Know What You Don’t Flow

Learned Generative Uncertainties

Lorenz Vogel

14 September 2026

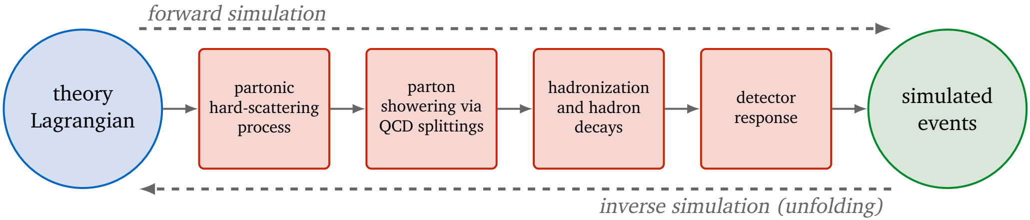
Institute for Theoretical Physics
Heidelberg University, Germany

ML4Jets Workshop 2026, Austrian Academy of Sciences

Know What You Don’t Flow, arXiv:2608.21509 [hep-ph]
A. Butter, S. Diefenbacher, T. Plehn, and L. Vogel

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Simulation chain for LHC physics



generative networks accelerate simulation chain and encode phase space density $p_\theta(x)$

but what is the **uncertainty** on the density a generative network encodes?

Two uncertainty-aware normalizing flows

Bayesian NF (BNF)

$$\mathcal{L}_{\text{BNF}} = D_{\text{KL}}[q_\phi(\theta), p(\theta)] - \big\langle \log p_\theta(\mathcal{D}_{\text{train}}) \big\rangle_{q_\phi}$$

$$\theta \sim q_\phi(\theta) \quad \Longrightarrow \quad \sigma^2_{\text{stat}}(x) = \text{Var}_{\theta|\mathcal{D}_{\text{train}}}\big[p_\theta(x)\big]$$

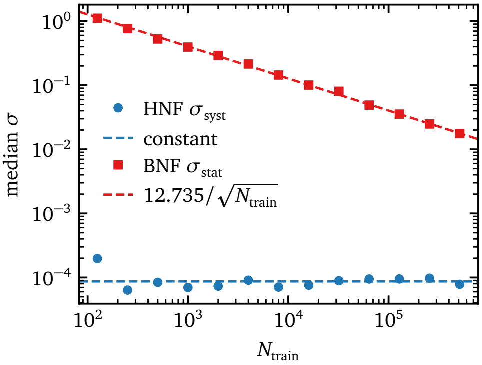
- learns variational **posterior** $q_\phi(\theta)$ over network weights instead of fixed values
- sampling from learned weight distributions provides uncertainty
- expected to capture **statistical uncertainty** (defined as vanishing in limit of infinite training data)

Heteroscedastic NF (HNF)

$$\mathcal{L}_{\text{HNF}} = \left\langle \frac{[p_\theta(x) - p_{\text{truth}}(x)]^2}{2\sigma^2_{\text{syst}}(x)} + \log \sigma_{\text{syst}}(x) \right\rangle_{\mathcal{D}_{\text{train}}}$$

- **NF** predicting density $p_\theta(x)$ and **MLP** predicting uncertainty $\sigma_{\text{syst}}(x)$, trained jointly
- expected to capture **systematic uncertainty** (defined as not vanishing in limit of infinite training data)
- needs **explicit target density** $p_{\text{truth}}(x)$

Toy model — statistical uncertainty

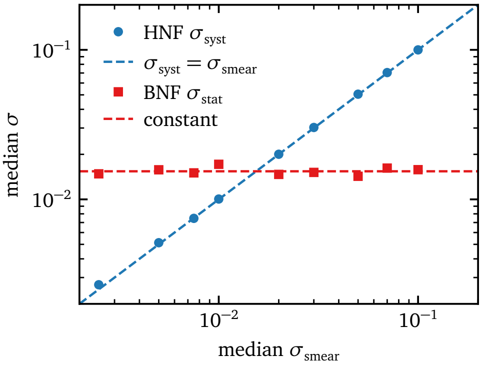


- 2D wedge ramp $p(x_1, x_2) = 2x_1$, **density known exactly**
- vary training dataset size N_{train}
- BNF uncertainty $\sigma_{\text{stat}}(x)$ **vanishes with more data**

$$\sigma_{\text{stat}}(x) \propto \frac{1}{\sqrt{N_{\text{train}}}}$$

- HNF uncertainty $\sigma_{\text{syst}}(x)$ stays **flat** in training dataset size N_{train}

Toy model — systematics from noisy data



- **controlled systematic**: inject relative Gaussian noise into target density, $f_{\text{smear}} = 0.25 \dots 10\,\%$

$$\sigma_{\text{smear}}(x) = f_{\text{smear}} \times p_{\text{truth}}(x)$$

- HNF uncertainty $\sigma_{\text{syst}}(x)$ **tracks injected target noise perfectly**
- BNF uncertainty $\sigma_{\text{stat}}(x)$ **does not capture noise**

Is the learned uncertainty calibrated?

Pull distribution

$$t_{\text{truth}}(x) = \frac{p_\theta(x) - p_{\text{truth}}(x)}{\sigma_{\text{syst}}(x)}$$

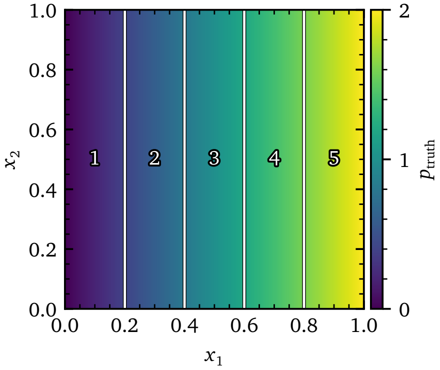
- calibrated **Gaussian** uncertainty: pull distribution follows **standard Gaussian** $\mathcal{N}(0,1)$
- width > 1: $\sigma_{\text{syst}}(x)$ **underestimated**
  width < 1: $\sigma_{\text{syst}}(x)$ **overestimated**

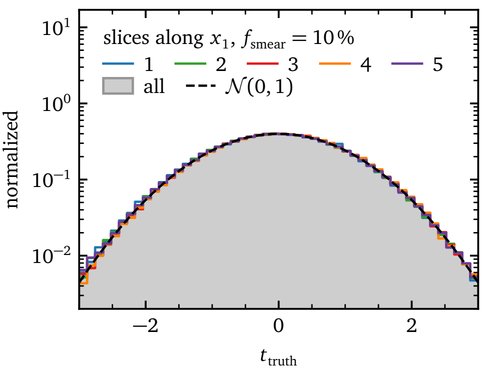
Empirical coverage

$$c_\gamma = \left\langle \mathbb{1}\left(\mathcal{P}_{\text{Gauss}}(p_{\text{truth}} \,|\, p_\theta, \sigma_{\text{syst}}) > 1-\gamma\right)\right\rangle_{\mathcal{D}_{\text{test}}}$$

- fraction of test points for which $p_{\text{truth}}(x)$ falls within network's γ-confidence region around $p_\theta(x)$
- calibrated: empirical coverage c_γ equals nominal coverage γ (**diagonal**)

Toy model — local and global uncertainty calibration

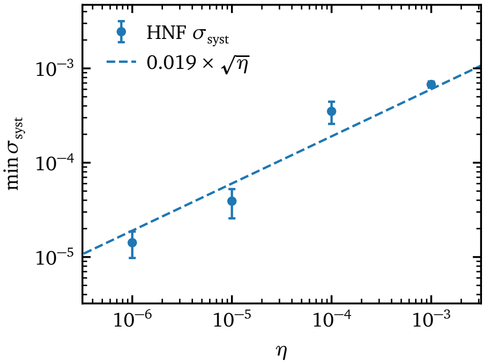




pull $t_{\mathrm{truth}}(x)$ for $f_{\mathrm{smear}} = 10\,\%$ in five slices along x_1,
from low to high density, and over the full phase space

standard Gaussian pulls in every slice — calibrated globally and locally

Toy model — systematics from learning rate

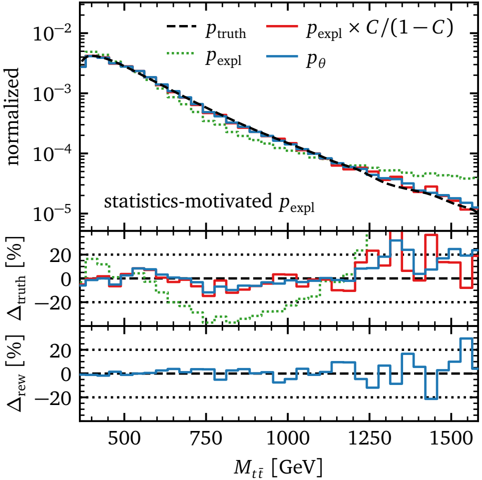


- **stochastic optimization** process in network training leads to **noisy network predictions**
- noise on prediction, just like noise on target
- captured by HNF uncertainty $\sigma_{\text{syst}}(x)$

$$\min_x \sigma_{\text{syst}}(x) \propto \sqrt{\eta}$$

- no noise injected by hand — network reports its **own training noise**

No explicit likelihood? Build one...



Problem: heteroscedastic loss needs $p_{truth}(x)$ — realistic LHC problems do not have it

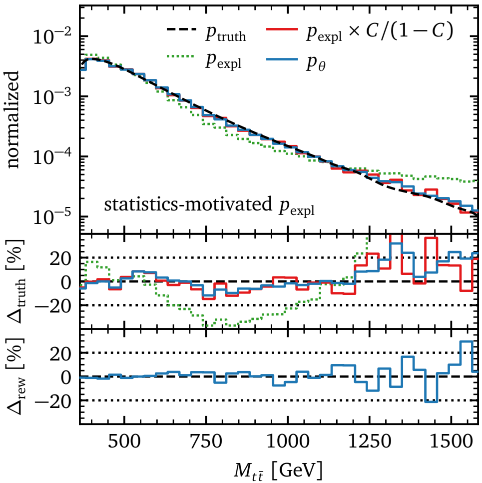
Uncertainty-aware HNF training:

1. train **approximate explicit density** $p_{expl}(x)$, here: NF undertrained on half the data, only 100 steps
2. train **classifier** $C(x)$ between $p_{expl}(x)$ and data
3. **reweight** — Neyman–Pearson, likelihood ratio trick [arXiv:1506.02169, arXiv:2009.03796]

$$p_{expl}(x) \times \frac{C(x)}{1 - C(x)} \approx p_{truth}(x)$$

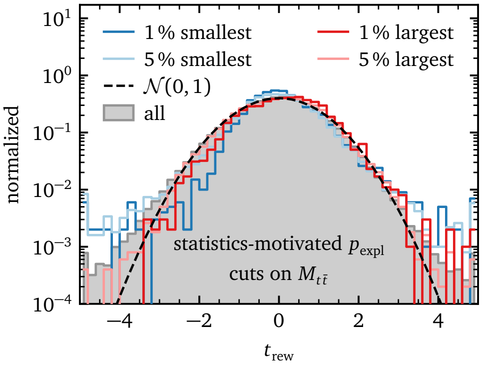
4. train **HNF** setup on this explicit target

Physics example — top pair production

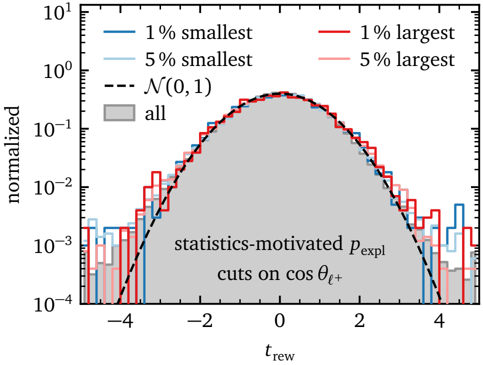


- physics benchmark: $gg \to t\bar{t} \to (be^+\nu)(\bar{b}e^-\bar{\nu})$
- top pair production with leptonic decays, **16-dim. phase space**
- classifier reweighting **repairs tails**, and **HNF follows its target** at percent level

Uncertainty calibration — pull distributions



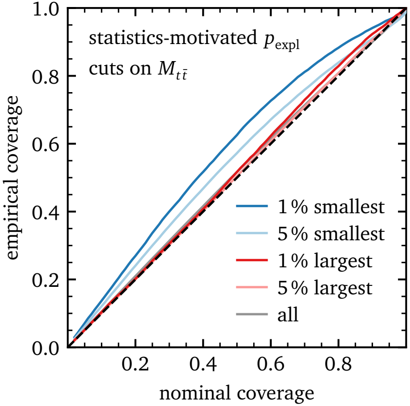
invariant mass of the $t\bar{t}$ pair



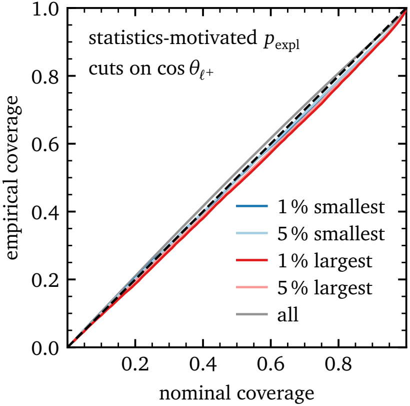
polar angle of e^+ in W^+ rest frame

learned systematic uncertainty is well-calibrated globally and locally

Uncertainty calibration — empirical coverage



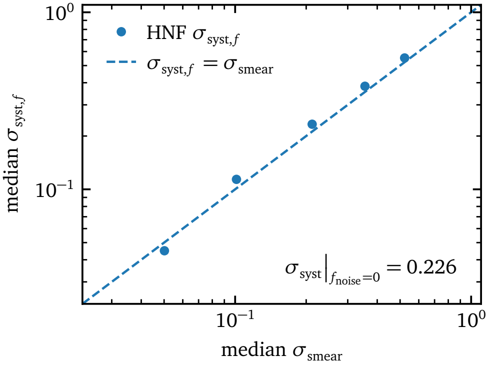
invariant mass of the $t\bar{t}$ pair

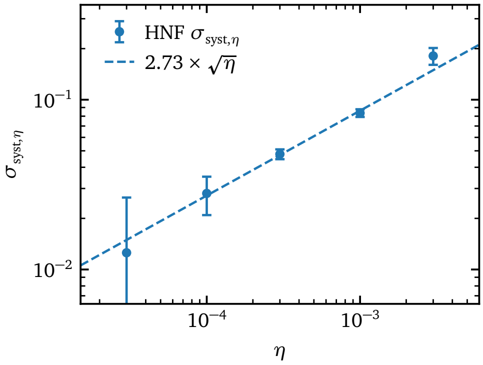


polar angle of e^+ in W^+ rest frame

learned systematic uncertainty is well-calibrated globally and locally

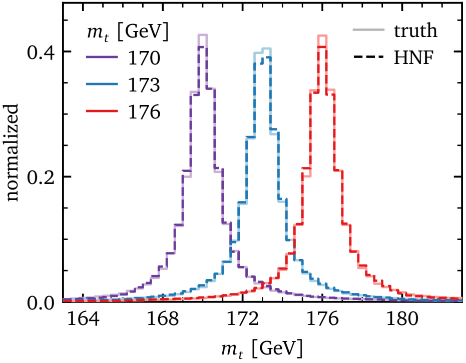
Top pair production — systematics





both toy-model scalings survive the realistic LHC problem

Nuisance parameters — conditional HNF

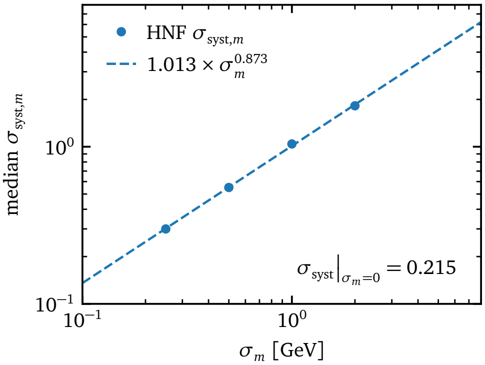


- top mass as **nuisance parameter**, $m_t = 167, 168, \ldots, 179\,\text{GeV}$
- conditioning on nuisance parameters is **established** [arXiv:2110.13632]
- $p_{\text{expl}}(x)$ trained on all top masses, **conditional classifier** $C(x|m_t)$ gives HNF target

$$p_{\text{expl}}(x) \times \frac{C(x|m_t)}{1 - C(x|m_t)}$$

- **conditional HNF** reproduces **top mass peaks** and correlated observables

Nuisance parameters — absorbed uncertainty



- **new**: train HNF on same target, but **without conditioning** on m_t
- train unconditioned HNF with noisy target density

$$p_{\text{expl}}(x) \times \frac{C(x|m_t)}{1 - C(x|m_t)}$$

$$m_t \sim 173\,\text{GeV} + \mathcal{N}(0, \sigma_m)$$

- target looks noisy, and heteroscedastic loss puts that spread into $\sigma_{\text{syst}}(x)$
- **HNF absorbs varying top mass** into learned systematic uncertainty $\sigma_{\text{syst}}(x)$

Thank you!

Summary

- BNF learns statistical uncertainty, vanishing with more training data
- HNF learns systematic uncertainty, from noisy targets and from its own training noise
- classifier reweighting provides explicit target density when no likelihood is available
- learned systematics are well-calibrated
- HNF absorbs nuisance parameters into its systematic uncertainty

Know What You Don’t Flow

arXiv:2608.21509 [hep-ph]