GANs for calorimeter simulation

Sofia Vallecorsa

Outline

- Introduction
  - Generative Adversarial Networks
- 3dGAN: the LCD high granularity calorimeter
- Summary & Plans

Introduction

- Started a fastsim R&D activity
- We want to have a generic interface capable of using different fastsim options including ML based
- Submitted a proposal for an IPPC including the development of such tools, including test of different ML techniques
- Started working on Generative Adversarial Networks

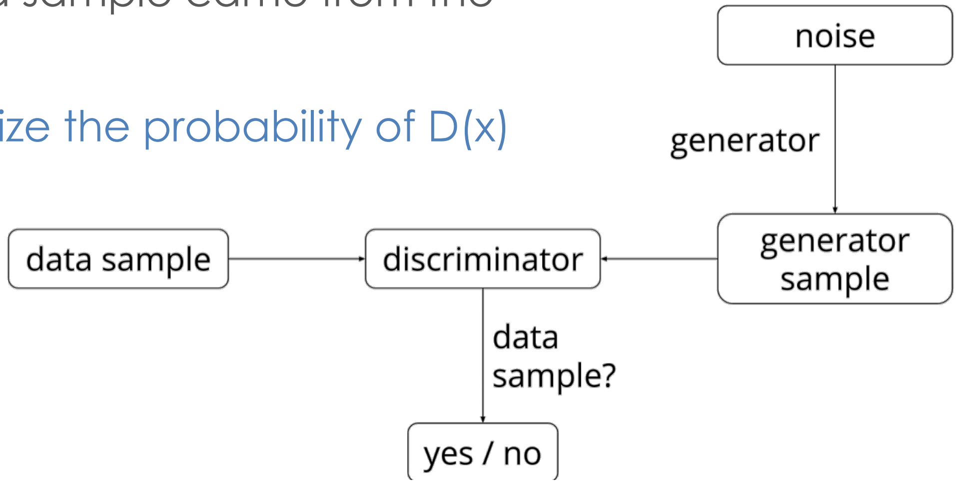
ML for (calorimeter) simulation

- Generative models (Generatirve Stochastic Netowrks, Variational Auto-Econders, Generative Adversarial Networks, ..) can be used for simulation
  - Realistic generation of samples
  - Use complicated probability distributions, optimise multiple output for a single input
  - Work well with missing data

- Can 3D imaging approaches be useful?
- Can we keep accuracy while doing things faster?
- Can we sustain the increase in detector complexity (future highly-granular calorimeters are more demanding)?
- What are the resources are needed?

Generative Adversarial Networks

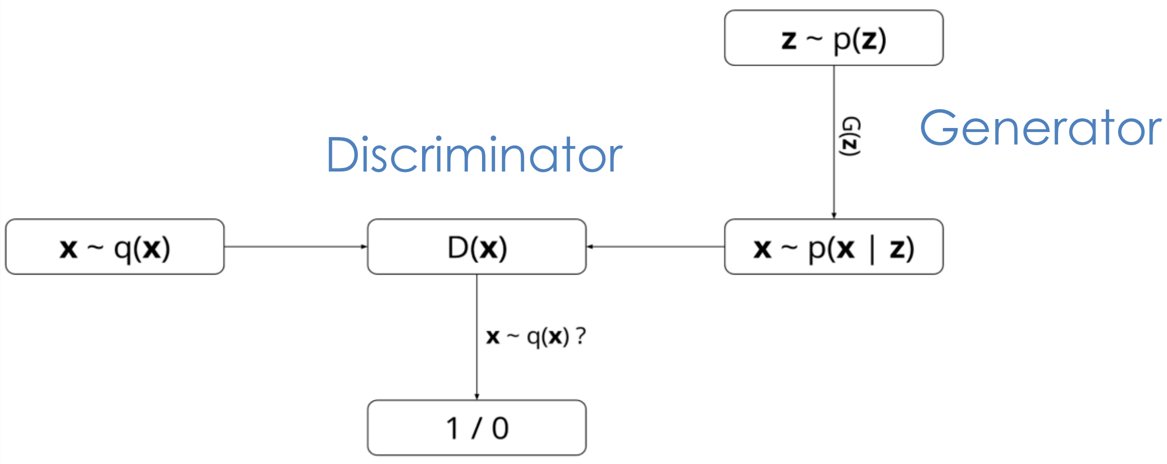
- Simultaneously train two models:
  - G(z) captures the data distribution
  - D(x) estimates the probability that a sample came from the training data rather than G
- Training procedure for G(z) is to maximize the probability of D(x) making a mistake



Goodfellow et al. 2014
Conditional GAN, arXiv: 1411.1744
Deep Convolutional GAN, arXiv:1511.06434
Auxiliary Classifier GAN, arXiv:1610.0958

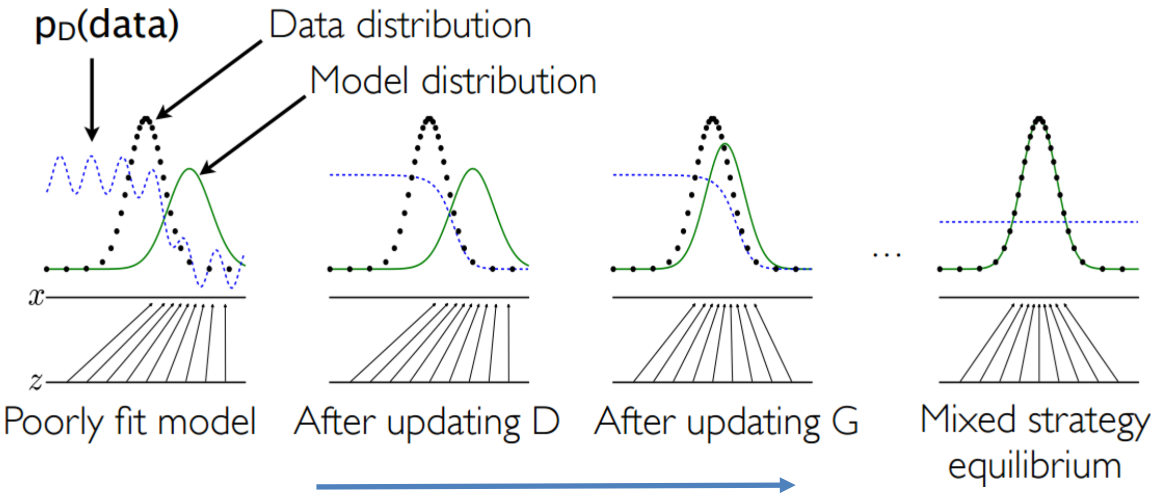
Generative Adversarial Networks

arXiv:1406.2661v1



$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

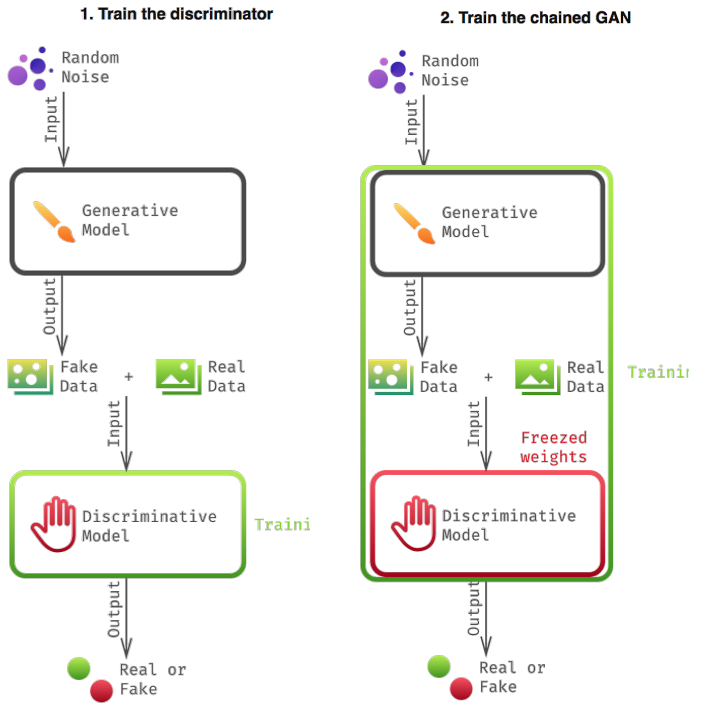
Generative Adversarial Networks



1. D is only a partially accurate classifier.
2. D is trained to discriminate samples from data
3. After updating G, gradient of D has guided G(z) to regions more likely to be classified as data.
4. G and D don't improve because $p_g = p_{data}$. D is unable to differentiate.

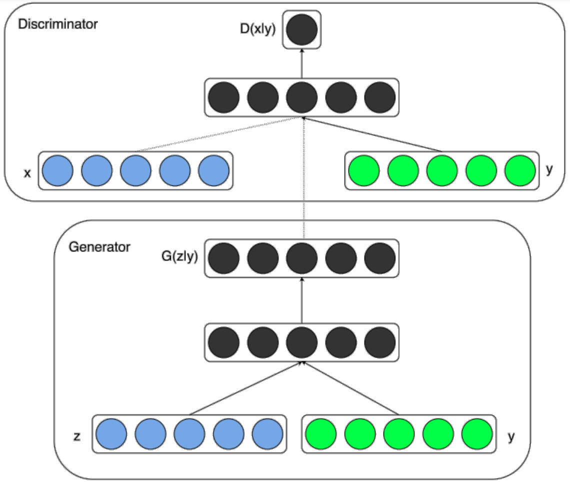
Training GANs is a many steps process:

- Sample noise and generate images with G
- Sample images from training dataset and train the D to recognize G data from real data
- Train combined G + D to tell you that G data it is real
  - At this stage D weights are frozen.
- Back feed info to discriminator and repeat for as many epochs as needed



Conditional GAN

- GAN framework can be extended to learn a parameterized generator $p_{model}(x \mid y)$
- D is trained on (x, y) pairs, G gets (z, y) as inputs
- Useful to obtain a single generator object for all y configurations
- Can be used to interpolate between distributions



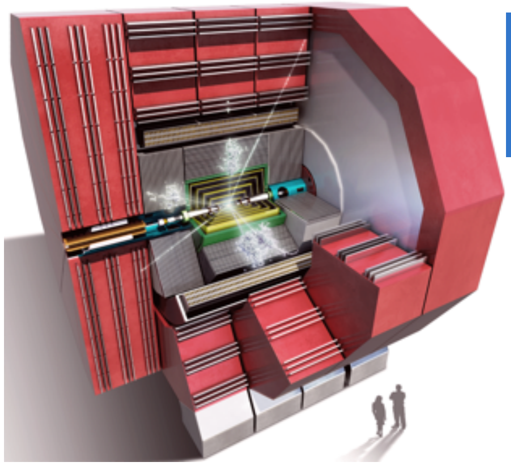
arXiv:1411.1784v

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

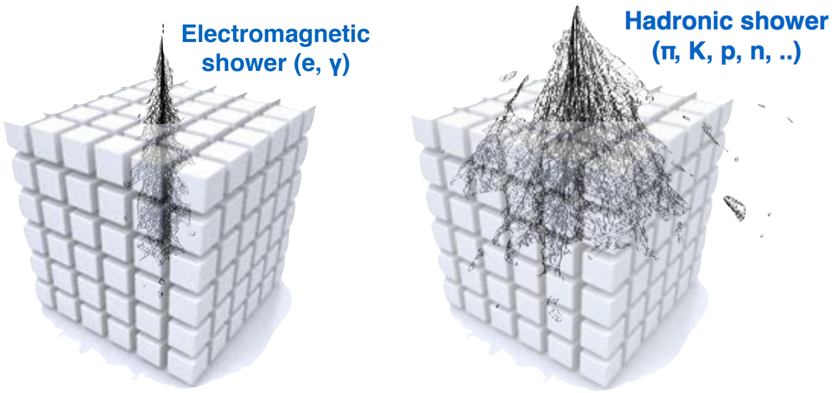
$$\Downarrow$$

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_{z}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

LCD calorimeter

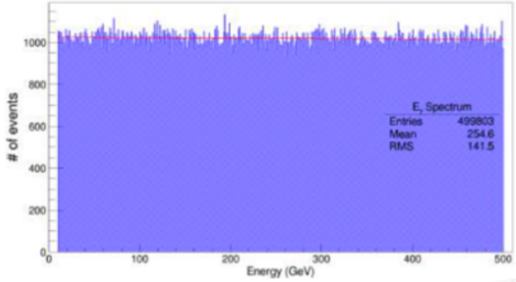


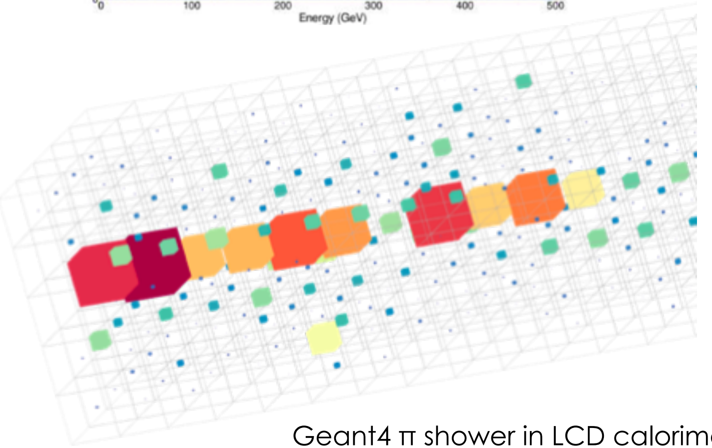
- Using as a benchmark the LCD detector design
- Accessible beyond the boundaries of different experiments
- Example of next-generation highly granular detector
- FullSIM available out of the box
- Simple calorimeter geometry with uniform cell sizes

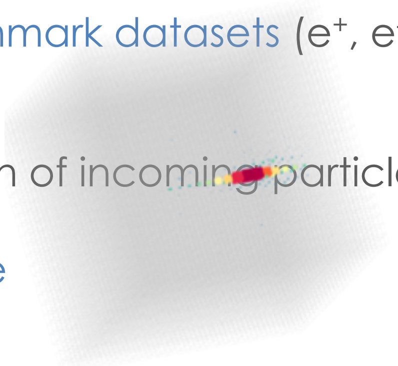


LCD calorimeter

- ECAL (1.5 m inner radius, 5 mm×5 mm segmentation)
  - 25 tungsten absorber layers + silicon sensors
- HCAL (3.0 cm×3.0 cm segmentation)
  - 60 steel absorber layers + polystyrene scintillators
- Defined single-particle benchmark datasets (e^+, e^-, γ, π)
  - Uniform energy distribution of incoming particles
- Data is essentially a 3D image





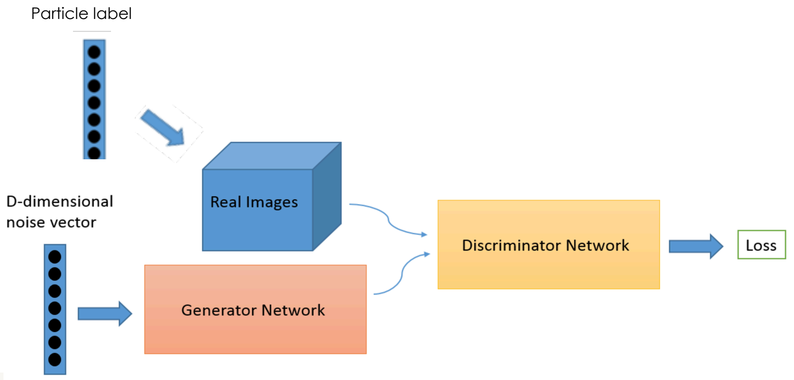


Geant4 π shower in LCD calorimeter

https://github.com/tpmccauley/ispy-hepml

3dGAN for calorimeter images

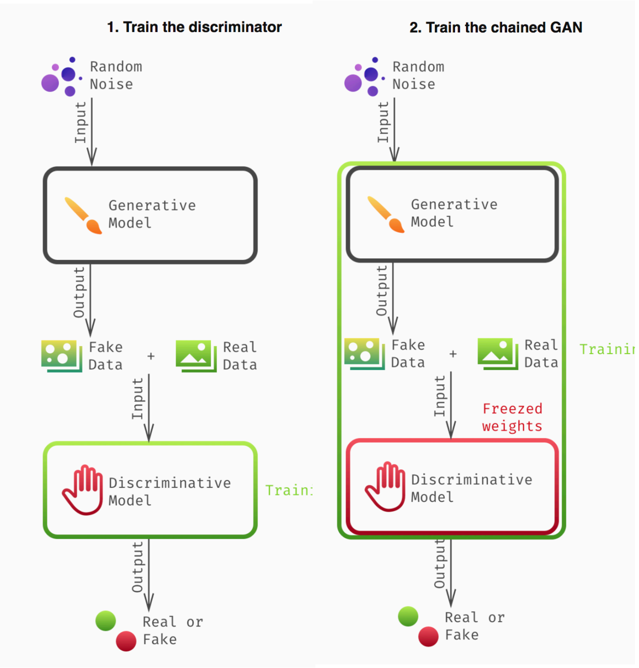
- Similar discriminator and generator models
  - 3D conv layers with different x,y,z filter sizes
  - Particle tag as auxiliary classifier
- Implemented tips&tricks found in literature
  - Some helpful (no batch normalisation in the last step, LeakyRelu, no hidden dense layers, no pooling layers)
  - Some not (Adam optimiser)
- Batch training
- Loss is combined cross entropy



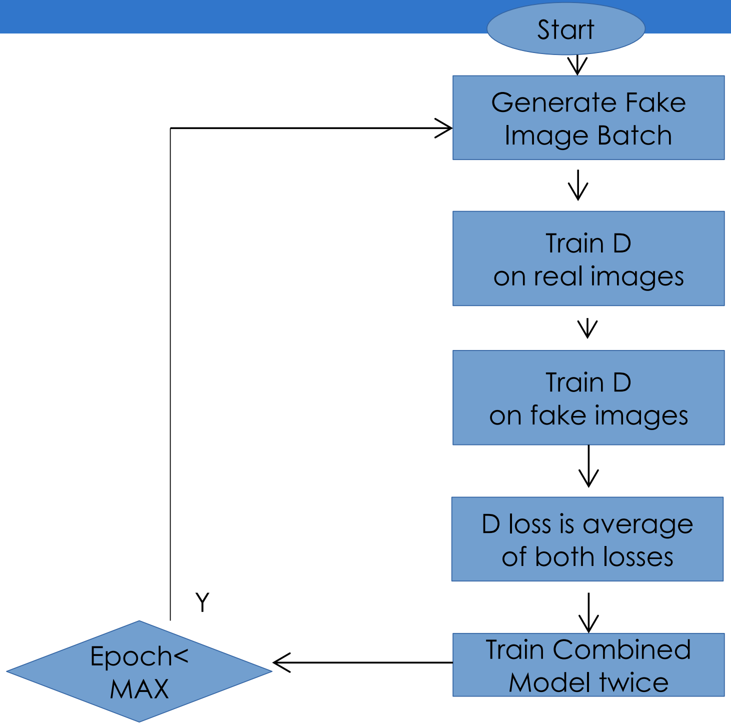
https://github.com/soumith/ganhacks

Training process

"Vanilla" GAN



3dGAN

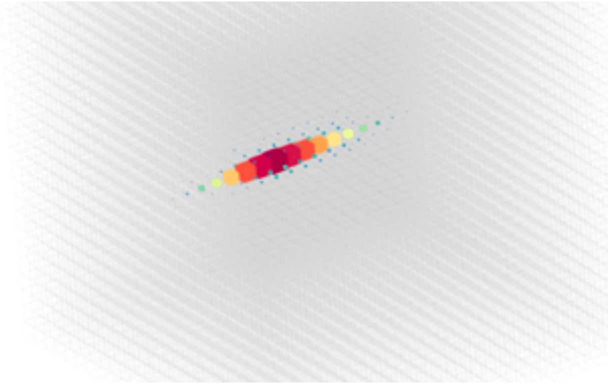
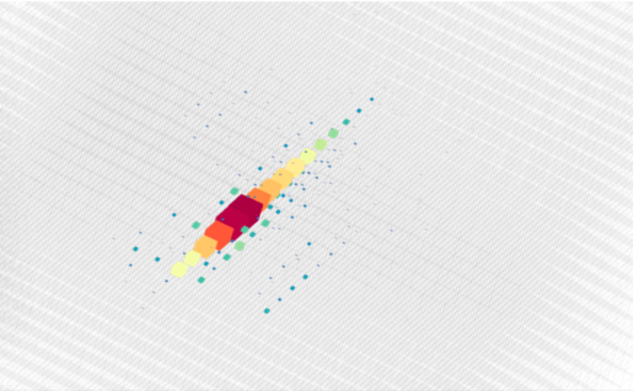
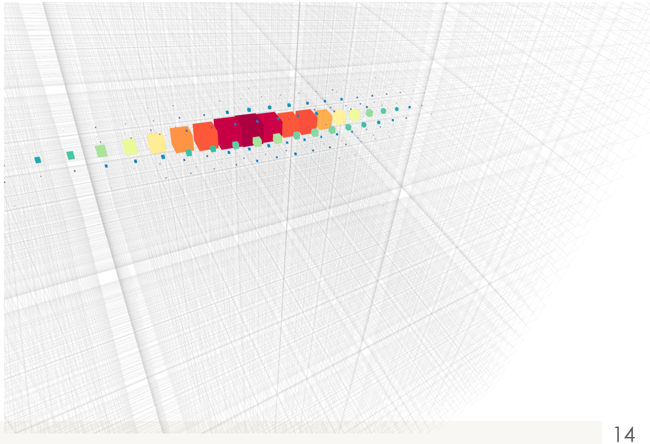


Training time is long: ~45 mins on GeForce GTX 1080 for 1 epoch

Preliminary

Some images

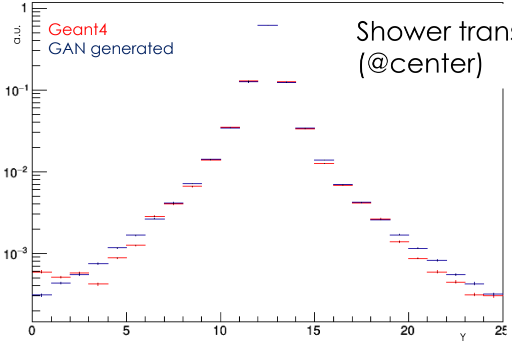
- Slice energy spectrum
- Start with photons & electrons in EM calorimeter
  - Use particle tag to condition training

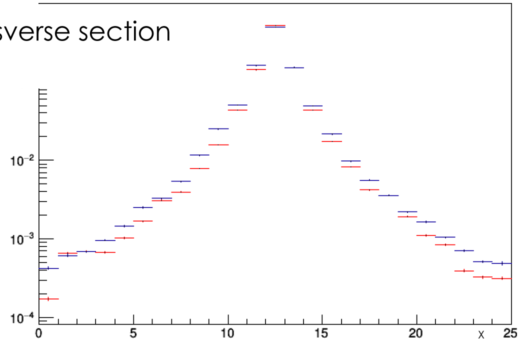





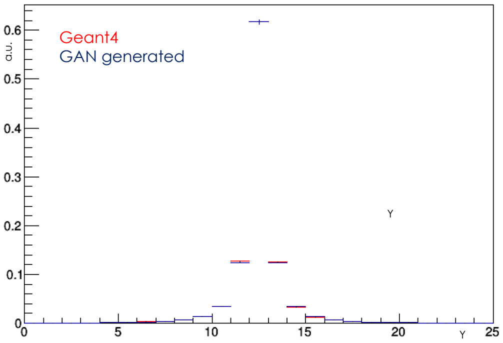
Preliminary

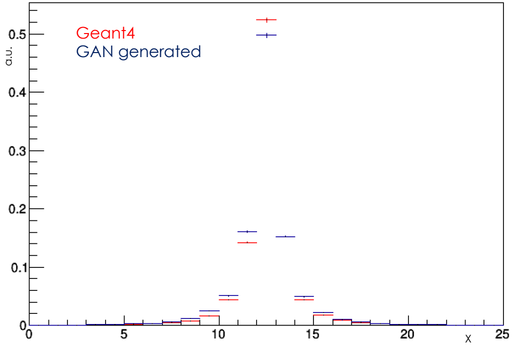
GAN generated electrons

Shower transverse section (@center)

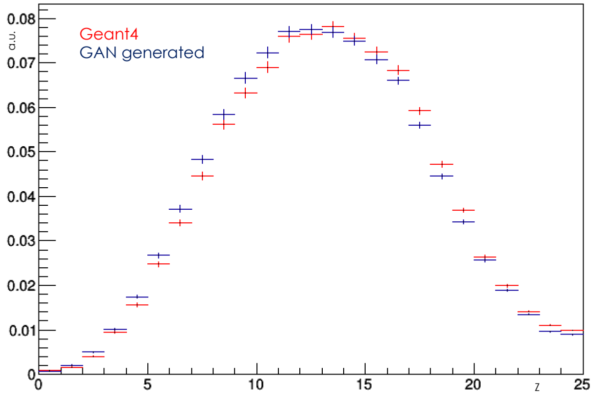






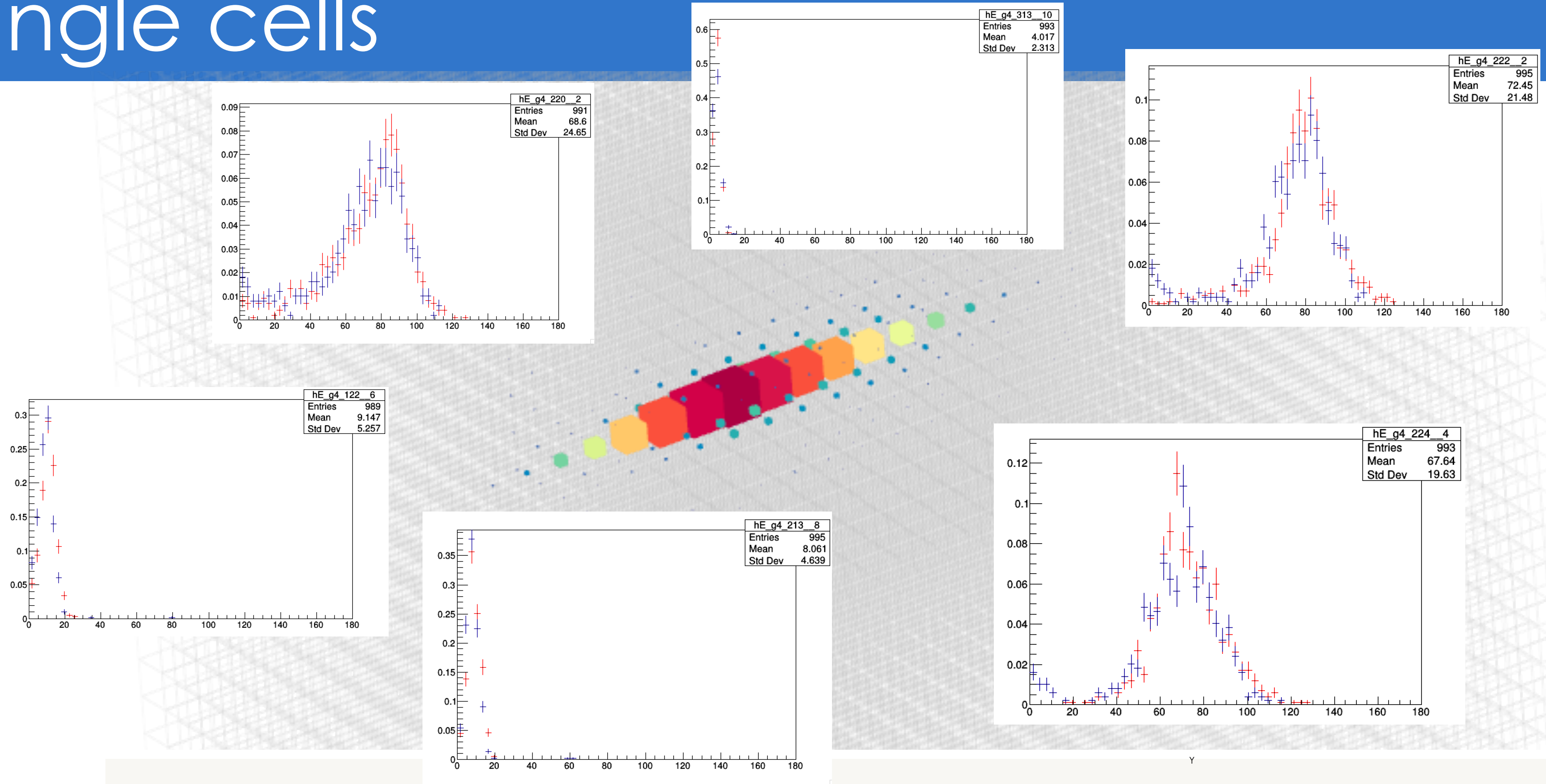


Shower longitudinal section



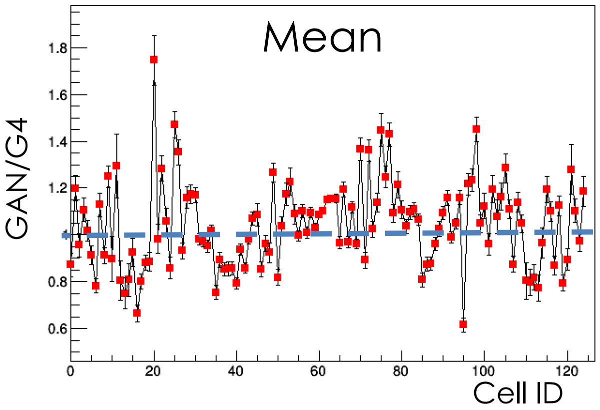
Preliminary

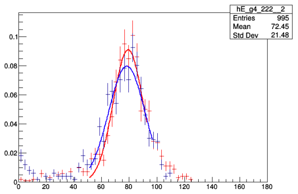
Single cells

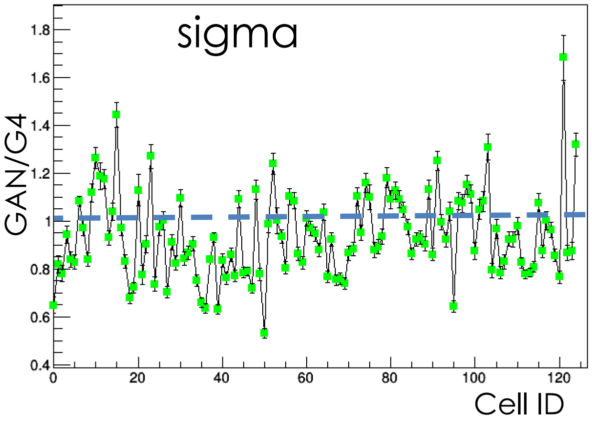


Preliminary

Single cells: energy mean and sigma





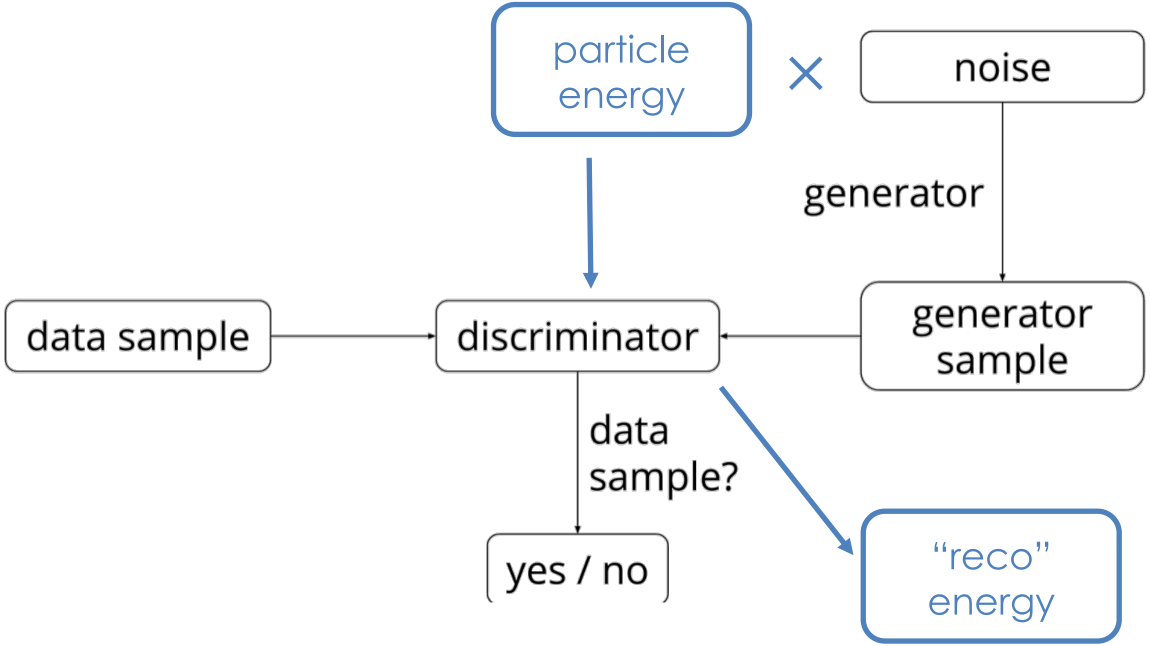


- Cell energy sigma is underestimated by GAN
- Set up higher level criteria for image validation (reconstructed variables)
- Check uncertainty due to training sample statistics

Conditioning on energy

Shruti Sharan

- Training the generator and the discriminator using initial particle energy
  - Initially discrete bins to test interpolation and extrapolation
  - Then test continuous spectrum



Parallel training

Gulrukh Khattak

- Study scaling on KNL cluster @CINECA. (IPCC framework)
  - Tensorflow does not scale to multinodes (no MPI implementation)
  - Migrate code to Intel Caffe
    - Our GAN implementation in Caffe is not straightforward (model and training)
  - Intel will release a completely new software stack (Nervana) → much more flexible
- Parallel test on GPU cluster
  - Collaboration with experts from the experiments (A. Farbin - ATLAS and M. Pierini -CMS)

Conclusion and plans

- First GAN application looks very promising
- Working on understanding and improving performance
  - Training sample statistics
  - Adding important features (energy)
  - Studying scaling
  - Try hyper-parameter scan to improve network design (an openlab summer student will join in July)

Adapted from P. Balaprakash