

Software development for LHCb ECAL upgrade

Zifan Li¹ (on behalf of the LHCb ECAL U-II group)

CLHCP2025, 2025/10/31

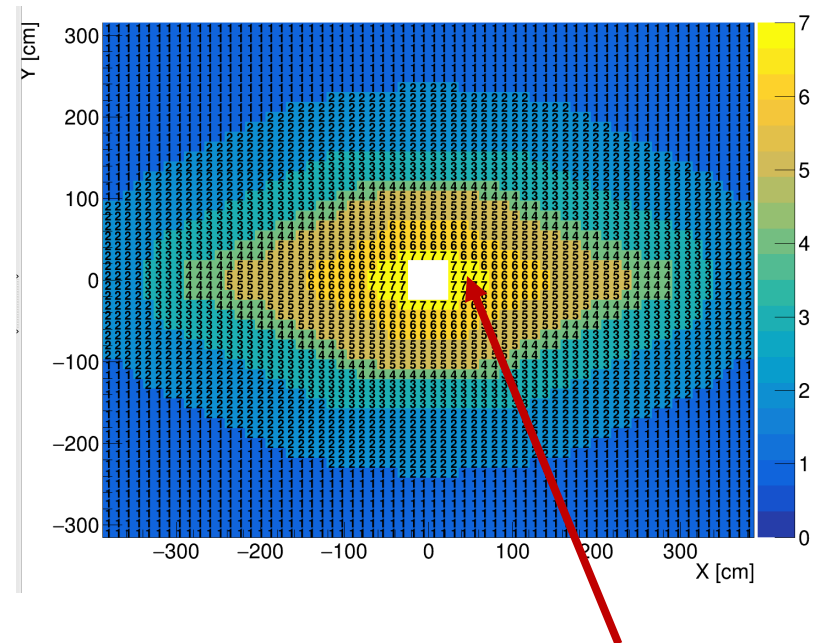
¹Peking University

Contents

1. The LHCb ECAL in Upgrade II
2. The LHCb Software framework
3. Software development
 1. Time Digitization
 2. Energy Digitization
 3. Reconstruction

PicoCal – the LHCb ECAL in Upgrade II

- In Upgrade II, the LHCb experiment will have:
 - Higher luminosity: $\mathcal{L} = 1.5 \times 10^{34} \text{ cm}^{-2} \text{ s}^{-1}$
 - 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



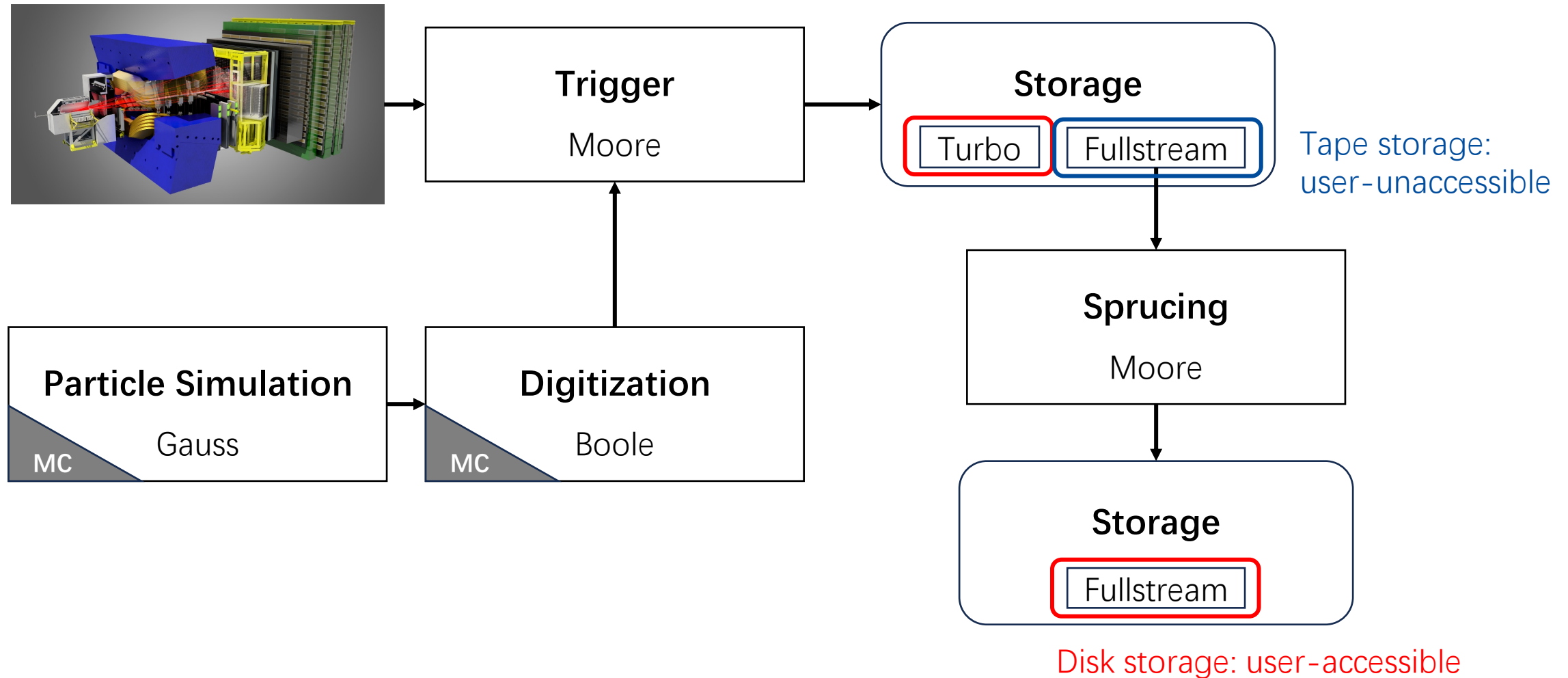
W-GAGG SPACAL

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

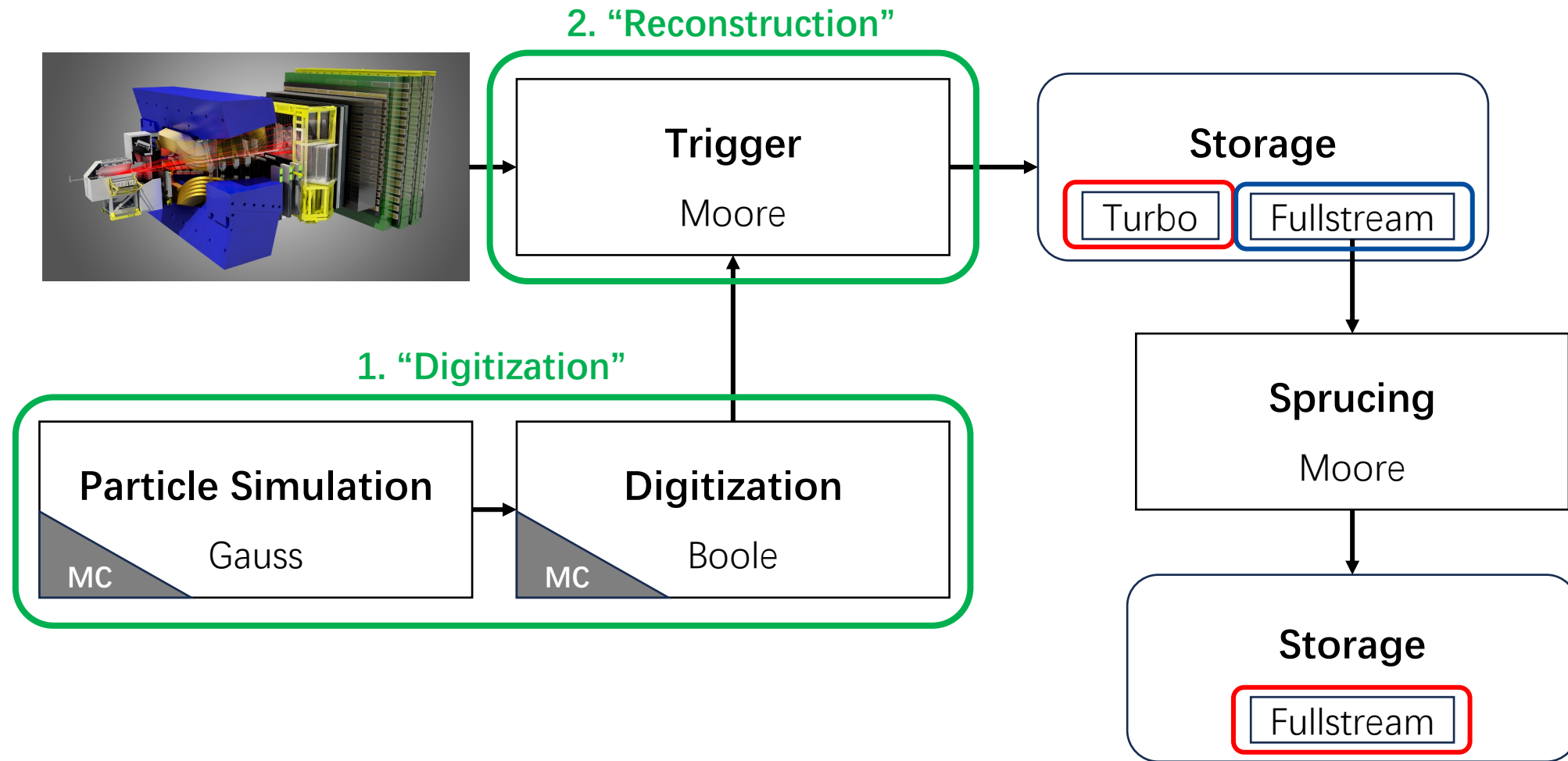
Key aspects for software development

- 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

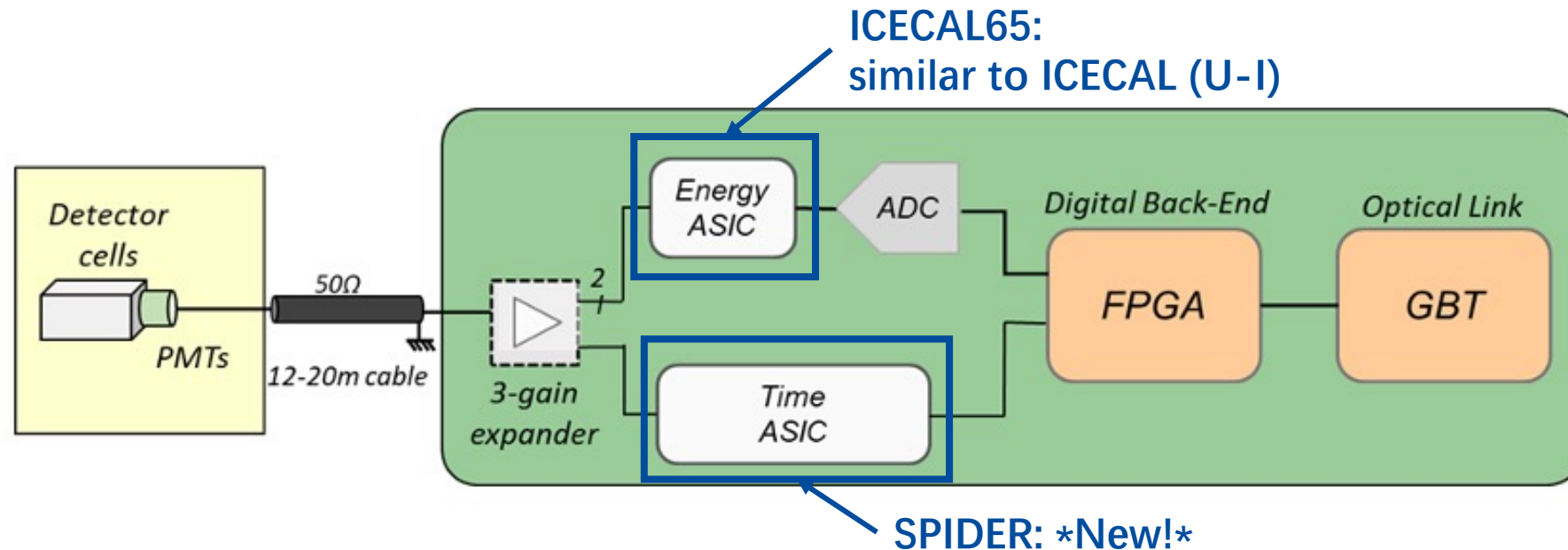


The LHCb software framework



Digitization in real electronics

- New readout electronics in PicoCal:



[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

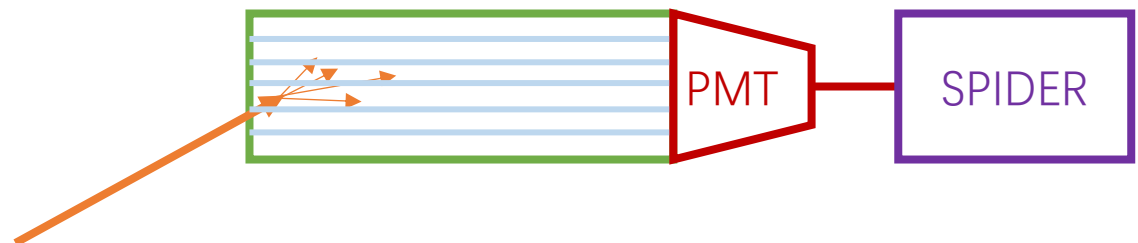
What happens when a particle hits an ECAL cell?

1. Energy deposition, producing scintillating photons
2. Photons transportation
3. Photons collected by the PMT
4. PMT generates pulse signal
5. SPIDER ASIC samples the pulse
6. Extract time information from the sampled 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. SPIDER ASIC samples the pulse
 6. Extract time information from the sampled pulse
 7. Output digital results
- Replaced with fast parameterization (Gauss)
- Simulated with template shape (Boole)
- Simulated with digitization algorithm (Boole)

Time digitization – software roadmap

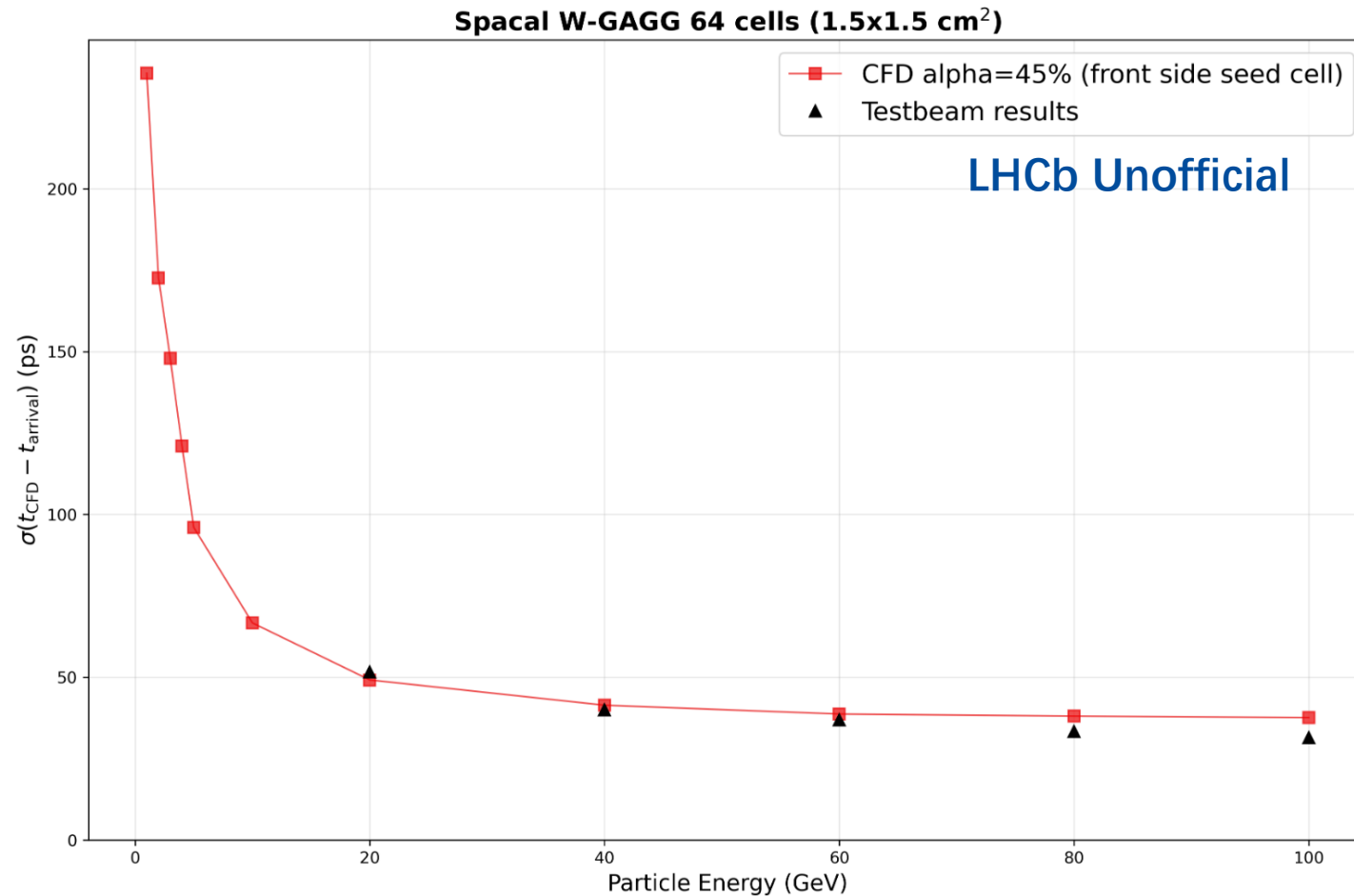
1. Read time and photon yield ($\sim$ energy)
information of simulated particles
 2. Generate pulses with template shapes
 - Cell PMT response = Σ (particle PMT response)
 - Electronic noises, shape variation etc.
are added in this step
 - Pulses are sampled in this step
 3. Simulate SPIDER response,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

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

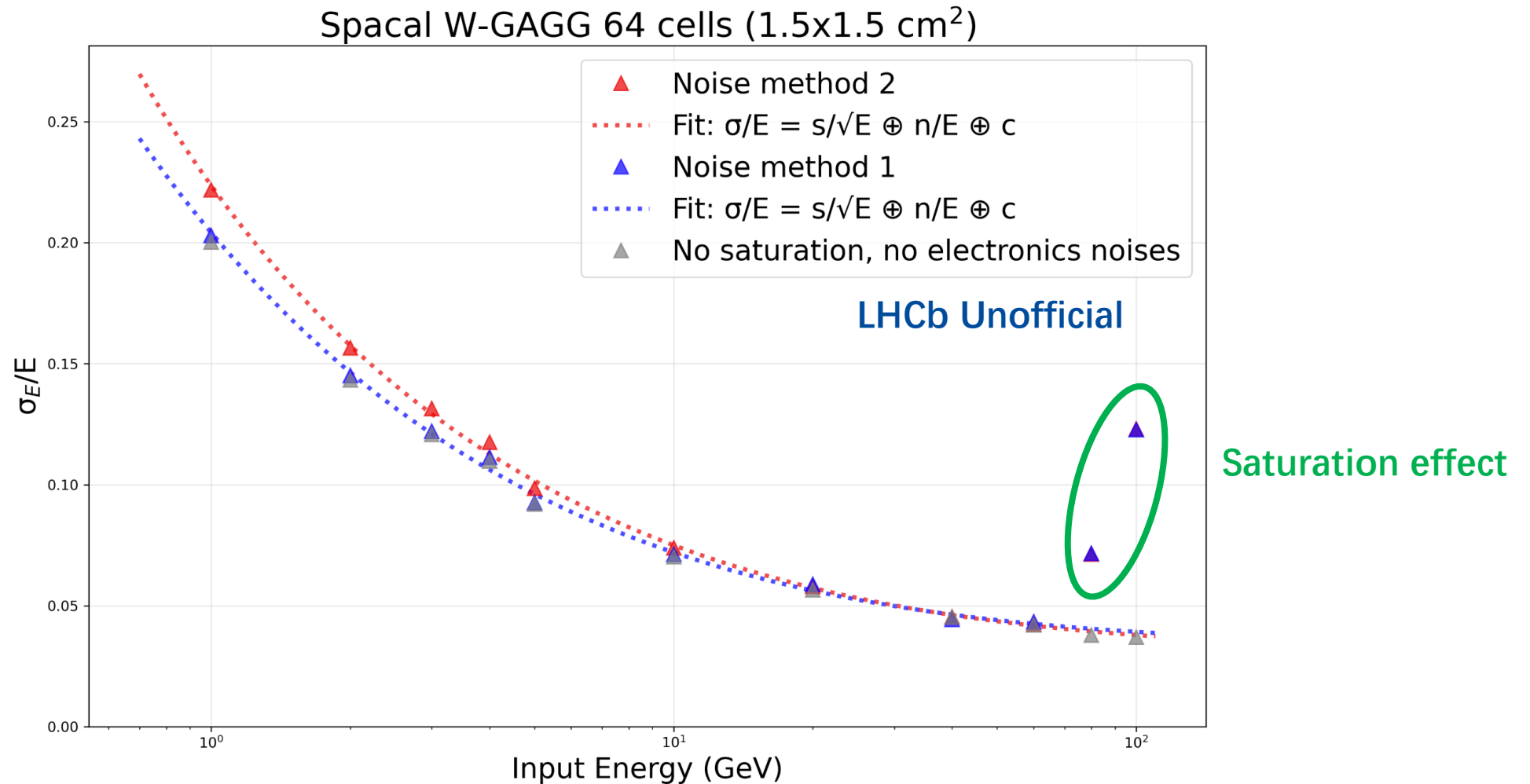
A fast, parameterized digitization method is implemented:

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

Energy digitization – software roadmap

1. Read energy information of simulated particles
2. Add photoelectron number smearing
3. Add electronic noise and gain noise
4. Determine if the cell is saturated
5. Output readout energy value

Energy digitization – results

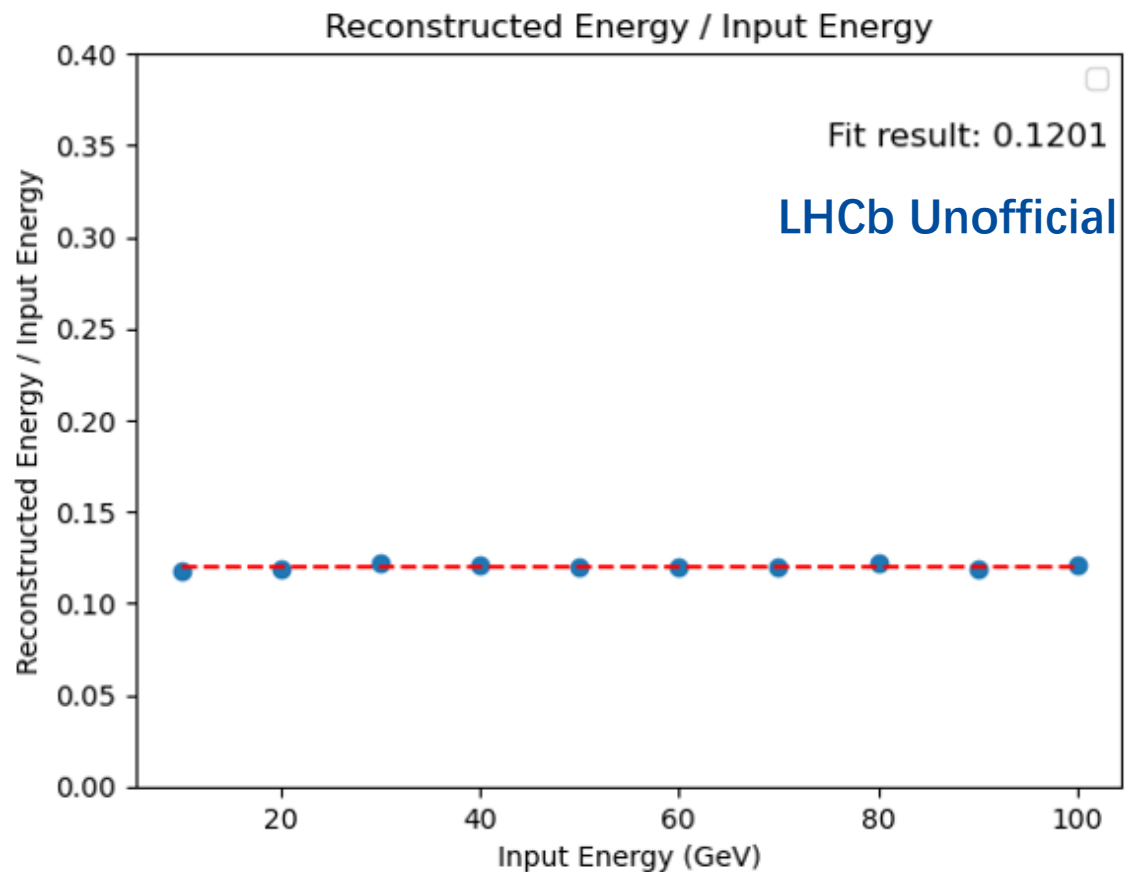


Software development for reconstruction

- 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

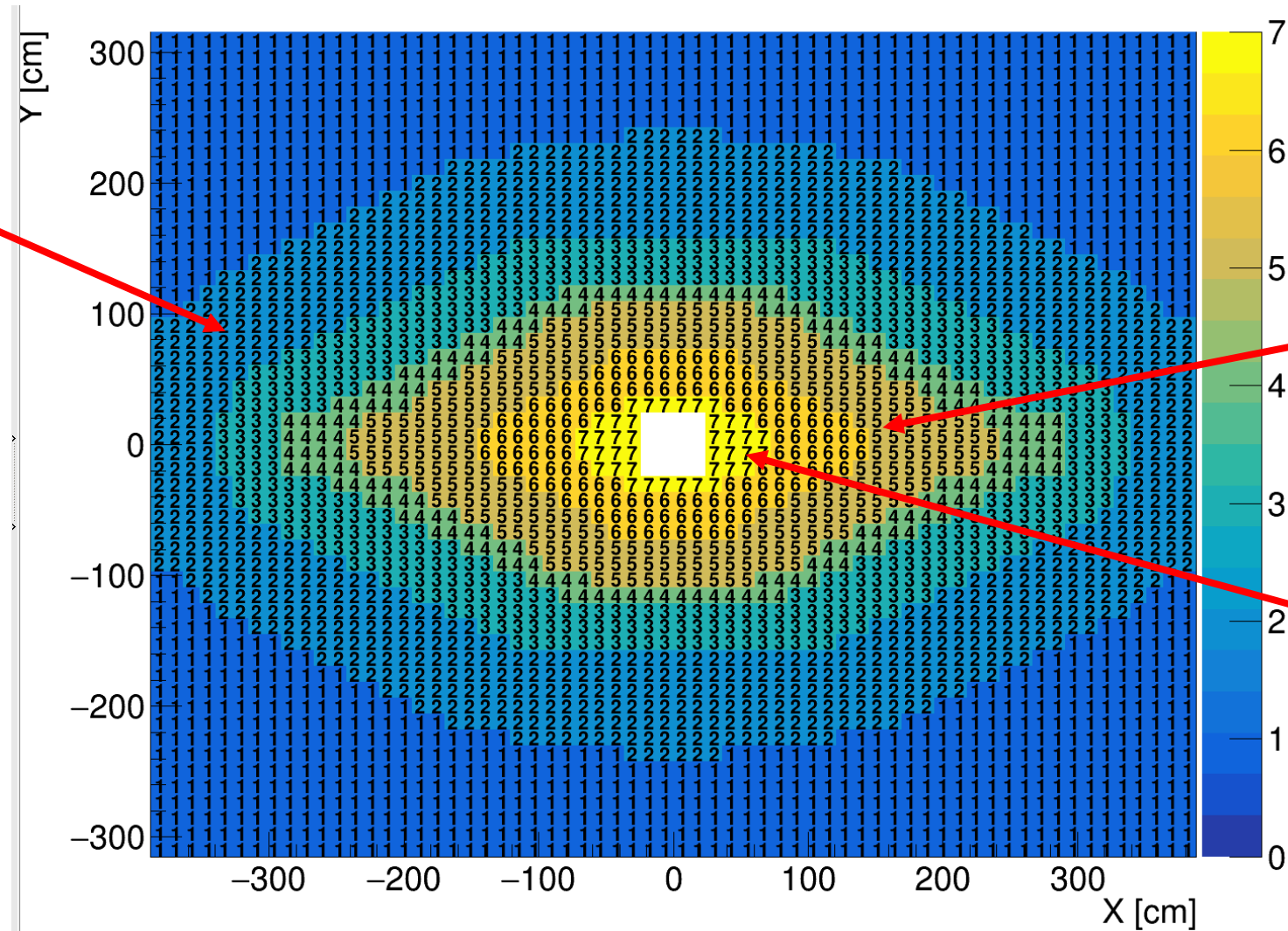
- Time digitization: New time digitization algorithm; successfully outputted time information with reasonable resolution
- Energy digitization: re-implemented digitization algorithm for PicoCal, σ - E relation consistent with model
- Reconstruction: implemented interfaces for the core algorithm, successfully reconstructed clusters with simulated hits, consistency verified
- Other prerequisite developments: data structure and output modification in Gauss, etc.

Thank you for listening!

BACKUP

Geometry of PicoCal (2025 baseline)

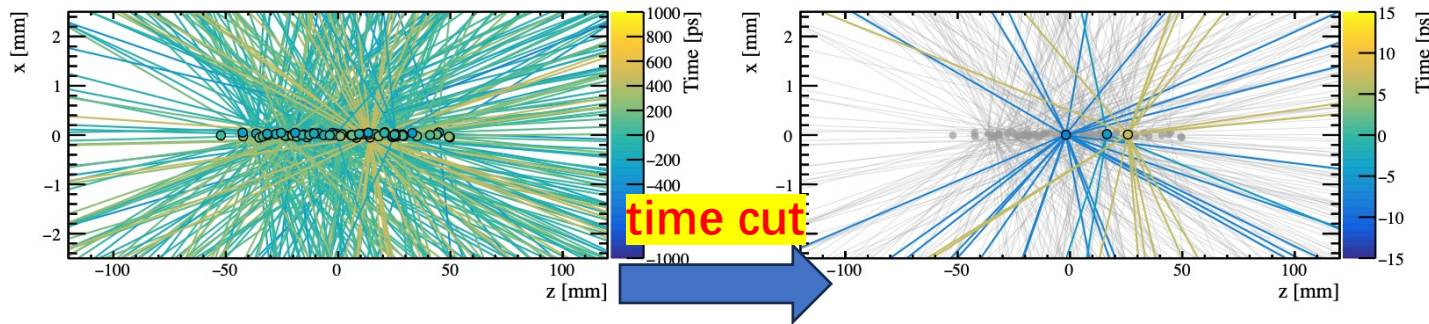
Shashlik
(Same design as U-I)



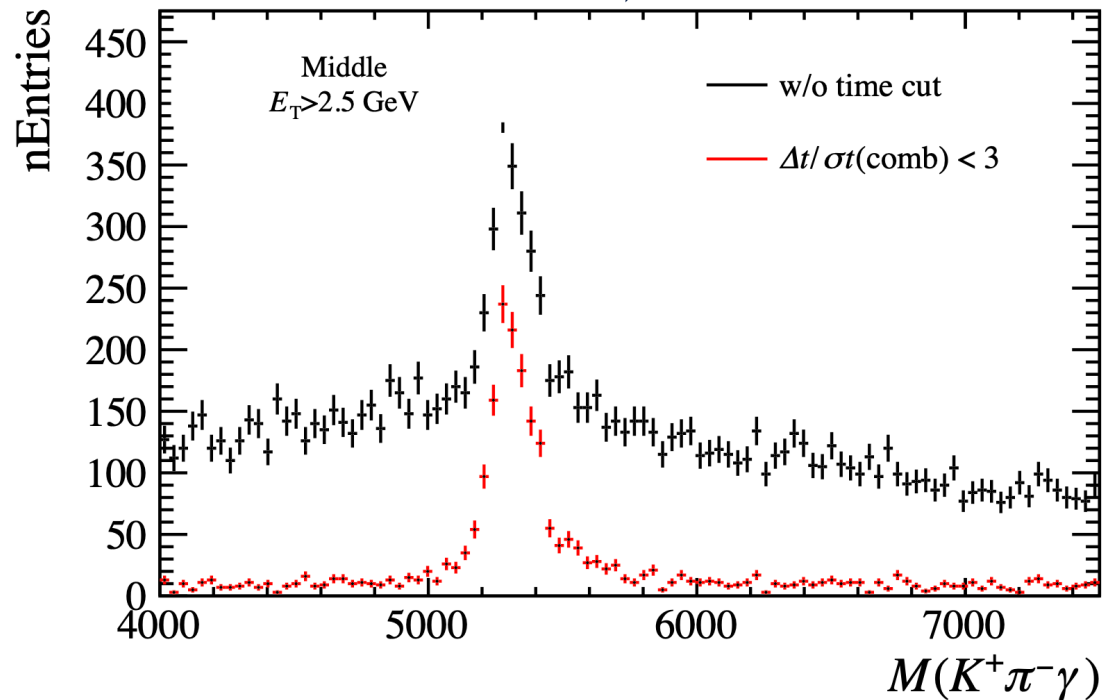
Pb-Poly SPACAL

W-GAGG SPACAL

Why we need timing for U-II?

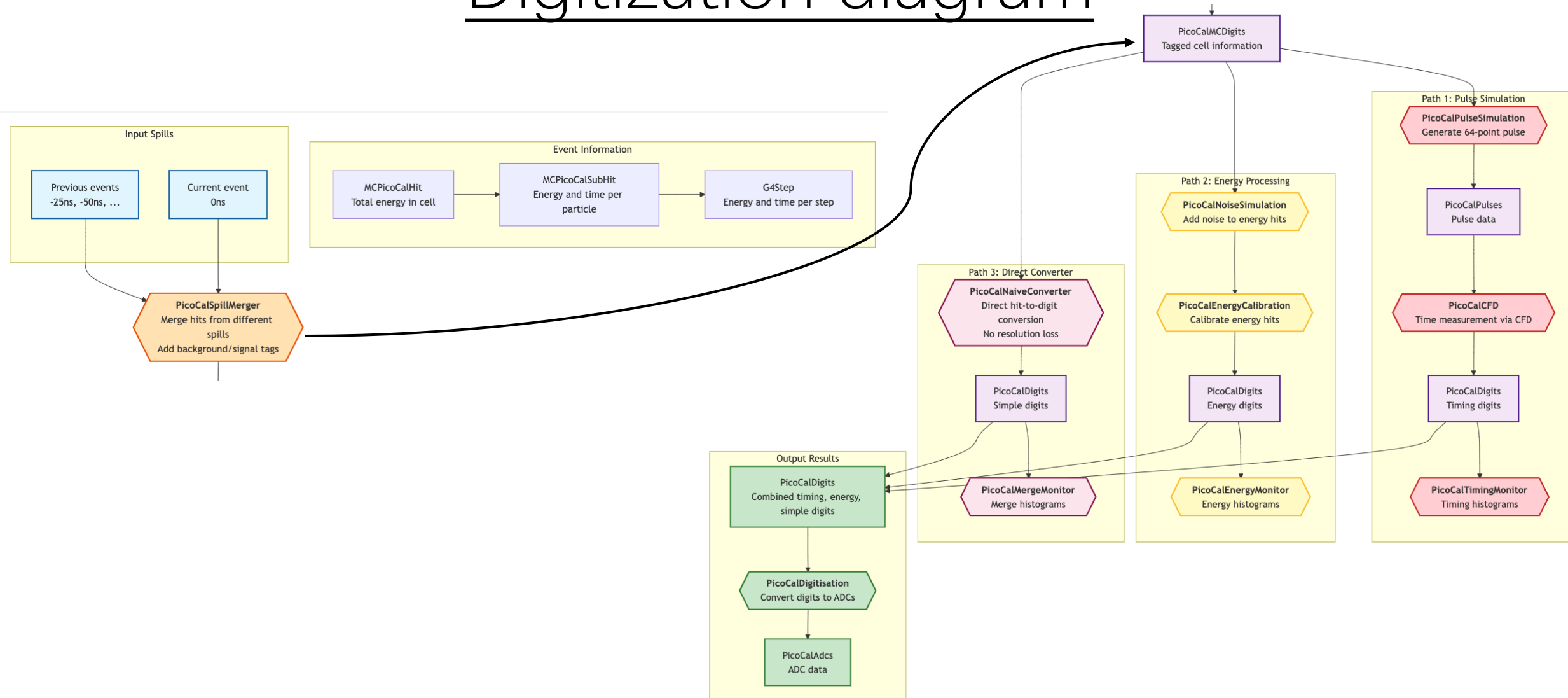


[LHCb Upgrade II Scoping Document, LHCb Collaboration, <https://cds.cern.ch/record/2903094/>]

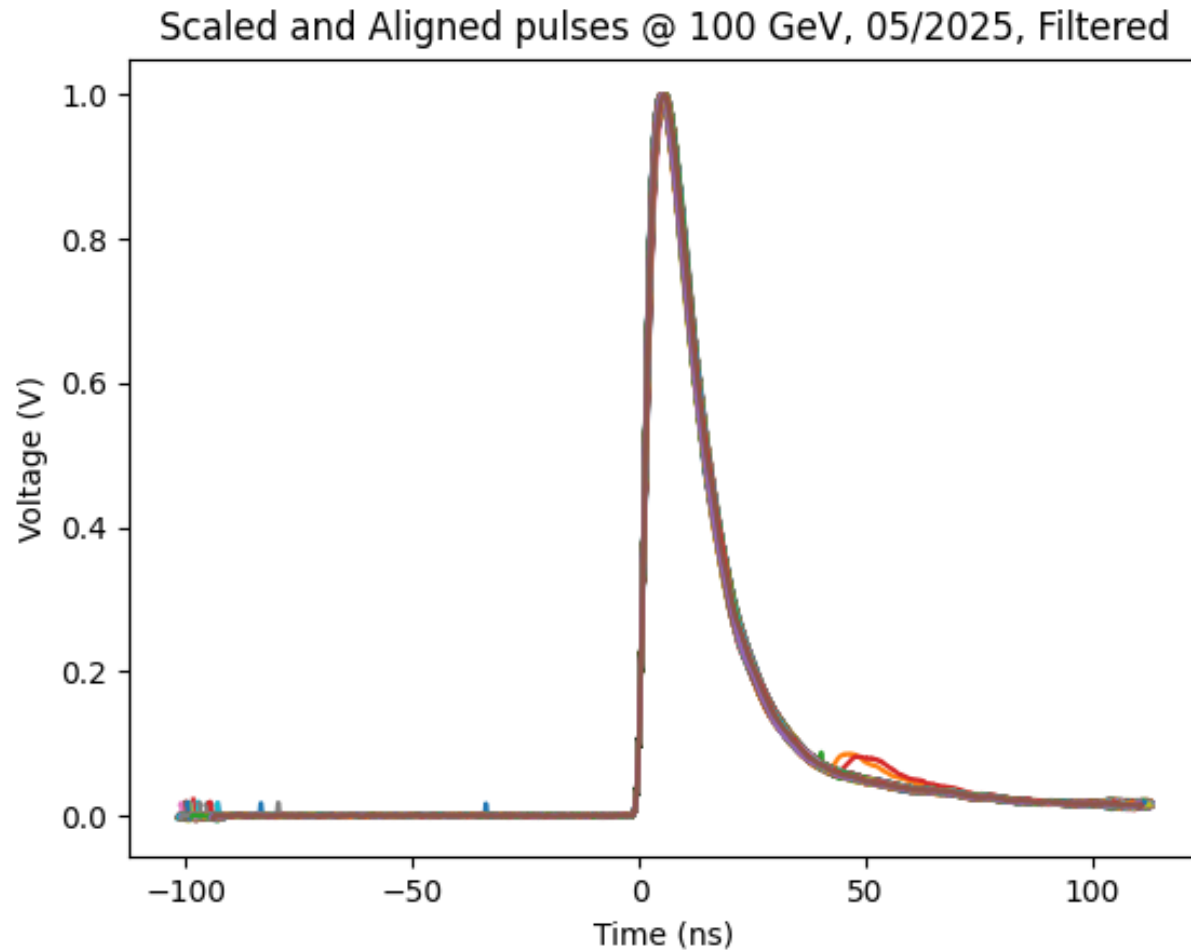


[LHCb Upgrade II Scoping Document, LHCb Collaboration, <https://cds.cern.ch/record/2903094/>]

Digitization diagram



Pulse shape consistency



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

Empirical function for the fit

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

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

t_0 : pulse start time

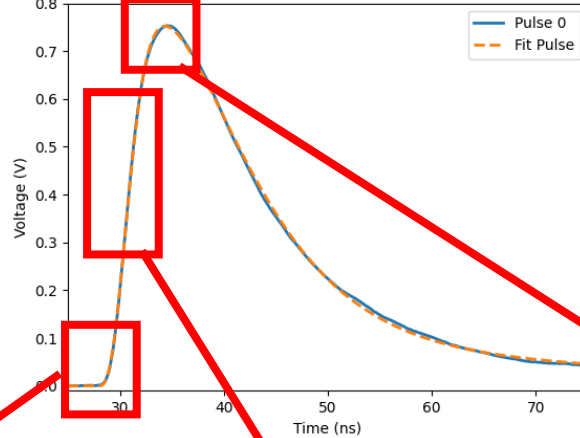
τ_0 : ~rise time

τ_i : photon decay time

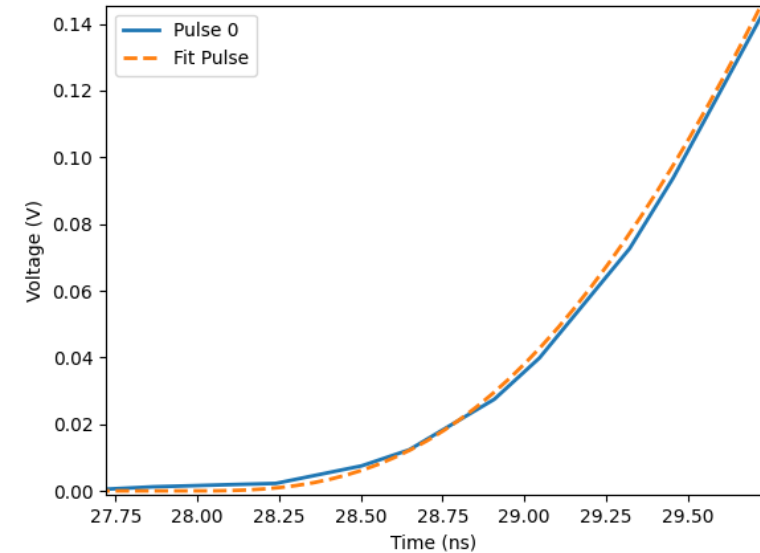
f_i : component fraction

Imperfectness of the template function

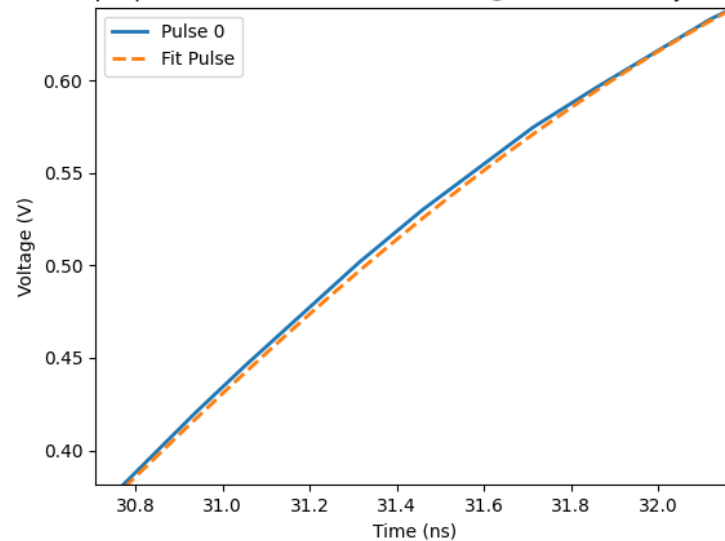
Example pulse 0 and fit result (test beam @ 100 GeV, WPoly, 05/2025)



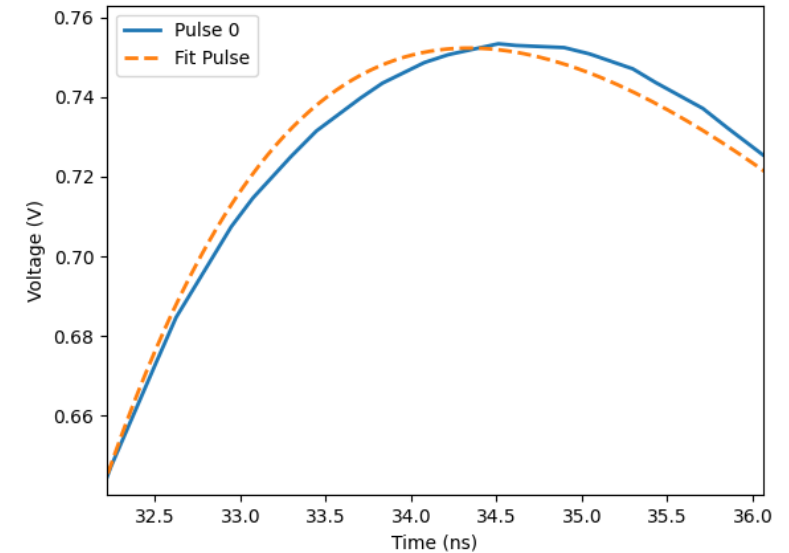
Example pulse 0 and fit result (test beam @ 100 GeV, WPoly, 05/2025)



Example pulse 0 and fit result (test beam @ 100 GeV, WPoly, 05/2025)



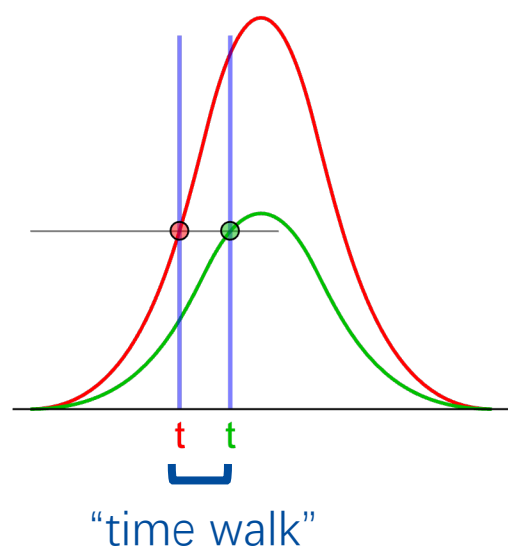
Example pulse 0 and fit result (test beam @ 100 GeV, WPoly, 05/2025)



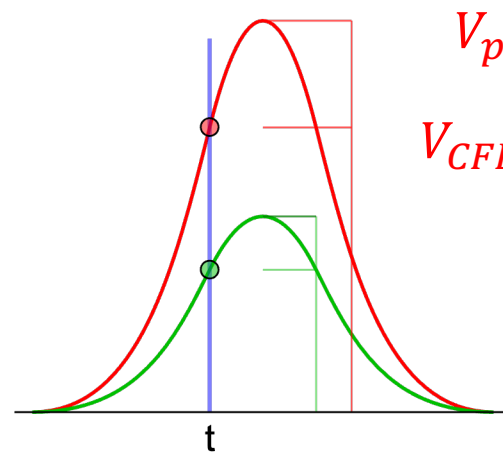
SPIDER design

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

Digital CFD algorithm

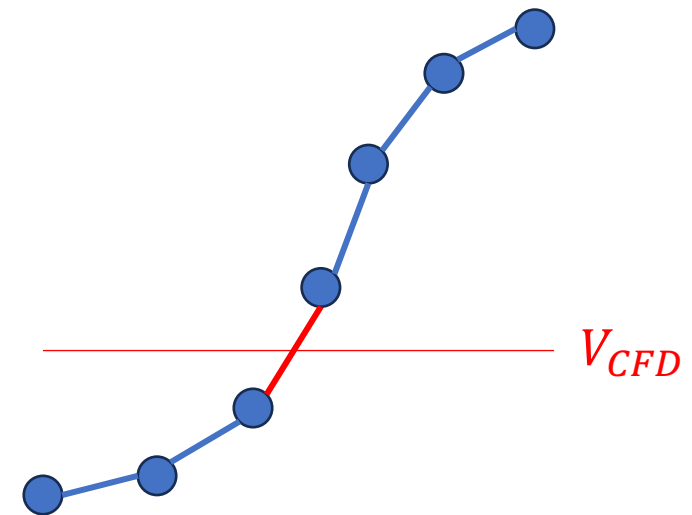


CFD diagram



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

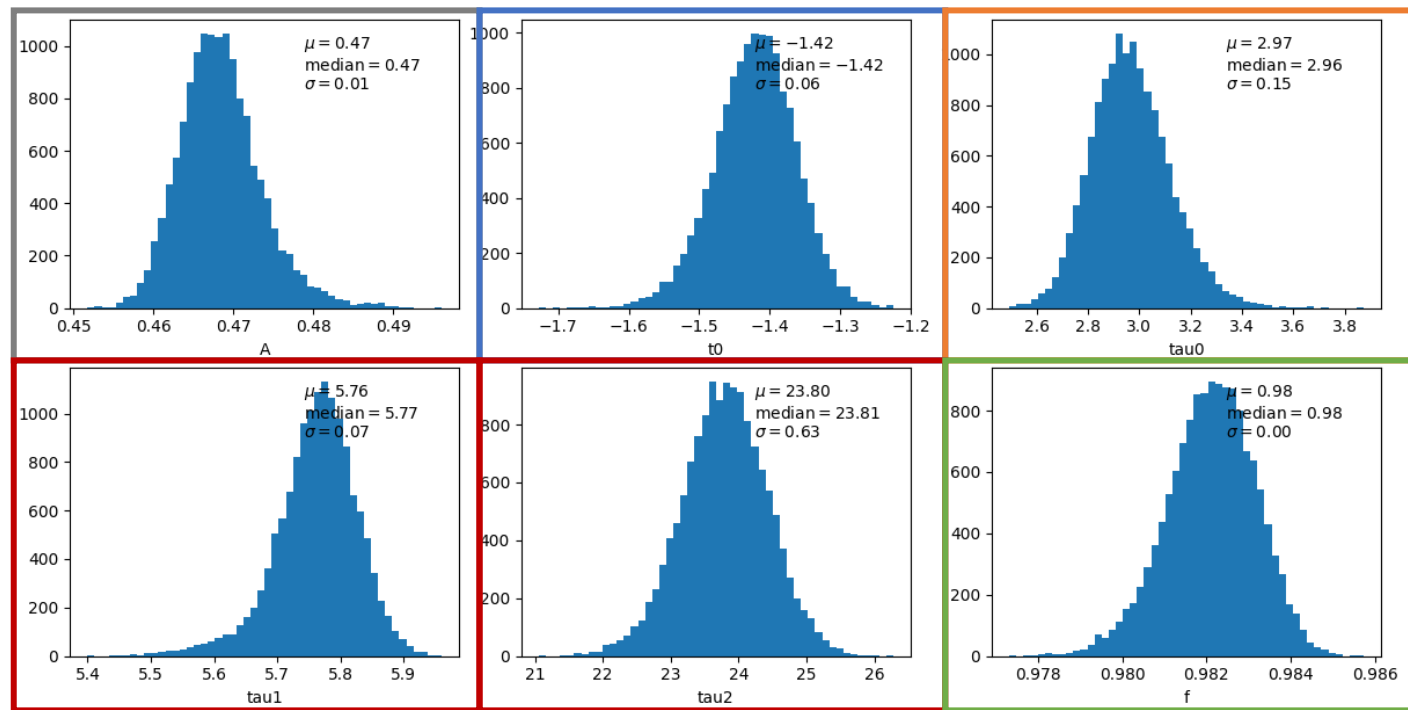
$$(f_{CFD} \in [0, 1])$$



Digital CFD

Method for getting templates

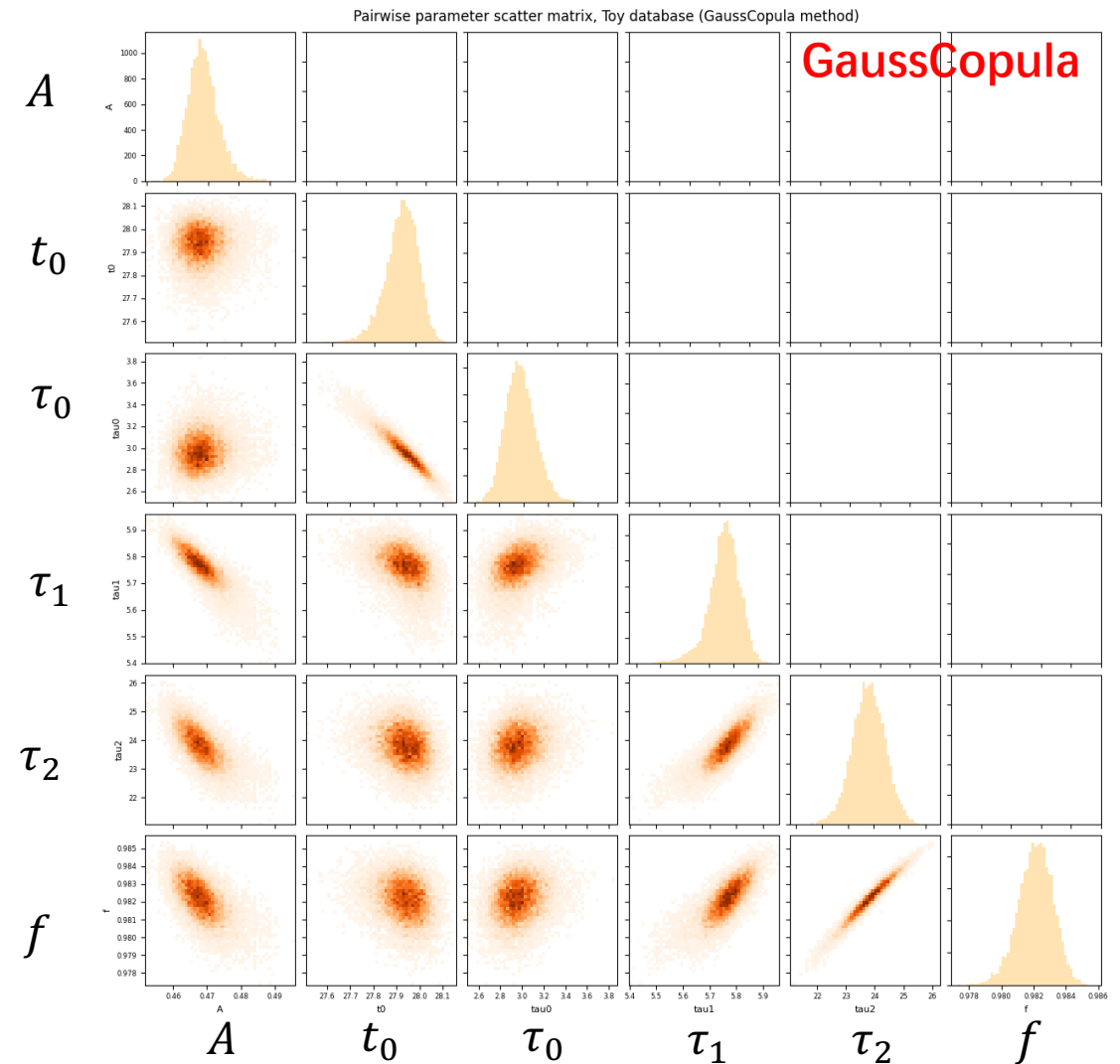
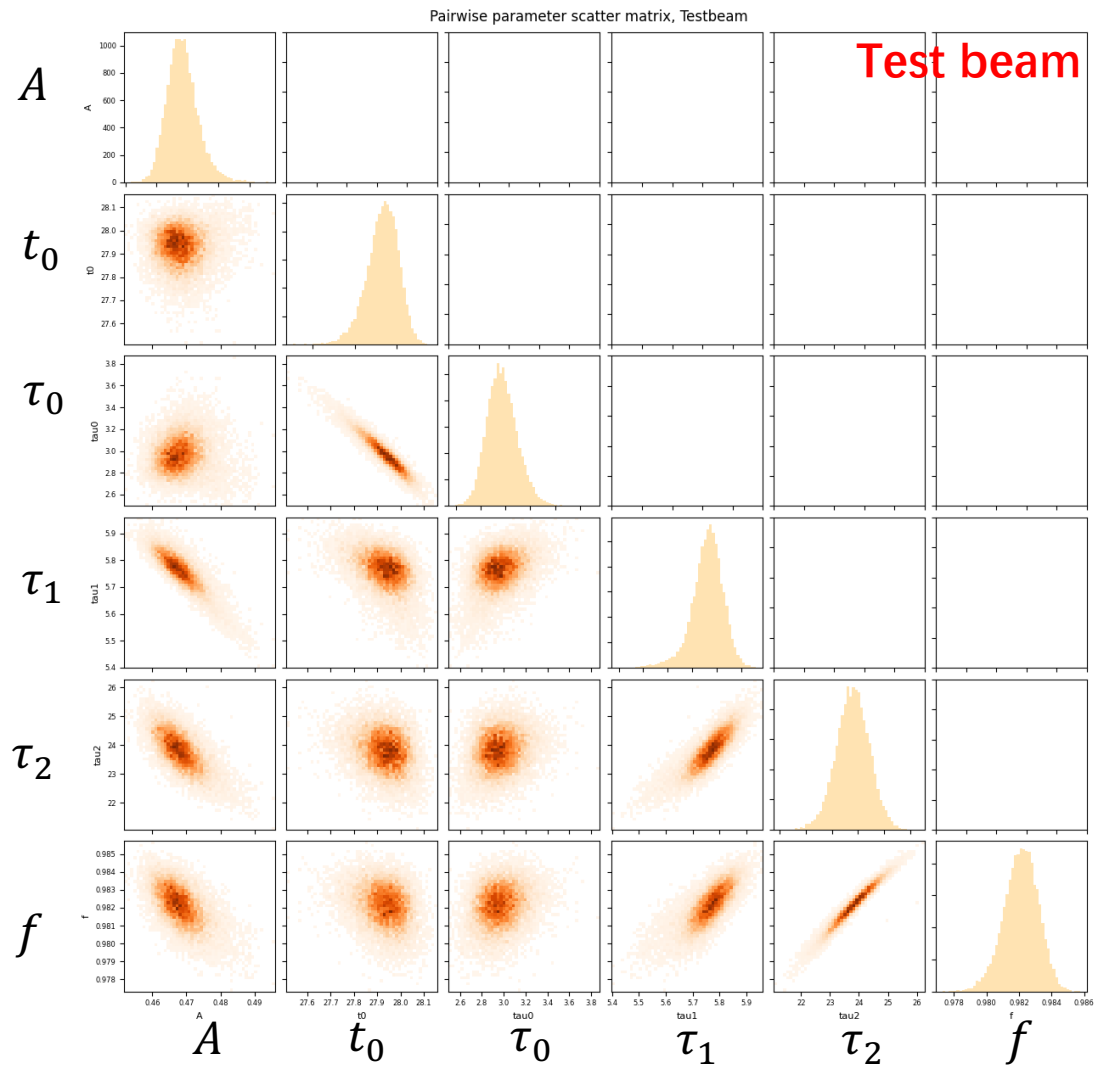
1. Determine the empirical function
2. Fit $\mathcal{O}(10^4)$ test beam pulses, get parameter sets
3. Produce toy parameter sets of the same distribution



GaussCopula method

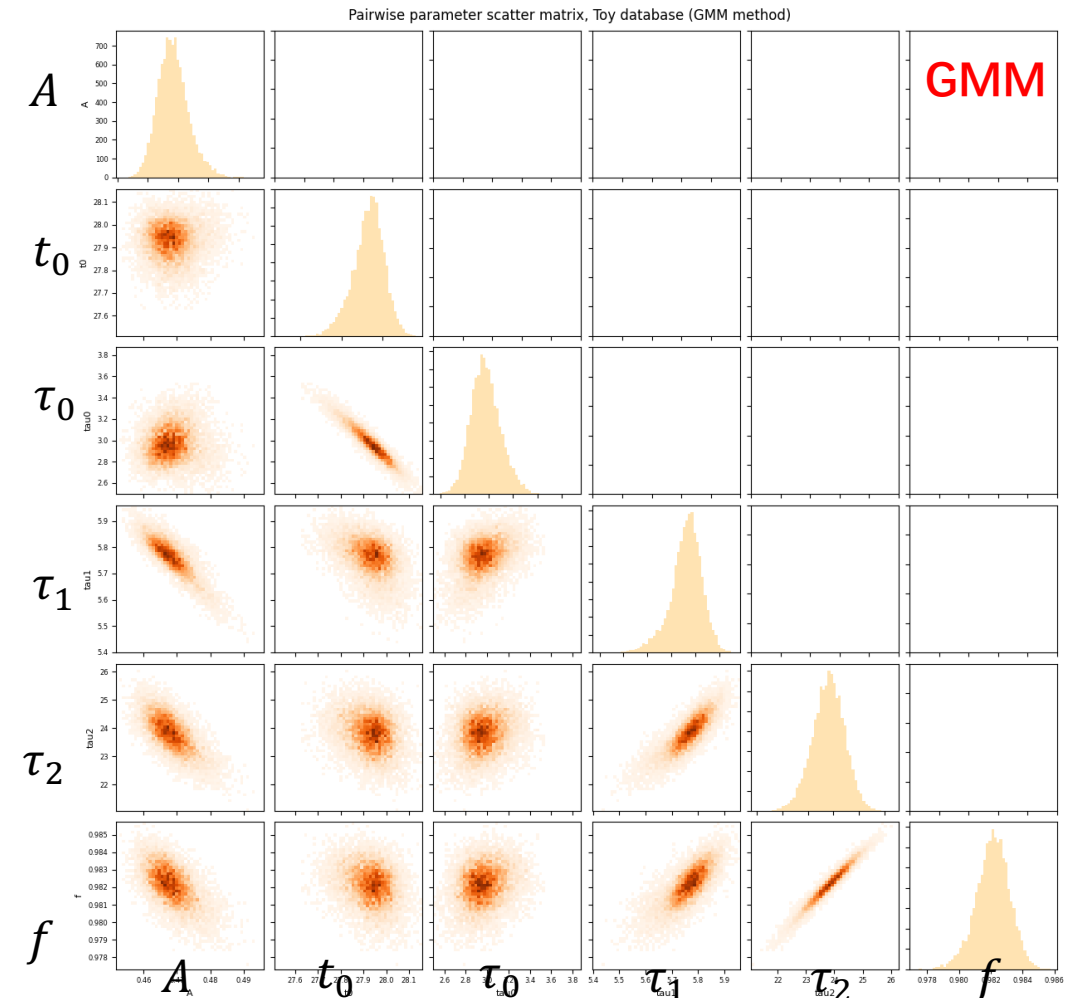
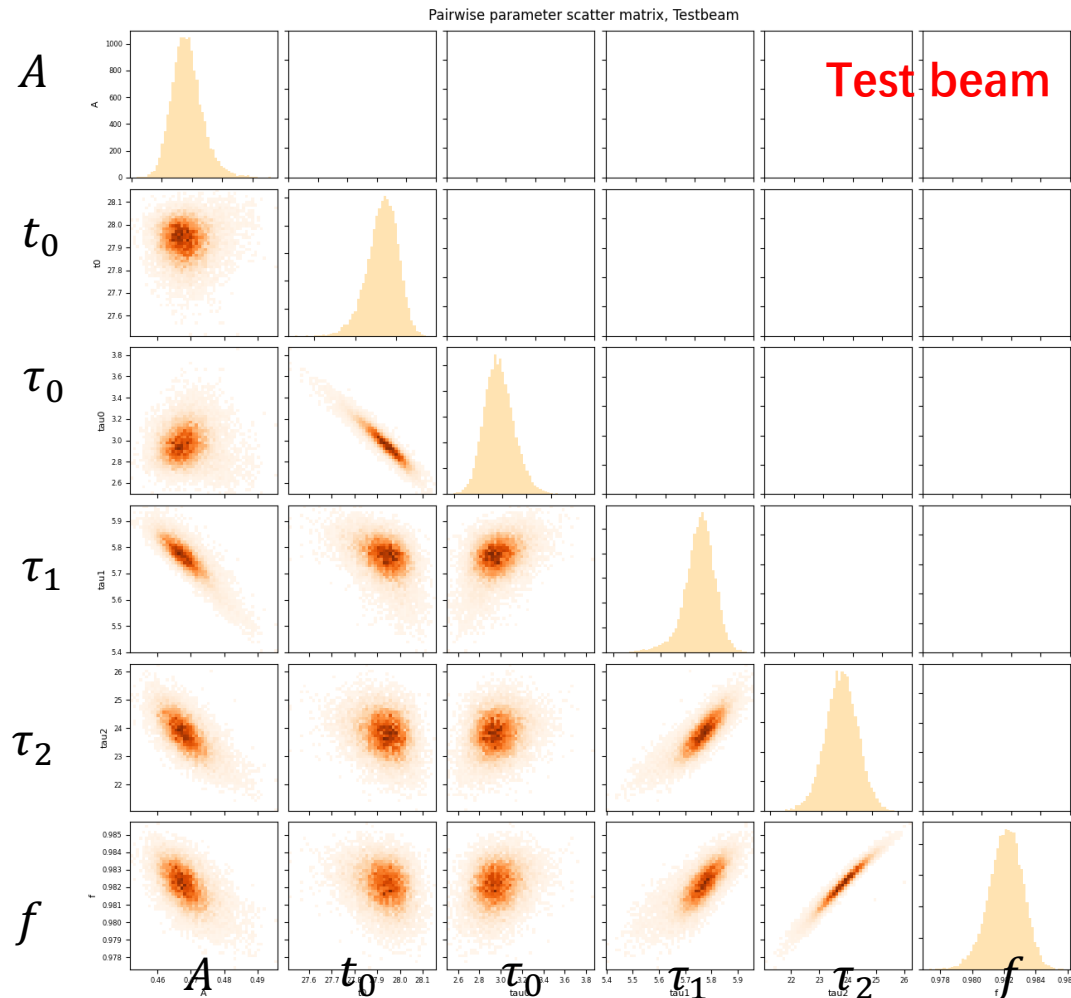
- Conduct a specific transformation T to the parameter sets
 - T regards only the marginal distributions
- Estimate the resulted joint distribution with n-D Gaussian distribution
 - The estimation have same covariant matrix with the joint distribution
- Generate toy parameter sets, do the inverse transformation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

GaussCopula method



Gaussian Mixture Model (GMM) method

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



Fraction of energy in active layers

