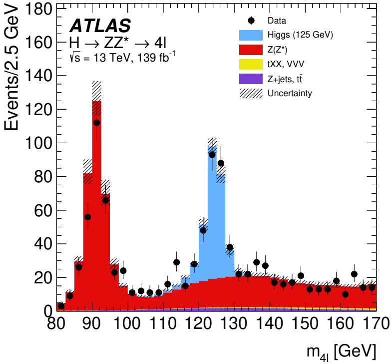
How to GAN LHC events

Anja Butter

ITP, Universität Heidelberg

MCnet Machine Learning School 2020

A typical LHC analysis



The HEP trinity

Theory

Fundamental Lagrangian

- Perturbative QFT

Standard Model vs. new physics

- Matrix elements, loop integrals

Experiment

Complex detector

- ATLAS, CMS, LHCb, ALICE, ...

Reconstruction of individual events

- Big data: jet images, tracks, ...

Precision simulations

First-principle Monte Carlo generators

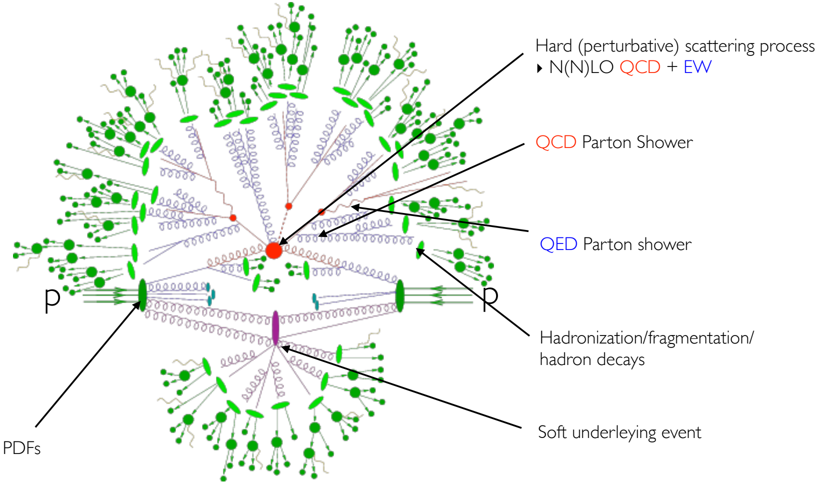
- Simulation of parton/particle-level events
- Herwig, Pythia, Sherpa, Madgraph, ...

Detector simulation

- Geant4, PGS, Delphes, ...

$\Rightarrow$ **Unweighted event samples**

Monte Carlo simulation of proton collision



A sherpa author & Jonas M. Lindert

Neural networks for precision simulations

Problems in MC simulations

- High-dimensional phase space
- Low unweighting efficiency
- CPU time increase per order in precision $\sim \times 100$
- Slow detector simulations

Solution with neural networks

- Flexible parametrisation
- Interpolation properties
- Fast evaluation
- Multiple generative models: GAN, VAE, normalizing flow

How to use ML for event generation

Estimate matrix element

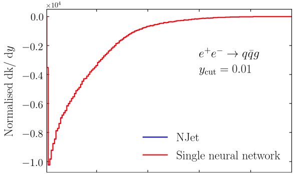
→ Regression

Optimize phase space mapping

→ Normalizing flow

Learn distribution of generated events

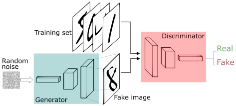
→ GAN, VAE



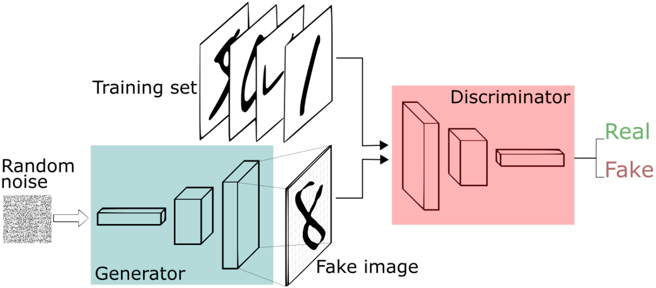
Badger & Bullock [2002.07516]

$$NF : x \to y$$

$$p_Y(y) = p_X(x) \det \frac{\partial y}{\partial x}$$



Generative Adversarial Networks



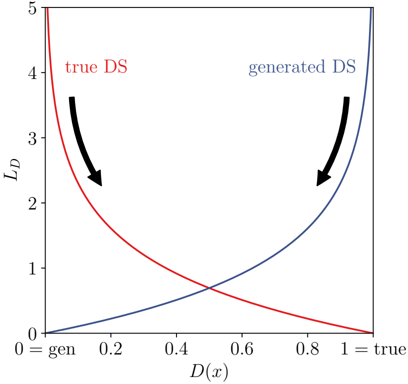
- Training data: $\{x_T\}$, Generated data: $\{x_G\}$
- **Discriminator** distinguishes $\{x_T\}, \{x_G\}$ $[D(x_T) \to 1,\ D(x_G) \to 0]$
- **Generator** fools discriminator $[D(x_G) \to 1]$

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

$$\max_G \min_D \Big[\big\langle -\log D(x) \big\rangle_{x\sim P_T} + \big\langle -\log(1 - D(x)) \big\rangle_{x\sim P_G} \Big]$$

Training the Discriminator

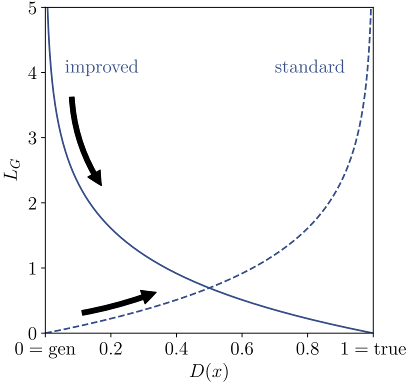
Discriminator loss



Minimize $$L_D = \big\langle -\log D(x) \big\rangle_{x\sim P_T} + \big\langle -\log(1 - D(x)) \big\rangle_{x\sim P_G}$$

Training the Generator

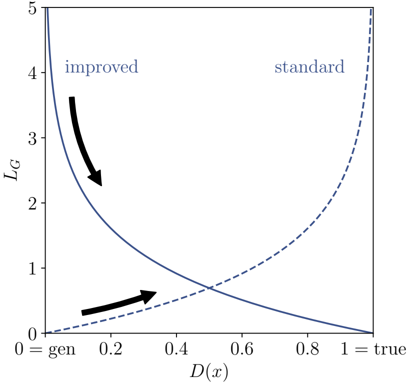
Generator loss



Maximize $$L_G = \big\langle -\log(1 - D(x)) \big\rangle_{x \sim P_G}$$

Training the Generator

Generator loss



Minimize $\quad L_G = \big\langle -\log D(x) \big\rangle_{x\sim P_G}$

Why GANs? Features, problems and solutions

+ Generate better samples than VAE
+ Large community working on GANs

Unstable Training

- Discriminator too strong → vanishing gradient

Solutions

Why GANs? Features, problems and solutions

+ Generate better samples than VAE
+ Large community working on GANs

Unstable Training

- Discriminator too strong → vanishing gradient

Solutions

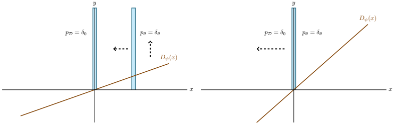
- Modified training objective:
  - Improved generator loss
  - Wasserstein GAN
  - Least square GAN
  - MMD-GAN
  - ...
- Check input dimensions!

Why GANs? Features, problems and solutions

+ Generate better samples than VAE
+ Large community working on GANs

Unstable Training

- Discriminator too strong → vanishing gradient
- Large gradient → no convergence



(a) $t = t_0$ (b) $t = t_1$

[1801.04406]

Adding gradient penalty

$$\phi(x) = \log \frac{D(x)}{1 - D(x)} \qquad \Rightarrow \qquad \frac{\partial \phi}{\partial x} = \frac{1}{D(x)} \frac{1}{1 - D(x)} \frac{\partial D}{\partial x}$$

$$L_D \to L_D + \lambda_D \left\langle (1 - D(x))^2 \, |\nabla \phi|^2 \right\rangle_{x \sim P_T} + \lambda_D \left\langle D(x)^2 \, |\nabla \phi|^2 \right\rangle_{x \sim P_G} ,$$

Why GANs? Features, problems and solutions

\+ Generate better samples than VAE

\+ Large community working on GANs

Unstable Training

- Discriminator too strong → vanishing gradient
- Large gradient → no convergence

Solutions

- Modified training objective:
  - Improved generator loss
  - Wasserstein GAN
  - Least square GAN
  - MMD-GAN
  - ...
- Check input dimensions!
- Regularization of the discriminator, eg. gradient penalty, weight clipping

Why GANs? Features, problems and solutions

\+ Generate better samples than VAE

\+ Large community working on GANs

Unstable Training

- Discriminator too strong $\rightarrow$ vanishing gradient
- Large gradient $\rightarrow$ no convergence

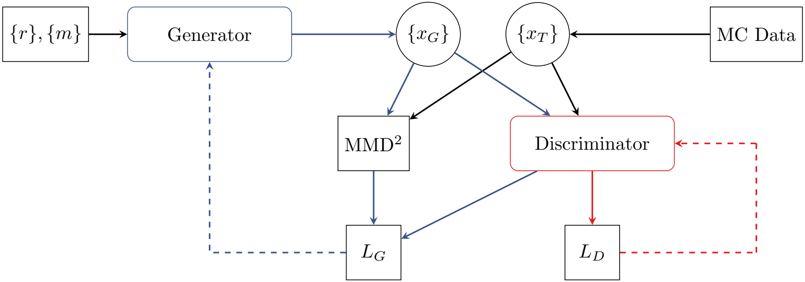
Solutions

- Modified training objective:
  - Improved generator loss
  - Wasserstein GAN
  - Least square GAN
  - MMD-GAN
  - ...
- Check input dimensions!
- Regularization of the discriminator, eg. gradient penalty, weight clipping
- Other possibilities to improve the training:
  - Use of symmetries
  - Whitening of data
  - Feature augmentation

Neural Networks for Event Generation?

- Input: random numbers, fixed parameters, eg. external masses
- Output: unweighted events

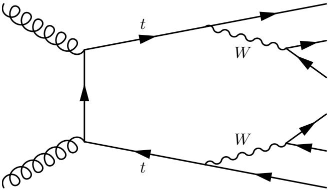
- Training data:
  - unweighted MC events or real data
  - can include parton showers, hadronization and detector effects



Top-Pair Production

GAN events for the $2 \to 6$ particle production process

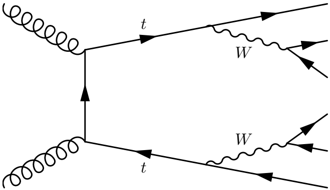
$$pp \to t\bar{t} \to (bW^-)\,(\bar{b}W^+) \to (bq_1\bar{q}_1')\,(\bar{b}q_2\bar{q}_2')\ .$$



Top-Pair Production

GAN events for the 2 → 6 particle production process

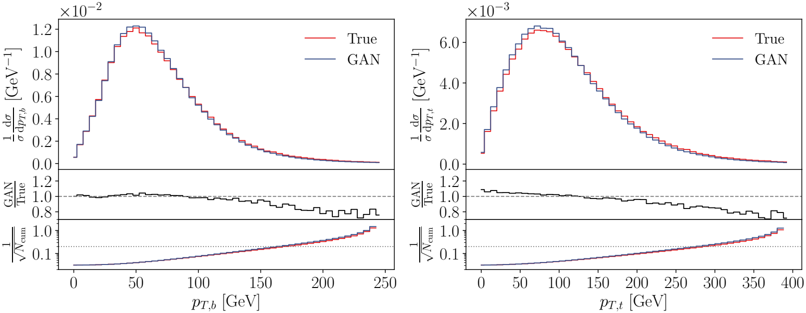
$$pp \to t\bar{t} \to (bW^-)\,(\bar{b}W^+) \to (bq_1\bar{q}_1')\,(\bar{b}q_2\bar{q}_2')\ .$$



Challenges: 16-dimensional phase-space, 4 resonances, phase-space boundaries, tails

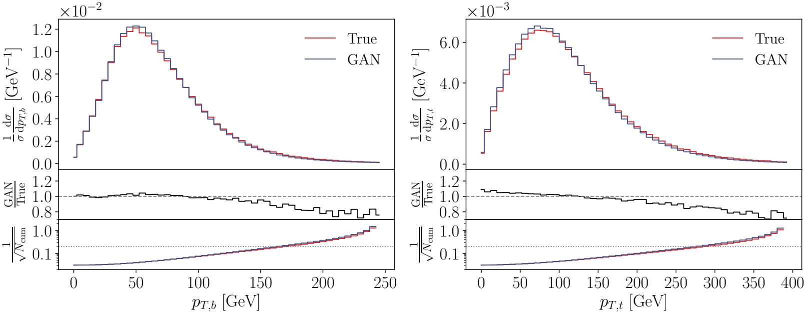
Remarks: fix masses of final state particles, no momentum conservation
→ generate 18 dim output

Momentum Distributions



$\rightarrow$ flat distributions easy to learn!

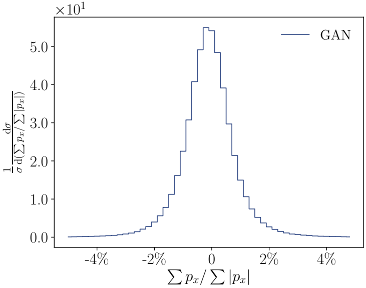
Momentum Distributions



$\rightarrow$ flat distributions easy to learn!

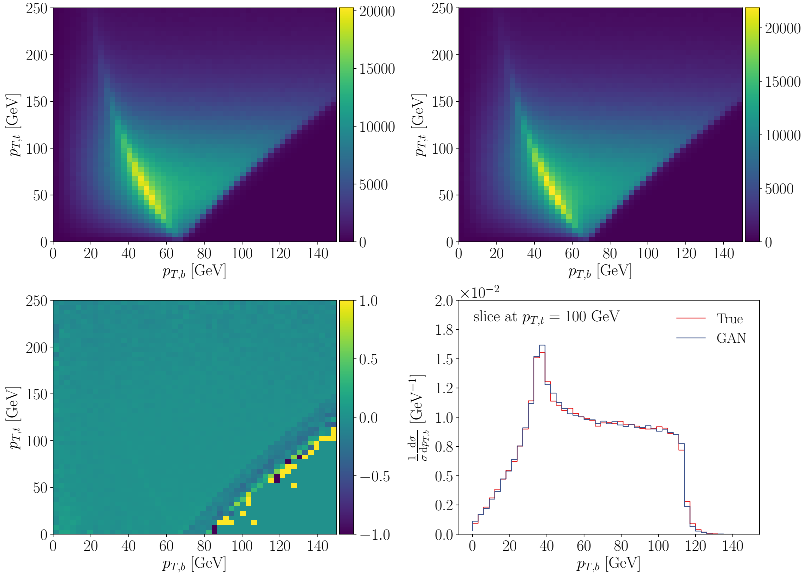
$\rightarrow$ Deviations scale with statistic uncertainty in the tail

Momentum Conservation by the Network



- generator learns to conserve momentum at a 1% level
- use correlations to evaluate performance

2-dimensional Correlations

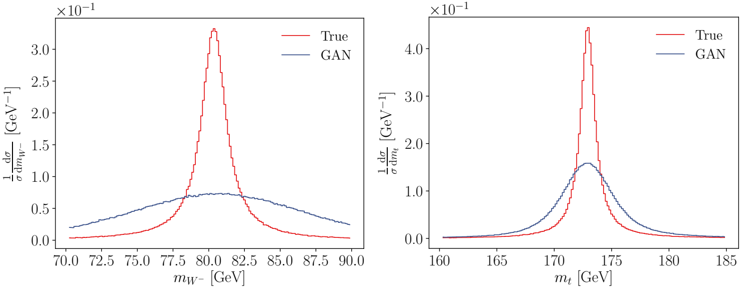


Invariant Mass Peaks

What about the resonances?

Invariant Mass Peaks

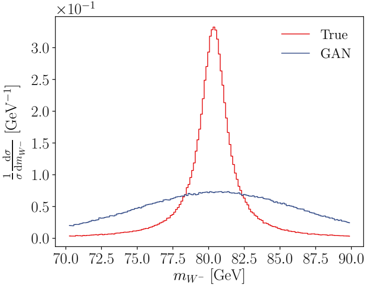
Simple GAN setup:

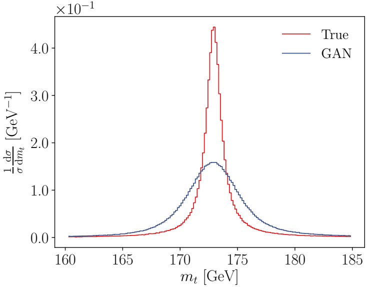


Challenge: resolve the mass peaks

Invariant Mass Peaks

Simple GAN setup:





Challenge: resolve the mass peaks

Standard solution: phase-space remapping

$$\int ds \frac{F(s)}{(s-m^2)^2 + m^2\Gamma^2} = \frac{1}{m\Gamma}\int dz\, F(s) \quad \text{with} \quad z = \arctan\frac{s-m^2}{m\Gamma}\,.$$

However: knowledge of m and Γ needed

Invariant Mass Peaks

Can we learn it simply from data?

Invariant Mass Peaks

Including the MMD Loss

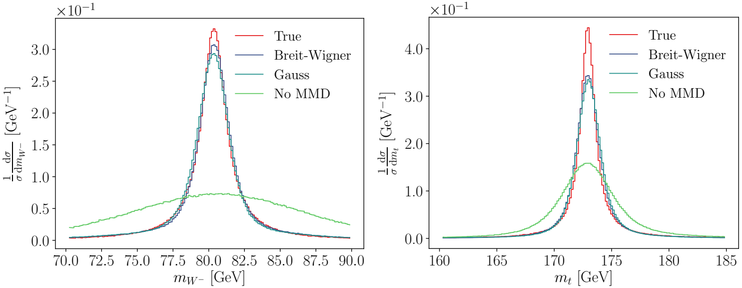
$$\mathrm{MMD}^2(P_T, P_G) = \big\langle k(x, x') \big\rangle_{x,x' \sim P_T} + \big\langle k(y, y') \big\rangle_{y,y' \sim P_G} - 2\big\langle k(x, y) \big\rangle_{x \sim P_T, y \sim P_G}$$

$$\mathrm{MMD}^2(P_T, P_G) = 0 \Leftrightarrow P_T = P_G$$

Invariant Mass Peaks

Including the MMD Loss

$$\mathrm{MMD}^2(P_T, P_G) = \big\langle k(x,x')\big\rangle_{x,x'\sim P_T} + \big\langle k(y,y')\big\rangle_{y,y'\sim P_G} - 2\big\langle k(x,y)\big\rangle_{x\sim P_T, y\sim P_G}$$



- free kernel choice → stable results
- **no** knowledge of m and Γ needed

First conclusion

- GAN is able to reproduce the full phase space structure of a realistic LHC process
- Flat distributions reproduced at arbitrary precison, limited only by statistics
- MMD loss to describe rich peaking resonances
- Possible to generate events from actual LHC event samples

Going further

Can we generate the difference of two distributions?

How to GAN event subtraction

Idea: sample based subtraction of distributions

1. Consistent multidimensional difference between two distributions
2. Beat bin-induced statistical uncertainty [interpolation of distributions]

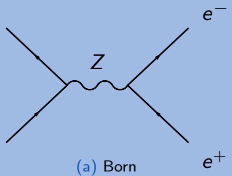
$$\Delta_{B-S} = \sqrt{n_B^2 N_B + n_S^2 N_S} > \max(\Delta_B, \Delta_S)$$

Physics case

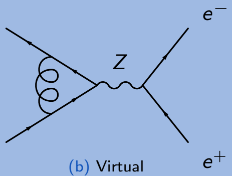
- Theory uncertainties have become a limiting factor for LHC analyses

$\rightarrow$ Need for better accuracy

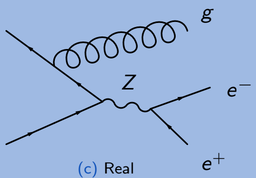
NLO in a nutshell



(a) Born



(b) Virtual



(c) Real

$$\sigma_{NLO} = \int d\Phi_B (B + V) + \int d\Phi_R R$$

Subtracting divergencies

- Virtual and real corrections diverge individually (eg. IR divergence)
- Sum of divergent contributions is finite

$\rightarrow$ Introduce dipoles D_i to cancel divergencies

Dipole subtraction

$$\sigma_{NLO} = \int d\Phi_B (B + V + \sum_i d\Phi_{R|B} D_i) + \int d\Phi_R (R - \sum_i D_i)$$

Subtracting divergencies

- Virtual and real corrections diverge individually (eg. IR divergence)
- Sum of divergent contributions is finite

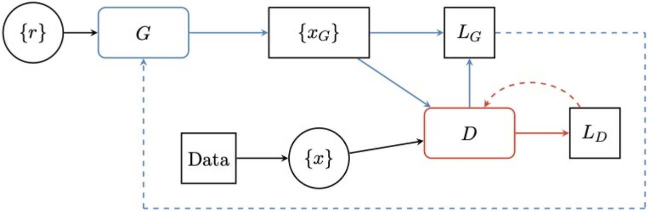
→ Introduce dipoles D_i to cancel divergencies

Dipole subtraction

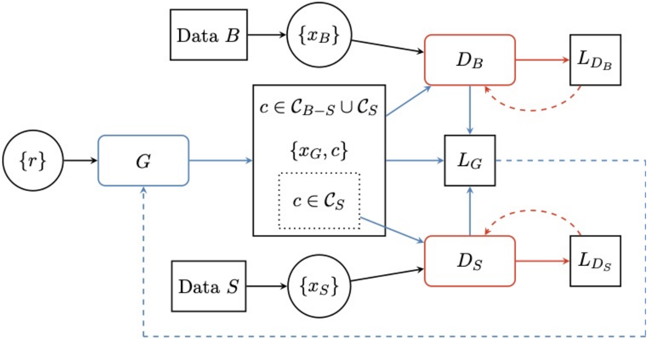
$$\sigma_{NLO} = \int d\Phi_B (B + V + \sum_i d\Phi_{R|B} D_i) + \int d\Phi_R (R - \sum_i D_i)$$

- Analytic solution only possible for simple processes
- Numeric subtraction of samples:
  → large statistic uncertainties
  → limits efficiency
- More applications:
  - Soft-collinear subtraction, multi-jet merging, on-shell subtraction
  - Background subtraction [4-body decays →preserves correlations]

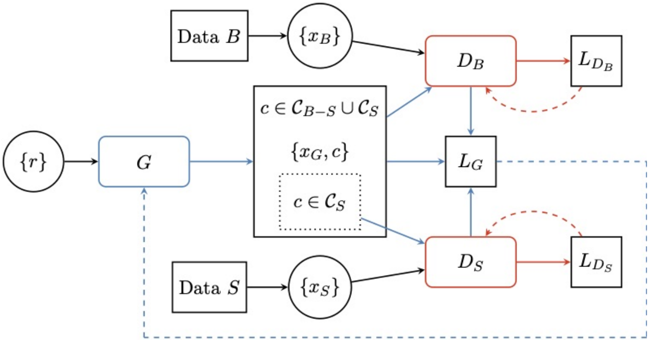
From a standard GAN ...



... to a subtraction GAN



... to a subtraction GAN



label vector $c = \begin{pmatrix} c_S \\ c_{B-S} \end{pmatrix}$

	c_{B-S}	c_S
Data B	1	1
Data S	0	1
B-S	1	0

Building the loss function

- Standard GAN loss for each discriminator

Building the loss function

- Standard GAN loss for each discriminator
- Differentiable function to count events of one type

$$f(c) = e^{-\alpha(\max(c)^2-1)^{2\beta}} \in [0,1] \qquad \text{for} \qquad 0 \leq c_i \leq 1\ .$$

Building the loss function

- Standard GAN loss for each discriminator
- Differentiable function to count events of one type

$$f(c) = e^{-\alpha(\max(c)^2-1)^{2\beta}} \in [0,1] \qquad \text{for} \qquad 0 \leq c_i \leq 1 \,.$$

- Reward clear class assignment

$$L_G^{(\text{class})} = \left(1 - \frac{1}{b} \sum_{c \in batch} f(c)\right)^2$$

Building the loss function

- Standard GAN loss for each discriminator
- Differentiable function to count events of one type

$$f(c) = e^{-\alpha(\max(c)^2-1)^{2\beta}} \in [0,1] \qquad \text{for} \qquad 0 \leq c_i \leq 1 \,.$$

- Reward clear class assignment

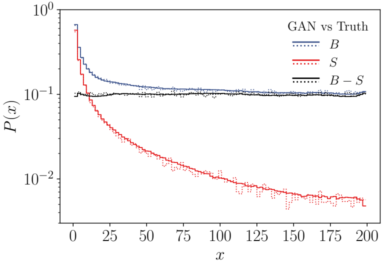
$$L_G^{(\text{class})} = \left(1 - \frac{1}{b} \sum_{c \in batch} f(c)\right)^2$$

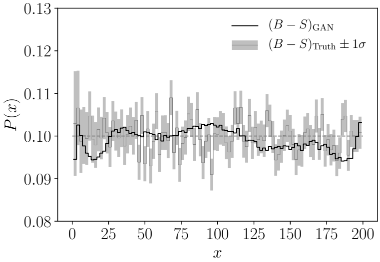
- Fix normalization

$$L_{G_i}^{(\text{norm})} = \left(\frac{\sum_{c \in \mathcal{C}_i} f(c)}{\sum_{c \in \mathcal{C}_B} f(c)} - \frac{\sigma_i}{\sigma_0}\right)^2$$

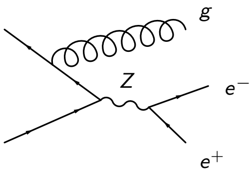
Toy example

$$P_B(x) = \frac{1}{x} + 0.1$$
$$P_S(x) = \frac{1}{x}$$
$$P_{B-S}(x) = 0.1$$





Back to the original problem

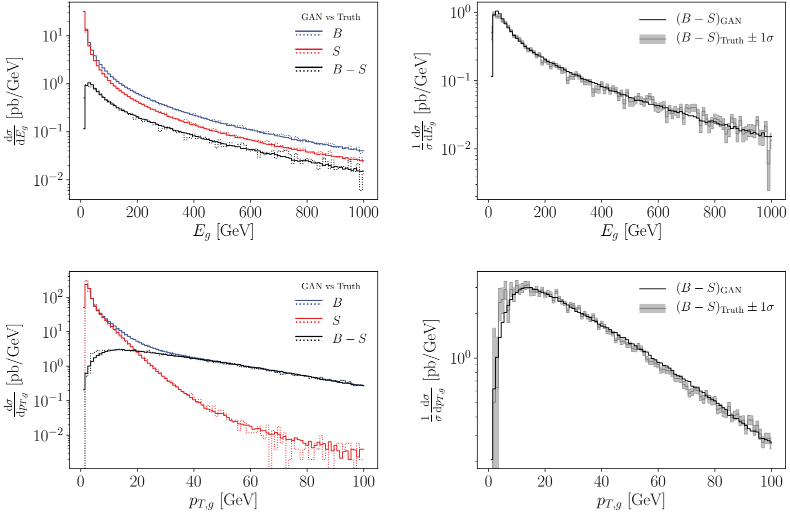


- Subtract the Catany Seymour Dipole from the real emission term

- For proof of concept we use a slightly modifed Catany Seymour kernel $\rightarrow$ increase difference

- Training
  - 10^5 samples per distribution
  - 4-vector representation of Z and g
  - $E_g > 5$ GeV

Results


GAN vs Truth
B
S
$B-S$
$\frac{d\sigma}{dE_g}$ [pb/GeV]
E_g [GeV]
$(B-S)_{\mathrm{GAN}}$
$(B-S)_{\mathrm{Truth}} \pm 1\sigma$
$\frac{1}{\sigma}\frac{d\sigma}{dE_g}$ [pb/GeV]
$\frac{d\sigma}{dp_{T,g}}$ [pb/GeV]
$p_{T,g}$ [GeV]
$\frac{1}{\sigma}\frac{d\sigma}{dp_{T,g}}$ [pb/GeV]

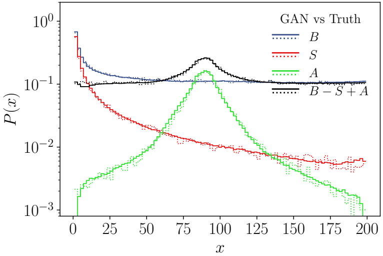
Include addition

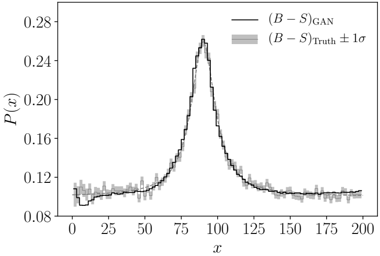
$$P_B(x) = \frac{1}{x} + 0.1$$

$$P_S(x) = \frac{1}{x}$$

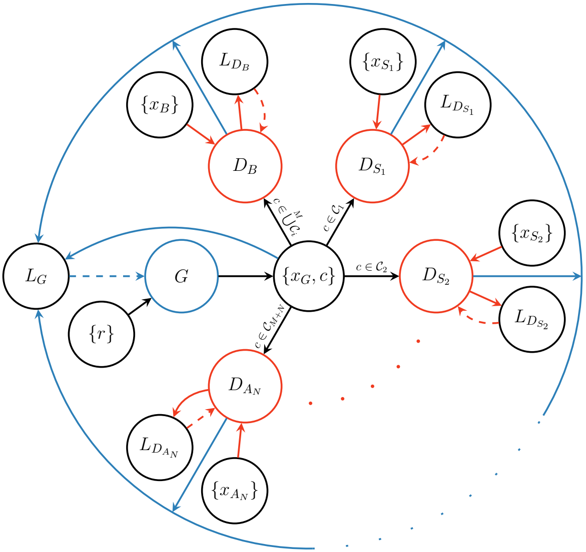
$$P_A(x) = \frac{5}{\pi} \frac{10}{10^2 + (x-90)^2}$$

	$\mathcal{C}_{B-S}$	$\mathcal{C}_S$	$\mathcal{C}_A$
Data B	1	1	0
Data S	0	1	0
Data A	0	0	1
B-S+A	1	0	1





Allowing for more datasets



Conclusion

- Build on GAN setup: learns underlying distributions

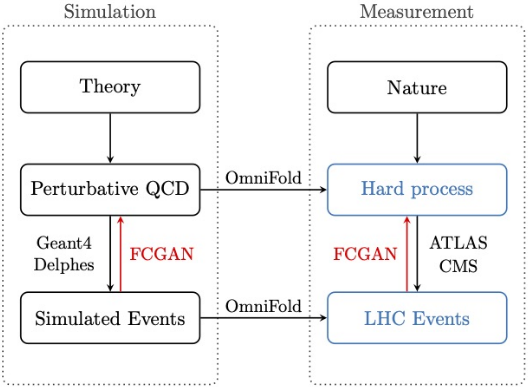
- More complex: subtraction GAN

$\rightarrow$ learn difference of two distributions

- Applications: subtract real-emission corrections to improve computation efficiency
- Background subtraction

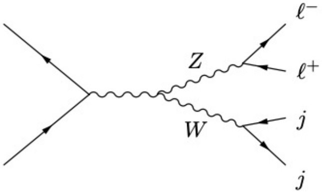
- New tool for our ML toolbox
  $\rightarrow$ other use cases?

Unfolding detector effects



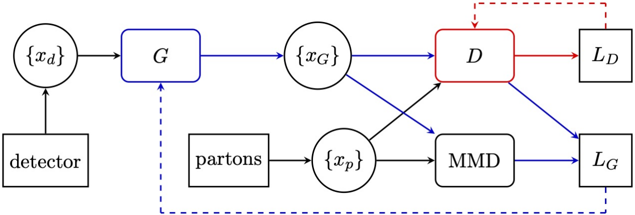
Setup

$$pp \to ZW^{\pm} \to (\ell^- \ell^+)\,(jj) \tag{1}$$



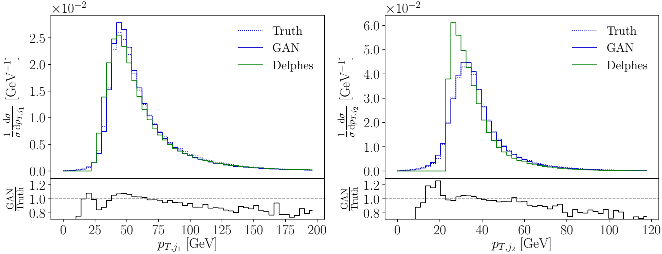
- 300k events using MadGraph+Pythia and Delphes, no ISR
- event selection:
  - exactly 2 jets and a pair of same-flavor opposite-sign leptons.
  - $p_{T,j} > 25$ GeV $\&|\eta_j| < 2.5$ GeV.
- Assign jet to a corresponding parton level object based on ΔR
- Assign leptons based on their charge

GAN setup



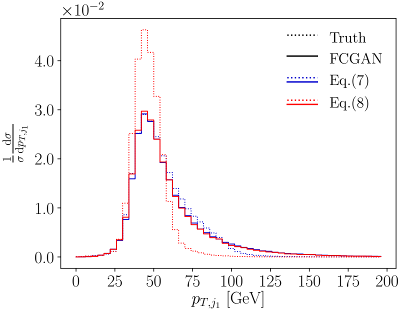
- Use GAN to map detector level events to parton level events

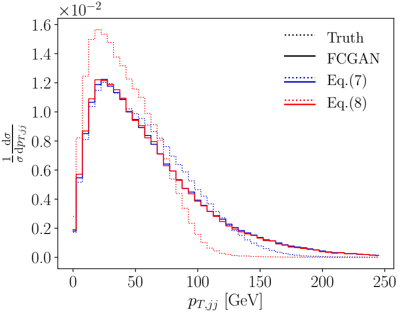
Unfolding the full distribution



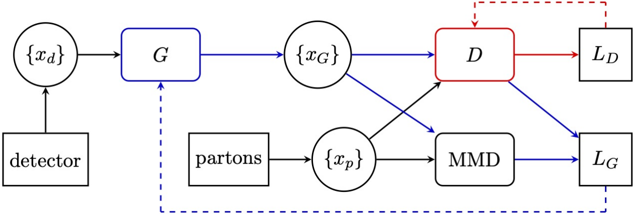
Slicing

$$\text{Eq.(7)}: \quad p_{T,j_1} = 30 \ldots 100 \text{ GeV}$$
$$\text{Eq.(8)}: \quad p_{T,j_1} = 30 \ldots 60 \text{ GeV} \quad \text{and} \quad p_{T,j_2} = 30 \ldots 50 \text{ GeV}$$

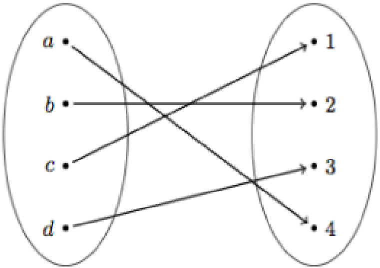




GAN setup



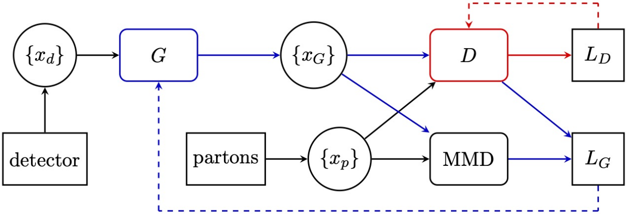
Problems



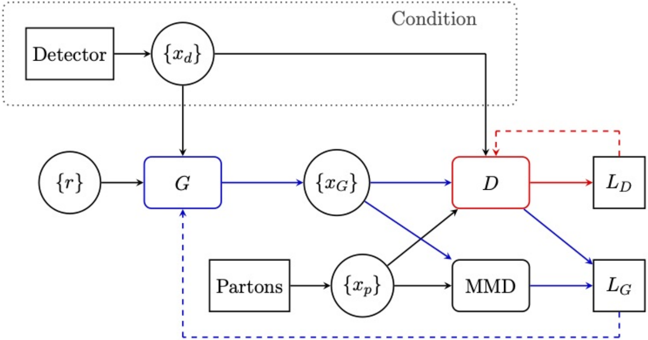
- No use of detector level information
- No concept of locality
- No stochastic mapping

$\rightarrow$ Conditional GAN

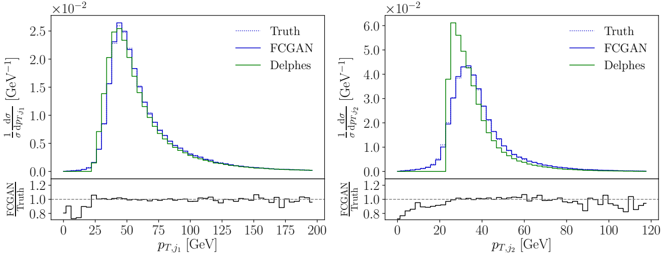
Conditional GAN I



Conditional GAN I



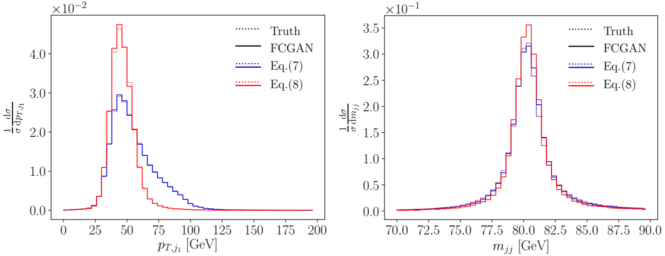
Full distributions



$\rightarrow$ Nice by-product: No systematic effect in the tails!

Slicing

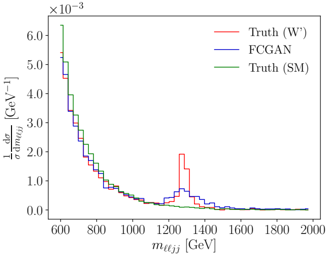
$$\text{Eq.}(7): \quad p_{T,j_1} = 30 \ldots 100 \text{ GeV} \quad (\sim 88\%)$$
$$\text{Eq.}(8): \quad p_{T,j_1} = 30 \ldots 60 \text{ GeV} \quad \text{and} \quad p_{T,j_2} = 30 \ldots 50 \text{ GeV} \quad (\sim 38\%)$$



→ Slices mapped correctly

Inserting a W' resonance

- Network trained on SM data



- Mean reproduced correct
- Width smeared out

Summary

- GANs can learn underlying distributions from event samples
- MMD improves performance for special features

- Generate difference of two event distributions

- Unfolding with standard GAN: No meaningful detector $\leftrightarrow$ parton matching
- Conditional GAN to link detector and parton level events