Universität Münster

Off-Shell Processes from Generative Networks

Anja Butter[1,3], Tomáš Ježo[2], Michael Klasen[2],
Mathias Kuschick[2], Sofia Palacios Schweitzer[1] and Tilman Plehn[1]

1 Institut für Theoretische Physik, Universität Heidelberg, Germany
2 Institut für Theoretische Physik, Universität Münster, Germany
3 LPNHE, Sorbonne Université, Université Paris Cité, CNRS/IN2P3, Paris, France

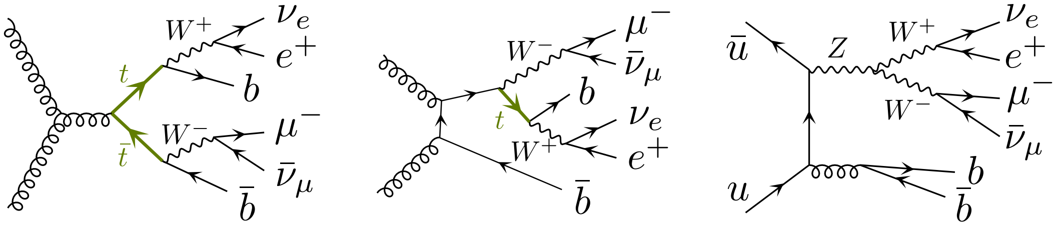
wissen.leben

Introduction

- Fast and precise predictions of event kinematics from first principles are the basis of every LHC analysis
- Two challenges:
  - Conceptual problems to overcome: e.g. dealing with loop diagrams with many scales
  - Technical problems: increased prescision comes with higher computational cost
- In this talk (and the corresponding paper) we focus on off-shell effects
  - Given the precision targets of the upcoming LHC runs, off-shell approximation is not justified
  - High computational cost of exact calculation
  - Neural-network surrogates: trained once, evaluated in parallel on GPUs

Off-shell vs on-shell effects

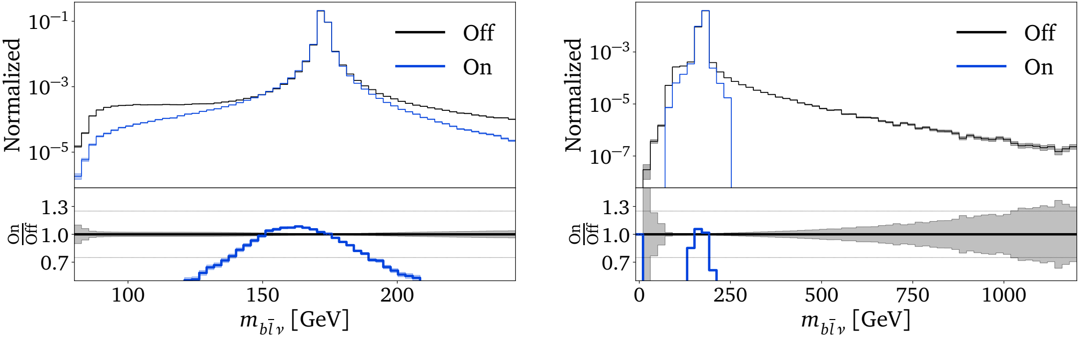
- For a proof of concept we are interested at the leading order in QCD dominated by $t\bar{t}$ production and dileptonic decay



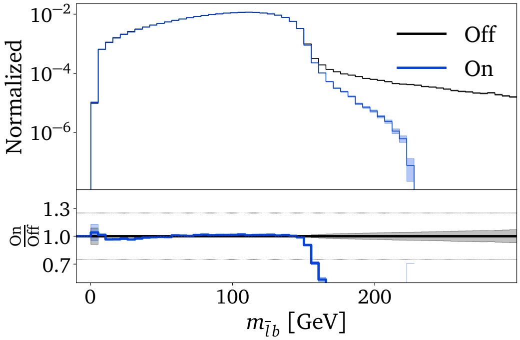
- Training datasets generated with hvq and bb4l containing 5 million events each
  - hvq data includes only approximate off-shell effects using finite top width
  - bb4l data includes full off-shell effects (including e.g non-resonant effects)

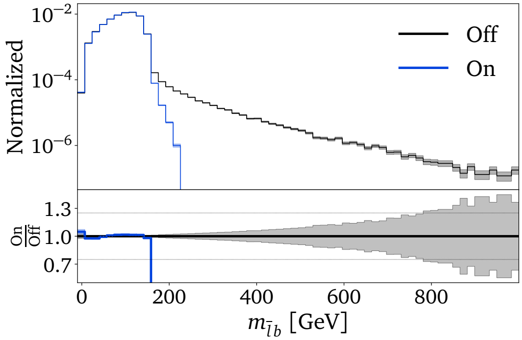
Universität
Münster

Off-shell vs on-shell effects



Off-shell vs on-shell effects



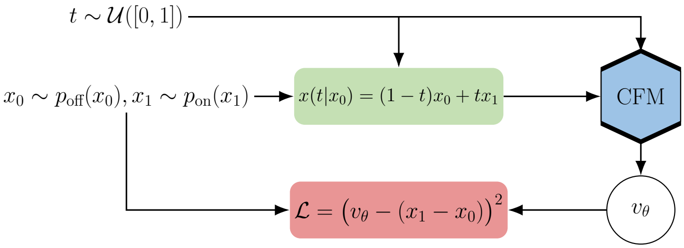


Off-shell vs on-shell effects - Problems and Solutions

- hard to generate complicated phase space
  - solution: transform easy to calculate phase space to hard to calculate phase space
- no pairings between on-shell and off-shell events
  - solution: choose method based on distributions
- We tried different methods
  - Train a classifier for event reweighting
    - no support in some regions of the phase space renders reweighting impossible
  - Flows4Flows
    - problems due to inflexibility of INNs
    - error amplification due to chaining of 2 INNs
  - Direct Diffusion
    - single feedforward DNN, no need for invertibility

Direct Diffusion

- We are using a setup called conditional flow matching (CFM) [arXiv:2209.15571, arXiv:2210.02747, arXiv:2209.03003, arXiv:2305.10475v2]
  - define $x(t=1) = x_1$ as a sample from the on-shell phase space
  - define $x(t=0) = x_0$ as a sample from the off-shell phase space



- For more details see talk by Sofia Palacios Schweizer (14:45, Main Auditorium)

Direct Diffusion

- We are using a setup called conditional flow matching (CFM):
  - Encoding transformation from on- to off-shell events as a continuous time evolution

$$\frac{dx}{dt} = v(x(t), t)$$

  - define $x(t = 1) = x_1$ as a sample from the on-shell phase space
  - define $x(t = 0) = x_0$ as a sample from the off-shell phase space
  - thus we get a time dependent probability density

$$p(x, t) \rightarrow \begin{cases} p_{\mathrm{off}}(x) & t \rightarrow 0 \\ p_{\mathrm{on}}(x) & t \rightarrow 1 \end{cases}$$

Direct Diffusion

- we adapt the linear trajectory between on- and off-shell events to be

$$x(t|x_0) = (1-t)x_0 + tx_1 \to \begin{cases} x_0 & t \to 0 \\ x_1 \sim p_{\text{on}} & t \to 1 \end{cases}$$

- hence the conditional velocity field becomes

$$v(x(t|x_0), t|x_0) = \frac{d}{dt}\left[(1-t)x_0 + tx_1\right] = -x_0 + x_1$$

Direct Diffusion

- from Bayesian statistics: $p(x,t) = \int dx_0\, p(x,t|x_0) p_{\mathrm{data}}(x_0)$
- making use of the continuity eq. to find unconditional $v(x,t)$:

$$\begin{aligned}
\frac{\partial p(x,t)}{\partial t} &= \int dx_0\, \frac{\partial p(x,t|x_0)}{\partial t} p_{\mathrm{data}}(x_0) \\
&= -\int dx_0\, \nabla_x \left(v(x,t|x_0) p(x,t|x_0) \right) p_{\mathrm{data}}(x_0) \\
&= -\nabla_x \left(p(x,t) v(x,t) \right)
\end{aligned}$$

we identify $$v(x,t) = \int dx_0\, \frac{v(x,t|x_0) p(x,t|x_0) p_{\mathrm{data}}(x_0)}{p(x,t)}$$

Direct Diffusion - Loss and Predictions

- the loss function used then is a simple MSE loss

$$\begin{aligned}\mathcal{L}_{\mathrm{CFM}} &= \left\langle [v_\theta(x,t) - v(x(t|x_0), t|x_0)]^2 \right\rangle \\ &= \left\langle [v_\theta((1-t)x_0 + tx_1, t) - (x_1 - x_0)]^2 \right\rangle_{t\sim\mathcal{U}(0,1), x_0\sim p_{\text{off}}, x_1\sim p_{\text{on}}}\end{aligned}$$

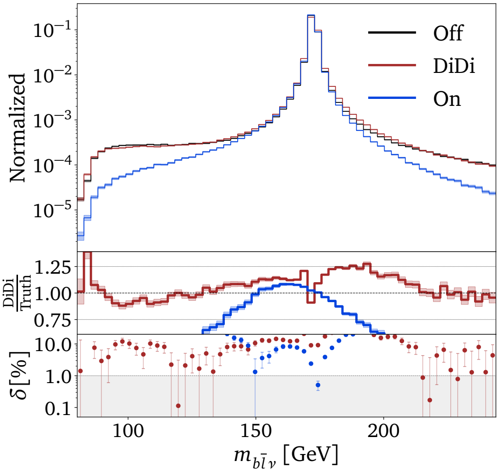
- predictions can be made by solving the ODE

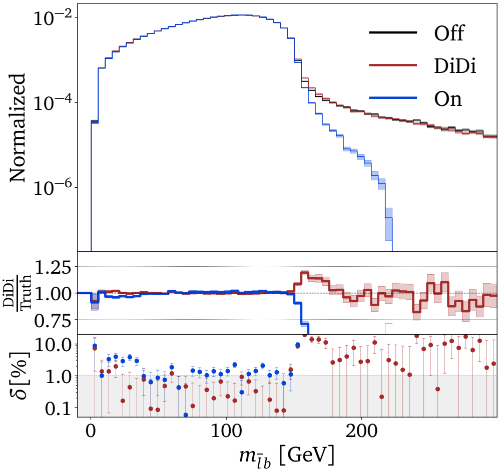
$$\begin{aligned}\frac{d}{dt}x(t) &= v_\theta(x(t), t) \\ \Rightarrow x_0 &= x_1 - \int_0^1 v_\theta(x,t)dt\end{aligned}$$

Phase Space Preprocessing

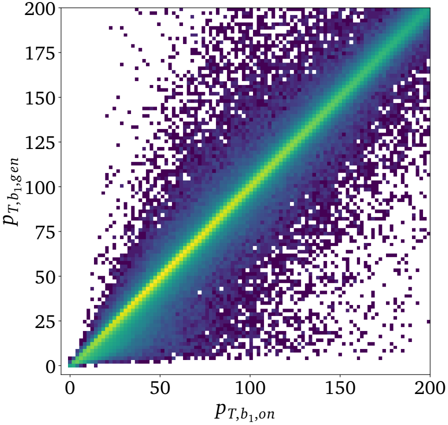
- Reduction of phase space
  - Phase space of 6 final state particles with 4 momentum components each (24D)
  - Transformed into p_T, η, ϕ, m with constant m (18D)
  - Aligning every event's coordinates to one ϕ (17D)
  - One p_x and one p_y is fixed due to $p_{\mathrm{T}}^{\mathrm{tot}} = 0$ (15D)
- Transformations:
  - $p_{\mathrm{T}} \to p_{\mathrm{T}}^{1/3}$
  - $\phi \to \operatorname{arctanh}(\phi/\pi)$
- Standardization

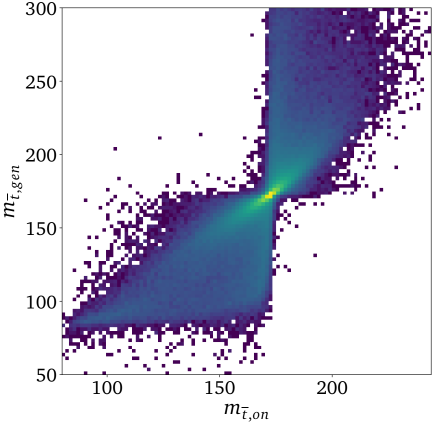
Direct Diffusion - Results





Direct Diffusion - Results





Reweighting

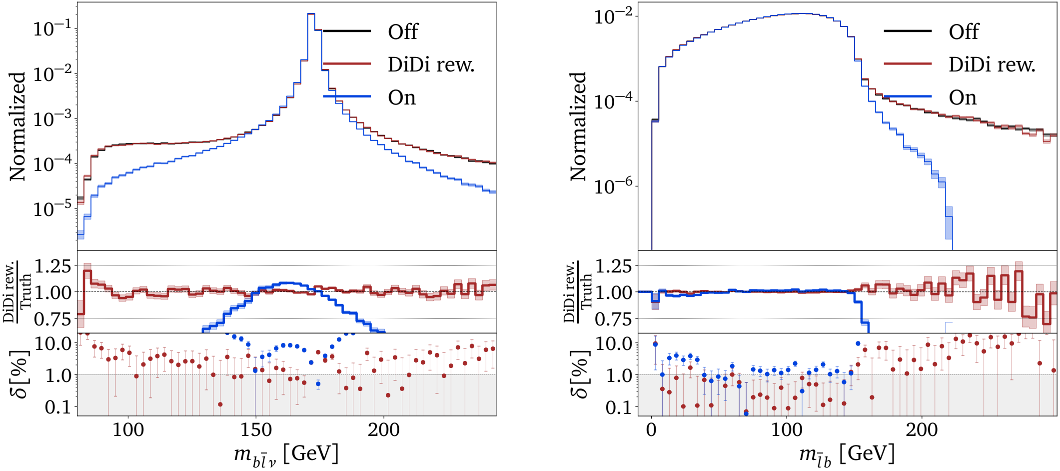
$$C(x) = \frac{p_{\mathrm{off,data}}(x)}{p_{\mathrm{off,data}}(x) + p_{\mathrm{off,model}}(x)}$$

$$w(x) = \frac{p_{\mathrm{off,data}}(x)}{p_{\mathrm{off,model}}(x)} = \frac{C(x)}{1 - C(x)}$$

- use p_{T}^{-1} instead of p_{T}

Universität Münster

Reweighting - Results



Conclusion

- Interesting problem, because it cannot be solved by modifying the amplitude at a give phase space point
- Instead, it requires a generative approach covering the complete off-shell phase space
- The advantage of this method is that the generative network only needs to learn a controlled deviation
- Small network with limited training effort can reproduce the target off-shell kinematics at the 10% level or better with only 5 million events
- Classifier reweighting improves its precision to the level of few percent even in challenging kinematic distributions

Universität Münster

Outlook

- Upcoming paper: Kicking it Off(-shell) with Direct Di-fusion

- Advancing to higher order processes
- Include processes that change final state structure
- Conditionalize training for different simulation parameters