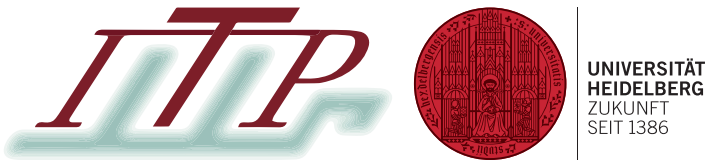
Accurate Surrogate Amplitudes with Calibrated Uncertainties

Henning Bahl

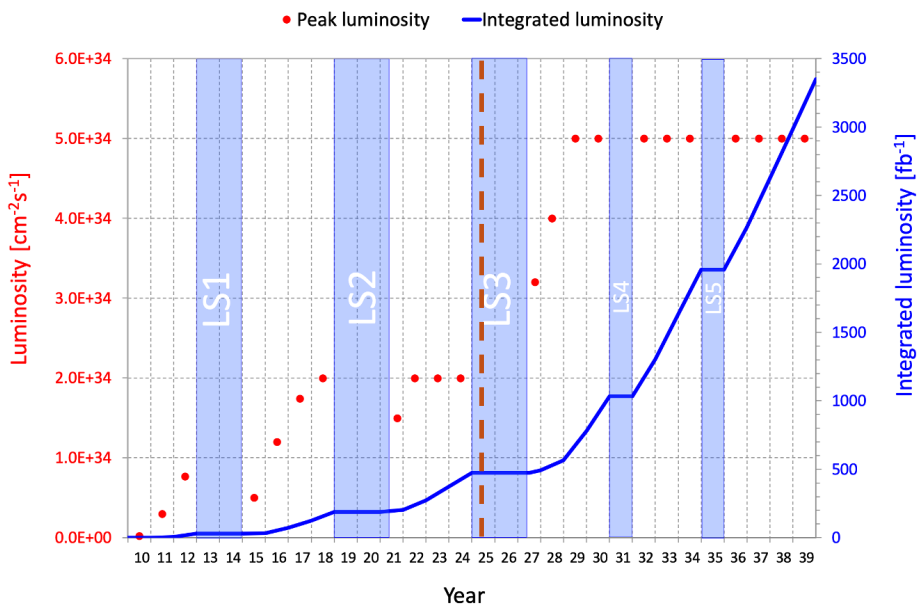
based on
2412.12069 in collaboration with Nina Elmer, Luigi Favaro,
Manuel Haußmann, Tilman Plehn, and Ramon Winterhalder



Particle Theory Seminar, Würzburg, 24.7.2025

The challenge ahead

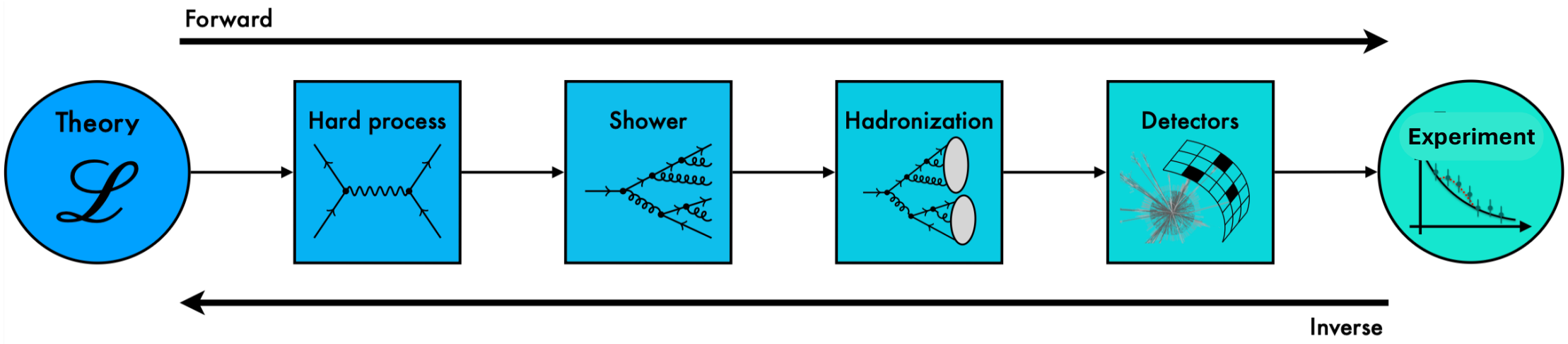
- general trend: larger-and-larger experiments collecting more-and-more data
- e.g. LHC: already enormous dataset will be further enlarged by a factor ~ 10
- costs for future experiments increasing



Fully exploit the available data!

- new analysis methods
- theory precision $\simeq$ experimental precision
- in particular: high-precision MC simulation

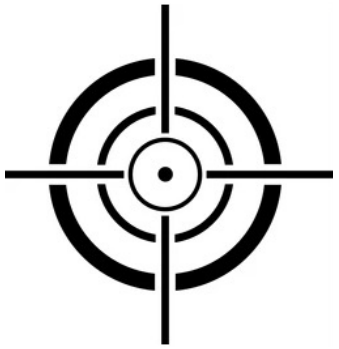
The particle physics workflow



ML can help with each of these steps by increasing

- accuracy/performance and/or
- increase speed

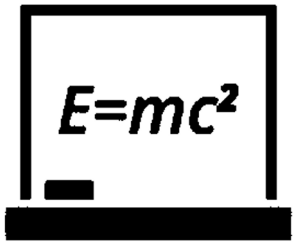
ML for particle physics



precision

speed

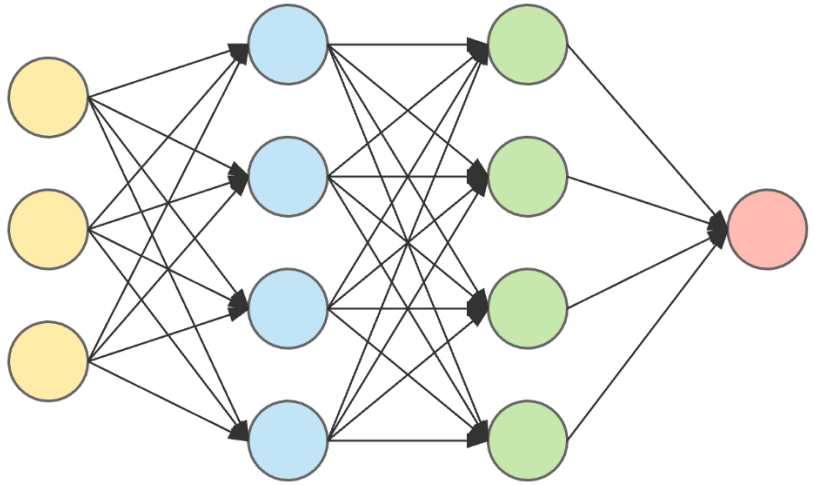
control



physics

Amplitude surrogates

$$|\mathcal{M}|^2 \approx$$



Case study for amplitude surrogates

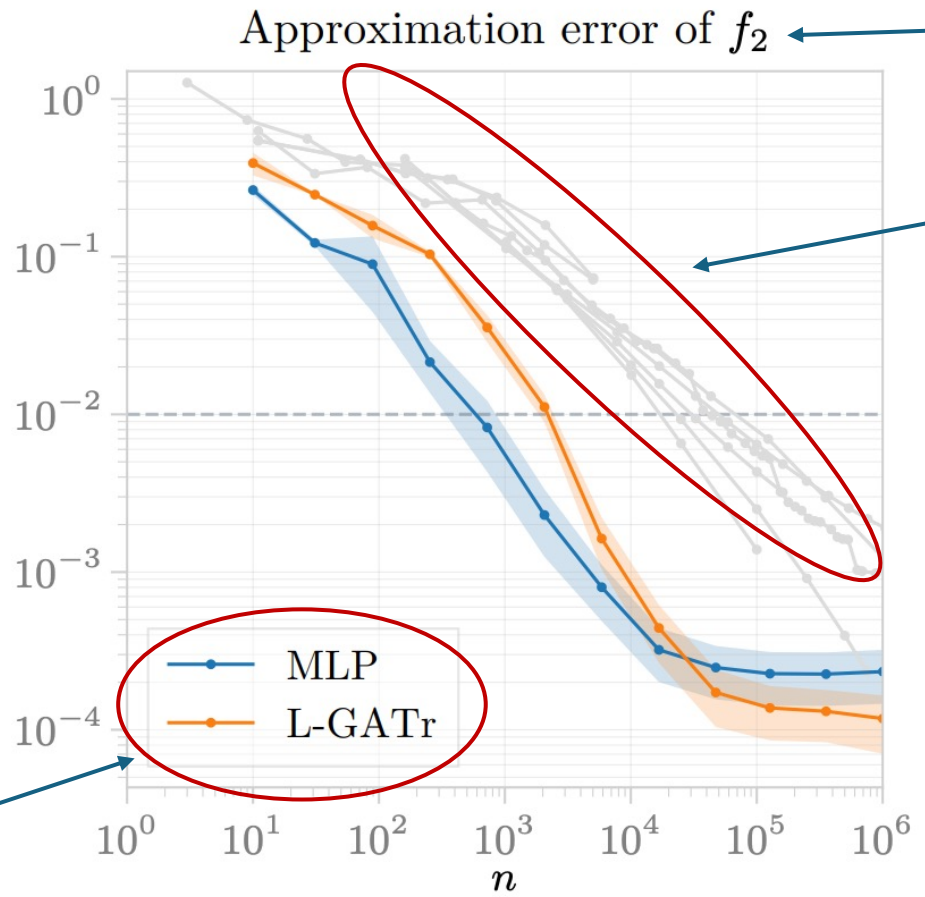
- evaluating analytic expressions for amplitudes $|\mathcal{M}|^2$ can be very expensive due to
  - higher-order corrections
  - large final-state multiplicities

- possible solution:
  - generate small training sample using full analytic expression
  - train a NN to approximate $|\mathcal{M}|^2$
  - generate events using NN surrogate, which is much faster to evaluate

➡ Does this work?

Comparison to classical interpolation

[Bresó et al., 2412.09534]



NLO $q\bar{q} \to t\bar{t}H$ amplitude

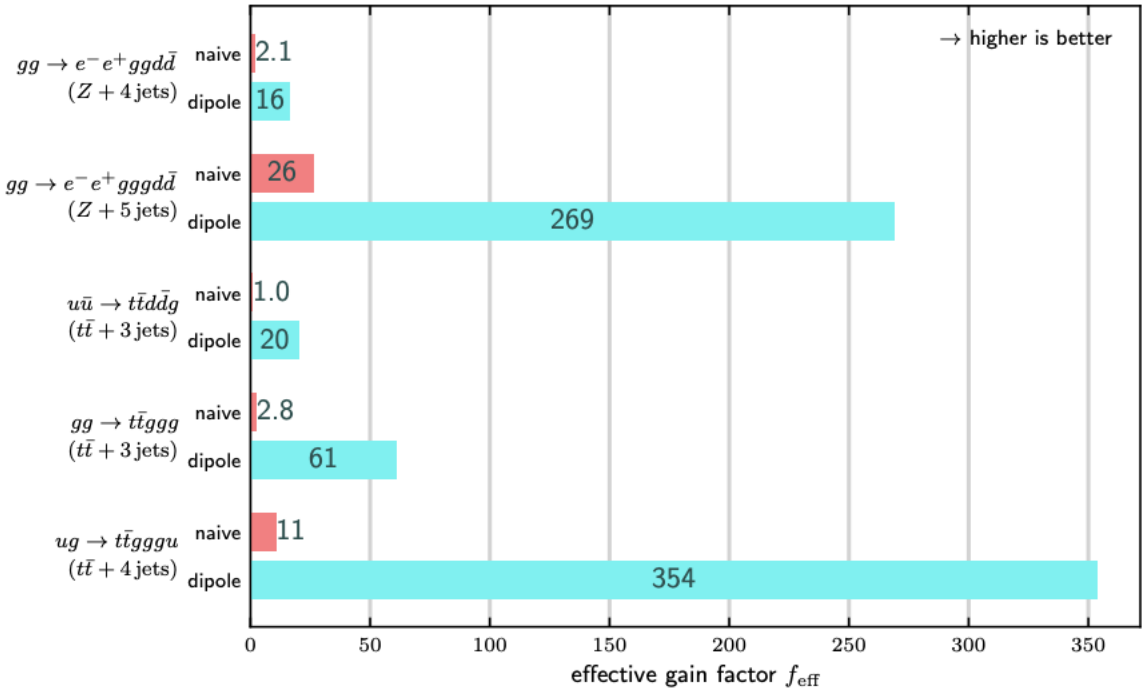
classical interpolation techniques

ML surrogates

ML surrogates outperform classical interpolation techniques

Speed comparison

[Janßen et al.,2301.13562]



	SHERPA default			with dipole-model surrogate				
Process	t_{ME}[ms]	t_{PS}[ms]	ϵ_{full}	t_{surr}[ms]	$x_{\max}$	$\epsilon_{\text{1st,surr}}$	$\epsilon_{\text{2nd,surr}}$	f_{eff}
$gg \to e^- e^+ ggd\bar{d}$	54	0.40	1.411 %	0.14	2.6	1.418 %	39 %	16
$gg \to e^- e^+ gggd\bar{d}$	16 216	5.70	0.076 %	0.20	3.6	0.085 %	29 %	269

$$f_{\text{eff}} = \frac{T_{\text{standard}}}{T_{\text{surrogate}}}$$

dipole vs naïve:
encode singularity structure of amplitudes

→ Large speed-ups possible!

↓

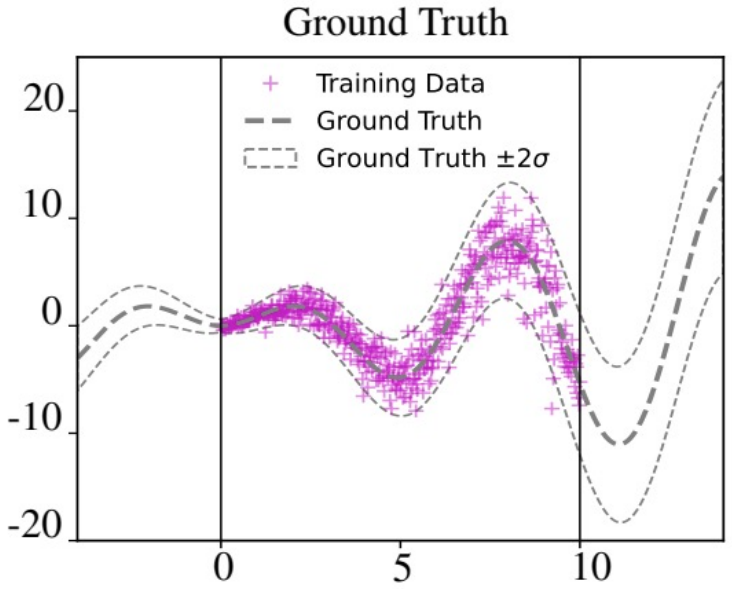
Can we also control the uncertainties?

Learning uncertainties

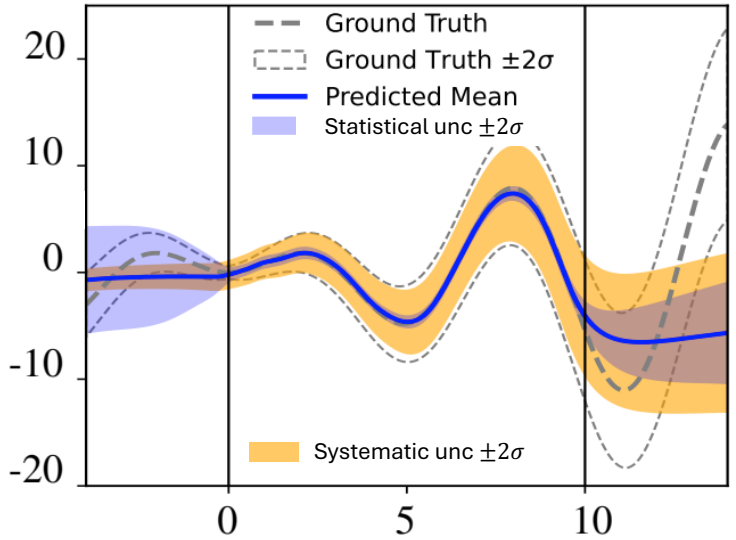
"All models are wrong, but some — those that know when they can be trusted — are useful!"

— George Box (adapted)

Regression with uncertainties



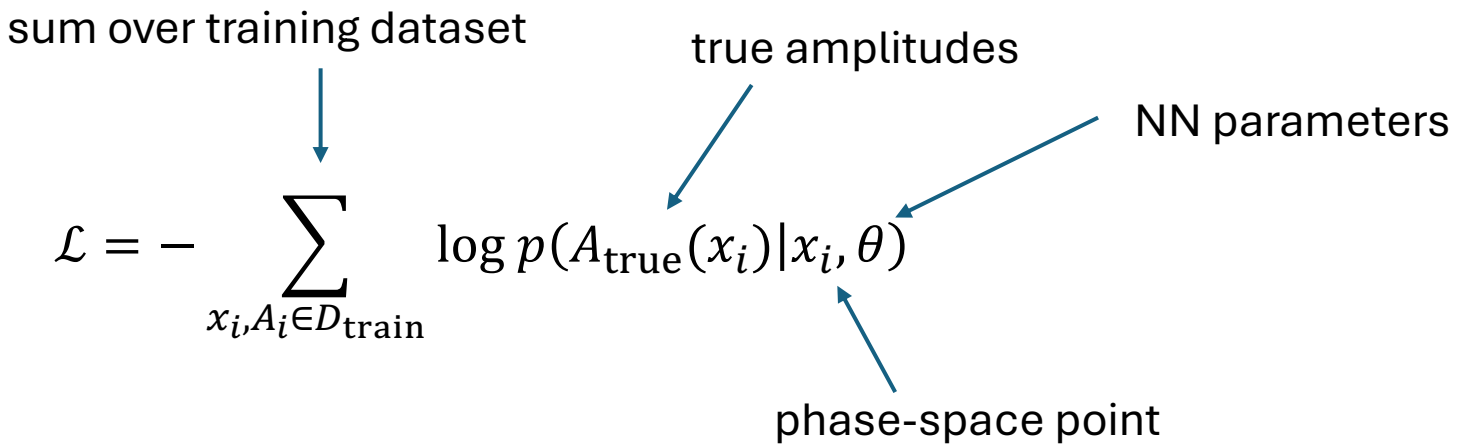
[Yi&Bessa, 2505.02743]



- statistical or epistemic uncertainty ≙ lack of training data
- systematic or aleatoric uncertainty ≙ noise in the data, lack in model expressivity

Systematic uncertainty: heteroskedastic loss

- log-likelihood loss:



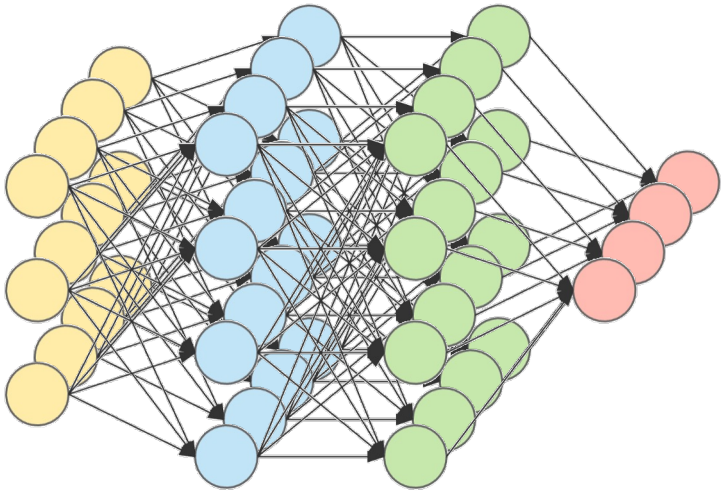
- assume Gaussian likelihood: $p(A|x,\theta) = \mathcal{N}(\overline{A}(x), \sigma^2_{\text{syst}}(x))$
- NN learns both: $\overline{A}(x)$ and $\sigma_{\text{syst}}(x)$

$$\Rightarrow \text{heteroskedastic loss: } \mathcal{L} = \sum_i \left[\frac{\left(\overline{A}(x_i) - A_{\text{true}}(x_i)\right)^2}{2\sigma^2_{\text{syst}}(x_i)} + \log\left(\sigma_{\text{syst}}(x_i)\right) \right]$$

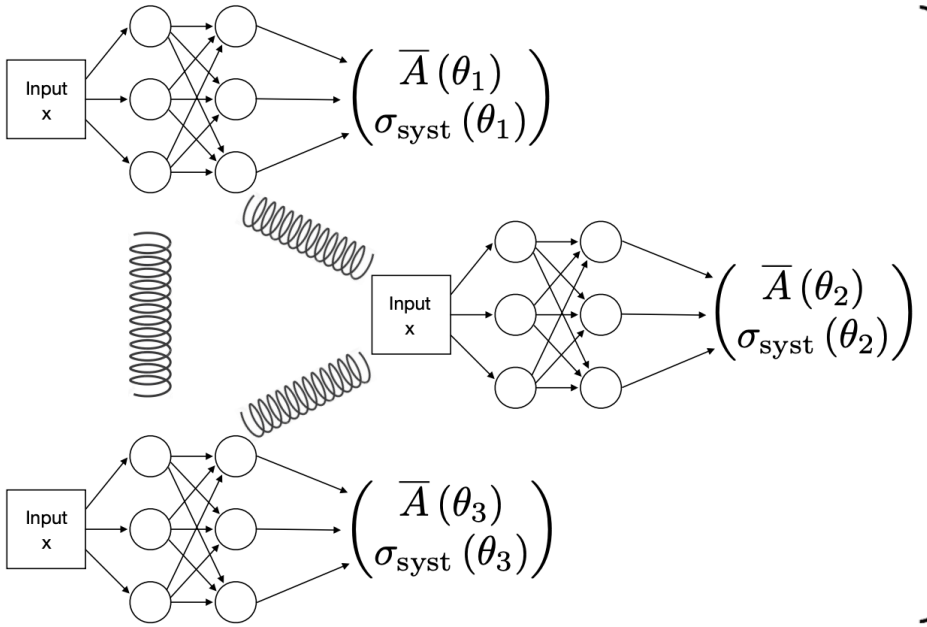
- constant $\sigma_{\text{syst}} \rightarrow$ recovers MSE loss

Statistical uncertainty: repulsive ensemble

- train ensemble of networks
- ensure convergence to correct posterior via repulsive interaction between ensemble members
- each networks leads to slightly different result
- spread of network predictions ~ statistical uncertainty
- less data → higher spread



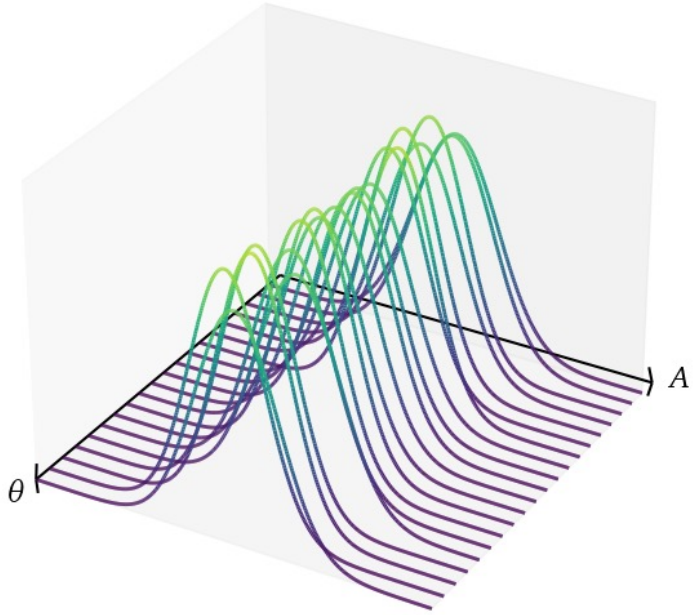
Repulsive ensemble + heteroskedastic loss



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

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

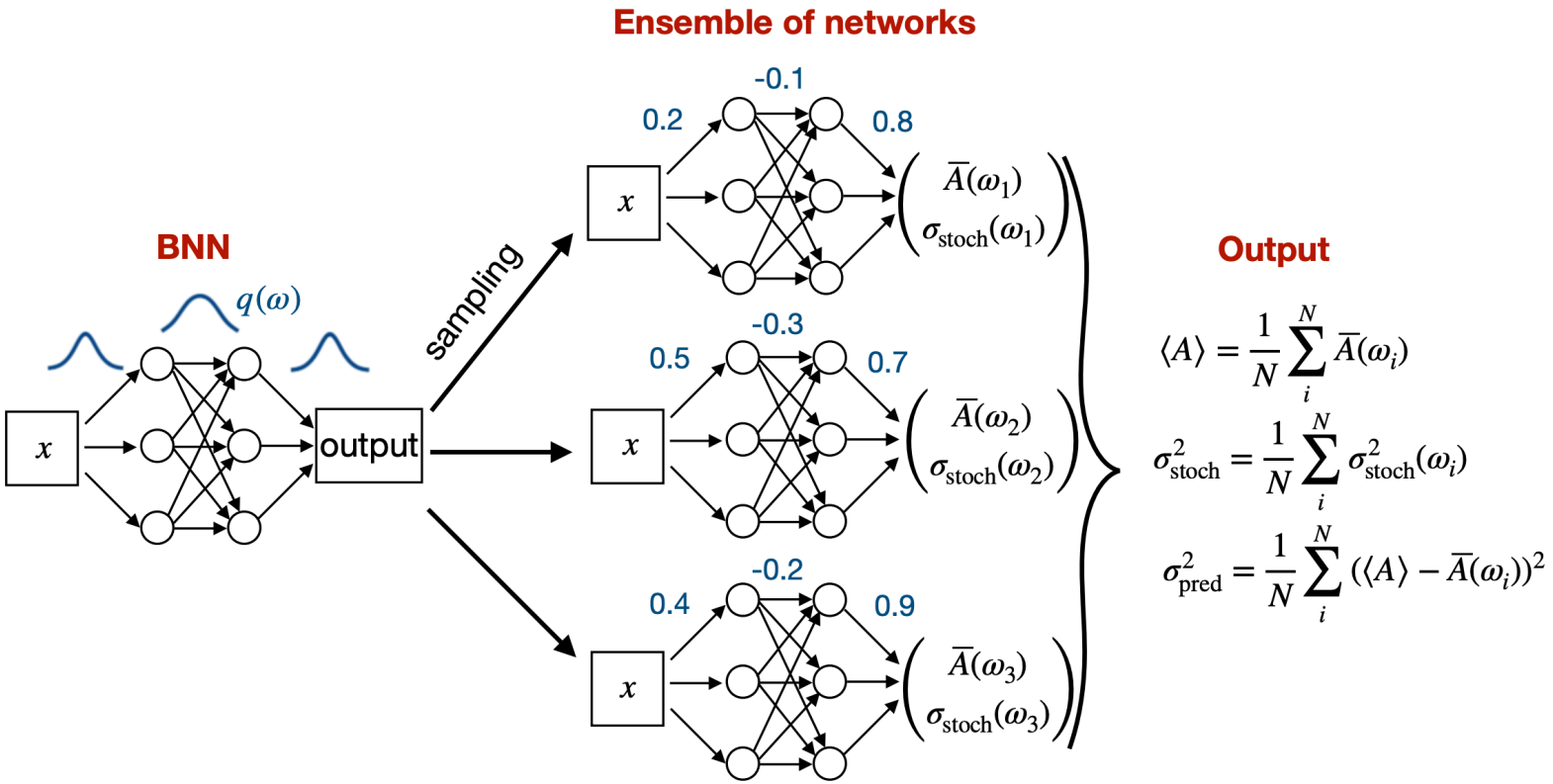
$$\sigma^2_{\text{stat}} = \frac{1}{N} \sum_i^N \left(\langle A \rangle - \overline{A}(\theta_i) \right)^2$$



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

Combined learnable modelling of systematic and statistical uncertainties!

Alternative: Bayesian NNs



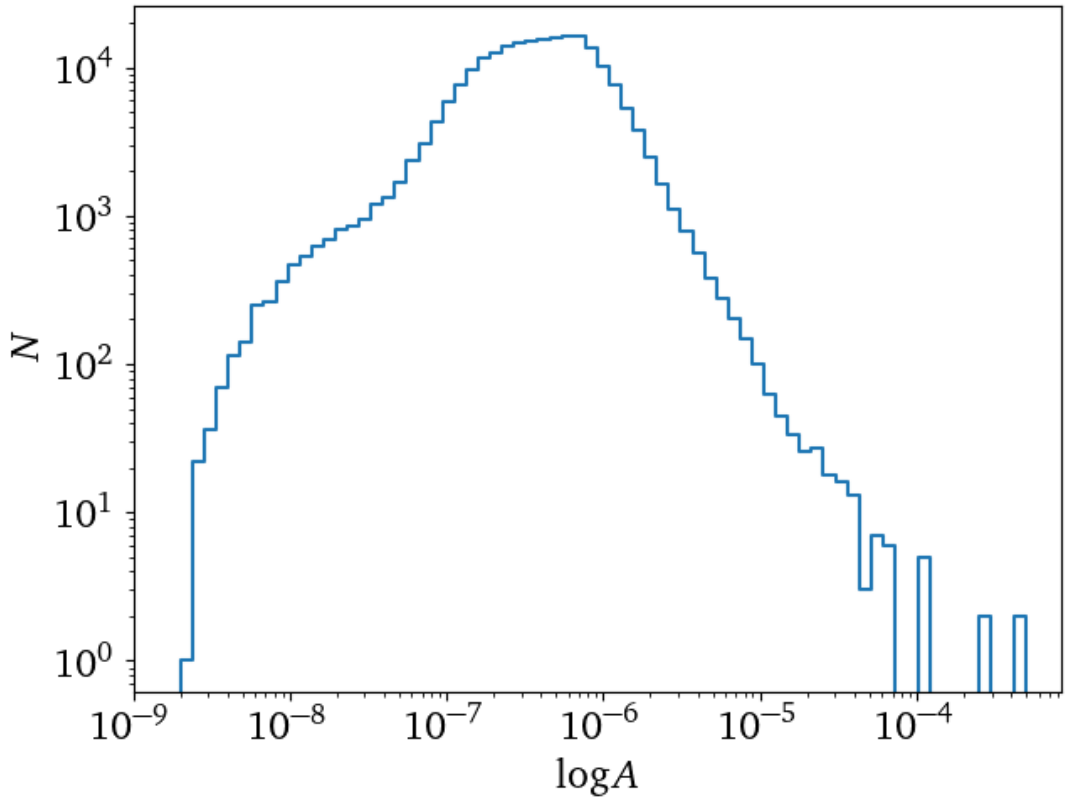
- promote NN parameters to Gaussians $q(\theta)$
- for each evaluation, sample from Gaussians
- learn means and widths

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

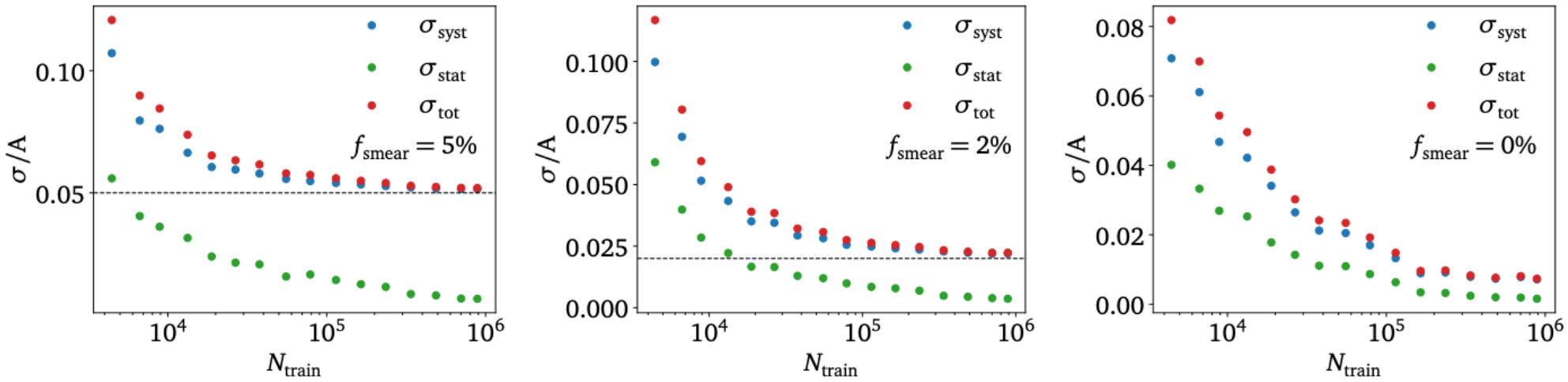
prior

$gg \to \gamma\gamma g$ dataset

- generated using Sherpa+Njet
- 1.1 million events, 70% used for training
- preprocessing:
  - learn logarithm of amplitude
  - rescale inputs and log-amplitudes to have zero mean and unit standard deviation
- amplitude symmetries
  - Lorentz-invariant
  - permutation-invariance w.r.t. identical particles



Behavior of learned uncertainties



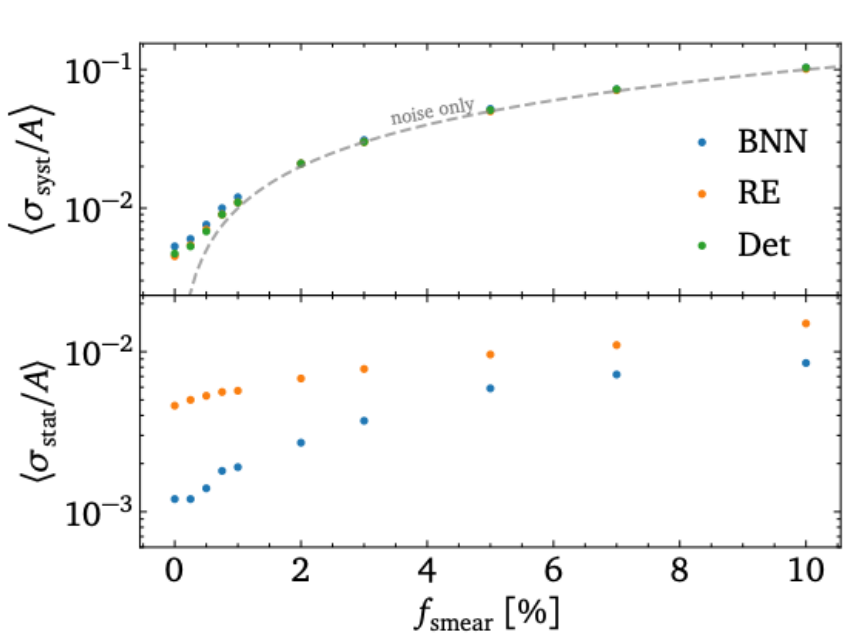
Test: apply different levels of Gaussian noise to amplitudes

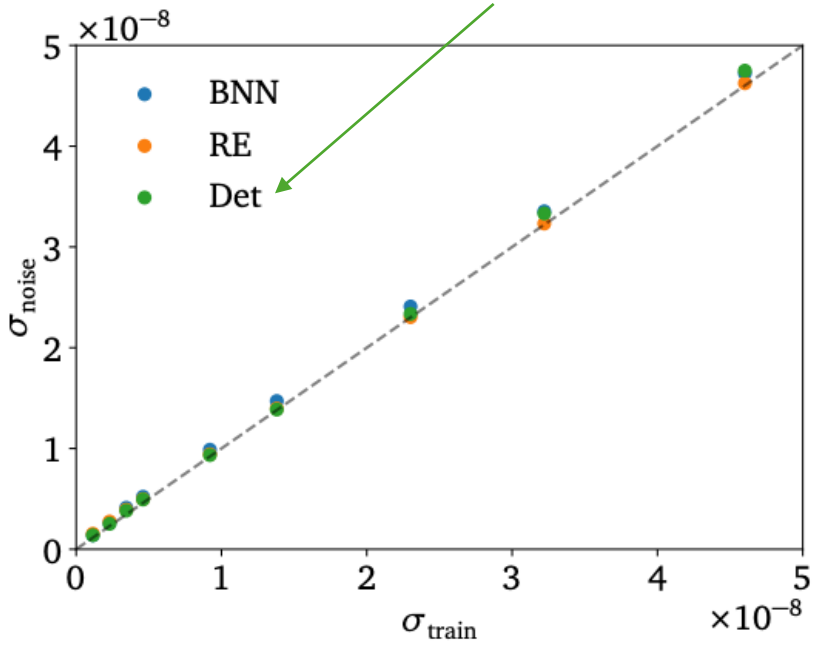
- statistical uncertainty decreases with more training data
- systematic uncertainty converges to level of applied noise

$$A_{\text{train}} \sim \mathcal{N}(A_{\text{true}}, \sigma^2_{\text{train}})$$
$$\sigma_{\text{train}} = f_{\text{smear}} A_{\text{true}}$$

Extracting the noise level

only heteroskedastic loss, no σ_{stat}





with $\sigma^2_{\text{noise}} = \sigma^2_{\text{syst}} - \sigma^2_{\text{syst},0}$ and $\sigma^2_{\text{syst},0}$ being the systematic unc. due to limited NN expressivity.

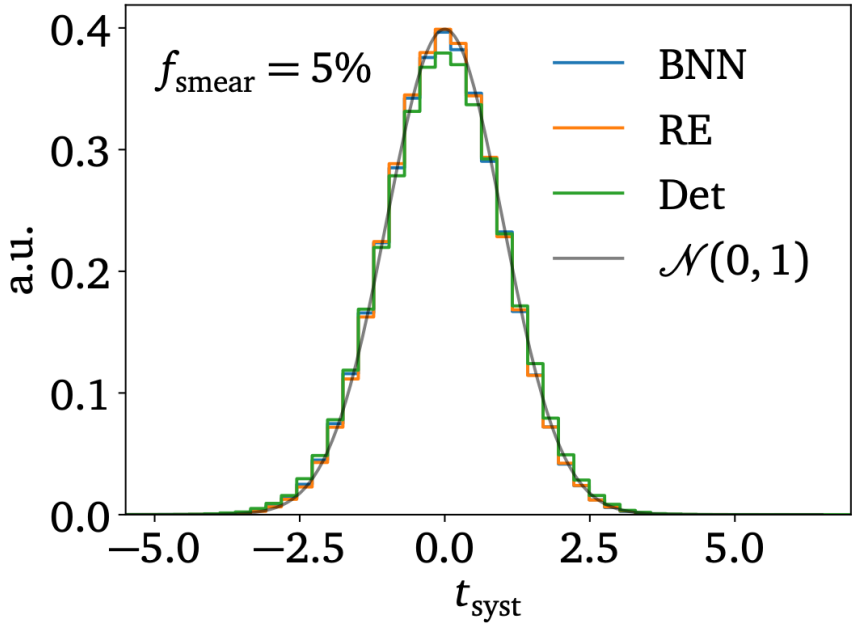
Able to reliably extract noise level!

Are these uncertainties calibrated?

- statistical uncertainties play minor role for amplitude regression
- define systematic pull:

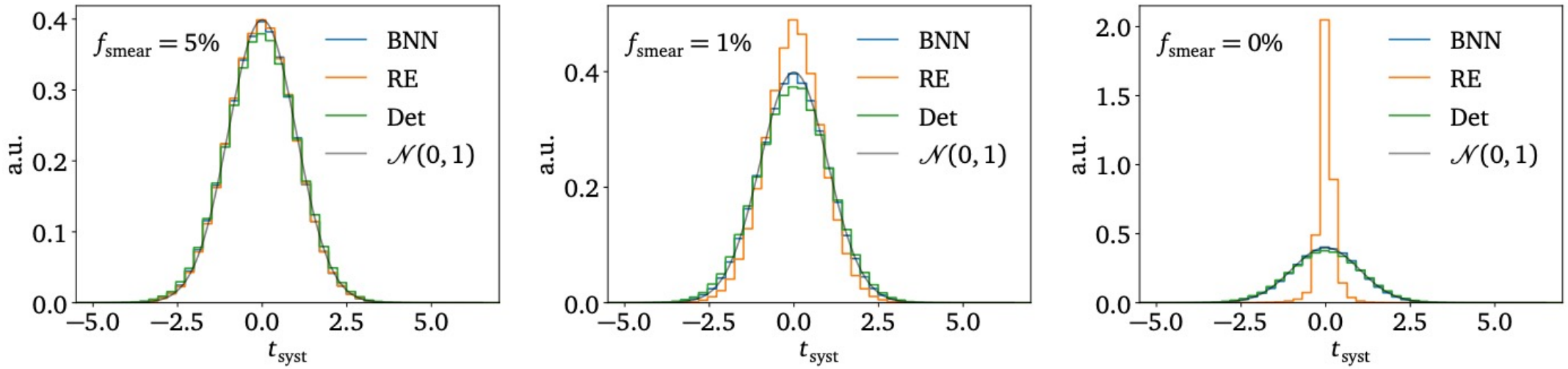
$$t_{\text{syst}} = \frac{\langle A \rangle(x) - A_{\text{train}}(x)}{\sigma_{\text{syst}}(x)}$$

- if calibrated, t_{syst} distribution should follow $\mathcal{N}(0, 1)$



Almost perfectly calibration → reliable uncertainty estimate

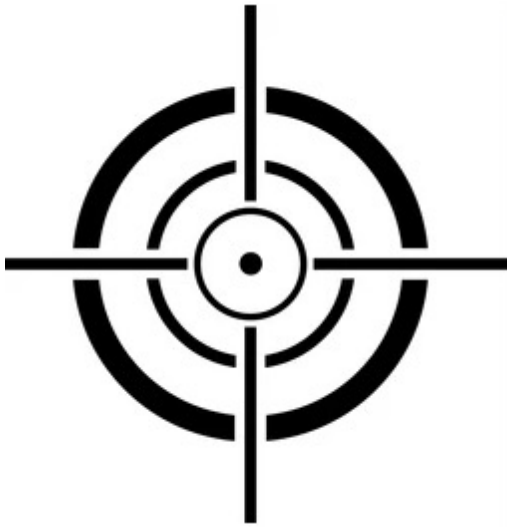
Dependence on smearing



- BNN and deterministic models
  → well calibrated for different smearing levels

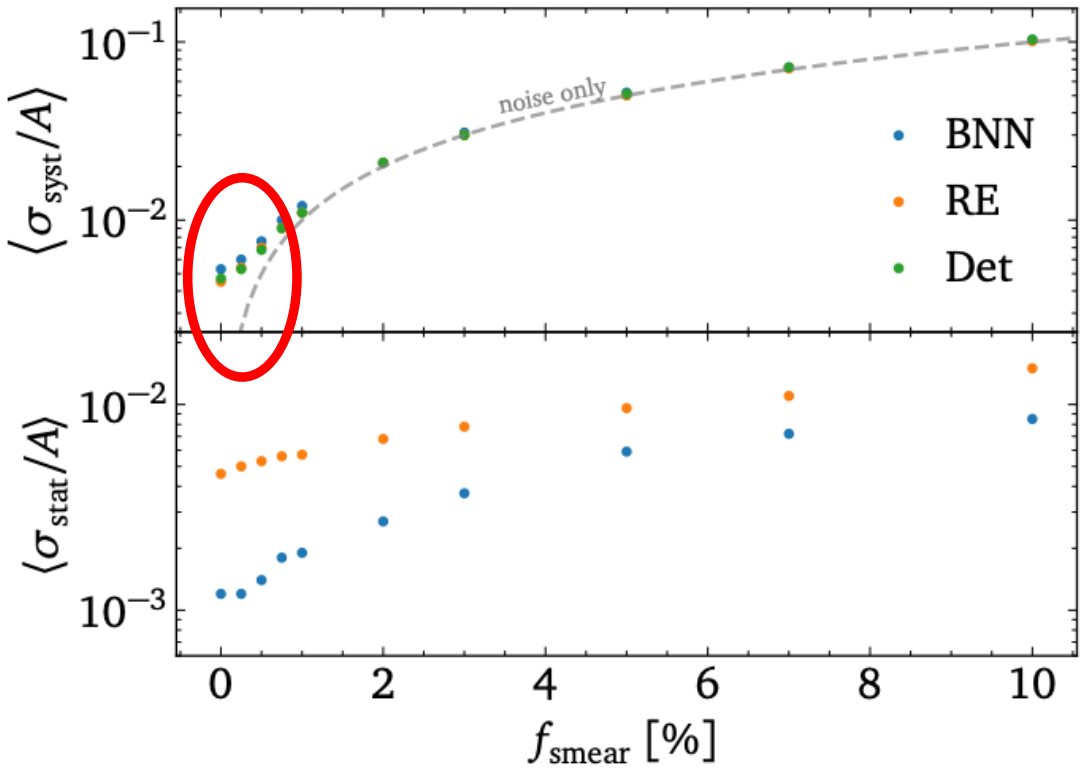
- repulsive ensemble overestimates uncertainty for low smearing
  → see below

Pushing for precision



activations, layers, and domain knowledge

Enhancing NN expressivity

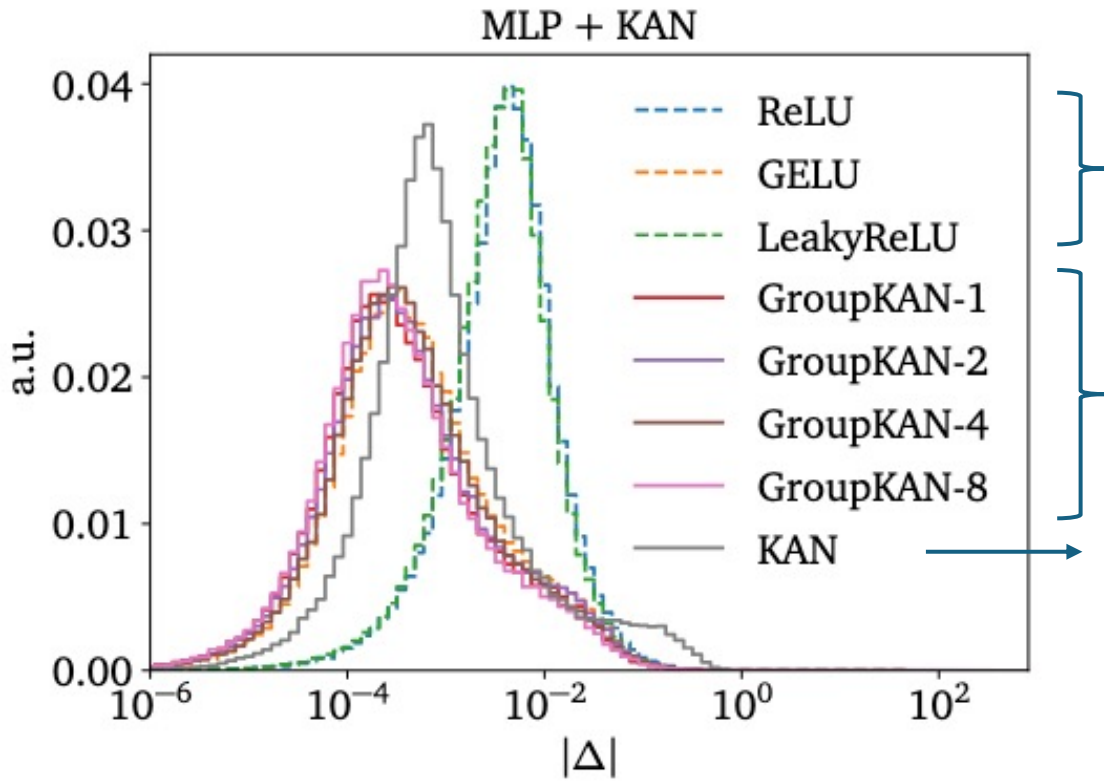


Can we improve accuracy for low smearing?

Improve NN expressivity:

- different activations, NN structures
- more layers,
- exploit symmetries

Activation functions and NN structure



static activation functions

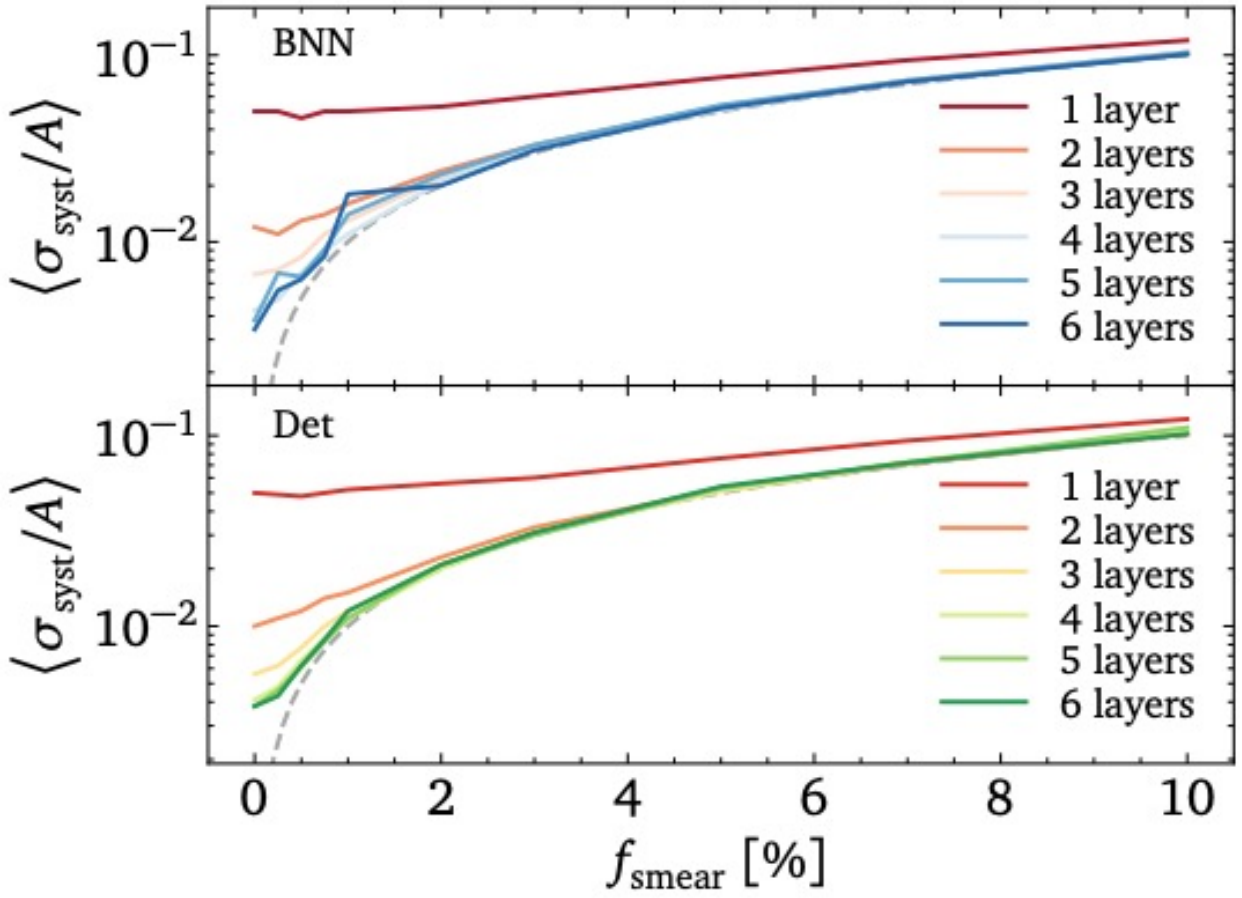
learned activation functions

Kolmogorov-Arnold Network

relative deviation: $\Delta = \frac{A_{\mathrm{NN}} - A_{\mathrm{true}}}{A_{\mathrm{true}}}$

well-chosen activation function can significantly improve performance

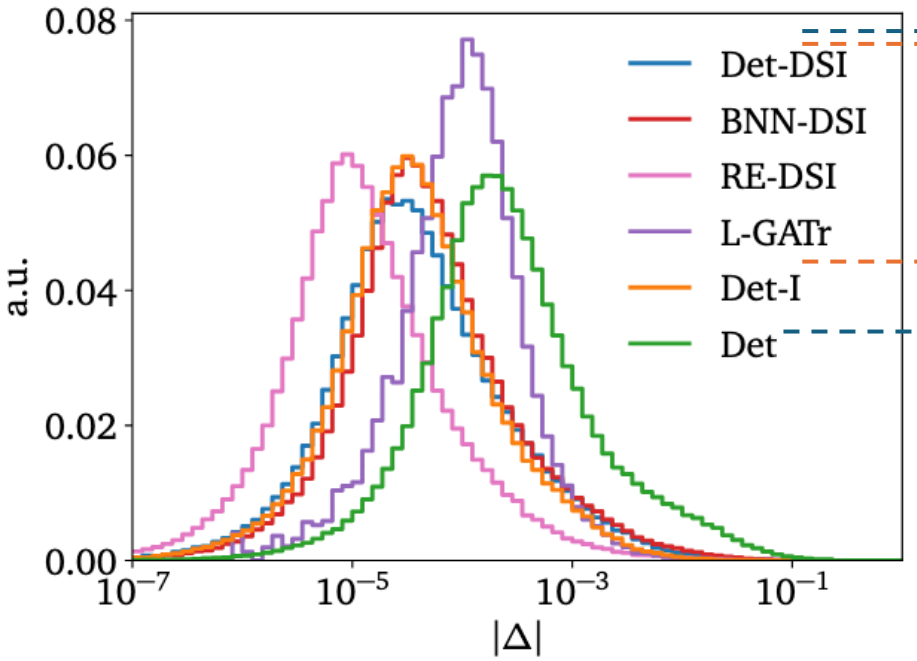
Adding more layers



adding more layers

convergence towards noise level

Encoding our physics knowledge

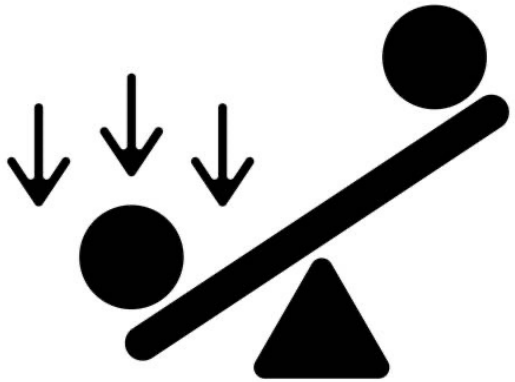


permutation invariance

Lorentz invariance

relative deviation: $\Delta = \frac{A_{\mathrm{NN}} - A_{\mathrm{true}}}{A_{\mathrm{true}}}$

Large gain in NN accuracy! Also found uncertainties still to be well calibrated.



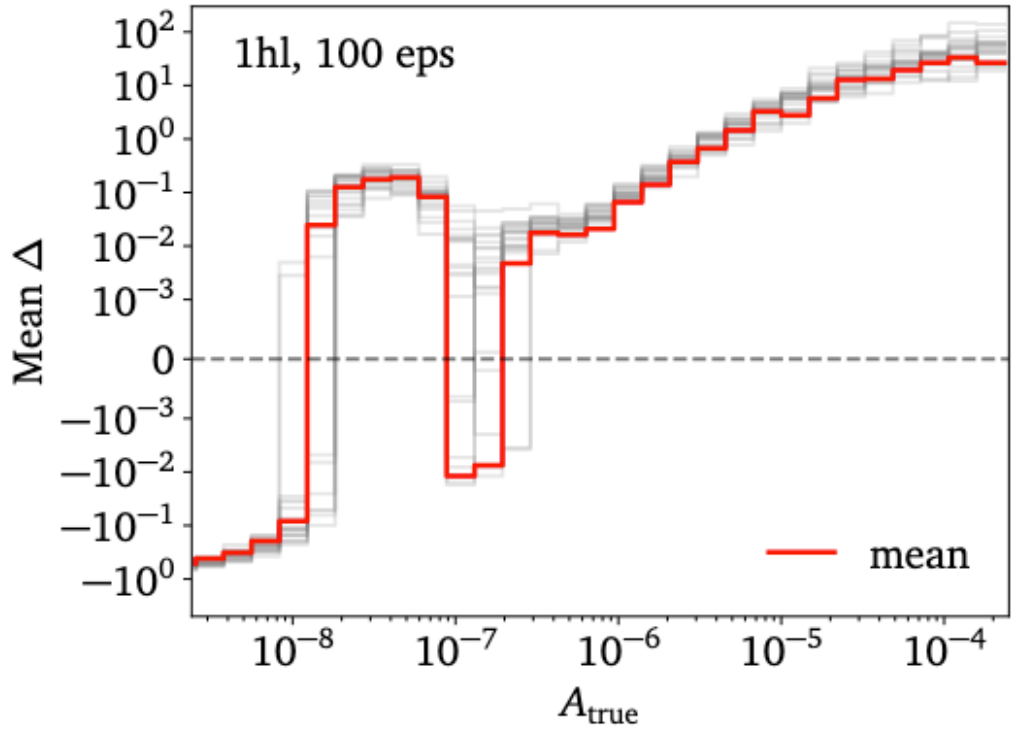
Ensembling and biases

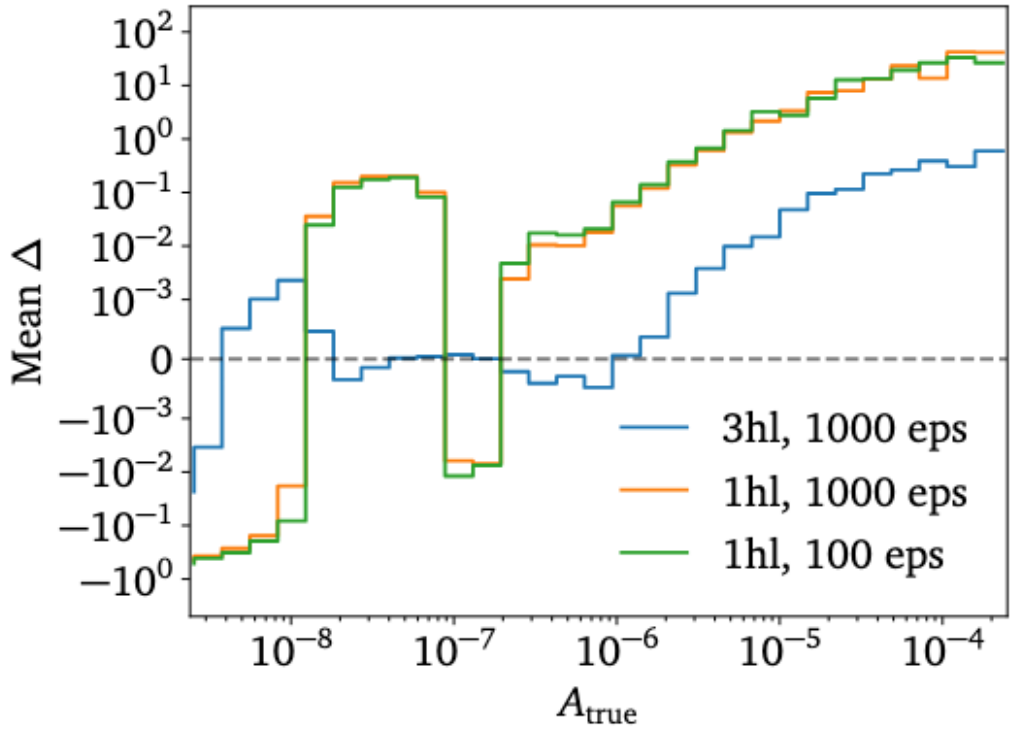
The wisdom (and biases) of the crowd.

[HB, Elmer, Plehn, Winterhalder, to appear]

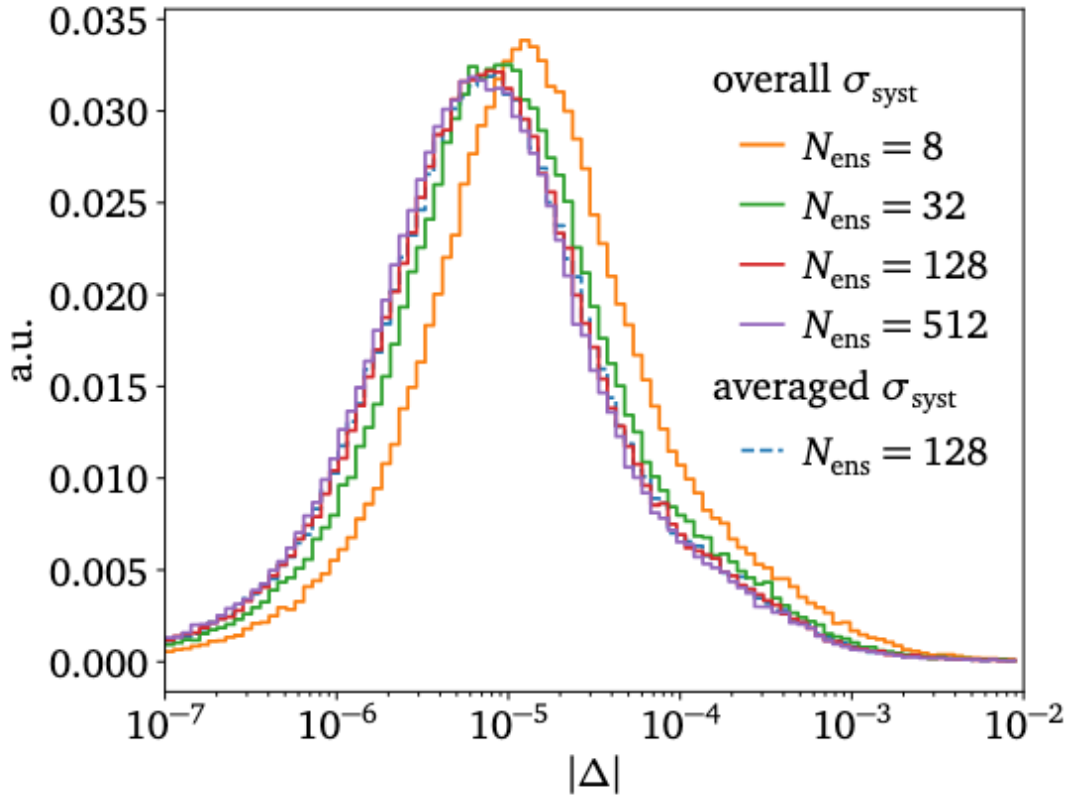
Ensembling and biases

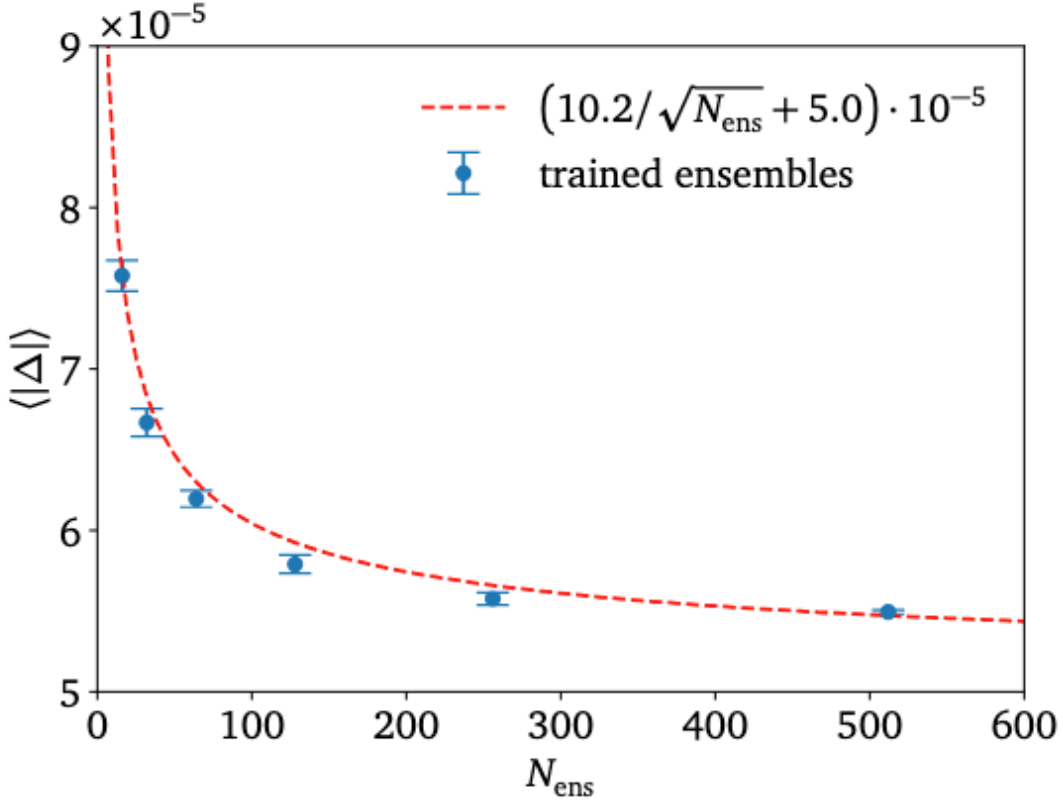
- ensembling averages out noise → widely used
- but, their can be systematic biases





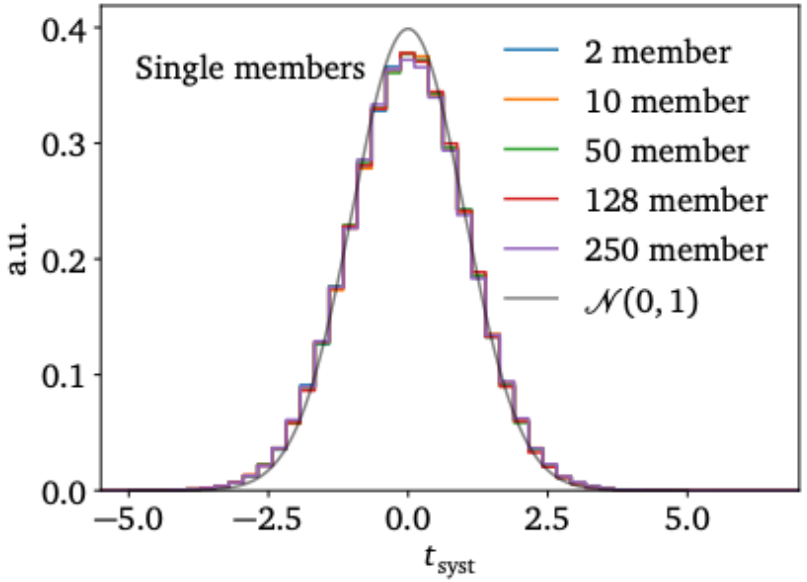
Extracting the biases

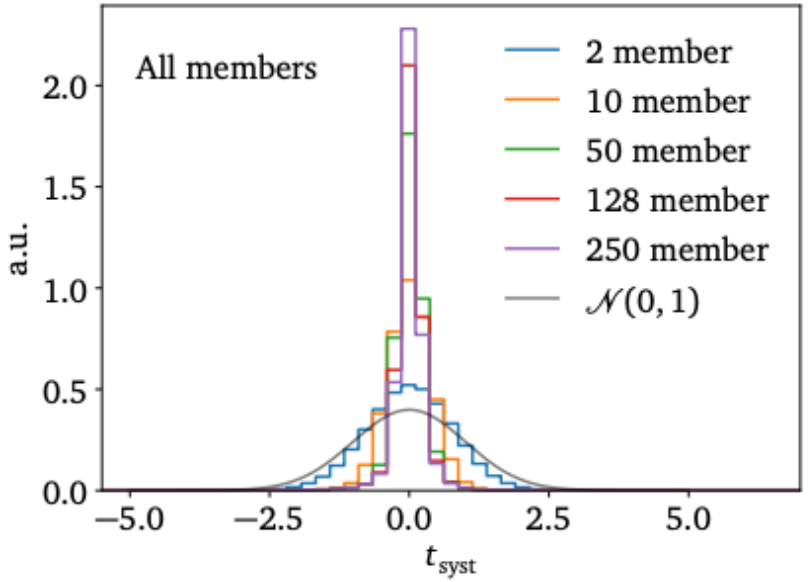




ensembling increases precision, but gains level off for $N_{\text{ens}} \gtrsim 100$

Miscalibration of ensemble systematic





$$A_{\text{NN}}(x) = \frac{1}{N_{\text{ens}}} \sum_{i=1}^{N_{\text{ens}}} A_{\text{NN},i}(x)$$

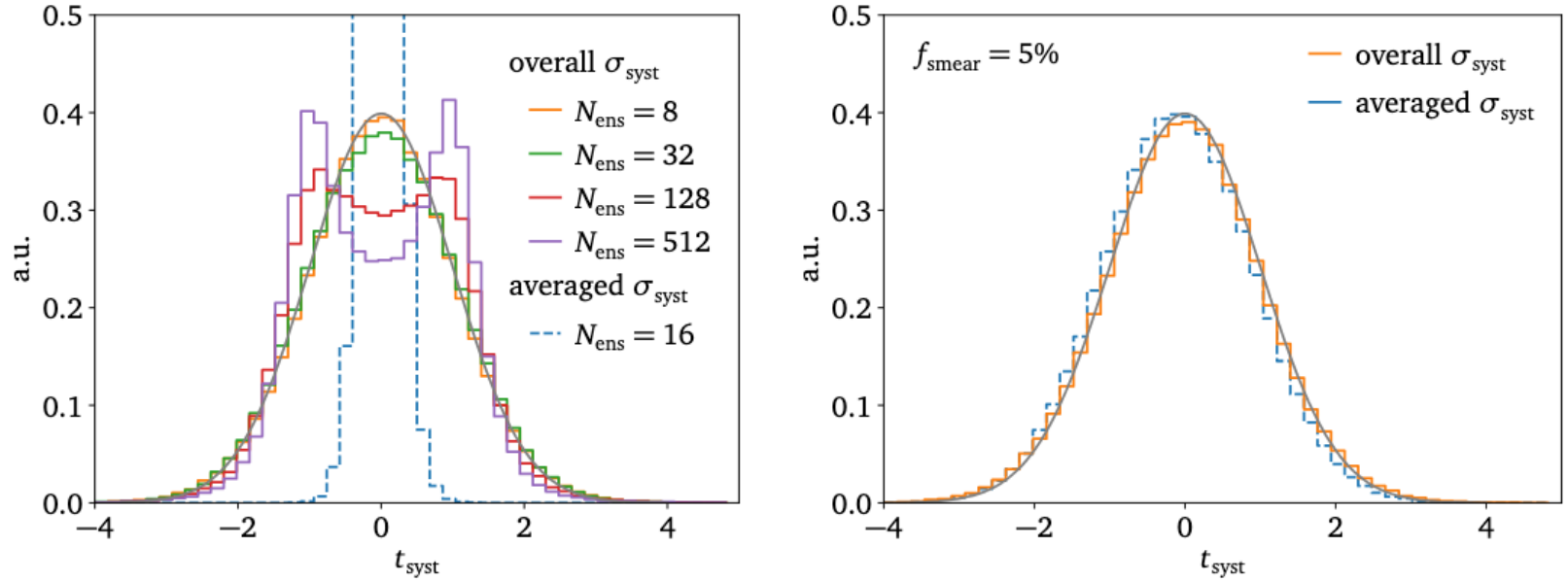
$$\sigma^2(x) = \frac{1}{N_{\text{ens}}} \sum_{i=1}^{N_{\text{ens}}} \sigma_i^2(x)\,.$$

- averaging systematic unc. of ensemble members ~ average uncertainty of each member
- but ensemble more precise

Fixing ensemble systematic unc.

$$A_{\mathrm{NN}}(x) = \frac{1}{N_{\mathrm{ens}}} \sum_{i=1}^{N_{\mathrm{ens}}} A_{\mathrm{NN},i}(x) \qquad \mathcal{L}_\sigma = \frac{1}{B} \sum_{b=1}^{B} \left[\frac{|A_{\mathrm{train}}(x_b) - A_{\mathrm{NN}}(x_b)|^2}{2\sigma^2(x_b)} + \log \sigma(x_b) \right]$$

solution: train separate network with loss $\mathcal{L}_\sigma$ to predict systematic uncertainty of ensemble



lower noise levels → non-Gaussian biases become visible

Localized smearing

Is the systematic uncertainty able to pick-up local noise?

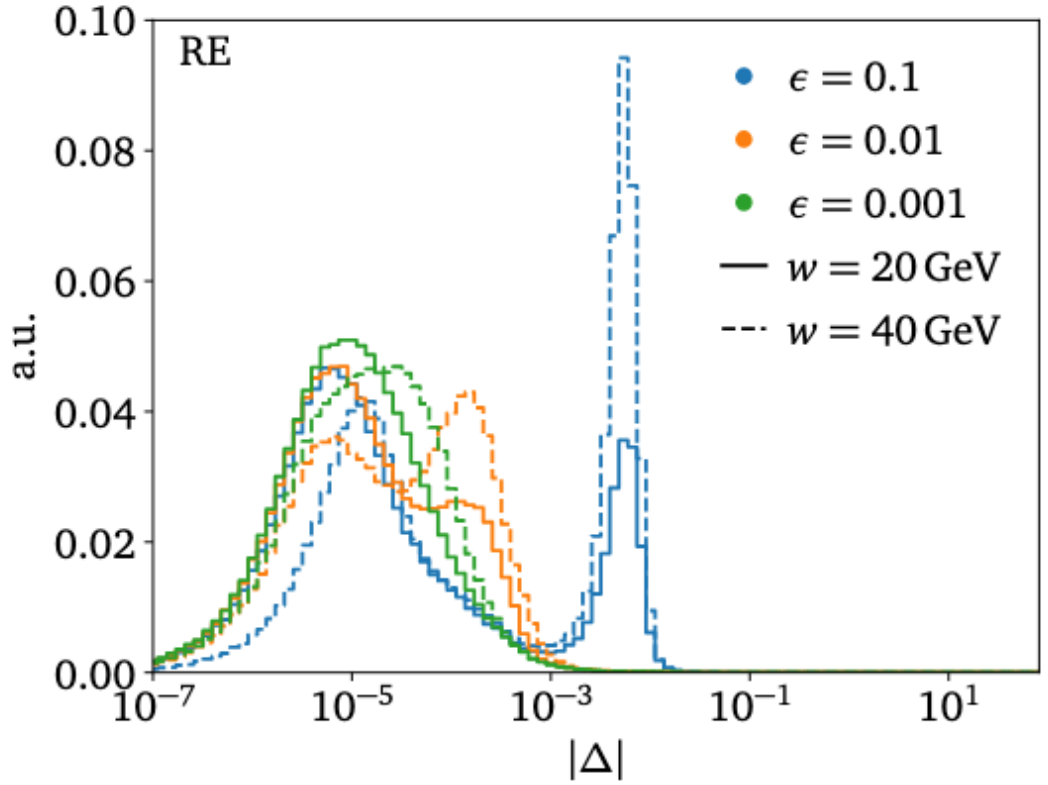
[HB et al., to appear]

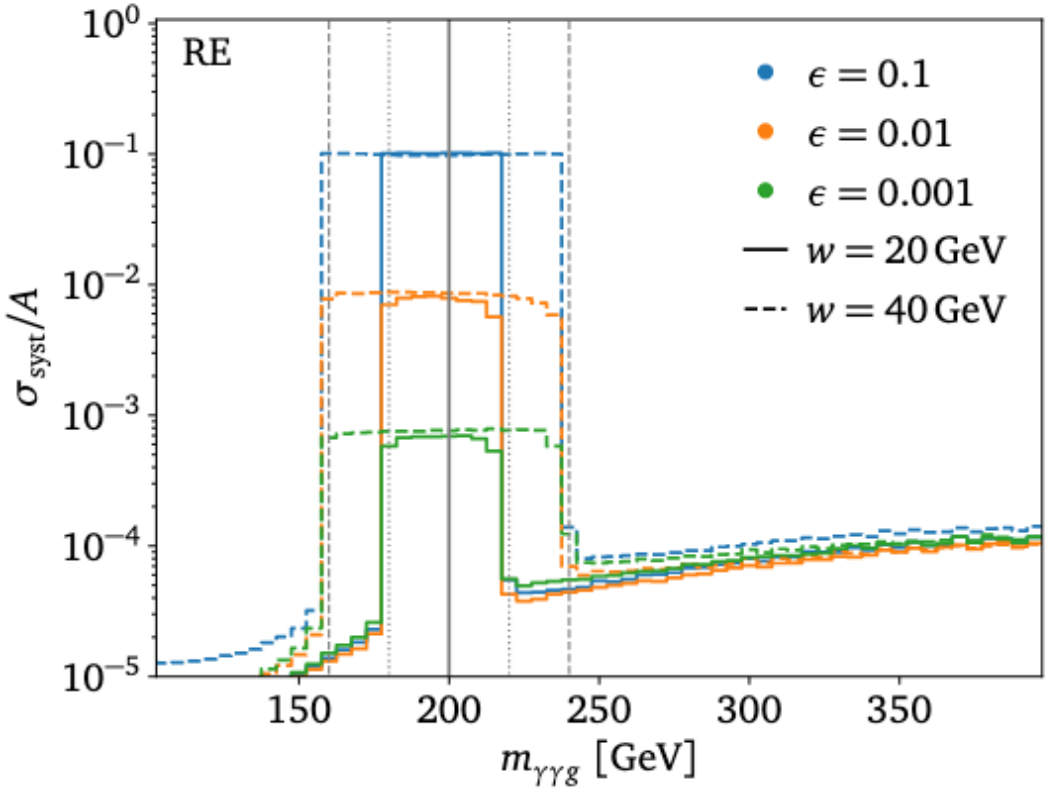
Box smearing

- analytic amplitude evaluation might loss precision close to thresholds, kinematic edges, …
- can the learned systematic uncertainties pick this up?
- test this by inducing local noise to our amplitudes
- simple start: box smearing with $m_{\text{thresh}} = 200\ \text{GeV}$

$$A_{\text{train}}(x) = \begin{cases} \mathcal{N}\left(A_{\text{true}}(x), \epsilon A_{\text{true}}(x)\right) & \text{if } |m_{\gamma\gamma g}(x) - m_{\text{thresh}}| < w \\ A_{\text{true}} & \text{if } |m_{\gamma\gamma g}(x) - m_{\text{thresh}}| \geq w \end{cases}$$

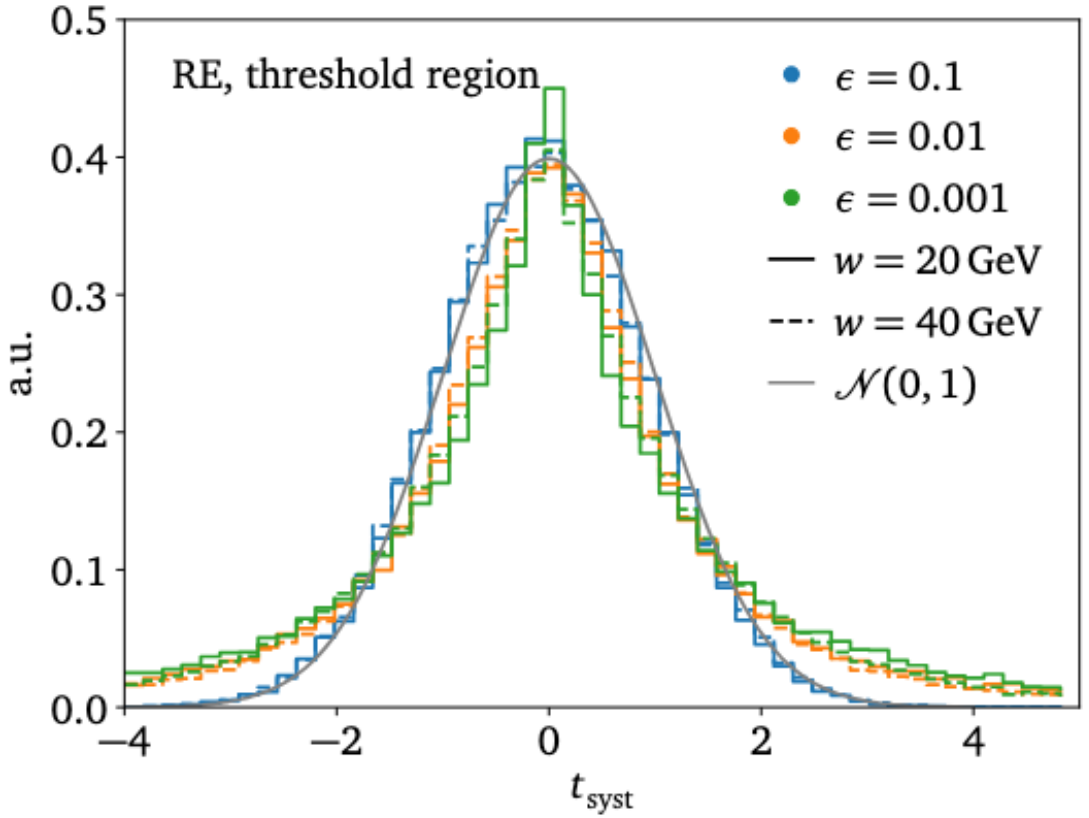
Box smearing — results

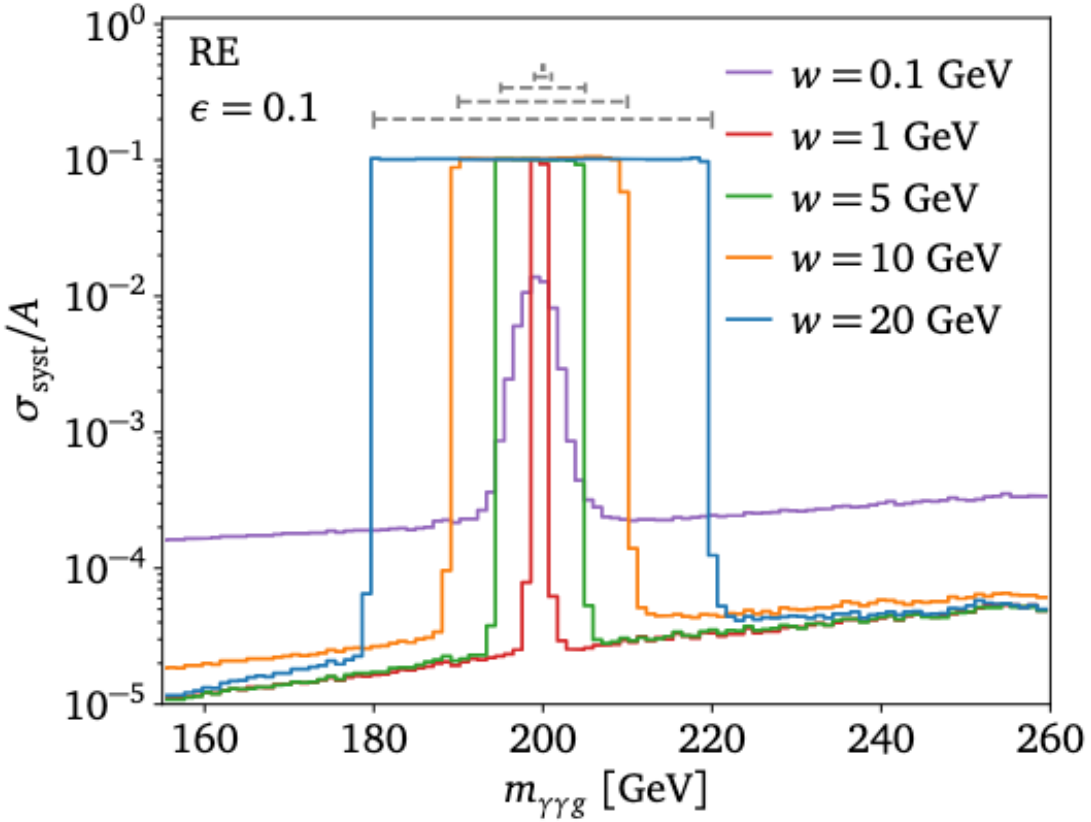




σ_{syst} closely follows expected behavior

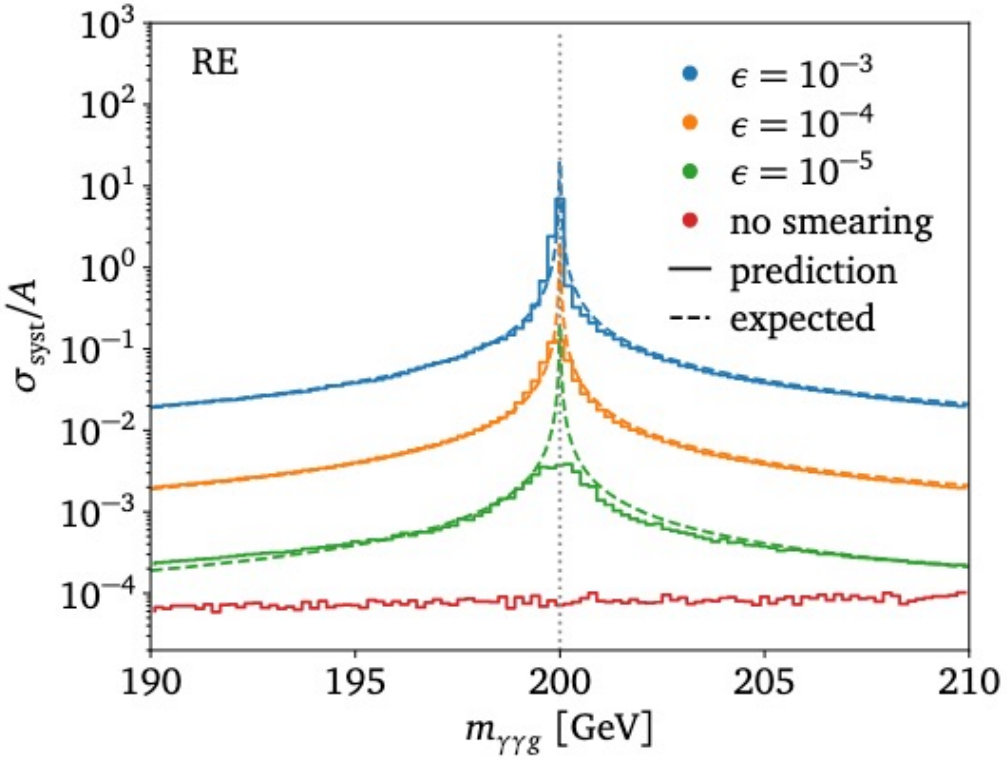
Box smearing — calibration

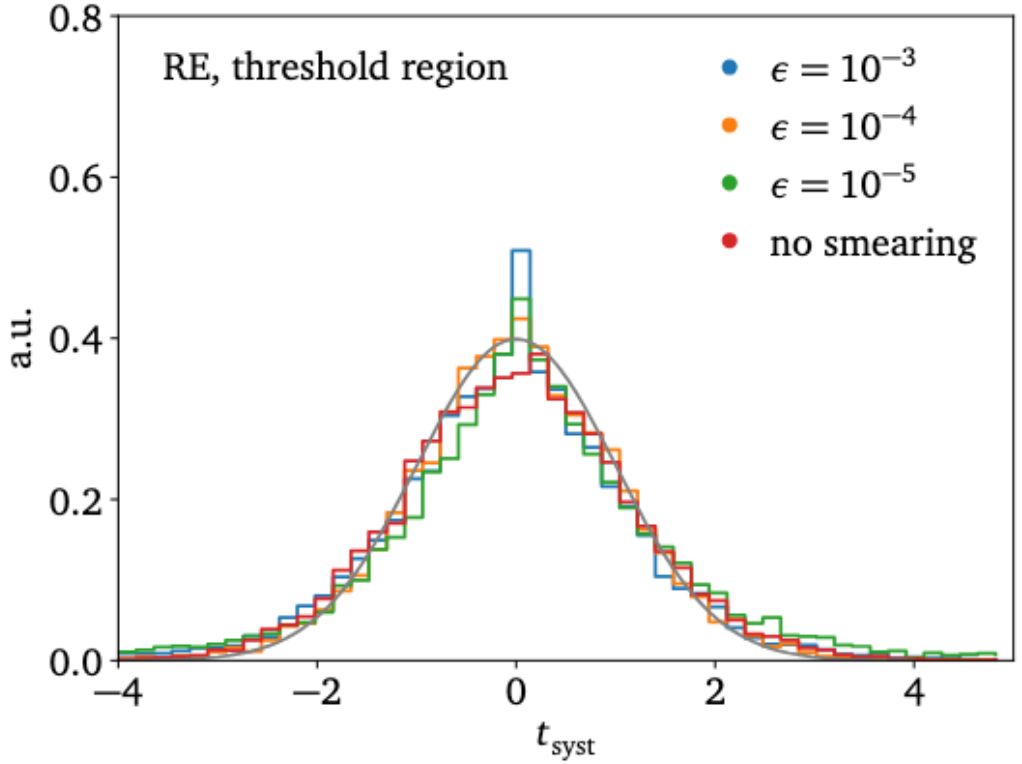




also good calibration

Peaked smearing





$$A_{\text{train}}(x) = \mathcal{N}\left(A_{\text{true}}(x), \frac{\epsilon\, m_{\text{thresh}}}{|m_{\gamma\gamma g}(x) - m_{\text{thresh}}|} A_{\text{true}}(x)\right)$$

well captured by systematic uncertainties

Conclusions

Conclusions

- amplitude surrogates → speed up MC generation
- uncertainty-aware NNs allow for controlled modelling of
  - systematic uncertainties: data-inherent noise + model expressivity
  - statistical uncertainties: lack of data
- methods: heteroskedastic loss, repulsive ensemble, Bayesian NNs
- learned uncertainties reflect actual deviations from truth well
- encoding physics knowledge increase accuracy with still well-calibrated unc.

same techniques also applicable to all kind of other problems

Appendix

Kolmogorov-Arnold theorem

Any $f: [0,1]^n \to \mathbb{R}$ can be written as a finite decomposition of univariate functions and addition.

$$f(x) = f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$$

where $\phi_{q,p} : [0,1] \to \mathbb{R}$ and $\Phi_q : \mathbb{R} \to \mathbb{R}$.

Kolmogorov-Arnold Networks (KANs)

- KAN layer: $$x_{l+1} = \underbrace{\begin{pmatrix} \phi_{l,1,1}(\cdot) & \phi_{l,1,2}(\cdot) & \cdots & \phi_{l,1,n_l}(\cdot) \\ \phi_{l,2,1}(\cdot) & \phi_{l,2,2}(\cdot) & \cdots & \phi_{l,2,n_l}(\cdot) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{l,n_{l+1},1}(\cdot) & \phi_{l,n_{l+1},2}(\cdot) & \cdots & \phi_{l,n_{l+1},n_l}(\cdot) \end{pmatrix}}_{\equiv \Phi_l} x_l \,,$$

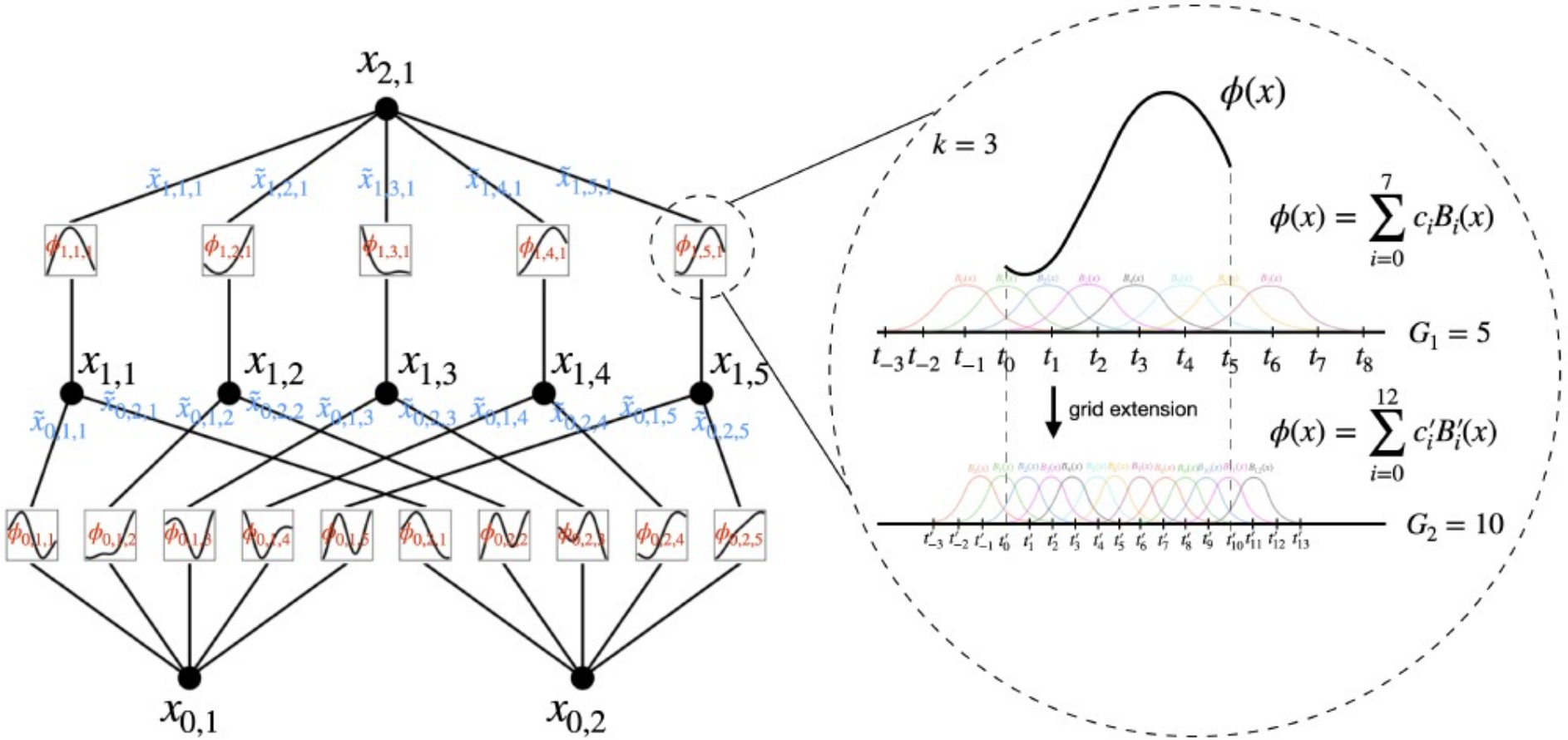
- KAN network: $$\mathrm{KAN}(x) = (\Phi_{L-1} \circ \Phi_{L-2} \circ \ldots \circ \Phi_1 \circ \Phi_0)x$$

- GroupKAN layer: $$\mathrm{activation}(x) \to \mathrm{GroupKANlayer}_l(x) = \begin{pmatrix} \phi_{l,g_l(1)}(x_1) \\ \phi_{l,g_l(2)}(x_2) \\ \vdots \\ \phi_{l,g_l(n_l)}(x_{n_l}) \end{pmatrix}$$

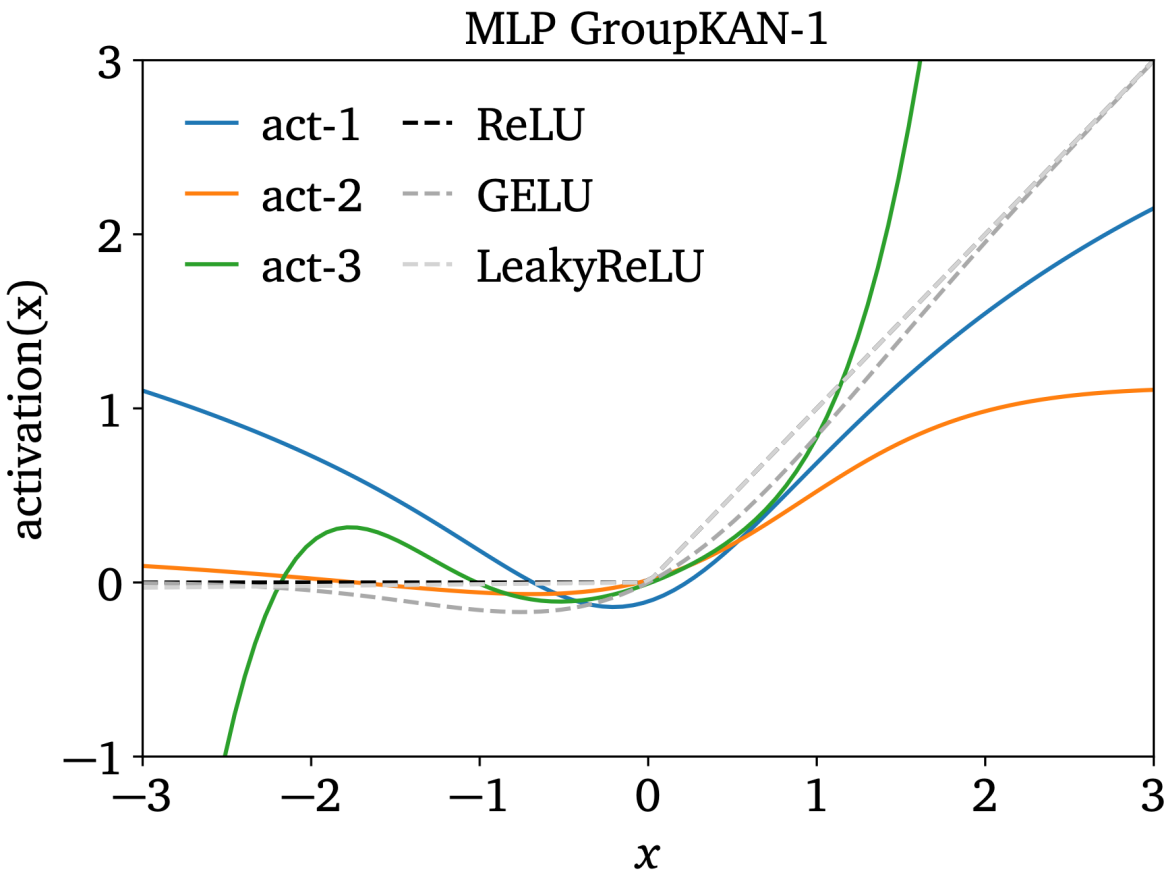
- GroupKAN network: $$\mathrm{GroupKAN}(x) = (W_{L-1} \circ \mathrm{GroupKANlayer}_{L-2} \circ W_{L-2} \circ \ldots \circ \mathrm{GroupKANlayer}_0 \circ W_0)x$$

- Normal MLP network $$\mathrm{MLP}(x) = (W_{L-1} \circ \mathrm{activation} \circ W_{L-2} \circ \ldots \circ \mathrm{activation} \circ W_0)x$$

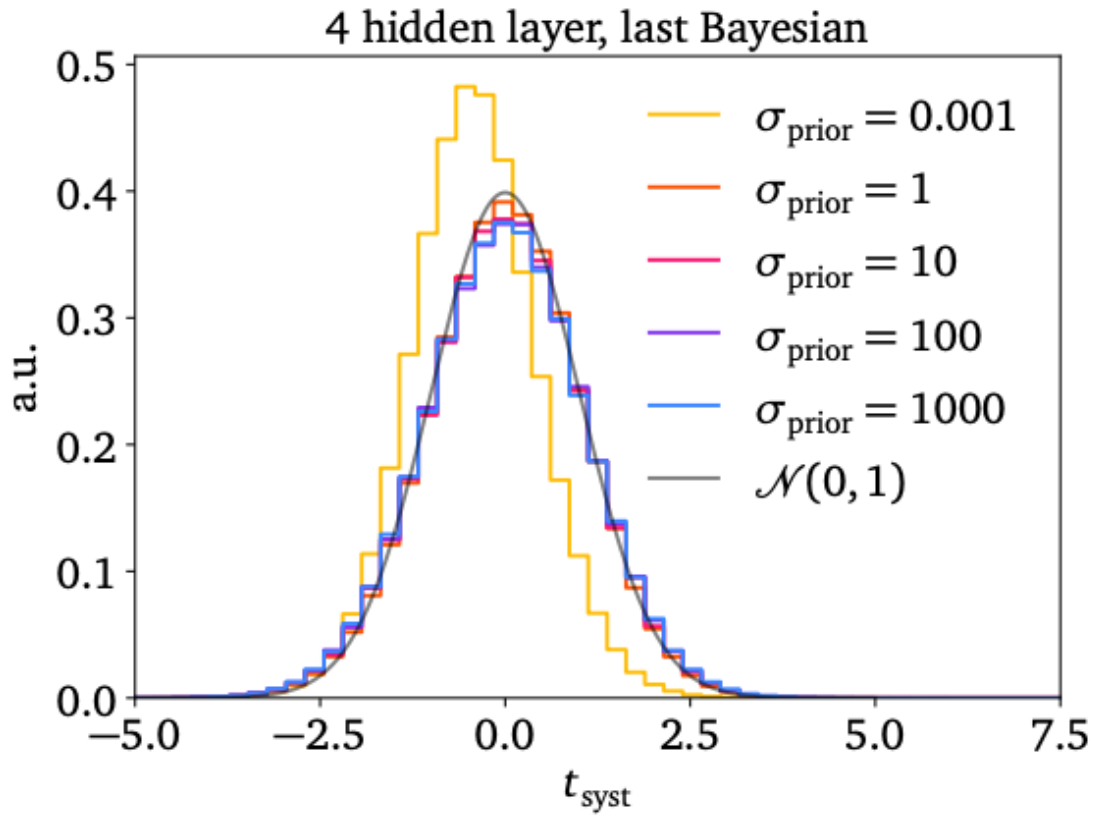
Kolmogorov-Arnold Networks (KANs)

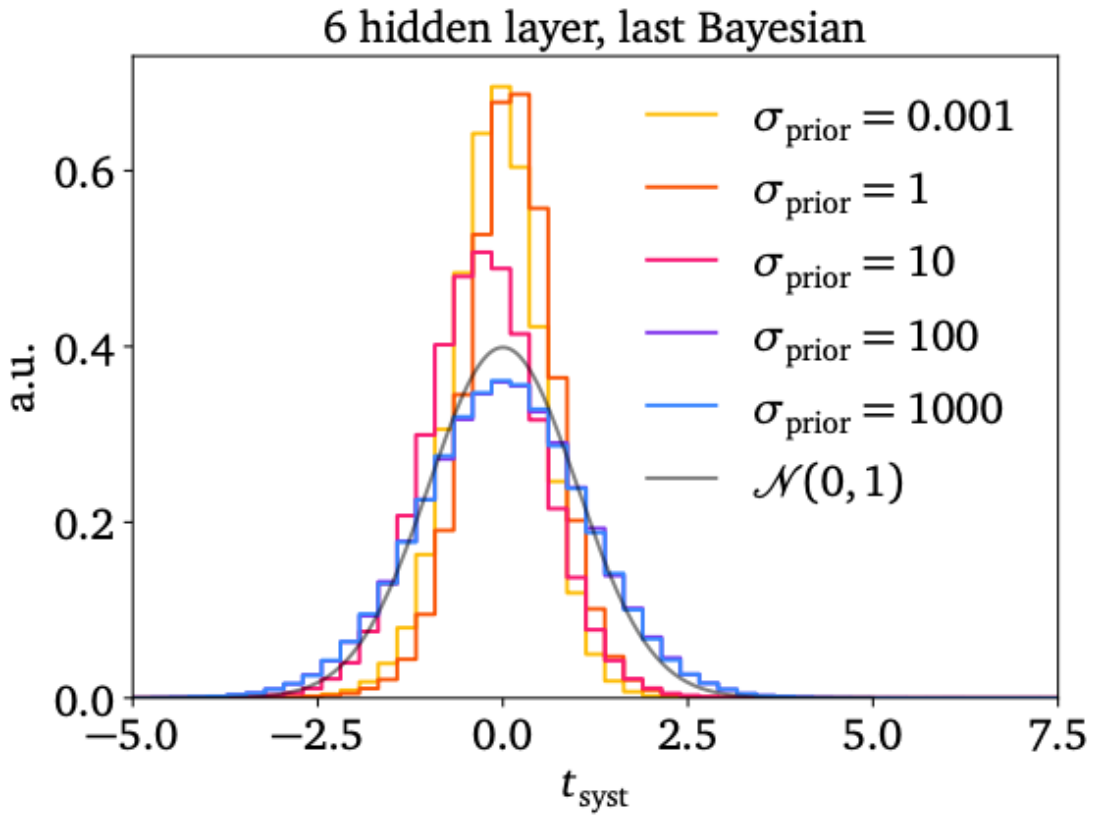


Learned activation functions



BNN prior dependence





Uncertainty overview