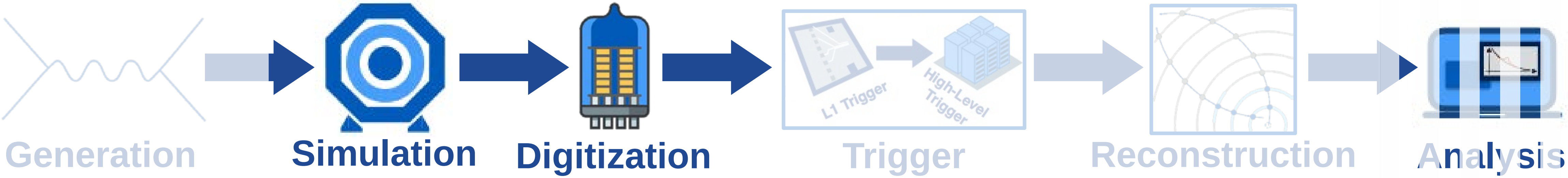
Machine Learning for Detector Simulation

Kevin Pedro
(Fermilab)
May 6, 2024

HEP Data Processing & Analysis



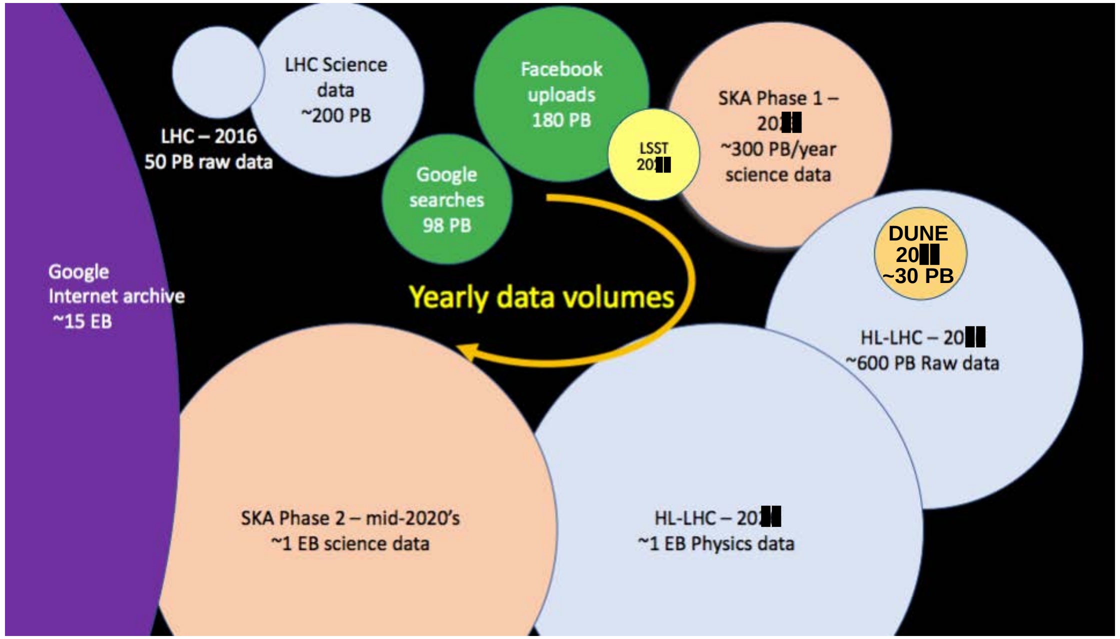
Geometry navigation
Material modeling
Physics interactions

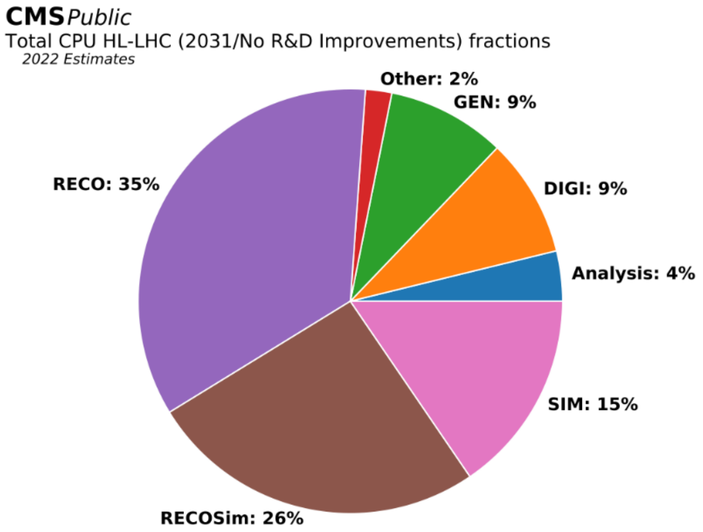
Electronics modeling
Pileup

Somewhat collider-centric view, but conceptually similar in neutrino physics, astrophysics, etc.

- Nearly all HEP results are built on simulations:
  - Detector design, analysis optimization, background estimation, etc.
- As we probe rarer processes, explore more complicated models, and make more precise measurements:
  - Accuracy and computational speed increase in importance!

Computing

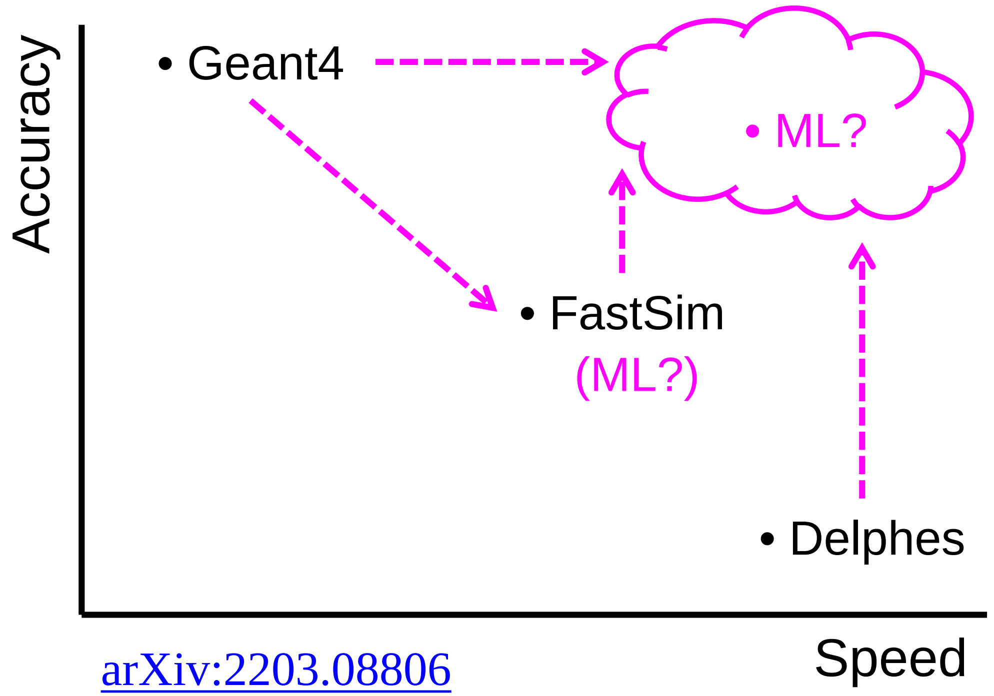




- A new precision era is imminent: HL-LHC, DUNE, LSST, SKA
  - 10× or more data compared to existing experiments
- **Simulation** needs to deliver more events with more complexity and more accuracy
  - Match growing data volumes and improved detectors… while using *smaller fraction* of computing!
    - To allow for increasing fraction of **reconstruction** (scales *superlinearly* with pileup)

ML4Sim Landscape

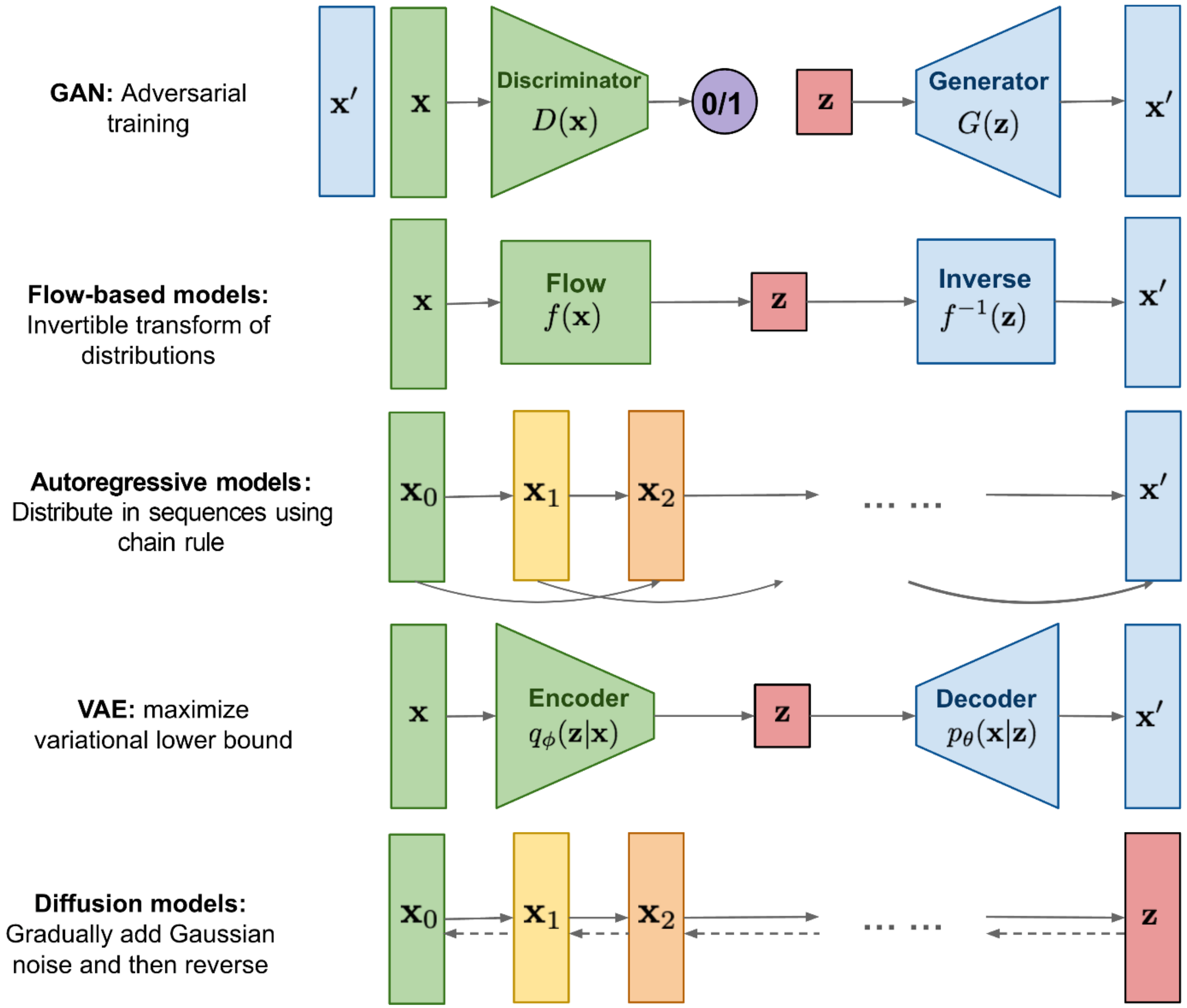
- Options to use ML for sim:
  1. Replace or augment (part or all of) Geant4
  2. Replace or augment (part or all of) FastSim
- Goals:
  1. Increase speed while preserving accuracy
  2. Preserve speed while increasing accuracy
- ML can also create faster, but less accurate simulation
  - à la existing classical FastSim
    - then augment w/ more ML to improve accuracy
- Another option: replace entire chain (“end-to-end”)
  - Complements other cases



arXiv:2203.08806

“replace” → generative ML
“augment” → ML refinement

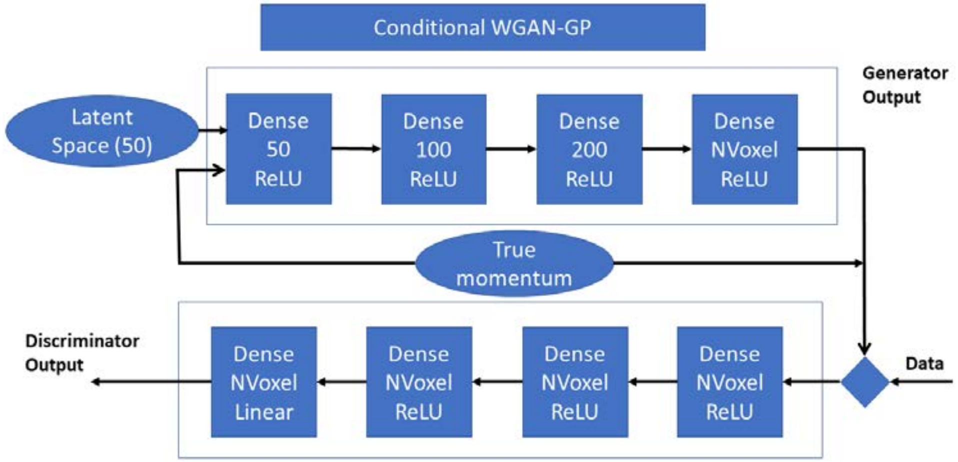
Generative Models



L. Weng

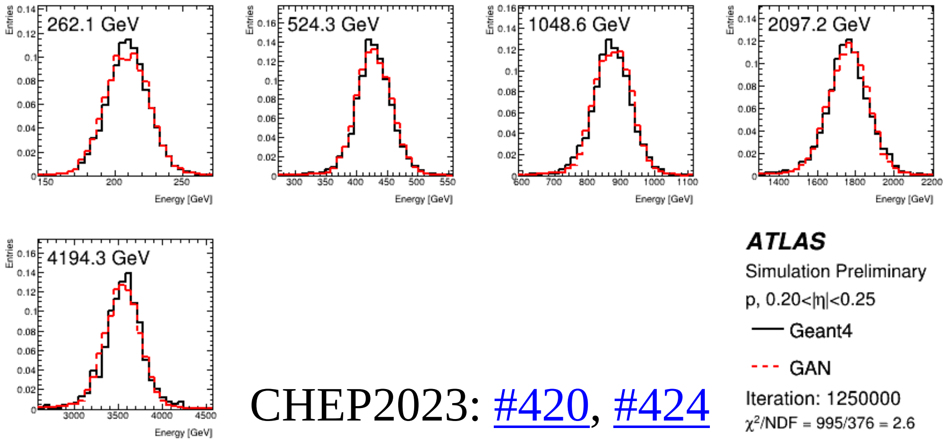
- *Implicit* density estimation: Generative Adversarial Networks (GANs)
  - Pros: fast
  - Cons: can suffer from mode collapse, lack of convergence, etc.
- *Exact* density estimation: Normalizing Flows (NFs), Autoregressive models (ARs)
  - Pros: accurate, fast in one direction
  - Cons: poor scaling, slow in other direction
- *Approximate* density estimation: Variational Autoencoders (VAEs), Diffusion Models (DMs)
  - VAEs: fast, but limited quality
  - DMs: high quality, but slow
- *Non-generative*: reweighting, refinement
  - Classification- or regression-based

Generative Models for ATLAS Simulation



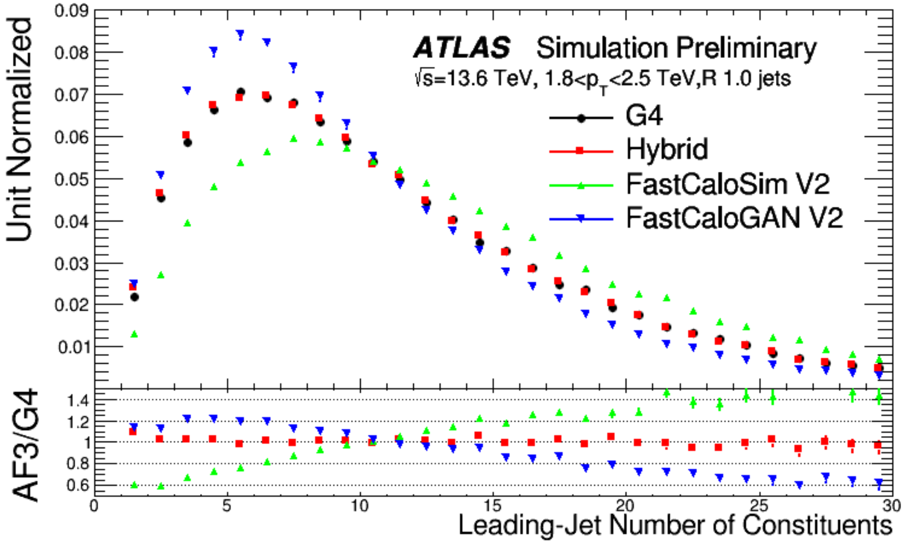
- FastCaloGAN architecture: Wasserstein loss prevents mode collapse
- Separate GANs trained for 100 η slices and for each particle type: γ, e, $\pi^{\pm}$, p → 600 total
  - o Hyperparameters optimized for each particle
  - o ~100 V100 GPU-days for final training
- Irregular geometry voxelized for training
- Incorporated in AtlFast3 along with FullSim and FastSim modules (depending on particle type, etc.)

- Good agreement for protons (new!)



CHEP2023: [#420](#), [#424](#)

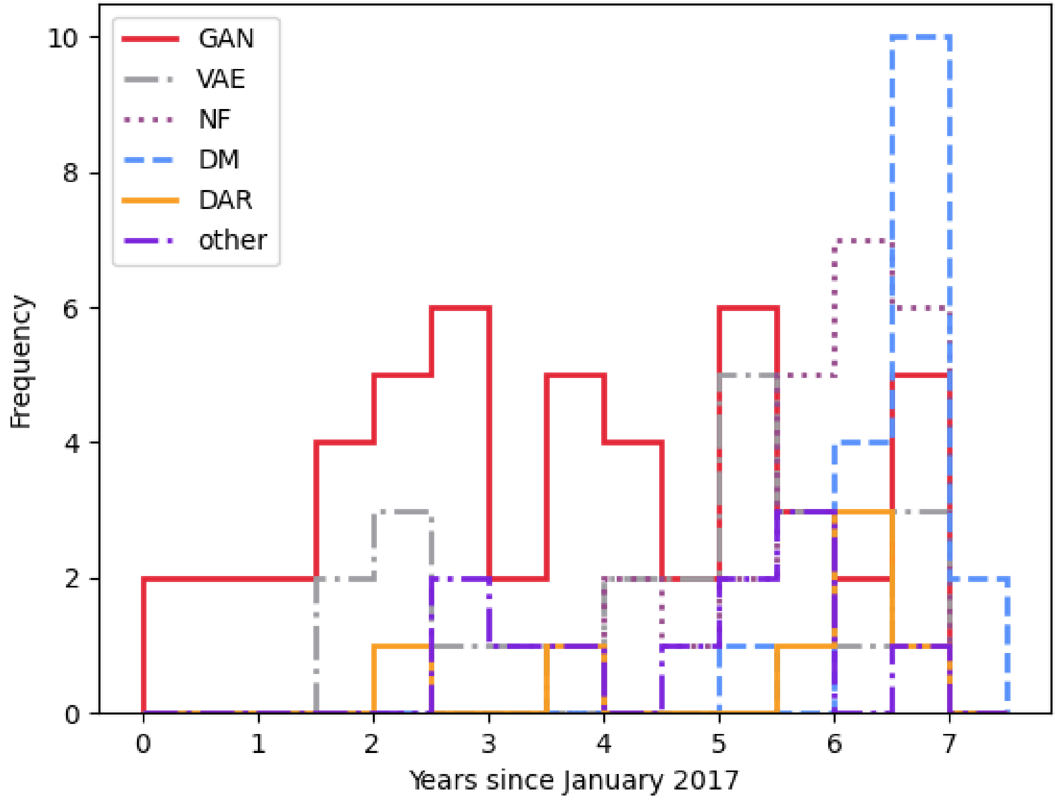
- Hybrid approach improves modeling of high-level quantities



7 Years of ML4Sim

Simulation

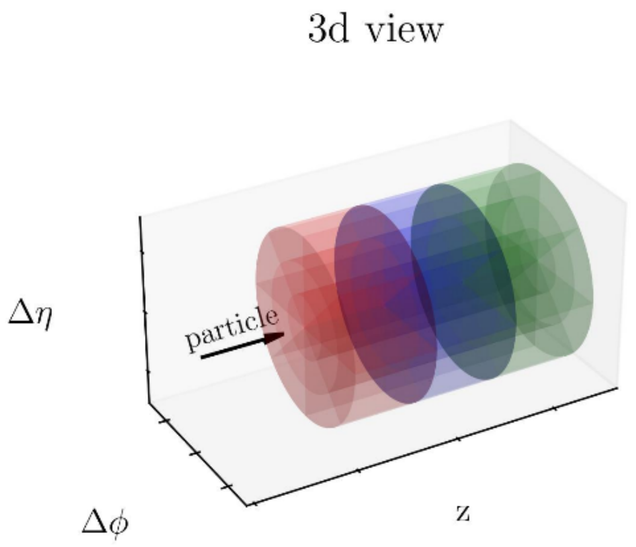
- From my database of 100+ ML4Sim-related papers
- Normalizing flows and diffusion models supplanting traditional GANs and VAEs
- Almost exponential takeoff for diffusion models
  - o Following industry dominance in image generation Stable Diffusion, DALL·E, Midjourney, etc.
- Some growing interest in autoregressive models
  - o Perhaps motivated by success in industry (GPT)
- Common datasets and metrics: big step forward to compare different approaches

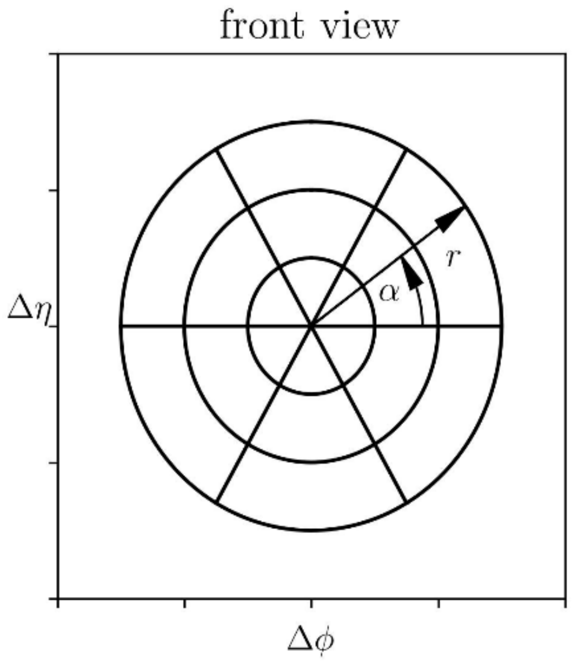


“Other” = non-generative models (FCNs, CNNs, GNNs), typically regression-based approaches

CaloChallenge

Simulation





- CaloChallenge: first competition for generative ML for detector simulation
- Three public datasets provided:
  1. Low granularity, irregular geometry (based on ATLAS calorimeter), photon & pion showers
  2. Medium granularity, silicon-tungsten sampling calorimeter, electron showers
  3. High granularity, otherwise same as #2
- Common datasets are crucial to compare different generative methods
  - Using metrics discussed on next slide
- Many new methods developed for the challenge
  - Preliminary comparisons will be shown

Metrics

Simulation

- Speed only matters if needed accuracy is achieved
  - Wrong answers can be obtained infinitely fast
- 1D histograms:
  - e.g. separation power $\langle S^2(g,h)\rangle = \frac{1}{2}\sum \frac{(g-h)^2}{(g+h)}$
  - Can miss high-dimensional correlations
- Best category: **integral probability metrics**

$$D_{\mathcal{F}}(p_{\text{real}}, p_{\text{gen}}) = \sup_{f\in\mathcal{F}} \left|\mathbb{E}_{\mathbf{x}\sim p_{\text{real}}} f(\mathbf{x}) - \mathbb{E}_{\mathbf{y}\sim p_{\text{gen}}} f(\mathbf{y})\right|$$

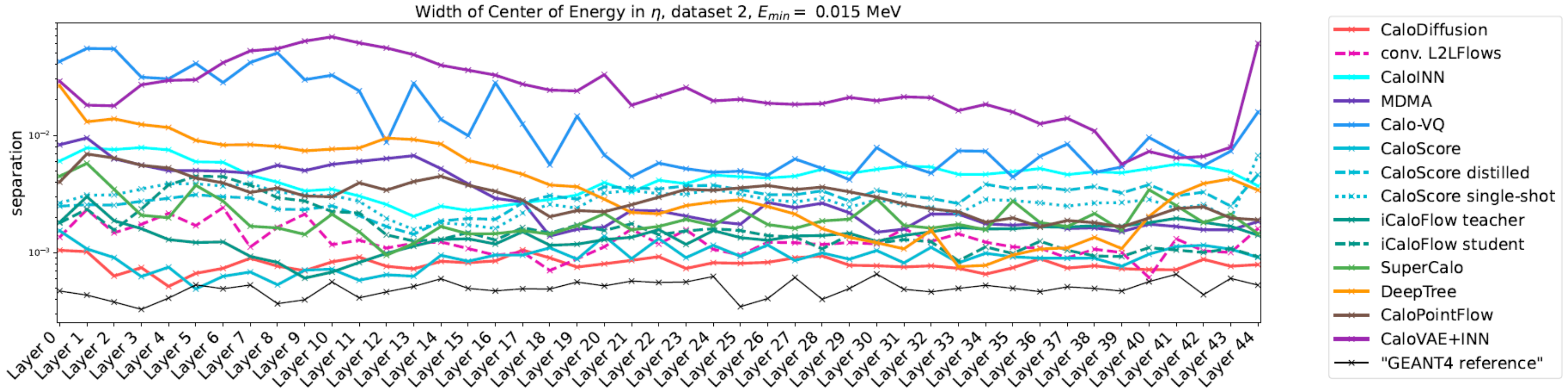
  - *Wasserstein distance* W_1: $\mathcal{F}$ is set of all K-Lipschitz functions
    - Only works well in 1D, biased in high-D
  - *Maximum mean discrepancy* (MMD): $\mathcal{F}$ is unit ball in reproducing kernel Hilbert space
    - Depends on choice of kernel
  - *Fréchet distance*: W_2 distance between Gaussian fits to (high-D) feature space
    - Features can be hand-engineered or obtained from NN activations
- Another interesting category: *classifier scores*
  - Train NN to distinguish real vs. generated
  - AUC score: ranges from 0.5 to 1.0
  - Log-posterior probability in multiclass case
- *Fréchet Particle Distance* most clearly distinguishes between two similar approaches

	FPD $\times 10^3$	KPD $\times 10^3$	W_1^M $\times 10^3$
Truth	0.08 ± 0.03	-0.006 ± 0.005	0.28 ± 0.05
MPGAN	$\mathbf{0.30 \pm 0.06}$	$\mathbf{-0.001 \pm 0.004}$	$\mathbf{0.54 \pm 0.06}$
GAPT	0.66 ± 0.09	0.001 ± 0.005	0.56 ± 0.08

arXiv:2211.10295

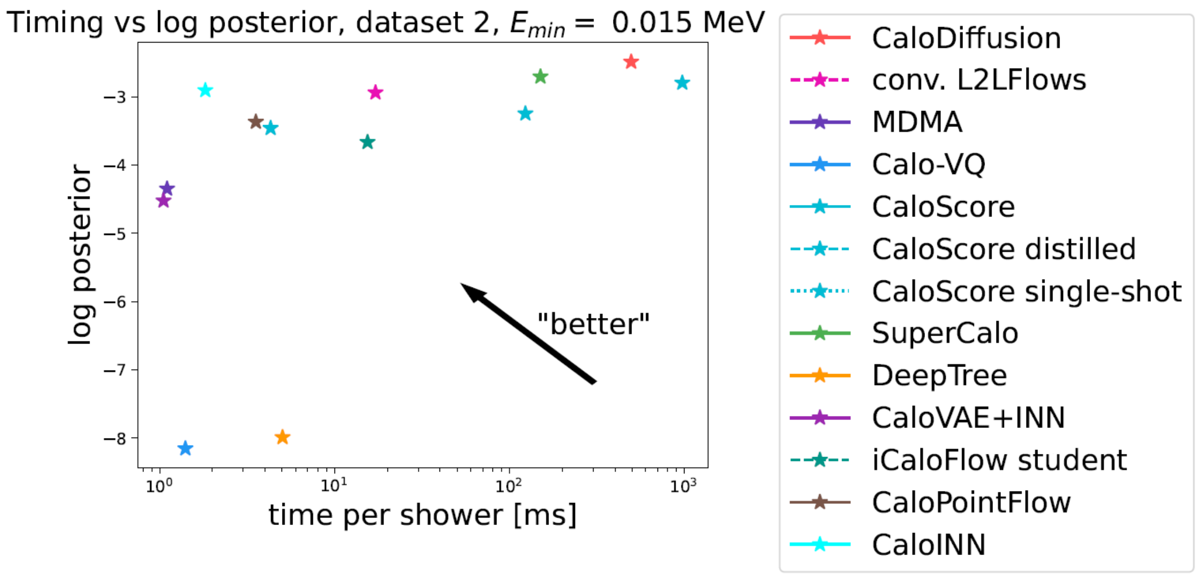
CaloChallenge Results

Simulation



C. Krause

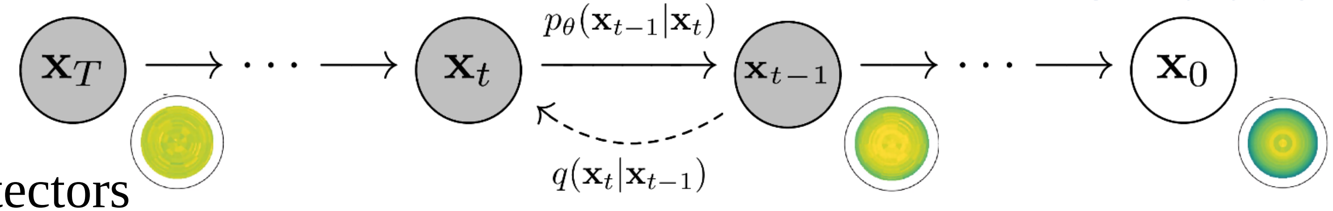
- Diffusion models and normalizing flows tend to have best performance
- However, diffusion models especially tend to be slower in inference
  - o Iterative process – multiple steps required to get highest accuracy
- Benefit of following industry trends: frequent papers with new methods to speed up diffusion models → easy to adopt in HEP



CaloDiffusion

Simulation

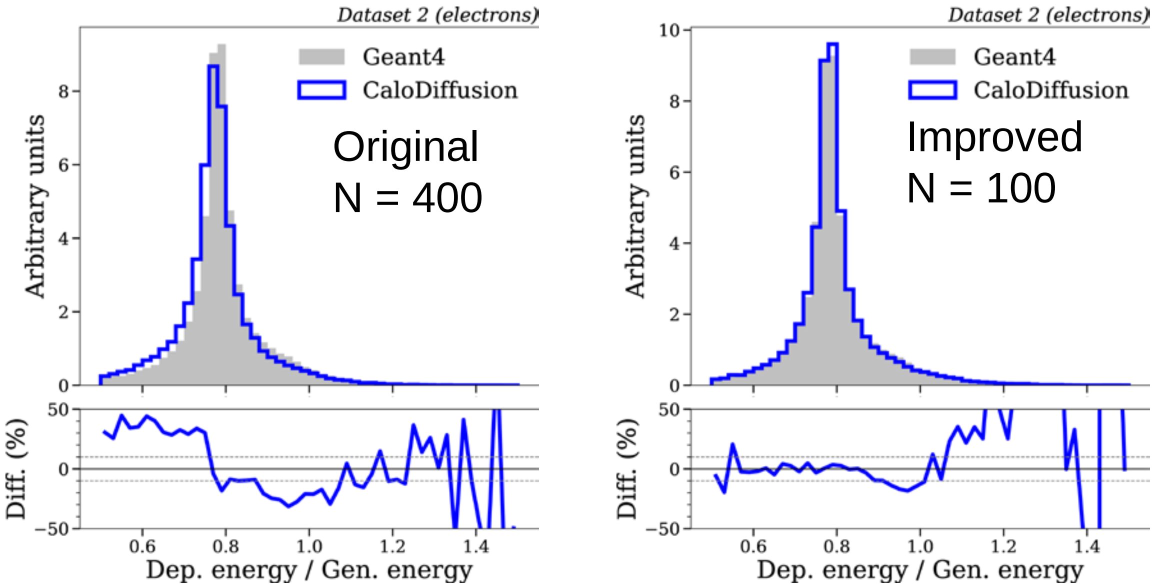
- Current state-of-the-art model: denoising w/ convolutional U-net architecture
  - o Various geometric adaptations:
    - Conditional cylindrical convolutions
    - Geometry latent mapping for irregular detectors
    - Attention layers for long-range correlations in *z*
- Comparison to other SOTA models:
  - o Best classifier AUC scores
  - o Low distance values compared to Geant4



Dataset	Classifier AUC (low / high) CaloDiffusion	CaloFlow	CaloScore v2
1 (photons)	**0.62** / 0.62	0.70 / **0.55**	0.76 / 0.59
1 (pions)	**0.65** / **0.65**	0.78 / 0.70	- / -
2 (electrons)	**0.56** / **0.56**	0.80 / 0.80	0.60 / 0.62
3 (electrons)	**0.56** / **0.57**	0.91 / 0.95	0.67 / 0.85

Dataset	FPD†	KPD
1 (photons)	0.014(1)	0.004(1)
1 (pions)	0.029(1)	0.004(1)
2 (electrons)	0.043(2)	0.0001(2)
3 (electrons)	0.031(2)	0.0001(1)

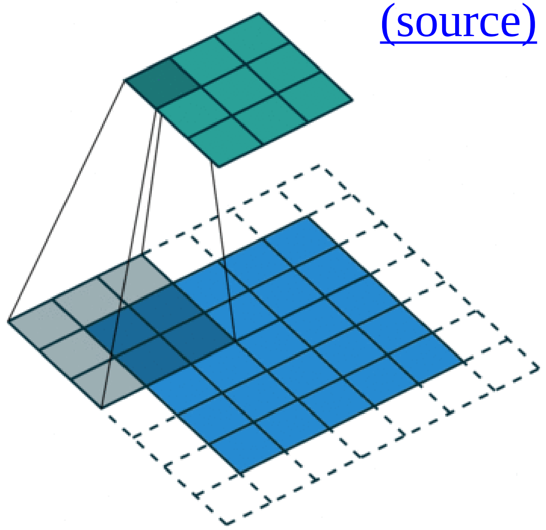
- Improvement from original: LayerDiffusion to predict total energy per layer $\rightarrow$ 4× speedup & better quality
  - o More speedups in [arXiv:2401.13162]



ML4Jets2023

Why Convolutions?

(source)



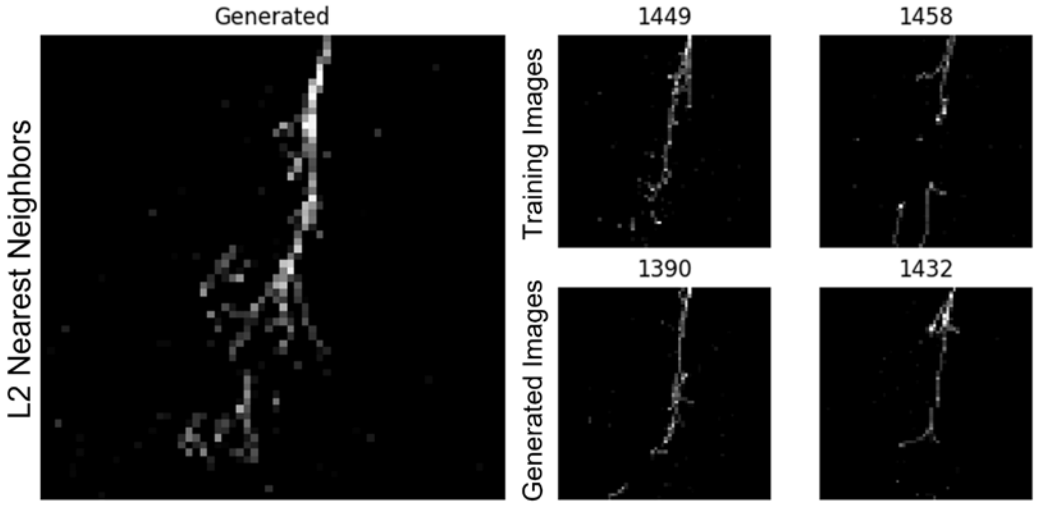
- Convolutions started the modern machine learning revolution (AlexNet, 2012)
  - *Spatial locality* and translational invariance
  - Shared weights → fewer parameters, *better scaling*
  - Highly *efficient* on GPUs: spatial locality implies memory locality
- Ideally suited for computer vision with rectangular images
  - Application to irregular geometries requires innovations
- Graph neural networks?
  - **Pro**: natural representation for irregular geometries
  - **Cons**: adjacency matrices consume substantial memory; operations less local/efficient; hard to generate arbitrary output (masking technique exists, but difficult to scale)
- Point clouds or transformers?
  - **Pro**: no adjacency matrix consuming memory
  - **Con**: discards useful geometric information, which then must be learned from (often sparse) inputs

➢ For generative applications, convolutions still have a lot to offer!
  - And they can keep up with transformers when trained properly… arXiv:2310.16764

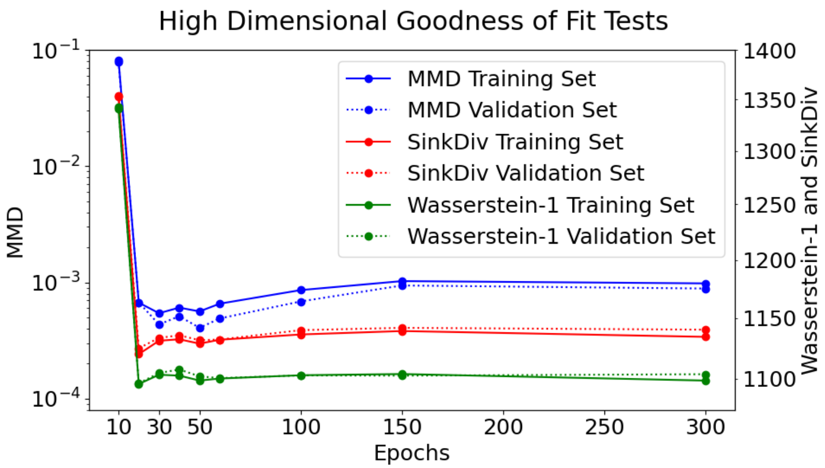
Diffusion for Liquid Argon TPCs

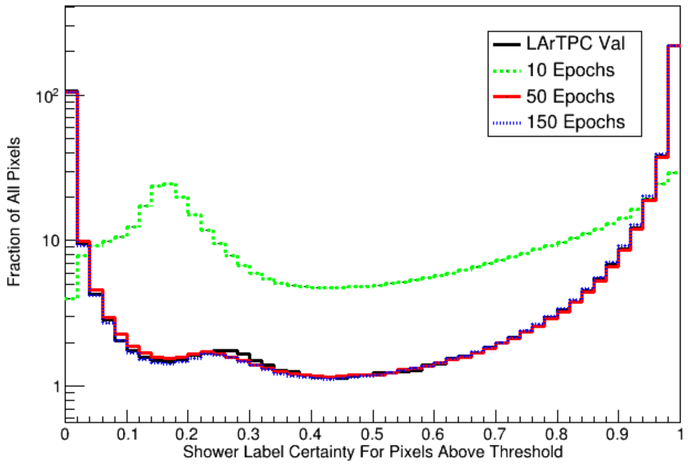
Simulation

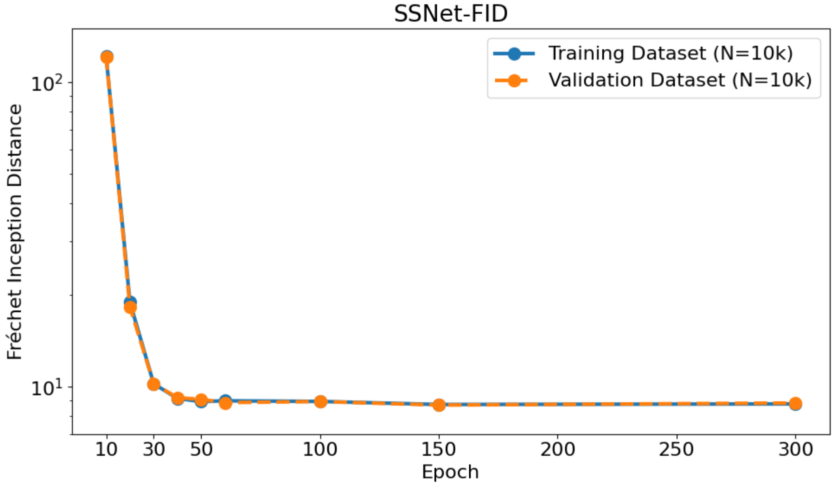
- Diffusion models can also simulate ionization deposits from charged tracks in LAr TPC detectors
  - o Here, score-based rather than denoising model is used
- Both visual and quantitative comparisons
  - o Various distance metrics, SSNet scores, Fréchet inception distance (from SSNet activations)
- Superior to previous attempts (VAE)



arXiv:2307.13687



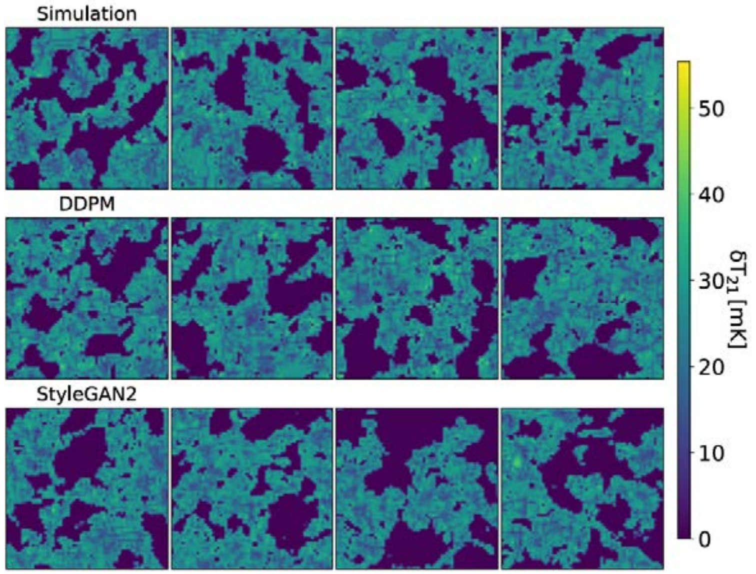


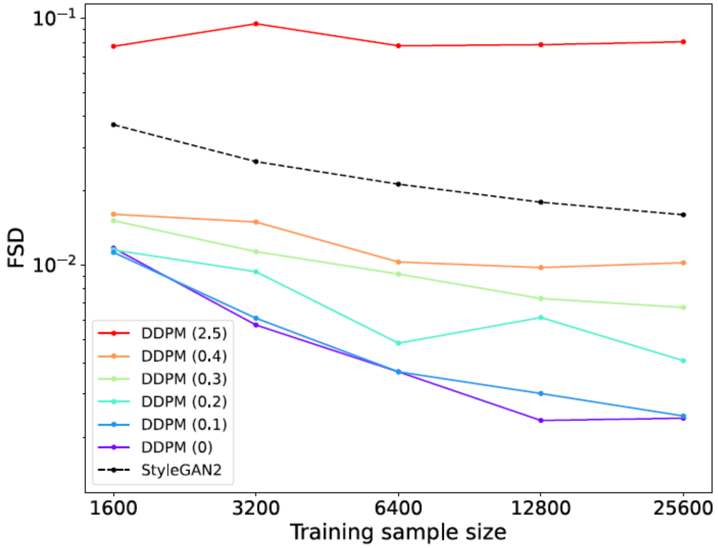


Diffusion for Astrophysical Images

Simulation

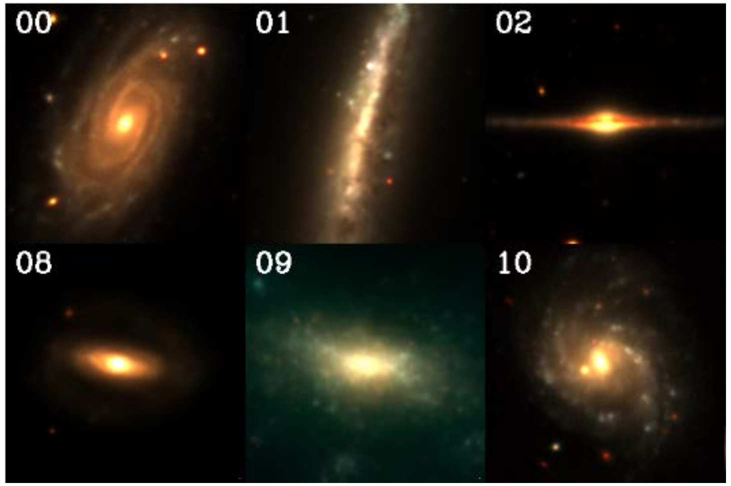
- Diffusion models can simulate various astrophysical phenomena
  - o Denoising DM for CMB maps (21 cm brightness temperature)
- Quantified using Fréchet scattering distance (from coefficients)
  - o Substantial improvement over GANs
  - o ~100× slower than GANs, but GPU inference still ~5× faster than traditional CPU-based simulation



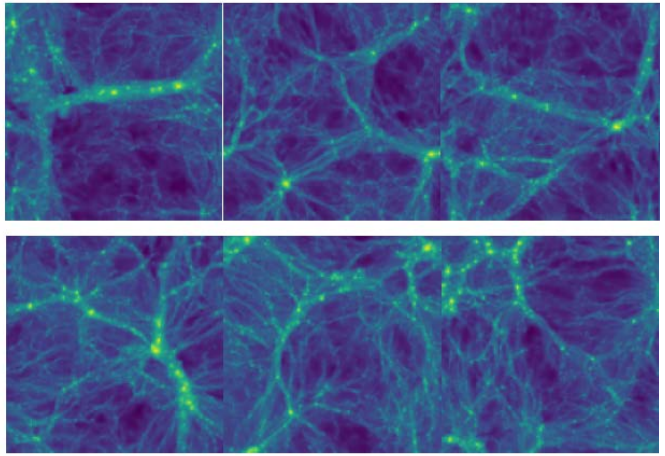


arXiv:2307.09568

- Also applied to:
  - o Galaxy images (arXiv:2111.01713)



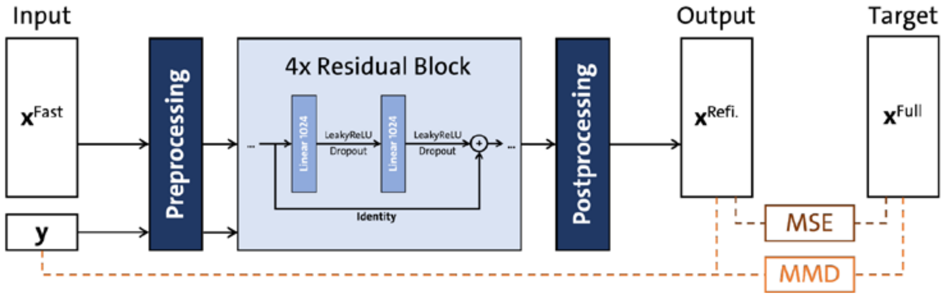
  - o Dark matter maps (arXiv:2211.12444)



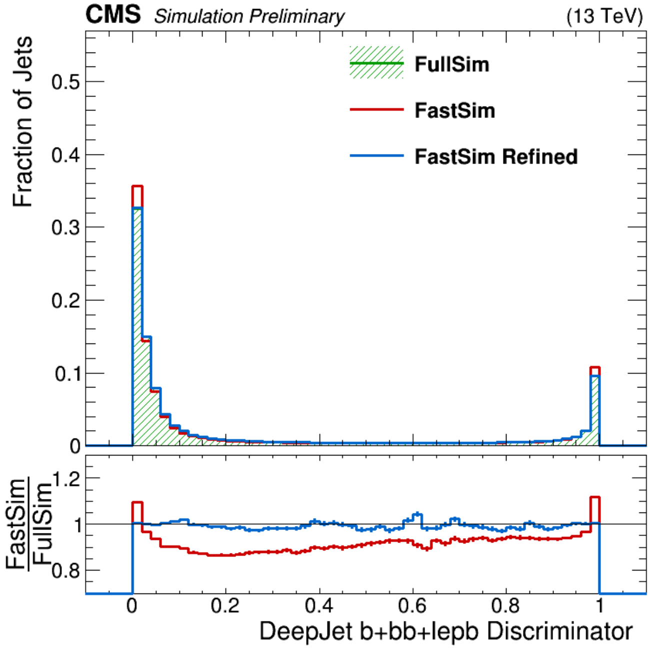
High-Level Refinement

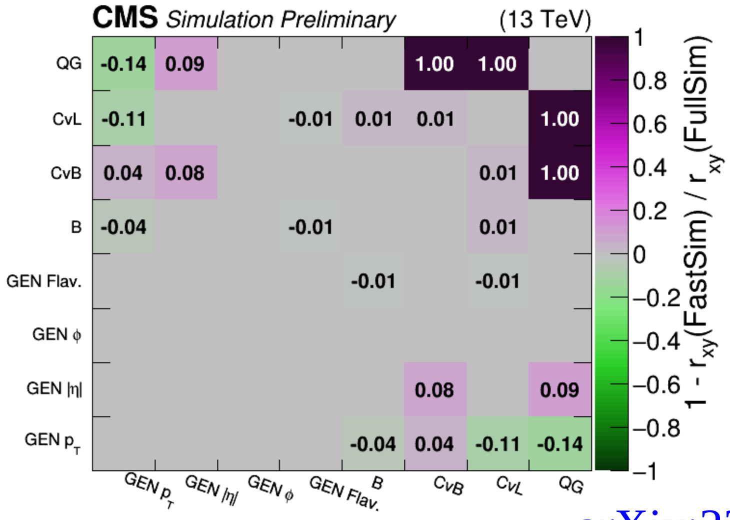
Analysis

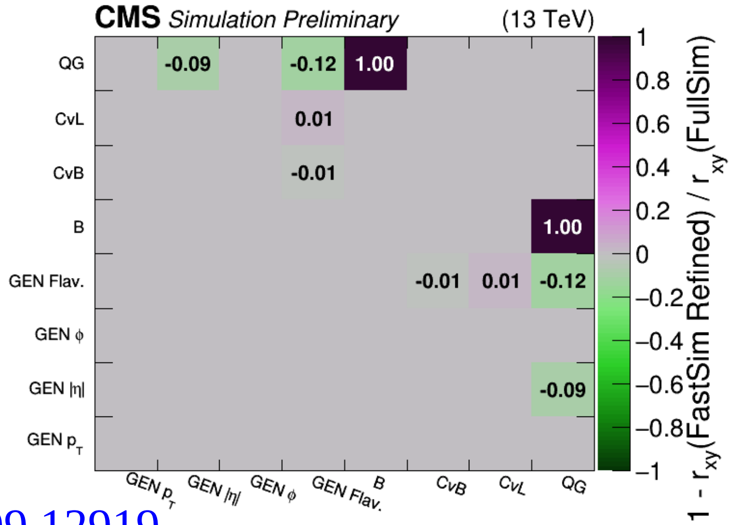
- Alternate approach: ML adjusts high-level quantities from existing CMS FastSim to match FullSim
  - Replaces coarse, manual correction factors
- Loss functions: ensemble & object-by-object comparisons
- Improves metrics, 1D distributions, correlations
- Generalizes to other processes; now being extended to more variables for Run 3 deployment



FullSim vs.	FPD $\times 10^3$	KPD $\times 10^3$
FastSim	0.801 ± 0.046	1.07 ± 0.58
FastSim Refined	0.071 ± 0.025	0.083 ± 0.418
FullSim	0.061 ± 0.029	-0.024 ± 0.250







[arXiv:2309.12919](https://arxiv.org/abs/2309.12919)

Constrained Optimization

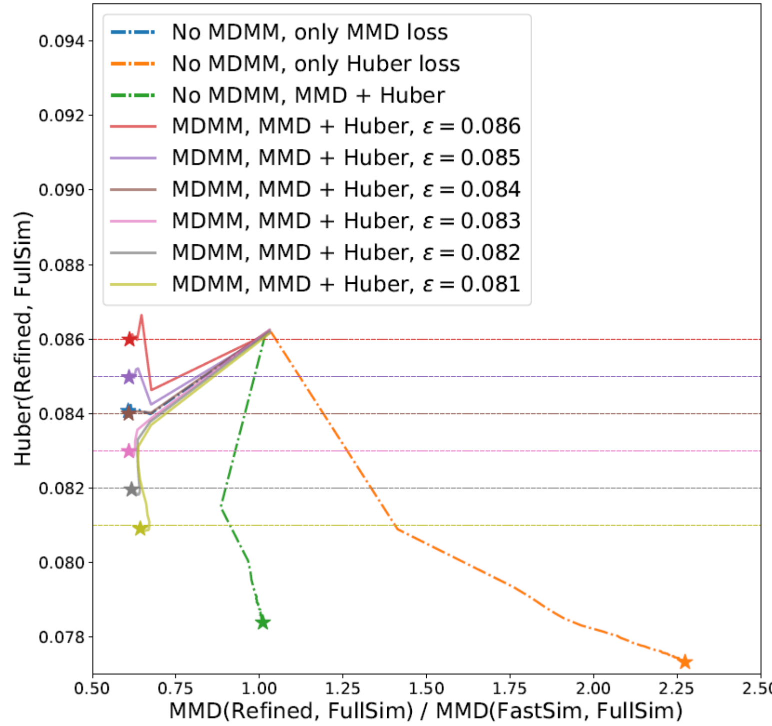
[arXiv:2309.12919]

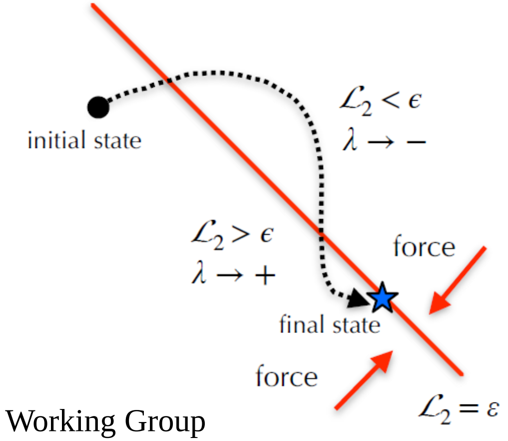
- *General principle*: you can't optimize for two things at once
  - Instead, optimize for one thing with *constraints* on others (Lagrange)
- Multiple loss terms are one approach to encode domain knowledge
- ☠ $\mathcal{L} \rightarrow \mathcal{L}_1 + \lambda\mathcal{L}_2 + \cdots$; set λ by trial and error
  - *No guarantee of optimality*; hard to control and understand
- ➢ modified differential method of multipliers (mdmm): [paper, blog, code]

learnable hyperparameter (convergence rate)

$$\mathcal{L} \rightarrow \mathcal{L}_1 - \lambda(\varepsilon - \mathcal{L}_2) + \underbrace{\delta(\varepsilon - \mathcal{L}_2)^2}_{\text{damping to ensure convergence}}$$

gradient *ascent* constraint damping to ensure convergence



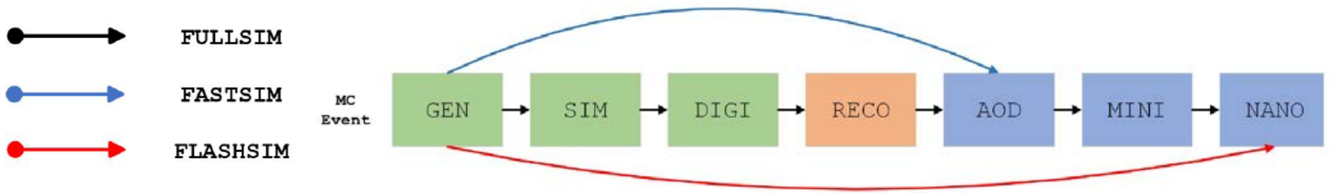


- In high-level refinement: balance per-event and ensemble losses (first known usage in HEP)
  - Minimize per-event: bad ensemble value
  - Minimize ensemble: per-event still good!
- ➢ Find Pareto front (concave or convex) and pick tradeoff

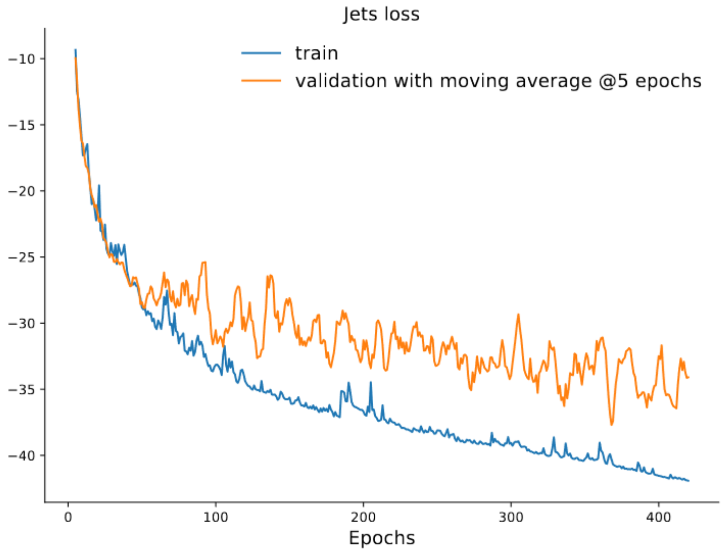
End-to-end: FlashSim

Analysis

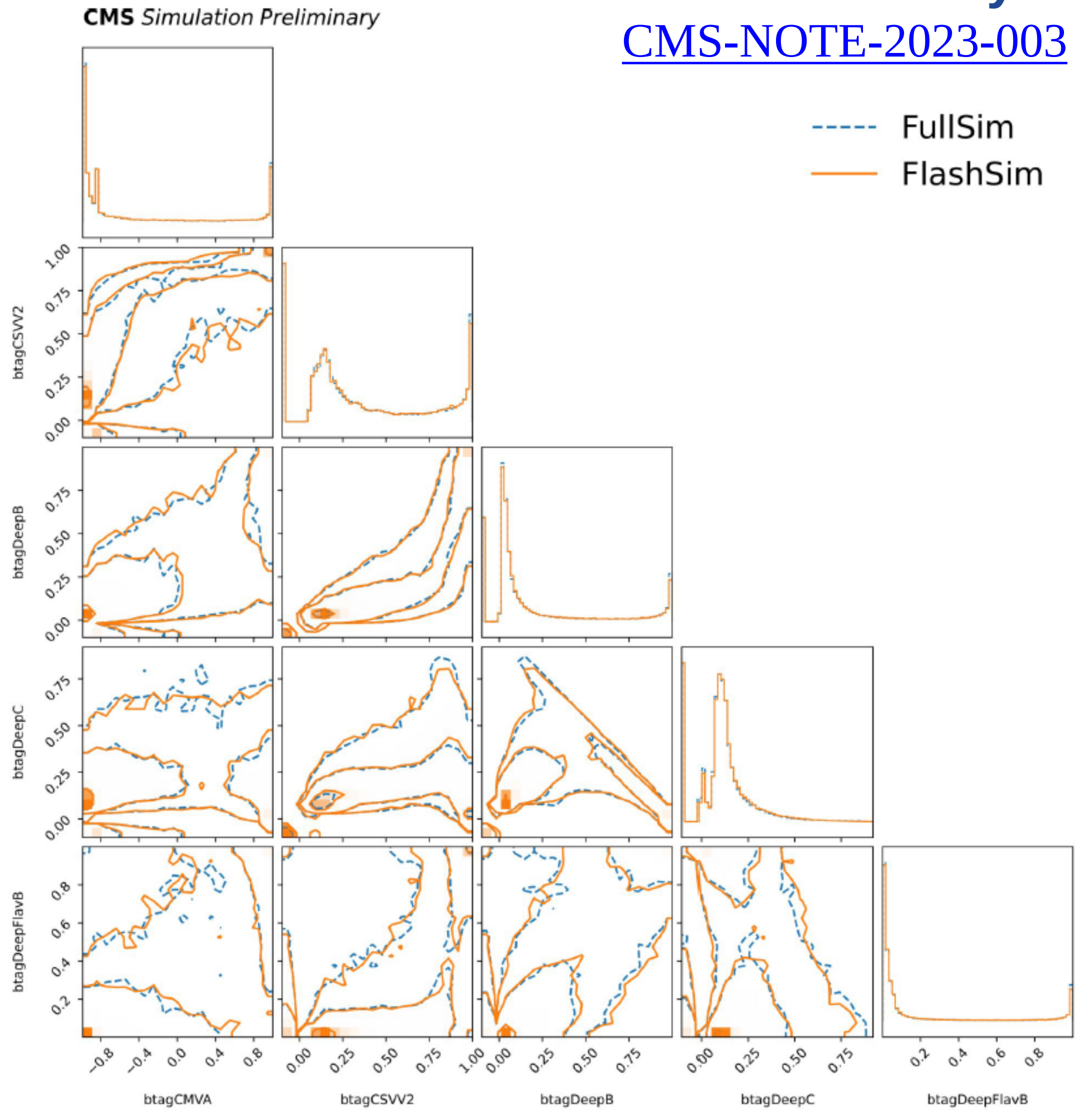
[CMS-NOTE-2023-003]



- Normalizing flow to predict high-level analysis quantities from generator-level information
  - o Need transformation for each variable: $\sim 10^7$ trainable parameters in total

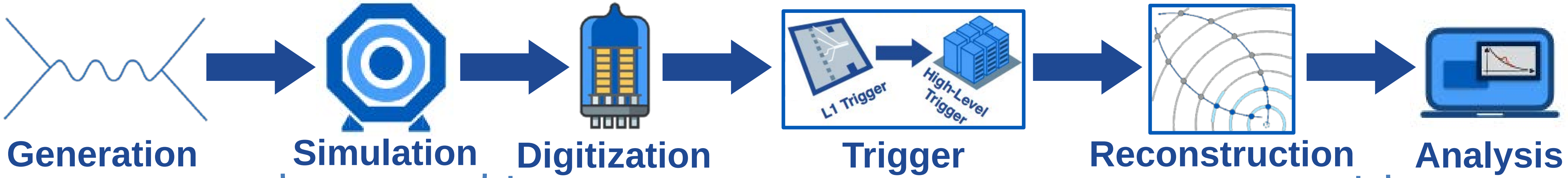


- Reproduces correlations even in ML b-tagging algorithm scores
- Currently covers: jets (real & fake), muons, electrons



- [Newer version] uses continuous flow matching

One Step vs. End-to-End



Changes O(1 year)
(geometry, physics models)

Changes O(1 second–1 month)
(calibrations, radiation damage, algorithms, …)

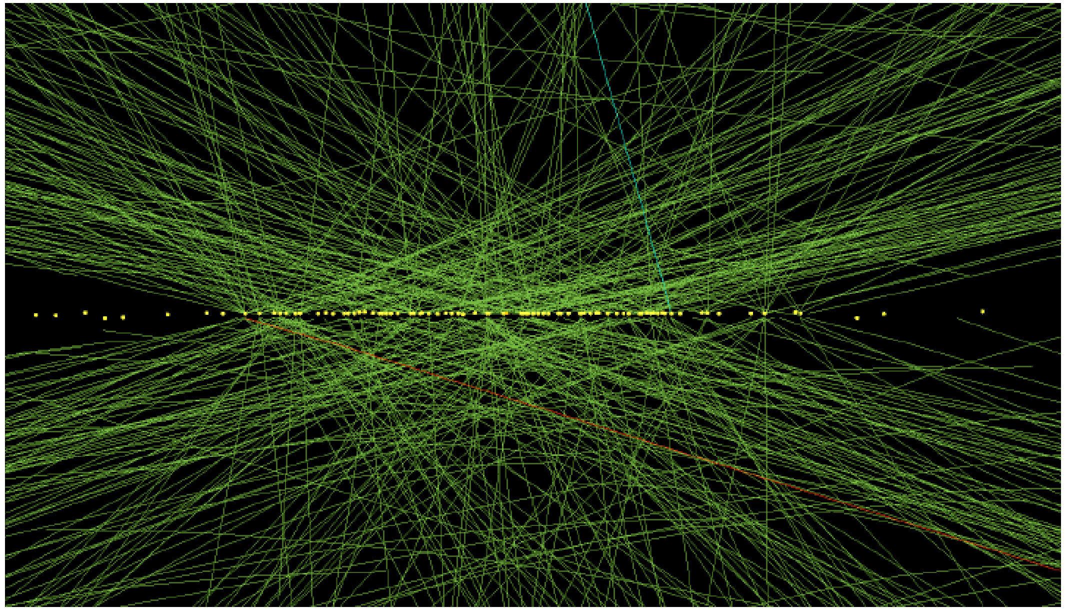
Changes O(1 year)
(mostly performed after data-taking finishes)

- End-to-end models like [FlashSim](#) that produce analysis-level observables from generator input have massive utility: essentially eliminate statistical fluctuations
  - …*for end-stage analysis*, where nothing is rapidly varying
- But accurate simulation is needed *throughout* the lifecycle of an experiment

➢ Models that target low-level simulated hits are *more broadly* applicable
  - Complementary use cases for both approaches

Pileup: An Overlooked Case

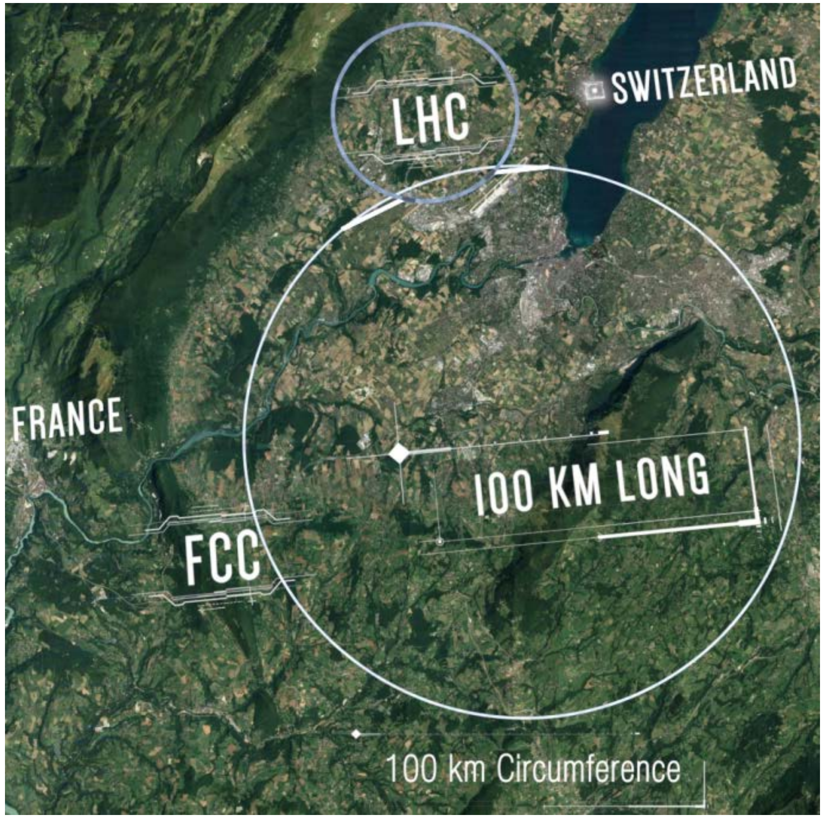
Digitization



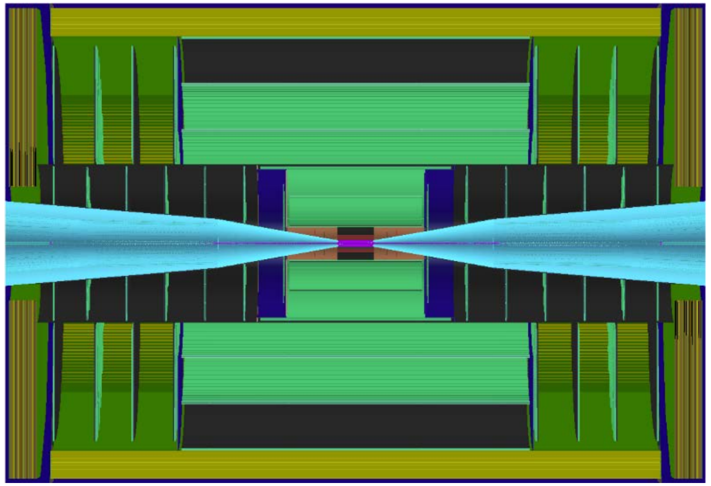
- "Classical" mixing: overlay n_{PU} *distinct* simulated minimum bias events *per bunch crossing* on top of signal event → massively I/O intensive
- "Premixing": perform overlay in advance, save hits after aggregation (digitized format)
  - Leads to O(PB) samples that have to be served throughout the grid with very high availability
  - Better than classical mixing, but still disk- and network-intensive
- Viewed as a solved problem… but substantial room for improvement
  - Generative ML could compress O(PB) samples into O(MB) model + random number generator & conditioning info → *completely eliminate* premixing resource usage (in exchange for training)
- Straightforward to repurpose detector simulation surrogates, but also possible improvements here
  - Train on data and realize long-awaited data mixing?

Future Colliders

- **FCC-hh**: ~100 km ring, $\sqrt{s}_{pp} \approx 100$ TeV
  - Expected pileup 1000: 2.5×10^5 > 100 MeV
  - *Significant escalation* from previous slide
  - More details in CERN-2022-002 (CDR)



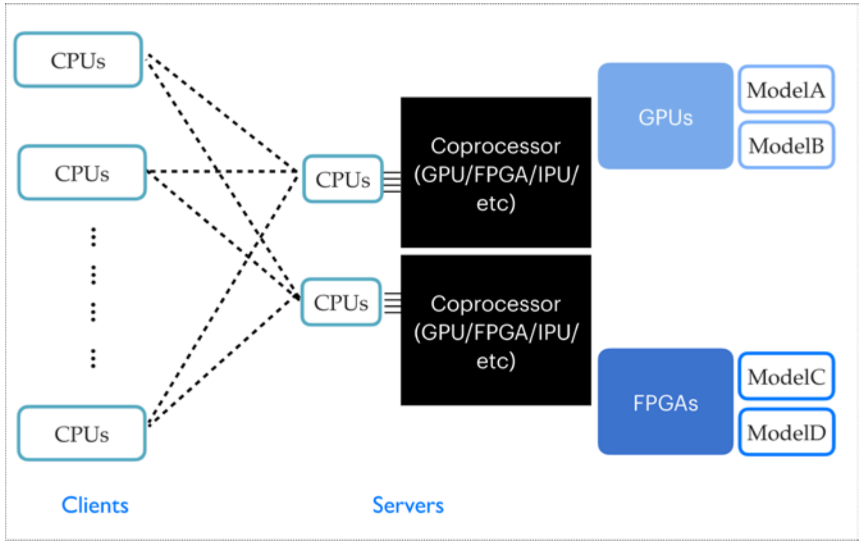
- **Muon Collider**: ~16 km ring, $\sqrt{s}_{\mu\mu} \approx 10$ TeV (≈70–150 TeV in $\sqrt{s}_{pp}$, arXiv:1901.06150)
  - *Beam-induced background*: arXiv:2209.01318
    - ~10^5 muon decays per meter
    - ~10^8 photons and neutrons per crossing
    - ➢ *24 hours* to simulate 1 event in Geant4
  - Designing & optimizing machine-detector interface (e.g. tungsten nozzle) requires substantial intensive simulation

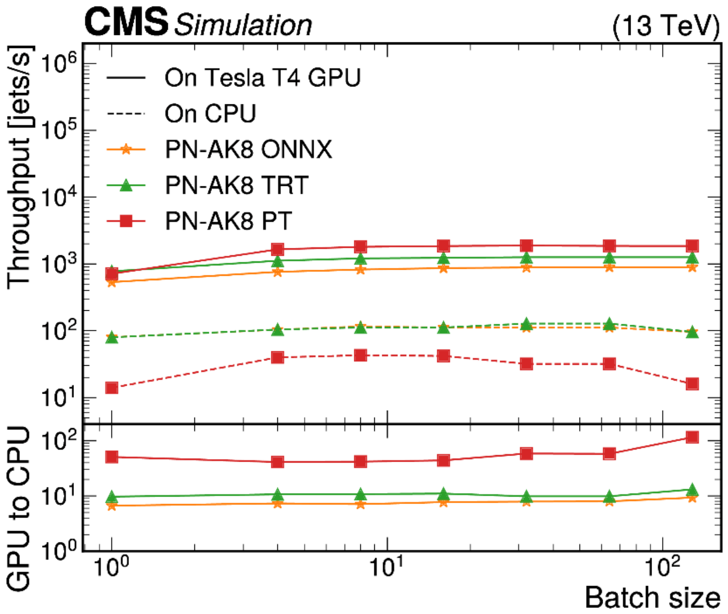


arXiv:2203.07964

Computing for ML

- ML algorithms use a restricted set of operations (mostly matrix multiplications)
  - Natural and easy to accelerate on specialized coprocessors
- *Most flexible* approach: inference as a service
  - Abstract away specific computing elements: client makes request, server delivers
  - Example: ParticleNet 10–100× faster on GPU vs. CPU
    - Algorithm latency becomes essentially *invisible* with asynchronous calls in offline processing
    - Can batch *across events* for optimal GPU utilization → maximize throughput
- Demonstrated for [CMS](), [protoDUNE](), [LIGO](), [analysis facilities]()
  - Use CPUs, GPUs, FPGAs, TPUs, IPUs… with zero code changes!
  - Optimally exploit new GPU-based High Performance Computing (HPC) facilities





Conclusion

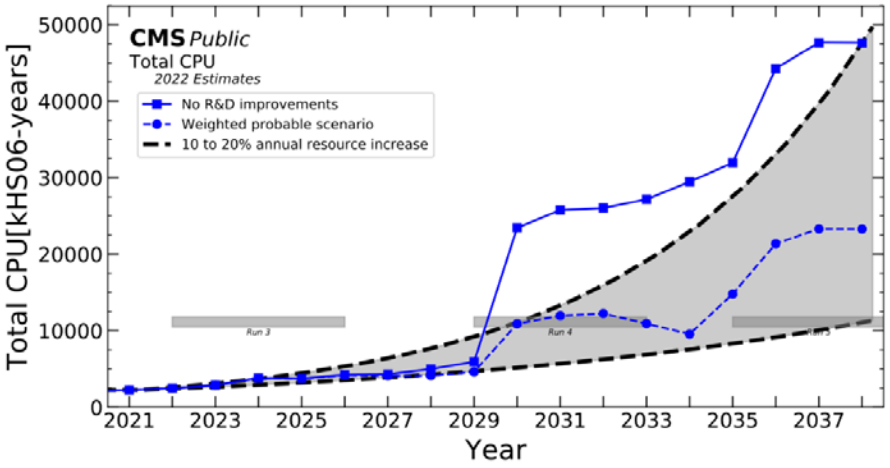
- Growing usage of AI/ML methods for detector simulation
  - Both generative models and non-generative classification/regression techniques are useful
- Increasing focus on resolving *practical problems*: improve both *accuracy* and *computing time*
  - Implementing in common or experiment software frameworks
  - Using ML at production scale – beyond proof of concept
- Applications throughout HEP
  - Primarily investigated for collider physics so far
  - Neutrino and astrophysics starting to see more adoption
- Diffusion models particularly powerful
  - Techniques like flow matching poised to unify normalizing flows and diffusion models
- Many more novel applications than could be discussed here
  - SIM reviews: arXiv:2203.08806, arXiv:2312.09597
  - Overall: HEPML-LivingReview

Background generated by SDXL 1.0 w/ prompt: “A GEANT4 simulation of a pion shower with energy 100 GeV in the Compact Muon Solenoid High Granularity Calorimeter at the CERN Large Hadron Collider, a particle physics experiment”

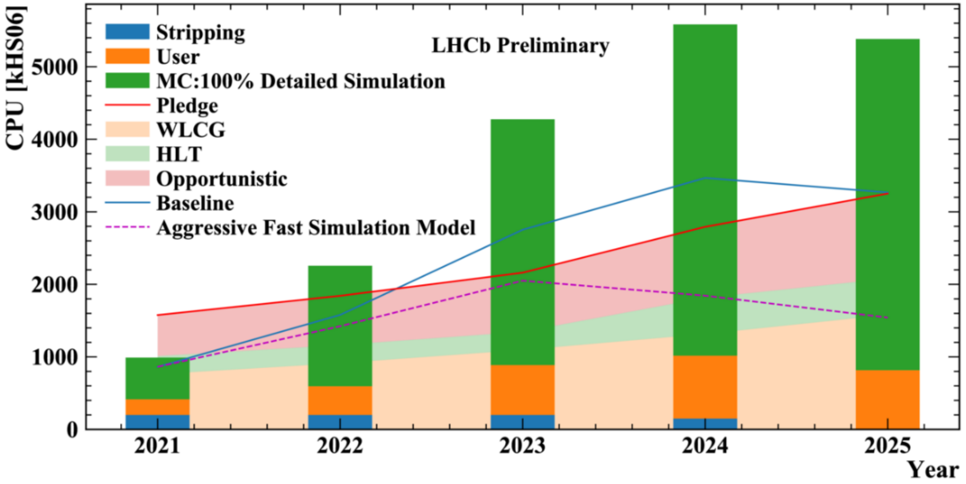
Backup

Projections

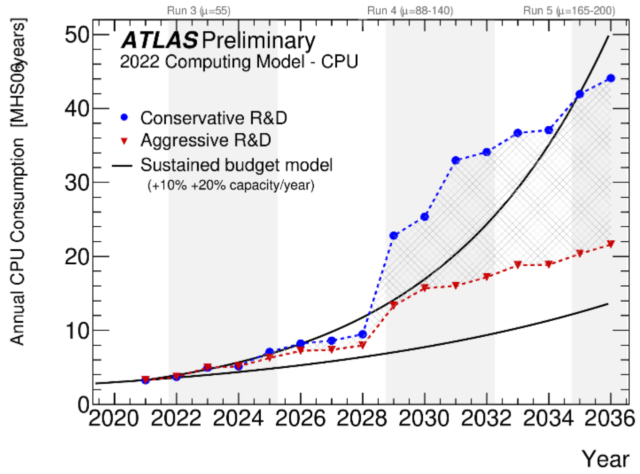
CMSOfflineComputingResults

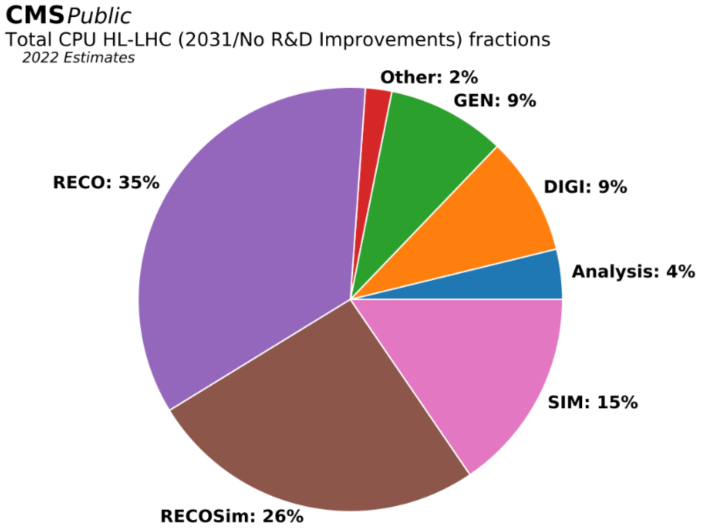


LHCb CPU Usage Forecast

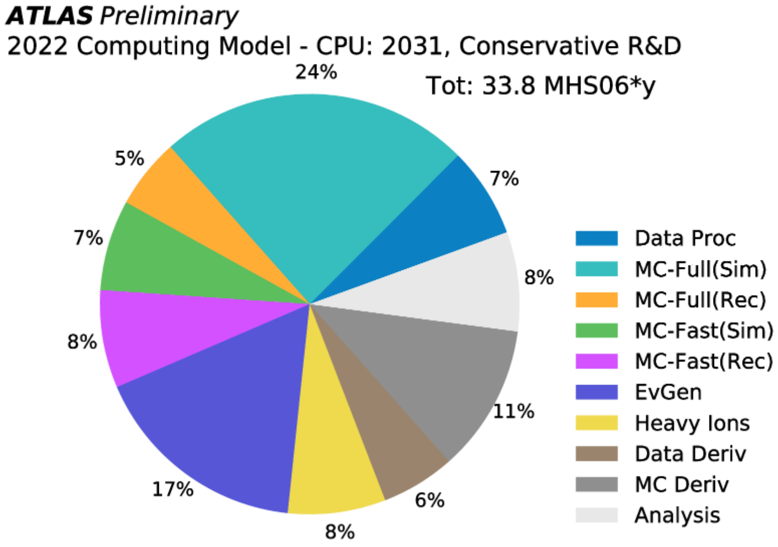


CERN-LHCC-2022-005

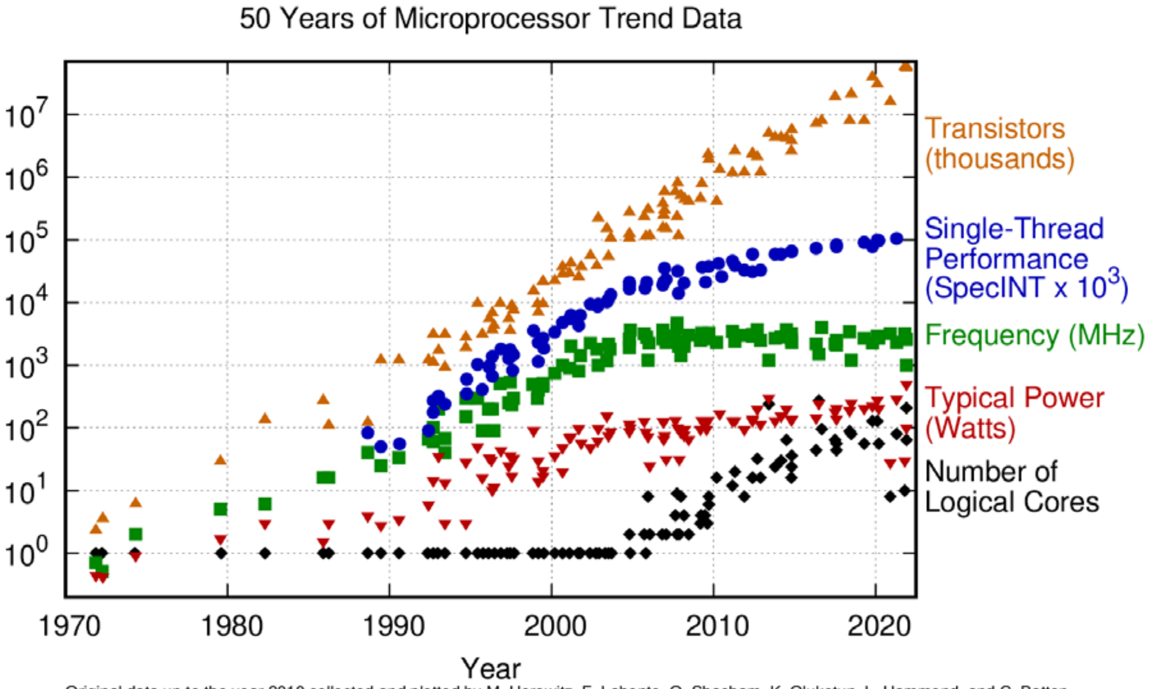




- Run 2: (full) simulation used ~40% (plurality) of grid computing resources for CMS & ATLAS [arXiv:1803.04165]
  - 70% for LHCb! [LHCb-PUB-2022-010]
- Run 4+: limit to ~10–20%, while simulating:
  - Complex detector upgrades
    - e.g. CMS High Granularity Calorimeter
  - More precise physics models
  - More events to reduce statistical uncertainty



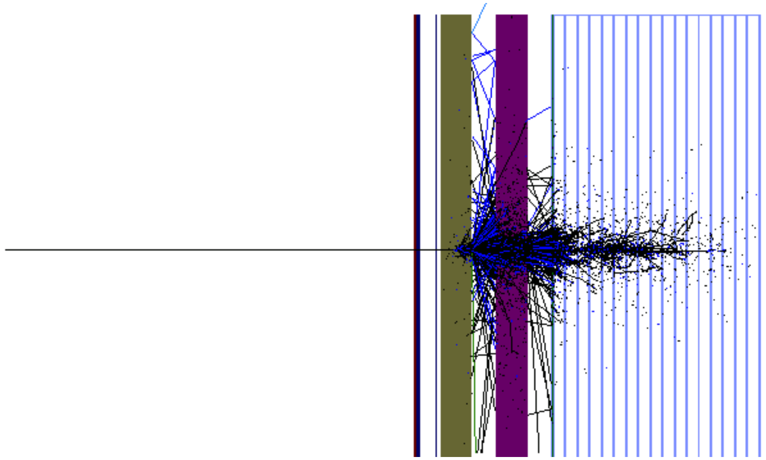
Processors: Old and New



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

- **CPUs**: Moore's Law continues, but Dennard scaling has broken down → stagnant performance/thread
- ➢ Heterogeneous **revolution**: rise of *specialized coprocessors* attached to general-purpose CPUs
  - **GPUs** (SIMD), **FPGAs** (spatial computing), **ASICs**
  - *Growing taxonomy*: even more specialized processors emerging, e.g. **IPUs** (MIMD for ML)
- *Deep learning* uses limited set of mathematical operations: perfect for acceleration on GPUs etc.
  - *Inference as a service*: most general/abstract way to offload tasks to coprocessors

Simulation Landscape

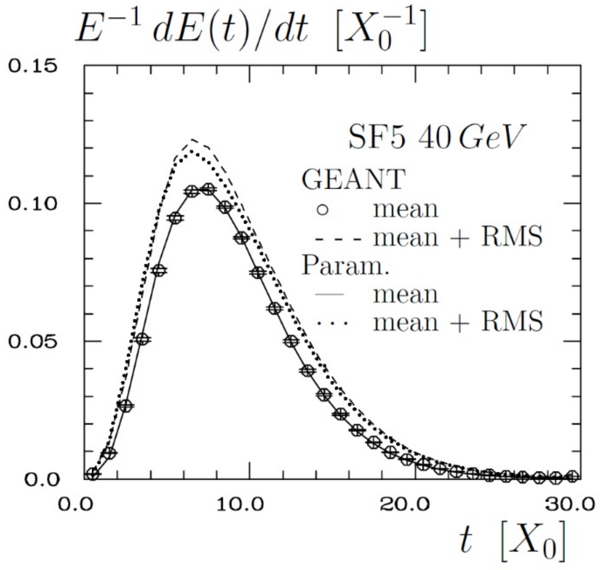


“FullSim”

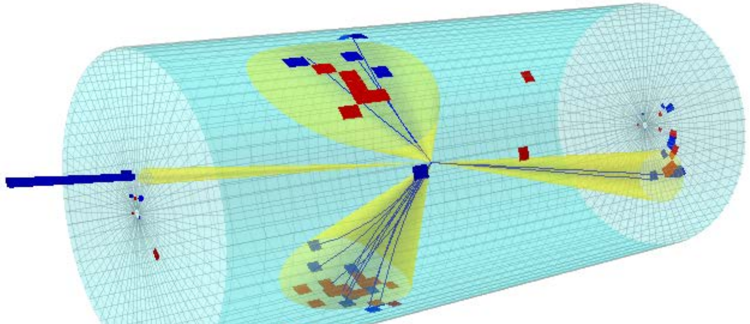
- Common software framework (i.e. Geant4)
  - Experiments can provide additional code via user actions
- Explicit modeling of detector geometry, materials, interactions w/ particles

“FastSim”

- Usually experiment-specific framework
- Implement approximations: analytical shower shapes (e.g. GFLASH), truth-assisted track reconstruction, etc.



[arXiv:hep-ex/0001020](https://arxiv.org/abs/hep-ex/0001020)

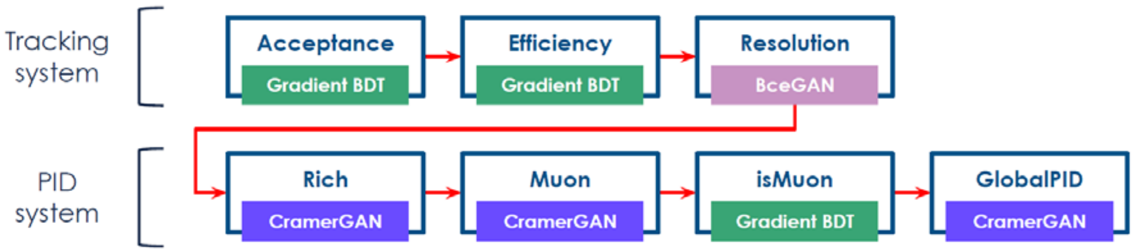


[M. Selvaggi]()

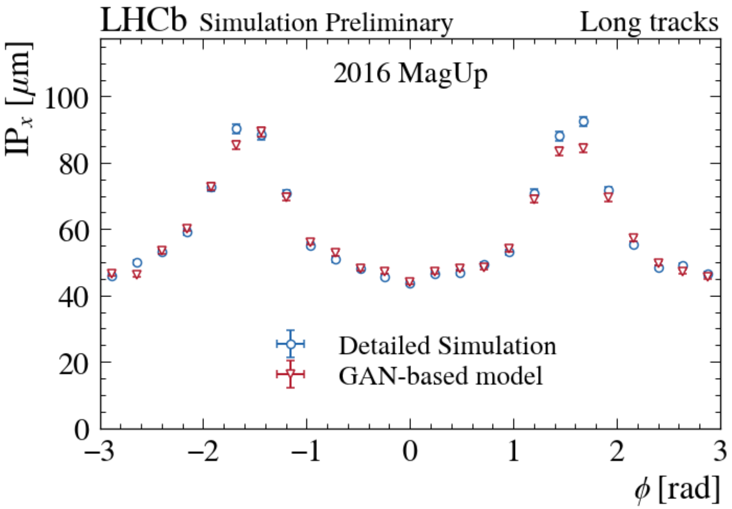
Delphes

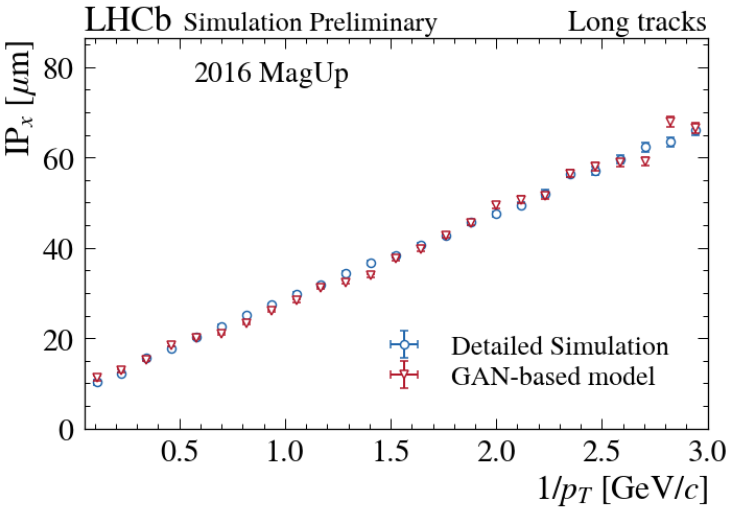
- Ultra-fast parametric simulation
- Used for phenomenological studies, future projections, etc.

Generative Models at Colliders: LHCb

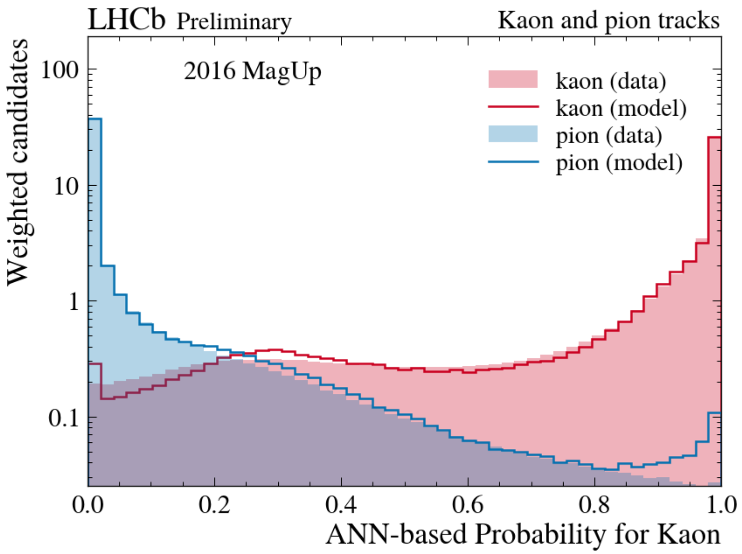


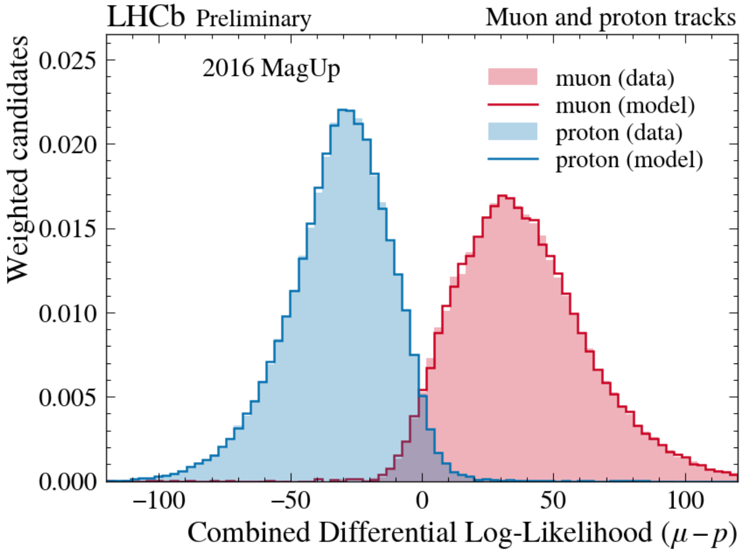
- "Stacked GAN" approach to parameterize different detector aspects
  - o Cramér distance related to W_1
- Tracking resolution: well reproduced in p_T & φ





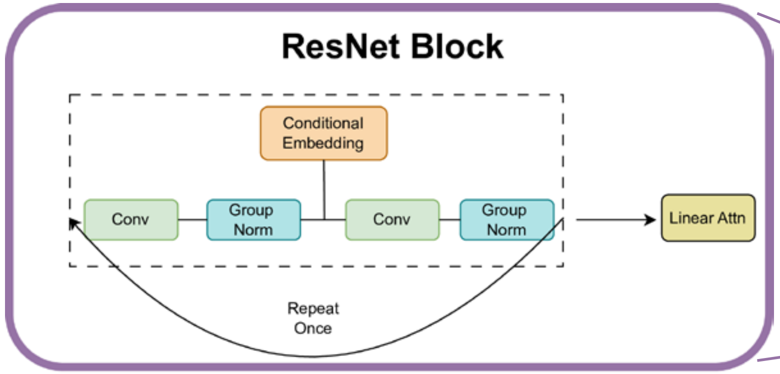
- Global PID variables also well reproduced:
  - o Top: $K^{\pm}$ vs. $\pi^{\pm}$
  - o Bottom: μ vs. p



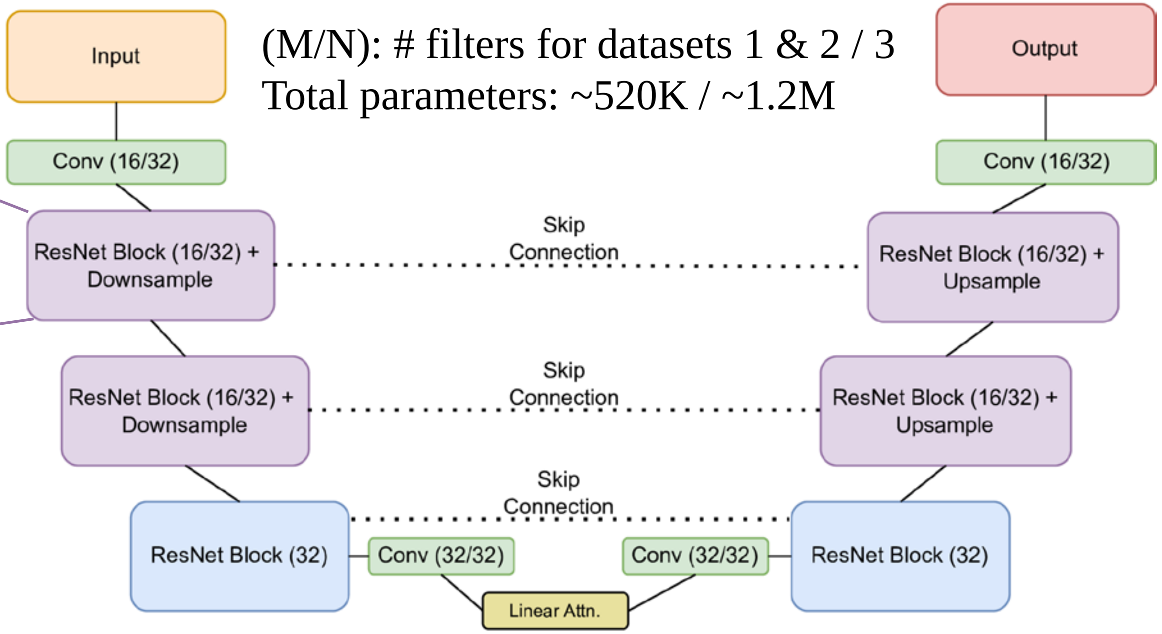


[LHCB-FIGURE-2022-004](#)

CaloDiffusion



(M/N): # filters for datasets 1 & 2 / 3
Total parameters: ~520K / ~1.2M

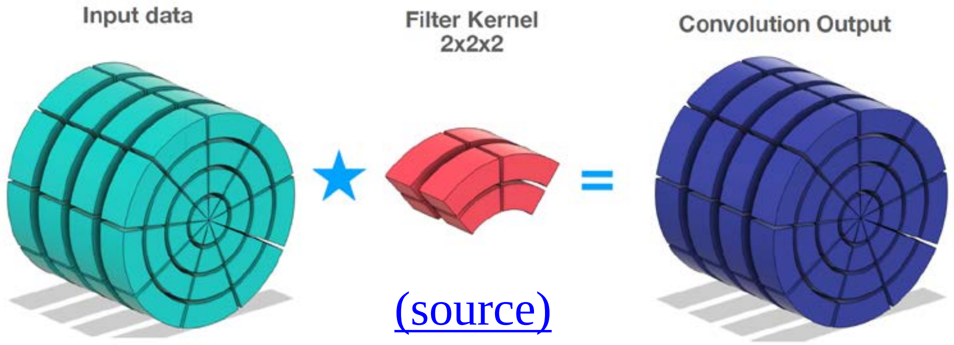


- Base architecture: U-net
  - Skip connections ensure no loss of information
- Linear self-attention layers applied to each convolutional ResNet block
  - Allows dimensionality reduction in *z* to handle longitudinal correlations in showers
- \+ numerous geometric innovations (next slide)
- Cosine noise schedule for training
- Stochastic sampling algorithm for generation

- Objectives:
  - Datasets 1 & 2: predict (normalized) noise
  - Dataset 3: predict weighted average of noise and denoised image
- Aim for highest achievable quality first
  - Then focus on improving speed
  - Wrong answers can be obtained infinitely fast

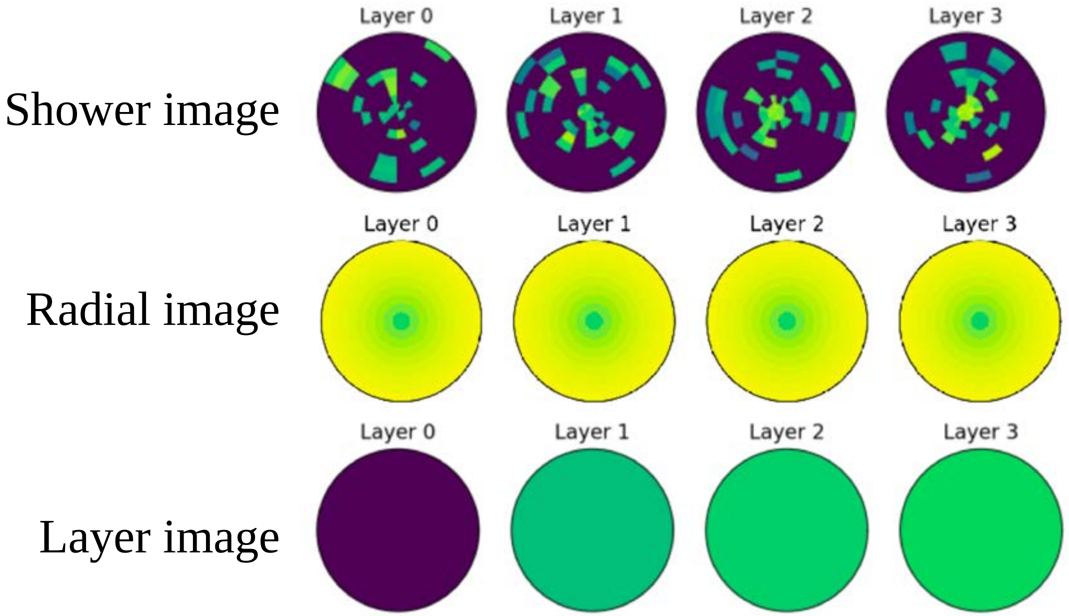
Geometric Innovations

- Particle showers are invariant & periodic in φ
  - Pad in φ so convolutions “wrap around”



[(source)]

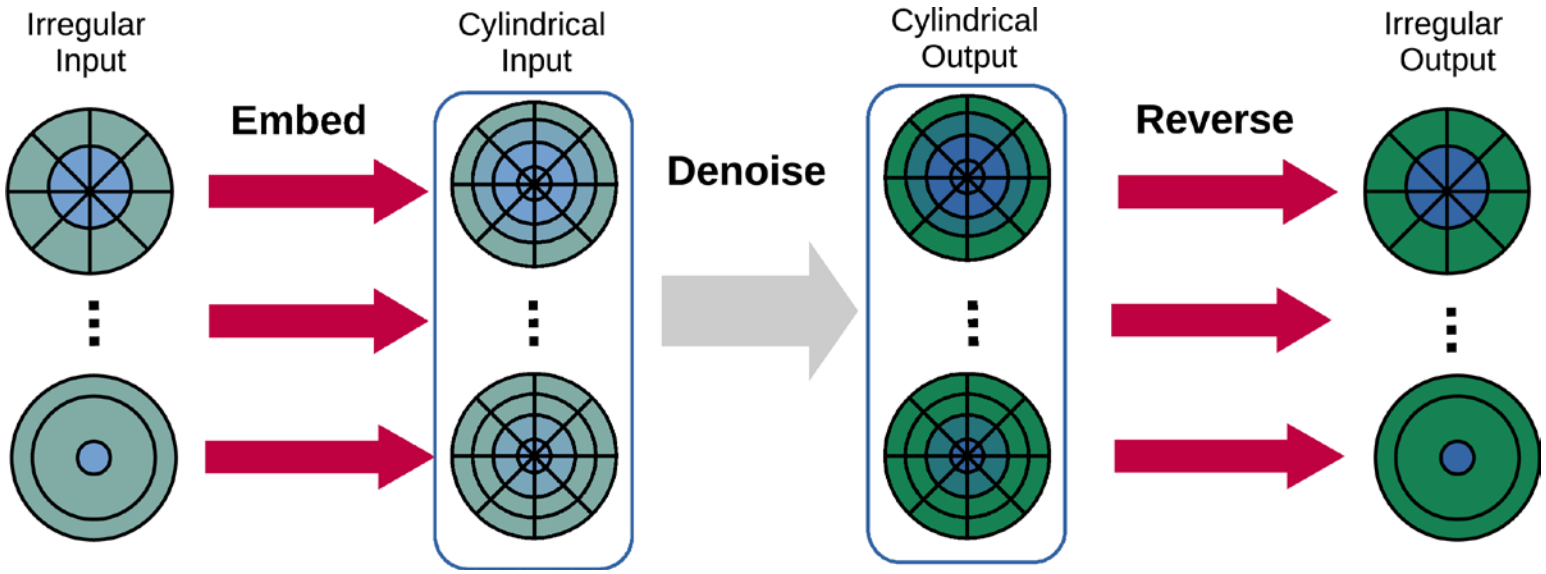
- Particle showers are *not* invariant in *r* or *z*
  - Provide *r* and *z* (layer) as extra per-pixel channels (input features)
  - Convolutions become *conditional*



➢ *Conditional cylindrical convolutions*

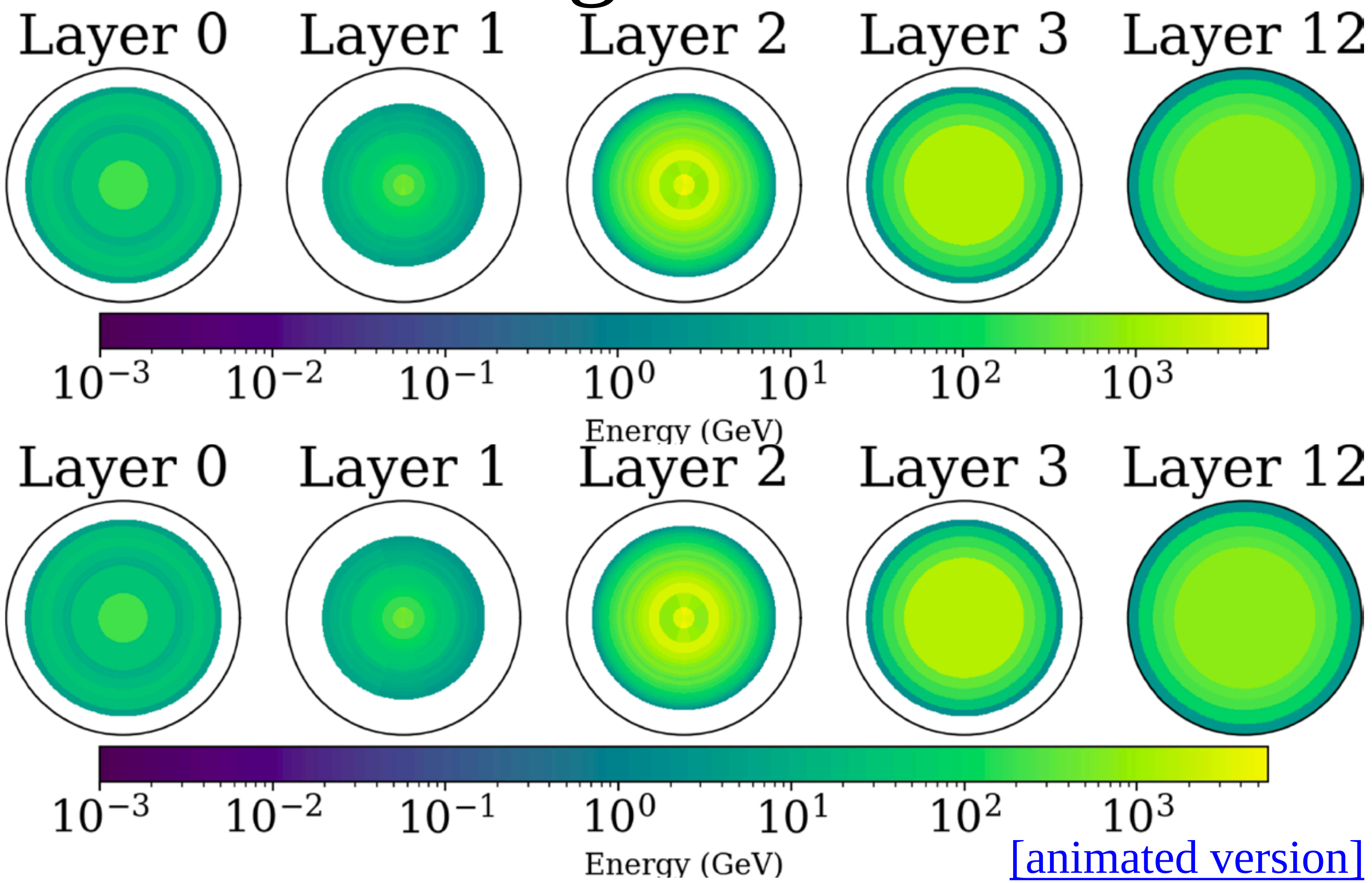
- Handle inherent features of particle detector geometry, distinct from rectangular images

Geometry Latent Mapping: GLaM



- Dataset 1 has different radial/angular bins in each layer
  - Can't directly apply convolutions, which require regular neighbor structure
- Learn forward and reverse embeddings to and from a regular geometry
  - Simple matrices C (NxM) and D (MxN)
    - C initialized to split or merge cells based on overlap between original and embedded geometries
    - D initialized as Moore-Penrose pseudoinverse of C
- Inspired by "latent diffusion" approach
  - But not necessarily lower-dimensional representation; actually higher-dimensional here

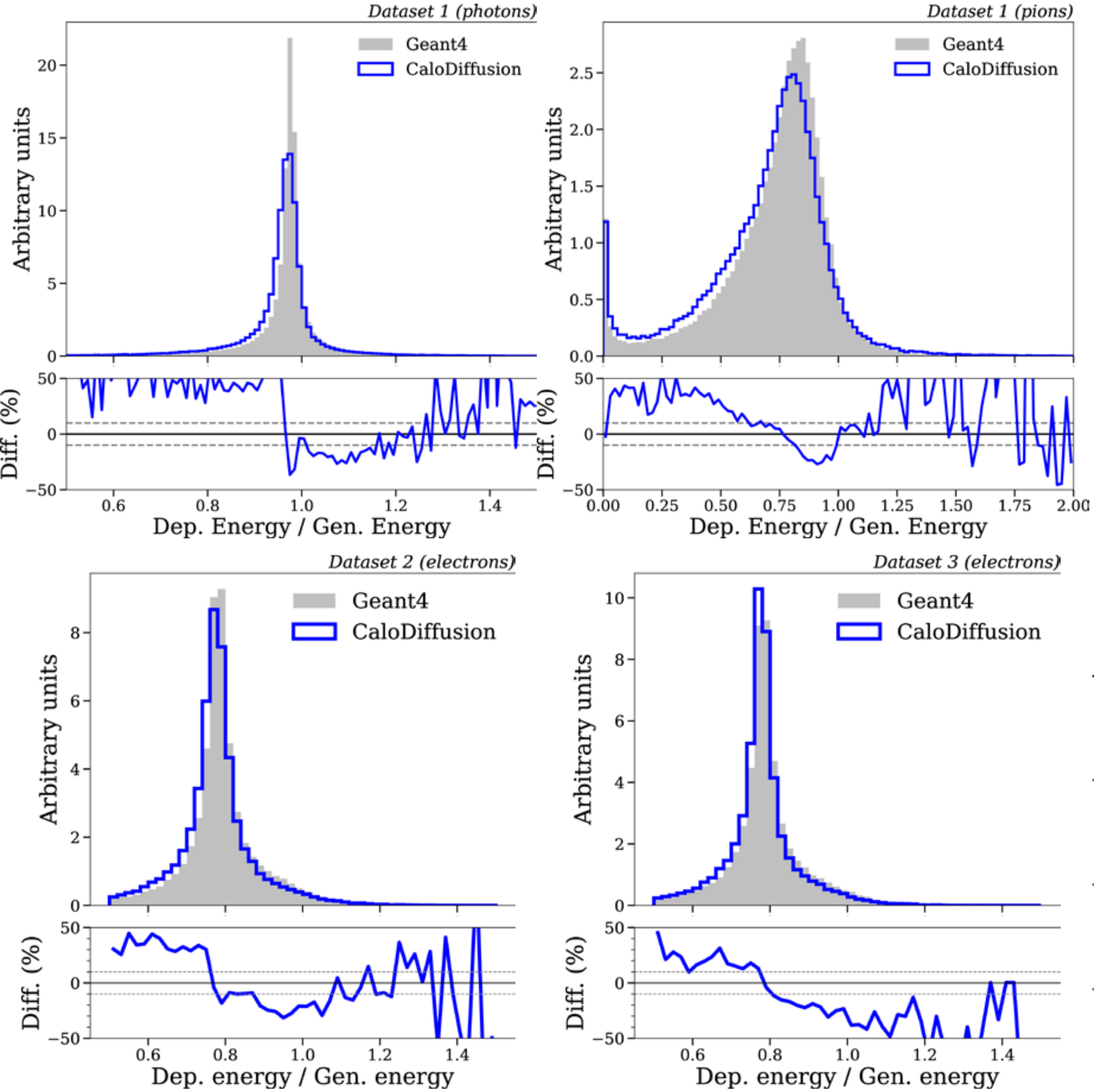
Average Showers



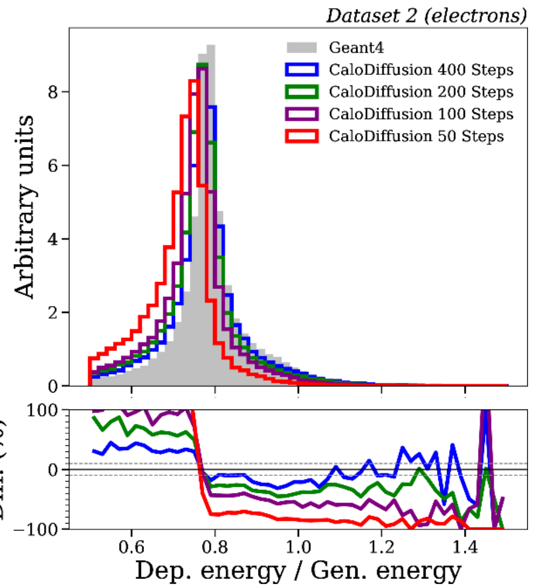
[animated version]

- Top: Geant4; bottom: CaloDiffusion (dataset 1, photons)
  - … or is it the other way around? Can you tell?

Original CaloDiffusion: Areas for Improvement



- Deficit in total energy modeling
- Need 400 diffusion steps to get acceptable quality
  - o Still faster than Geant4 (~100s) w/ batching on GPU
- Fewer steps:
  - o Linear speed improvement
  - o But even less accurate in this quantity



Dataset	Batch Size	Time/Shower [s] CPU	Time/Shower [s] GPU
1 (photons)	1	9.4	6.3
(368 voxels)	10	2.0	0.6
	100	1.0	0.1
1 (pions)	1	9.8	6.4
(533 voxels)	10	2.0	0.6
	100	1.0	0.1
2 (electrons)	1	14.8	6.2
(6.5K voxels)	10	4.6	0.6
	100	4.0	0.2
3 (electrons)	1	52.7	7.1
(40.5K voxels)	10	44.1	2.6
	100	-	2.0

Num. Steps	Classifier AUC (low / high)	FPD	E Ratio Sep. Power
400	0.56 / 0.55	0.043(1)	0.011
200	0.61 / 0.56	0.046(1)	0.036
100	0.69 / 0.59	0.065(3)	0.079
50	0.83 / 0.67	0.110(4)	0.251