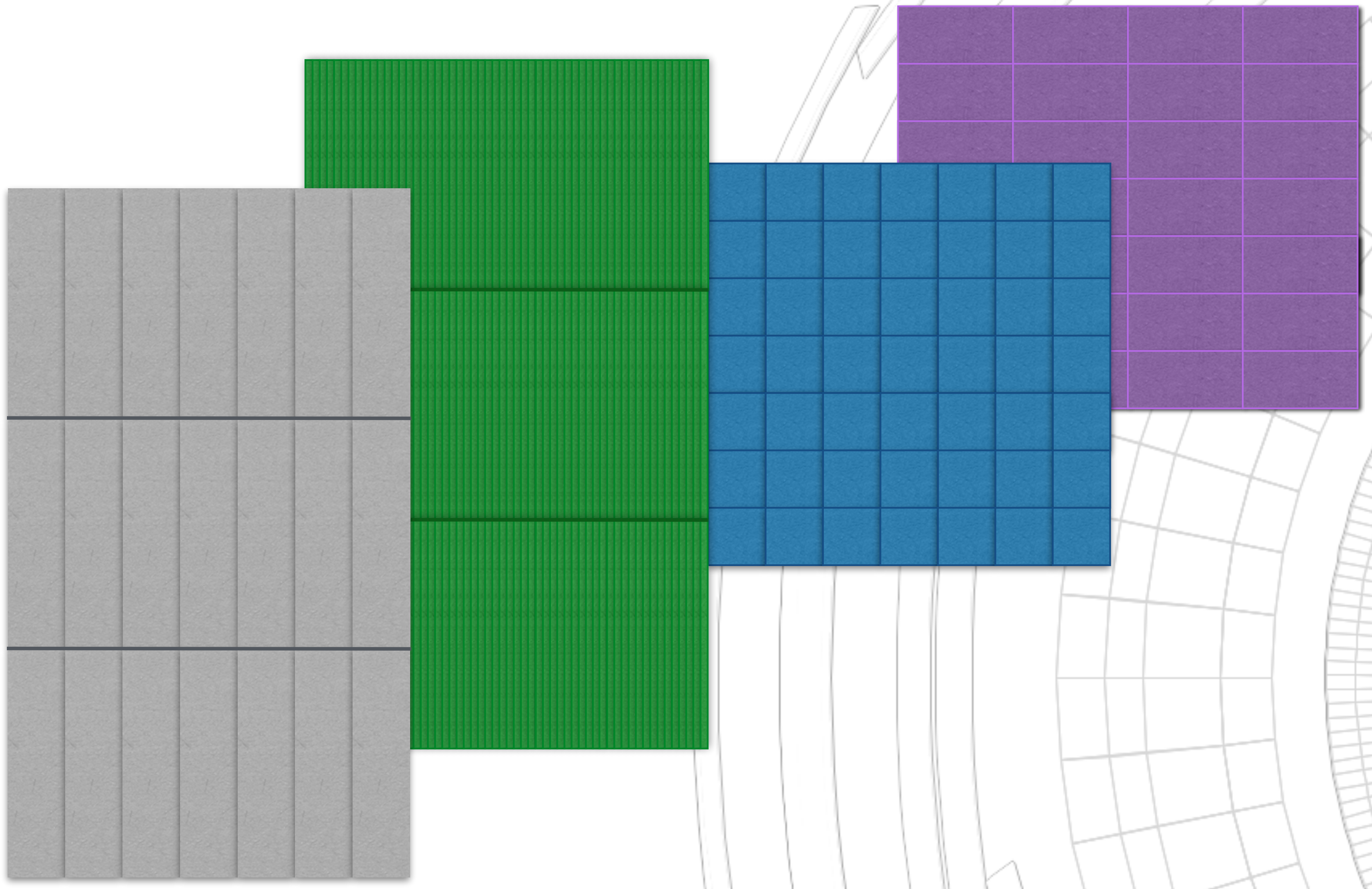
Generative Adversarial Networks for Fast Shower Simulation in ATLAS

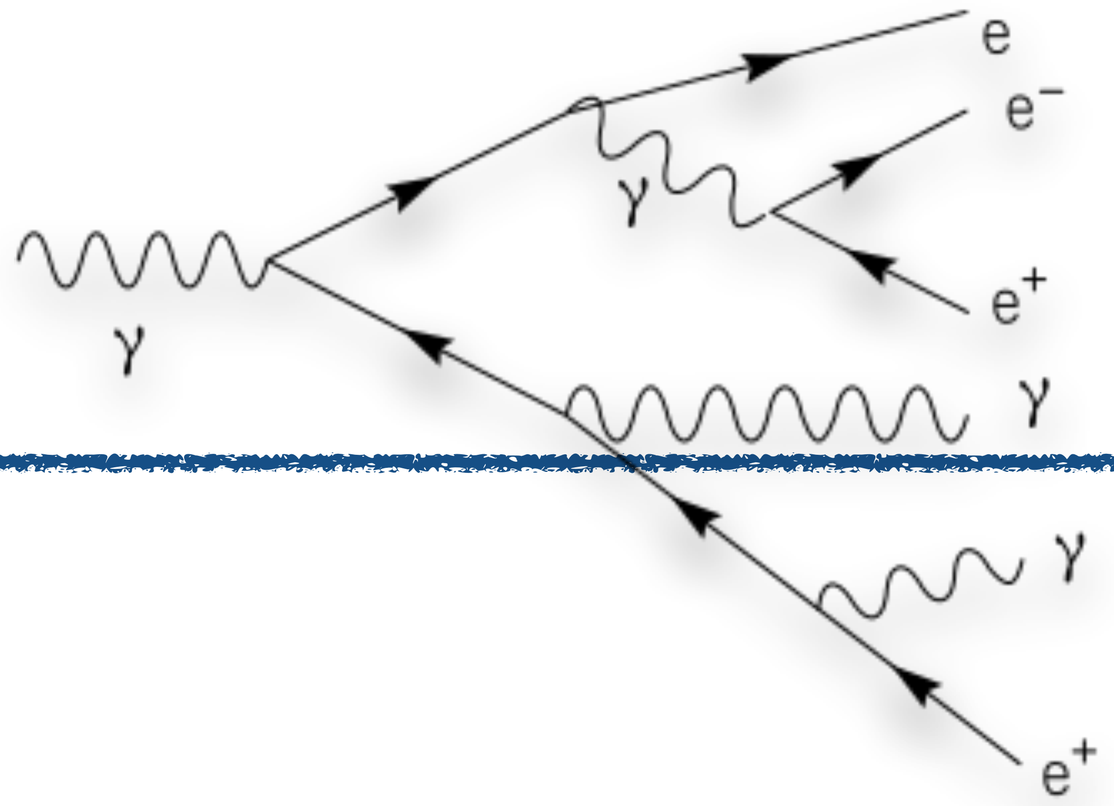
Aishik Ghosh, David Rousseau

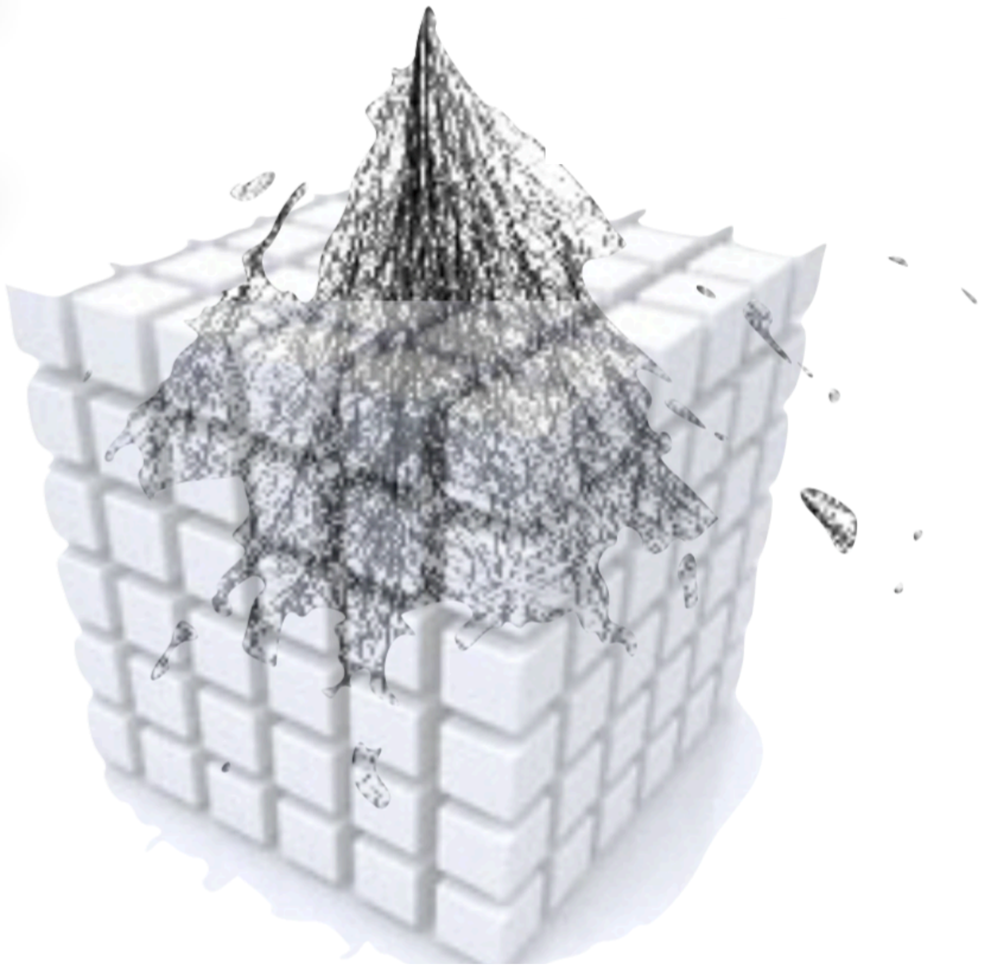
IN2P3/IRFU ML Workshop
22 January 2019

IJCLab



Motivation

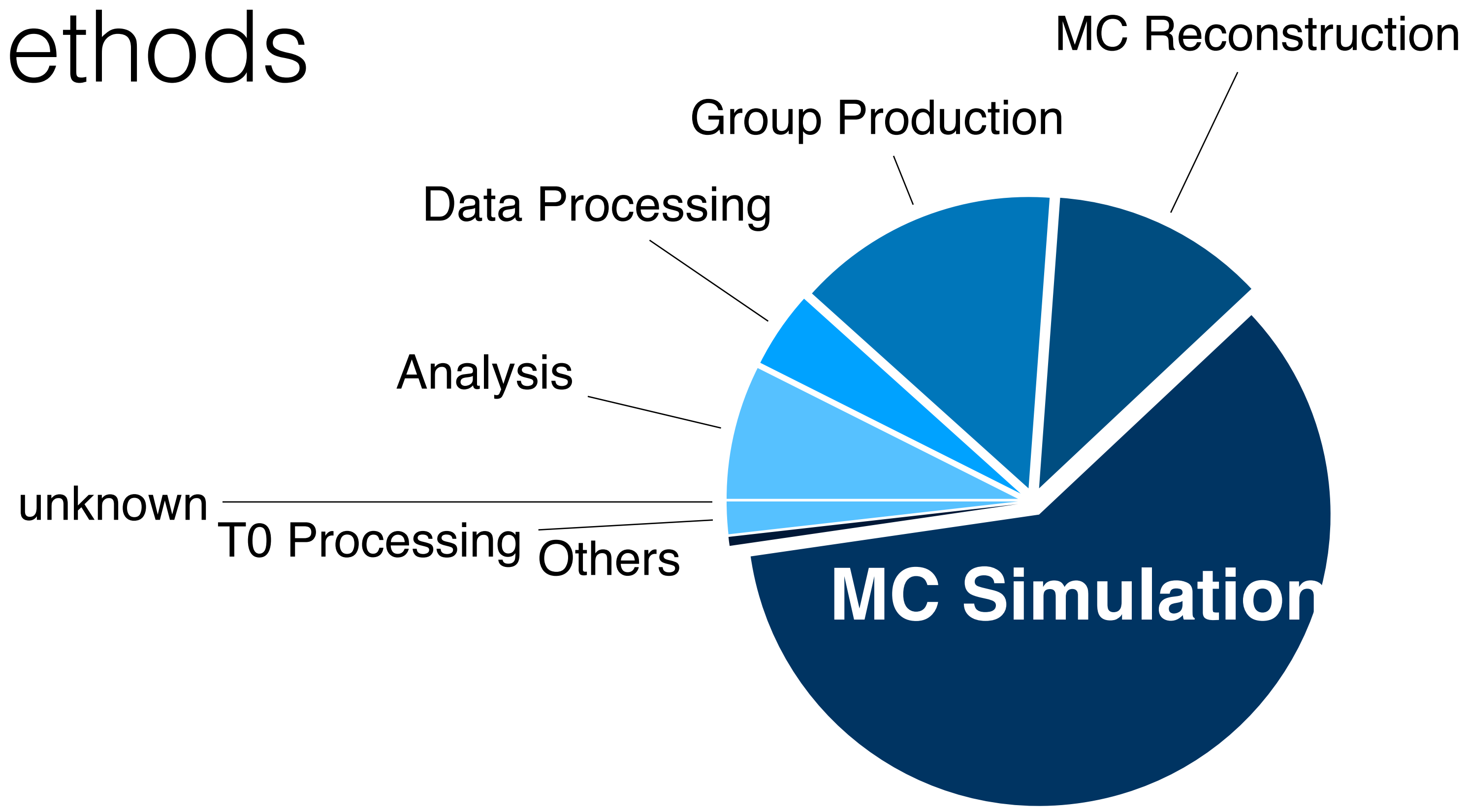


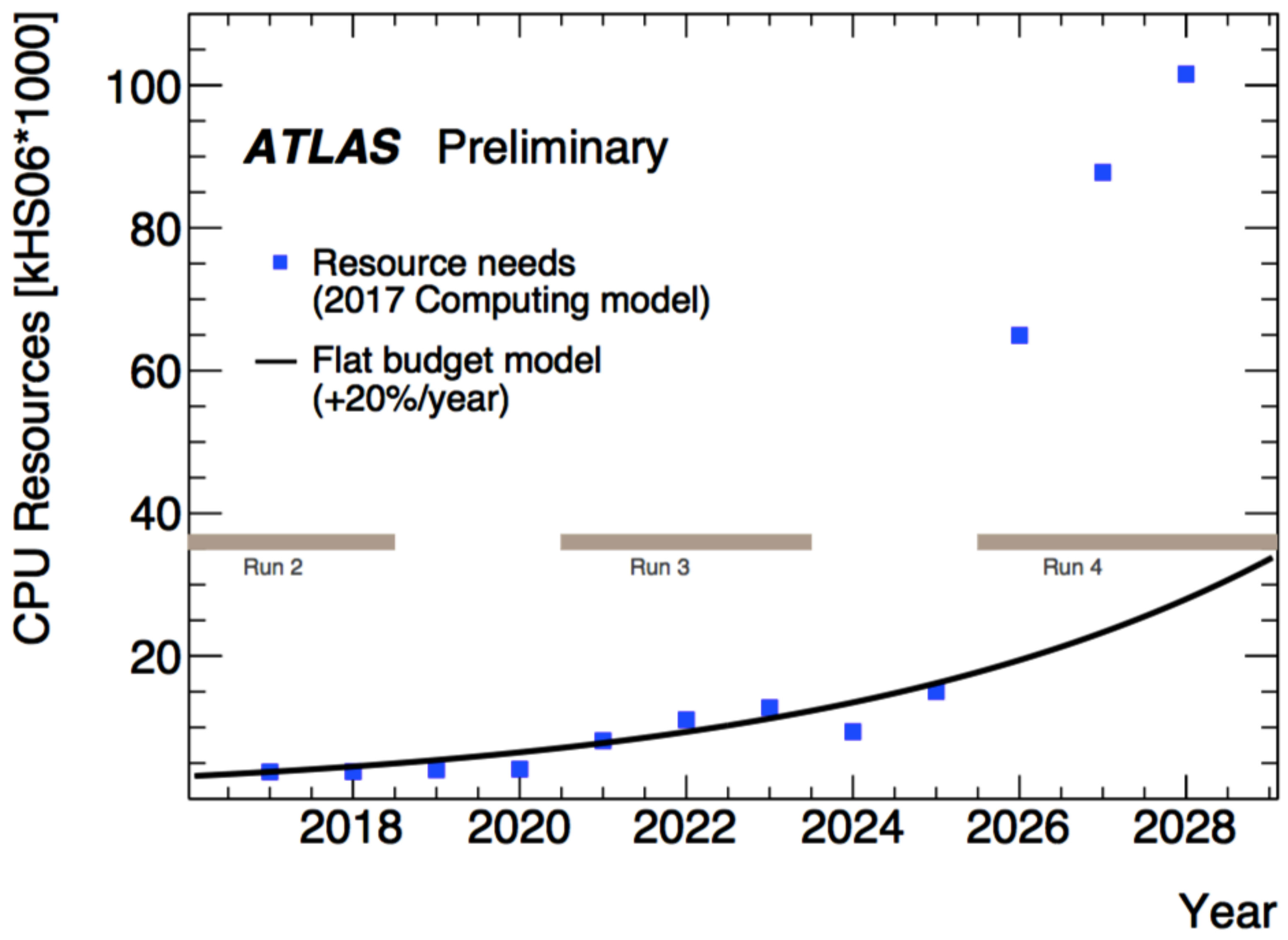


Simulate showers 100-1000x *faster* than Geant4

Less human time intensive, *higher accuracy* than current fast simulation methods

Have it run inside ATLAS C++ software and be *less resource* hungry than current fast simulation methods



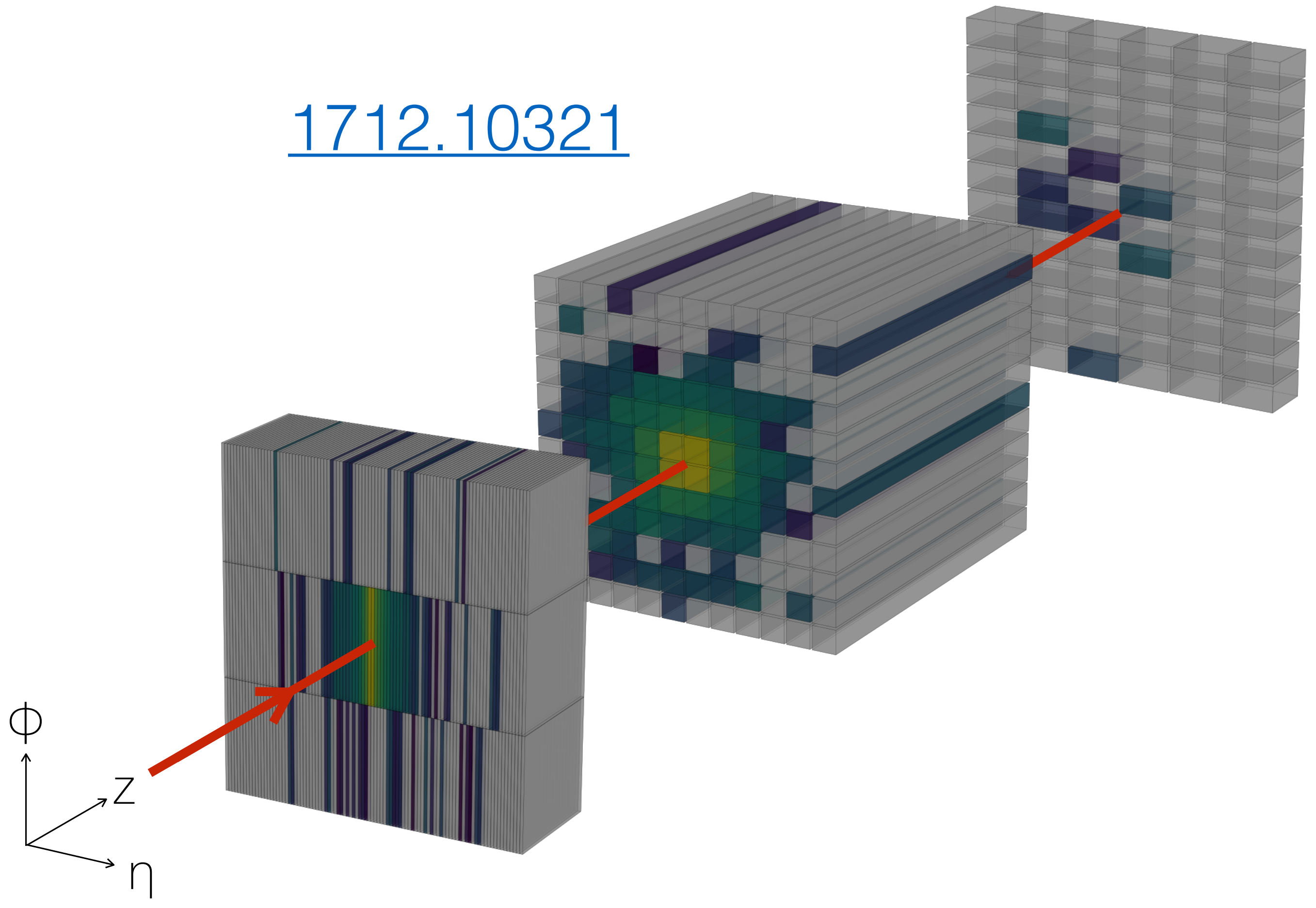


Imperative to develop fast accurate shower simulations

Generative Models for EM Shower Simulation

CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks

1712.10321

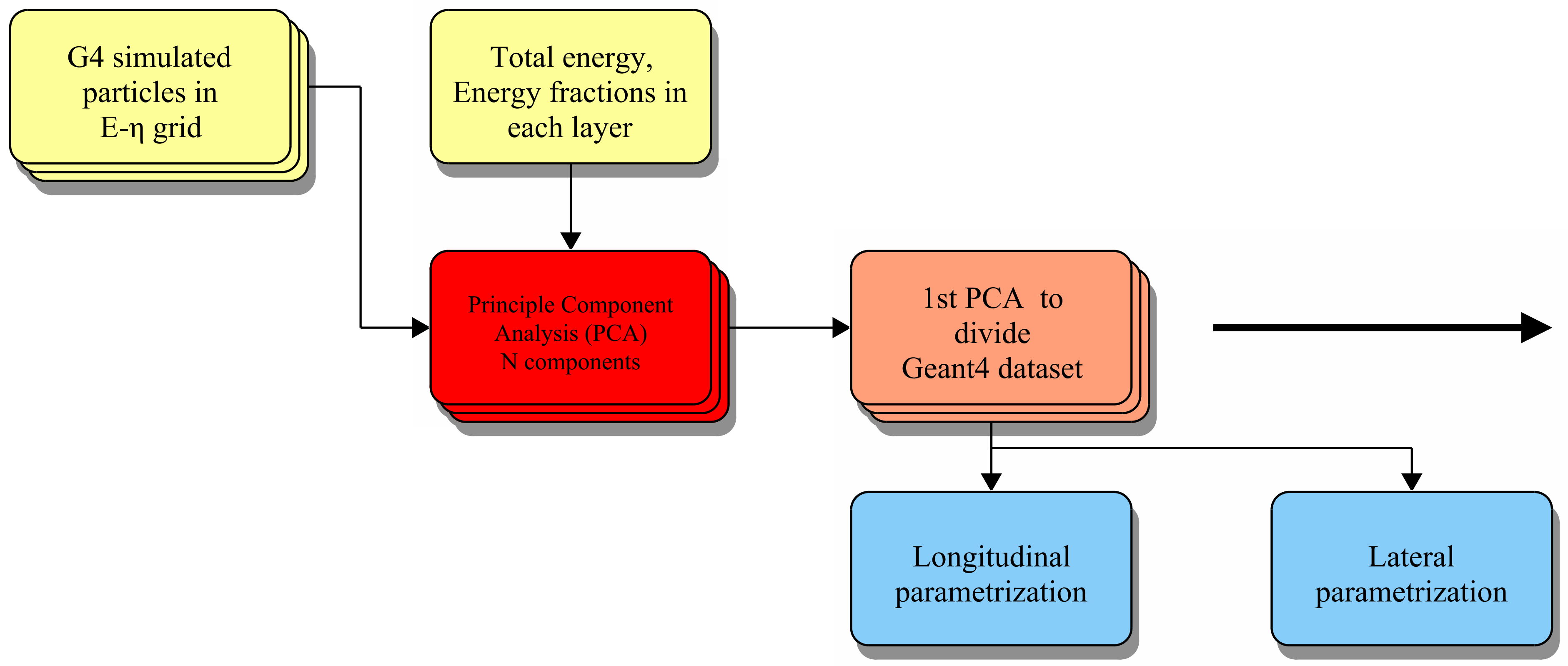


- CaloGAN showed that it is possible to simulate EM showers for a detector like ATLAS using GANs
- Since then we've seen many GANs for particle physics
- ATLAS calorimeter more complicated than CMS, strange geometry compared to high granularity future :<u>major simulation bottleneck!</u>

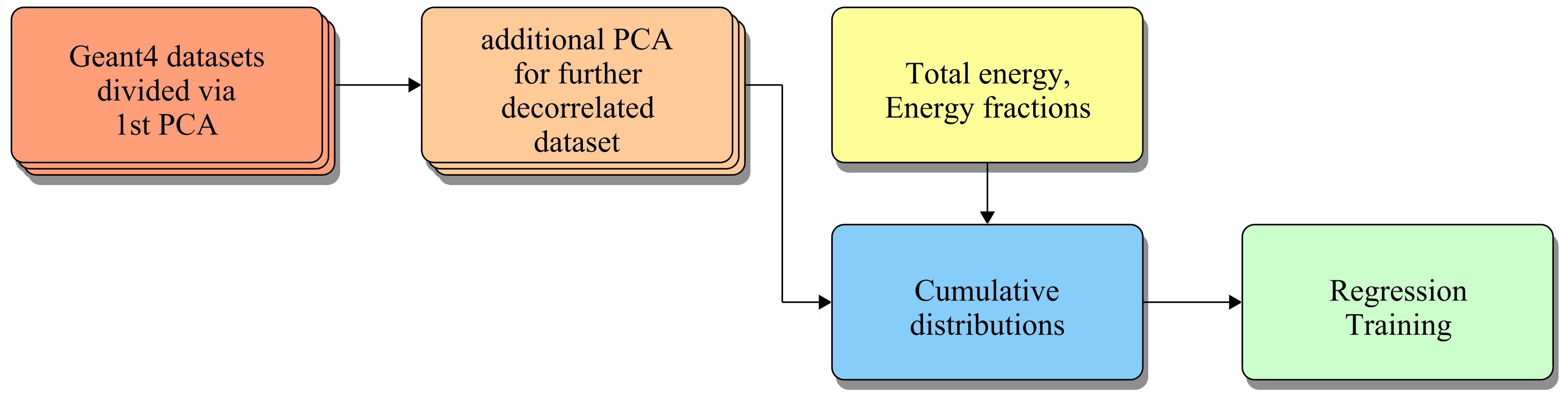
Baseline: FastCaloSim

Human designed parameterisation techniques being developed over many years -> A **high benchmark** for GAN / VAE

FastCaloSim V2 already using Machine Learning at various points in the chain



Additional PCA transformation to further decorrelation



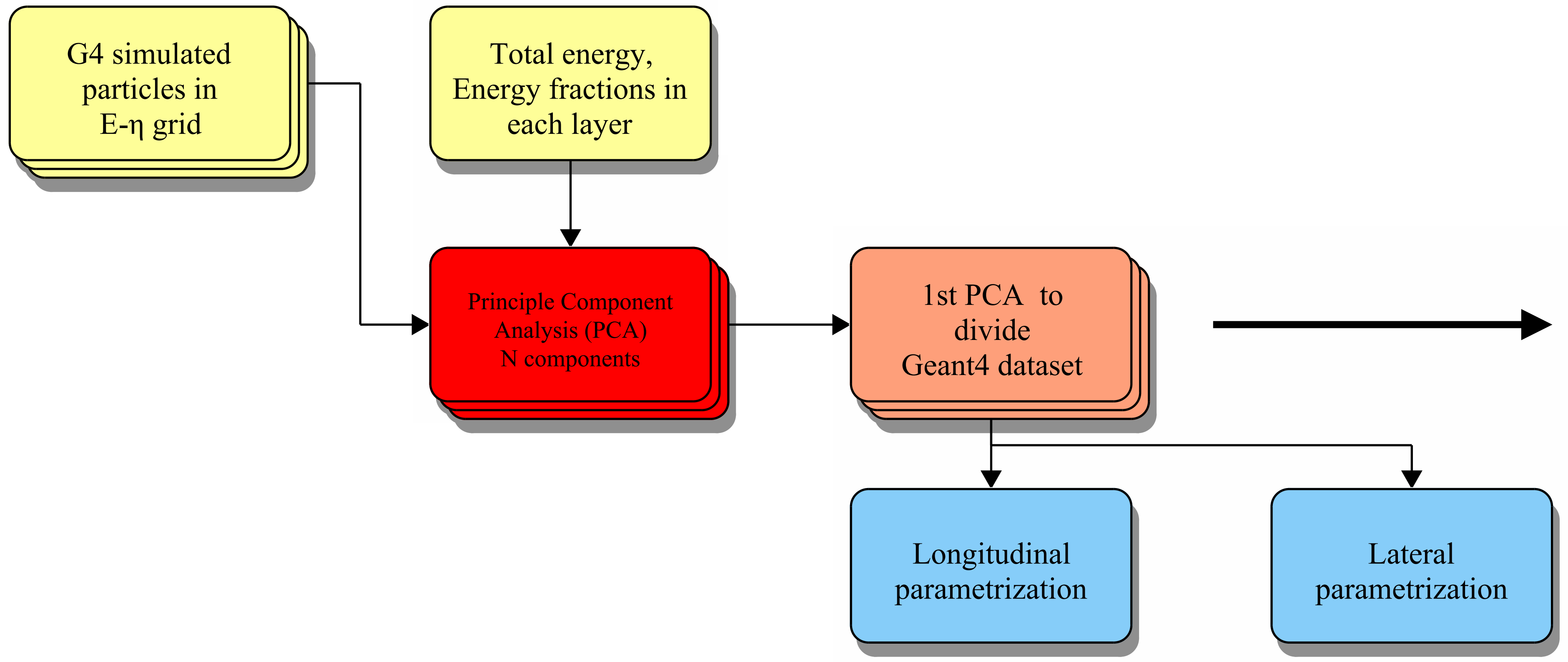
Done separately for each η slices, energy

additional interpolation mechanism

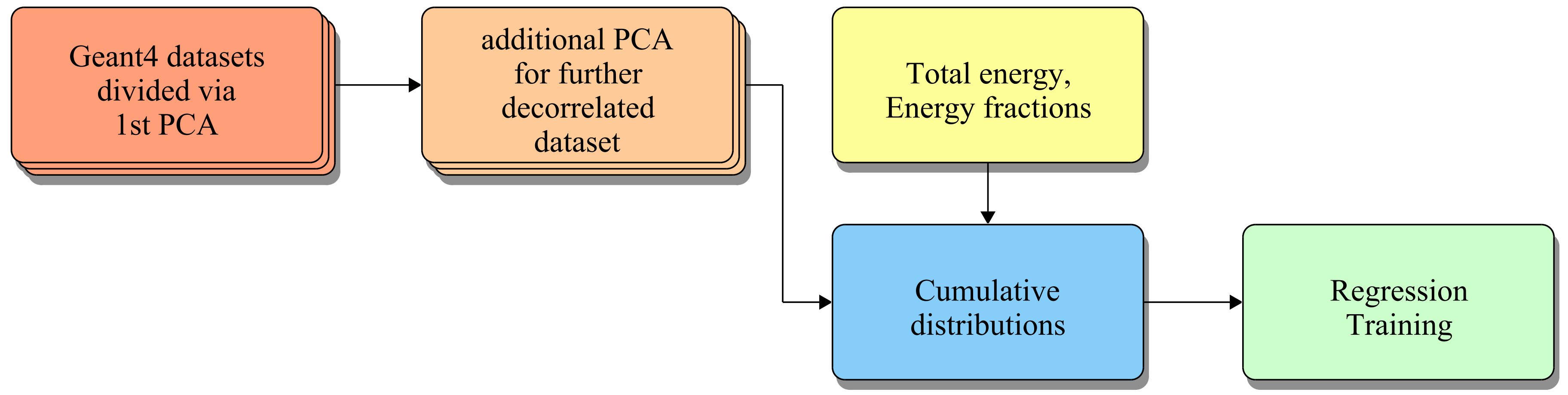
Baseline: FastCaloSim

Human designed parameterisation techniques being developed over many years -> A **high benchmark** for GAN / VAE

FastCaloSim V2 already using Machine Learning at various points in the chain



Additional PCA transformation to further decorrelation



"GAN" this

Done separately for each η slices, energy

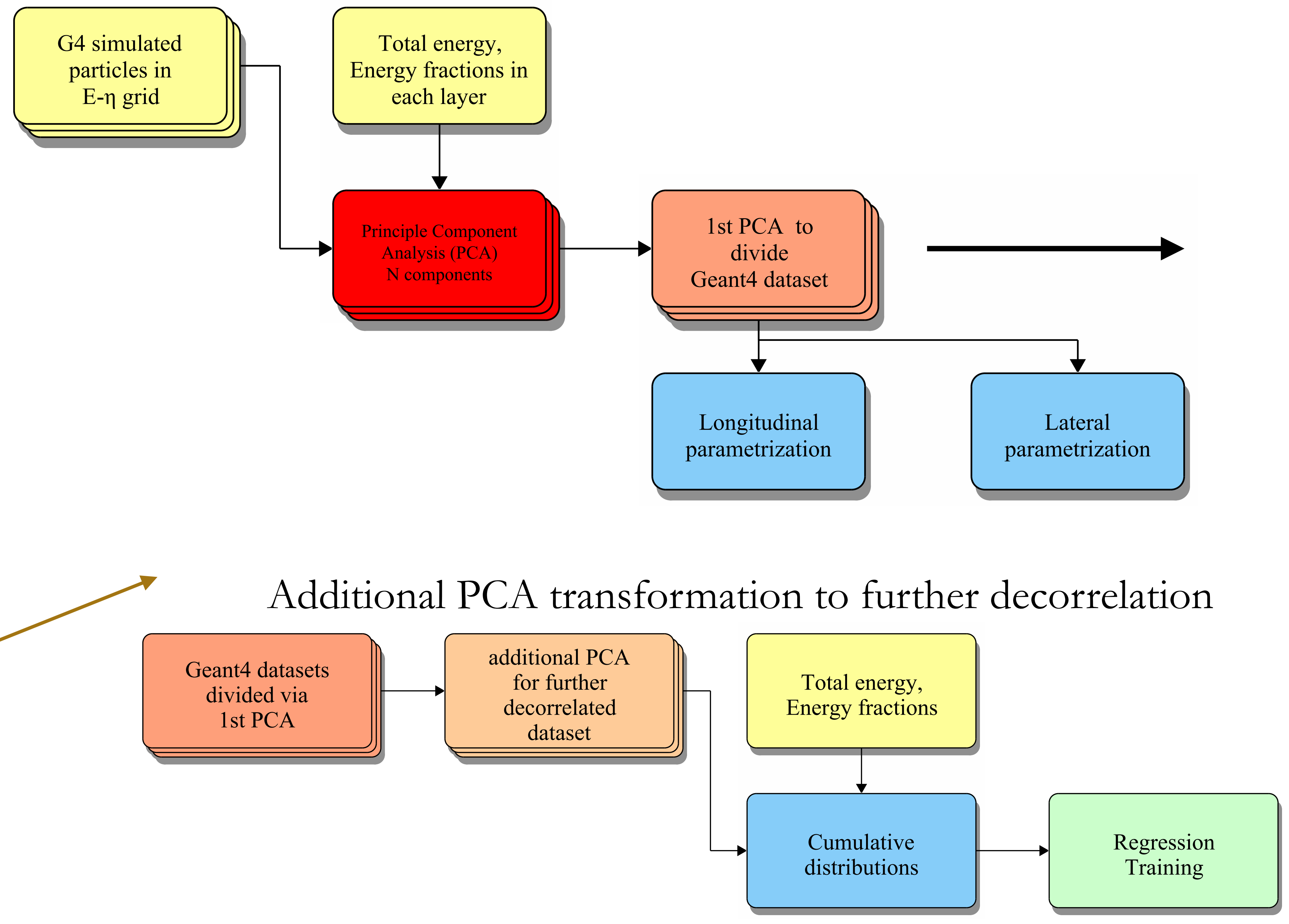
additional interpolation mechanism

Baseline: FastCaloSim

Human designed parameterisation techniques being developed over many years -> A **high benchmark** for GAN / VAE

Validation with “EGamma” group defined **list of complex variables**

FastCaloSim V2 already using Machine Learning at various points in the chain



“GAN” this

Done separately for each η slices, energy

additional interpolation mechanism

Baseline: FastCaloSim

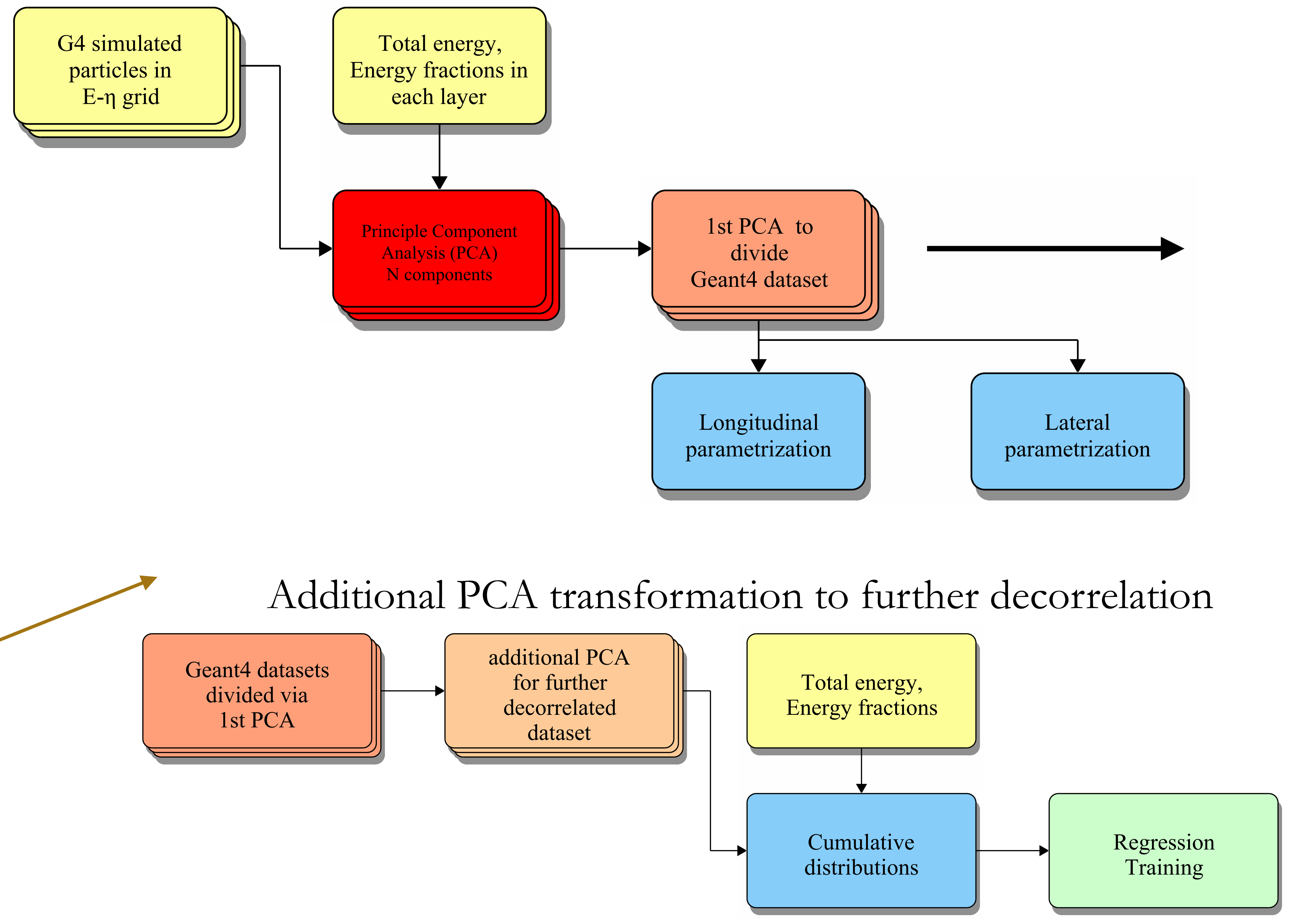
Human designed parameterisation techniques being developed over many years -> A **high benchmark** for GAN / VAE

Validation with "EGamma" group defined **list of complex variables**

Validation cross-check frameworks already in place for FastCaloSim: same level of scrutiny for all fast simulation approaches.

Need to get all distributions right simultaneously, **average distributions might look right** but must **verify also the distributions per energy point / section of the calorimeter**

FastCaloSim V2 already using Machine Learning at various points in the chain



"GAN" this

Done separately for each η slices, energy

additional interpolation mechanism

GAN research moving towards better quality images

that some features are not represented such as the cigarette in the left image.

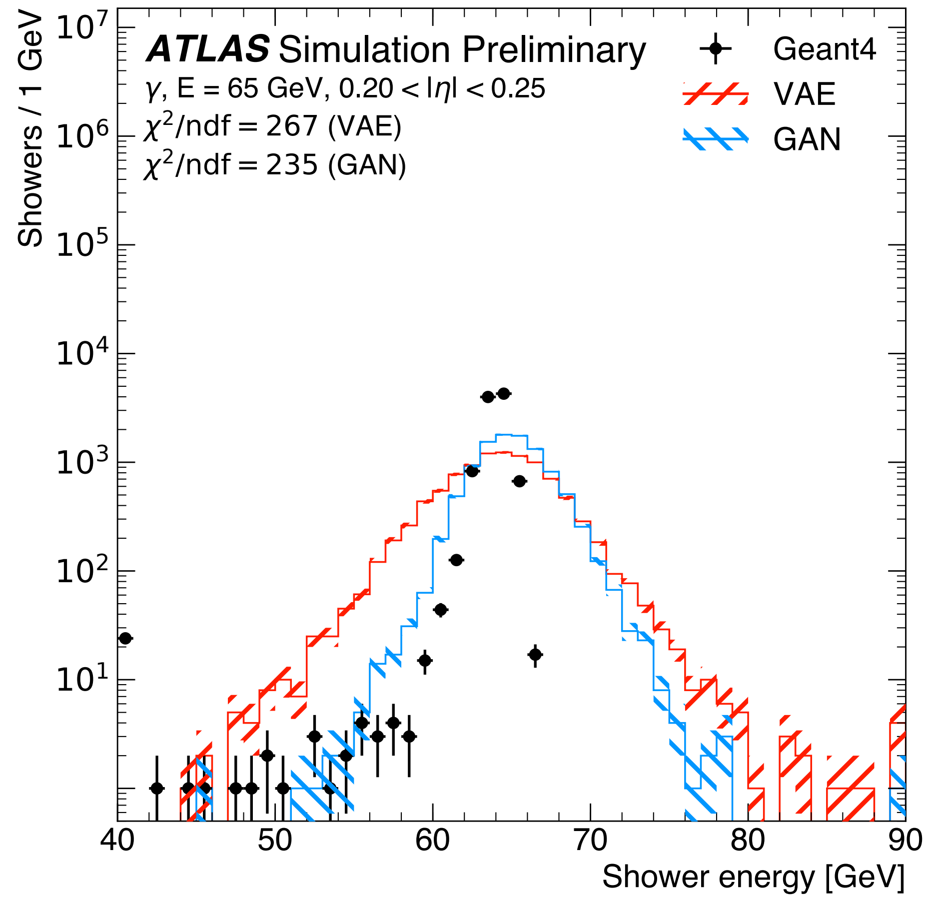
(BE)GAN seems to produce more attractive faces than in training dataset

We observe varied poses, expressions, genders, skin colors, light exposure, and facial hair. However we did not see glasses, we see few older people and there are more women than men. For comparison

Generative Adversarial Networks

GAN research moving towards better quality images

But probability densities are another thing

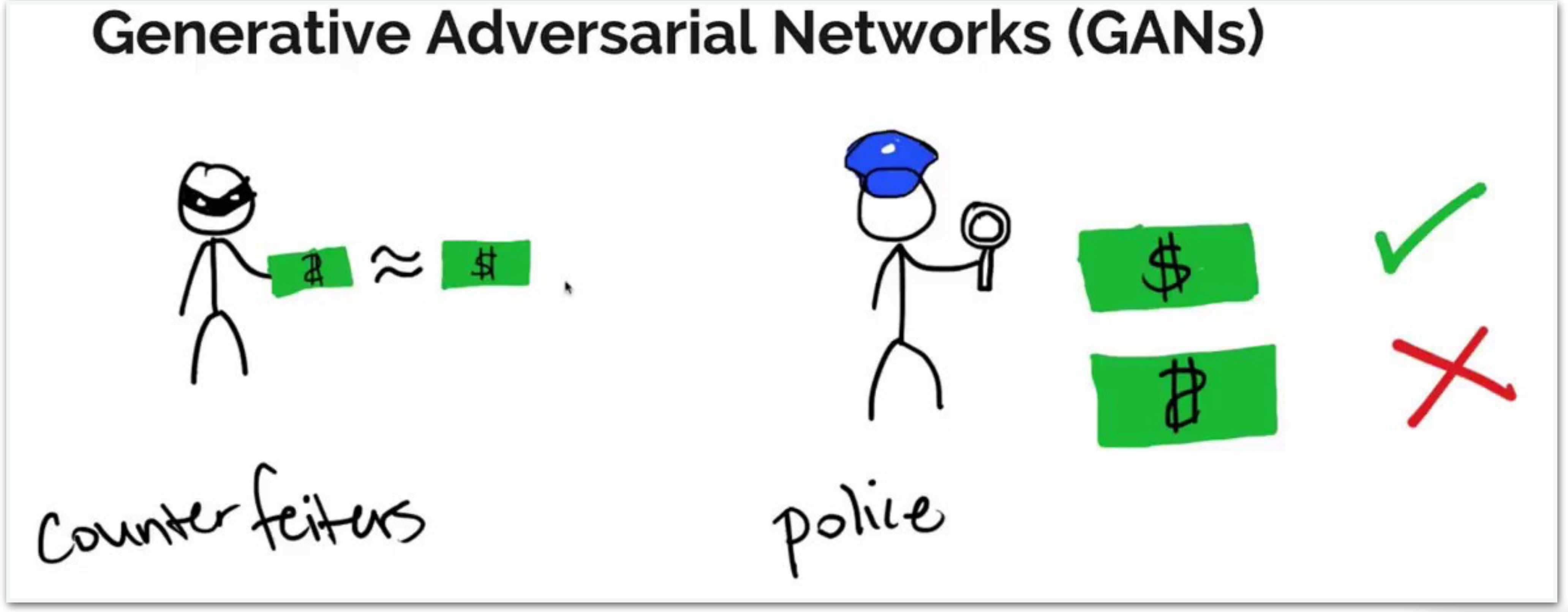


that some features are not represented such as the cigarette in the left image.

(BE)GAN seems to produce more attractive faces than in training dataset

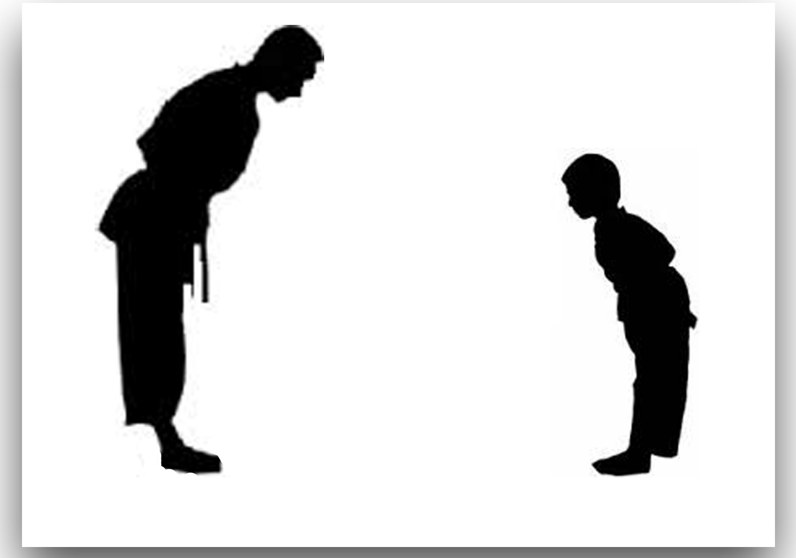
We observe varied poses, expressions, genders, skin colors, light exposure, and facial hair. However we did not see glasses, we see few older people and there are more women than men. For comparison

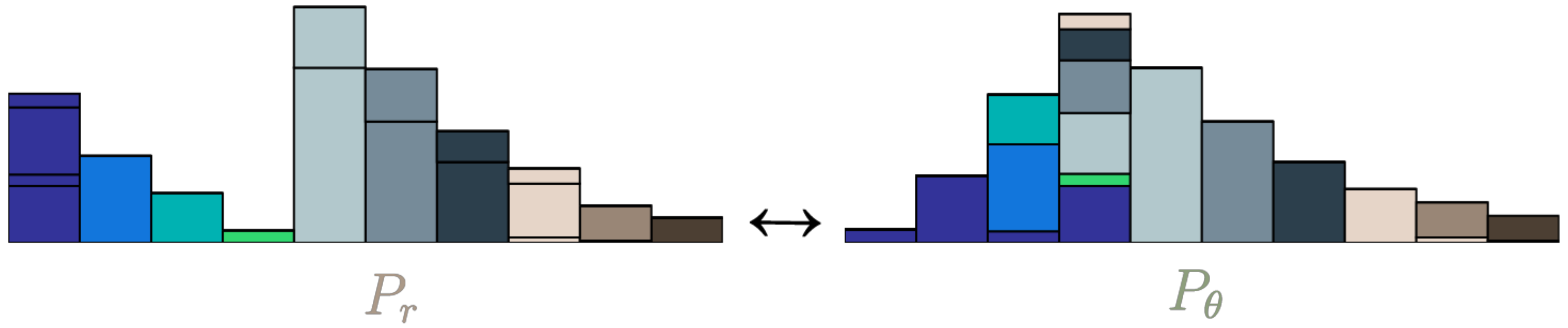
Wasserstein GAN with Gradient Penalty



- Stable GAN training, no vanishing grads, no mode collapse
- Long training time
- Other GAN favours were tried

iWasserstein GANs:
Gradient Penalty on Critic





$$L_{\mathrm{GAN}} = \mathop{E}_{\tilde{x}\sim p_{\mathrm{gen}}}[D(\tilde{x})] - \mathop{E}_{x\sim p_{\mathrm{Geant4}}}[D(x)] + \lambda \mathop{E}_{\hat{x}\sim p_{\hat{x}}}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

The Calorimeter

2-D Axis: η vs ϕ

Particle goes through 4 layers in this order:

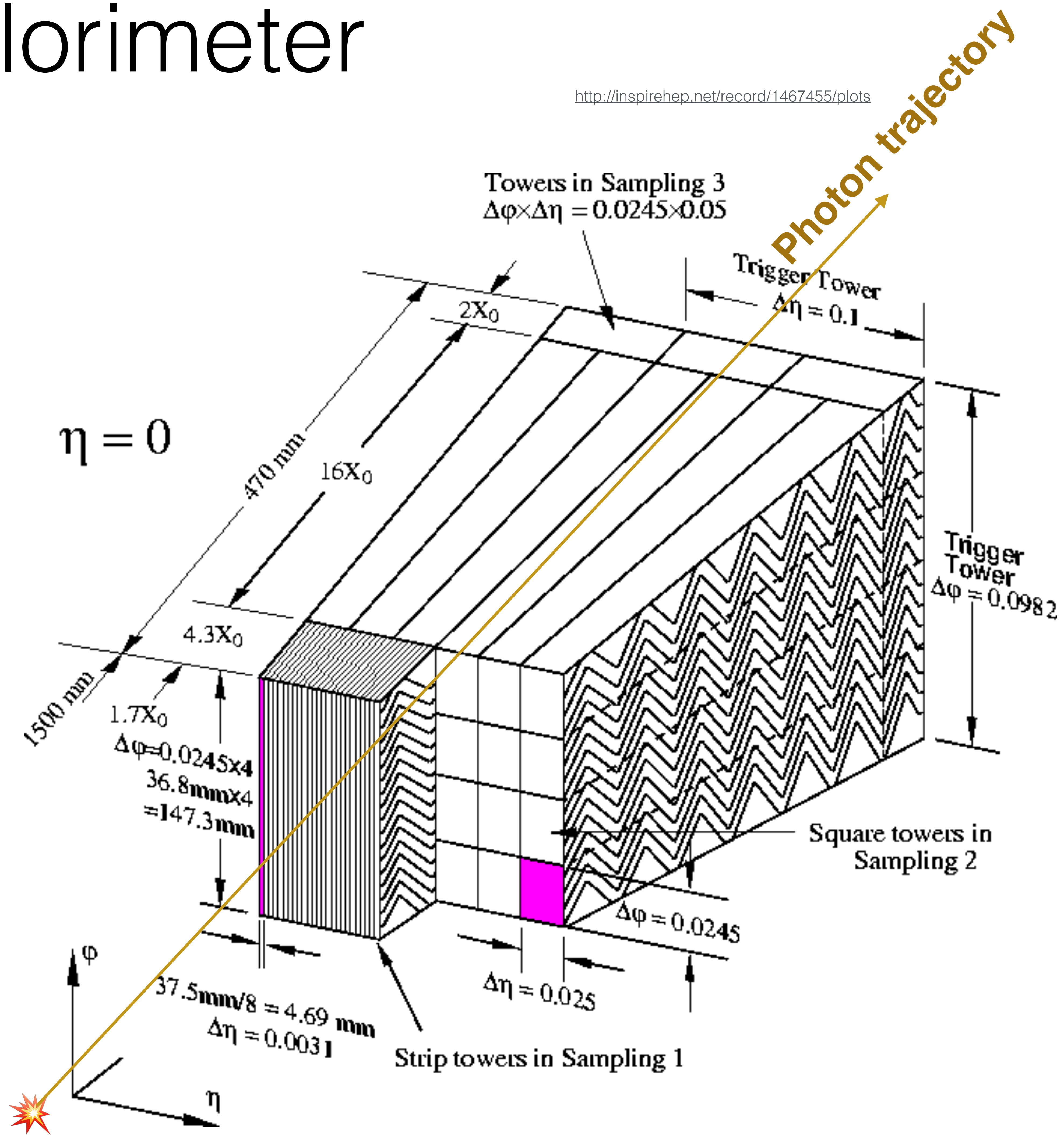
0. Pre-Sampler : (7x3) Some energy deposit

1. Strips: (56x3) Very granular in η; more energy deposit

2. Middle: (7x7) Thickest layer, maximum energy deposit

3. Back: (4x7) Little Energy deposits

http://inspirehep.net/record/1467455/plots



The Calorimeter

2-D Axis: η vs ϕ

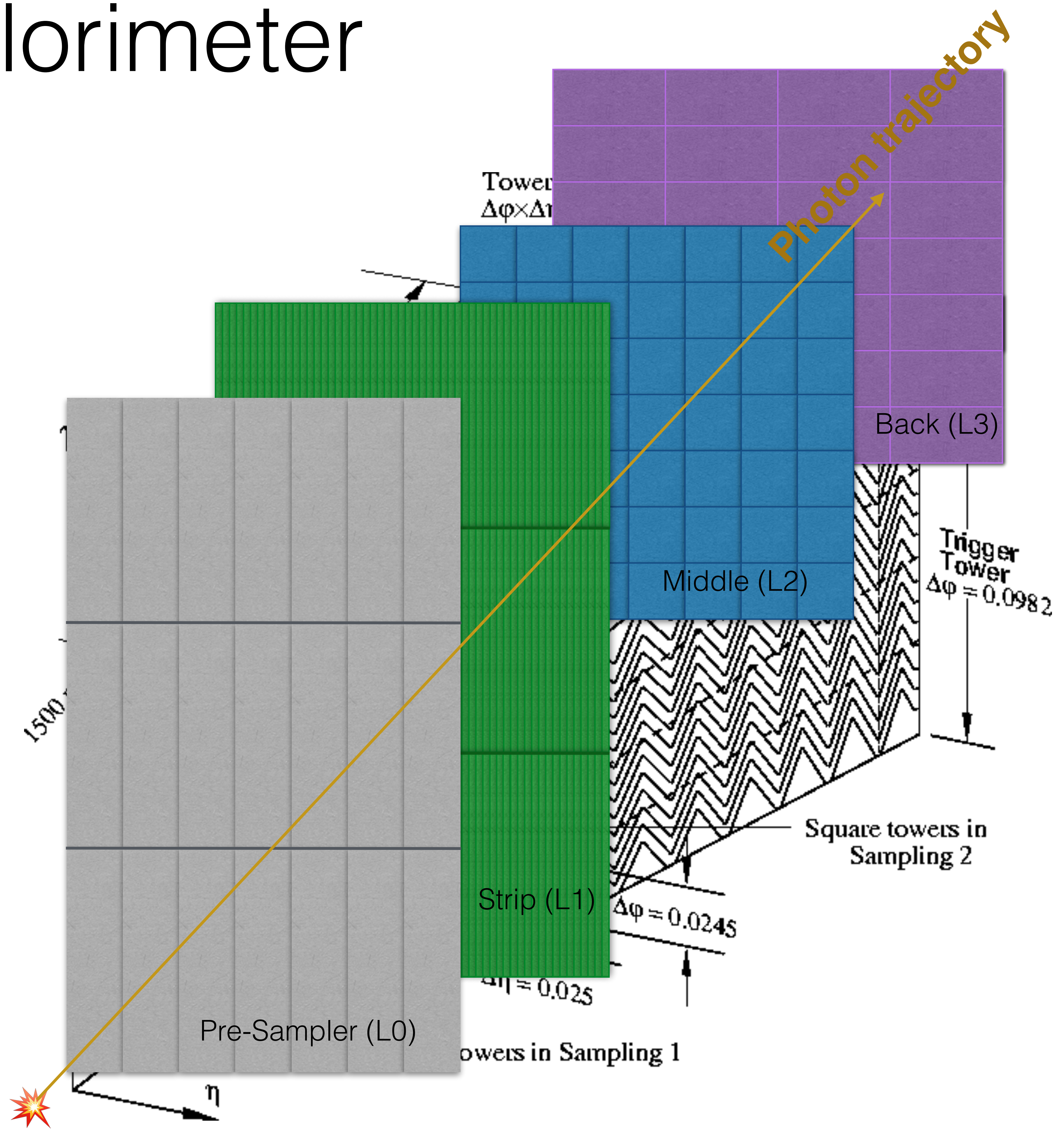
Particle goes through 4 layers in this order:

0. Pre-Sampler : (7x3) Some energy deposit

1. Strips: (56x3) Very granular in η; more energy deposit

2. Middle: (7x7) Thickest layer, maximum energy deposit

3. Back: (4x7) Little Energy deposits



The Calorimeter

2-D Axis: η vs ϕ

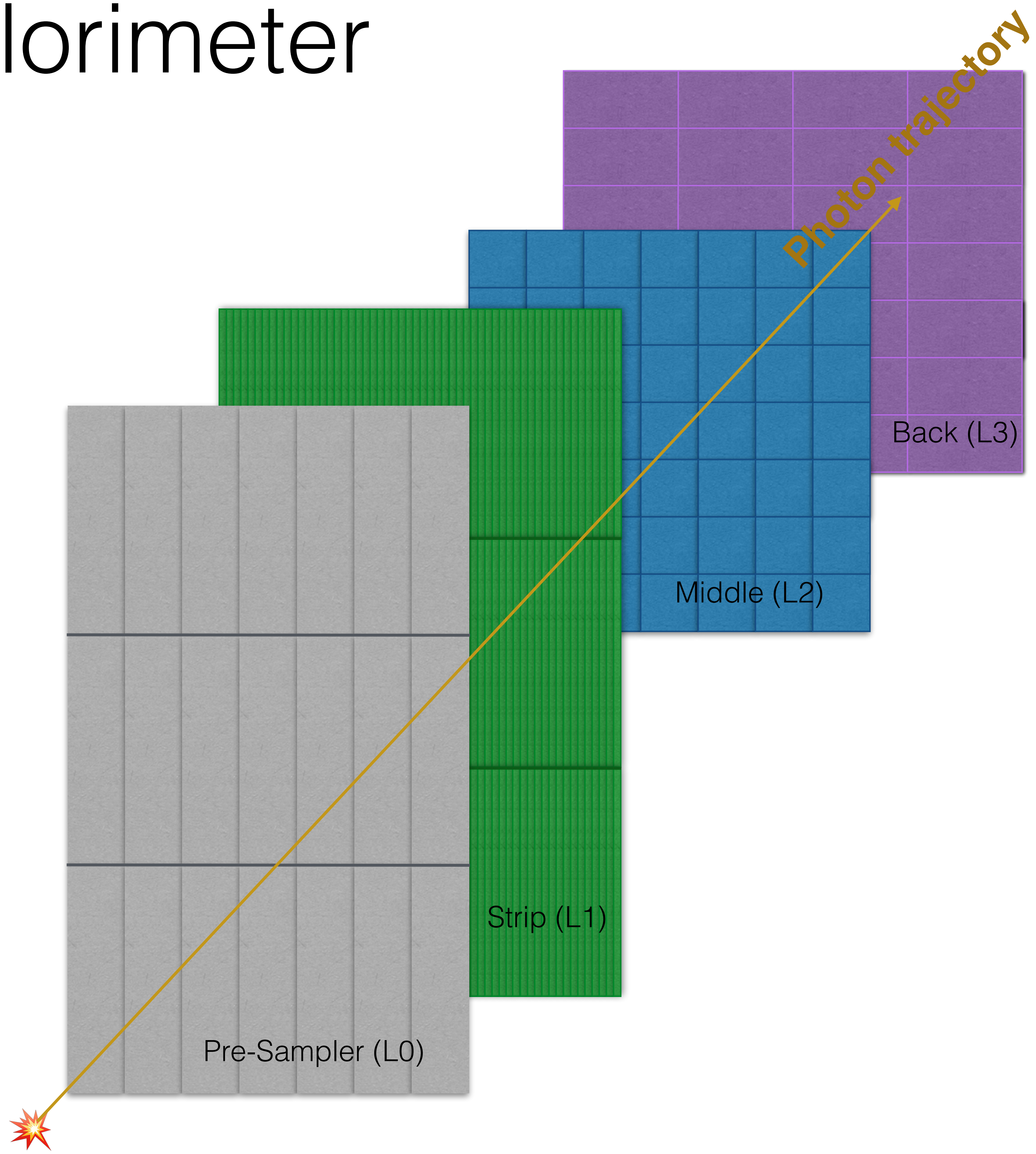
Particle goes through 4 layers in this order:

0. Pre-Sampler : (7x3) Some energy deposit

1. Strips: (56x3) Very granular in η; more energy deposit

2. Middle: (7x7) Thickest layer, maximum energy deposit

3. Back: (4x7) Little Energy deposits



The Calorimeter

2-D Axis: η vs ϕ

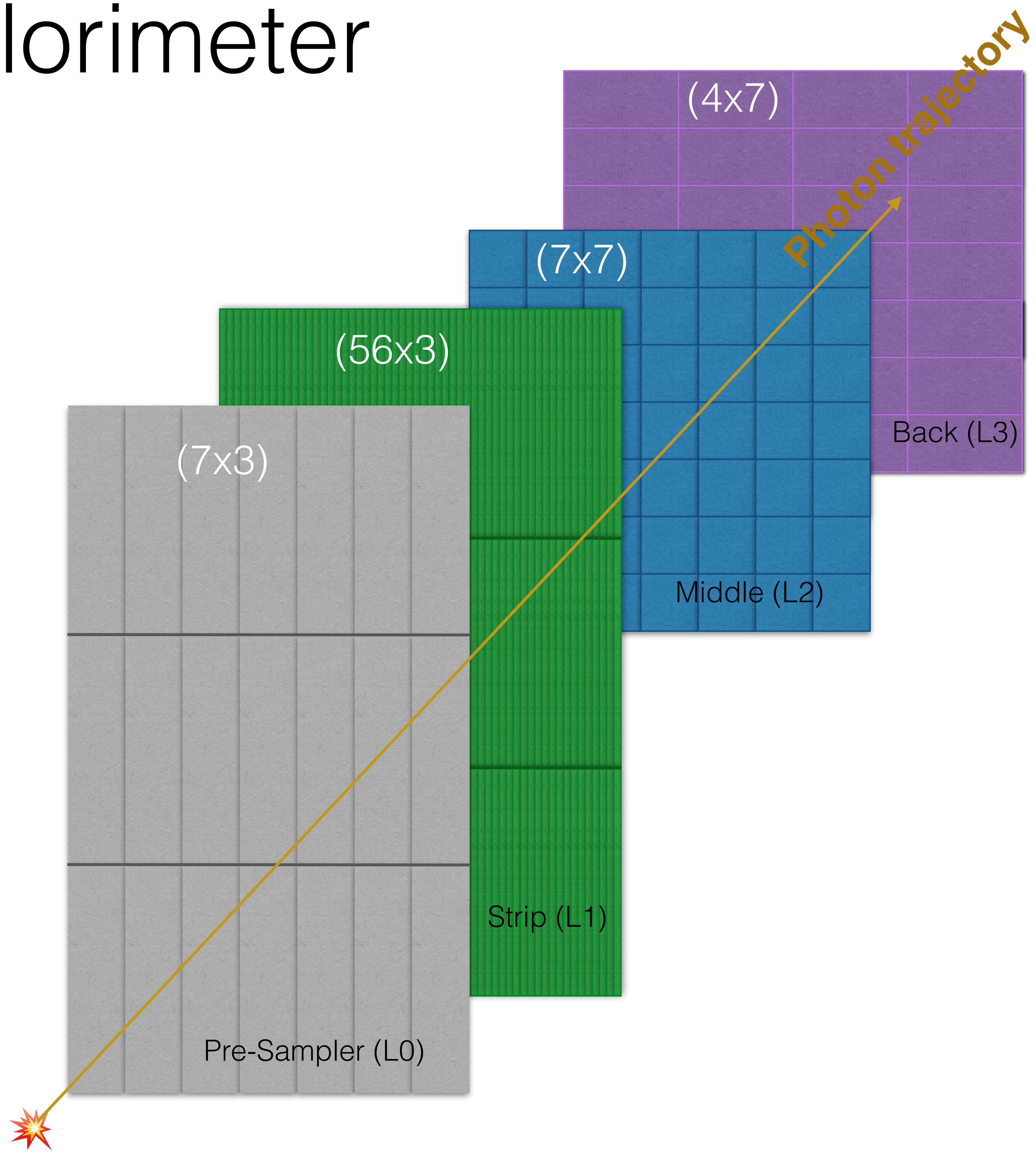
Particle goes through 4 layers in this order:

0. Pre-Sampler : (7x3) Some energy deposit

1. Strips: (56x3) Very granular in η; more energy deposit

2. Middle: (7x7) Thickest layer, maximum energy deposit

3. Back: (4x7) Little Energy deposits



The Calorimeter

2-D Axis: η vs ϕ

Particle goes through 4 layers in this order:

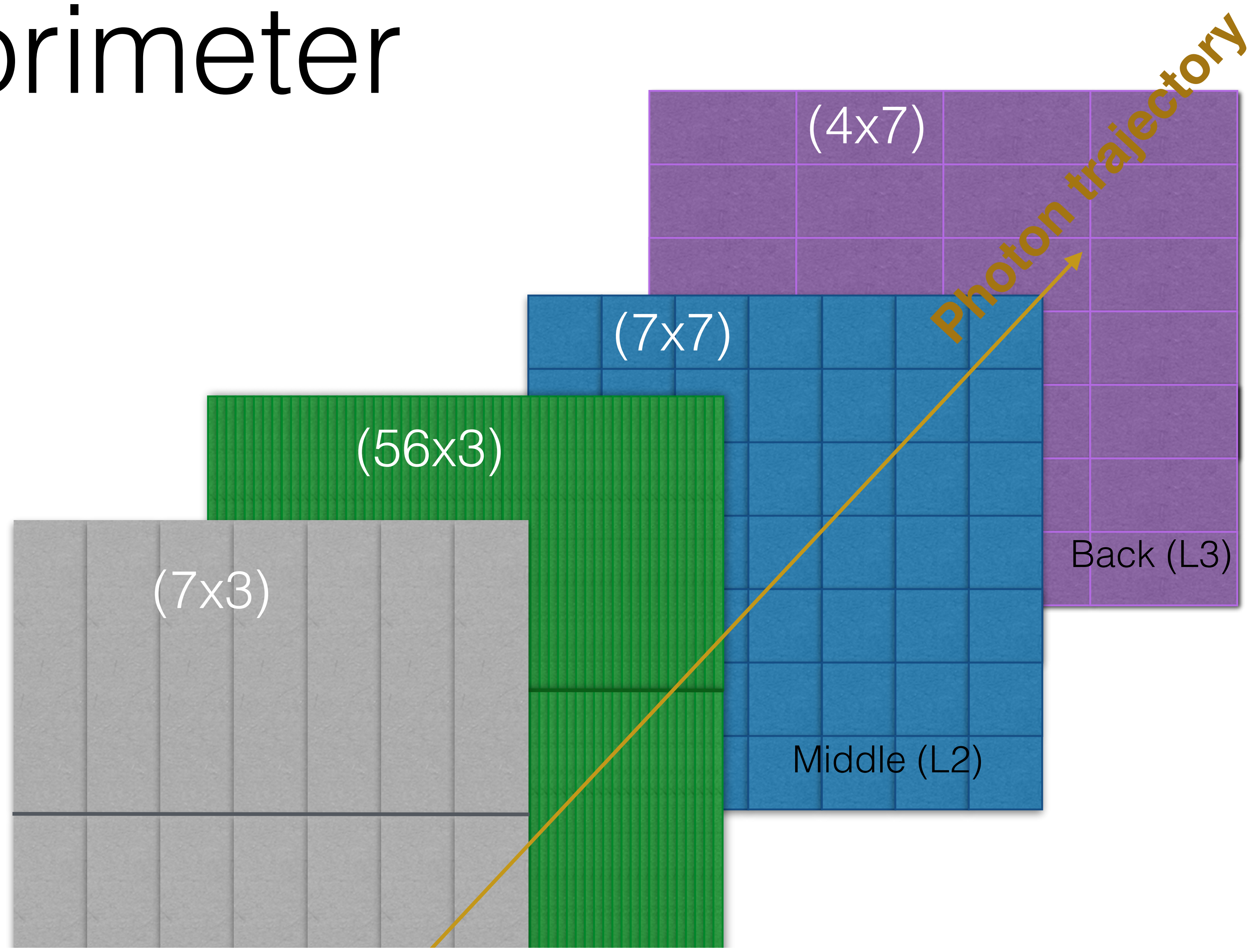
0. Pre-Sampler : (7x3) Some energy deposit

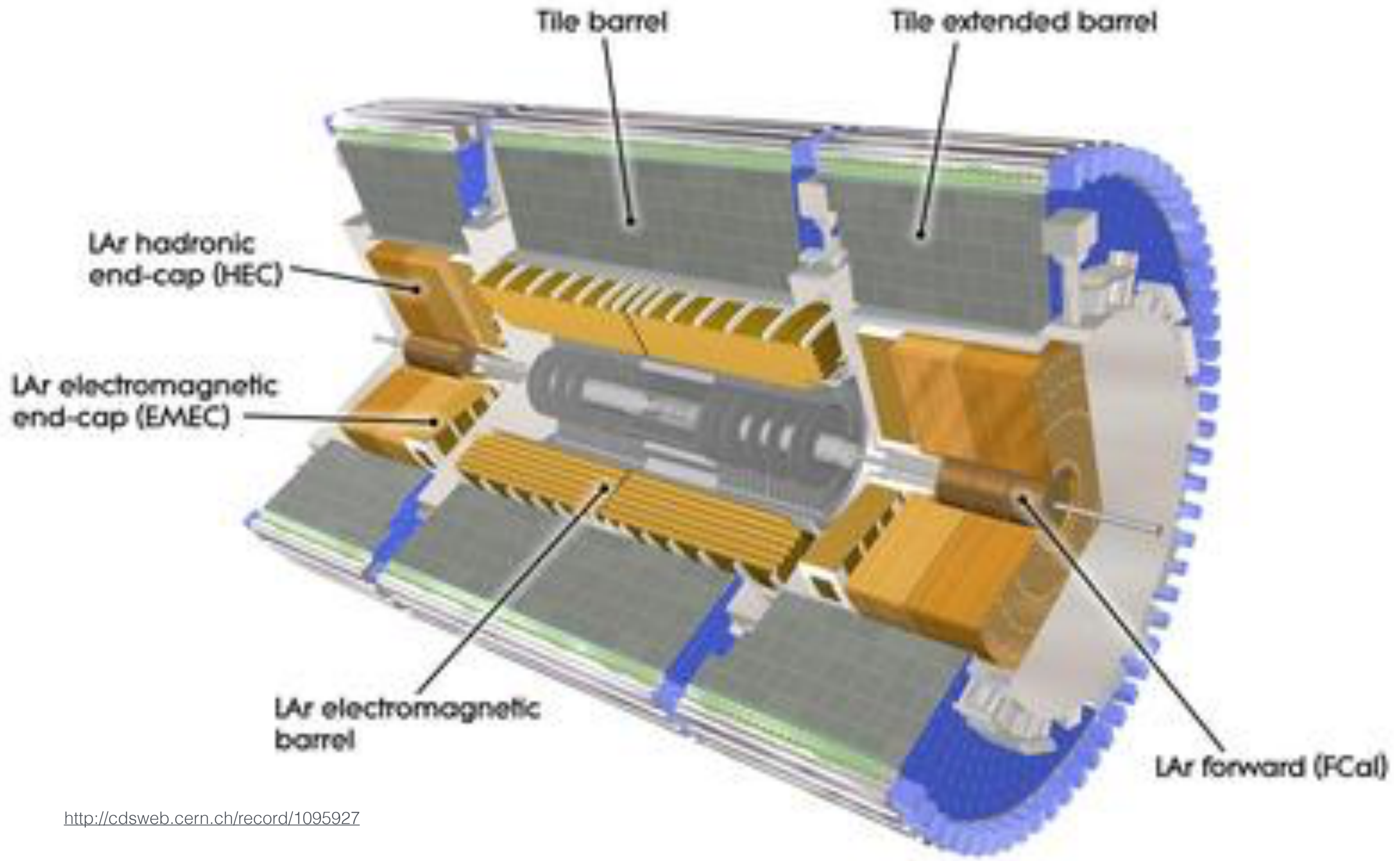
1. Strips: (56x3) Very granular in η; more energy deposit

2. Middle: (7x7) Thickest layer, maximum energy deposit

3. Back: (4x7) Little Energy deposits

Disagreement between “raw” designed and actual position of the cells for practical reasons





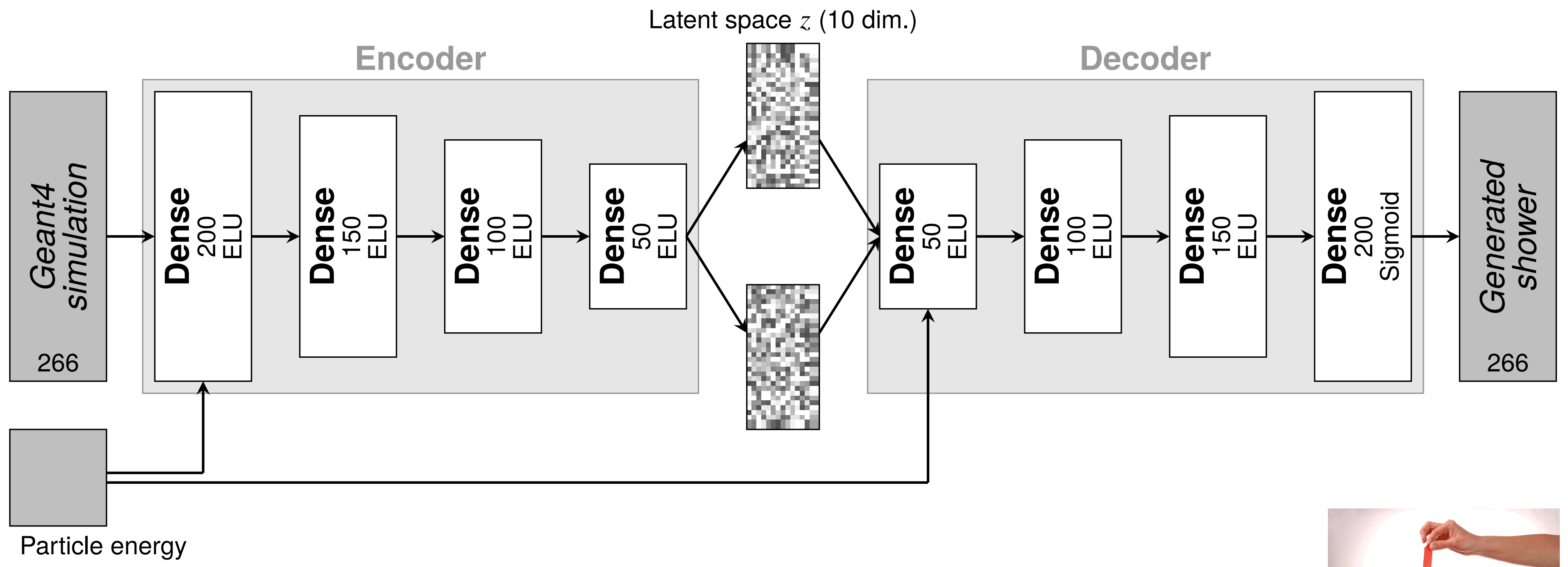
http://cdsweb.cern.ch/record/1095927

https://cds.cern.ch/record/2630433/files/ATL-SOFT-PUB-2018-001.pdf

Graeme Stewart (CERN), Aishik Ghosh, David Rousseau (LAL, Orsay), Kyle Cranmer (NYU), Stefan Gadatsch, Tobias Golling, Dalila Salamani (UniGe), Gilles Louppe (ULiège)

PubNote 2018: VAE and GAN

VAE:



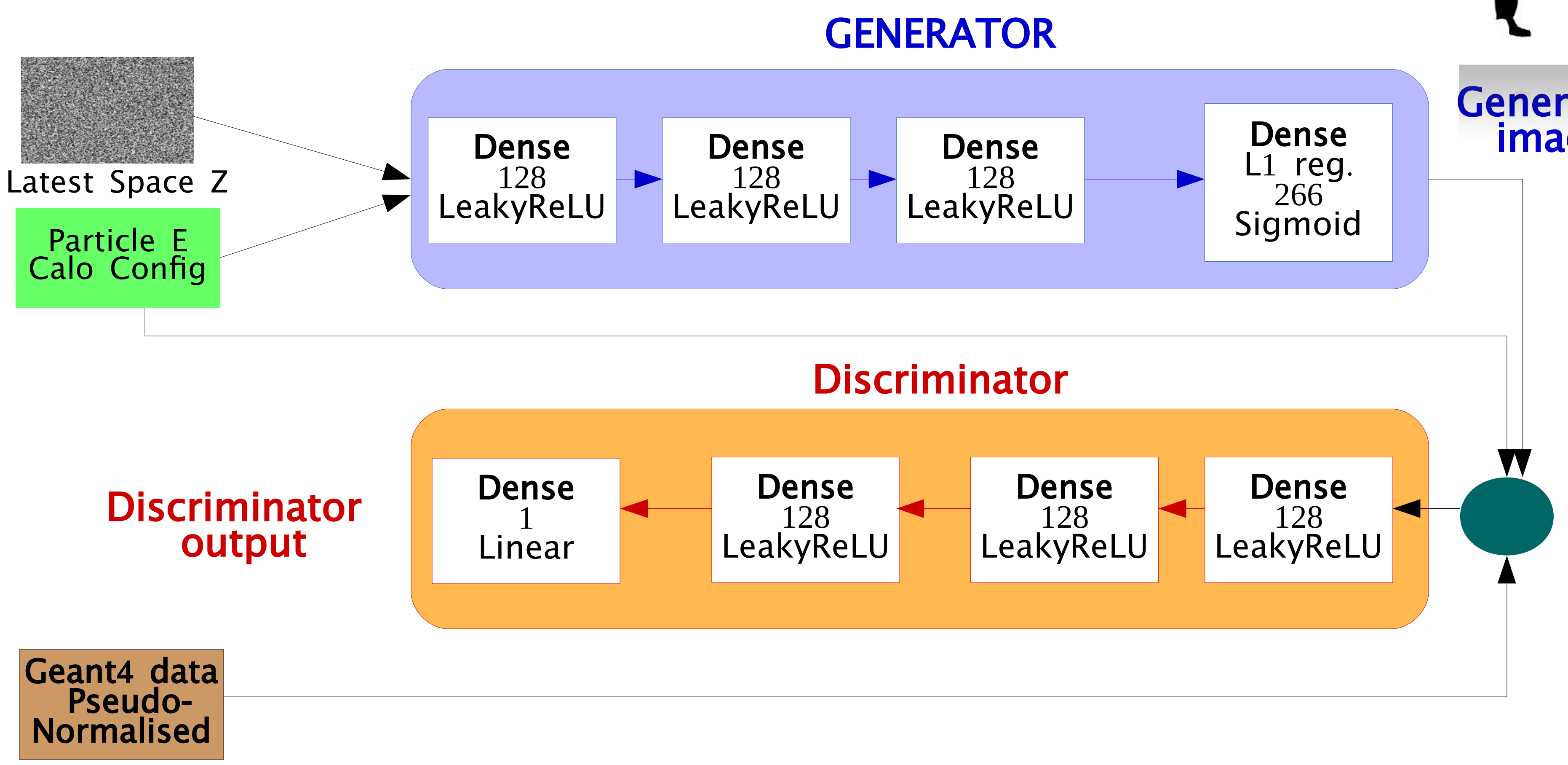
100 epochs, 2 mins, CPU

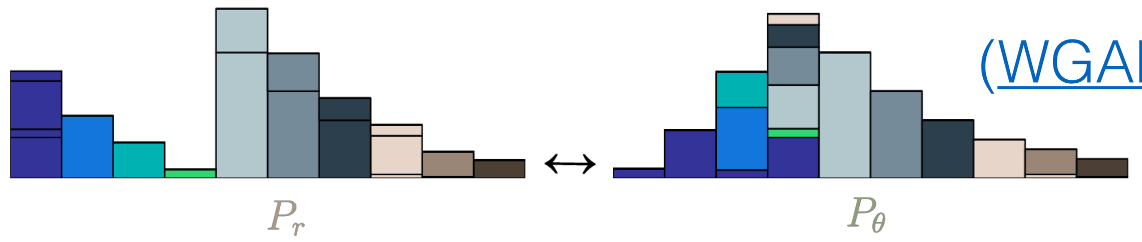
Flat vector of 266 cells are the output of both generators

Not an ideal training dataset

50k 'epochs', 7 hours training, 1 GPU

GAN:





Keras TensorFlow

Training dataset:

- Single **photon** samples from Geant4
- 88000 events
- 9 **discrete energy points** : {1, 2, 4, 8, 16, 32, 65, 131, 262} GeV
- $0.20 < |\eta| < 0.25$
- 4 electromagnetic calorimeter layers

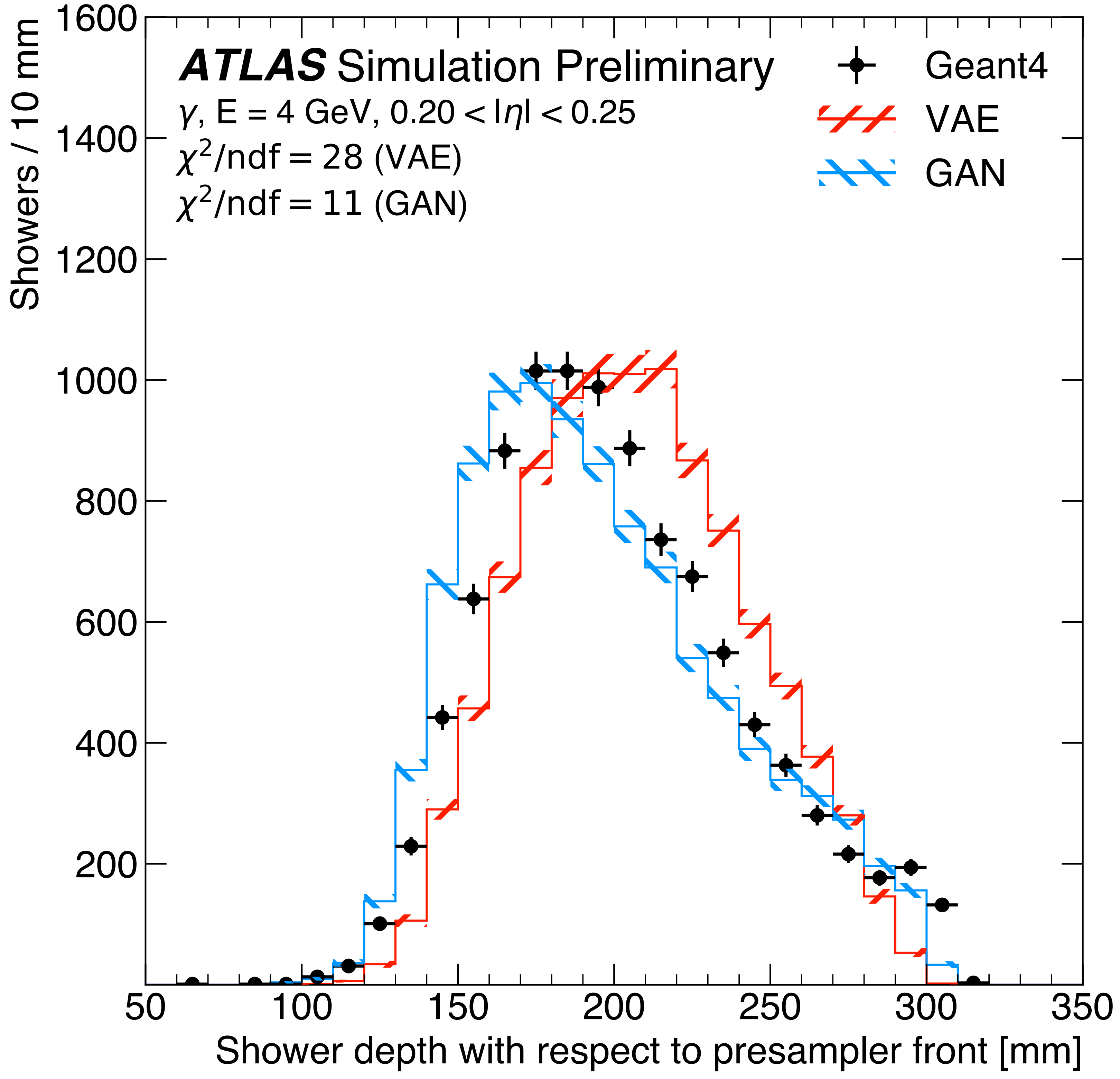
Data preprocessing

- Negative energies set to 0
- Mirror $\eta < 0$

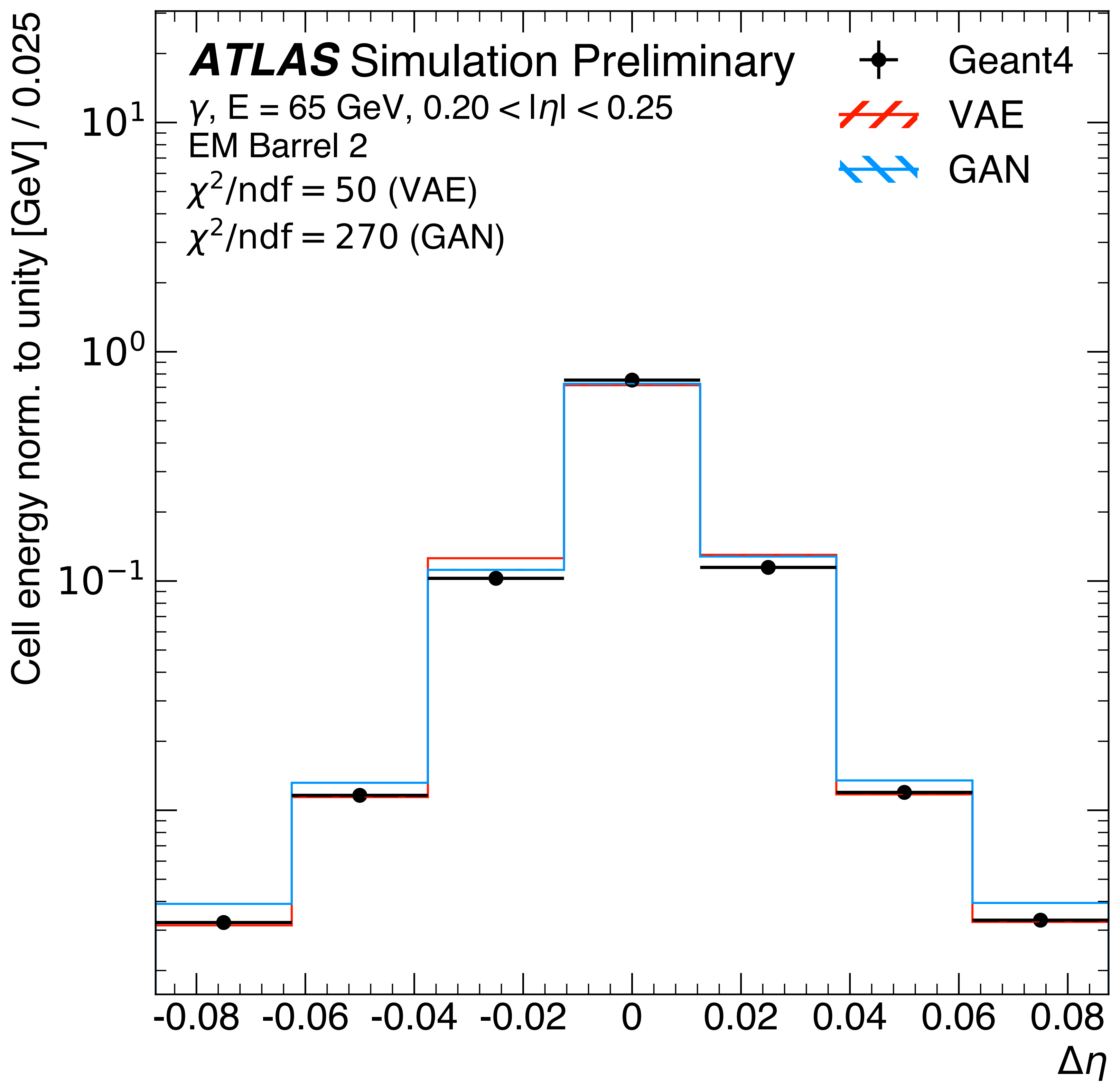
(WGAN-GP, Improved WGAN-GP nightmare on Keras!)

2018 Results(1/2)

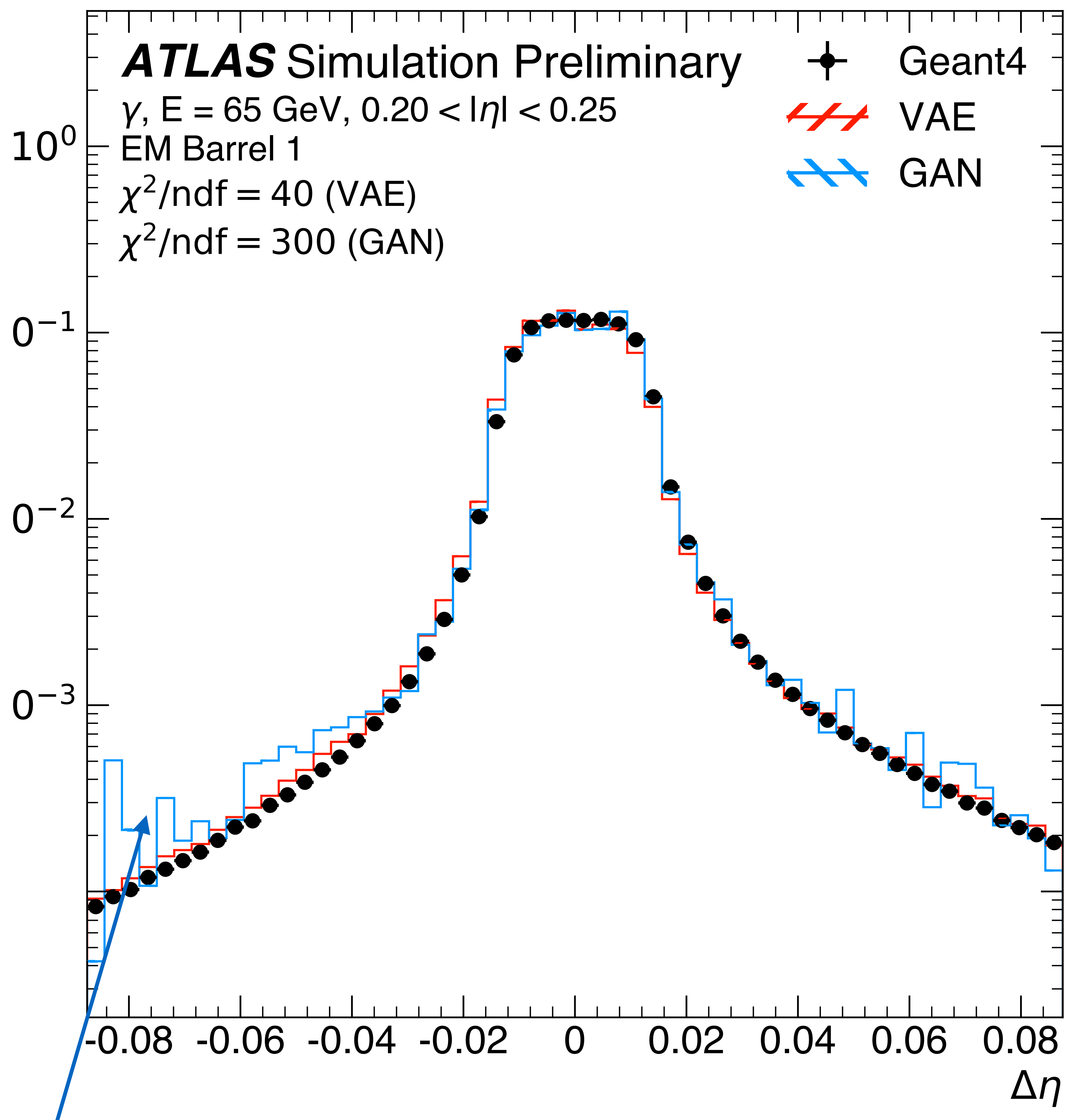
From summer PubNote 2018



Shower Depth



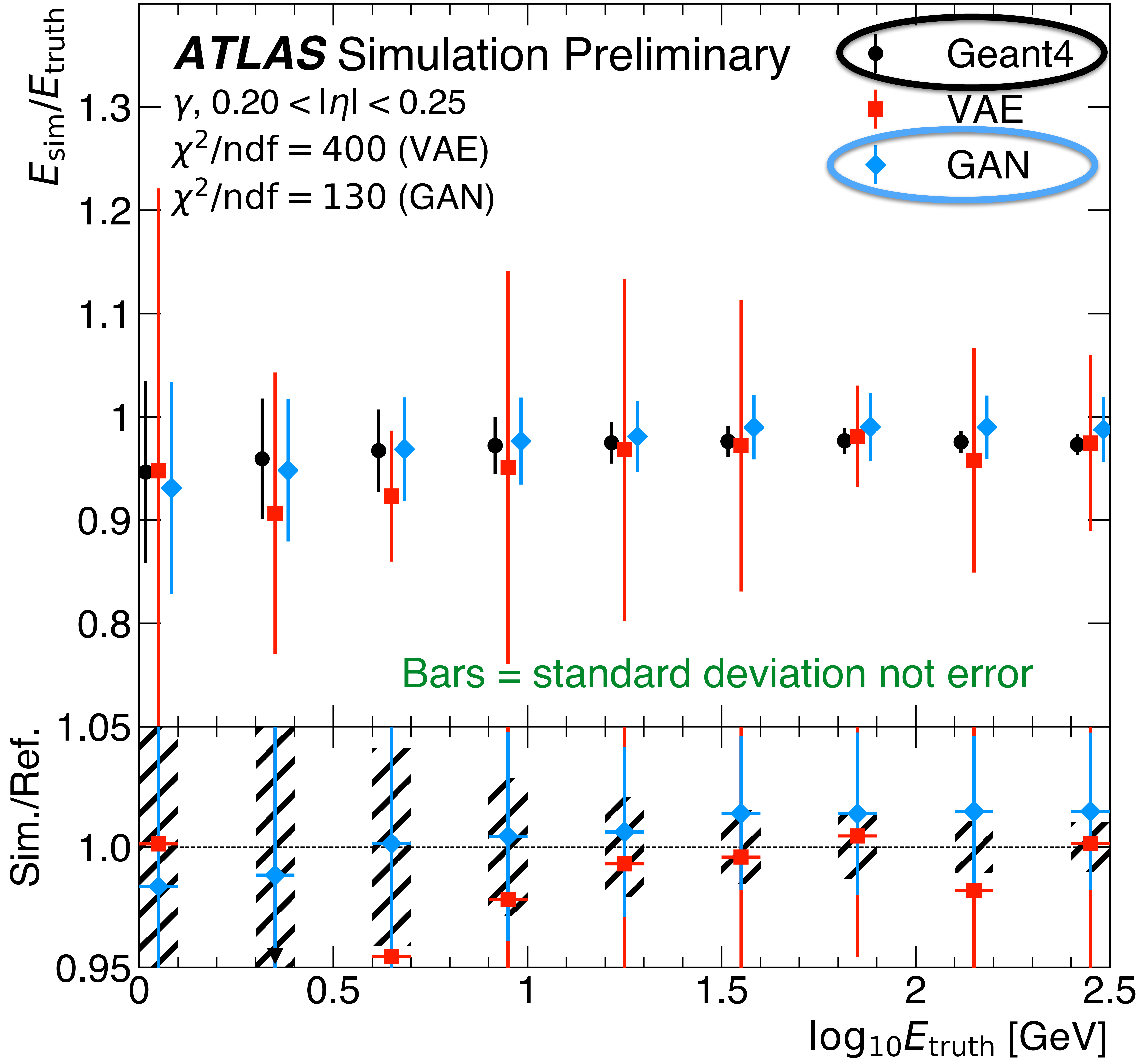
Average η in Middle



Average η in Strip

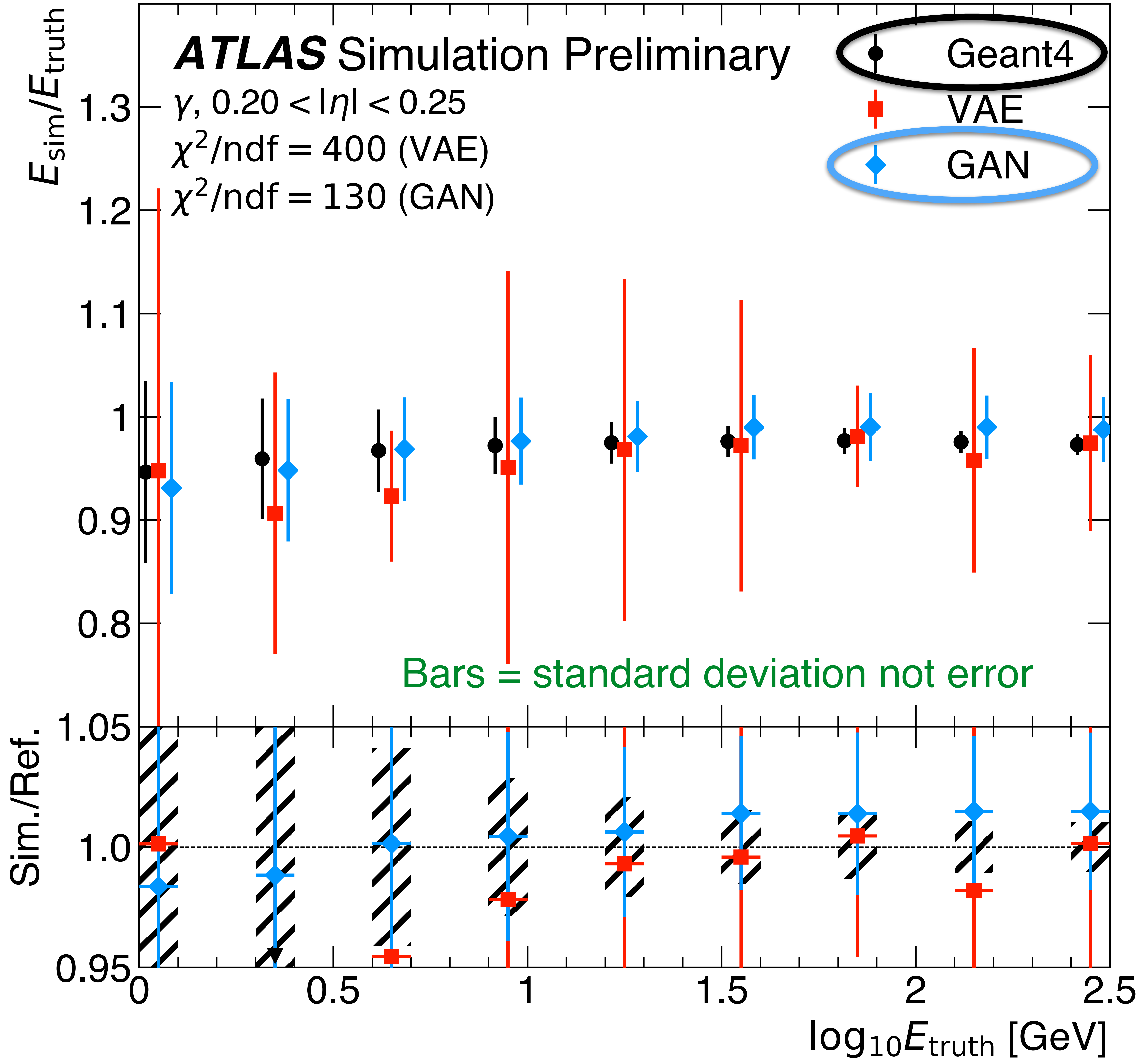
GAN: Fluctuations due to small training size, fixed since PubNote. Train on 4% -> 50% of dataset by **removing momentum**, lowering number of updates to generator

Disaster: Cannot Model Detector Resolution



Well known detector resolution: **σE/E ~ 10% /√E**

Disaster: Cannot Model Detector Resolution



η, ϕ, other distributions not so bad but for total energy…

GAN gets the means but not the **widths** of the energies

Well known detector resolution: **σE/E ~ 10% /√E**

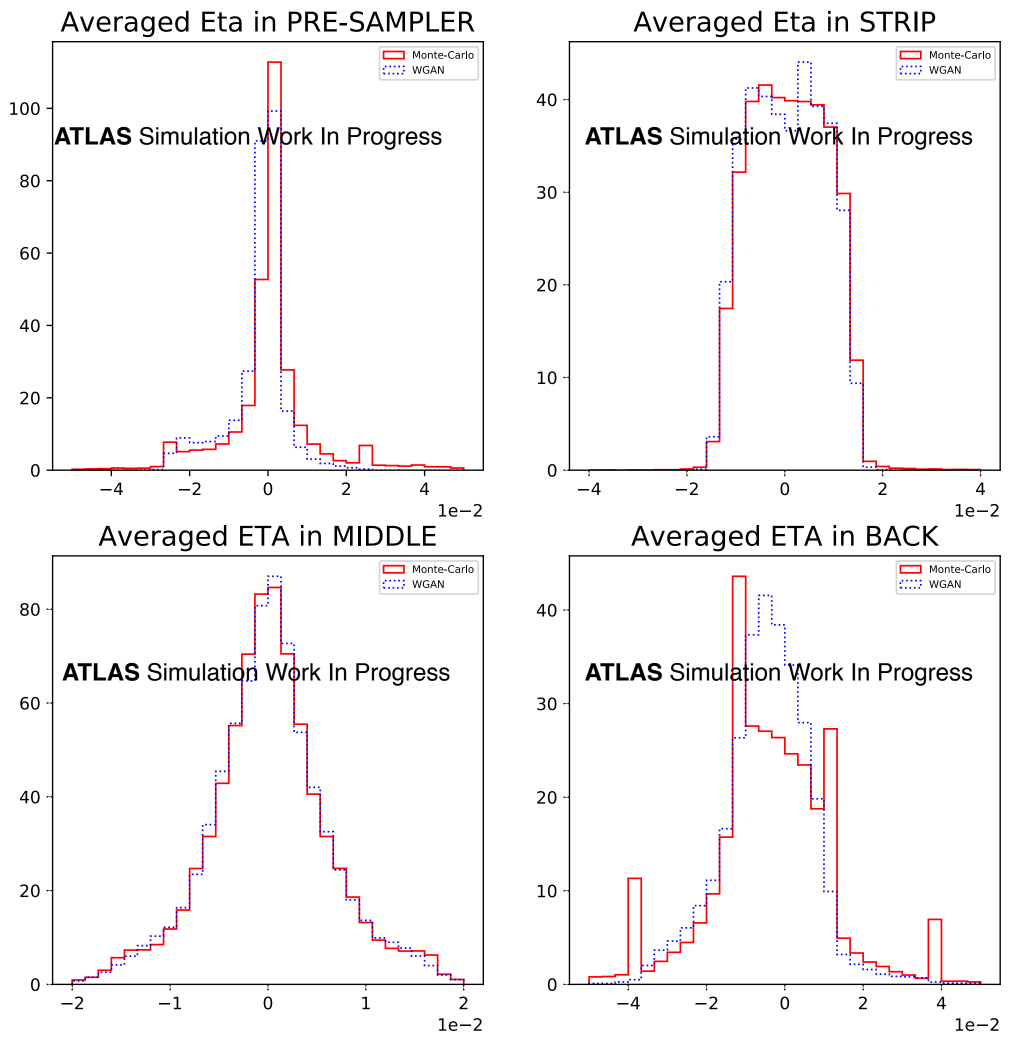
Critic can't see the difference b/w real and fake images.

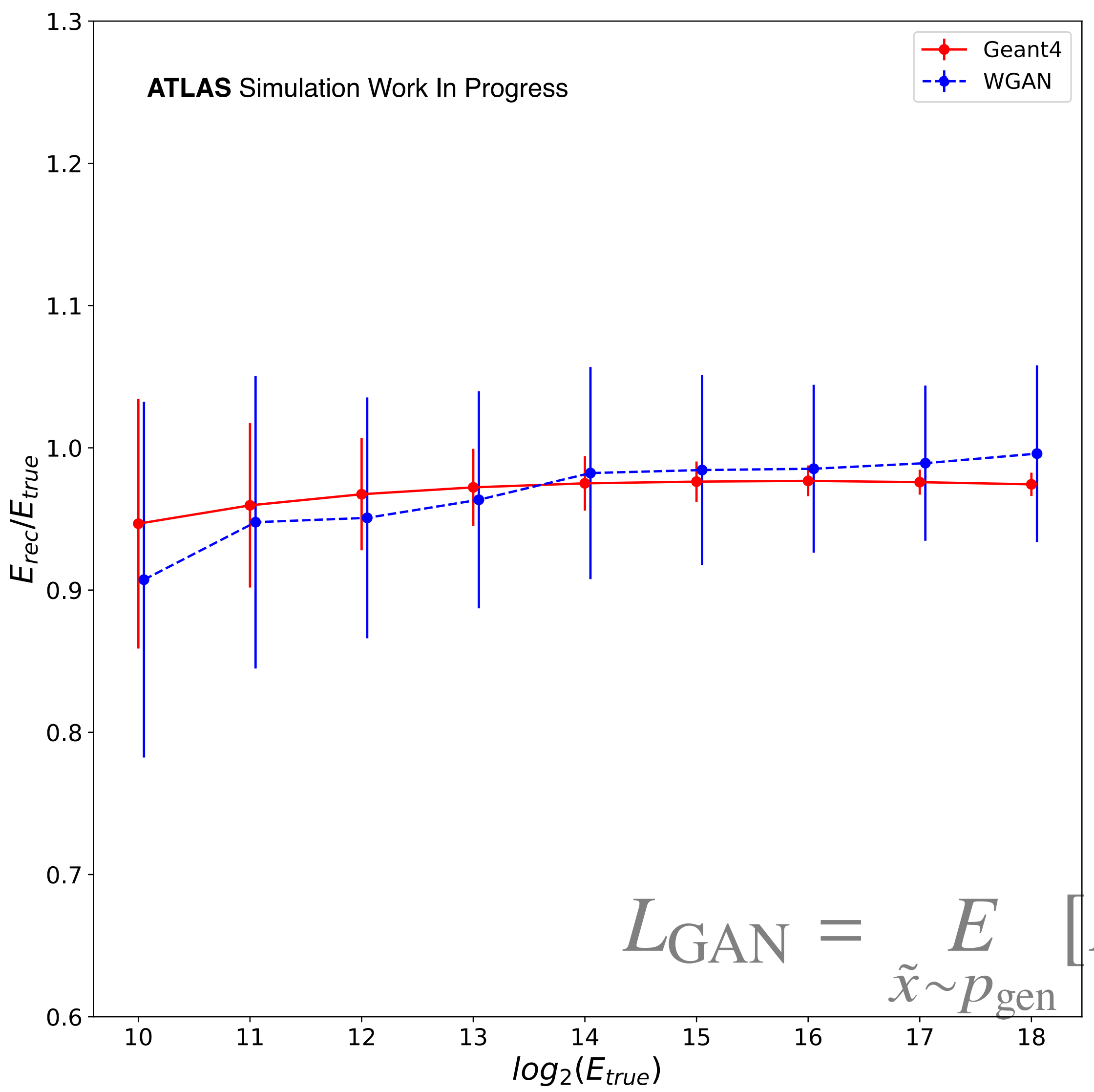
Tried training on single high energy point,
Minibatch discrimination, various other tricks. No result.

What is a good range of hyper-parameters to try anyway?

Results were stable for usual GP values, ∈ (1,500)

Gradient Penalty = 10



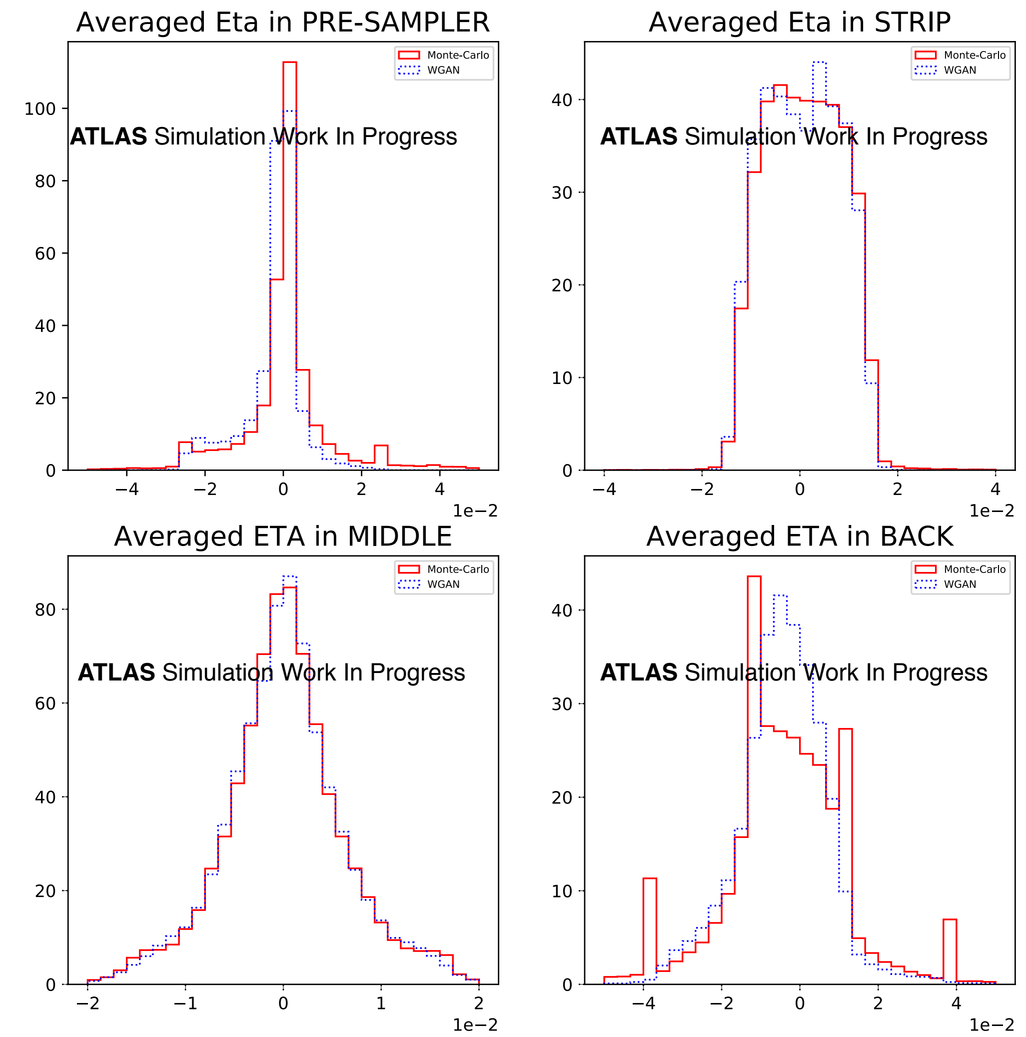


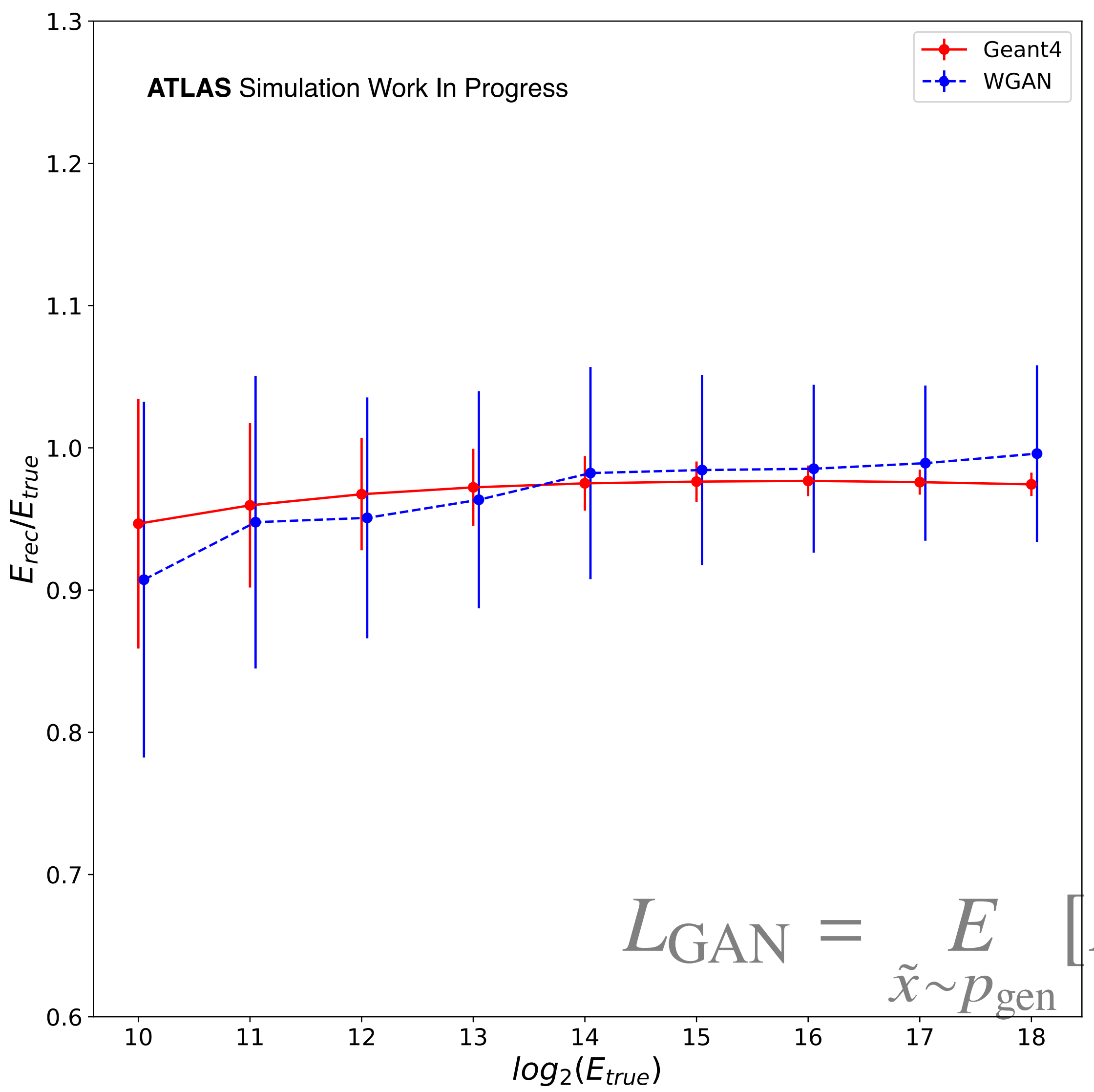
$$L_{\mathrm{GAN}} = \underset{\tilde{x}\sim p_{\mathrm{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\mathrm{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}}D(\hat{x})||_2 - 1)^2]$$

What is a good range of hyper-parameters to try anyway?

Results were stable for usual GP values, ∈ (1,500)

Gradient Penalty = 10

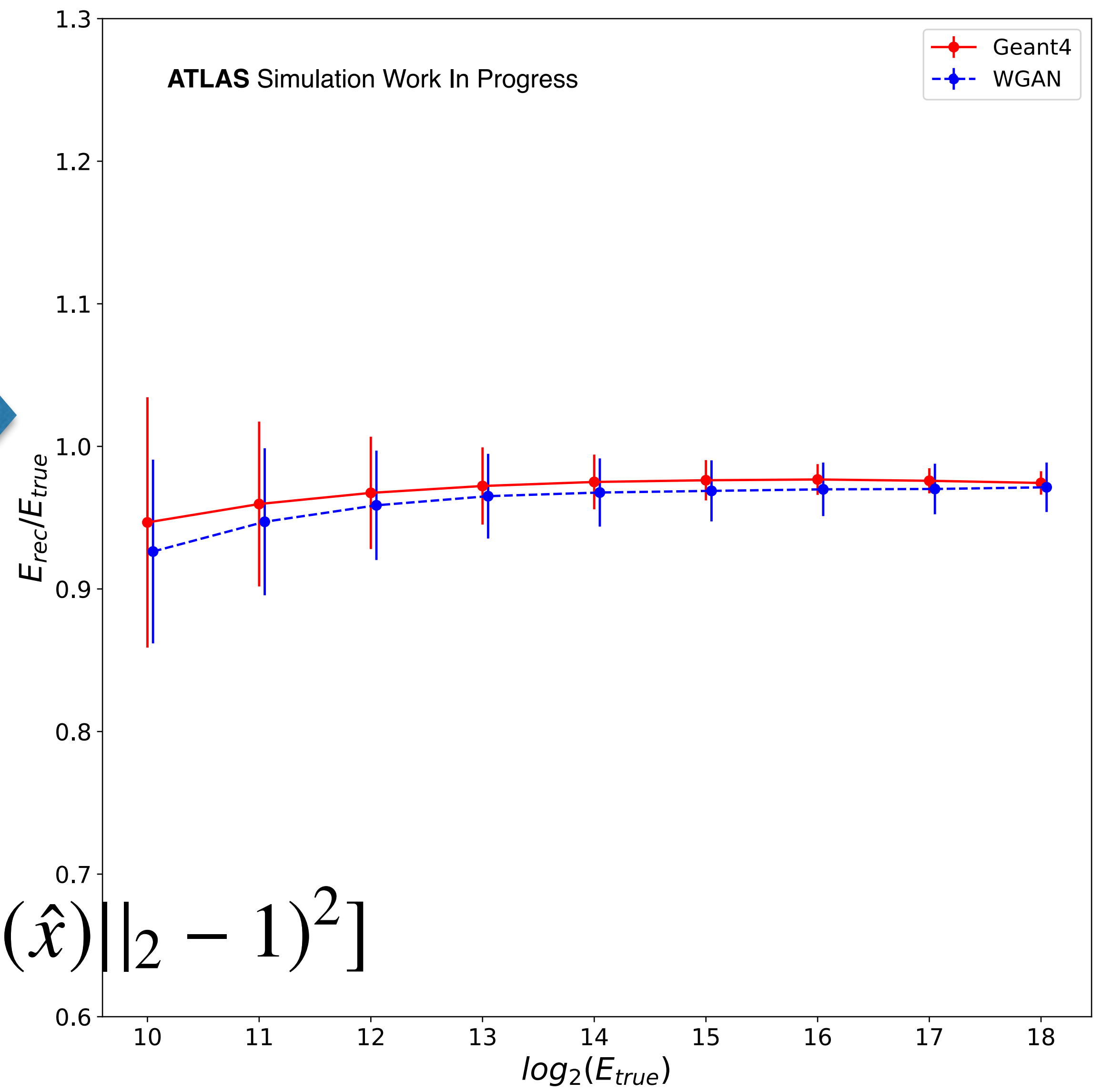




Energy gets better

Gradient Penalty = 1e-13

Never seen such a number in literature

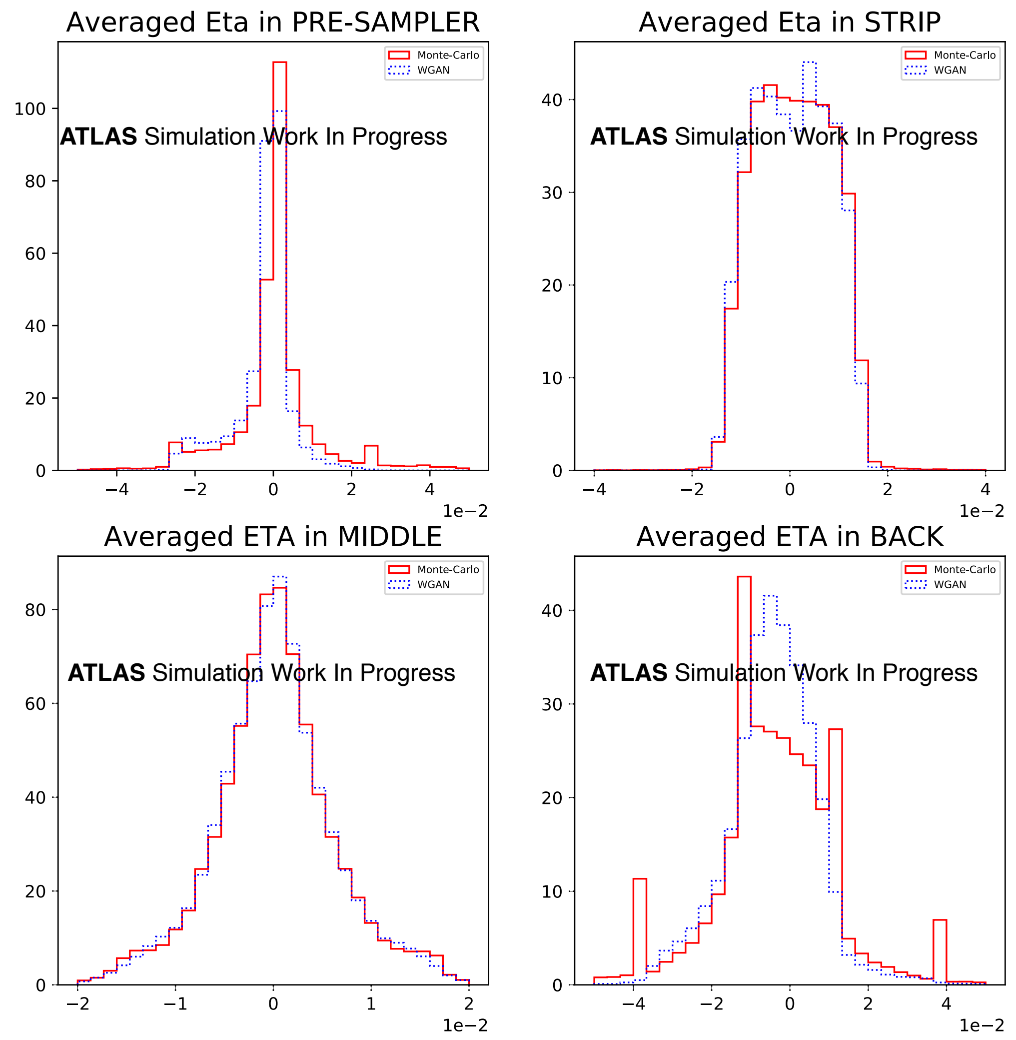


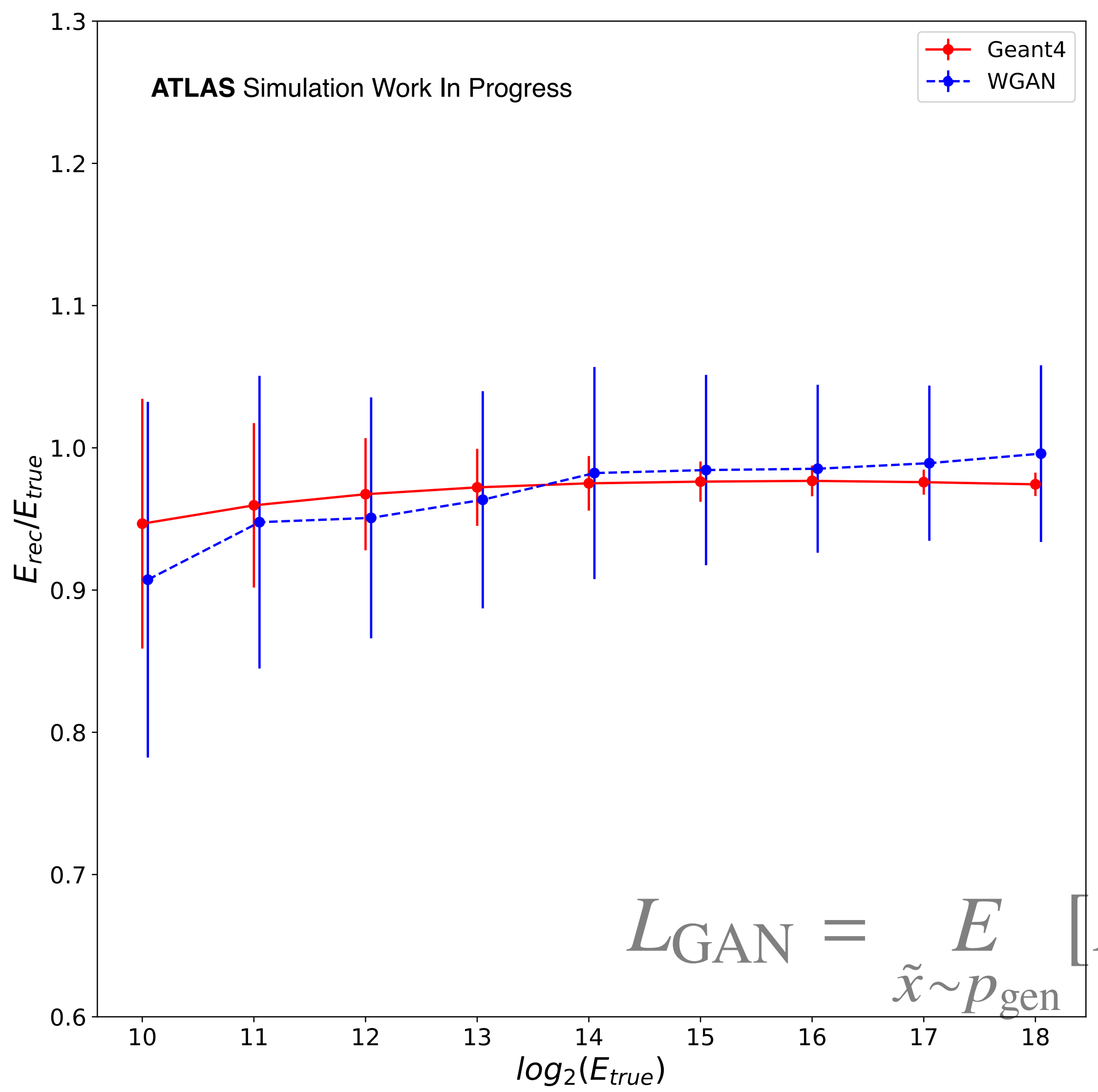
$$L_{\mathrm{GAN}} = \underset{\tilde{x}\sim p_{\mathrm{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\mathrm{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

What is a good range of hyper-parameters to try anyway?

Results were stable for usual GP values, ∊ (1,500)

Gradient Penalty = 10

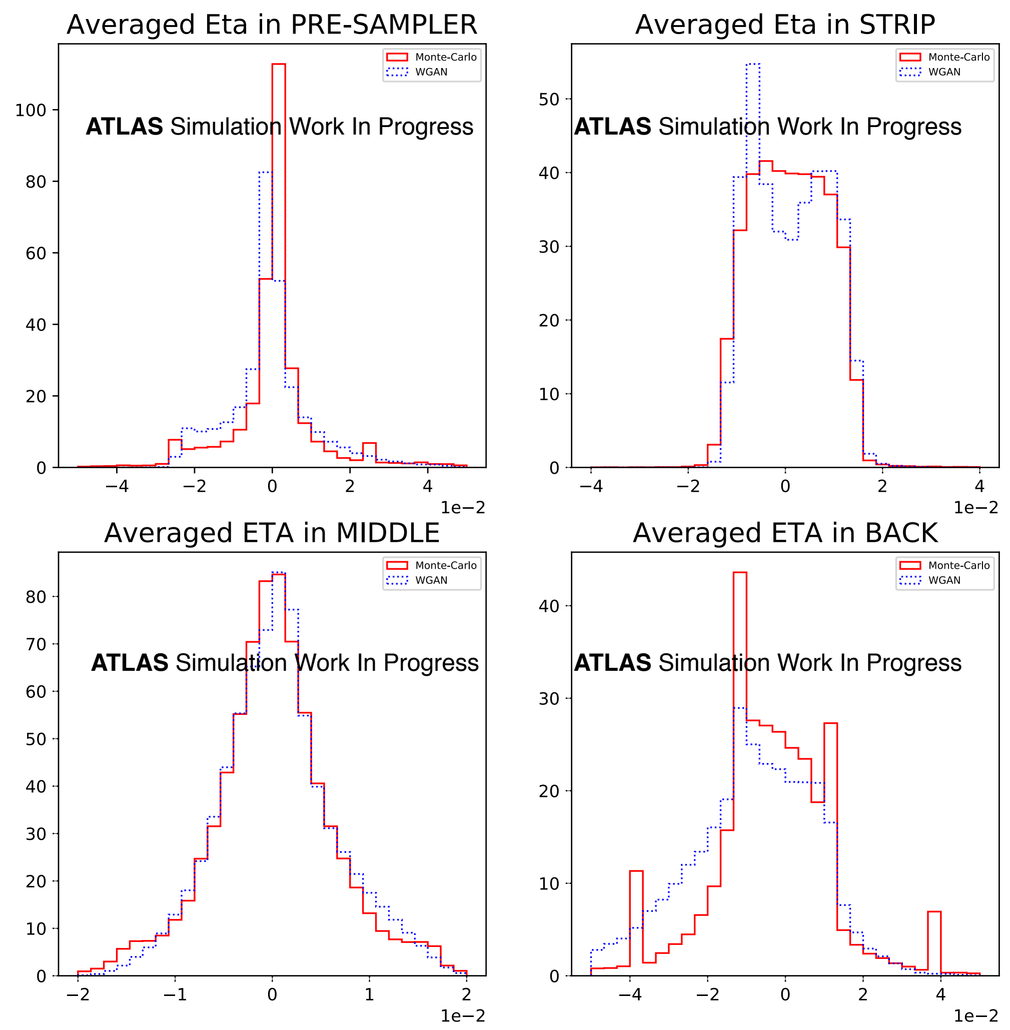


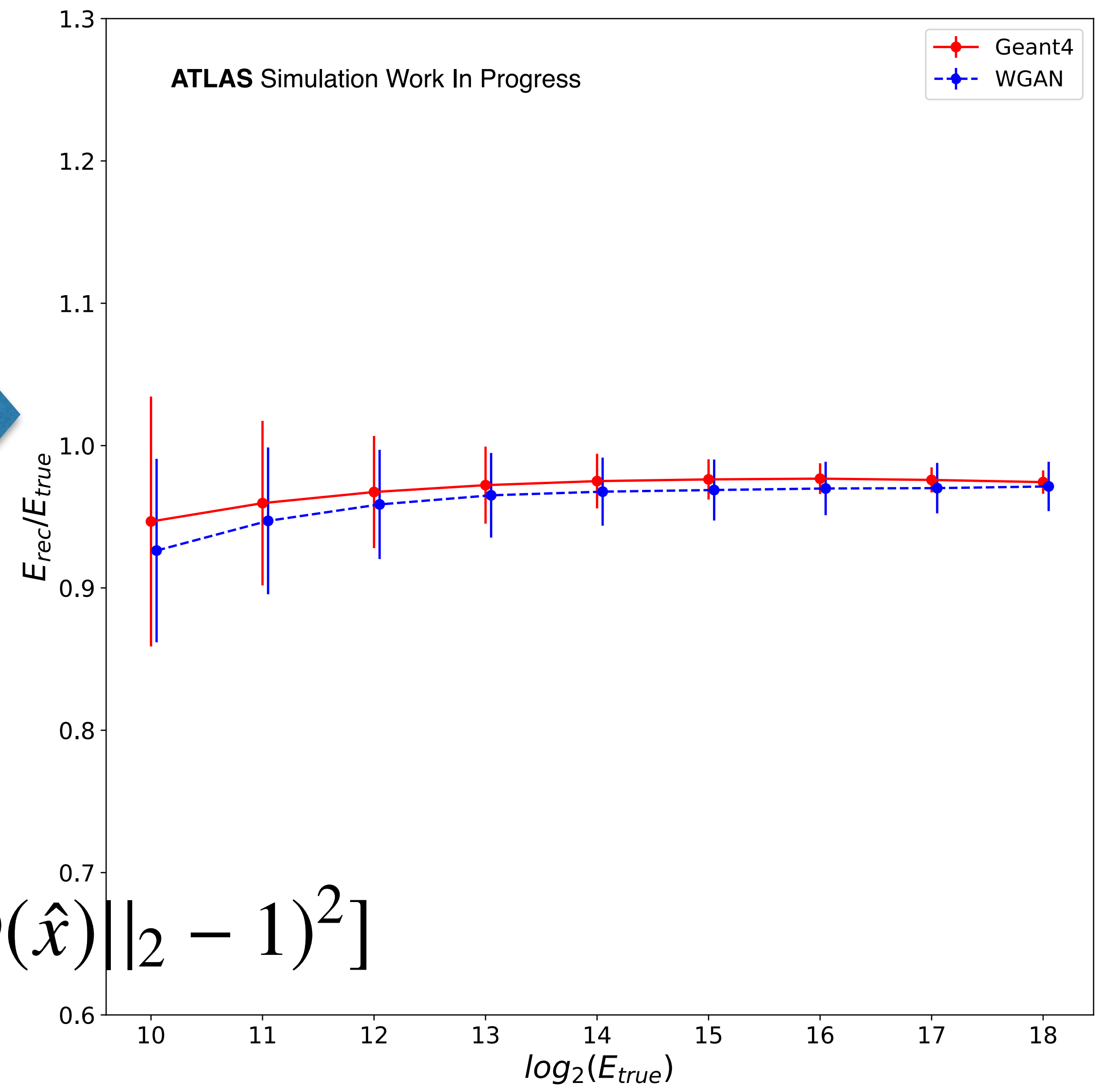


Plots get worse

Energy gets better

Gradient Penalty = 1e-13



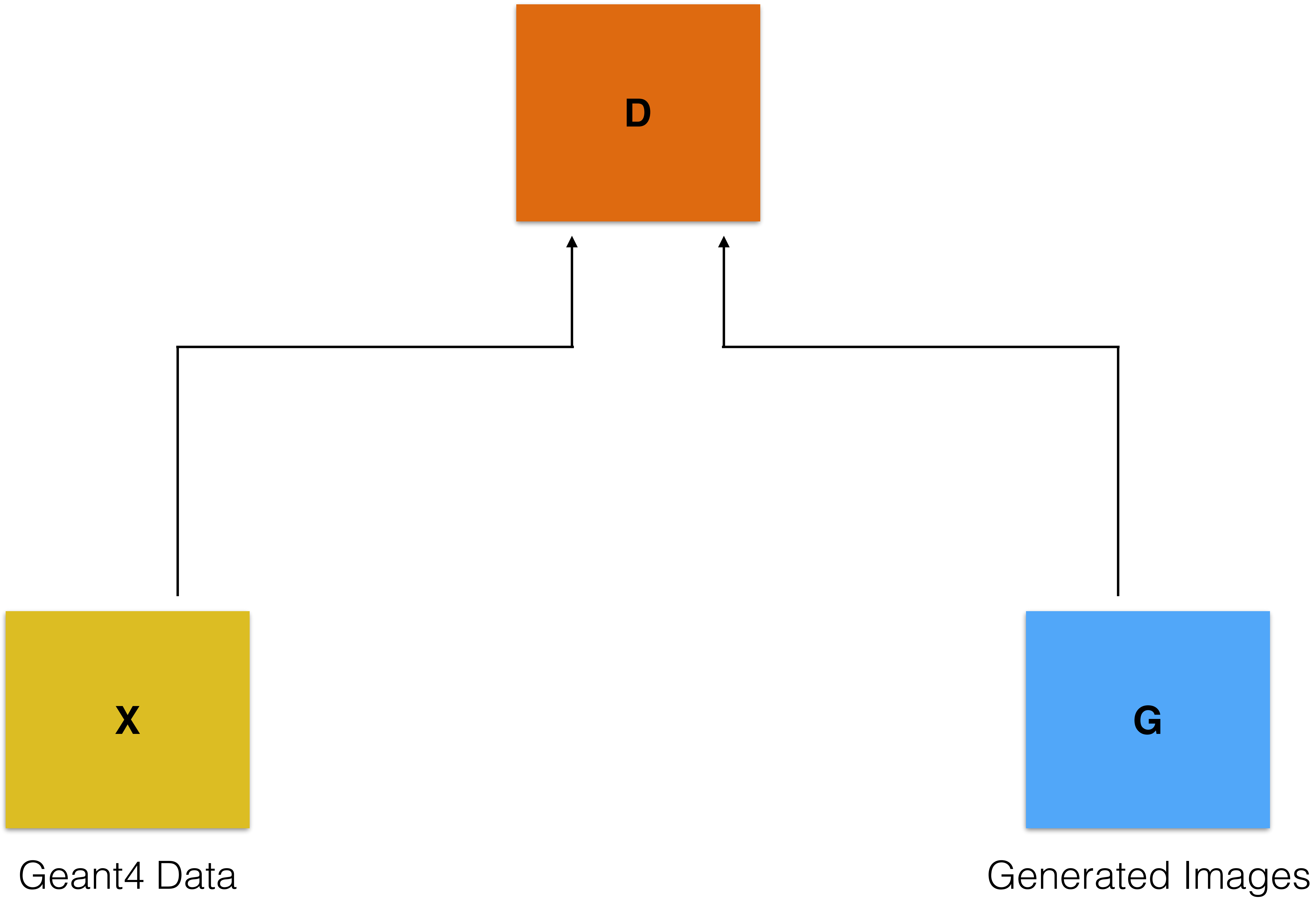


Never seen such a number in literature

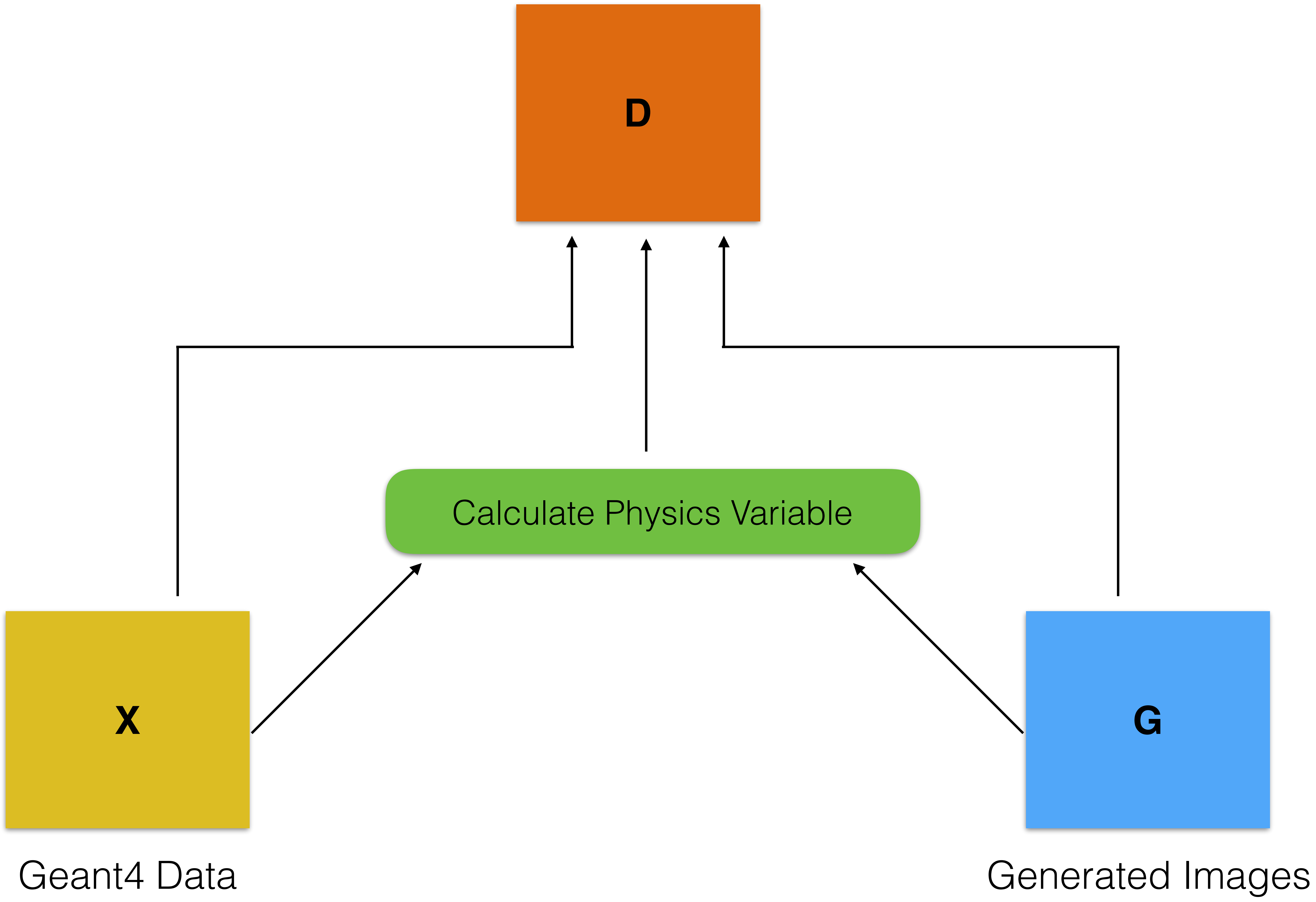
And highly unstable training

$$L_{\mathrm{GAN}} = \underset{\tilde{x}\sim p_{\mathrm{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\mathrm{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

Add Physics Variables in Training

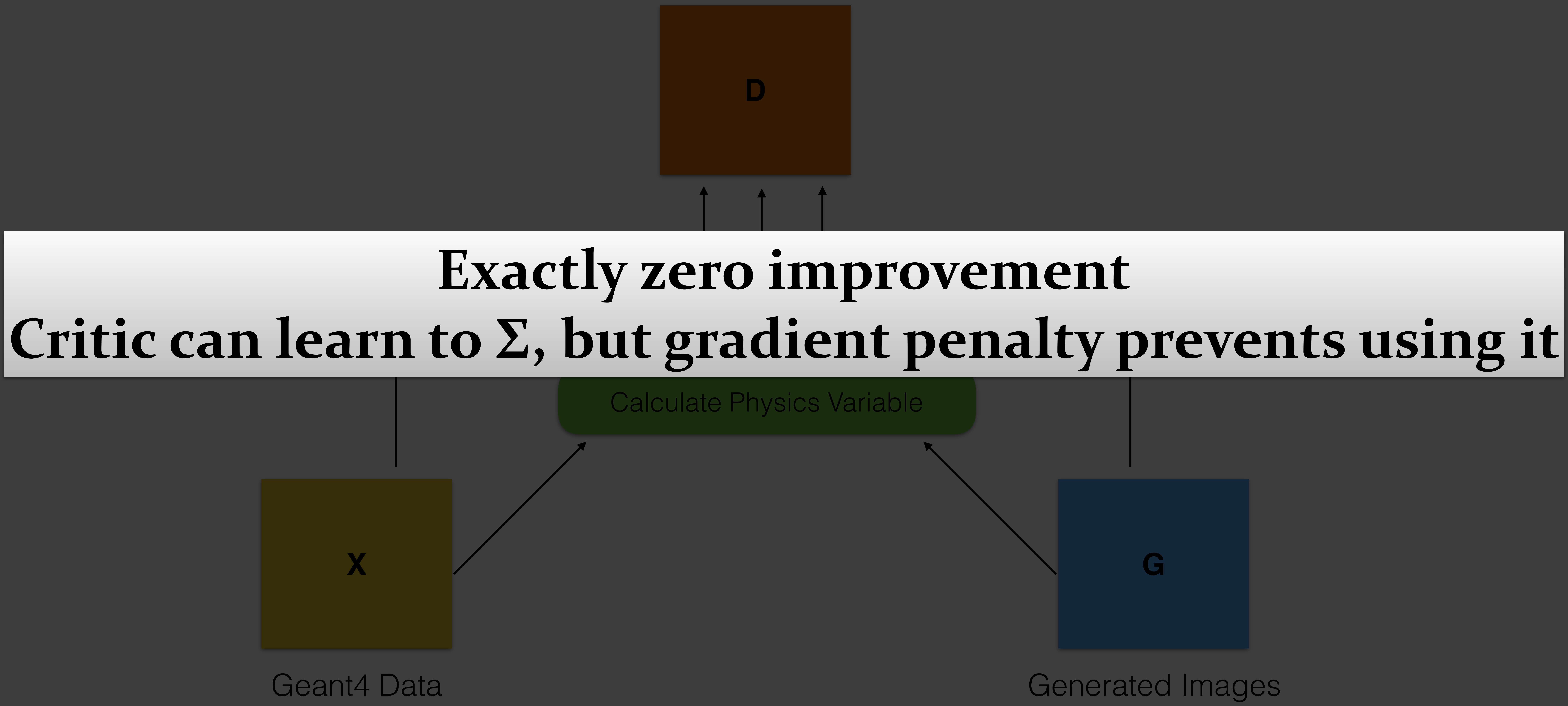


Add Physics Variables in Training



Help the discriminator see physics

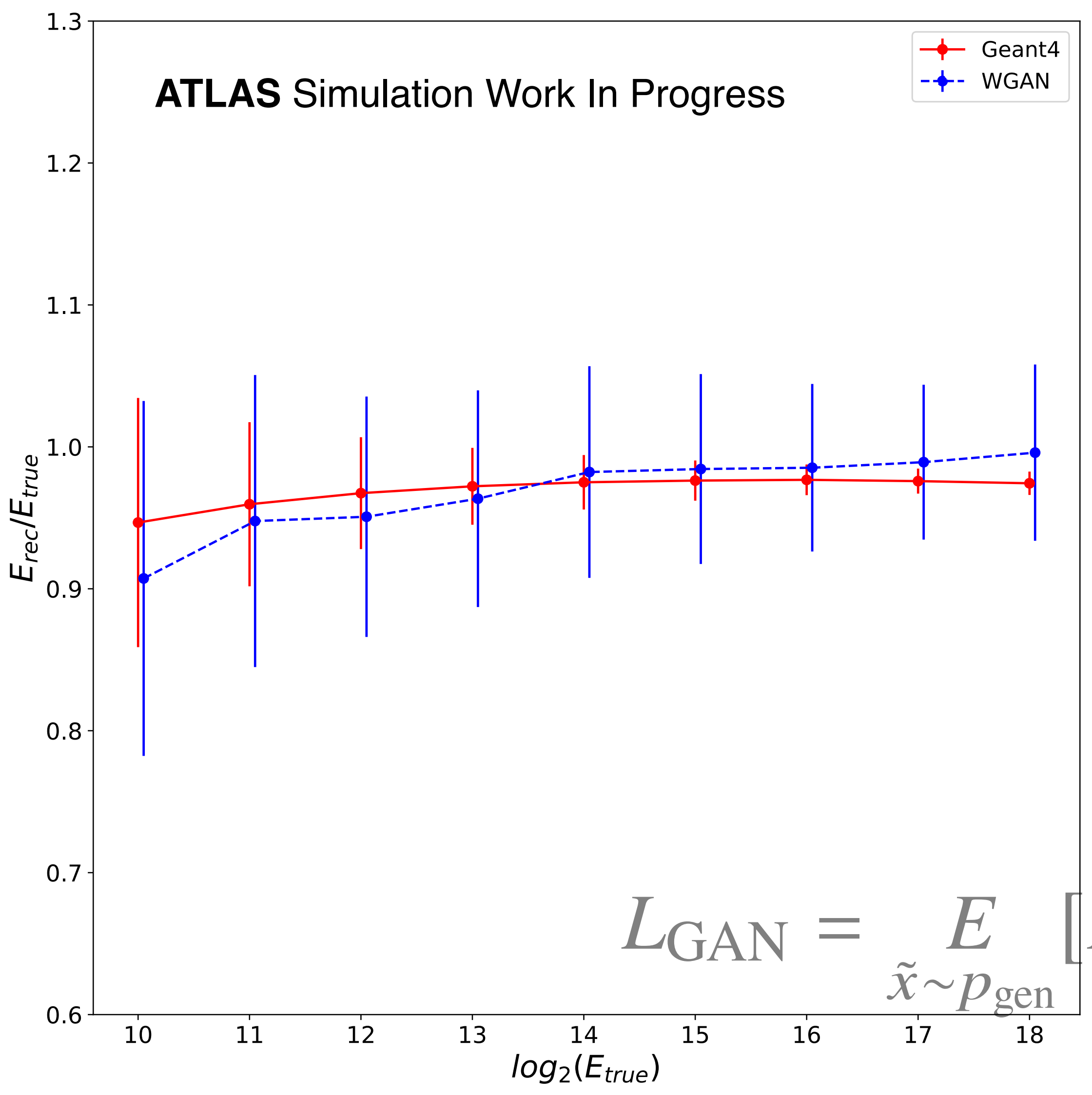
Add Physics Variables in Training



Help the discriminator see physics

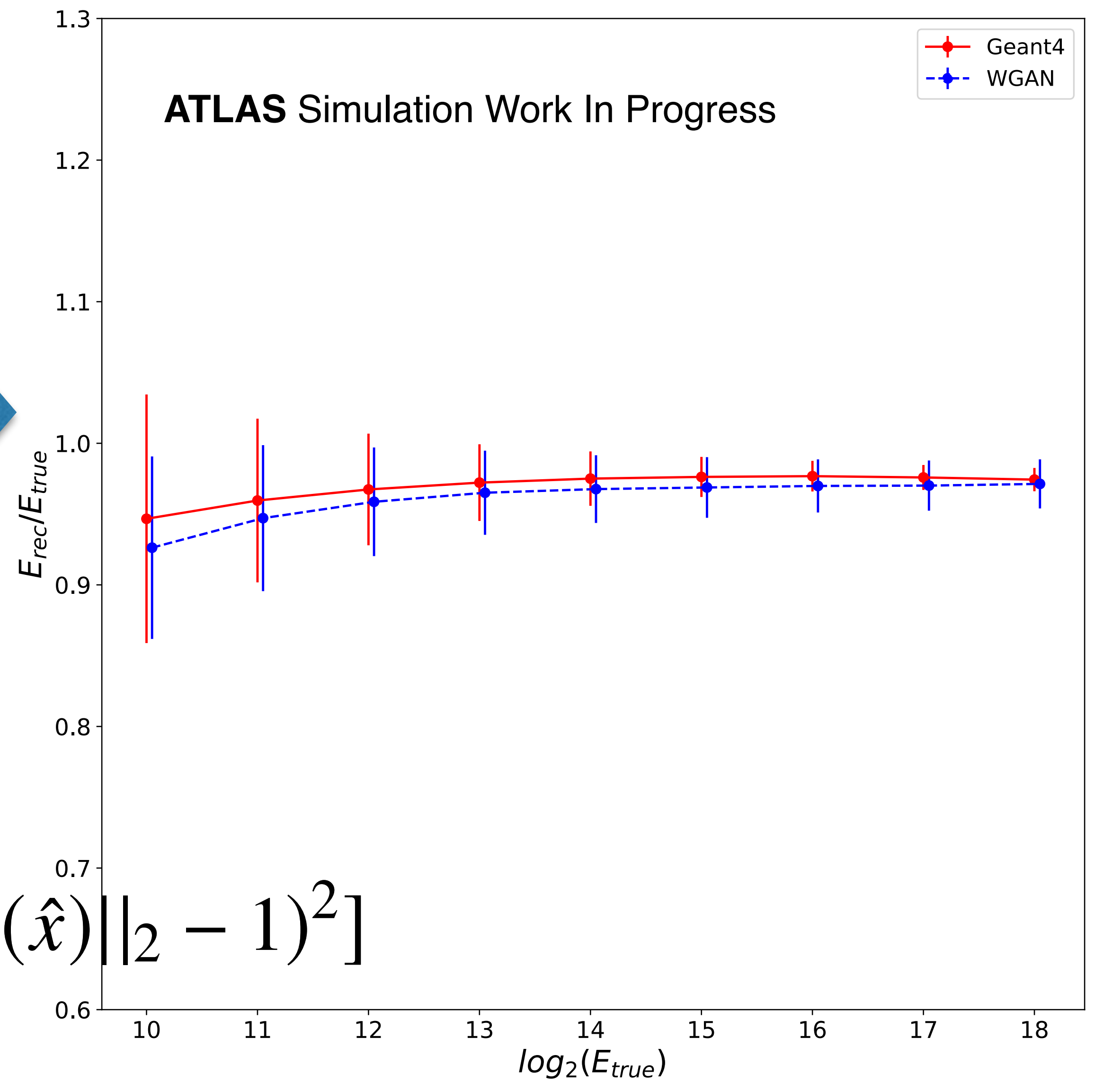
Remove Grad Penalty for Total Energy (TE)

Critic Input : 266 Cells



Energy gets better

Critic Input : 266 Cells + Sum

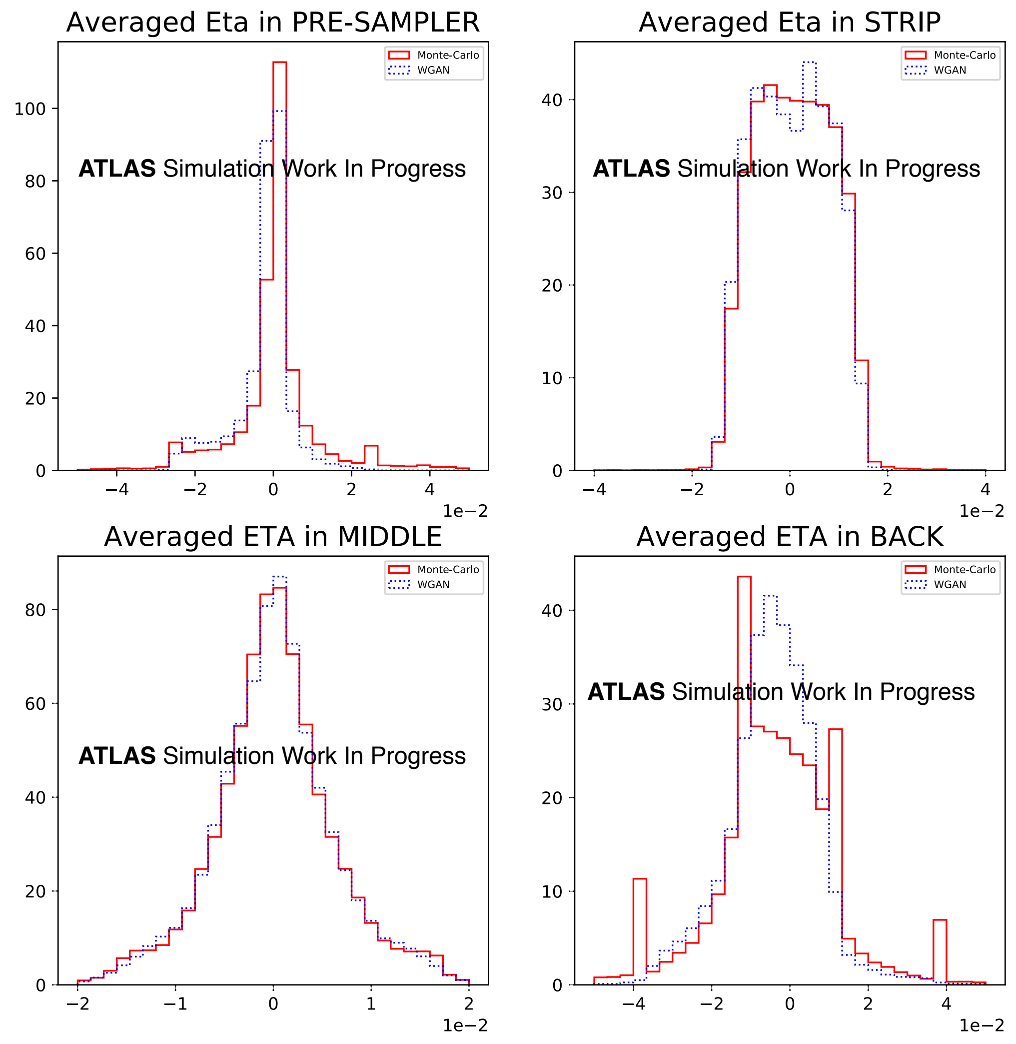


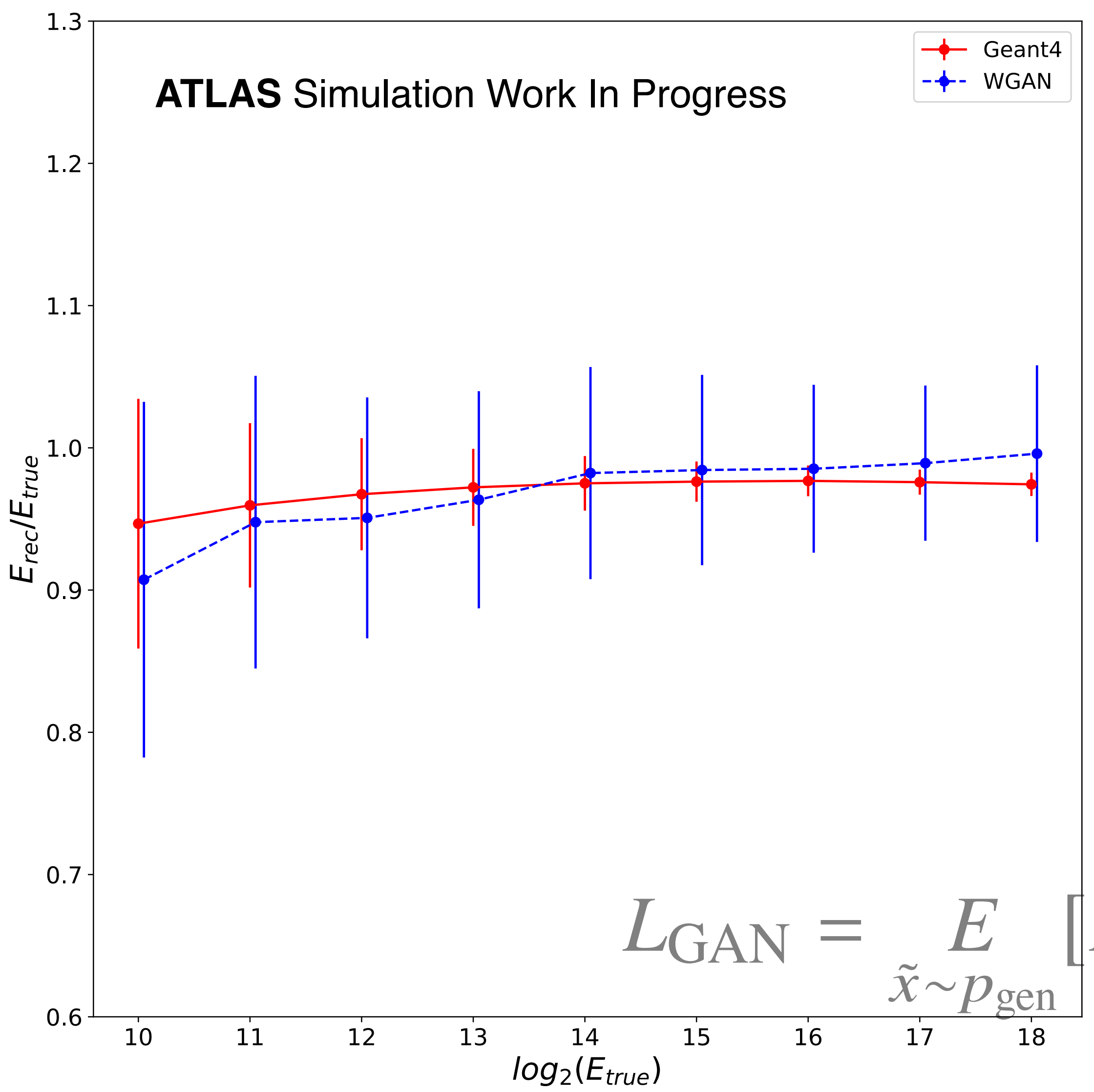
No Grad Penalty on TE

$$L_{\text{GAN}} = \underset{\tilde{x}\sim p_{\text{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\text{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

Remove Grad Penalty for Total Energy (TE)

Critic Input : 266 Cells

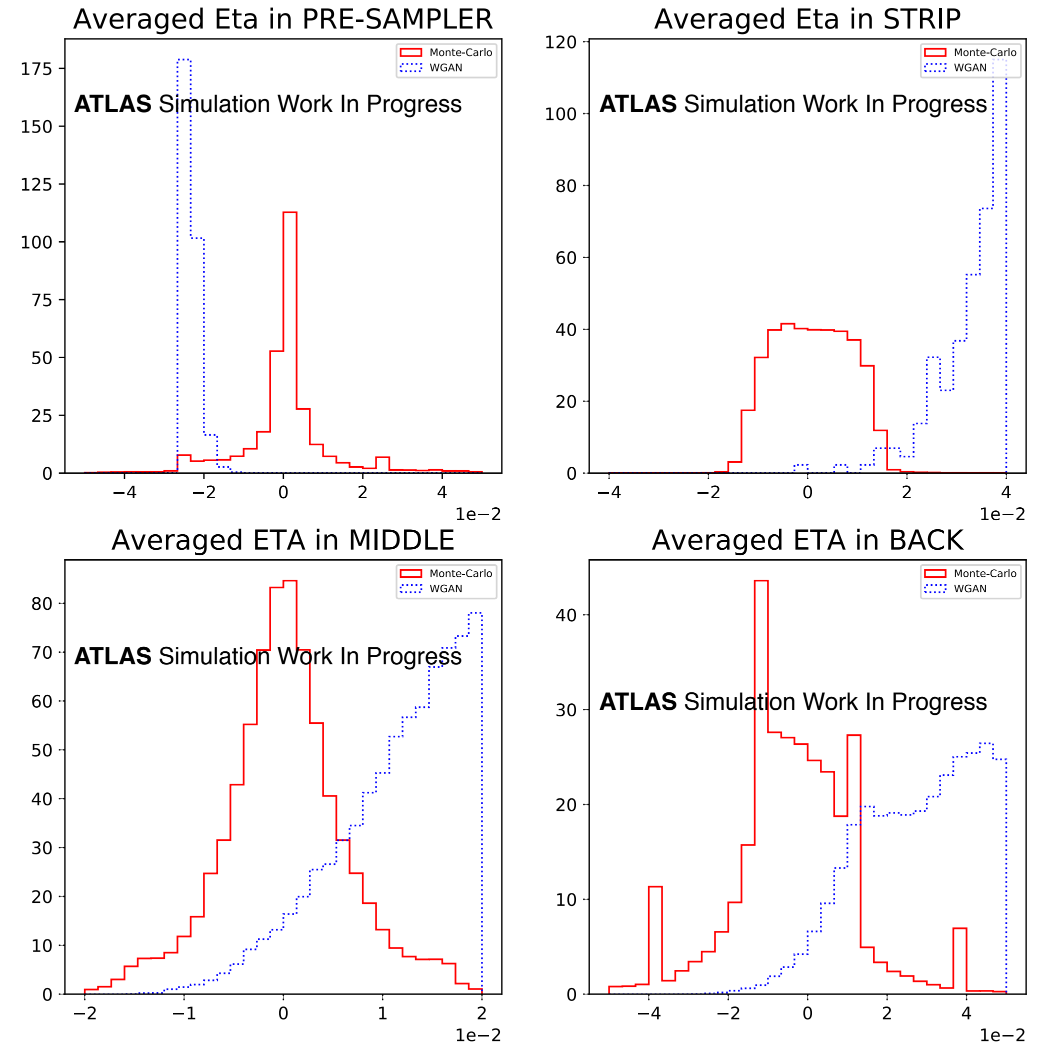


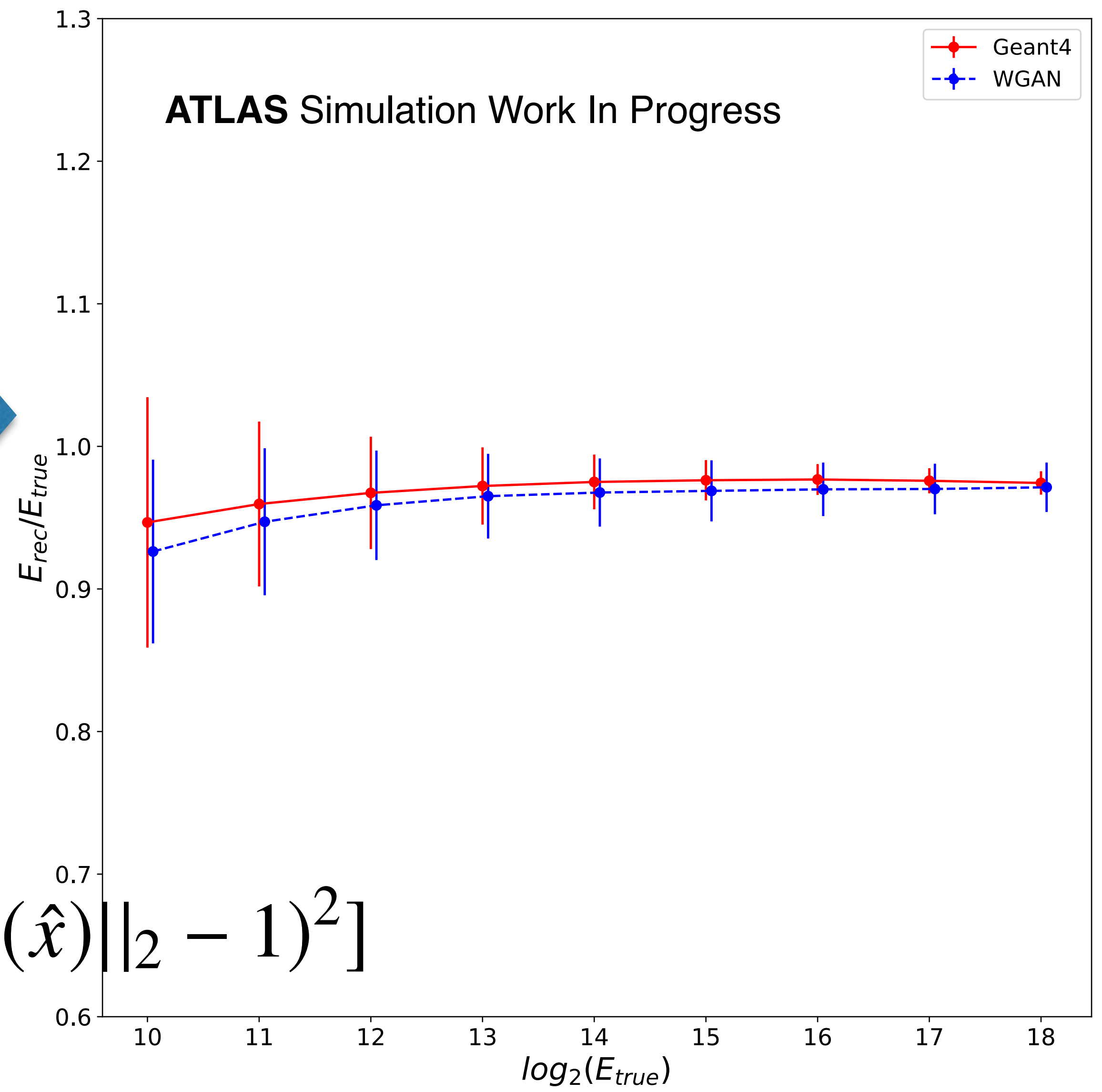


Plots get worse

Energy gets better

Critic Input : 266 Cells + Sum





No Grad Penalty on TE

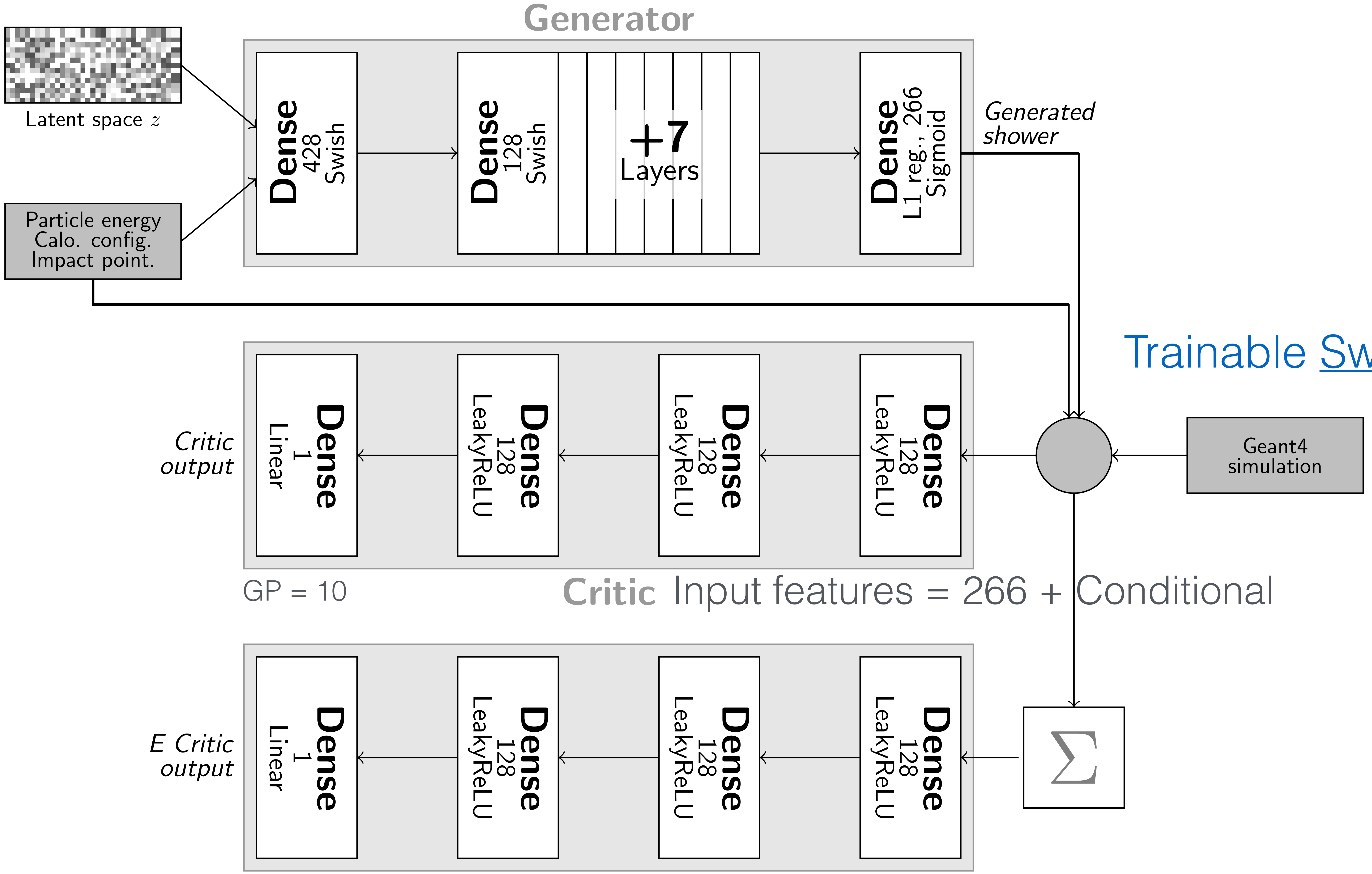
$$L_{\text{GAN}} = \underset{\tilde{x}\sim p_{\text{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\text{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}}D(\hat{x})||_2 - 1)^2]$$

Trade-Off b/w Distributions and Total Energy: How to get the best of both?

Trade-Off b/w Distributions and Total Energy: How to get the best of both?

"Train the Generator against a Critic of each type!"
-Gilles Louppe (ATLAS ACE)

New GAN Architecture



GP = 10 **Critic** Input features = 266 + Conditional

GP = 1e-8 **E Critic** Input features = **1** + Conditional

2 Critics

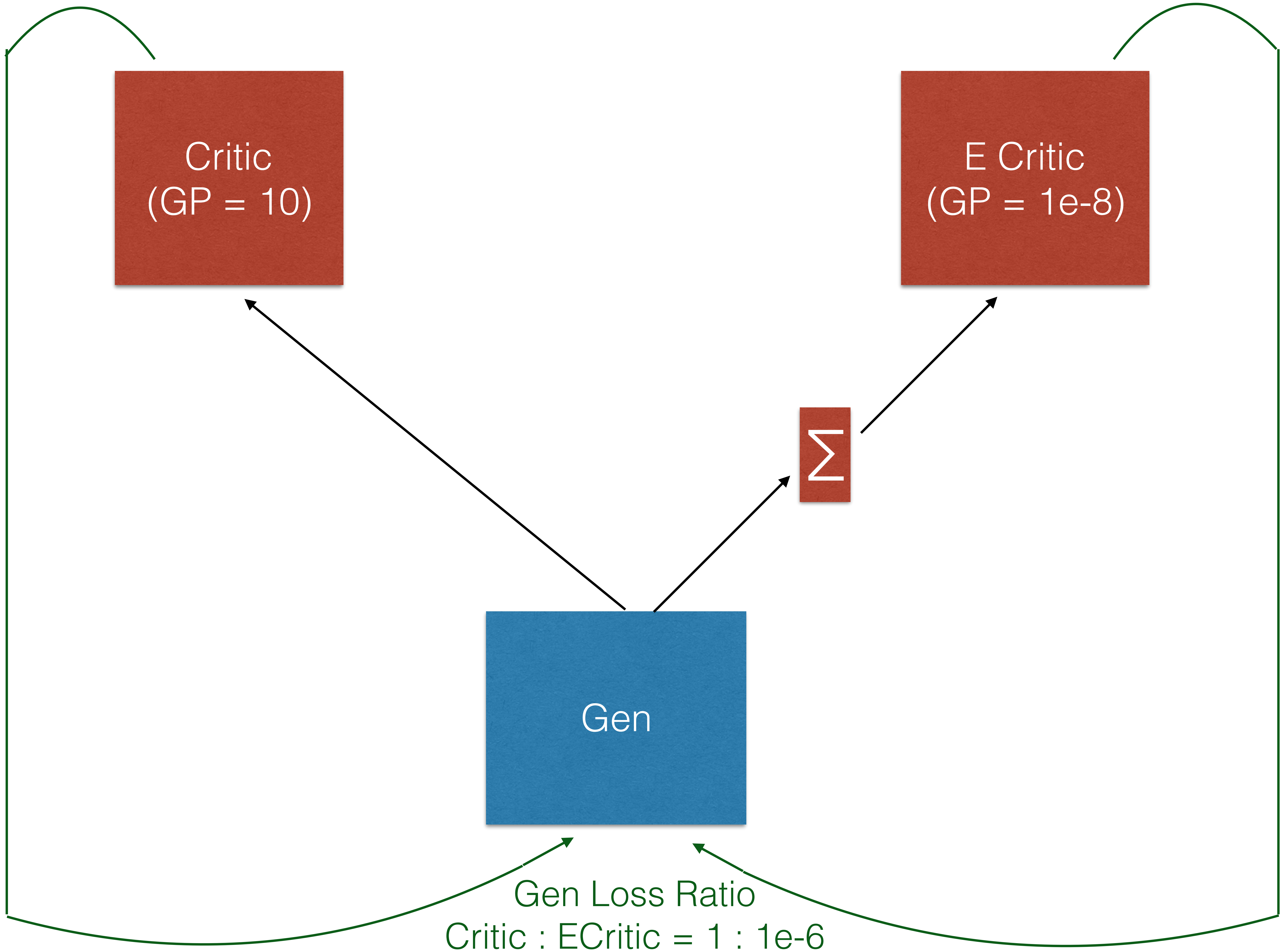
Deeper Generator needed

Trainable Swish activation for Generator

$$Swish(x) = x \cdot \mathrm{sigmoid}(\beta x)$$

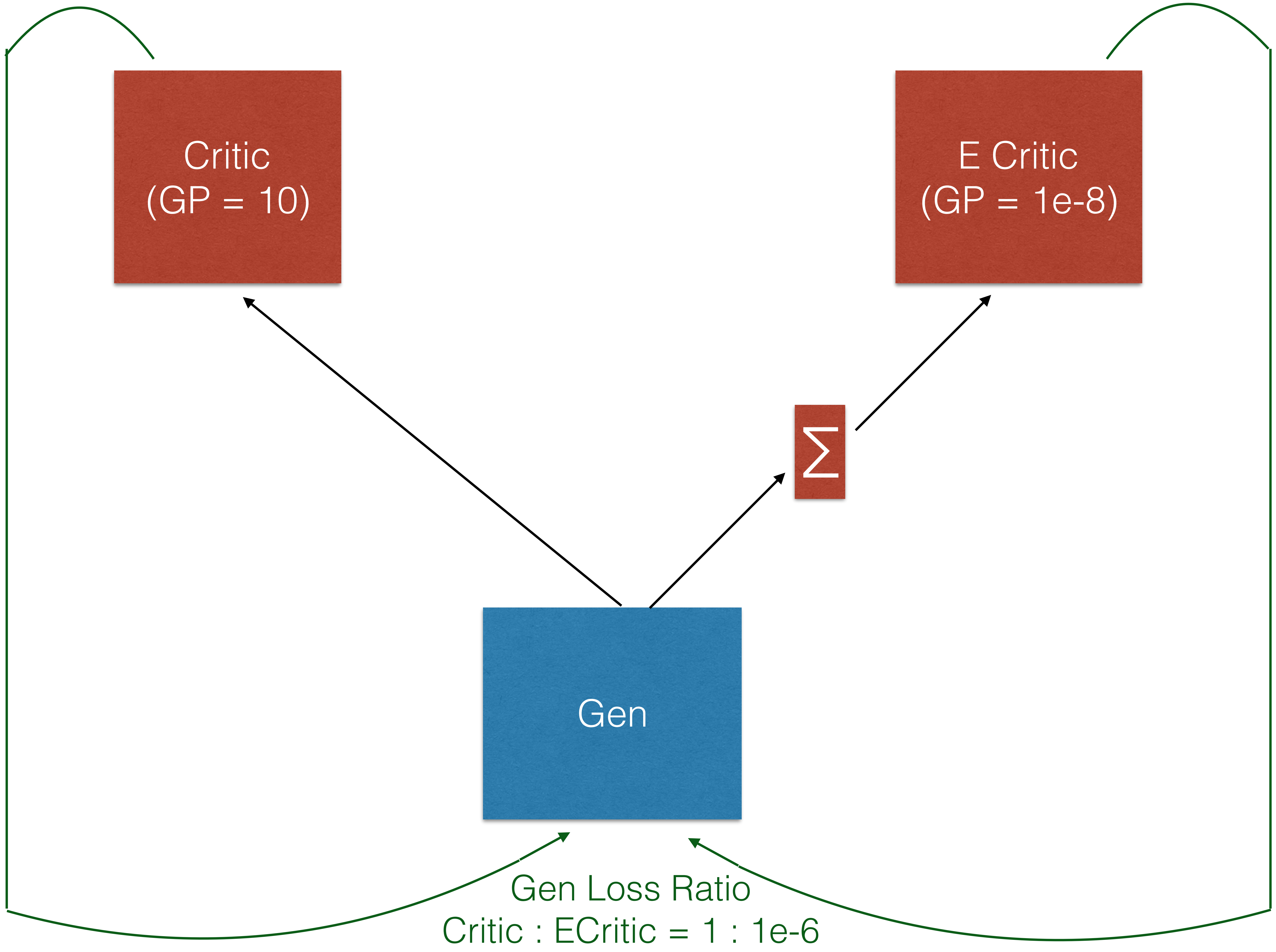
72 desecrate conditional combinations + 2 continuous conditions, doesn't even fit in one batch (64)

New GAN Architecture



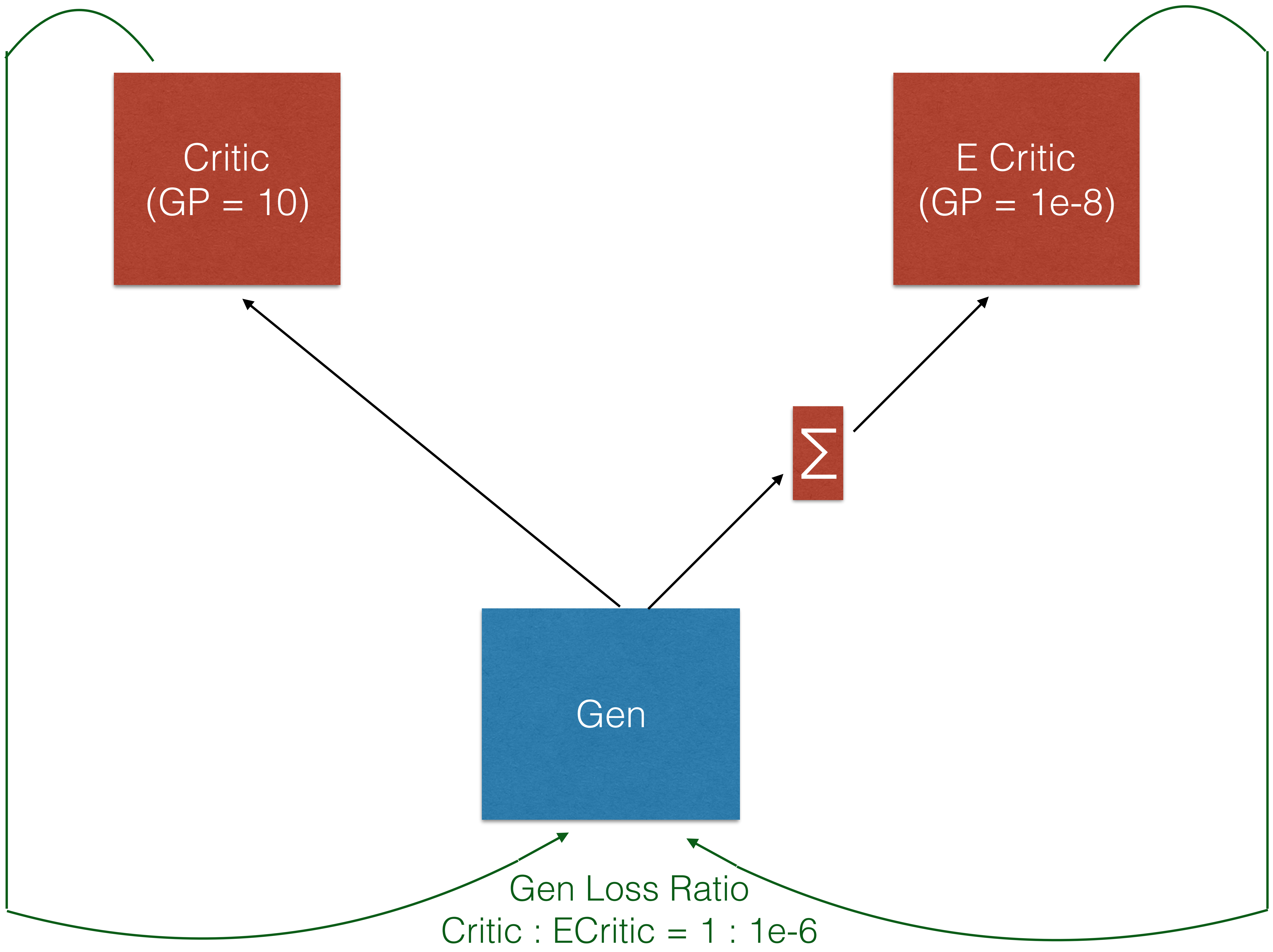
GP: Two Sided Gradient Penalty

New GAN Architecture



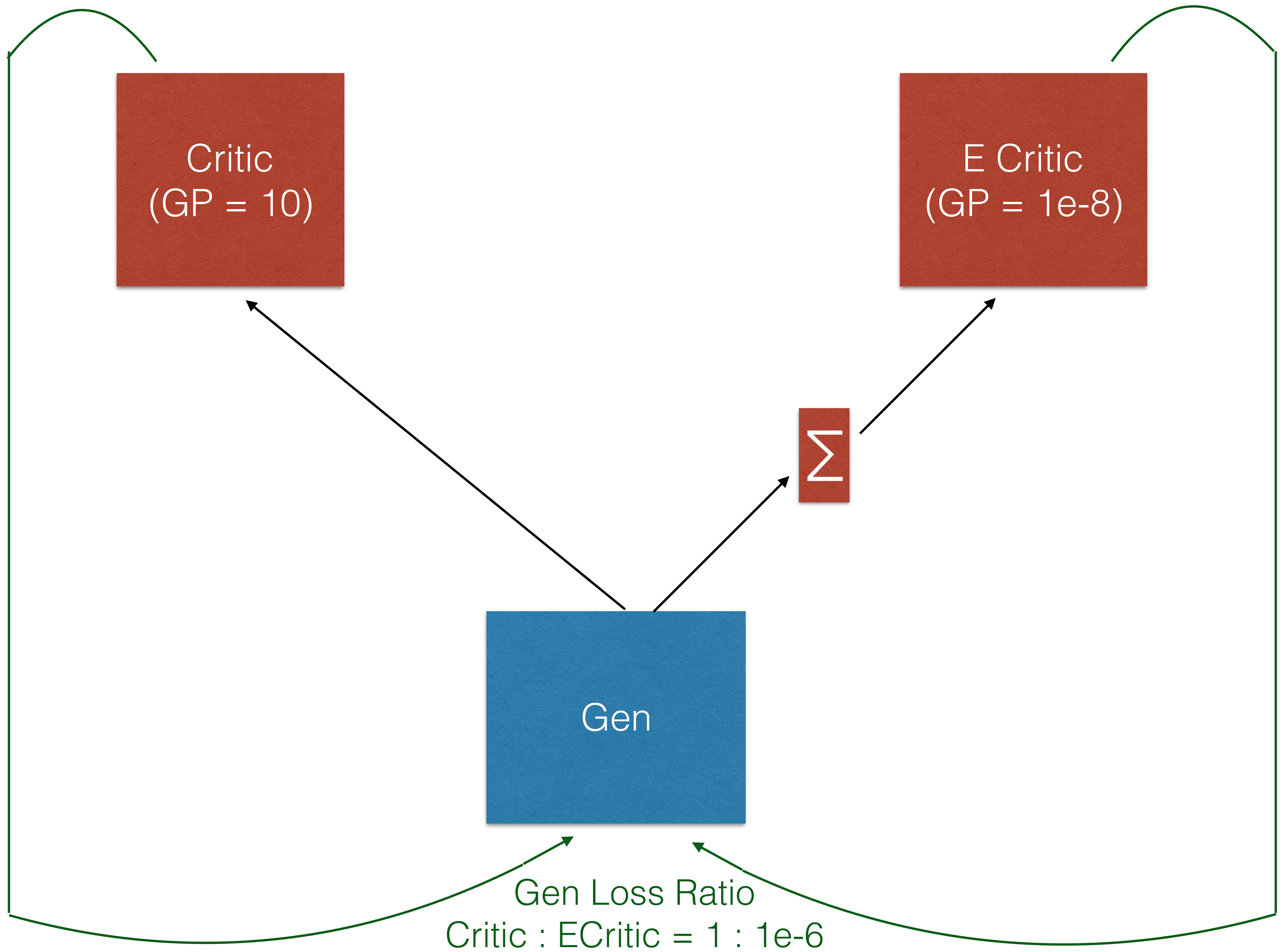
GP: Two Sided Gradient Penalty

New GAN Architecture



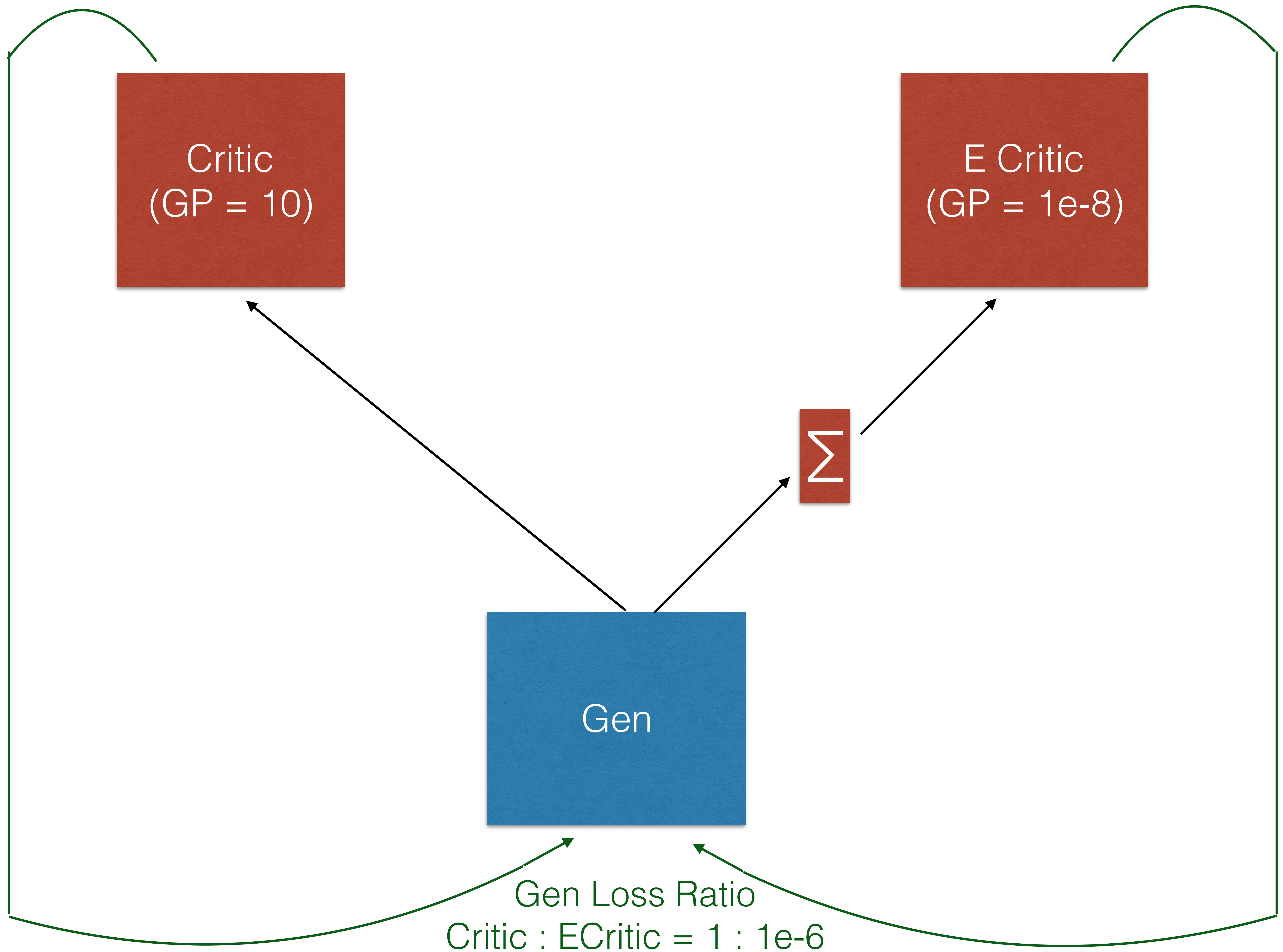
GP: Two Sided Gradient Penalty

New GAN Architecture



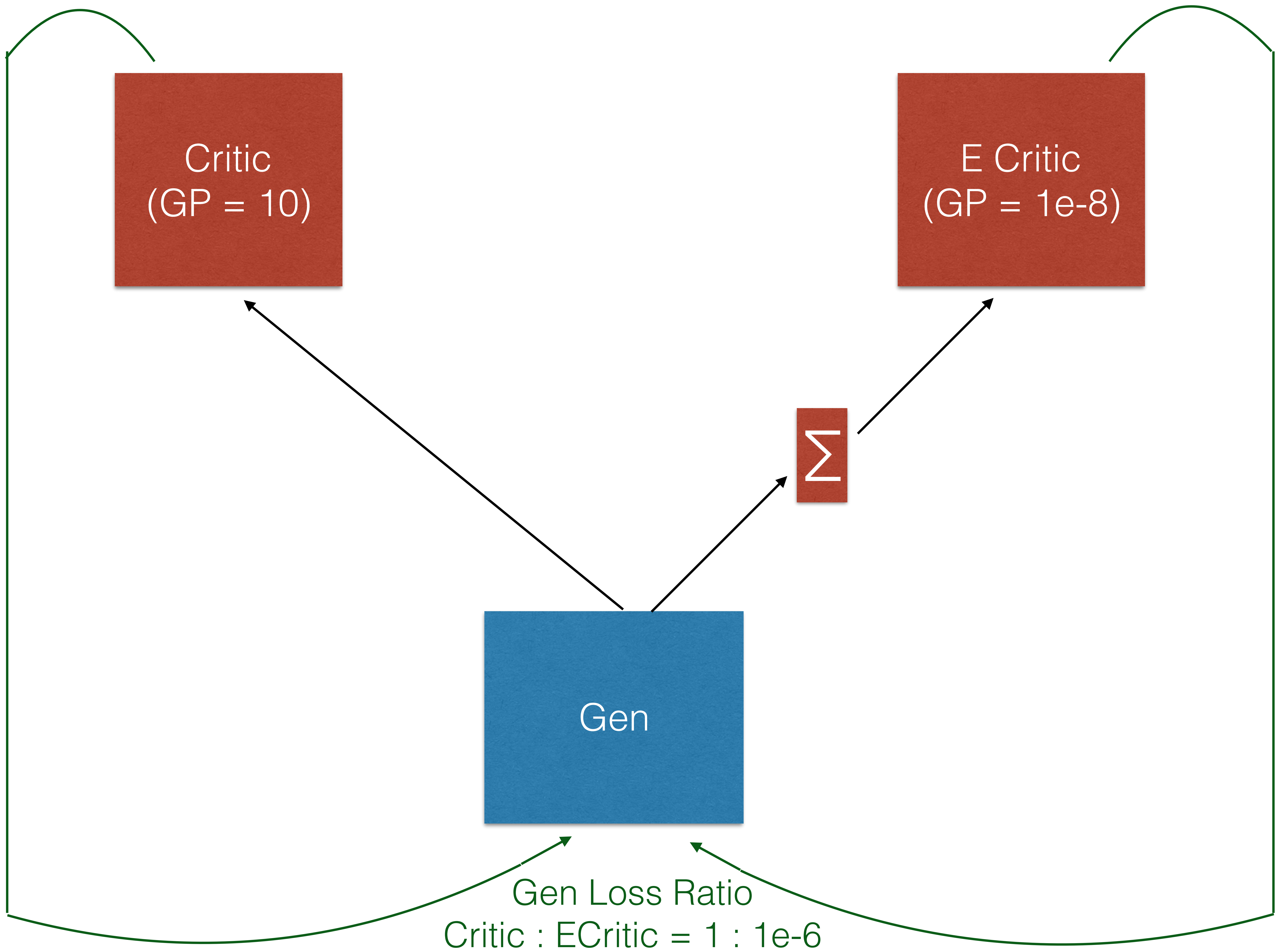
GP: Two Sided Gradient Penalty

New GAN Architecture



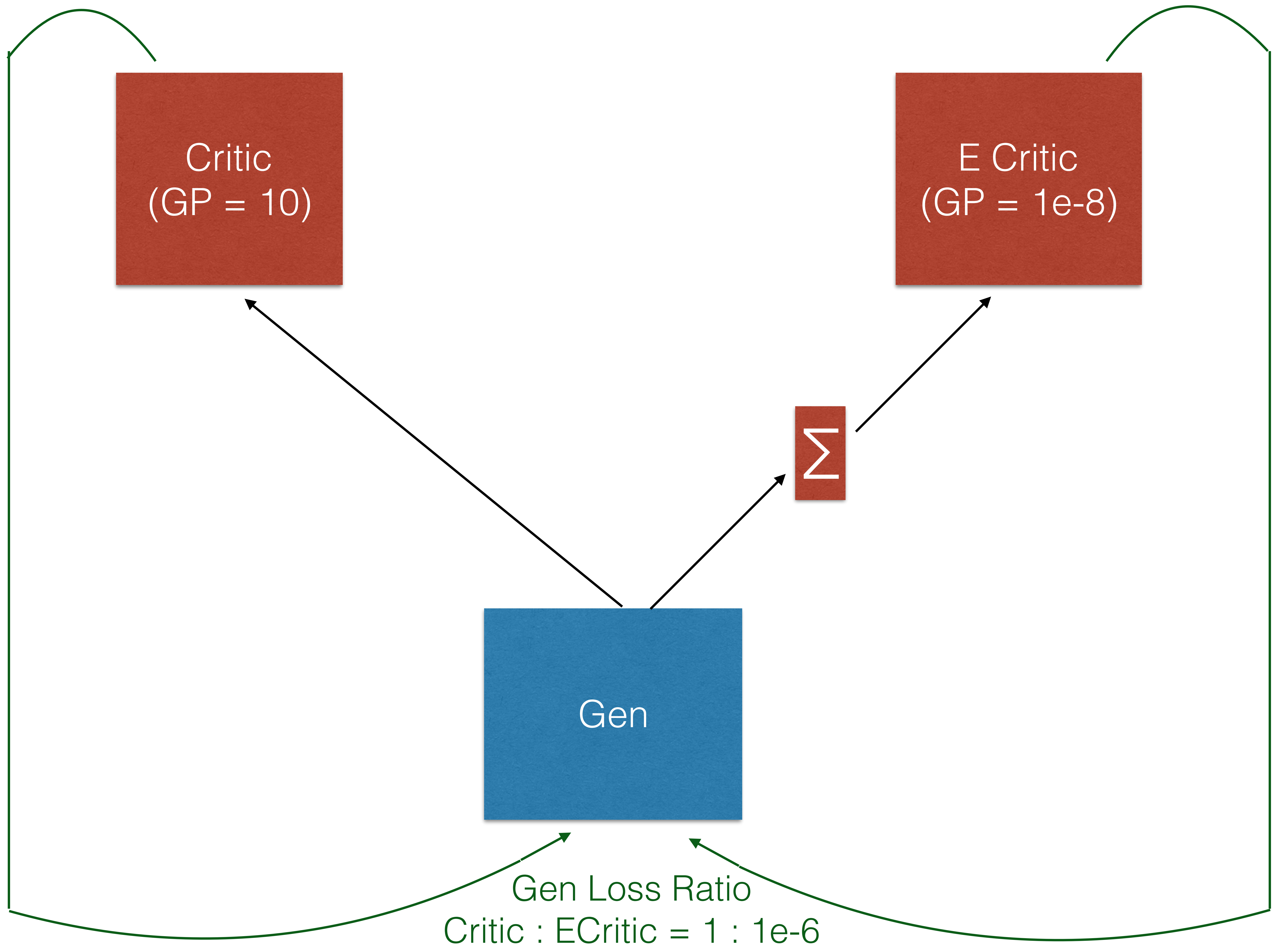
GP: Two Sided Gradient Penalty

New GAN Architecture



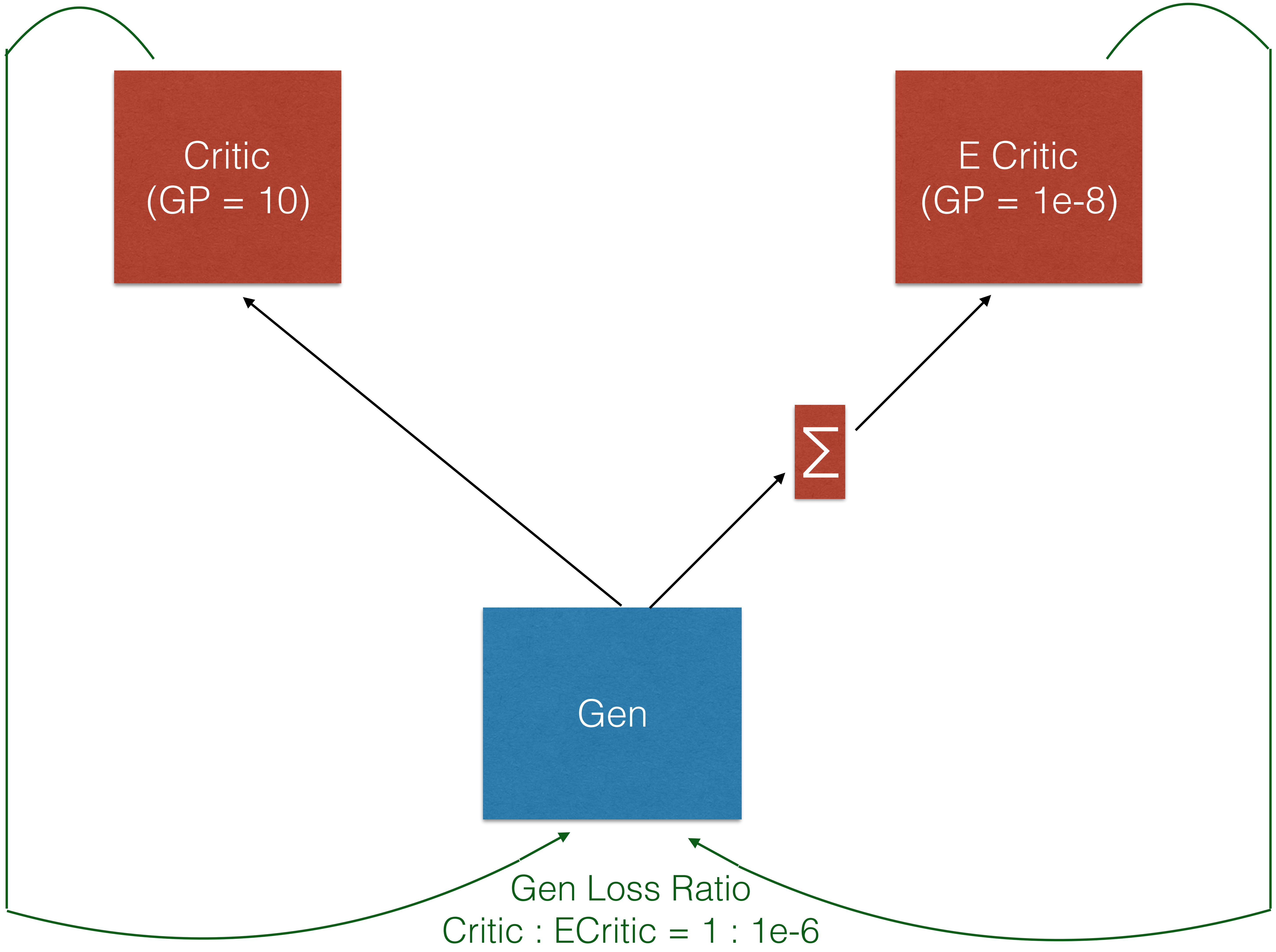
GP: Two Sided Gradient Penalty

New GAN Architecture



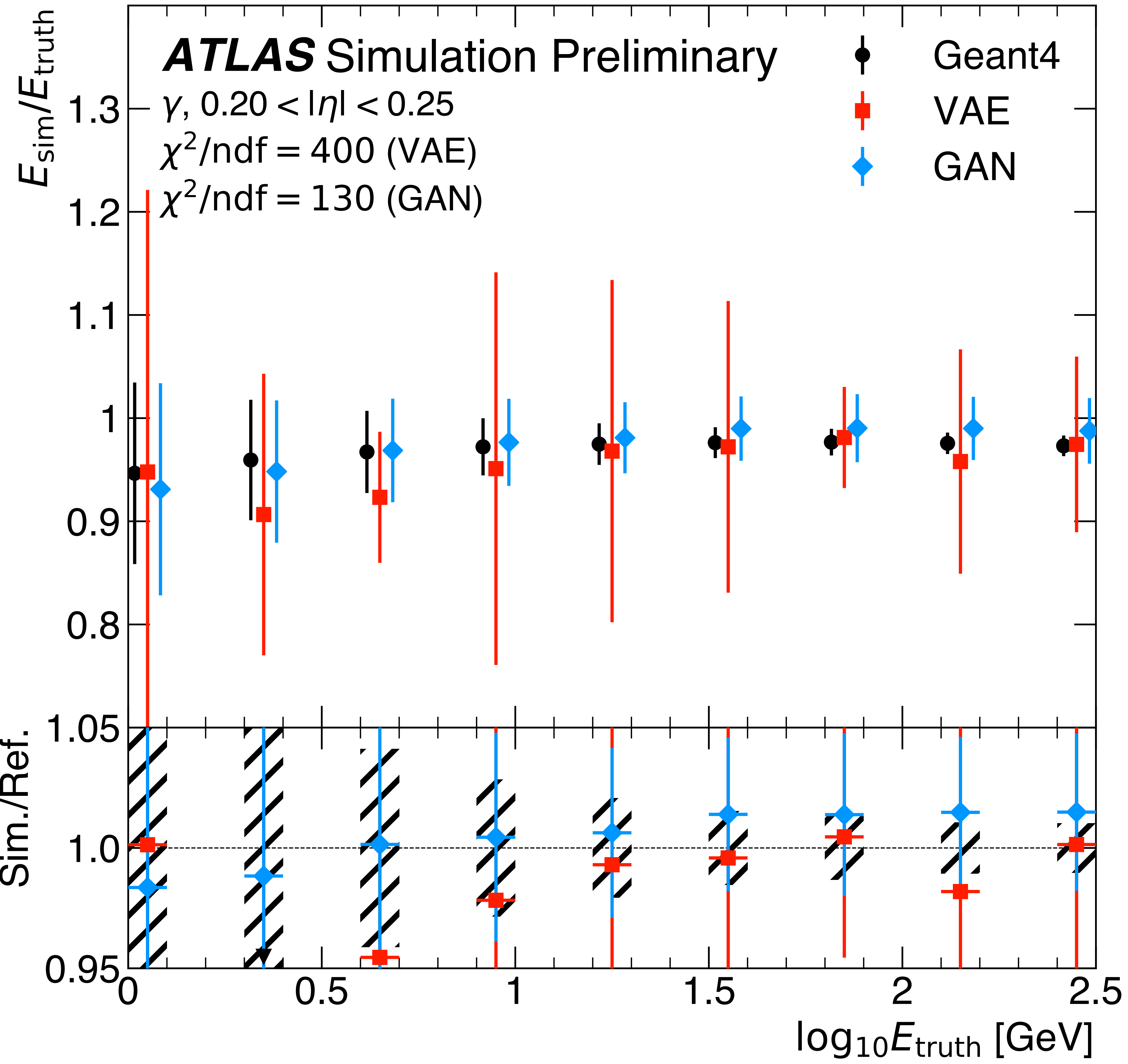
GP: Two Sided Gradient Penalty

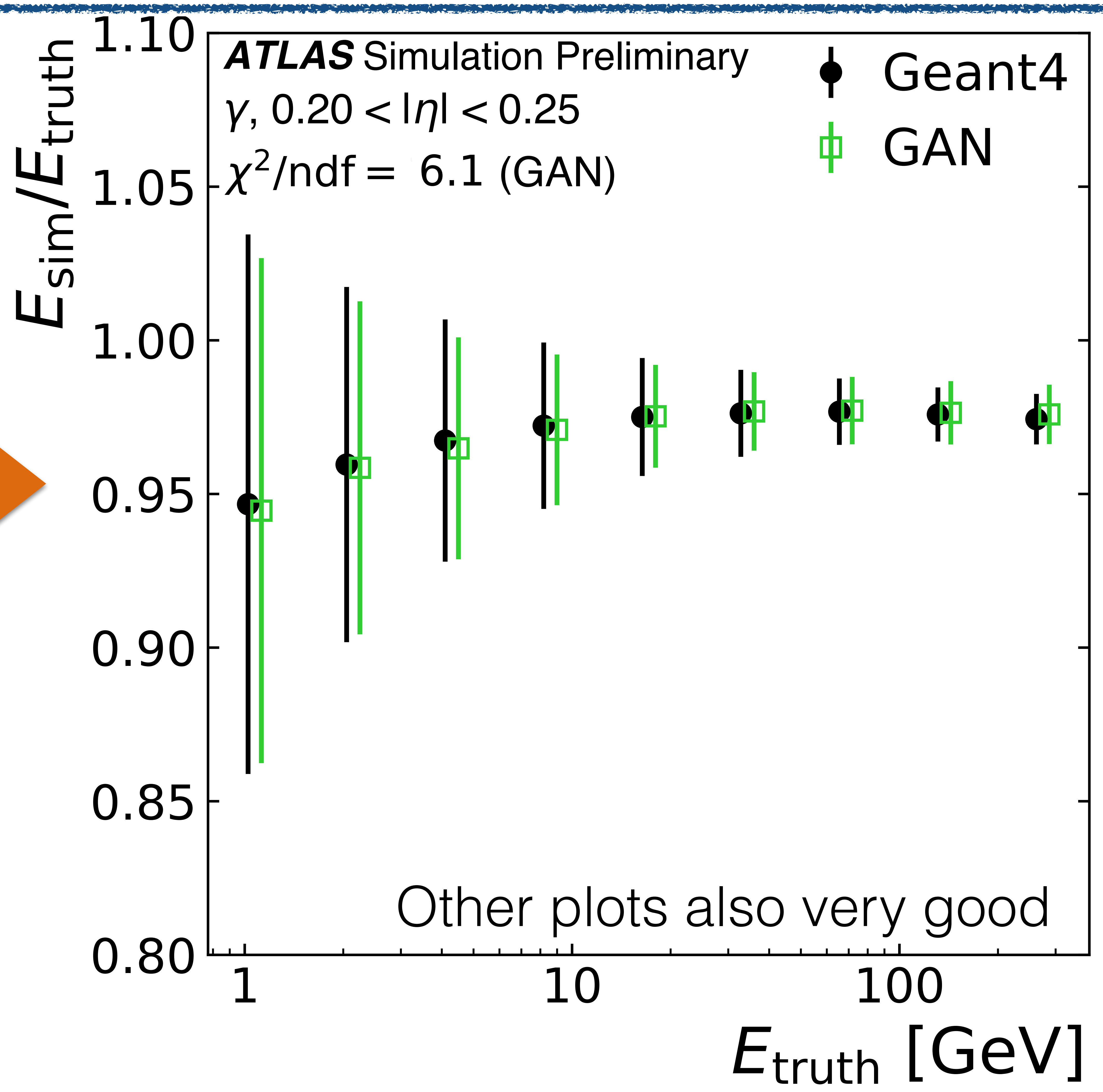
New GAN Architecture



GP: Two Sided Gradient Penalty

GAN: Improved Energy Resolution



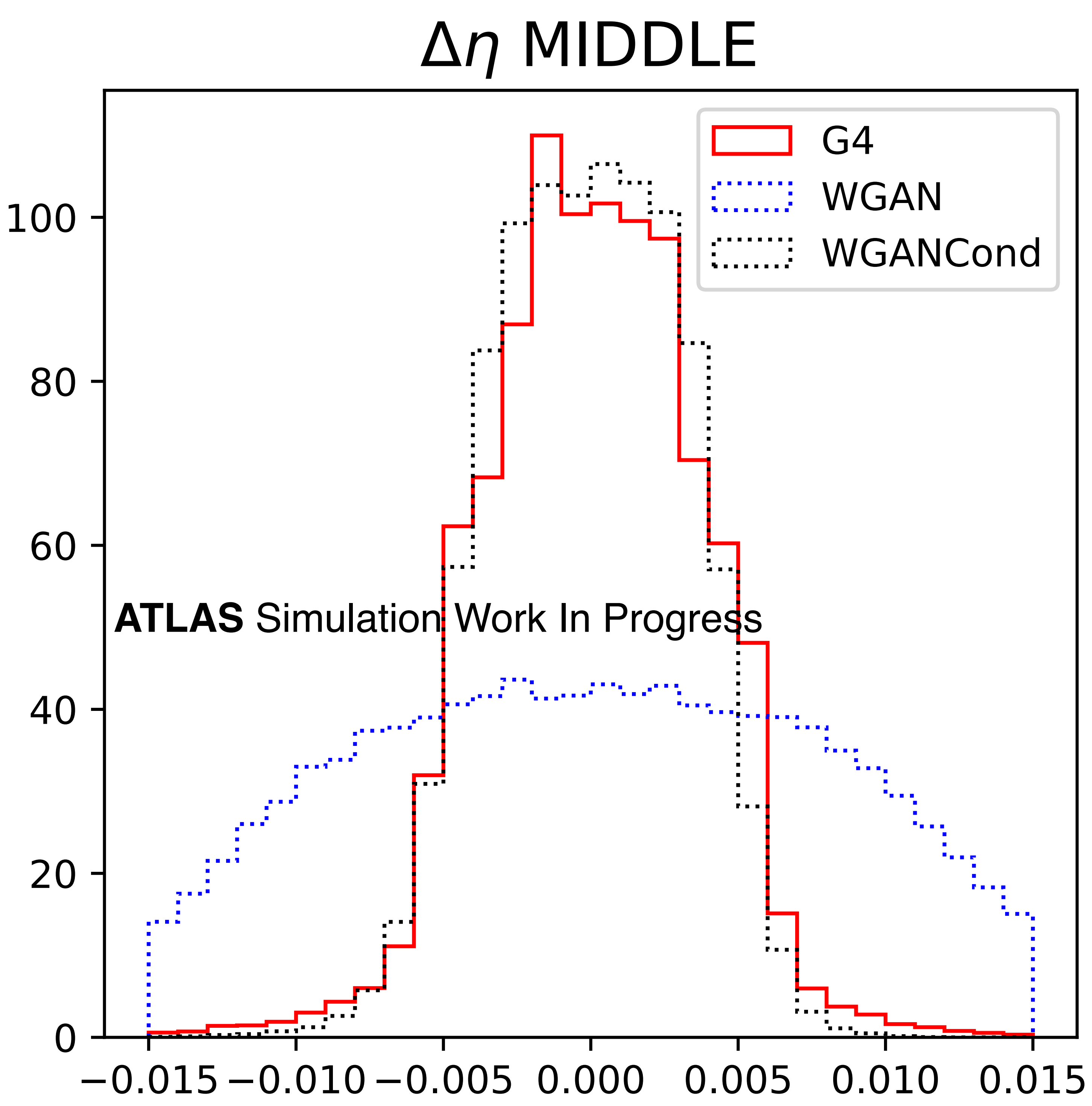


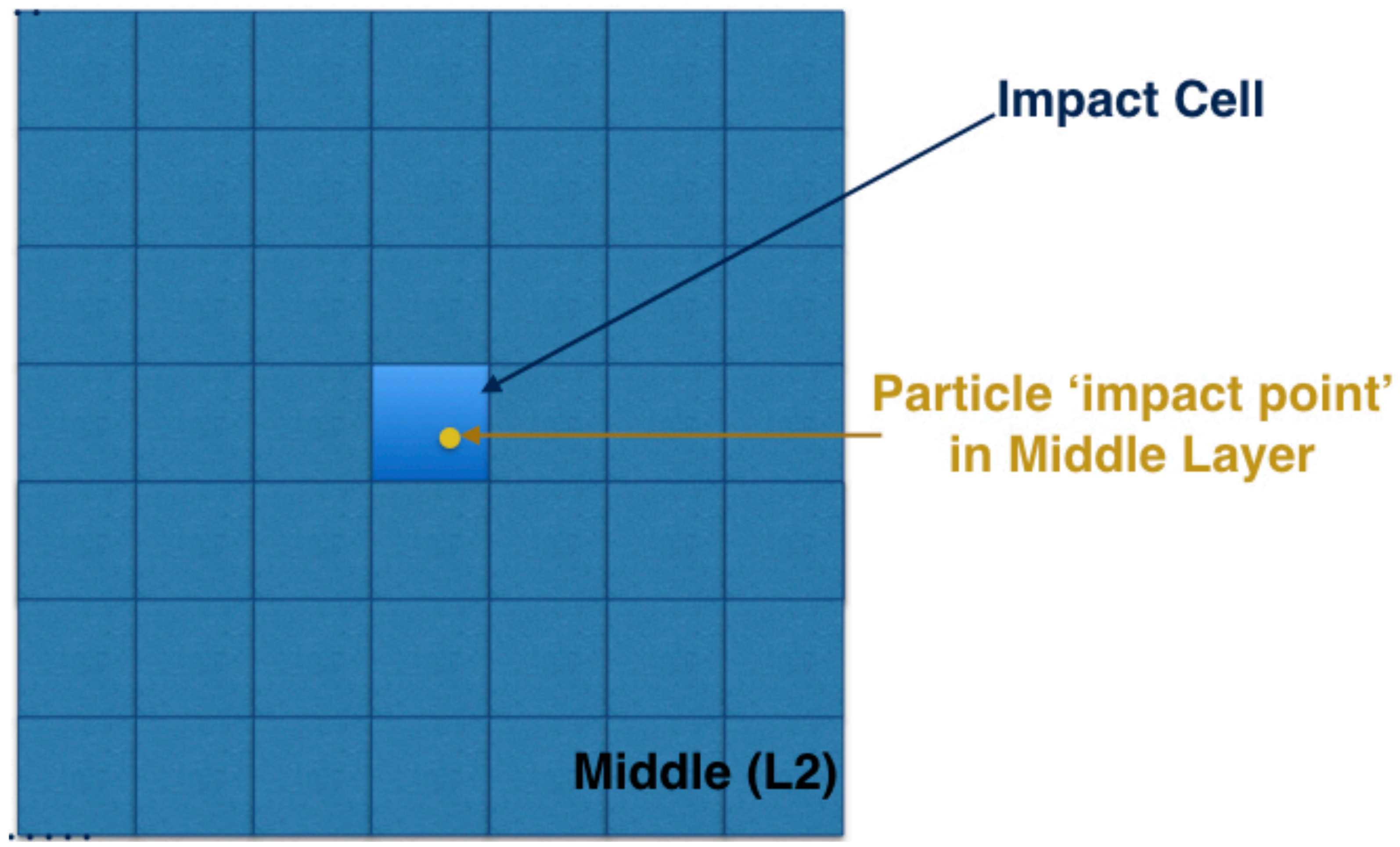
Reference

Simplified validation, before ATLAS software integration

Condition GAN also on Impact Position of Particle

<u>Average η - Particle's Impact η</u>

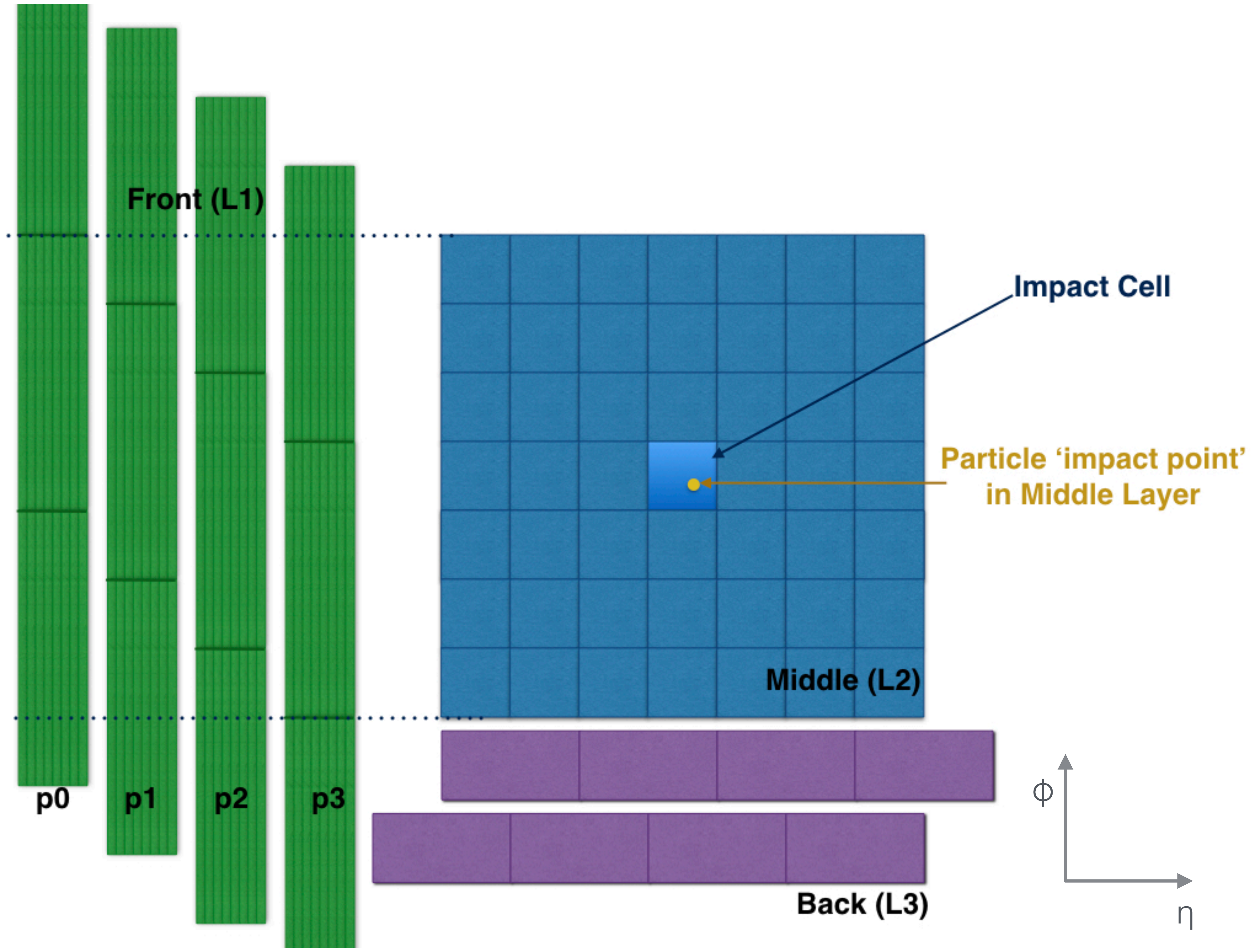


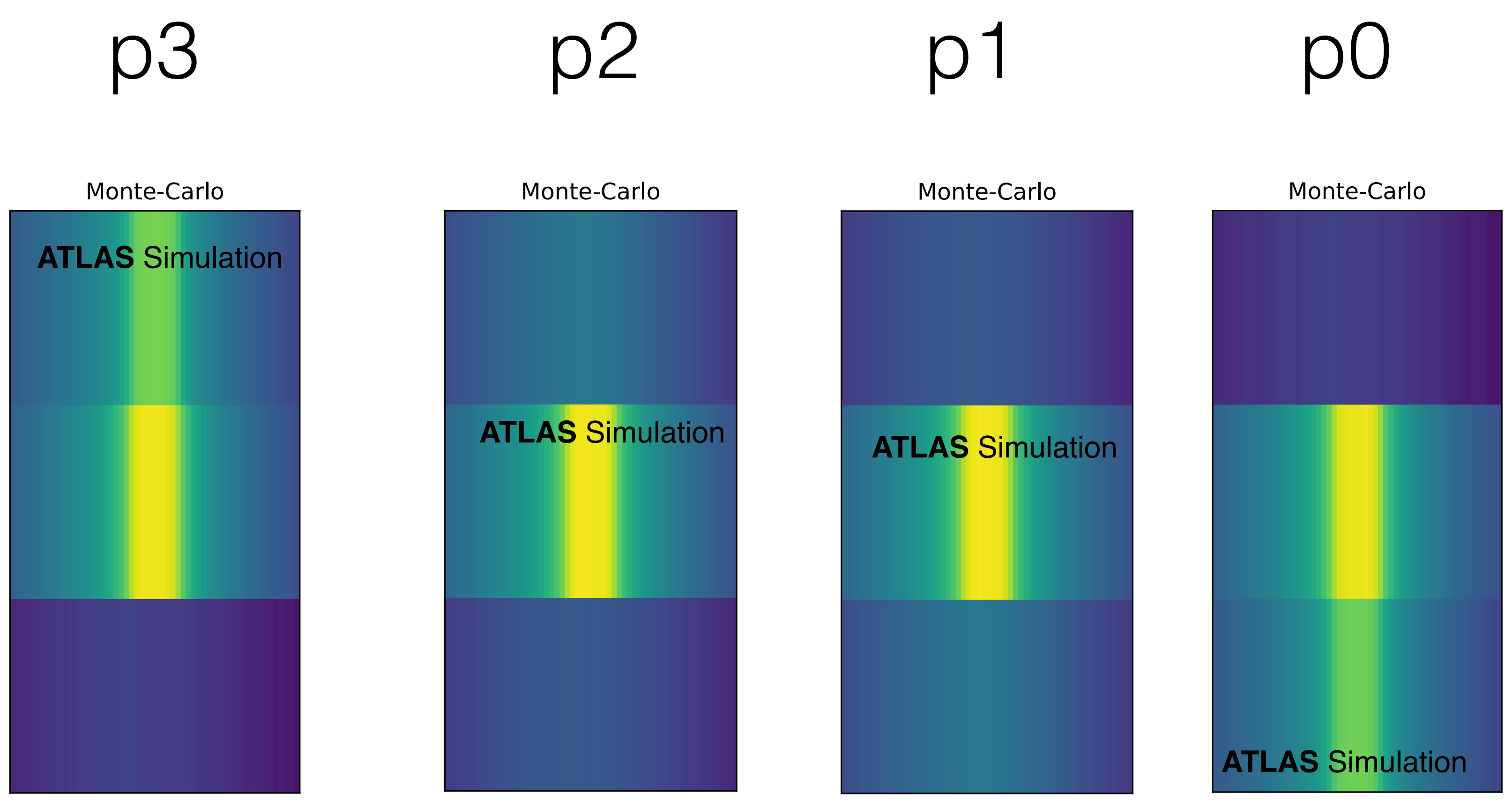


Continuous variable,
not class conditioning

GAN learns to centre the shower
around the particle position

Calorimeter Alignment Conditioning

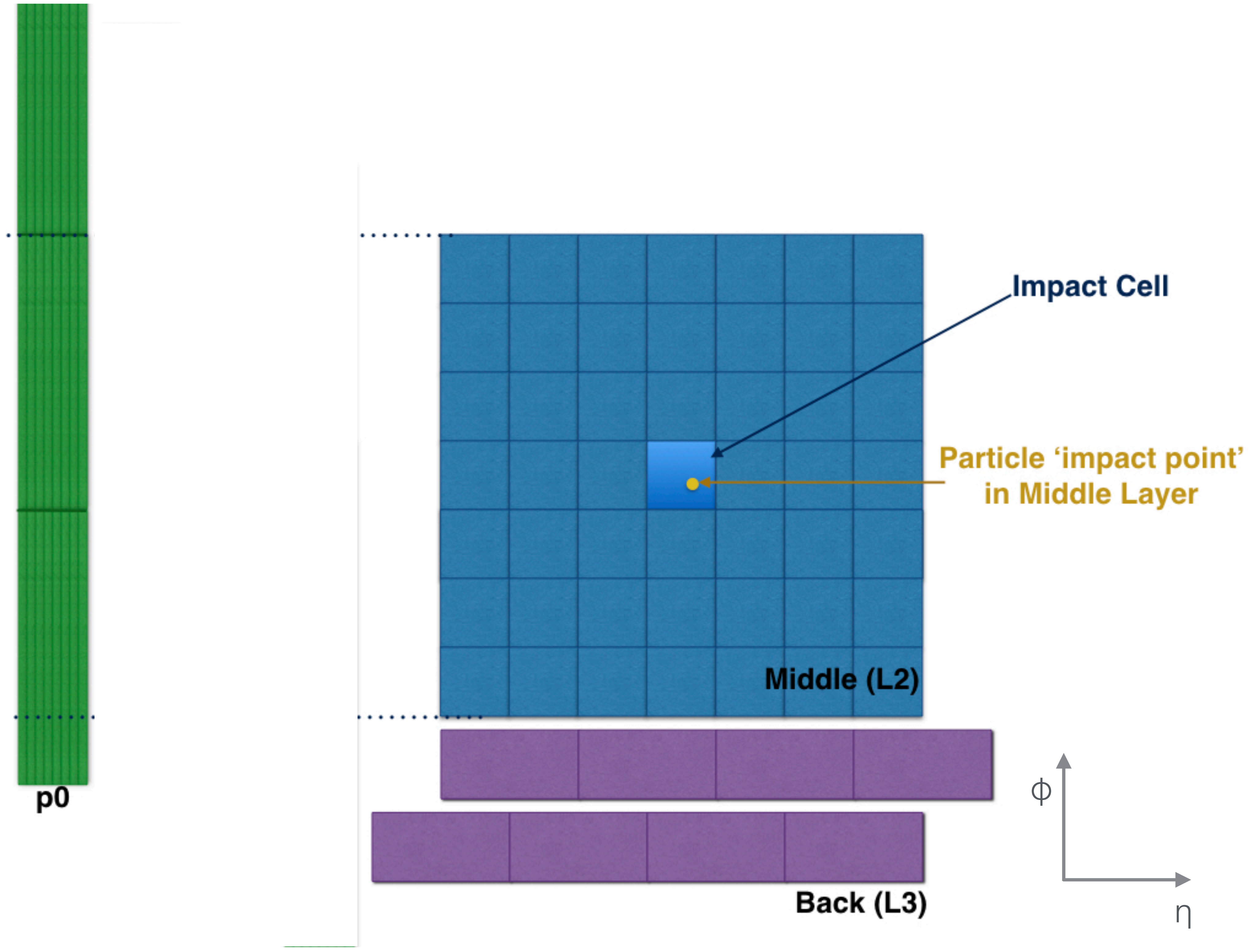


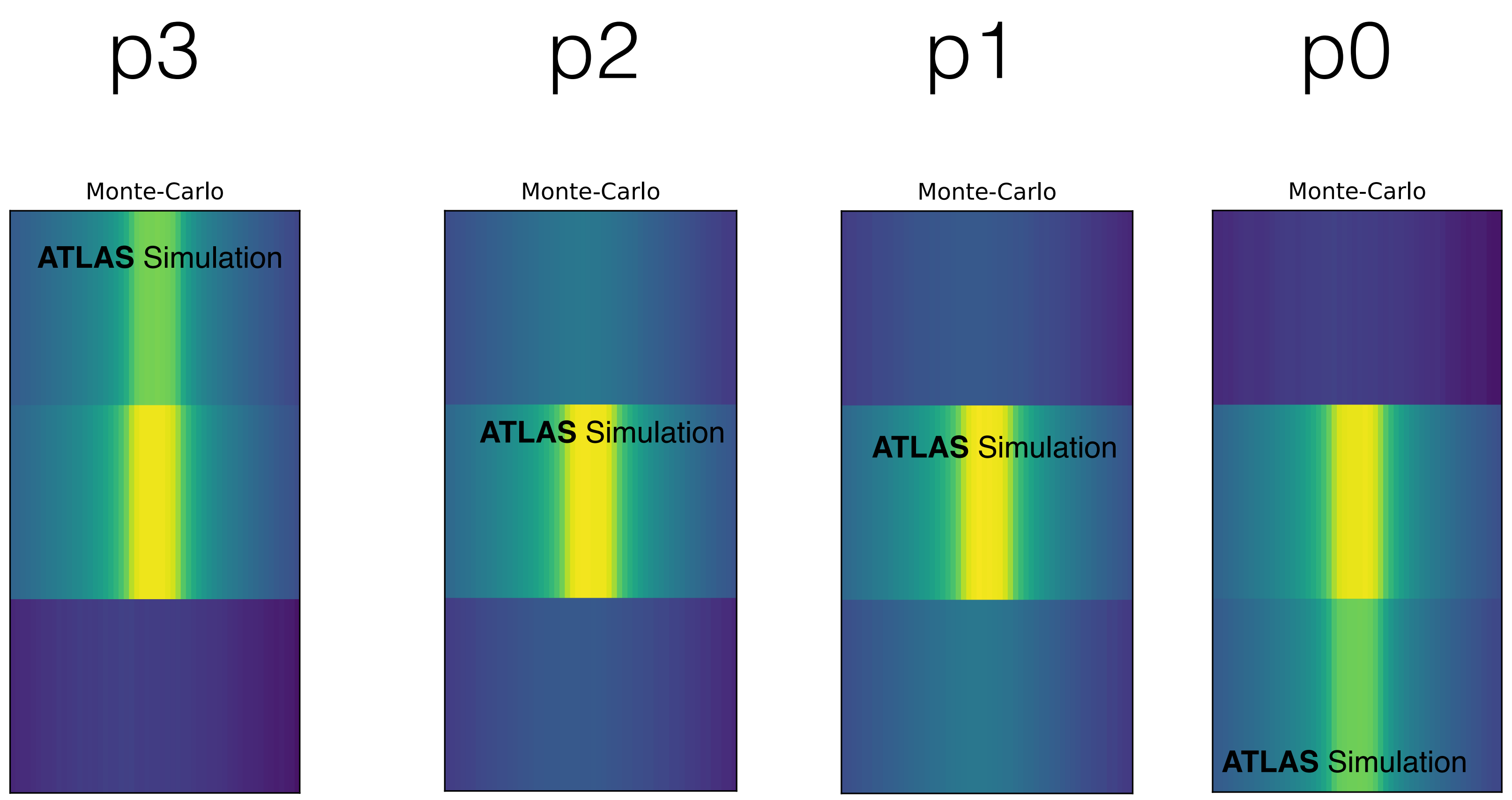


Hardest conditioning to get correct (HPO)

Two hot vector encoding

Calorimeter Alignment Conditioning

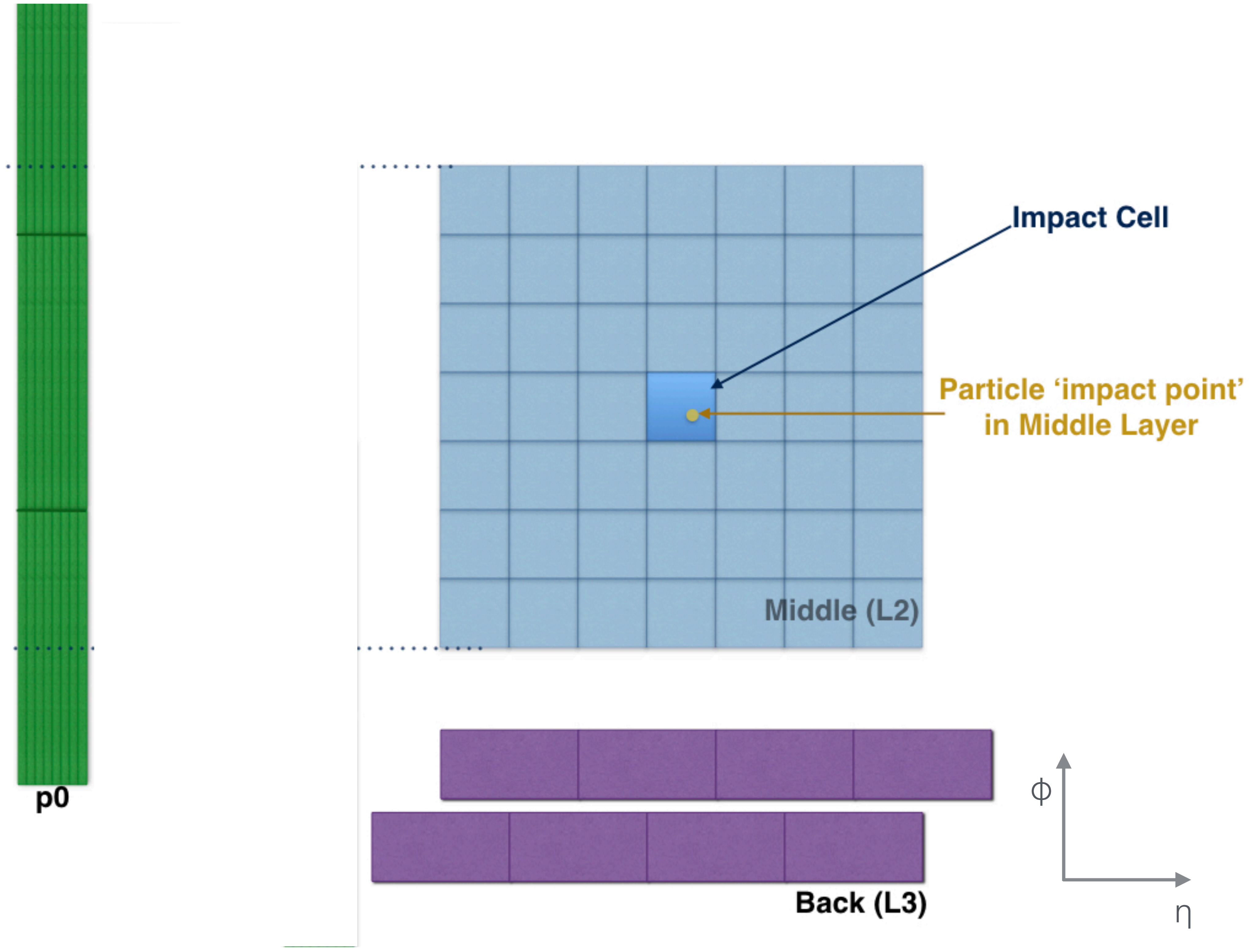


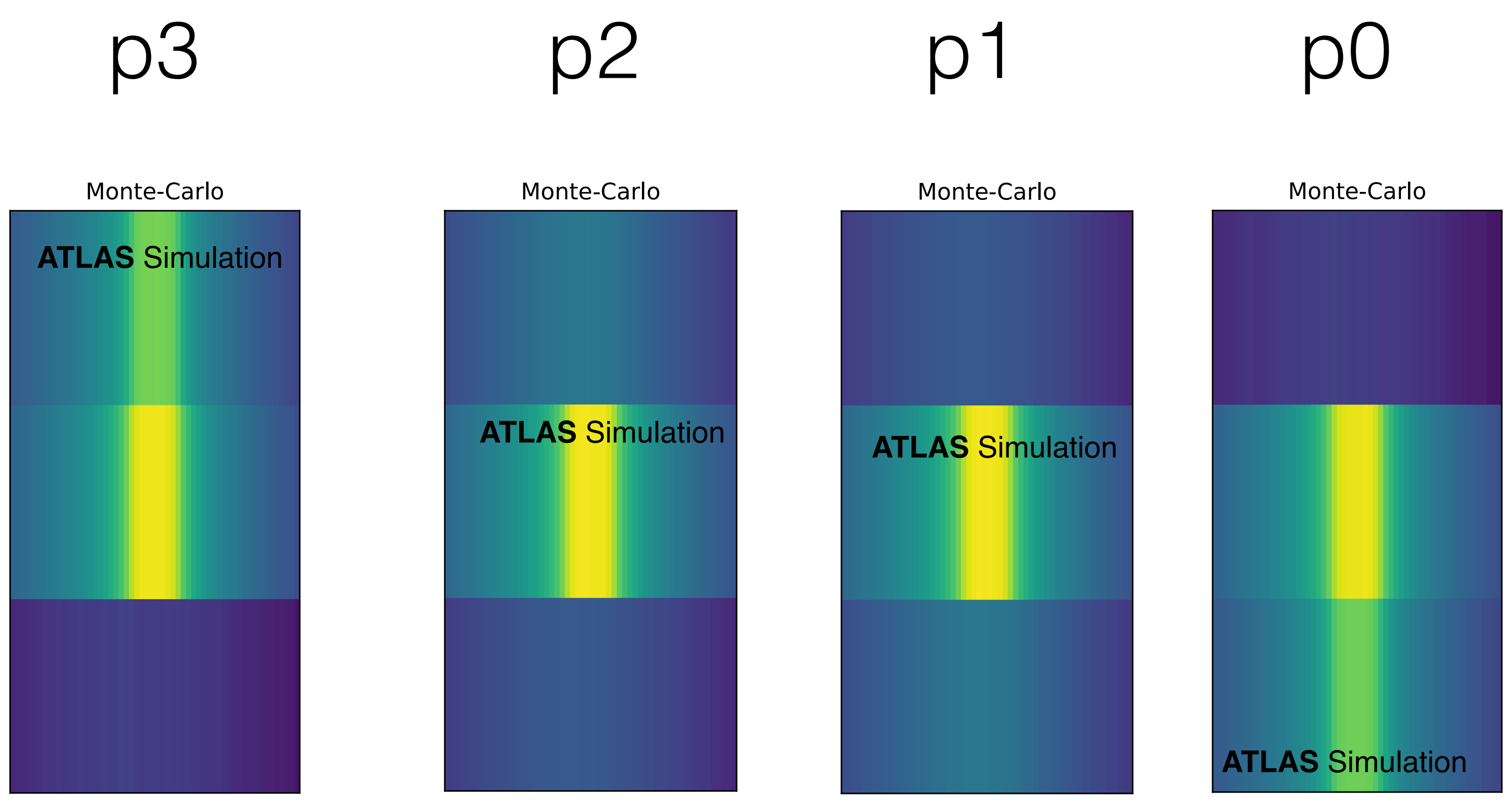


Hardest conditioning to get correct (HPO)

Two hot vector encoding

Calorimeter Alignment Conditioning

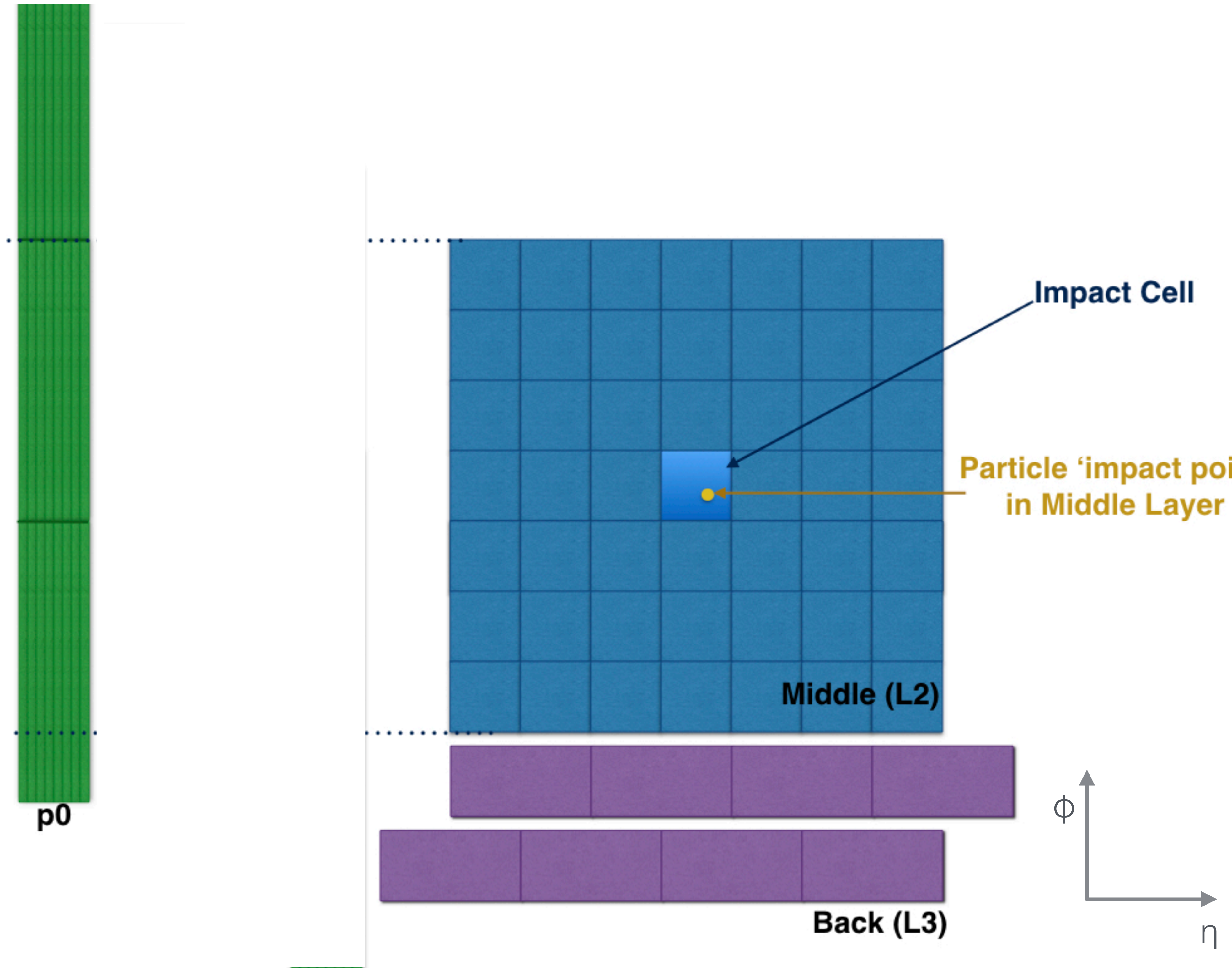


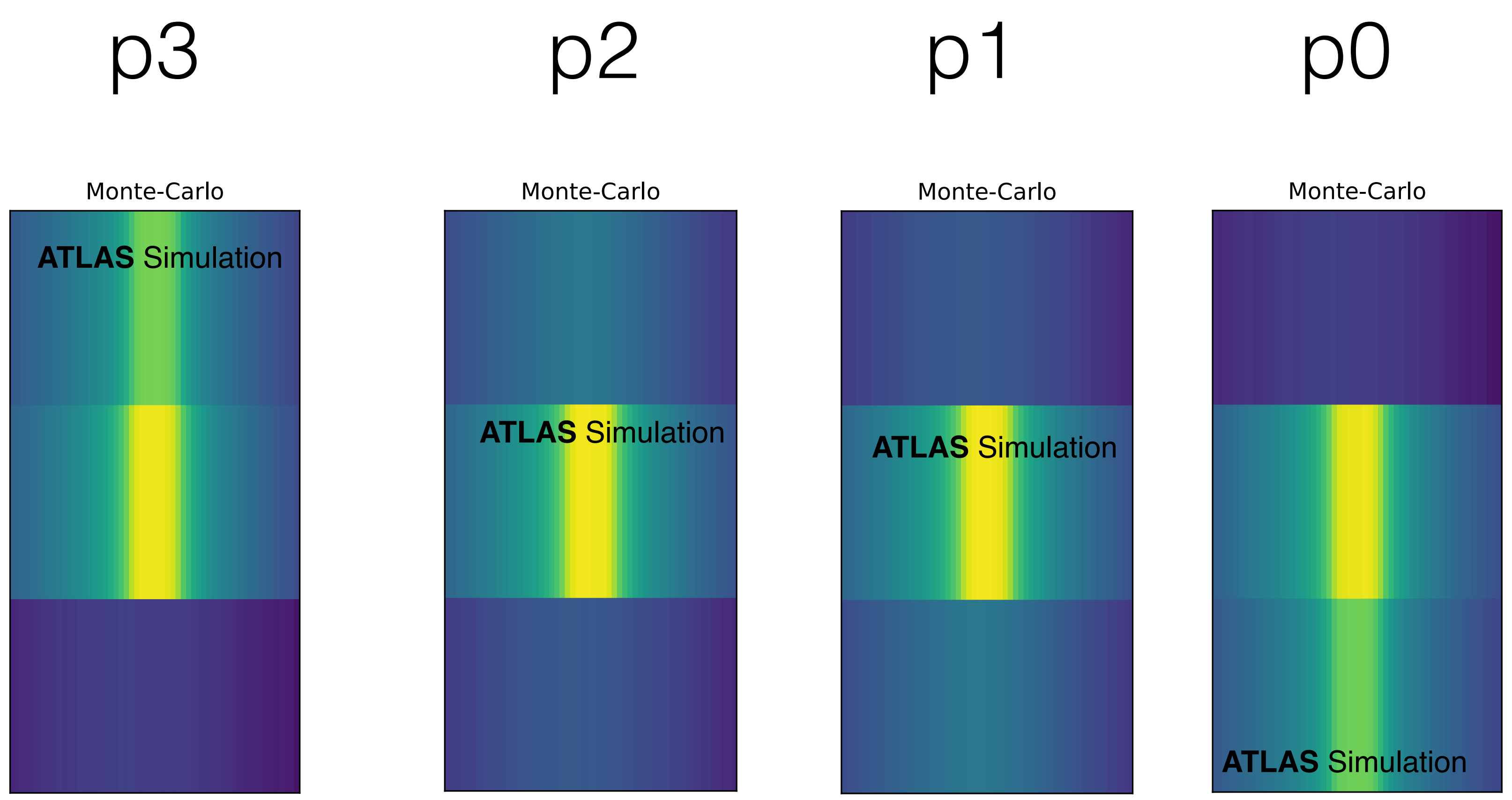


Hardest conditioning to get correct (HPO)

Two hot vector encoding

Calorimeter Alignment Conditioning

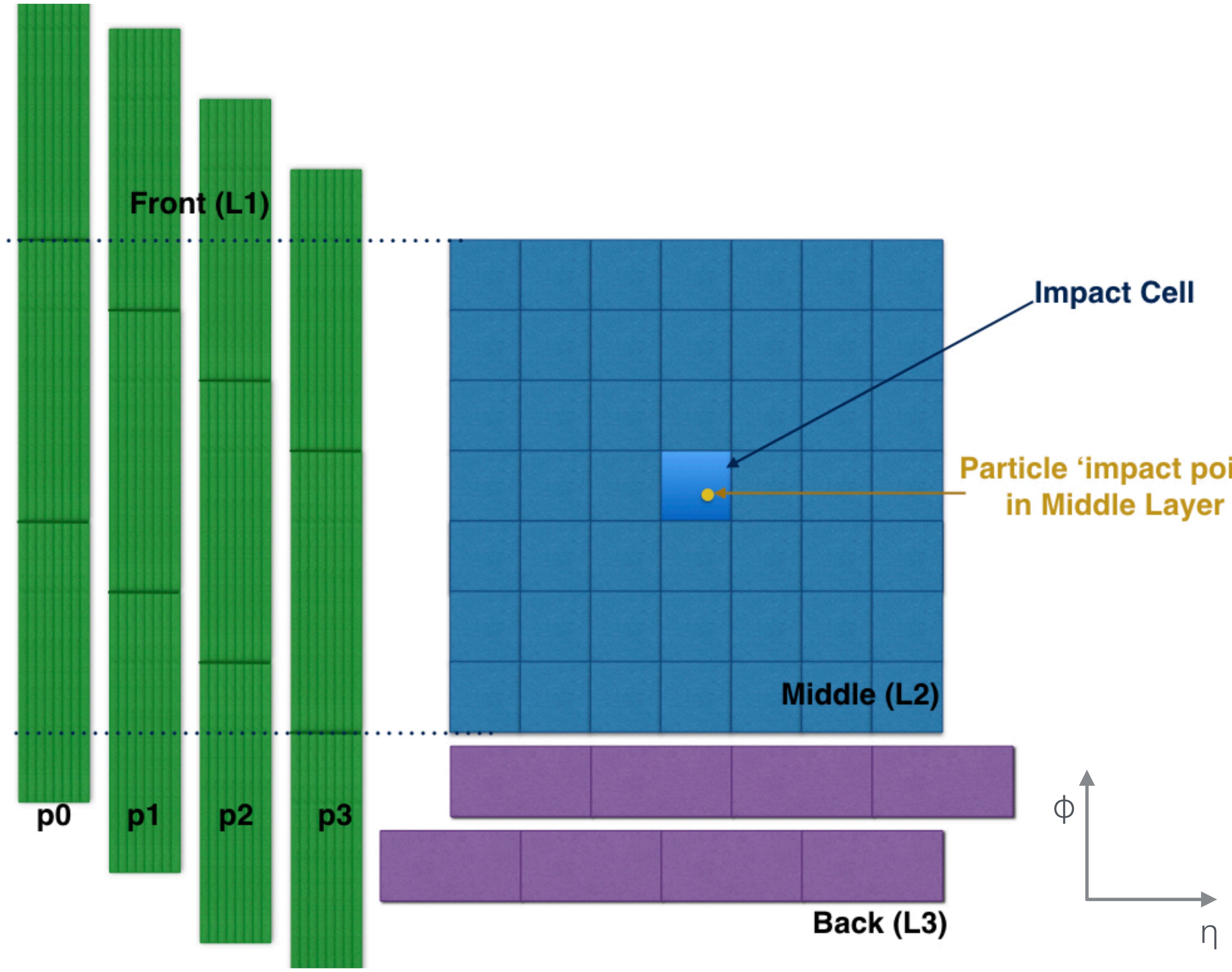


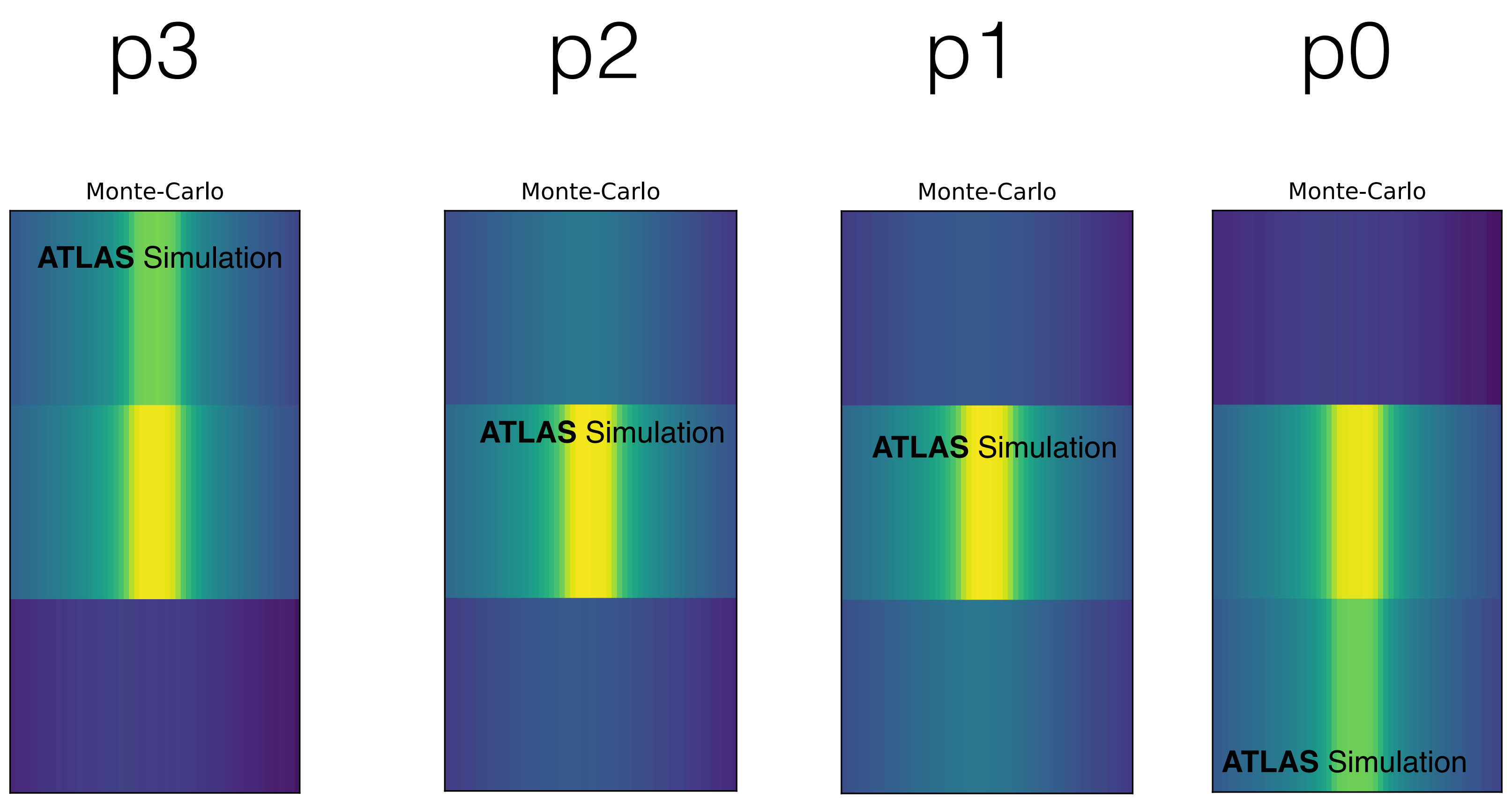


Hardest conditioning to get correct (HPO)

Two hot vector encoding

Calorimeter Alignment Conditioning

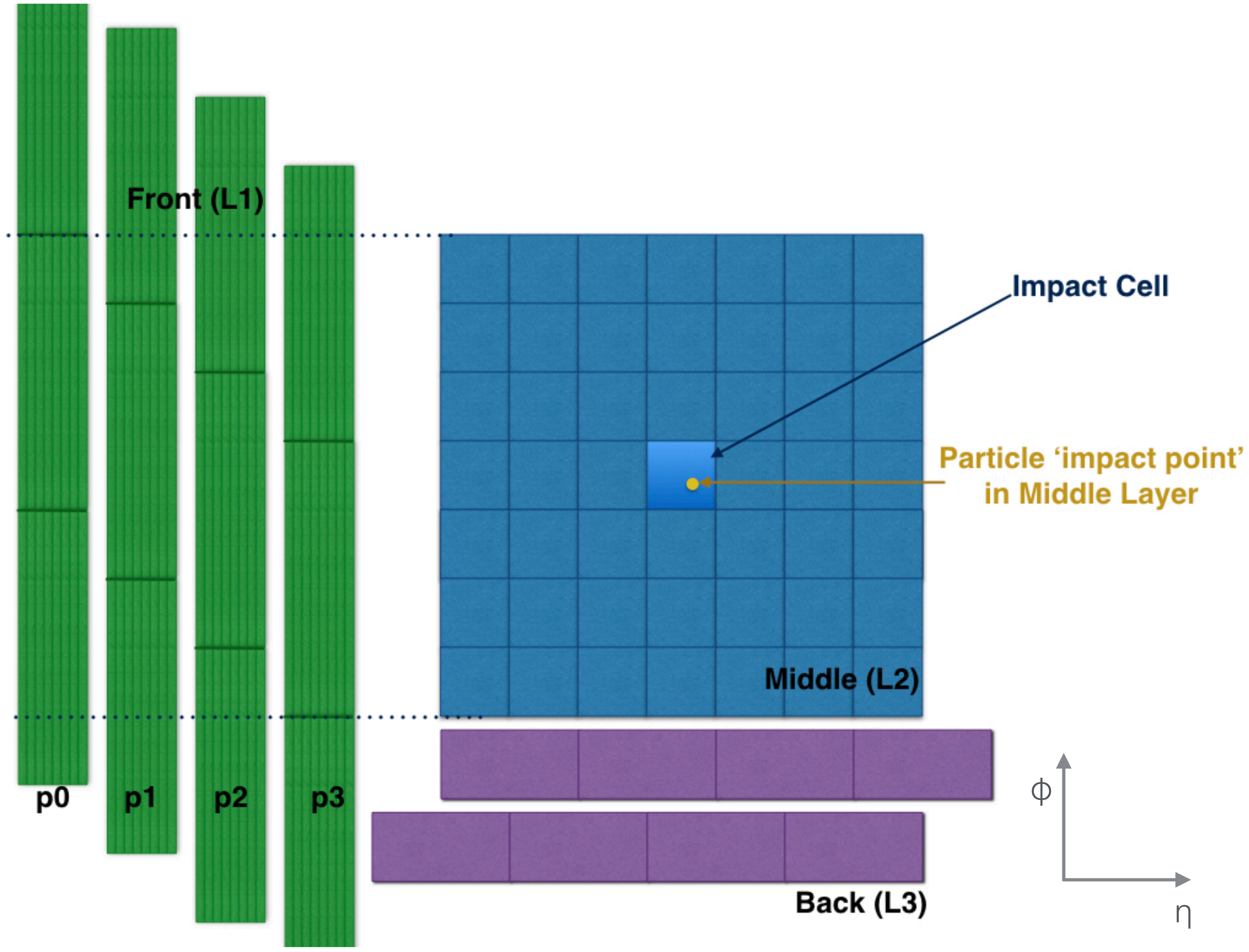


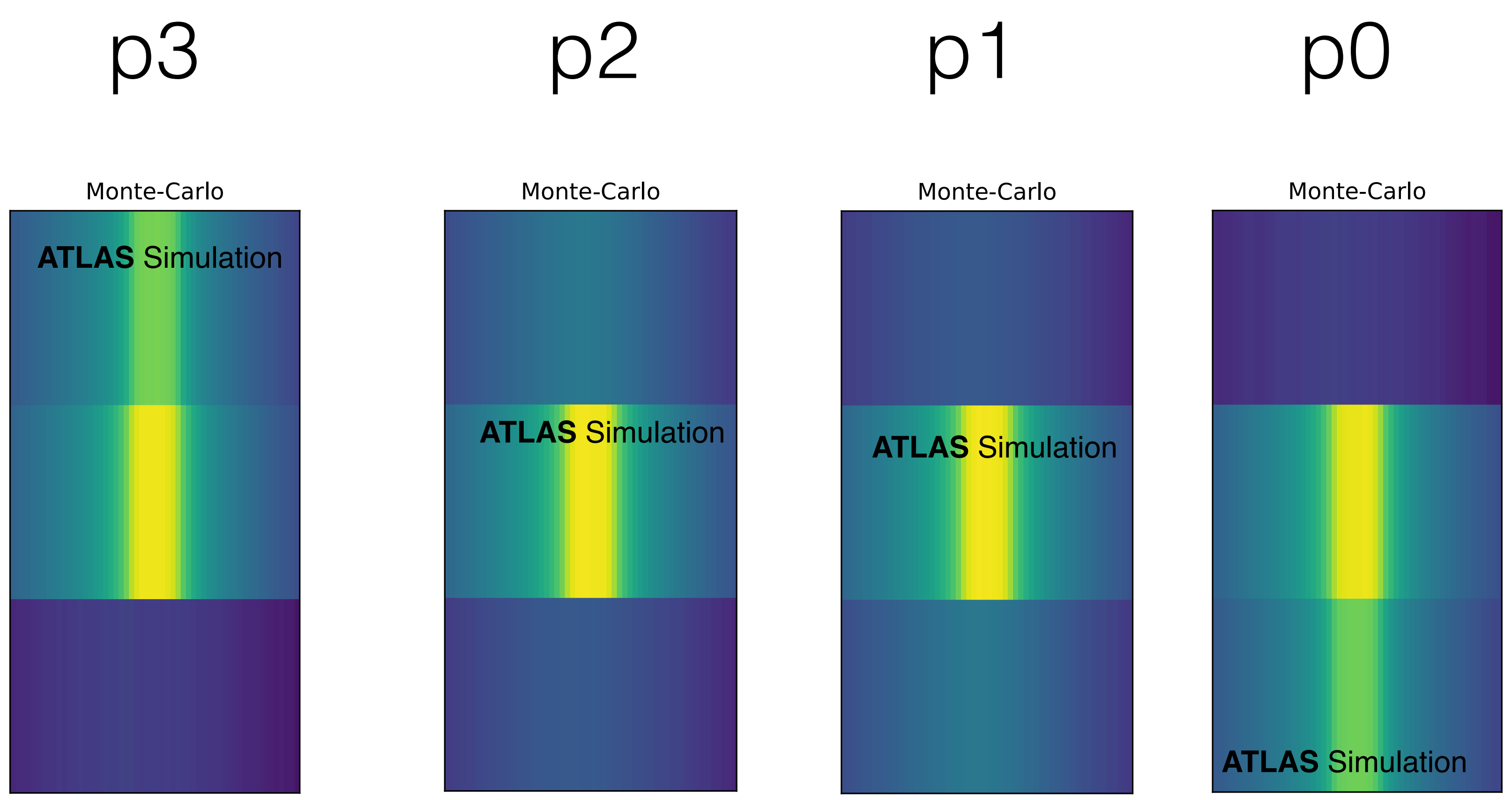


Hardest conditioning to get correct (HPO)

Two hot vector encoding

Calorimeter Alignment Conditioning

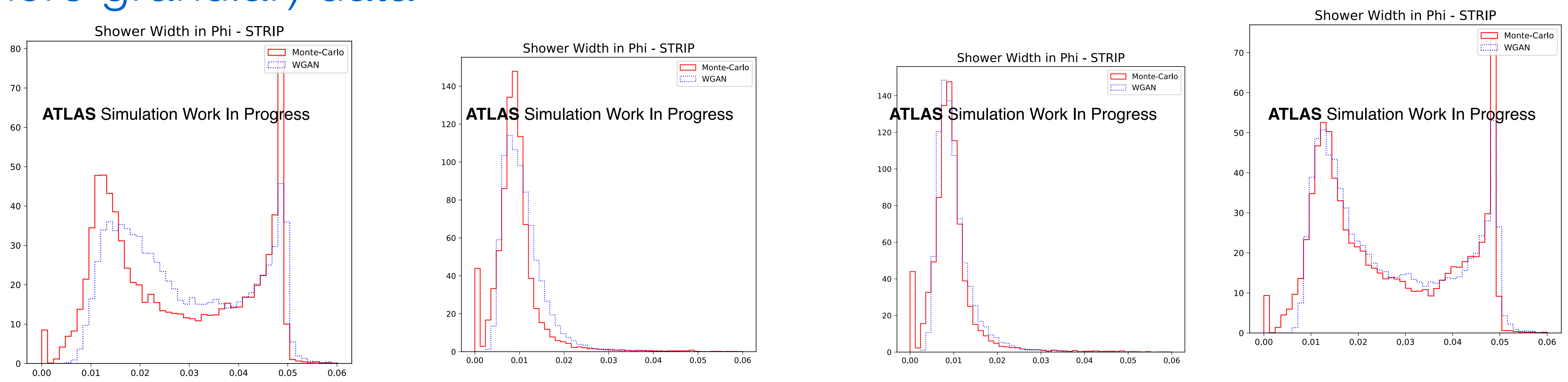




Hardest conditioning to get correct (HPO)

Motivating to move to Hits level (more granular) data

Two hot vector encoding



Smart Approximations

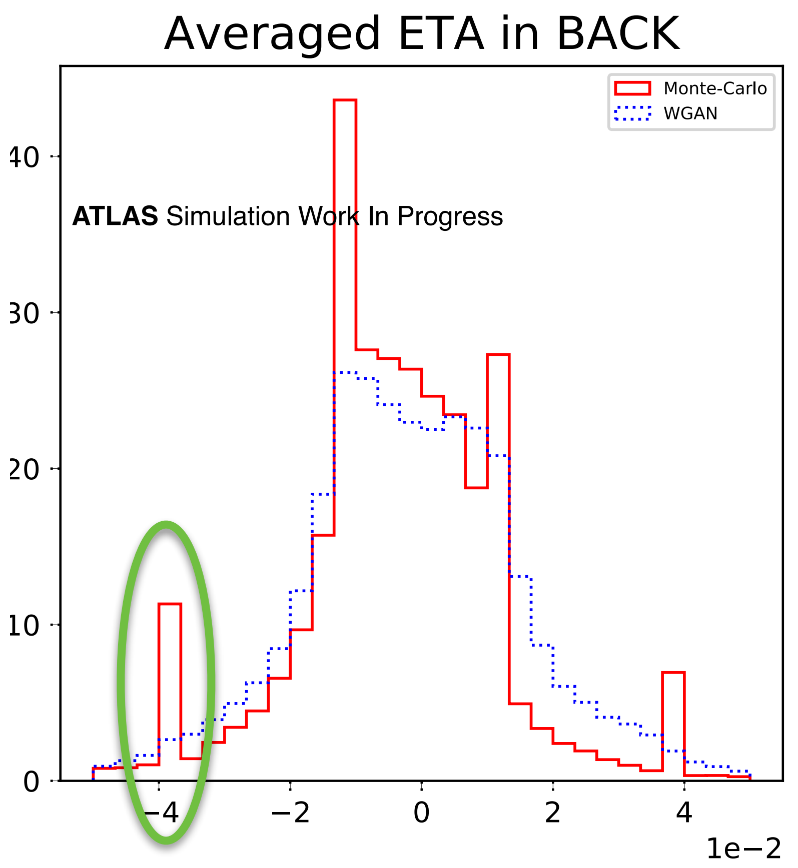
Aim is to **plug into Atlas C++ infrastructure** and hope that the GAN does well when **validation done with complex variables**

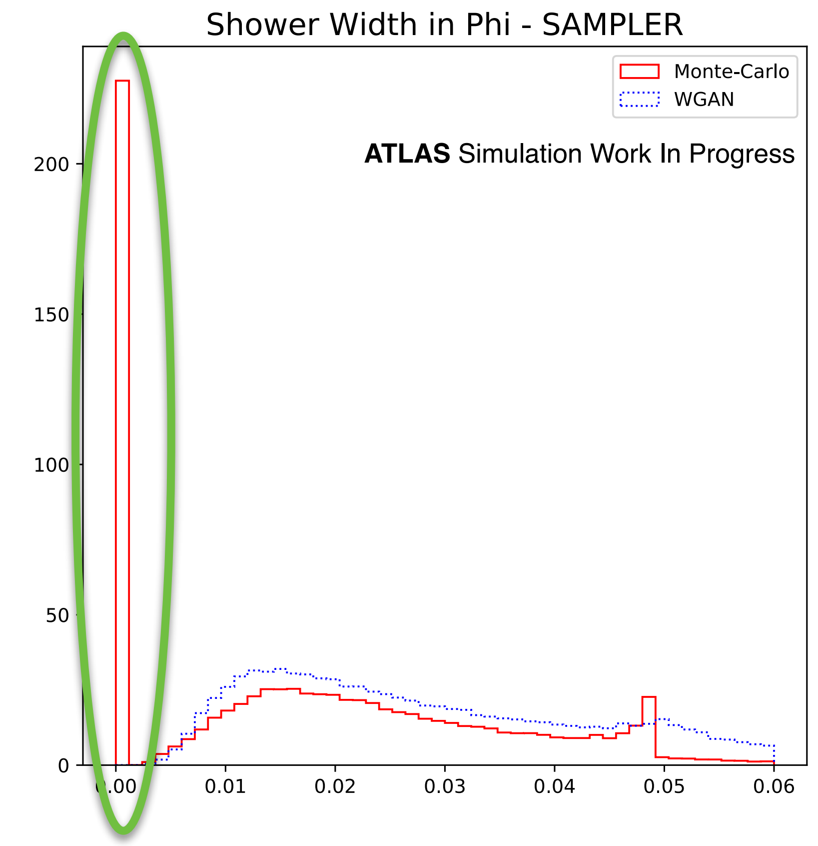
Currently calculating **simpler variable** that we **hope will adequately describe the performance** for **GAN optimisation**

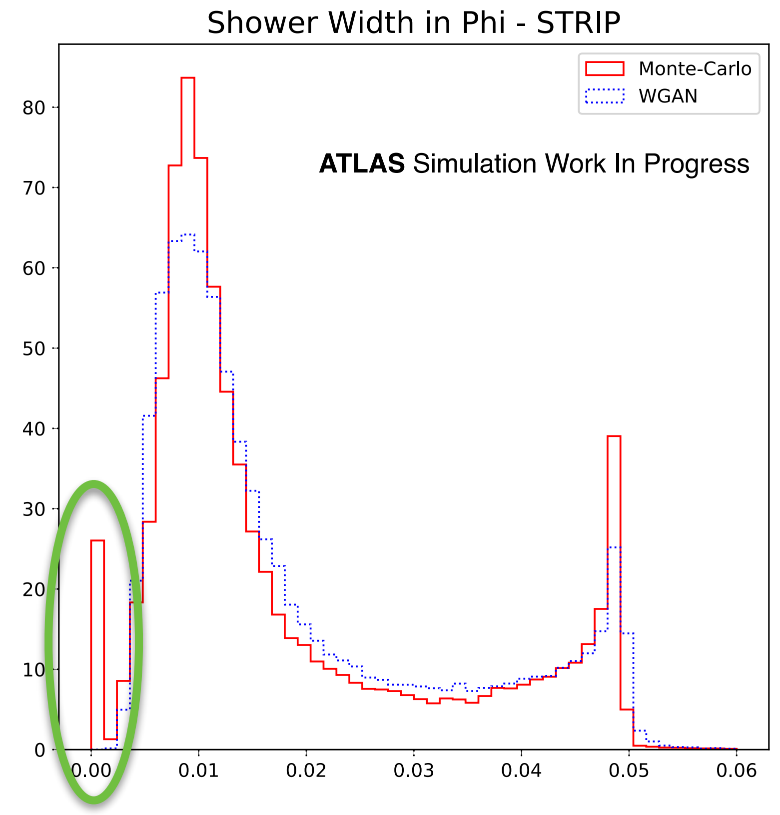
Train on dataset **without electronic noise**, cross talk (which the **ATLAS software adds later**), make other approximations of real validation phase

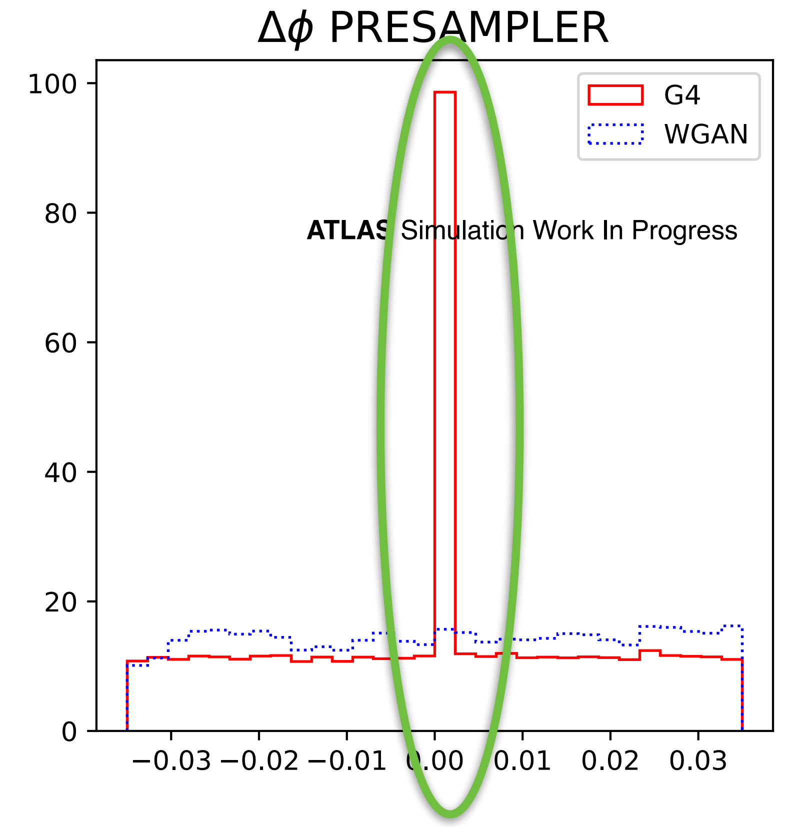
Do we care about modelling the sharp **single-bin peaks? Can reproduce with ReLu** activation, **but** we expect the noise to wash these **unphysical peaks** out

Don’t want to condition energy with one-hot encoding, **need to interpolate later**









Integration of DNN into ATLAS (C++) Software

Lightweight Trained Neural Network **Eigen based NN inference package for C++**

build passing | coverity passed | DOI 10.5281/zenodo.597221

✓

- Light Weight Trained Neural Network package built for fast inference in C++ framework:
  - Minimal dependencies
  - Avoid integrating heavy Tensorflow/PyTorch into software (CMS had multithreading nightmares)

Now we can make fair comparisons

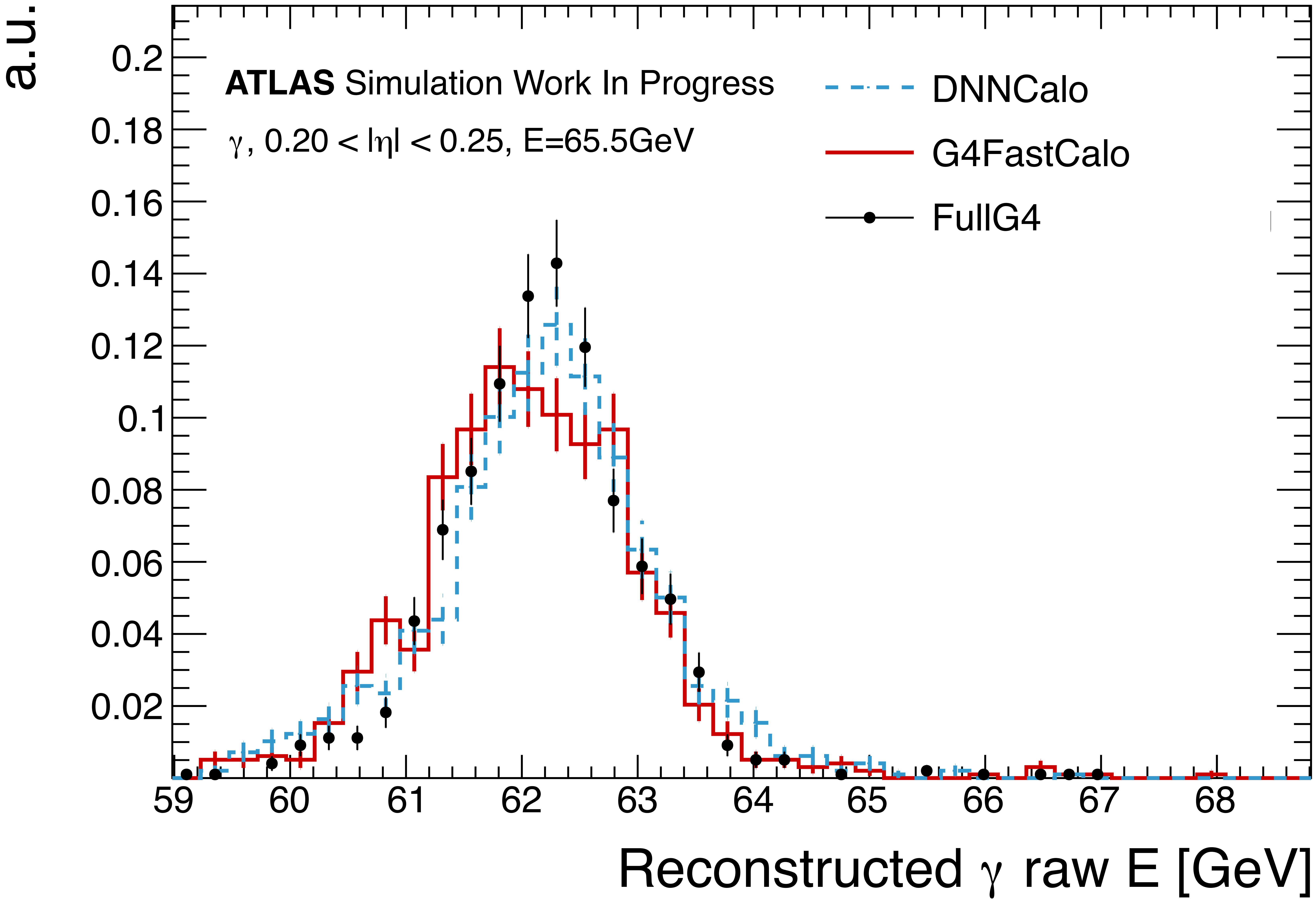
Speed & Resource utilisation (No GPUs, No Batch Parallelism):

- DNNCaloGAN ~ FastCaloSimV2 ~**70ms** (vs **~10s** for Geant4)
  - LWTNN takes **<1 ms per shower**, rest is overhead (being optimised)
- DNNCaloGAN **memory footprint small**
  - 5 MB for LWTNN JSON file vs order GB for FCS parameterisation file

Now the real test:

How are we doing in a high level validation inside Atlas software?

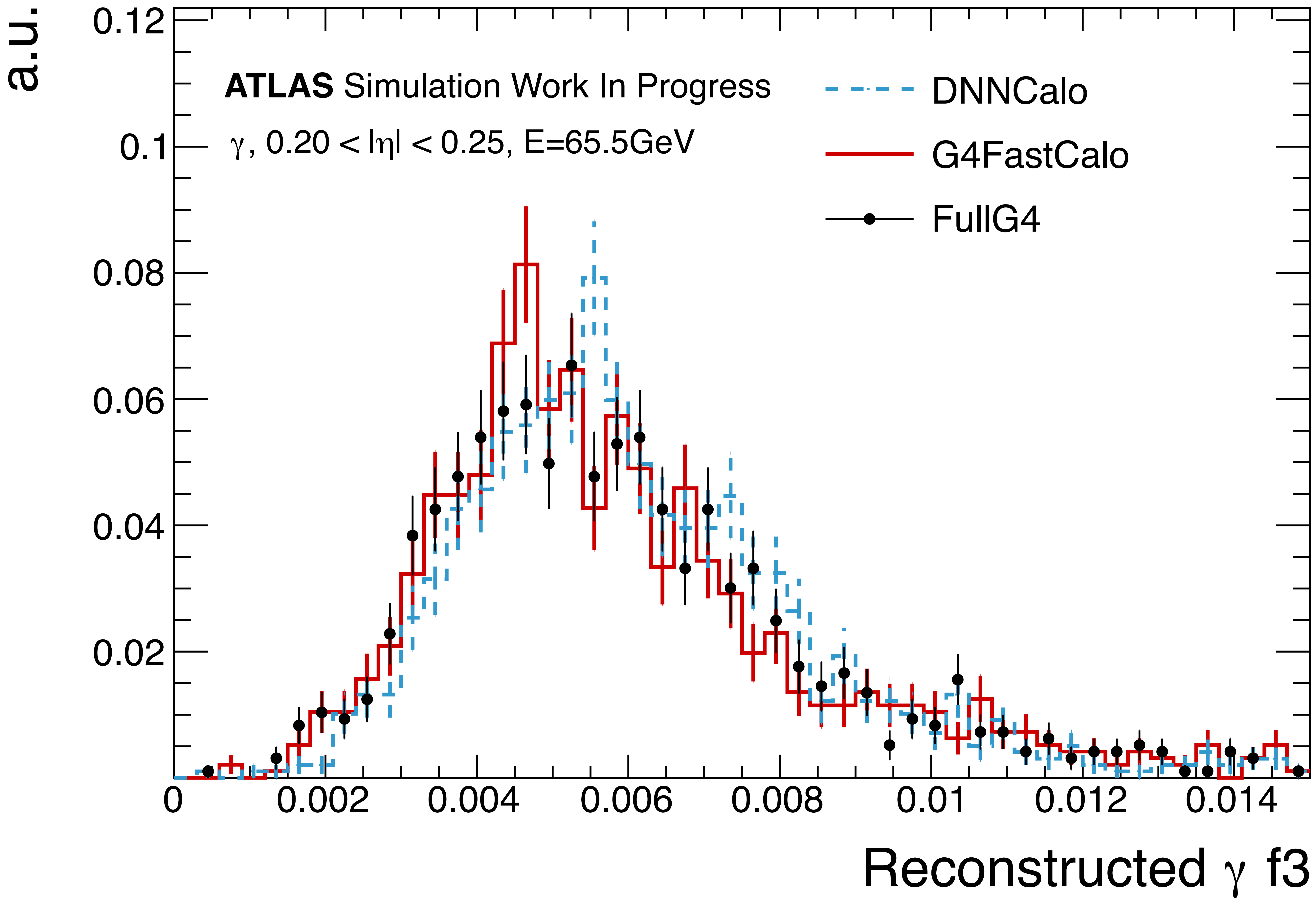
Total Uncalibrated Energy for 65 GeV Photons



Disclaimer:
FastCaloSim versions moving fast with improvements, the FCS plot (which is no longer up to date) not to be used for ranking methods but rather to get a rough idea

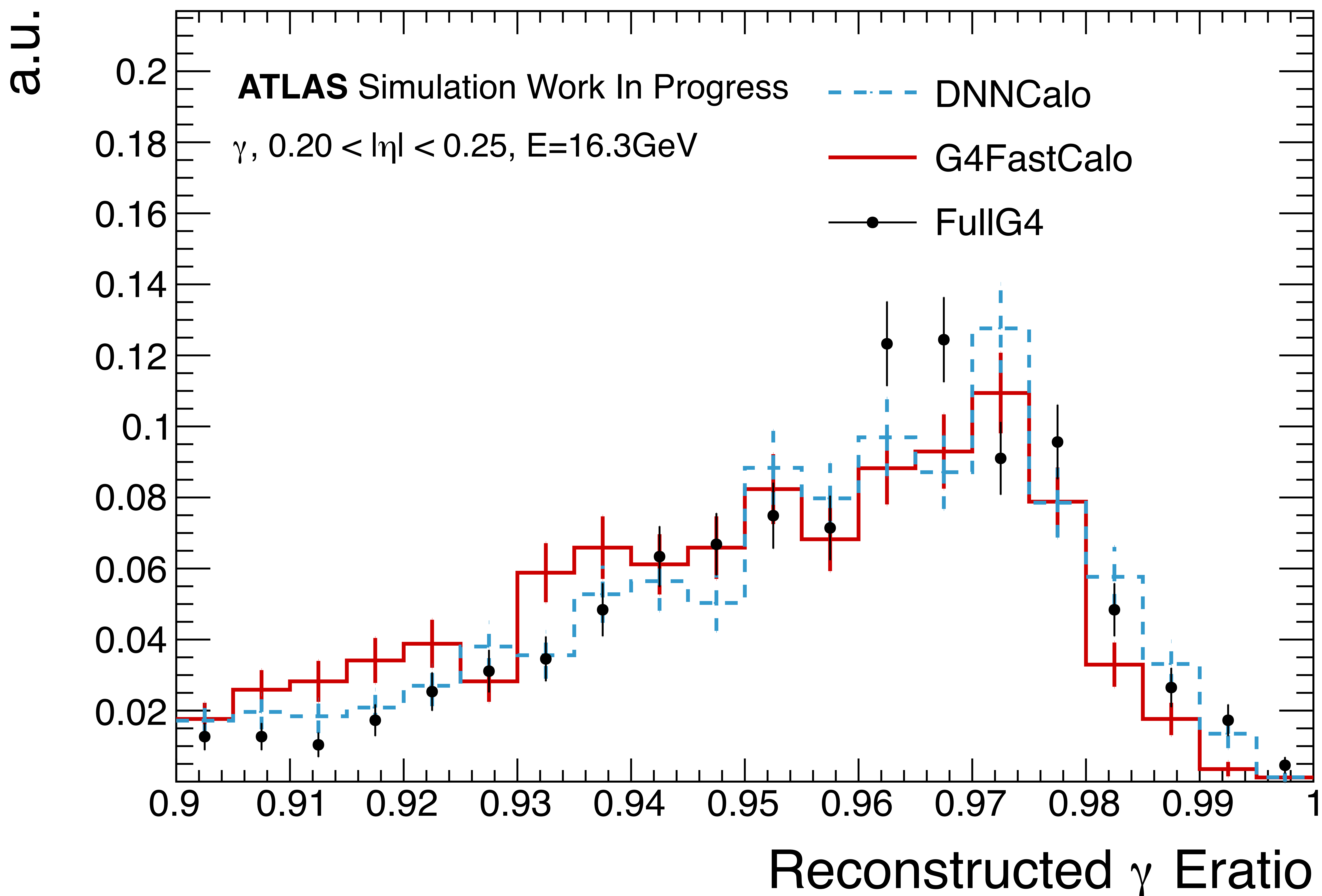
Second Critic doing it's job!

f3:Fraction of Energy in the Back



Eratio (16 GeV)

(First_Max_Strip -Second_Max_Strip)/(First_Max_Strip+Second_Max_Strip)

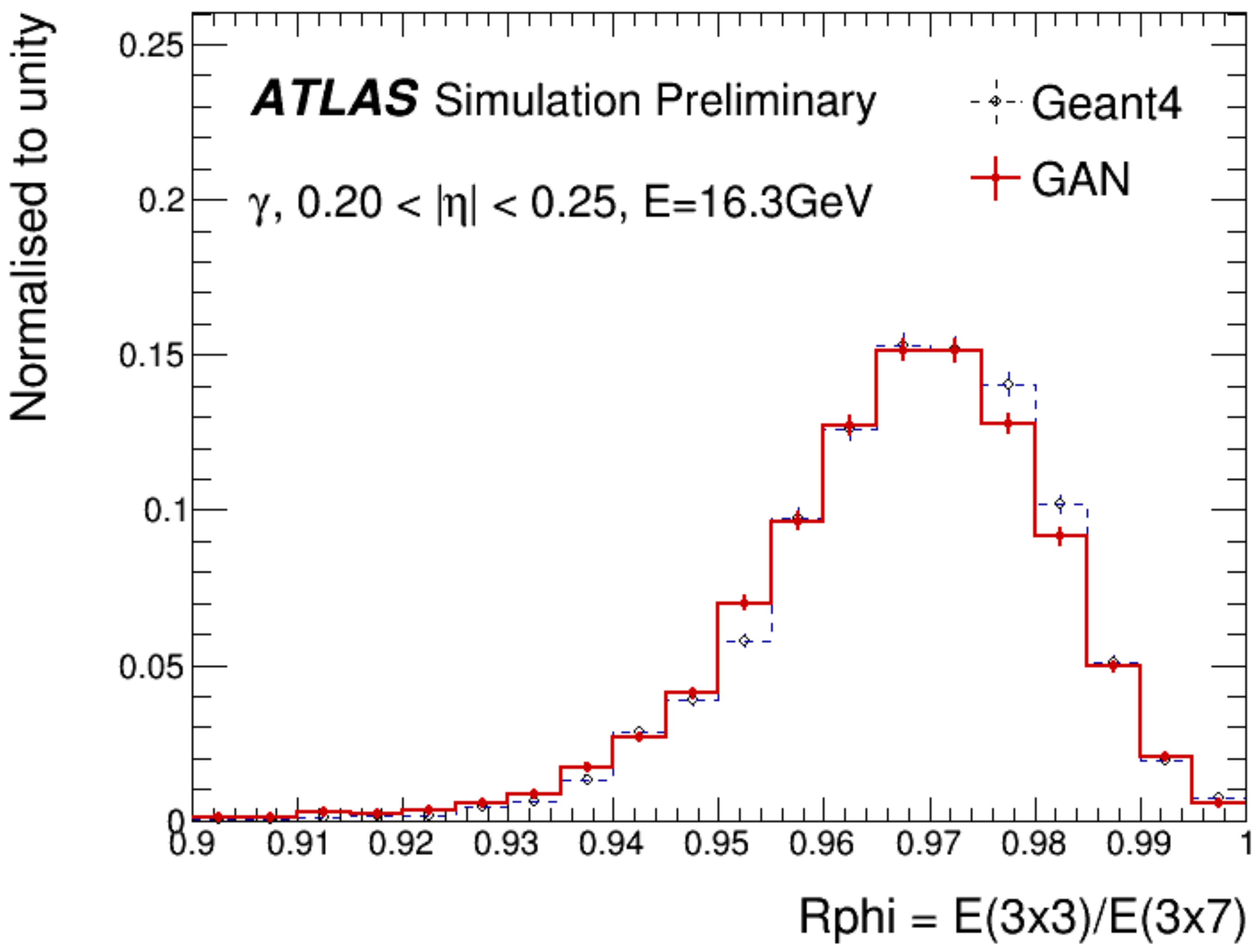


Performs worse at some <u>other</u> energies

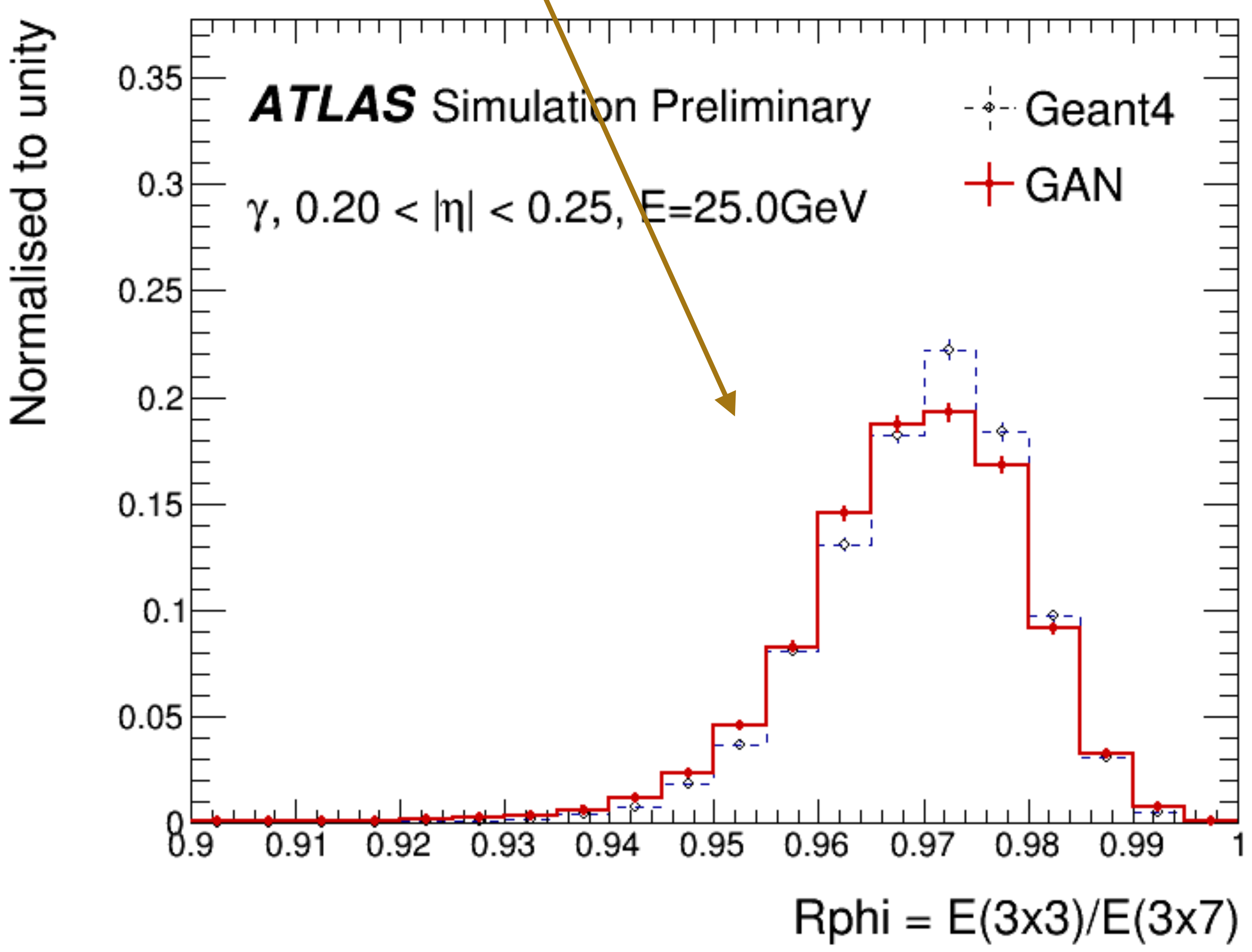
High Statistics Public Plots with Interpolation

GAN never trained at 25 GeV!

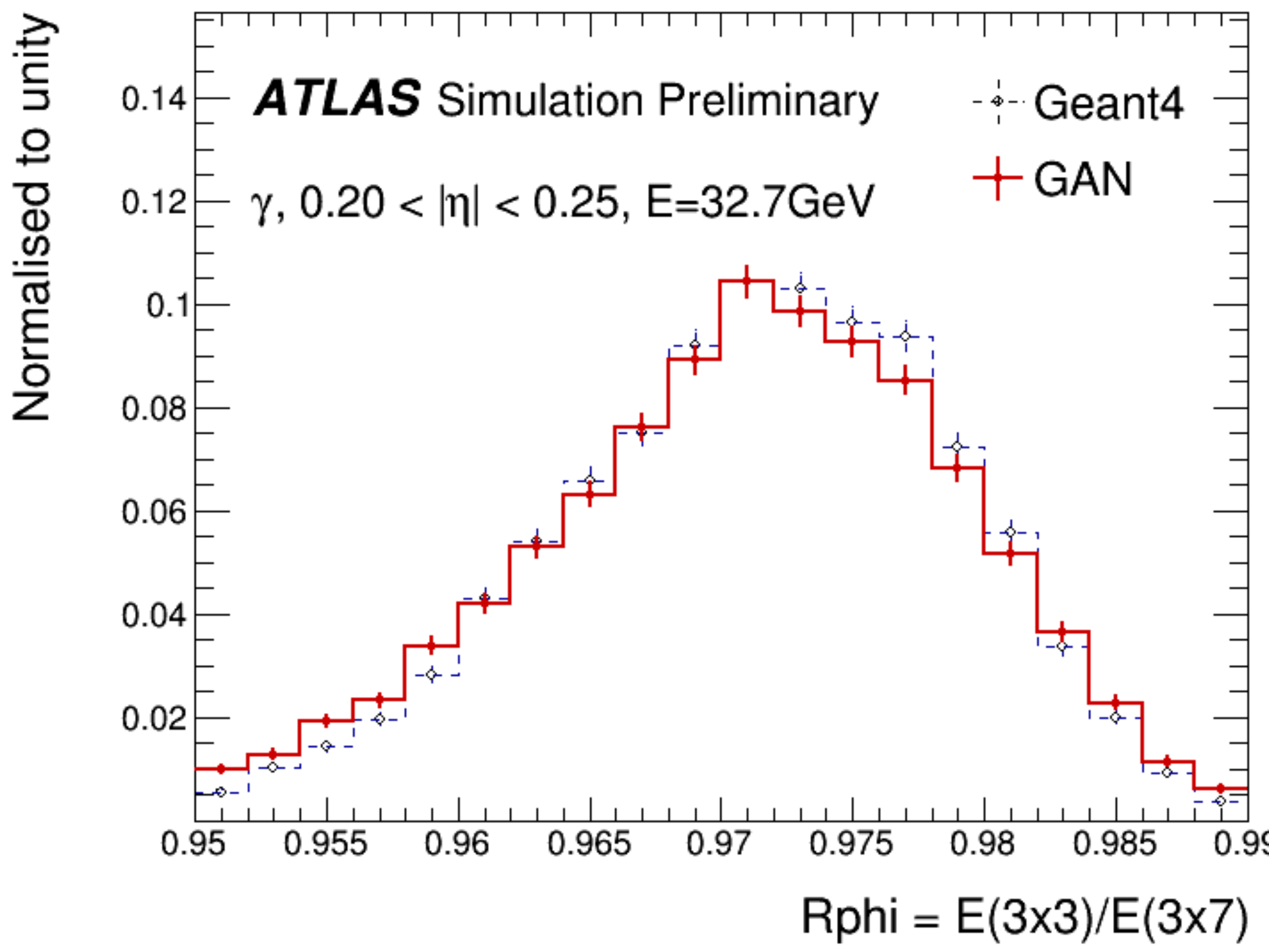
GAN



16 GeV



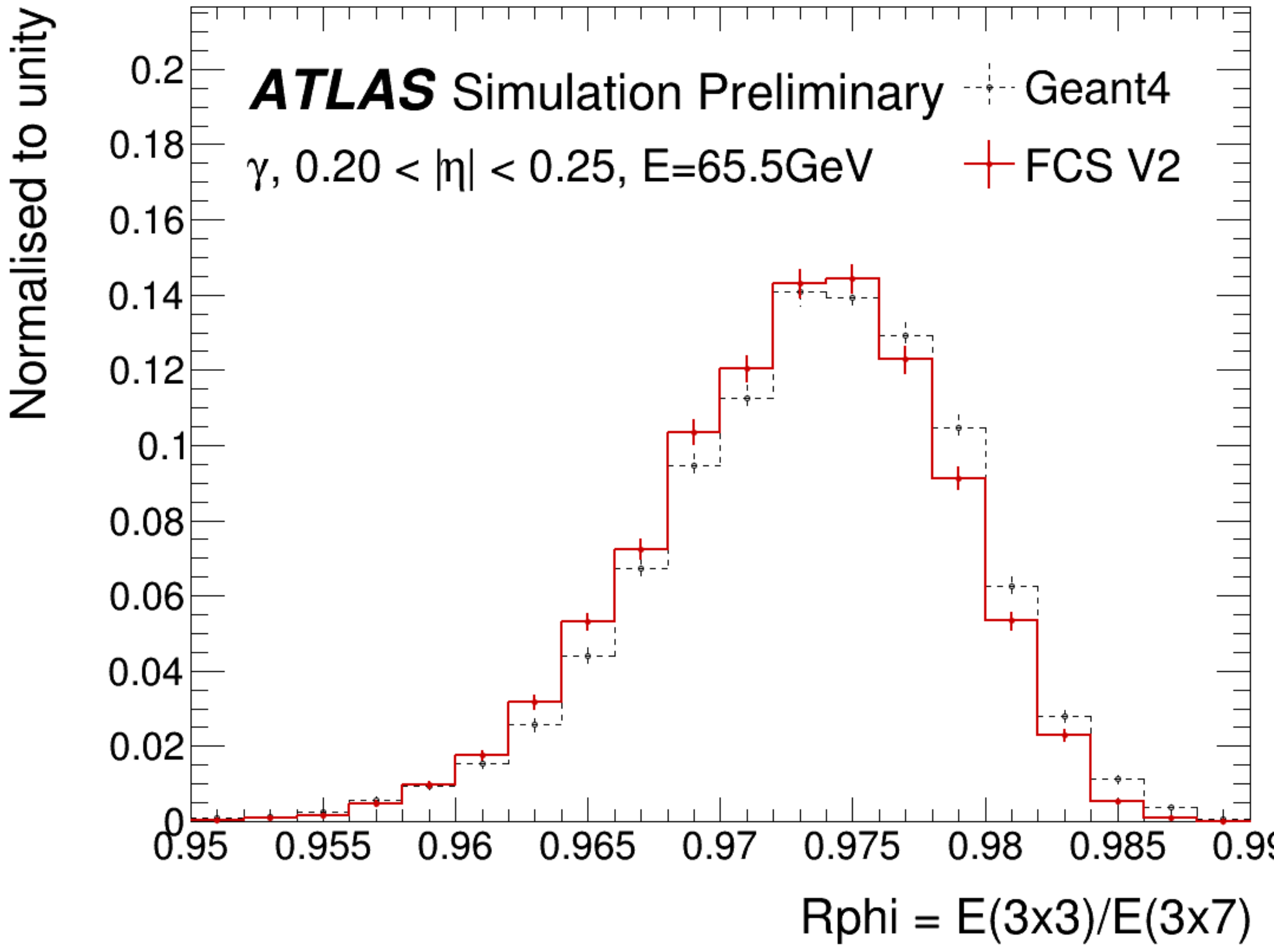
25 GeV



32 GeV

High Statistics Public Plots with Interpolation

FCS

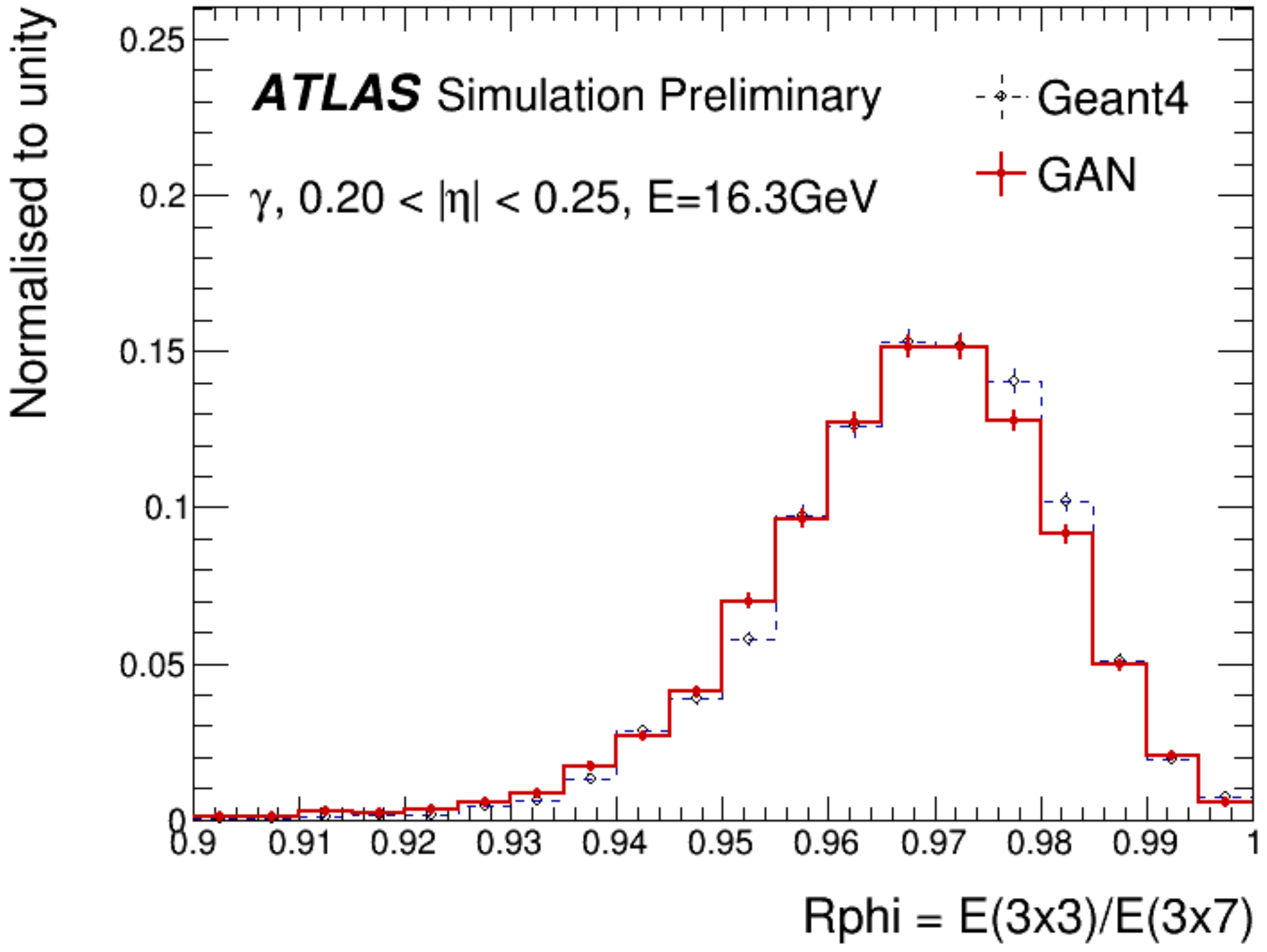


65 GeV

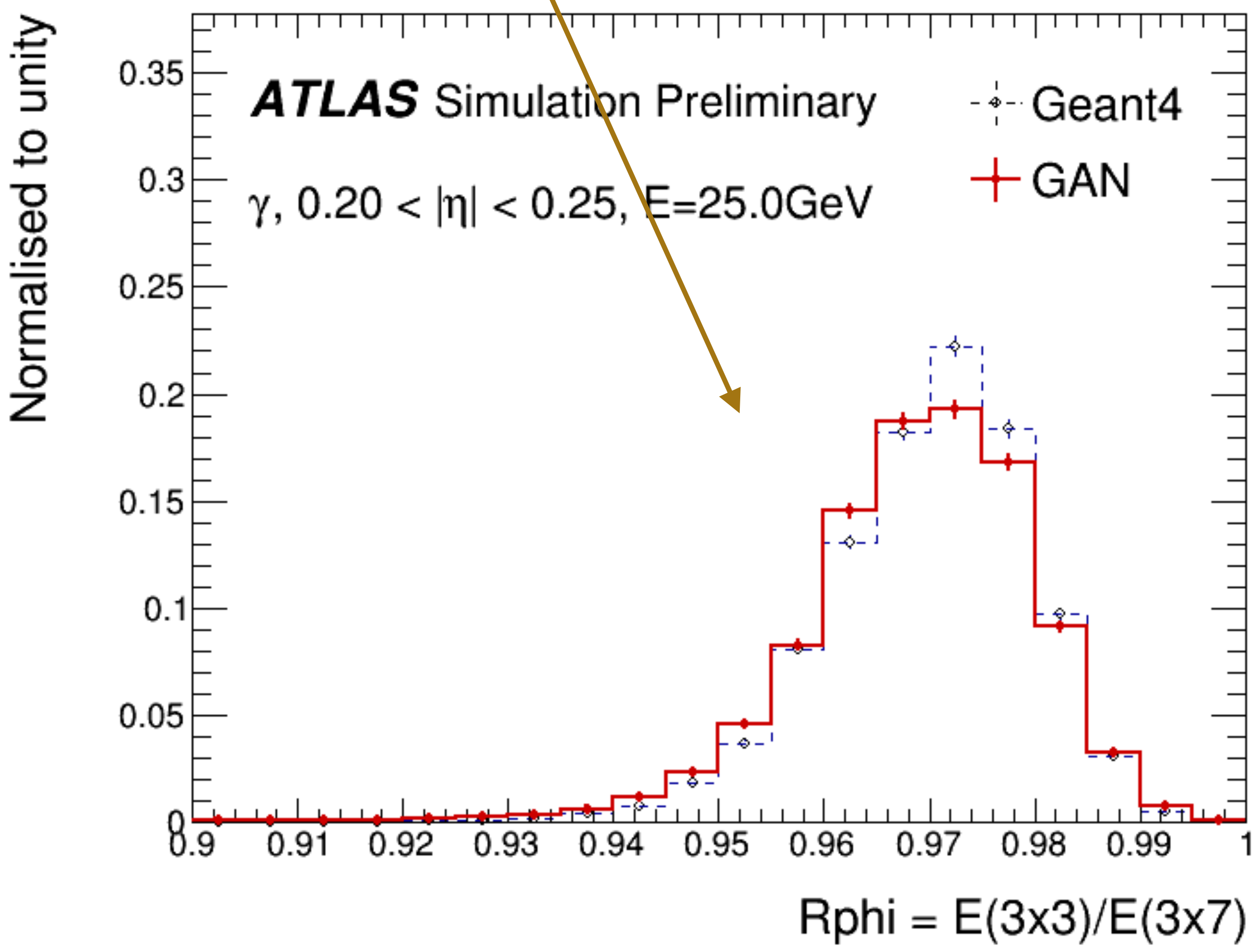
Fractional energy deposit in the φ direction for the second EM barrel layer for a 65 GeV photon reconstructed cluster in the range 0.20 < |η| < 0.25. The 3x3 and 3x7 refers to the rectangle of cells considered around the cluster centre. FCSV2 (red solid line) is compared to Geant4 (black dashed line). FCS V2 has an improved treatment of the parameterisation of the later shower with respect to [ATL-SOFT-PUB-2018-002](). The new version of FCS V2 improved the treatment of cross-talk between cells.

GAN never trained at 25 GeV!

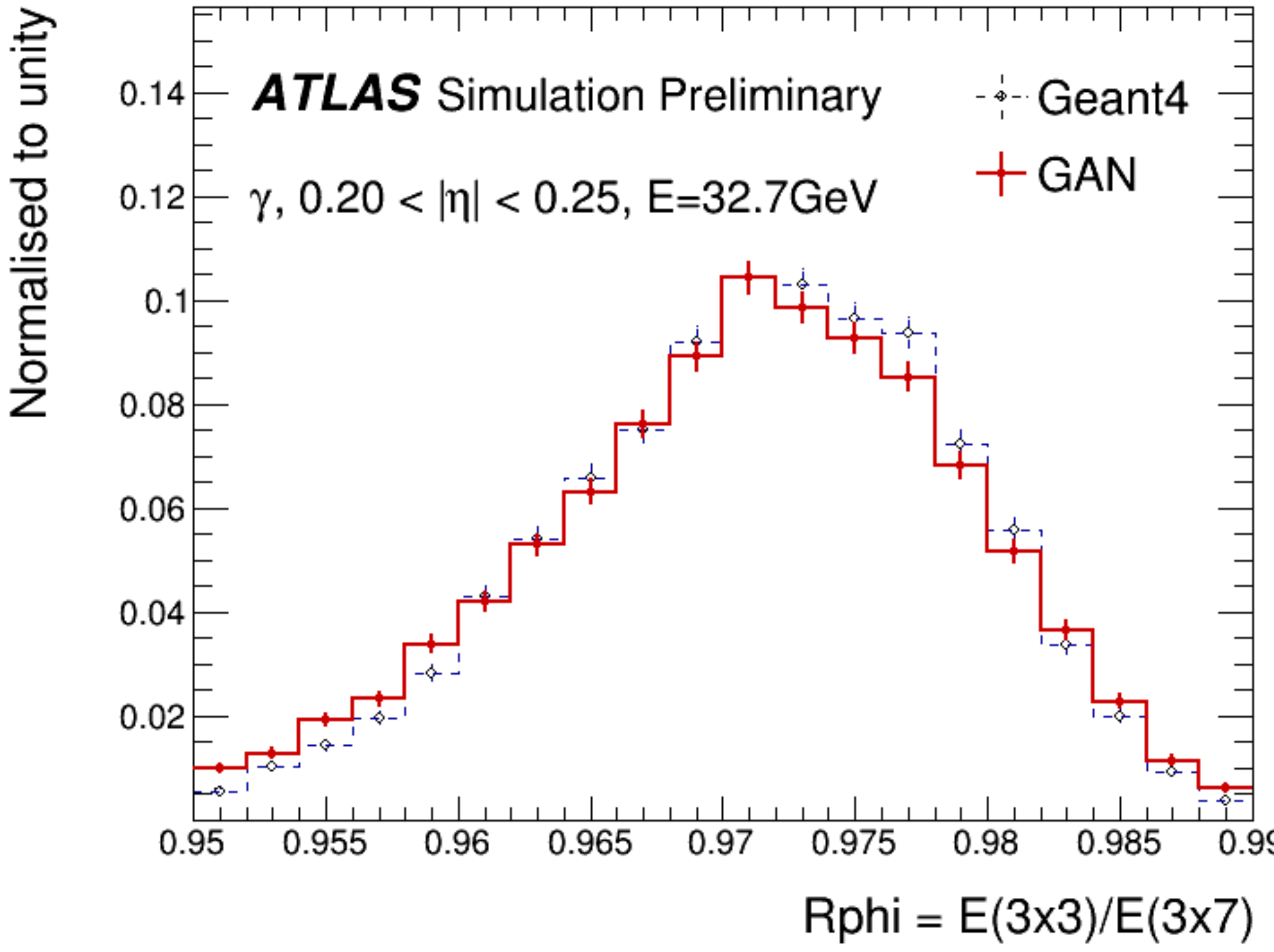
GAN



16 GeV



25 GeV



32 GeV

Conclusion

- Highly conditioned GAN working inside Atlas software, other GAN, VAE groups are following suit
- GAN can interpolate
- Successfully conditioned on energy (hardest), particle position (easy), calorimeter geometry (hard), other DNNCaloSim approaches also trying
    - Motivates the possibility to have one conditioned GAN for full calorimeter
- Wasserstein GANs (with Gradient Penalty) stable to train but limited, can't make sharp decisions
    - Additional Critic (with low Gradient Penalty) can be used for important physics variables that need attention
- Infusing physics knowledge was essential to push the final frontiers
- Project could be taken further to include the two ends of the calorimeter, Hadronic Cal, other particles, Z vertex spread in the future

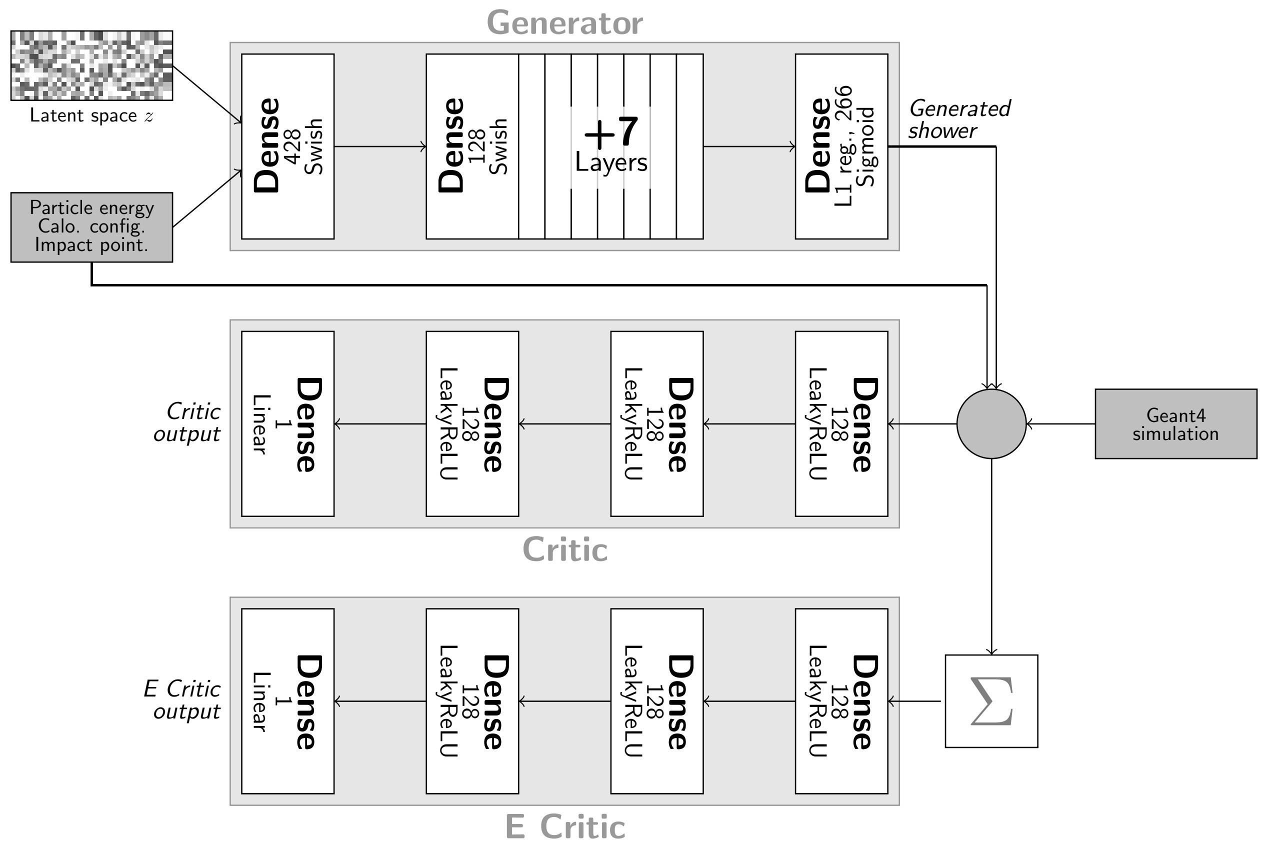
Backup

Scale Up with GPUs, Distributed Deep Learning?

Cannot speed-up even with massive GPU farms:
- no gain from model parallelism or data parallelism
- time per epoch very small, **number of epochs very large**
- training **dataset changes after every 5 batches**
- Best we can do is **parallel Hyper-Parameter Optimisation**

Alternative: Gradient Reversal Layer + **simultaneously train 3 networks** with different learning rates rather than training ratio

- Training time: **2-7 Days** on 1 GPU
- **Epochs: 7k-50k**
- Training Size: 44000 events (50% of Dataset), ~300 features
- CPU = 2 x GPU training time at 52% GPU utilisation



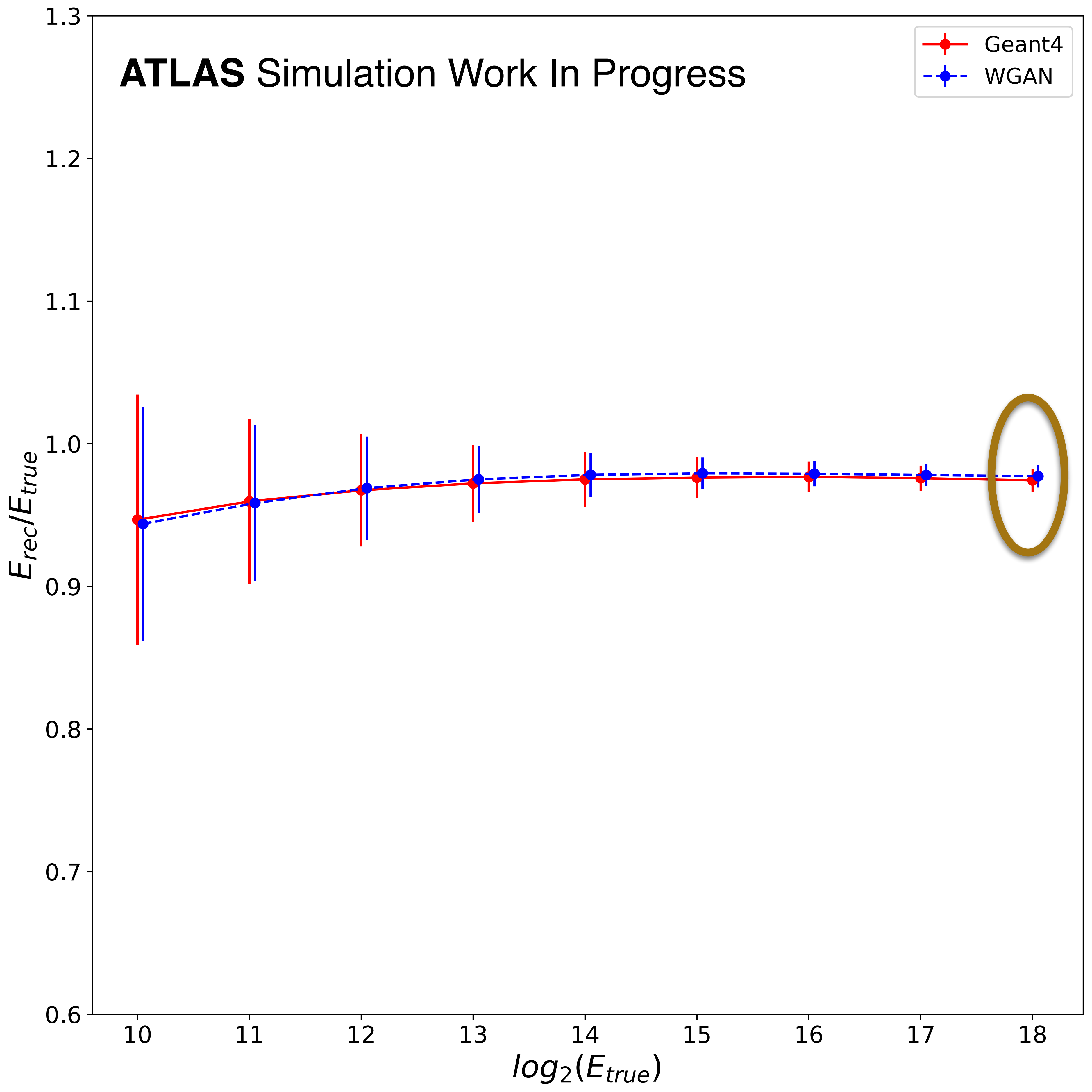
Reminder: A conditioned WGAN-GP takes many many epochs to train, much beyond when the loss looks converged

https://arxiv.org/pdf/1409.7495.pdf

Take Aways

- Do not start with an oversimplified problem, GANs don't scale that way
- Is a feature important to model well? : **Think of final use case**
- Wasserstein GANs (with Gradient Penalty) stable to train but limited
    - Takes much longer to train than a vanilla GAN, specNorm GAN etc
    - People will not believe you but the conditional WGAN-GP continues to train long after the loss has "converged"
    - No mode collapse but performance might be **limited by Gradient Penalty**, find creative ways out (**oppose of mode collapse**)
- **Infusing physics knowledge:**
    - Even if the GAN could learn on it's own, if you have the information, give it to the GAN either as input or as an auxiliary task
- **Distributed Deep Learning not a solution** for long training time of WGANs if the problem is number of updates to the model
- Lots of data not always the answer, just a **small representative sample can go a long way**. With conditioning, **we don't need balanced number of samples** for each category (see soft extrapolation), GAN will still interpolate
- Additional Critic can also be used for **Transfer Learning on Data** for specific features when Geant4 isn't good enough
- Large number of discrete conditioning was harder for us than smooth continuous ones
    - Getting all plots right simultaneously requires luck (multiple runs)

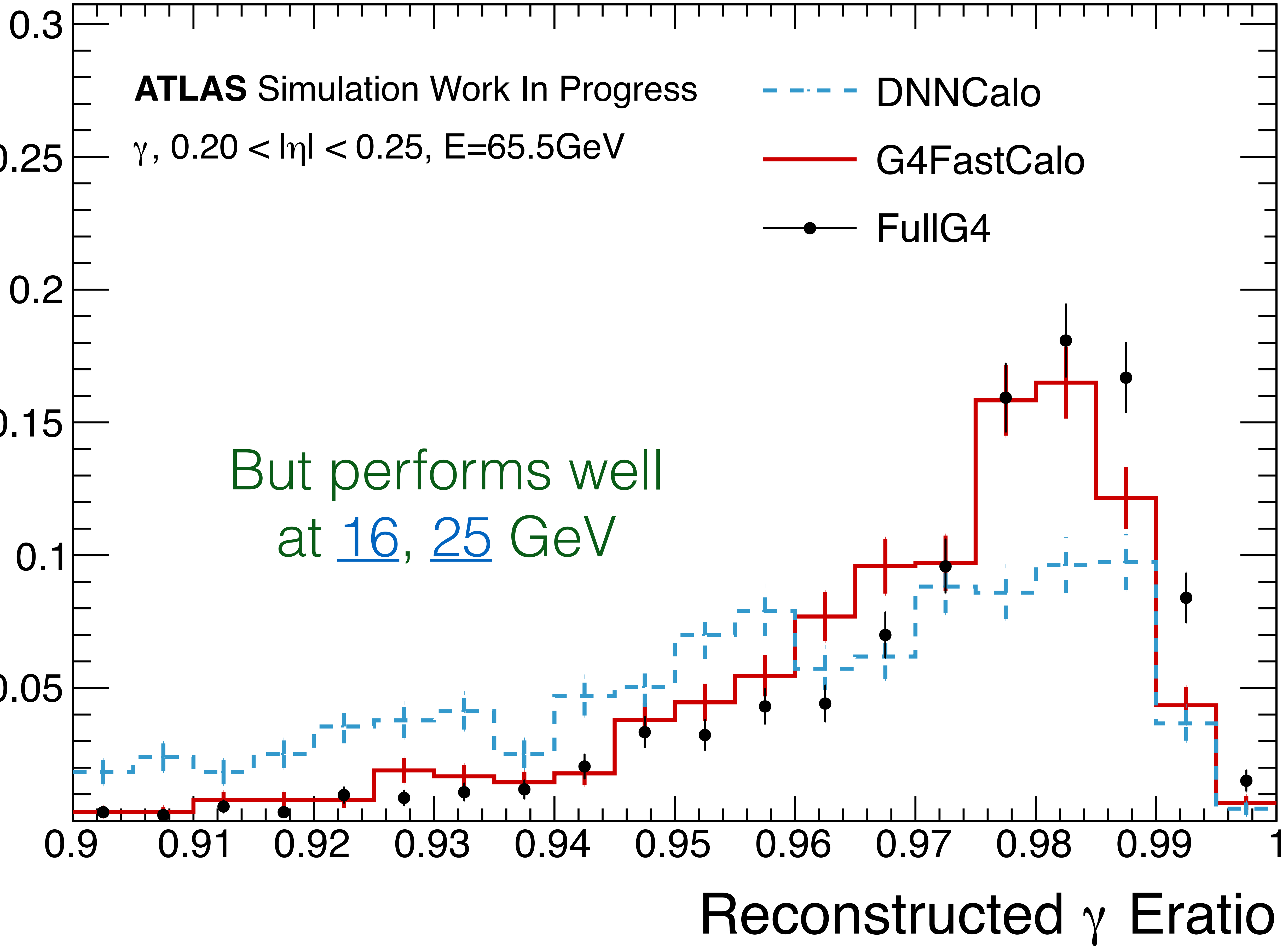
Soft Extrapolation



Train on only 10 events at 262 GeV, 5k events at other Energy points

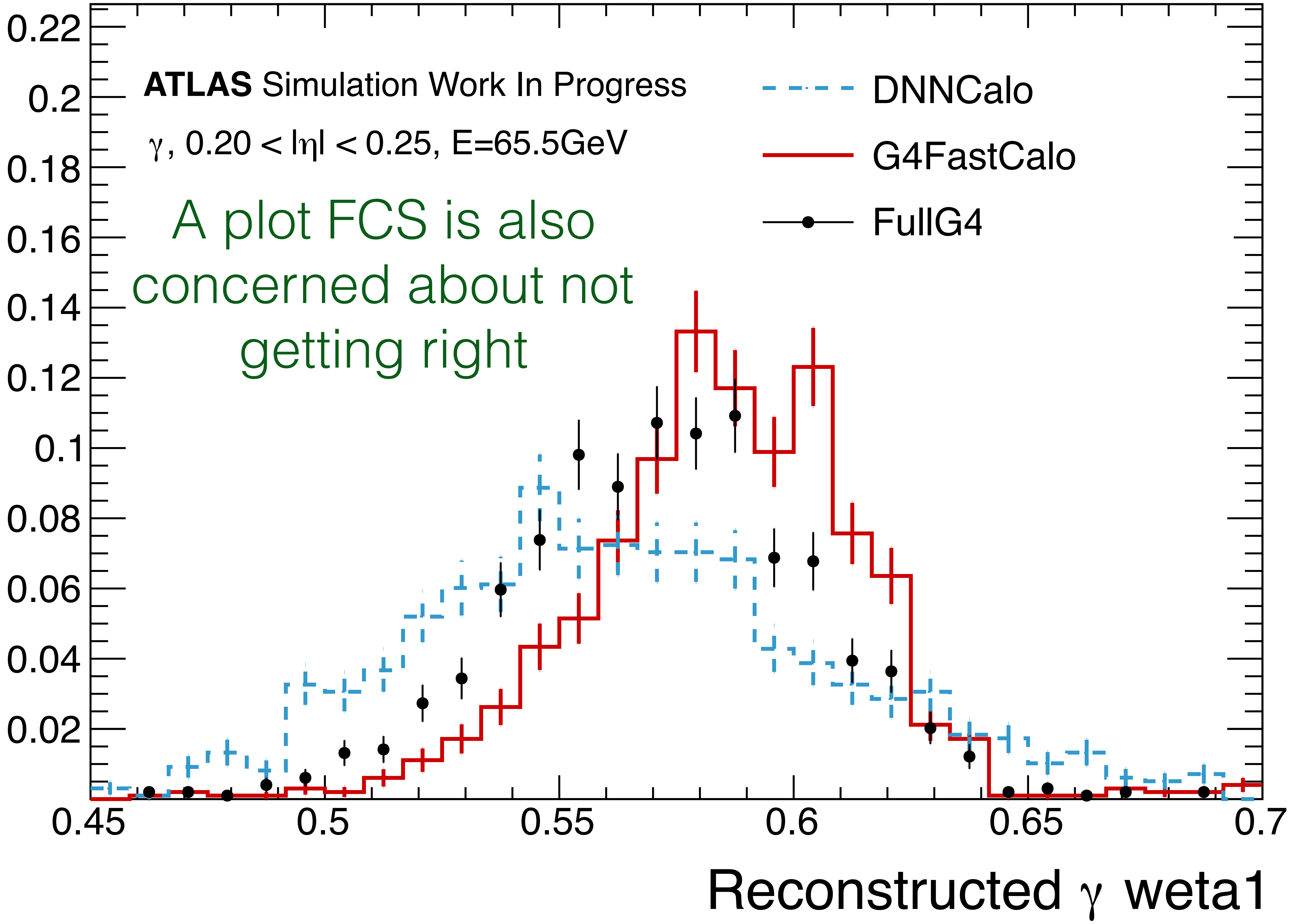
Strips (65 GeV)

Eratio = (First_Max_Strip -Second_Max_Strip)/(First_Max_Strip+Second_Max_Strip)



But performs well at 16, 25 GeV

Width in Eta (in Strip Cell units)



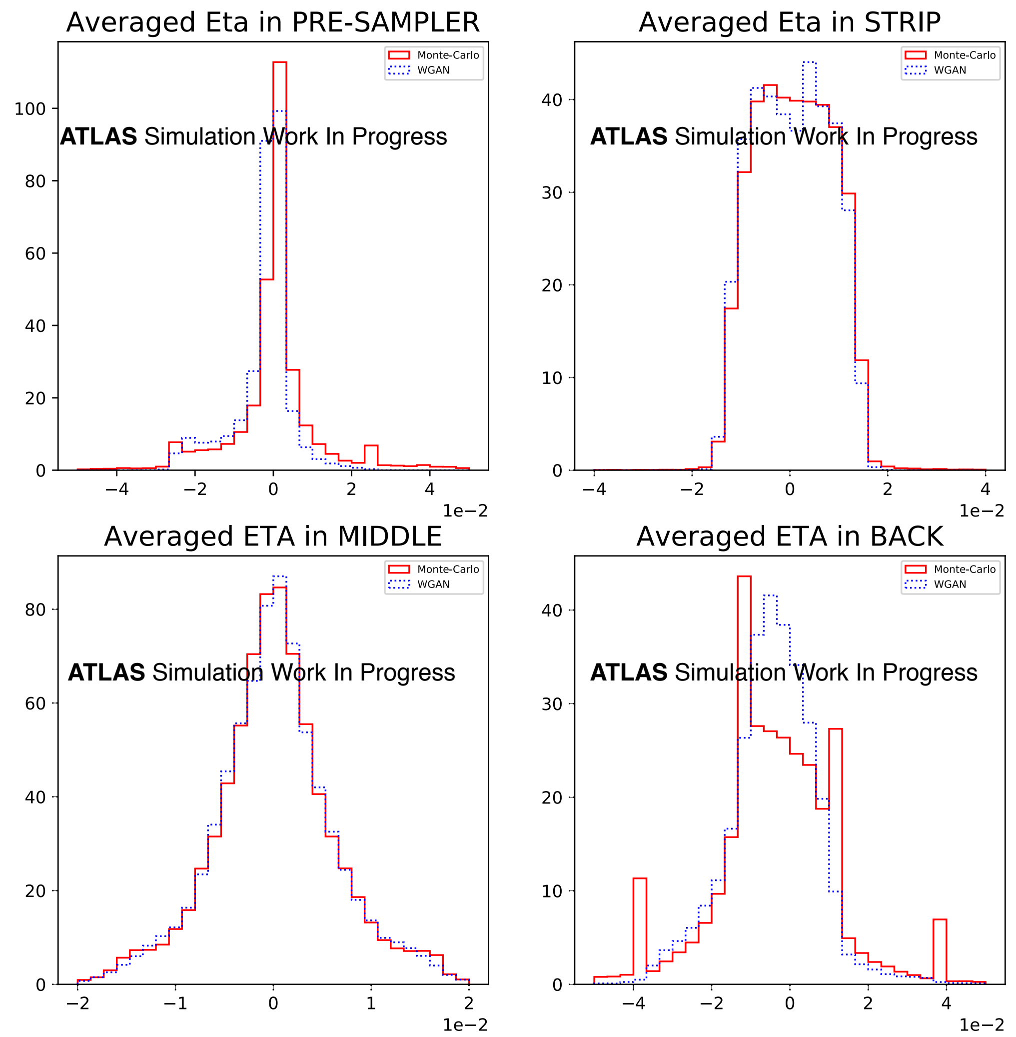
A plot FCS is also concerned about not getting right

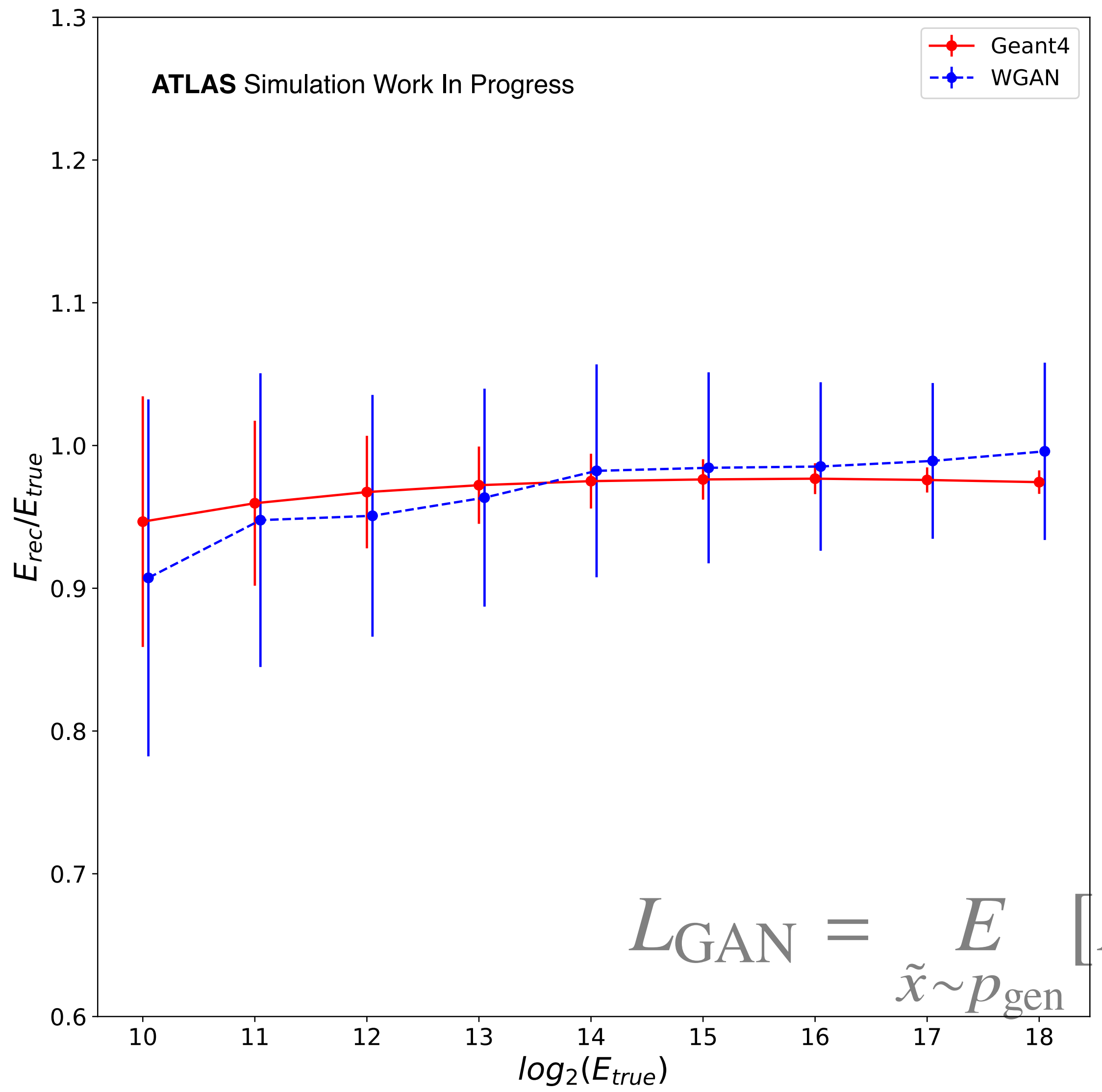
⇒ Need detailed simulation of Strip

What is a good range of hyper-parameters to try anyway?

Results were stable for usual GP values, ∈ (1,500)

Gradient Penalty = 10



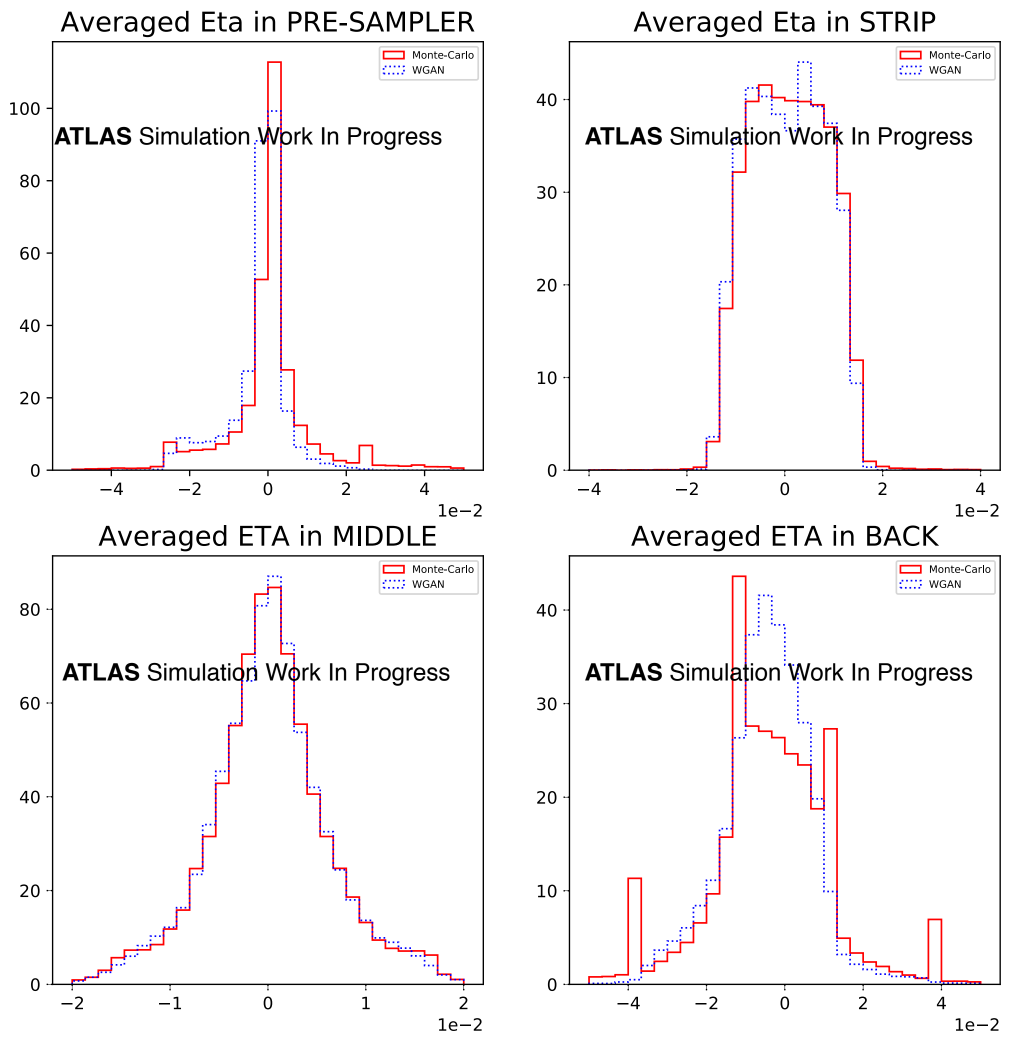


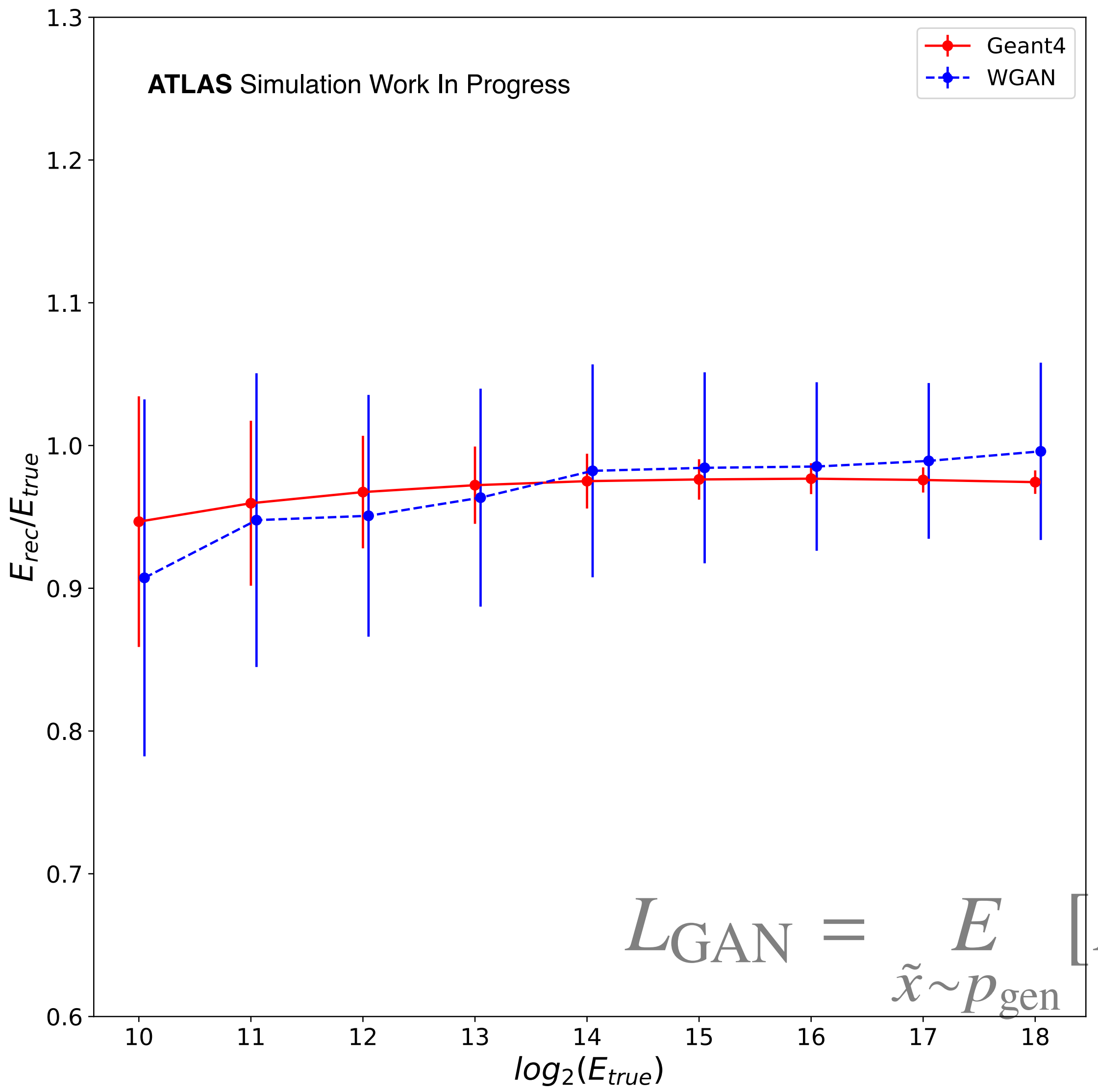
$$L_{\mathrm{GAN}} = \underset{\tilde{x}\sim p_{\mathrm{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\mathrm{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

What is a good range of hyper-parameters to try anyway?

Results were stable for usual GP values, ∊ (1,500)

Gradient Penalty = 10

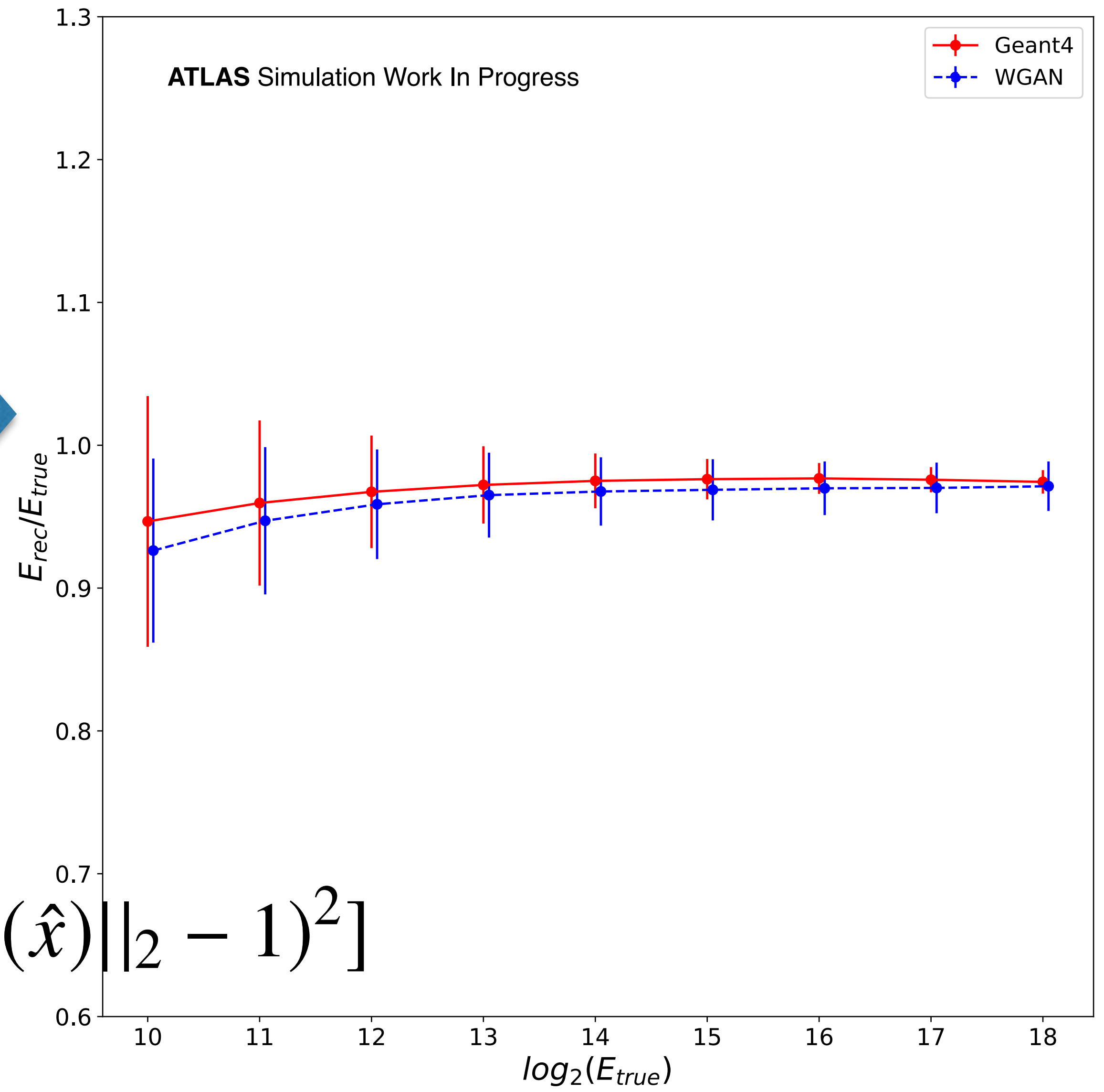




Energy gets better

Gradient Penalty = 1e-13

Never seen such a number in literature

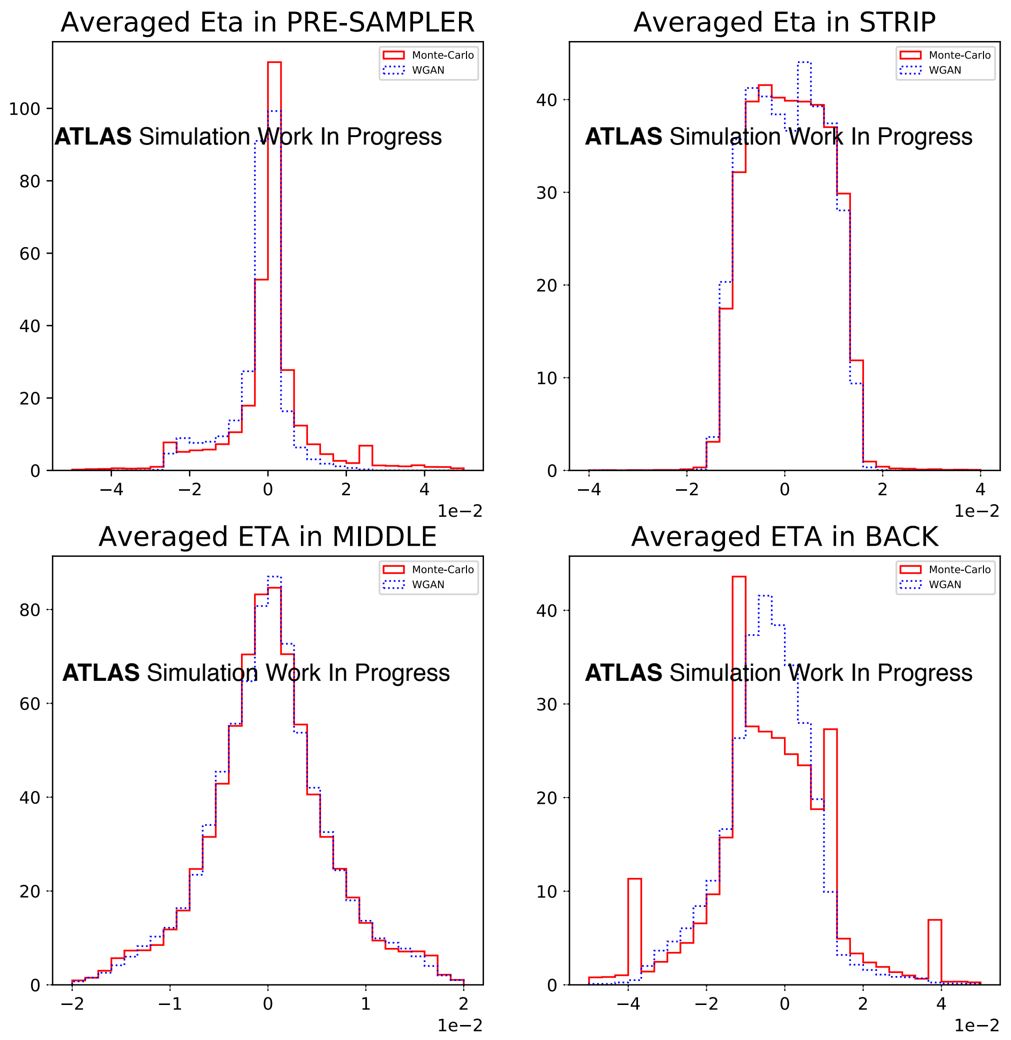


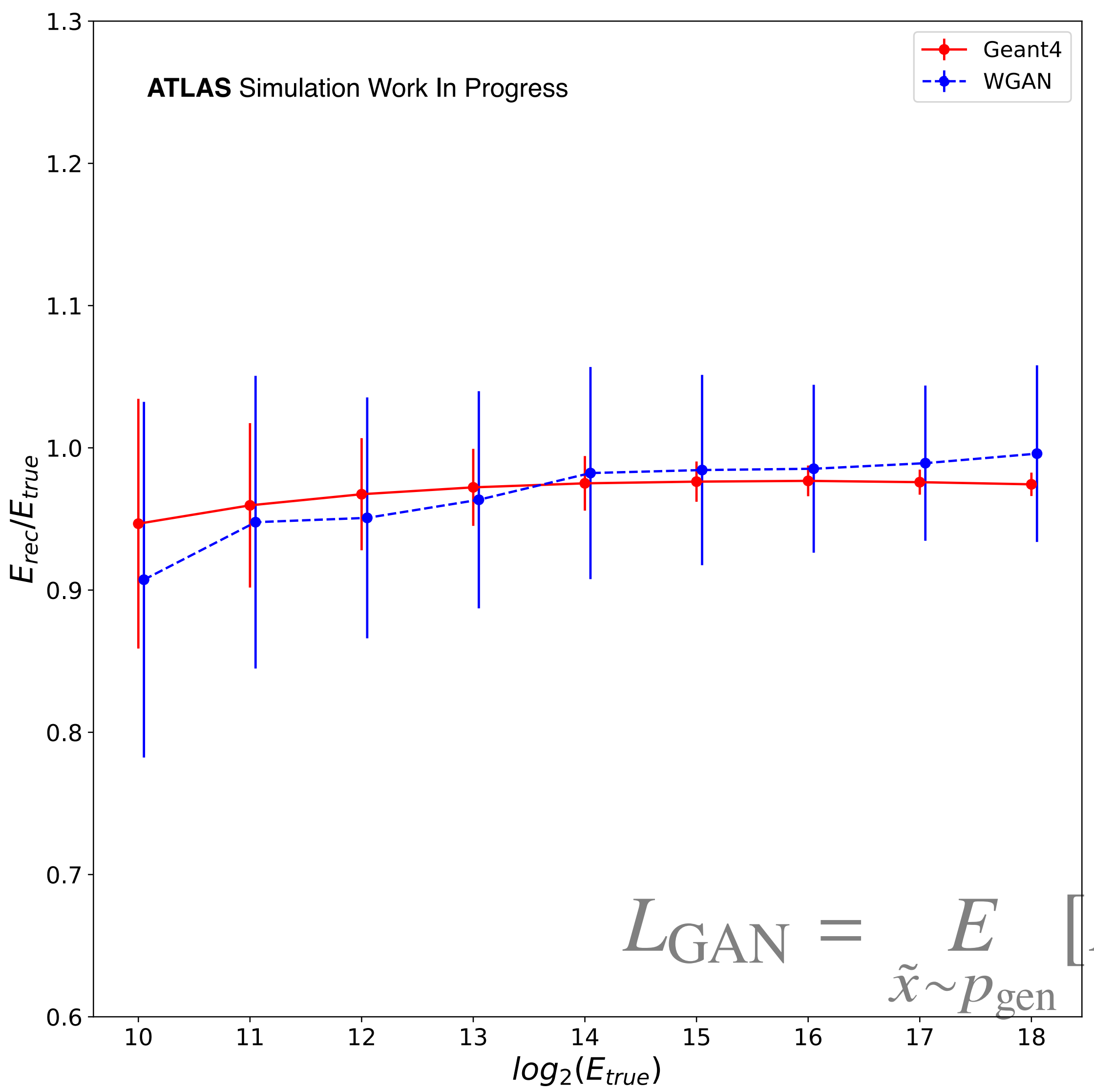
$$L_{\text{GAN}} = \underset{\tilde{x}\sim p_{\text{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\text{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

What is a good range of hyper-parameters to try anyway?

Results were stable for usual GP values, ∈ (1,500)

Gradient Penalty = 10

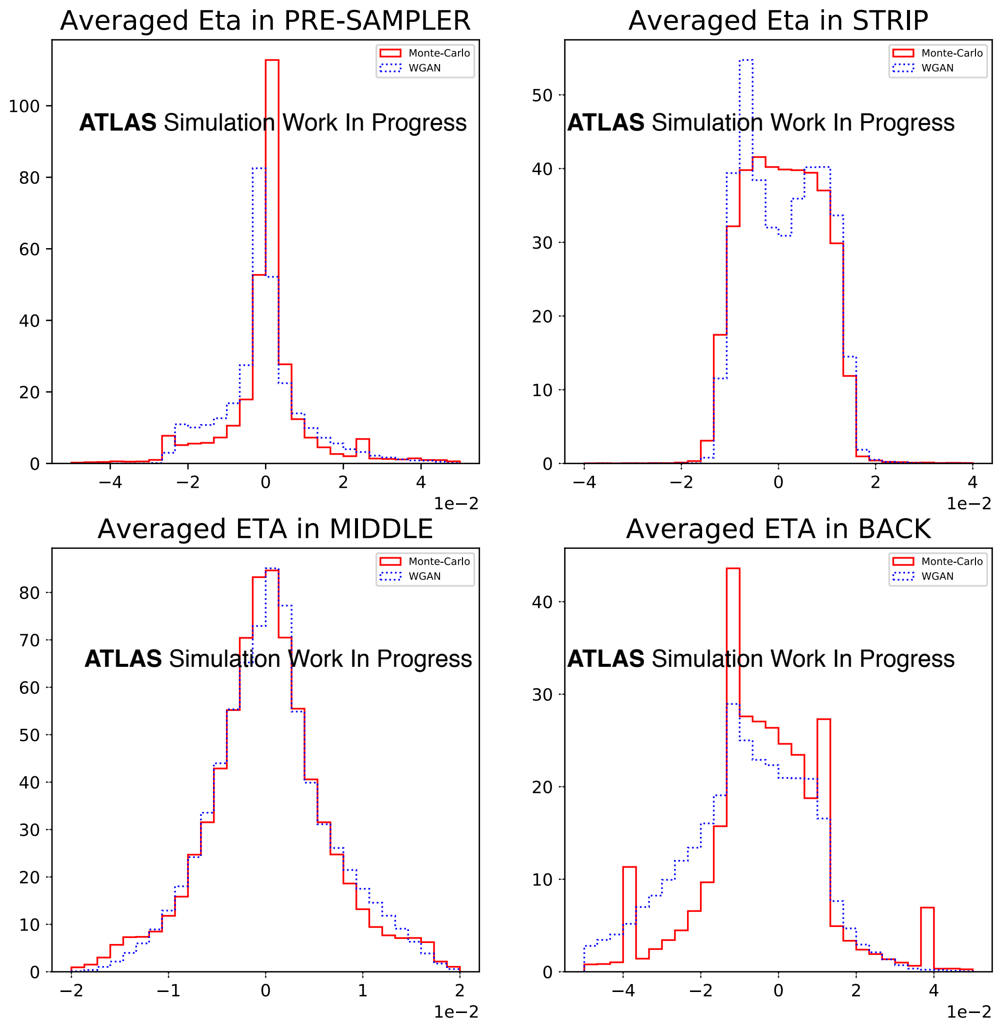


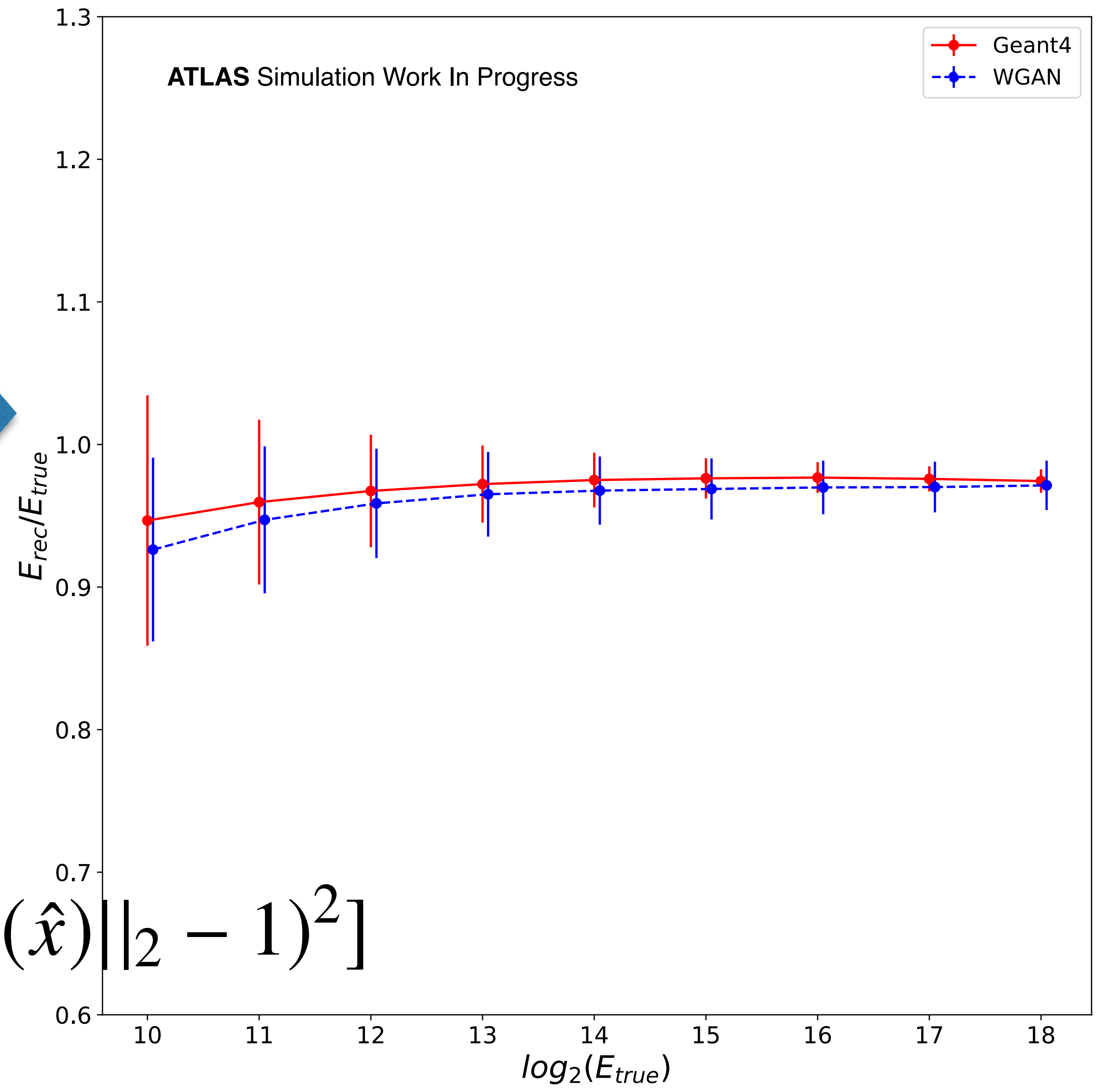


Plots get worse

Energy gets better

Gradient Penalty = 1e-13



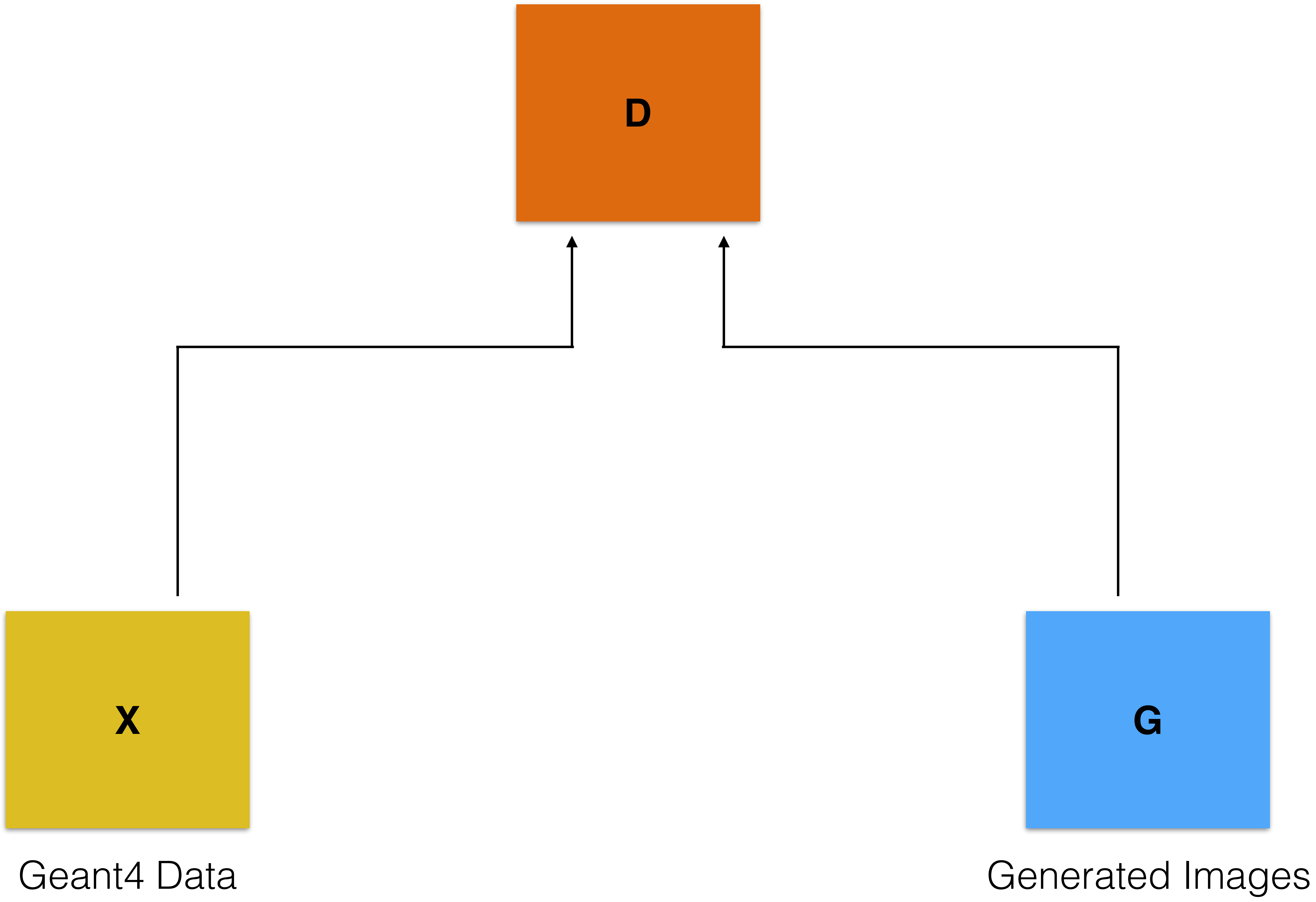


Never seen such a number in literature

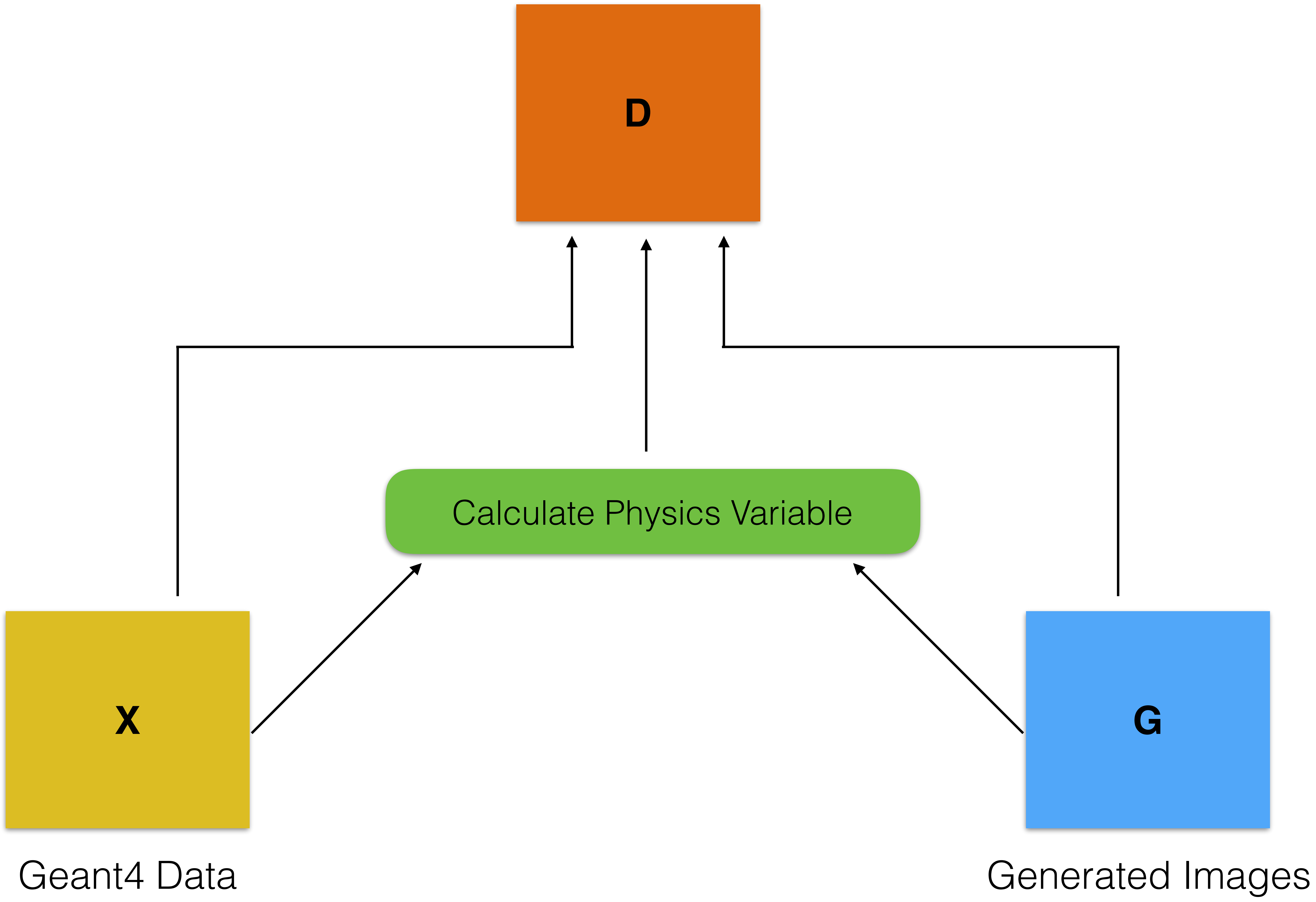
And highly unstable training

$$L_{\mathrm{GAN}} = \underset{\tilde{x}\sim p_{\mathrm{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\mathrm{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

Add Physics Variables in Training

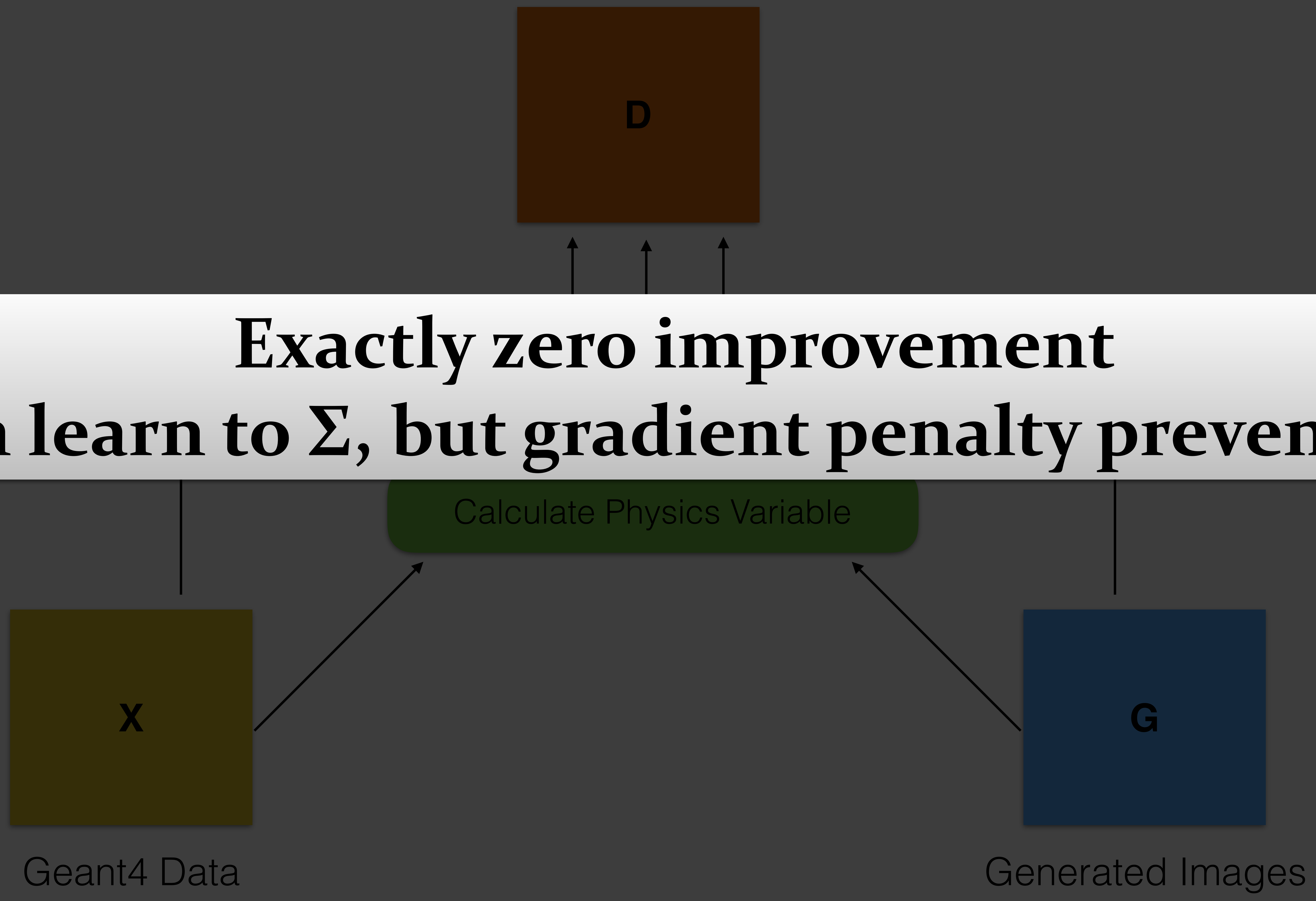


Add Physics Variables in Training



Help the discriminator see physics

Add Physics Variables in Training



Exactly zero improvement

Critic can learn to Σ, but gradient penalty prevents using it

Help the discriminator see physics

Stop Gradients through Σ

When you train the Generator

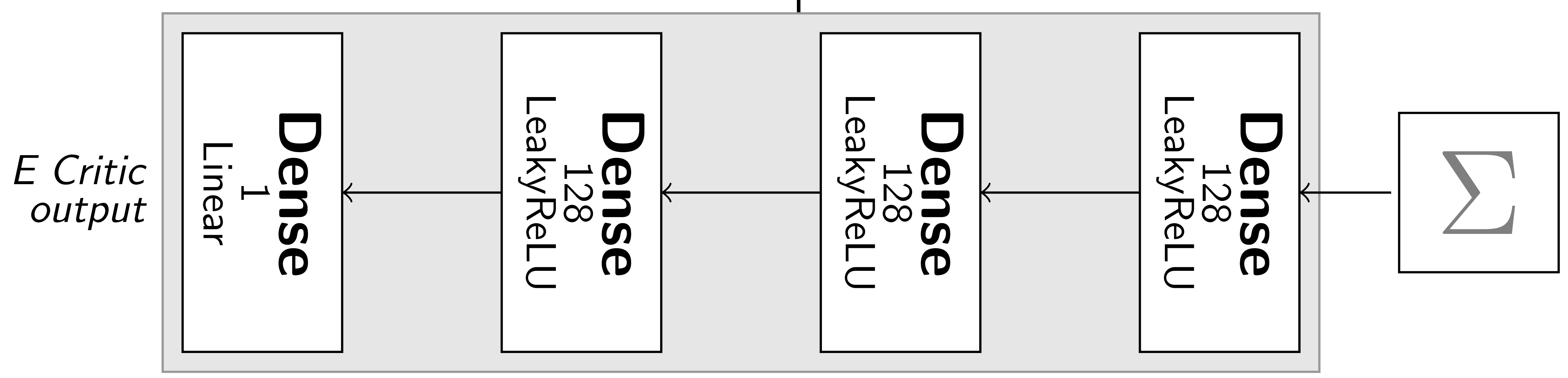
Gradients useful for Generator

When you train the Critic

Treat Σ as independent input feature, not as a sum of the other 266 features

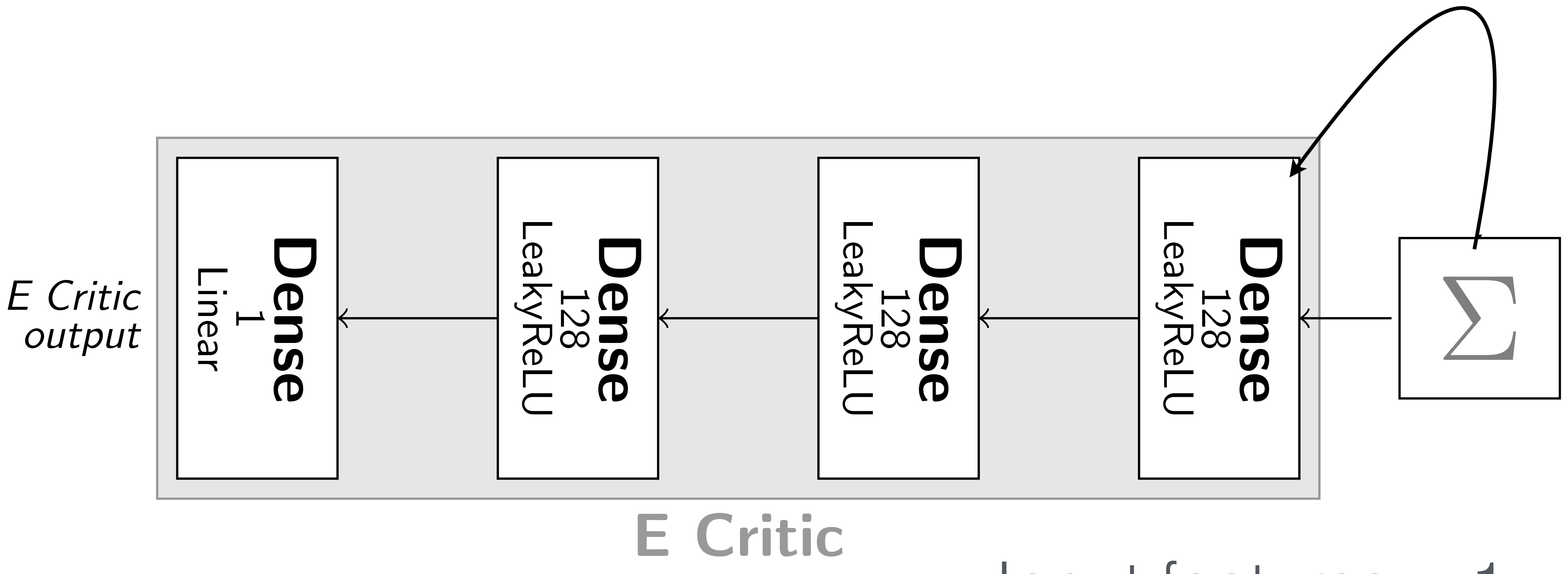
No gradient penalty on it

$$L_{\text{GAN}} = \mathop{E}_{\tilde{x}\sim p_{\text{gen}}}[D(\tilde{x})] - \mathop{E}_{x\sim p_{\text{Geant4}}}[D(x)] + \lambda \mathop{E}_{\hat{x}\sim p_{\hat{x}}}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$



Careful: Sum Inside or Outside the Network?

$\boxed{\sum}$ `= Lambda(sumFunc)(m_input_image)`



Input features = 1 + Conditional

They are not equivalent, need to tune hyper parameters differently

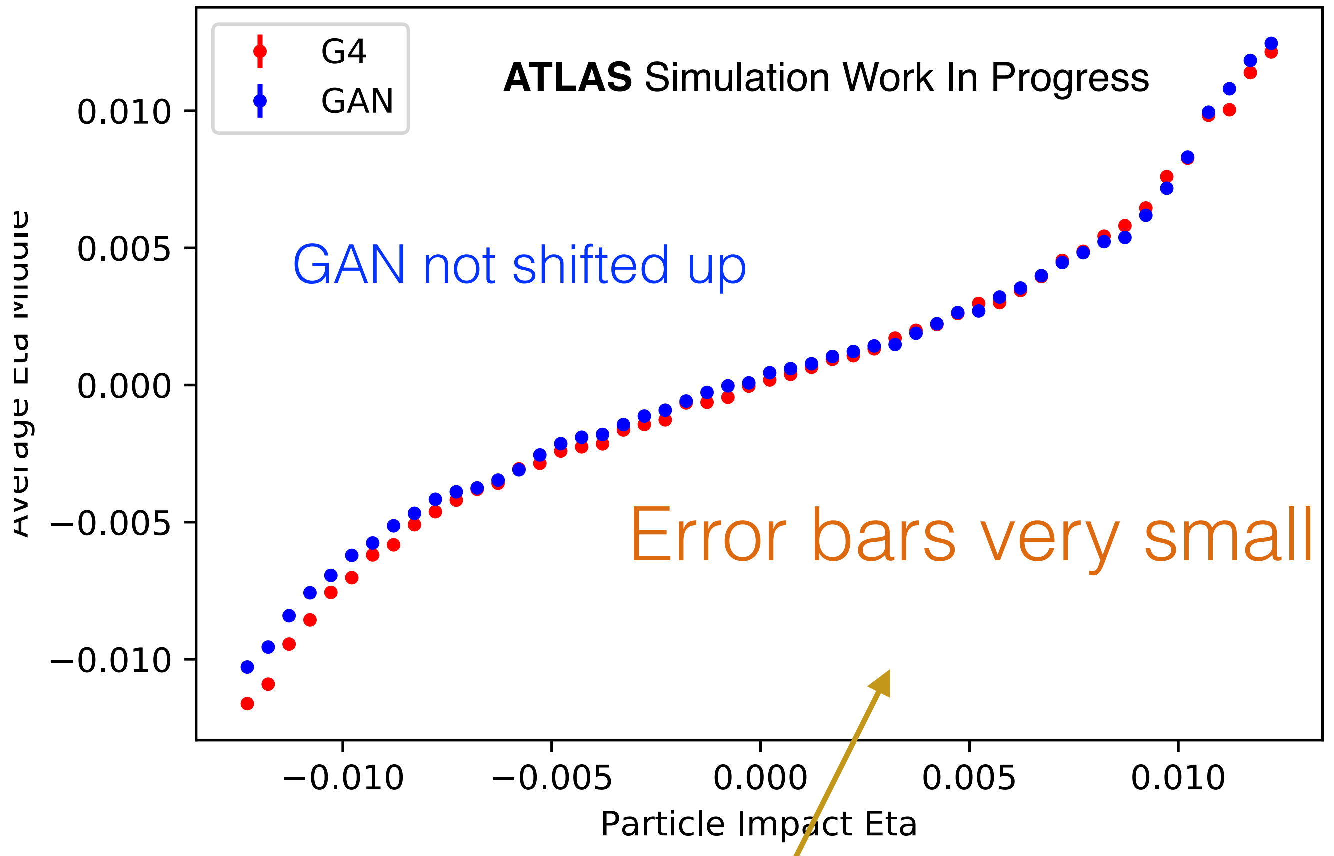
$$L_{\mathrm{GAN}} = \underset{\tilde{x}\sim p_{\mathrm{gen}}}{E}[D(\tilde{x})] - \underset{x\sim p_{\mathrm{Geant4}}}{E}[D(x)] + \lambda \underset{\hat{x}\sim p_{\hat{x}}}{E}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2]$$

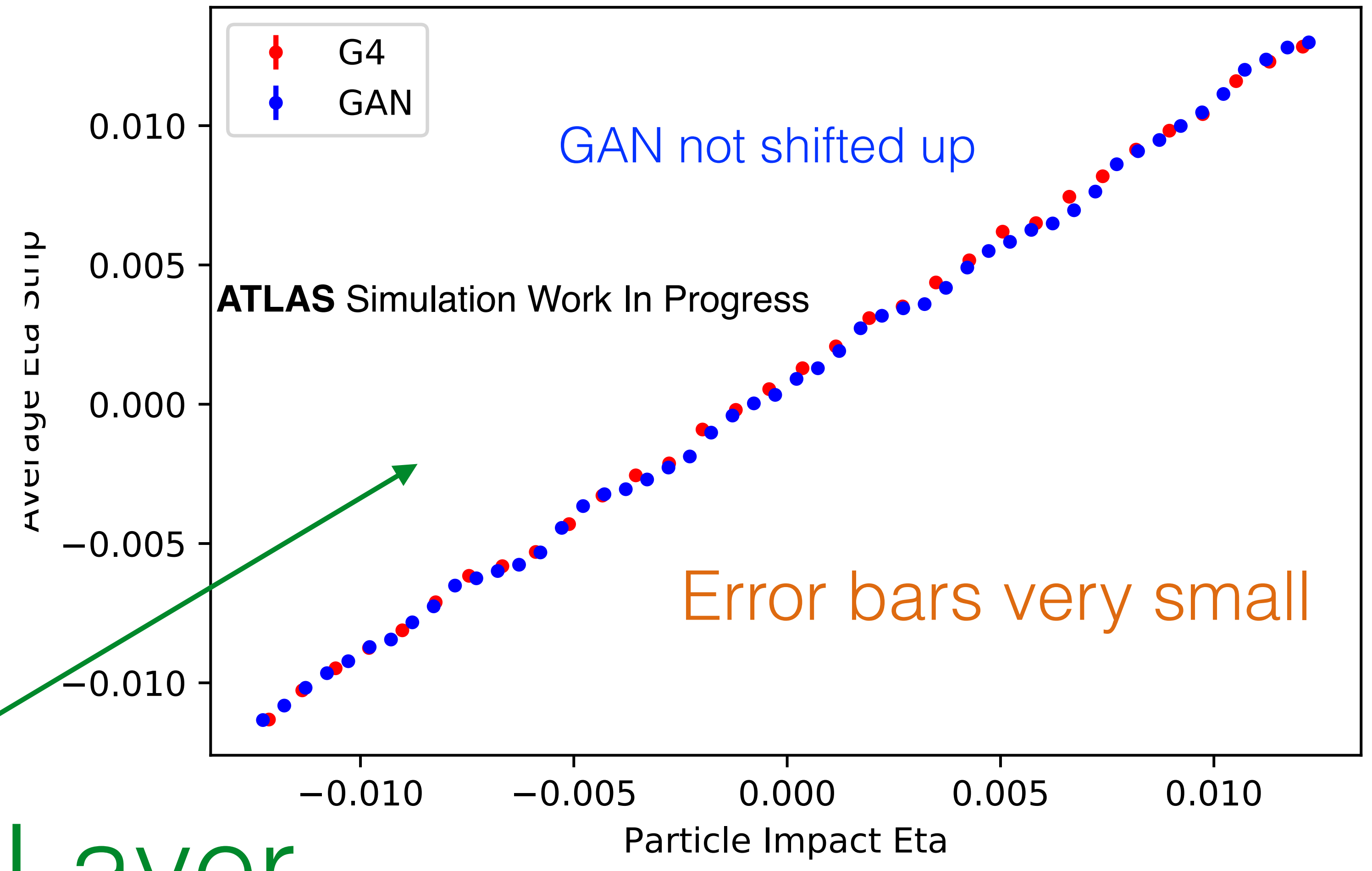
Gradient Penalty on 1 input vs 266 inputs

S Shape

65 GeV Photons Only

Avg Eta vs Particle Eta

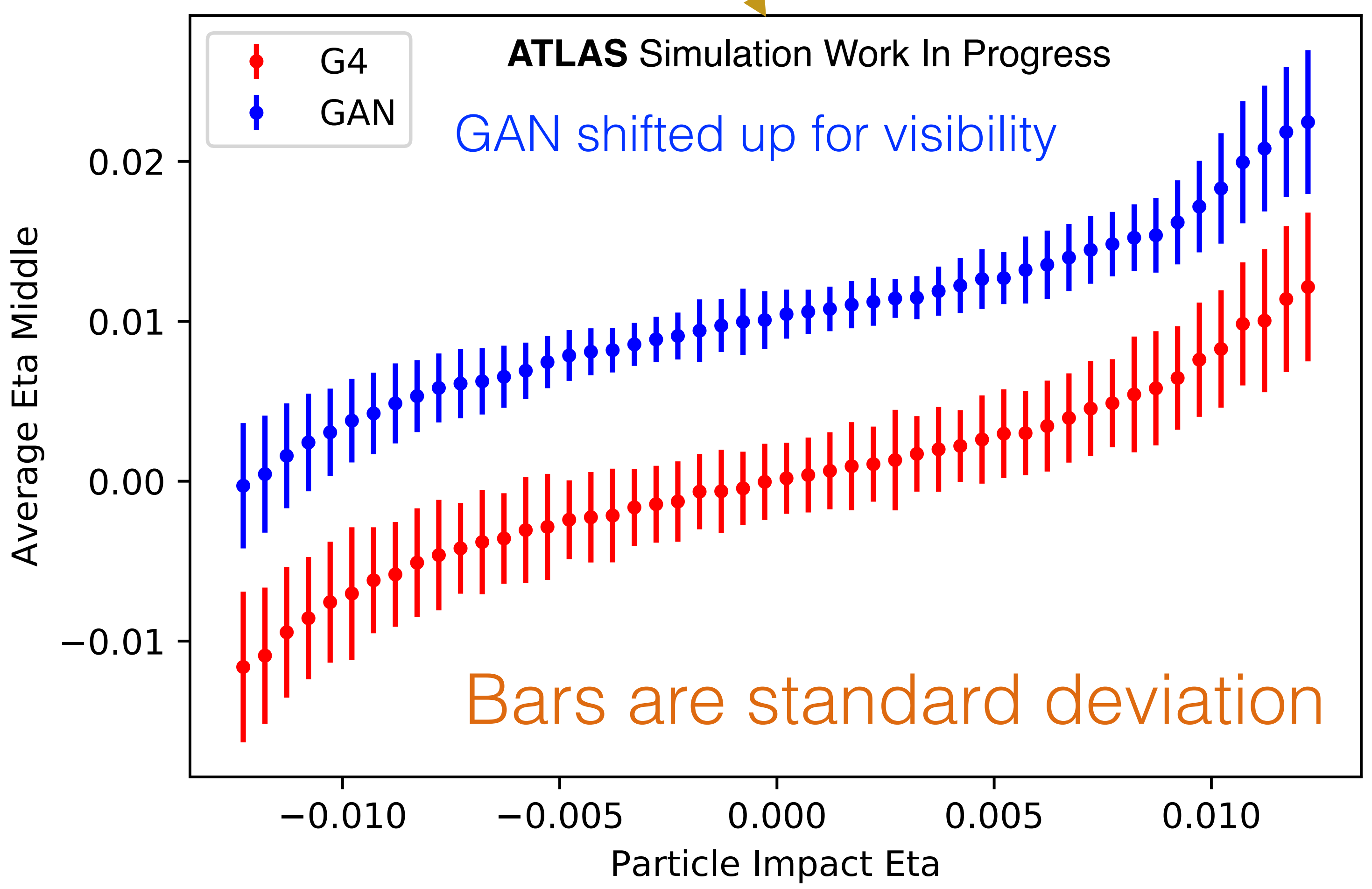


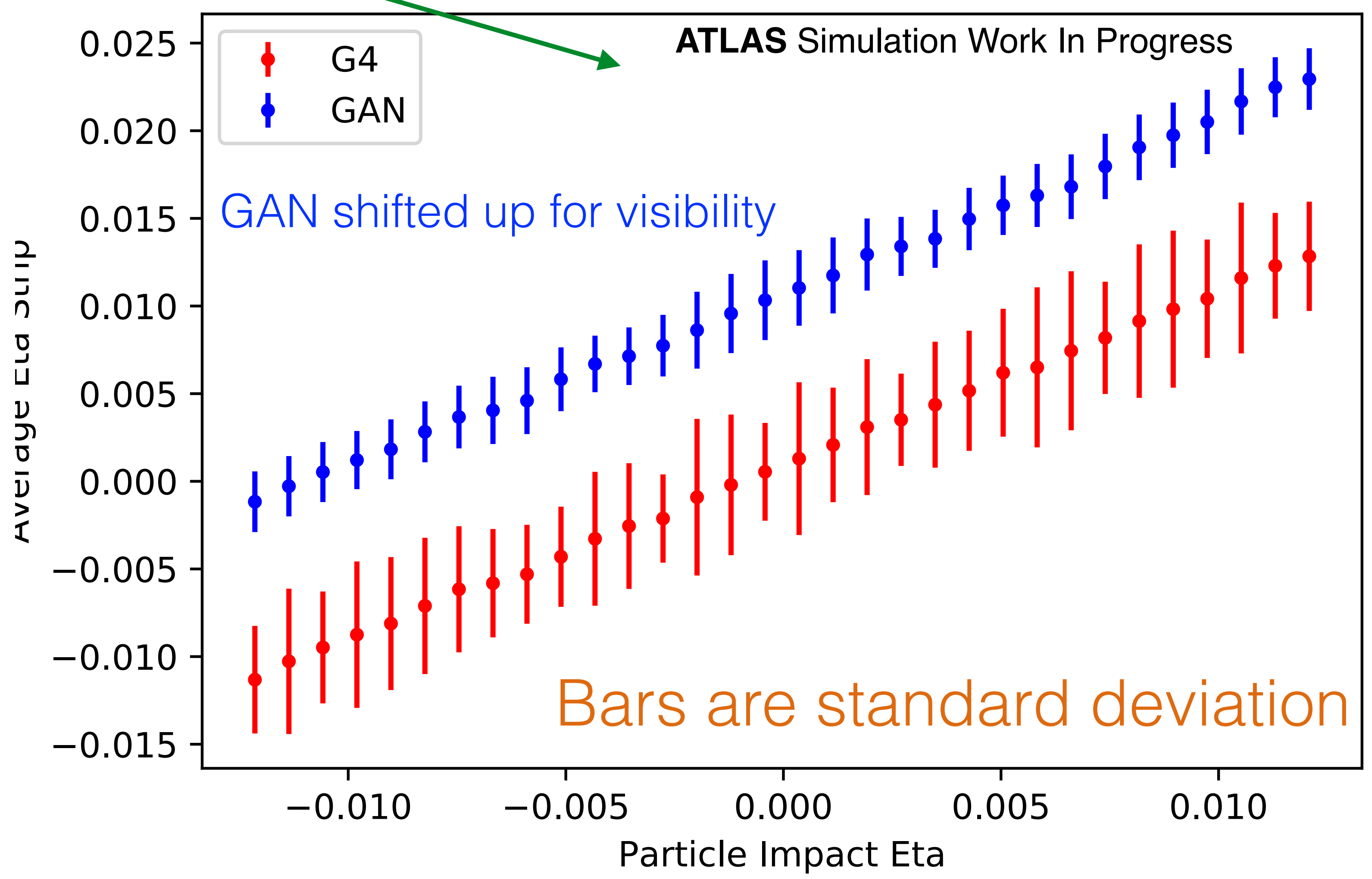


Middle Layer

Strip Layer

8 stip cells = 1 Middle Cell => 8 bumps



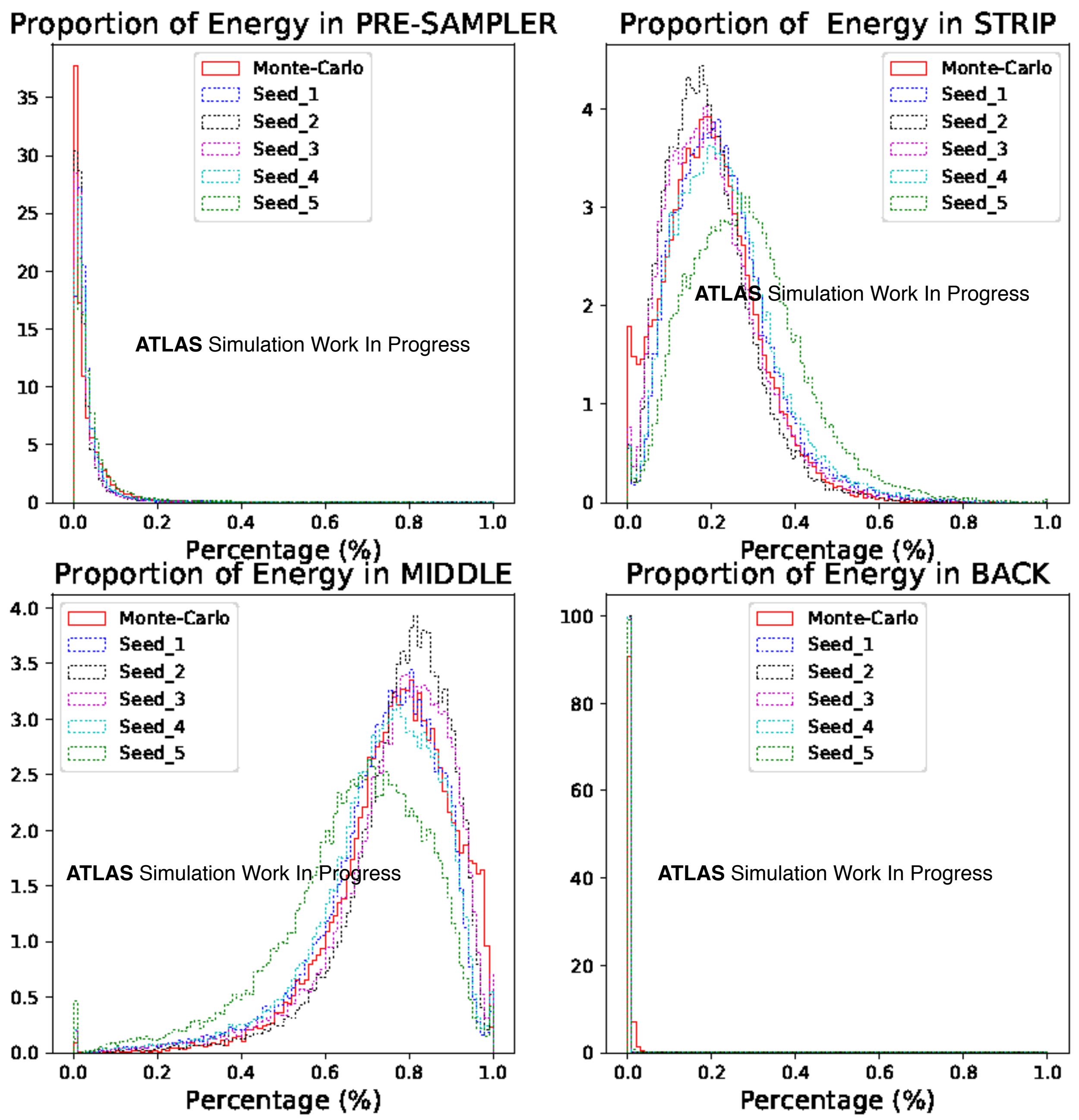


Statistical analysis of HPO results

Chi2, KS, AD tests not useful

Statistical analysis of HPO results

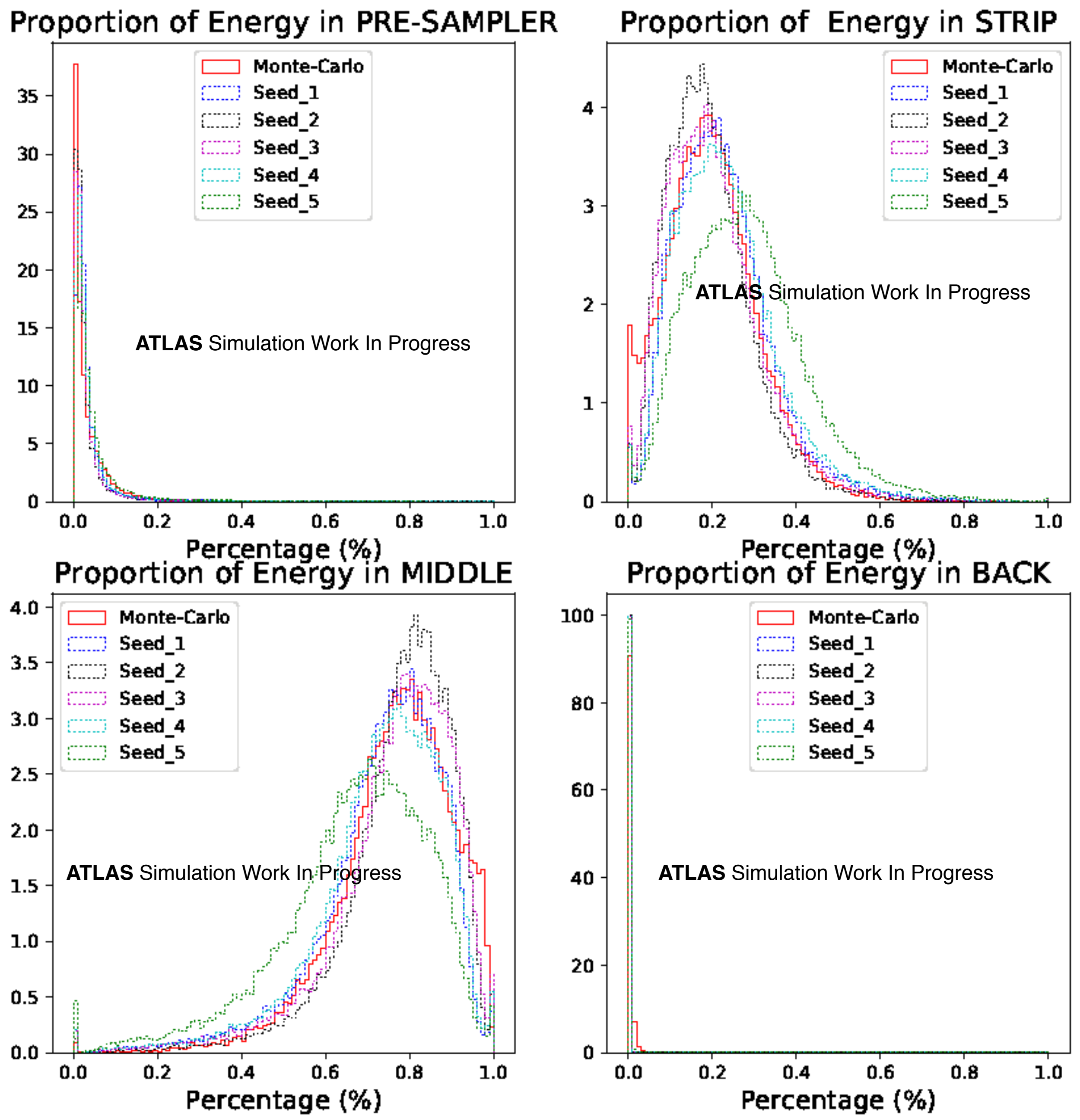
Different training seeds



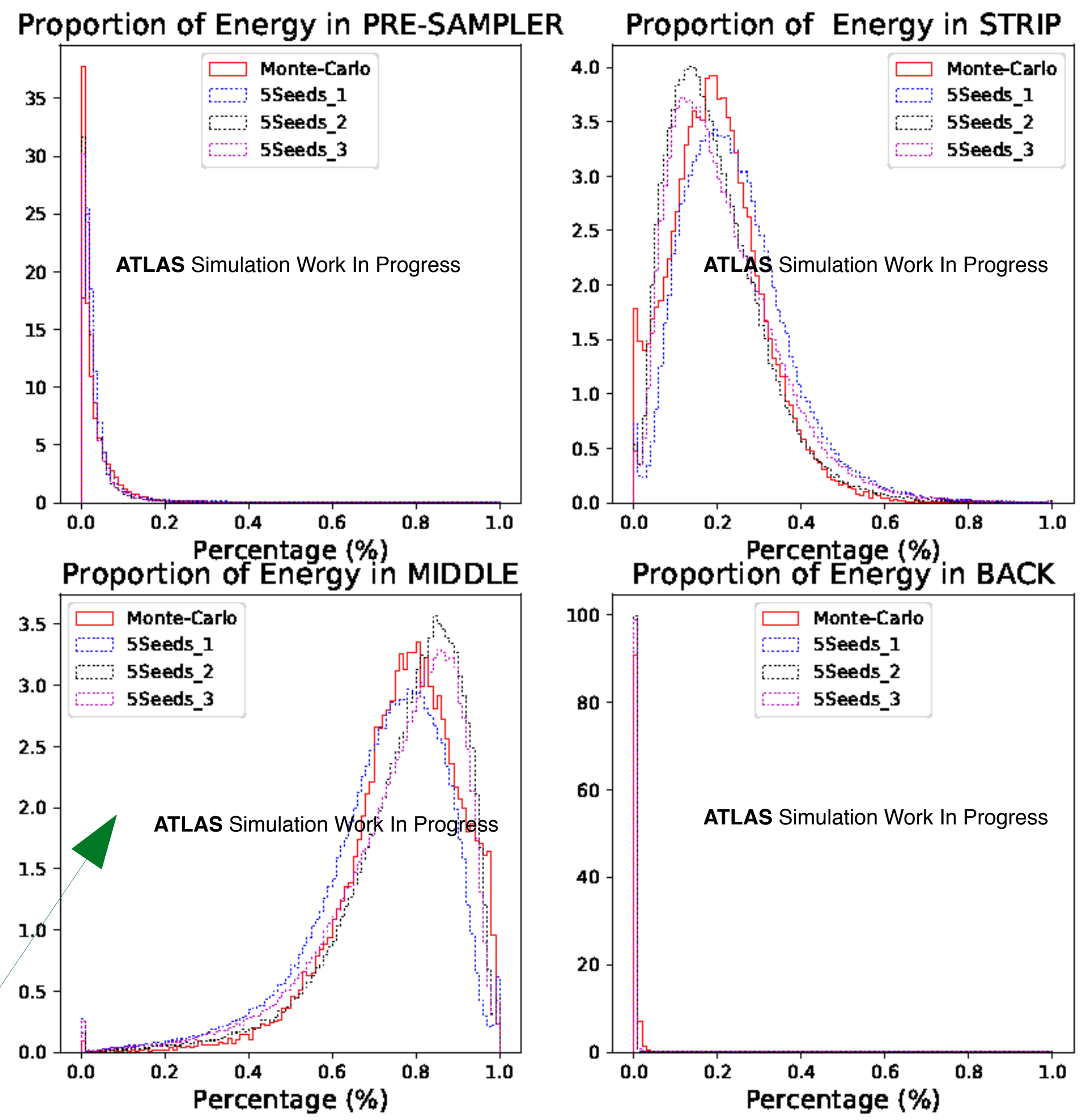
Chi2, KS, AD tests not useful

Statistical analysis of HPO results

Different training seeds



Average 5 trainings
(3 sets averages)

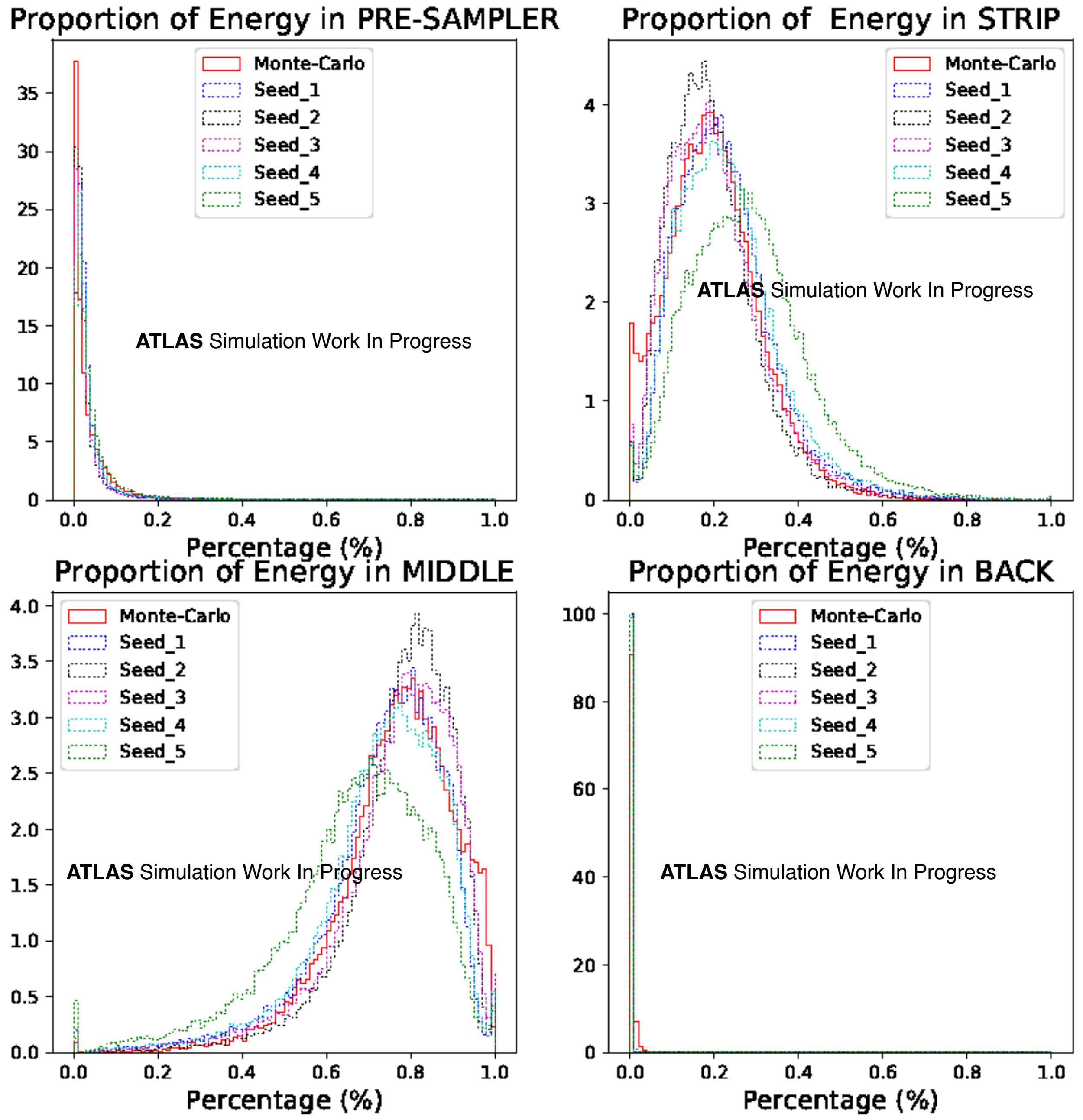


“Baseline”

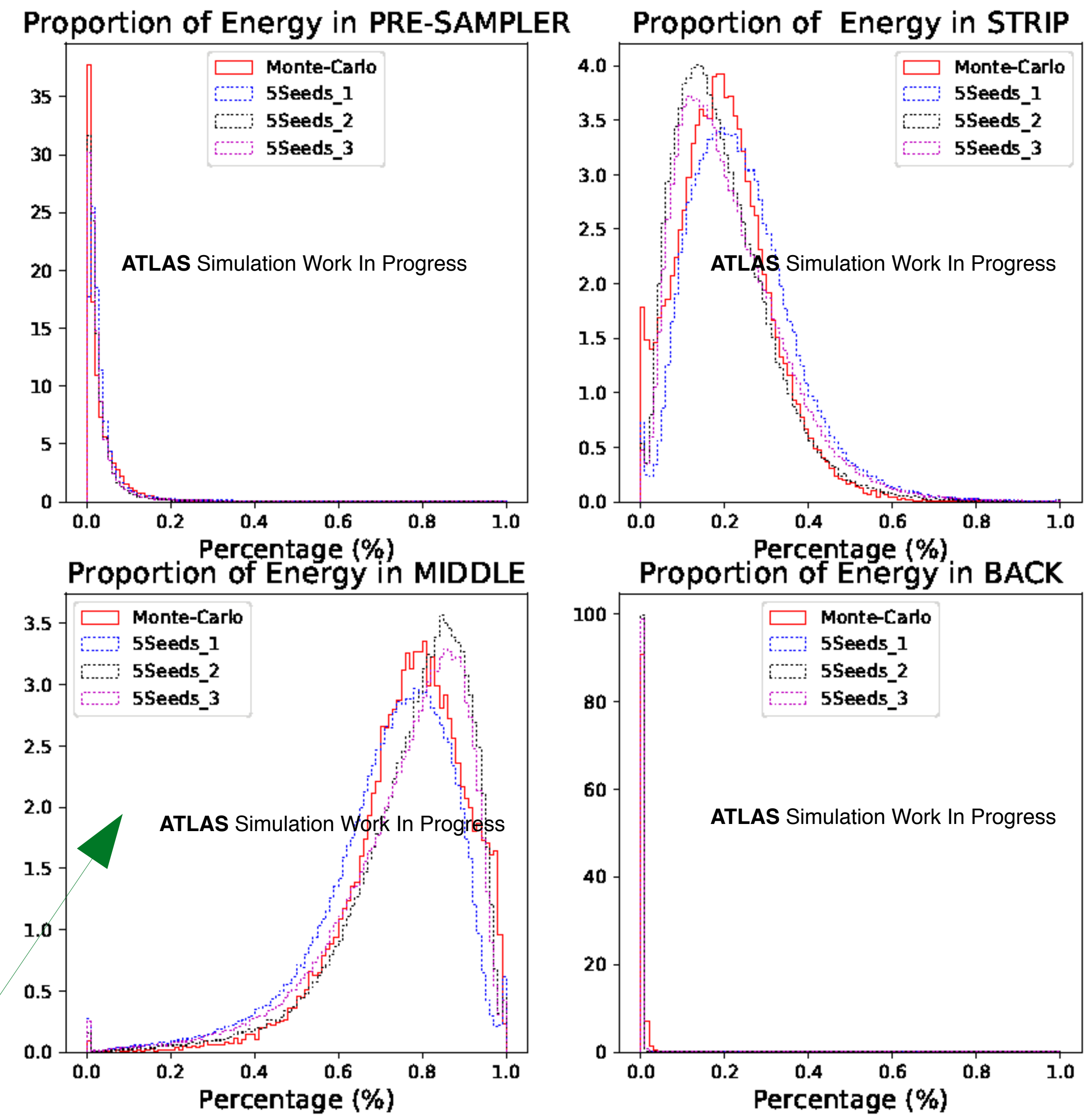
Chi2, KS, AD tests not useful

Statistical analysis of HPO results

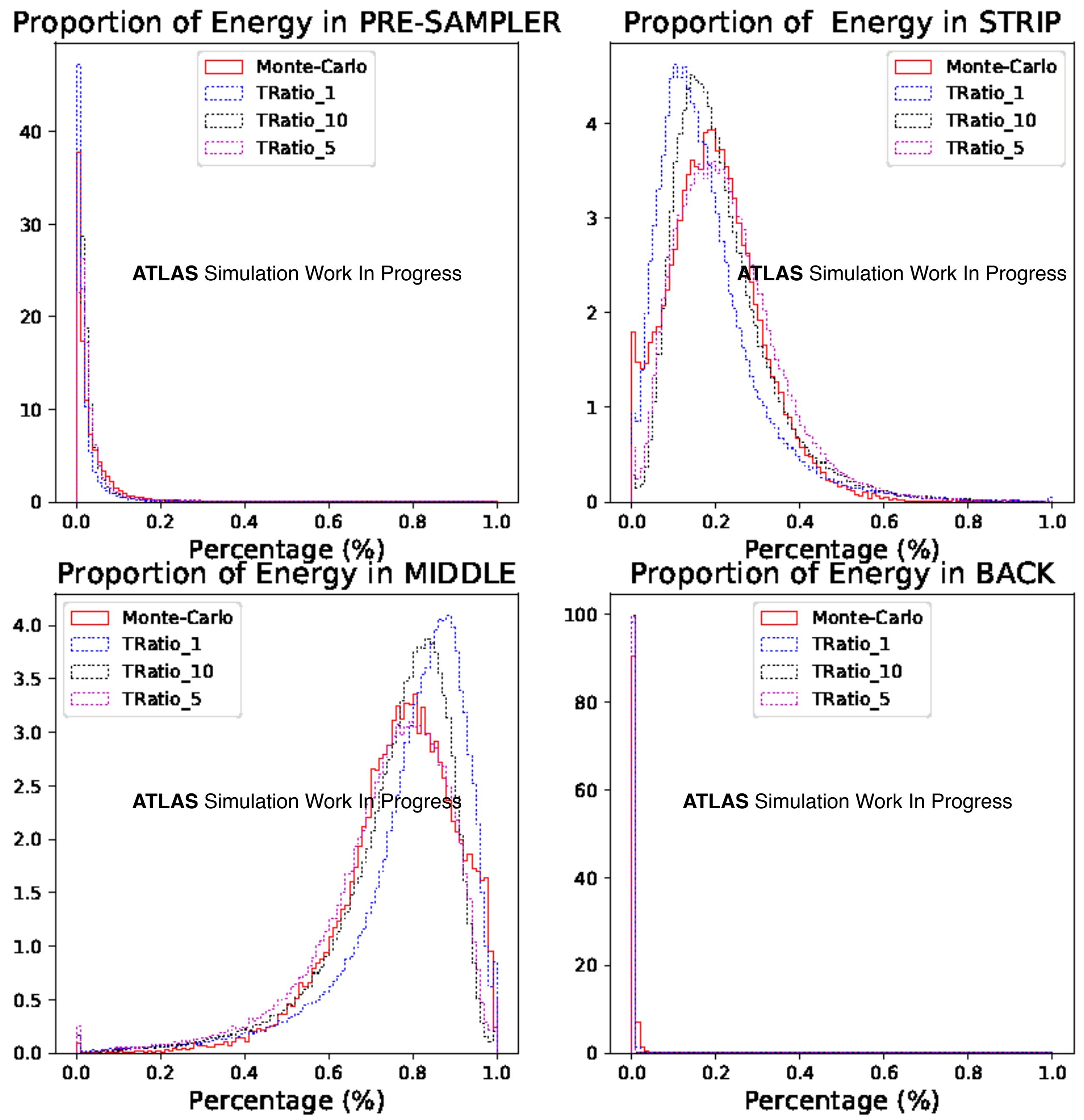
Different training seeds



Average 5 trainings
(3 sets averages)



1,5,10 Ratio; average 5 seed training for each

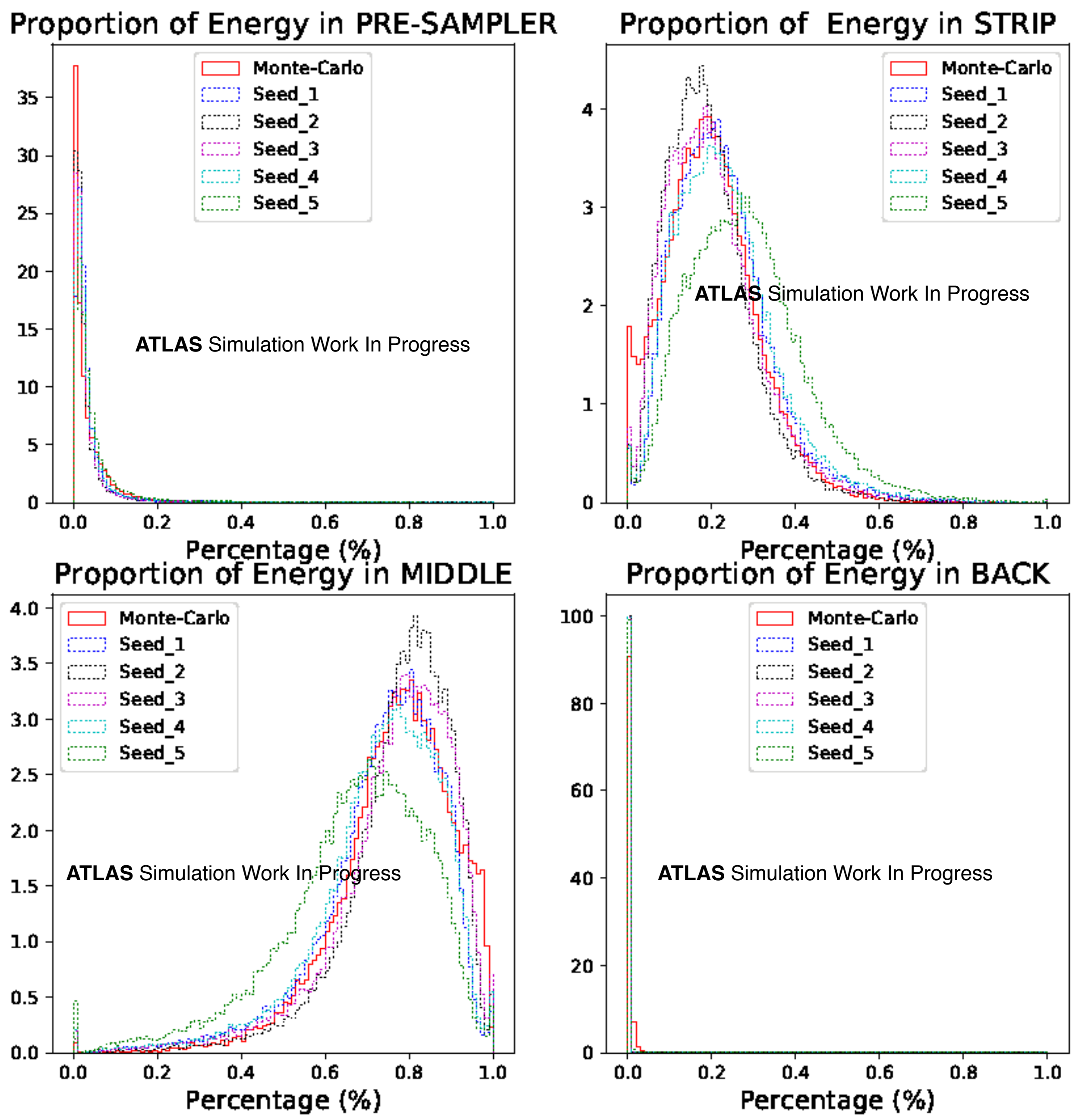


“Baseline”

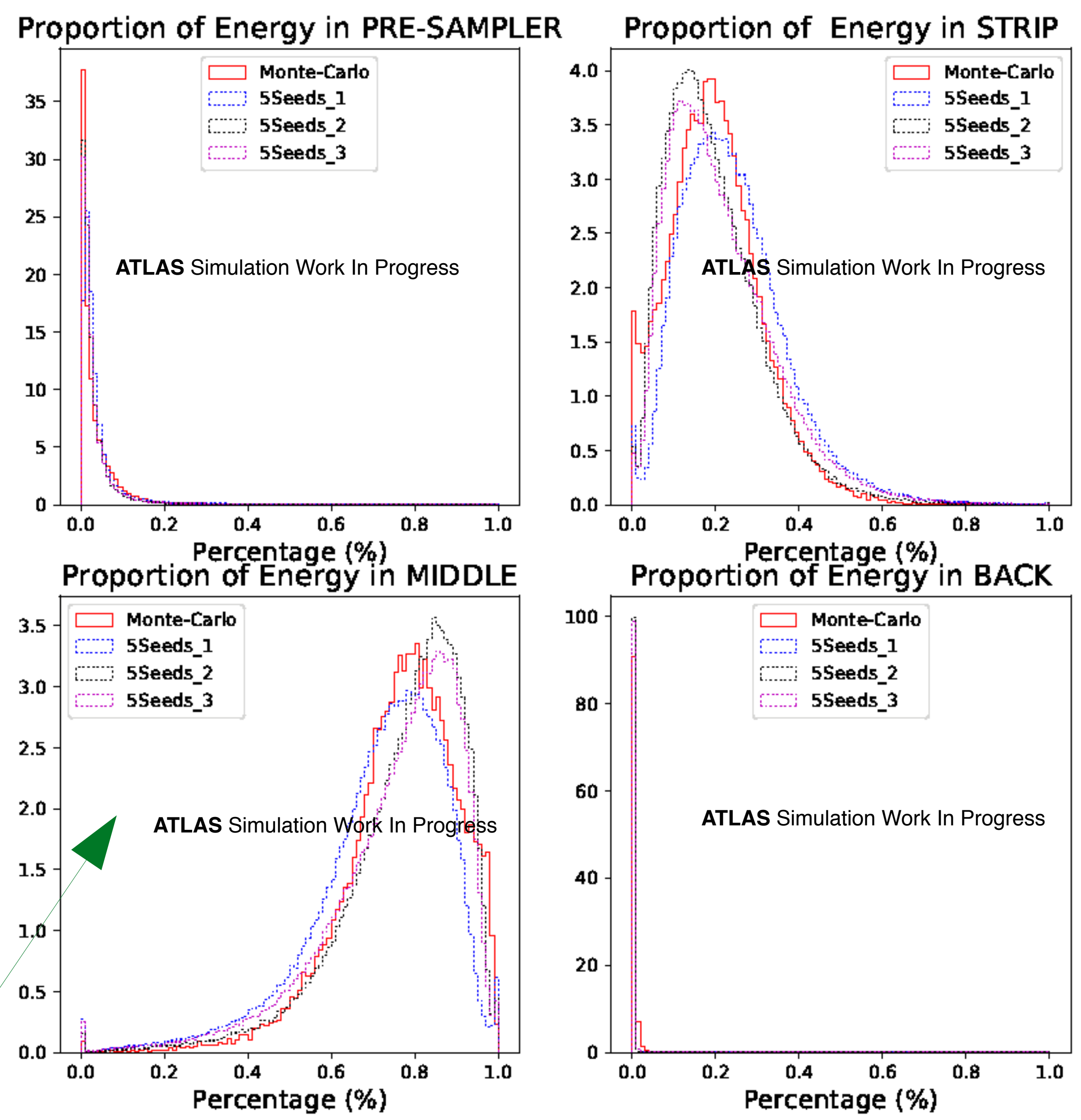
Chi2, KS, AD tests not useful

Statistical analysis of HPO results

Different training seeds

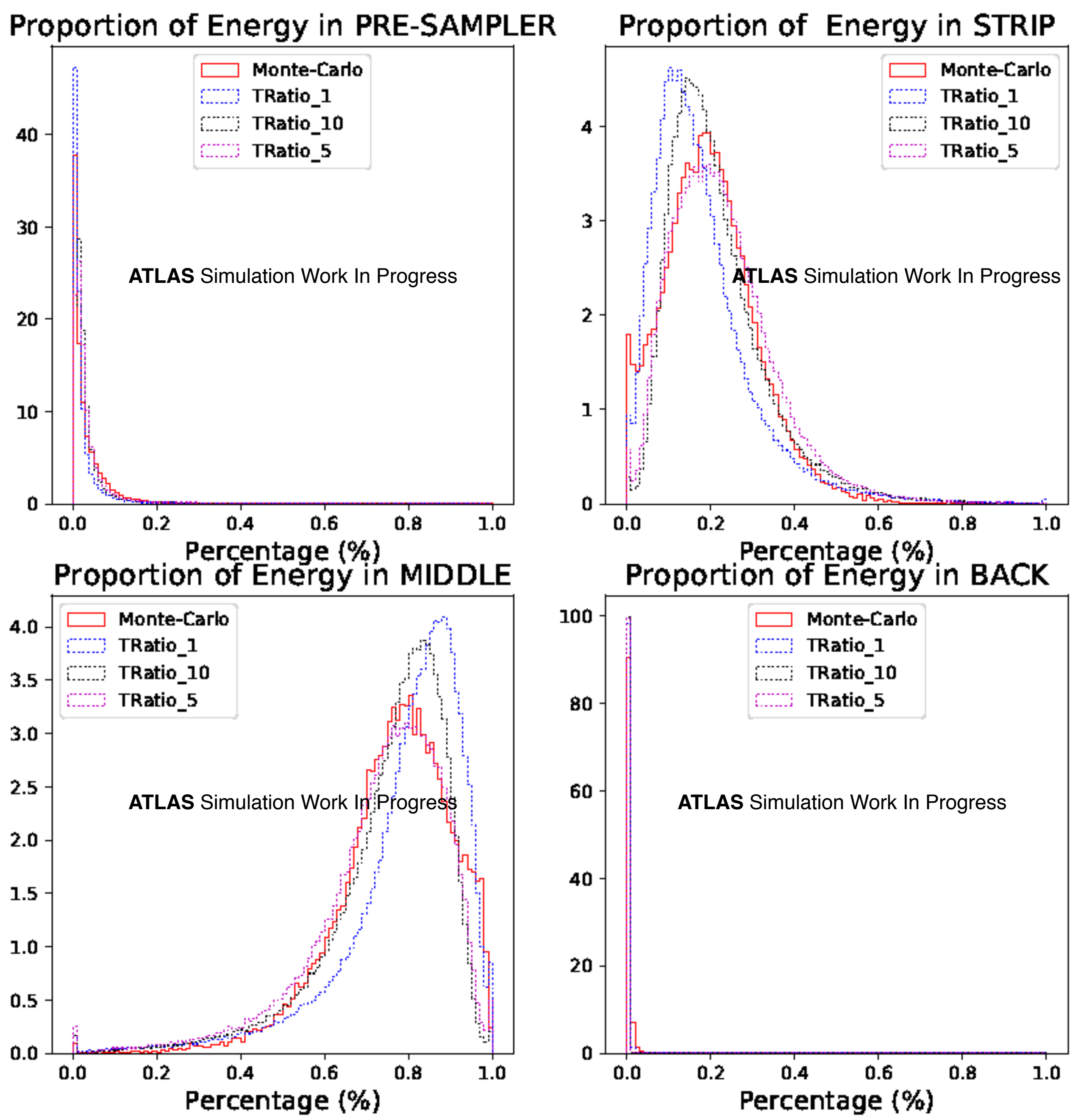


Average 5 trainings
(3 sets averages)



“Baseline”

1,5,10 Ratio; average 5 seed training for each



Claim: Training Ratio of 5 is good

Chi2, KS, AD tests not useful

Useful to make such assessments at initial R&D stage

GAN Alchemy (May not generalise)

- Momentum can work against you in adversarial training (Adam -> RMSProp)
- Results peak at certain epoch, then consistently deteriorate
  - Which epoch is a function of number of updates to generator (smaller epoch for more data / smaller batch size)
- Upgrading Keras TF versions consistently improves results
  - Despite deep investigation, no explanation
  - Older versions were more stable, newer ones require epoch picking
- Best way to HPO ? : Grad Student Decent
- Getting all conditionings right simultaneously requires luck, epoch picking
  - We want to stick to hyper-parameters that get plots right more consistently during R&D stage
  - Do whatever is necessary to get the best model at the final stage
- WGAN-GP hyper-parameter can suddenly have meaningful impact at 1e-13 ! Never seen in literature

Trainable Swish Activation

$$Swish(x) = x \cdot \mathrm{sigmoid}(\beta x)$$

Trainable β

SEARCHING FOR ACTIVATION FUNCTIONS

Prajit Ramachandran,* Barret Zoph, Quoc V. Le
Google Brain
{prajit,barretzoph,qvl}@google.com

Discovered using Reinforcement Learning + Exhaustive Search

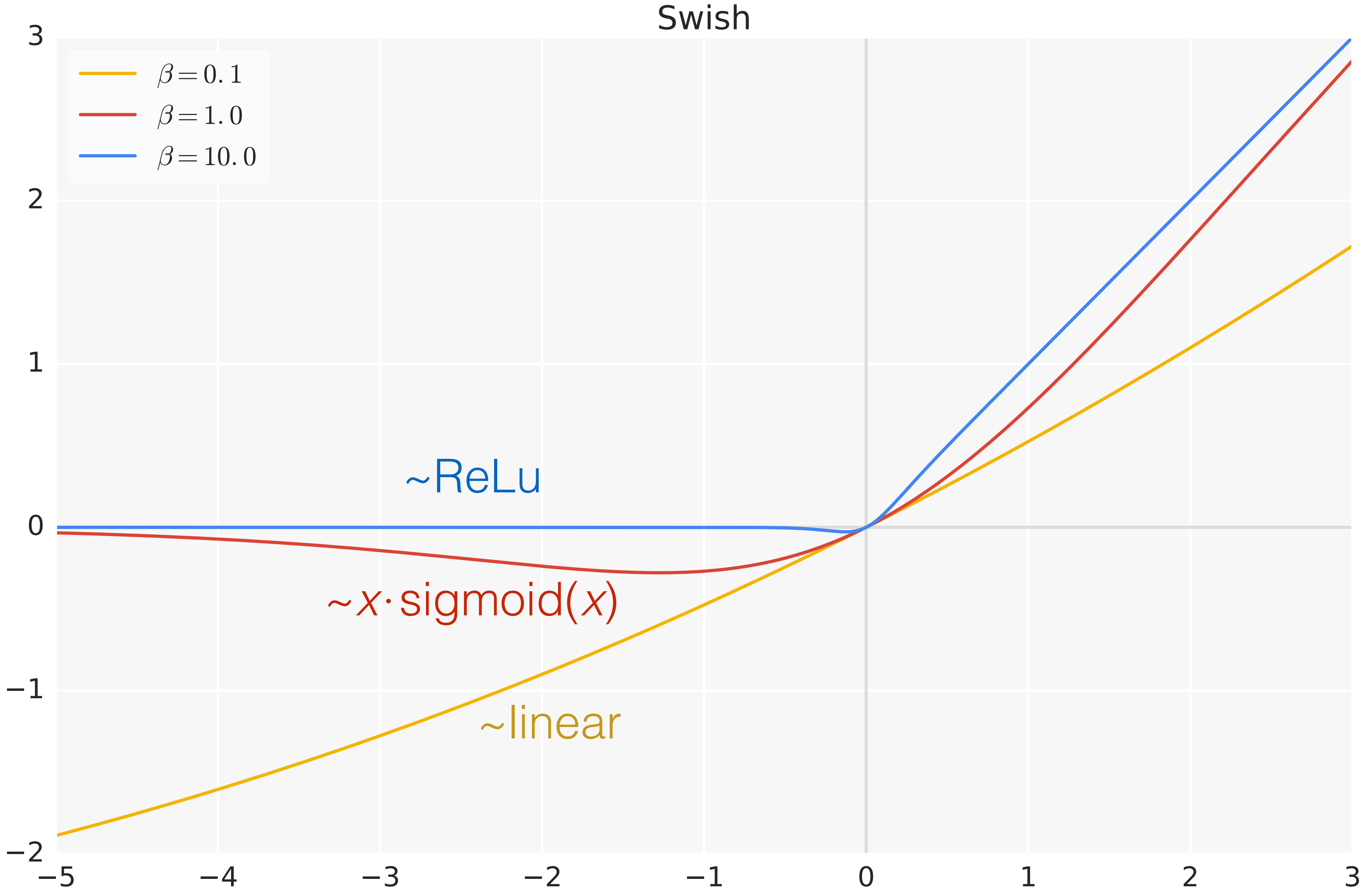


Figure 4: The Swish activation function.

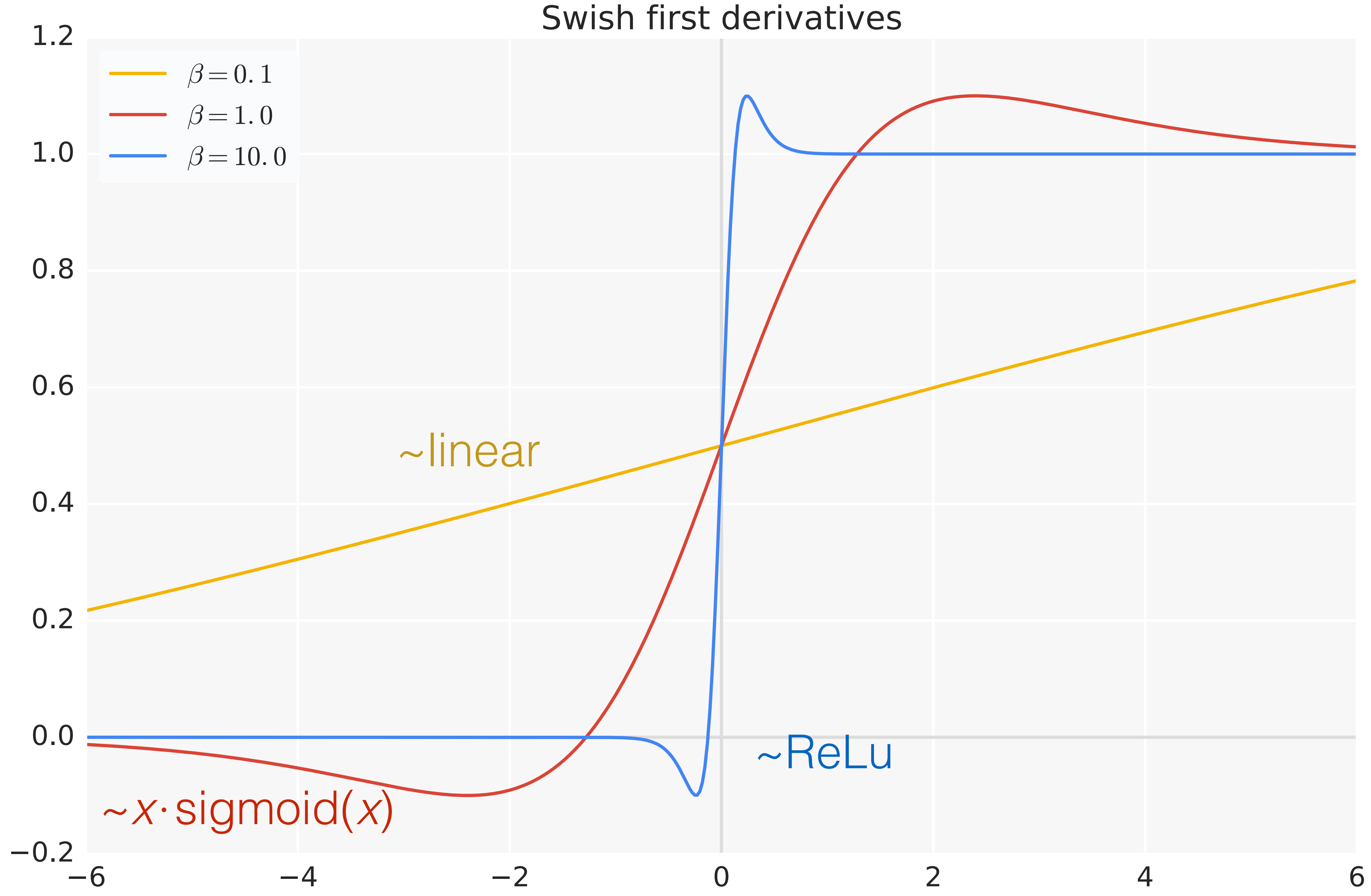
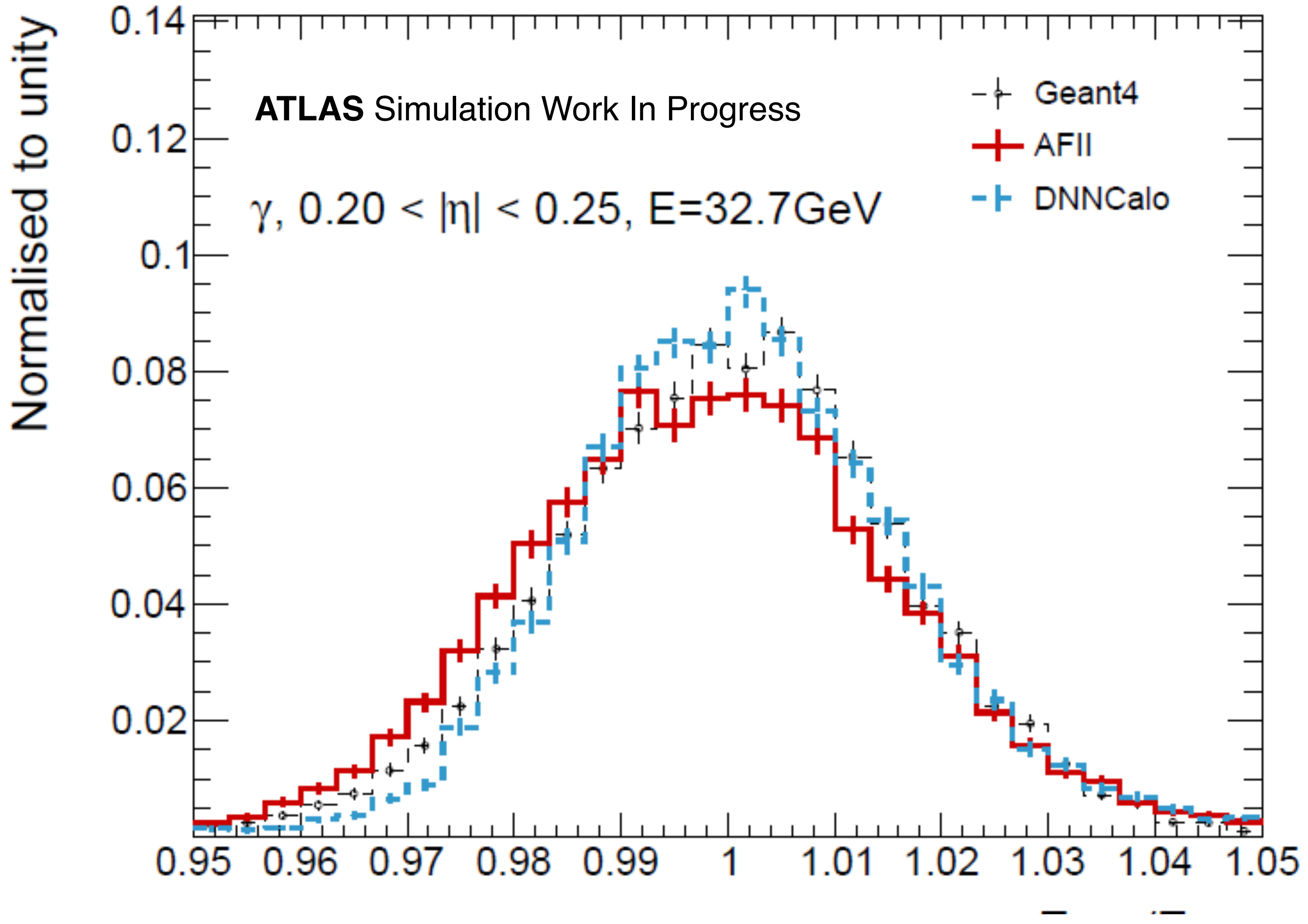
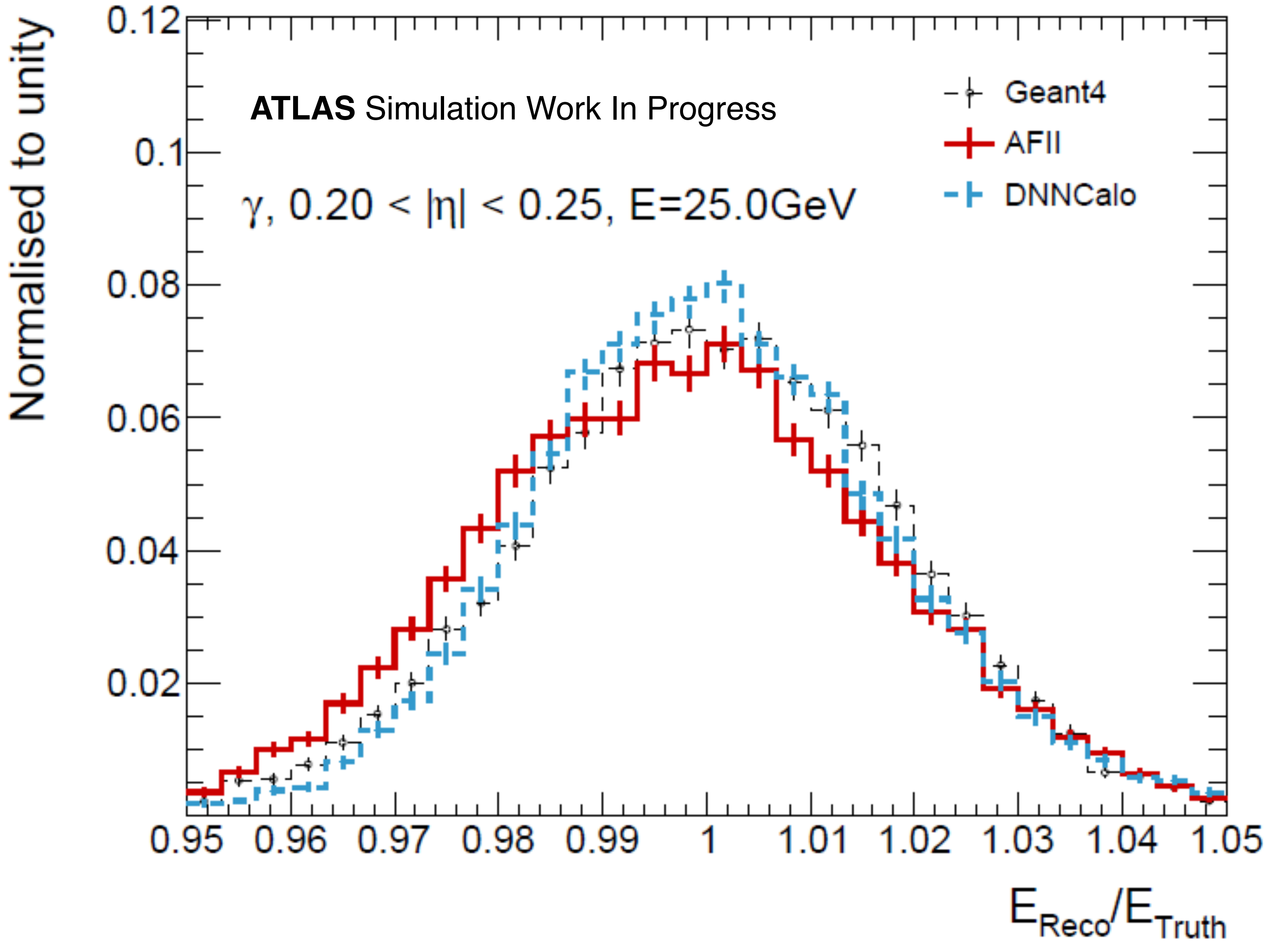
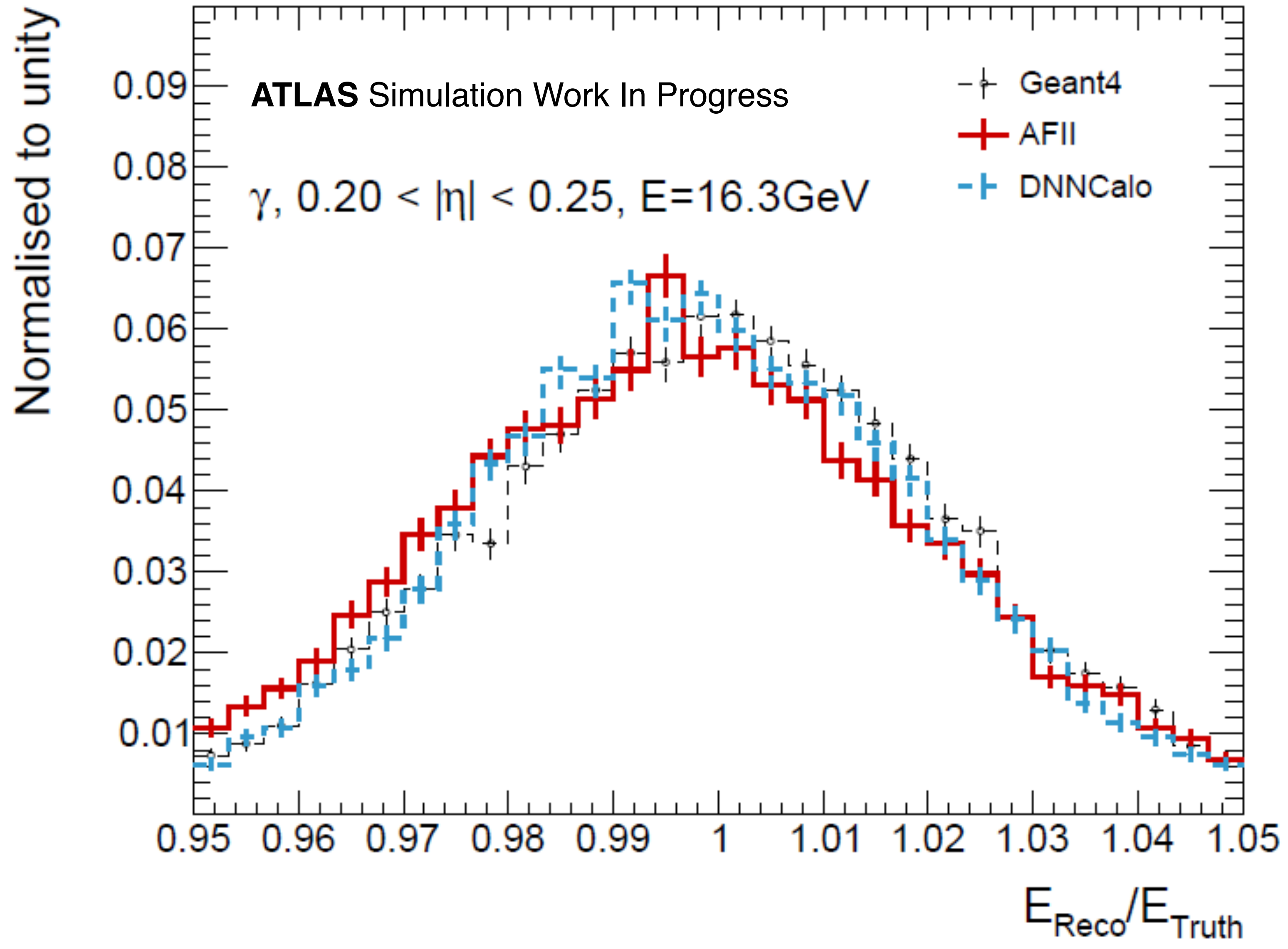


Figure 5: First derivatives of Swish.

High Stats Comparison With AF2



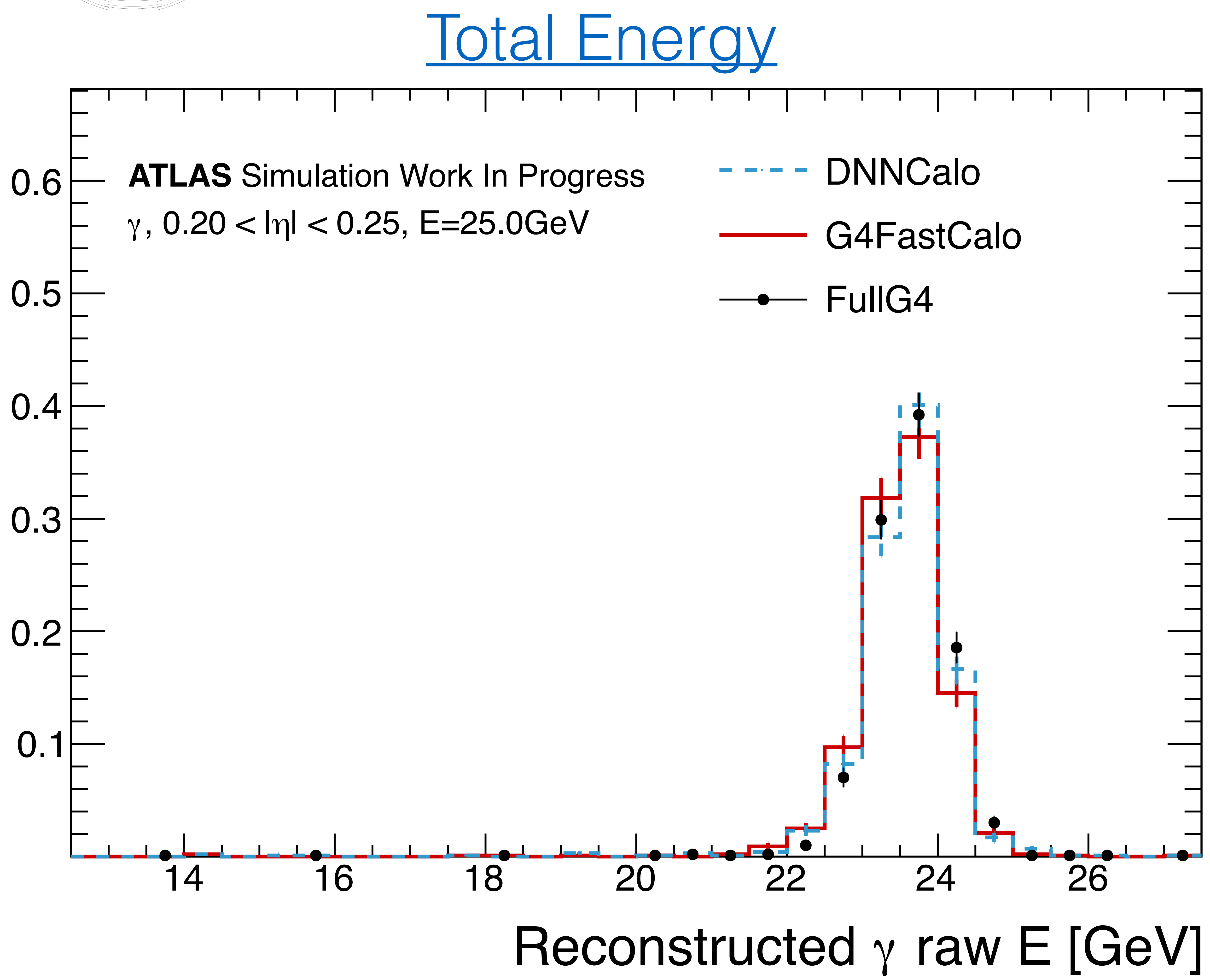


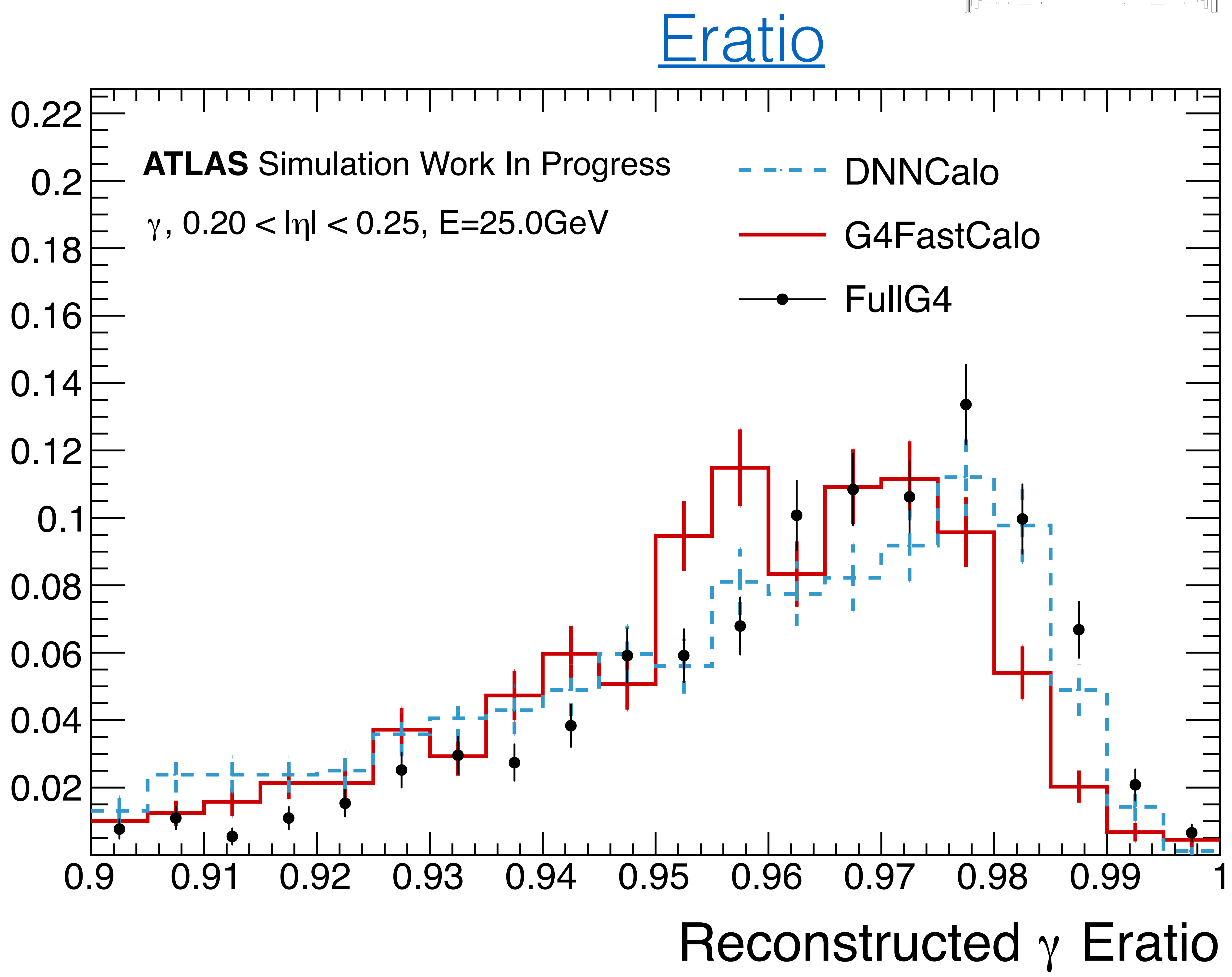


Really good agreement in cluster energy
Significantly better than AF2

Even for the interpolated point at 25 GeV

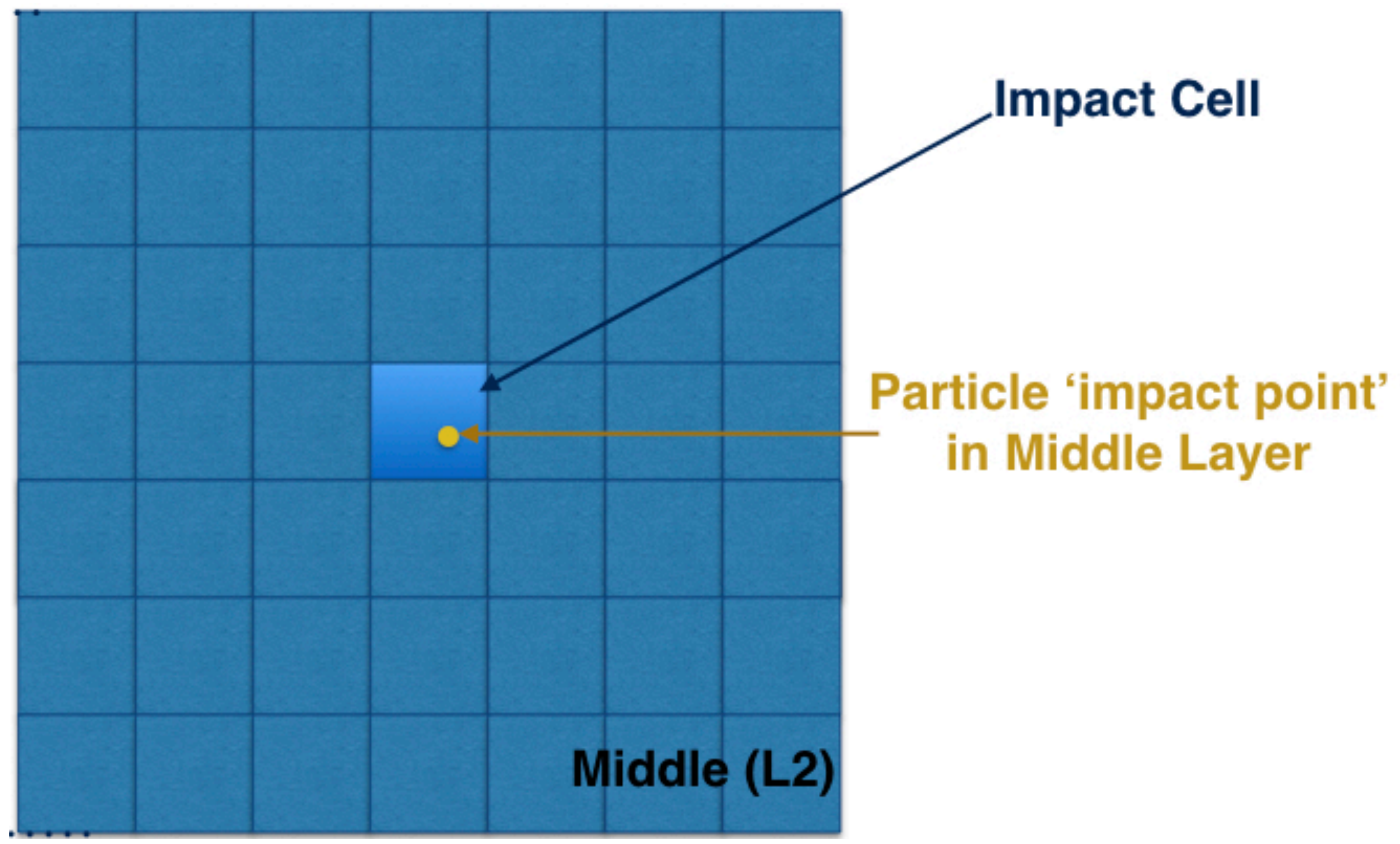
Interpolate Untrained 25 GeV





Proof of concept that 1 GAN can interpolate parameter space: extendible to eta, phi conditioning …

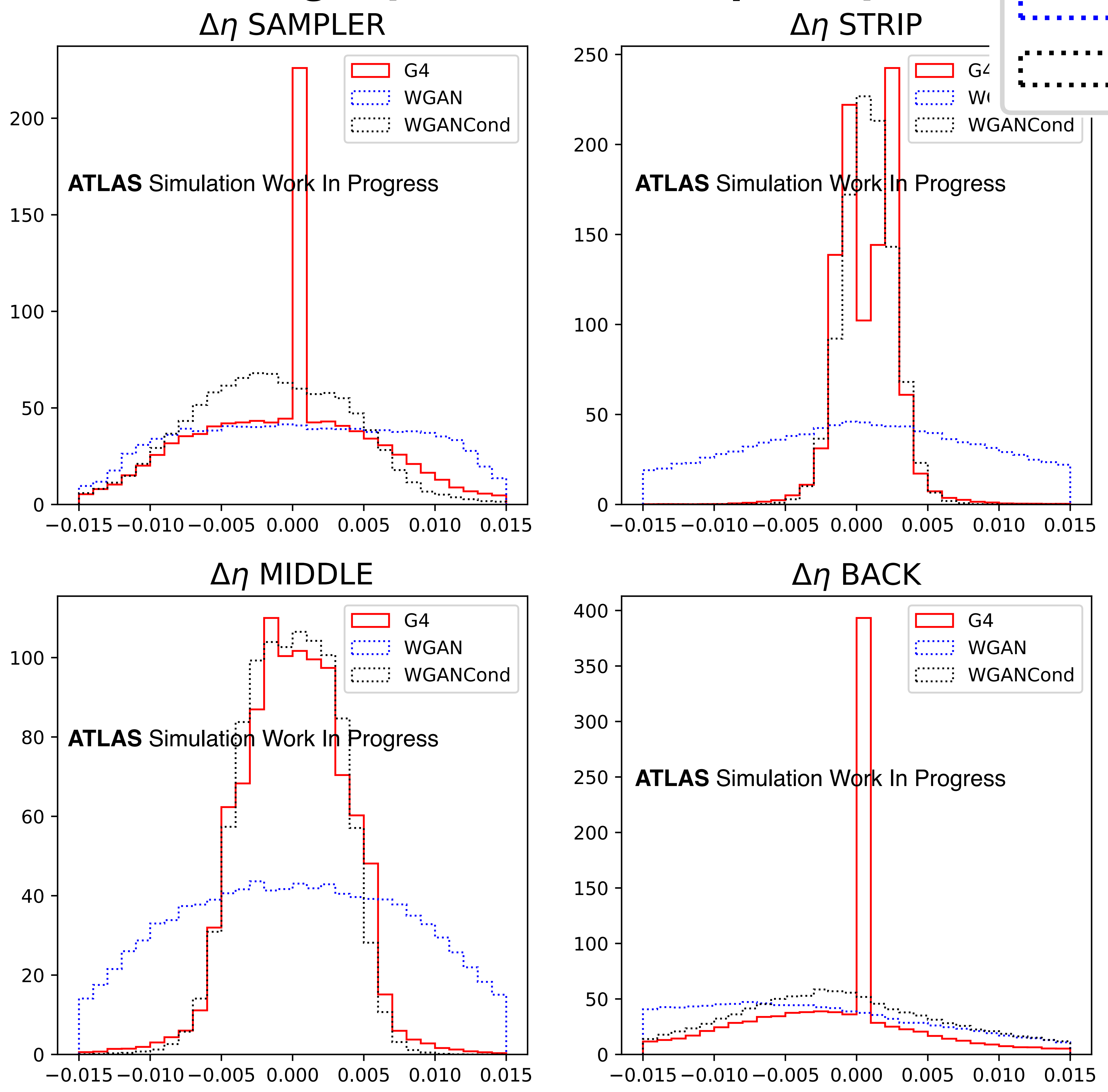
Condition GAN also on Impact Position of Particle

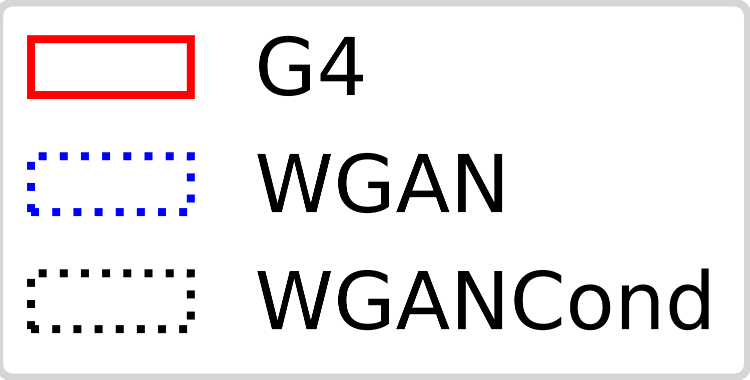


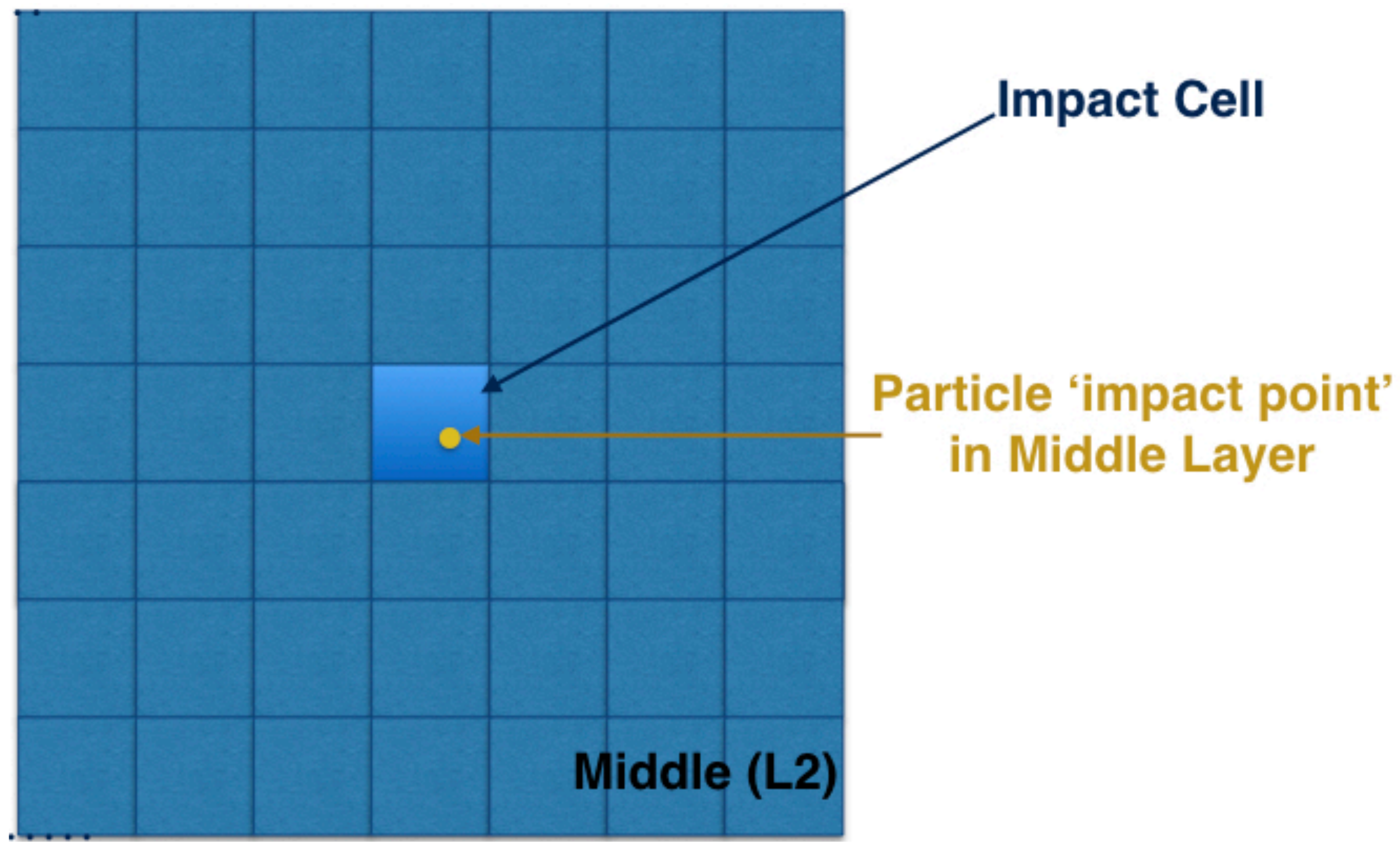
Continuous variable,
not class conditioning

Condition GAN also on Impact Position of Particle

<u>Average η - Particle's Impact η</u>



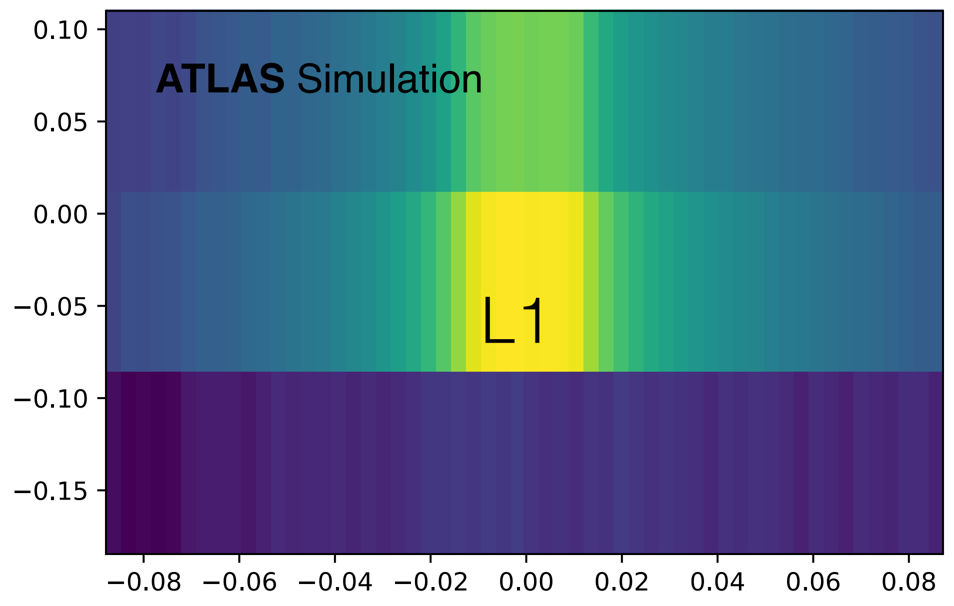




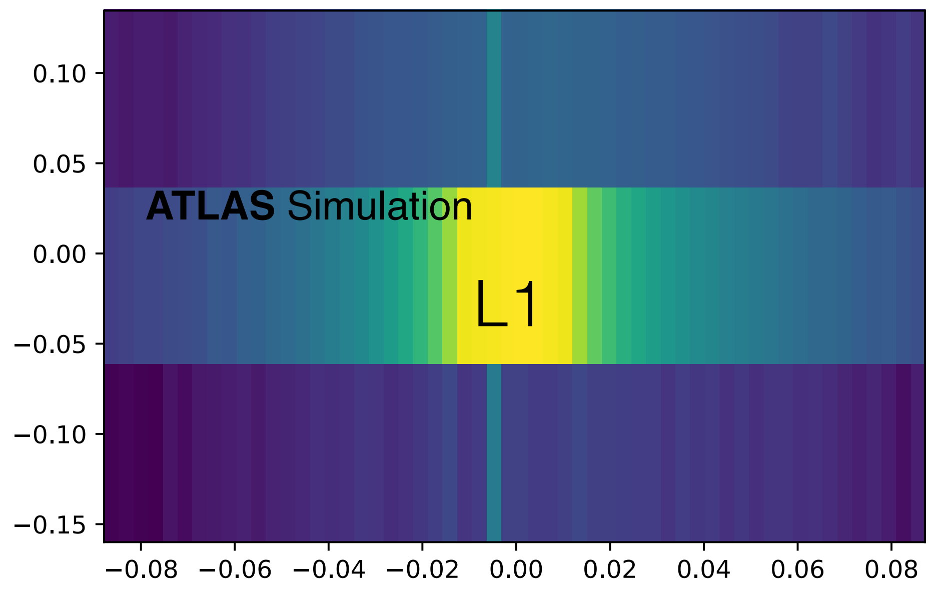
Continuous variable,
not class conditioning

GAN learns to centre the shower
around the particle position within the middle cell

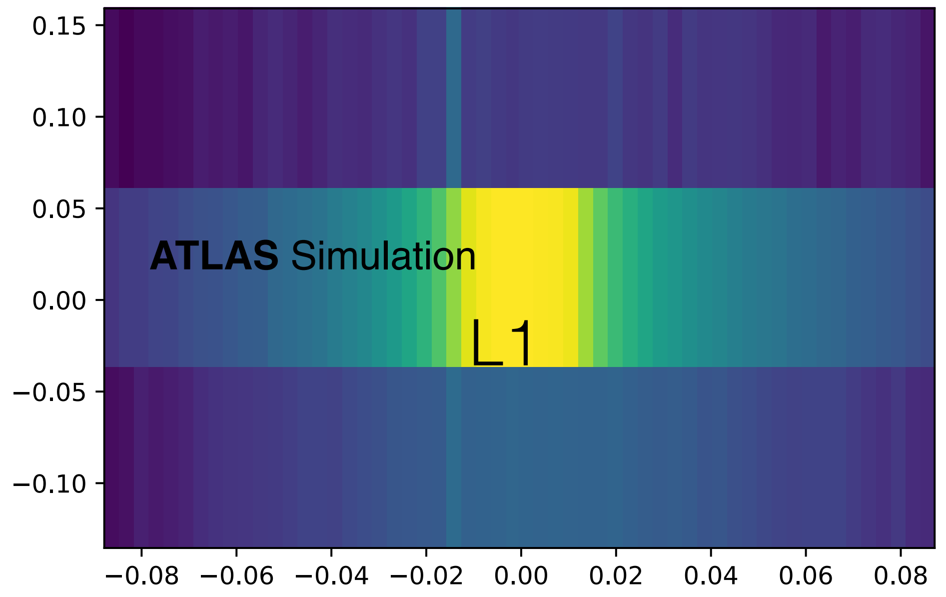
Strip (L1)



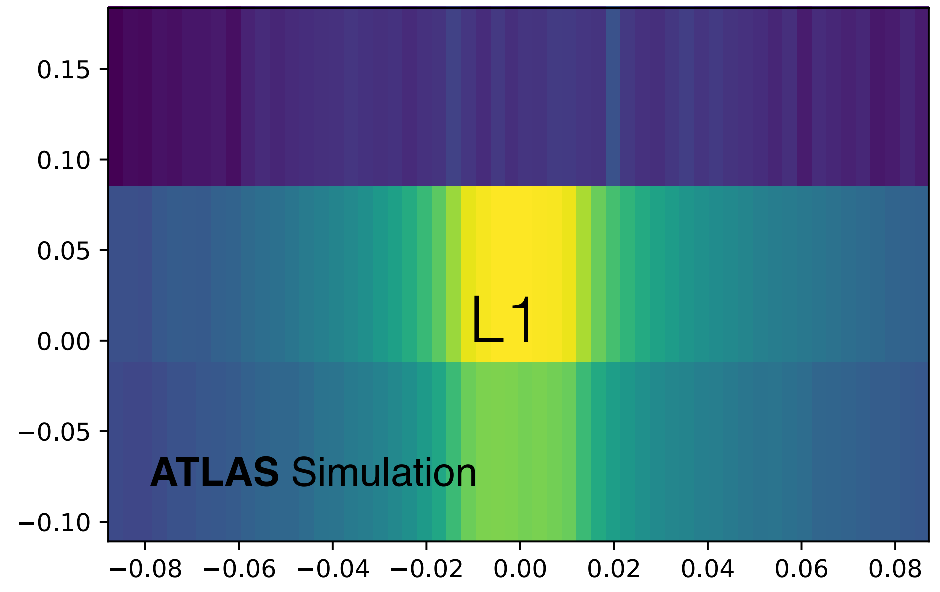
Config 7



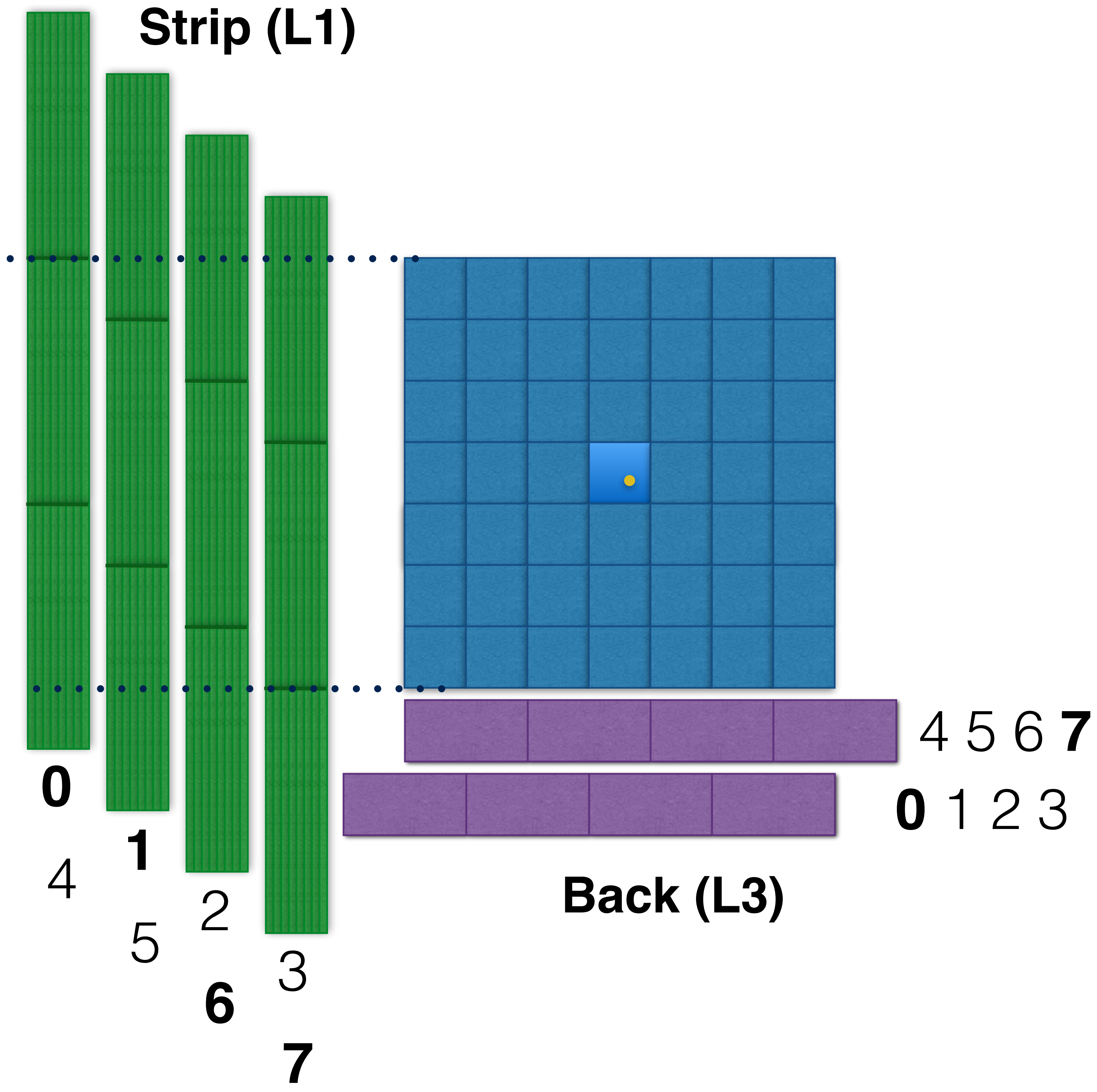
Config 6



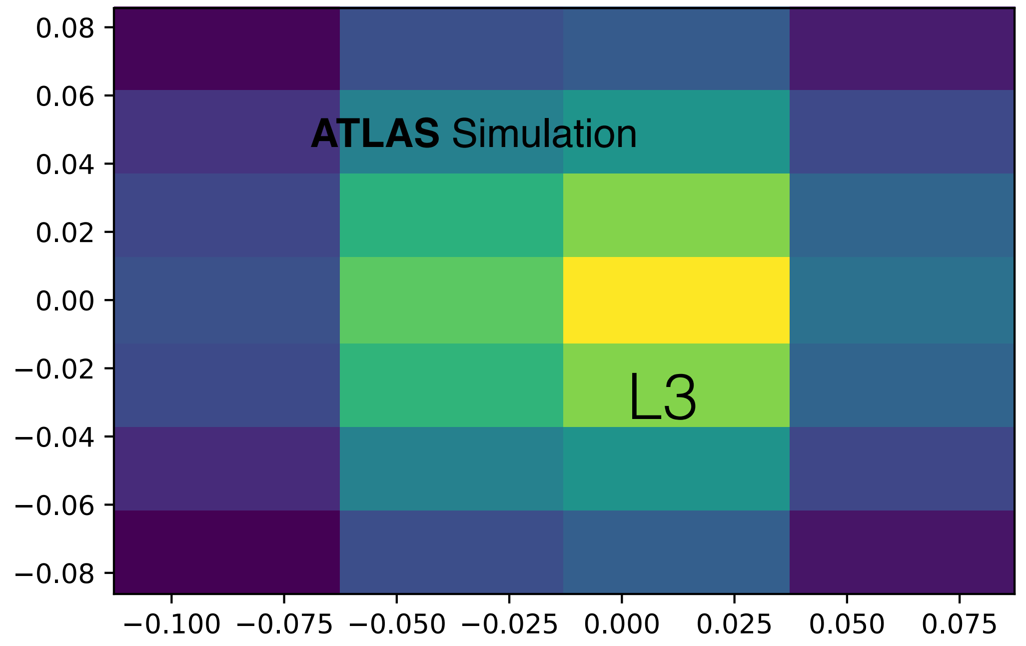
Config 1



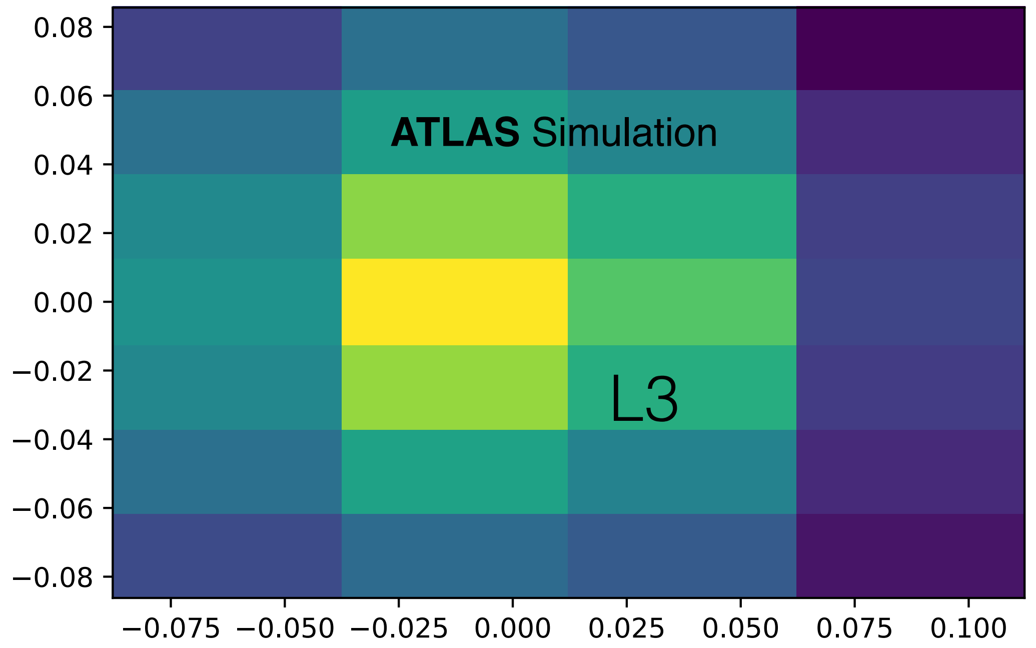
Config 0



Back (L3)



Config 0



Config 7

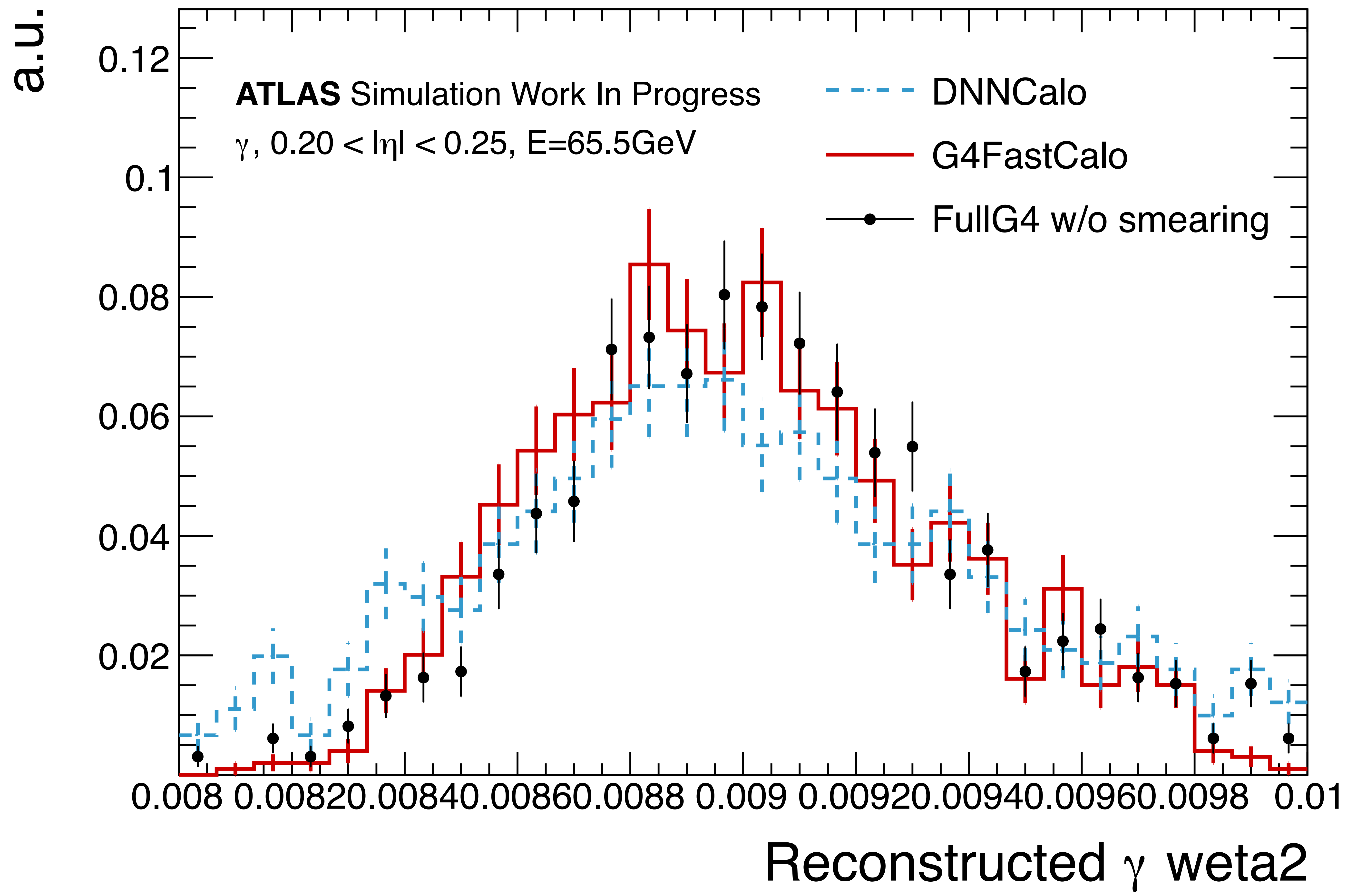
Conditional GAN Algorithm

```python
nb_epochs = 50000; optimizer=Adam(lr=0.00005, beta_1=0.5, beta_2=0.5);
g4Images_train = g4Data_train.images()
cond_info_train = g4Data_train.cond_info()
for epoch in range (nb_epochs):
    g4Images_train, cond_info_train = shuffle(g4Images_train, cond_info_train)
    for bigBatchImg, bigBatchCond in yieldChunk(g4Images_train,cond_info_train,n_images=64*5):
        for real_images, cond_features in yieldChunk(bigBatch, bigBatchCond,n_images=64):
            noise = random.gaussian(latent_size=300, n_images=64)
            fakes_images = generator.predict(noise, cond_features)
            real_images = concatenate(real_images, cond_features, axis=1)
            fakes_images = concatenate(fakes_images, cond_features, axis=1)
            train_set = shuffle(concatenate(real_images, fakes_images, axis=0))
            critic_output = critic(train_set)
            critic_loss = Wasserstein_loss(critic_output) + Grad_Penalty(critic_output)
            critic.backpropagate(critic_loss)
        noise = random.gaussian(latent_size=300, n_images=64)
        if (sampleFromTrainingSet):

            # select 64 cond_features from entire training set of 8800 images
            cond_features = random.choice(cond_info_train, size=64, replace=False)
        else if (reuseWithResample):
            # select 64 cond_features from the 320 bigBatchCond at random
            cond_features = random.choice(bigBatchCond, size=64, replace=False)

        else:
            # use cond_features from the last critic iteration
            pass
        fake_images = generator.predict(noise, cond_features)
        critic_output = critic(fake_images, cond_features)
        generator_loss = - Wasserstein_loss(critic_output)
        generator.backpropagate(generator_loss)
```

Width in Eta for Middle Layer

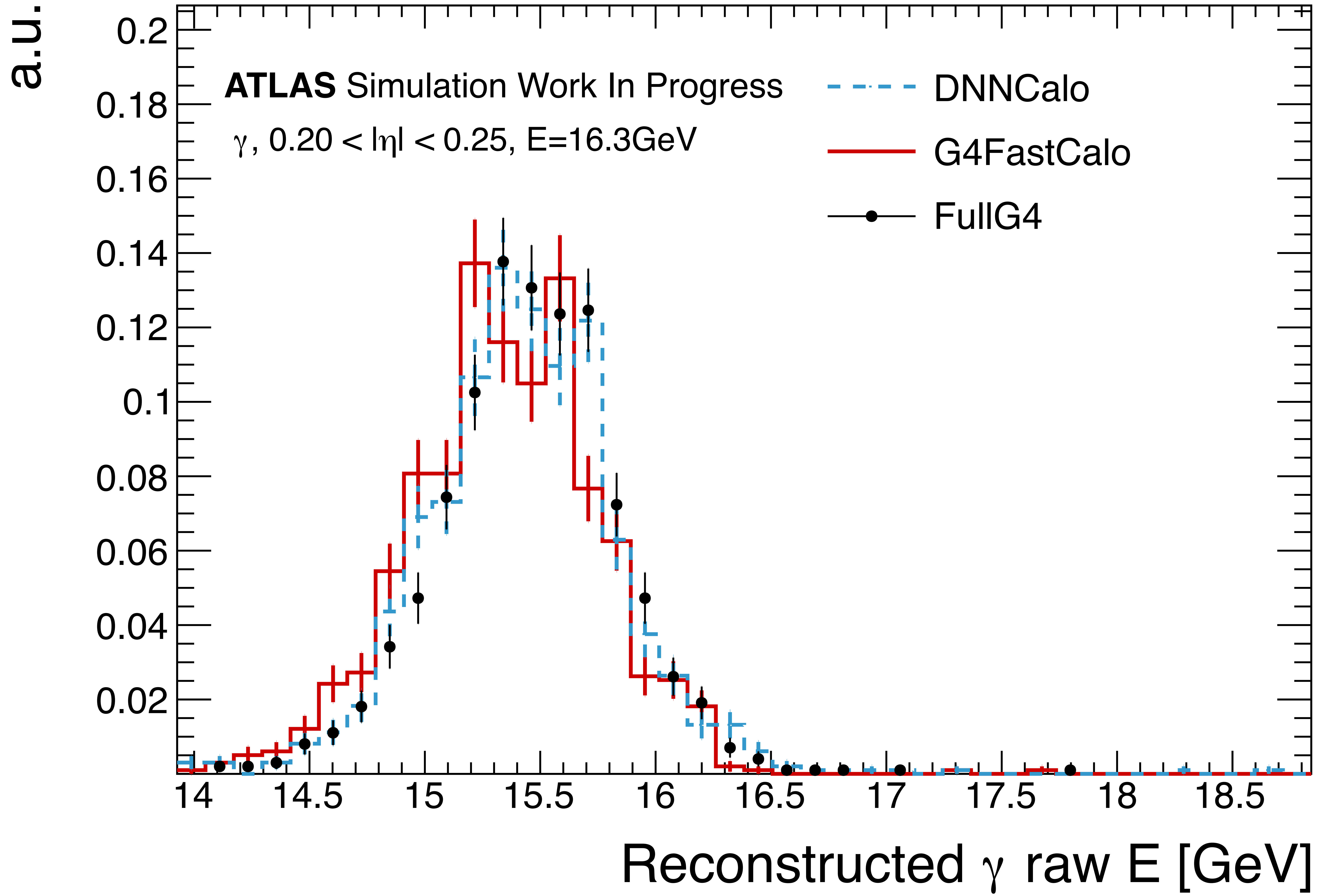


WEta Middle (in η units)

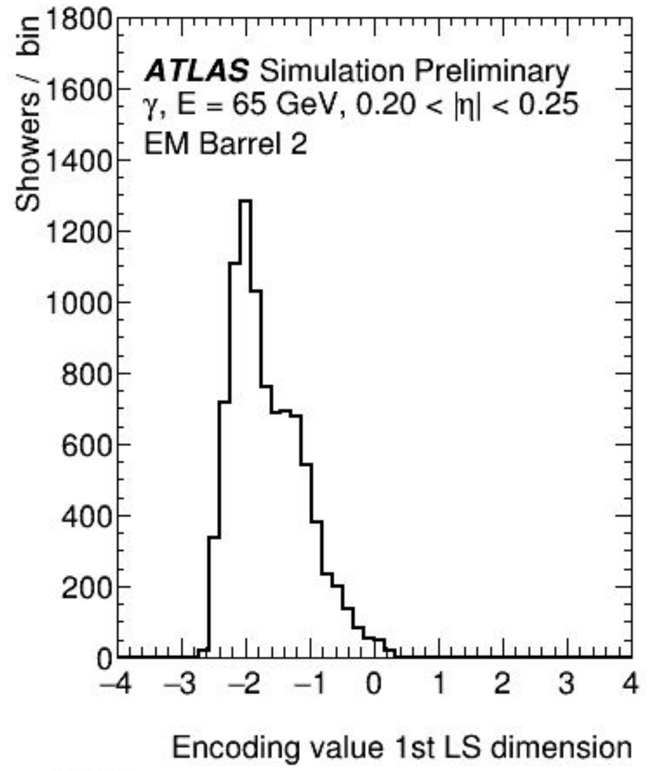
Test on Untrained Energy Point: 25 GeV

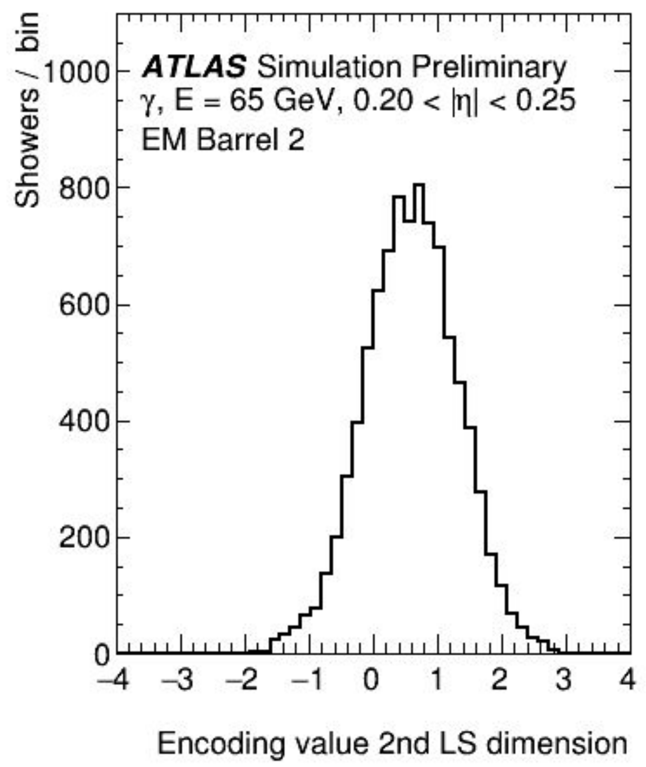
Remember, GAN trained on 9 discrete energy points:{1,2,4,8,16, 32, 65, 131, 262} GeV

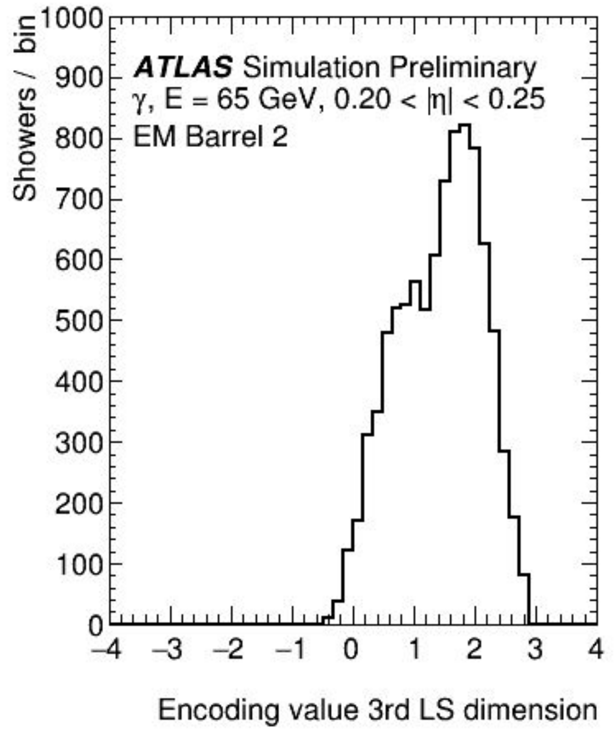
16 GeV Sample: Total E

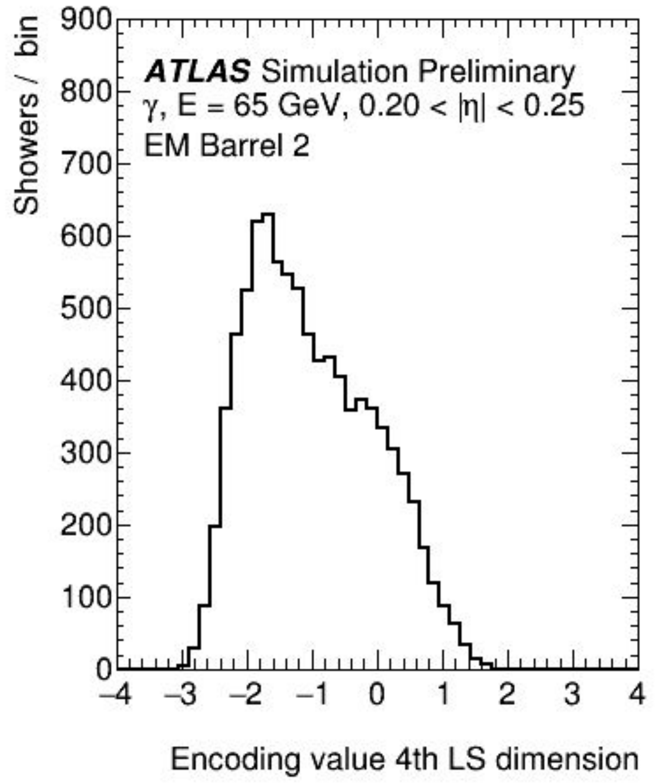


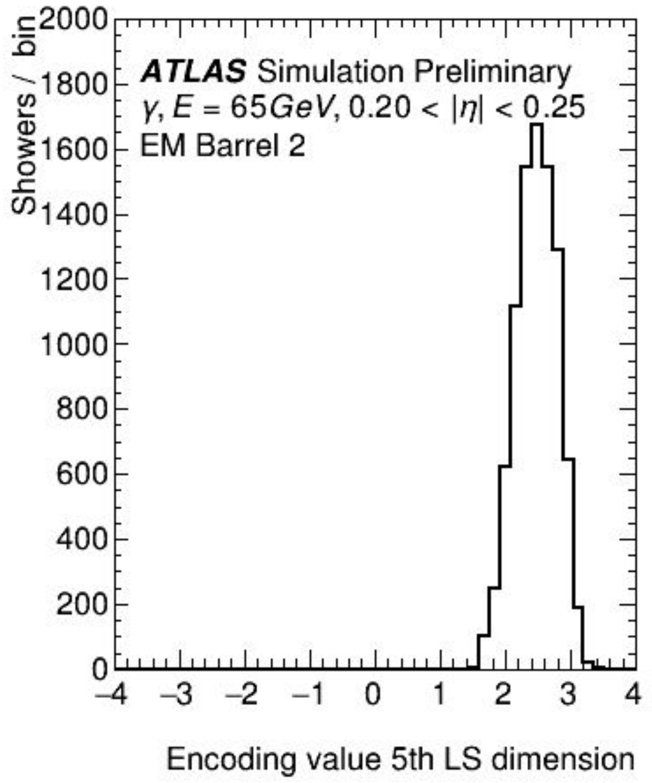
VAE Latent Space











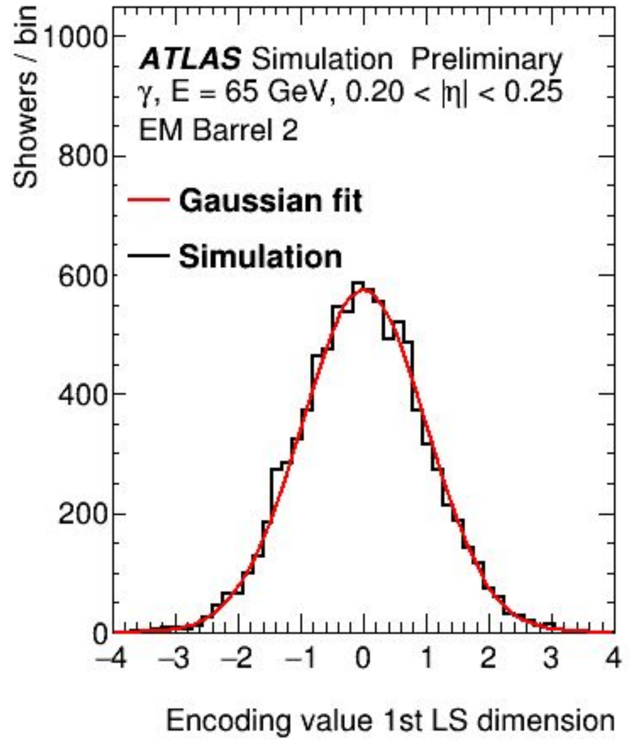
- Input : a variable with some specified ordering (multidimensional tensor)
- Output : (μ,σ) for each element of the input variable conditioned on the previous elements.

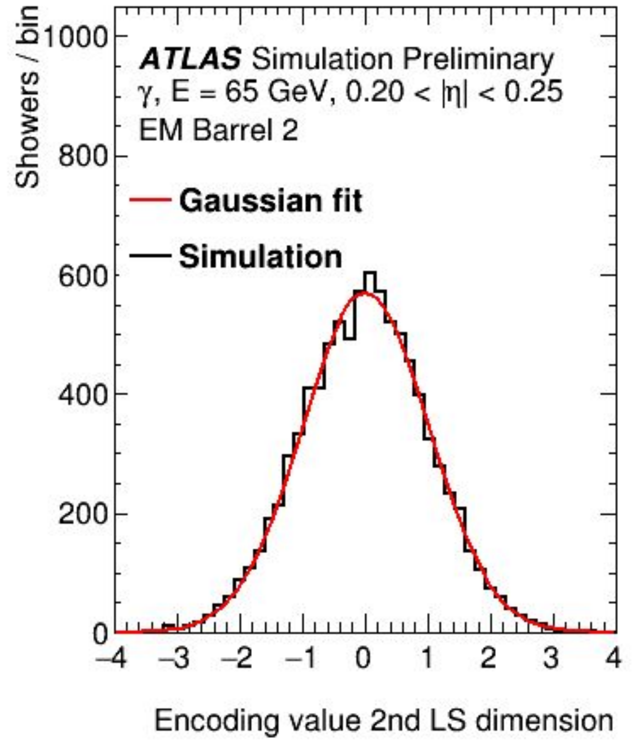
5D Latent Space don't look Gaussian

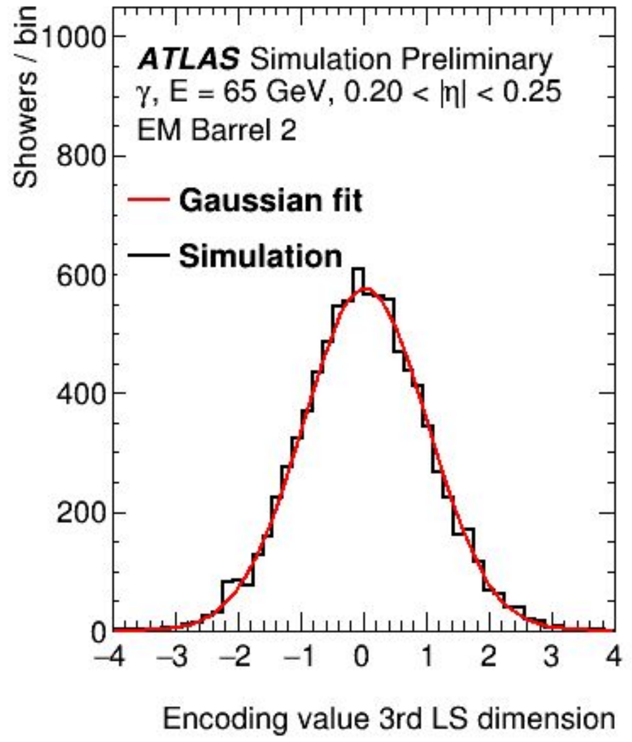
Inverse Autoregressive transformations

a type of Normalizing Flow to make the latent space more Gaussian

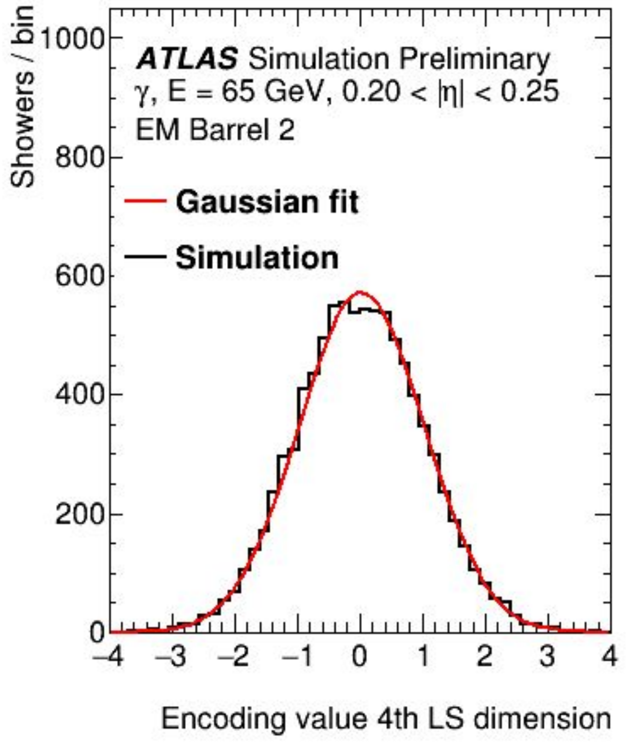
IAF transformations make the latent space distributions more Gaussian like.

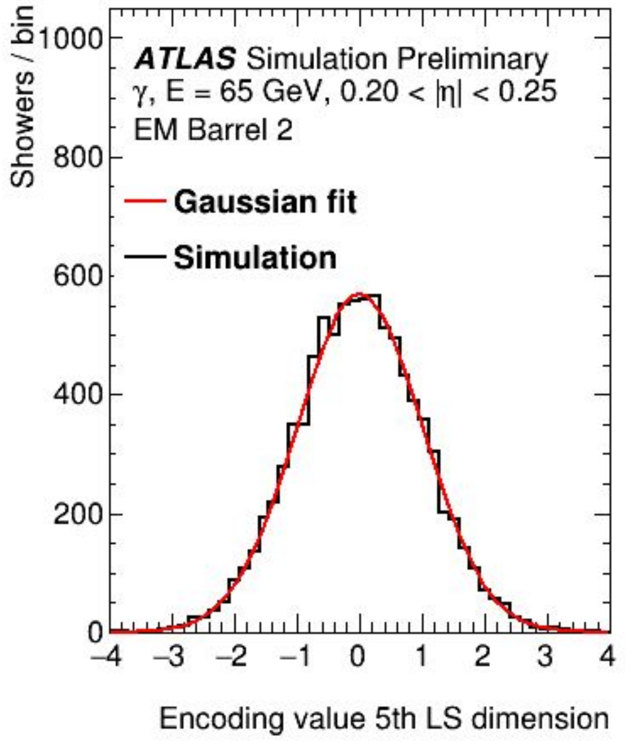




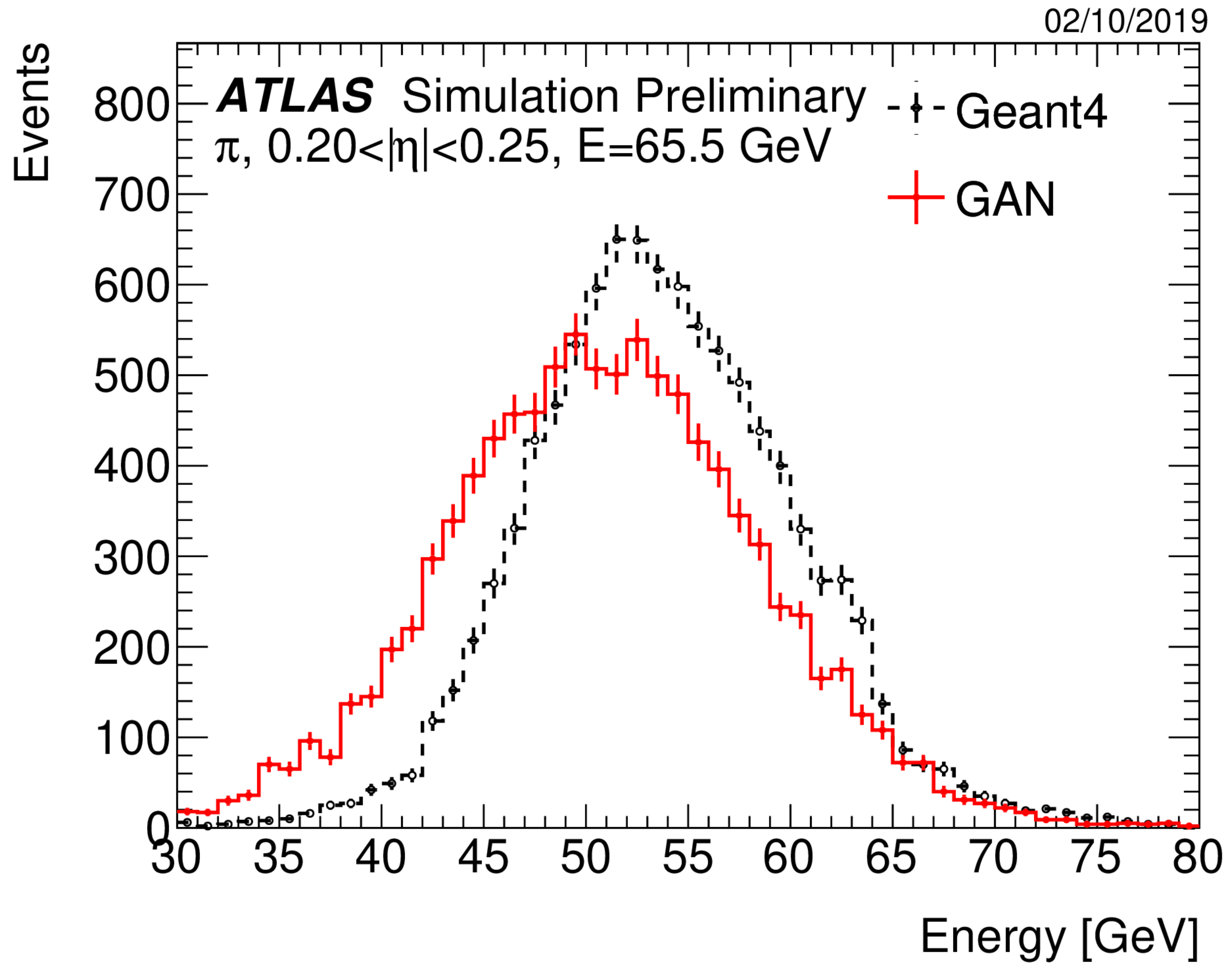


When we use the Decoder as a generator, it will be more correct to sample from a Gaussian distribution, impact on physics under study





GAN on Voxels (Edinbergh team)



Trained at single energy point and works
When moving to conditioned GAN, old setup doesn't work, need to re-start from scratch !

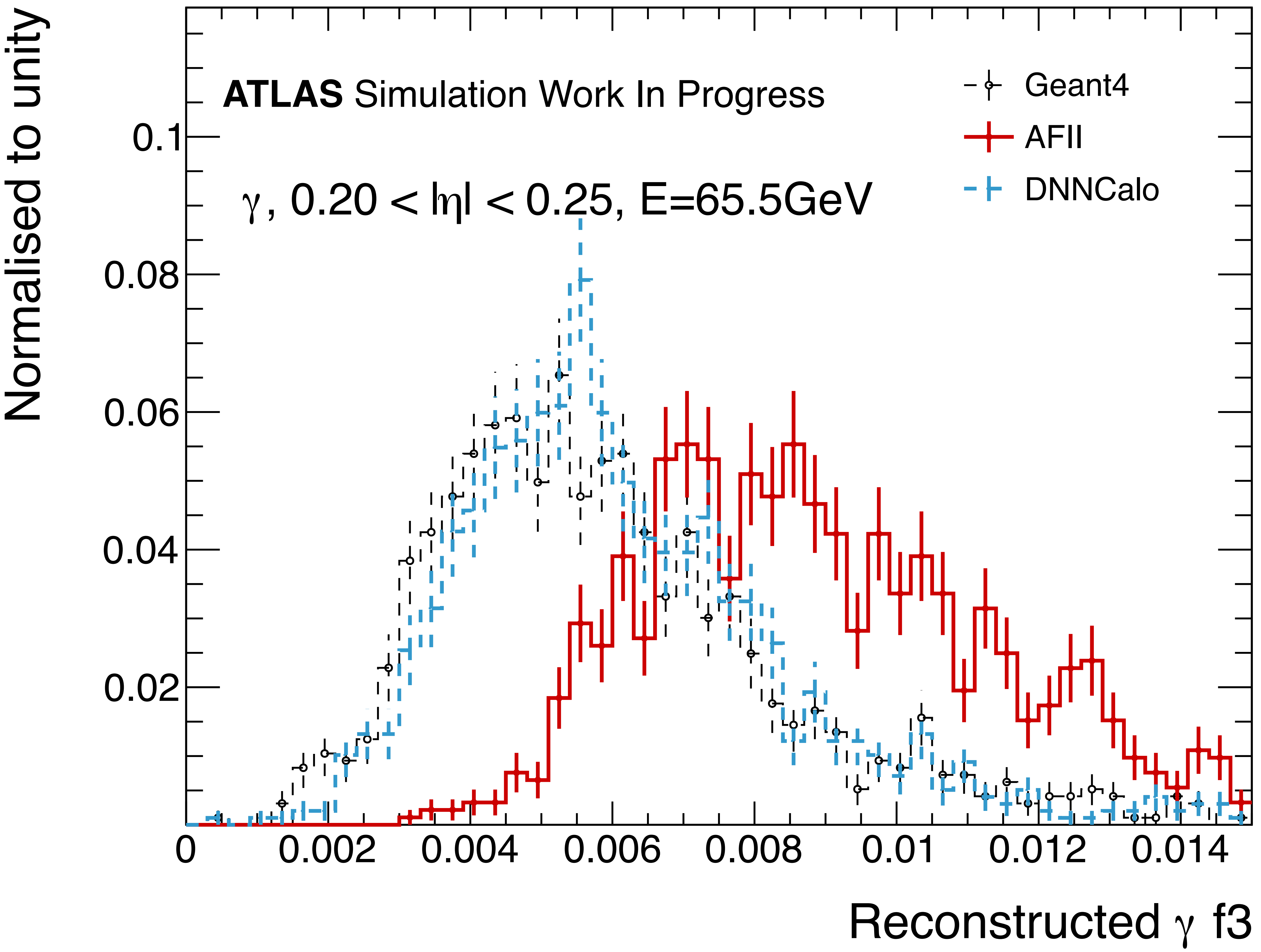
Voxelisation was tuned a lot to get good results
Possible that it was overturned for this energy, eta point

How do we compare with the current AFII (FCS V1)?

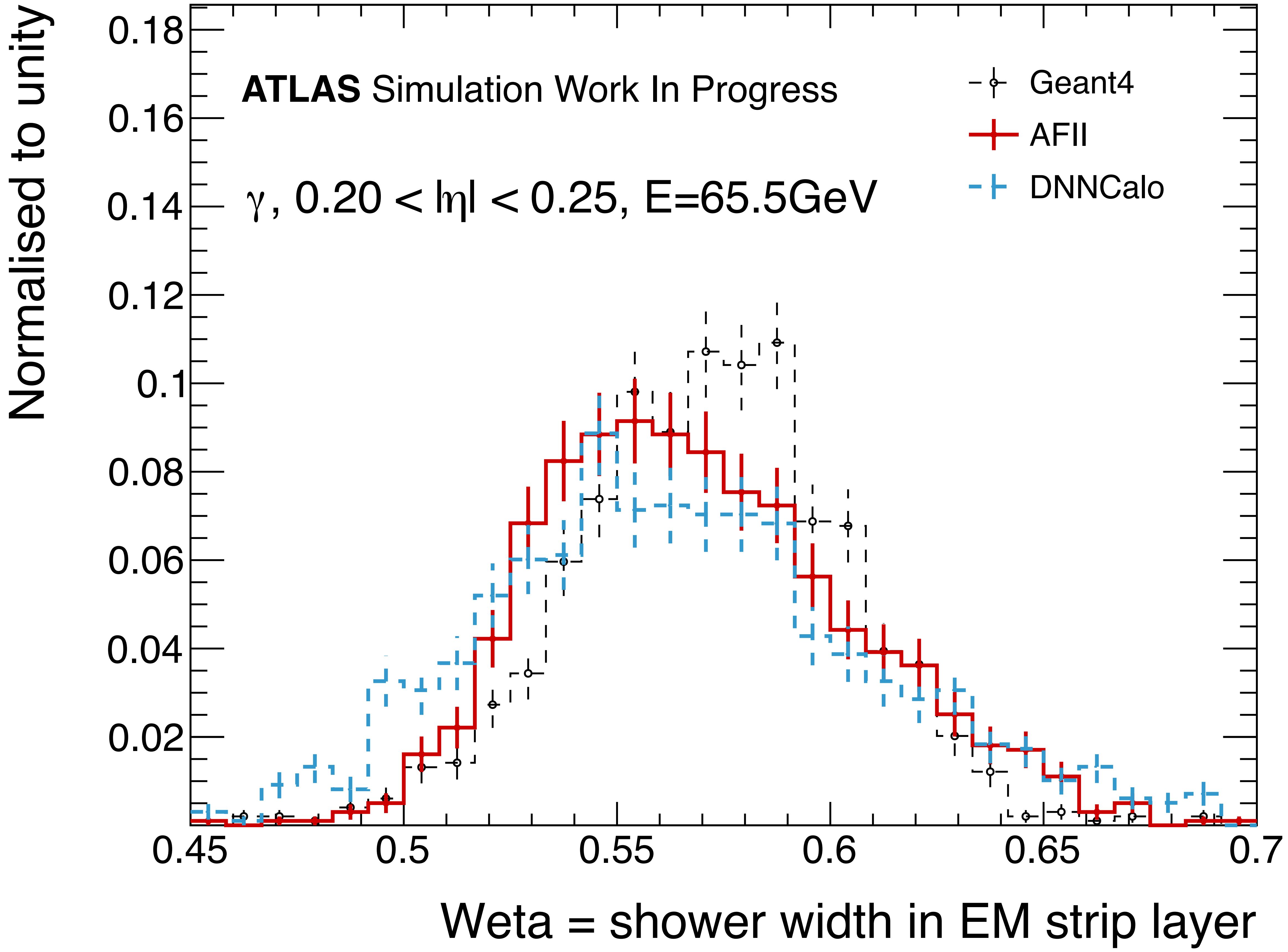
This is our first look at AFII
(includes data tuning)

G4 vs **AF2** vs DNNCalo

Fraction of Energy in Back



WEta1

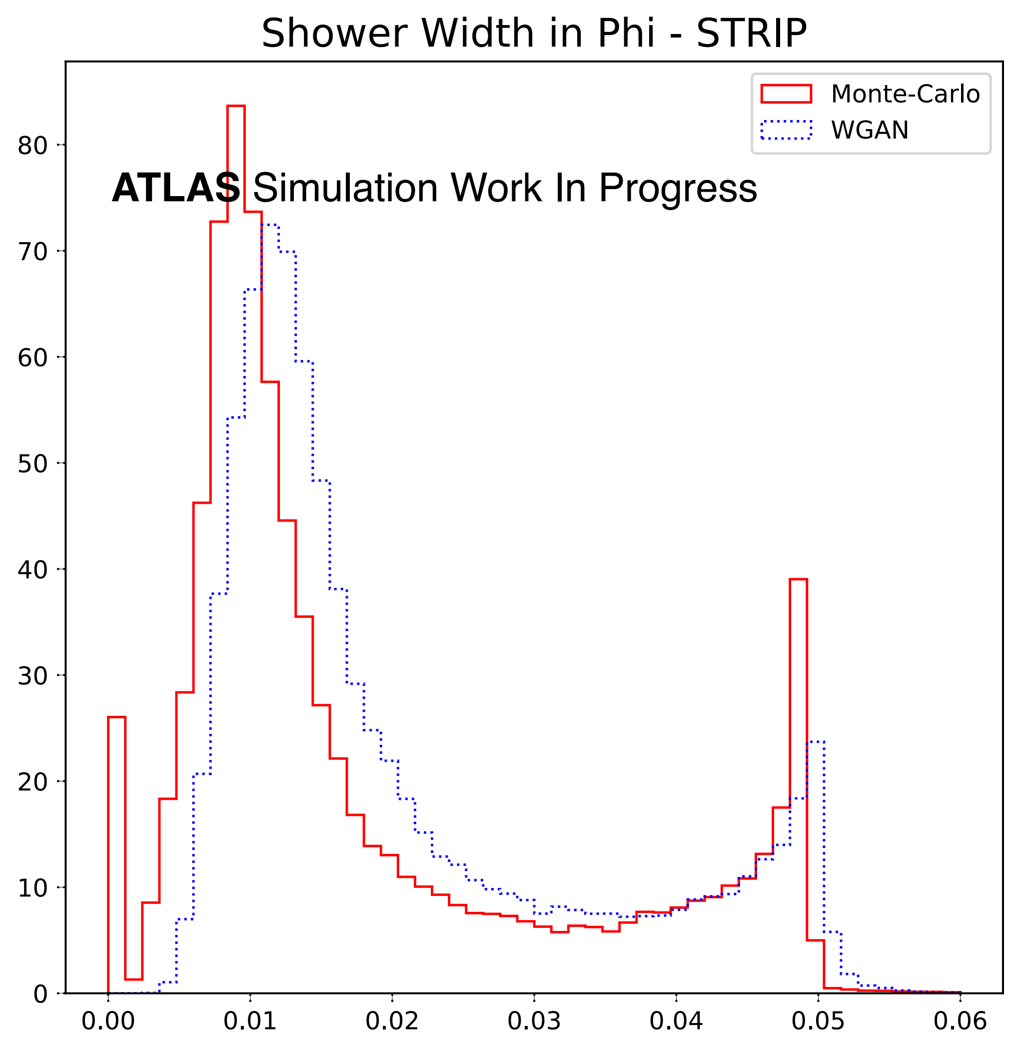


Much better than AF2 in the back, not as good at Strip Width

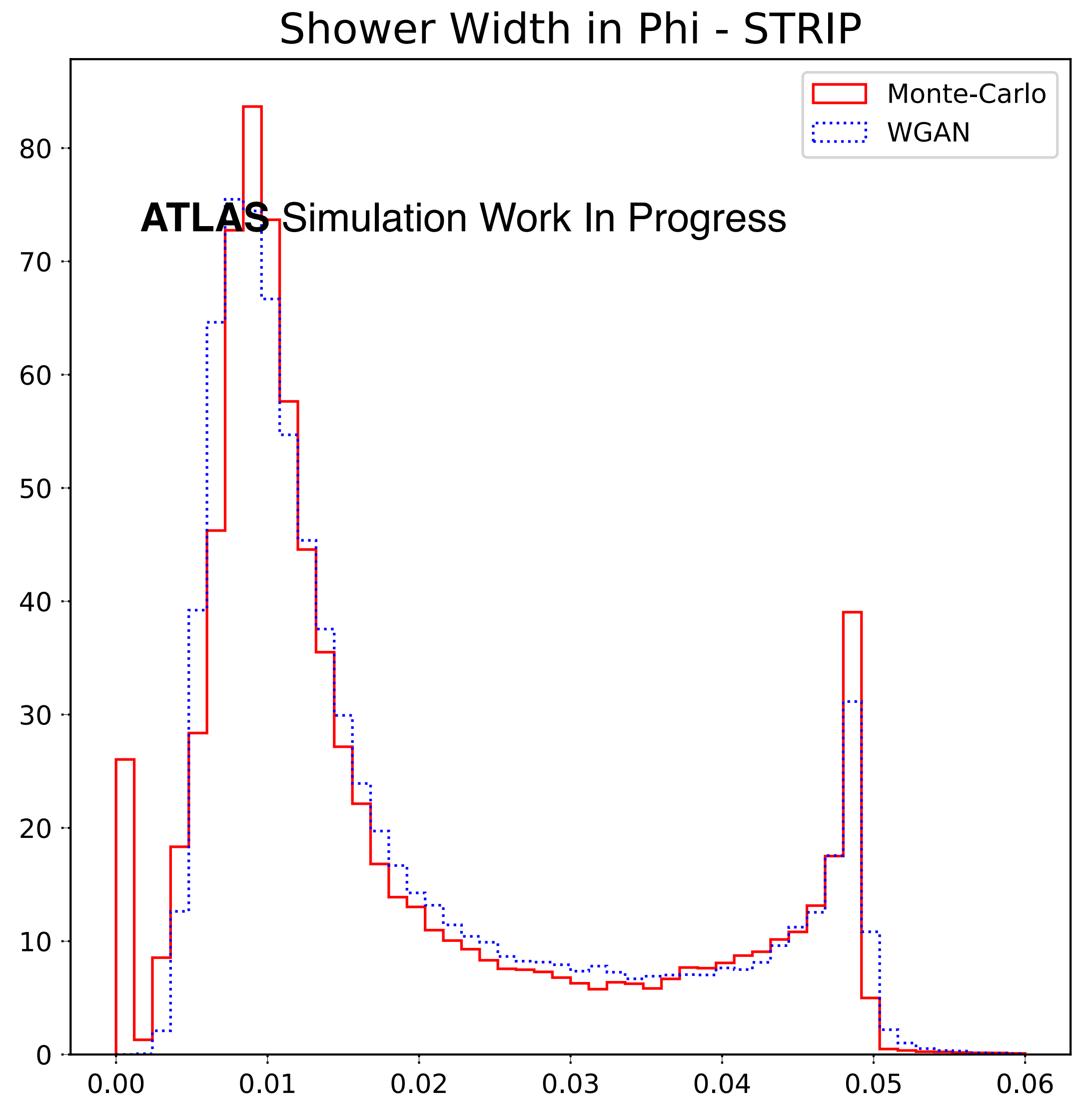
Keras Version

Inexplicable improvement in results and convergence by upgrading from [Keras 2.0.8 with TF-GPU 1.3.0] to [Keras 2.1.5 with TF-GPU 1.4.1]

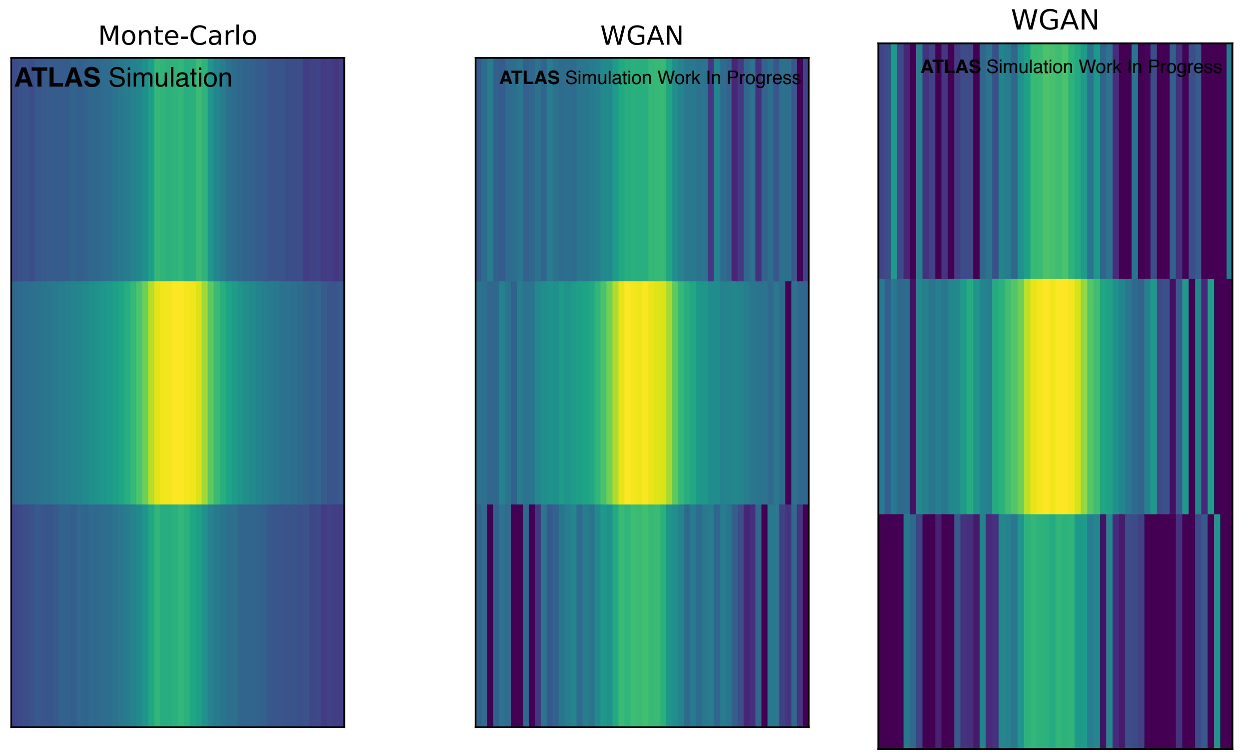
- No hints from release notes
- Same improvement also seen in [Keras 2.1.2 TF 1.4.1] the CPU version



Old Keras +TF



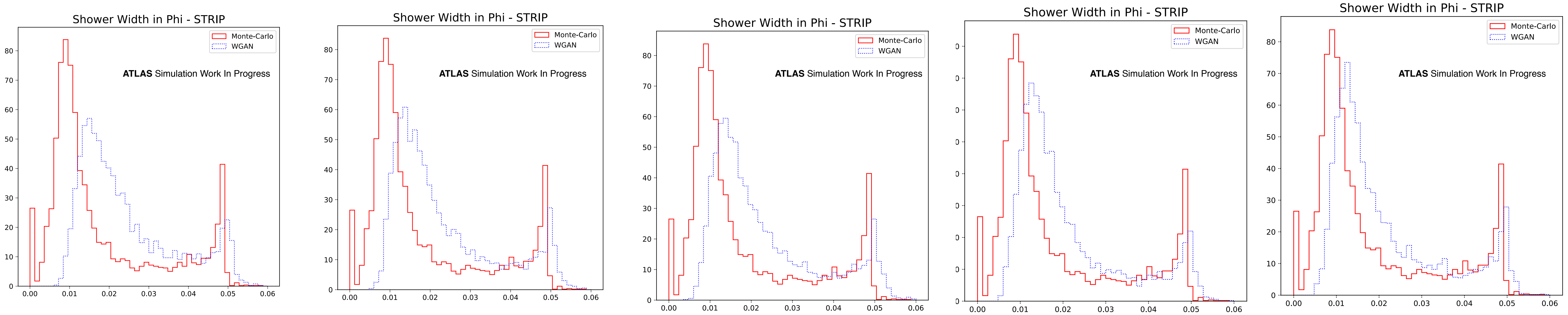
New Keras +TF



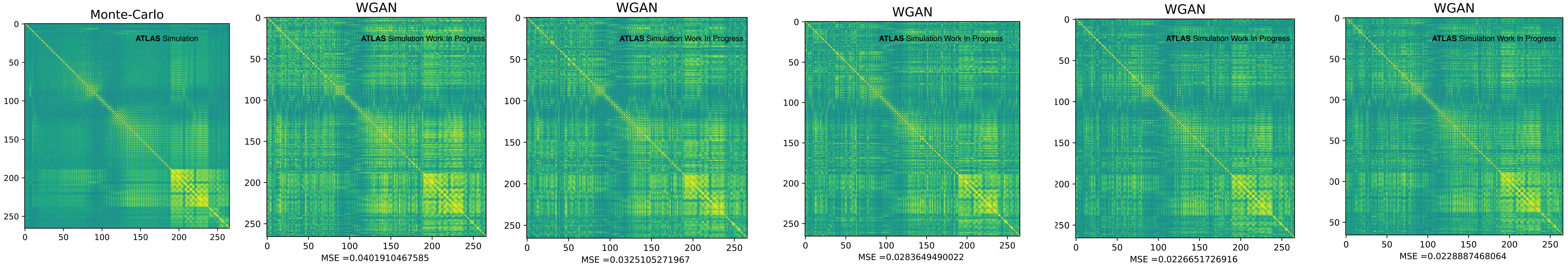
G4 Old Keras New Keras

Changed after new Keras

Performance with Epoch

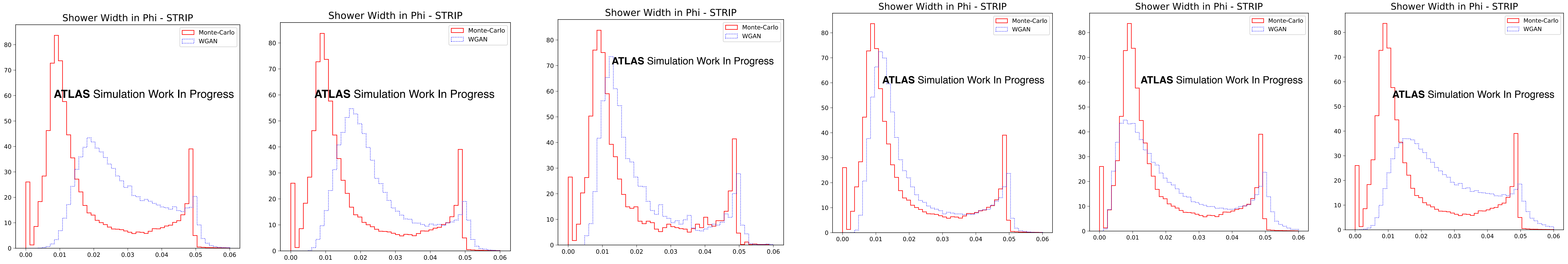


G4 10k 20k 30k 45k 50k



Improved after dropping momentum, fewer epochs for larger training size

Performance with Training Set Size



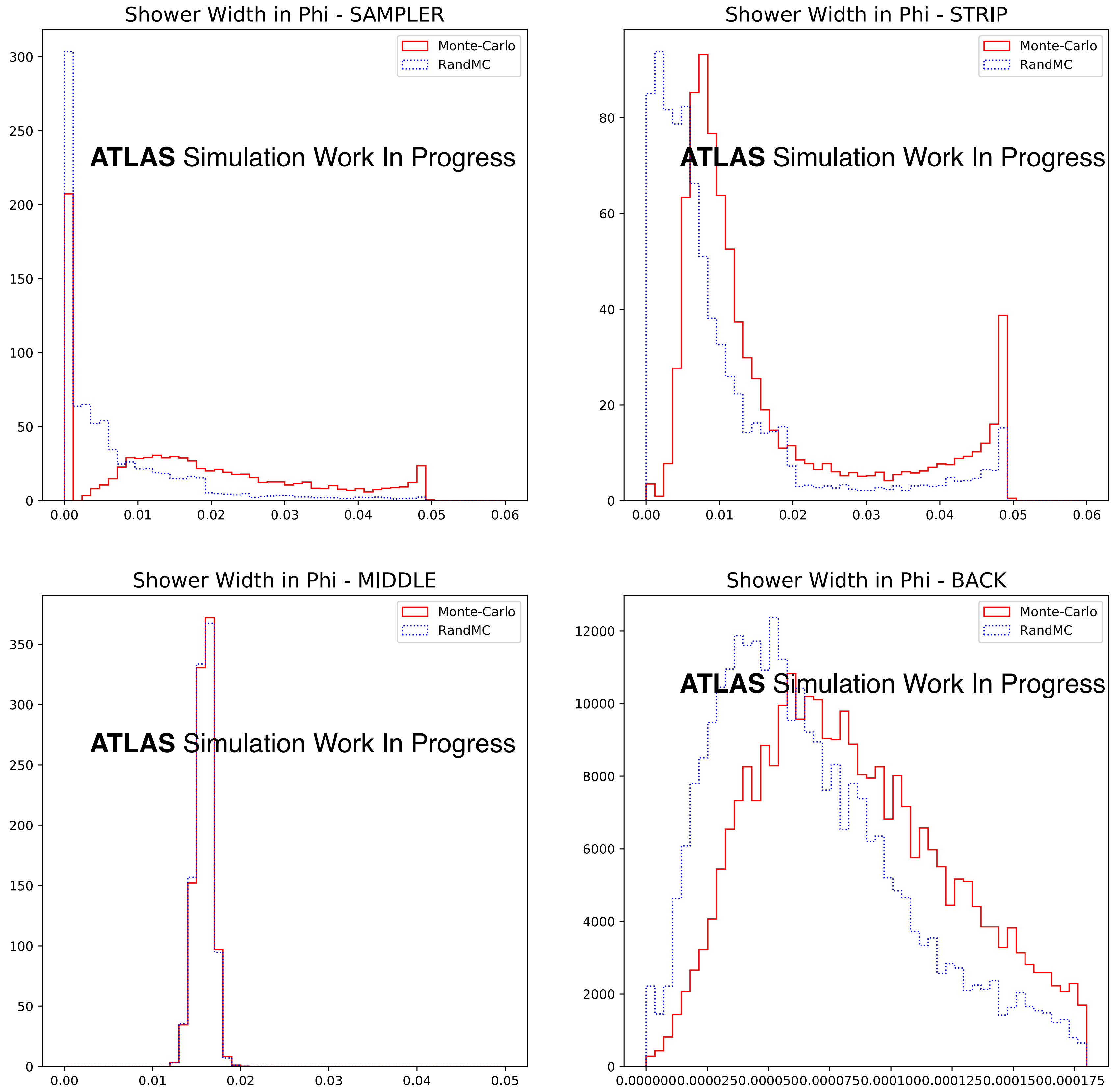
1% 2% 4% 10% 20% 50%

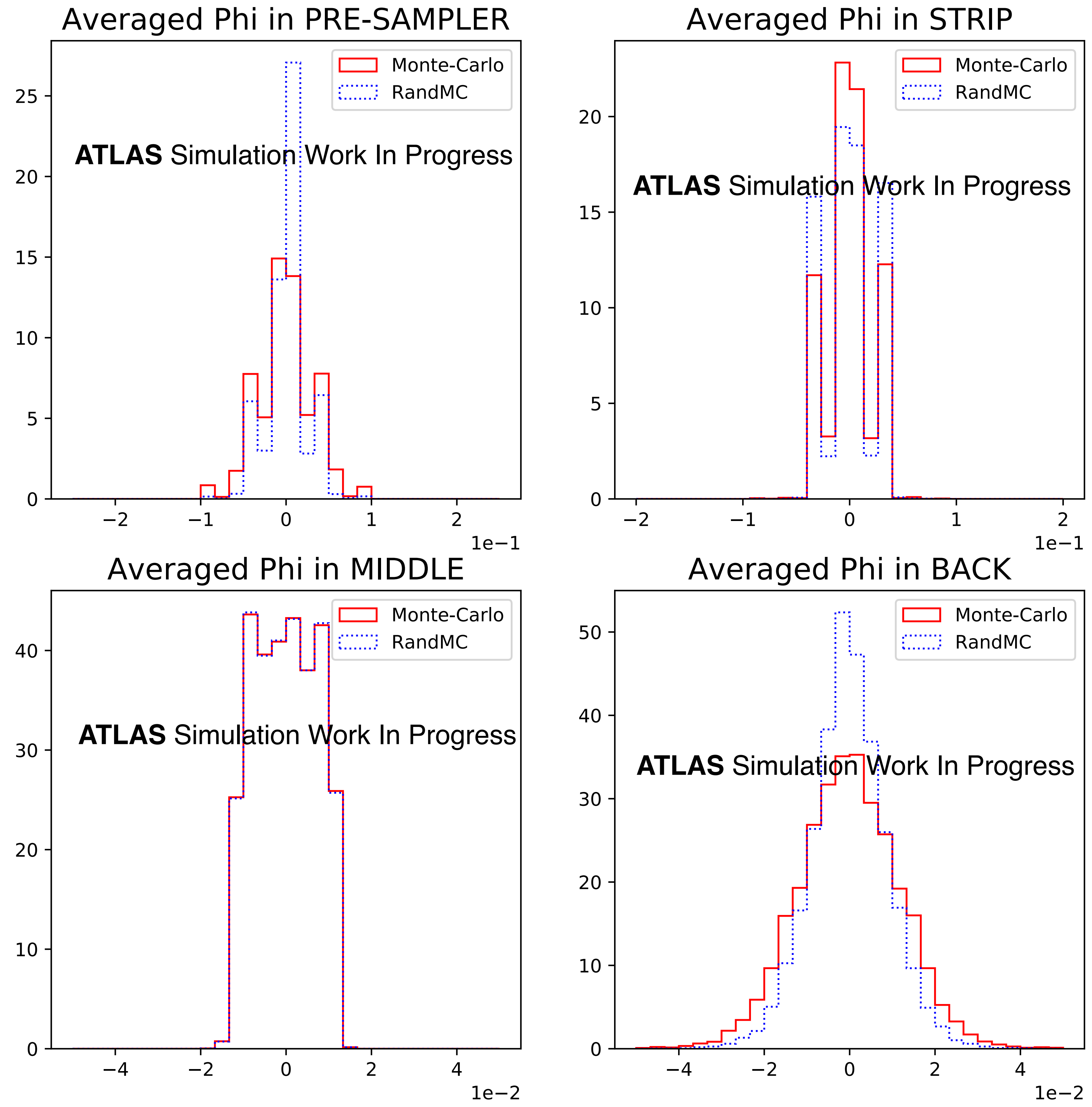
We are not limited by number of training events,
a more representative dataset however would help

MC vs MC

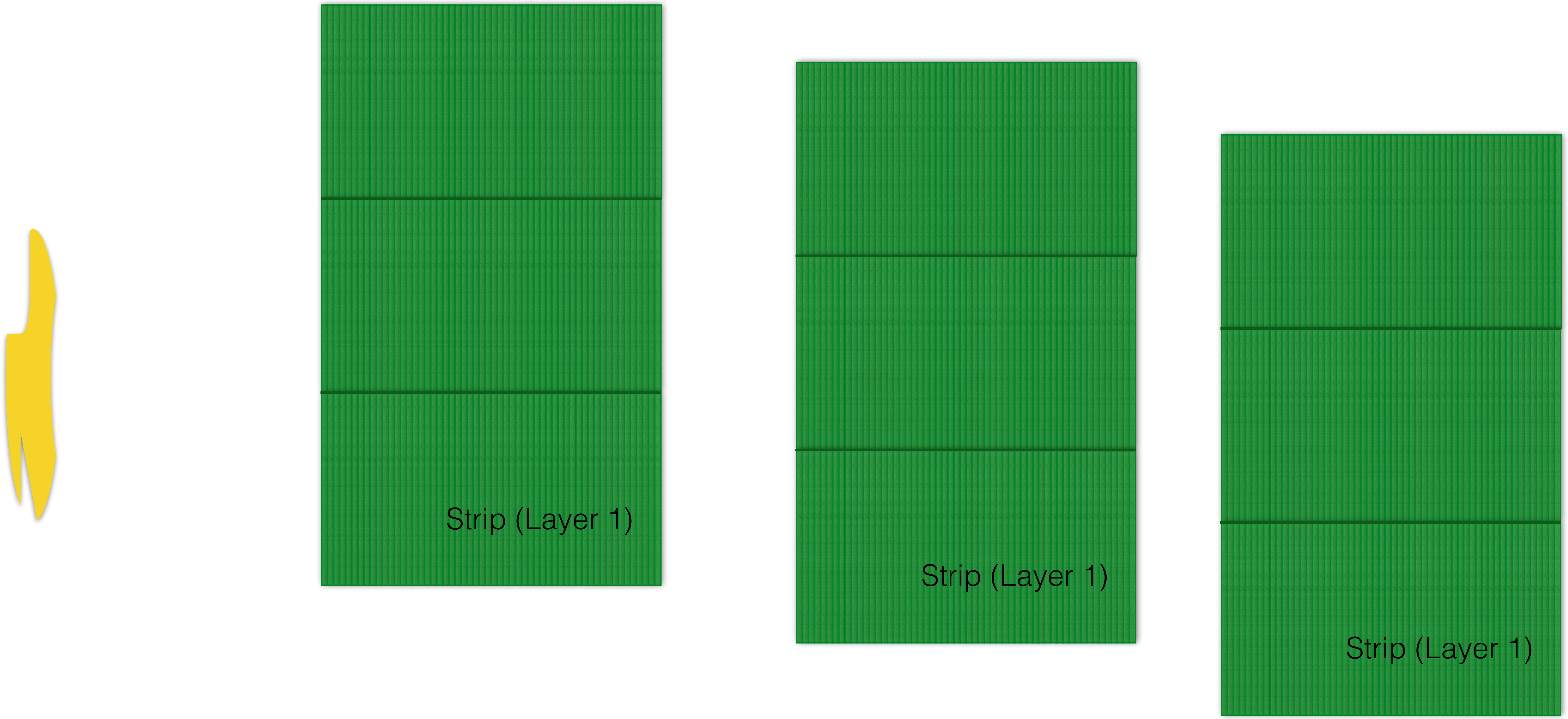
Phi from same Event vs Phi from another random event

Blue here is also Geant4 data (not GAN!)

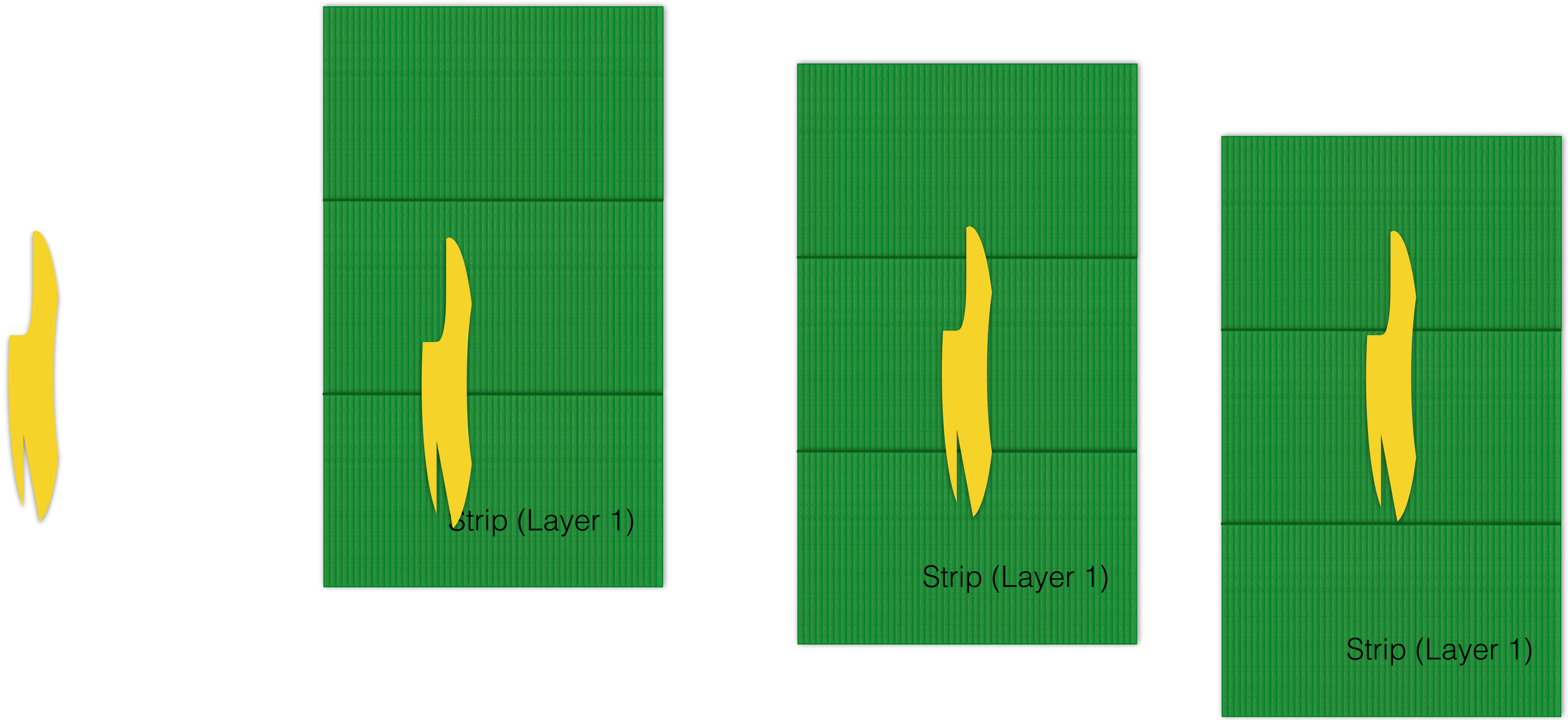




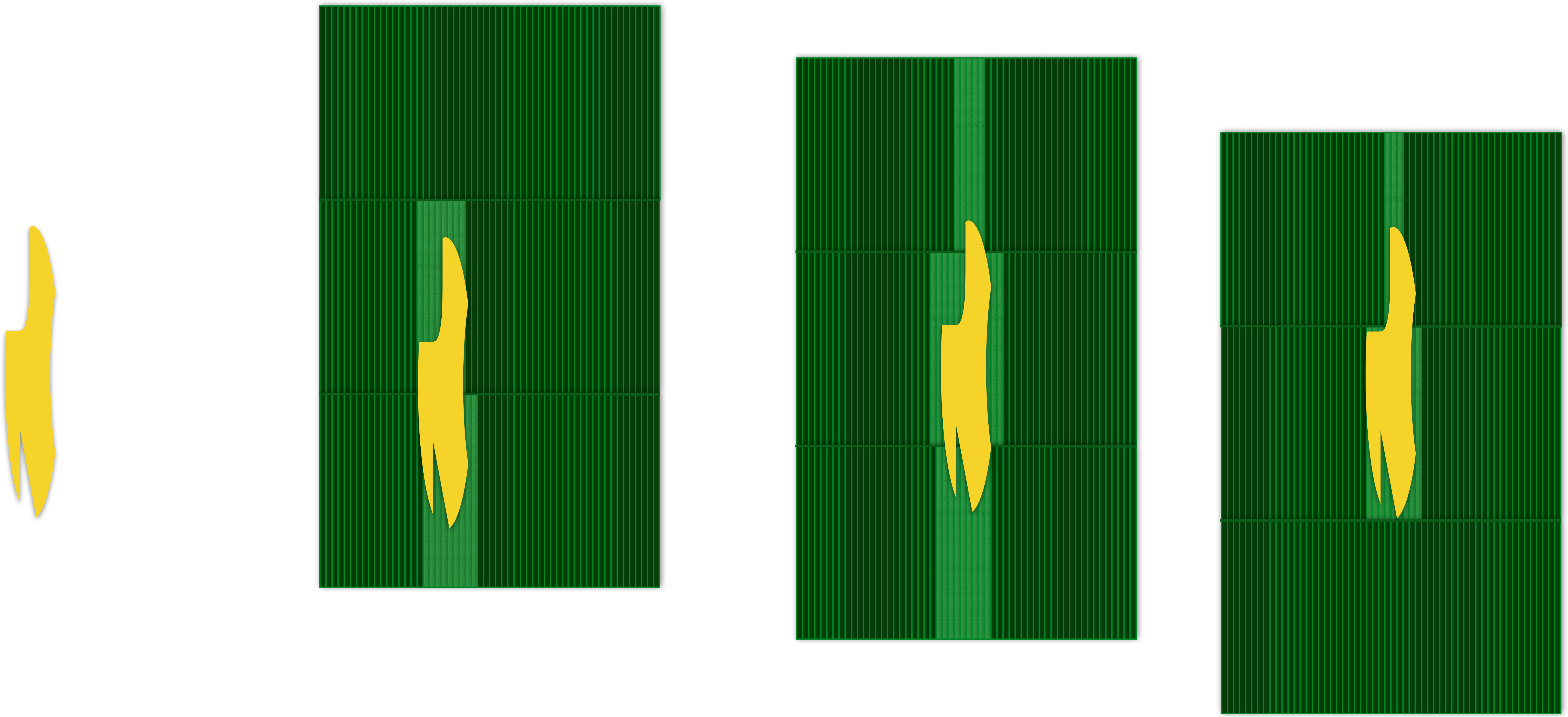
Distributions for Middle Layer are almost perfect



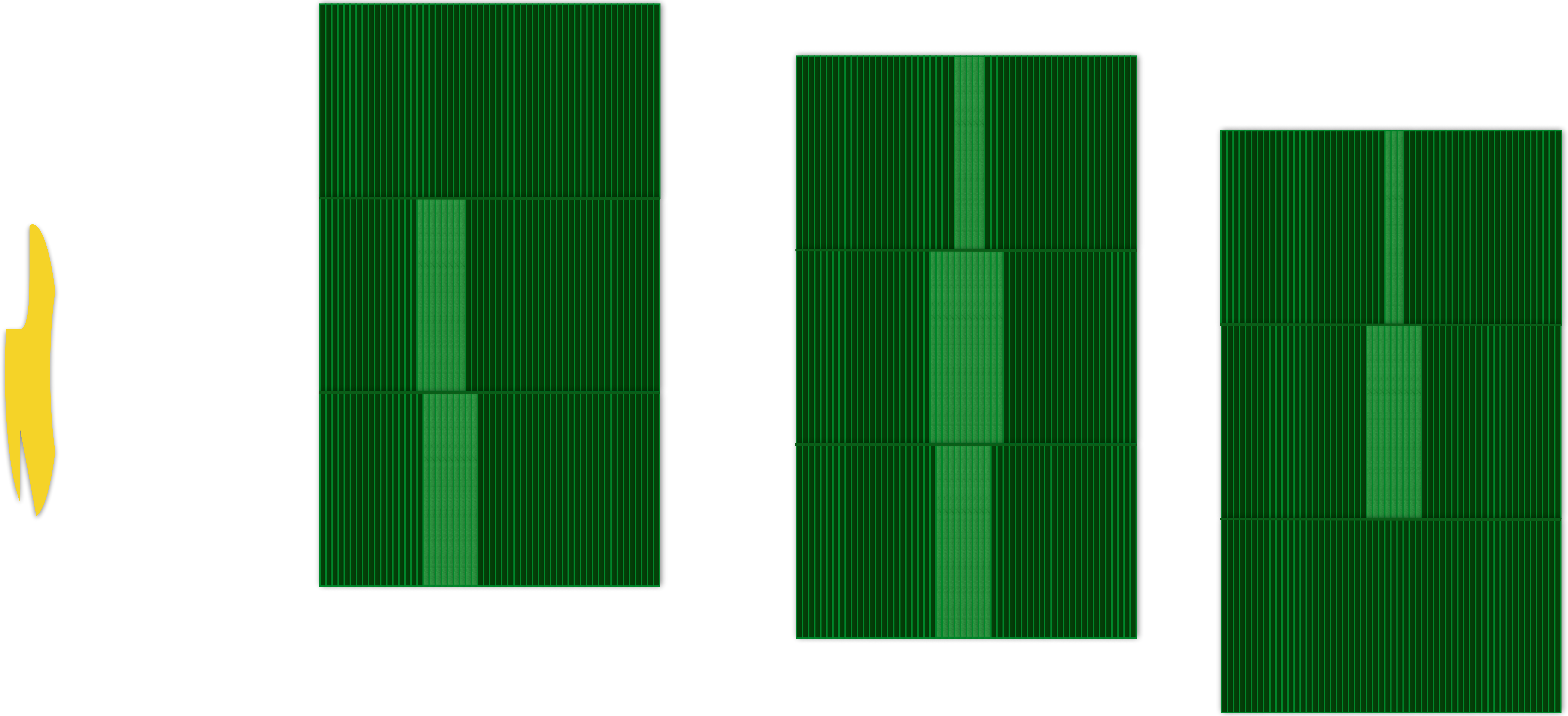
Same shower pattern, different image!



Same shower pattern, different image!



Same shower pattern, different image!



Same shower pattern, different image!