Software development for LHCb ECAL upgrade

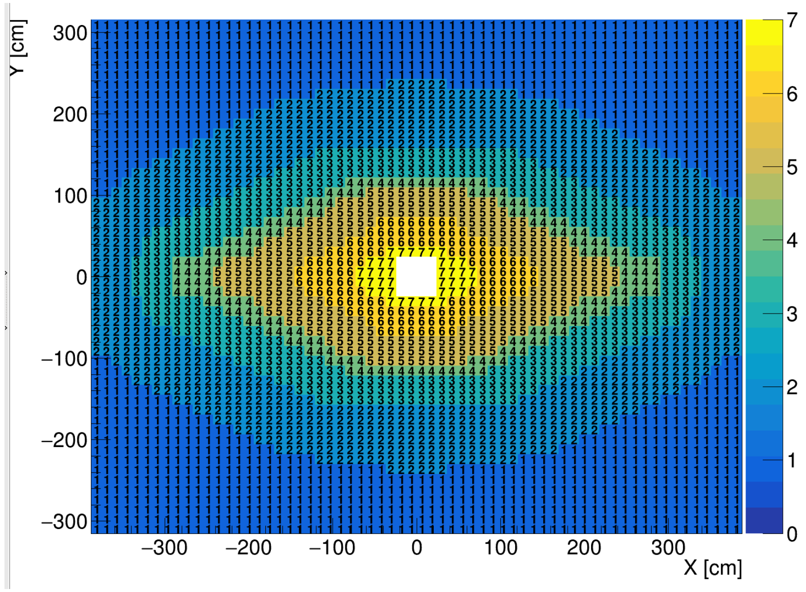
Zifan Li[1] (on behalf of the LHCb ECAL U-II group)
CLHCP2025, 2025/10/31

[1]Peking University

Contents

PicoCal – the LHCb ECAL in Upgrade II

- In Upgrade II, the LHCb experiment will have:
  - Higher luminosity: $\mathcal{L} = 1.5\times10^{34}\ \mathrm{cm}^{-2}\ \mathrm{s}^{-1}$
  - Higher pile-up: ~ 42
- As a result, the LHCb ECAL will need:
  - Deployment of new radiation-hard technologies
  - New cell design for higher spatial granularity: Spaghetti Calorimeter (SpaCal)
  - New geometry (cell types, double sided readout, ⋯)
  - Add hit time readout

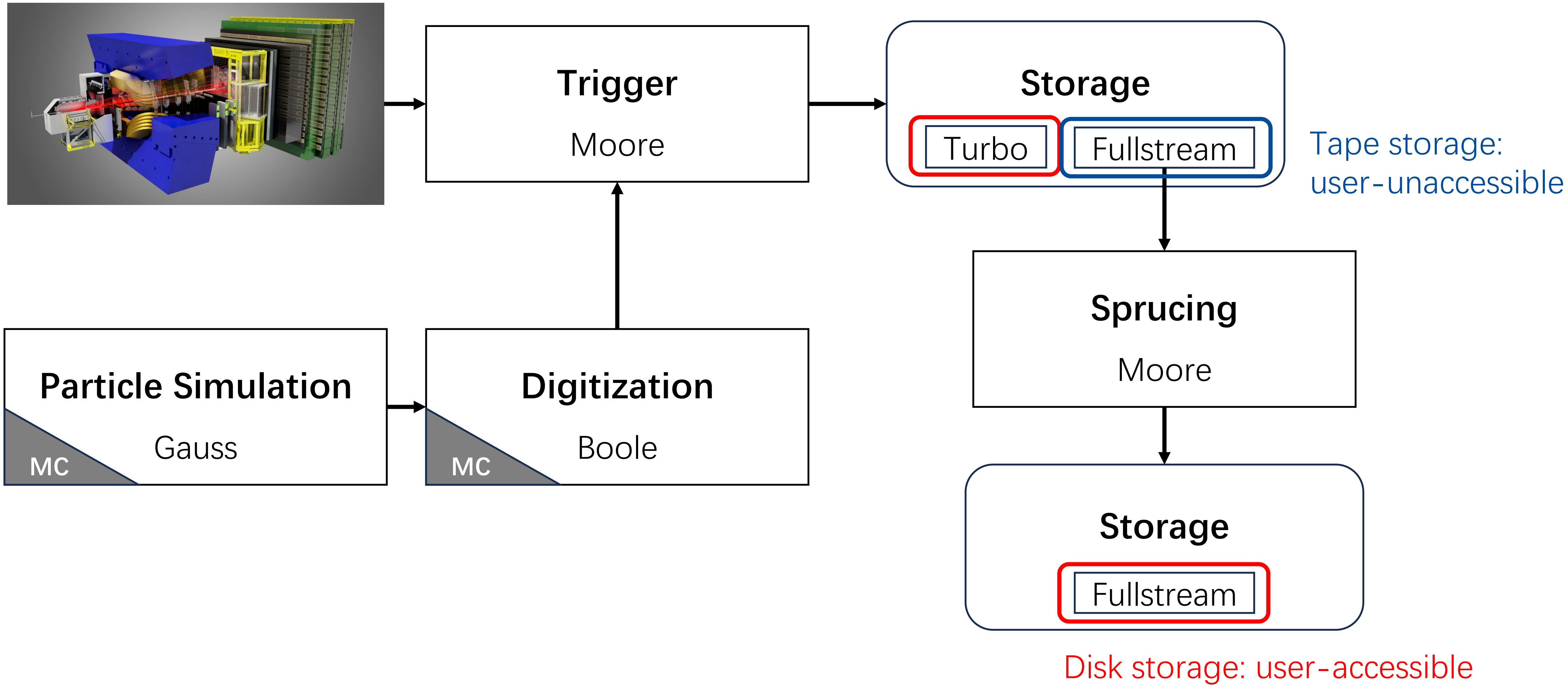


- **All these changes need to be reflected in LHCb software!**

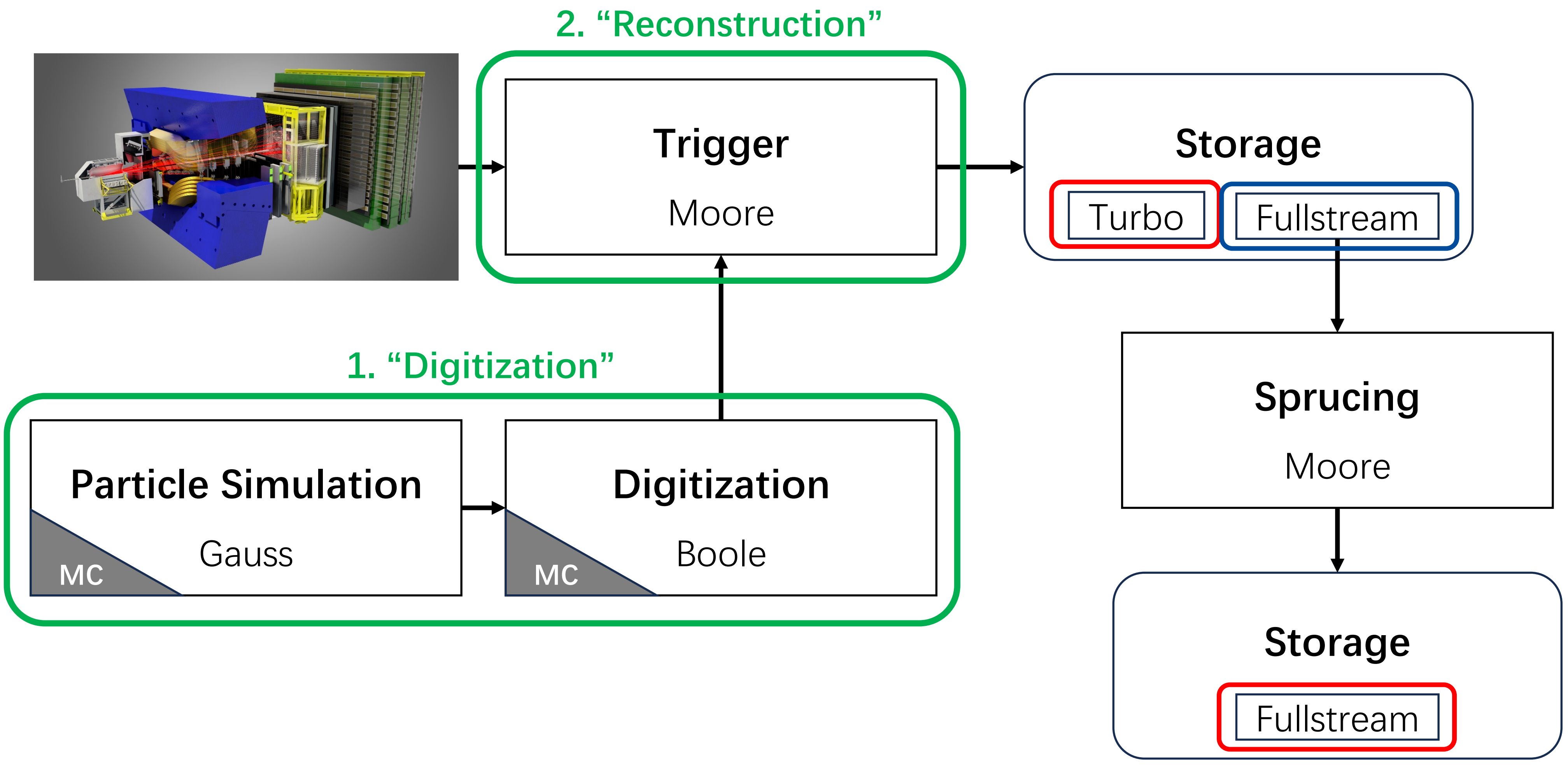
Software development for PicoCal

- New ECAL geometry description
- Update algorithm interfaces
- Re-implement pre-existing algorithms with new interfaces
- Implement new algorithms for new functionality in PicoCal
    - New frontend readouts
    - New reconstruction algorithm for time information and new geometry
- …

The LHCb software framework



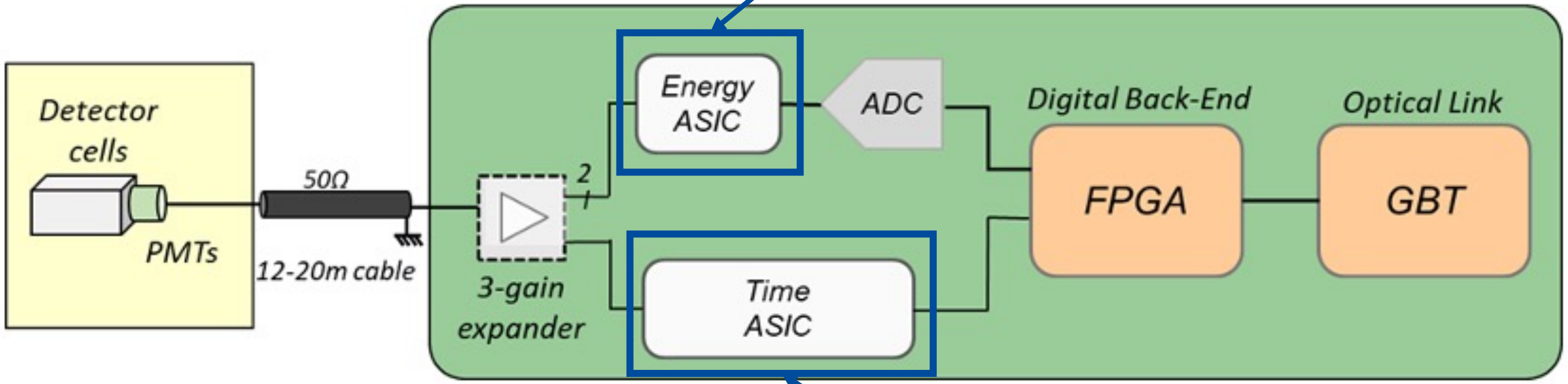
The LHCb software framework



Digitization in software

- Simulate the behavior of readout electronics
  - Information of simulated particles (truth-level) → Simulated readout
- New readout electronics in PicoCal:

ICECAL65: similar to ICECAL (U-I)



SPIDER: *New!*

[Summary of performance of the SPIDER ASIC v0, Baptiste Joly, Philippe Vallerand et. al @ PicoCal electronic meeting, April 10 2025]

Time readout in real electronics

What happens when a particle hits an ECAL cell?

1. Energy deposition, producing scintillating photons
2. Photons transportation
3. Photons collected by the PMT
4. PMT generates pulse signal
5. Sampling of the pulse
6. SPIDER ASIC extracts time and energy information from the pulse
7. Output digital results

Time digitization in software

- Can simulate the whole process with LHCb software, but too inefficient and resource-consuming for mass simulation production
- To save computing resources, a **faster method using templated pulse** is implemented
  - Based on a fact: the PMT pulses have consistent shape across different energies

The templated pulse approach

1. Energy deposition, producing scintillating photons
2. Photons transportation
3. Photons collected by the PMT
4. PMT generates pulse signal
5. Sampling of the pulse
6. SPIDER ASIC extracts time and energy information from the pulse
7. Output digital results

Replaced with fast parameterization (Gauss) (steps 1–3)

Simulated with template shape (Boole) (step 4)

Simulated with digitization algorithm (Boole) (steps 5–7)

Time digitization – software roadmap

1. Read time and photon yield (~energy) information of simulated particles
2. Generate pulses with template shapes
   - Cell PMT response = Σ(particle PMT response)
   - Electronic noises, shape variation etc. are added in this step
   - Pulses are sampled in this step
3. Simulate SPIDER response, output result

Generate pulse (steps 1–2)

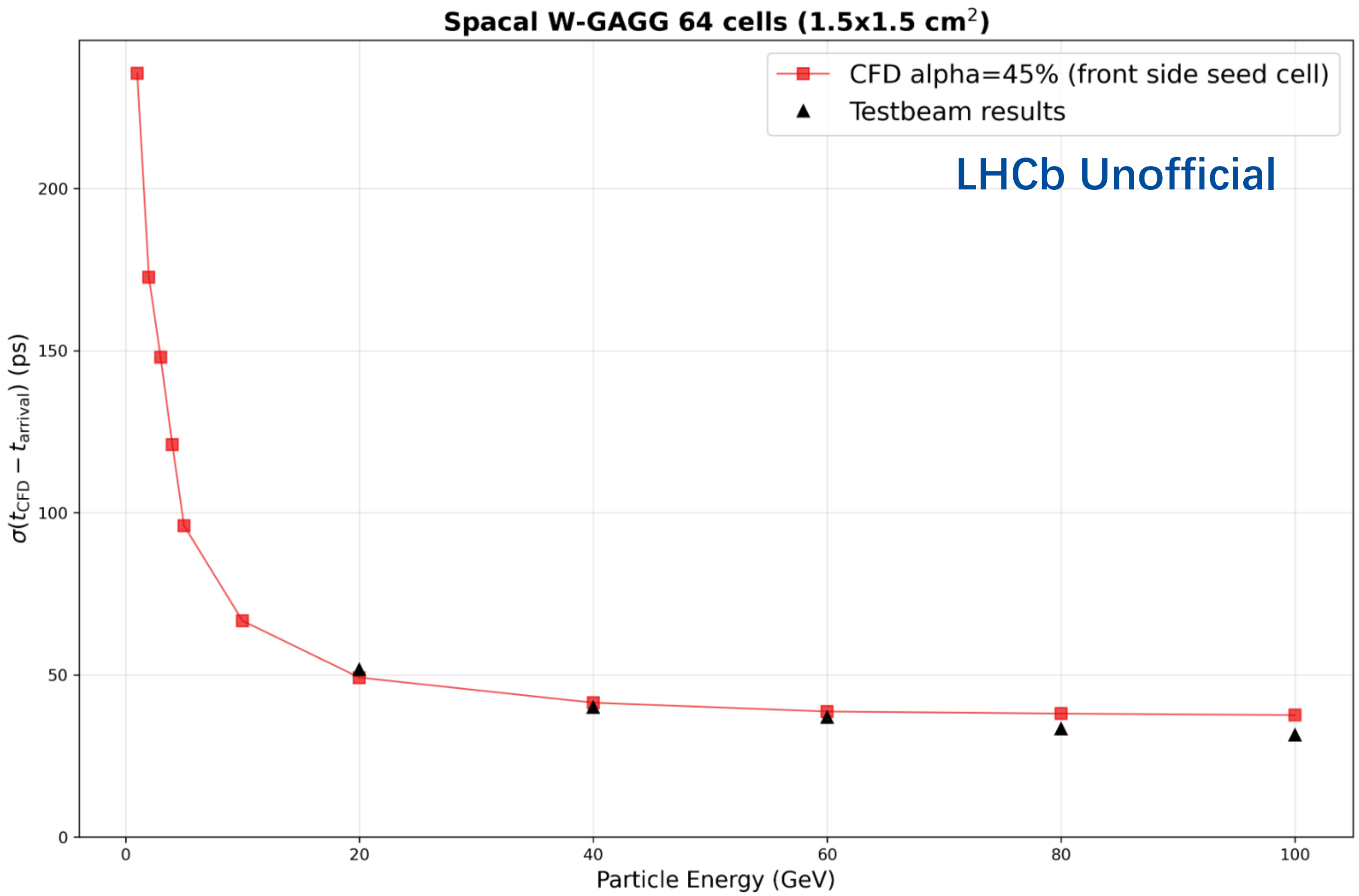
Calculate timestamp (step 3)

Finding template shape(s)

- The consistent PMT pulse shape we want can be found in test beam!
- To avoid:
  - … using a certain pulse shape from a single event
  - … including the electronic noises in the template

the (initial) templates are acquired by **fitting test beam pulses** with a **parameterized empirical function**

Time digitization – Validation



Energy readout in real electronics

Similar to time digitization:

1. Energy deposition, producing scintillating photons
2. Photons transportation
3. Photons collected by the PMT
4. PMT generates pulse signal
5. ICECAL65 ASIC: pulse re-shaping, integration
6. Output digital results

Energy digitization in software

A fast, parameterized digitization method is implemented:

1. Energy deposition, producing scintillating photons
2. Photons transportation
3. Photons collected by the PMT

Replaced with fast parameterization (Gauss)

4. PMT generates pulse signal
5. ICECAL65 ASIC: pulse re-shaping, integration
6. Output digital results

Replaced with fast parameterization (Boole)

Energy digitization – software roadmap

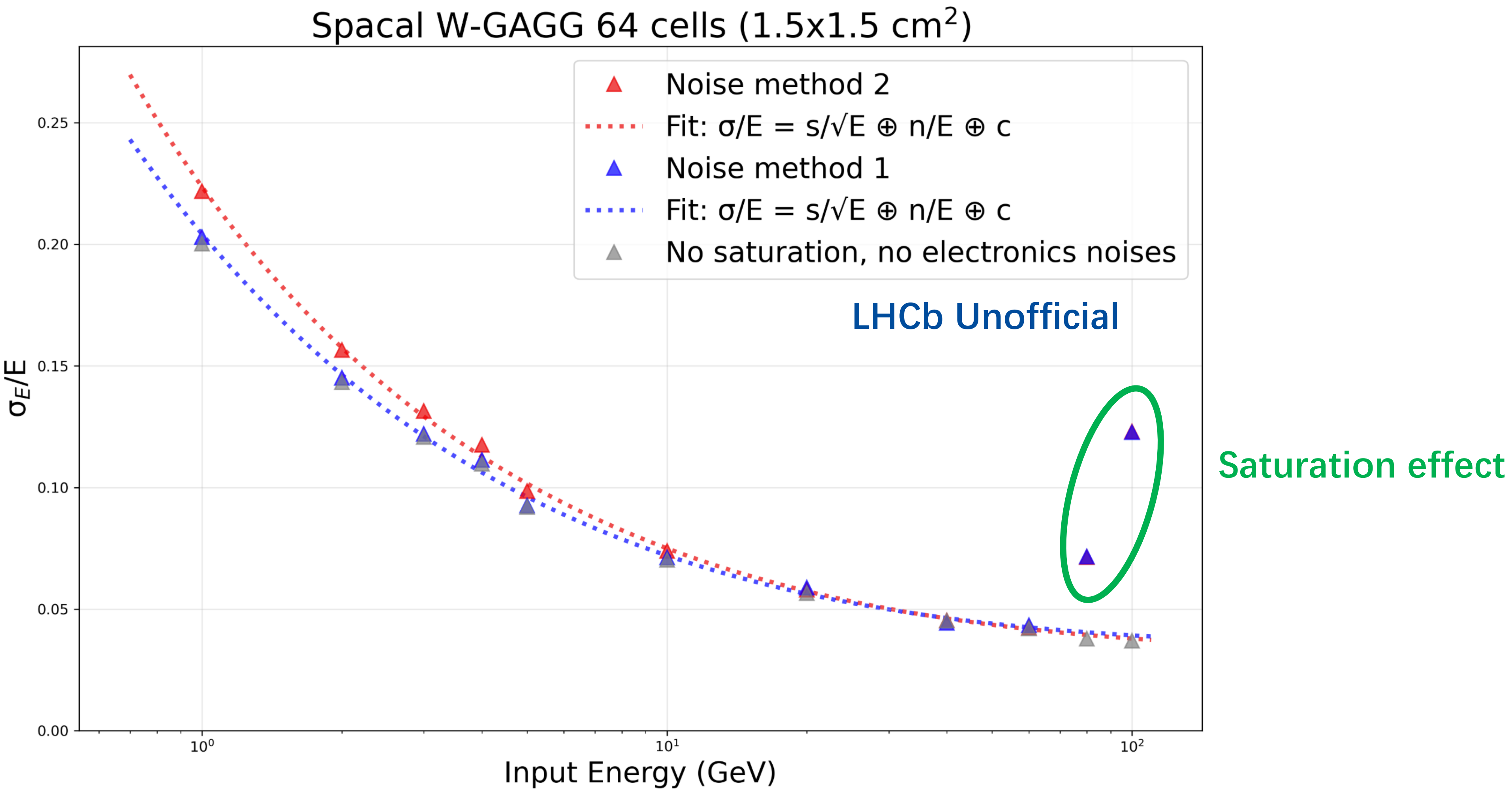
1. Read energy information of simulated particles

2. Add electronic noise and gain noise to the true energy

3. Determine if the cell is saturated

4. Output readout energy value

Energy digitization – results

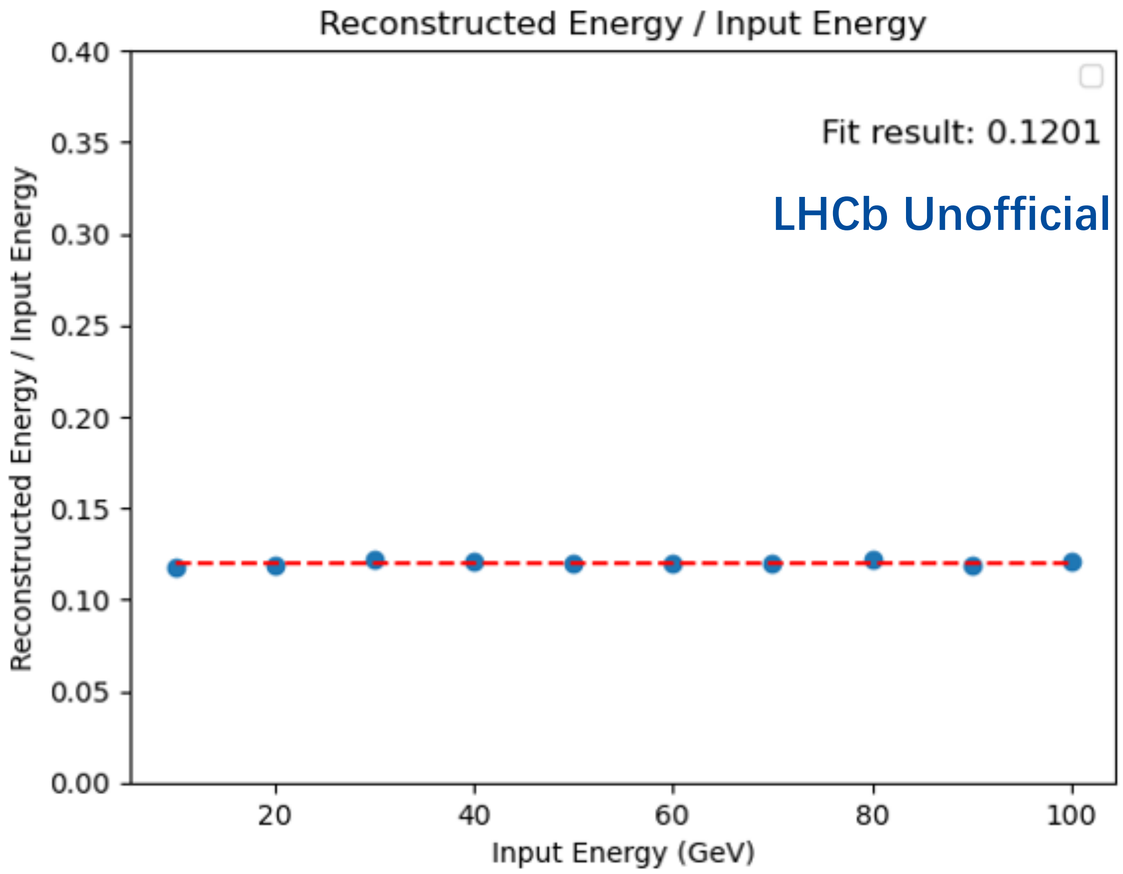


Software development for reconstruction

- Integrated the new PicoCal namespaces in the related LHCb software (Rec and Moore)
- Implemented data objects and interfaces required by the core reconstruction algorithm GraphClustering
- Successfully reconstructed Run 5 ECAL clusters from simulated hits (MCHit) using the new framework

Reconstruction algorithm – Validation

- **The reconstructed cluster energy is consistent with particle energy across a wide range of energies**



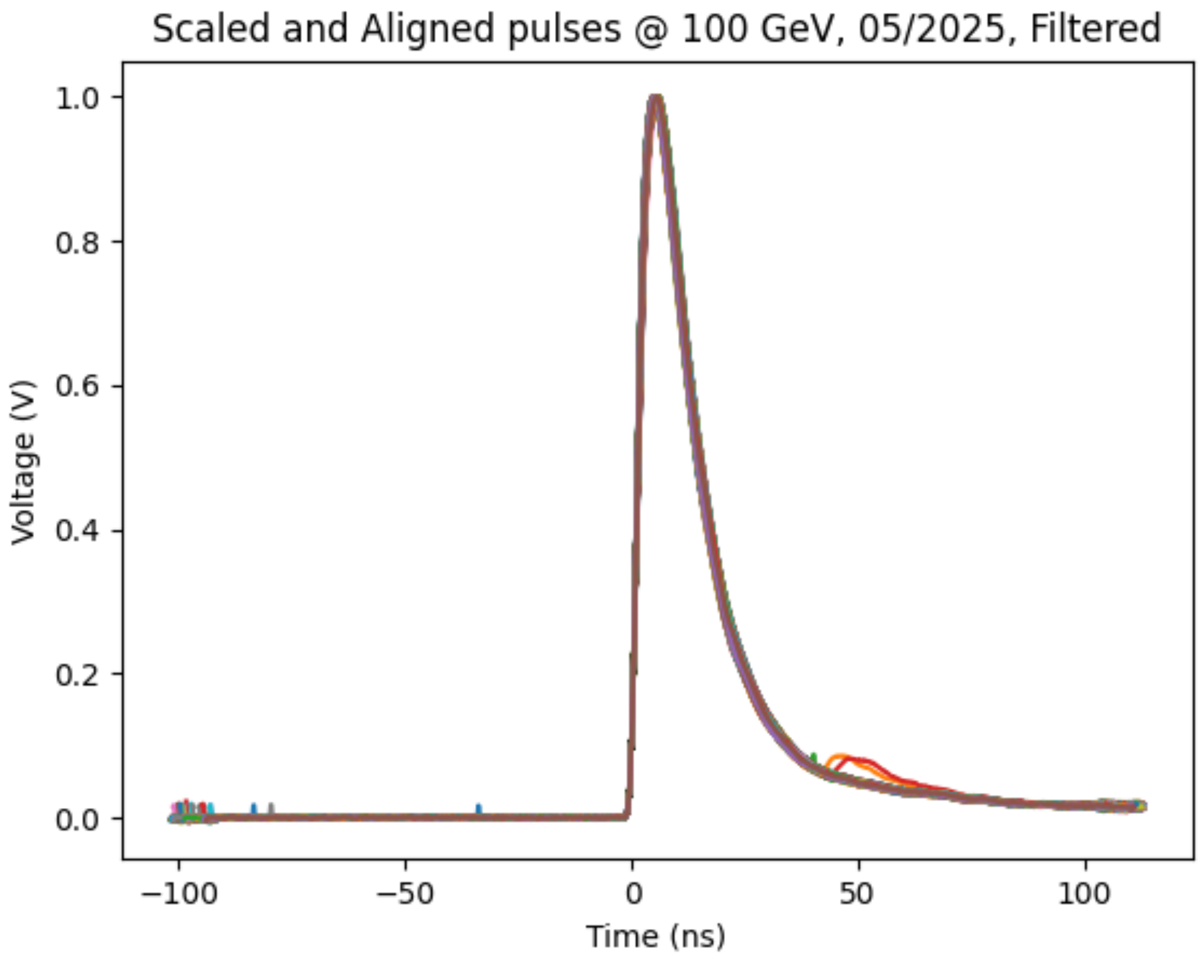
Summary

- The following developments in the LHCb software for the PicoCal project:
  - Time digitization: implemented the new time digitization algorithm; successfully outputted time information for ECAL cells with workable time resolution
  - Energy digitization: re-implemented digitization algorithm for PicoCal, σ-E relation consistent with model
  - Reconstruction: implemented interfaces for the core reconstruction algorithm, successfully reconstructed clusters with simulated hits, consistency verified
  - Other prerequisite developments: data structure and output modification in Gauss, etc.

Thank you for listening!

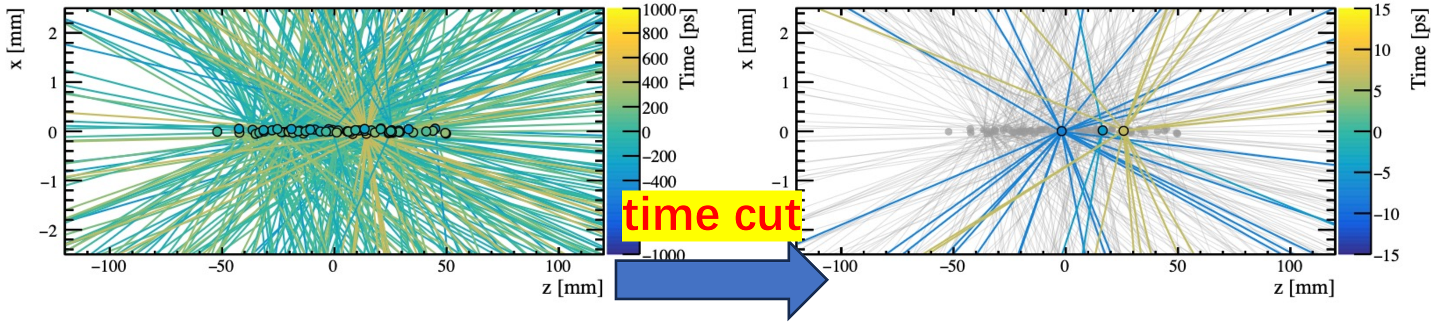
BACKUP

Pulse shape consistency

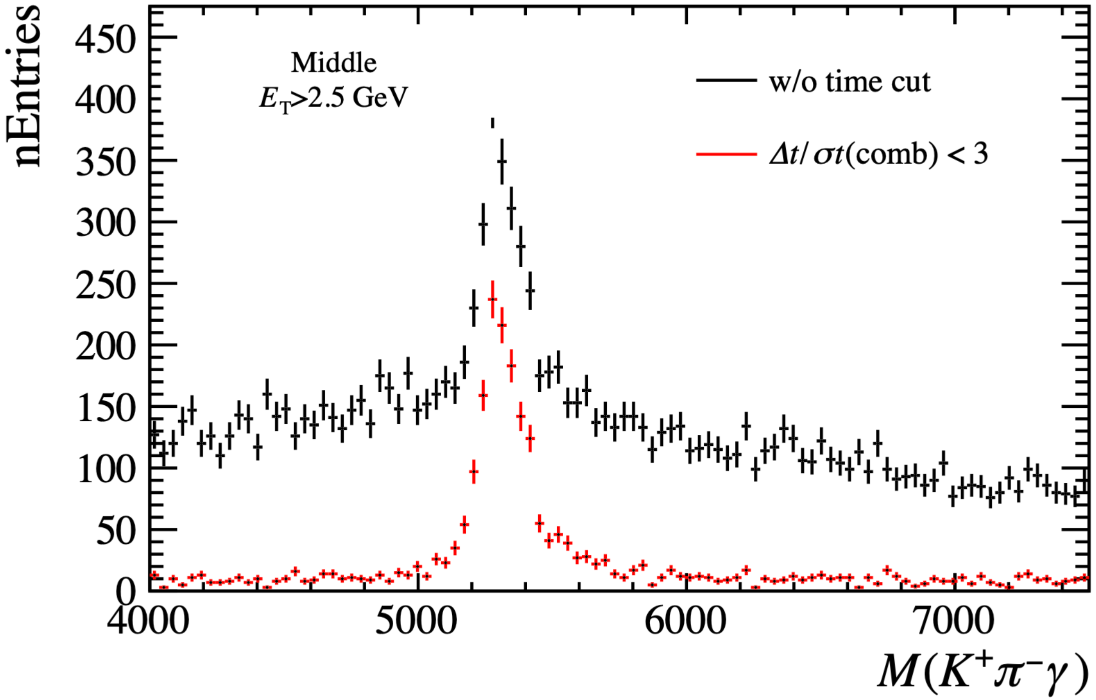


[Test beam data of May 2025, 100GeV @ SPS, from Loris Martinazzoli & Emile Caire]

Why we need timing for U-II?



Track visualization for a 42 PV event (Left) Without timing and (Right) Within a 30ps time window
[LHCb Upgrade II Scoping Document, LHCb Collaboration, https://cds.cern.ch/record/2903094/]



$B^0 \to K^{*0}\gamma$ spectrum w/ & w/o ECAL timing cut
[LHCb Upgrade II Scoping Document, LHCb Collaboration, https://cds.cern.ch/record/2903094/]

Empirical function for the fit

$$F(t; A, t_0, \tau_0, f_i, \tau_i) = A \cdot (t - t_0) \cdot \left[1 - e^{-((t-t_0)/\tau_0)^2}\right] \cdot \sum_{i=1}^{2} f_i \, e^{-(t-t_0)/\tau_i}$$

$(\sum_{i=1}^{2} f_i = 1)$

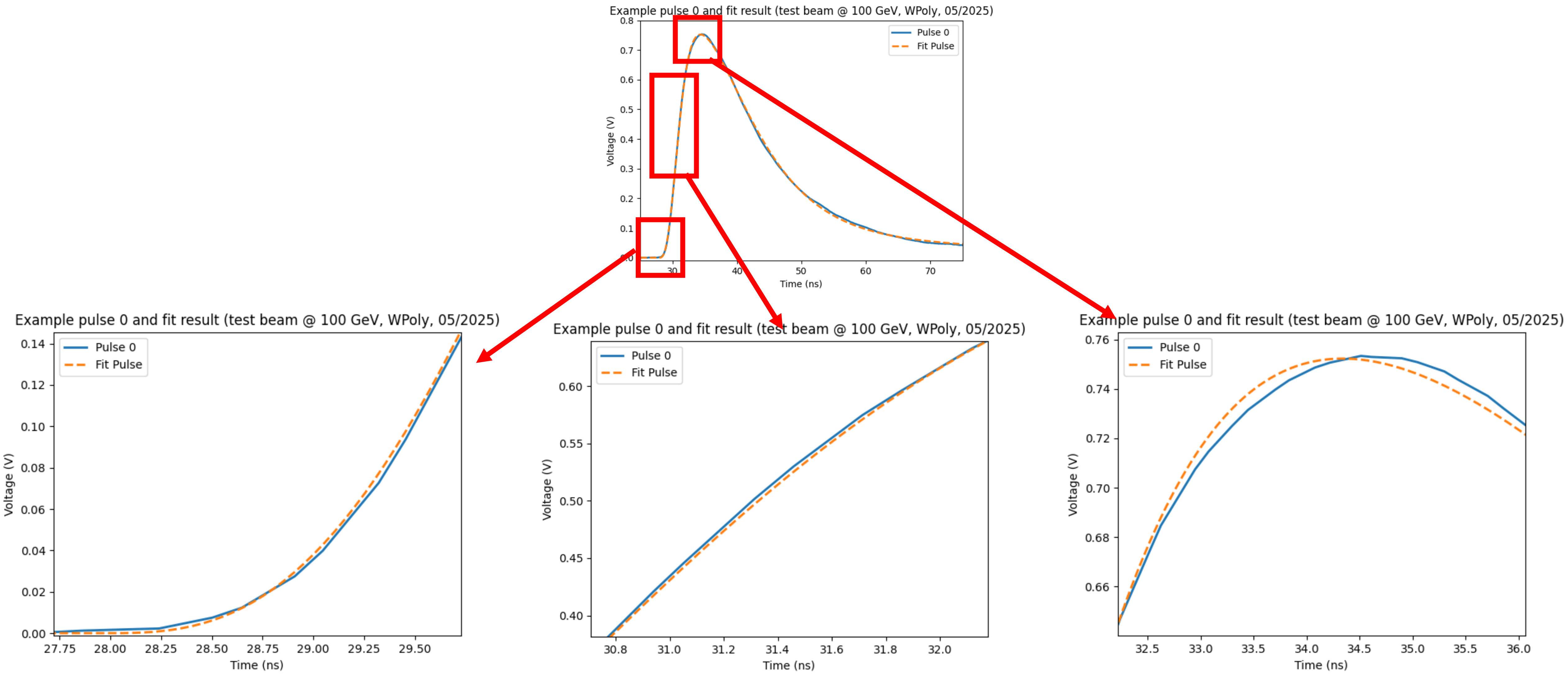
t_0: pulse start time

τ_0: ~rise time

τ_i: photon decay time

f_i: component fractionx

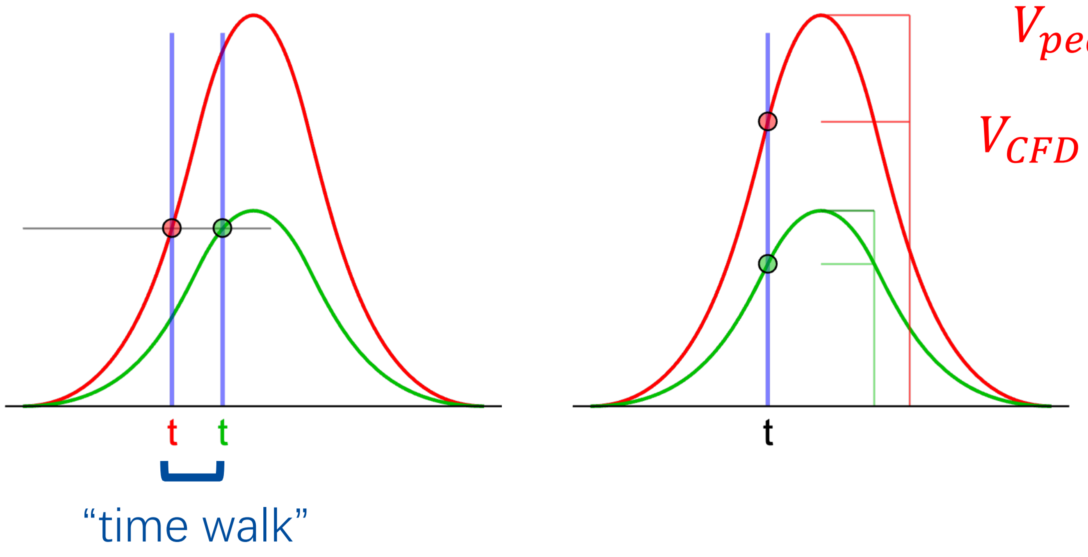
Imperfectness of the template function



SPIDER design

- ADC readout noise: $< 0.5\ \mathrm{mV}$
- Input voltage dynamic range: $V_{in} \in [8\ \mathrm{mV},\ 800\ \mathrm{mV}]$
  - PicoCal target: time measurement for $E_T \in [50\ \mathrm{MeV},\ 5\ \mathrm{GeV}]$

Digital CFD algorithm

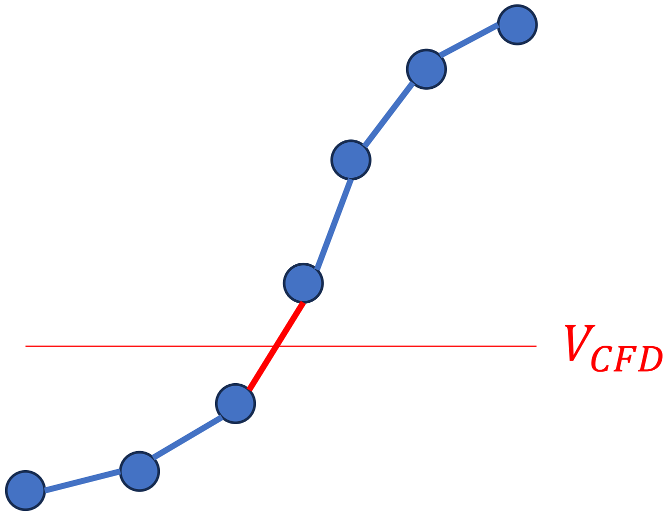


V_{peak}

$V_{CFD} = f_{CFD} \cdot V_{peak}$

$(f_{CFD} \in [0, 1])$

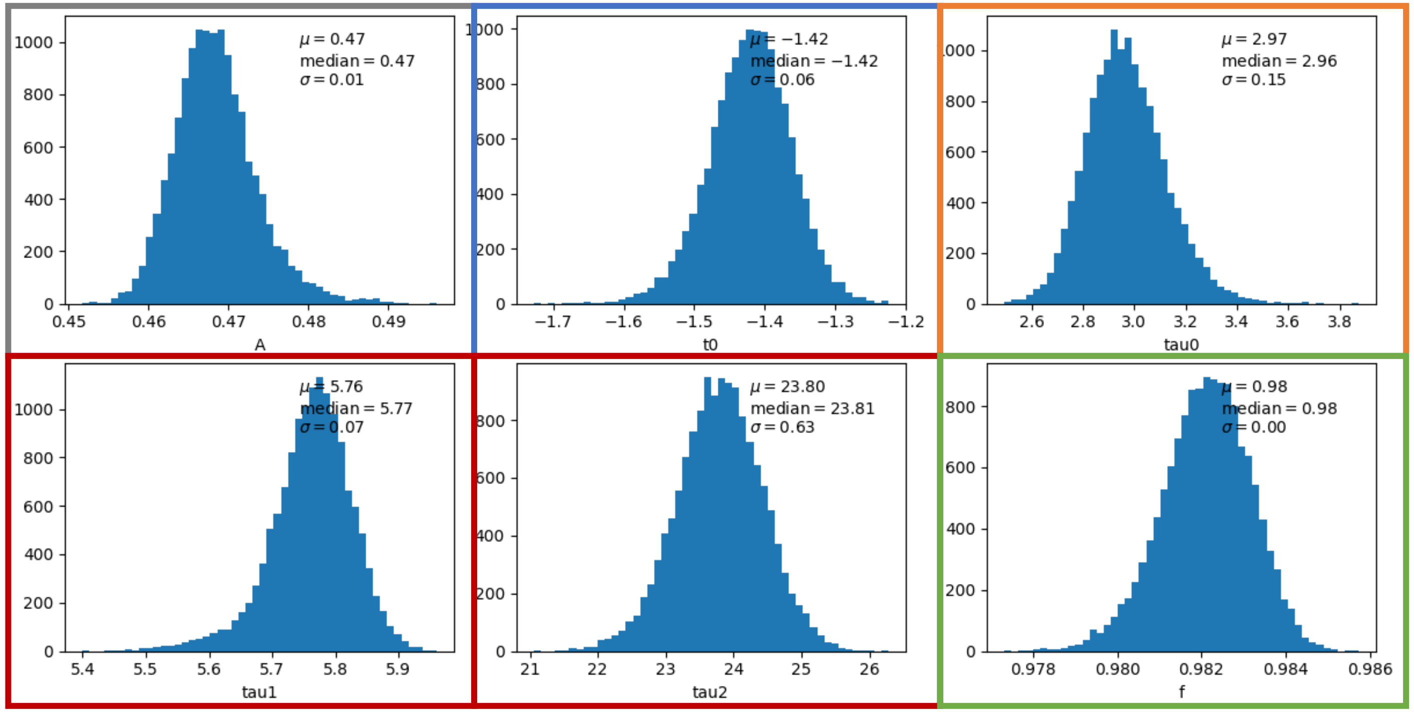
CFD diagram



Digital CFD

Method for getting templates

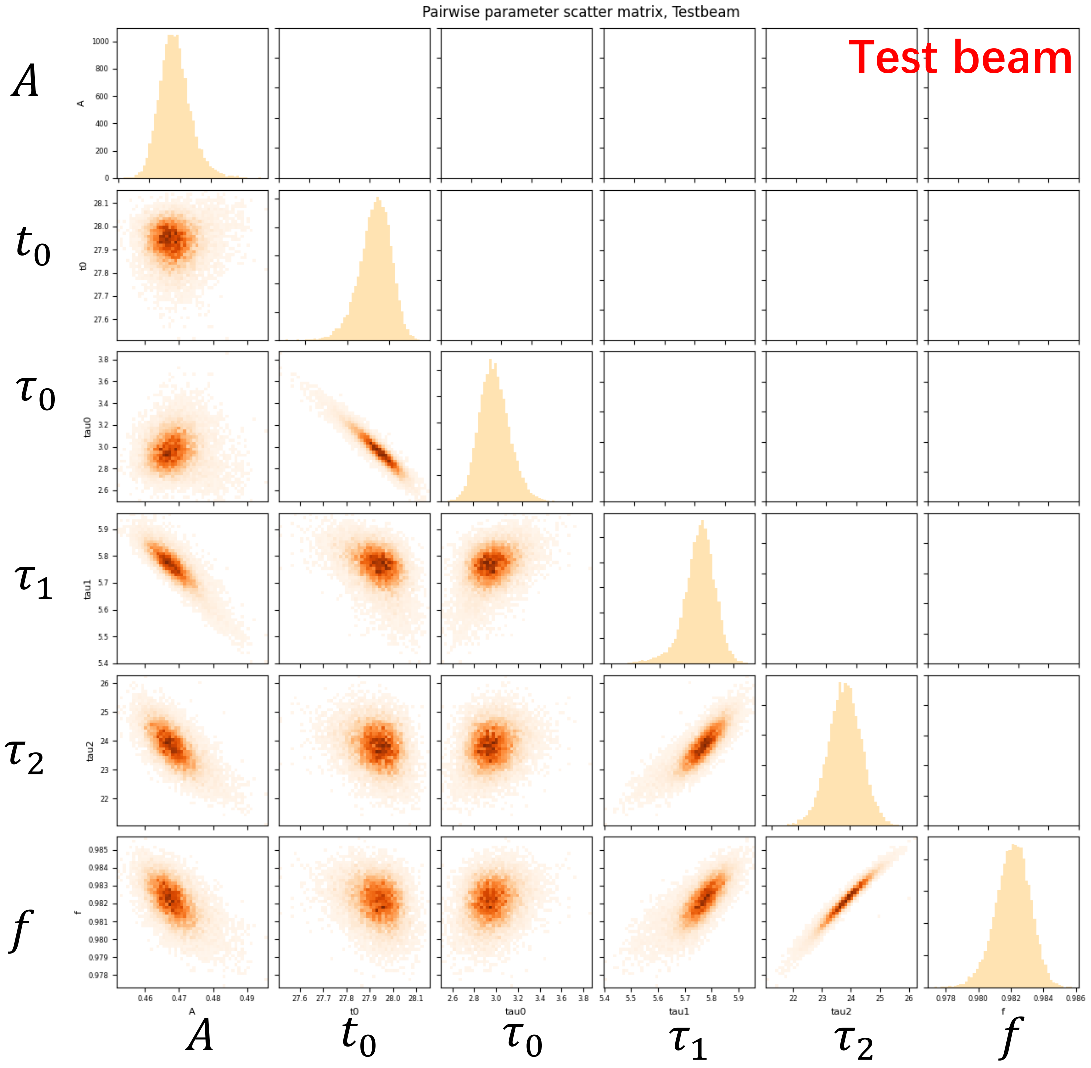
1. Determine the empirical function
2. Fit $\mathcal{O}(10^4)$ test beam pulses, get parameter sets
3. Produce toy parameter sets of the same distribution

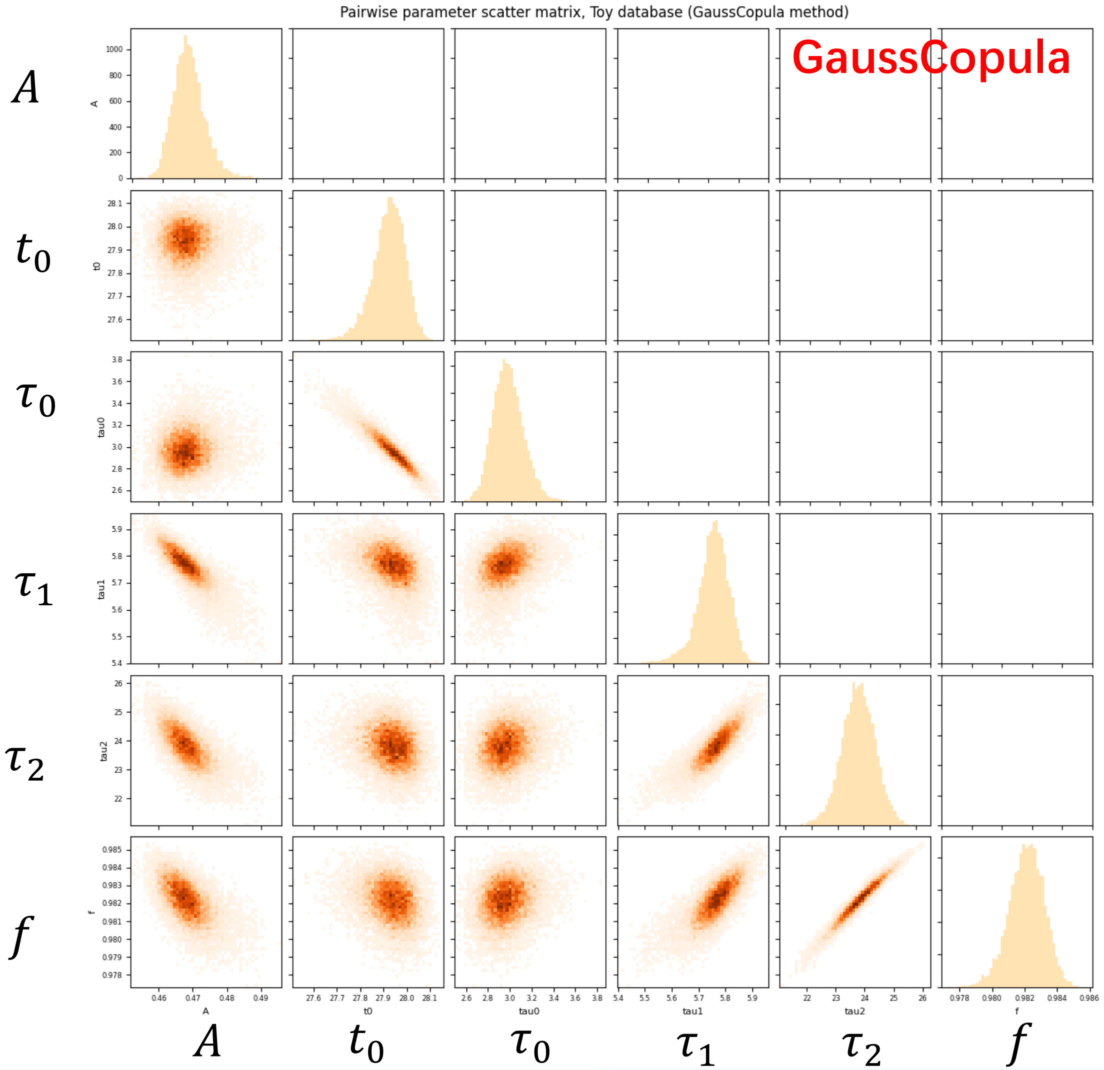


GaussCopula method

- Conduct a specific transformation T to the parameter sets
  - T regards only the marginal distributions
- Estimate the resulted joint distribution with n-D Gaussian distribution
  - The estimation have same covariant matrix with the joint distribution
- Generate toy parameter sets, do the inverse transformation T^{-1}

GaussCopula method





Gaussian Mixture Model (GMM) method

- Estimate the joint distribution with several n-D Gaussian distribution