Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Istituto Nazionale di Fisica Nucleare
SEZIONE DI FIRENZE

ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Generative Models at the LHC

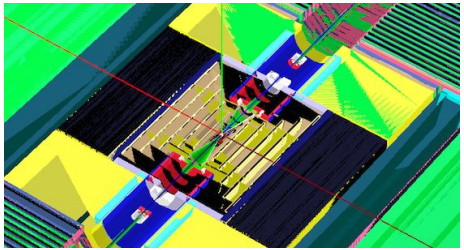
Lucio Anderlini

Trento, November 23rd, 2023

ALPACA WORKSHOP

Why do we need simulation?

Design detectors and experiments

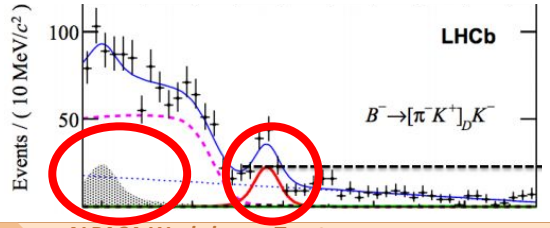


```
P/export_scikinC.py  P/pipeline.C  P/trainMuon.py
import sys

from FastQuantileLayer import FastQuantileLayer
sys.path.append("/pclhcb06/landerli/LamarrGanTrainer/
from AddRandomFeatures import AddRandomFeatures


##################################################
def retrieve_n_features (model_path, loaded_model):
    h5 = h5py.File(os.path.join(model_path,'keras_ge
    layer = loaded_model.layers[2]
    kernel_shape = h5[f"{layer.name}/sequential/dens
    bias_shape = h5[f"{layer.name}/sequential/dense/
```

Design selection strategies (e.g. for the trigger)

Evaluate selection efficiencies for **physics signal** and **backgrounds**

Build **statistical models** for physics contributions



Event size of the LHC experiments

Different physics

Different experiments

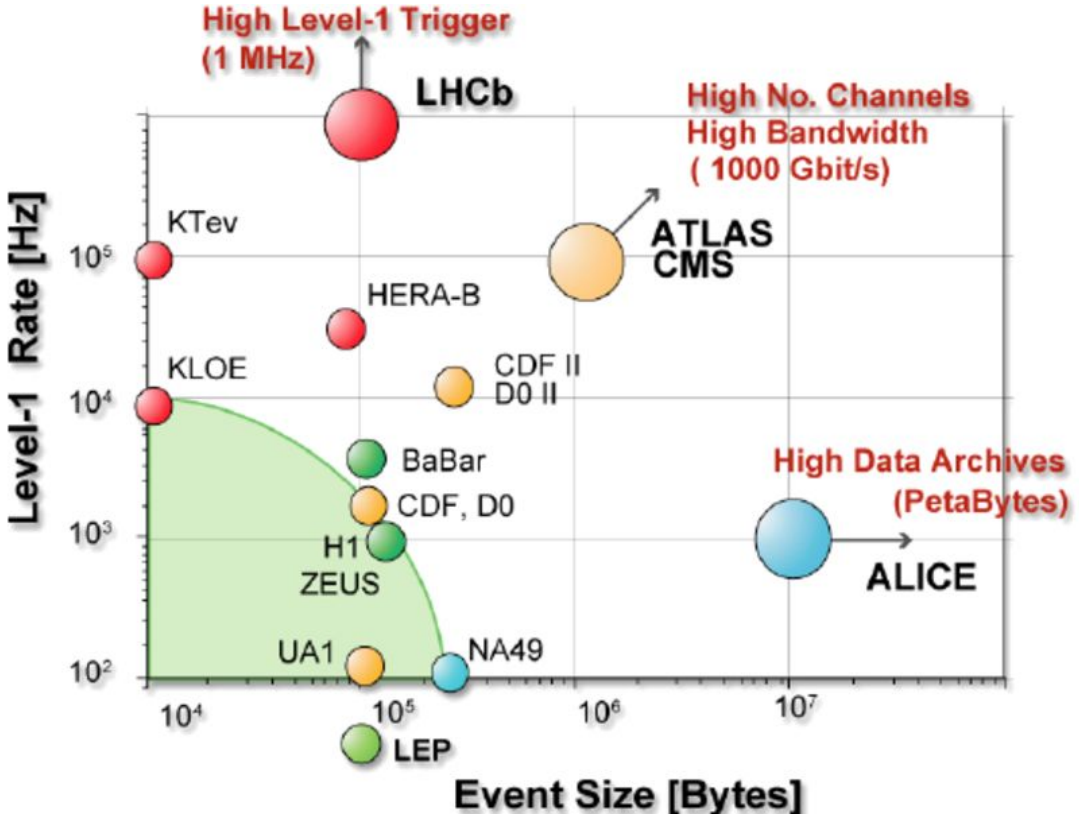
Different data processing

Different data formats

Different needs in terms of simulation.

Same problem:
simulation costs too much

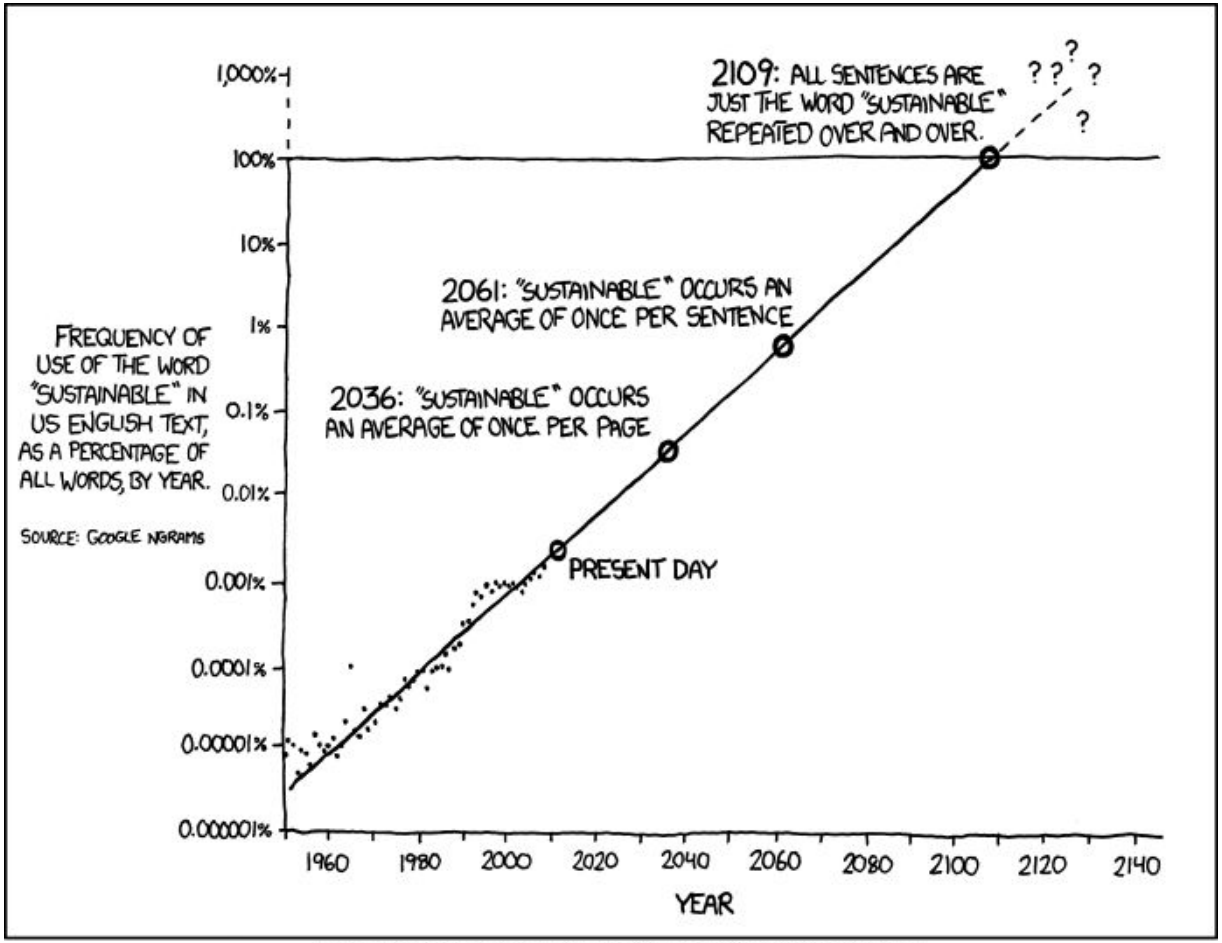
Different solutions



Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Analysing larger datasets requires larger (or at least more and diverse) **simulated samples.**

The proportionality between the analysed luminosity and the simulation requests has been observed during Run1 and 2.

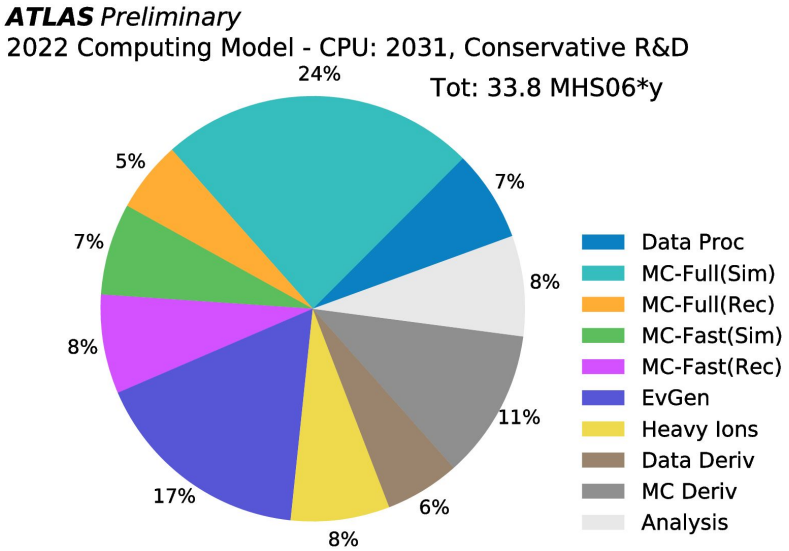


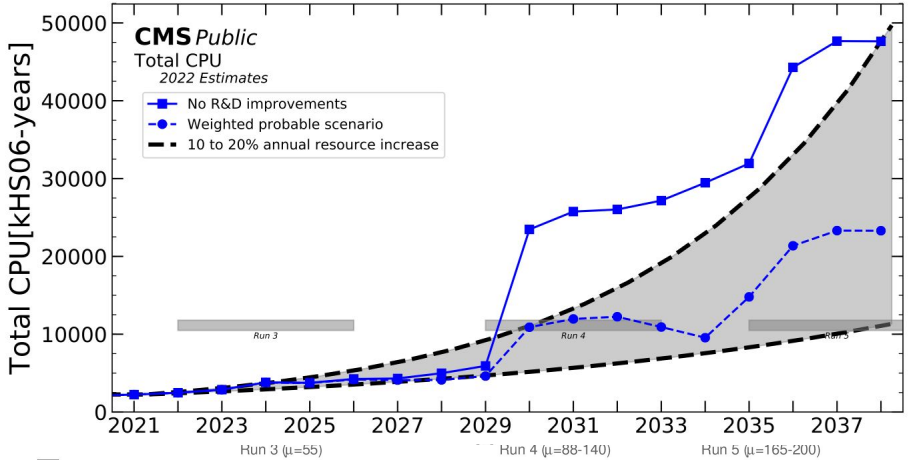
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

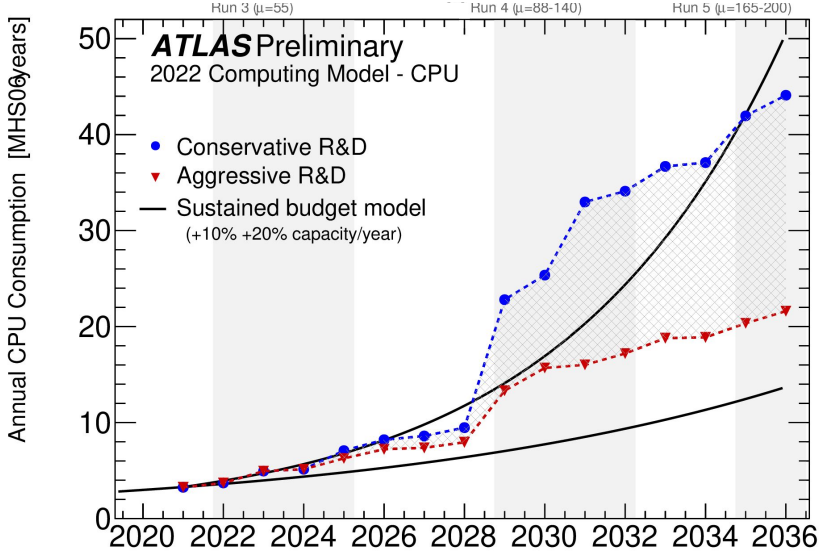
The simulation challenge

Aggressive R&D is needed to decrease the impact (read the money) to obtain the necessary simulated samples.

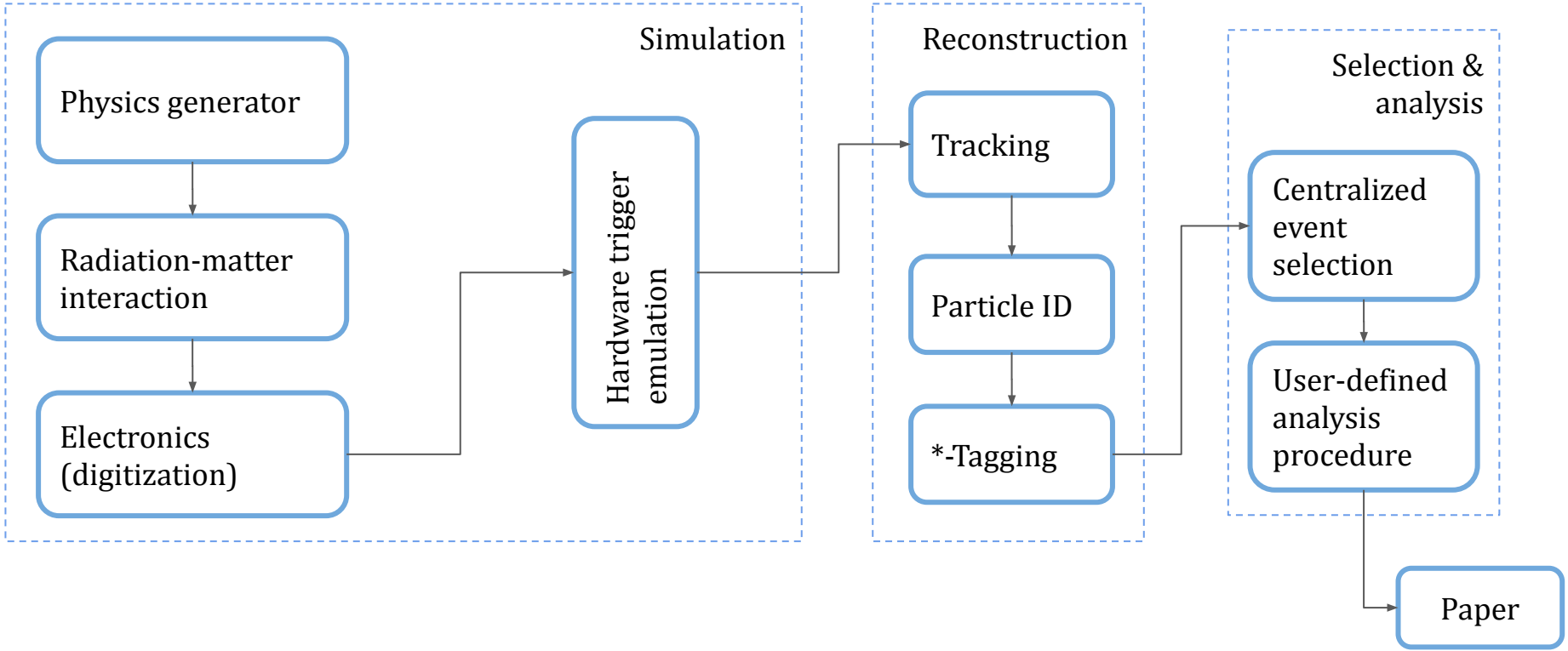
It goes under the name of ***Fast Simulations***







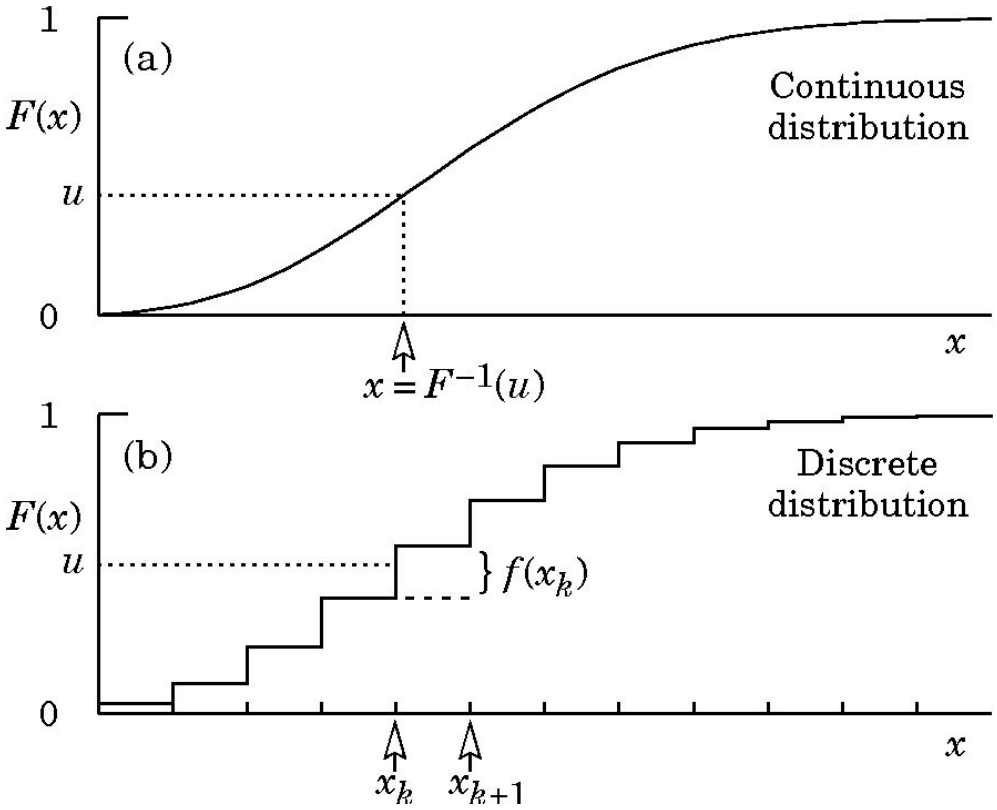
Standard simulation: the big picture



Parametric simulation

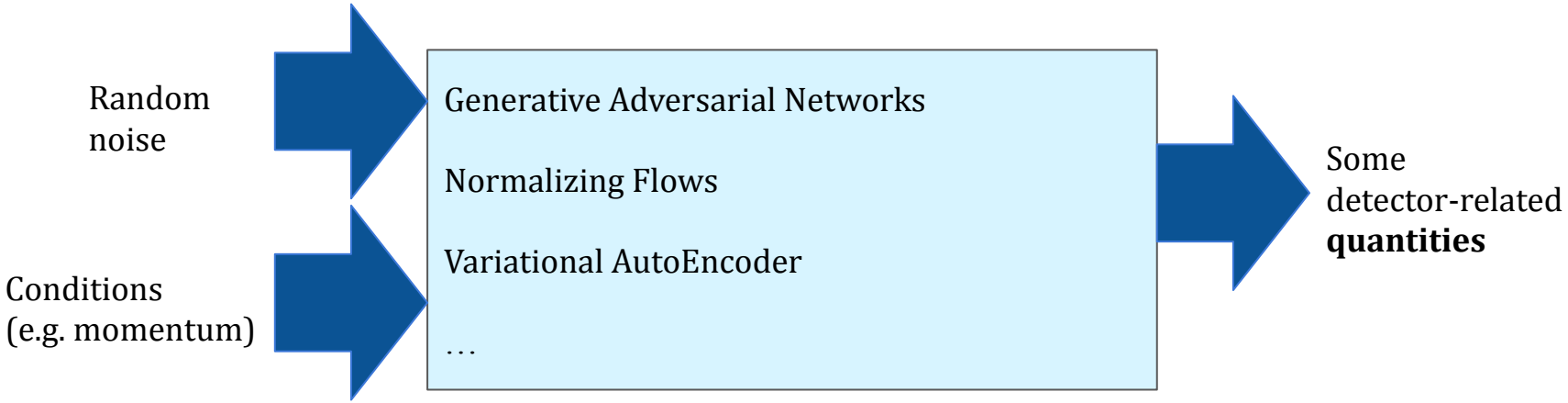
We know from statistics that we can generate samples according to a given distribution, for example, through the inverse CDF method.

We use machine learning to learn a multidimensional equivalent of the inverse of the cumulative *F of the target distribution*
as a function of some parameter, e.g. the momentum of a particle.

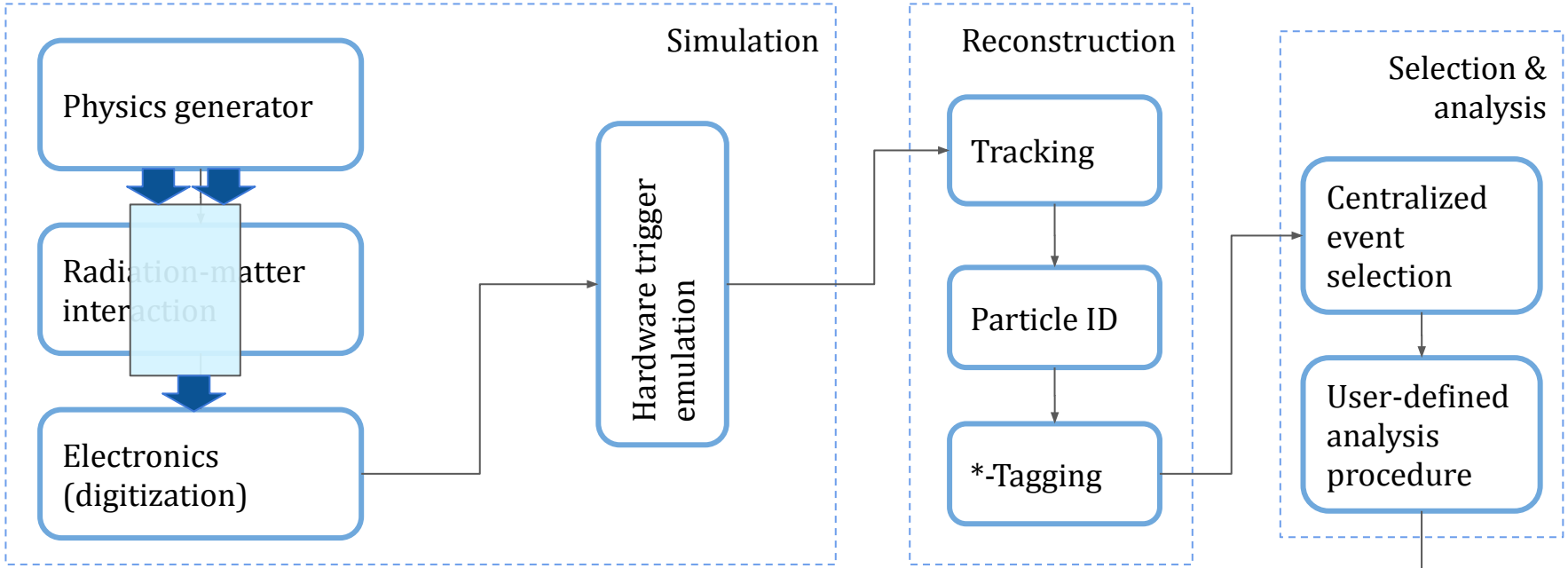


Our ML building block

$$\underbrace{x}_{\text{Target variable}} = F^{-1}\left(\underbrace{u}_{\text{Random noise}}; \underbrace{p, \eta, \dots}_{\text{Conditions}}\right)$$



Approaches to ML in simulation: *Speed-up Geant*



Only replace with ML parts of the radiation-matter interaction, in the showers.

S. Vallecorsa *et al.* EPJ Web Conf., 214 (2019) 02010

👉 see William Korcari's talk

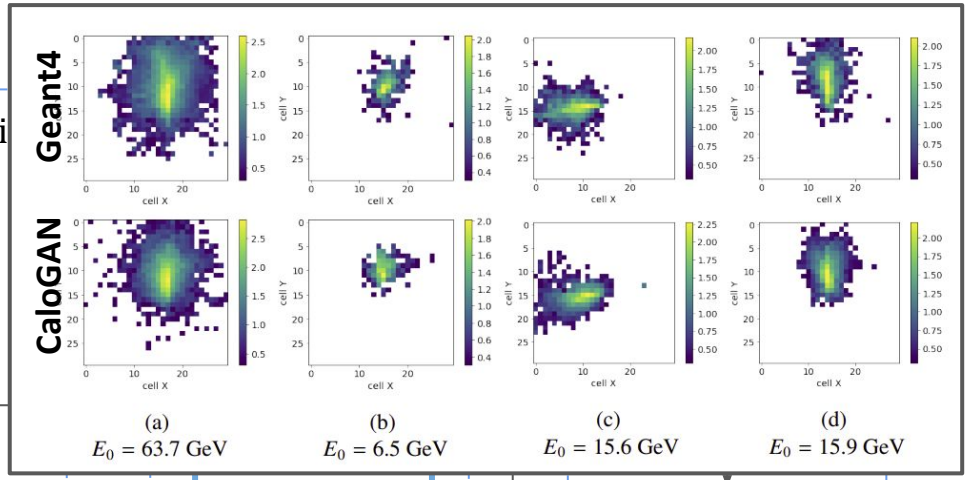
Approaches to ML in simulation: *Fast Simulation*

Physics generator

Radiation-matter interaction

Electronics (digitization)

Hardware trigger emulation



*-Tagging

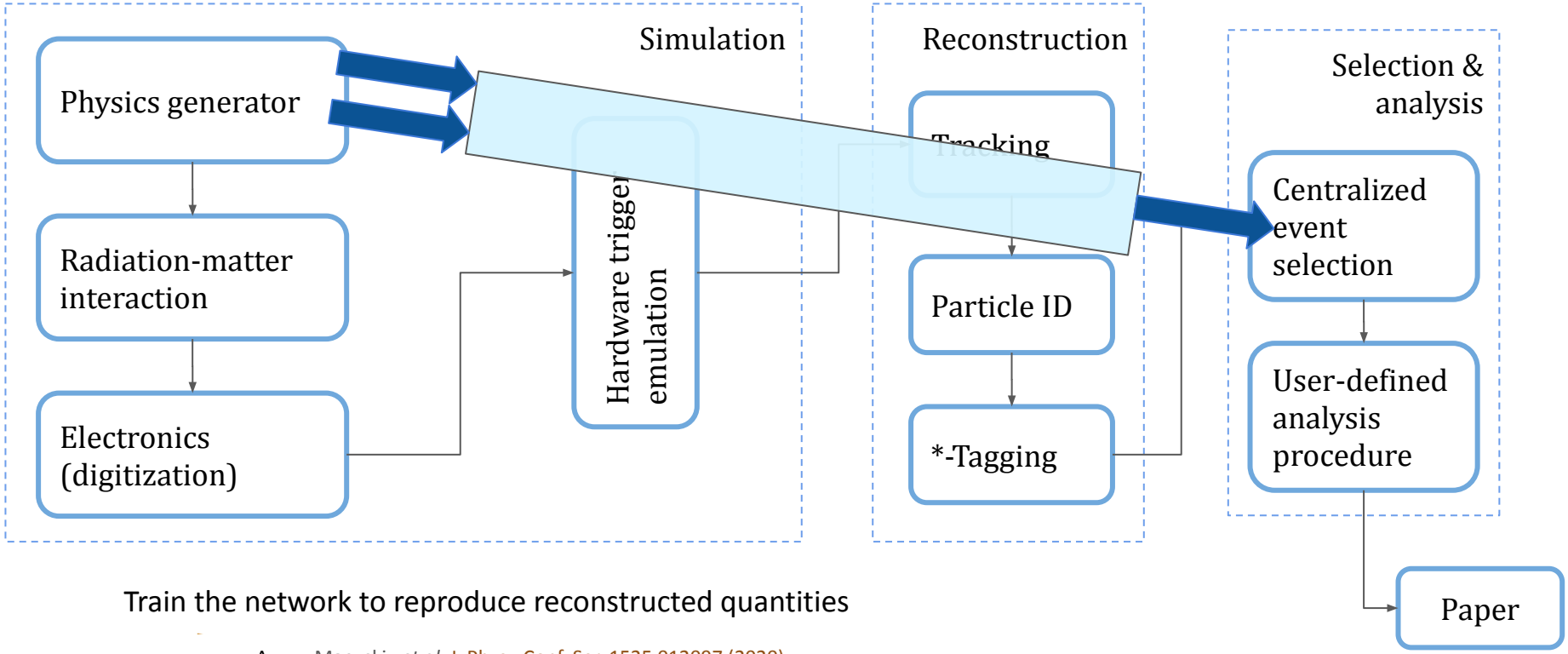
User-defined analysis procedure

Paper

Take advantage of the granularity of the calorimeter readout to simplify the network

F. Ratnikov *et al.* EPJ Web Conf., 245 (2020) 02026

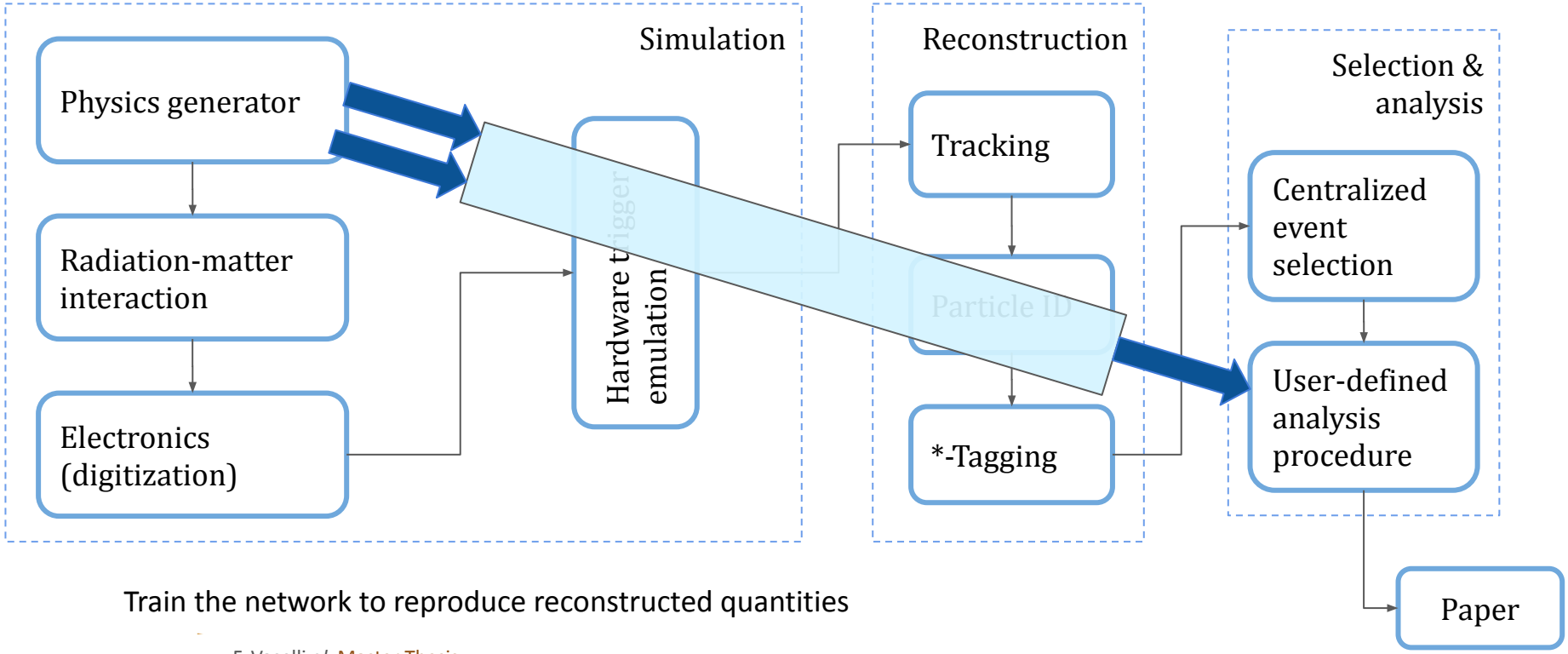
Approaches to ML in simulation: *Ultra-Fast Simulation*



Train the network to reproduce reconstructed quantities

A. Maevskiy *et al.* J. Phys.: Conf. Ser. 1525 012097 (2020)

Approaches to ML in simulation: *Flash Simulation*



Train the network to reproduce reconstructed quantities

F. Vaselli *al.* Master Thesis

How to choose? *It depends…*

Lower-level parametrization (Fast simulation):

- enabable to run (a fraction of) the experiment reconstruction software: ***all variables/concepts will be available to the final user***
- are usually ***very tricky to train*** because we wish the reconstructed quantities, rather than the output ot the model, will be compatible with full simulation
- **modest speed-up**

Higher-level parametrization (Flash simulation):

- **only simulates variables needed in the final analysis**
- ***training is made easier*** by the availability of the metrics on the relevant variables
- ***major speed-up is possible***

Two intermediate ideas from LHCb

Auxiliary regressor [ACAT`21, 2207.06329]

One can push Fast ⟶ Flash Simulation by training the generative model to be aware of the reconstructed quantities.

This is achieved by training an additional neural network to emulate the reconstruction procedure as a differentiable function, and including it in the training procedure of the generative model.

Ultra-fast simulation framework [ICHEP`22]

Push Flash ⟶ Full Simulation by parametrizing with different “flash simulations” different high-level objects (particles, vertices, particle IDs).

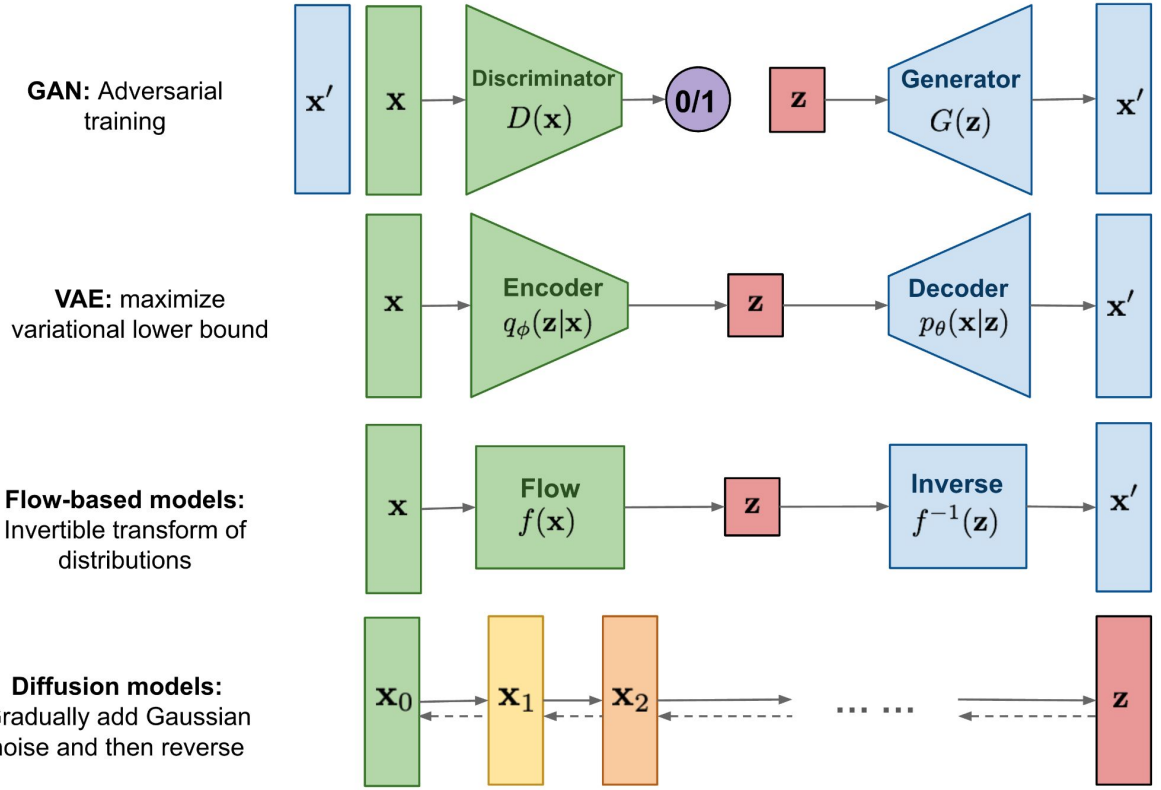
The analyst will still need to combine those objects to “interpret” the event by reconstructing particle decays with custom strategies.

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Foundational models

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Fundational models: *GAN, VAE, NF and Diffusion models*



Scheme stolen from Lilian Weng's blog post

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Common building blocks!

Latent space

Generative Models

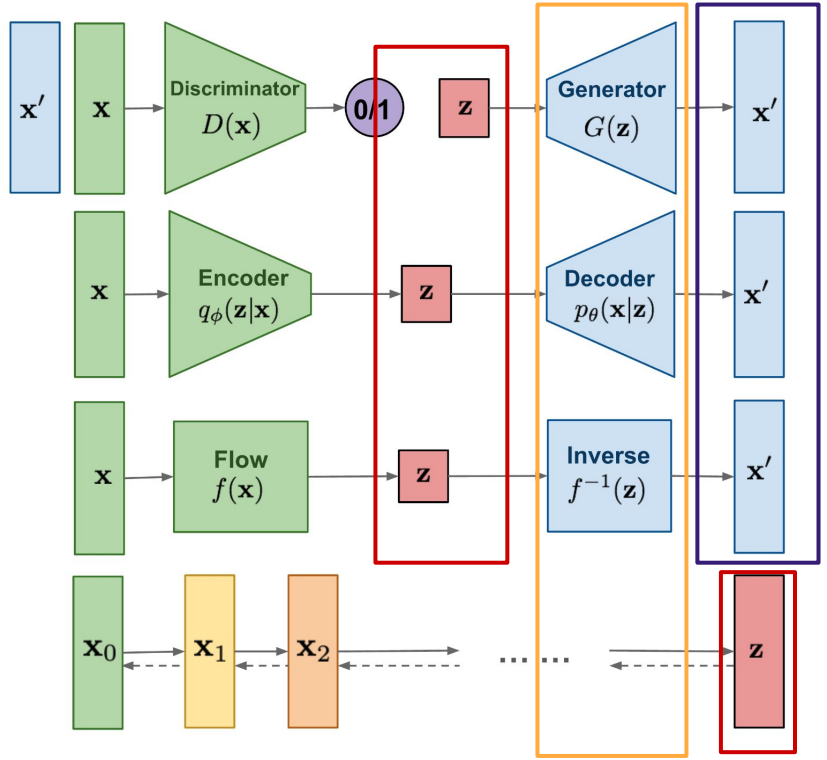
Synthetic data

GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

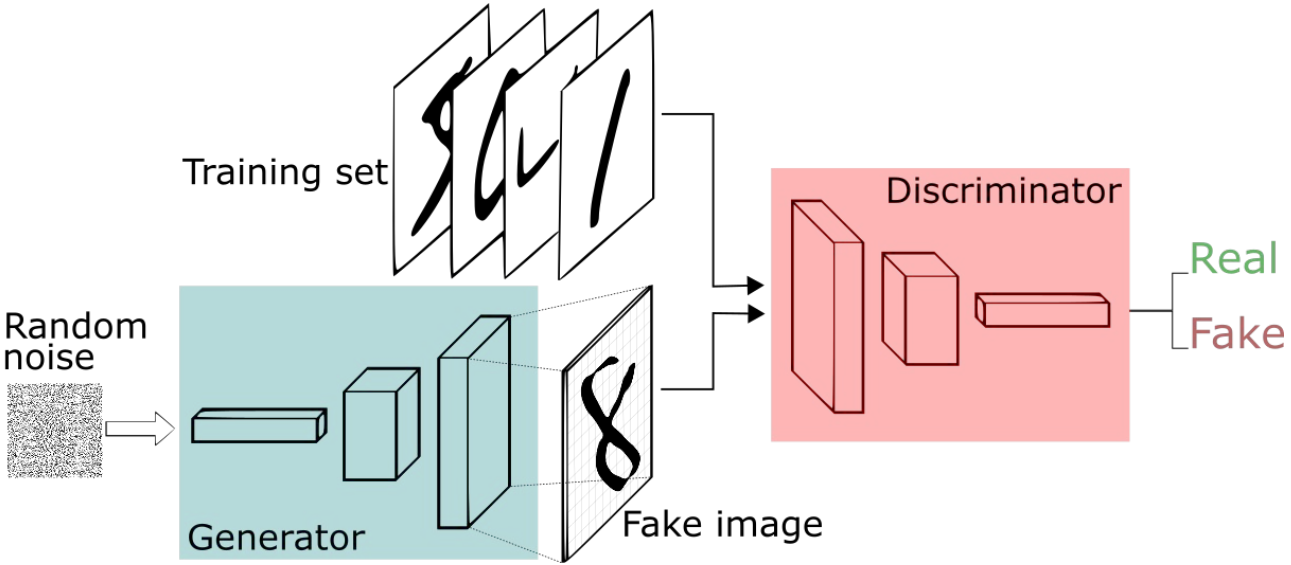
Diffusion models: Gradually add Gaussian noise and then reverse



Generative adversarial networks: game-theory inspired training dynamic

Discriminator D: estimates the probability of a given sample coming from the real dataset

Generator G: outputs synthetic samples given a noise variable input (brings in stochasticity)



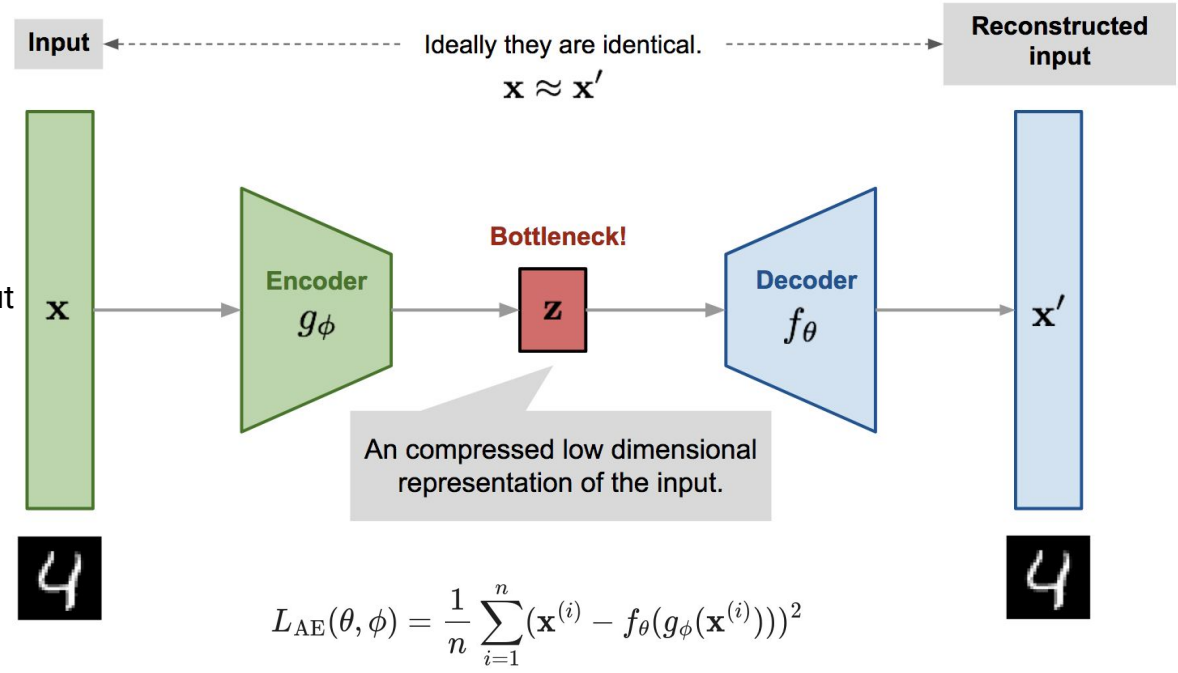
$$\min_G \max_D L(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$
$$= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1-D(x)]$$

Autoencoders can be a great starting point

Encode **x** into low dimensional representation!

Encoder network: It translates the original high-dimension input into the latent low-dimensional code. The input size is larger than the output size.
Decoder network: The decoder network recovers the data from the code

Why can't I use this for generation?



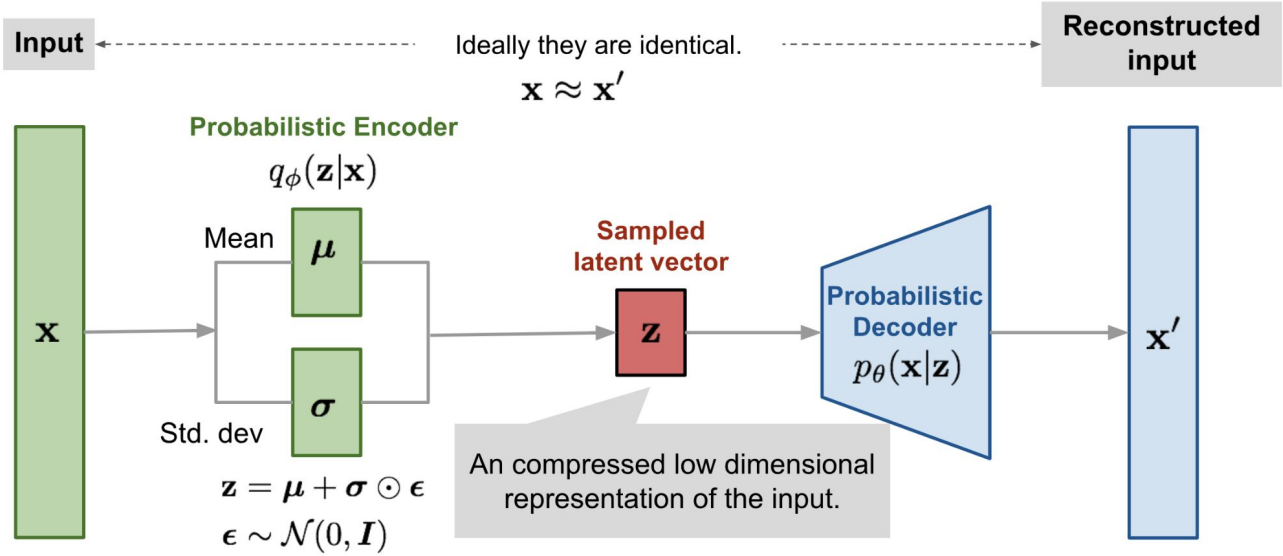
$$L_{\text{AE}}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}^{(i)} - f_\theta(g_\phi(\mathbf{x}^{(i)})))^2$$

Reconstruction Loss

Finally, a *variational* autoencoder!

We can produce new, original data samples z with arbitrary mean and sigma!

- **Latent Space Representation**: meaningful, smooth latent space representations, great for tasks like anomaly detection or data compression.
- **Flexibility**
- **Interpolation**: VAEs can interpolate between data points in a meaningful way in the latent space.



The basic idea: change of variables

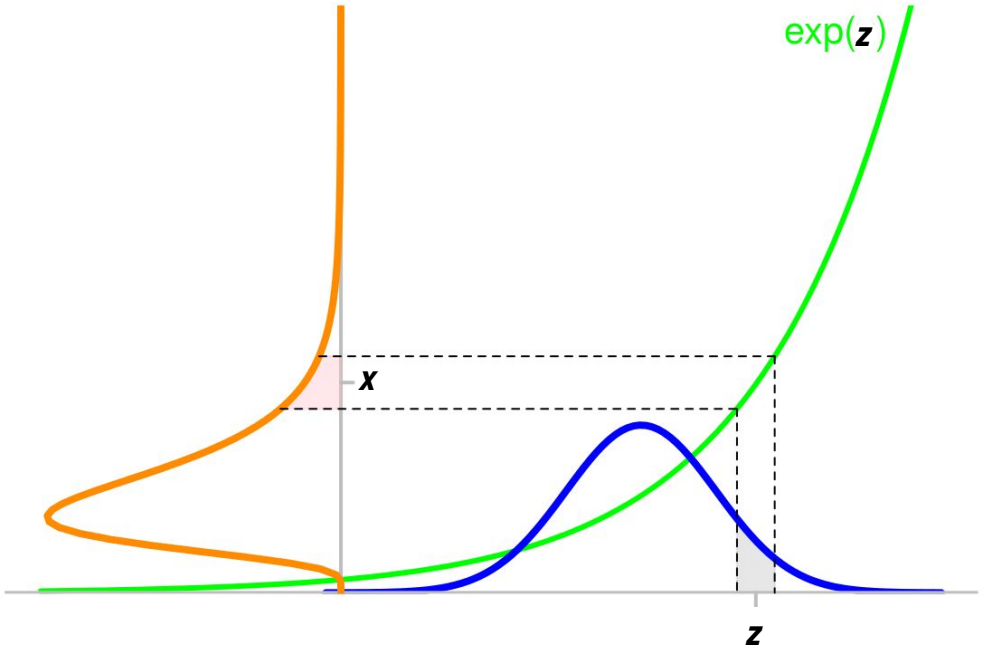
We define a transform *f* such that:

$$\mathbf{x} = f(\mathbf{z})$$
$$\mathbf{z} = f^{-1}(\mathbf{x})$$

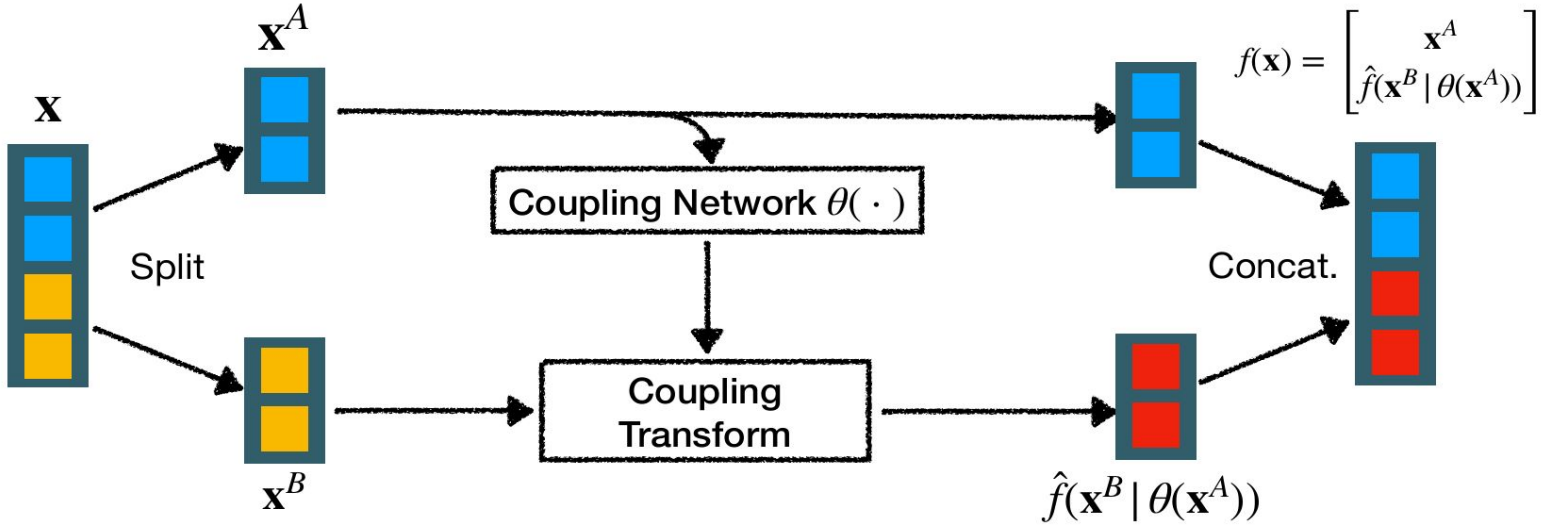
The two pdfs are related:

$$p_x(\mathbf{x})d\mathbf{x} = p_z(\mathbf{z})d\mathbf{z}$$

$$p_x(\mathbf{x}) = p_z(f^{-1}(\mathbf{x}))\,\det\left|\frac{d\mathbf{z}}{d\mathbf{x}}\right|$$



Coupling layers are a way of addressing jacobian complexity



$$\mathbb{J}_f(\mathbf{x};\,\phi) = \begin{pmatrix} \frac{\partial \vec{x}_{1:d}}{\partial \vec{z}_{1:d}} & \frac{\partial \vec{x}_{1:d}}{\partial \vec{z}_{d+1:D}} \\ \frac{\partial \vec{x}_{d+1:D}}{\partial \vec{z}_{1:d}} & \frac{\partial \vec{x}_{d+1:D}}{\partial \vec{z}_{d+1:D}} \end{pmatrix} = \begin{pmatrix} \mathbb{I} & 0 \\ A & \mathbb{J}^* \end{pmatrix}$$

the Jacobian becomes triangular!

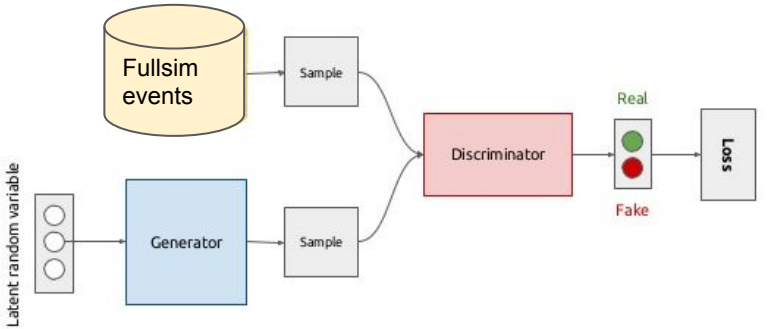
from Jason Yu

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Some applications

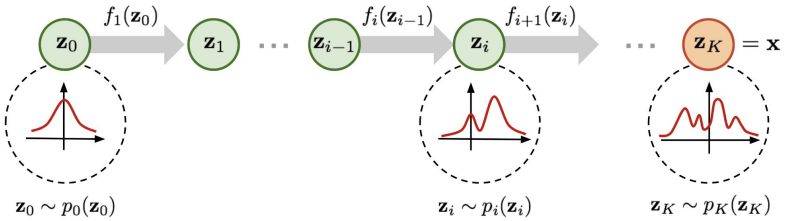
Simulation

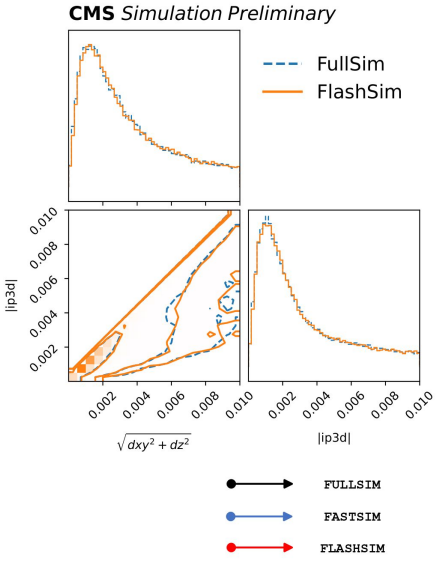
- Detailed (full) simulation (GEANT4 based)
  - ✓ Speed up some slow parts (e.g. shower models)
- Fast-simulation
  - ✓ Various versions of “fastsim” exists (w/o ML)
    - Re-produce low-level detector data is hard
      - Do not attempt to simulate “tracking hits” and just short cut to “tracks”
    - Only produce high level (analysis) simulated data formats (e.g. “Delphes like” simulation)
  - ✓ **Many inputs, arbitrary parameters in the algorithms, high complexity ⇒ ideal AI field!**
- Ideally ML may allows to have a simulation that is almost as accurate as Geant4 **for analysis purpose** but could run as fast as Delphes

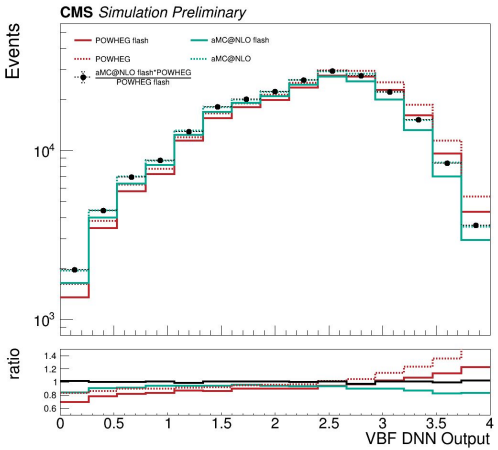


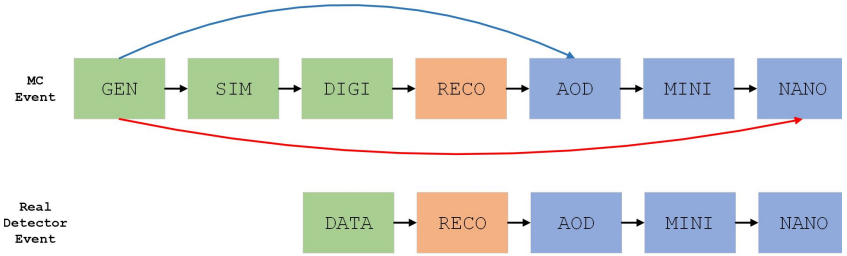
Fast simulation with Normalizing Flows

- Delphes like simulation “from generator to analysis ntuples” using machine learning
  - ✓ I.e. “skip GEANT”
- CMS prototyped a generic “Flashsim” using normalizing flows
- ATLAS used NF for analysis specific simulation
- LHCb implemented fast simulations with GANs





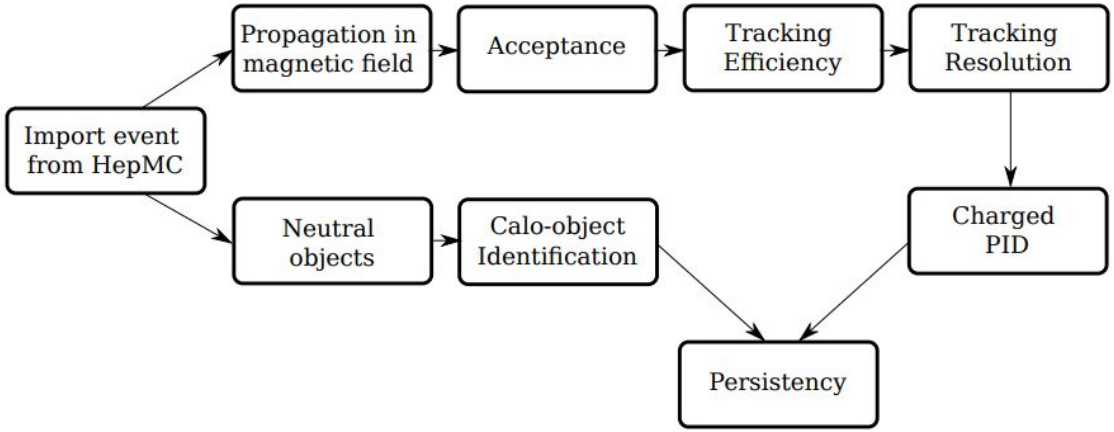




Lamarr: *a pipeline of parameterizations embedded in Gauss*

Lamarr is a pipeline of **modular parametrizations**, integrated with the LHCb analysis framework:

- compatibility of the same, LHCb-tuned, **generators**
- compatibility with the **distributed computing** middleware (LHCbDirac) and production environment
- producing datasets with same **persistency** format

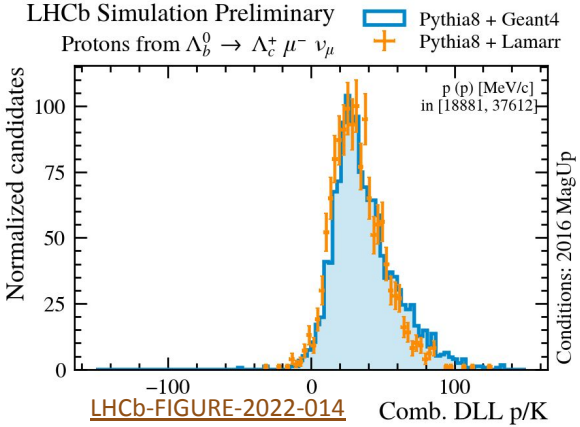


Efficiencies are modeled with binary classifiers (*e.g.* ***BDT***) predicting the probability of being reconstructed/selected…

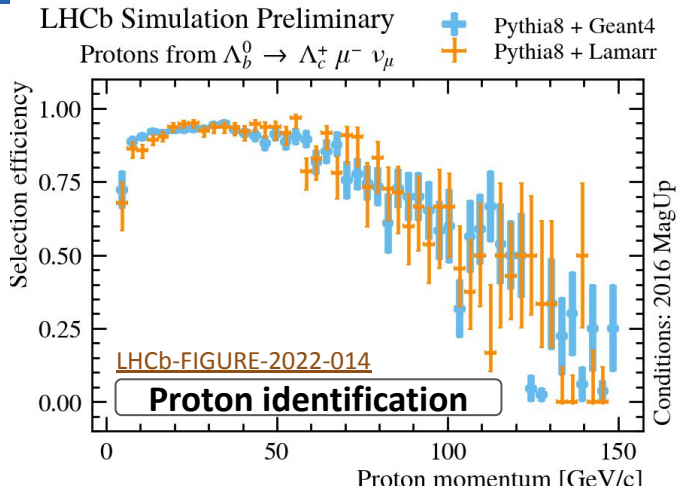
Reconstructed quantities/errors are modeled with **GANs**

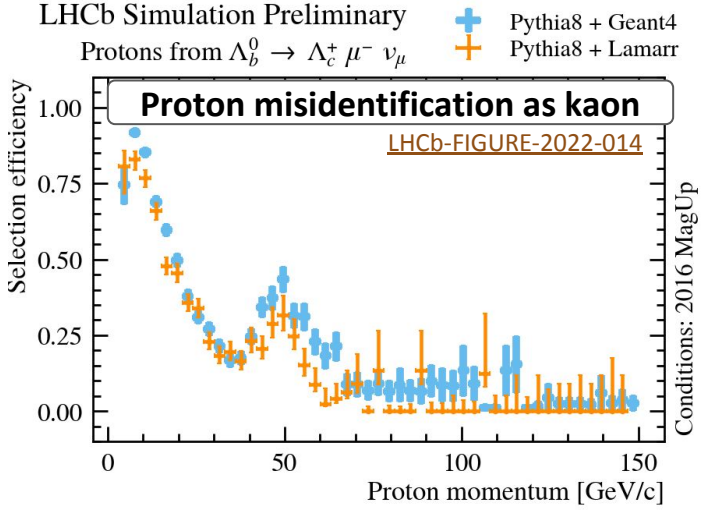
Proton identification

Lamarr simulates the distribution of the detector response. Analysts often inject the detector response in some analysis-specific classifier.



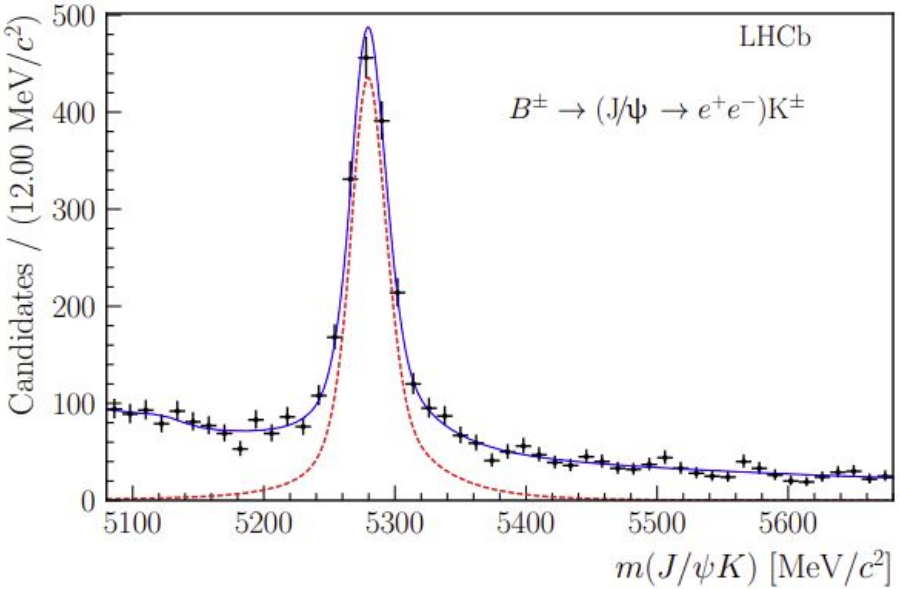
Here, we define cuts to visualize the ability of the trained models to describe the **dependence of the detector response on occupancy and kinematics**.





Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
M. Bovsiak, N. Kazeev - JINST 14 P08029 (2019)

Training on background-subtracted data



Background contribution can be modeled effectively studying the **invariant mass of some particle** involved in the decay process.

The effect of the **contamination** on any variable uncorrelated to the mass can then be statistically subtracted with the *sPlot* technique.

The **loss function of the machine learning algorithm** is modified to be compliant with the *sPlot* hypotheses and learn from the signal component while ignoring the contamination.

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

A word on **Resources & Infrastructure**

Training

- **Training generative models is computationally expensive and requires GPU**;
- GPUs are not granted to the LHC experiments as part of the WLCG pledges, but they are often provisioned in an opportunistic way.;
- Services for hyperparameter optimization may be needed [*e.g.* *hopaas*]
- **Commercial cloud** providers are too expensive.

INFN is trying to coordinate **cloud-based access to GPU resources** for training models with two initiatives:

- A flagship on ***Advanced Machine Learning*** in the Spoke2 of the *Centro Nazionale*
- An initiative of CSN5, named ***"AI_INFN"*** officially starting in January 2024.

Take part!

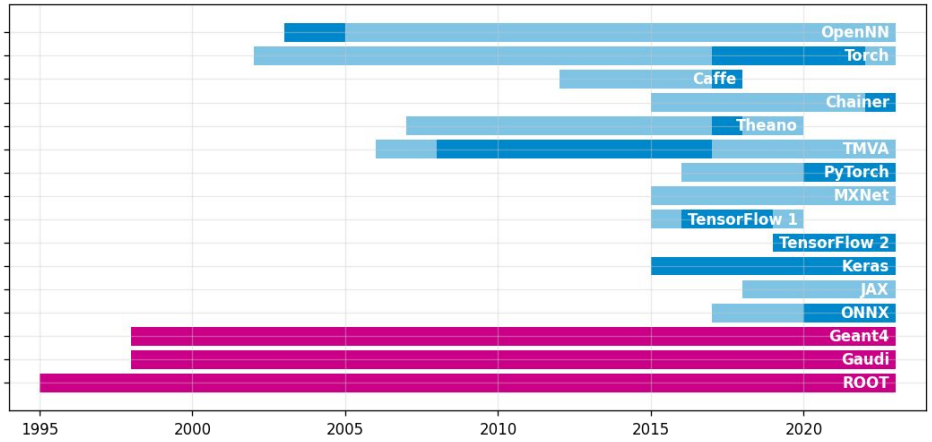
AI_INFN
Artificial Intelligence technologies for INFN research

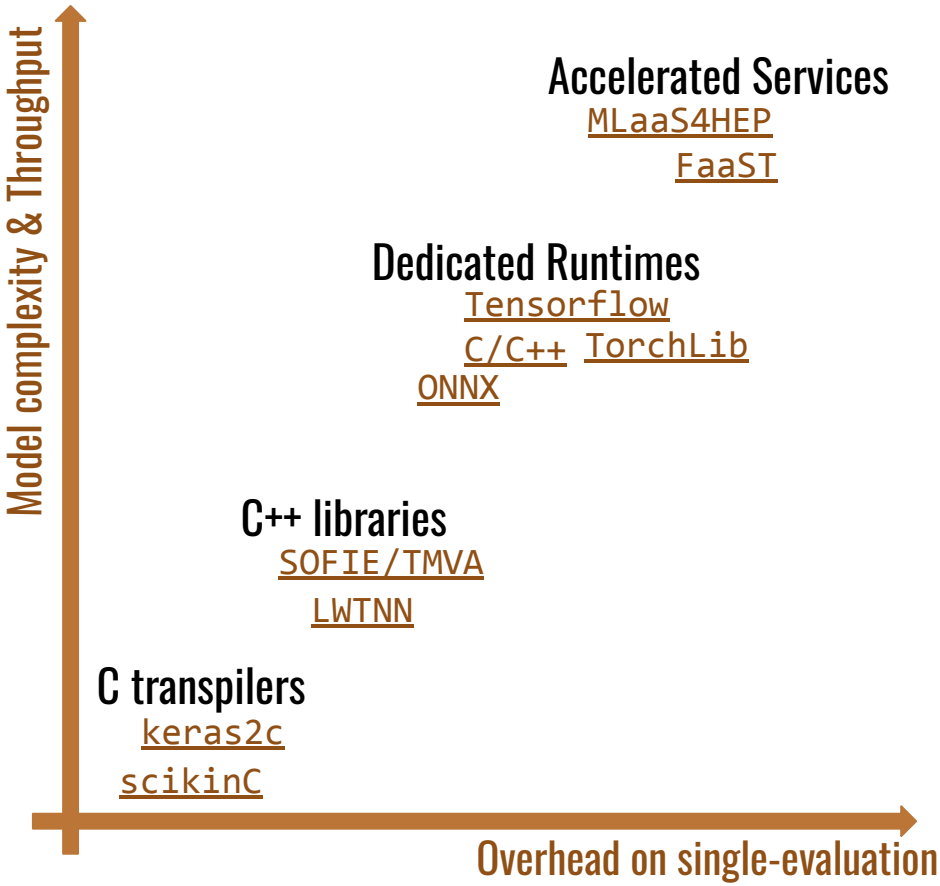
Deployment on software

Multiple options are available and are being explored for deploying ML in HEP applications (C++ distributed on WLCG).

- *project lifecycle;*
- *throughput;*
- *single-call overhead*

should all be considered for an optimal selection

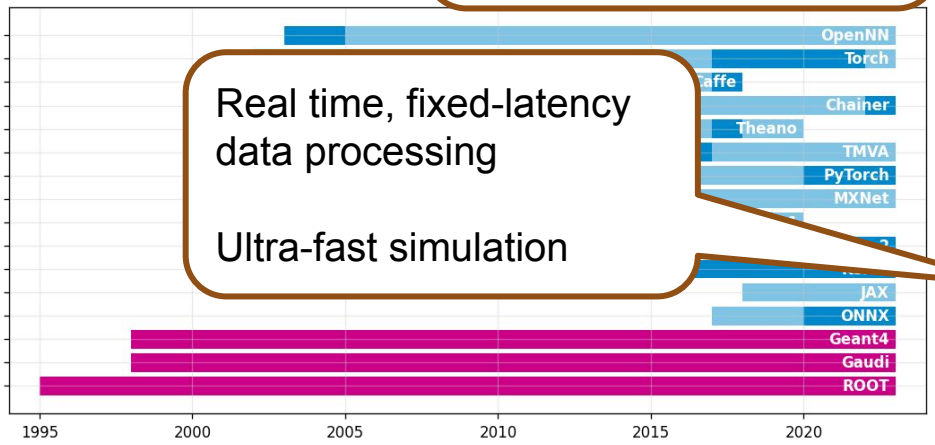




Deployment on software

Multiple options are available and ar
being explored for deploying ML in H
applications (C++ distributed on WLCG).

- *project lifecycl*
- *throughput;*
- *single-call over*



Real time, fixed-latency data processing

Ultra-fast simulation

Reconstruction-level classifiers and regressors on single objects

Integration in experiment software implies risky dependencies

Full-event processing on very large models; multiple event batches

Model complexity & T

Accelerated Services
MLaaS4HEP
FaaST

Dedicated Runtimes
Tensorflow
C/C++ TorchLib
ONNX

C++ libraries
SOFIE/TMVA
LWTNN

C transpilers
keras2c
scikinC

Overhead on single-evaluation

Slide from **Lorenzo Sestini** as presented at the ***Fifth ML_INFN Hackathon***, Pisa

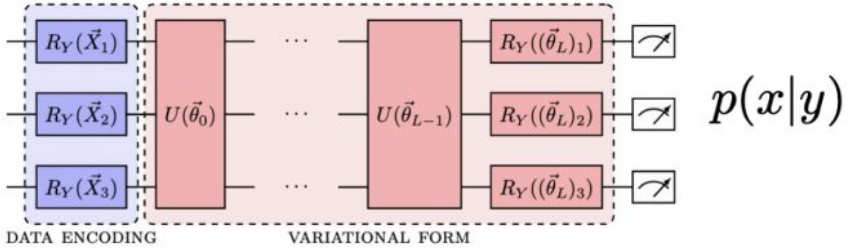
Generative QML: Quantum Born Machines

- **Quantum Circuit Born Machines (QCBM)** make use of the stochastic nature of quantum measurements, no classical analogs
- Each base element of the quantum space is mapped to a specific configuration of the system we want to simulate

$$p_\theta(x) = |\langle x|\psi(\theta)\rangle|^2$$

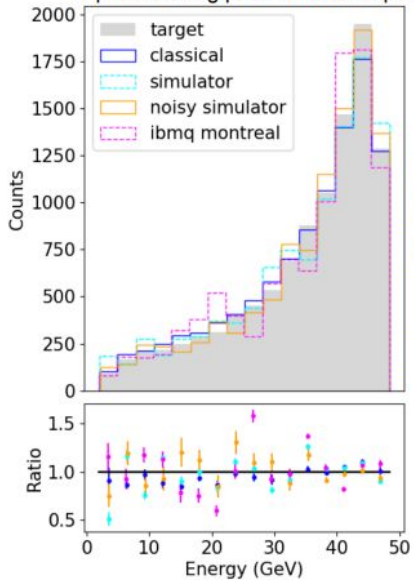
- As an example if we have N qubits we can simulate a distribution in 2^N bins
- Variational Quantum Circuits are trained to obtain the best compatibility with respect to the original dataset. **The initial state has a negligible impact**.

Conditional Born Machines: conditions are given in input to the circuit



$p(x|y)$

https://arxiv.org/pdf/2205.07674.pdf



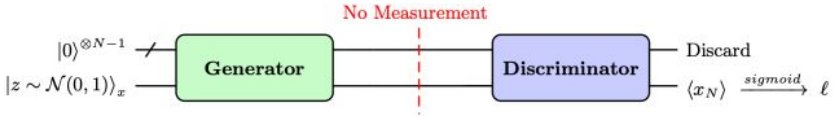
Example: Muonic Force Carriers energy distribution

QCBM are pretty stable and reliable, but many qubits are needed for multi-dimensional simulations

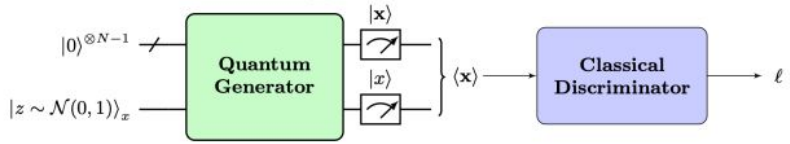
Slide from **Lorenzo Sestini** as presented at the ***Fifth ML_INFN Hackathon***, Pisa

Generative QML: qGAN

- **Quantum Generative Adversarial Networks**: a quantum generator is trained against a discriminator (classical or quantum)
- In general, GAN (not only qGAN) could replace time-consuming program as Geant4
- With qGAN, N qubits can be used to simulate 2^N features (NOT 2^N configurations as in Born Machines)
- The problem is the stability and convergence: it is useful to increase the latent space dimension, e.g. adding ancillary qubits



(a) *Fully quantum model.*

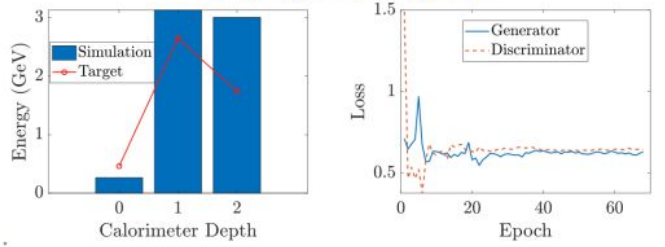


(b) *Hybrid model.*

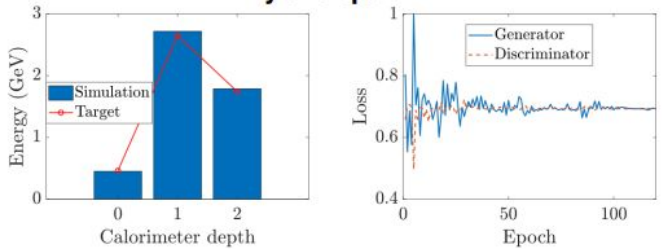
https://arxiv.org/pdf/2101.11132.pdf

Calorimeter simulation: energy as a function of the depth (3 bins

Full quantum qGAN



Hybrid qGAN



16/11/2023 Lorenzo Sestini 30/36

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
ICSC

Conclusion

Generative models represent a viable strategy to significantly reduce the computing time to produce simulated samples for the LHC experiments.

The fundamental idea is that we can **parametrize** very effectively some step in the simulation pipeline.
Which step we consider as worth approximating depends on the application.

Independently of the exact model chosen for the parametrization, training and deploying generative models in the HEP software environment is not trivial and deserves dedicated effort.

The research community is extremely fresh and active! Take part!