UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

ACAT 2024, 12.03.2024

Precision-Machine Learning for the Matrix Element Method

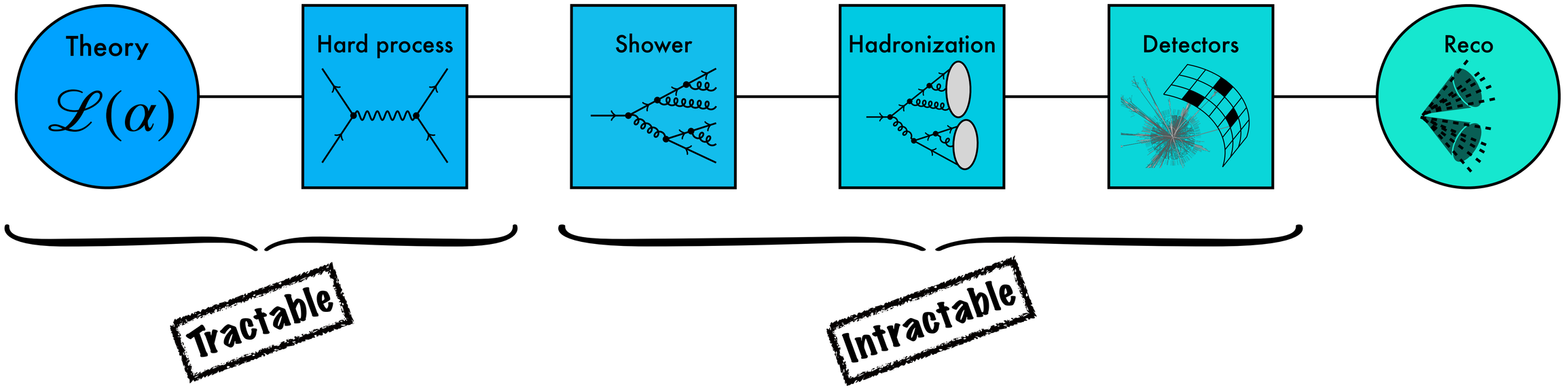
T. Heimel, **N. Huetsch**, R. Winterhalder, T. Plehn, A. Butter
arXiv: 2310.07752

Building on: A. Butter, T. Heimel, T. Martini, S. Peitzsch, T. Plehn: arXiv:2210.00019

SPONSORED BY THE

Federal Ministry
of Education
and Research

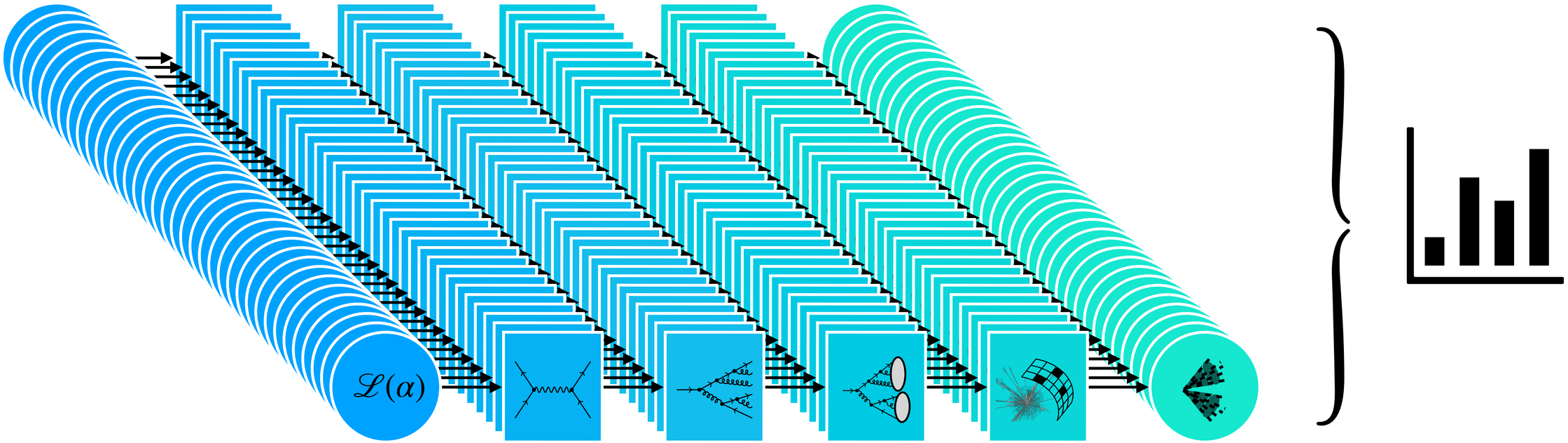
From Theory to Experiment in LHC Physics



In HEP we can never analytically calculate what we measure

We rely on simulations to connect theory and experiment

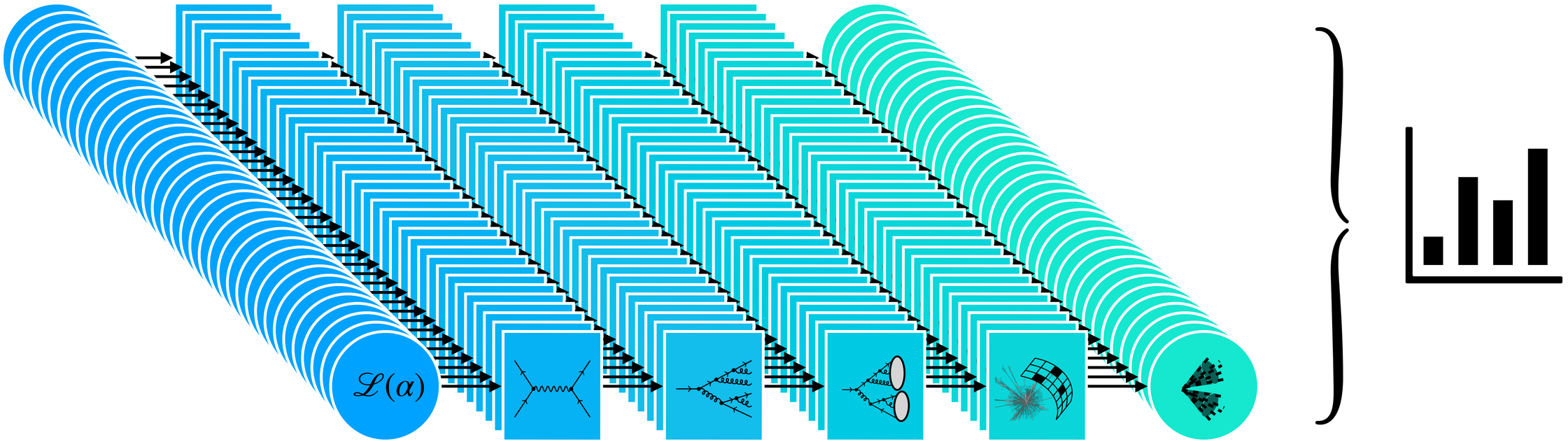
From Theory to Experiment in LHC Physics



Event samples are combined into observable histogram

Simulated and measured histograms are compared

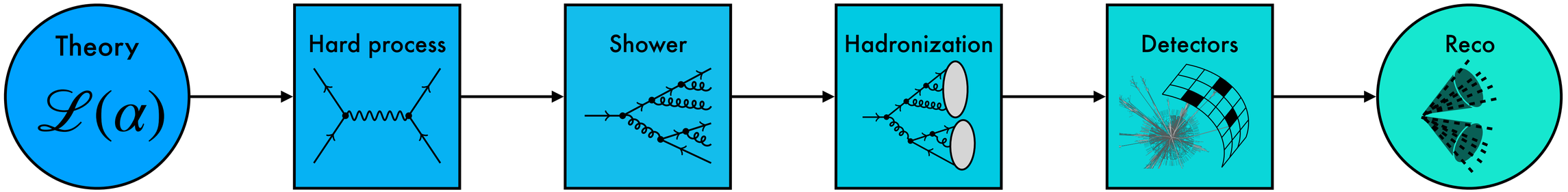
From Theory to Experiment in LHC Physics



We lose a lot of information doing this!

What if we could go for the likelihood instead?

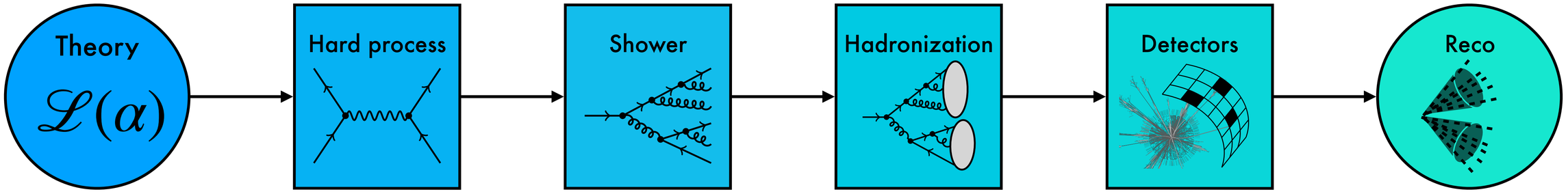
Going for the likelihood



$$p(x_{hard} \mid \alpha)$$

Hard-scattering level likelihood known via differential cross-section $p(x_{hard} \mid \alpha) = \frac{1}{\sigma(\alpha)} \frac{d\sigma(\alpha)}{dx_{hard}}$

Going for the likelihood

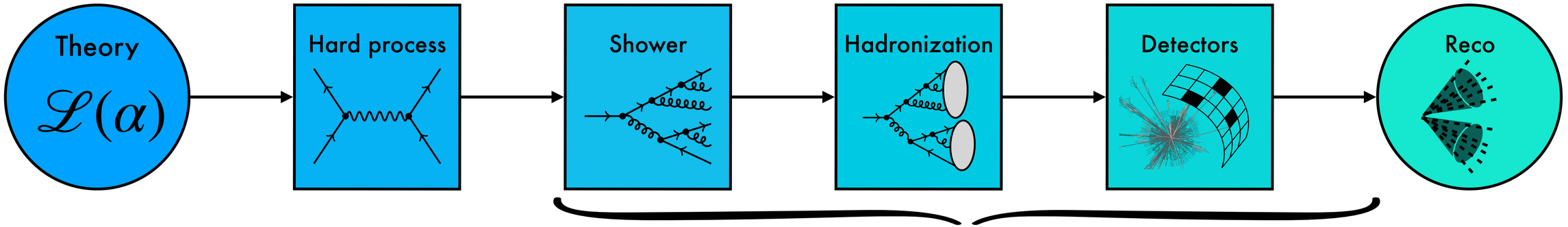


$p(x_{hard} \mid \alpha)$

$p(x_{reco} \mid \alpha)$

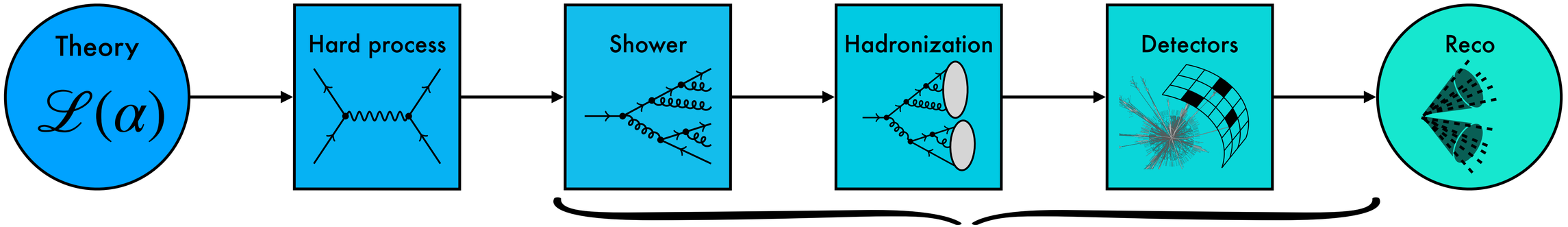
Can we access the likelihood at reconstruction level?

Going for the likelihood



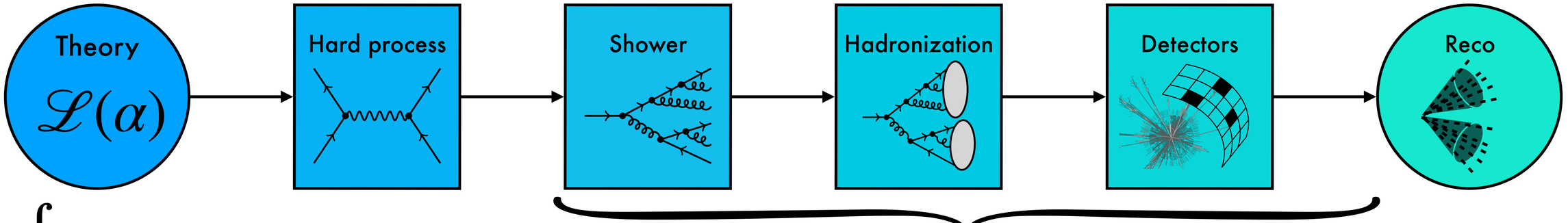
Hard-scattering and reconstruction linked by forward transfer probability

Going for the likelihood



Include an efficiency term to account for acceptance of events

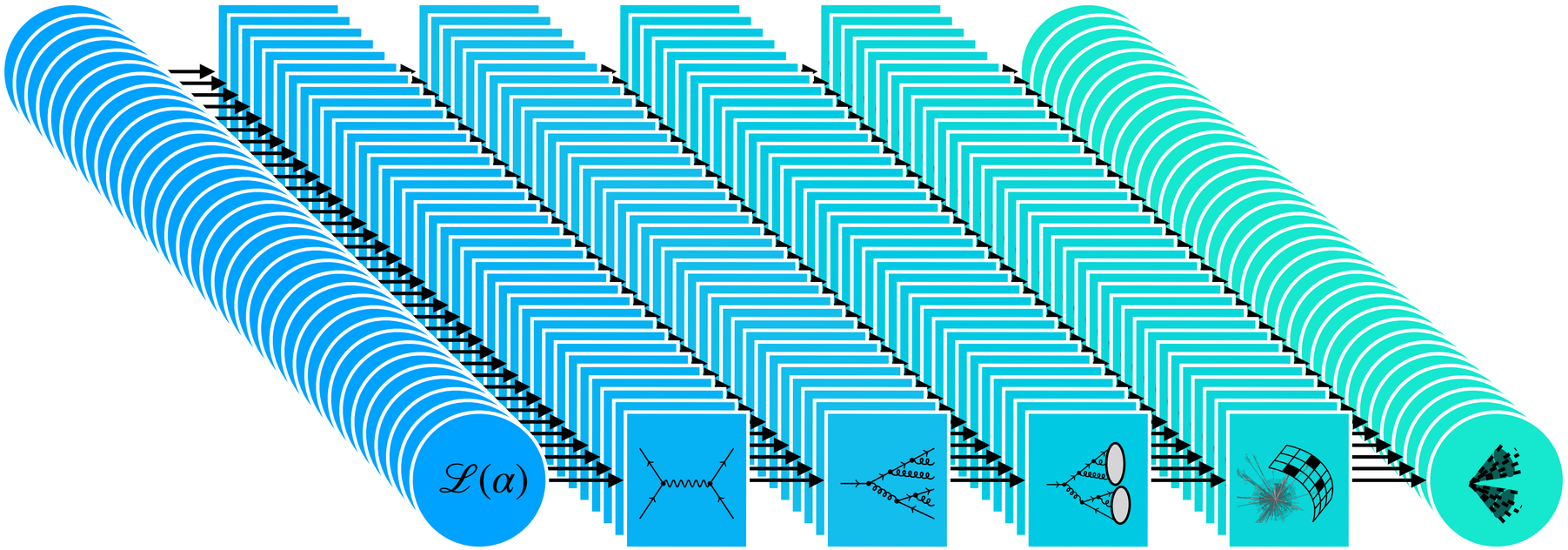
Going for the likelihood



$$\int dx_{hard} \quad p(x_{hard}\,|\,\alpha) \quad p(x_{reco}\,|\,x_{hard},\alpha)\,\epsilon(x_{hard},\alpha) \;=\; p(x_{reco}\,|\,\alpha)$$

Integrate over all possible hard-scattering configurations

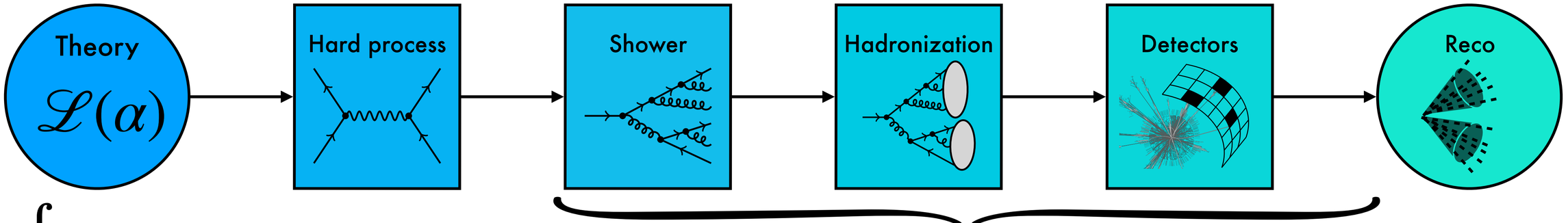
Going for the likelihood



$$\prod_{events\ i} p(x_{reco,i} \mid \alpha)$$

Event likelihoods are combined into sample likelihoods

The Matrix Element Method



$$\int dx_{hard} \quad p(x_{hard}\,|\,\alpha) \quad p(x_{reco}\,|\,x_{hard},\alpha)\,\epsilon(x_{hard},\alpha) \;=\; p(x_{reco}\,|\,\alpha)$$

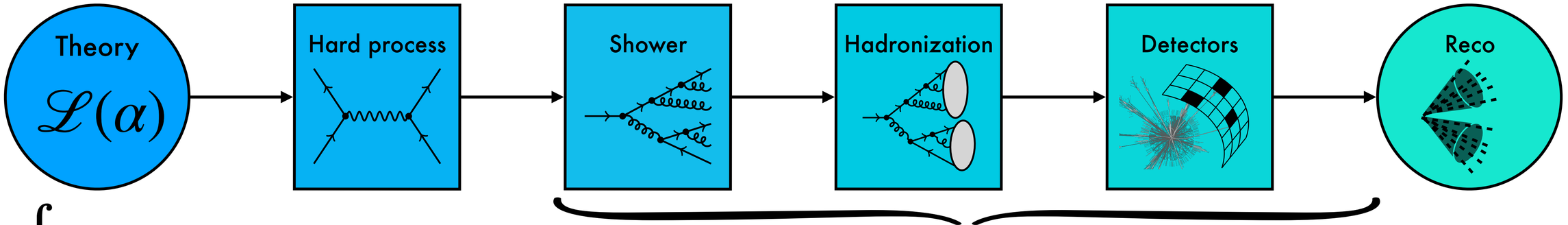
+++ Unbinned and multivariate by design

+++ Optimal use of information derived from Neyman-Pearson lemma

—— Transfer probability and efficiency not known

—— Integral numerically very challenging

The Matrix Element Method



$$\int dx_{hard} \quad p(x_{hard}\,|\,\alpha) \quad p(x_{reco}\,|\,x_{hard},\alpha)\,\epsilon(x_{hard},\alpha) \;=\; p(x_{reco}\,|\,\alpha)$$

+++ Unbinned and multivariate by design

+++ Optimal use of information derived from Neyman-Pearson lemma

—— ~~Transfer probability and efficiency not known~~ USE MACHINE LEARNING

—— ~~Integral numerically very challenging~~ USE MACHINE LEARNING

The Transfer Network

Intractable

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard},\alpha)\; \epsilon(x_{hard},\alpha) \;=\; p(x_{reco}|\alpha)$$

The Transfer Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard},\alpha) \;=\; p(x_{reco}|\alpha)$$

Intractable

The Transfer Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard},\alpha) \;=\; p(x_{reco}|\alpha)$$

Intractable

Transfer probability is analytically intractable

Transfer can be simulated to generate paired data x_{hard} , x_{reco}

$$x_{hard} \xrightarrow{\text{Simulation}} x_{reco}$$

The Transfer Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard},\alpha) \;=\; p(x_{reco}|\alpha)$$

Intractable

Transfer probability is analytically intractable

Transfer can be simulated to generate paired data x_{hard} , x_{reco}

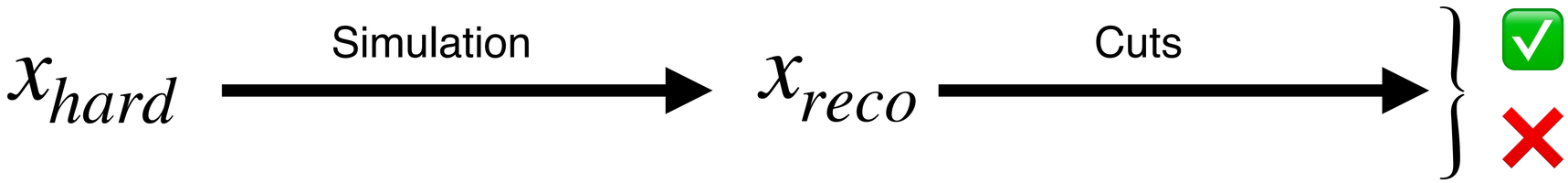
x_{hard} —Simulation→ x_{reco}

Conditional Generative Neural Network to encode the transfer probability

The Transfer Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard},\alpha) \;=\; p(x_{reco}|\alpha)$$

Intractable



The transfer network is trained only on accepted pairs

Bias in the learned transfer probability $p(x_{reco}|x_{hard})$ that we need to correct!

The Acceptance Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}, \alpha) = p(x_{reco}|\alpha)$$

Unknown

The Acceptance Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) = p(x_{reco}|\alpha)$$

Unknown

The Acceptance Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) = p(x_{reco}|\alpha)$$

Unknown

Need to encode the efficiency at hard-scattering level

Transfer can be simulated to generate labeled data $x_{hard} \rightarrow x_{reco}(x_{hard}) \rightarrow$ ✅ ❌

The Acceptance Network

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) = p(x_{reco}|\alpha)$$

Unknown

Need to encode the efficiency at hard-scattering level

Transfer can be simulated to generate labeled data $x_{hard} \rightarrow x_{reco}(x_{hard}) \rightarrow$ ✅ / ❌

Classifier Neural Network to encode the acceptance probability

The Sampling Network

Challenging

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \;=\; p(x_{reco}|\alpha)$$

The Sampling Network

Challenging

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \;=\; p(x_{reco}|\alpha)$$

$$= \left\langle \frac{1}{q(x_{hard}|x_{reco},\alpha)} p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \right\rangle_{x_{hard}\sim q}$$

The Sampling Network

Challenging

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \;=\; p(x_{reco}|\alpha)$$

$$= \left\langle \frac{1}{q(x_{hard}|x_{reco},\alpha)} p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \right\rangle_{x_{hard}\sim q}$$

Integral becomes trivial if : $q(x_{hard}|x_{reco},\alpha) \propto p(x_{hard}|x_{reco},\alpha)\epsilon(x_{hard})$

The Sampling Network

Challenging

$$\int dx_{hard}\; p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \;=\; p(x_{reco}|\alpha)$$

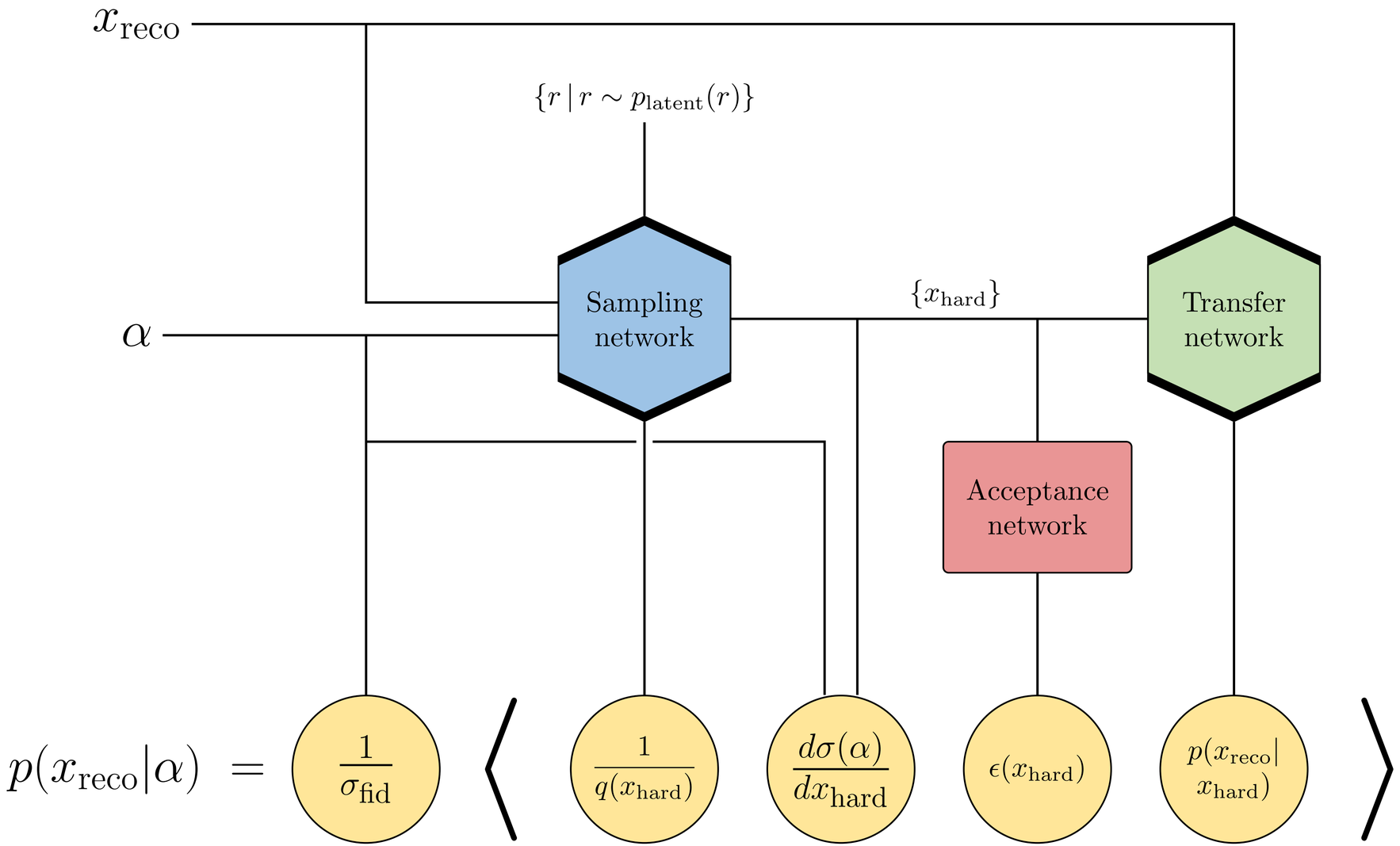
$$= \left\langle \frac{1}{q(x_{hard}|x_{reco},\alpha)} p(x_{hard}|\alpha)\; p(x_{reco}|x_{hard})\; \epsilon(x_{hard}) \right\rangle_{x_{hard}\sim q}$$

Integral becomes trivial if : $q(x_{hard}|x_{reco},\alpha) \propto p(x_{hard}|x_{reco},\alpha)\epsilon(x_{hard})$

Generative Neural Network to encode sampling distribution

$$r \sim p_{latent}(r) \longleftrightarrow x_{hard}(r) \sim q_\phi(x_{hard}|x_{reco},\alpha)$$

Machine-learned MEM



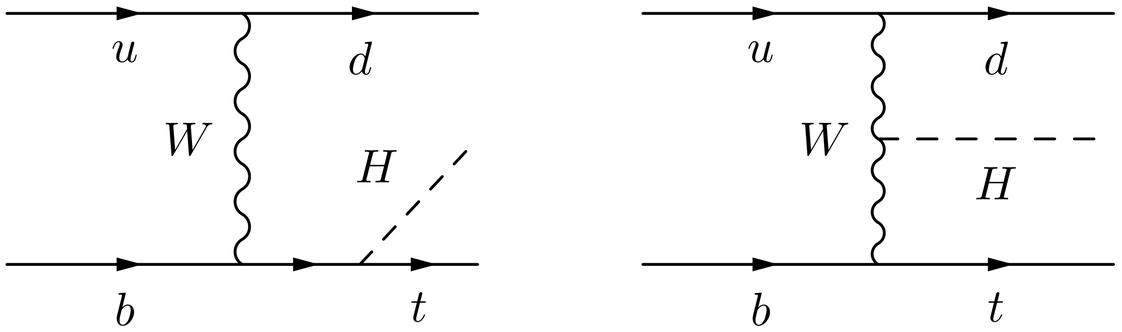
The physics example

The problem: Measuring a CP-phase in the top Yukawa coupling

$$\mathscr{L}_{t\bar{t}H}(\alpha) = -\frac{y_t}{\sqrt{2}}\left[\cos\alpha\bar{t}t + \frac{2}{3}i\sin\alpha\bar{t}\gamma_5 t\right]H$$

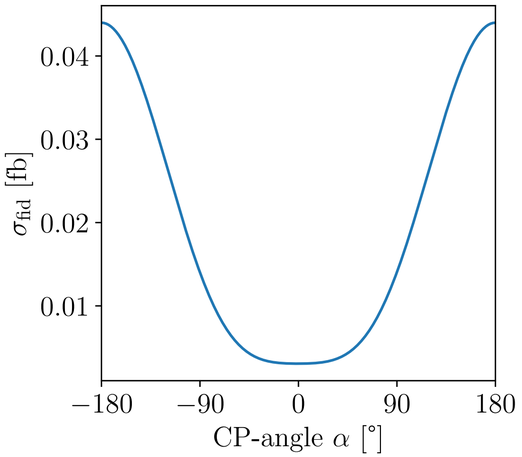
The process: Associated single-top and Higgs production

$$pp \to tHj \to (bjj)\,(\gamma\gamma)\,j \;+\; \text{ISR Jets}$$

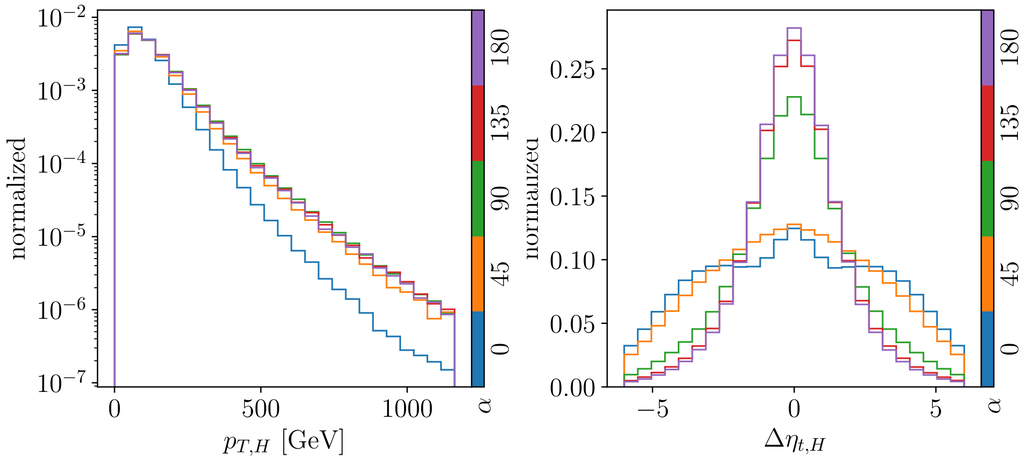


[Following Butter et al arXiv:2210.00019]

Why MEM here?



Cross-section very small and insensitive to variations in α



Hard-scattering $p(x_{hard} \mid \alpha)$ kinematics are sensitive

Ideal use case for the Matrix Element Method

[Figures taken from Butter et al arXiv:2210.00019]

The workflow

1) Start from Lagrangian $\mathscr{L}_{t\bar{t}H}(\alpha)$ and process $pp \to tHj \to (bjj)\,(\gamma\gamma)\,j$ + ISR Jets

2) Simulate hard-scattering events $p(x_{hard}\,|\,\alpha)$ for a range of parameter values

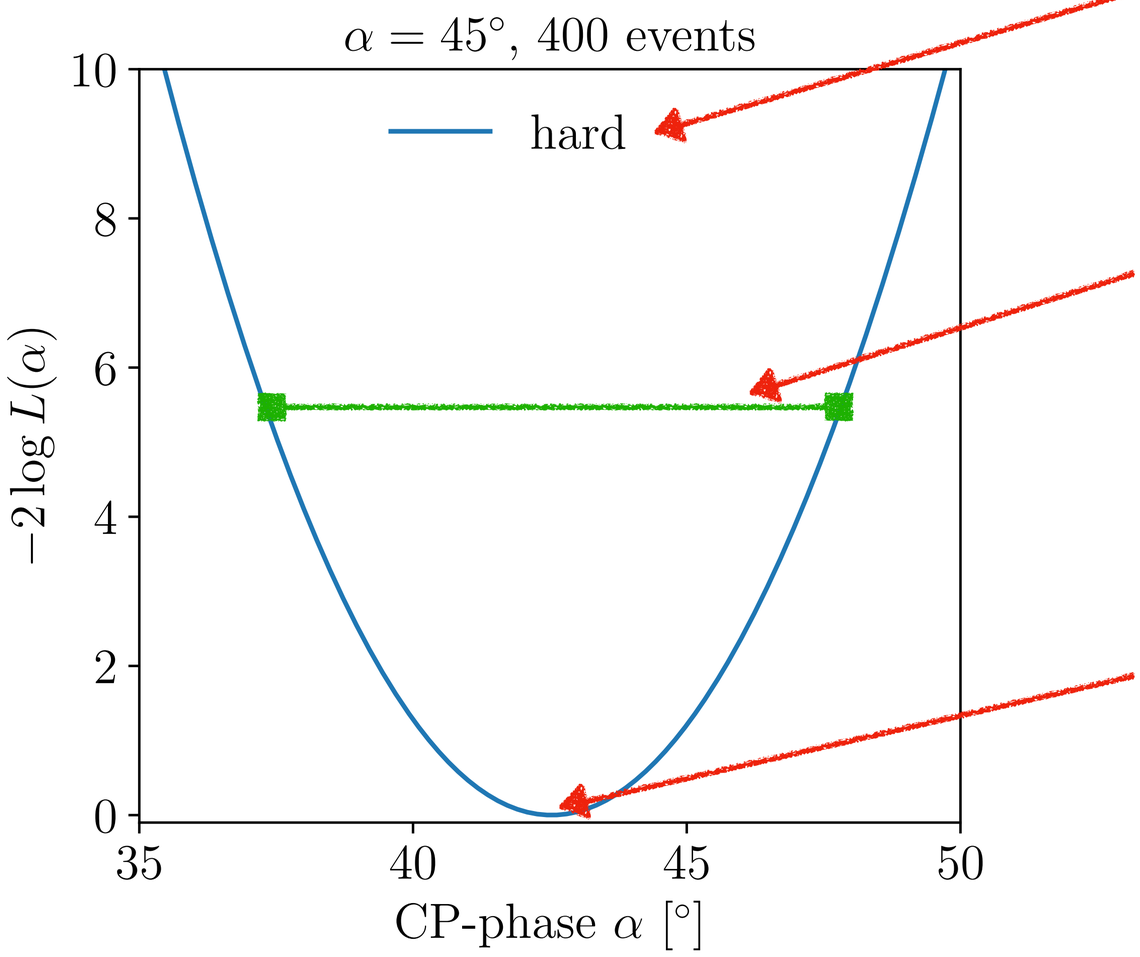
3) Simulate transfer $x_{hard} \to x_{reco}$ to obtain paired data $(\alpha, x_{hard}, x_{reco})$

4) Train the networks: transfer $p(x_{reco}\,|\,x_{hard})$, efficiency $\epsilon(x_{hard})$, sampling $q(x_{hard}\,|\,x_{reco}, \alpha)$

5) Solve the MEM integral for each data sample

$$p(x_{reco}\,|\,\alpha) = \left\langle \frac{1}{q(x_{hard}\,|\,x_{reco}, \alpha)} p(x_{hard}\,|\,\alpha)\; p(x_{reco}\,|\,x_{hard})\; \epsilon(x_{hard}) \right\rangle_{x_{hard}\sim q}$$

What do results look like?

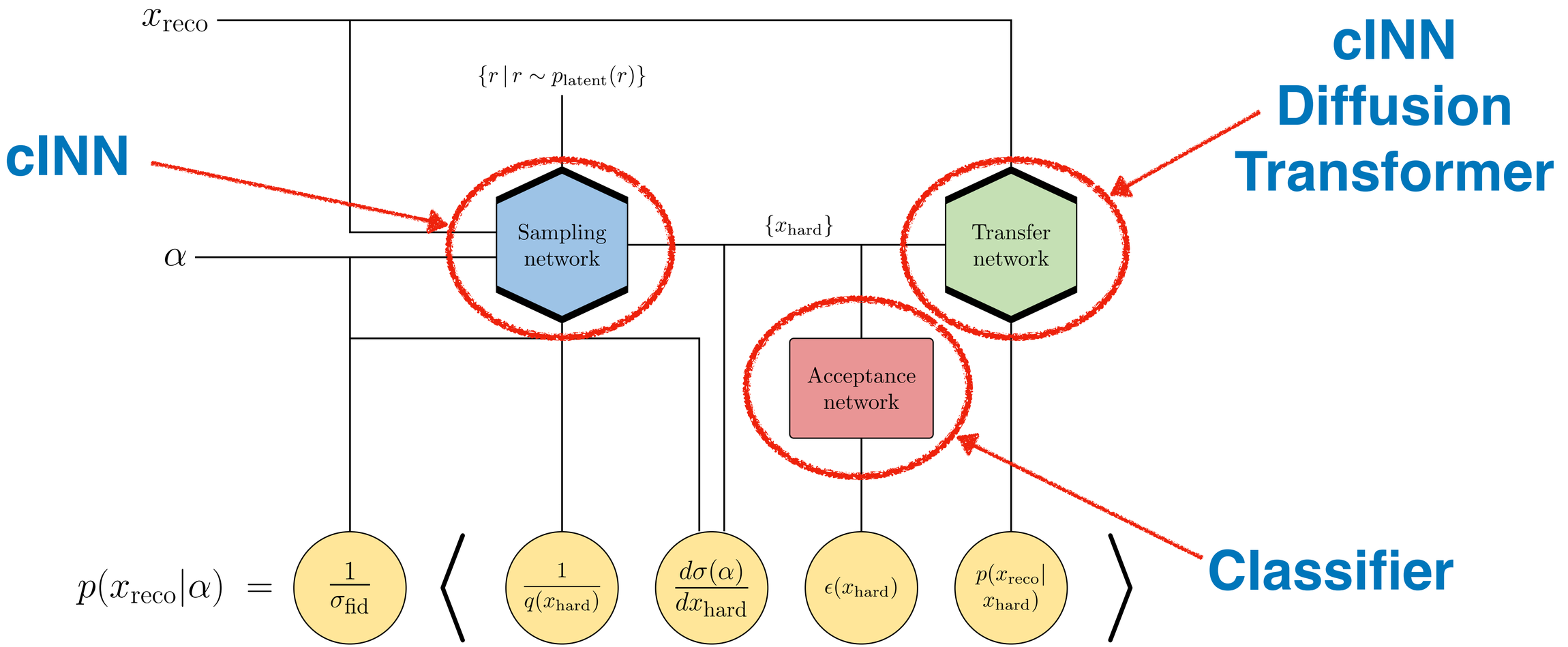


Hard-level likelihood as truth

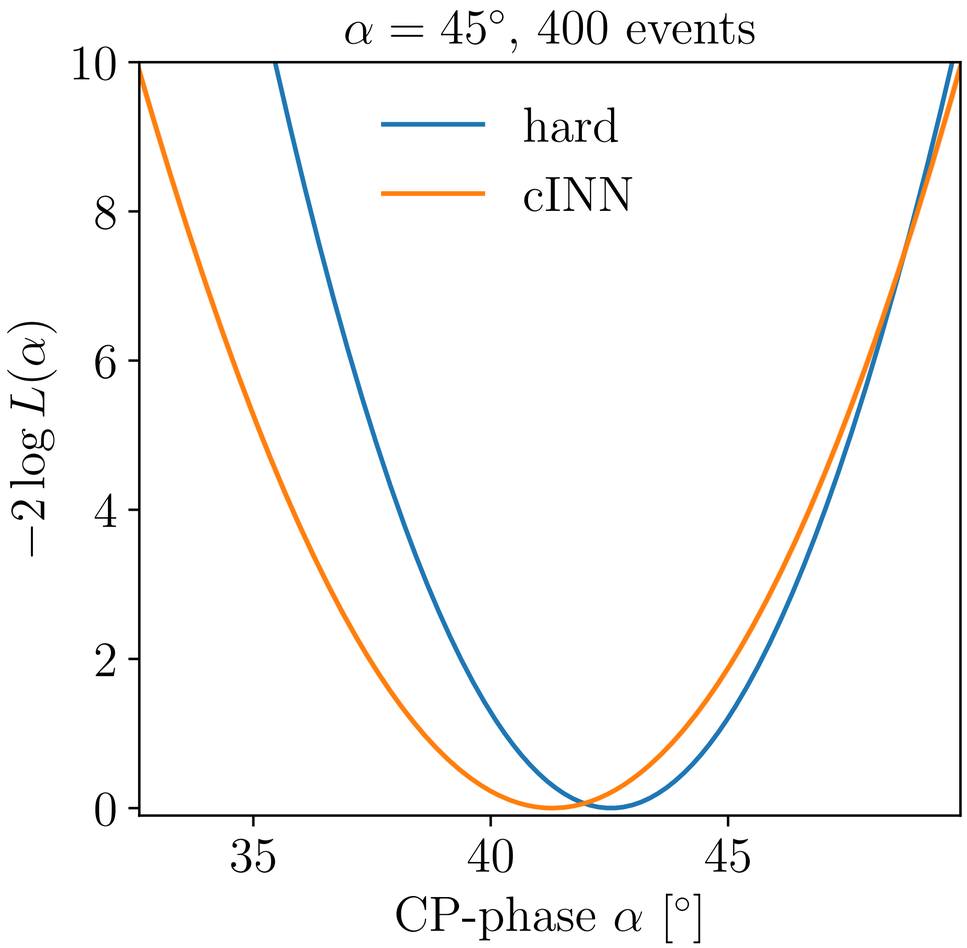
Width of the Parabola:
Uncertainty of the ML estimator

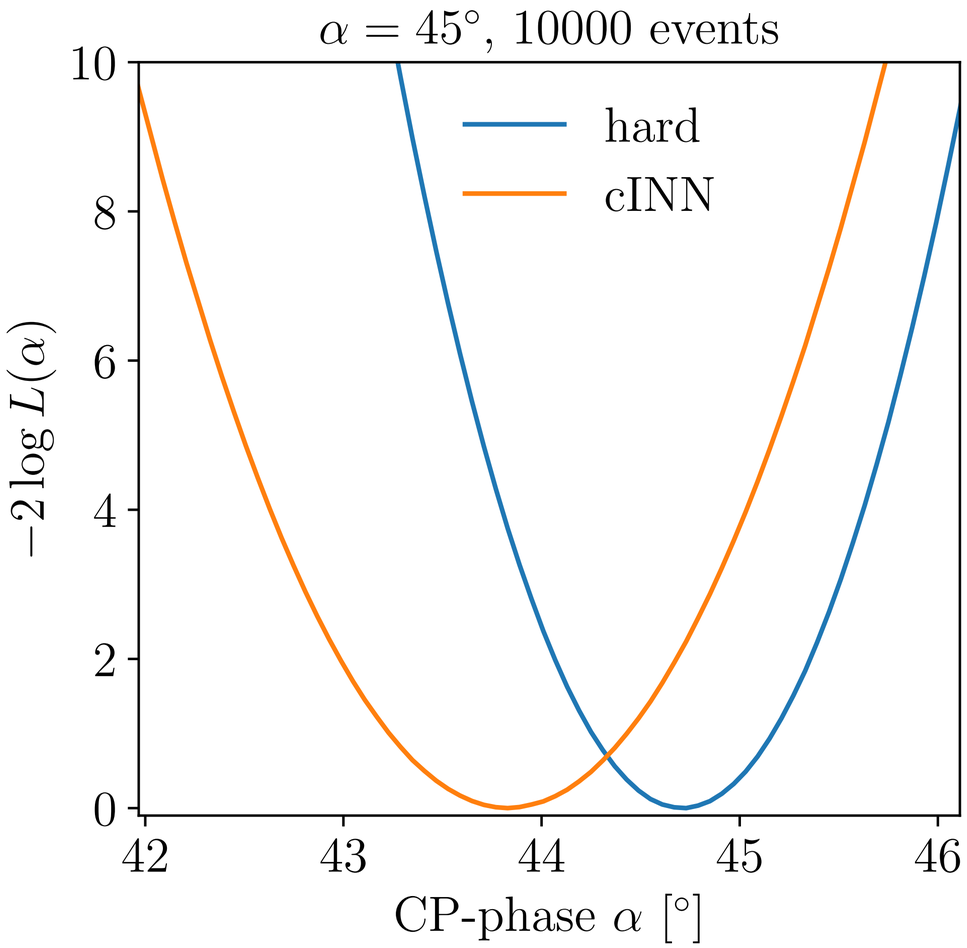
Position of the Minimum:
Maximum-Likelihood estimator of α

Machine-learned MEM

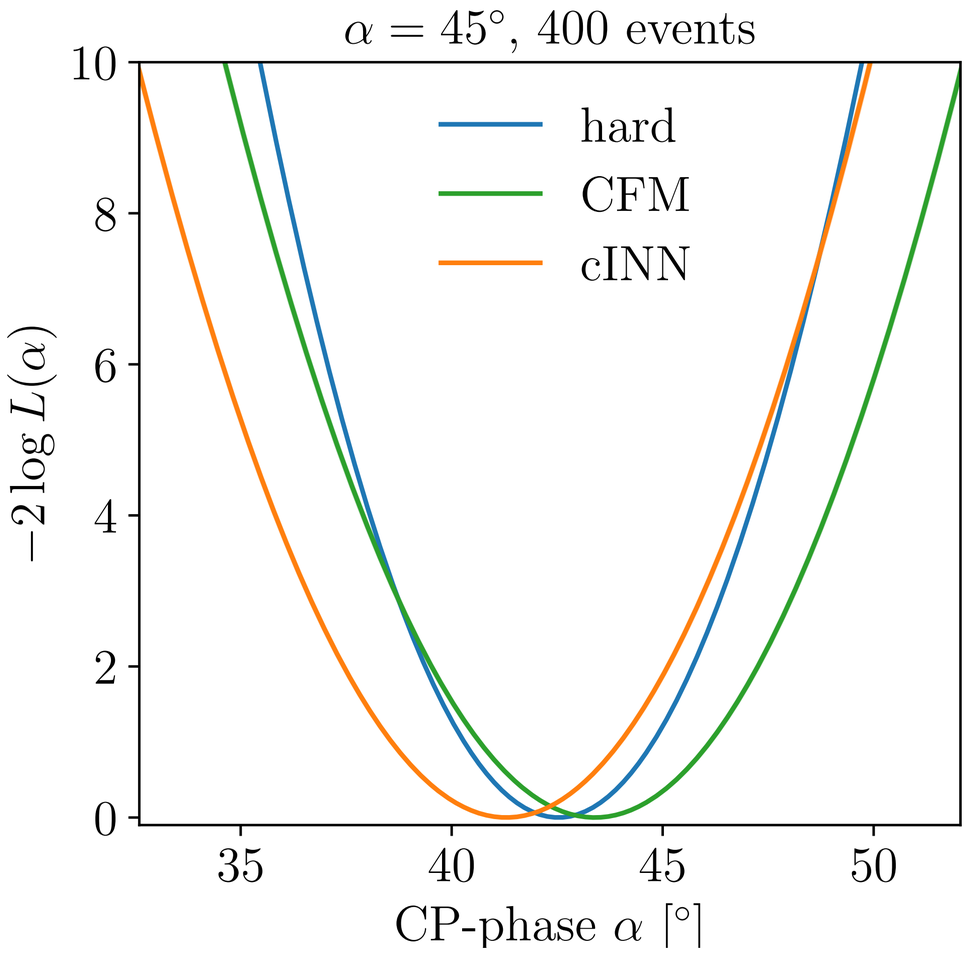


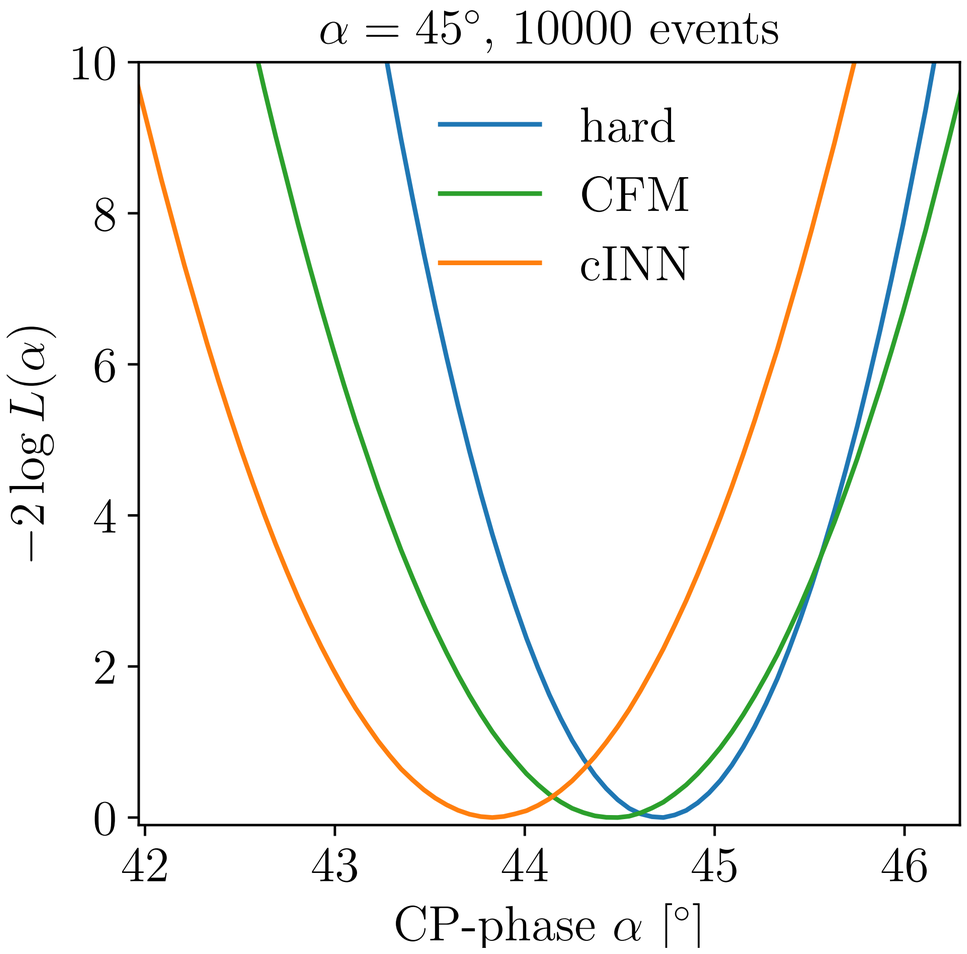
Transfer-cINN Results



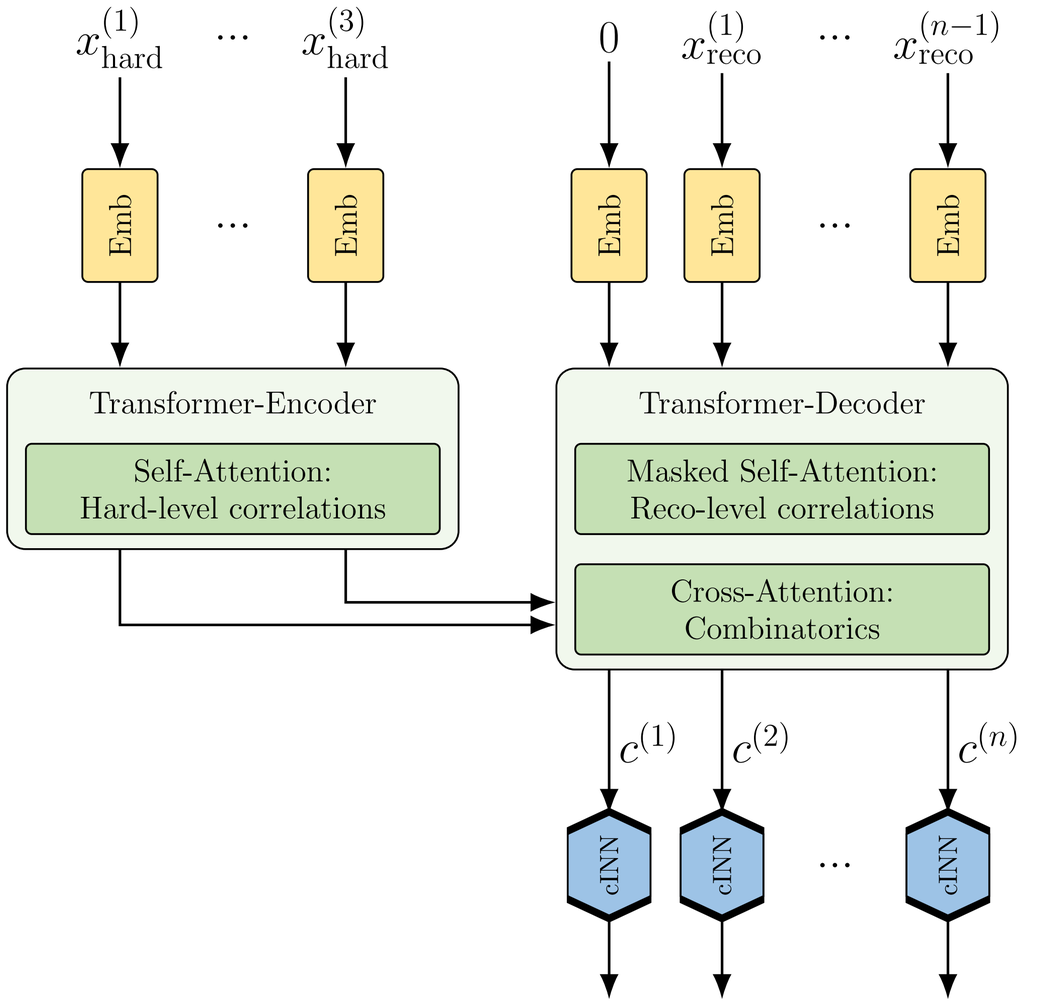


Transfer-Diffusion Results

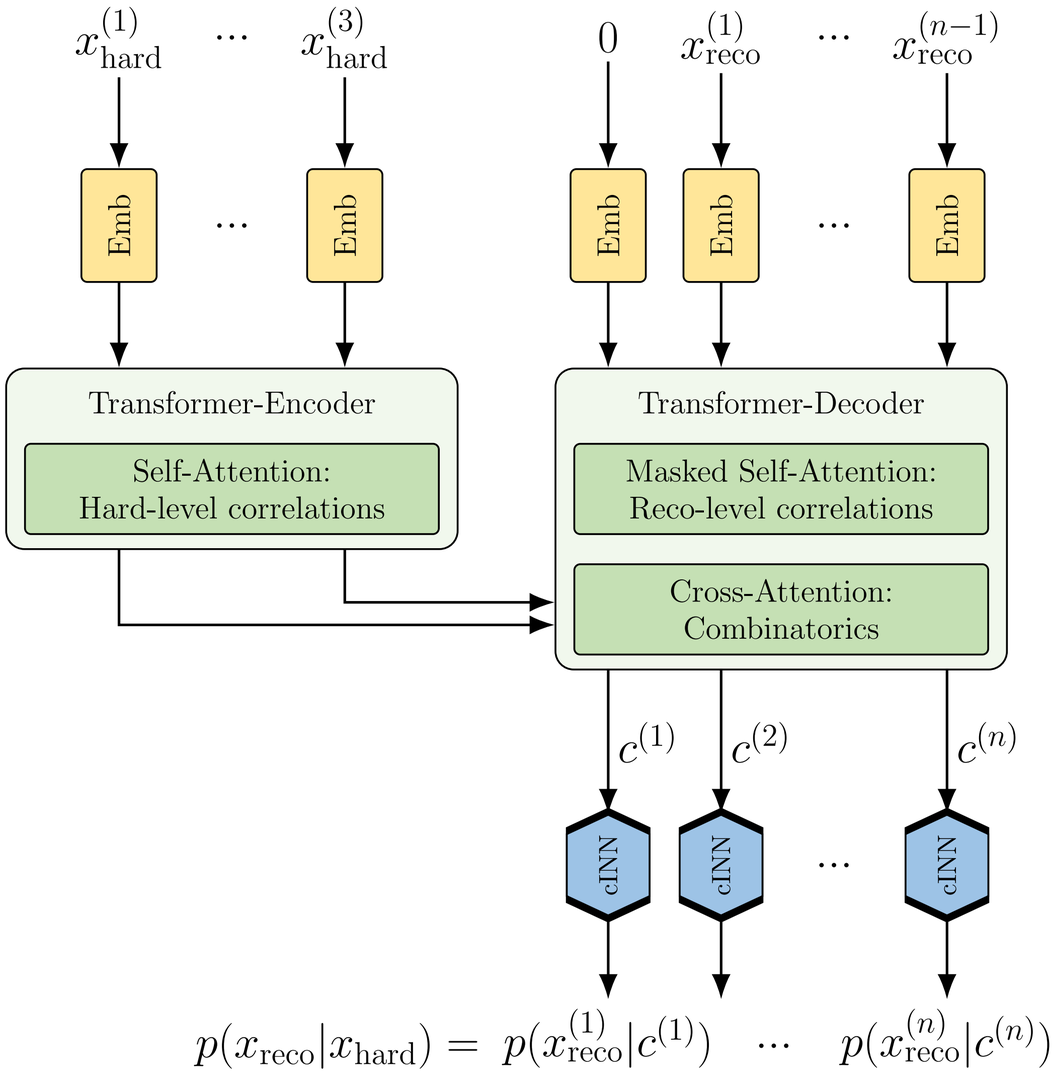




The Transfermer(Transfer-Transformer)



The Transfermer(Transfer-Transformer)



Why a Transformer here?

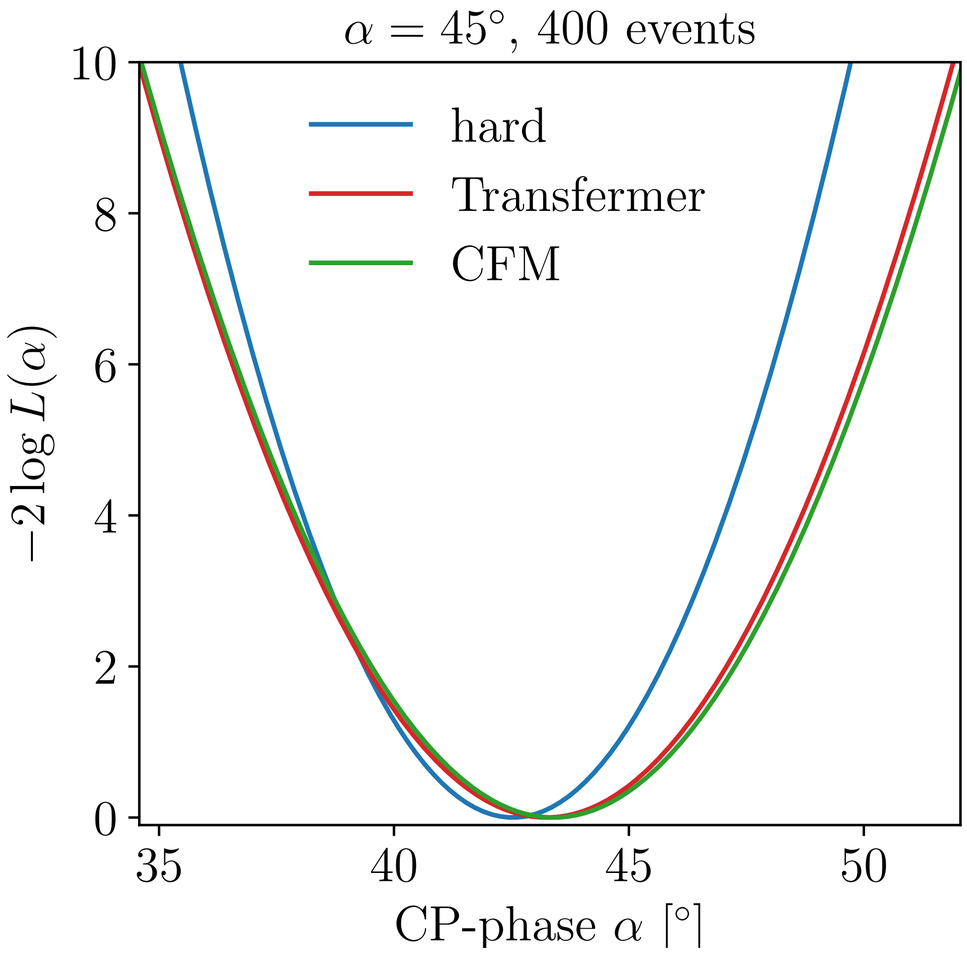
1. Jet combinatorics

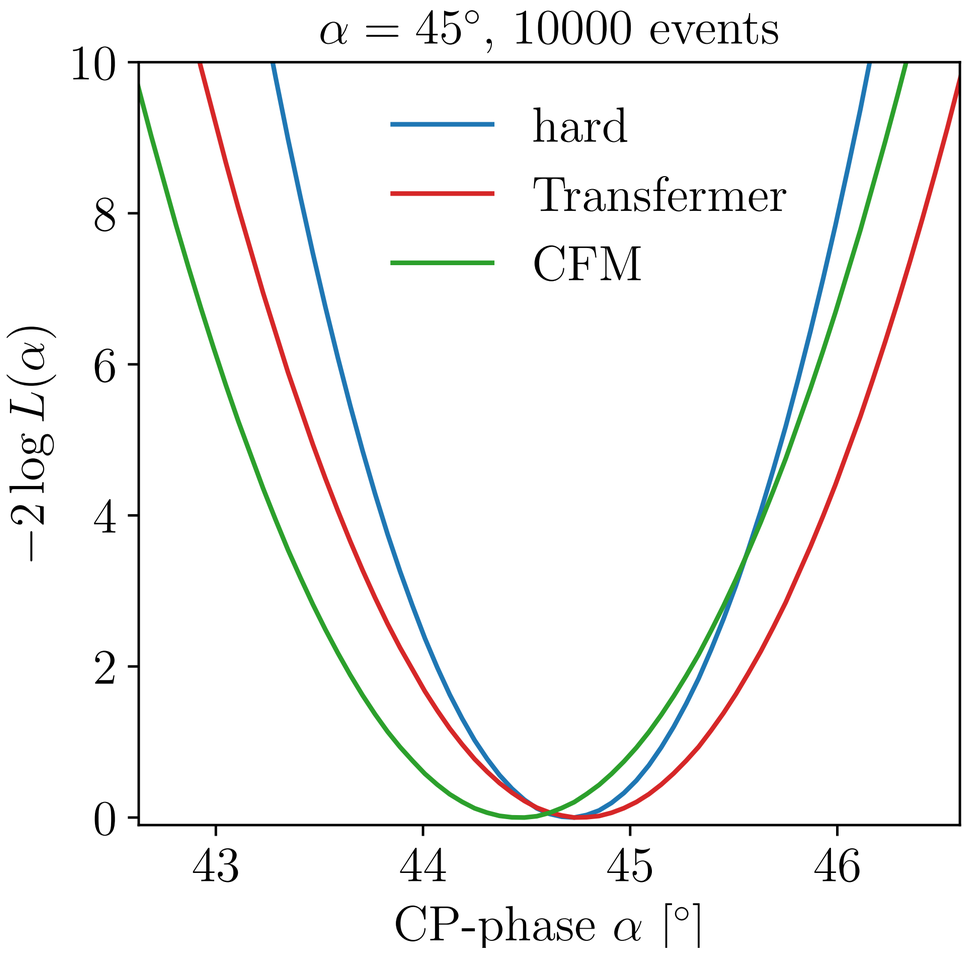
$$pp \to tHj \to (bjj)\ (\gamma\gamma)\ j\ +\ \text{ISR Jets}$$

2. Can handle varying particle multiplicity

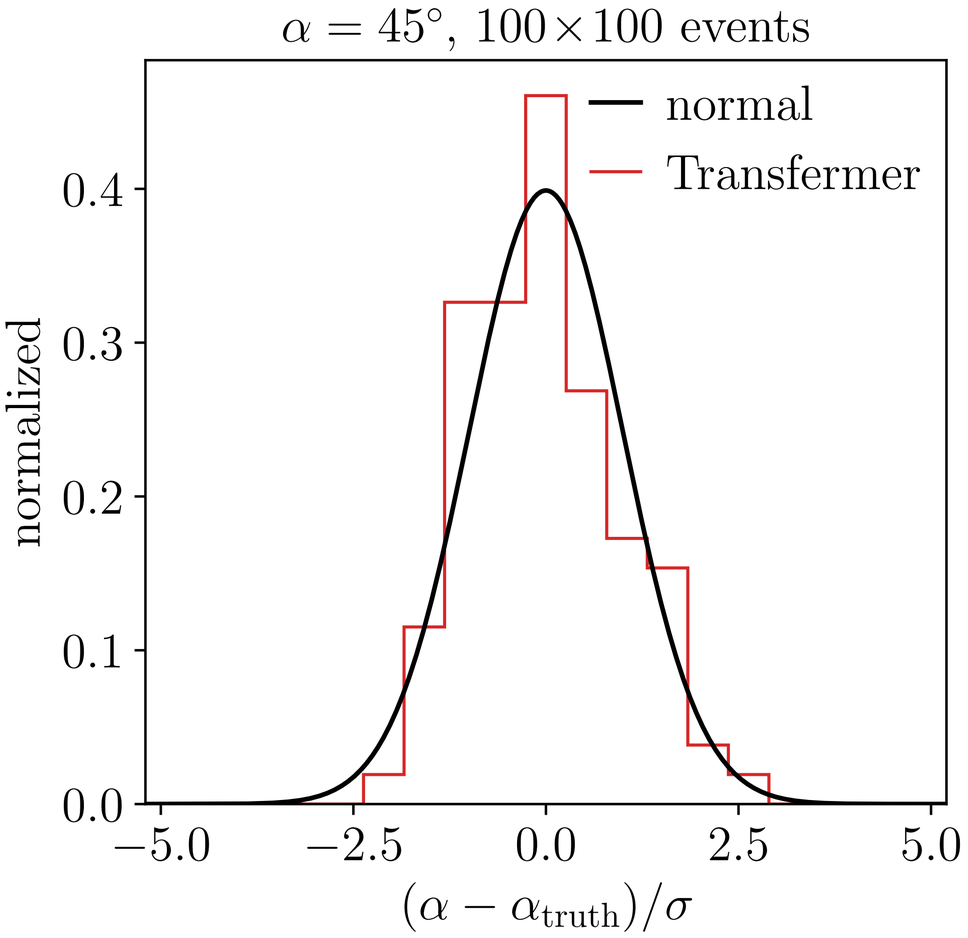
⟶ NLO

Transfermer Results





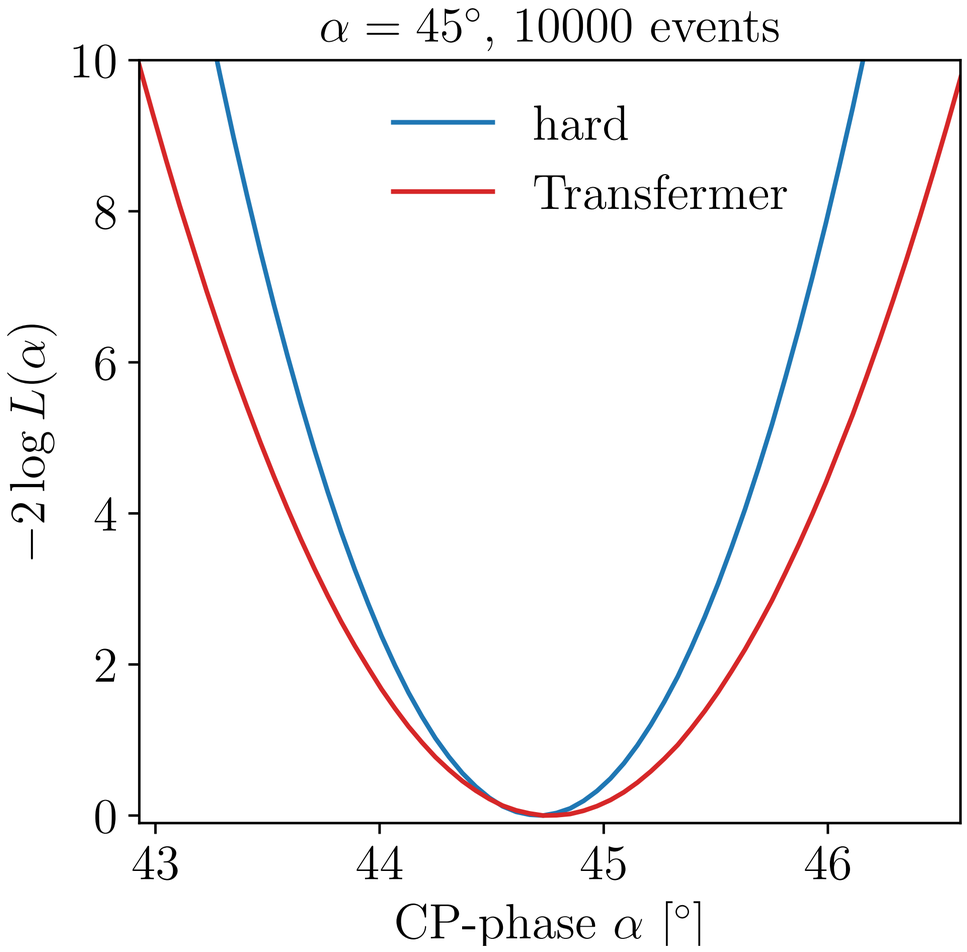
Estimating Calibration



Divide the 10k events into 100 samples of 100 events

Look at this distribution of the minima around the true α

Estimating Uncertainty



Estimate uncertainty using replicas:

Integration uncertainty:
Resample the MC points using bootstrapping

Network uncertainty:
Use Bayesian NN and resample for each replica

Summary and Outlook

SciPost Physics 2310.07752 Submission

Precision-Machine Learning for the Matrix Element Method

Theo Heimel[1], Nathan Huetsch[1], Ramon Winterhalder[2],
Tilman Plehn[1], and Anja Butter[1,3]

1 Institut für Theoretische Physik, Universität Heidelberg, Germany
2 CP3, Université catholique de Louvain, Louvain-la-Neuve, Belgium
3 LPNHE, Sorbonne Université, Université Paris Cité, CNRS/IN2P3, Paris, France

Summary and Outlook

SciPost Physics 2310.07752 Submission

Precision-Machine Learning for the Matrix Element Method

Theo Heimel[1], Nathan Huetsch[1], Ramon Winterhalder[2],
Tilman Plehn[1], and Anja Butter[1,3]

1 Institut für Theoretische Physik, Universität Heidelberg, Germany
2 CP3, Université catholique de Louvain, Louvain-la-Neuve, Belgium
3 LPNHE, Sorbonne Université, Université Paris Cité, CNRS/IN2P3, Paris, France

Three network setup makes the **MEM tractable**

1) **Transfer-Network** encoding the transfer probability $p(x_{reco} \mid x_{hard})$
2) **Acceptance-Network** encoding the efficiency $\epsilon(x_{hard})$
3) **Sampling-Network** encoding the proposal distribution $q(x_{hard})$

Pipeline gives **precise** and **well-calibrated** results

Bootstrapping and **Bayesian NNs** to estimate **uncertainties**

Summary and Outlook

SciPost Physics 2310.07752 Submission

Precision-Machine Learning for the Matrix Element Method

Theo Heimel[1], Nathan Huetsch[1], Ramon Winterhalder[2],
Tilman Plehn[1], and Anja Butter[1,3]

1 Institut für Theoretische Physik, Universität Heidelberg, Germany
2 CP3, Université catholique de Louvain, Louvain-la-Neuve, Belgium
3 LPNHE, Sorbonne Université, Université Paris Cité, CNRS/IN2P3, Paris, France

Three network setup makes the **MEM tractable**

1) **Transfer-Network** encoding the transfer probability $p(x_{reco} \,|\, x_{hard})$
2) **Acceptance-Network** encoding the efficiency $\epsilon(x_{hard})$
3) **Sampling-Network** encoding the proposal distribution $q(x_{hard})$

Method gives **precise** and **well-calibrated** results

Bootstrapping and **Bayesian NNs** to estimate **uncertainties**

NLO

Theory $\neq$ Practice

Improving integration convergence

$$\left\langle \frac{1}{q(x_{hard})} p(x_{hard}\,|\,\alpha)\; p(x_{reco}\,|\,x_{hard})\; \epsilon(x_{hard}) \right\rangle_{x_{hard}\sim q(x_{hard})}$$

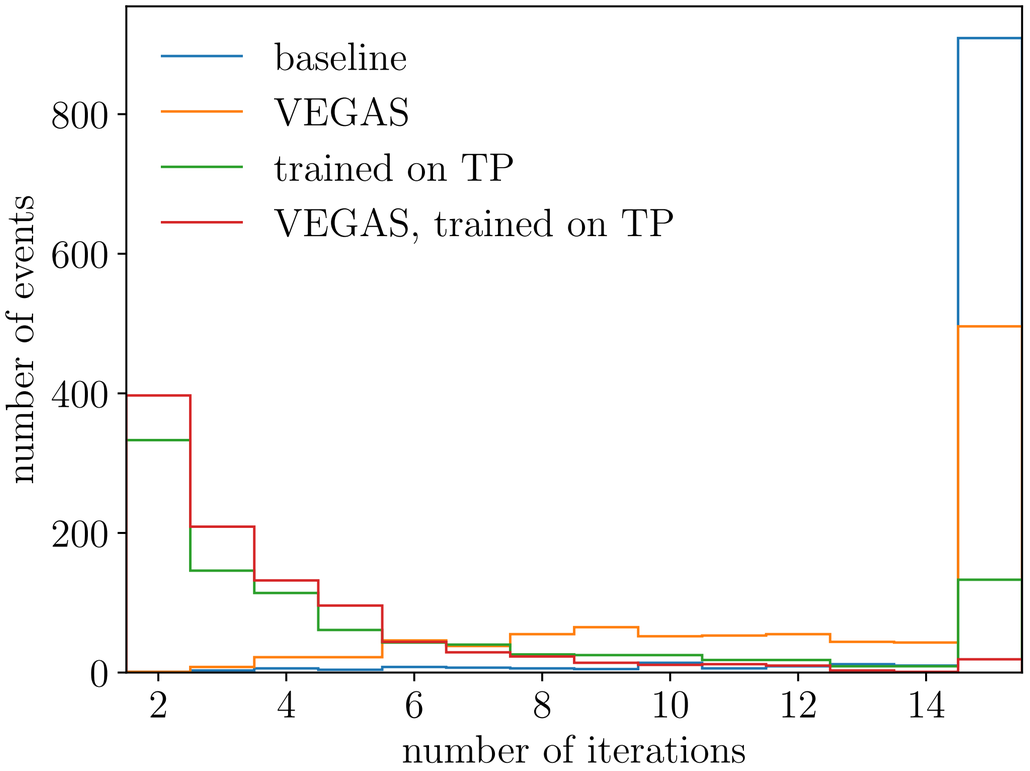
1 In practice $p_\theta(x_{reco}\,|\,x_{hard}) \neq p(x_{reco}\,|\,x_{hard})$

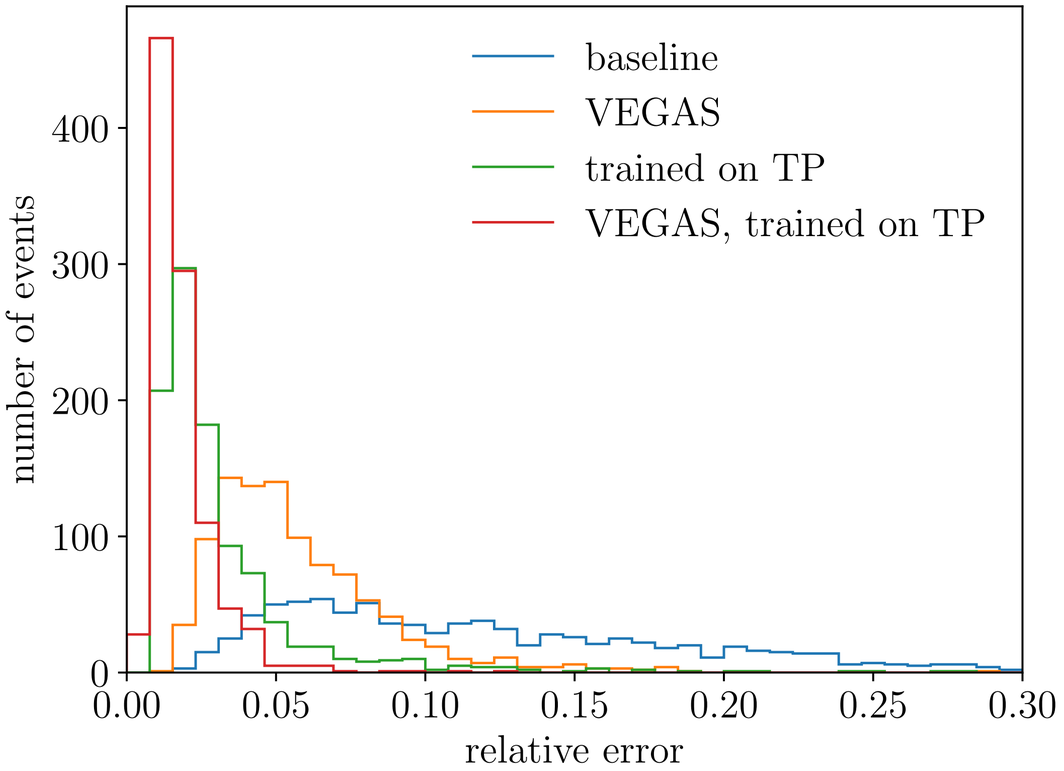
→ Train Sampling Network on trained Transfer Probability

2 In practice $q_\theta(x_{hard}) \neq q_{ideal}(x_{hard})$

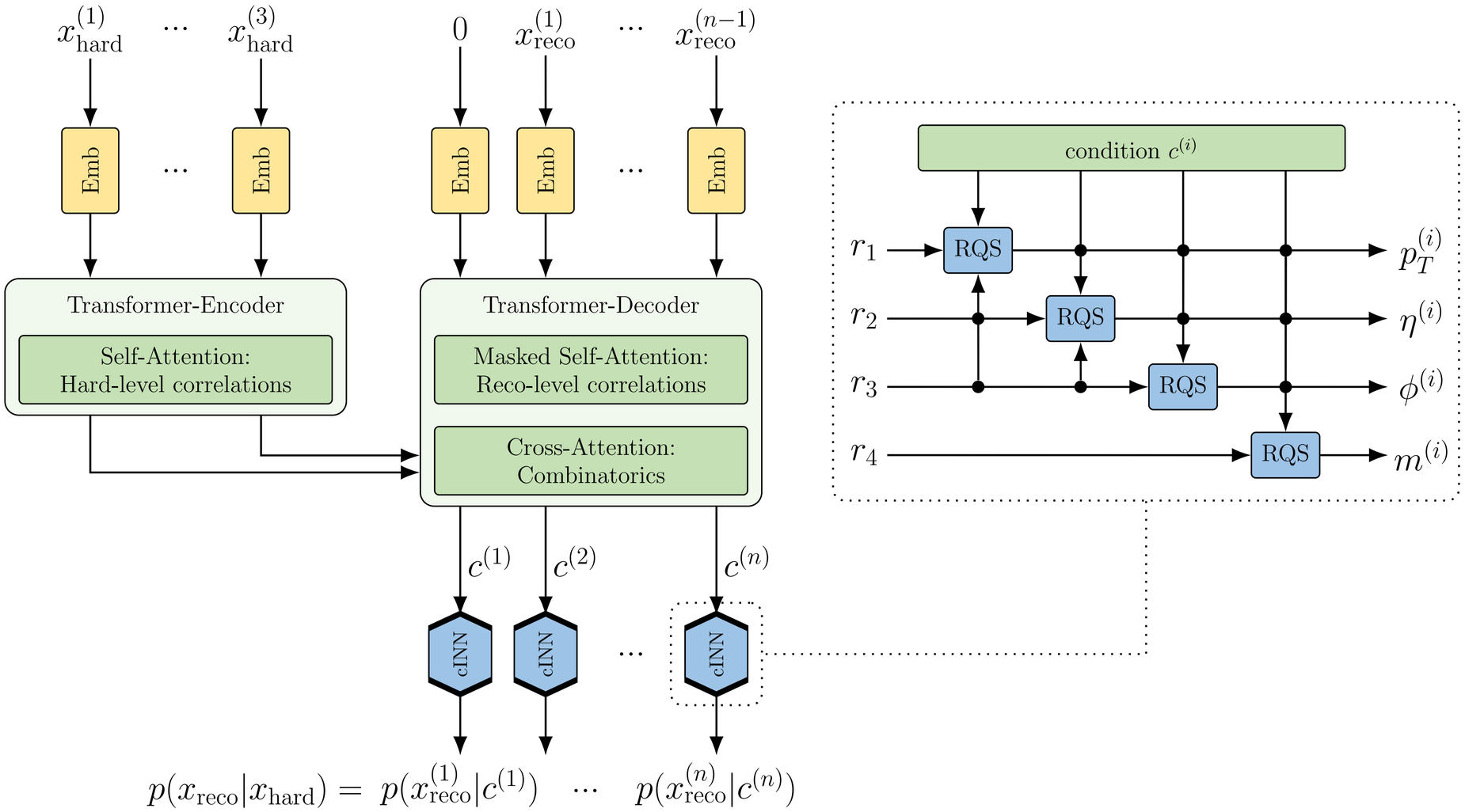
→ Further refine the cINN latent space with a VEGAS grid

Improving integration convergence

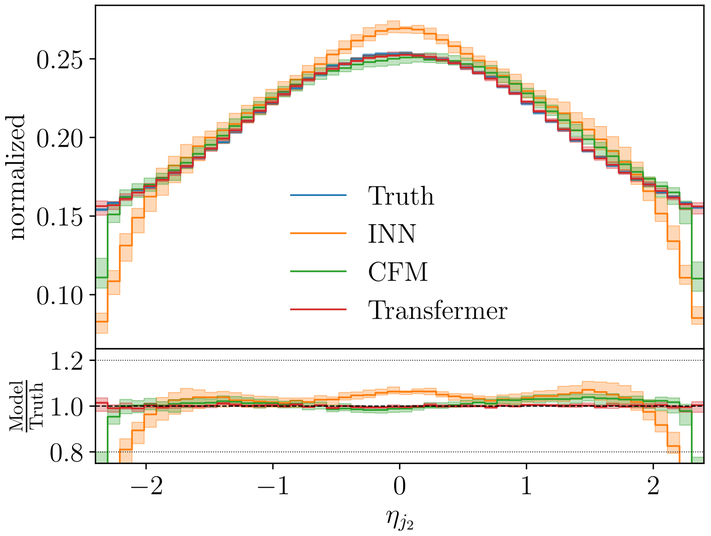


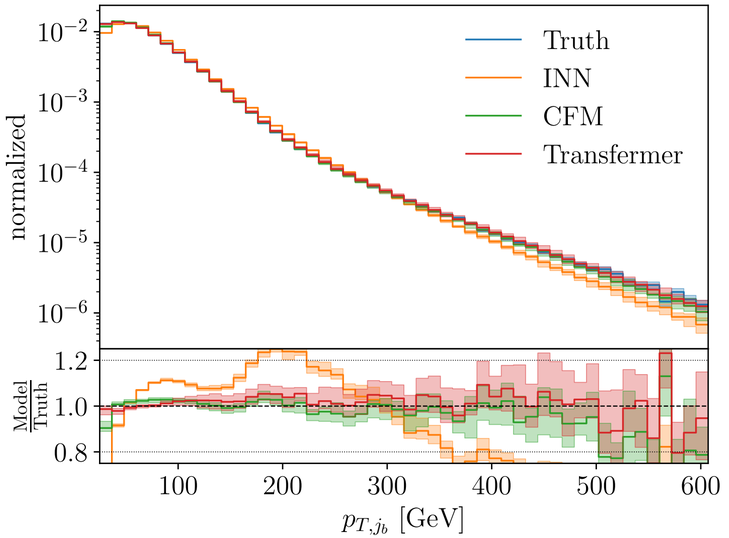


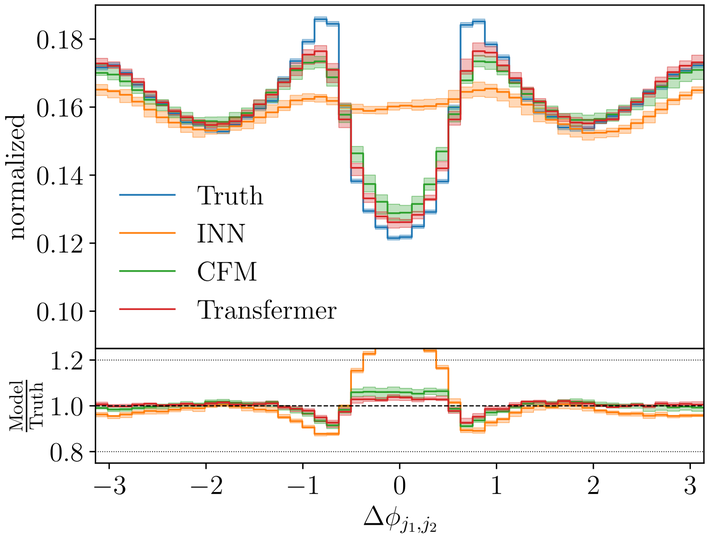
The Transfermer(Transfer-Transformer)

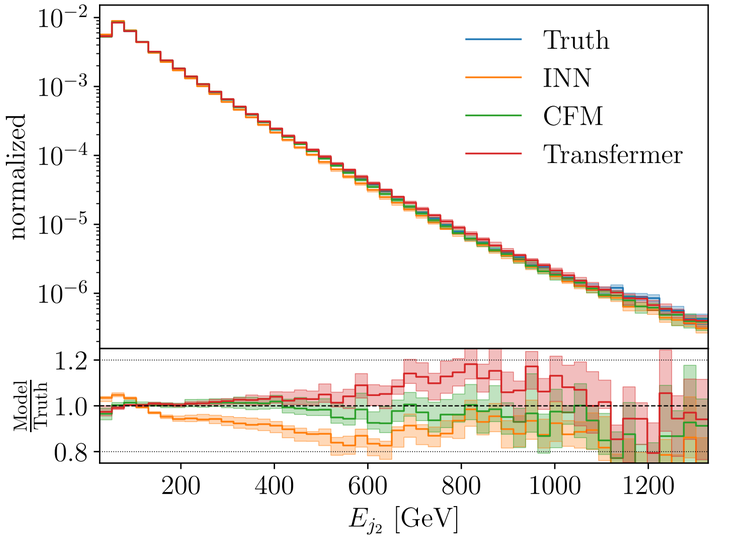


Transfer-Network observables

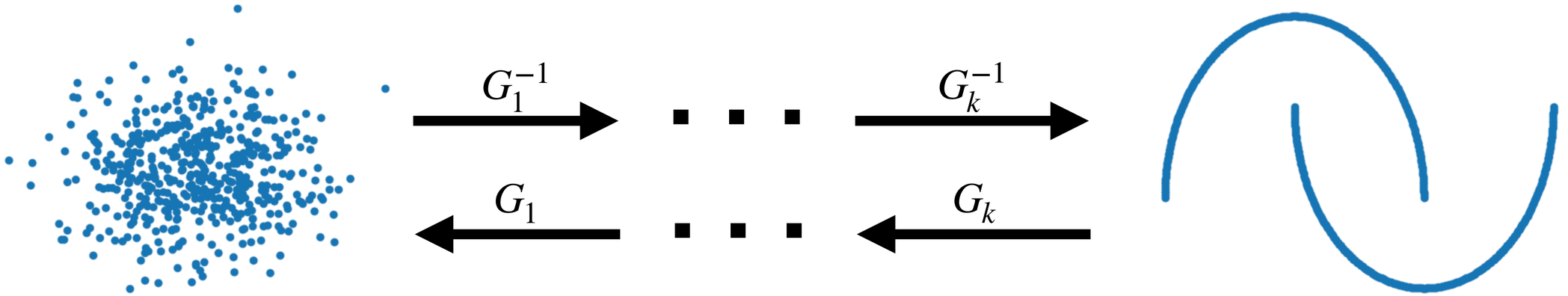






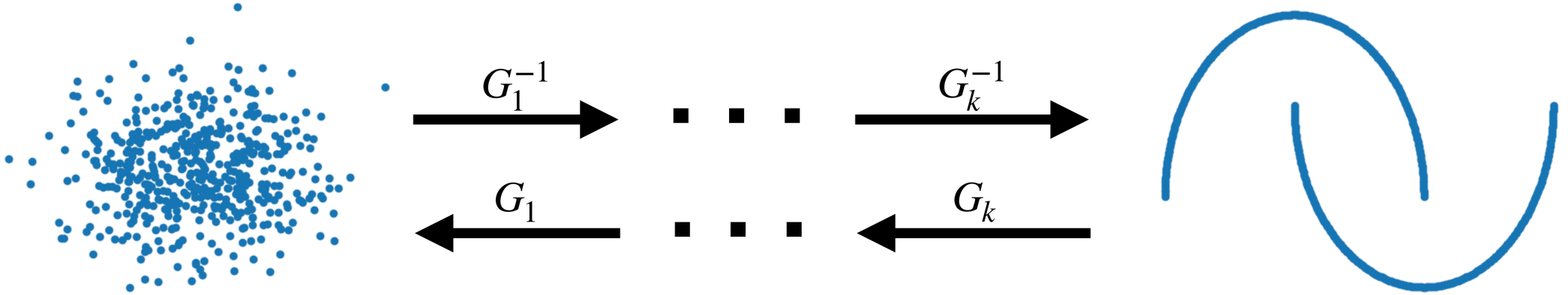


Invertible Neural Networks (or Normalizing Flows)



Invertible mapping between data and latent space

Invertible Neural Networks (or Normalizing Flows)



Invertible mapping between data and latent space

$$p_\theta(x) = p_{latent}(G_\theta(x)) \underbrace{\left| det\ J_{G_\theta} \right|}_{\left| det\ J_{G_1} \right| \left| det\ J_{G_2} \right| \dots \left| det\ J_{G_k} \right|}$$

Conditional Flow Matching

$$\frac{dr_t}{dt} = v_\theta(r_t, t)$$

time

$r_1 \sim p_{latent}(r)$

$r_0 \sim p_{target}$

ODE defines invertible mapping: $r_1 = G_\theta^{-1}(r) = r_1 + \int_1^0 v_\theta(r_t, t)\, dt$

$$p_\theta(x) = p_{latent}(G_\theta(x)) \overbrace{\left| det\ J_{G_\theta} \right|}^{\exp\left(\int_0^1 \nabla_r v_\theta(r_t, t)\, dt \right)}$$

Transfermer Results