27.07.2020

Muon Collider validation software

Overview of the workflow

P. Andreetto [a], N. Bartosik [b], L. Buonincontri [a,d]
M. Casarsa [c], A. Gienelle [a], D. Lucchesi [a,d], L. Sestini [a]

[a] INFN Padova, [b] INFN Torino, [c] INFN Trieste, [d] University of Padova

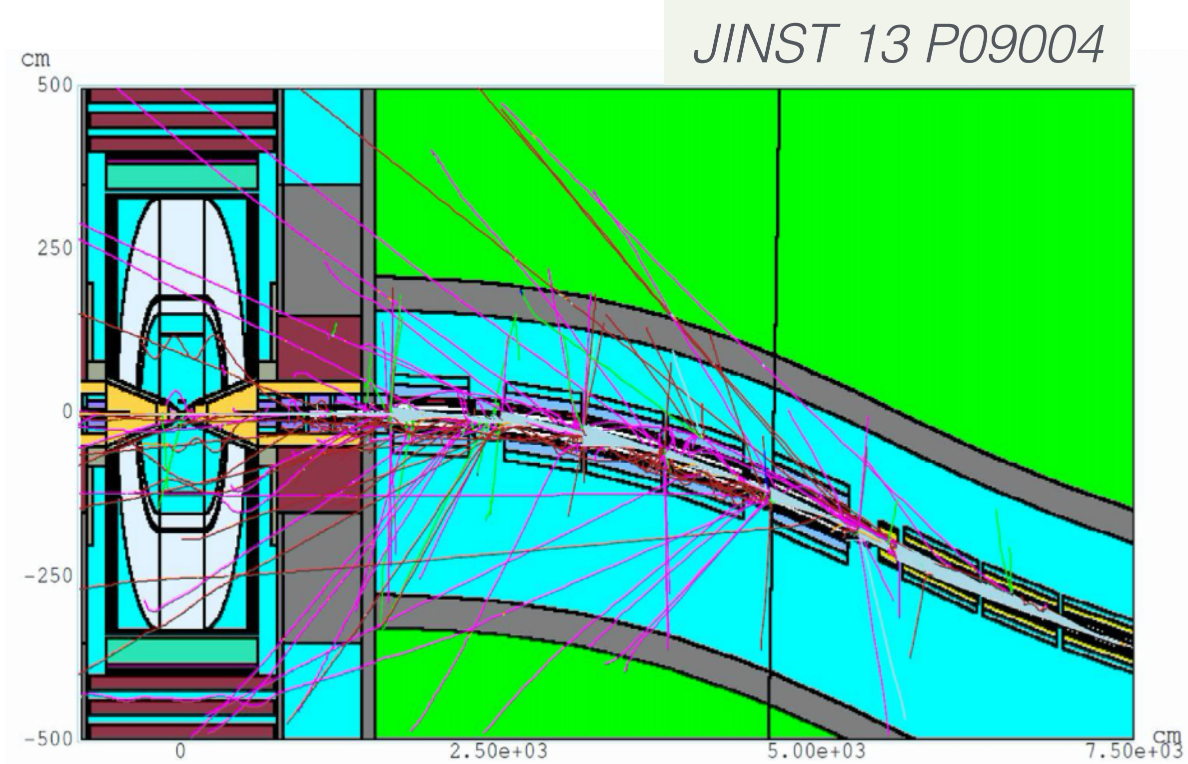
Muon Collider: BIB challenge

Muon Collider is a lepton-lepton collider: colliding elementary particles should provide extremely clean events compared to hadron-hadron colliders *at first sight*

- in reality the situation is not so simple: muons have short lifetime
- with **0.75 TeV** muon beams at **2×10^{12} muons/bunch**
  ↳ **4×10^{5} muon decays/m** in single bunch crossing

MAP (Muon Accelerator Program) developed a realistic simulation of beam-induced backgrounds (BIB) in the detector: implementing a model of the tunnel and accelerator complex ±200 m from the interaction point

- accelerator + detector design are strongly related and have to be studied together



- interaction of secondary electrons with the whole accelerator have been simulated with MARS15 and transported to the detector for further simulation

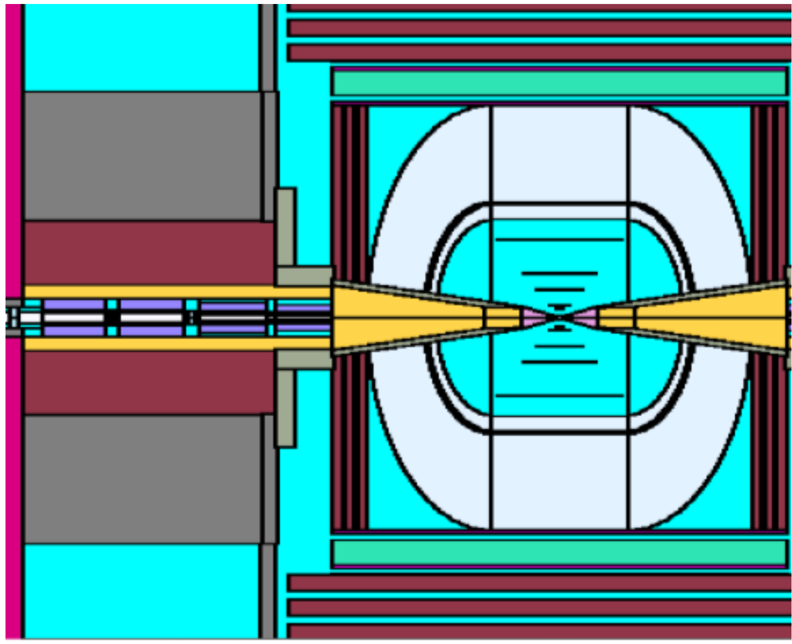
At the level of MARS15 output only the detector and MDI are relevant

Physical BIB supression

Currently we are limited to the beam energies and accelerator complex design that have been studied by MAP: $\sqrt{s}$=1.5 TeV and $\sqrt{s}$=3 TeV

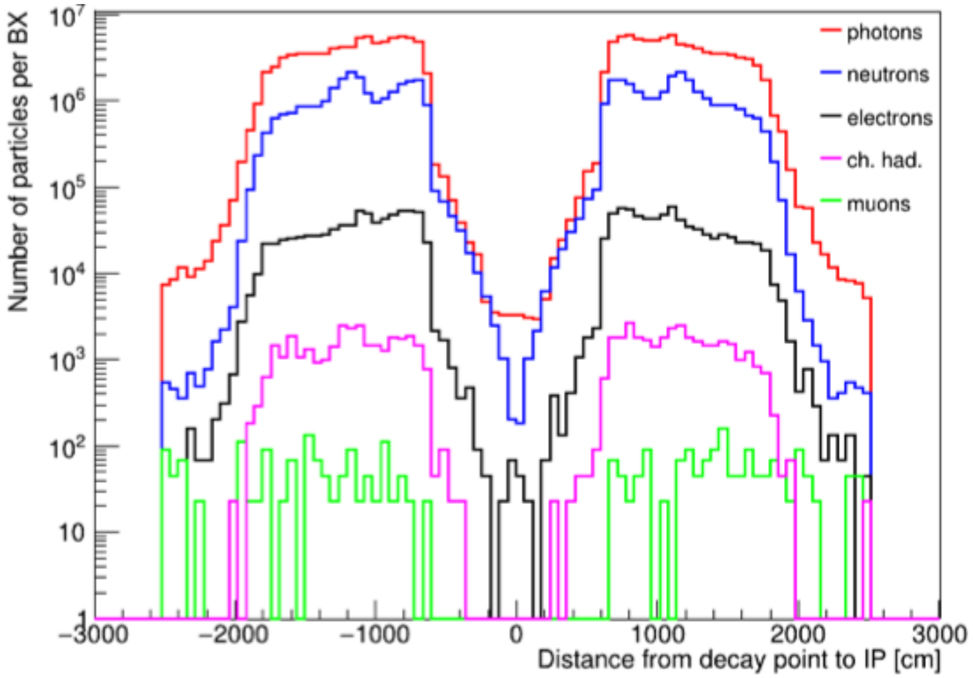
- lists of simulated BIB particles arriving to the detector are available from MAP

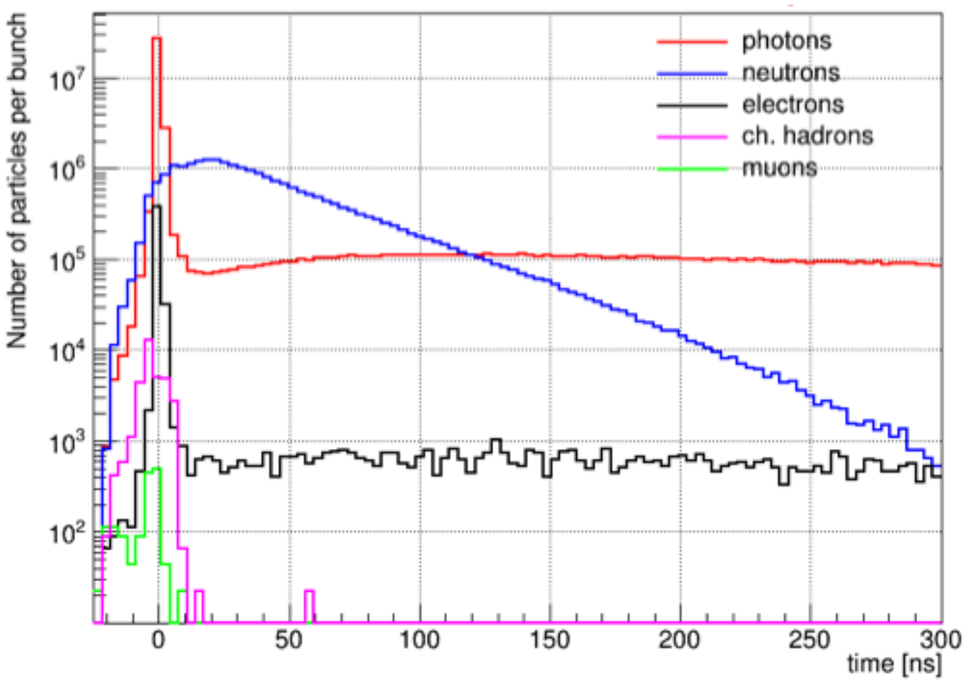
1.5 TeV case has been studied in greatest detail, including the BIB-absorbing tungsten nozzles
with an optimised shape to suppress
the BIB rate by ~500



Huge amounts of particles arriving to the detector

- after suppression by the nozzles





timing will be a crucial discriminator

BIB simulation

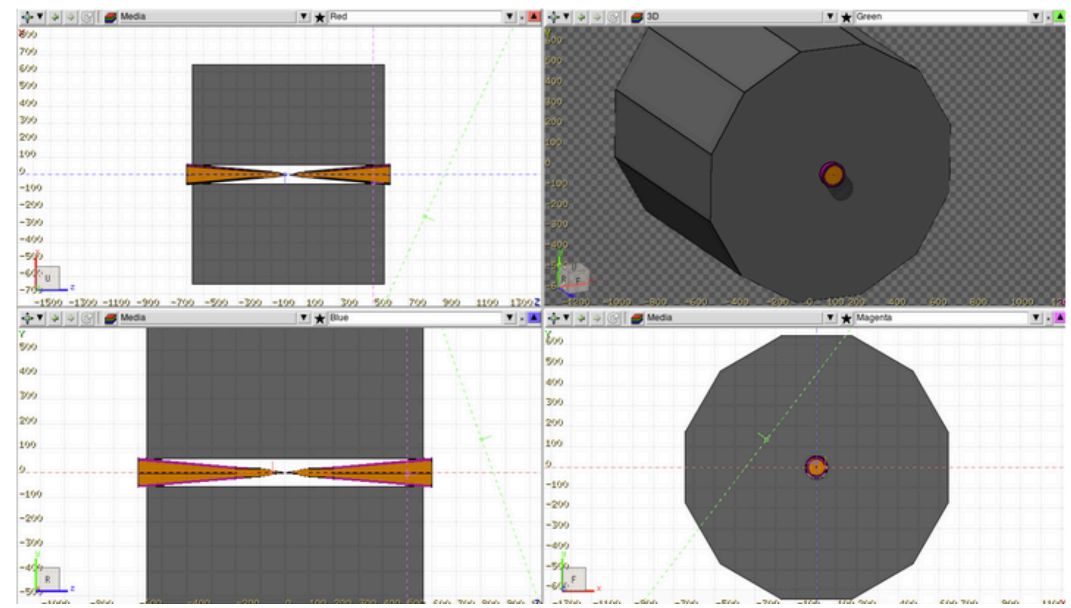
New workflow is under development to simulate the BIB at any tunnel/ accelerator layout using:

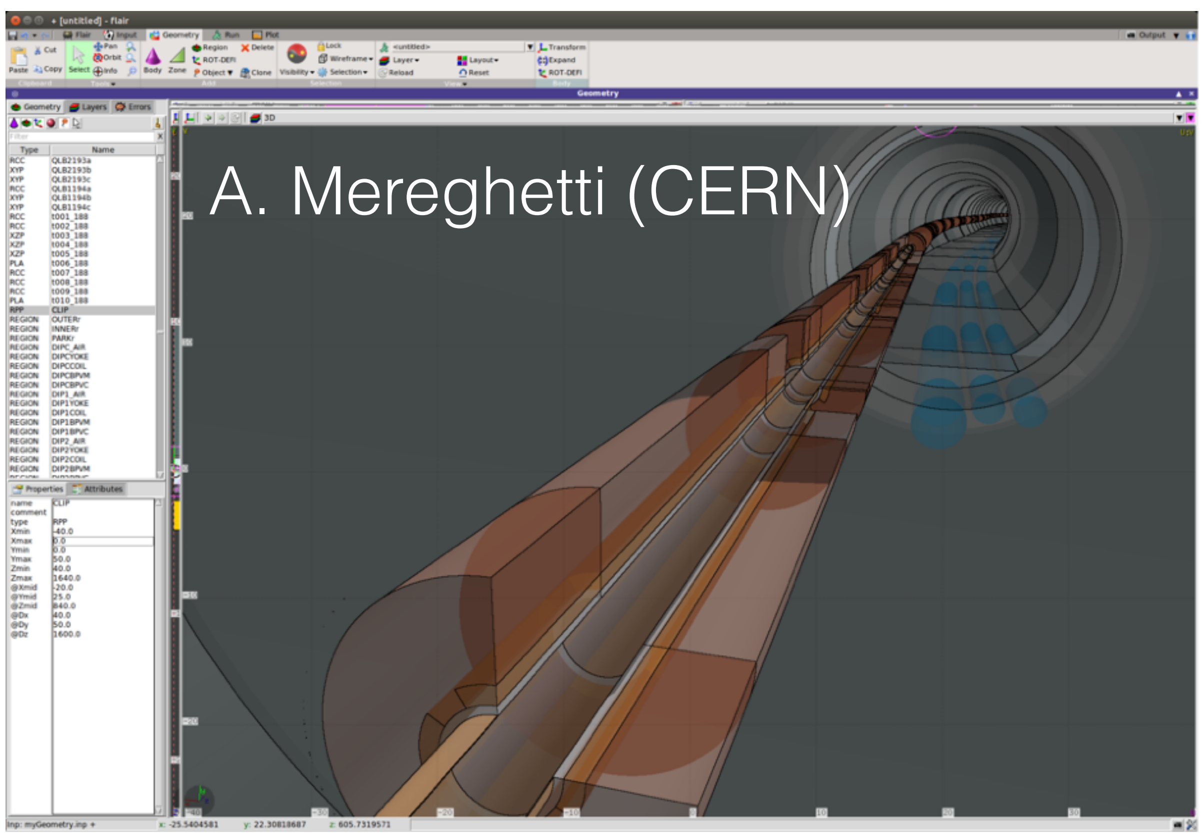
- **FLUKA** for simulating muon decays and interaction with material
- **Fluka Line Builder** for efficient construction of the accelerator complex

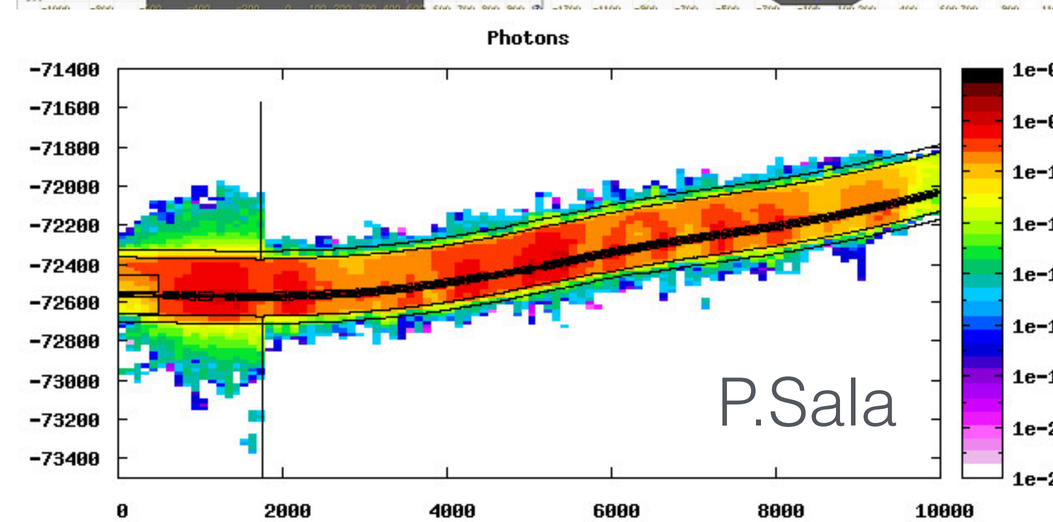
The tools are not yet for public use

The simulated background can be used by anyone interested

F. Collamati, C. Curatolo







Detector performance simulation

The presence of BIB particles has to be taken into account in the detector design

Ultimately the detector performance needs to be evaluated for specific analyses

Key components of a physics analysis:

1. **generation of the process of interest (ME + PS)**
2. **simulation of the detector response to incoming particles**
3. **reconstruction of particles and other relevant quantities**
4. **event classification**

BIB particles directly enter at the very early stage
and affect the rest of the analysis chain

↳ **detector design + readout logics + event reconstruction strategy must all be studied coherently with a fast turn-around cycle**

Currently all the studies are performed within the ILCSoft framework
that is actively used/maintained and was developed for ILC and CLIC studies

A number of modifications and additions specific to the Muon Collider case
are maintained in a separate public Muon Collider Software repository

Key components of ILCSoft

1. LCIO [Linear Collider I/O]
   Provides consistent storage of event data (MCParticles, simulated/reconstructed hits, higher-level and custom objects) using *.slcio file format
   - the most generic and basic part with almost no user intervention needed

2. DD4hep [Detector Description for High Energy Physics]
   Efficient and flexible detector geometry description with interface to GEANT4 and simulation/reconstruction software
   - consists of C++ implementations of detector components assembled together via flexible XML configuration files
   - conceptual changes in the detector design require corresponding extensions of the underlying C++ code

3. Marlin [Modular Analysis & Reconstruction for the LINear collider]
   Collection of processors for isolated tasks that can be chained into the necessary workflow by means of XML configuration files
   - everything after hits simulated by GEANT4 is handled by processors within the Marlin framework: *digitization, hit/track/jet reconstruction, b-tagging, etc.*

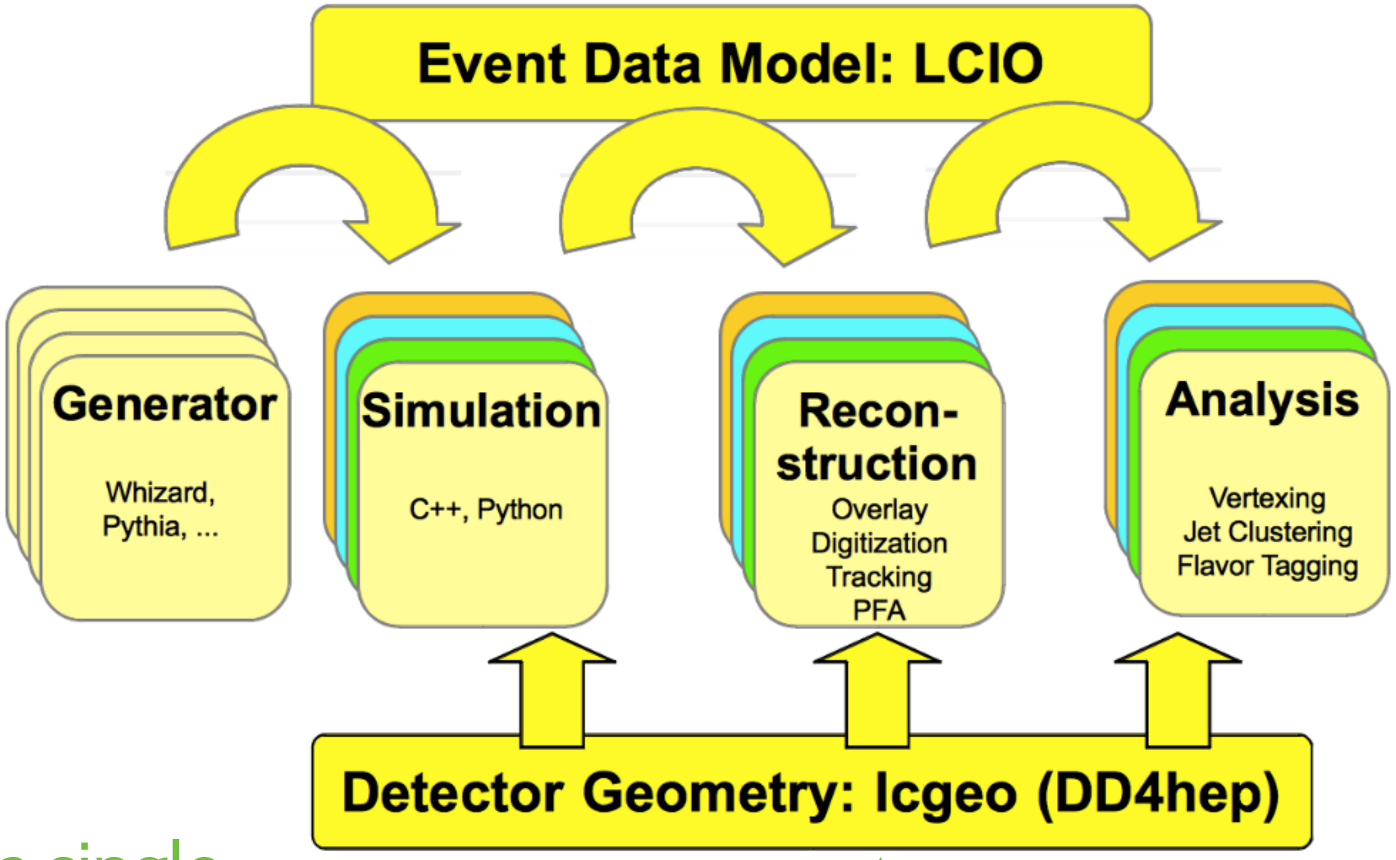
Key components of ILCSoft

In a specific physics analysis the process of interest is generated by a dedicated event generator (outside the ILCSoft framework)

- Simulation step is handled by DD4hep: `ddsim --steeringFile clic_steer.py`
- STDHEP or SLCIO formats for input are supported

BIB particles can be added in either of the two places:

- **Simulation** (as MCParticles)
  - fluctuations in GEANT4 simulation are not significant
- **Reconstruction** (as SimHits)
  - much more efficient



Optimal solution:

- perform Simulation of BIB from a single bunch crossing and overlay hits on every signal event

BIB particles: from MARS15

Provided my MAP as MARS text files: lists of particle definitions *(μ^+ and μ^- beams)*

- each line represents a single particle crossing the outer detector/nozzle surface
- only a fraction of all particles actually included
  - each particle has an associated weight to calculate the proper normalisation

Dedicated C++ macro converts to SLCIO files, compatible with ILCSoft framework

- 1 line → 1 MCParticle with corresponding position, momentum, pdgId, etc.
- + copies of the particle randomly distributed in φ to account for the weight
- particles split into multiple events (default: 2000 lines/event)
  - to use a fraction of all particles in the simulation
  - to run the GEANT4 simulation of subsets of particles in parallel

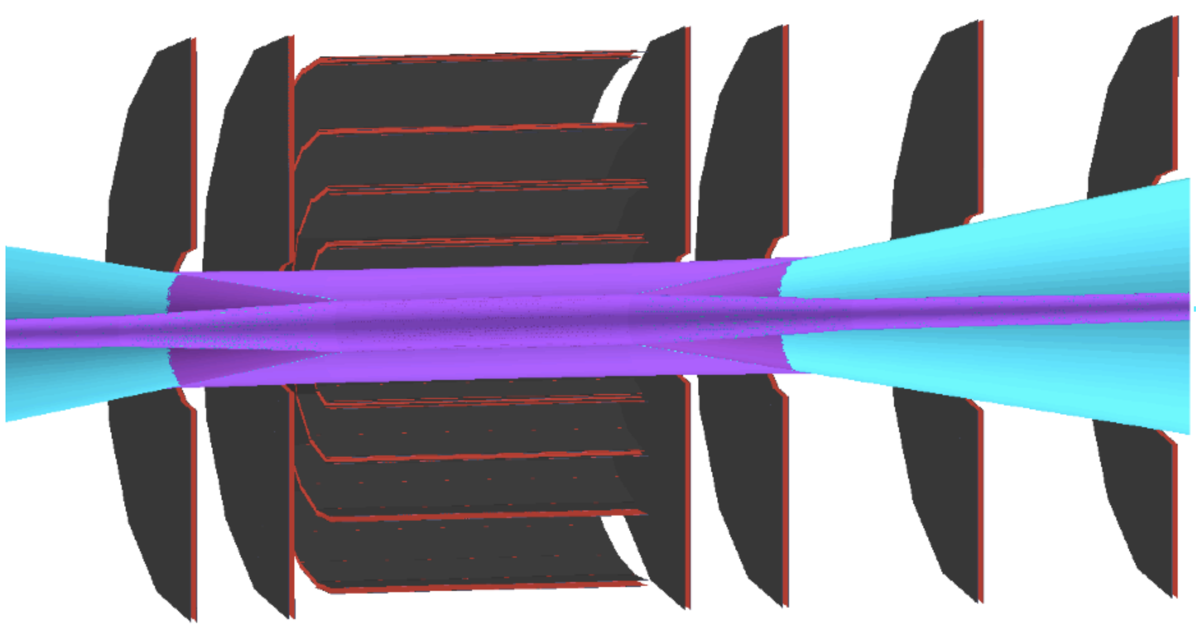
Possible to exclude particles based on certain selection criteria

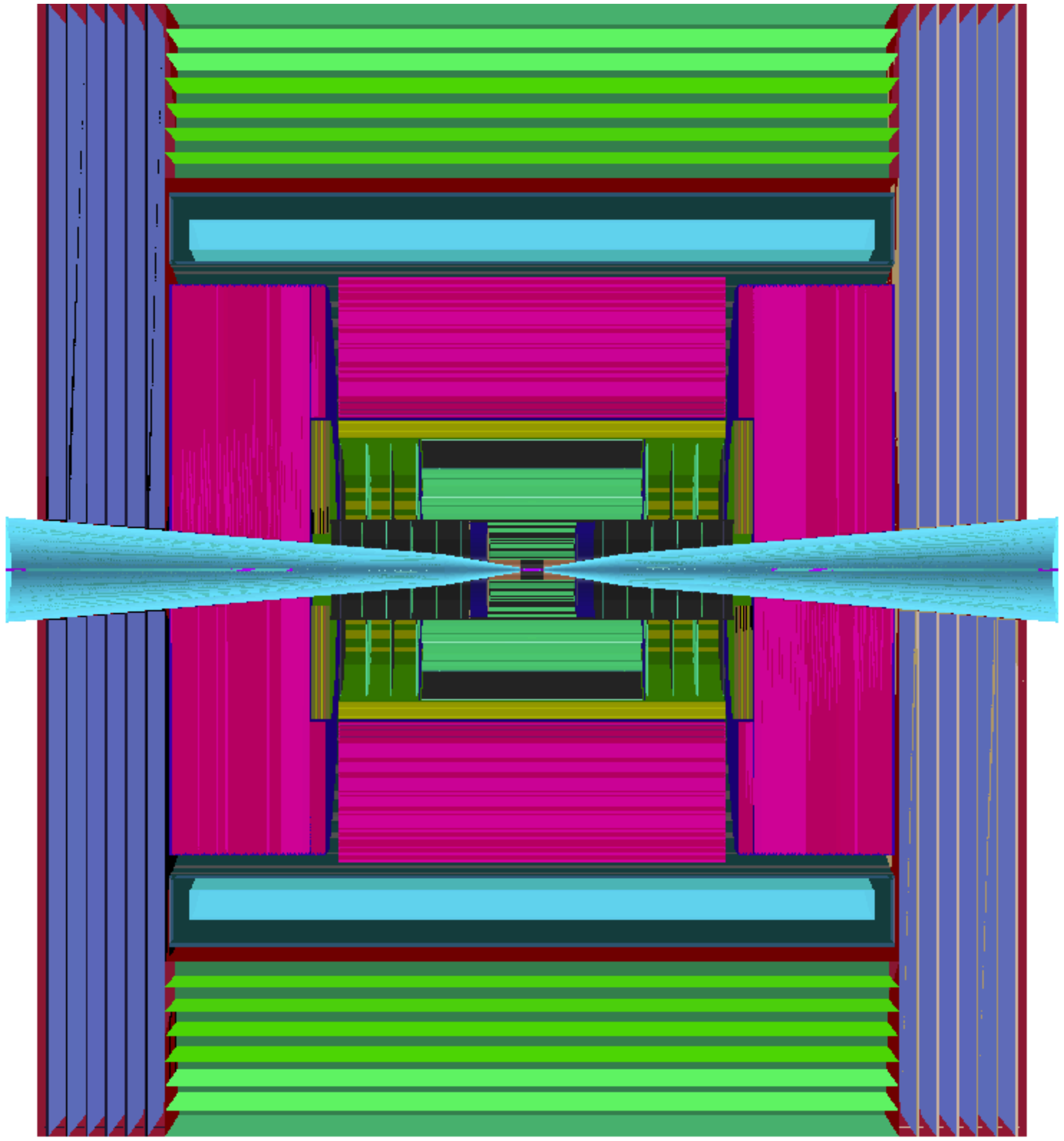
- time of arrival of the particle
- energy of the particle if it's a neutron *(more in the following slides)*

Detector geometry: derived from CLIC

Currently used geometry is derived from the CLIC detector:

- inserted BIB-absorbing tungsten nozzles developed by MAP
- inner openings of endcap detectors increased to fit the nozzles
- optimised layout of the Vertex detector to reduce occupancy
- Vertex segmentation along the beamline





Using the forked version of lcgeo to support the modified geometry components

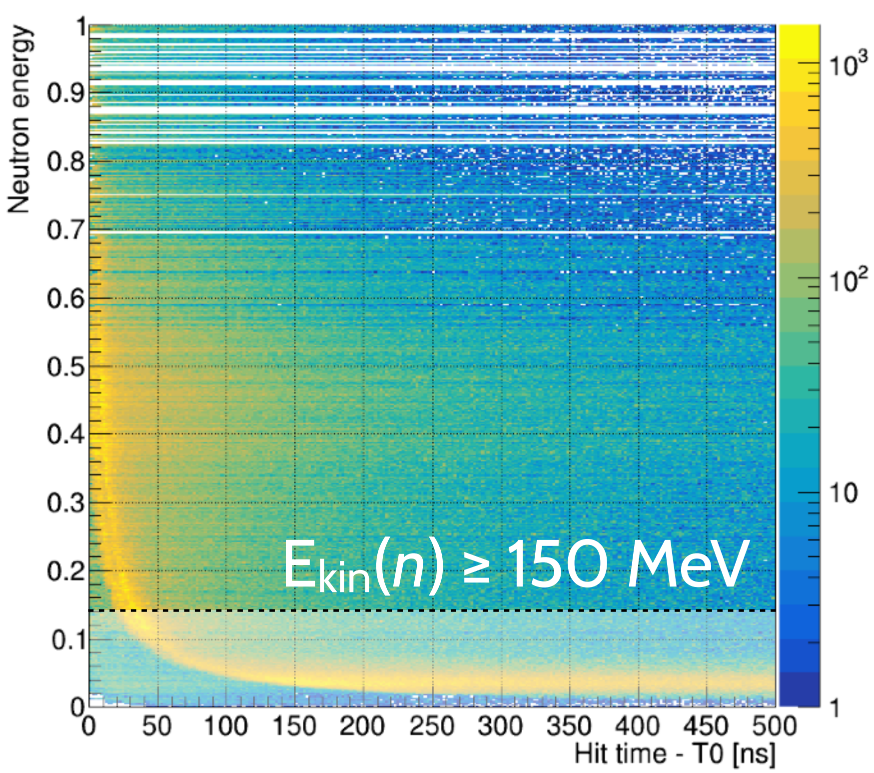
Optimising the background simulation

We start with 190M particles from each beam *[μ⁺ and μ⁻]*

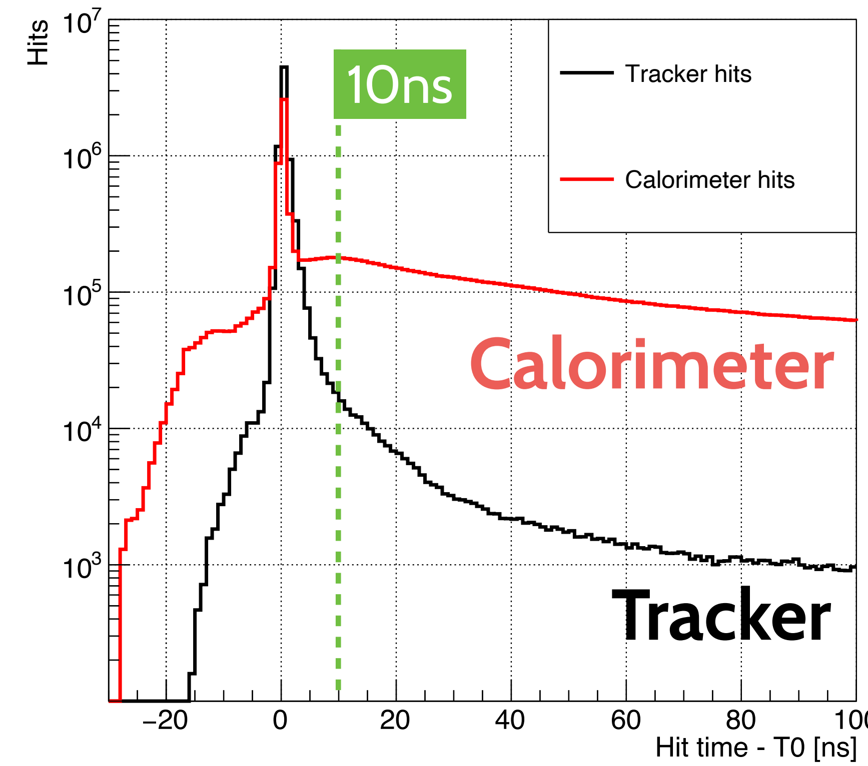
- simulating 1 bunch crossing takes 8 days *[at 8 parallel threads]*

Only hits in the short readout time window are relevant [~0.1-10 ns]

Slow neutrons create calorimeter hits at very large time offsets



	# of particles	*CPU time*
all	**380M**	**380 h**
time cut	98M (26%)	60 h (18%)
$E_{kin}(n)$ cut	**78M (20%)**	**25 h (6.6%)**



Significant speed up of the BIB simulation by excluding irrelevant particles at the earliest stage: MARS15 → SLCIO conversion

BIB overlay

Adding SimHits from extra SLCIO files performed by the Overlay module forked

- more flexible definition of acceptance **time window** for the considered SimHits
- possibility to disable merging of the MCParticles collections
  - too many BIB particles → output file size explodes
  - SimHits↔MCParticle association is not relevant

SimHits come in two conceptually different types:

SimHits:	SimTrackerHits	SimCalorimeterHits
digitization	↓	↓
Hits:	TrackerHits	CalorimeterHits

TrackerHits:

- **1 SimHit** → **1 Hit**
  ↳ no pixelisation for now
- smeared U-V coordinates
- smeared time

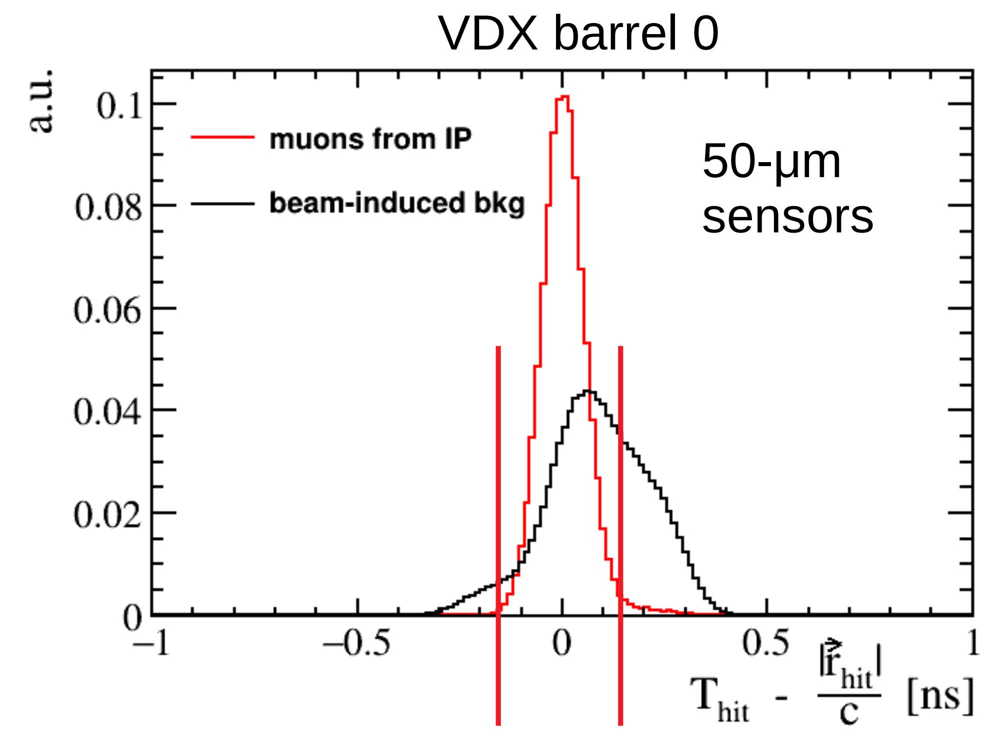
under development: realistic digitization with spatial and time granularity

CalorimeterHits:

- **1 Cell** × **Δt** → **1 Hit**
  ↳ $\sum_{edep}$ in Δt - resolution
- either keep only first Δt or all Δt clusters

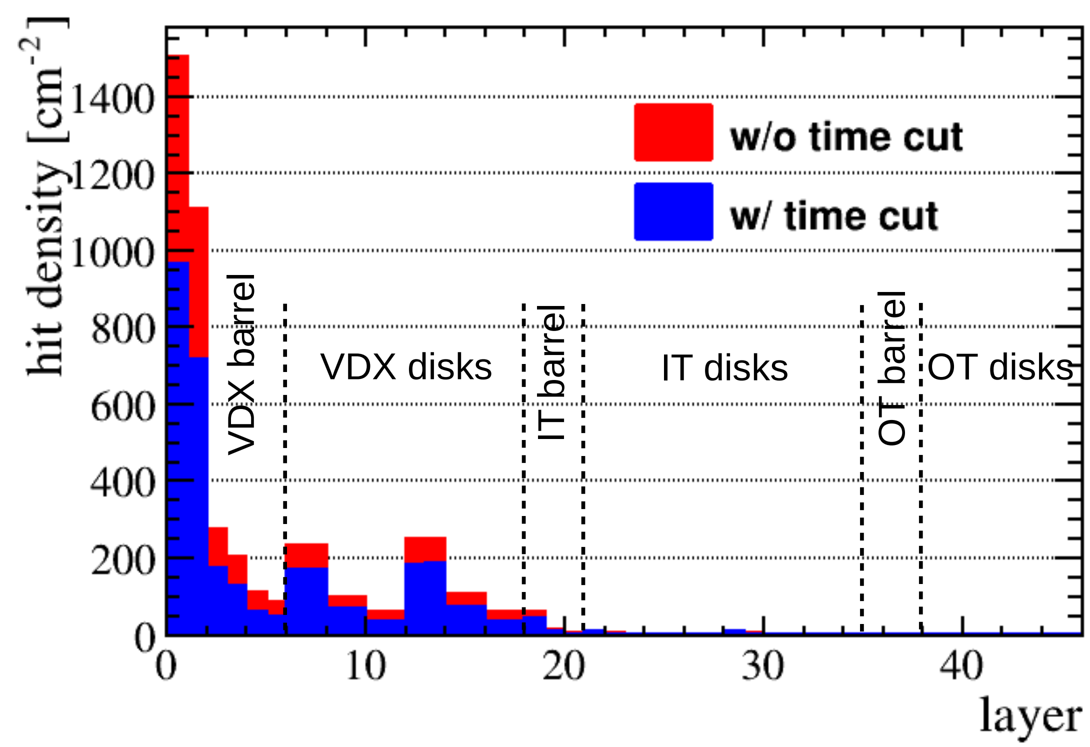
Number of tracker hits from BIB is high but manageable after dropping hits outside the readout time window

- BIB particles arrive to the tracker at different time compared to particles from the IP



The biggest issue: <u>reconstructing tracks with so many hits using conformal tracking algorithm</u>

- developed by CLIC, geometry-agnostic, minimal intrusion when changing geometry
- pattern recognition extremely slow at present occupancy



Two complementary solutions under study:

- further reducing occupancy *(better time resolution, pair-wise hit readout, etc.)*
- changing track-reconstruction workflow *(seeding from less-occupied layers, splitting in sectors, use ROIs, etc.)*

Technical issues: Calorimeters

ECAL and HCAL have high spatial granularity + fine segmentation along R

Photons (ECAL) and Neutrons (HCAL) from BIB produce ~10^8 hits

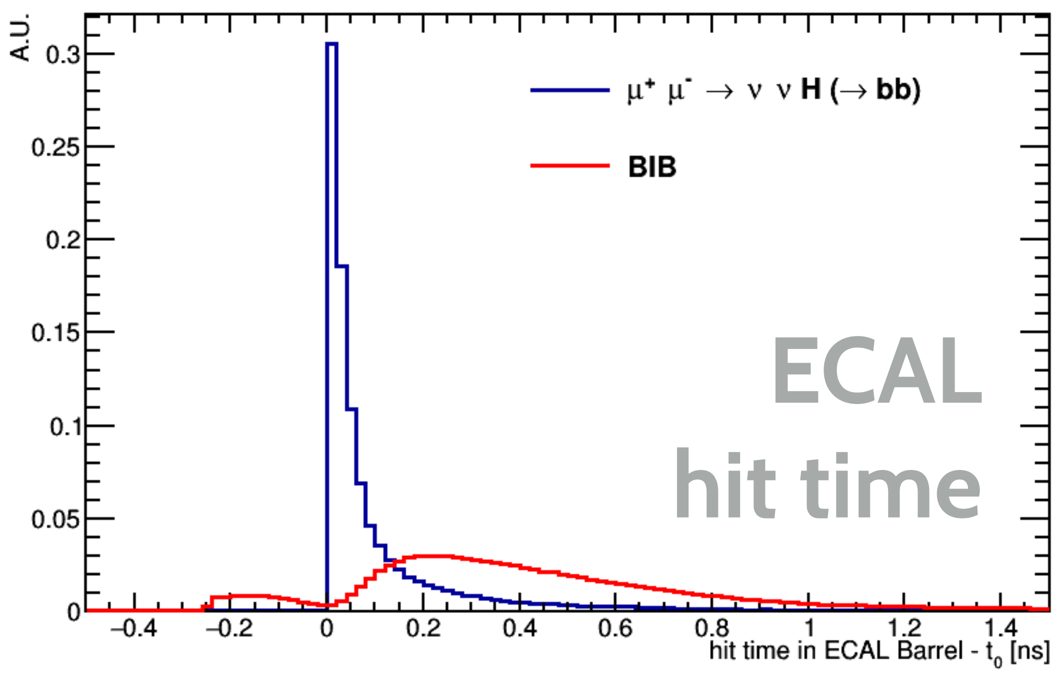
- after removing slow neutrons at the MARS → LSCIO conversion stage
- after applying readout time window cuts

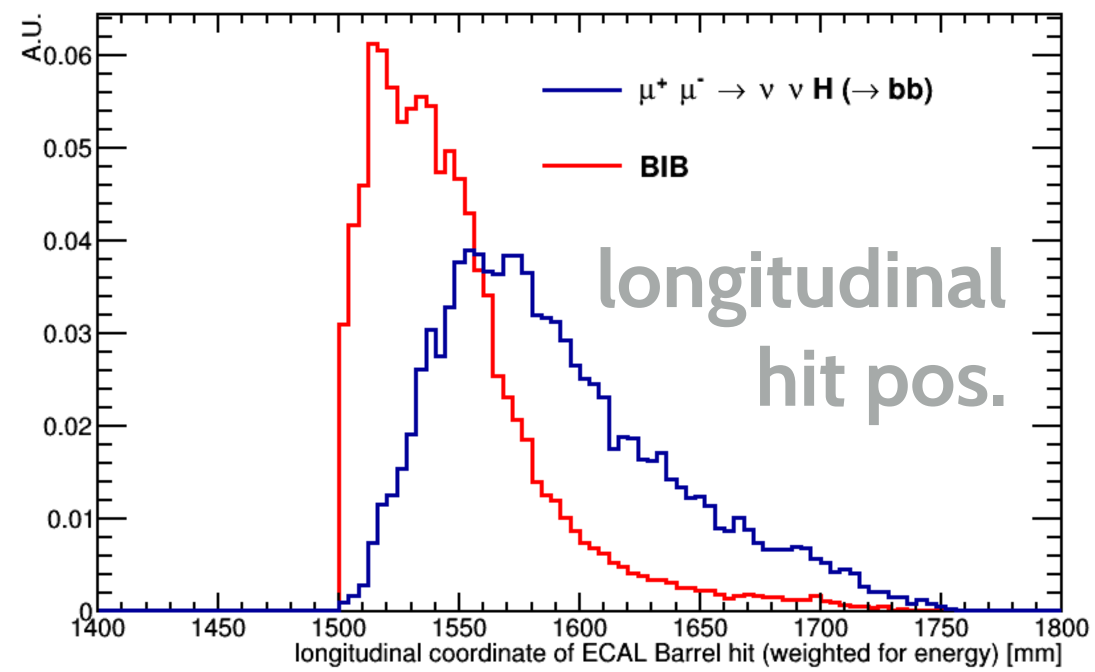
Storing all hits to SLCIO file for further use in event reconstruction fails

- presumably a limitation in LCIO, which was designed for events with significantly smaller numbers of objects
- the corresponding limitation (or bug?) to be fixed in the near future
- currently working with more isolated regions of the detector with smaller numbers of hits

Calorimeter optimisation

Timing and longitudinal shower distribution provide a handle on BIB in ECAL





Various BIB mitigation approaches for ECAL can be studied

- possibly adding a preshower for absorbing the initial part of BIB in ECAL
- subtraction of BIB depositions using the hit time+depth information

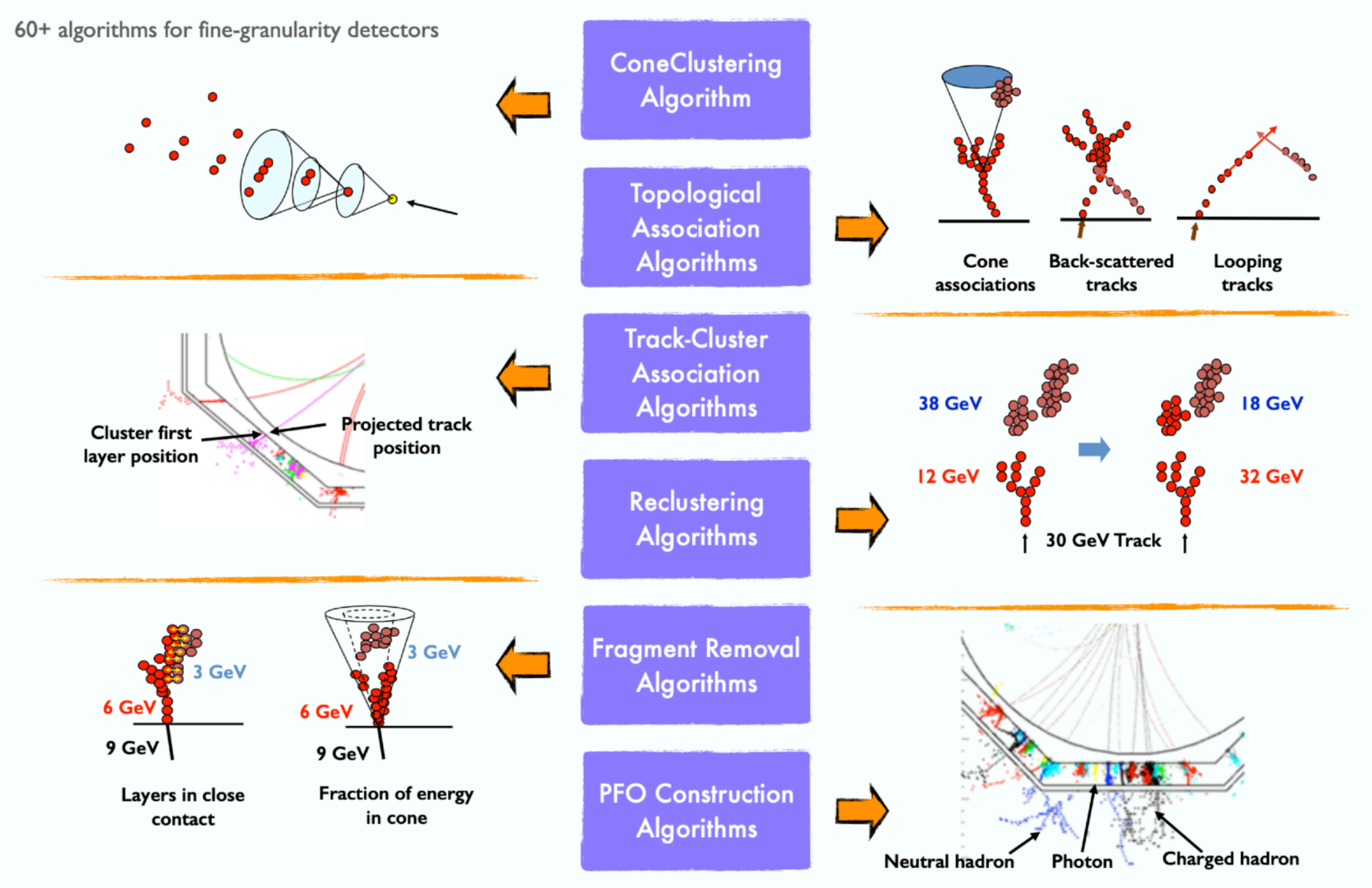
Hadronic showers have longer development time → timing not critical

- the most straightforward approach: evaluate the average BIB energy deposition and consider only energy deposits above the BIB level

Higher-level objects

CLIC uses PandoraPFA for Particle Flow reconstruction within Marlin

- relies on two main building blocks: tracks + high-granularity calorimeter data



Can't be used yet in the presence of BIB

Summary

We are performing our studies on a solid foundation of ILCSoft that was proven to be an effective tool by ILC and CLIC

Beam-induced-background at Muon Collider poses a number of challenges for the framework in its current state

- coping with the very high number of particles/hits and suppressing it as much as possible
- efficiently reconstructing tracks
- realistic treatment of time measurements in the simulation

Plenty of specific directions available for anyone interested to contribute

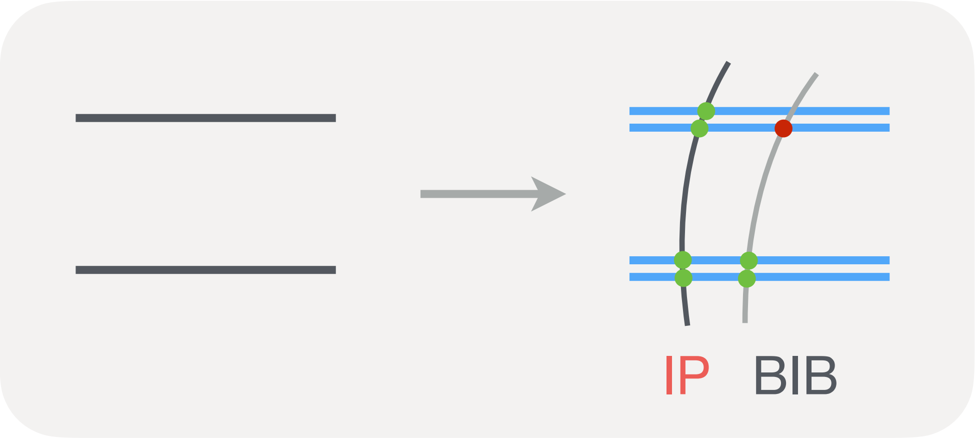
- development of the realistic tracking detector digitization
- study of optimized track reconstruction approaches
- study of alternative calorimeter designs
- jet reconstruction studies
- etc.

BACKUP

Double-layer hit rejection

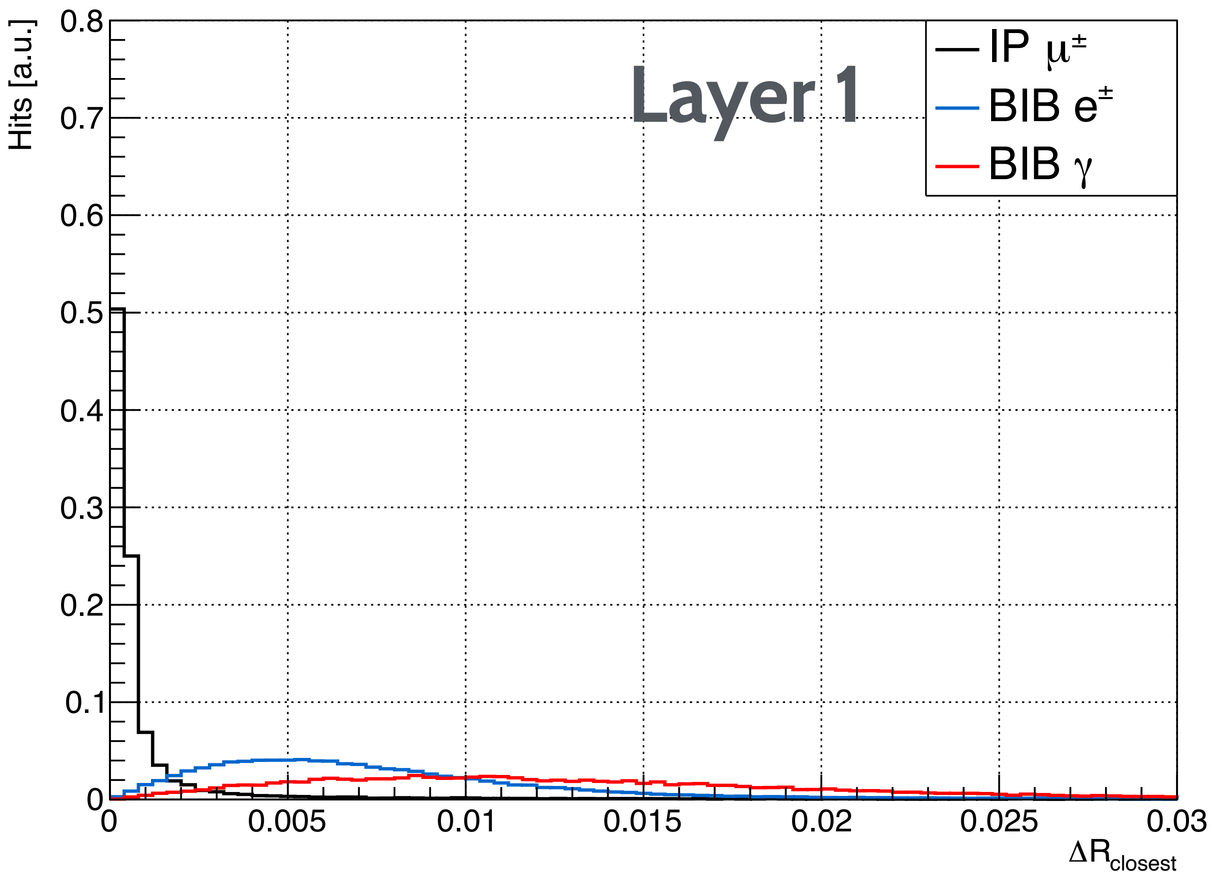
Double-layer structure provides an additional handle on BIB rejection

- soft BIB tracks have lower probability of reaching the second sublayer
- soft BIB tracks are deflected more by the magnetic field



Readout only pairs of hits that are close in η-φ across two sublayers

- sensors at the same angular position from the two sublayers should be paired for natural hit pair selection
- having better time synchronisation between pairs of sensors could allow tighter time window at this step

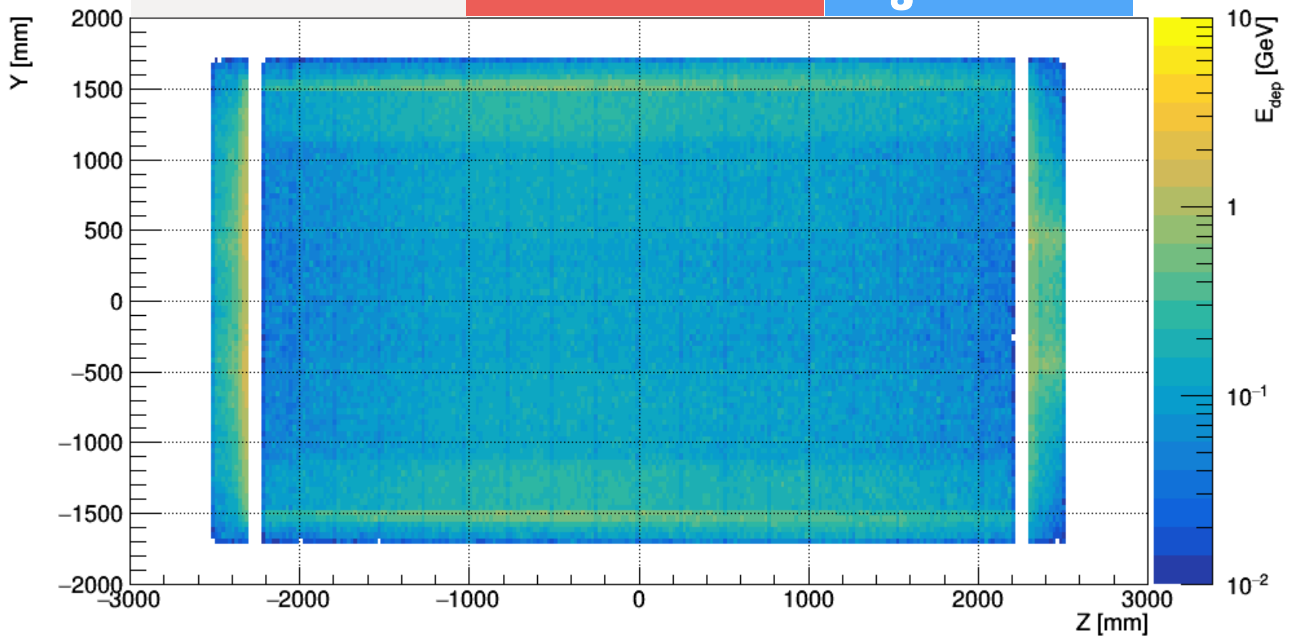


Calorimeter is almost uniformly lit by the BIB particles

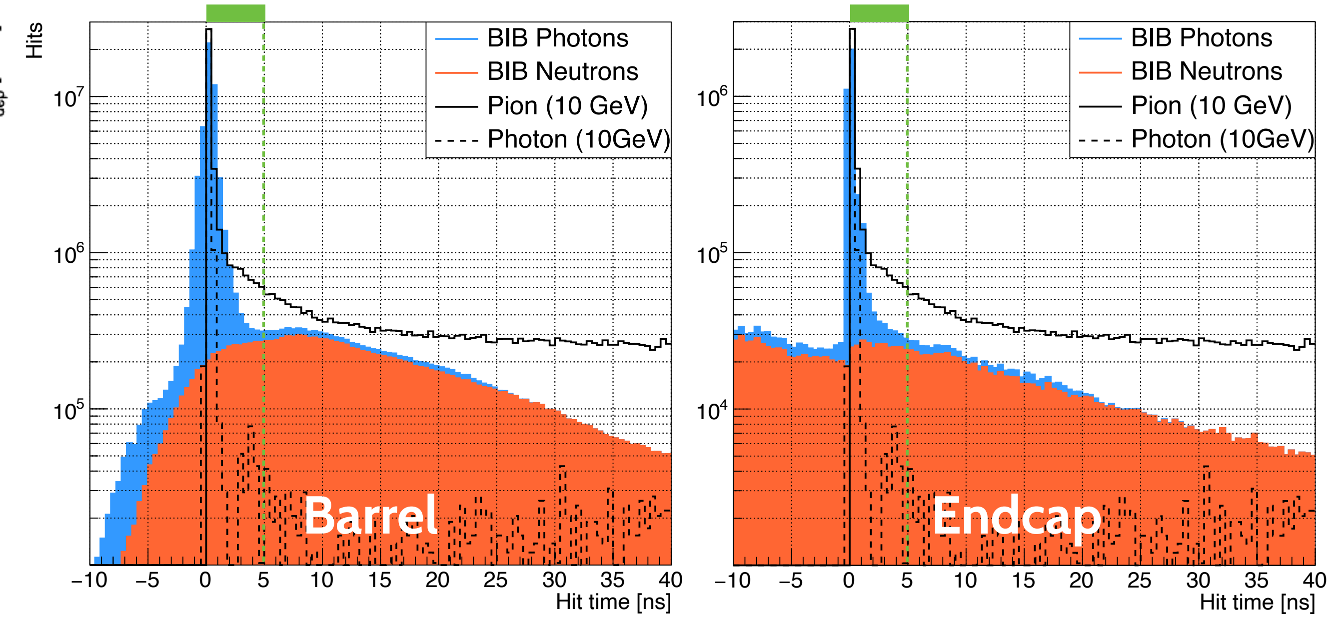
μ^- beam

ECAL

n: 48% γ: 51%

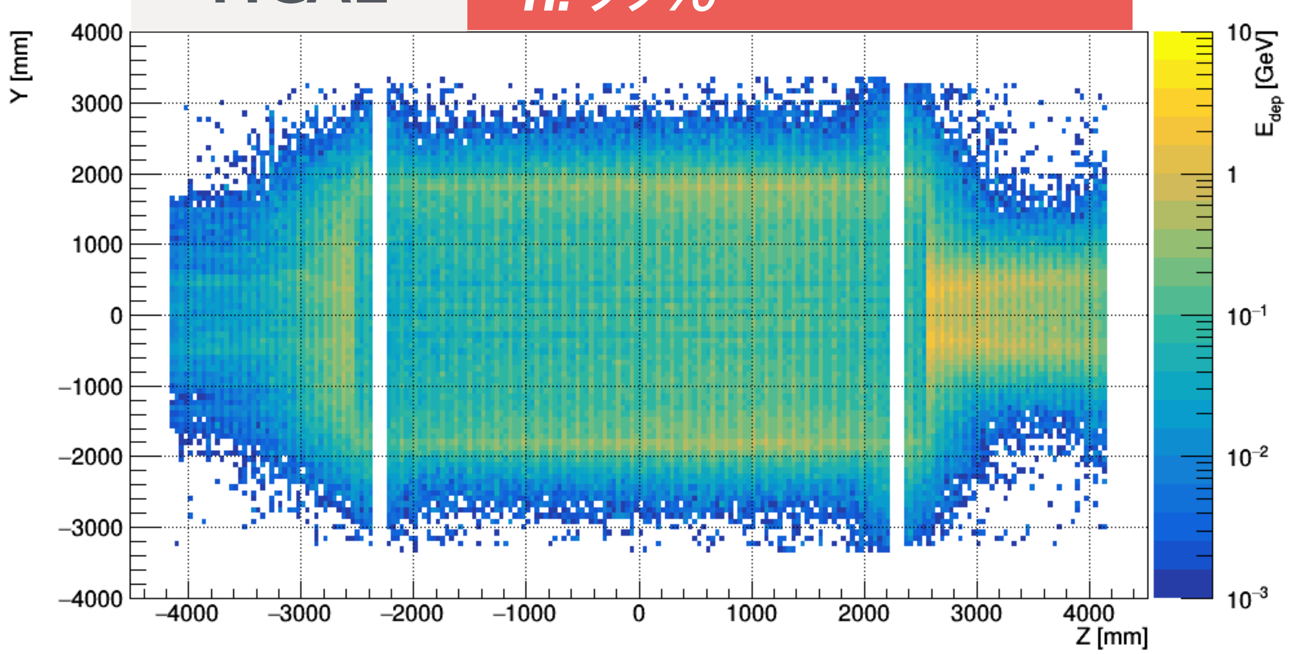


5ns

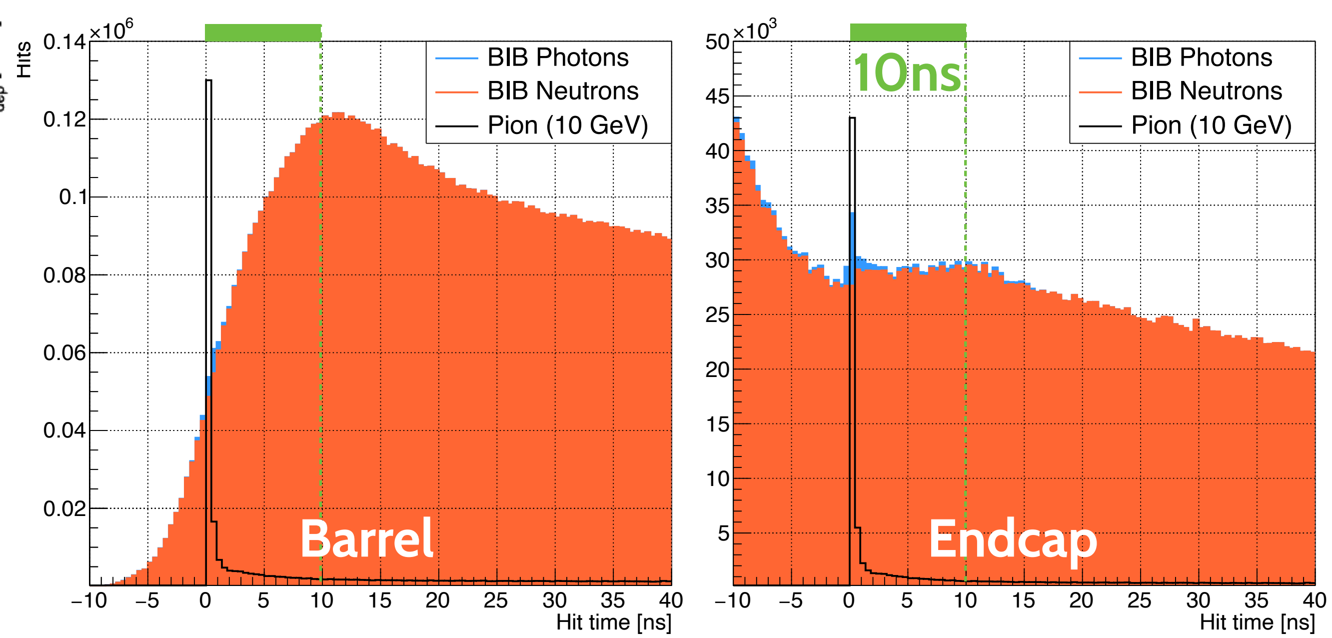


HCAL

n: 99%



A short readout time window needed

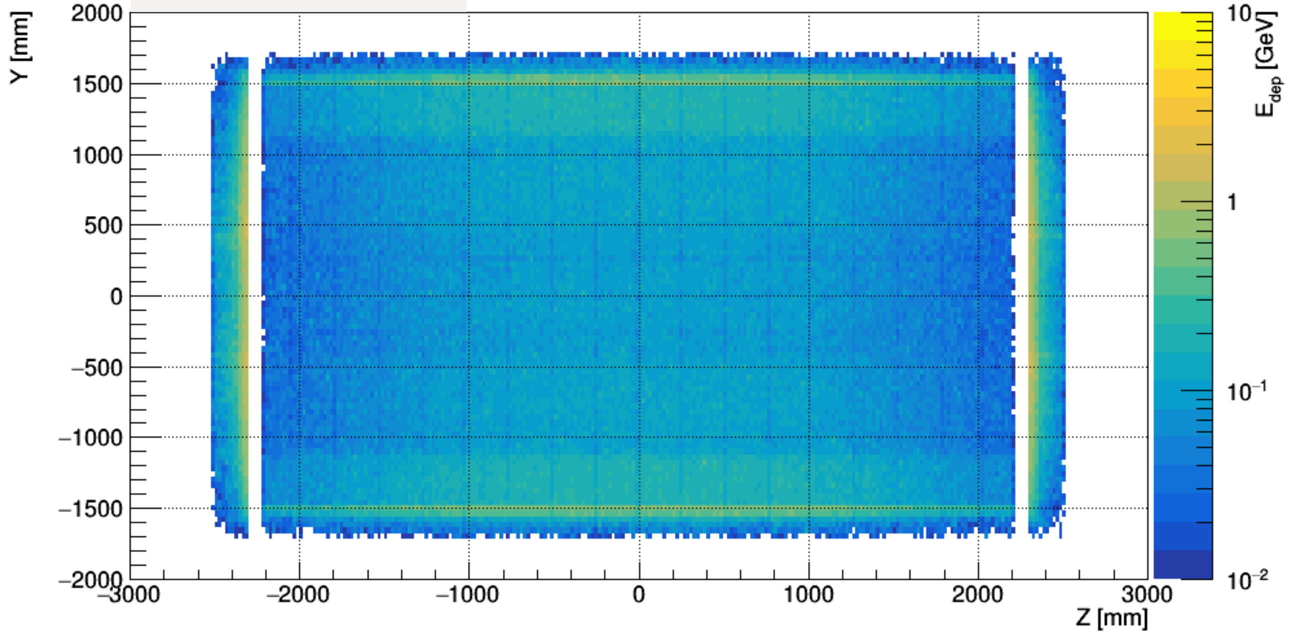


Calorimeter hit distribution

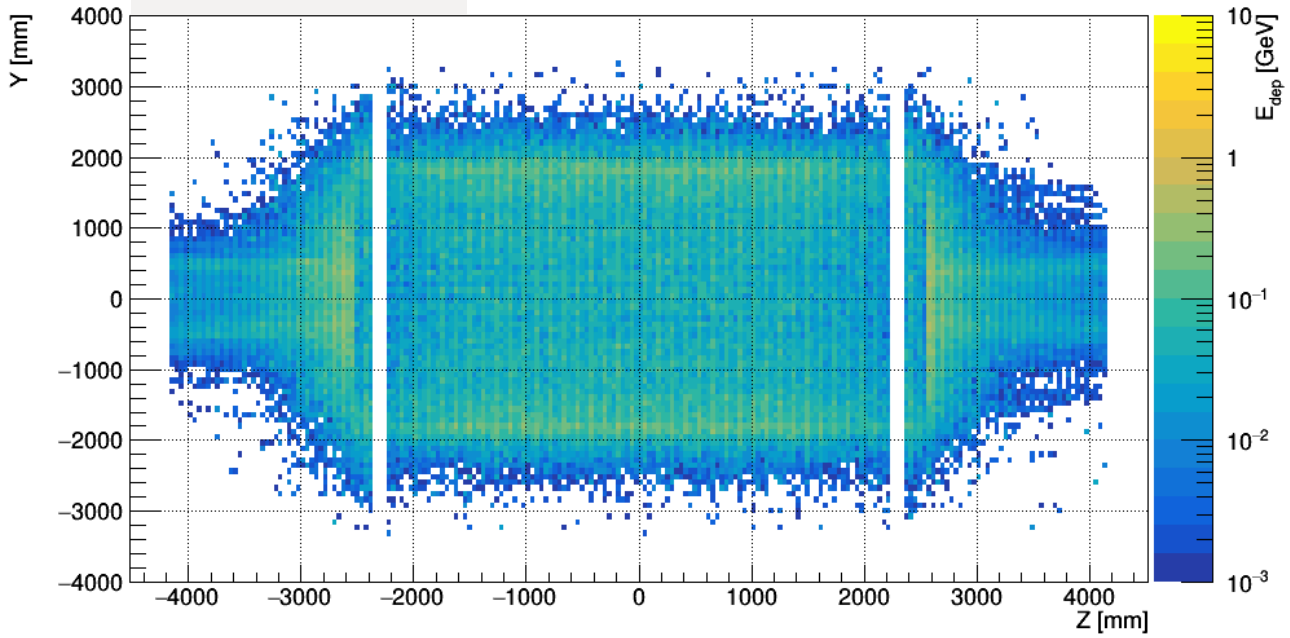
After applying the time selection + adding μ^+ beam

$\mu^+ + \mu^-$ beams

ECAL



HCAL



Energy deposited by BIB reduced from

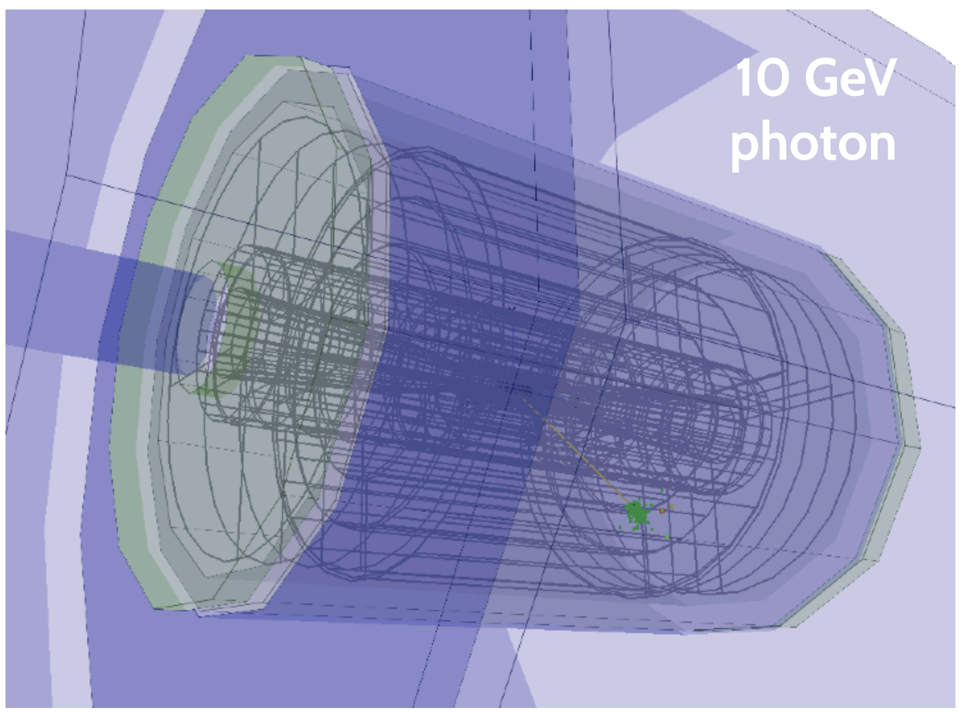
ECAL	HCAL
6 TeV	2.5 TeV

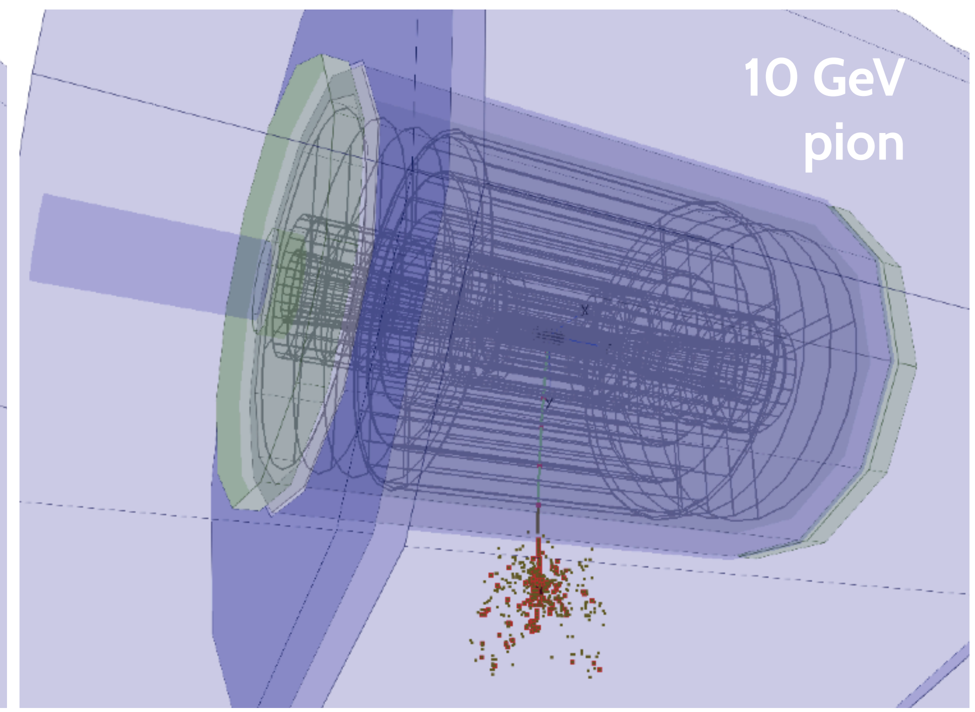
to

ECAL	HCAL
2.5 TeV	0.5 TeV

Having a look at the effect of timing cuts on the reconstructed energy resolution

- using the Pandora particle flow algorithm [arXiv: 0907.3577]
  - relies on reconstructed tracks and calorimeter hit clusters, which are not yet appropriately handled with the full BIB included in the event
  - no BIB included at the reconstruction step yet





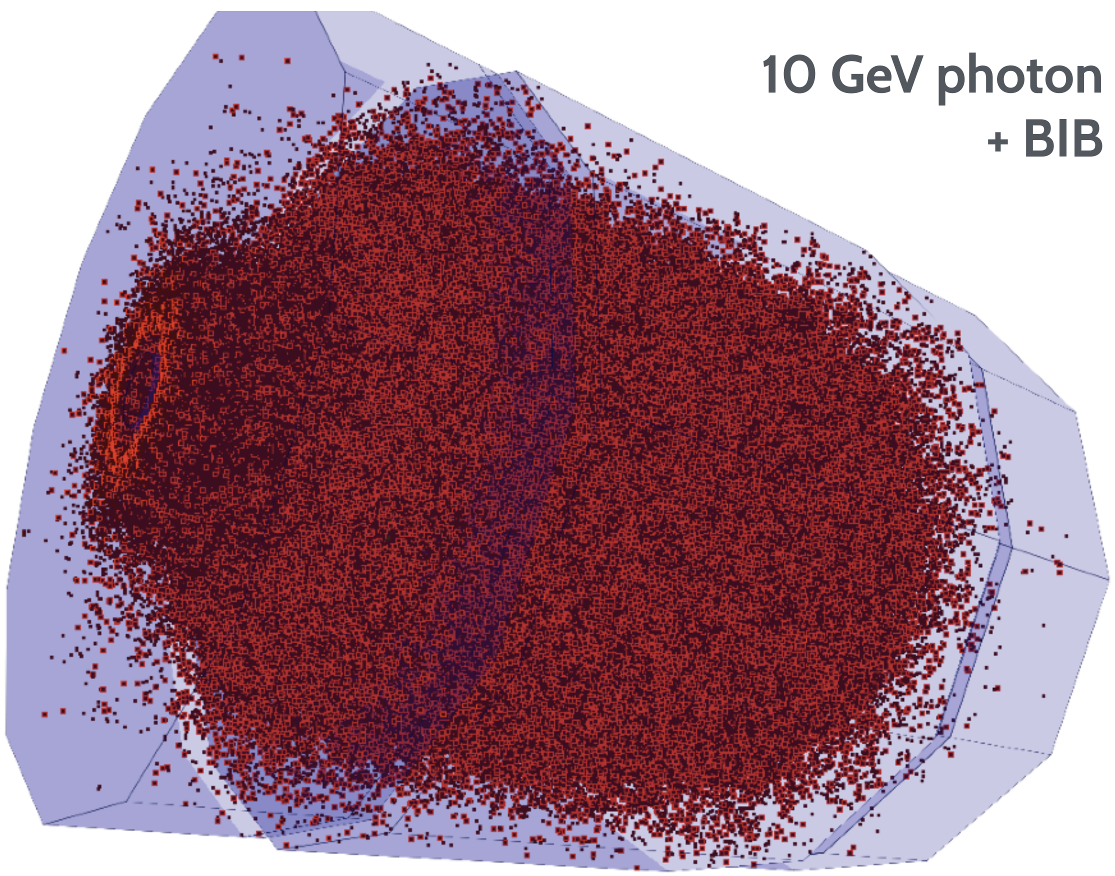
100 ns	10 GeV	100 GeV
photon	5%	1,6%
pion	19%	6%

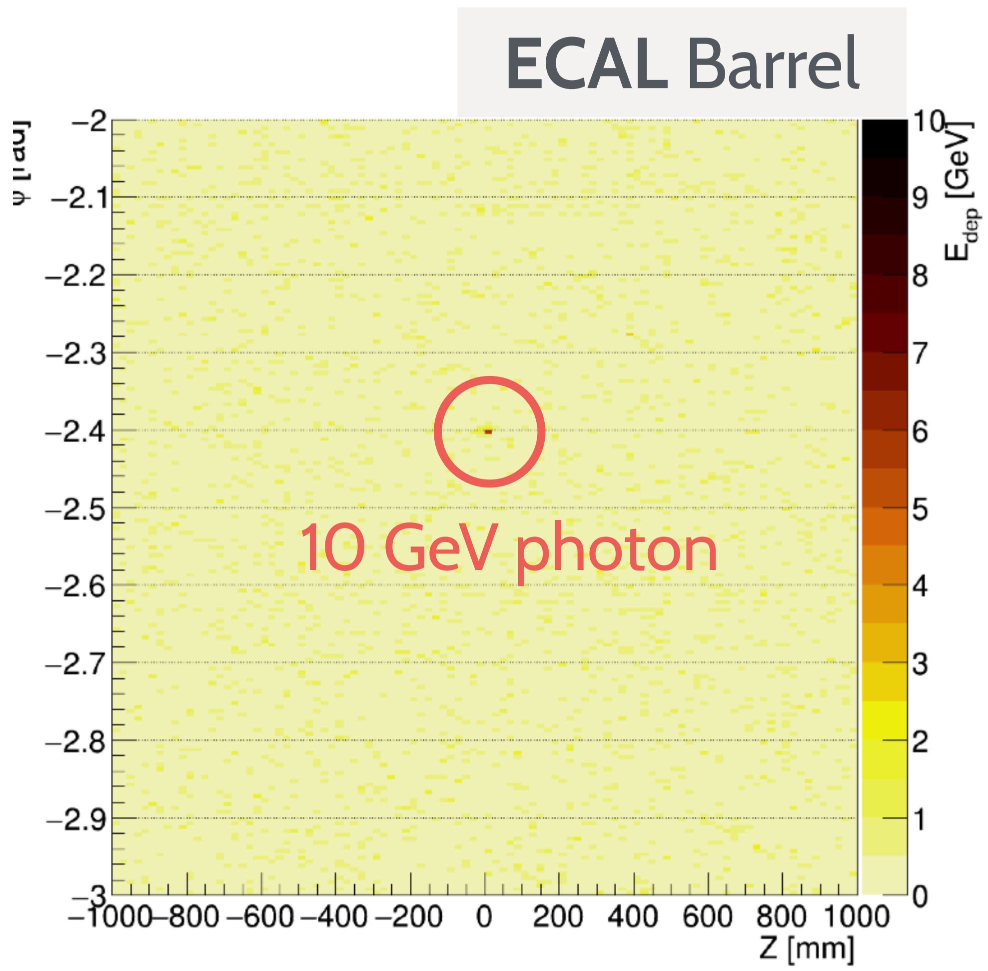
→

5/10 ns	10 GeV	100 GeV
photon	5%	1,7%
pion	20%	7%

Reconstruction with BIB

Things get very busy once energy deposits from BIB are added





Traditional calorimeter hit clustering algorithms are not applicable in such a busy environment

Proper subtraction of energy deposited by the BIB has to be implemented at the clustering stage of the particle flow algorithm