
Simulation R&D

Fastsim Updates

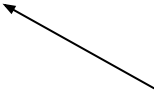
Piyush Raikwar, Peter McKeown

03.09.24

Datasets

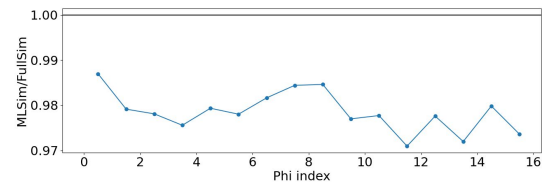
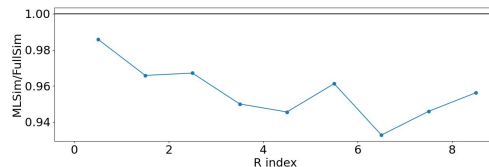
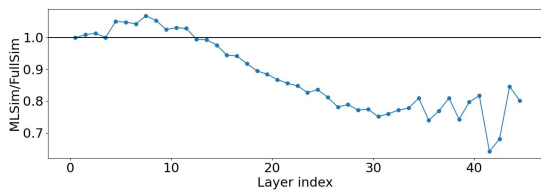
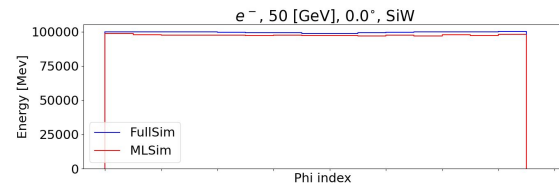
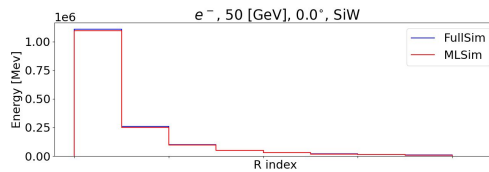
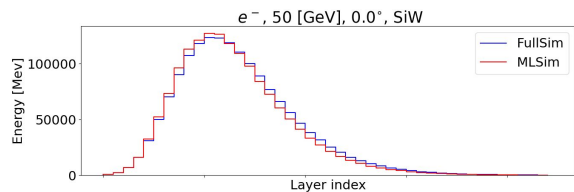
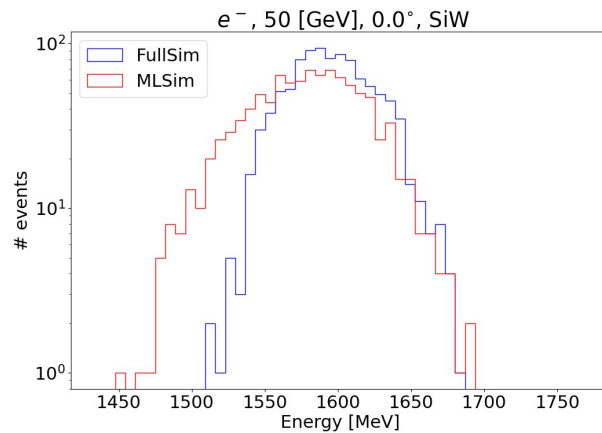
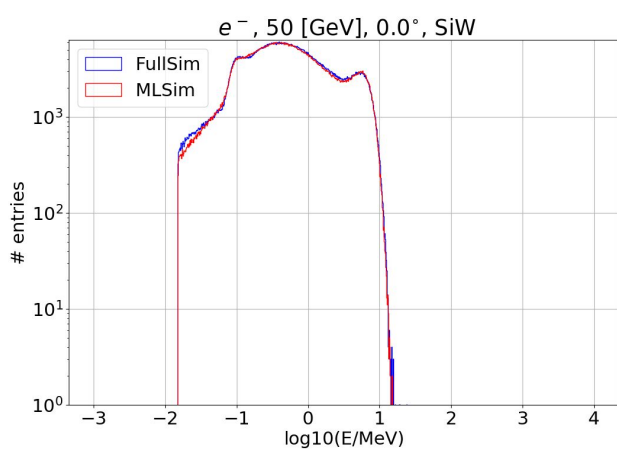
- All the datasets are ready
 - Par04 SiW
 - Par04 SciPb
 - Par04 PbWO4
 - ODD
 - FCCee CLD (1GeV - 100 GeV)
 - FCCee ALLEGRO (1GeV - 100 GeV)
- Some events are discarded in FCCee detectors if they create secondary particles before reaching the calorimeter (same conditions are being replicated when doing the validation in DDG4)
 - 0.1% in ALLEGRO
 - 10% in CLD
- Datasets are unscaled
 - Initially sampling fraction was included while creating the dataset
 - During training on multiple geometries, either the model learns how various calorimeters differ or we somehow take this into account to bring the voxel energies to same scale across geometries

To be addressed for future studies

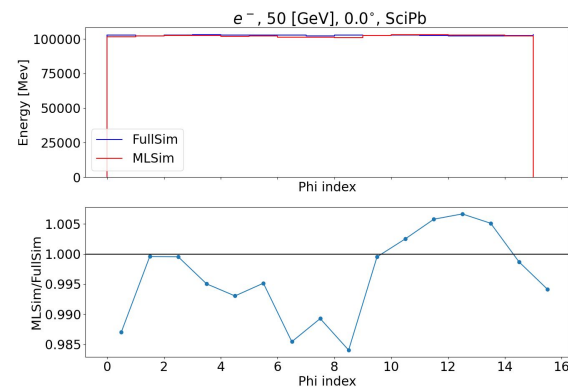
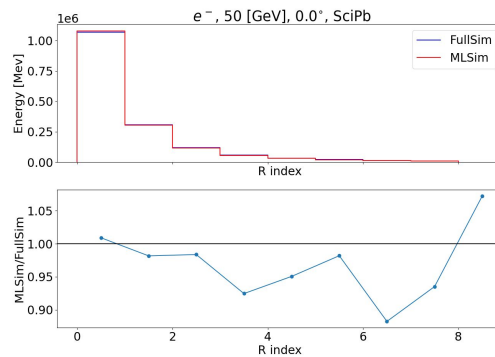
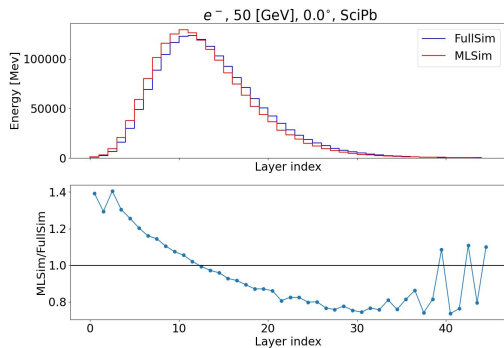
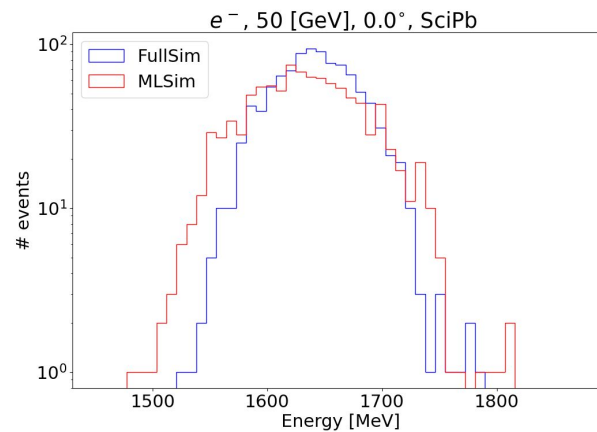
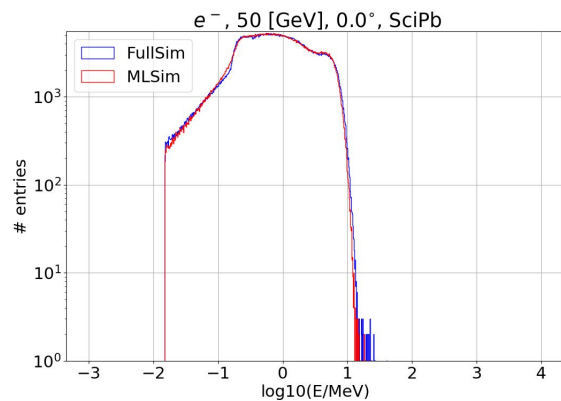


Single geometry training using DDPM

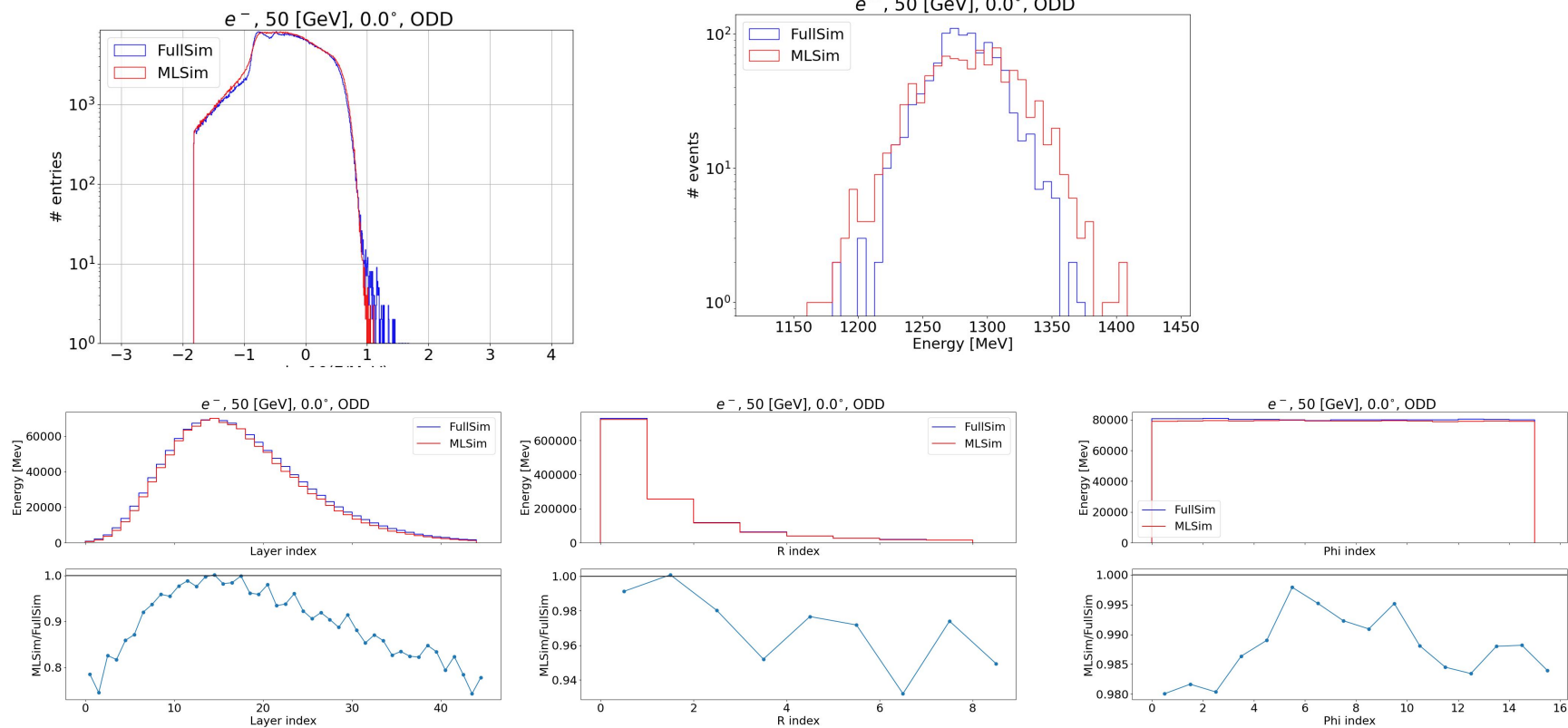
Paro4 SiW



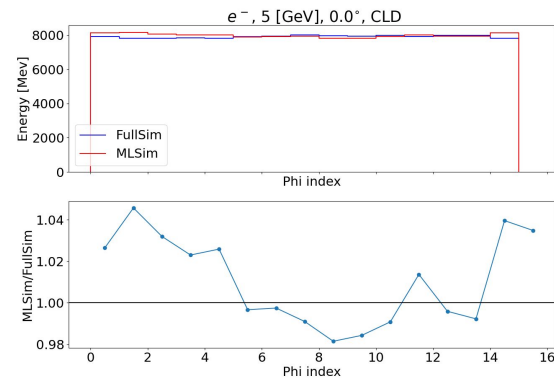
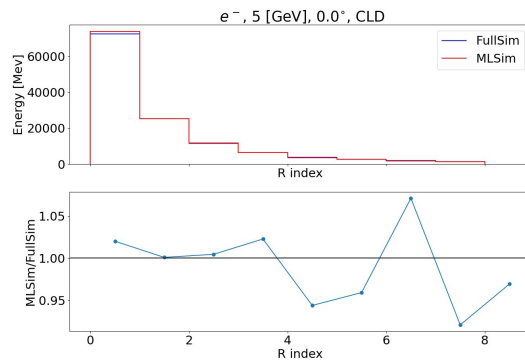
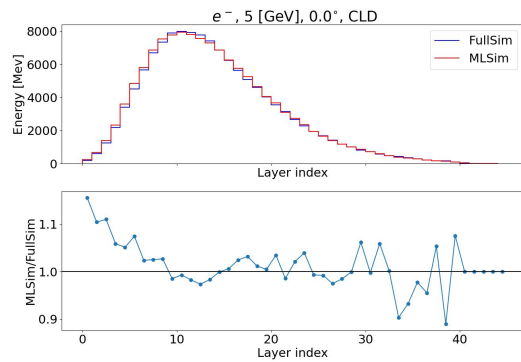
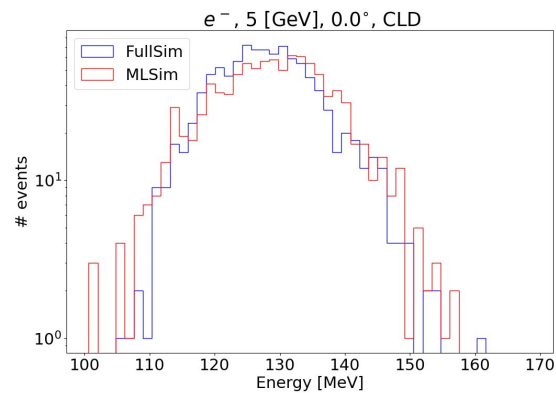
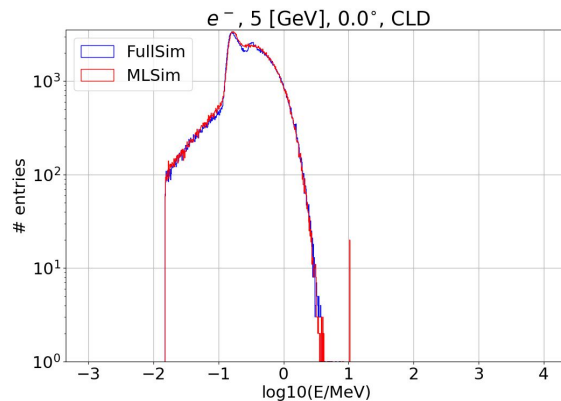
Par04 SciPb



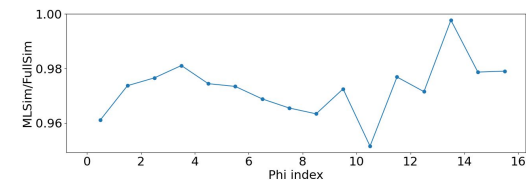
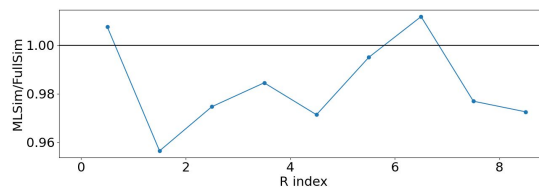
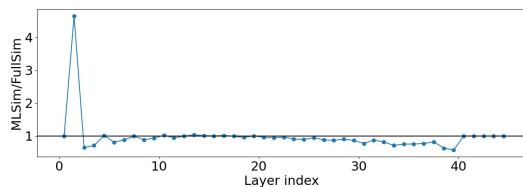
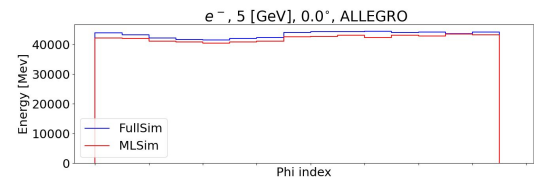
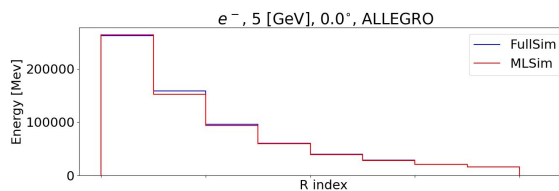
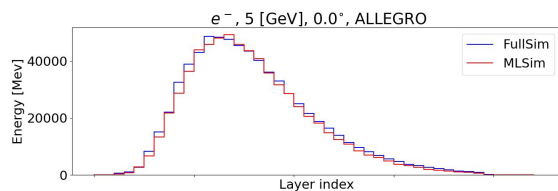
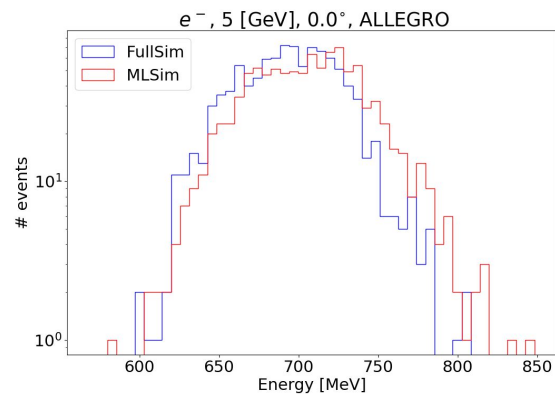
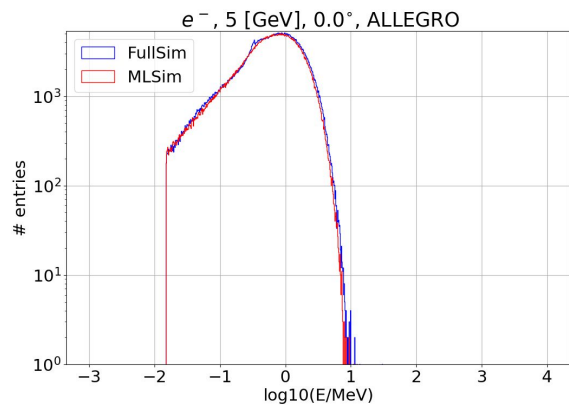
ODD



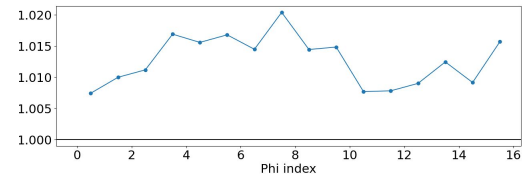
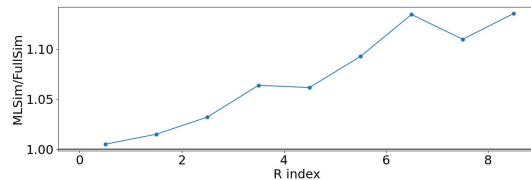
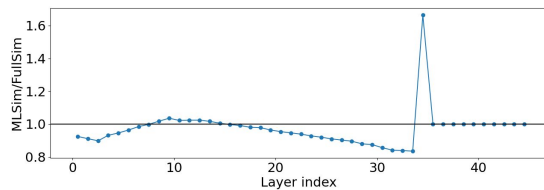
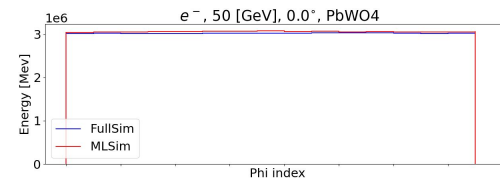
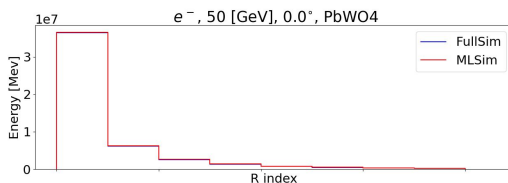
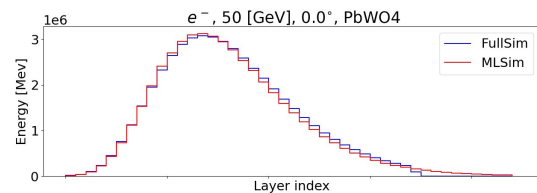
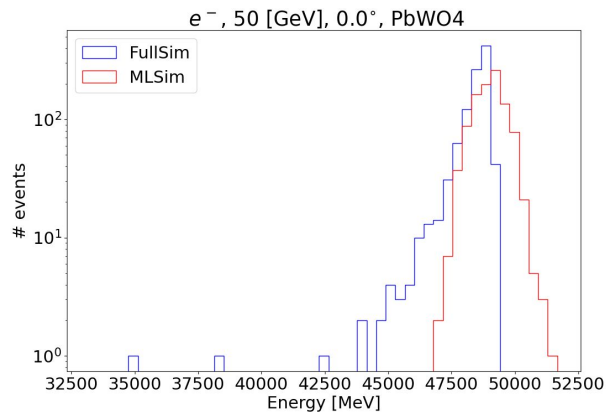
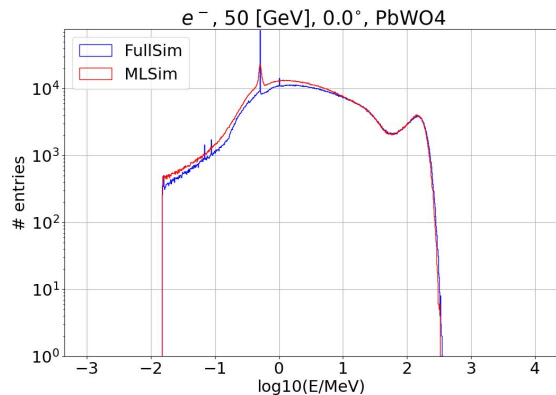
FCCeeCLD



FCCeeALLEGRO



Par04 PbWO4

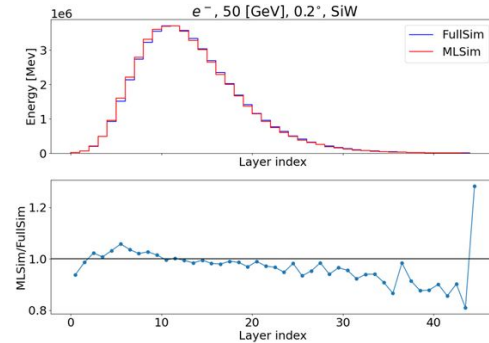
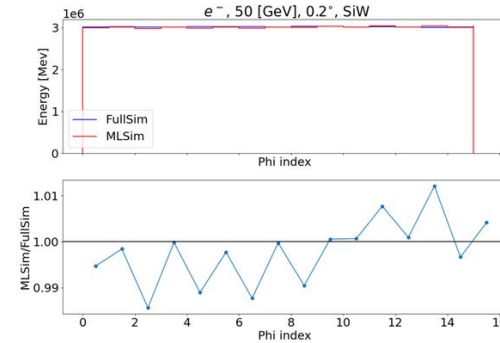
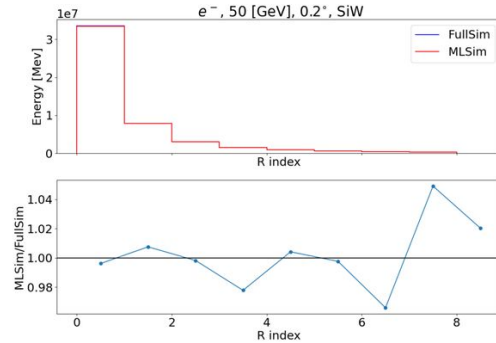
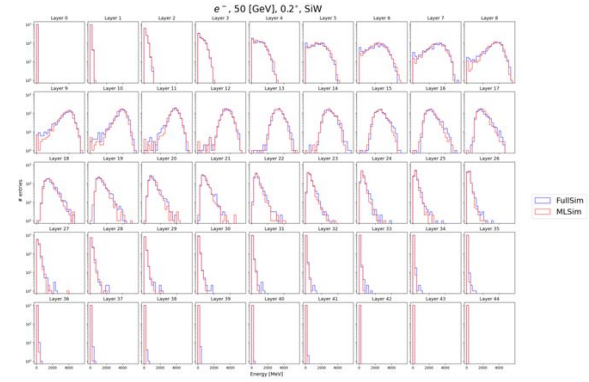
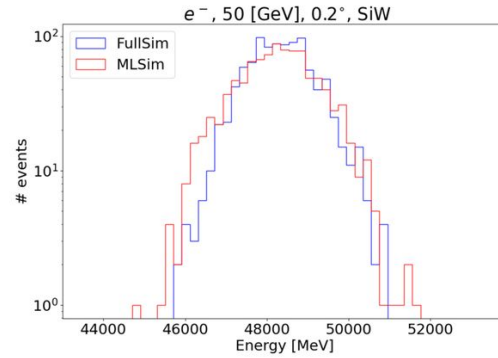
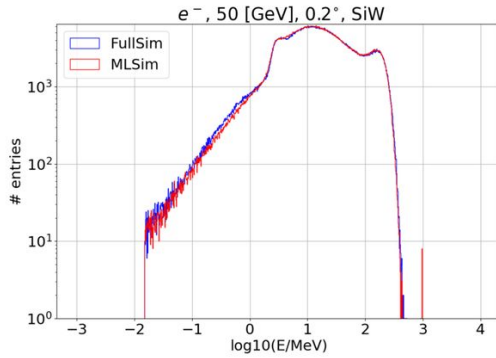


Architectural improvements

- Simpler preprocessing
 - Previously, $\text{logit}(x)$
 - $\log(x + \epsilon)$
 - ϵ value matters, $1e-6$ is working well
- Dynamic scaling
 - Usually, based on history
 - μ, σ are predicted based on the conditions
 - Added in the transformer blocks
 - Improves results a bit
 - *Could be useful for different detector geometries*
- Results on the previous slides only has simpler preprocessing

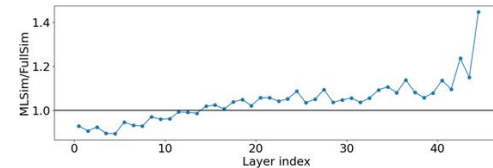
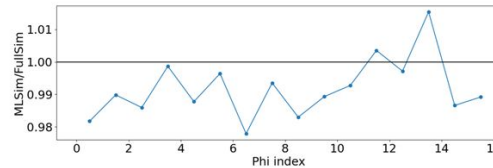
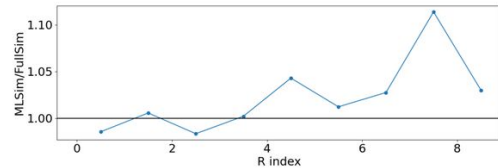
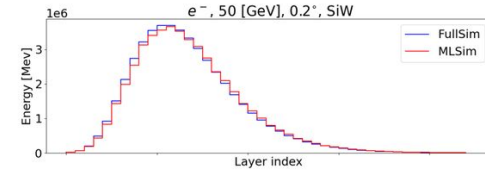
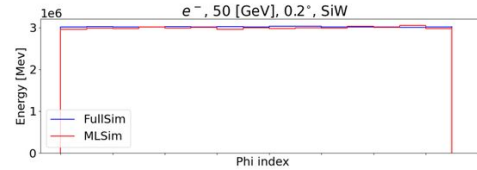
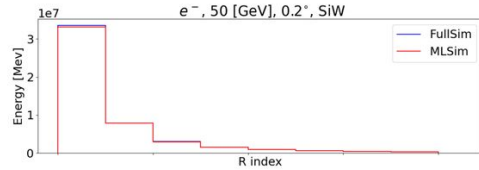
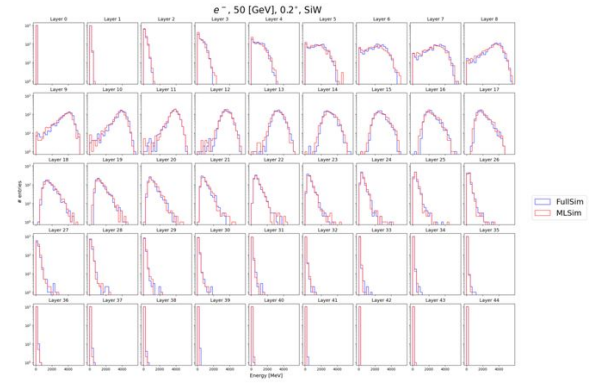
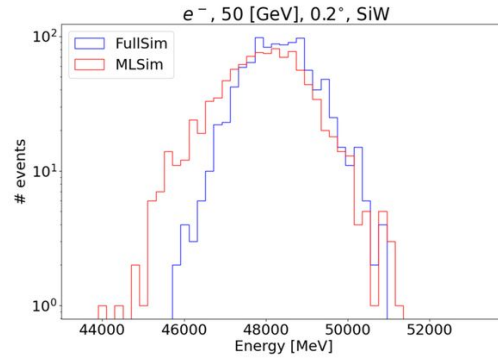
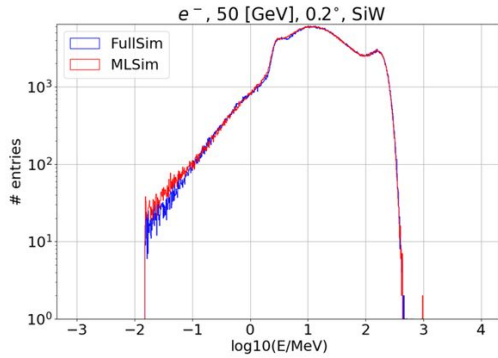
Faster Sampling of Diffusion Models

Original DDPM - 400 steps (solves SDE)

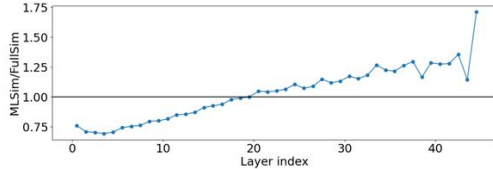
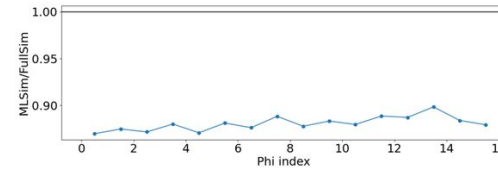
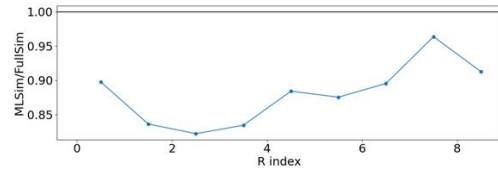
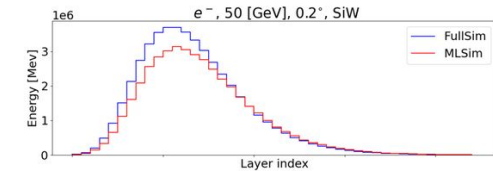
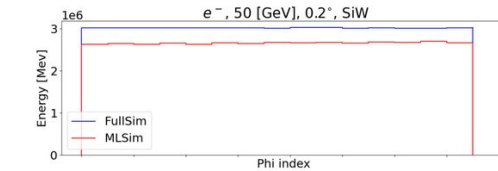
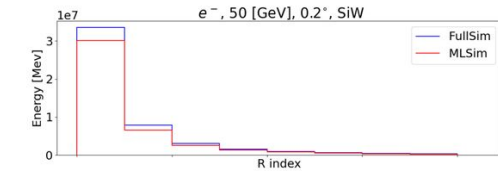
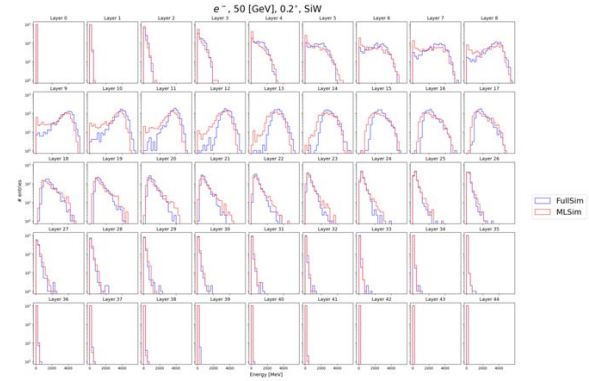
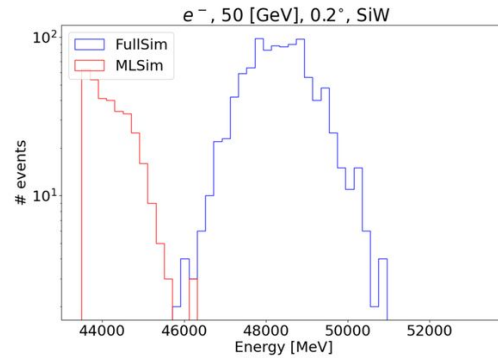
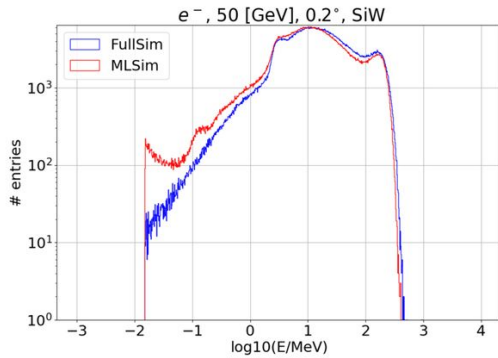


DDIM - 100 steps

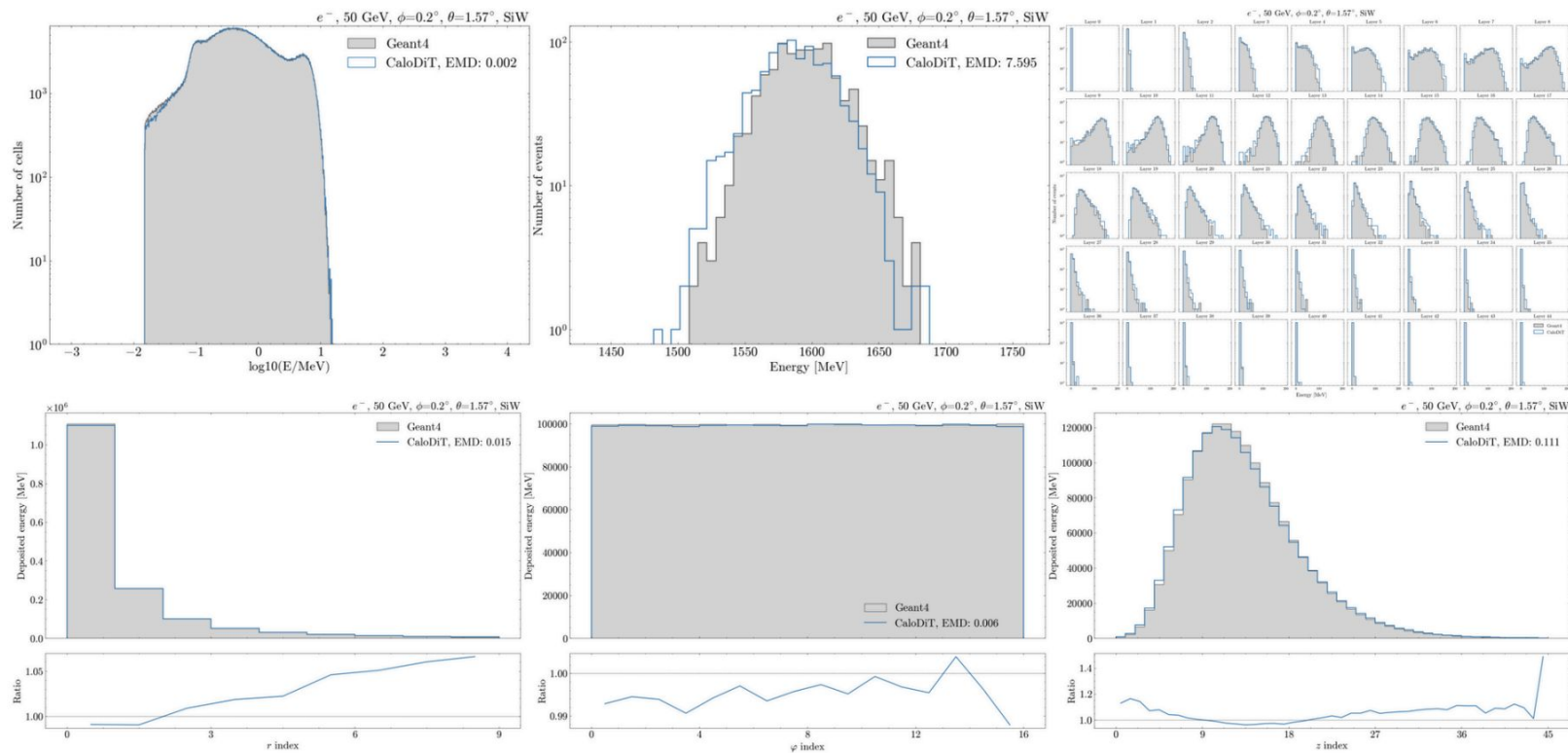
(solves ODE using a NN, can skip some steps)



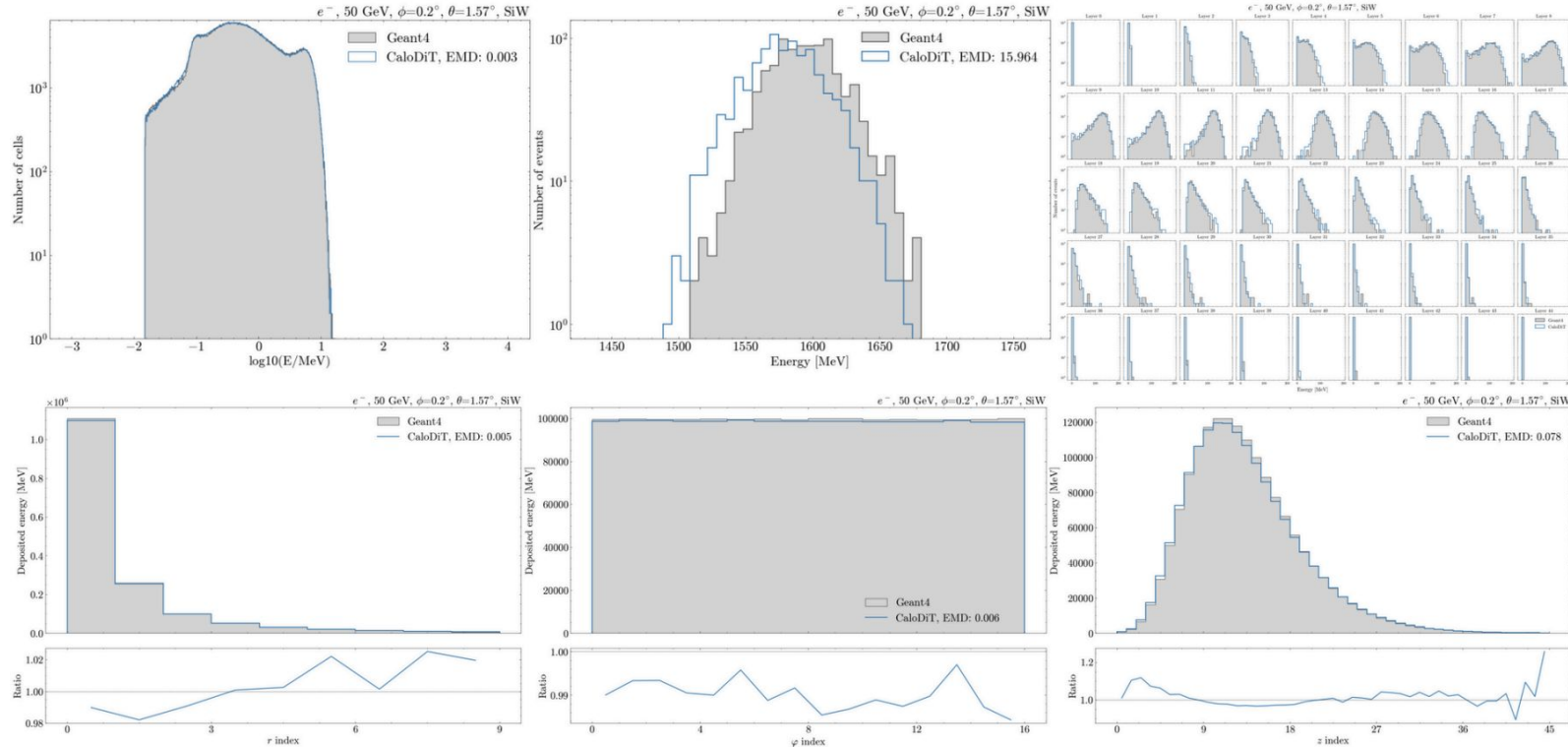
DDIM - 20 steps (too much skipping and it gets worse)



EDM - Heun 40 steps (improved Euler, solves ODE)

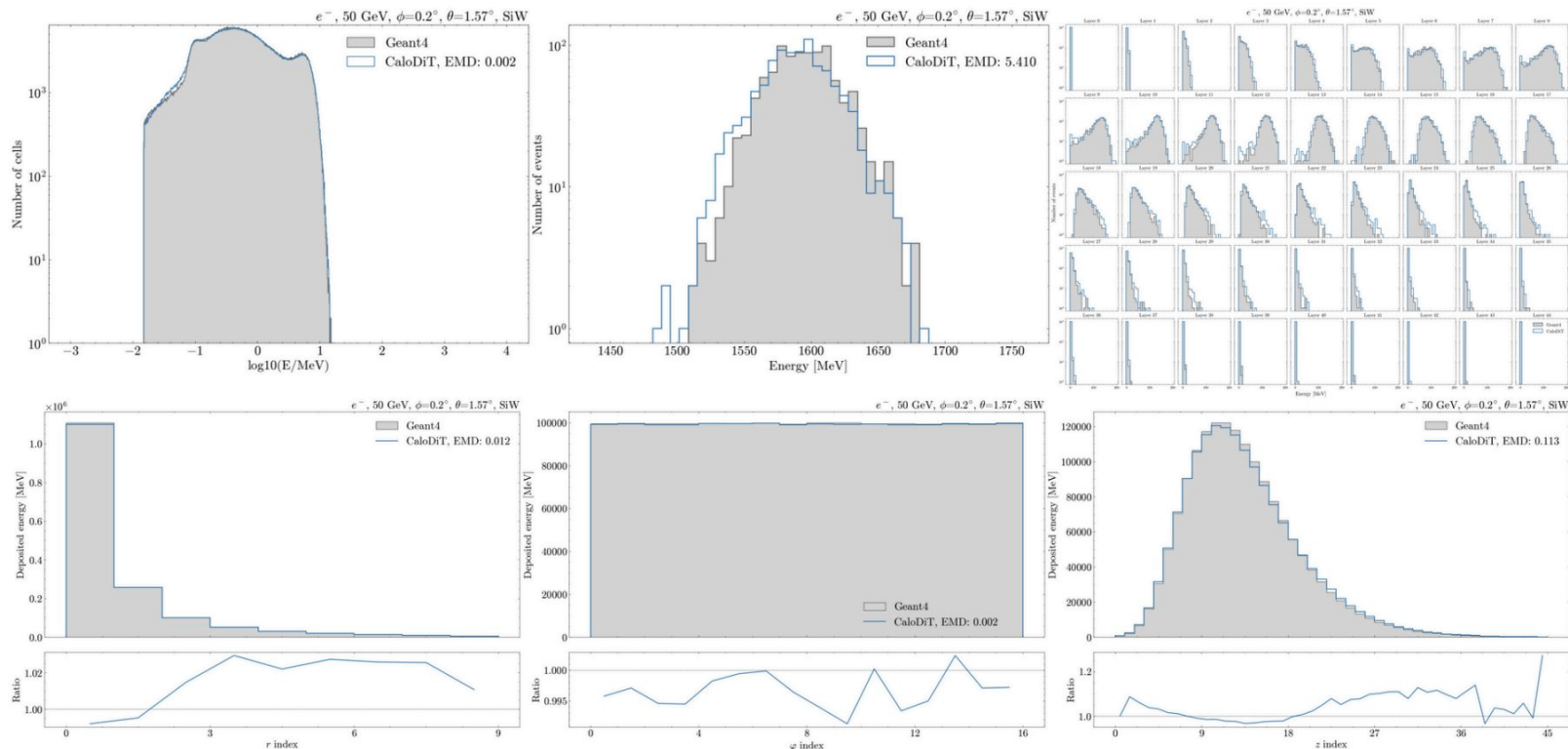


EDM - LMS 40 steps (another classical method to solve ODE)

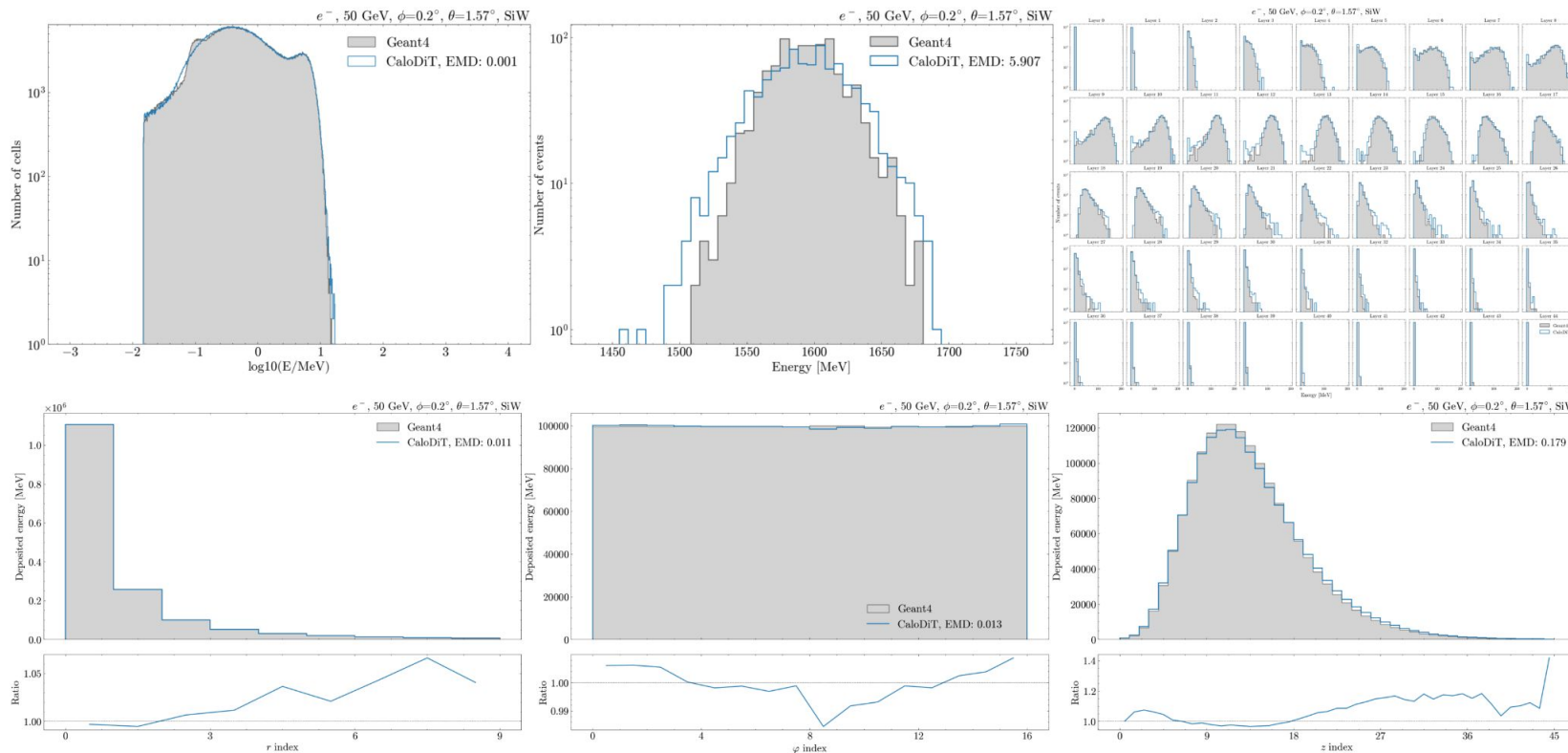


DPM-Solver++ 20 steps

(ODE solver based on NN specifically for diffusion)



Consistency distillation (1 step)



Timings

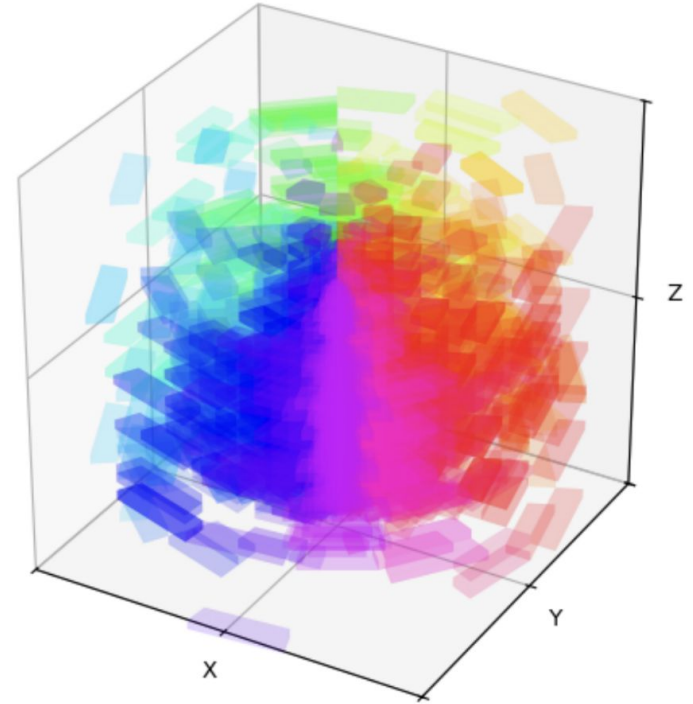
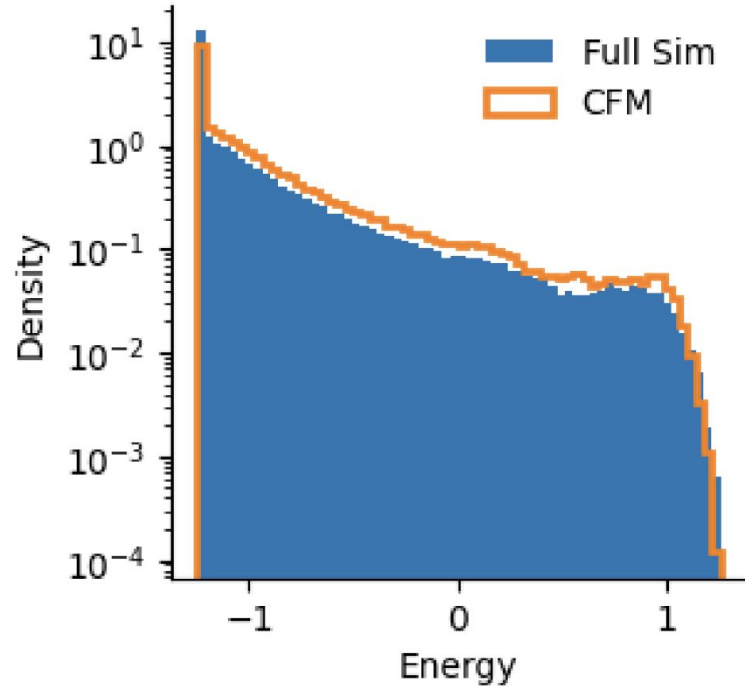
Geant4 timings picked up from another paper

- 10GeV - 90GeV, uniformly

Device	Method	NFE	Batch size	Time / Shower [ms]	Speed-up
	Geant4			3914.8 ± 74.09 ¹⁰	x1
CPU (single core)	Heun	79	1	12358.573 ± 73.218	x0.3
	LMS	40	1	6284.26 ± 25.509	x0.6
	DPM-Solver++(2M)	25	1	4085.132 ± 17.304	x1
	DPM-Solver++(3M)	25	1	4092.242 ± 21.617	x1
	CD	2	1	316.209 ± 0.188	x12
	CD	1	1	158.747 ± 0.85	x25
CPU (multi-core)	Heun	79	1	1882.3 ± 1.49	x2
	LMS	40	1	1022.525 ± 1.243	x4
	DPM-Solver++(2M)	25	1	682.637 ± 10.97	x6
	DPM-Solver++(3M)	25	1	652.725 ± 1.598	x6
	CD	2	1	51.743 ± 0.408	x76
	CD	1	1	25.409 ± 0.286	x154
GPU	Heun	79	64	105.271 ± 0.288	x37
	LMS	40	64	52.91 ± 0.259	x74
	DPM-Solver++(2M)	25	64	34.353 ± 0.193	x114
	DPM-Solver++(3M)	25	64	34.389 ± 0.197	x114
	CD	2	64	2.612 ± 0.005	x1499
	CD	1	64	1.312 ± 0.008	x2983

Mikołaj Piórczyński

Flow Matching

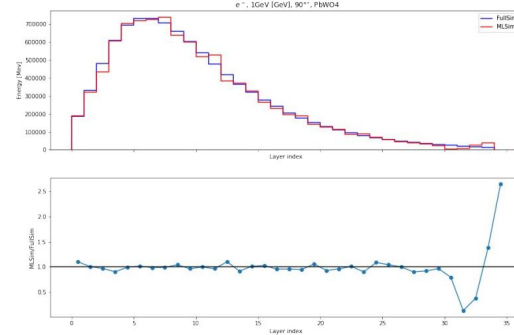
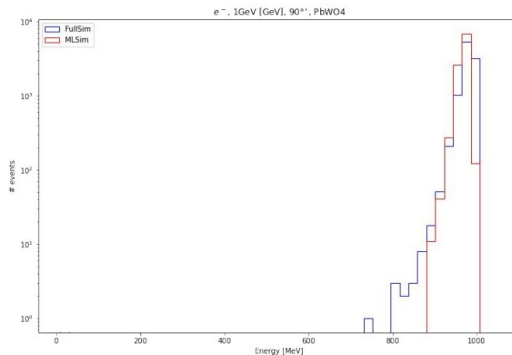
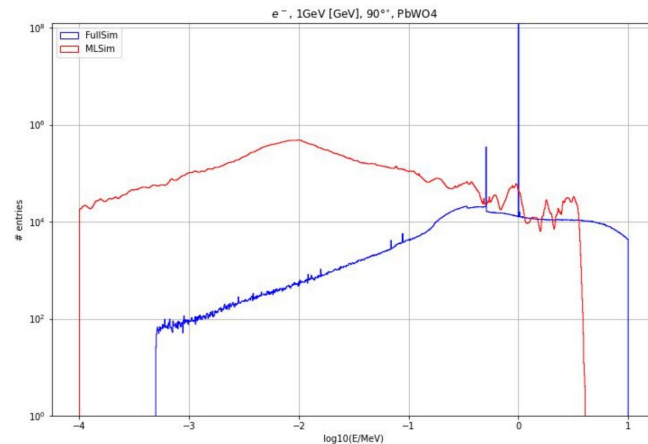


Very preliminary results. Need further investigation but not urgent.

Low energy showers

Summarized

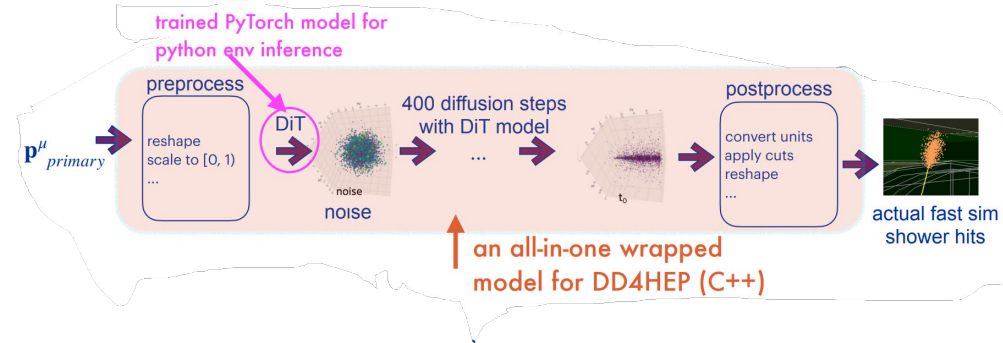
- Dataset
 - PbWO4 geometry
 - 100MeV to 1GeV, 500k events
 - Granularity same as other datasets
- Model
 - Explored VAE, vanilla and with stochastic decoder
 - + Normalizing flows to generate layer-wise energies
- Need further investigation into what exactly the requirements are



Integration

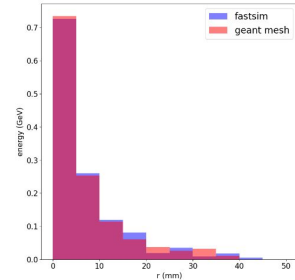
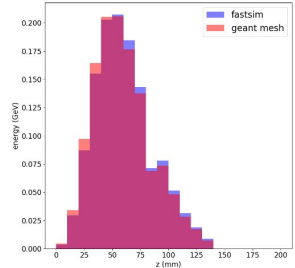
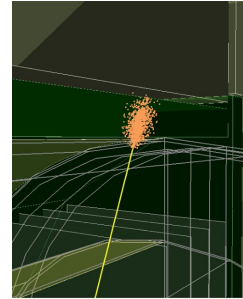
Model conversion

- Mixing tracing and scripting
 - Earlier - Trace one forward pass, implement the for-loop, pre/post-processing in C++
 - Now - Trace one forward pass, script the rest
 - A comparison between the two approaches is pending
- Explored two frameworks
 - ONNX
 - wrong CPU results
 - GPU results are good
 - TorchScript
 - faster than ONNX
 - same results as Python
 - both CPU & GPU works without issues



DDFastShowerML (DDML) Integration

- As first step, focus on integrating CaloDiT for CLD
 - Make use of existing infrastructure for Calice-style sandwich calorimeters in this library
 - Use new cylindrical mesh placement
- Integrated 400 diffusion step CaloDiT model using TorchScript inference
 - CPU, batchsize 1
- Initial comparison with pythonic inference in cylindrical mesh to validate model- limited statistics so far (400 diffusion steps)
- Next steps:
 - Explore placement from cylindrical scoring mesh into detector readout using Geant4 showers-add functionality to DDML
 - Other detectors (Allegro etc.)
 - Integrate distilled diffusion model
 - Implement physics benchmarks (di-photon separations etc.)- how good do we need to be?



Next steps

- Model conversion
 - ONNX CPU debugging
- PbWO4 issues.
- **Start multi-geometry training and adaptation.**
- Fixing some numerical overflows in the model.
- Faster sampling
 - Consistency training (simpler pipeline)
 - Easy consistency distillation (significantly faster distillation)
- Flow Matching (not urgent)
- Continue progress on integrating CaloDiT

News

- Abstracts accepted for [ML4Jets workshop](#) (Nov 4-8, Paris)
 1. Faster diffusion and generalizable model
 2. DDML
- A new collaboration in consideration