Istituto Nazionale di Fisica Nucleare
SEZIONE DI FIRENZE

Machine Learning for the LHCb Simulations

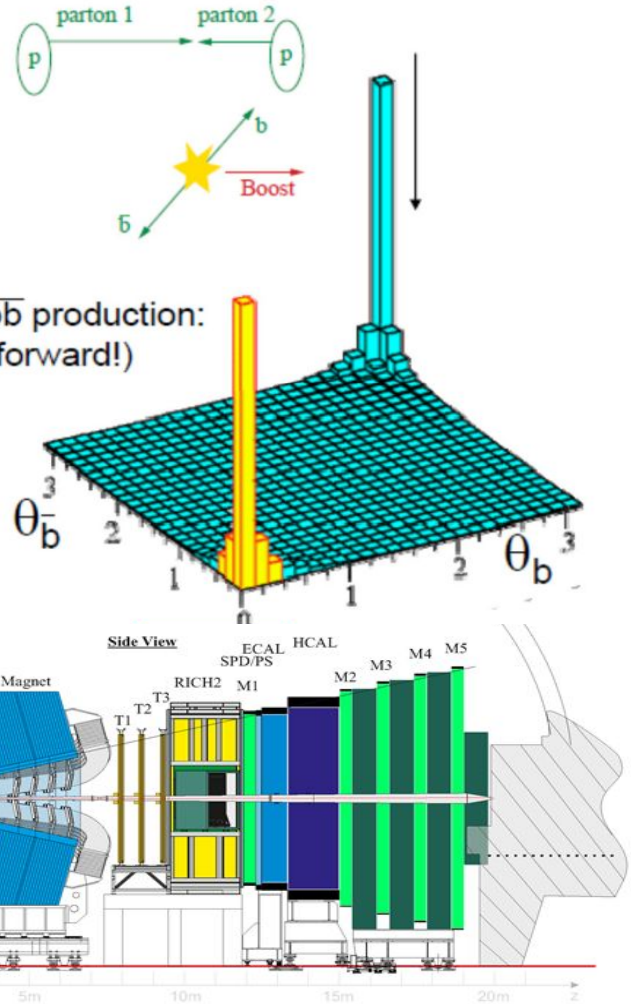
Lucio Anderlini *on behalf of the LHCb Collaboration*
Istituto Nazionale di Fisica Nucleare – Sezione di Firenze

2021 - 09 - 07

Hadron physics with LHCb

The LHC is the most **abundant source of heavy quarks** on Earth. It allows for **precision measurements** challenging the Standard Model.

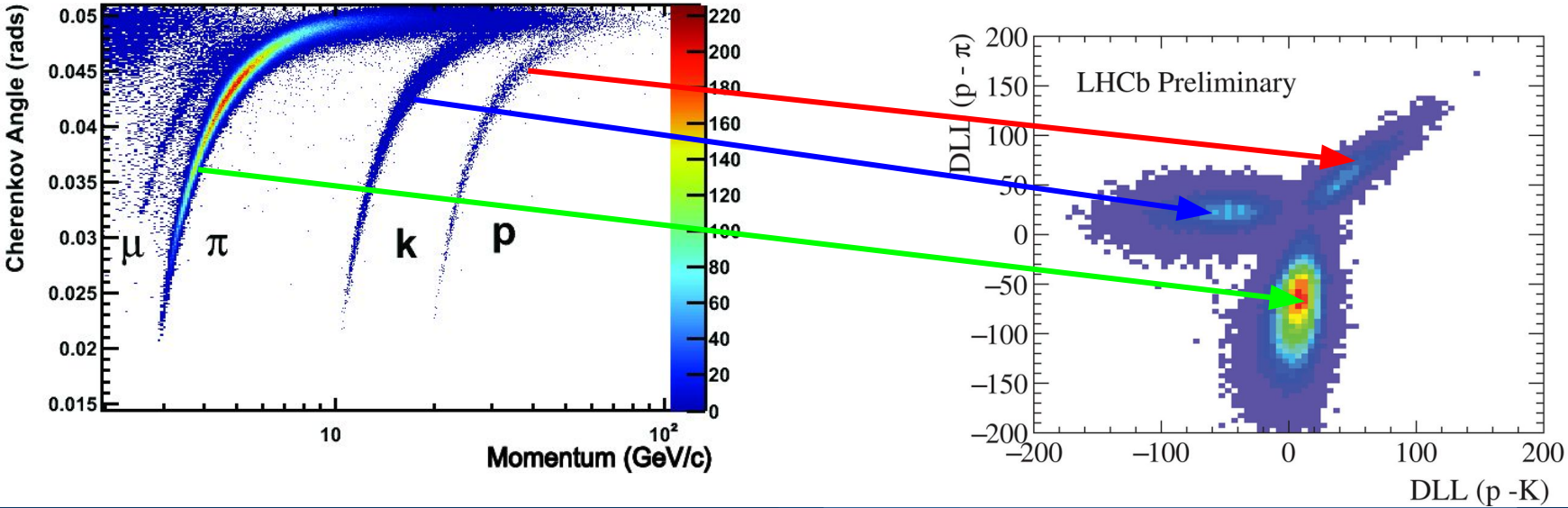
LHCb studies lower-p_T hadrons that cluster in in the "**forward region**" at low angle with respect to the beam axis.



Special at LHCb: *the RICH detectors for hadron identification*

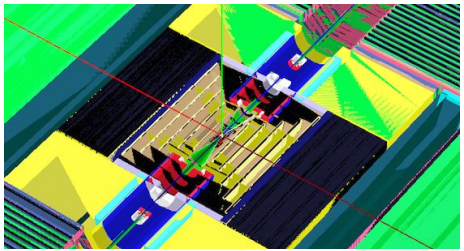
To classify heavy hadron decays an outstanding capability of identifying hadrons is necessary. The LHCb experiment uses two **Ring Image Cherenkov** (RICH) detectors to separate pions, kaons and protons.

The Cherenkov angle is then converted into ***Differential Log Likelihoods***.



Why do we need simulation?

Design detectors and experiments

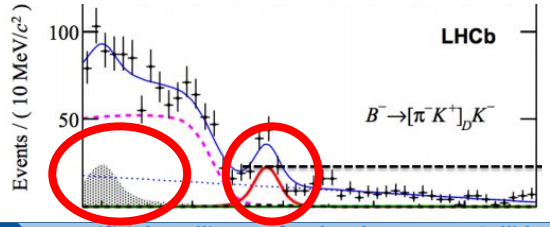


```
P/export_scikinC.py | P/pipeline.C | P/trainMuon.py
import sys

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/dense
    bias_shape = h5[f"{layer.name}/sequential/dense/b
```

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

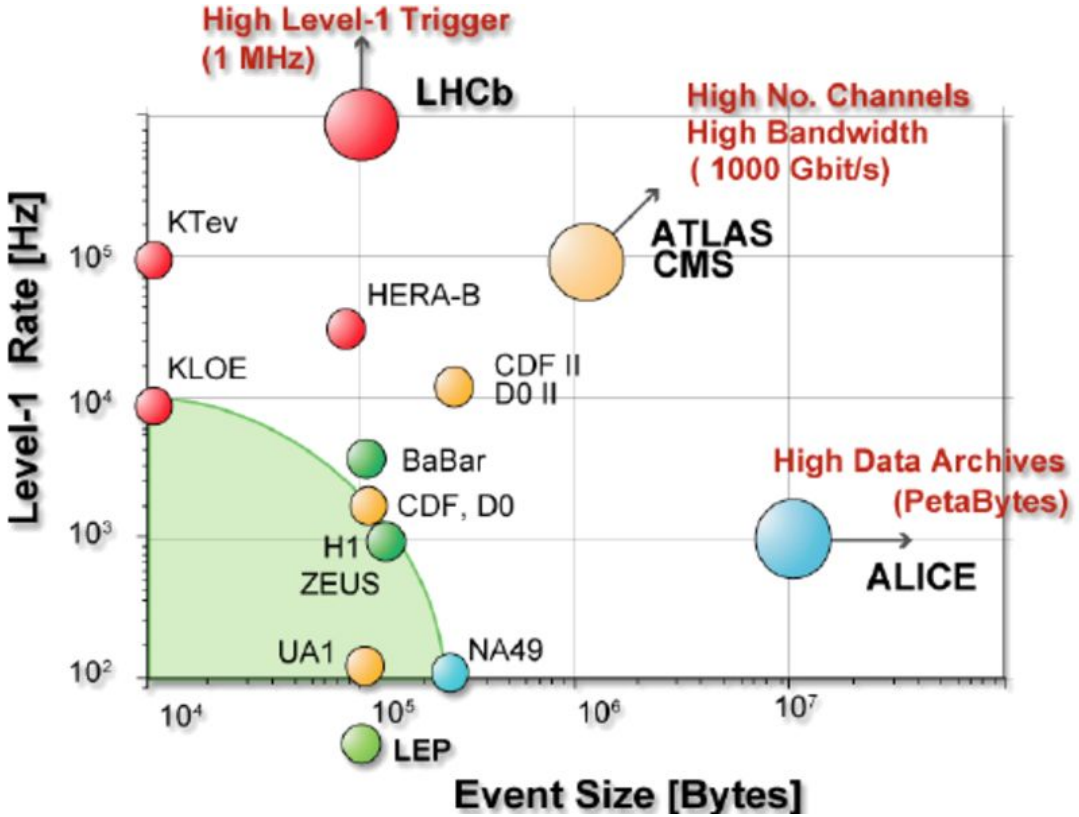
Different physics

Different experiment

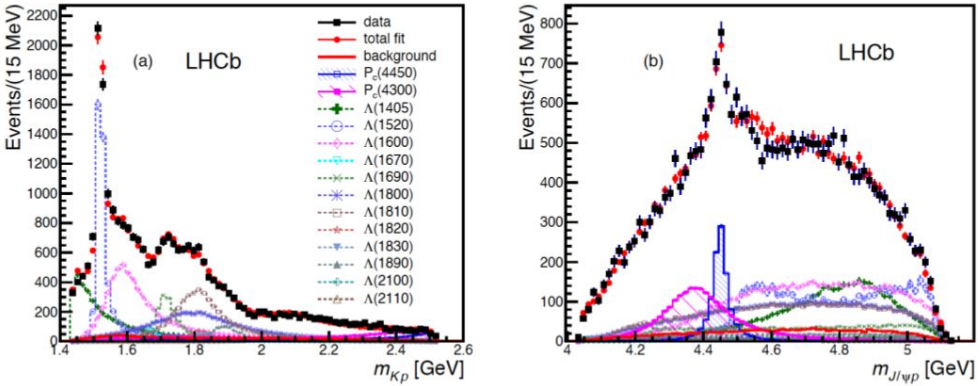
Different data processing

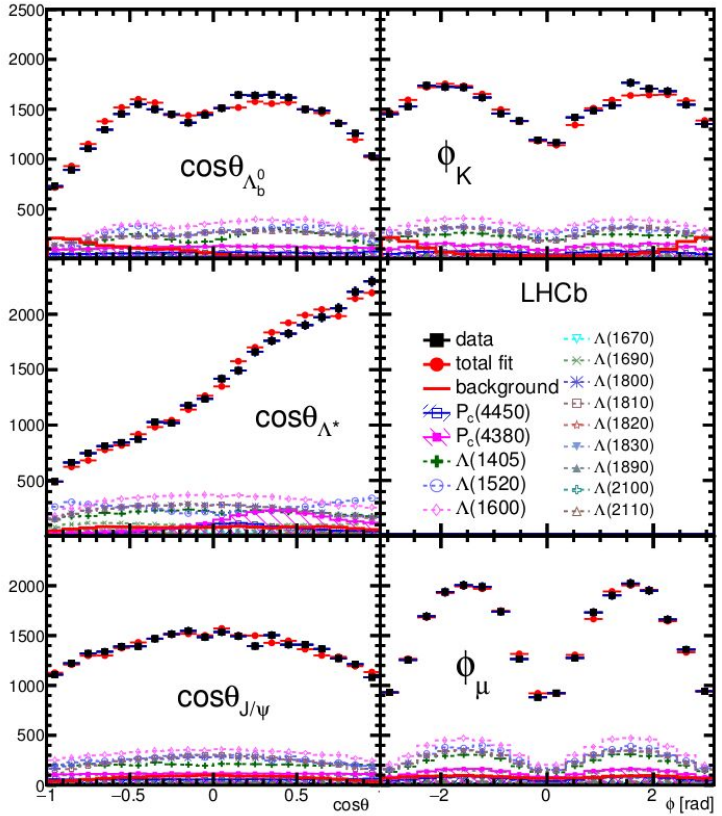
Different needs in terms of simulation.

LHCb focus on precision measurements require very large simulation samples



Amplitude analysis: *the example of the pentaquark*

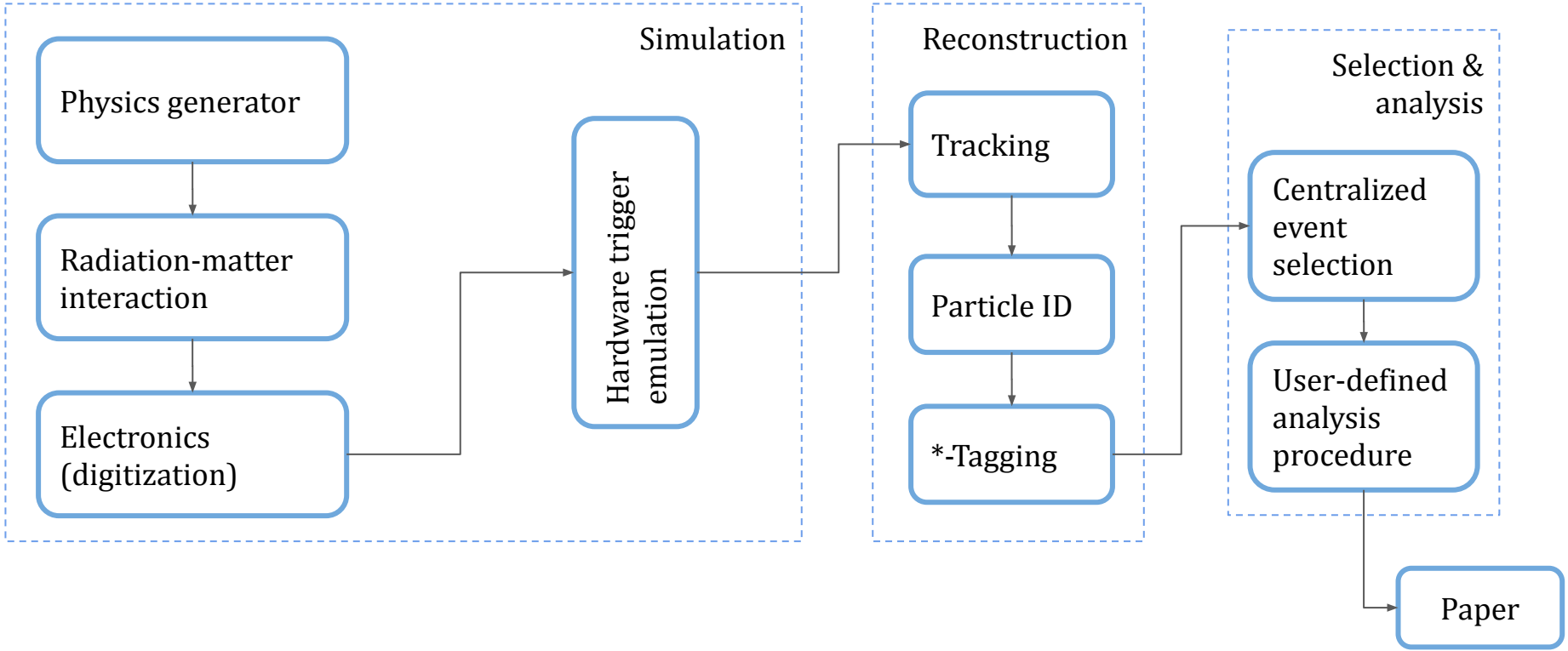




An outstanding modeling of the efficiency in the six-dimensional space used for the likelihood fit is needed.

Imperfection in the efficiency determination must be negligible in front of the statistical uncertainty obtained with 2.50×10^5 signal events.

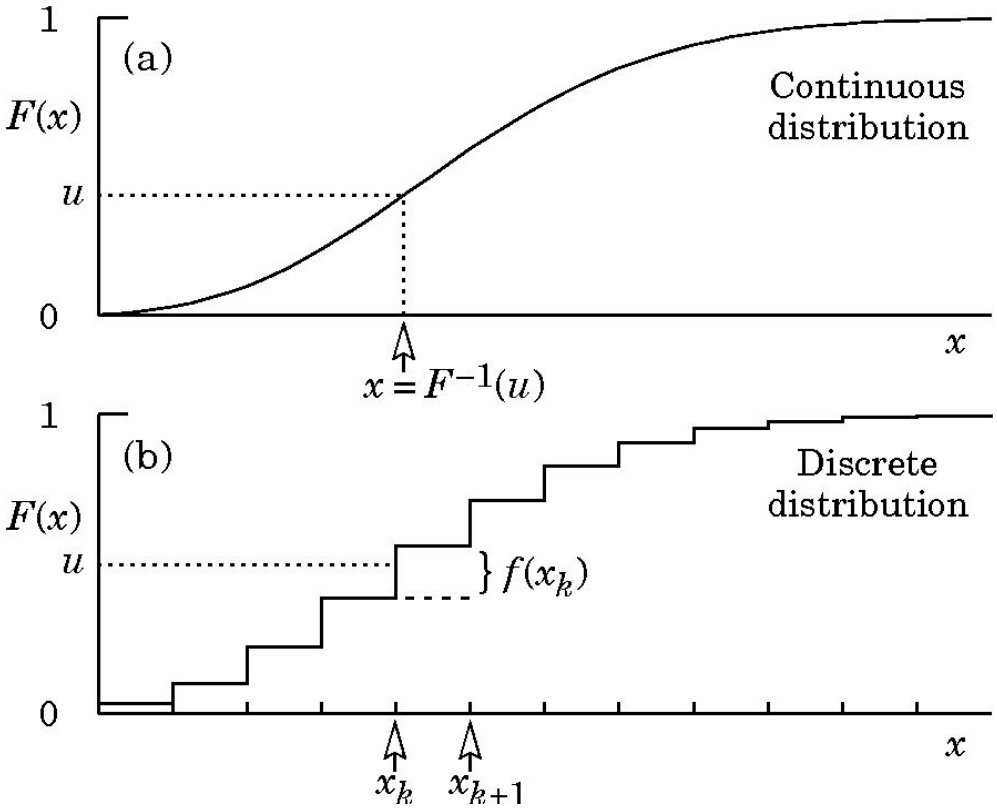
Standard simulation: the big picture



Parametric simulation

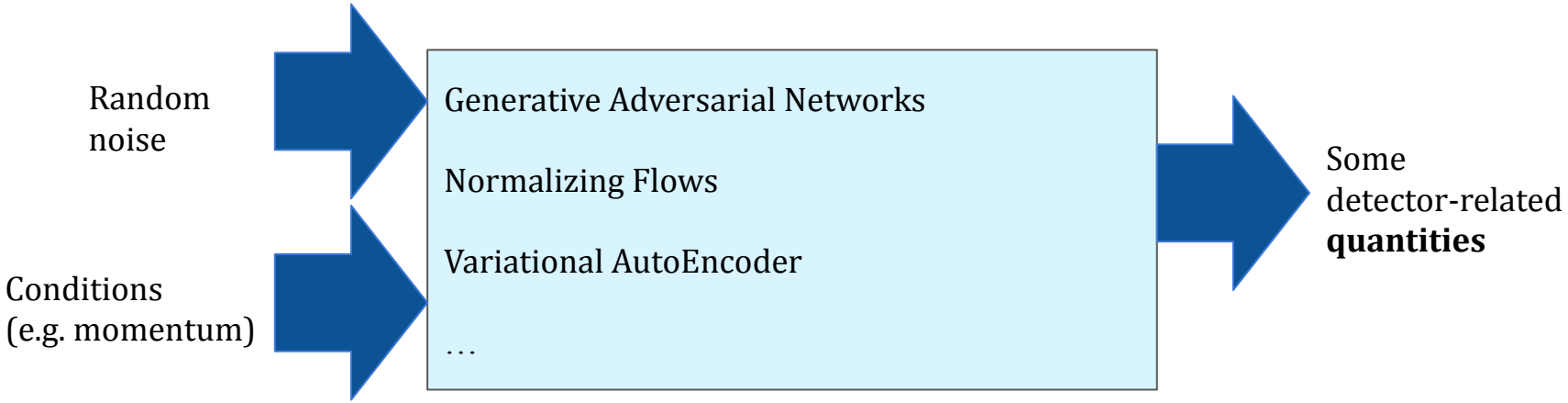
We know from statistics that we can generate samples according to a given distribution, for example, through the inverse CDF method.

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.

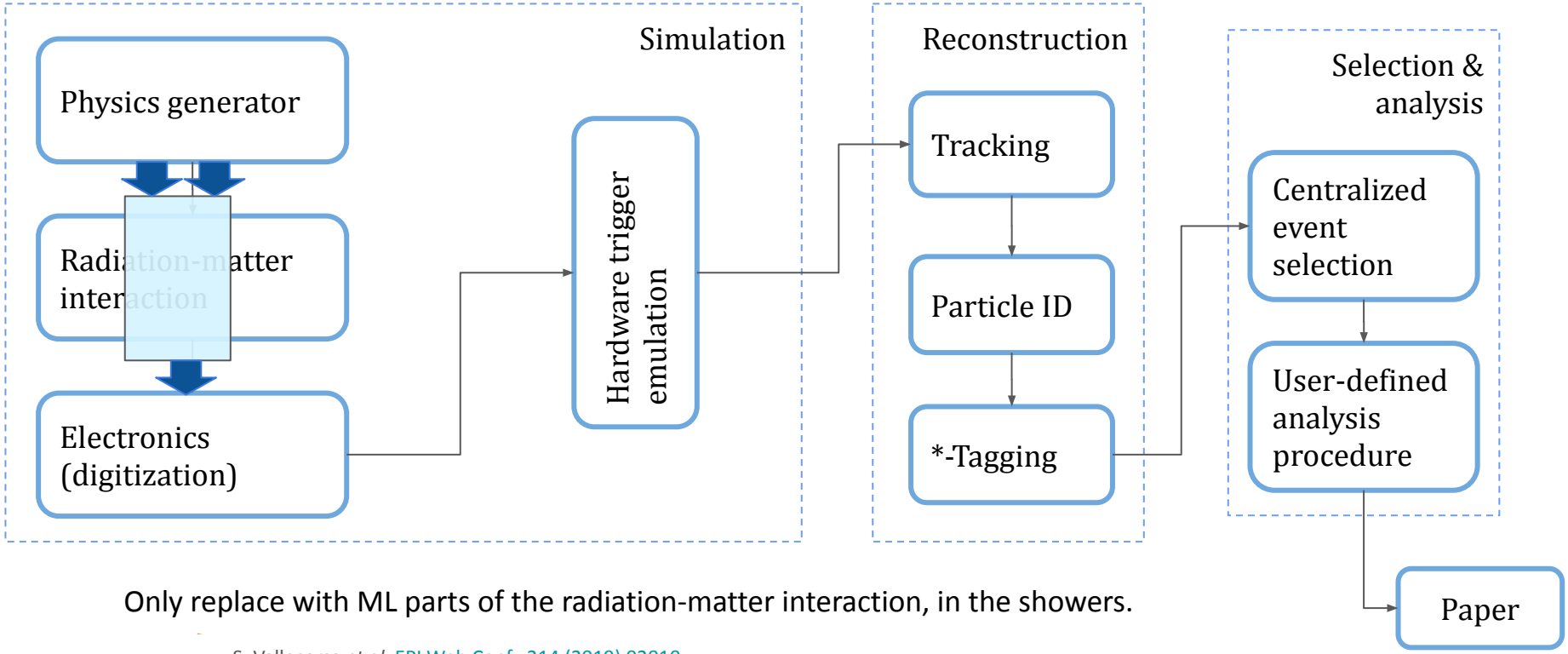


Our ML building block

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



Approaches to ML in simulation



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

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

Approaches to ML in simulation

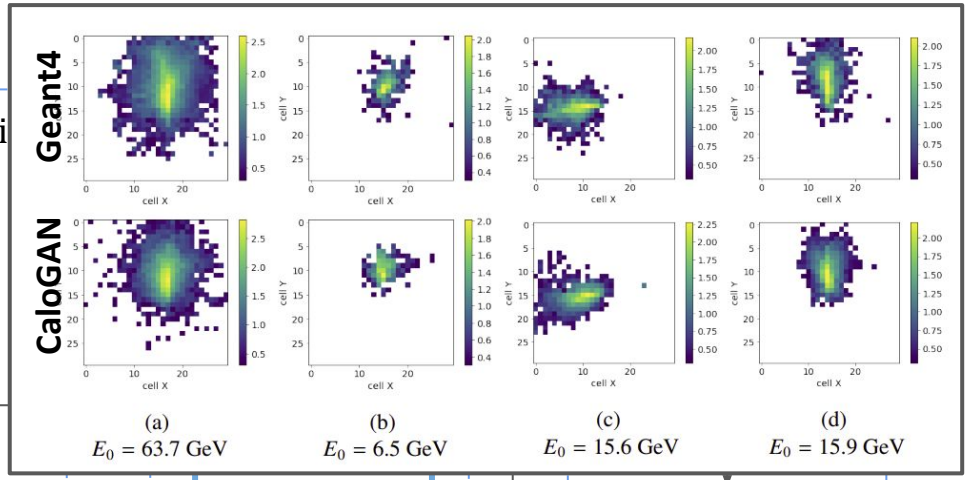
Simulati

Physics generator

Radiation-matter interaction

Electronics (digitization)

Hardware trigger emulation



*-Tagging

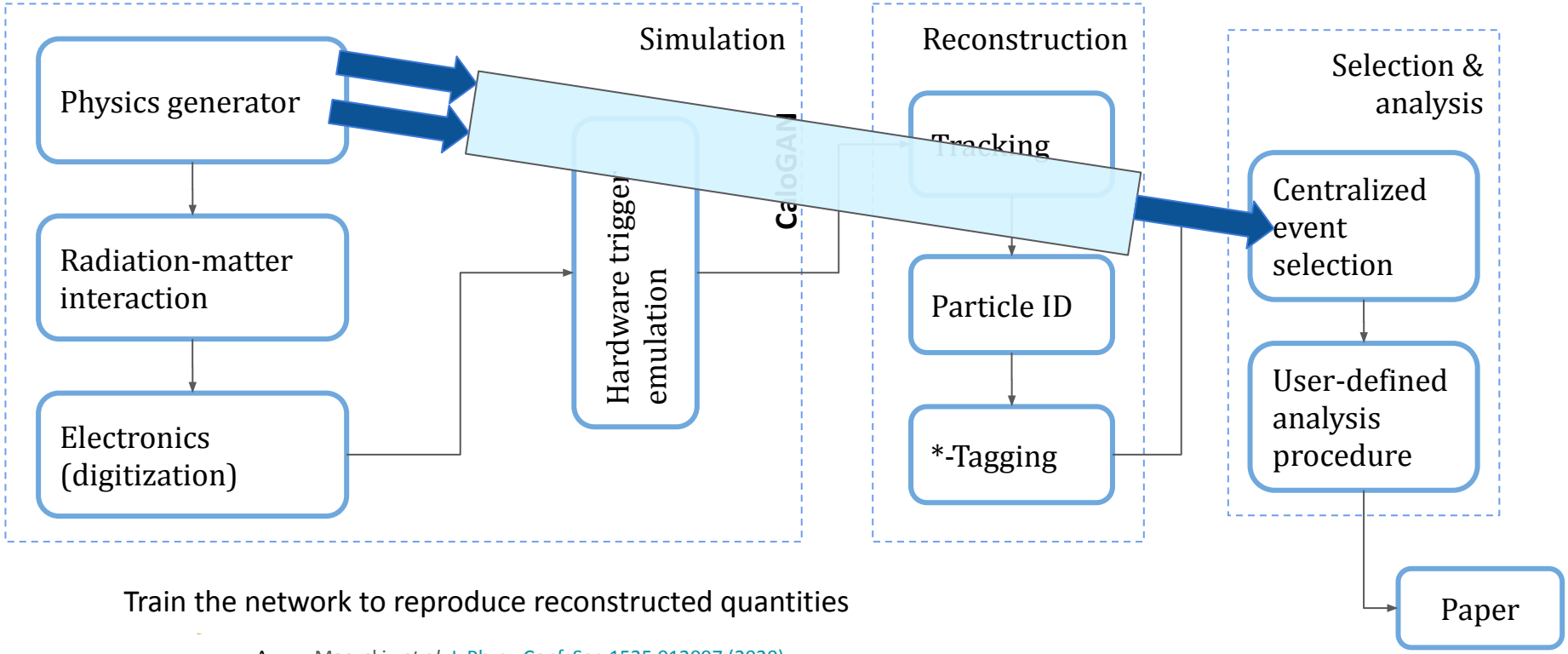
User-defined analysis procedure

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



Train the network to reproduce reconstructed quantities

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

Which approach is the best one?

Simulating the rad-matter interaction does not change the philosophy of the simulation: everything working with the detailed simulation will also work when replacing a detector with its parametrization (**full-featured simulation**).

Unfortunately, small imperfections in the parametrization may result into large errors of the analysis level quantities at the end of the reconstruction & analysis pipeline.

Simulating the analysis level features allows for exceptional-quality description of the simulated analysis-level quantities (possibly better than full simulation!!!) and it is **faster** than the reconstruction step alone (not to mention simulation)

But, collective features and those depending on secondary particles are very difficult to describe, making this simulation "**feature-limited**".

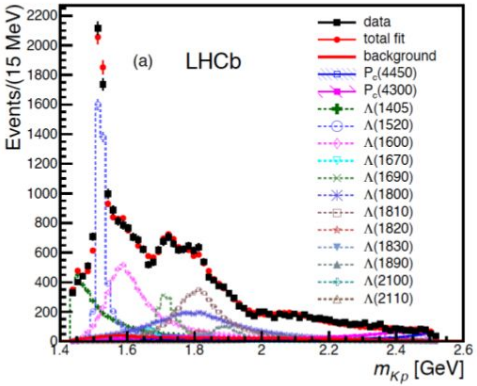
And then? Which approach is the best one?

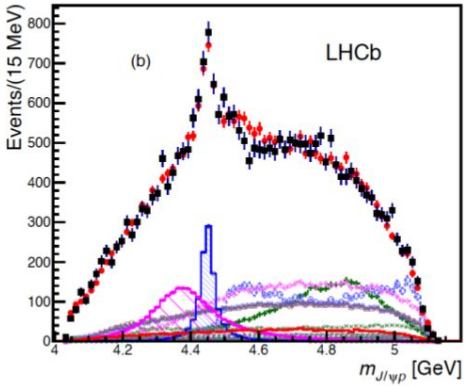
We aim at covering the majority of the analysis needs with **an ultra-fast, feature-limited simulation.**

For analyses requiring collective features such as flavour tagging we will make **full-featured fast simulations** available.

Finally, for those studies requiring to access to the raw data of the simulated detector, or to perform detector studies, the **detailed simulation** will a totally viable option.

Thinking again to amplitude analyses...





The six quantities composing our dataset are “simple” kinematic quantities.

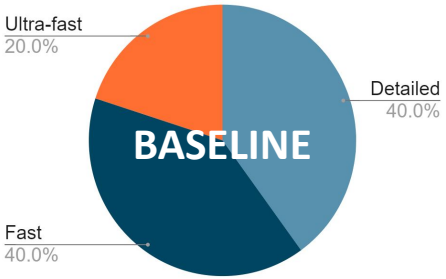
The theoretical uncertainties on the decay model are large.

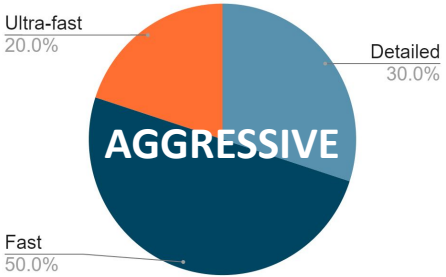
The number of simulated events required to get a good description with correlation in a 6-dimensional space is very large.

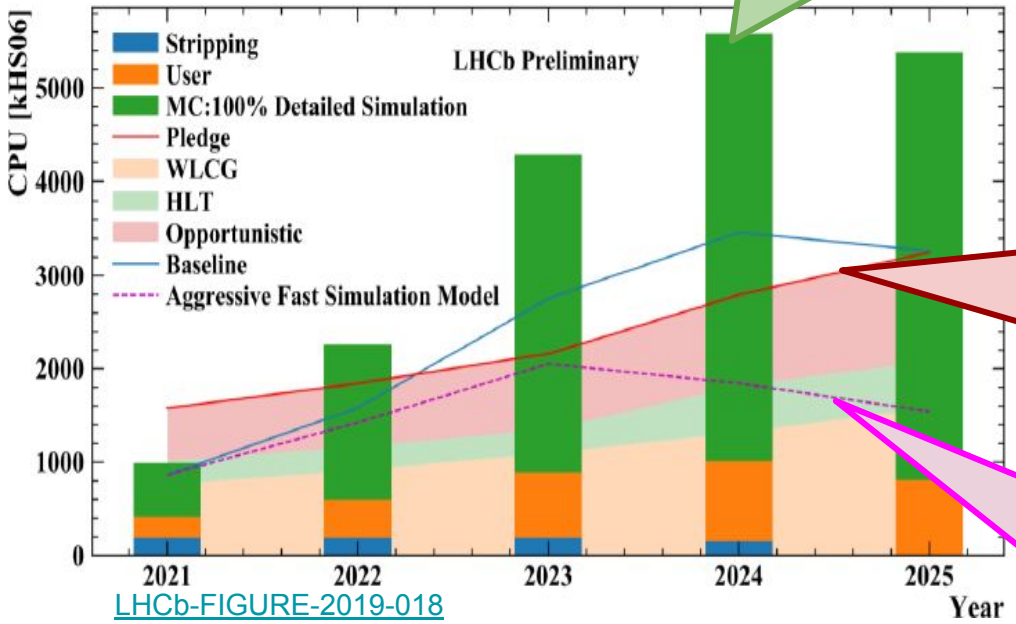
→ Use *ultra-fast simulation* of analysis level-quantities (possibly validating the projections with full simulation) to assess efficiency and background contributions.

Do we really really need that stuff?

CPU resources needed for a simulation scheme only including Geant4-based *"Detailed Simulation"* .







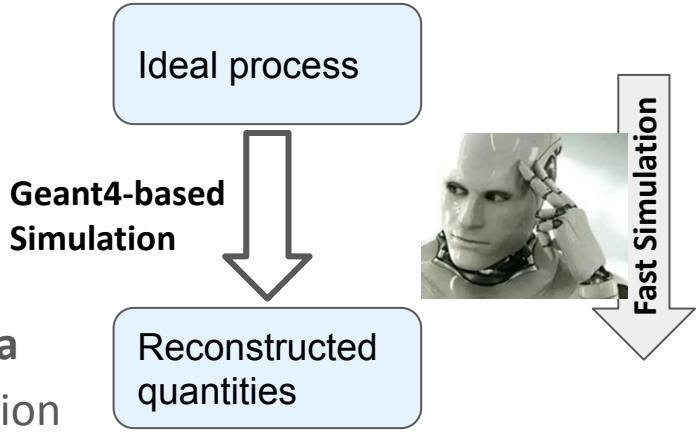
LHCb-FIGURE-2019-018

Available resources according to the planned funding scheme

What would be achievable with **excellent** *fast* and *ultra-fast simulations*

Training with synthetic (simulated) data

The most obvious training scheme is training the *Generative Models* for the (ultra-)fast simulation to mimic the *detailed simulation.*



In practice,

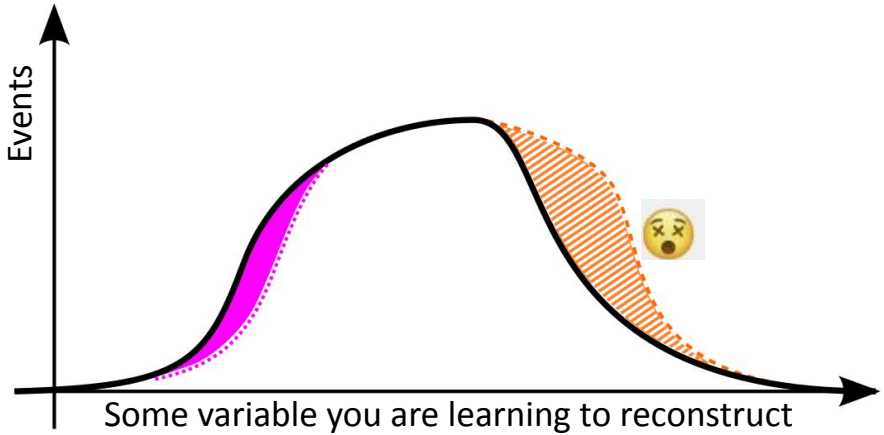
1. **Generate a large dataset of Geant4-simulated data**
2. Train a generative model to reproduce the connection between the MC-truth and the reconstructed quantities

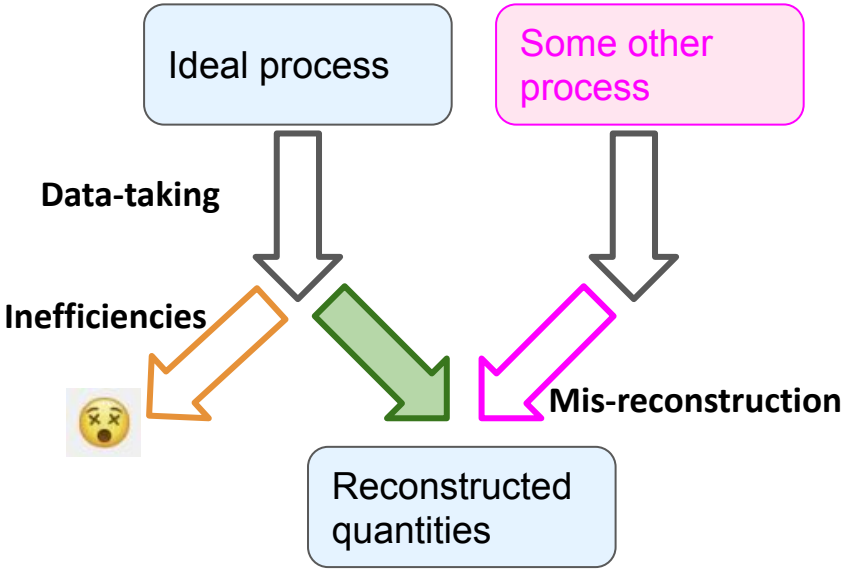
This approach is **simple** and **works anywhere**, but the fast simulation is an approximation of the detailed simulation and inherit all its imperfections.

A fast simulation trained on synthetic data will be at most as good as the full simulation.

Training on real data

Training on real data is therefore appealing, but real data is often **biased** and **contaminated**.





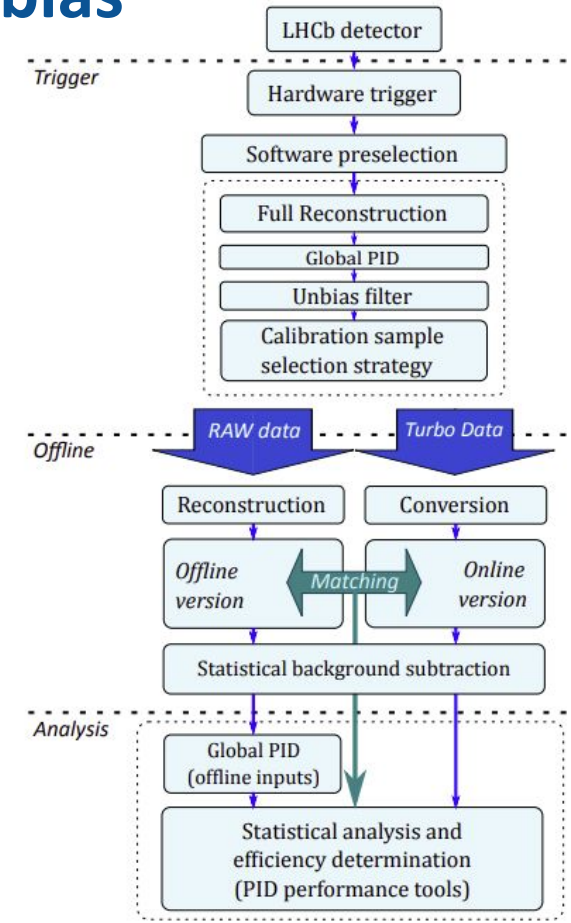
Dedicated selection strategies to avoid bias at LHCb

Biases are avoided by applying selection criteria to some particles and using for training some other, **never relevant** to the selection procedure.

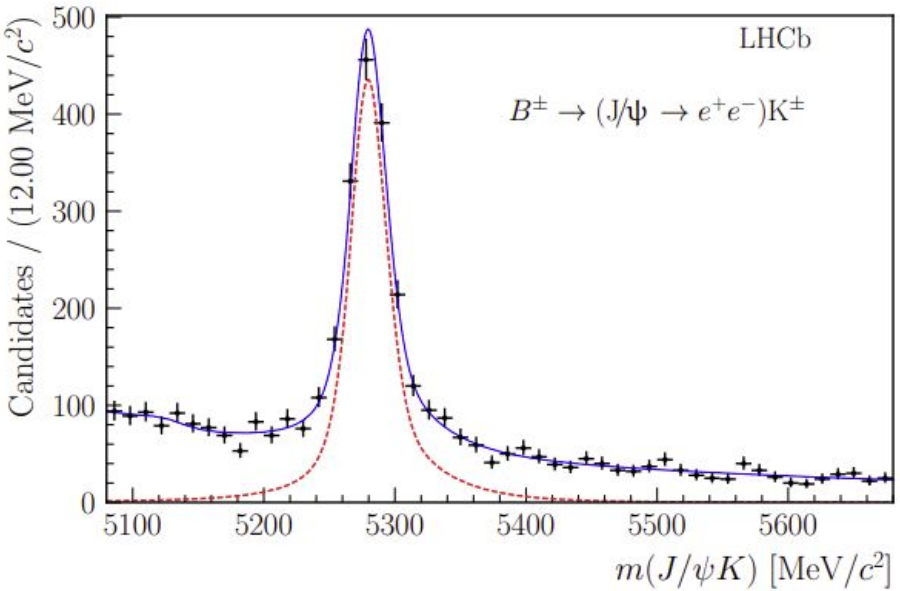
A special Data Acquisition Pipeline is necessary for the calibration samples in order to:

- avoid bias at already at trigger level
- keep track of the variables reconstructed by the trigger

For example, in $B^{\pm} \to (J/\psi \to e^+e^-)K^{\pm}$ one can apply cuts on the **kaon** and the **positron** to then train on the **electron**.



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.

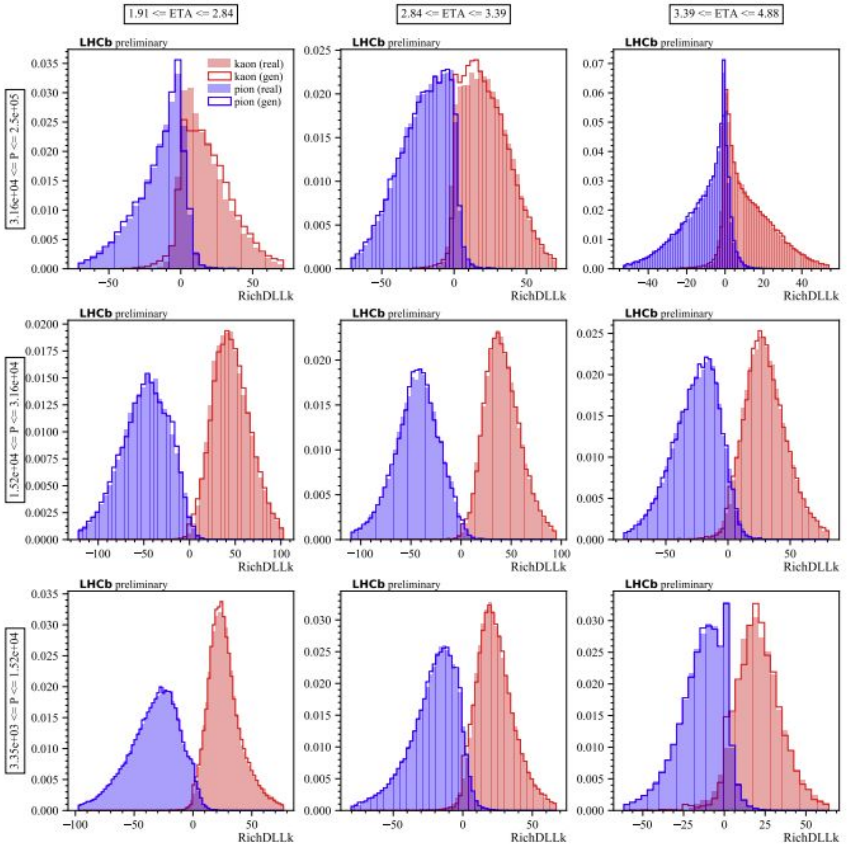
The **loss function of the machine learning algorithm** is modified to be compliant with the *sPlot* hypotheses and learn from the signal component while ignoring the contamination.

No inheritance from Geant4 simulation

When this is done, the inheritance from the detailed simulation is broken.

The two simulations are then independent and can validate each other.

With models good enough, the fast simulation can be even “better” than the full simulation.



Example
modelling of the RICH detectors at the **LHCb** experiment

kaon (real)
kaon (gen)
pion (real)
pion (gen)

Goodness of the GAN model

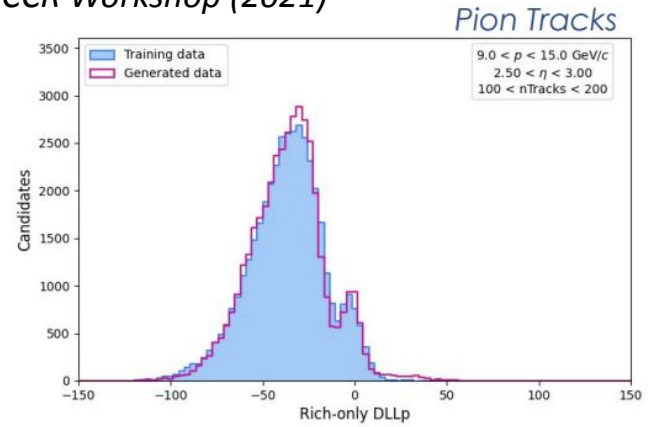
GAN quality is assessed by training one (or more) BDTs to distinguish the generated dataset from the one used for training.

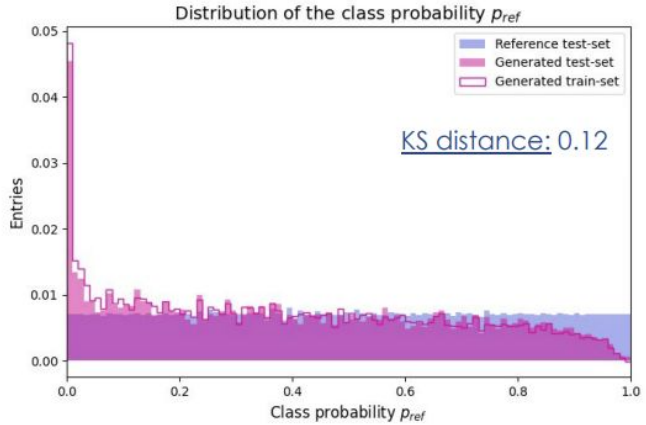
We use the ***Kolmogorov-Smirnov*** distance between the two datasets as metric for the quality of the GAN training.

You can read it as an upper limit to the absolute error introduced by using the GAN model to measure the efficiency of a requirement on a sample **identical to that used in training**.

*. Larger errors are still possible considering corners of the phase space.

Stolen from M. Barbetti,
Simulating the LHCb Detector with GANs, CCR Workshop (2021)





Deployment environment

The physics cases of the four LHC experiment are different, and so are the software solutions, but in general:

- They are C++ based
- They are built with a complicated build system
- Rely on sophisticated schedulers to profit from multithreading

DNN libraries such as TensorFlow and PyTorch are designed to make the most out of parallel computing: even when running on a single working-thread, the scheduling may interfere with the underlying application.

Deployment does not just happen...

Want to deploy your ML model

Can you afford **frequent updates** of TensorFlow/Torch dependencies?

Yes!

Can you integrate TF/Torch in the **build system** of the experiment?

Yes!

Integrate TF/Torch in your C++ using **C++** APIs

No

Integrate TF/Torch in your C++ using plain **C** APIs

No

Compile the NN **together with the software stack?** (loading weights)

Yes!

"Transpile" the NN to C++ with RNNgenerator

No

Export your model to LWTNN format

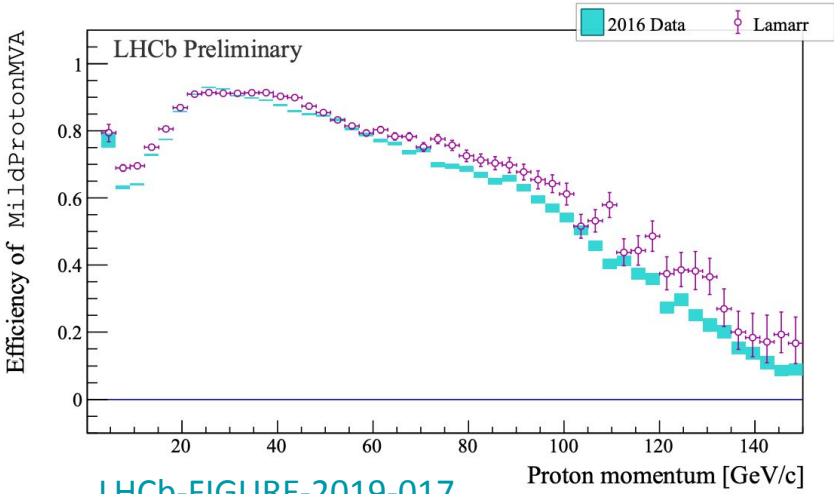
Do your pipelines include scikit-learn steps?

scikinC is a transpiler of some commonly-used scikit-learn (and some keras) steps to dynamically linkable libraries (.so).

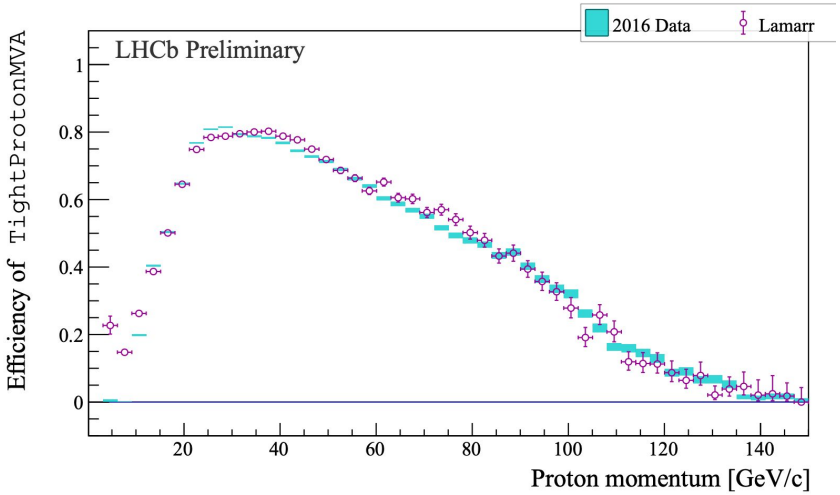
Going through the pipeline

Final validation is performed, for example, by comparing the efficiency of the Particle Identification (PID) selective criteria with those obtained from real data, through decay modes **not used for training**.

Note this is a kind of closure-test exercise because **production mechanism**, detector **acceptance**, tracking **efficiency** and **resolution** as well as **trigger** and offline **selection** strategy all contribute to the definition of the PID distributions in both simulated and real data.



LHCb-FIGURE-2019-017

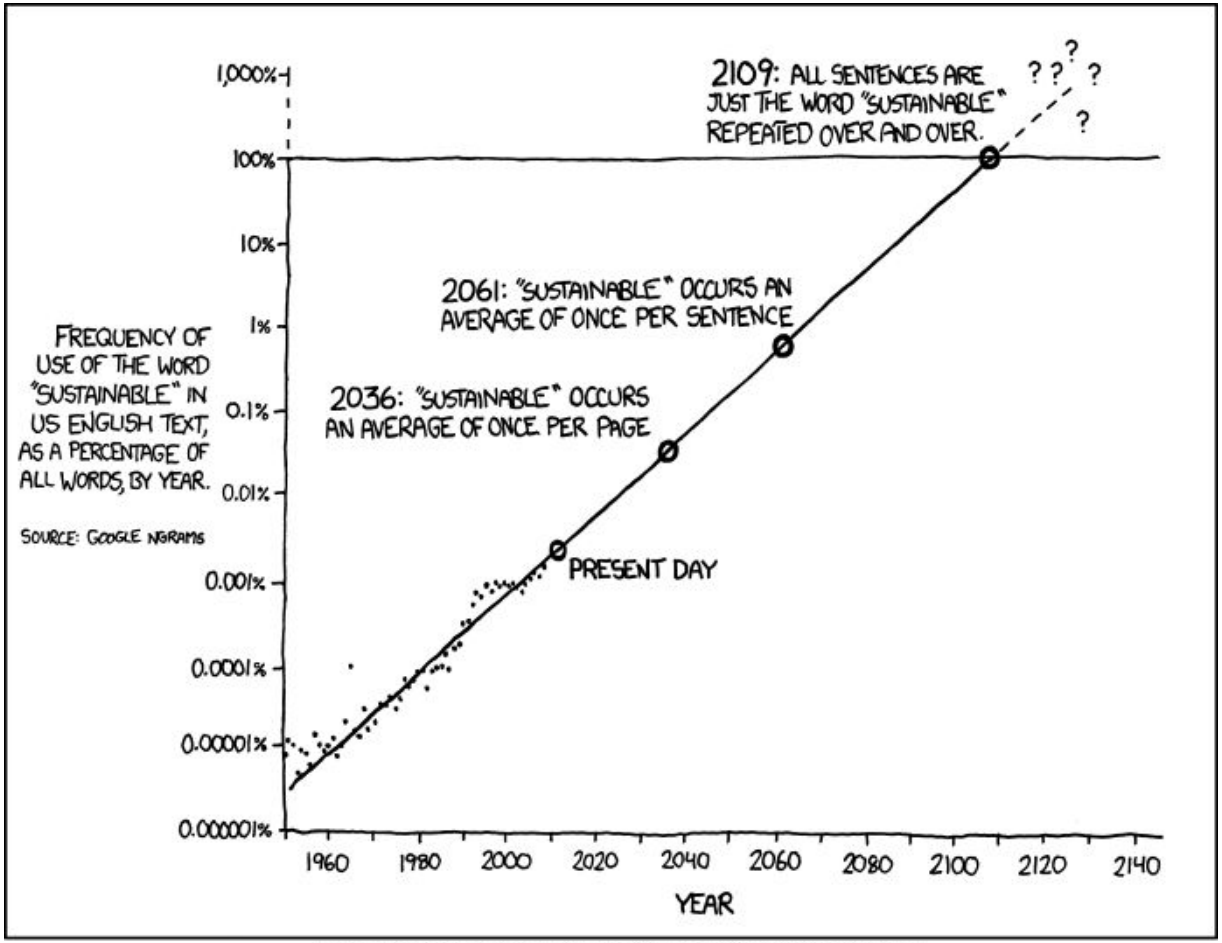


Conclusion

Conclusion 1

The Simulation of the collision data for the experiments of the Large Hadron Collider is crucial for a number of applications.

Data analyses are the most important consumers of *simulated data*. Scaling the their demand to the future runs shows that simulation is not sustainable.



Conclusion 2

Many studies are ongoing to use machine learning to speed up the simulation, taking different approaches:

- simulating the radiation-matter interaction faster
- simulating the response of some detector
- simulating the whole simulation pipeline to reconstructed analysis level quantities

The various solutions are all important as they can all help speeding the simulation up.