Representation Learning for Particle Physics

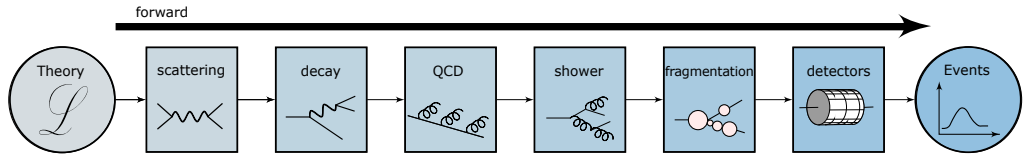
Tilman Plehn

Universität Heidelberg

AI Boninchi Workshop 2025, Geneva

LHC Theory goal

Turn data into fundamental physics



- QFT
  start with Lagrangian
- compute hard scattering
  compute decays
  compute jet radiation
- partons inside protons
  hadron-level QCD
- detector simulations

$\rightarrow$ First-principle simulations, not data modeling

HL-LHC: inference with 10$\times$more data

- statistical improvement $\sqrt{10} > 3$
- rate over phase space to $< 0.1\%$
- SBI starts with Simulation $\leftrightarrow$ theory
- speed the key to precision
- MadGraph7, ML-Sherpa, ML-Pythia, ML-(N)NLO, NNPDF...

$\rightarrow$ Summarize LHC physics in a Lagrangian

LHC vs language

Similar to fit

- approximate $f_\theta(x) \approx f(x)$
- x low-D phase space
- f_θ numerical function

$\rightarrow$ θ data representation

Probabilities over phase space

- regression $x \to A_\theta(x)$
- classification $x \to p_\theta(x)$ [likelihood ratio]
- generation $r \sim \mathcal{N} \to x \sim p_\theta(x)$
- conditional generation $r \sim \mathcal{N} \to x \sim p_\theta(x|y)$

LHC representations

- accuracy
- precision
- structure

$\rightarrow$ All related by physics knowledge

LHC representation: L-GATr

Encode known symmetries [Brehmer, Breso, de Haan, TP, Qu, Spinner, Thaler]

1- permutation invariance $\to$ graph or transformer
transformer $\equiv$ learned representation with scalar product

2- Lorentz co-variance/equivariance $[S, f_\theta](x) = 0$

- input: 4-vectors vs Mandelstams $\to$ scalars, vectors, etc?
- geometric product extending vector space

$$xy = \frac{\{x,y\}}{2} + \frac{[x,y]}{2} \qquad \text{with} \qquad \{\gamma^\mu, \gamma^\nu\} = 2g^{\mu\nu}$$

- metric reducing, bivector increasing grade

$$\gamma^\mu \gamma^\nu = \frac{\{\gamma^\mu, \gamma^\nu\}}{2} + \frac{[\gamma^\mu, \gamma^\nu]}{2} \equiv g^{\mu\nu} + \sigma^{\mu\nu}$$

- multivector of geometric algebra [just like supersymmetry in 90s]

$$x = x^S\, 1 + x^V_\mu\, \gamma^\mu + x^B_{\mu\nu}\, \sigma^{\mu\nu} + x^A_\mu\, \gamma^\mu\gamma^5 + x^P\, \gamma^5 \qquad \text{with} \qquad \begin{pmatrix} x^S \\ x^V_\mu \\ x^B_{\mu\nu} \\ x^A_\mu \\ x^P \end{pmatrix} \in \mathbb{R}^{16}$$

- example input (PID, p_μ)

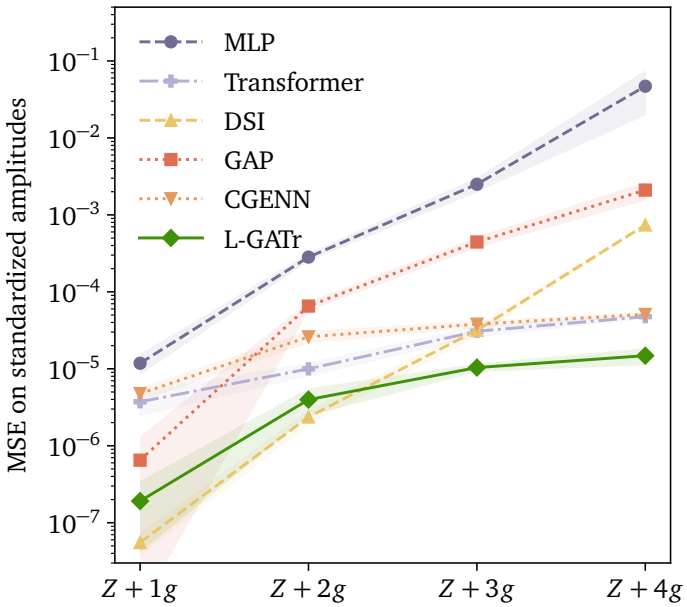
$$x^S = \text{PID} \qquad x^V_\mu = p_\mu \qquad x^T_{\mu\nu} = x^A_\mu = x^P = 0$$

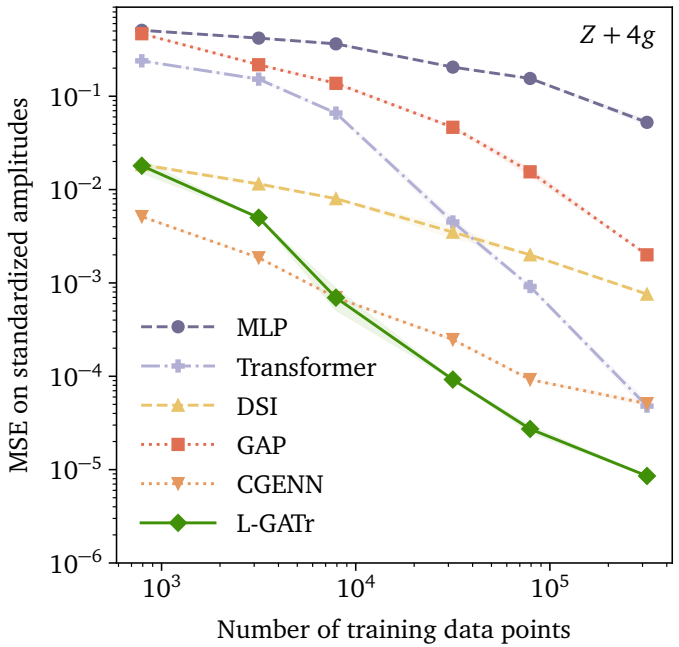
$\to$ Organize network layers by grade: L-GATr

L-GATr amplitudes

Performance is all you need

- LO transition amplitudes $q\bar{q} \to Z + 1...4\ g$
- amplitude regression: cost scaling with dimensionality
  $\to$ low-dimensional representation better
- performance scaling with multiplicity and training data
  $\to$ winning transformers



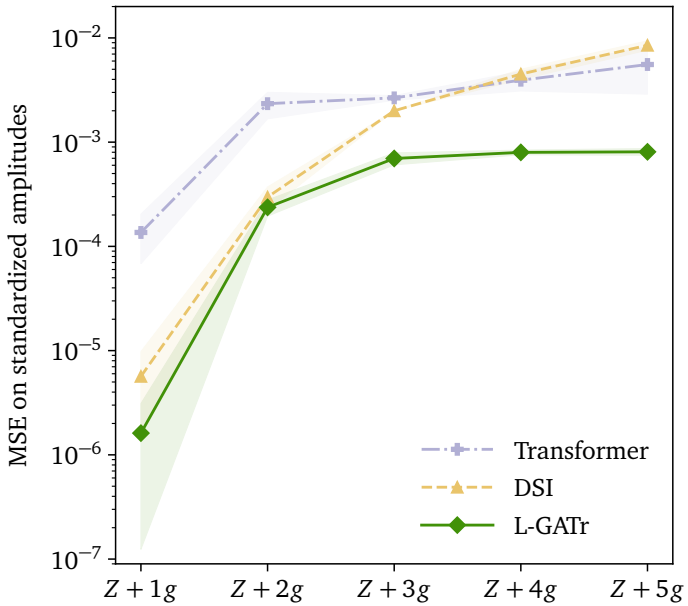


L-GATr amplitudes

Performance is all you need

- LO transition amplitudes $q\bar{q} \to Z + 1\ldots4\ g$
- amplitude regression: cost scaling with dimensionality
  $\to$ low-dimensional representation better
- performance scaling with multiplicity and training data
  $\to$ winning transformers
- more scaling with less data
  $\to$ equivariance with additional advantage

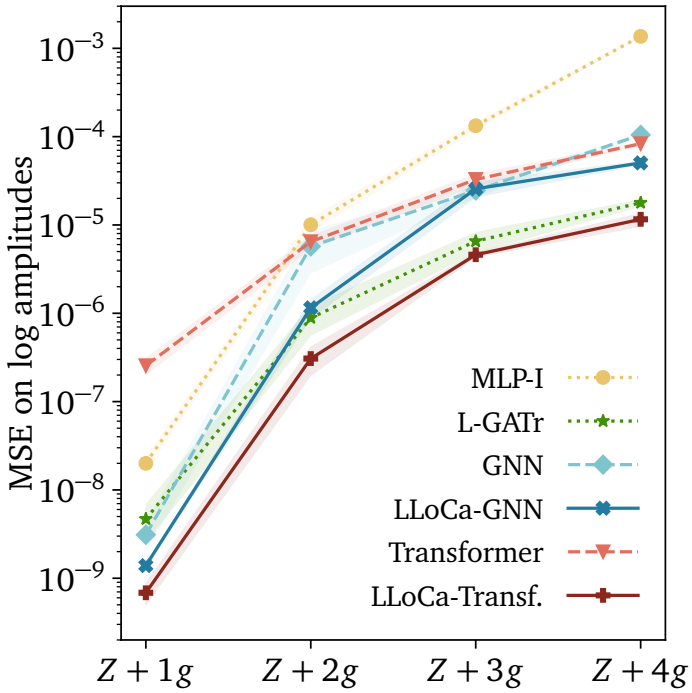
$\to$ Equvariant transformers current winner

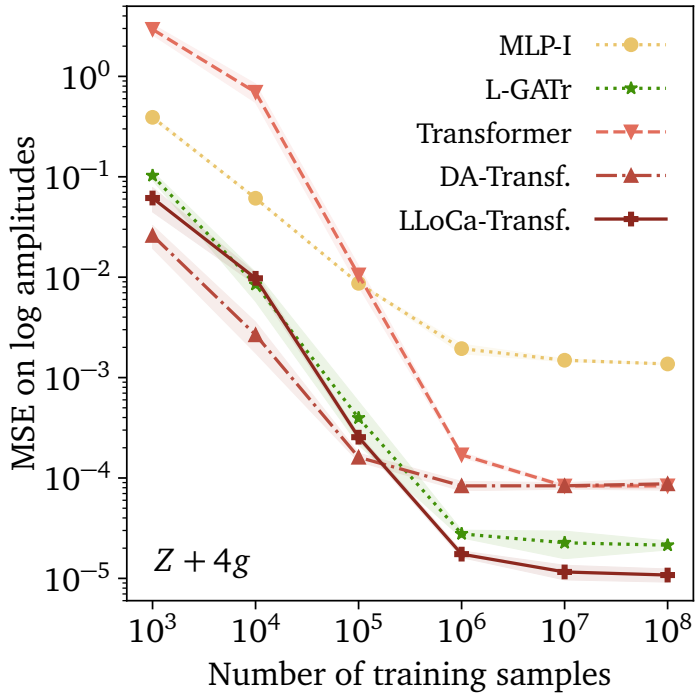


LLoCa vs L-GATr amplitudes

Alternative implementation [Favaro, TP, Spinner + Hamprecht group]

- Better performance is all you need
- local Lorentz frame for each particle
  message passing between particles in local frames
- scaling with multiplicity and training data
  $\rightarrow$ L-GATs and LLoCa comparable



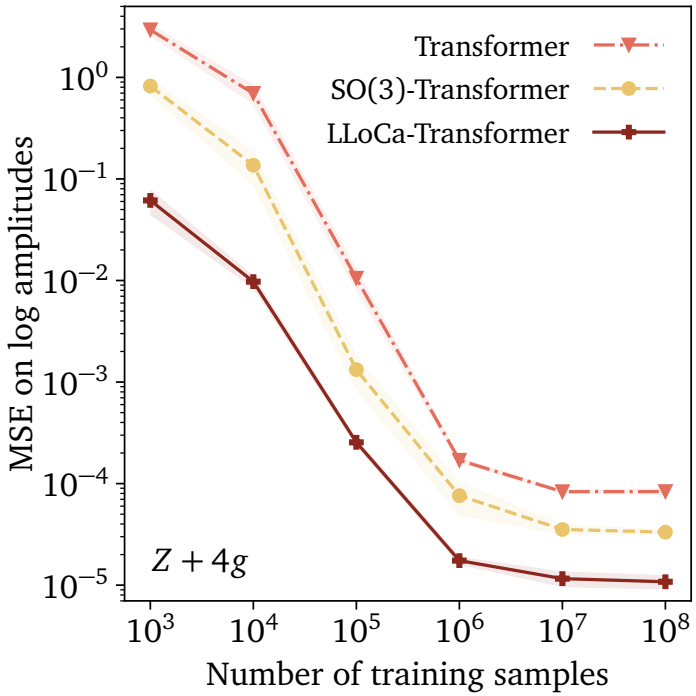


LLoCa vs L-GATr amplitudes

Alternative implementation [Favaro, TP, Spinner + Hamprecht group]

- Better performance is all you need
- local Lorentz frame for each particle
  message passing between particles in local frames
- scaling with multiplicity and training data
  $\to$ L-GATs and LLoCa comparable
- broken symmetries: rotations
  $\to$ intermediate symmetries means intermediate performance

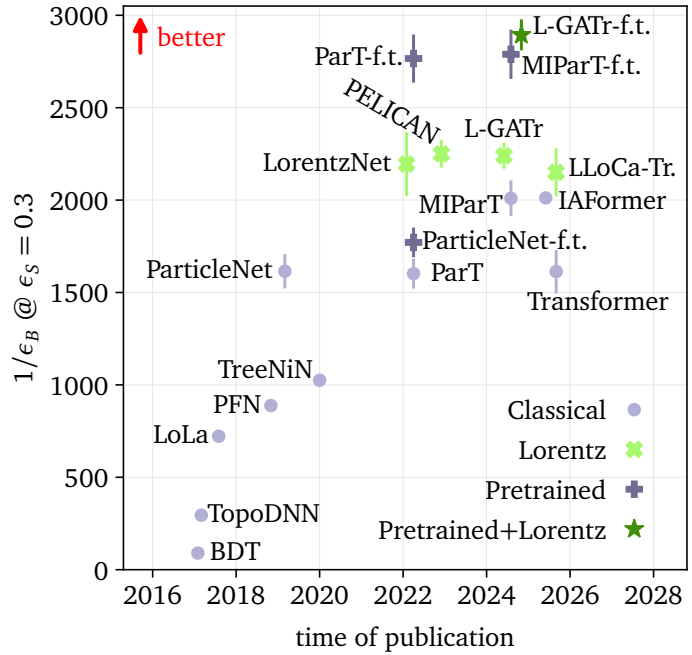
$\to$ Advantage of equivariance for all implementations



Equivariant jet tagging

Fat jet tagging benchmarks [same bottom line as flavor tagging]

- problem: jet axis breaking Lorentz-symmetry
  1- give jet axis explicitly as additional 4-vector
  2- reduce LLoCa symmetry
- top tagging performance
  $\rightarrow$ equivariance + pre-training leading
  $\rightarrow$ 30-fold background rejection from best BDT



Equivariant jet tagging

Fat jet tagging benchmarks [same bottom line as flavor tagging]

- problem: jet axis breaking Lorentz-symmetry
  1- give jet axis explicitly as additional 4-vector
  2- reduce LLoCa symmetry
- top tagging performance
  $\to$ equivariance + pre-training leading
  $\to$ 30-fold background rejection from best BDT
- multi-class performance
  $\to$ equivariance still leading

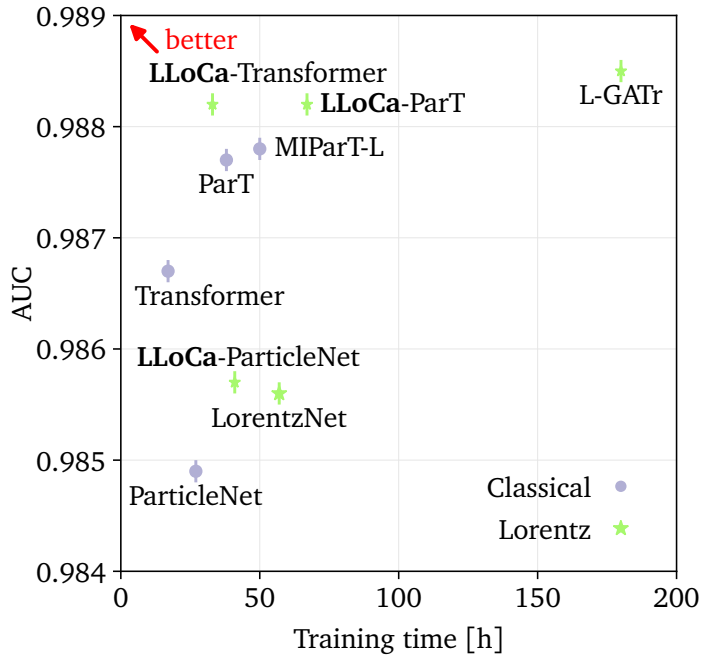
	$H \to b\bar{b}$	$H \to c\bar{c}$	$H \to gg$	$H \to 4q$	$H \to l\nu q\bar{q}'$	$t \to bq\bar{q}'$	$t \to bl\nu$	$W \to q\bar{q}'$	$Z \to q\bar{q}$
	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{99\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{99.5\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$
PFN [72]	2924	841	75	198	265	797	721	189	159
P-CNN [73]	4890	1276	88	474	947	2907	2304	241	204
MIParT-L [68]	10753	4202	123	1927	5450	31250	16807	542	402
LorentzNet [22]	8475	2729	111	1152	3515	13889	10257	400	303
L-GATr [26]	12987	4819	128	2311	6116	47619	20408	588	432
ParticleNet [18]	7634	2475	104	954	3339	10526	11173	347	283
LLoCa-ParticleNet*	7463	2833	105	1072	3155	10753	9302	403	306
ParT [19]	10638	4149	123	1864	5479	32787	15873	543	402
LLoCa-ParT*	11561	4640	125	2037	5900	41667	19231	552	419
Transformer	10753	3333	116	1369	4630	24390	17857	415	334
LLoCa-Transformer*	11628	4651	125	2037	5618	39216	17241	548	410

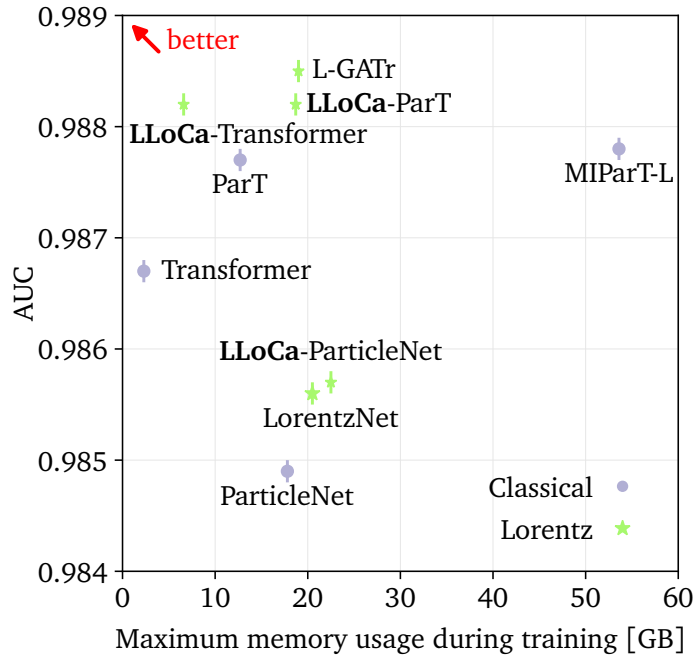
Equivariant jet tagging

Fat jet tagging benchmarks [same bottom line as flavor tagging]

- problem: jet axis breaking Lorentz-symmetry
  1- give jet axis explicitly as additional 4-vector
  2- reduce LLoCa symmetry
- top tagging performance
  → equivariance + pre-training leading
  → 30-fold background rejection from best BDT
- multi-class performance
  → equivariance still leading

→ Efficiency is what you also need...

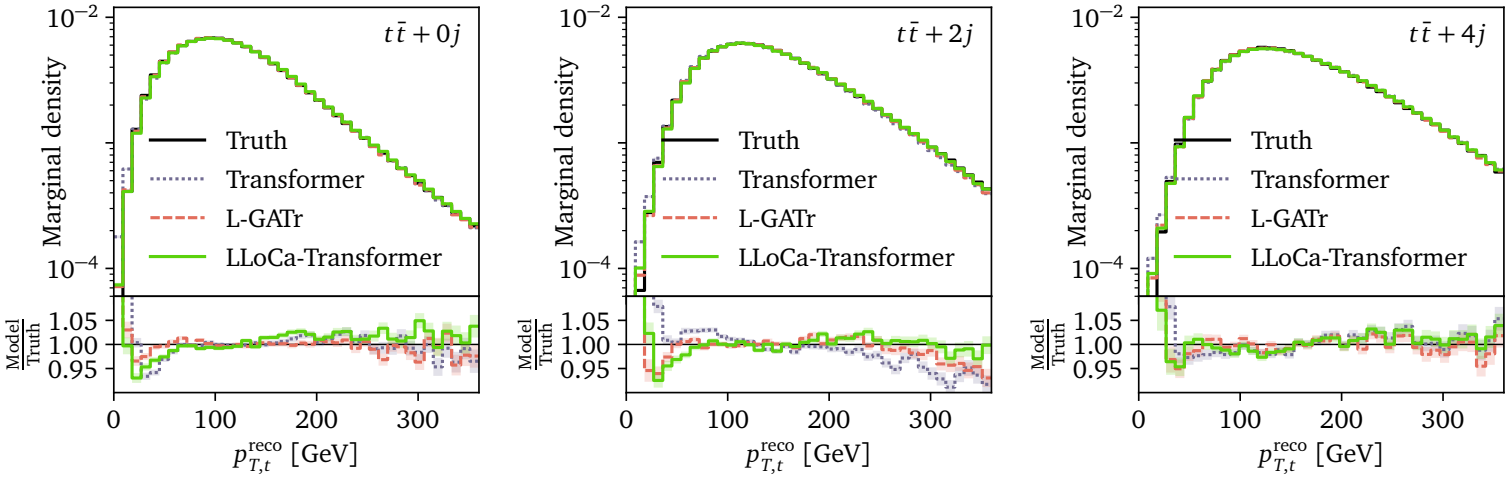




Equivariant event generation

Conditional flow matching with transformer-velocity

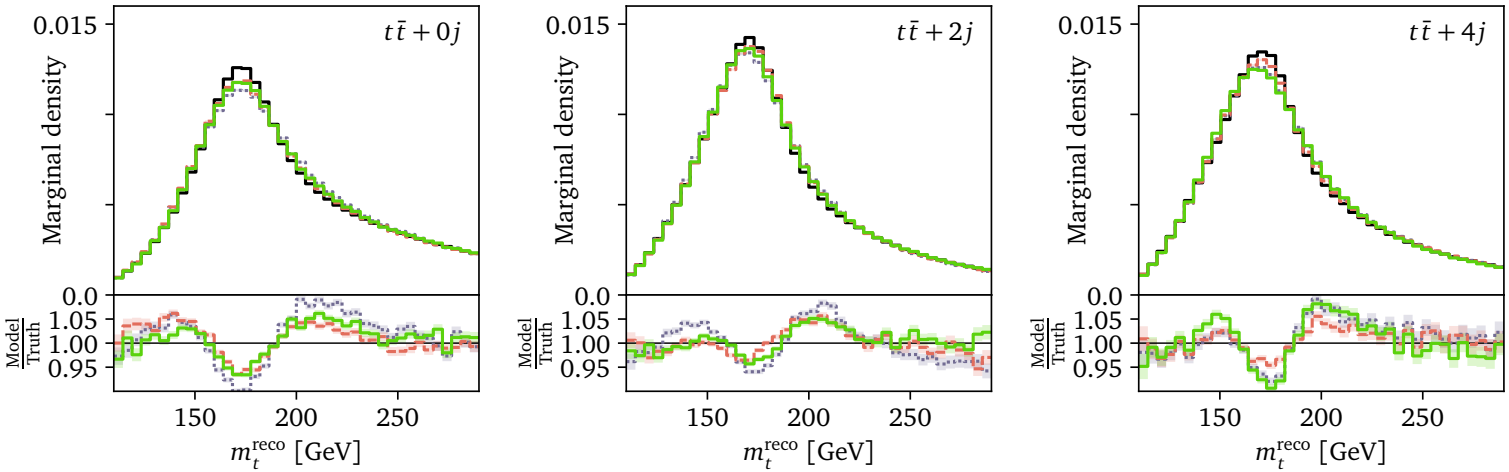
- end-to-end event generation
- all-hadronic events $pp \to t_{\text{had}}\bar{t}_{\text{had}} + 0\ldots4j$ [22M training events]
- per-cent kinematics for all particles



Equivariant event generation

Conditional flow matching with transformer-velocity

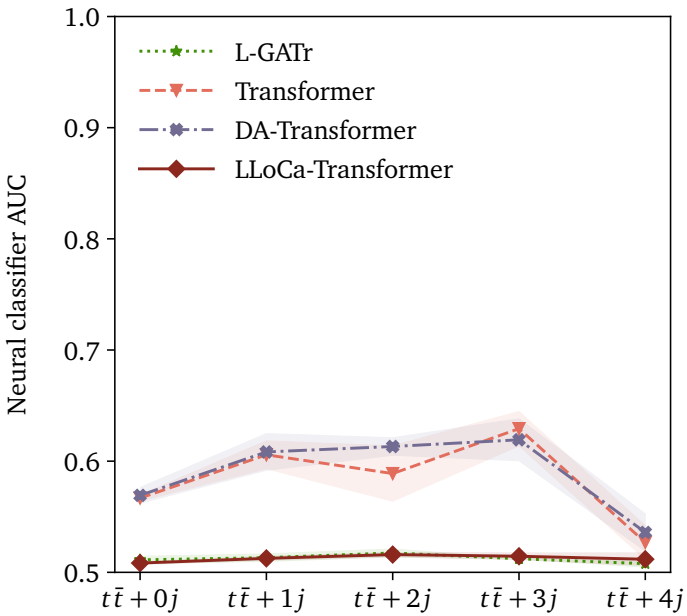
- end-to-end event generation
- all-hadronic events $pp \to t_{\text{had}}\bar{t}_{\text{had}} + 0\ldots4j$ [22M training events]
- per-cent kinematics for all particles
- invariant top mass

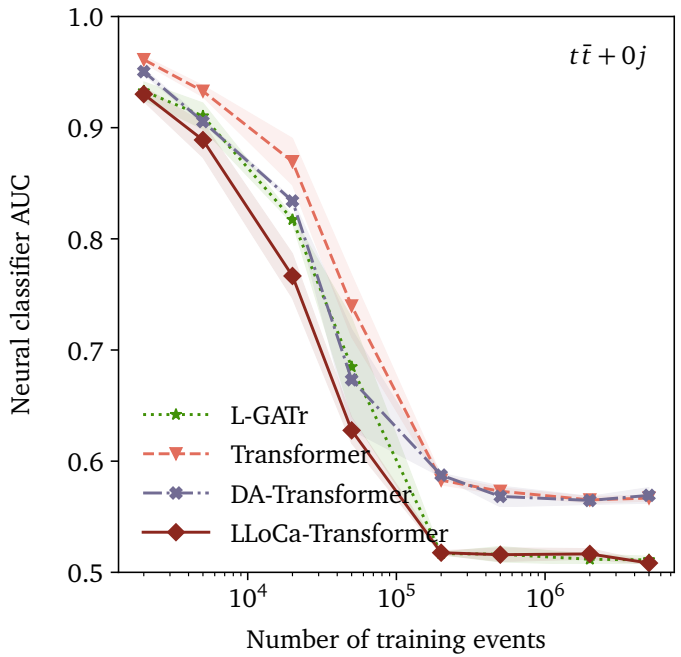


Equivariant event generation

Conditional flow matching with transformer-velocity

- end-to-end event generation
- all-hadronic events $pp \to t_{\text{had}}\bar{t}_{\text{had}} + 0\ldots4j$ [22M training events]
- per-cent kinematics for all particles
- invariant top mass
- classifier between training and generated datasets



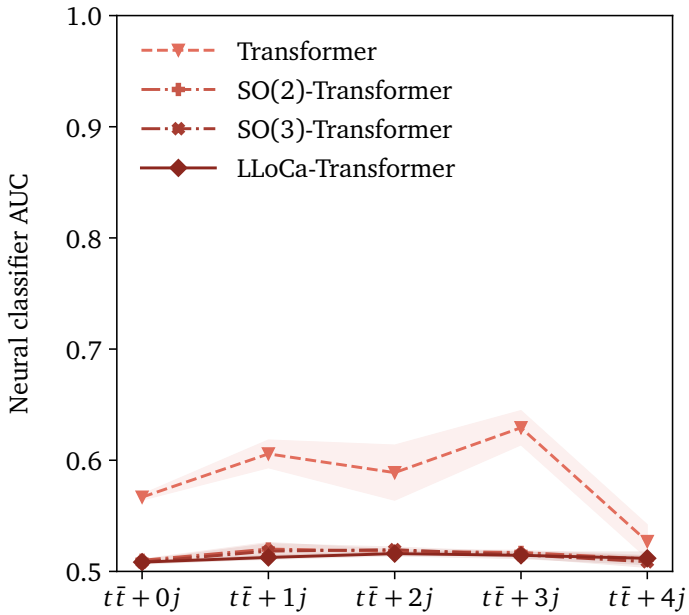


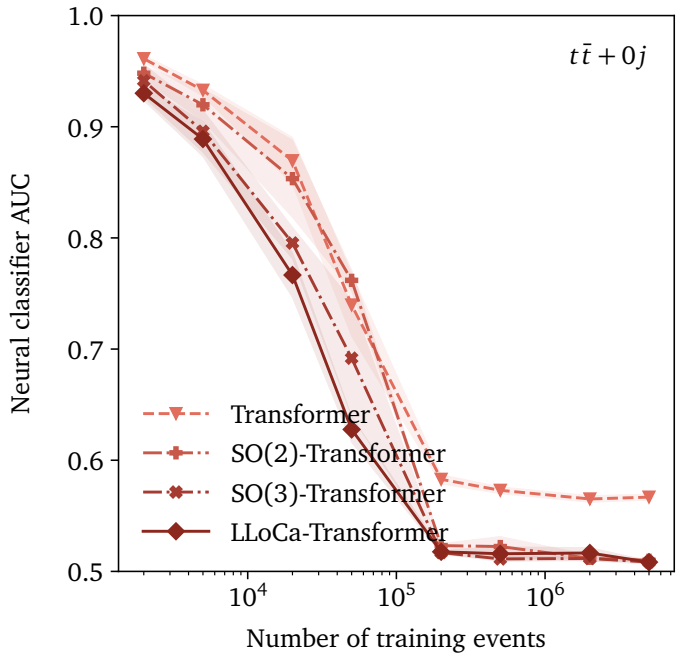
Equivariant event generation

Conditional flow matching with transformer-velocity

- end-to-end event generation
- all-hadronic events $pp \to t_{\text{had}}\bar{t}_{\text{had}} + 0\ldots4j$ [22M training events]
- per-cent kinematics for all particles
- invariant top mass
- classifier between training and generated datasets

$\to$ Precision-phase space a solved problem...





Accurate, precise, controlled generative ML

Performance increase

- Variational Autoencoder [too old] $\longrightarrow$ GAN [2019]
- normalizing flow [2020] $\longrightarrow$ diffusion [2023]
- JetGPT [2023] $\longrightarrow$ vision transformer [2024]

Accurate, precise, controlled generative ML

Performance increase

- Variational Autoencoder [too old] $\longrightarrow$ GAN [2019]
- normalizing flow [2020] $\longrightarrow$ diffusion [2023]
- JetGPT [2023] $\longrightarrow$ vision transformer [2024]

Classifier test [Das, Favaro, Heimel, TP, Shih]

- generation: unsupervised density
- classify training vs generated events $D(x)$
  learned density ratio [Neyman-Pearson]

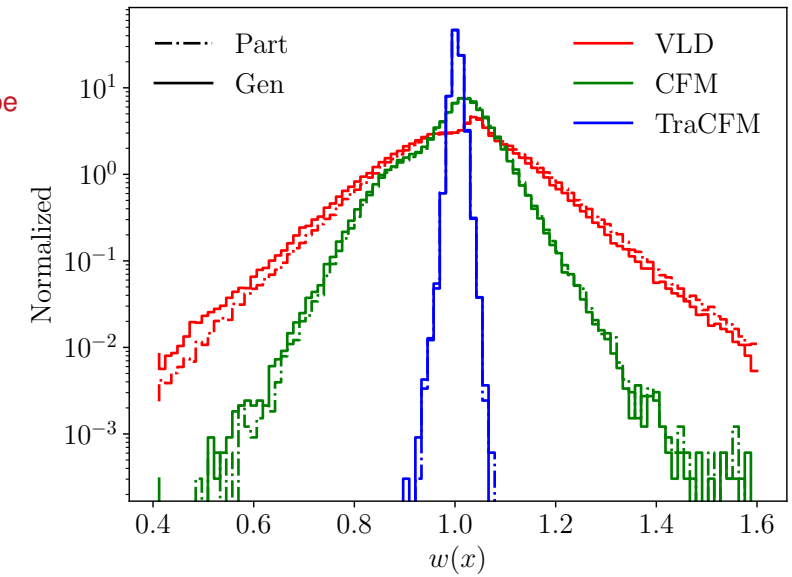
$$w(x_i) = \frac{D(x_i)}{1 - D(x_i)} = \frac{p_{\text{data}}(x_i)}{p_{\text{model}}(x_i)}$$

$\rightarrow$ Local quality test

E.g. unfolding generators [Berkeley-Irvine-Heidelbe

- width of weight distribution
- tails indicating failure mode

$\rightarrow$ Systematic generator benchmarking



Encoding mean and uncertainty

Network training as a fit

- learn scalar field $f_\theta(x) \approx f(x)$
- statistics: maximize parameter probability given (f_j, σ_j)

$$p(\theta|x) = \frac{p(x|\theta)\; p(\theta)}{p(x)}$$

$\to$ maximize tractable likelihood instead

$$p(x|\theta) = \prod_j \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{|f_j - f_\theta(x_j)|^2}{2\sigma_j^2}\right)$$

$$\Rightarrow \qquad \mathcal{L} \equiv -\log p(x|\theta) = \sum_j \frac{|f_j - f_\theta(x_j)|^2}{2\sigma_j^2} + \text{const}(\theta)$$

Encoding mean and uncertainty

Network training as a fit

- learn scalar field $f_\theta(x) \approx f(x)$
- statistics: maximize parameter probability given (f_j, σ_j)

$$p(\theta|x) = \frac{p(x|\theta)\; p(\theta)}{p(x)}$$

$\to$ maximize tractable likelihood instead

$$p(x|\theta) = \prod_j \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{|f_j - f_\theta(x_j)|^2}{2\sigma_j^2}\right)$$

$$\Rightarrow \qquad \mathcal{L} \equiv -\log p(x|\theta) = \sum_j \frac{|f_j - f_\theta(x_j)|^2}{2\sigma_j^2} + \text{const}(\theta)$$

Learned local uncertainty

- Gaussian log-likelihood with normalization

$$\mathcal{L}_\text{heteroscedastic} = \frac{|f(x) - f_\theta(x)|^2}{2\sigma_\theta(x)^2} + \log \sigma_\theta(x) + \cdots$$

- if needed replace $\sigma_\theta(x)$ by mixture model

$\to$ learn $f_\theta(x)$ and $\sigma_\theta(x)$ together

1- Bayesian networks

Learned function statistically

- amplitude over phase phase

$$\langle A \rangle = \int dA \; A \; p(A)$$

- internal representation θ of training data T [think Gaussian with mean and width]

$$p(A) = \int d\theta \; p(A|\theta) \; p(\theta|T)$$

$\rightarrow$ θ-distribution defining Bayesian NN

1- Bayesian networks

Learned function statistically

- amplitude over phase phase

$$\langle A \rangle = \int dA \; A \; p(A)$$

- internal representation θ of training data T [think Gaussian with mean and width]

$$p(A) = \int d\theta \; p(A|\theta) \; p(\theta|T)$$

$\rightarrow$ θ-distribution defining Bayesian NN

Variational approximation

- definition of training

$$p(A) = \int d\theta \; p(A|\theta) \; p(\theta|T) \approx \int d\theta \; p(A|\theta) \; q(\theta)$$

- similarity through minimal KL-divergence [Bayes' theorem to remove unknown posterior]

$$\begin{aligned} D_{\text{KL}}[q(\theta), p(\theta|T)] &= \int d\theta \; q(\theta) \; \log \frac{q(\theta)}{p(\theta|T)} \\ &= \int d\theta \; q(\theta) \; \log \frac{q(\theta)p(T)}{p(T|\theta)p(\theta)} \\ &\approx D_{\text{KL}}[q(\theta), p(\theta)] - \int d\theta \; q(\theta) \; \log p(T|\theta) \equiv \mathcal{L} \end{aligned}$$

$\rightarrow$ Two-term loss: likelihood + prior

1- Bayesian networks

Statistical network evaluation

- expectation value from network $q(\theta)$

$$\langle A \rangle = \int dA d\theta \; A \; p(A|\theta) \; q(\theta)$$
$$\equiv \int d\theta \; q(\theta) \overline{A}(\theta) \qquad \text{with} \qquad \overline{A}(\theta) = \int dA \; A \; p(A|\theta)$$

- corresponding variance

$$\sigma^2_{\text{tot}} = \int dA d\theta \;\; (A - \langle A \rangle)^2 \;\; p(A|\theta) \; q(\theta)$$
$$= \int d\theta \; q(\theta) \left[\overline{A^2}(\theta) - \overline{A}(\theta)^2 + \Big(\overline{A}(\theta) - \langle A \rangle \Big)^2 \right] \equiv \sigma^2_{\text{syst}} + \sigma^2_{\text{stat}}$$

Two uncertainties

- statistical — vanishing for perfect training: $q(\theta) \to \delta(\theta - \theta_0)$

$$\sigma^2_{\text{stat}} = \int d\theta \; q(\theta) \left[\overline{A}(\theta) - \langle A \rangle \right]^2$$

- systematic — vanishing for perfect data: $p(A|\theta) \to \delta(A - A_0)$

$$\sigma^2_{\text{syst}} = \int d\theta \; q(\theta) \left[\overline{A^2}(\theta) - \overline{A}(\theta)^2 \right]$$

- (ir)-reducible errors in ML language

$\to$ Systematics dominant for LHC

2- Repulsive ensembles

Posterior from network ensemble

- OED vs continuity equation

$$\frac{d\theta}{dt} = v(\theta, t) \qquad \Leftrightarrow \qquad \frac{\partial \rho(\theta, t)}{\partial t} = -\nabla_\theta \left[v(\theta, t) \rho(\theta, t) \right]$$

- Fokker-Planck equation with stationary $\rho(\theta, t) = \pi(\theta)$

$$\frac{d\theta}{dt} = -\nabla_\theta \log \frac{\rho(\theta, t)}{\pi(\theta)}$$

- ODE describing training progress

$$\theta^{t+1} - \theta^t \propto -\nabla_{\theta^t} \left[\log \rho(\theta^t) - \log \pi(\theta^t) \right]$$

$$= -\nabla_{\theta^t} \left[\log \sum_j k(\theta^t, \theta_j^t) - \log p(\theta | x_{\text{train}}^t) \right] \equiv -\nabla_{\theta^t} \mathcal{L}_{\text{RE}}$$

$\rightarrow$ Joint ensemble training

2- Repulsive ensembles

Posterior from network ensemble

- OED vs continuity equation

$$\frac{d\theta}{dt} = v(\theta, t) \qquad \Leftrightarrow \qquad \frac{\partial \rho(\theta, t)}{\partial t} = -\nabla_\theta \left[v(\theta, t) \rho(\theta, t) \right]$$

- Fokker-Planck equation with stationary $\rho(\theta, t) = \pi(\theta)$

$$\frac{d\theta}{dt} = -\nabla_\theta \log \frac{\rho(\theta, t)}{\pi(\theta)}$$

- ODE describing training progress

$$\theta^{t+1} - \theta^t \propto -\nabla_{\theta^t} \left[\log \rho(\theta^t) - \log \pi(\theta^t) \right]$$

$$= -\nabla_{\theta^t} \left[\log \sum_j k(\theta^t, \theta_j^t) - \log p(\theta | x_{\text{train}}^t) \right] \equiv -\nabla_{\theta^t} \mathcal{L}_{\text{RE}}$$

$\rightarrow$ Joint ensemble training

Repulsive ensembles

- train network ensemble
- apply repulsive force
  kernel in function space

$\rightarrow$ Alternative for statistical uncertainty

3- Evidential regression

Uncertainties from latent distribution, without sampling [Bahl, Elmer, TP, Winterhalder]

- evidential distribution

$$p(A) = \int d\lambda\, p(A|\lambda)\, p(\lambda|T) \approx \int d\lambda\, p(A|\lambda)\, p(\lambda|\theta_0)$$

assuming $p(A|\lambda) = \mathcal{N}\left(A|\,\overline{A}, \sigma^2\right)$ with $\lambda \equiv (\overline{A}, \sigma^2)$

- choose $p(\lambda|\theta_0)$ as conjugate prior

$$p(\lambda|\theta_0) = \frac{\beta^\alpha \sqrt{v}}{\Gamma(\alpha)\sqrt{2\pi\sigma^2}} \left(\frac{1}{\sigma^2}\right)^{\alpha+1} \exp\left(-\frac{2\beta + v(\gamma - \overline{A})^2}{2\sigma^2}\right)$$

with $\{\gamma, v, \alpha, \beta\}\,(x, \theta_0)$

- analytic likelihood: Student-t

$$p(A) = \text{St}\left(A\middle|\gamma, \frac{\beta(1+v)}{v\alpha}, 2\alpha\right)$$

$$A_\text{NN} = \int d\lambda\, \overline{A}\, p(\lambda|\theta_0) = \gamma$$

$$\sigma^2_\text{syst} = \int d\lambda\, \sigma^2\, p(\lambda|\theta_0) = \frac{\beta}{\alpha - 1}$$

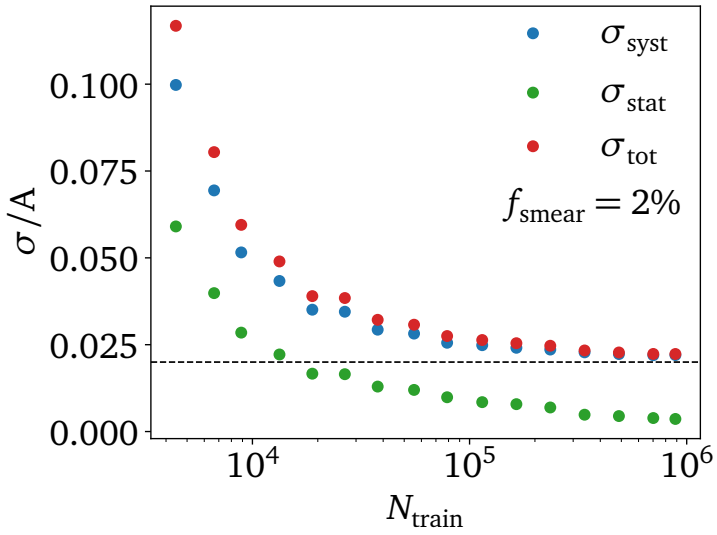
$$\sigma^2_\text{stat} = \int d\lambda \left[\overline{A} - A_\text{NN}\right]^2 p(\lambda|\theta_0) = \frac{\beta}{v(\alpha - 1)}$$

→ Another alternative for learned uncertainty

Amplitudes with calibrated uncertainties

Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

- systematics: artificial noise
- statistics plateau

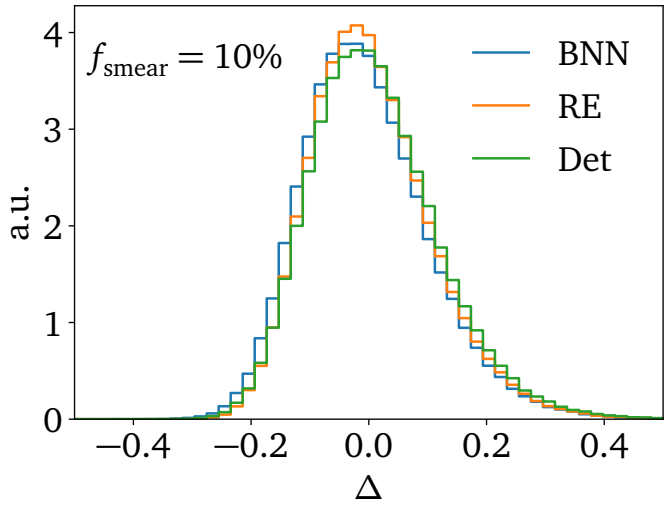


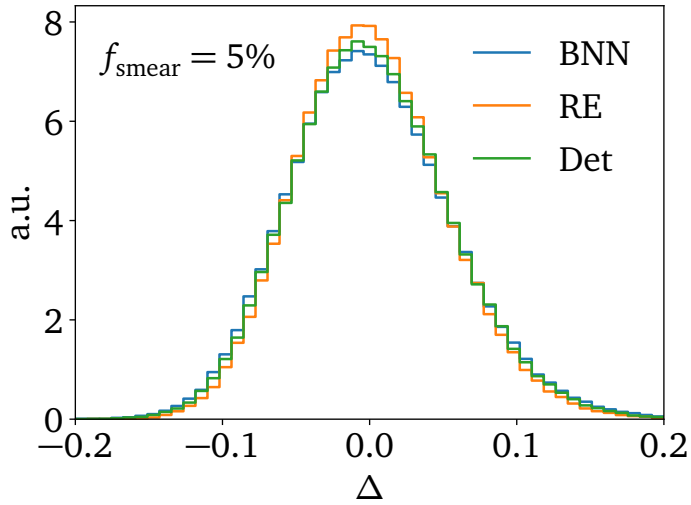
Amplitudes with calibrated uncertainties

Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

- systematics: artificial noise
- statistics plateau
- accuracy over phase space

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





Amplitudes with calibrated uncertainties

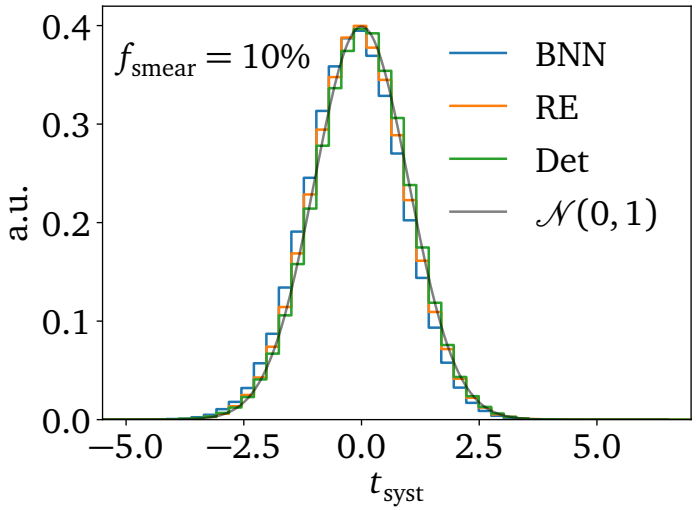
Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

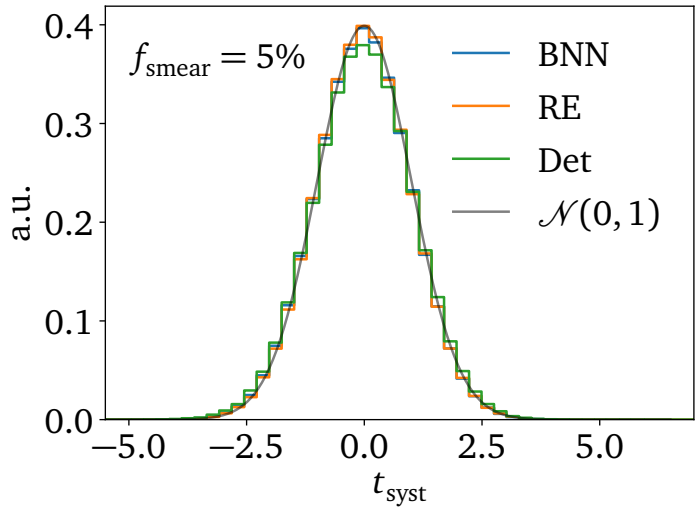
- systematics: artificial noise
- statistics plateau
- accuracy over phase space

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

- pull over phase space

$$t_{\text{syst}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{\sigma_{\text{syst}}(x)}$$





Amplitudes with calibrated uncertainties

Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

- systematics: artificial noise
- statistics plateau
- accuracy over phase space

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

- pull over phase space

$$t_{\text{syst}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{\sigma_{\text{syst}}(x)}$$

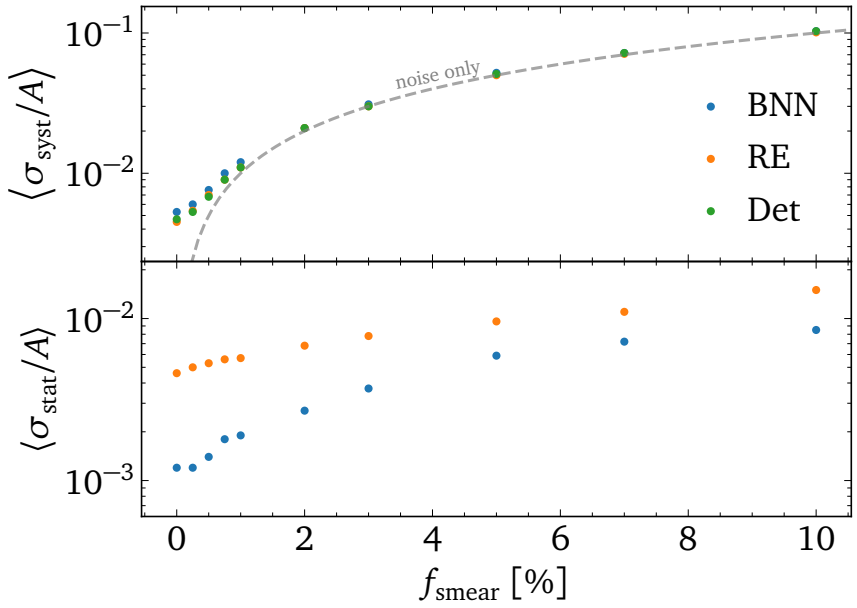
Towards zero noise

- scaling

$$\sigma^2_{\text{syst}} - \sigma^2_{\text{syst},0} \approx \sigma^2_{\text{train}}$$

- plateau $\langle \sigma_{\text{syst}}/A \rangle \sim 0.4\%$

$\to$ Limiting factor??



Amplitudes with calibrated uncertainties

Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

- systematics: artificial noise
- statistics plateau
- accuracy over phase space

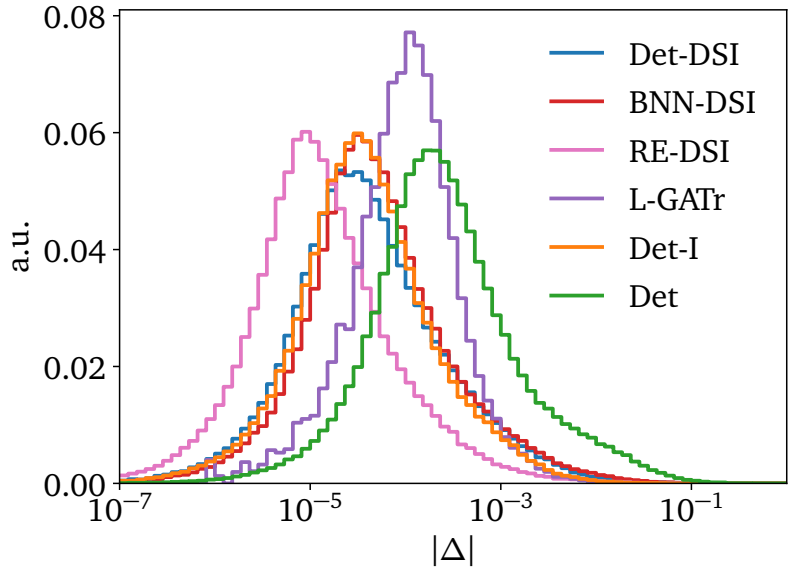
$$\Delta(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{A_{\text{true}}(x)}$$

- pull over phase space

$$t_{\text{syst}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{\sigma_{\text{syst}}(x)}$$

Data representation [Breso, Heinrich, Magerya, Olsson]

- amplitude from invariants
- learn Minkowski metric
- Deep-sets-invariant network



Amplitudes with calibrated uncertainties

Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

- systematics: artificial noise
- statistics plateau
- accuracy over phase space

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

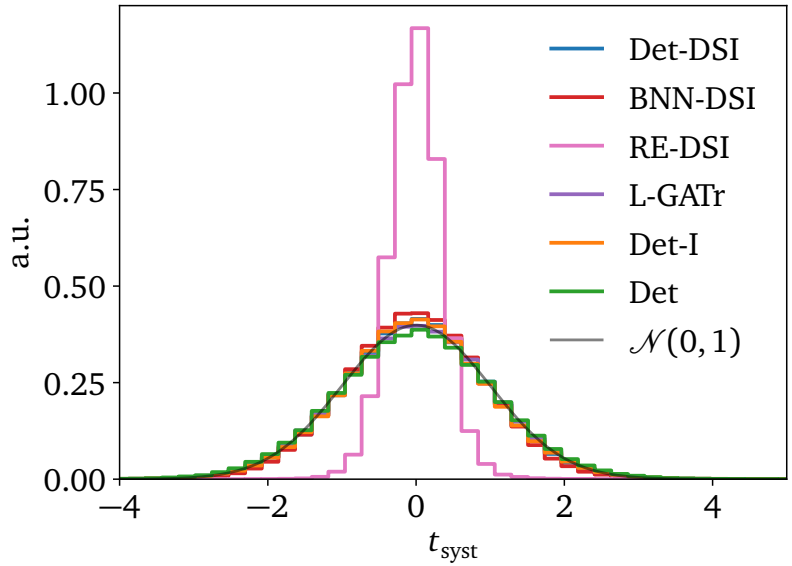
- pull over phase space

$$t_{\text{syst}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{\sigma_{\text{syst}}(x)}$$

Data representation [Breso, Heinrich, Magerya, Olsson]

- amplitude from invariants
- learn Minkowski metric
- Deep-sets-invariant network

$\to$ Calibrated systematics



Amplitudes with calibrated uncertainties

Loop amplitude $gg \to \gamma\gamma g(g)$ [Bahl, Elmer, Favaro, Haußmann, TP, Winterhalder]

- systematics: artificial noise
- statistics plateau
- accuracy over phase space

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

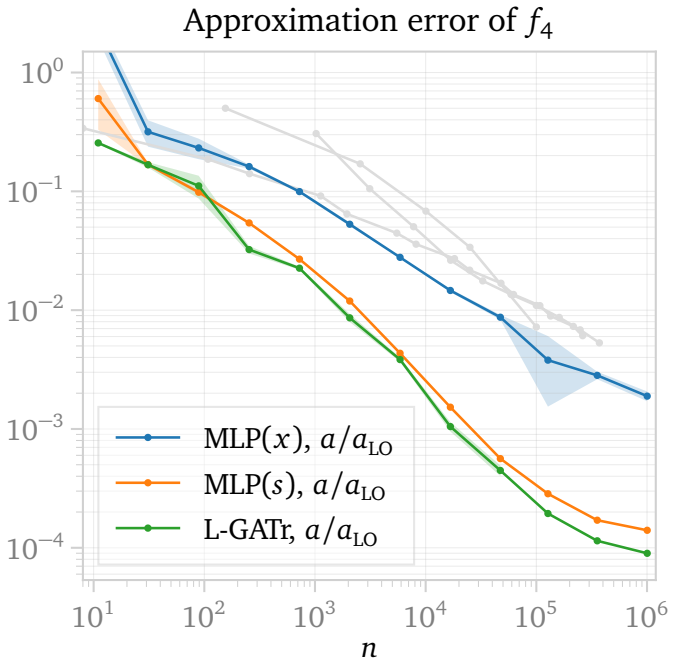
- pull over phase space

$$t_{\text{syst}}(x) = \frac{A_{\text{NN}}(x) - A_{\text{true}}(x)}{\sigma_{\text{syst}}(x)}$$

Data representation [Breso, Heinrich, Magerya, Olsson]

- amplitude from invariants
- learn Minkowski metric
- Deep-sets-invariant network

$\to$ On to real loop integrals



Phase space gaps

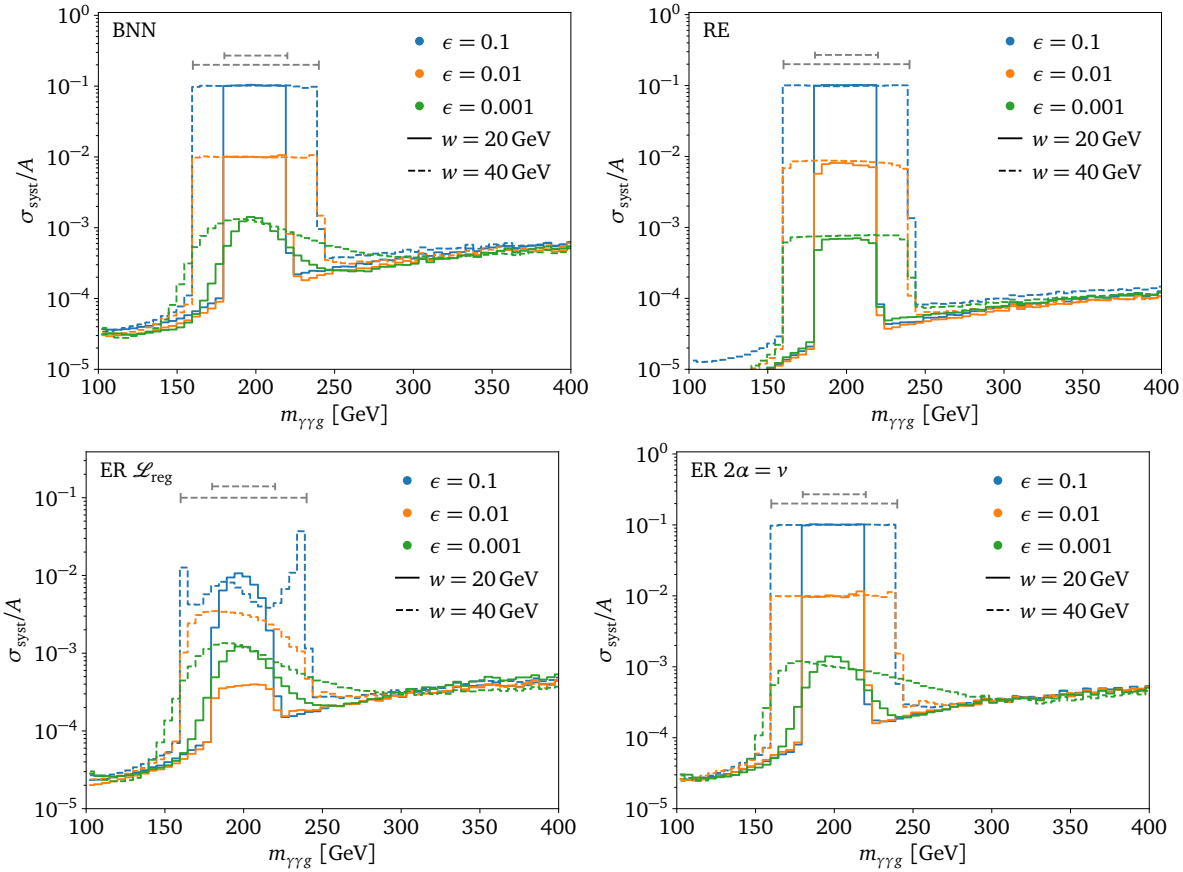
Noisy amplitude gap [Bahl, Elmer, TP, Winterhalder]

- step-function relative noise in $m_{\gamma\gamma g}$ gap

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

- compare Bayesian NN, ensembles, evidential regression

$\rightarrow$ Only little noise corrected, any noise learned



More about ensembles

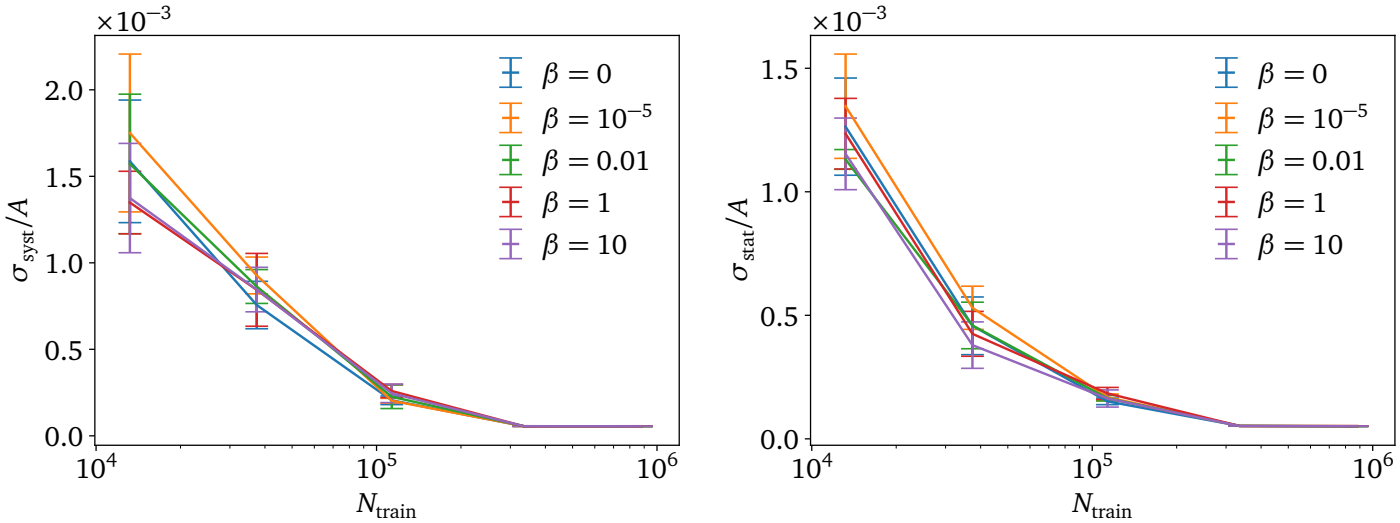
How repulsive?

- two-term repulsive ensemble loss

$$\mathcal{L}_{\text{RE}} = \sum_{i=1}^{N_{\text{ens}}} \left[-\frac{1}{B} \sum_{b=1}^{B} \log p(A|x_b, \theta_i) + \frac{\beta}{N} \frac{\sum_{j=1}^{N_{\text{ens}}} k(\bar{A}(x,\theta_i), \hat{A}(x,\theta_j))}{\sum_{j=1}^{N_{\text{ens}}} k(\hat{A}(x,\theta_i), \hat{A}(x,\theta_j))} + \dots \right]$$

- $\beta = 1$ from derivation, turned into hyperparameter [I hate to do that!]
- $\beta \to 1$ means standard independent ensemble

$\to$ Repulsive ensembles hardly repel



More about ensembles

How repulsive?

- two-term repulsive ensemble loss

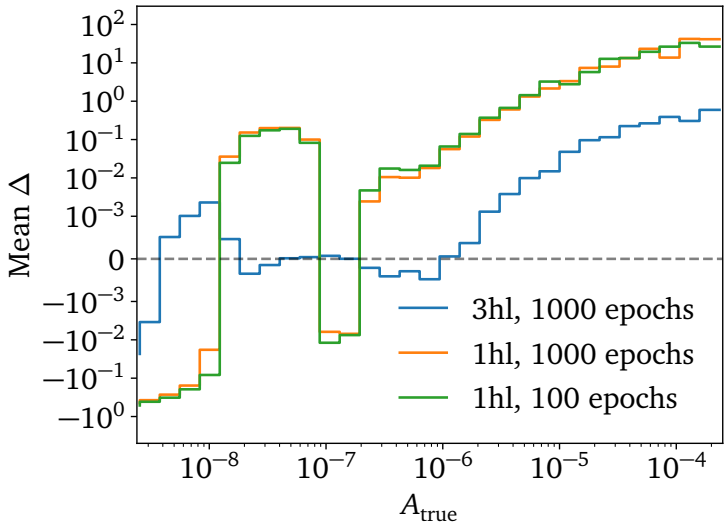
$$\mathcal{L}_{\text{RE}} = \sum_{i=1}^{N_{\text{ens}}} \left[-\frac{1}{B} \sum_{b=1}^{B} \log p(A|x_b, \theta_i) + \frac{\beta}{N} \frac{\sum_{j=1}^{N_{\text{ens}}} k(\bar{A}(x,\theta_i), \hat{A}(x,\theta_j))}{\sum_{j=1}^{N_{\text{ens}}} k(\hat{A}(x,\theta_i), \hat{A}(x,\theta_j))} + \dots \right]$$

- $\beta = 1$ from derivation, turned into hyperparameter [I hate to do that!]
- $\beta \to 1$ means standard independent ensemble

$\to$ Repulsive ensembles hardly repel

Ensemble bias

- ensemble of crappy network vs accurate architecture?
- limited model expressivity
  $\to$ challenging amplitude peaks [QCD, B-
  $\to$ local failure mode



More about ensembles

How repulsive?

- two-term repulsive ensemble loss

$$\mathcal{L}_{\text{RE}} = \sum_{i=1}^{N_{\text{ens}}} \left[-\frac{1}{B} \sum_{b=1}^{B} \log p(A|x_b, \theta_i) + \frac{\beta}{N} \frac{\sum_{j=1}^{N_{\text{ens}}} k(\bar{A}(x,\theta_i), \hat{A}(x,\theta_j))}{\sum_{j=1}^{N_{\text{ens}}} k(\hat{A}(x,\theta_i), \hat{A}(x,\theta_j))} + \dots \right]$$

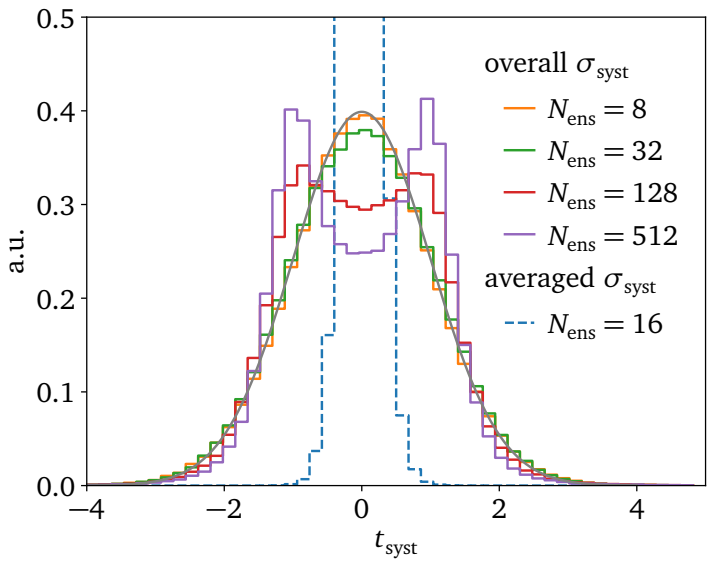
- $\beta = 1$ from derivation, turned into hyperparameter [I hate to do that!]
- $\beta \to 1$ means standard independent ensemble

$\to$ Repulsive ensembles hardly repel

Ensemble bias

- ensemble of crappy network vs accurate architecture?
- limited model expressivity
  $\to$ challenging amplitude peaks [QCD, B-W]
  $\to$ local failure mode
- systematics calibration

$\to$ Ensemble of networks not calibrated



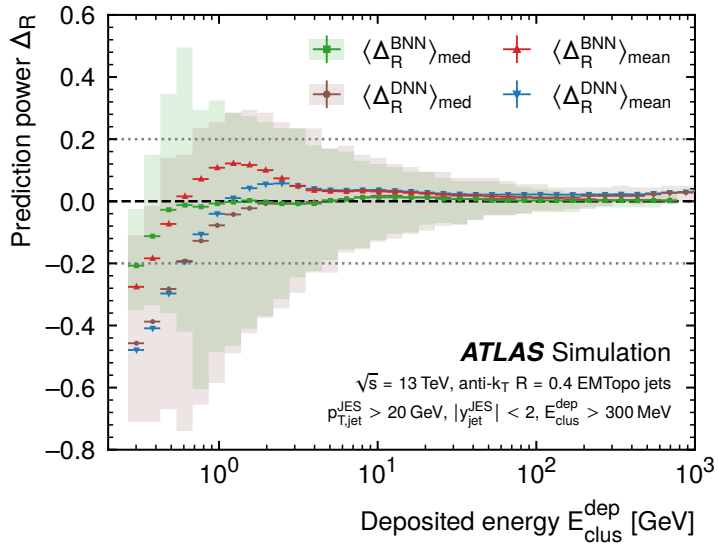
ATLAS calibration

Energy calibration with uncertainties [ATLAS + TP, Vogel]

- interpretable calorimeter phase space x
- learned calibration function

$$\mathcal{R}_{\text{NN}}(x) \pm \Delta\mathcal{R}_{\text{NN}}(x) \approx \frac{E^{\text{obs}}(x)}{E^{\text{dep}}(x)}$$

- systematics: noise in data
  network expressivity
  data representation
  definitely not Gaussian



ATLAS calibration

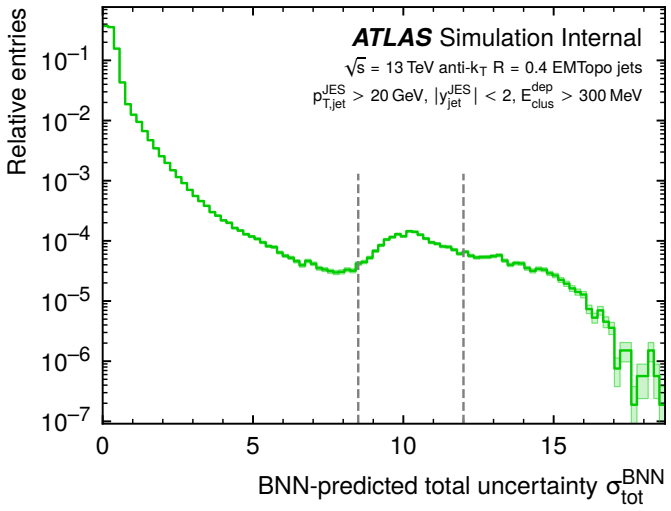
Energy calibration with uncertainties [ATLAS + TP, Vogel]

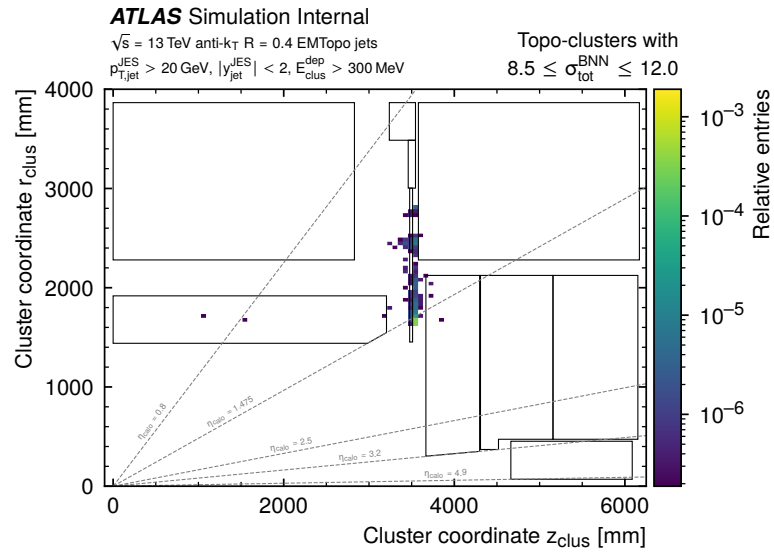
- interpretable calorimeter phase space x
- learned calibration function

$$\mathcal{R}_{\text{NN}}(x) \pm \Delta\mathcal{R}_{\text{NN}}(x) \approx \frac{E^{\text{obs}}(x)}{E^{\text{dep}}(x)}$$

- systematics: noise in data
  network expressivity
  data representation
  definitely not Gaussian

$\rightarrow$ Understand (simulated) detector





GANplification

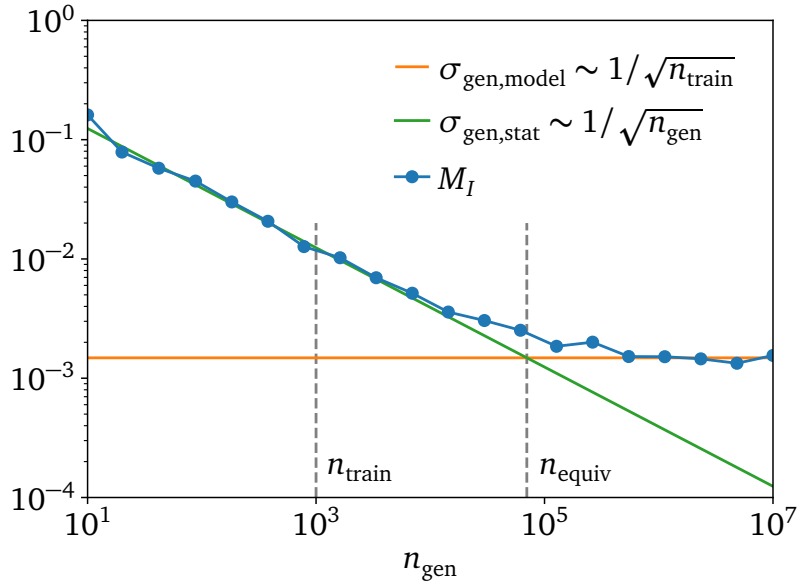
Amplification from generative networks [Bahl, Diefenbacher, Elmer, TP, Spinner]

- phase space densities

$$D_{\text{true}}^{n_{\text{train}}} \sim p_{\text{true}} \approx p_{\text{gen}}(x) \quad \Rightarrow \quad p_{\text{gen}}(x) \overset{?}{\sim} p_{\text{true}}(x)$$

- 'How many events can I sample from a network trained on n events?'
  fewer, unless trained perfectly, $p_{\text{gen}}(x) \neq p_{\text{train}}(x)$
  more, generative network smoothen $p_{\text{train}}(x)$
- scaling of two uncertainties

$$M_I = \sigma_{\text{stat}}^2 + \sigma_{\text{model}}^2 = \begin{cases} \dfrac{a}{n_{\text{gen}}} & n_{\text{gen}} \ll n_{\text{train}} \\ \dfrac{b}{n_{\text{train}}} & n_{\text{gen}} \gg n_{\text{train}} \end{cases} \qquad \text{transition} \quad G = \frac{n_{\text{gen}}}{n_{\text{train}}} = \frac{a}{b}$$



GANplification

Amplification from generative networks [Bahl, Diefenbacher, Elmer, TP, Spinner]

- phase space densities

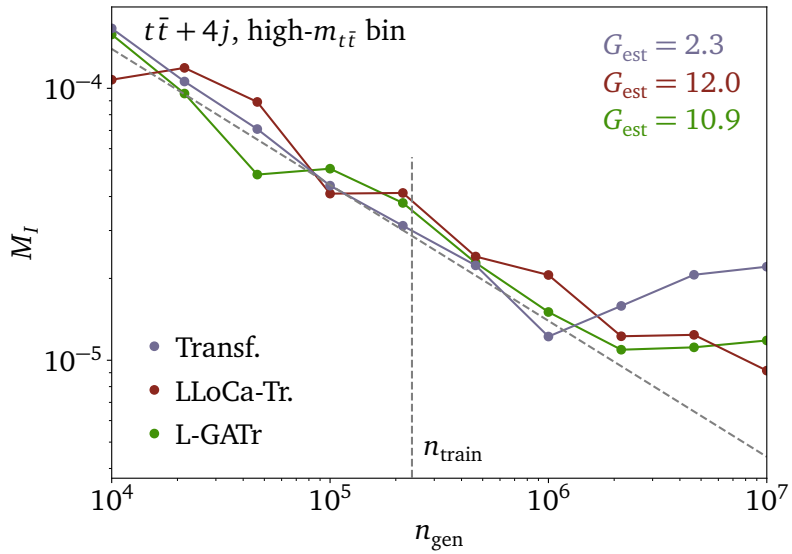
$$D_{\text{true}}^{n_{\text{train}}} \sim p_{\text{true}} \approx p_{\text{gen}}(x) \quad \Rightarrow \quad p_{\text{gen}}(x) \overset{?}{\sim} p_{\text{true}}(x)$$

- 'How many events can I sample from a network trained on n events?'
  fewer, unless trained perfectly, $p_{\text{gen}}(x) \neq p_{\text{train}}(x)$
  more, generative network smoothen $p_{\text{train}}(x)$
- scaling of two uncertainties

$$M_I = \sigma^2_{\text{stat}} + \sigma^2_{\text{model}} = \begin{cases} \dfrac{a}{n_{\text{gen}}} & n_{\text{gen}} \ll n_{\text{train}} \\ \dfrac{b}{n_{\text{train}}} & n_{\text{gen}} \gg n_{\text{train}} \end{cases} \qquad \text{transition} \quad G = \frac{n_{\text{gen}}}{n_{\text{train}}} = \frac{a}{b}$$

- compare p_{gen} and p_{train}
- scale KS-test
- obtain gen-equivalent training size

→ L-GATr GANplifies...



Extrapolating ISR

Universal QCD jet radiation [$Z + 1 \ldots 8$ jets]

- from n to $n+1$ jets

$$R_{(n+1)/n} = \frac{\sigma_{n+1}}{\sigma_n} \quad \text{and} \quad P(n) = \frac{\sigma_n}{\sigma_{\text{tot}}} \quad \text{with} \quad \sigma_{\text{tot}} = \sum_{n=0}^{\infty} \sigma_n \; .$$

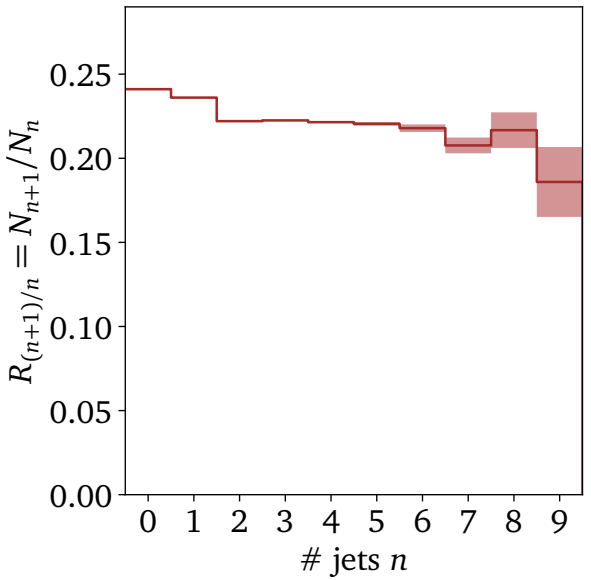
- large scale drop: Poisson scaling

$$R_{(n+1)/n} = \frac{\bar{n}}{n+1} \quad \Leftrightarrow \quad P(n) = \frac{\bar{n}^n e^{-\bar{n}}}{n!} \; .$$

- democratic scales: staircase scaling

$$R_{(n+1)/n} = e^{-b} = 1 - \tilde{\Delta}_g(Q^2) \equiv P(n+1|n)$$

$\rightarrow$ Universal pattern learnable?



Extrapolating ISR

Universal QCD jet radiation [$Z + 1 \ldots 8$ jets]

- democratic scales: staircase scaling

$$R_{(n+1)/n} = e^{-b} = 1 - \tilde{\Delta}_g(Q^2) \equiv P(n+1|n)$$

$\rightarrow$ Universal pattern learnable?

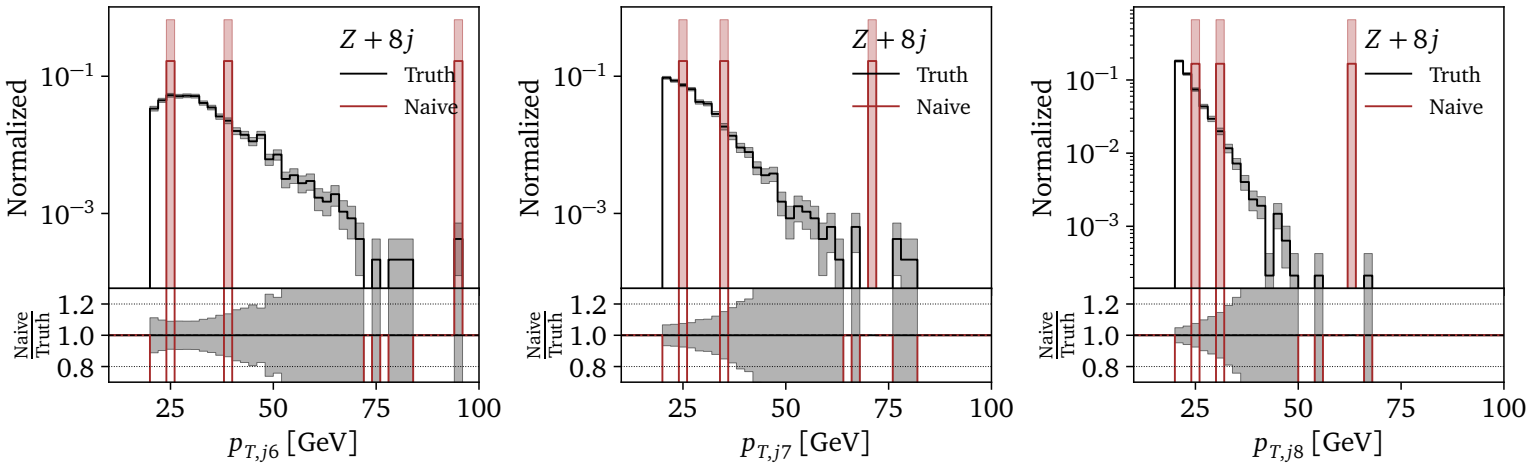
Autoregressive transformer [Butter, Charton, Villadamigo, Ore, TP, Spinner]

- factorized probability and loss function

$$p(x_i|x_{1:i-1}) = p_{\text{kin}}(x_i|x_{1:i-1})\; p_{\text{split}}(x_{1:i-1})$$

$$p(x_{1:n}) = \left[\prod_{i=1}^{n} p_{\text{kin}}(x_i|x_{1:i-1})\right] \left[\prod_{i=1}^{n} p_{\text{split}}(x_{1:i-1})\right] \left[1 - p_{\text{split}}(x_{1:n})\right] ,$$

- naive extrapolation, train up to 6 jets, generate 7 and 8



Extrapolating ISR

Universal QCD jet radiation [$Z + 1 \ldots 8$ jets]

- democratic scales: staircase scaling

$$R_{(n+1)/n} = e^{-b} = 1 - \tilde{\Delta}_g(Q^2) \equiv P(n+1|n)$$

$\rightarrow$ Universal pattern learnable?

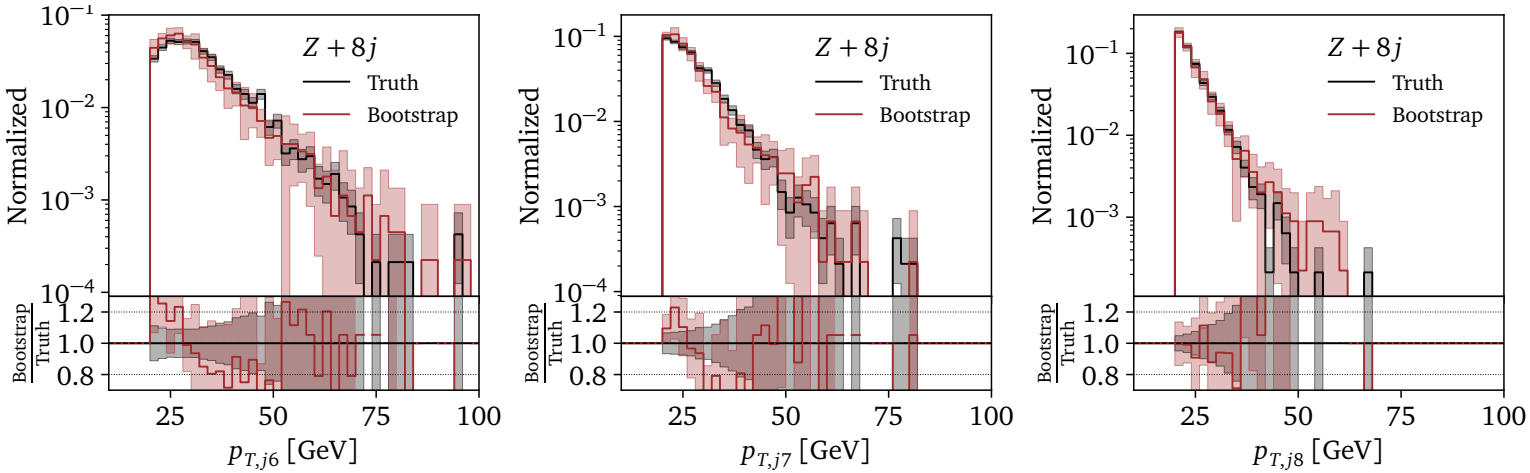
Autoregressive transformer [Butter, Charton, Villadamigo, Ore, TP, Spinner]

- factorized probability and loss function

$$p(x_i|x_{1:i-1}) = p_{\text{kin}}(x_i|x_{1:i-1})\; p_{\text{split}}(x_{1:i-1})$$

$$p(x_{1:n}) = \left[\prod_{i=1}^{n} p_{\text{kin}}(x_i|x_{1:i-1})\right] \left[\prod_{i=1}^{n} p_{\text{split}}(x_{1:i-1})\right] \left[1 - p_{\text{split}}(x_{1:n})\right] ,$$

- bootstrapped extrapolation, train up to 6 jets, generate 7 and 8



Extrapolating ISR

Universal QCD jet radiation [$Z + 1 \ldots 8$ jets]

- democratic scales: staircase scaling

$$R_{(n+1)/n} = e^{-b} = 1 - \tilde{\Delta}_g(Q^2) \equiv P(n+1|n)$$

$\rightarrow$ Universal pattern learnable?

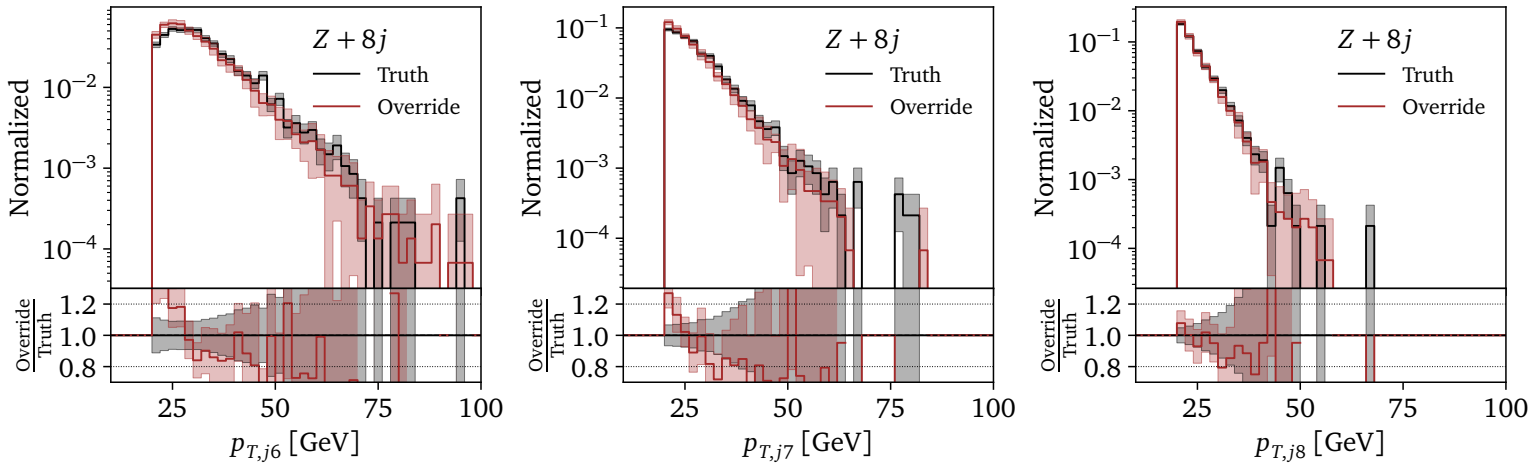
Autoregressive transformer [Butter, Charton, Villadamigo, Ore, TP, Spinner]

- factorized probability and loss function

$$p(x_i|x_{1:i-1}) = p_{\text{kin}}(x_i|x_{1:i-1})\; p_{\text{split}}(x_{1:i-1})$$

$$p(x_{1:n}) = \left[\prod_{i=1}^{n} p_{\text{kin}}(x_i|x_{1:i-1})\right] \left[\prod_{i=1}^{n} p_{\text{split}}(x_{1:i-1})\right] \left[1 - p_{\text{split}}(x_{1:n})\right] ,$$

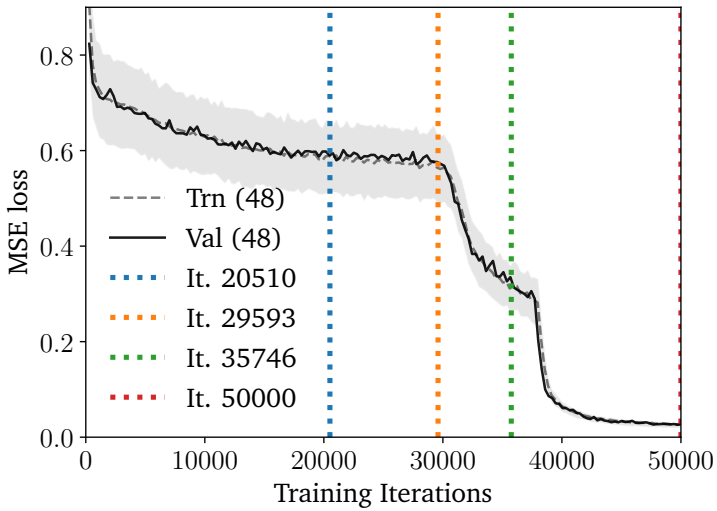
$\rightarrow$ full extrapolation with right latent representation



Symmetry-aware representations?

Amplitude ratio regression [Villadamigo, Frederix, TP, Vitos, Winterhalder]

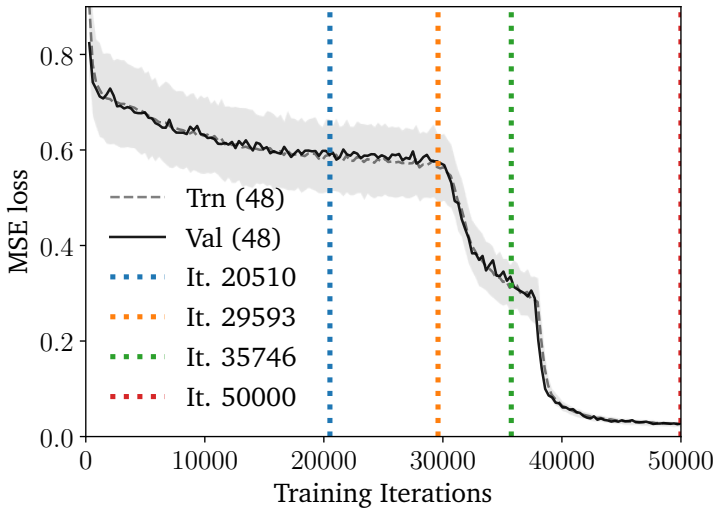
- full vs. leading color for $gg \to 4g$
- standard transformer training [not for MLP, GNN, L-GATr]

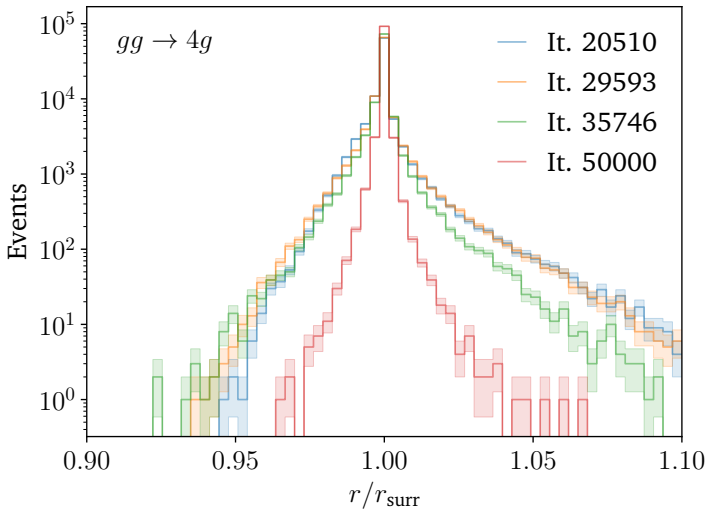


Symmetry-aware representations?

Amplitude ratio regression [Villadamigo, Frederix, TP, Vitos, Winterhalder]

- full vs. leading color for $gg \to 4g$
- standard transformer training [not for MLP, GNN, L-GATr]
- related to performance gain
- What is happening?





Symmetry-aware representations?

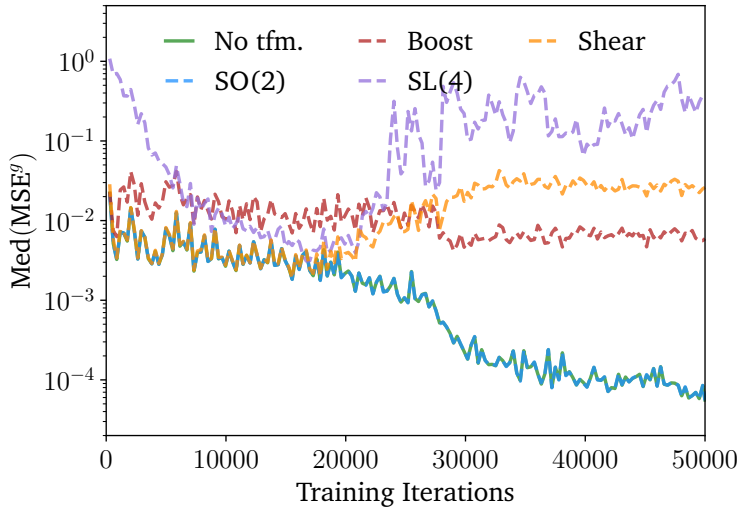
Amplitude ratio regression [Villadamigo, Frederix, TP, Vitos, Winterhalder]

- full vs. leading color for $gg \to 4g$
- standard transformer training [not for MLP, GNN, L-GATr]
- related to performance gain
- What is happening?

Symmetries?

- evaluate MSE on transformed data

  wrong: 4D rotation SL(4), $y - z$-shear
  right: Lorentz boosts, SO(2) rotations
- initially learning general structures
  then penalizing wrong symmetries
  Lorentz symmetries ...
  SO(2) always good symmetry
- What can we do with it?



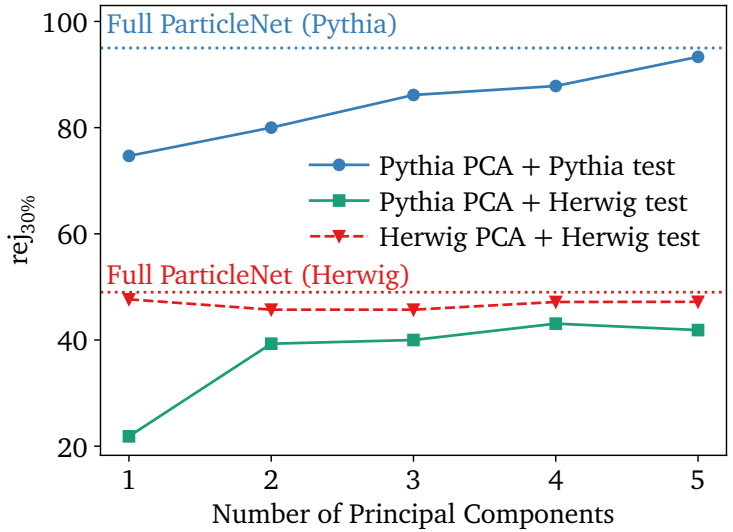
Physics from latent representation

Quarks vs gluons from trained ParticleNet [Vent, Winterhalder, TP]

- sensitive substructure variables

$$n_{\text{pf}} = \sum_i 1 \quad w_{\text{pf}} = \frac{\sum_i p_{T,i} \Delta R_{i,\text{jet}}}{p_{T,\text{jet}}} \quad p_T D = \frac{\sqrt{\sum_i p_{T,i}^2}}{\sum_i p_{T,i}} \quad C_\beta = \frac{\sum_{i<j} p_{T,i} p_{T,j} (\Delta R_{ij})^\beta}{\left(\sum_i p_{T,i}\right)^2}$$

- related to max 5 principle components? [linear decorrelation]



Physics from latent representation

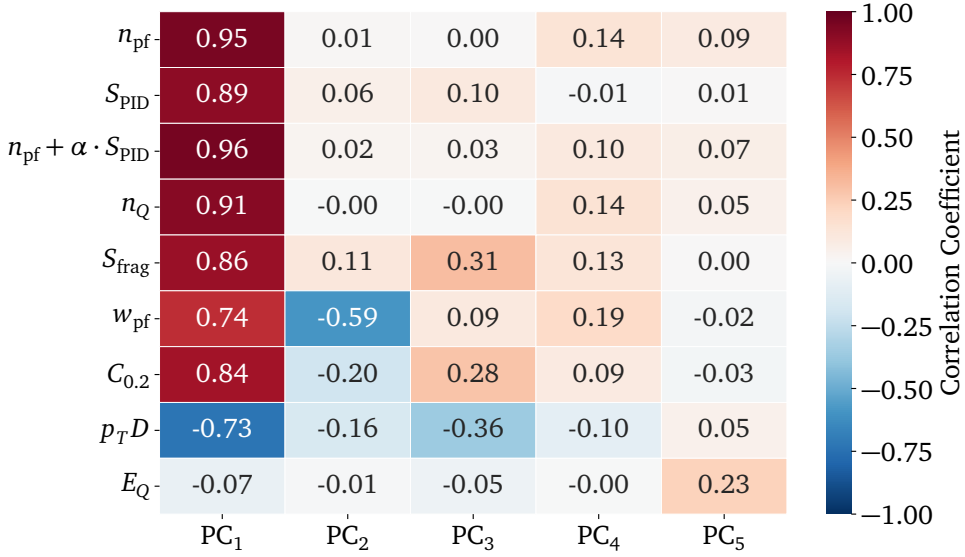
Quarks vs gluons from trained ParticleNet [Vent, Winterhalder, TP]

- sensitive substructure variables

$$n_{\text{pf}} = \sum_i 1 \quad w_{\text{pf}} = \frac{\sum_i p_{T,i} \Delta R_{i,\text{jet}}}{p_{T,\text{jet}}} \quad p_T D = \frac{\sqrt{\sum_i p_{T,i}^2}}{\sum_i p_{T,i}} \quad C_\beta = \frac{\sum_{i<j} p_{T,i} p_{T,j} (\Delta R_{ij})^\beta}{\left(\sum_i p_{T,i}\right)^2}$$

- related to max 5 principle components? [linear decorrelation]
- PC_1: constituent number and diversity

$$n_{\text{pf}} + \alpha \cdot S_{\text{PID}} \qquad \text{with} \qquad S_{\text{PID}} = -\sum_{\text{type}j} f_j \log f_j$$



Physics from latent representation

Quarks vs gluons from trained ParticleNet [Vent, Winterhalder, TP]

- sensitive substructure variables

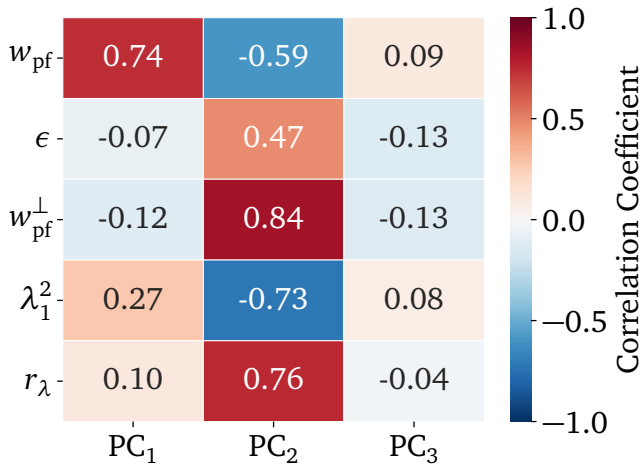
$$n_{\text{pf}} = \sum_i 1 \qquad w_{\text{pf}} = \frac{\sum_i p_{T,i} \Delta R_{i,\text{jet}}}{p_{T,\text{jet}}} \qquad p_T D = \frac{\sqrt{\sum_i p_{T,i}^2}}{\sum_i p_{T,i}} \qquad C_\beta = \frac{\sum_{i<j} p_{T,i} p_{T,j} (\Delta R_{ij})^\beta}{\left(\sum_i p_{T,i}\right)^2}$$

- related to max 5 principle components? [linear decorrelation]
- PC_1: constituent number and diversity

$$n_{\text{pf}} + \alpha \cdot S_{\text{PID}} \qquad \text{with} \qquad S_{\text{PID}} = -\sum_{\text{type}j} f_j \log f_j$$

- PC_2: radial energy profile

$$w_{\text{pf}}^\perp = \alpha \cdot n_{\text{pf}} - w_{\text{pf}} \qquad \text{and} \qquad r_\lambda = \frac{\lambda_{0.5}^1}{\lambda_1^2} \qquad \lambda_k^\beta = \sum_i z_i^\beta \Delta R^k$$



Physics from latent representation

Quarks vs gluons from trained ParticleNet [Vent, Winterhalder, TP]

- sensitive substructure variables

$$n_{\text{pf}} = \sum_i 1 \quad w_{\text{pf}} = \frac{\sum_i p_{T,i} \Delta R_{i,\text{jet}}}{p_{T,\text{jet}}} \quad p_T D = \frac{\sqrt{\sum_i p_{T,i}^2}}{\sum_i p_{T,i}} \quad C_\beta = \frac{\sum_{i<j} p_{T,i} p_{T,j} (\Delta R_{ij})^\beta}{\left(\sum_i p_{T,i}\right)^2}$$

- related to max 5 principle components? [linear decorrelation]
- PC_1: constituent number and diversity

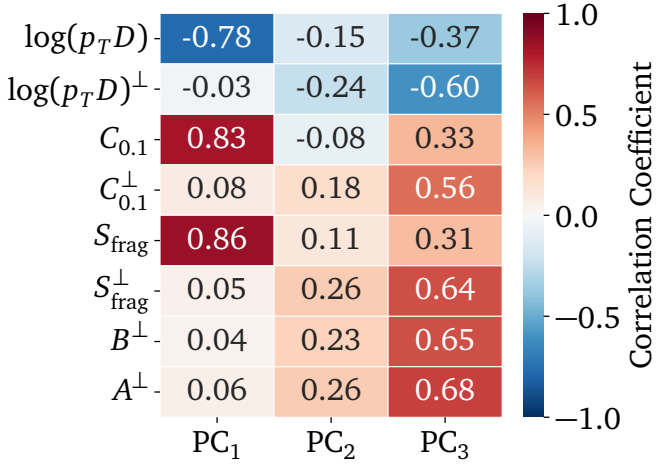
$$n_{\text{pf}} + \alpha \cdot S_{\text{PID}} \quad \text{with} \quad S_{\text{PID}} = -\sum_{\text{type}j} f_j \log f_j$$

- PC_2: radial energy profile

$$w_{\text{pf}}^\perp = \alpha \cdot n_{\text{pf}} - w_{\text{pf}} \quad \text{and} \quad r_\lambda = \frac{\lambda_{0.5}^1}{\lambda_1^2} \quad \lambda_k^\beta = \sum_i z_i^\beta \Delta R^k$$

- PC_3: fragmentation and energy dispersion

$$S_{\text{frag}} = -\sum_i z_i \log z_i$$



Physics from latent representation

Quarks vs gluons from trained ParticleNet [Vent, Winterhalder, TP]

- sensitive substructure variables

$$n_{\text{pf}} = \sum_i 1 \quad w_{\text{pf}} = \frac{\sum_i p_{T,i} \Delta R_{i,\text{jet}}}{p_{T,\text{jet}}} \quad p_T D = \frac{\sqrt{\sum_i p_{T,i}^2}}{\sum_i p_{T,i}} \quad C_\beta = \frac{\sum_{i<j} p_{T,i} p_{T,j} (\Delta R_{ij})^\beta}{\left(\sum_i p_{T,i}\right)^2}$$

- related to max 5 principle components? [linear decorrelation]
- PC_1: constituent number and diversity

$$n_{\text{pf}} + \alpha \cdot S_{\text{PID}} \qquad \text{with} \qquad S_{\text{PID}} = -\sum_{\text{type}j} f_j \log f_j$$

- PC_2: radial energy profile

$$w_{\text{pf}}^\perp = \alpha \cdot n_{\text{pf}} - w_{\text{pf}} \qquad \text{and} \qquad r_\lambda = \frac{\lambda_{0.5}^1}{\lambda_1^2} \qquad \lambda_k^\beta = \sum_i z_i^\beta \Delta R^k$$

- PC_3: fragmentation and energy dispersion

$$S_{\text{frag}} = -\sum_i z_i \log z_i$$

- $\text{PC}_{4,5}$: charge information etc

$$E_Q = \frac{E_{\text{charged}}}{E_{\text{jet}}} \qquad \text{and} \qquad A^\perp = S_{\text{frag}} \frac{C_{0.1}}{C_{0.05}} - 0.03 \cdot n_{\text{pf}} + 1.95 w_{\text{pf}}^\perp$$

$\rightarrow$ Latent distributions learn physics

ParticleNet beyond PCA

Disentangled latent classifier

- learning compressed, decorrelated representation

$$\mathcal{L} = \underbrace{\sum_{i=1}^{N} |x_i - \hat{x}_i|^2}_{\mathcal{L}_\text{reco}} + \underbrace{\sum_{i=1}^{N} \Big[y_i \log \sigma(z_i) + (1-y_i) \log(1-\sigma(z_i)) \Big]}_{\mathcal{L}_\text{class}} + \underbrace{\sum_{j\neq k} \Big[\text{Cov}(z_j, z_k) \Big]^2}_{\mathcal{L}_\text{disentangle}}$$

$\rightarrow$ 5 latent dimensions plenty

Latent Dim	1	2	3	4
AUC	0.893(2)	0.9001(4)	0.9024(4)	0.9034(2)
$\text{rej}_{30\%}$	72(3)	77(3)	´ 95(5)	95(3)
ΔC	1.8(3)	0.93(5)	1.0(16)	0.9(15)

ParticleNet beyond PCA

Disentangled latent classifier

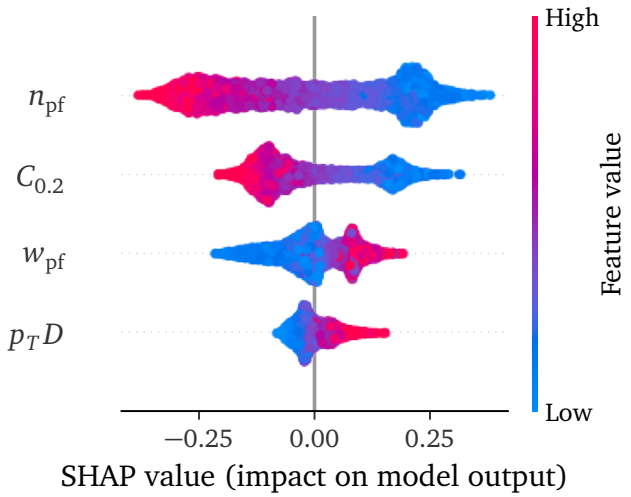
- learning compressed, decorrelated representation

$$\mathcal{L} = \underbrace{\sum_{i=1}^{N} |x_i - \hat{x}_i|^2}_{\mathcal{L}_\text{reco}} + \underbrace{\sum_{i=1}^{N} \Big[y_i \log \sigma(z_i) + (1-y_i) \log(1-\sigma(z_i)) \Big]}_{\mathcal{L}_\text{class}} + \underbrace{\sum_{j\neq k} \Big[\text{Cov}(z_j, z_k) \Big]^2}_{\mathcal{L}_\text{disentangle}}$$

$\rightarrow$ 5 latent dimensions plenty

Shapley variables

- pretty pictures, with weird patterns [quark



ParticleNet beyond PCA

Disentangled latent classifier

- learning compressed, decorrelated representation

$$\mathcal{L} = \underbrace{\sum_{i=1}^{N} |x_i - \hat{x}_i|^2}_{\mathcal{L}_\text{reco}} + \underbrace{\sum_{i=1}^{N} \Big[y_i \log \sigma(z_i) + (1-y_i) \log(1-\sigma(z_i)) \Big]}_{\mathcal{L}_\text{class}} + \underbrace{\sum_{j \neq k} \Big[\text{Cov}(z_j, z_k) \Big]^2}_{\mathcal{L}_\text{disentangle}}$$

$\rightarrow$ 5 latent dimensions plenty

Shapley variables

- pretty pictures, with weird patterns [quark jets: low multiplicity and low girth]

$\rightarrow$ Little insight from SHAP implementation

ParticleNet beyond PCA

Disentangled latent classifier

- learning compressed, decorrelated representation

$$\mathcal{L} = \underbrace{\sum_{i=1}^{N} |x_i - \hat{x}_i|^2}_{\mathcal{L}_\text{reco}} + \underbrace{\sum_{i=1}^{N} \Big[y_i \log \sigma(z_i) + (1-y_i) \log(1-\sigma(z_i)) \Big]}_{\mathcal{L}_\text{class}} + \underbrace{\sum_{j \neq k} \Big[\text{Cov}(z_j, z_k) \Big]^2}_{\mathcal{L}_\text{disentangle}}$$

$\rightarrow$ 5 latent dimensions plenty

Shapley variables

- pretty pictures, with weird patterns [quark jets: low multiplicity and low girth]

$\rightarrow$ Little insight from SHAP implementation

Symbolic Regression

- learn formula with given complexity
- classifier output not power series
- AUC and calibration double goal

$$p_\text{quark} = \tanh^3 \Big[0.55 \cdot C_{0.2} + 2 \left(-0.02 \cdot r_\lambda \cdot \left(C_{0.2} \cdot p_T D \cdot S_\text{PID} \cdot S_\text{frag} - 0.25 \right) + 1 \right)^3 \Big]$$

observables	model	AUC	$\text{Rej}_{30\%}$
$(n_\text{pf}, p_T D, C_{0.2}, r_\lambda, S_\text{PID}, S_\text{frag}, E_Q)$	MLP	0.872	66.87
	PySR	0.871	66.58

ParticleNet beyond PCA

Disentangled latent classifier

- learning compressed, decorrelated representation

$$\mathcal{L} = \underbrace{\sum_{i=1}^{N} |x_i - \hat{x}_i|^2}_{\mathcal{L}_\text{reco}} + \underbrace{\sum_{i=1}^{N} \Big[y_i \log \sigma(z_i) + (1-y_i) \log(1-\sigma(z_i)) \Big]}_{\mathcal{L}_\text{class}} + \underbrace{\sum_{j \neq k} \Big[\text{Cov}(z_j, z_k) \Big]^2}_{\mathcal{L}_\text{disentangle}}$$

→ 5 latent dimensions plenty

Shapley variables

- pretty pictures, with weird patterns [quark jets: low multiplicity and low girth]

→ Little insight from SHAP implementation

Symbolic Regression

- learn formula with given complexity
- classifier output not power series
- AUC and calibration double goal

$$p_\text{quark} = \tanh^3 \Big[0.55 \cdot C_{0.2} + 2 \left(-0.02 \cdot r_\lambda \cdot \left(C_{0.2} \cdot p_T D \cdot S_\text{PID} \cdot S_\text{frag} - 0.25 \right) + 1 \right)^3 \Big]$$

→ Formulas as physics regularizers? [Bahl, Fuchs, Menem, TP]

AI for fundamental physics

Develop AI for the best science

1 just another tool for a numerical field

2 transformative new language

· many applications in LHC theory

MadGraph7
MLhad
NNLO
Simulation-based inference
Unfolding
SFitter global analyses

→ Make complexity our friend

→ Remember we still do LHC theory

Modern Machine Learning for LHC Physicists

Tilman Plehn[a*], Anja Butter[a,b], Barry Dillon[a],
Theo Heimel[a], Claudius Krause[c], and Ramon Winterhalder[d]

[a] Institut für Theoretische Physik, Universität Heidelberg, Germany
[b] LPNHE, Sorbonne Université, Université Paris Cité, CNRS/IN2P3, Paris, France
[c] HEPHY, Austrian Academy of Sciences. Vienna, Austria
[d] CP3, Université catholique de Louvain, Louvain-la-Neuve, Belgium

March 19, 2024

Abstract

Modern machine learning is transforming particle physics fast, bullying its way into our numerical tool box. For young researchers it is crucial to stay on top of this development, which means applying cutting-edge methods and tools to the full range of LHC physics problems. These lecture notes lead students with basic knowledge of particle physics and significant enthusiasm for machine learning to relevant applications. They start with an LHC-specific motivation and a non-standard introduction to neural networks and then cover classification, unsupervised classification, generative networks, and inverse problems. Two themes defining much of the discussion are well-defined loss functions and uncertainty-aware networks. As part of the applications, the notes include some aspects of theoretical LHC physics. All examples are chosen from particle physics publications of the last few years.[1]