Exploring Multi-Parameter Spaces with ML/AI

CosPT Workshop 2021 - IJCLab

Anja Butter

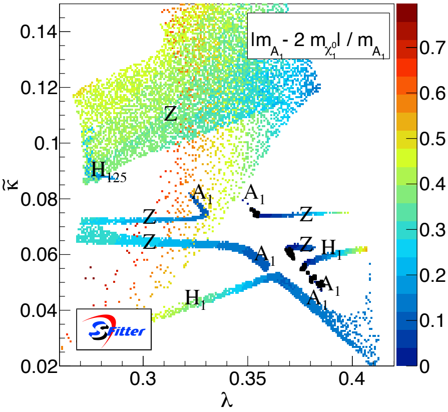
ITP, Universität Heidelberg

Exploring the SUSY parameter space

1. Choose parameter point
2. Calculate observables
3. Compare with data

→ Likelihood

- Challenges:
  - Curse of dimensionality
  - Complex hyperplanes due to relic density
- Traditional solution: MCMC
- New: learn likelihood with ML?



SUSY exploration vs Event generation

High dim. parameter space $\longleftrightarrow$ High dim. final state

Likelihood $\mathcal{L} \longleftrightarrow$ Cross section $\frac{d\sigma}{dp}$

Narrow DM funnel $\longleftrightarrow$ Narrow Breit Wigner

Expensive observables $\longleftrightarrow$ Higher order cross section

$\Rightarrow$ Explore similar techniques

Event generation in a nutshell

1. Generate phase space points

2. Calculate event weight

$$w_{event} = \boxed{f(x_1, Q^2) f(x_2, Q^2)} \times \boxed{\mathcal{M}(x_1, x_2, p_1, \dots p_n)} \times \boxed{J(p_i(r))^{-1}}$$

3. Unweighting via importance sampling
$\rightarrow$ optimal for $w \approx 1$

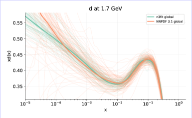
Event generation in a nutshell

$$w_{event} = \underbrace{f(x_1, Q^2)f(x_2, Q^2)}_{\text{PDF}} \times \overbrace{\mathcal{M}(x_1, x_2, p_1, \dots p_n)}^{\text{Matrix element}} \times \underbrace{J(p_i(r))^{-1}}_{\text{Phase space mapping}}$$

Event generation in a nutshell

Matrix element

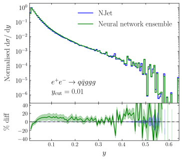
$$w_{event} = \boxed{f(x_1, Q^2) f(x_2, Q^2)} \times \boxed{\mathcal{M}(x_1, x_2, p_1, \dots p_n)} \times \boxed{J(p_i(r))^{-1}}$$



- NNPDF since 2002(!)
- S. Carrazza, J. Cruz-Martinez [1907.05075]

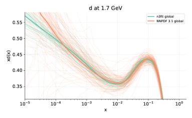
Phase space mapping

Event generation in a nutshell



- Amplitude estimation
- S. Badger, J. Bullock [2002.07516]
- J. Bendavid [1707.00028]

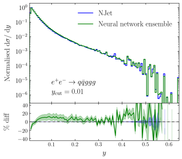
$$w_{event} = f(x_1, Q^2) f(x_2, Q^2) \times \mathcal{M}(x_1, x_2, p_1, \dots p_n) \times J(p_i(r))^{-1}$$



- NNPDF since 2002(!)
- S. Carrazza, J. Cruz-Martinez [1907.05075]

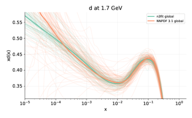
Phase space mapping

Event generation in a nutshell

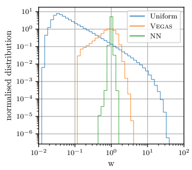


- Amplitude estimation
- S. Badger, J. Bullock [2002.07516]
- J. Bendavid [1707.00028]

$$w_{event} = f(x_1, Q^2) f(x_2, Q^2) \times \mathcal{M}(x_1, x_2, p_1, \ldots p_n) \times J(p_i(r))^{-1}$$

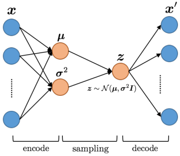


- NNPDF since 2002(!)
- S. Carrazza, J. Cruz-Martinez [1907.05075]

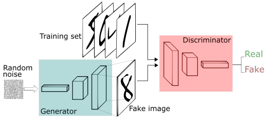


- Learn phase space mapping ($\to w \approx 1$)
- Gao et al. [2001.10028]
- Bothmann et al. [2001.05478]

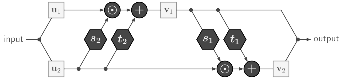
ML solutions for generative models



VAE

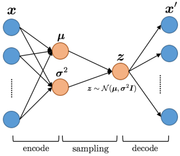


GAN

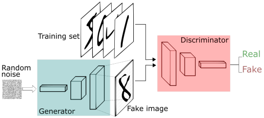


NF

ML solutions for generative models

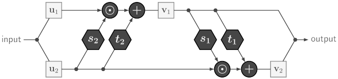


VAE



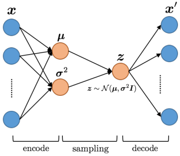
GAN

all kinds of hybrids

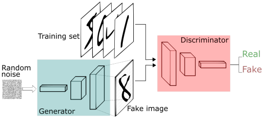


NF

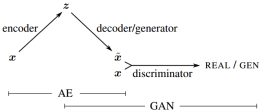
ML solutions for generative models



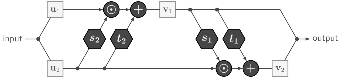
VAE



GAN

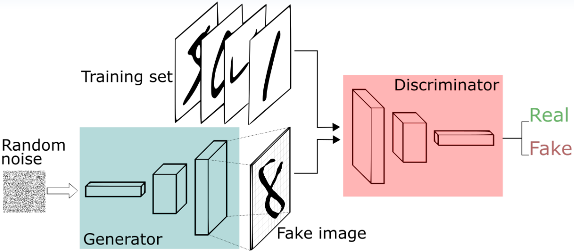


VAE-GAN



NF

Generative Adversarial Networks



Discriminator $[D(x_T) \to 1,\ D(x_G) \to 0]$

$$L_D = \Big\langle -\log D(x) \Big\rangle_{x\sim P_{Truth}} + \Big\langle -\log(1-D(x)) \Big\rangle_{x\sim P_{Gen}} \to -2\log 0.5$$

Generator $[D(x_G) \to 1]$

$$L_G = \Big\langle -\log D(x) \Big\rangle_{x\sim P_{Gen}}$$

$\Rightarrow$ **Equilibrium**

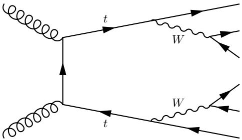
$\Rightarrow$ **New statistically independent samples**

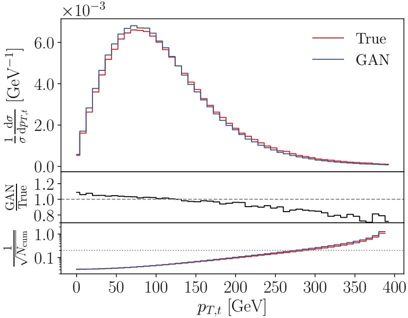
How to GAN LHC events [1907.03764]

- $t\bar{t} \to$ 6 quarks
- 18 dim output
  - external masses fixed
  - no momentum conservation

+ Flat observables ✓

– Systematic undershoot in tails [10-20% deviation]



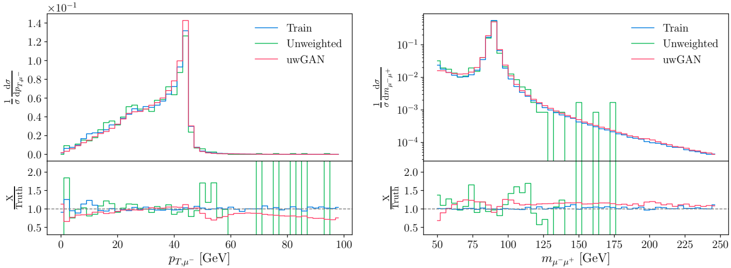


Training on weighted events

Low unweighting efficiencies $\rightarrow$ bottleneck before training

$\rightarrow$ Train on weighted events

$$\rightarrow L_D = \big\langle -w \log D(x) \big\rangle_{x \sim P_{Truth}} + \big\langle -\log(1 - D(x)) \big\rangle_{x \sim P_{Gen}}$$

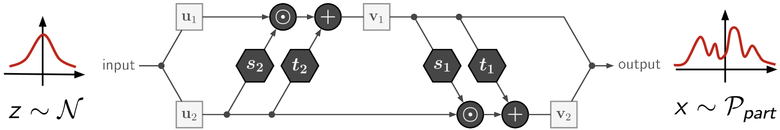


Populates high energy tails

Large amplification wrt. unweighted data!

Normalizing flows

Invertible neural networks



+ Bijective mapping

+ Fast evaluation in both directions

+ Tractable Jacobian

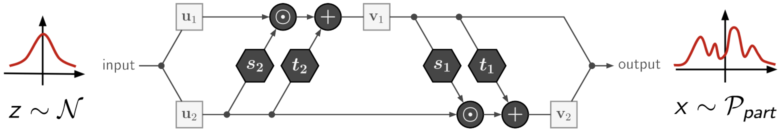
+ Enable correction for perfect precision

+ Extendable to Bayesian invertible networks

+ Trainable on either density or samples

Training on density

Sherpa [2001.05478, 2001.10028]



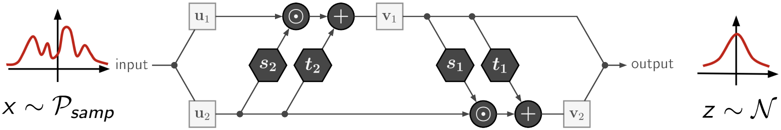
- $z \sim \mathcal{N} \rightarrow$ NN $\rightarrow x \sim p_x$
- $p_x(x) = p_z(z) \cdot J_{\text{NN}}$
- Given target density $t(x)$

$\rightarrow$ Train NN to minimize $\log(p_z(z) \cdot J_{\text{NN}}/t(x))$

- Problem: Calculate $f(x)$ each time

Training on samples

with T, Heimel, S. Hummerich, T. Krebs, T. Plehn, A. Rousselot, S. Vent [arXiv:2110.XXXXX]



- $x \sim p_{\text{samples}} \rightarrow$ NN $\rightarrow z$

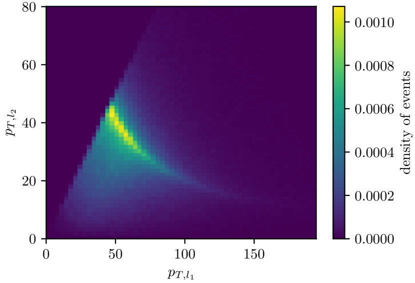
$\rightarrow$ Train NN to ensure $z \sim \mathcal{N}$

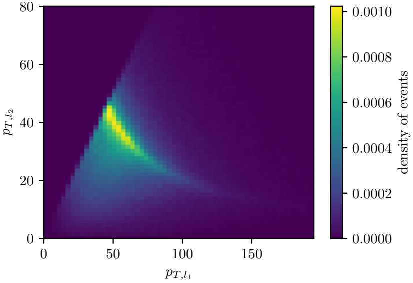
- Loss: Maximize posterior over network weights:

$$
\begin{aligned}
-\log(p(\theta|x)) &= -\log(p(x|\theta)) - \log(p(\theta)) + \text{const.} \\
&= -\log(p(z|\theta)) - \log(J) - \log(p(\theta)) + \text{const.}
\end{aligned}
$$

Results

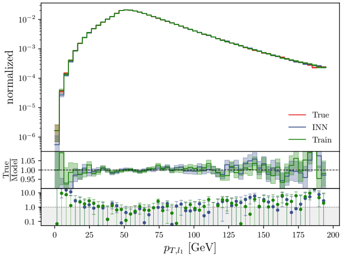
Z+ jets





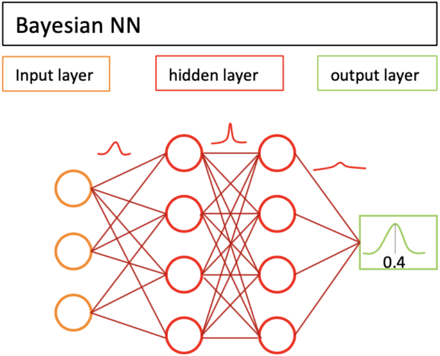
Results

Z+ jets



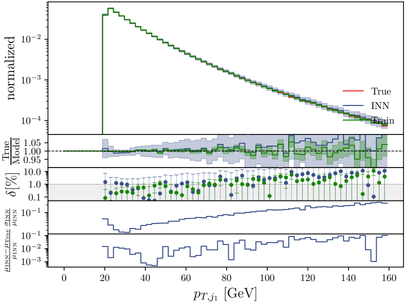
Uncertainties

- Bayesian approach to network parameters:
  - replace each weight by Gaussian $\mathcal{N}(\mu, \sigma)$
  - prior is normal distribution
  - sampling over network weights yields distribution



Results

Inclusive Z+jets production



How can we explore parameter spaces with ML?

- Speed up expensive calculations (regression networks)
- Explore parameter space with generative models (GAN, NF/INN)
- Develop iterative procedure to train while exploring

How can we explore parameter spaces with ML?

- Speed up expensive calculations (regression networks)
- Explore parameter space with generative models (GAN, NF/INN)
- Develop iterative procedure to train while exploring