selection

Pattern recognition

$\mathscr{L}$

Quantum Theory

Scattering Amplitudes

MC sampler

Shower + hadron. simulation

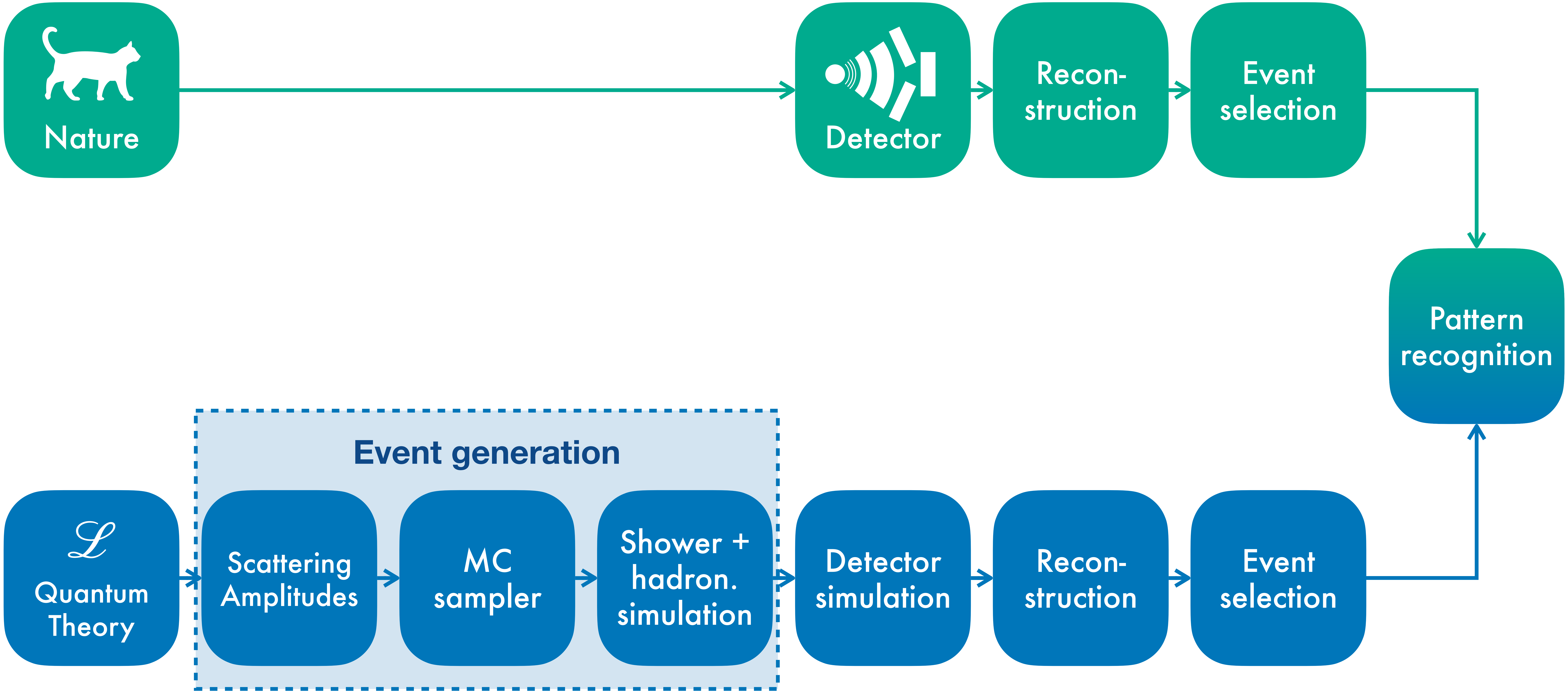
Detector simulation

Recon-struction

Event selection

Credits: R. Winterhalder

LHC physics and machine learning



Nature

Detector

Recon-struction

Event selection

Pattern recognition

$\mathscr{L}$

Quantum Theory

Event generation

Scattering Amplitudes

MC sampler

Shower + hadron. simulation

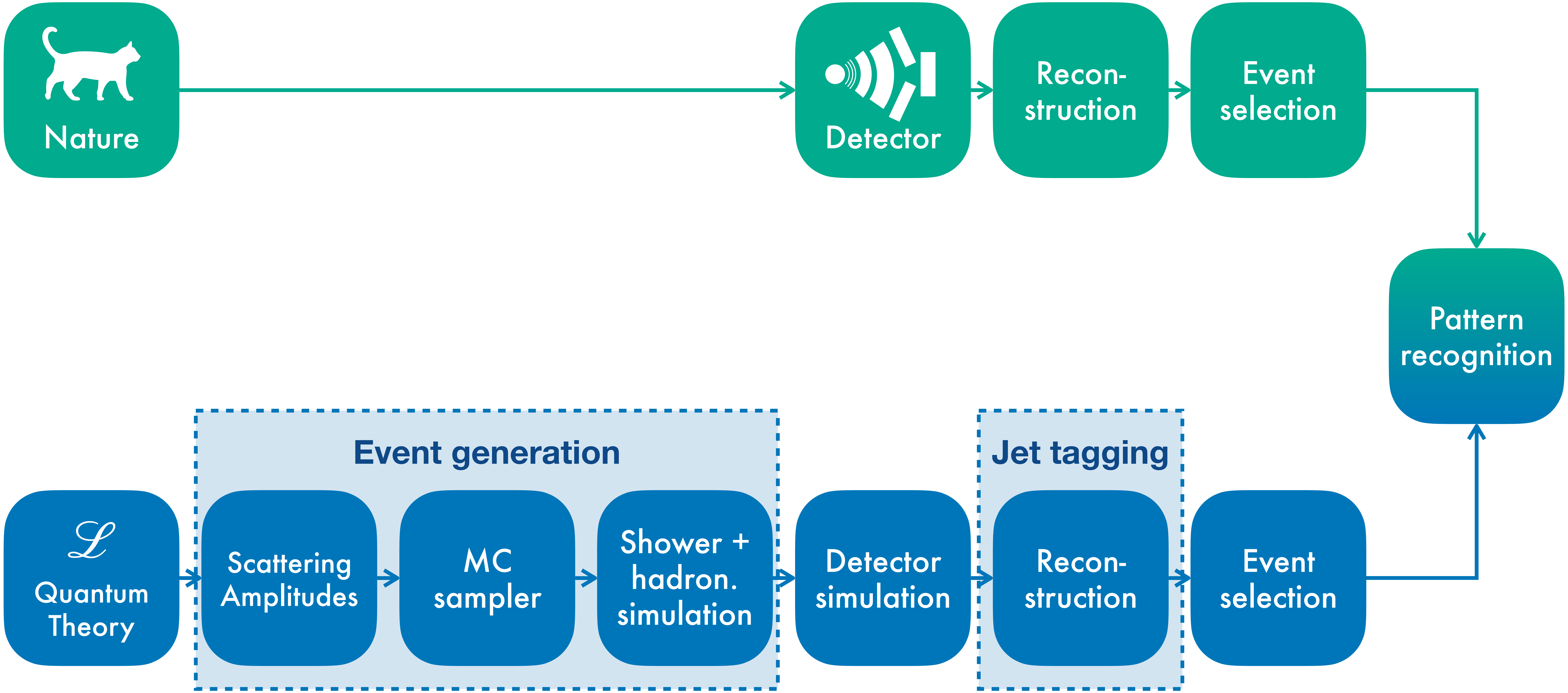
Detector simulation

Recon-struction

Event selection

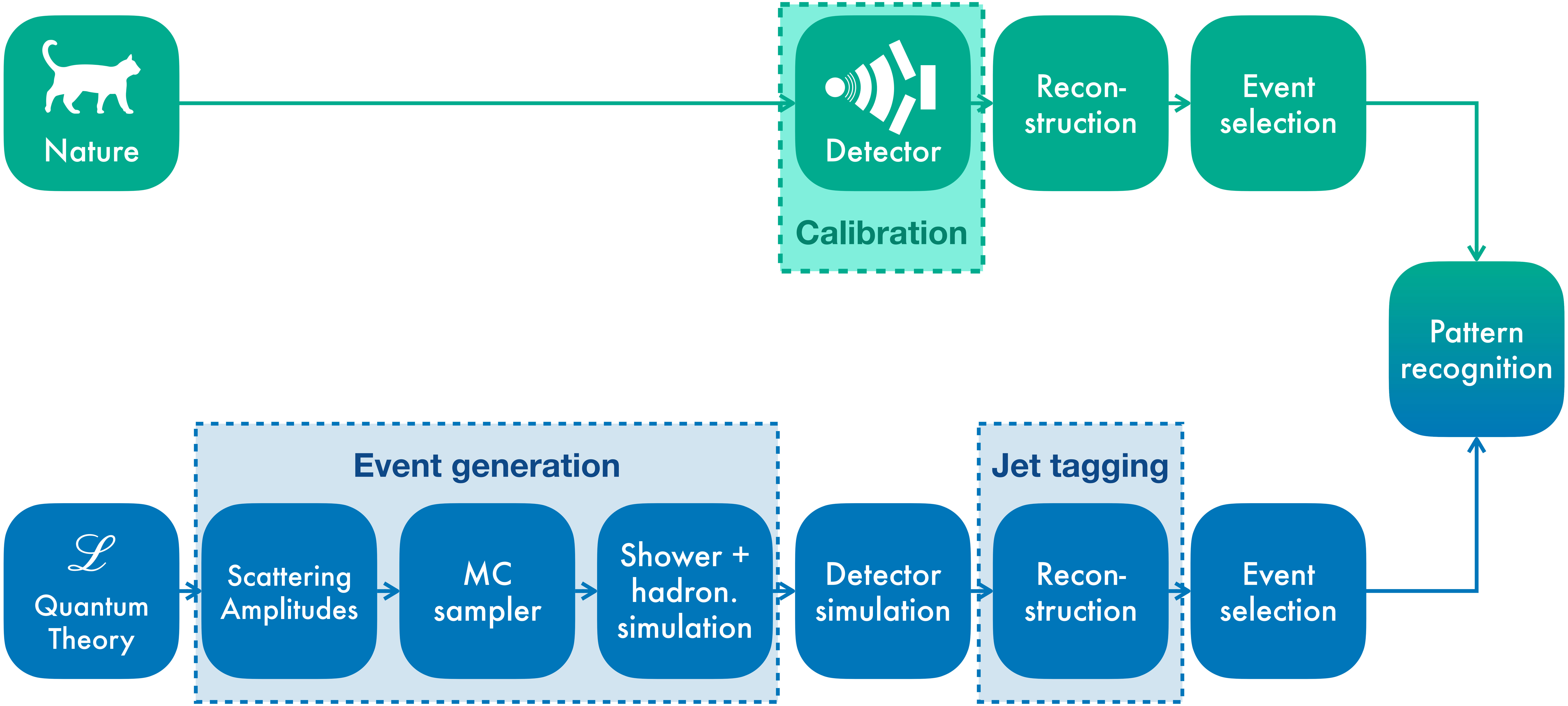
Credits: R. Winterhalder

LHC physics and machine learning



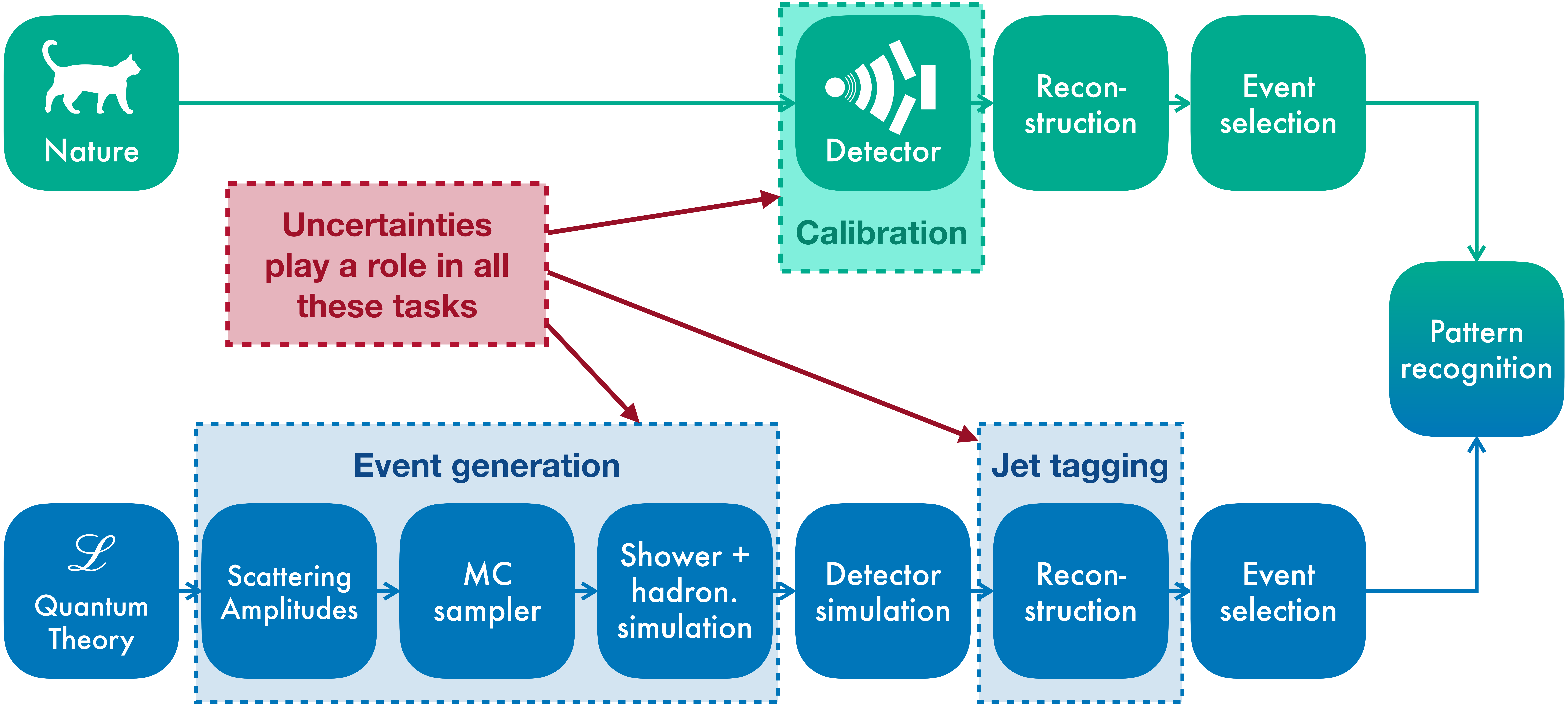
Credits: R. Winterhalder

LHC physics and machine learning



Credits: R. Winterhalder

LHC physics and machine learning



Nature

Detector

Calibration

Recon-struction

Event selection

Pattern recognition

Uncertainties play a role in all these tasks

Event generation

Jet tagging

$\mathscr{L}$ Quantum Theory

Scattering Amplitudes

MC sampler

Shower + hadron. simulation

Detector simulation

Recon-struction

Event selection

Credits: R. Winterhalder

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)

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 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

Motivation

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

network distribution

network output

- Standard heteroscedastic loss:

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

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

- Standard heteroscedastic loss:

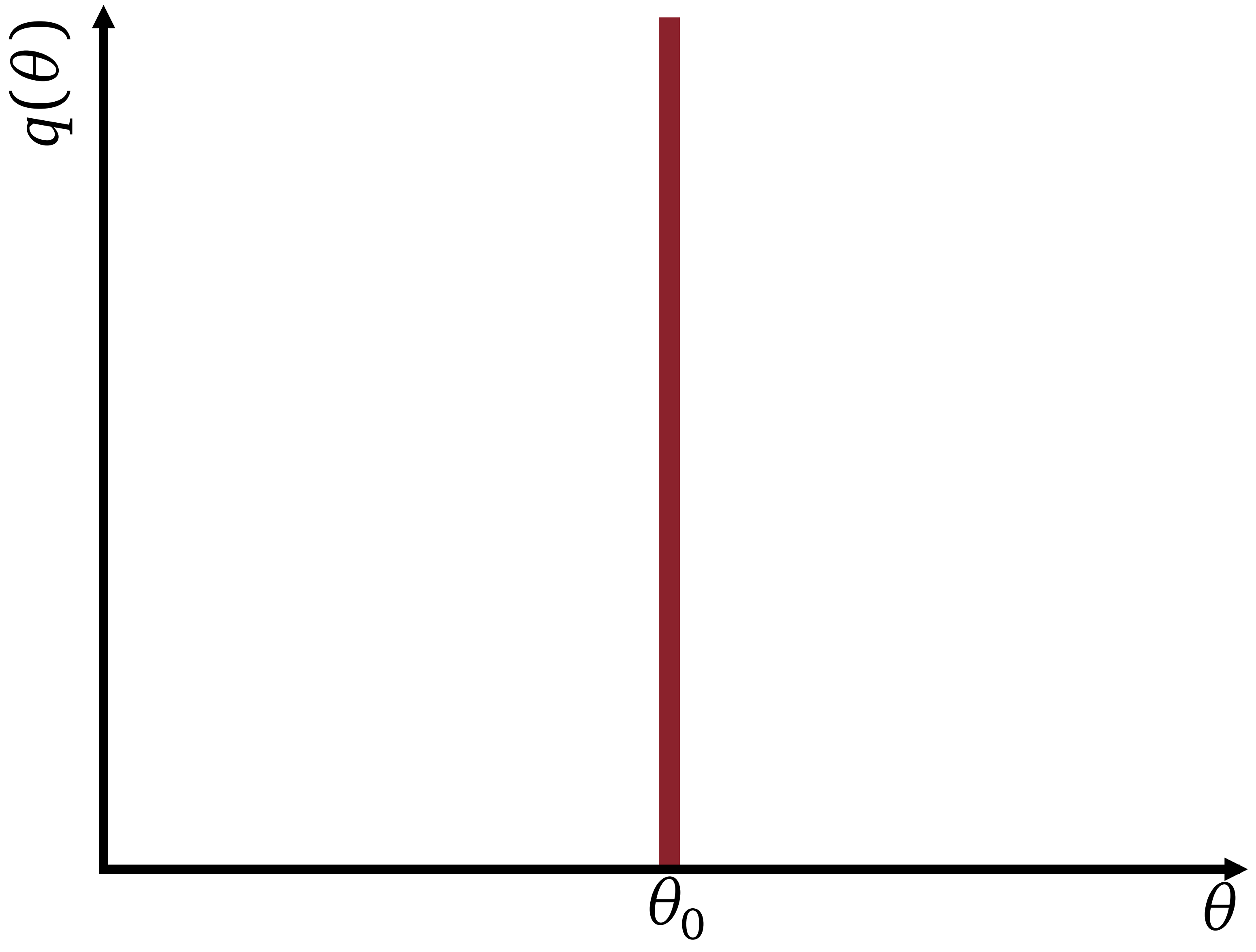
$$\mathscr{L}_{\text{heteroscedastic}} = \sum_i \frac{|f(x_i) - f_\theta(x_i)|^2}{2\sigma(x_i)^2} + \log \sigma(x_i) + \ldots$$

- 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)

Neural network



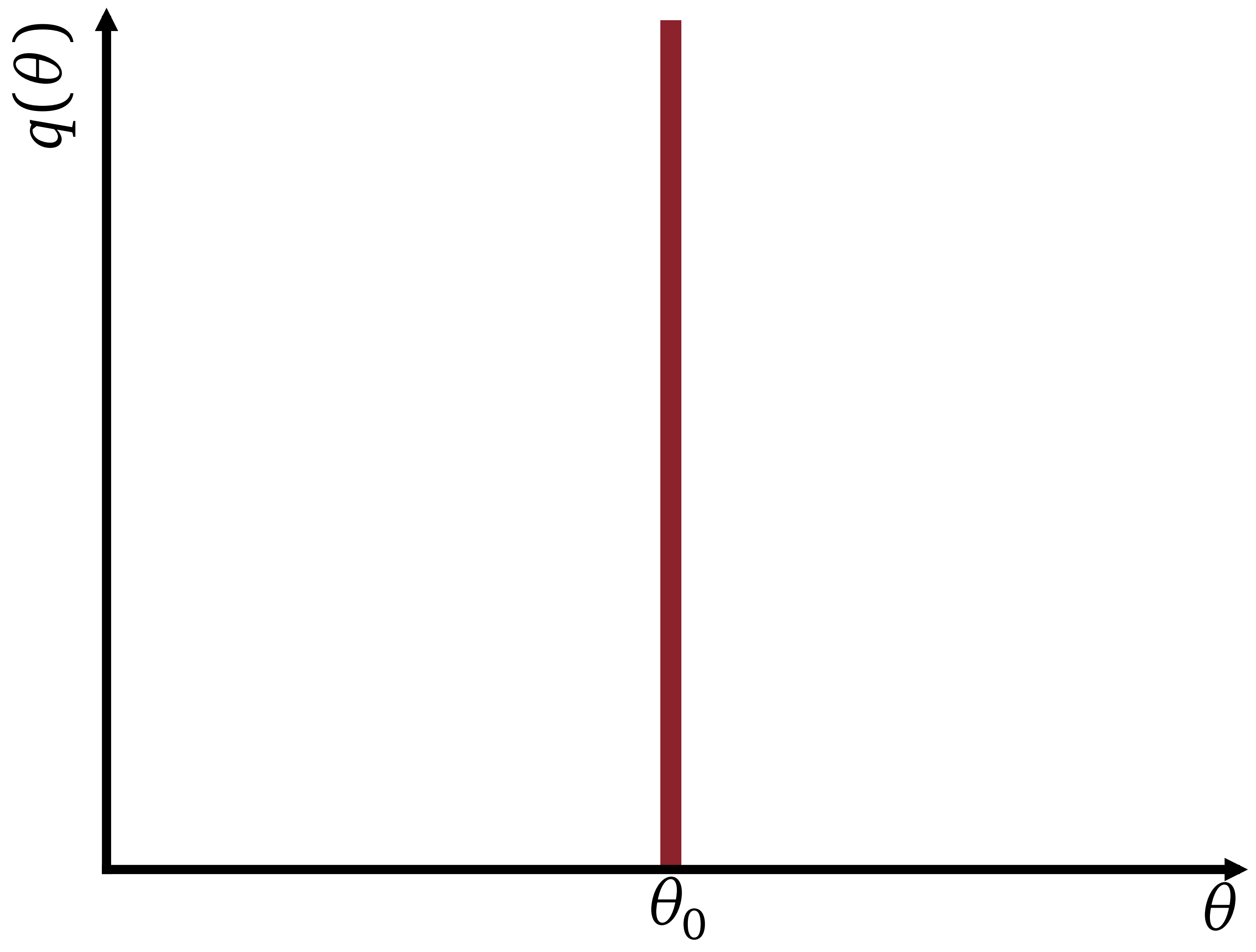
Network weights are deterministic

$$\theta = \theta_0$$

Credits: R. Winterhalder

Bayesian neural networks (BNNs)

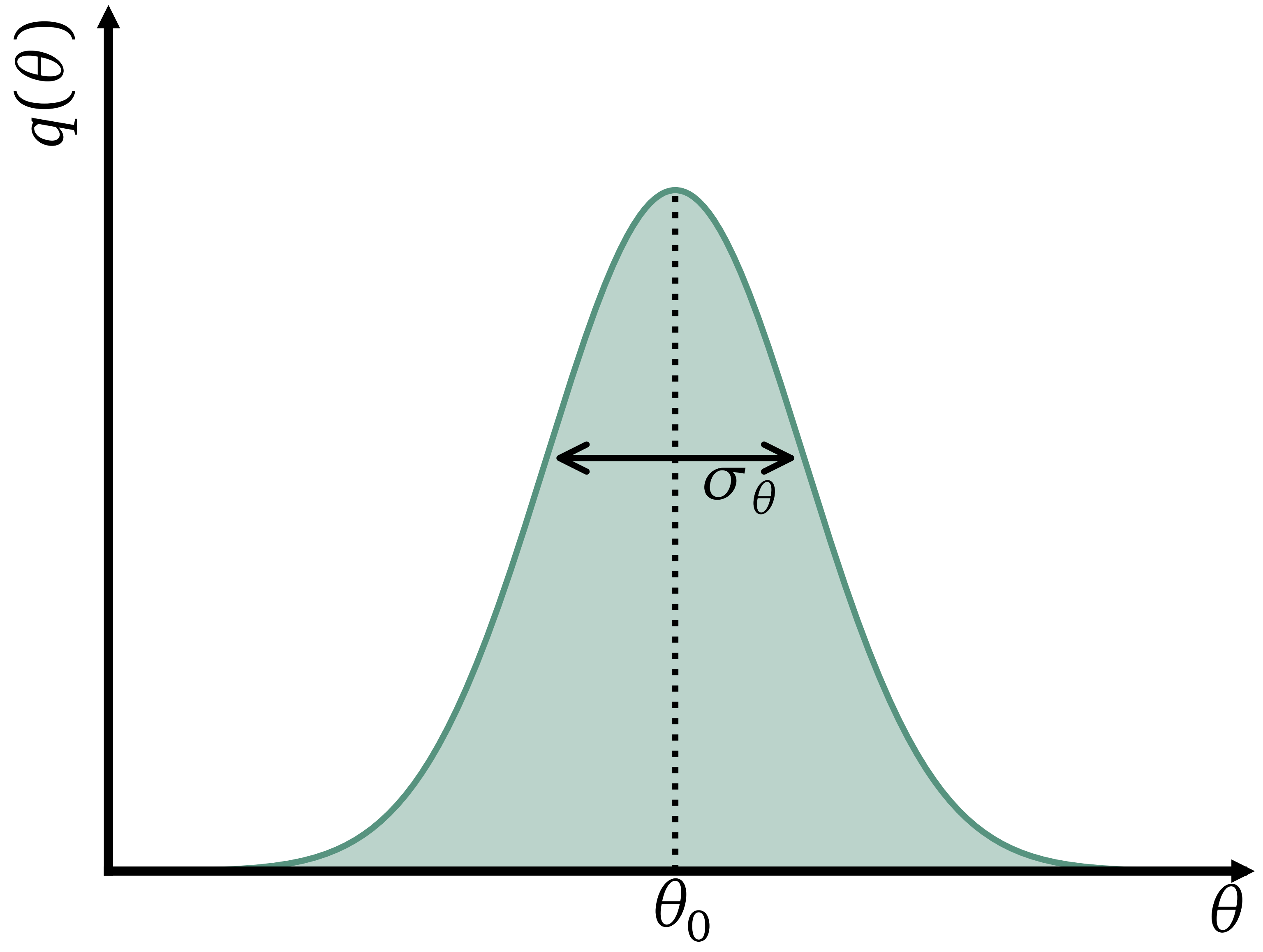
Neural network



Network weights are deterministic

$$\theta = \theta_0$$

Bayesian neural network



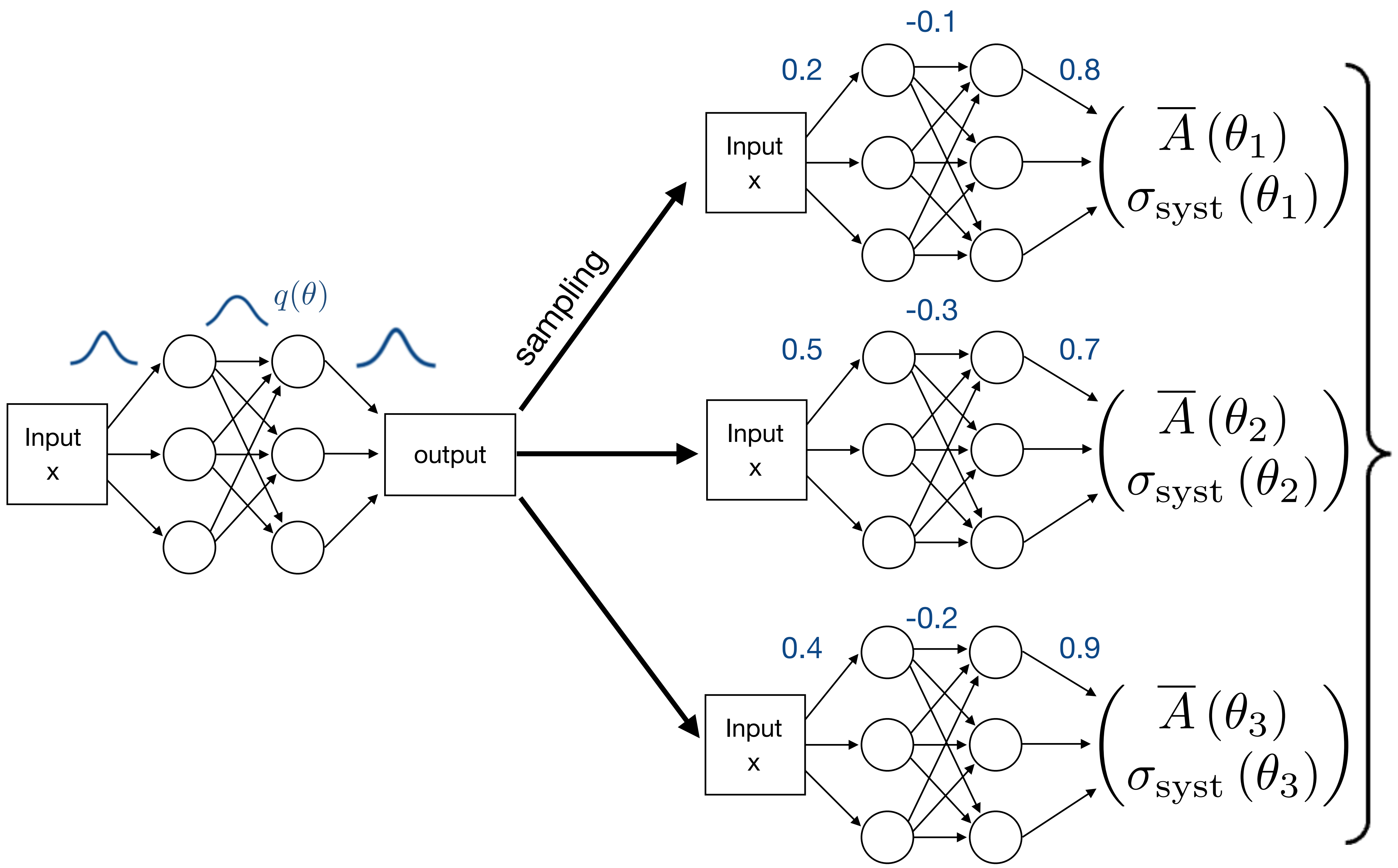
Network weights are drawn from distribution

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

Credits: R. Winterhalder

Bayesian neural networks (BNNs)

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



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

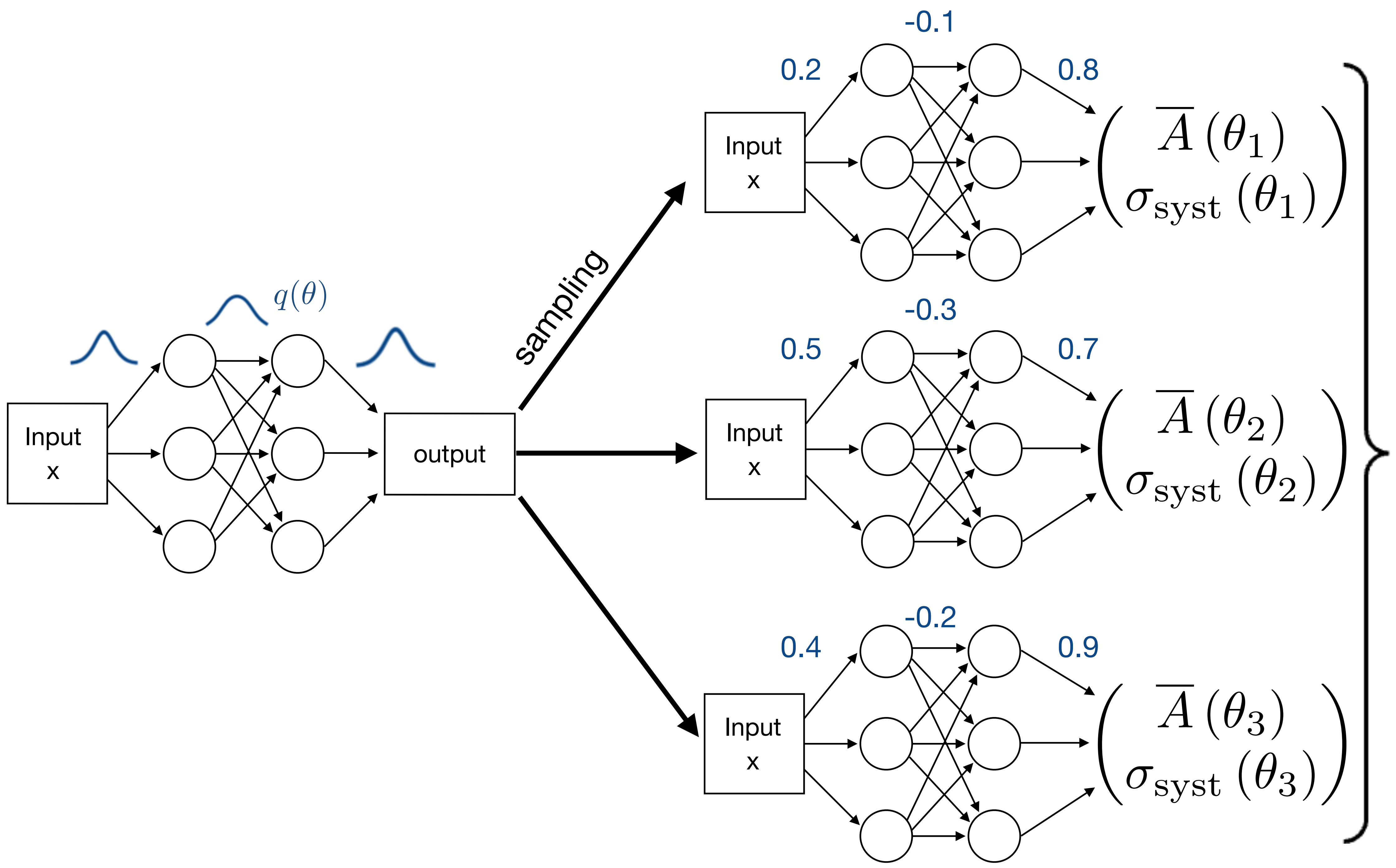
$$\sigma^2_{\text{syst}} = \frac{1}{N} \sum_i^N \sigma^2_{\text{syst}}(\theta_i)$$

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

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

Bayesian neural networks (BNNs)

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



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

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

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

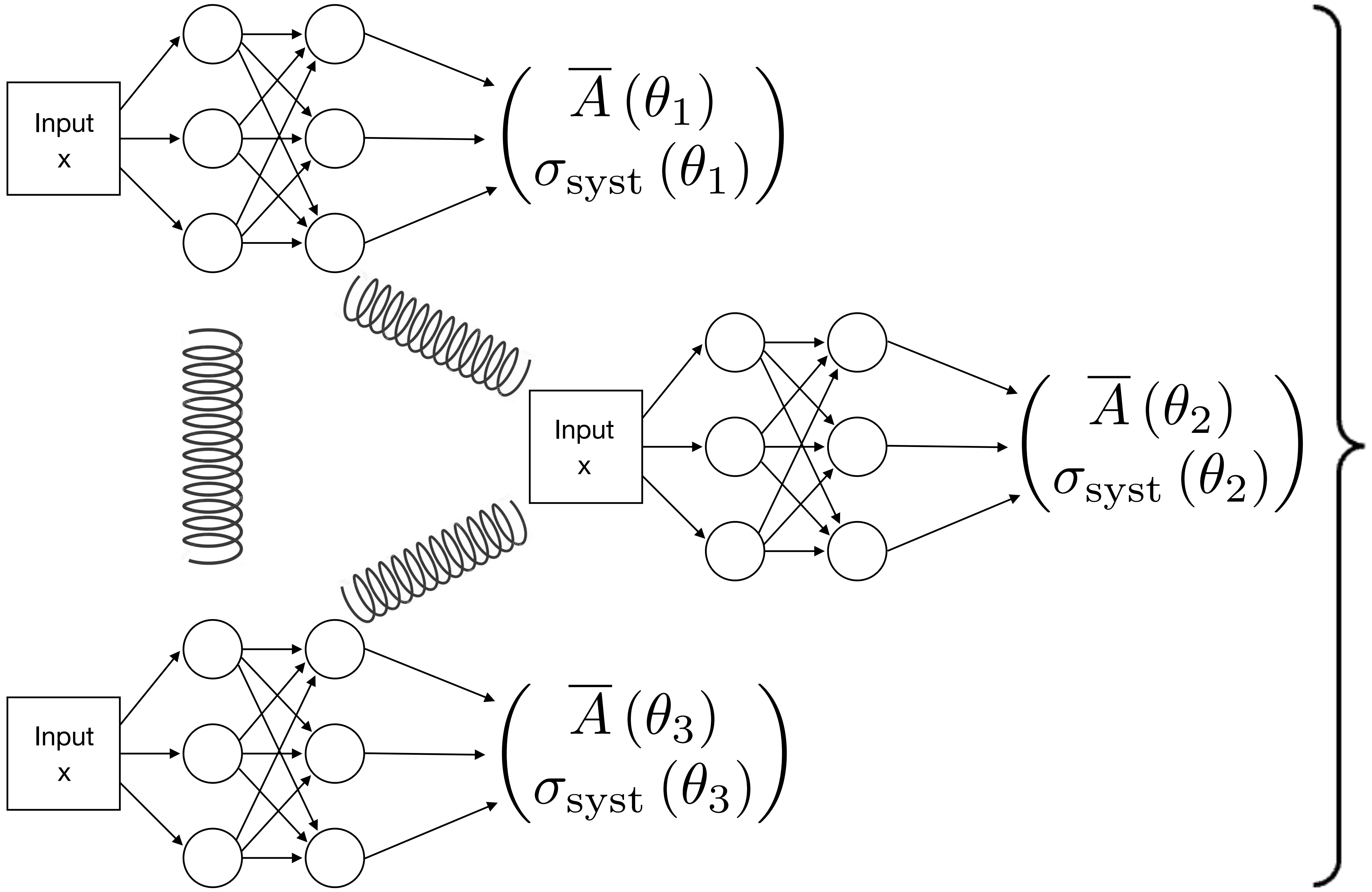
$$\mathcal{L}_{\text{BNN}} = \sum_x \Big[\text{KL}[q(\theta), p(\theta)] - \big\langle \log p(D_{\text{train}}|\theta)\big\rangle_{\theta\sim q(\theta)}\Big]$$

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

Repulsive ensembles (REs)

Ensemble of networks

Output



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

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

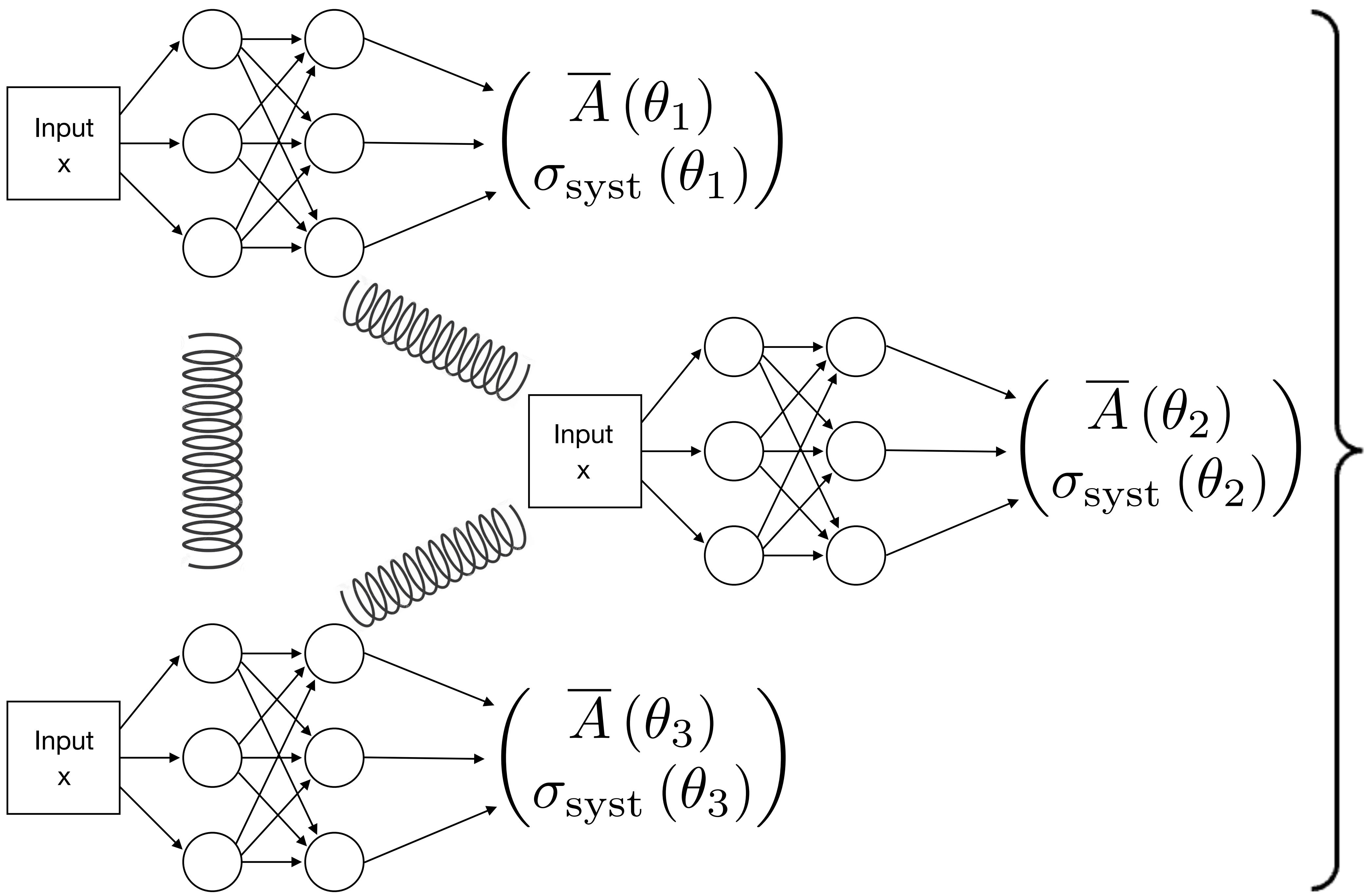
$$\sigma^2_{\text{stat}} = \frac{1}{N}\sum_i^N \left(\langle A\rangle - \overline{A}(\theta_i)\right)^2$$

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

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**

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

Learning Amplitudes

- Learning scattering amplitudes $|M|^2$

- One-loop partonic process: $gg \to \gamma\gamma g$

- 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:

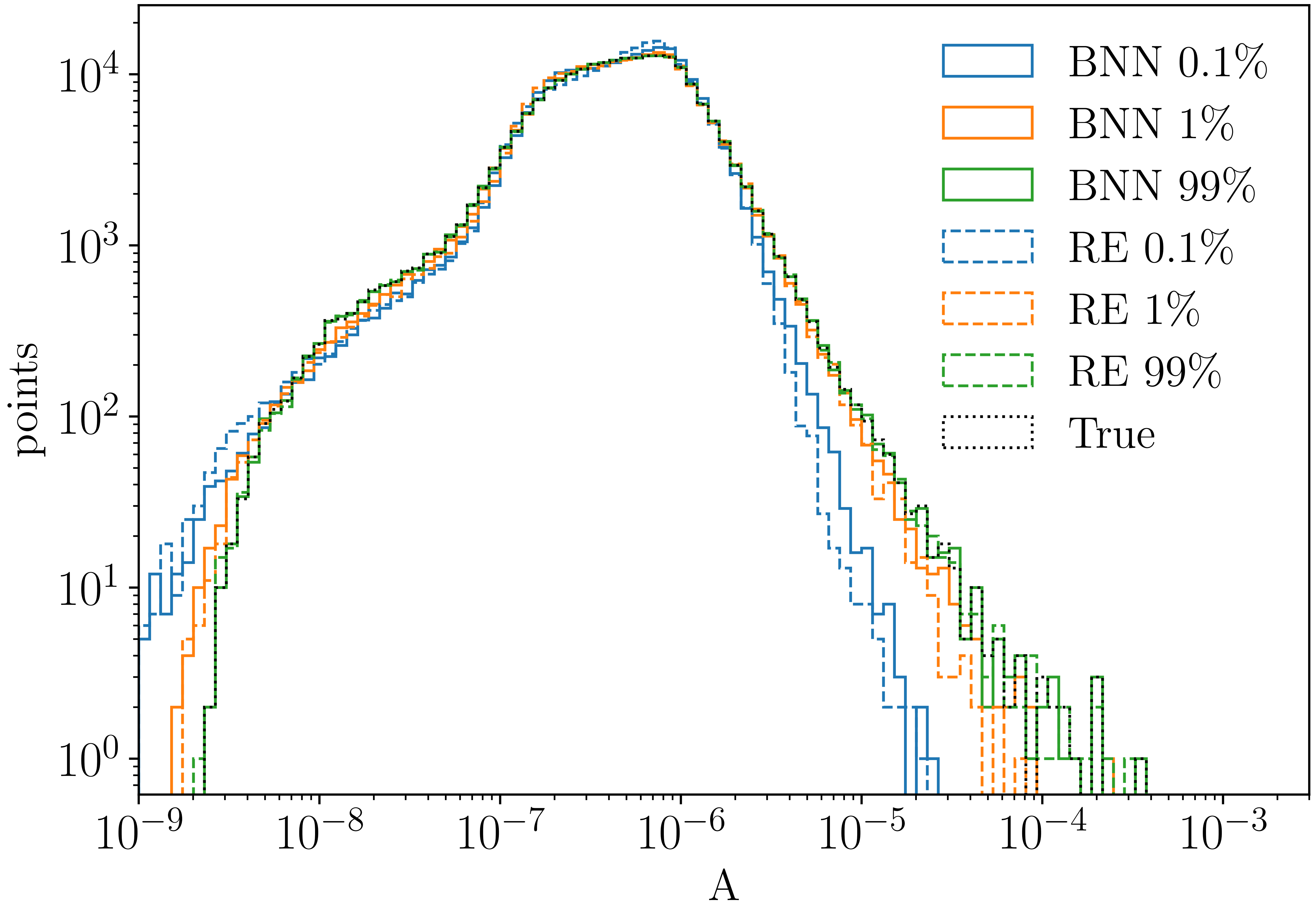
$$p_{T,j} > 20\,\mathrm{GeV}$$
$$p_{T,\gamma} > 40, 30\,\mathrm{GeV}$$
$$|\eta_j| < 5$$
$$|\eta_\gamma| < 2.37$$
$$R_{jj,j\gamma,\gamma\gamma} > 0.4$$

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:

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

training with different amounts of data



Outline

Part I: Different networks and architectures

Part II: **Systematic uncertainties**

Part III: Statistical uncertainties

Adding Gaussian noise

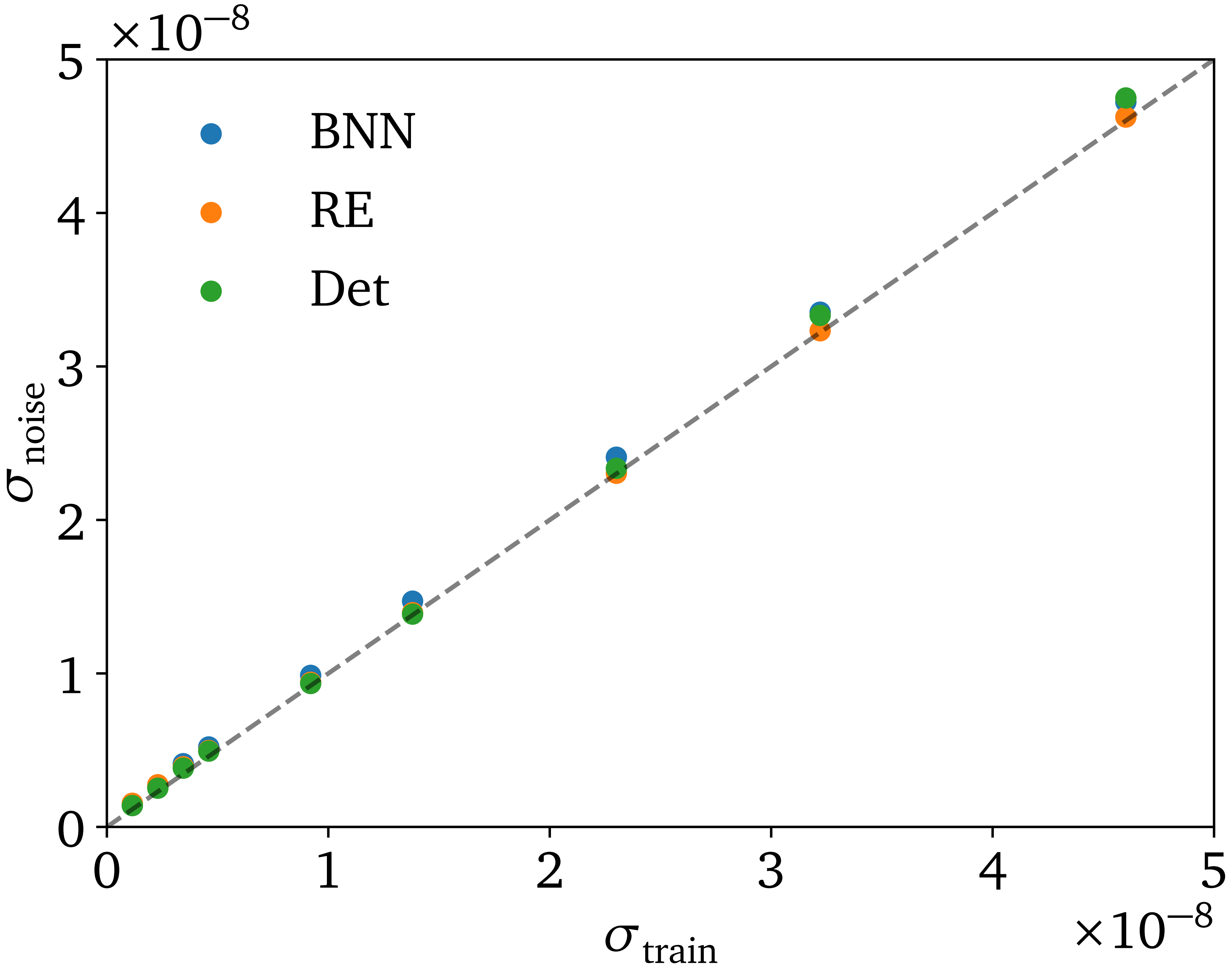
$$\sigma^2_{\text{tot}} = \sigma^2_{\text{syst},0} + \sigma^2_{\text{noise}} + \sigma^2_{\text{stat}}$$

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

Adding Gaussian noise

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

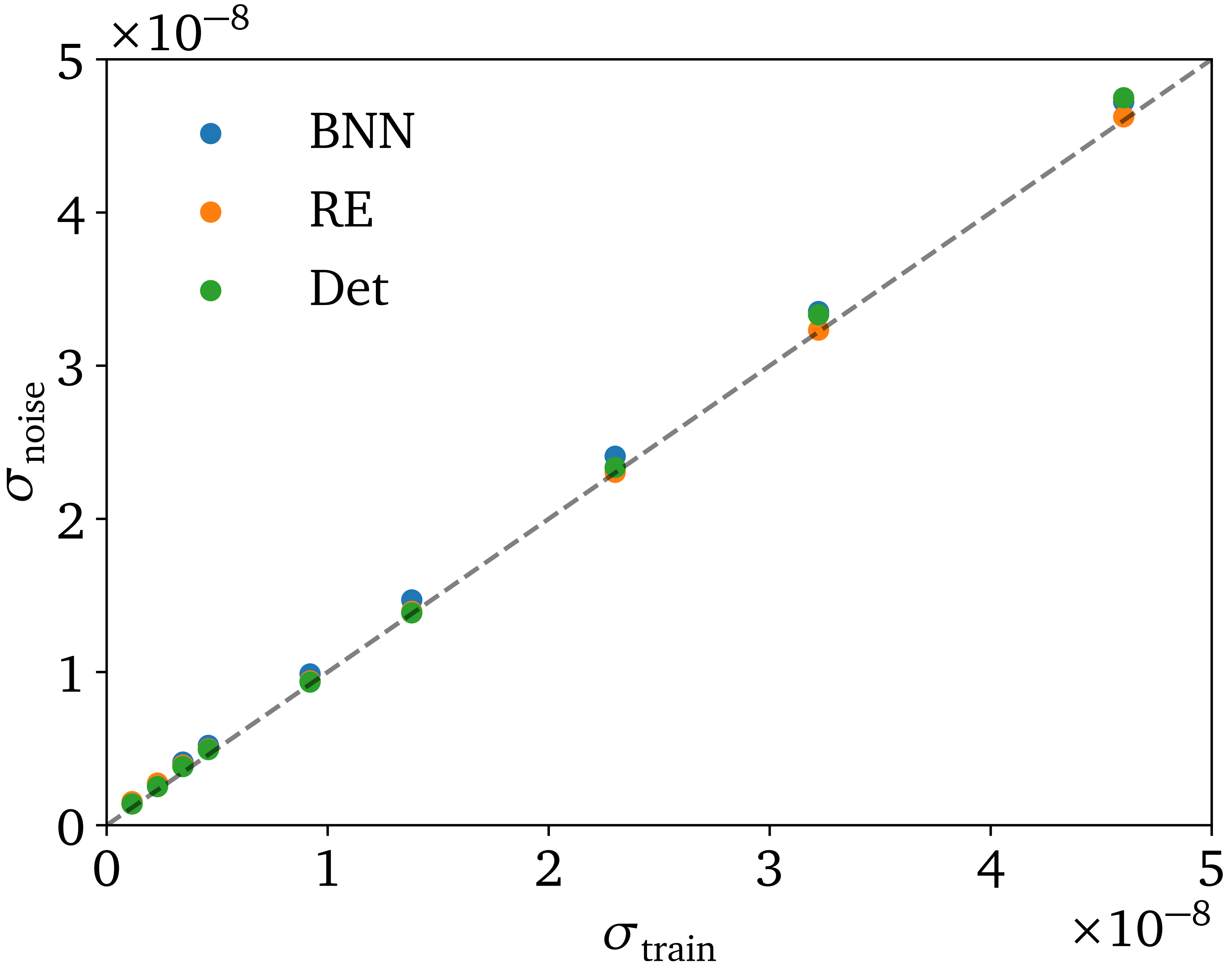
$$\sigma_{\text{train}} = f_{\text{smear}}\, A_{\text{true}}$$

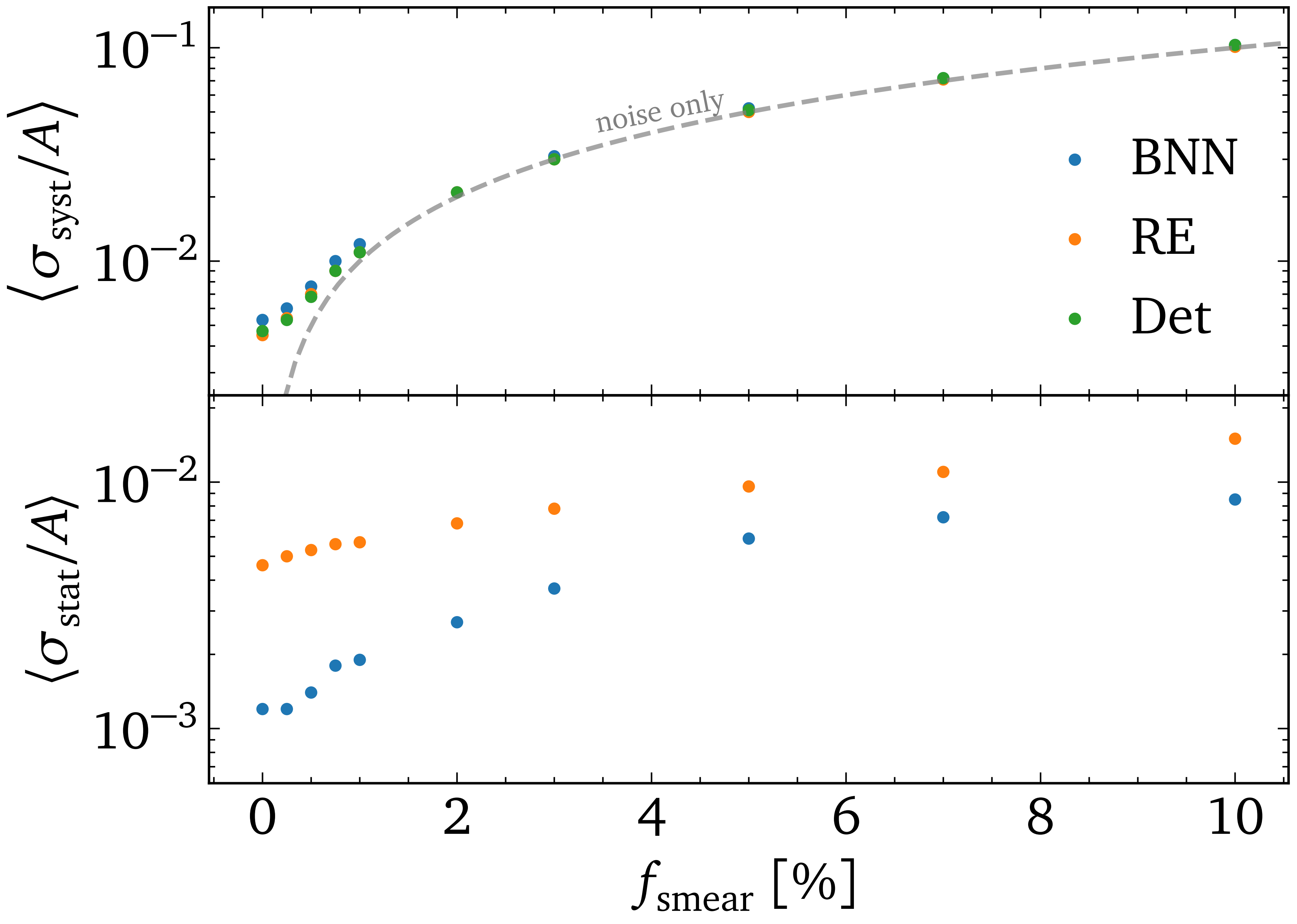


Adding Gaussian noise

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

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

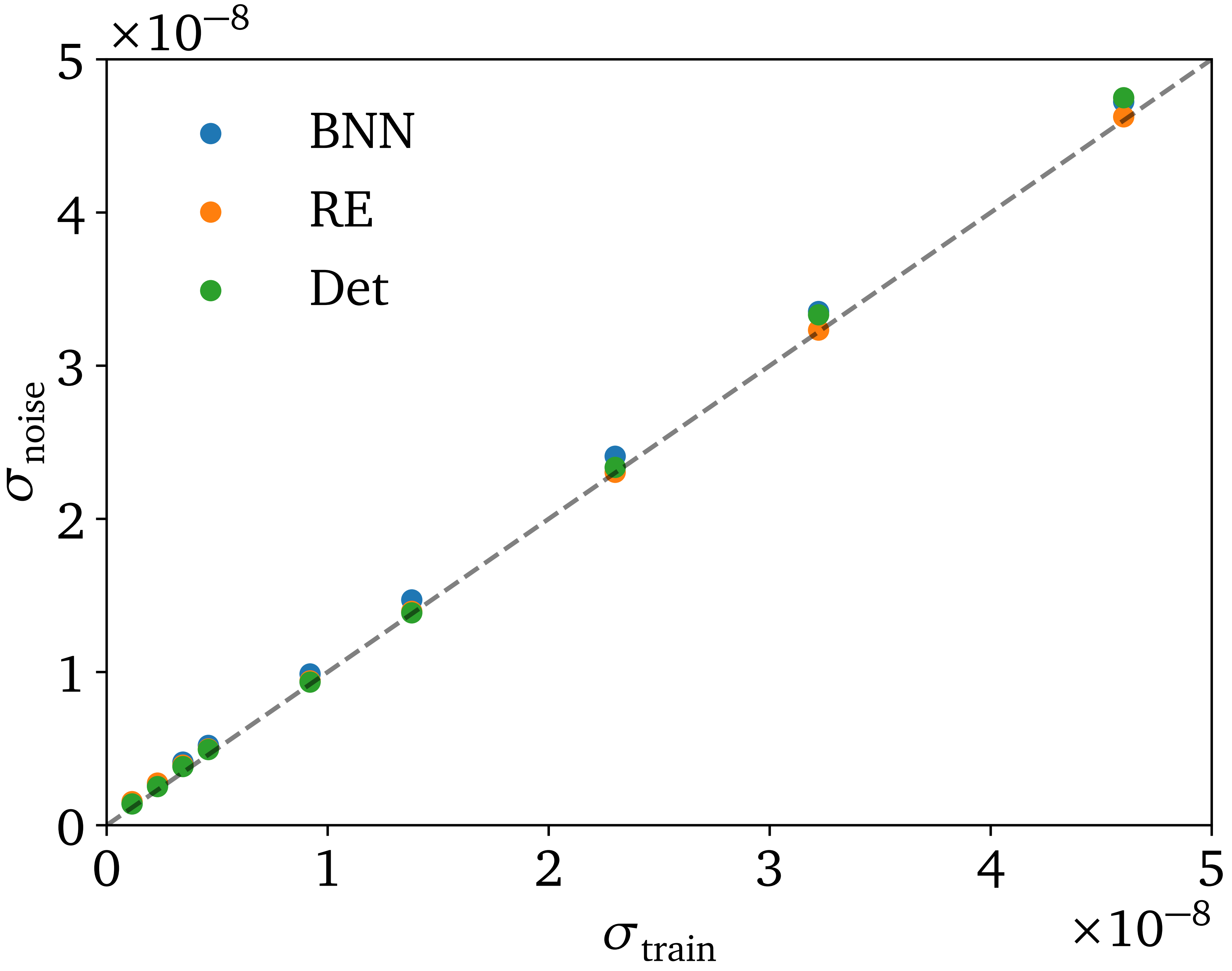


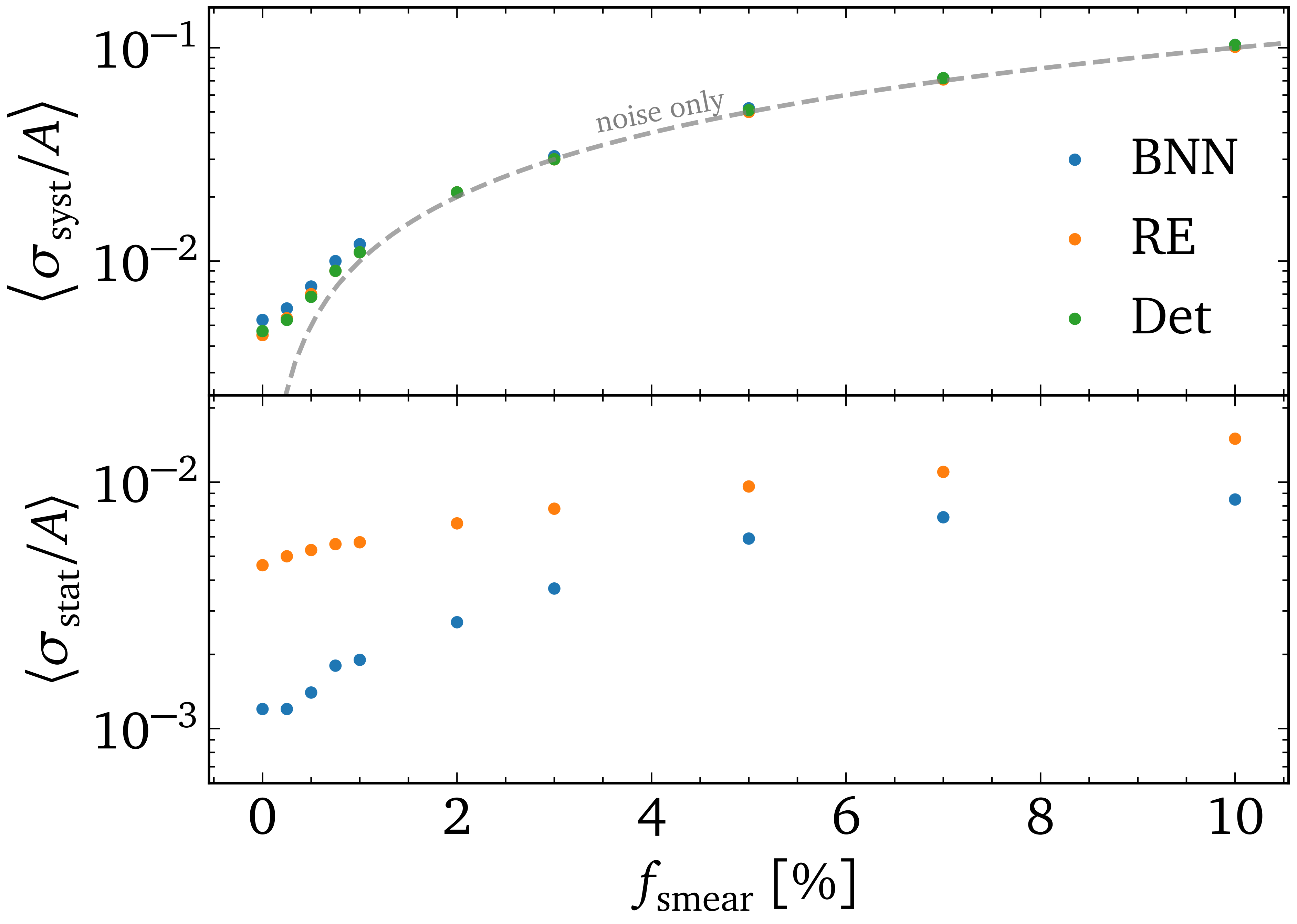


Adding Gaussian noise

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

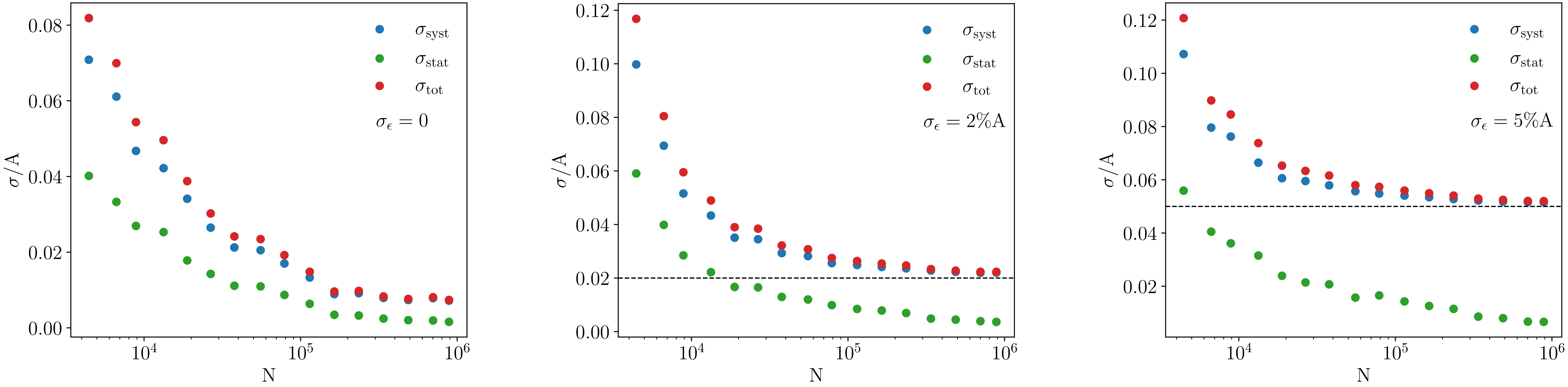
$$\sigma_{\mathrm{train}} = f_{\mathrm{smear}}\, A_{\mathrm{true}}$$





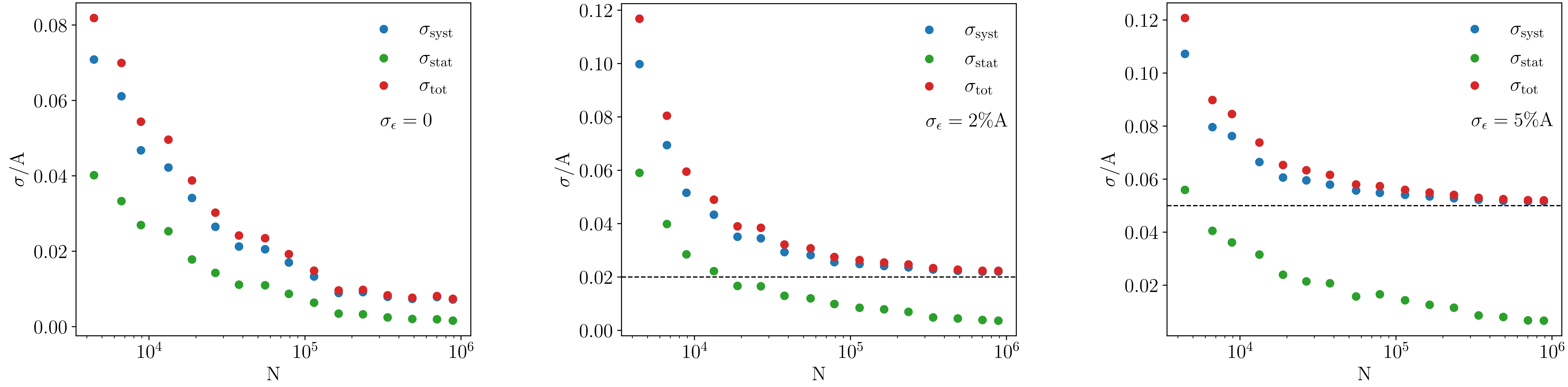
➡ Networks learn noise as **systematic** uncertainties

Uncertainty behavior



BNN only

Uncertainty behavior



BNN only

1. Statistical uncertainty **independent** of noise

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

Calibration of the results

Bias calibration

Uncertainty calibration

Calibration of the results

Bias calibration

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

Uncertainty calibration

Calibration of the results

Bias calibration →

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

Uncertainty calibration →

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

Calibration of the results

Bias calibration

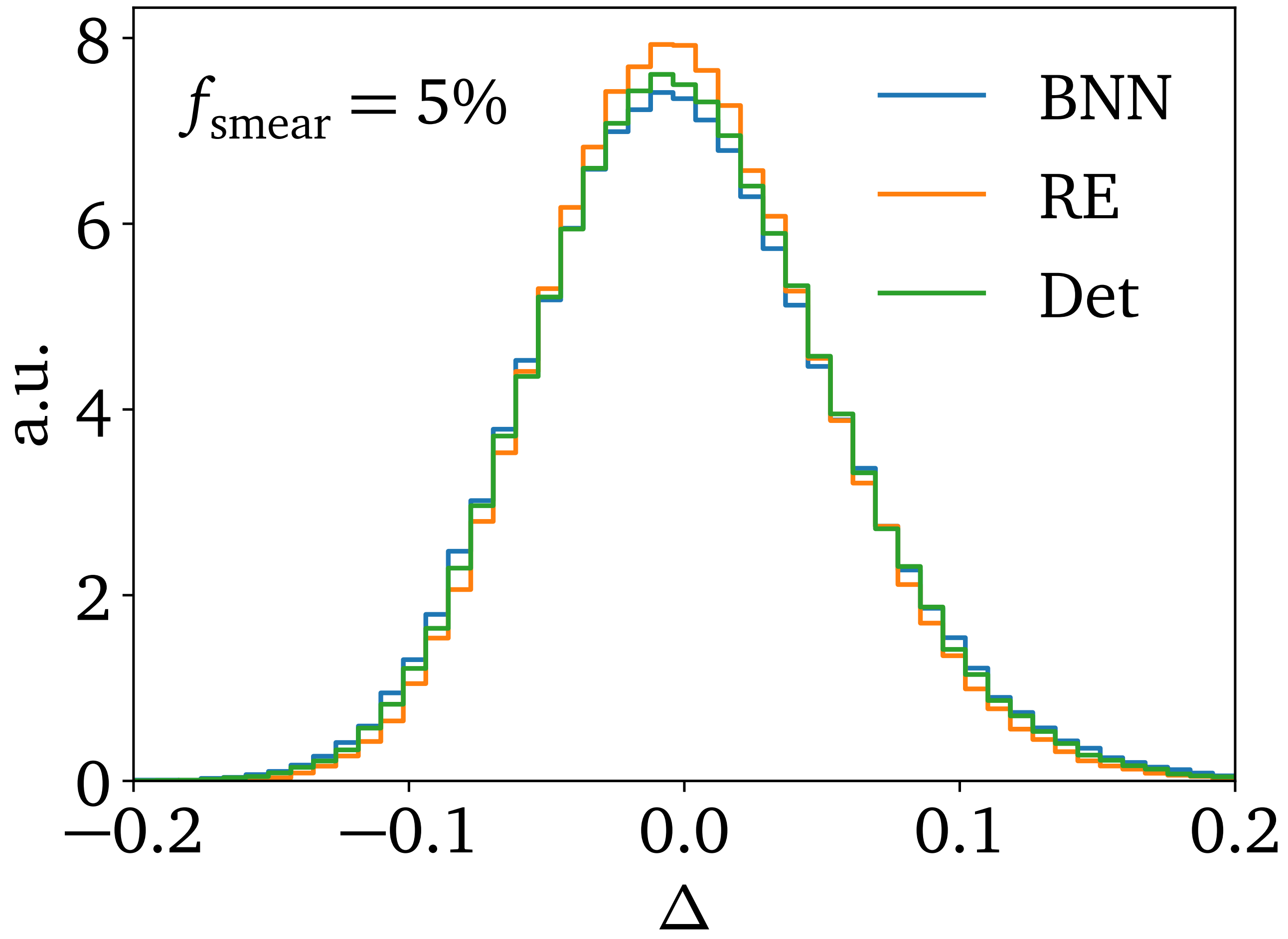
$$\Delta(x) = \frac{A_{\mathrm{NN}}(x) - A_{\mathtt{true}}(x)}{A_{\mathtt{true}}(x)}$$

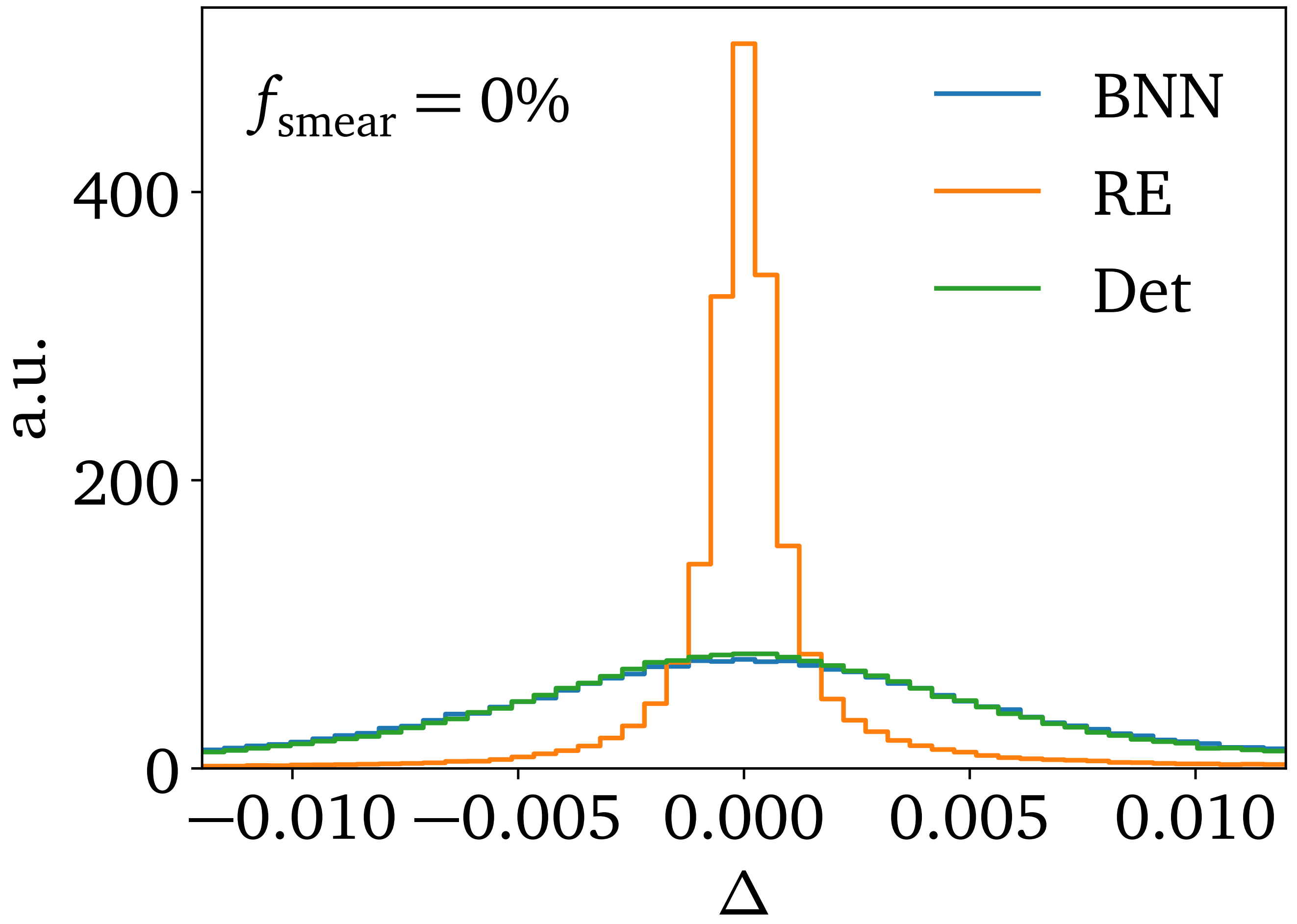
Uncertainty calibration

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

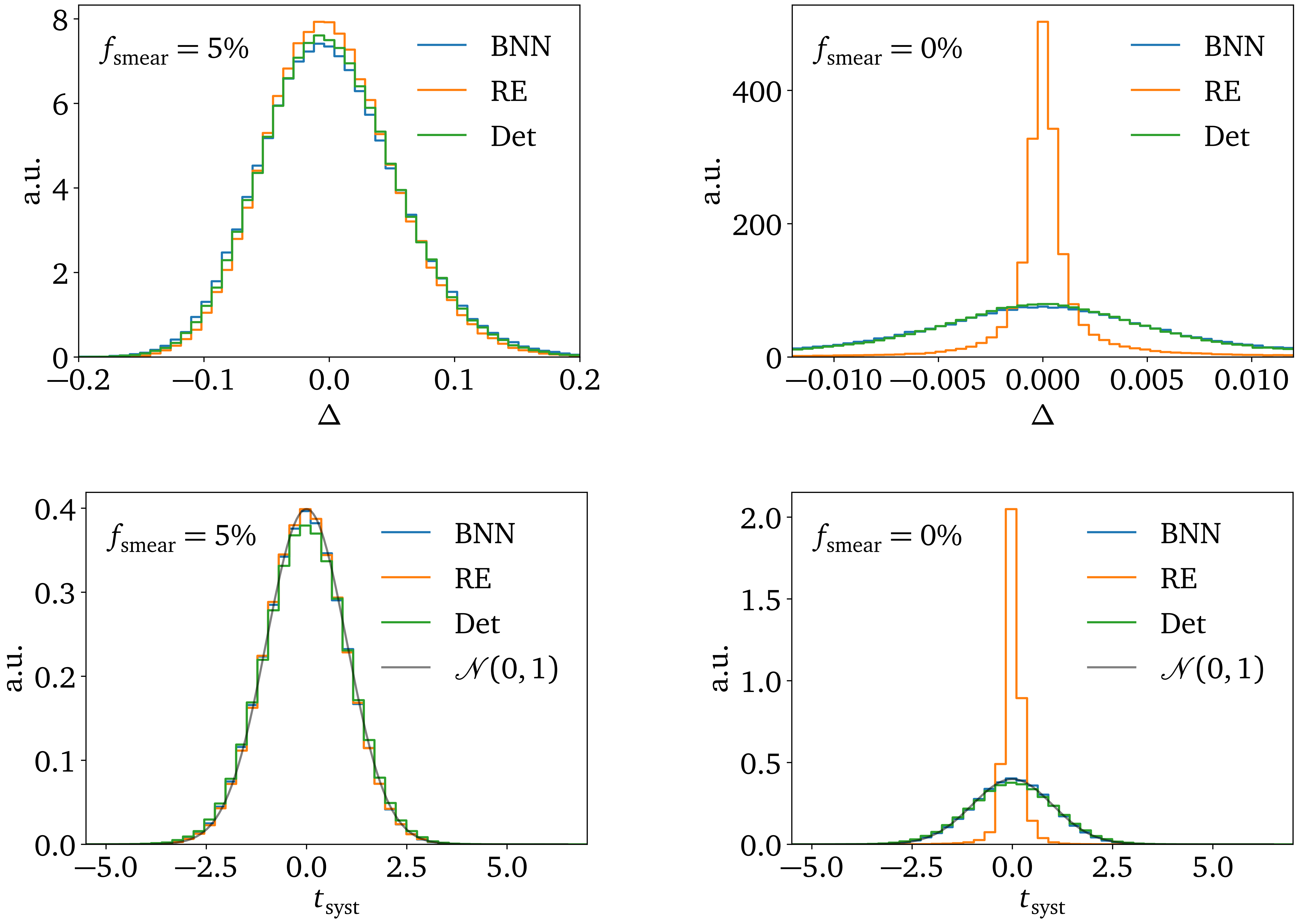
Correctly learned: follow Gaussian

Systematic pull - Results

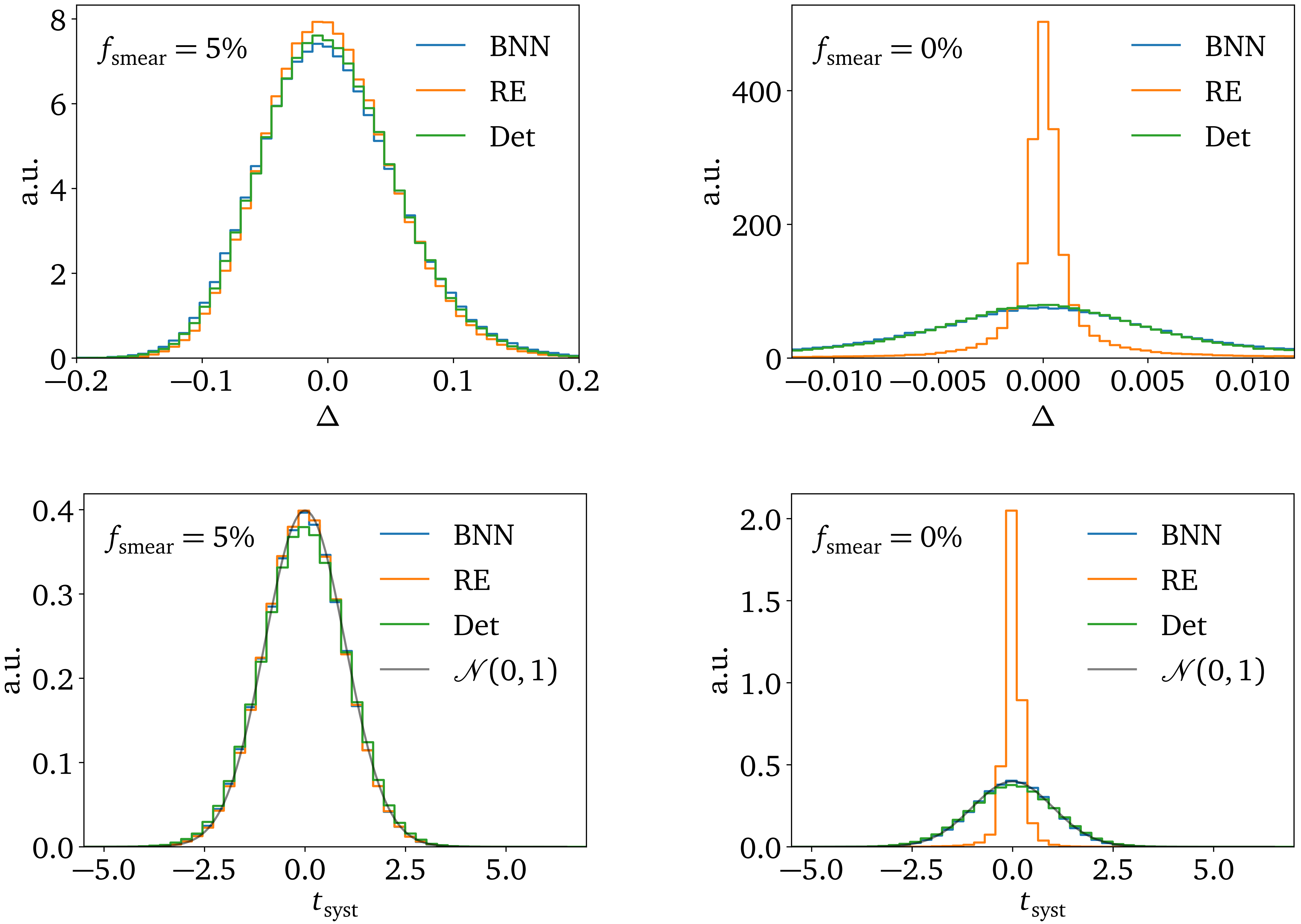




Systematic pull - Results



Systematic pull - Results



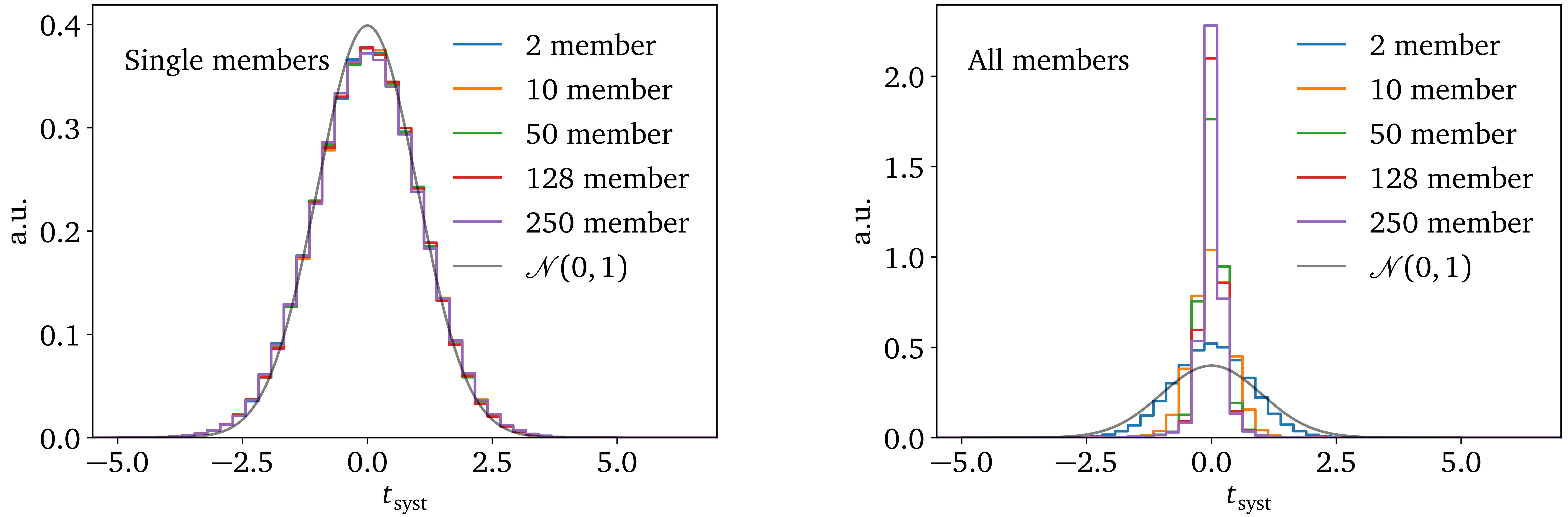
➡ BNN advantage for systematic uncertainties

Systematic pull of REs

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

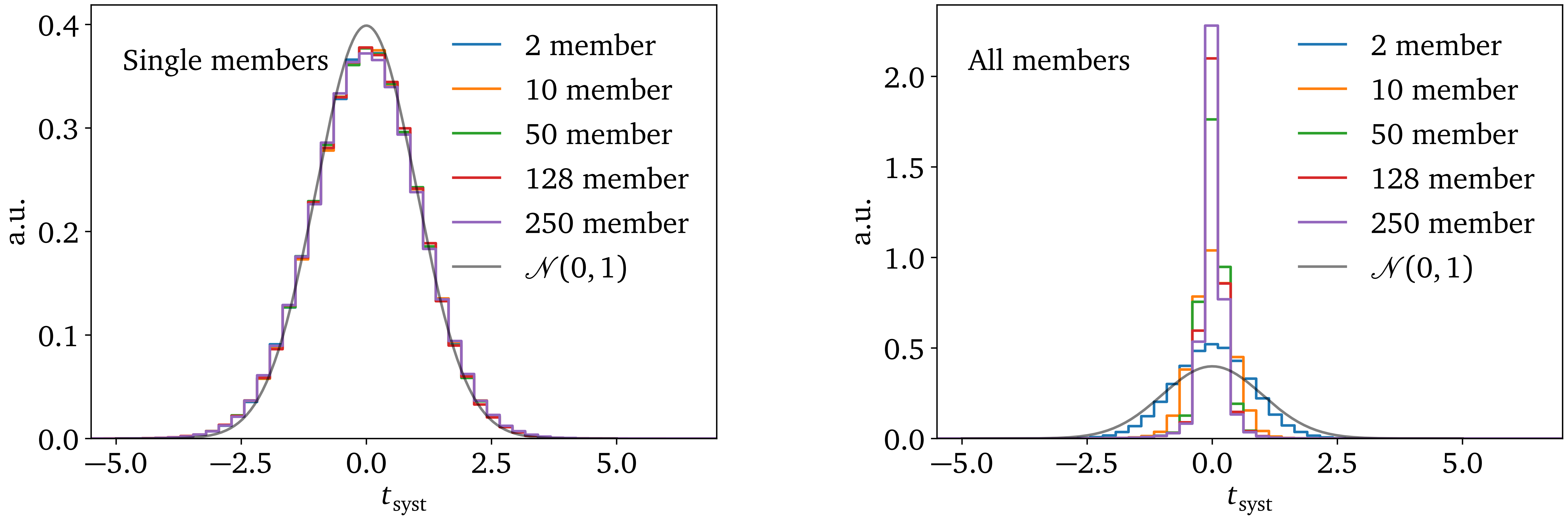
Systematic pull of REs

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



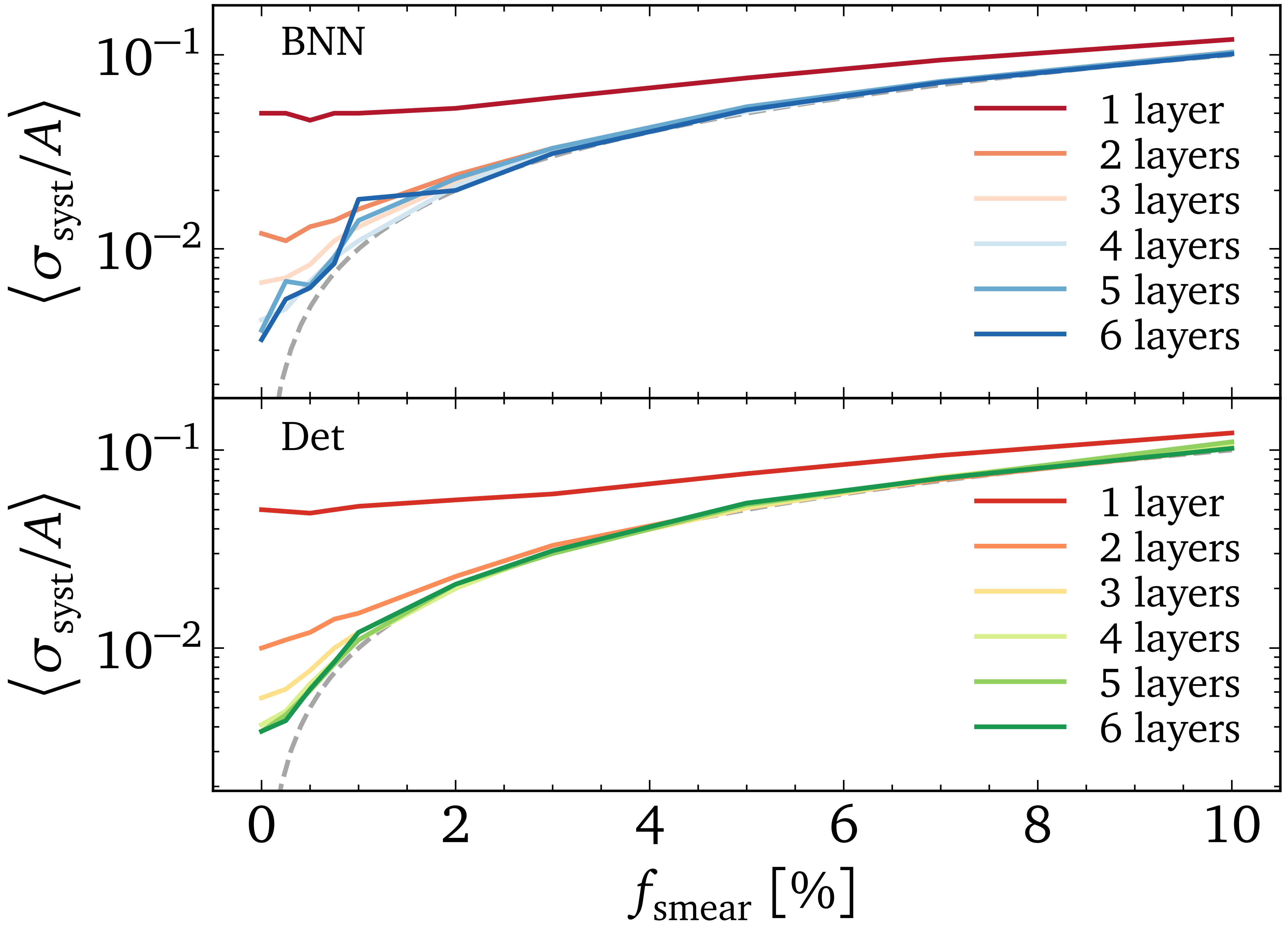
Systematic pull of REs

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

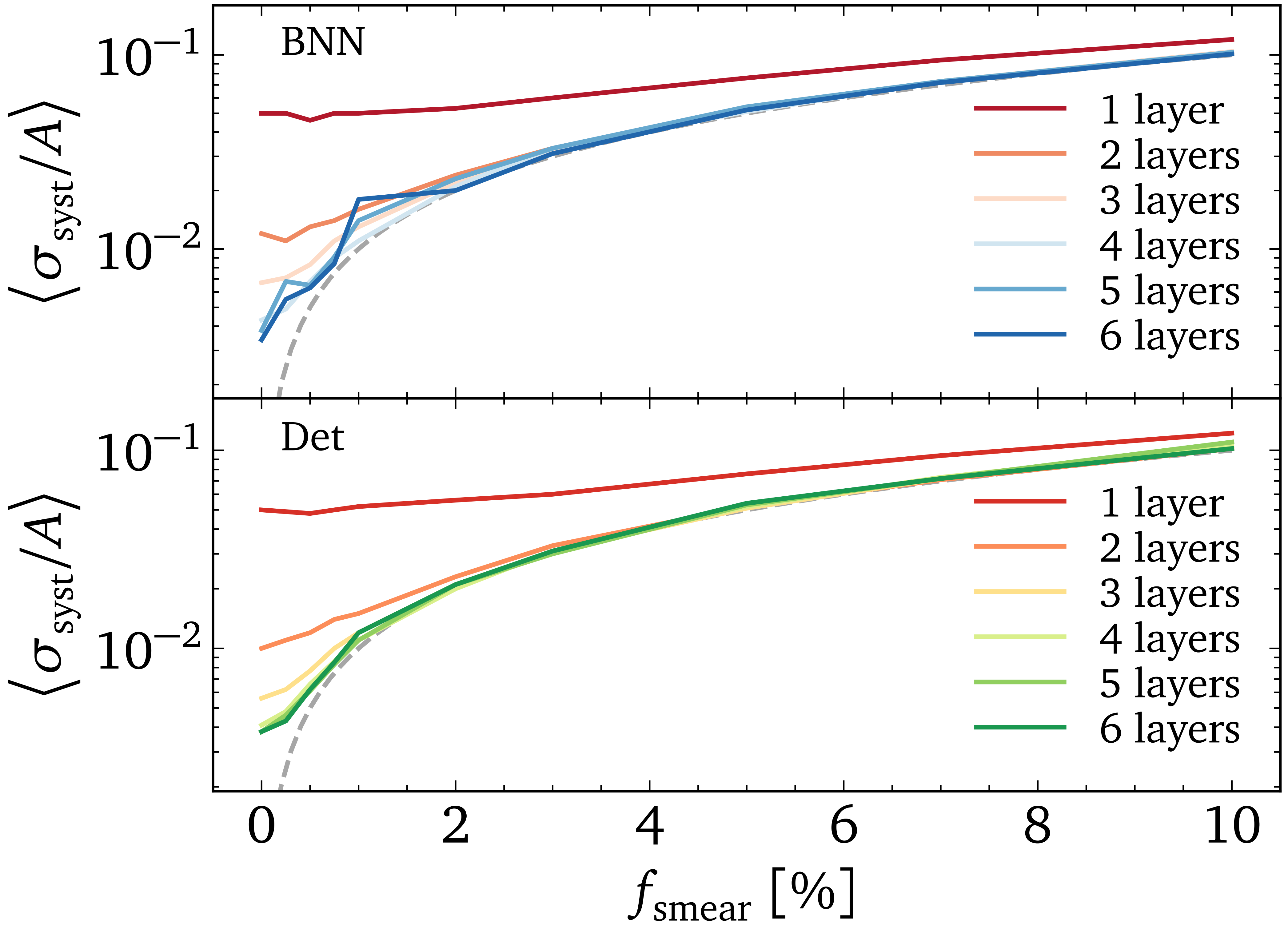


➡Prediction benefits from ensemble nature but not σ_{syst}

Systematics from network expressivity



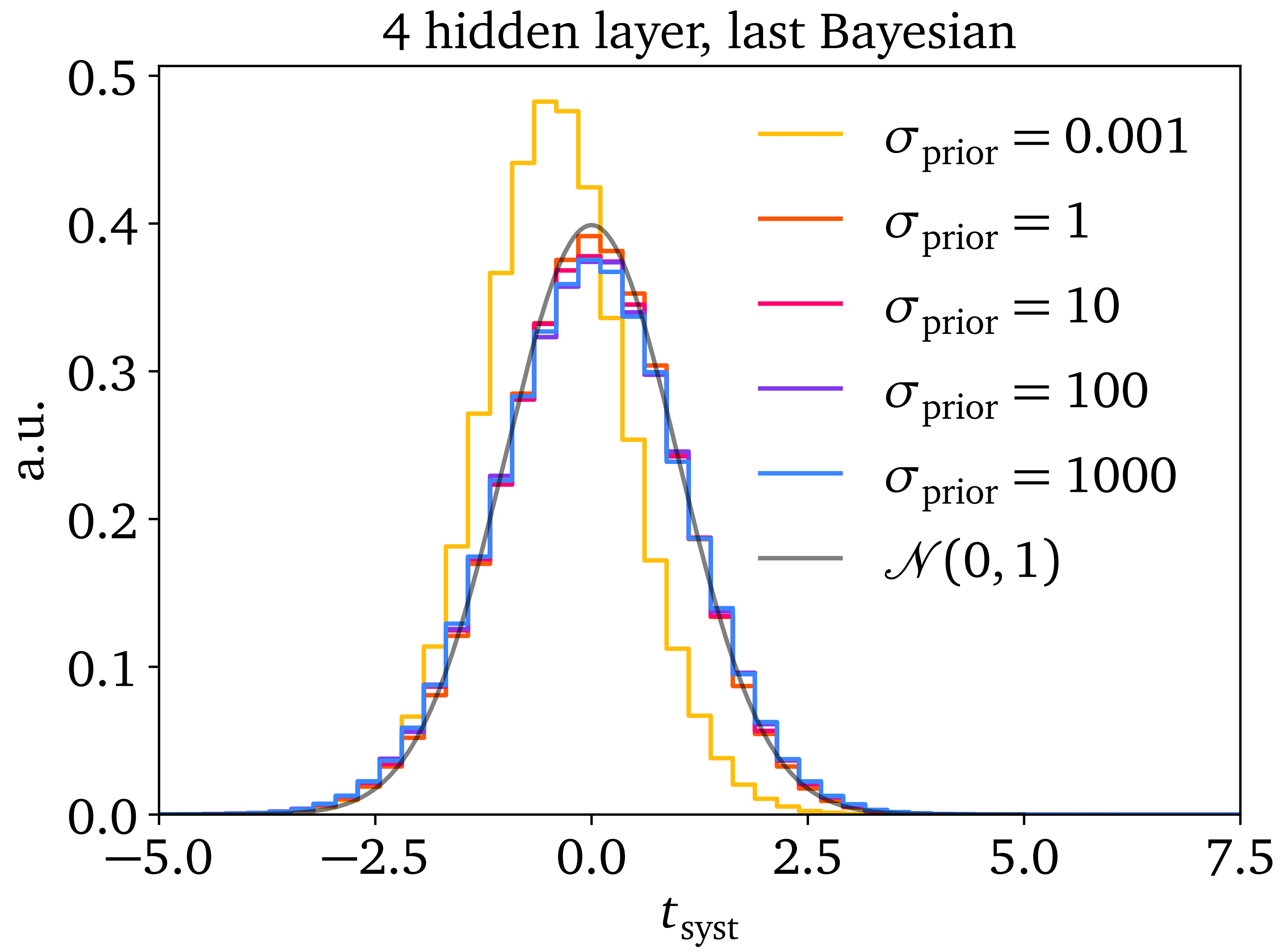
Systematics from network expressivity

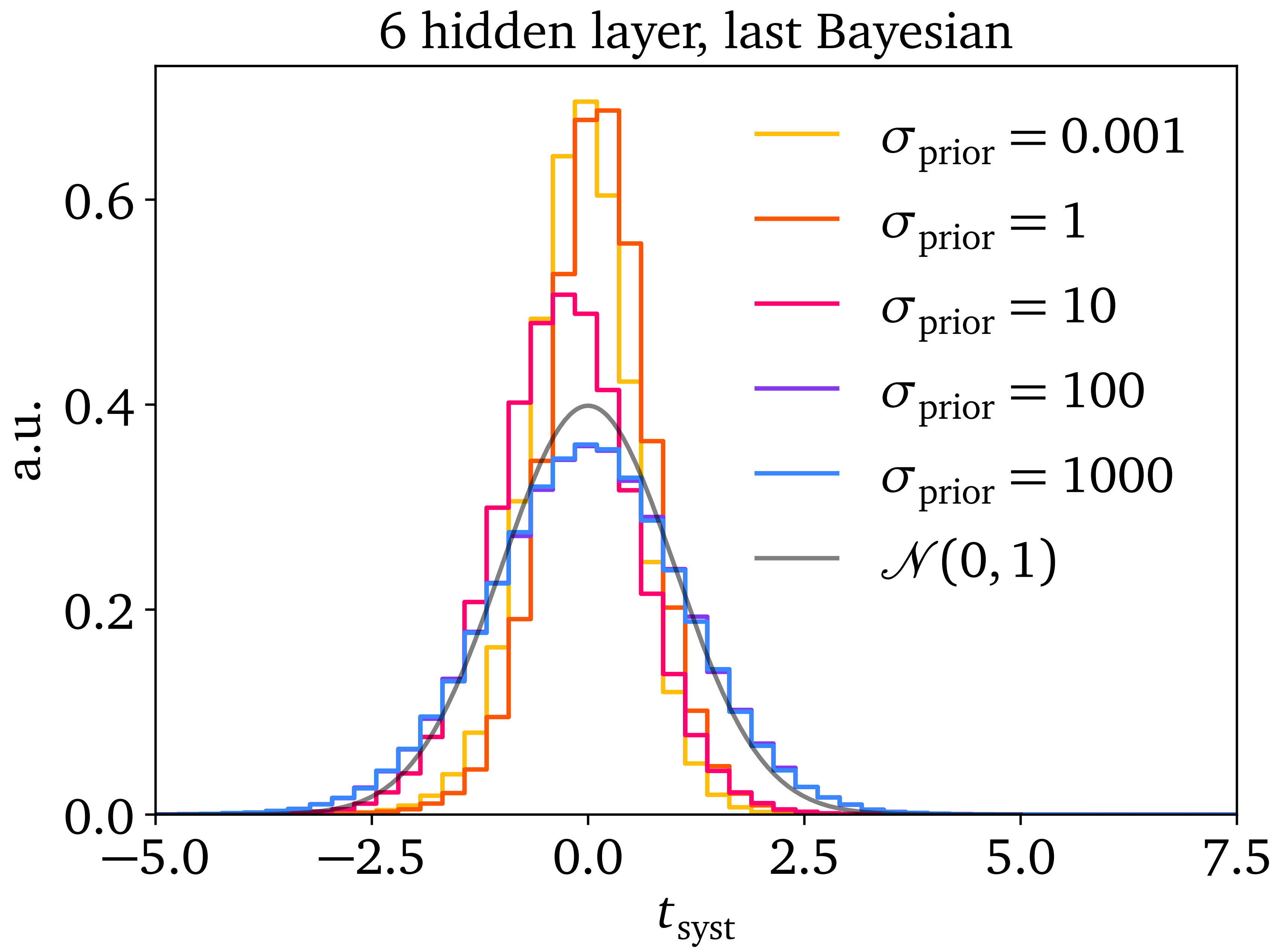


- Need at least three layers
- BNN: only last layer Bayesian
- Six layer: network gets too large

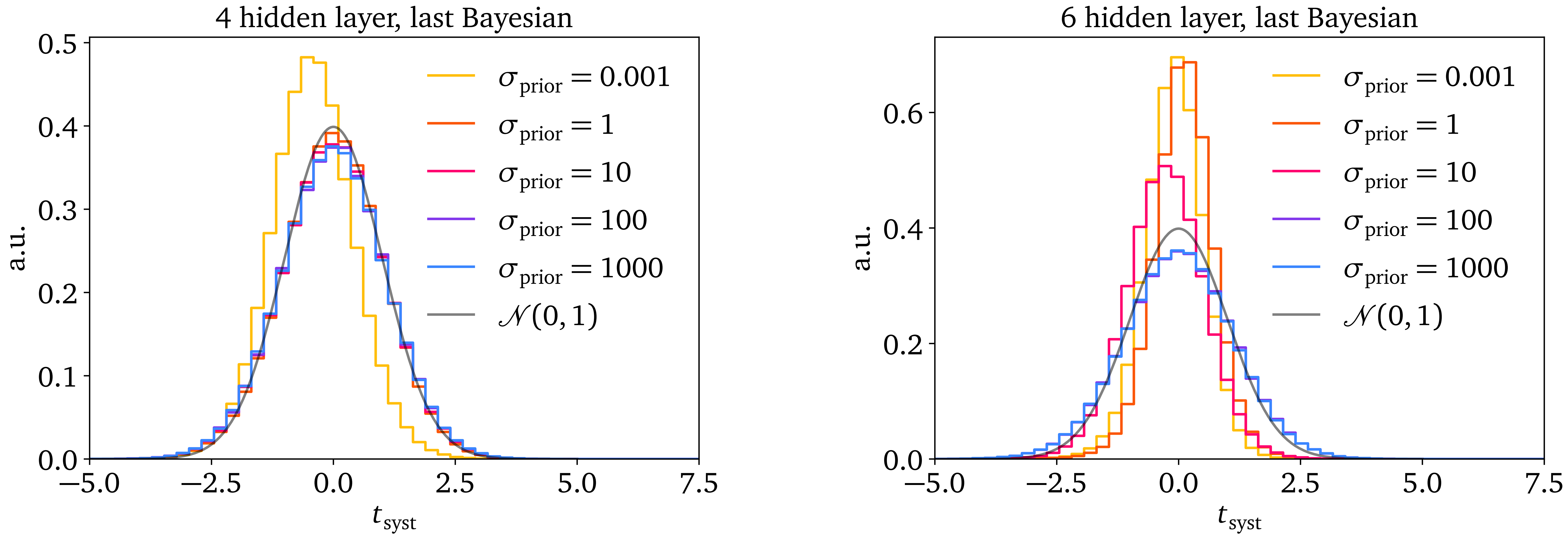
➡ More expressivity and better sensitivity for small noise with more layers

Prior influence in the BNN





Prior influence in the BNN



➡Results don't depend on prior

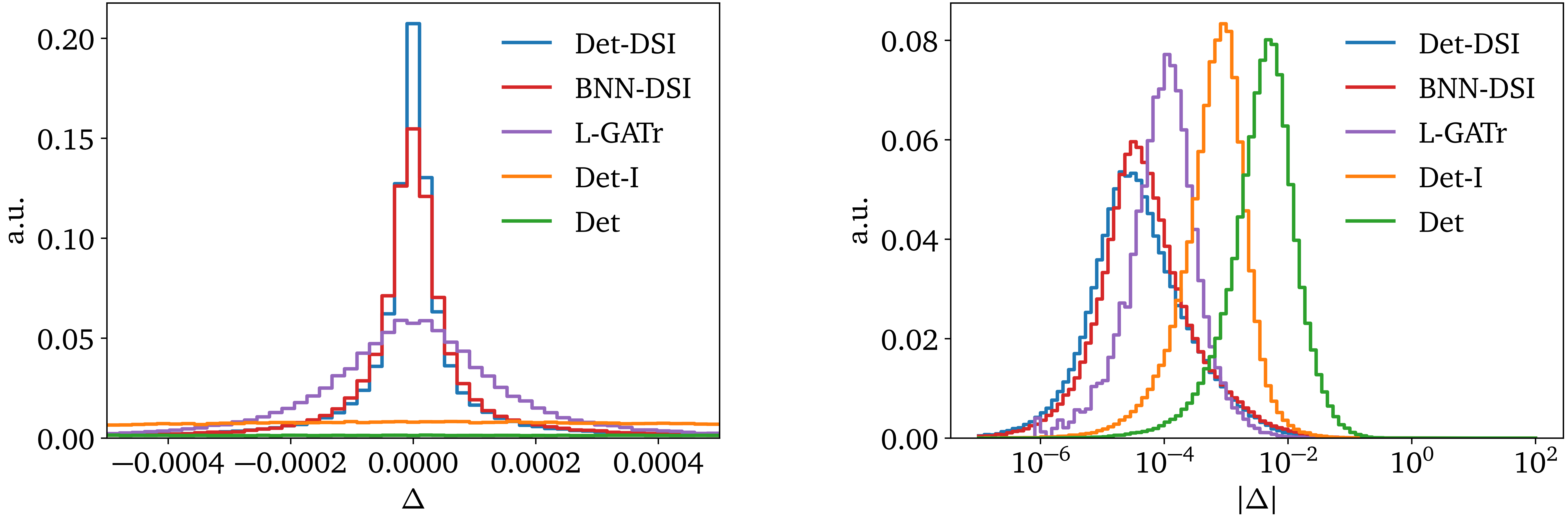
➡For more layers a larger prior is needed

Improving by using advanced architectures

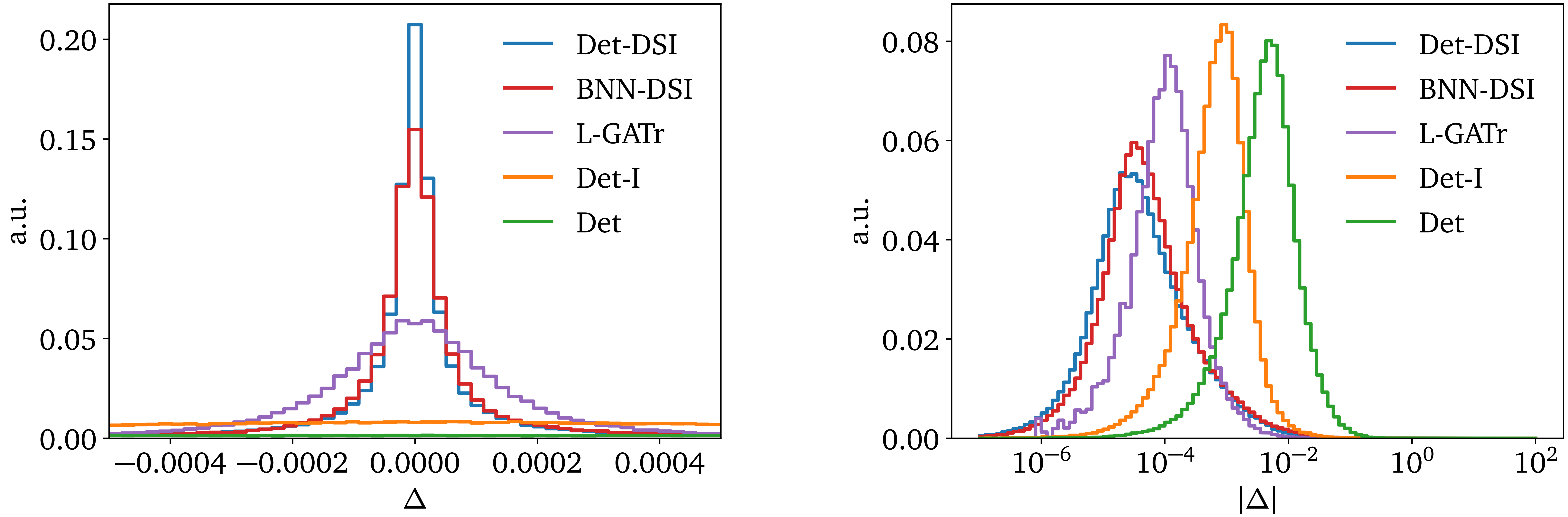
- Enhance standard networks through representation learning:

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]

Accuracy for advanced architectures

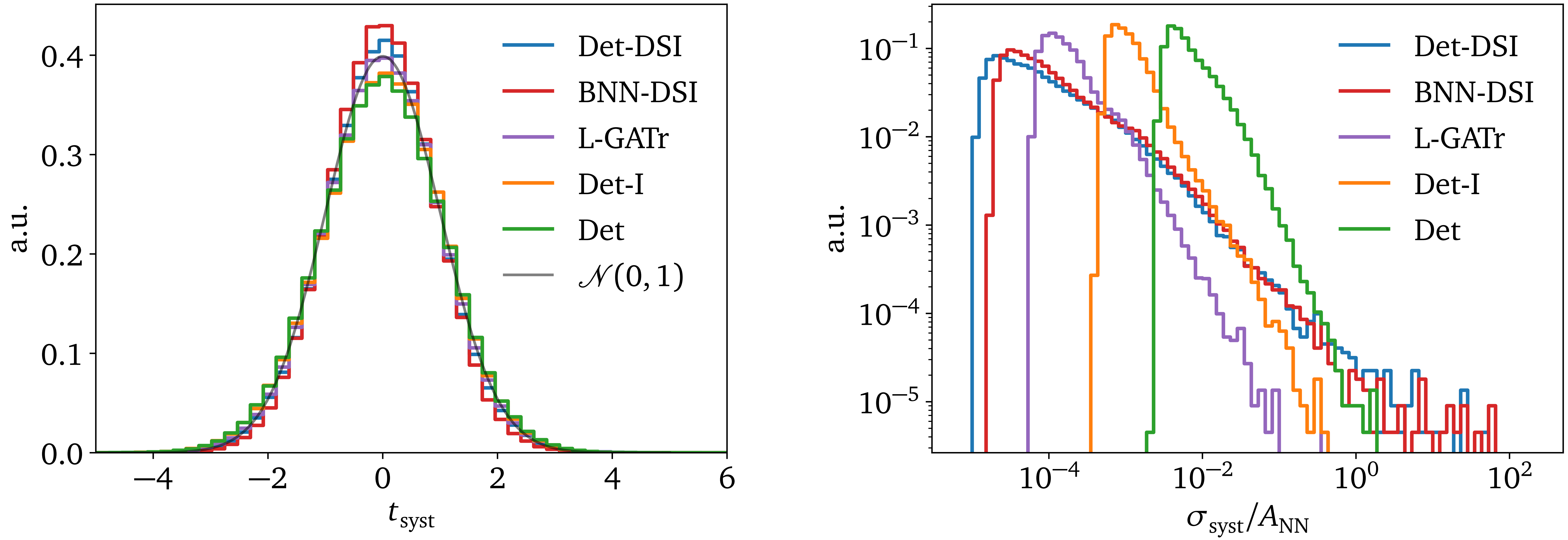


Accuracy for advanced architectures

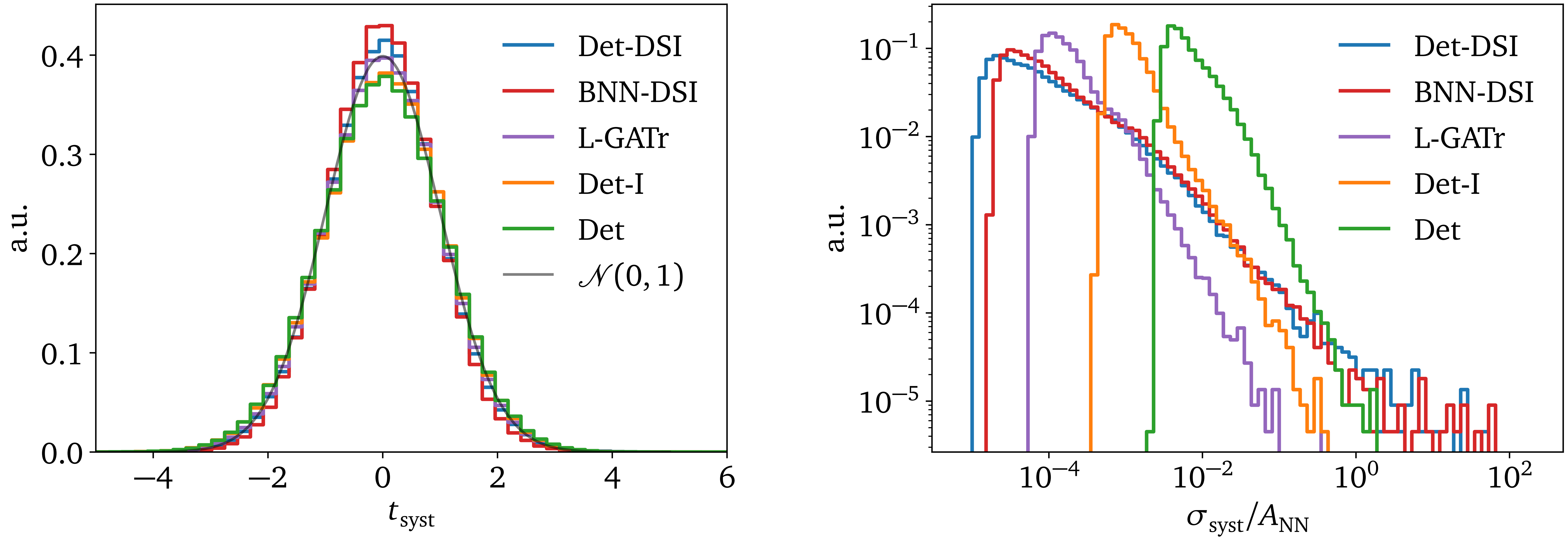


➡ **Controlled accuracy** to 10^{-5} level

Uncertainties from advanced architectures



Uncertainties from advanced architectures



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

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

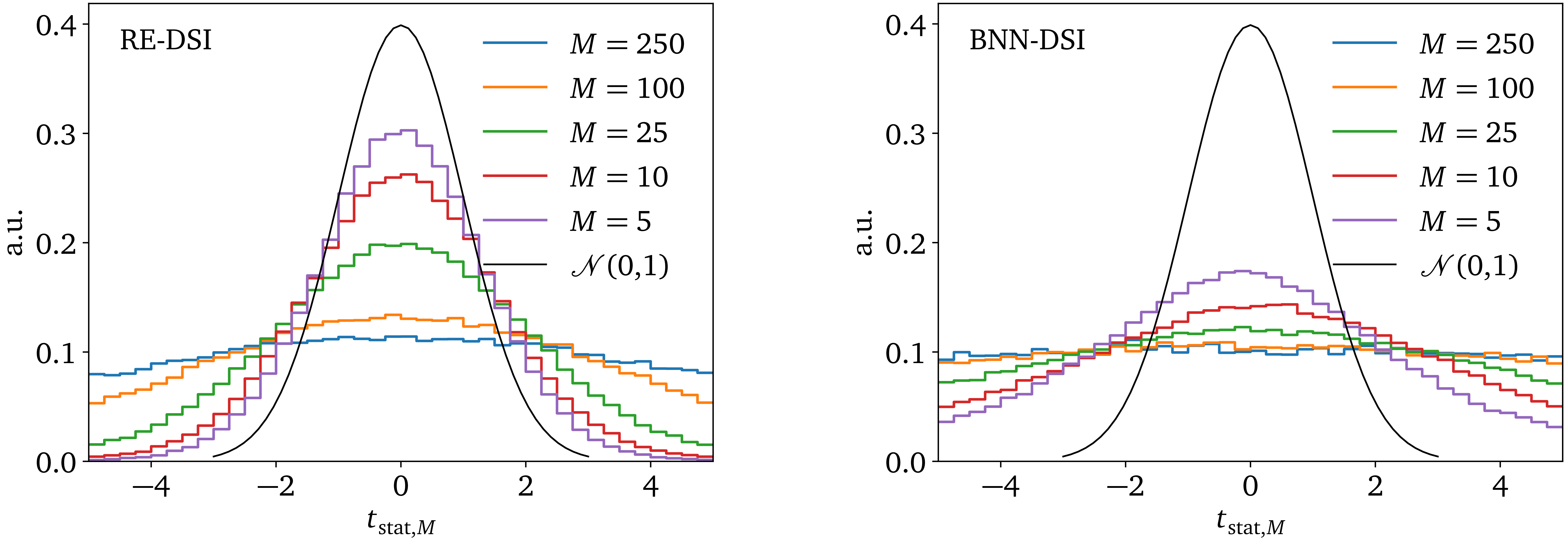
Outline

Part I: Different networks and architectures

Part II: Systematic uncertainties

Part III: **Statistical uncertainties**

Scaled statistical pull



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

Reducing the training size

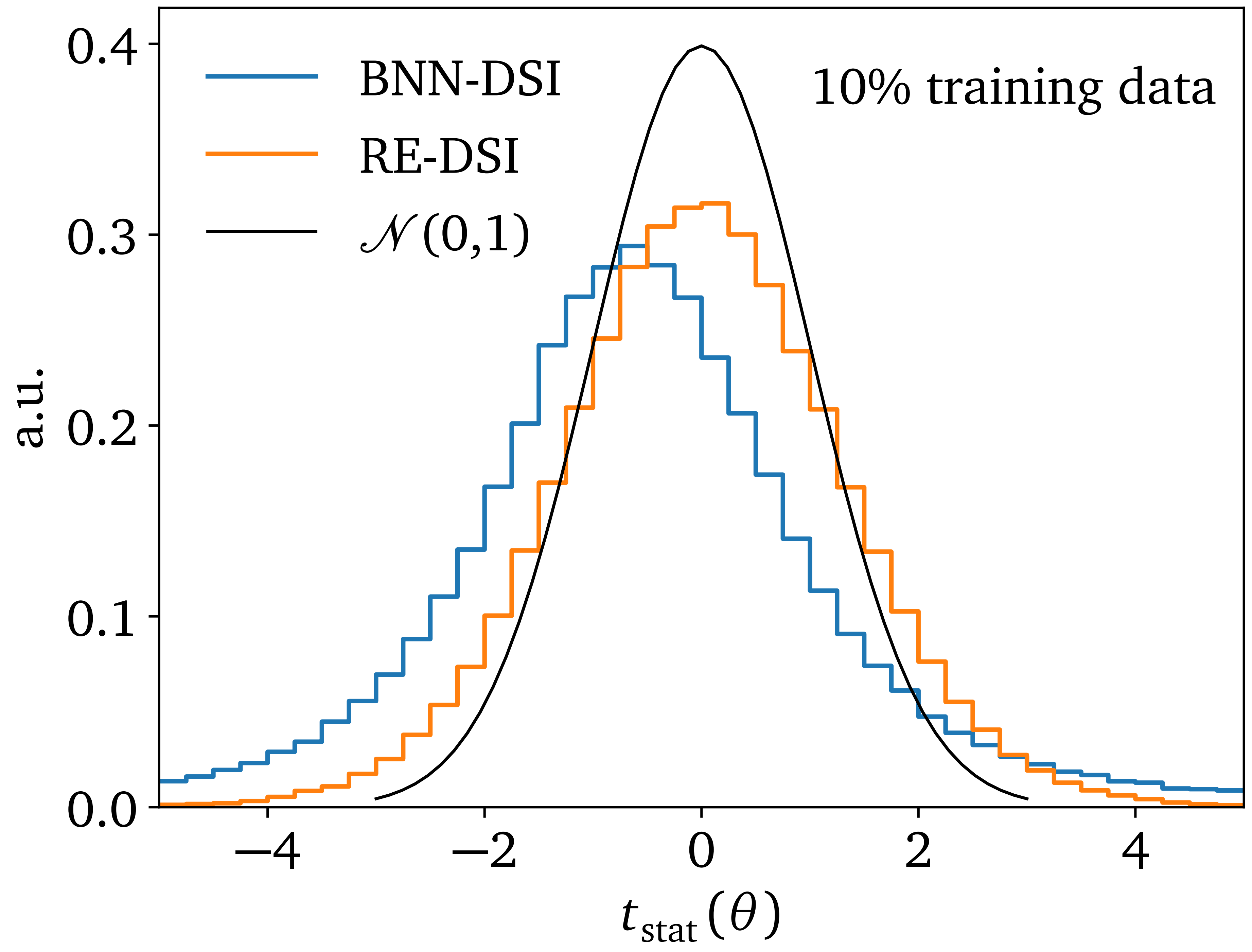
- Systematic uncertainty dominant over statsitical
- Training on 700000 phase space points: $\sigma_{\mathrm{tot}}(x) \approx \sigma_{\mathrm{syst}}(x) \gg \sigma_{\mathrm{stat}}(x)$
- Reducing training data to 100000 phase space points

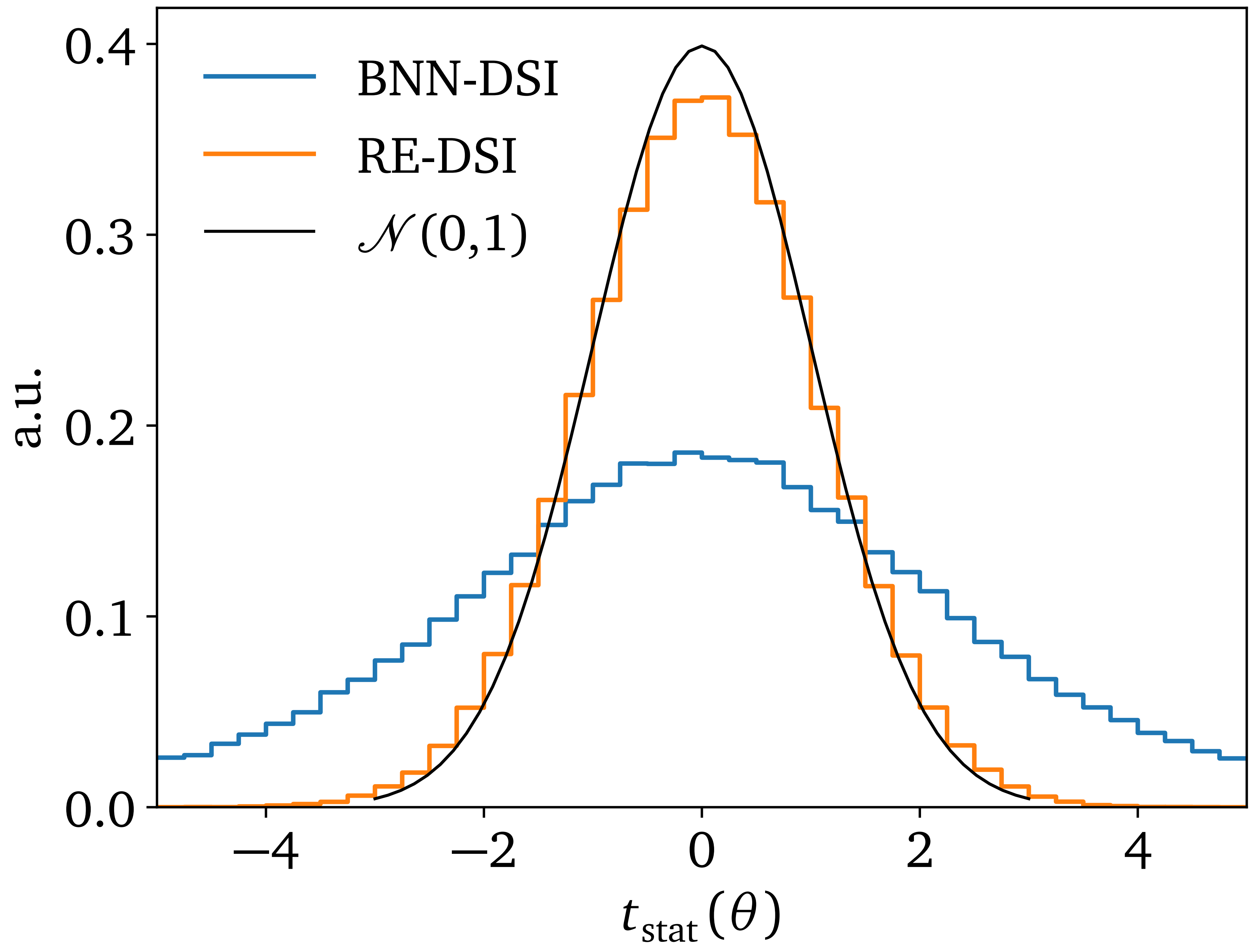
Reducing the training size

- 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

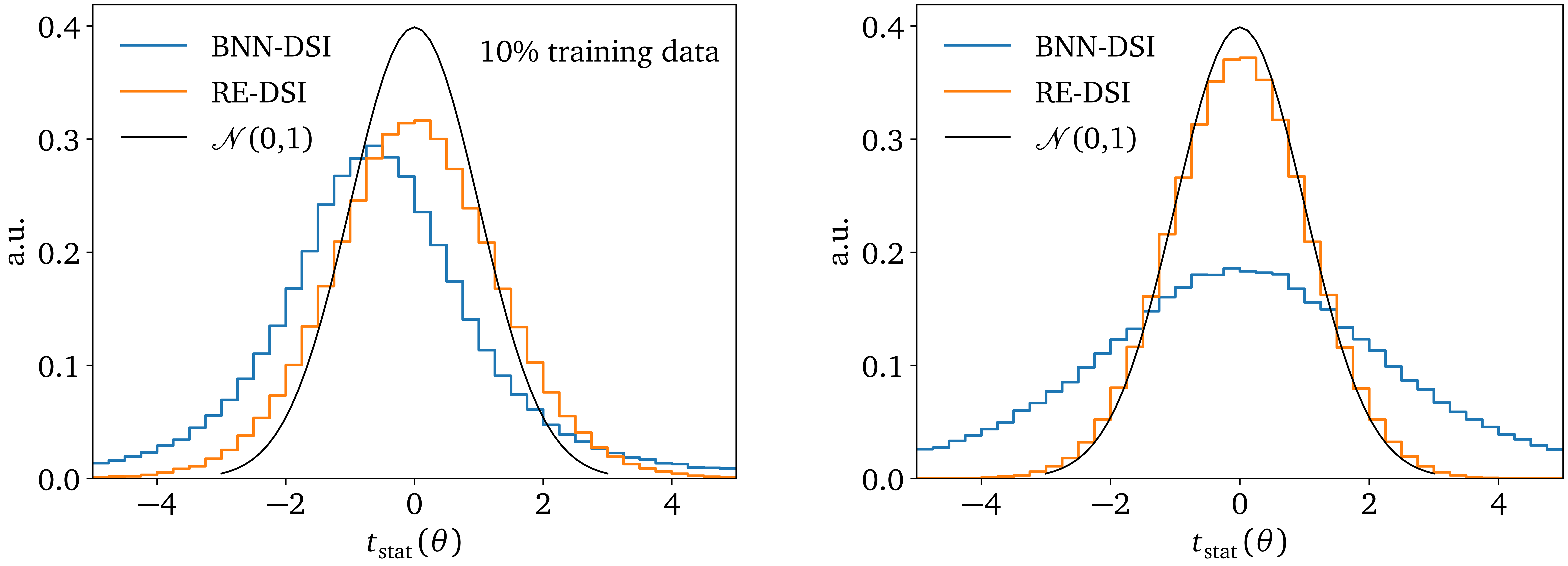
	70%	10%
$\langle \sigma_{\text{syst, BNN-DSI}}/A \rangle$	$8.7 \cdot 10^{-5}$	$2.5 \cdot 10^{-4}$
$\langle \sigma_{\text{stat, BNN-DSI}}/A \rangle$	$3.6 \cdot 10^{-5}$	$1.5 \cdot 10^{-4}$
$\langle \sigma_{\text{syst, RE-DSI}}/A \rangle$	$5.1 \cdot 10^{-5}$	$2.9 \cdot 10^{-4}$
$\langle \sigma_{\text{stat,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

Dependence on training size



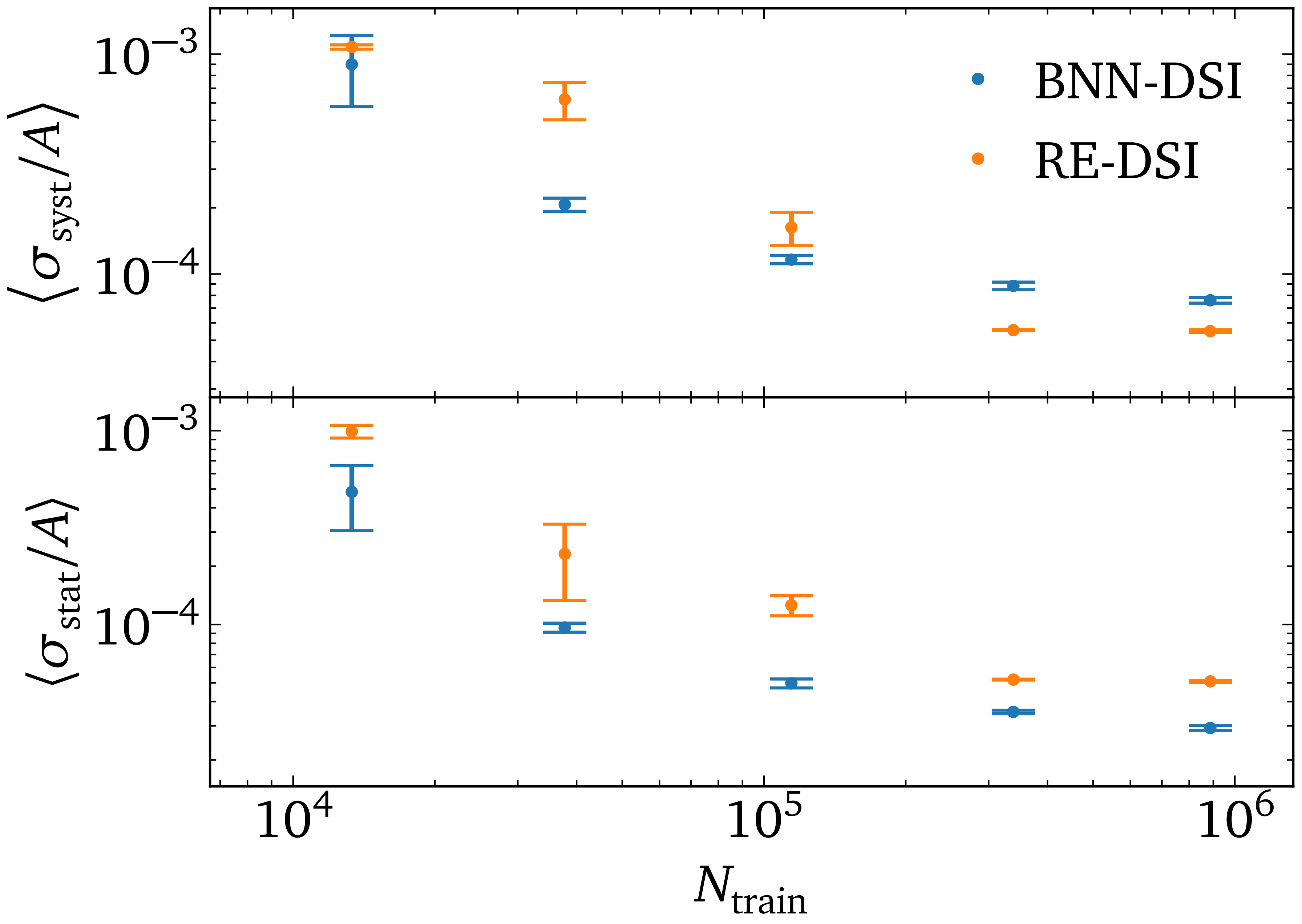


Dependence on training size

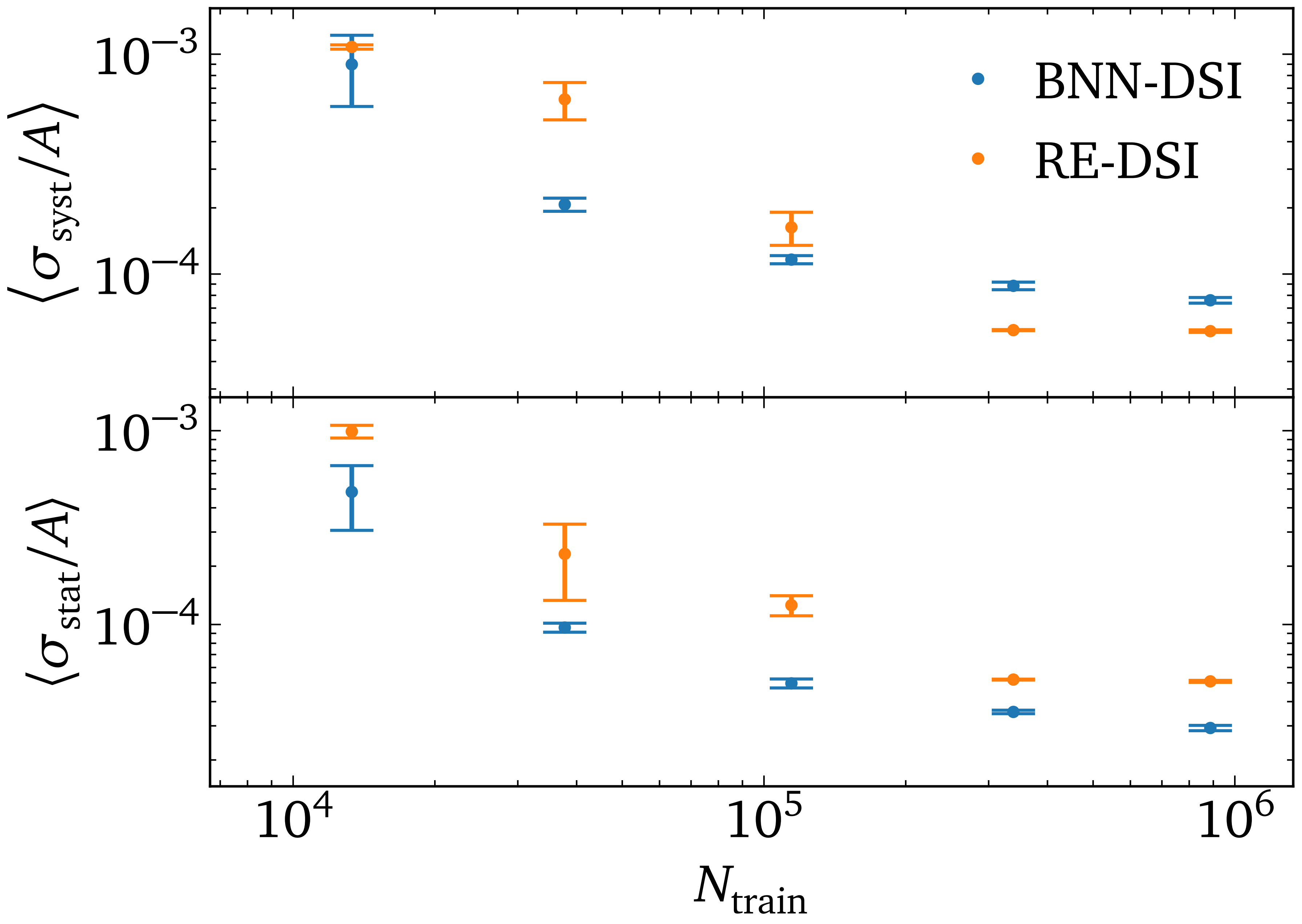


- Repulsive ensemble advantage in statistical uncertainty

Relative uncertainty vs training size



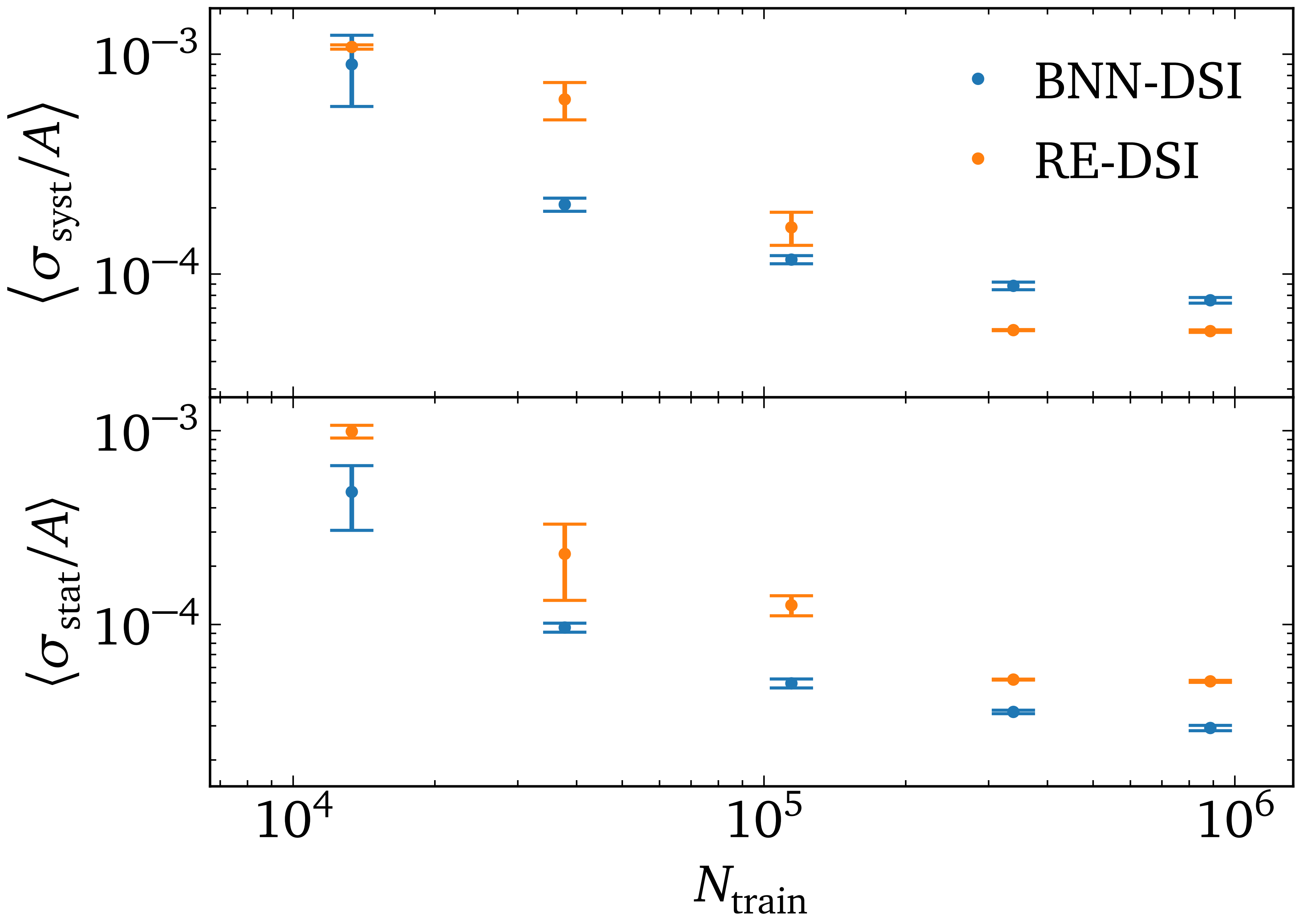
Relative uncertainty vs training size

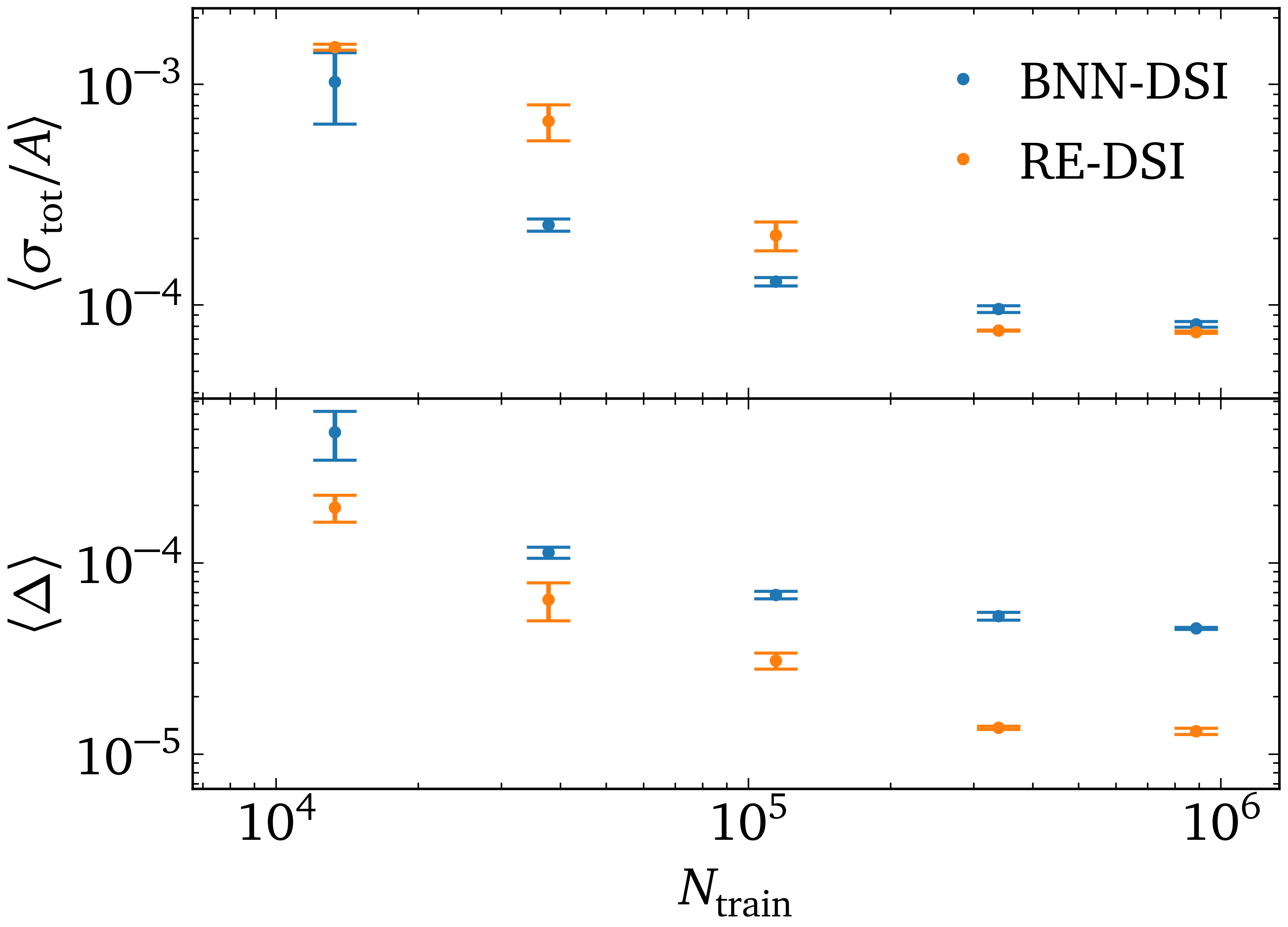


➡σ_{syst} always larger

➡σ larger for RE-DSI than BNN-DSI

Relative uncertainty vs training size

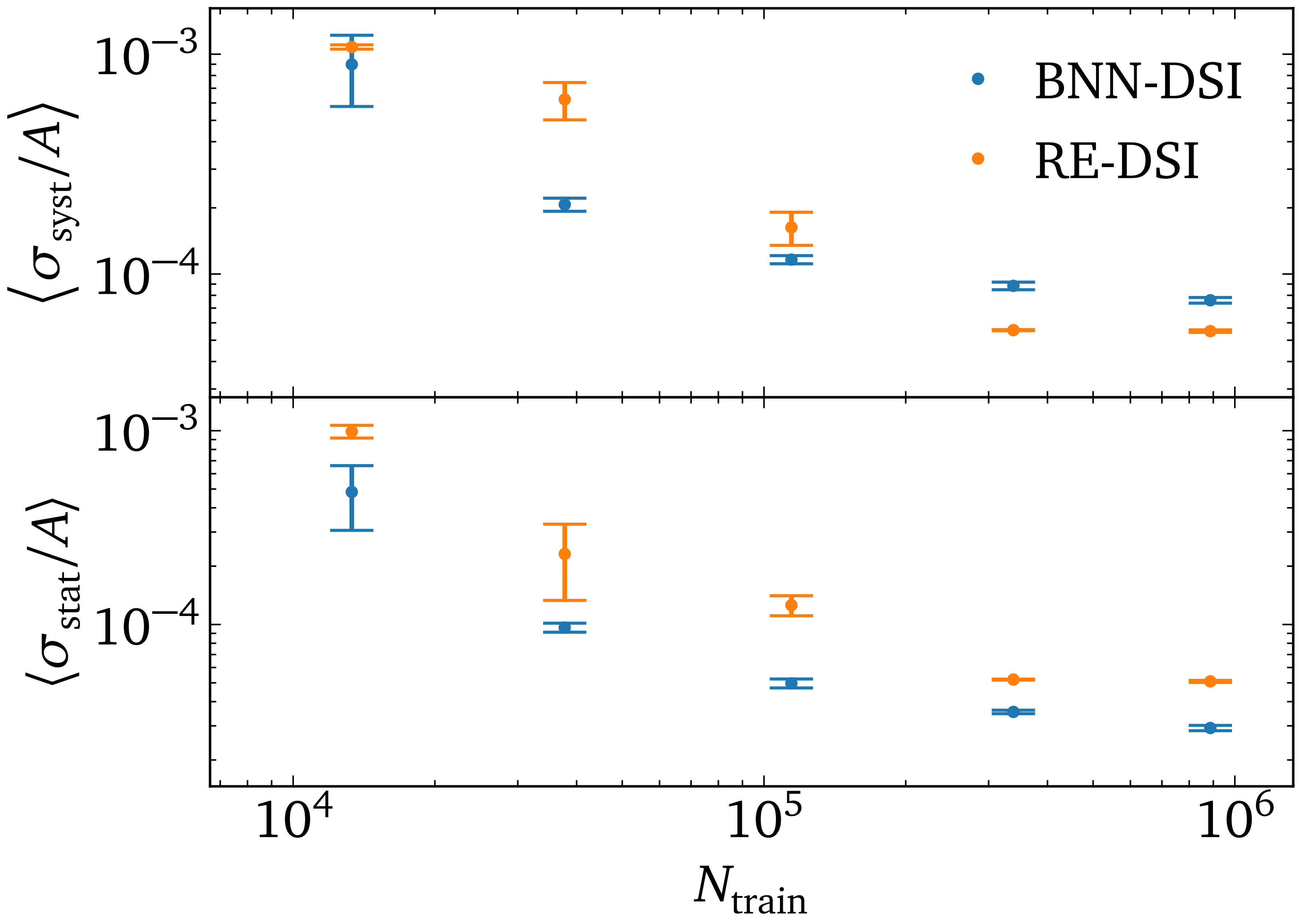


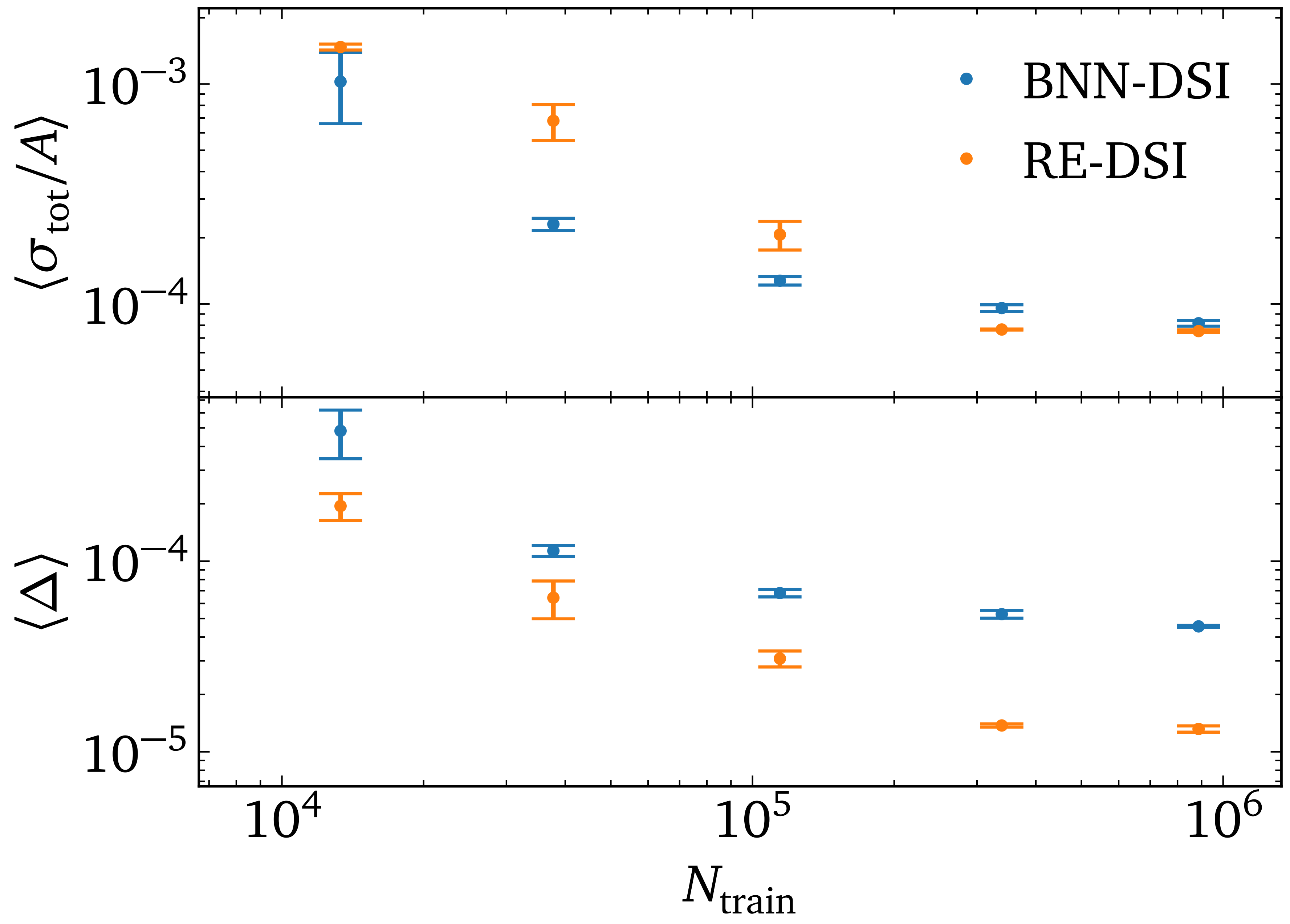


➡ σ_{syst} always larger

➡ σ larger for RE-DSI than BNN-DSI

Relative uncertainty vs training size





➡σ_{syst} always larger

➡σ larger for RE-DSI than BNN-DSI

➡Difference in σ_{tot} for small data

➡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!