

WP7 Software

Graeme Stewart and André Sailer, for the WP7 team



Software for Future Detectors

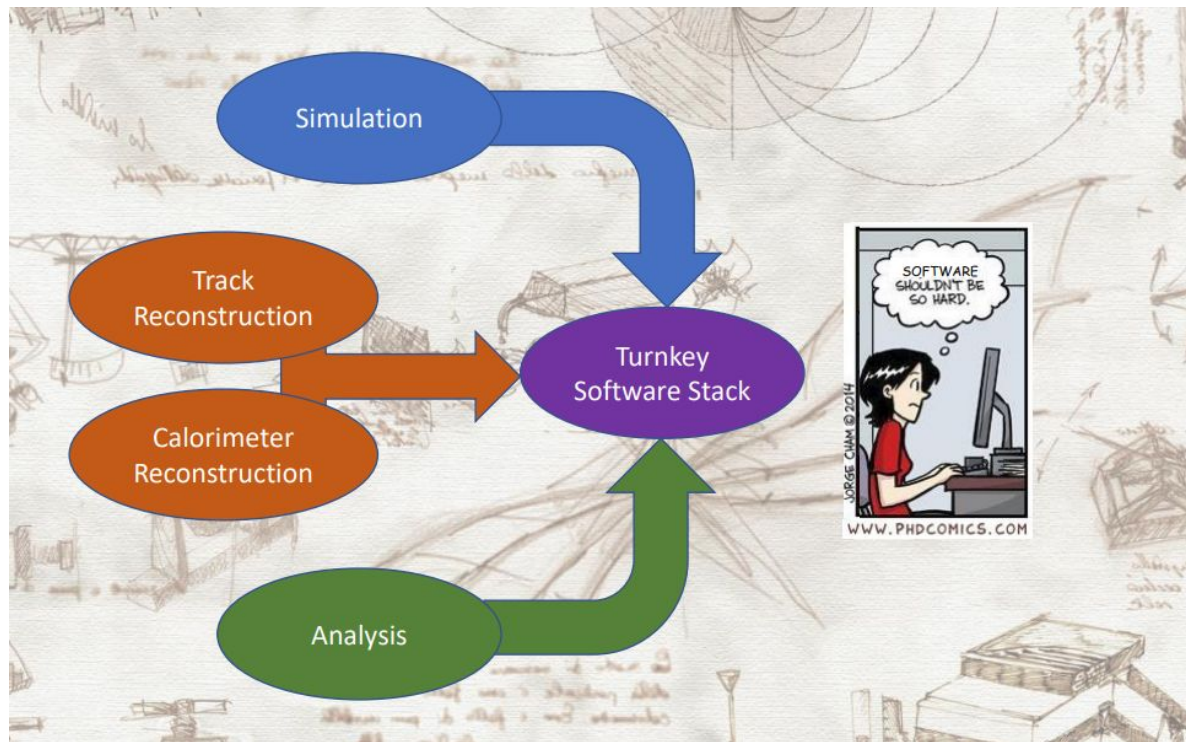
Challenges...

- Event rates
- Event complexity
- Precision physics

Plus...

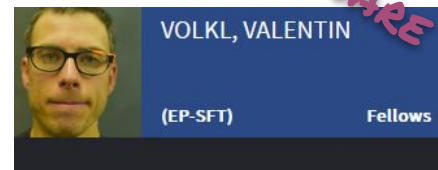
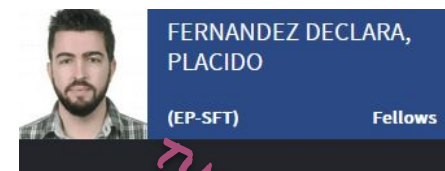
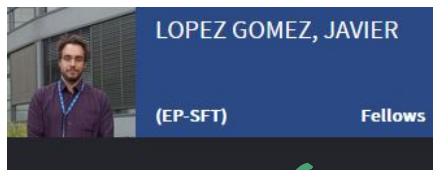
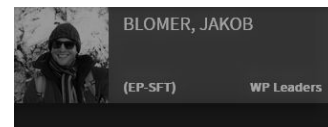
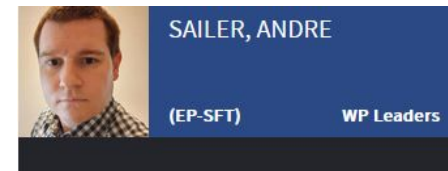
- Computing hardware
- Storage technology

For great detectors we need great software



WP7 - Meet the Team!

- Graeme Stewart leads the WP
- André Sailer has taken over from Jakob Blomer as deputy WP leader
 - Huge thanks to Jakob for all his work in WP7, in preparation and execution!



Task Leaders: Anna Zaborowska, Marco Rovere, Felice Pantaleo, Andi Salzburger, Jakob Blomer, André Sailer

SIMULATION

ANALYSIS

TURNKEY SOFTWARE

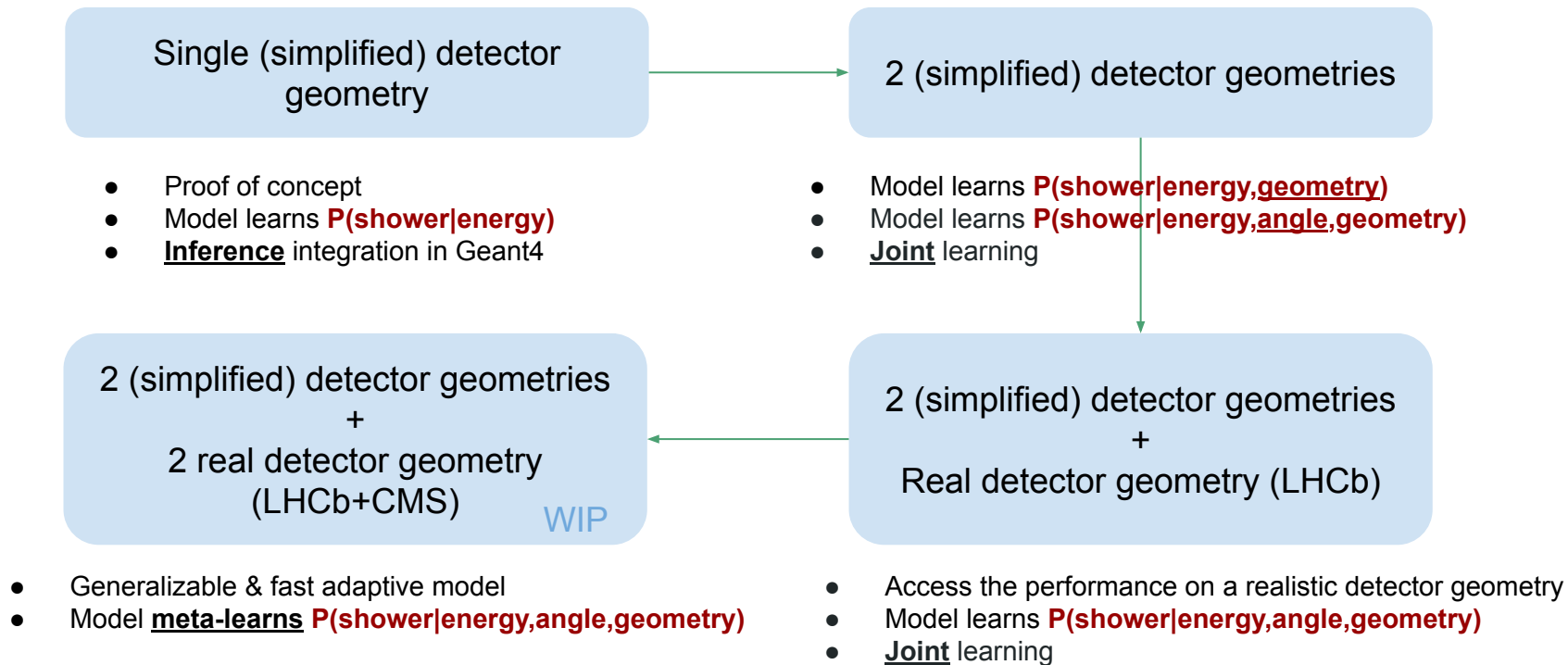
RECONSTRUCTION

Fast Simulation

Fast Simulation Task : Overview

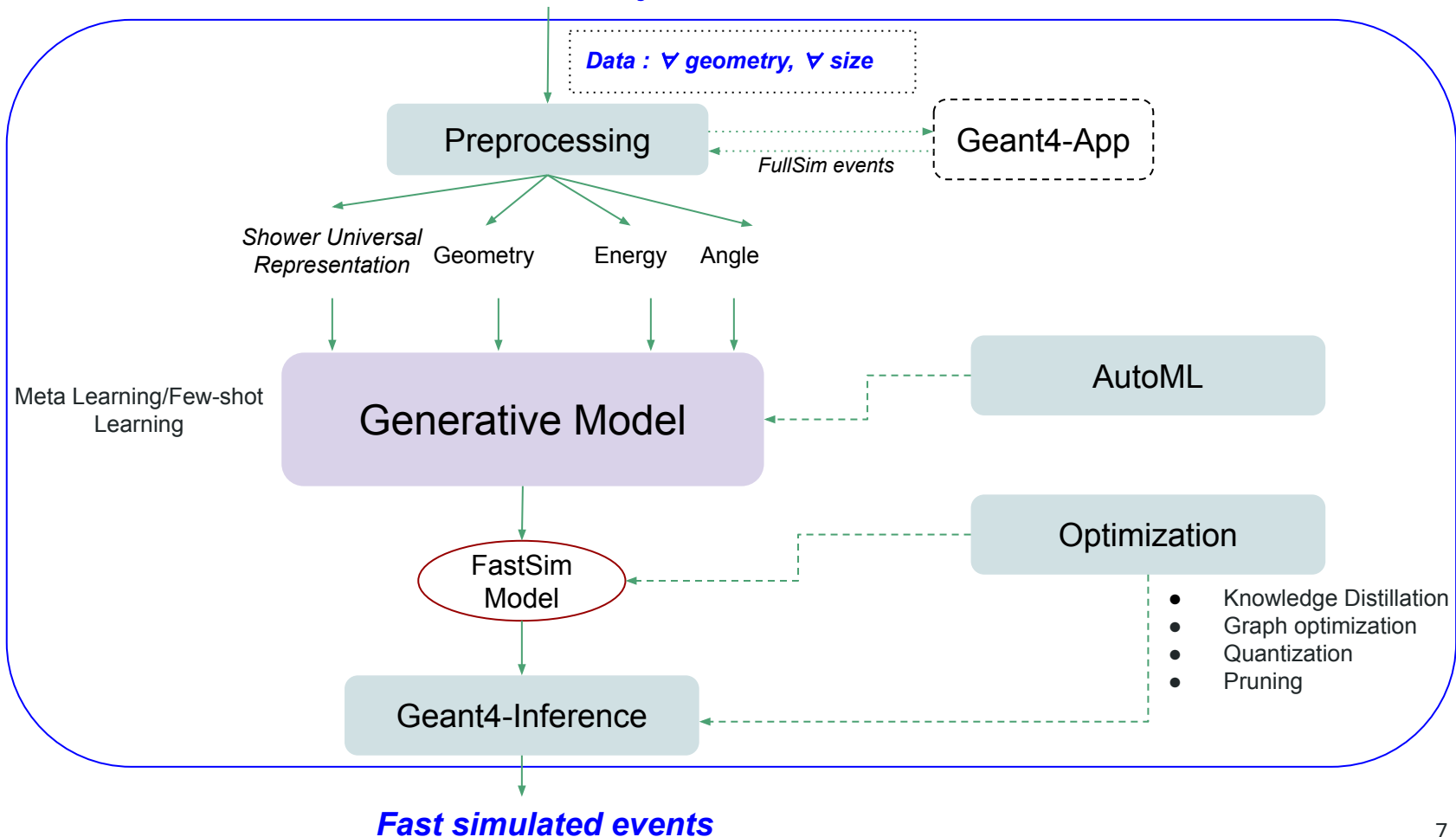
- Tools facilitating use of fast-sim models implemented into Geant4
- ML-based fast simulation
 - Demonstrated generalisation on sample geometries, for different energies and incident angles
 - Fast adaptation step allows to tune to any geometry
 - LWTNN and ONNX for inference within Geant4
 - Network optimization to reduce memory footprint of inference
 - Ongoing work on validation for use on realistic geometries, with geometry-independent scoring
 - Ongoing work on [Geant4 example](#)
- Synergy with AIDAInnova: recruiting a CERN junior fellow (starting in 2022)
- [ML4Sim](#) spin-off series of meetings
 - Machine learning for simulation discussion forum
 - Hybrid format of topical meetings and lectures
 - Discussions cover successful and unsuccessful ML tests
 - Keep in touch [mailing list](#)!

Towards a generalizable and a fast adaptive ML simulator

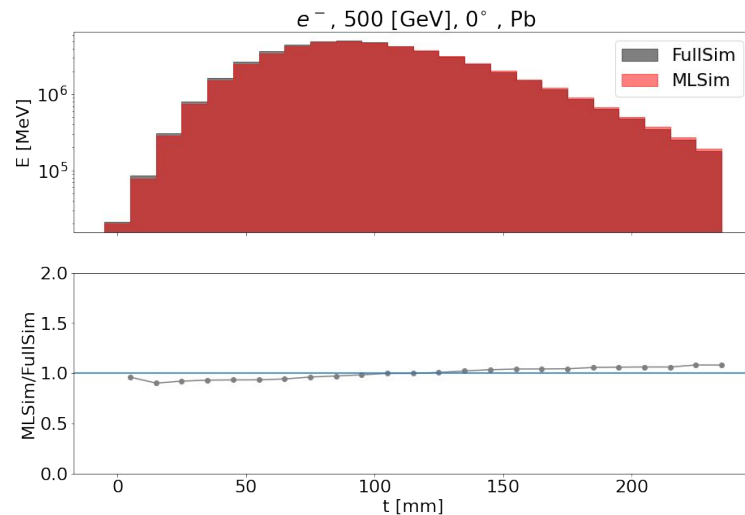
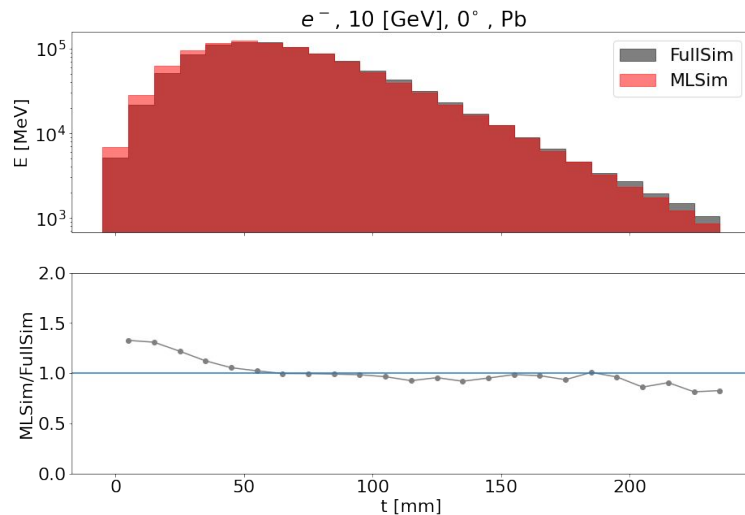


GDML File , Geometry features

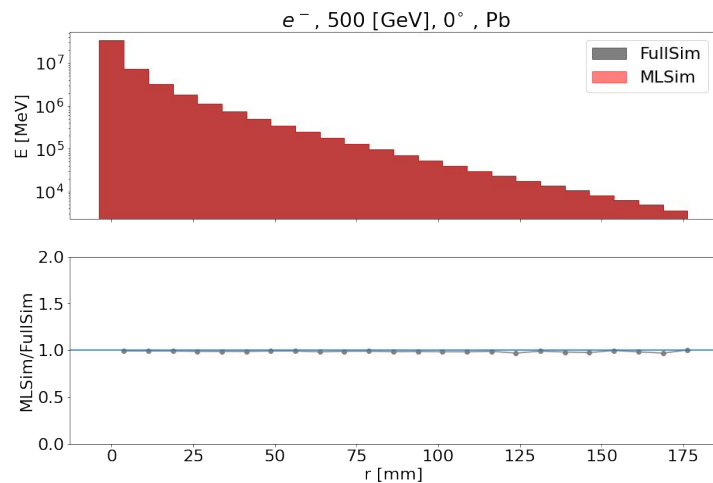
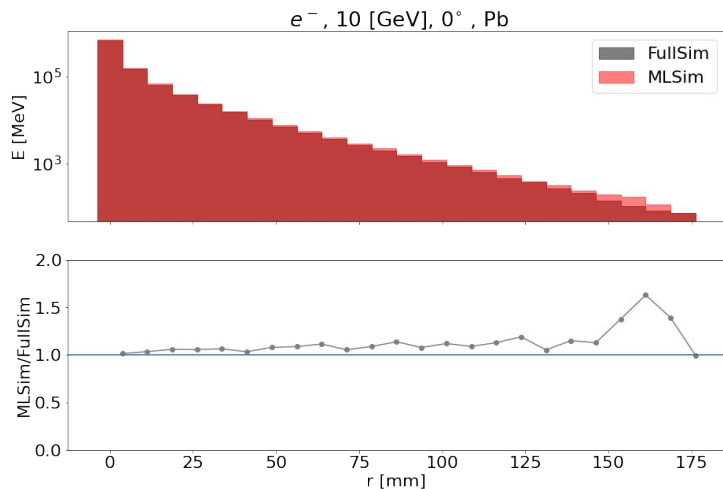
Pipeline ML-Simulator



Longitudinal profiles PBWO₄



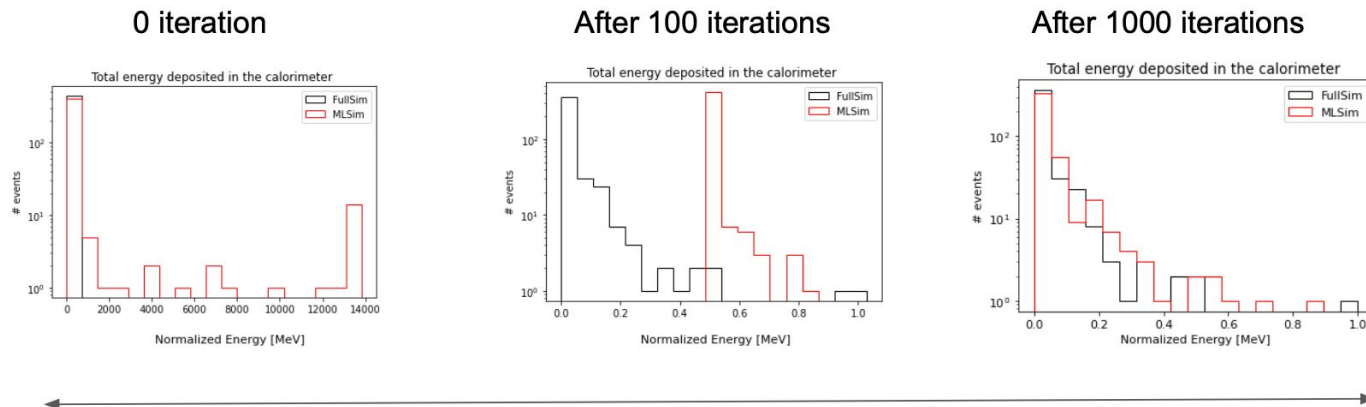
Transverse profiles PBWO₄



Adapting to a new geometry : CMS

WIP

Energy 60 GeV
Angle 20°



3mn on CPU, local machine (4 cores, 16Gb memory)

- Meta learning step: model trained on three detector geometries, two simplified geometries (PBWO₄ and SiW) and one real detector geometry (LHCb)
- Fast adaptation step: the pretrained model is adapted to the CMS geometry

ML-Inference & optimization with ONNX & LWTNN

| | LWTNN | ONNX |
|-----------------|-------|------|
| Resident Memory | 4GB | 61MB |
| Virtual Memory | 4GB | 52MB |

| Optimization with ONNX | Raw Model (without optimization) | Quantization | Quantization + Graph Optimization |
|------------------------|----------------------------------|--------------|-----------------------------------|
| Resident memory (MB) | 2265.34 | 650.414 | 555.828 |
| Virtual memory(MB) | 3205.26 | 1339.22 | 1073.21 |

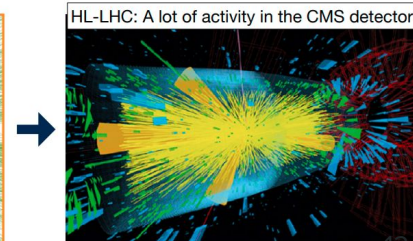
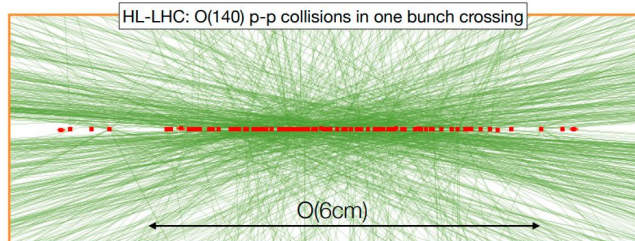
↑
Very large network

Calorimeter Reconstruction



Reconstruction at high-pileup

- Particle shower reconstruction in **high-granularity calorimeter** is very interesting and crucial task in high-density environments
 - Typical situation at HL-LHC → Many showers tend to **overlap**
 - Standard reconstruction algorithms using combinatorics are expected to fail due to **memory/timing explosion**
 - **Fertile ground** for new techniques and algorithms: clustering, machine learning, graph theory, and modern computer architectures
 - Planned and designed, taking into account the information from the tracking system and timing detectors
- Development can profit from experience with CMS Particle Flow techniques
- New flexible framework can be reused in other (future) experiments using high-granularity calorimeters

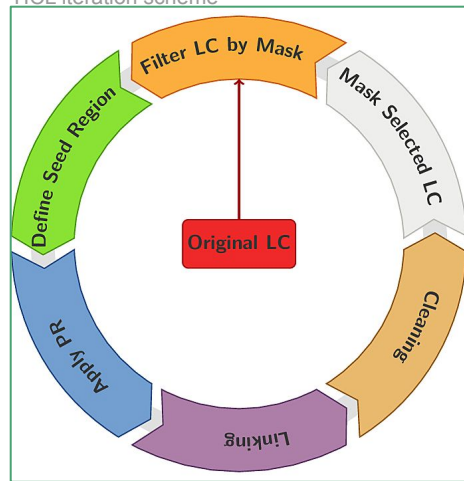




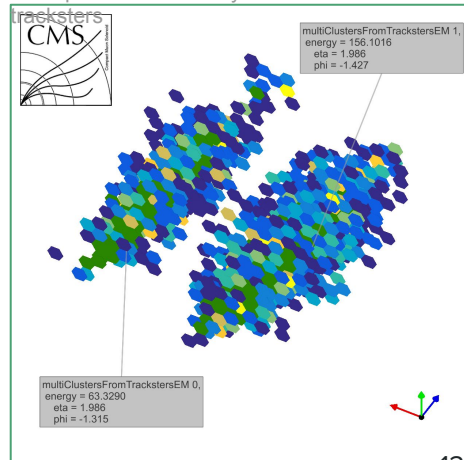
Reconstruction in HGCAL

- TICL (**The Iterative Clustering**) is a **modular framework** integrated and under development in CMSSW
- Main **purpose**: processing calo rec-hits (x, y, z, t, E) and returning particle properties and probabilities
- In a nutshell: grouping 2D layer clusters (**CLUE**) into 3D clusters (Tracksters) **iteratively** to reconstruct different particle species
- Important features:
 - No prior knowledge of CMSSW needed to contribute
 - Modules are designed such that new algorithms or techniques (e.g. Machine Learning) can be plugged on top easily
 - Algorithms are designed as swappable plugins, with heterogeneous architectures / portability in mind
 - Mostly geometry independent
- [Documentation](#)

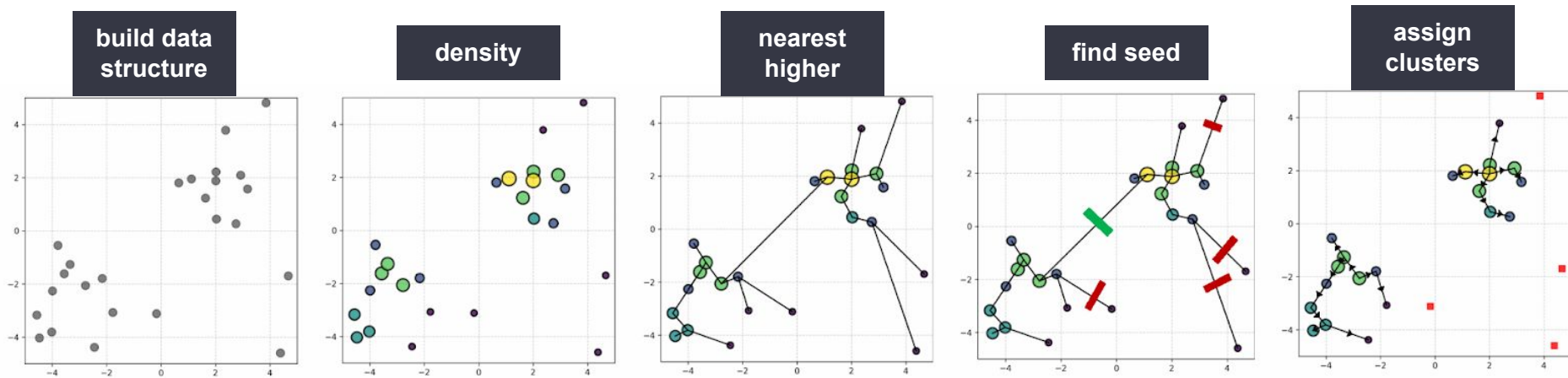
TICL iteration scheme



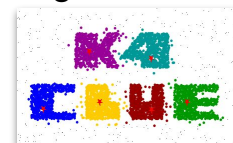
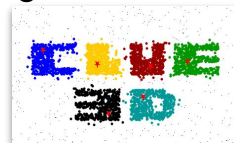
Example of two close-by reconstructed



CLUstering of Energy



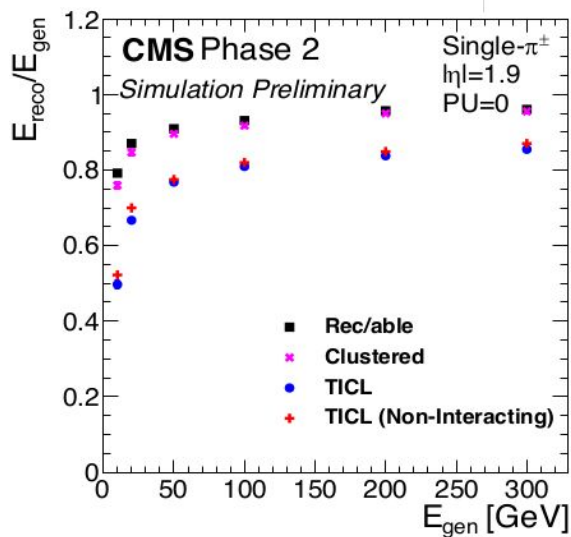
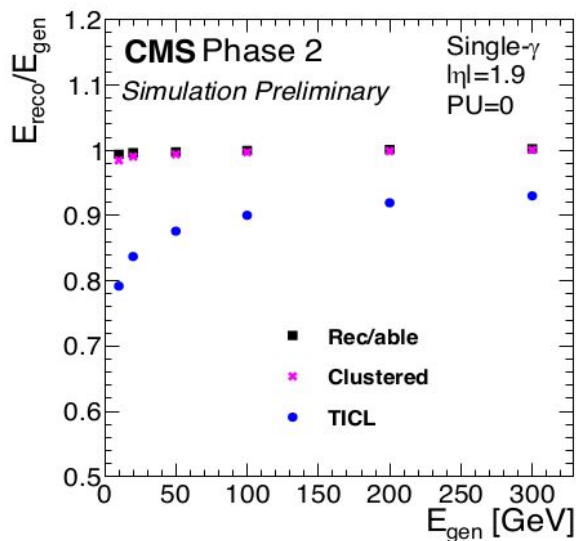
- CLUE (**CLUstering of Energy**) is an clusterisation algorithm based on *Energy density* - rather than individual cell energy - used to define ranking, seeding threshold, etc...
- GPU-friendly, i.e. suitable for the upcoming era of heterogeneous computing in HEP
- CLUE lives in a standalone repository



CLUE as 2D clustering algo



- CLUE (and full TICL) has been tested on electromagnetic showers and hadronic showers
 - Current approach: reconstruct the whole shower as a single trackster
- Preliminary results are very promising



Rec/able: Sum of energy of all RecHits produced by the generated particle

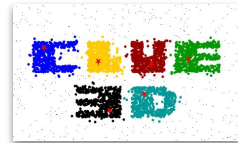
CLUE: Sum of the energy in the 2D clusters returned by the CLUE algorithm

TICL: sum of the energy in the shower reconstructed using TICL

TICL (non-interacting): sum of the energy in the shower reconstructed using TICL produced by pions that did not interacted in the tracker volume in front of HGCal



CLUE as 3D pattern reco algo



- CLUE3D is **not** the implementation of CLUE in 3 dimensions
- It starts from CLUE Layer clusters and **clusters** them in 3 dimensions, thus CLUE2D+1
 - Clustering happens in z, η, ϕ space, and is projective
 - Clustering within the same layer is off by default
- Integrated in CMSSW as additional **two new iterations** depending on the minimum local density (energy) to be promoted as a seed:
 - CLUE3DHIGH
 - CLUE3DLOW
- Valid proof of concept for clusterisation in “3”D
 - TICL framework extremely versatile
 - Keep on exploiting its potential, “using one more dimension”, but its optimal tuning and best usage have yet to be fully studied and understood

Track Reconstruction

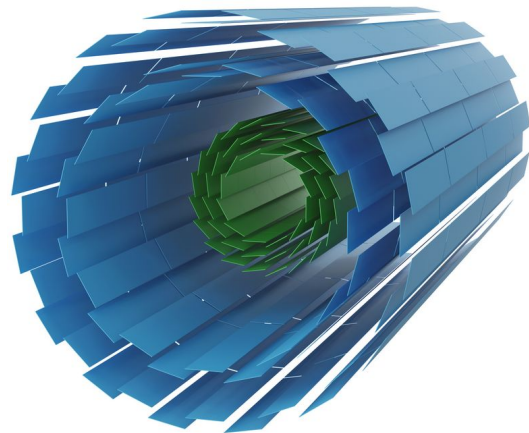
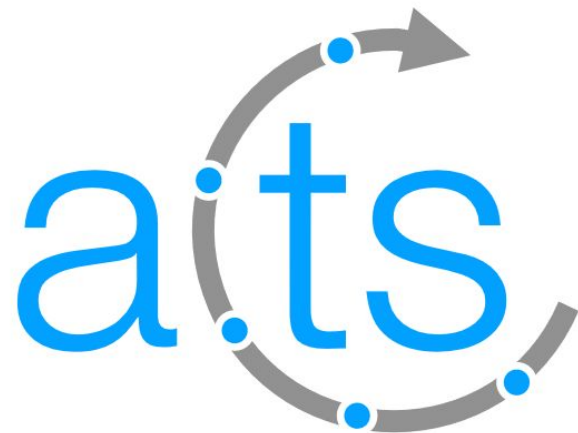
ACTS Tracking

ACTS

- Experiment independent toolkit for track reconstruction
- Modern architecture and code, unit tested, continuous integration
- Minimal external dependencies, easy to build
- Robust concurrency through thread-safety by design
- Community platform for R&D across various experiments




Addressing similar challenges found for calorimeter reco (pileup, overlapping clusters, cpu/memory scaling).




ACTS is the natural project into which EP R&D work fits, excellent synergy with AIDAinnova



EP R&D Tracking work in 2021

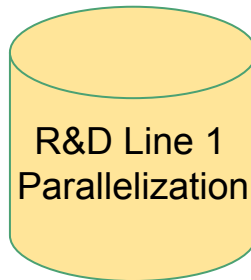






`acts-project` finalize
proof-of-concept demonstrator for
fully multithreaded (CPU) appl.   





Extensive profiling & CPU/thread
utilization studies (next slide)   





Further optimization of geometry &
navigation   

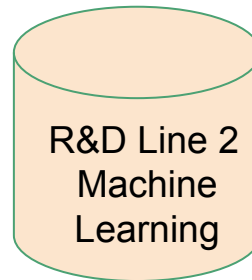
Event Data model optimization 







`traccc` demonstrator as a
simplified tracking chain for
EDM & algorithm R&D    




`detray` non-polymorphic
tracking geometry
description    

`vecmem` unified EDM
memory & container
approach CPU/GPU for
`traccc` and `detray`    



finalize study on machine
learning based navigator  

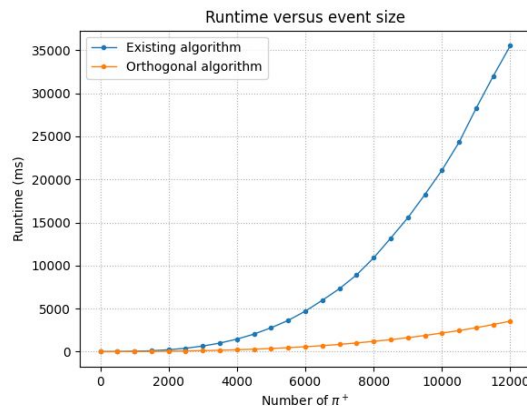
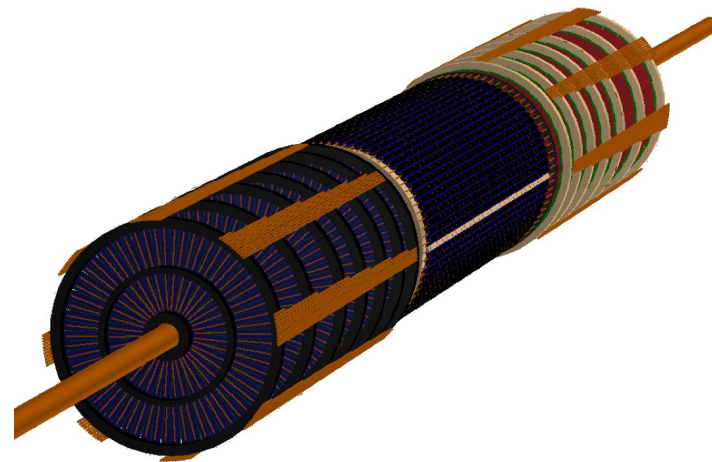
finalize Open Data Detector
dataset as replacement for
TrackML dataset  

ML based vertex type
classifier for ACTS   

EP R&D Tracking project status

Core development:

- New modules being developed
 - Gaussian Sum Filter, new seed finding on KD-Trees
- Full chain tracking demonstrator established (TrackML, OpenDataDetector)
 - Large scale validation and tuning efforts ongoing (next slide)
 - [OpenDataDetector](#) will be a proving ground for R&D that is OpenScience
- ACTS Paper [[2106.13593](#)] accepted for Computing and Software for Big Science (CSBS)



New seed finding strategy on based of KD-trees,
achieving very similar tracking performance

EP R&D Tracking project status

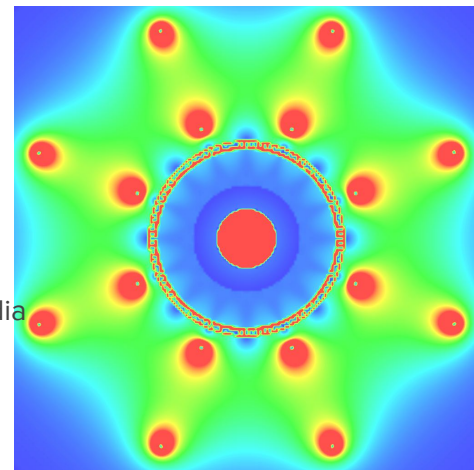
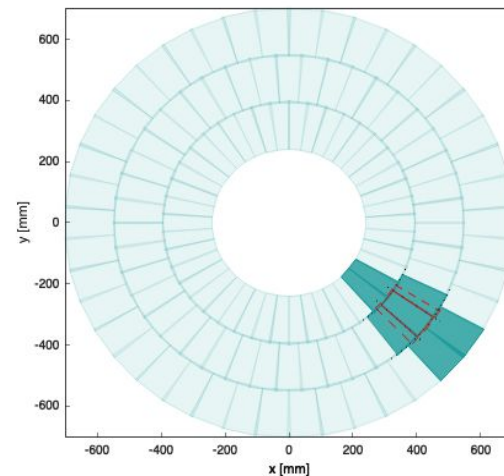
R&D line 1: `acts-parallelization`:

- `vecmem` (datamodel), `detray` (geometry) and `traccc` (algorithms) libraries are well progressing, aiming at cross-compilation of large parts of code for heterogeneous architectures
- ACTS GPU Kalman filter paper published (CSBS), follow-up project started using the libraries above
- First prototype of representing ATLAS magnetic field using GPU texture memory

R&D line 2: `acts-machinelearning`:

- prototype of the parameter tuning setup (with iterative steering)
- work with Exa.Trk on full pipeline integration

TrackML detector layer in n detray



ATLAS magnetic field slice on Nvidia GPU's texture memory
[S Swatmann]

Efficient Analysis



I/O for Analysis at Future Colliders

1. HL-LHC as a baseline for requirements from future colliders:
 - From 300fb^{-1} in Run 1-3 to 3000fb^{-1} in Run 4-6
 - 10B events/year to 100B events/year
 - **x10 the current demand**
 - **Remote I/O** and **beyond-grid** (HPC & cloud) storage systems
 - Data pipelines into **machine-learning accelerators**
2. Full exploitation of modern storage hardware
 - Ultra fast networks and SSDs: 10GB/s per device reachable (HDD: 250MB/s)
 - Flash storage is inherently parallel \rightarrow asynchronous, parallel I/O key
 - Heterogeneous computing hardware \rightarrow I/O needs to efficiently feed many-core applications and GPU kernels
 - Distributed storage moves from file systems to object stores



Blurring between I/O and compute



ROOT RNTuple R&D Overview

RNTuple: major upgrade of the event data file format and access API

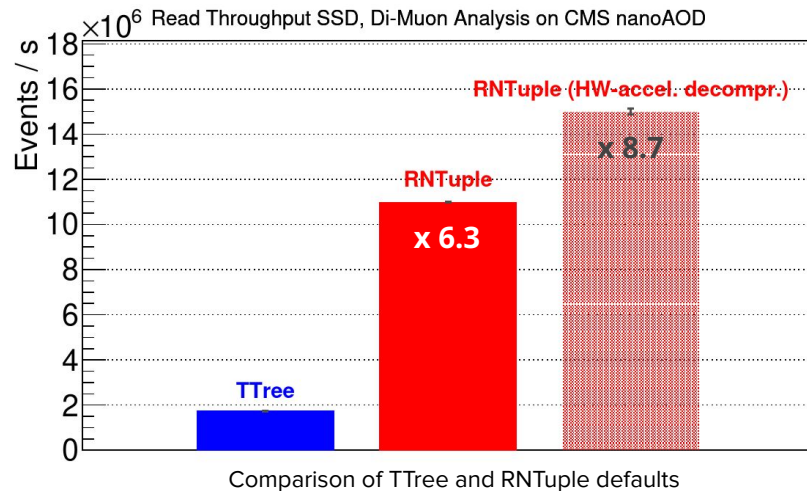
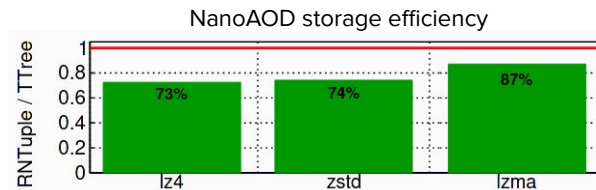
- Less disk and CPU usage for same data content
 - **10-20% smaller** files, **order of magnitude better throughput**
- Native support for HPC and cloud object stores
- Lossy compression
- Support for memory mapped I/O
- Systematic use of checksumming and exceptions to prevent silent I/O errors

I/O system closely integrated with analysis workflows

- Optimized for selective reads (skims etc.)
- Support for rich event data models (EDMs)
- Vertical and horizontal data set joins (“friends”, “chains”, ...)
- Good integration with multi-threaded frameworks
 - New: NanoAOD RNTuple output module in the CMS software framework (supported by IRIS-HEP)
- Support for code & data evolution over decades

Performance and functionality unmatched by any other available data format / API

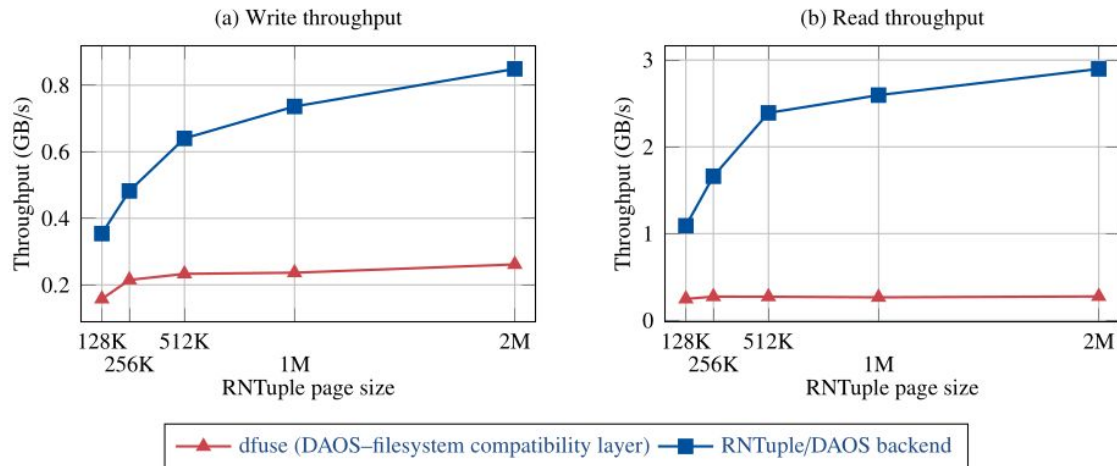
Redesign of the I/O system based on 25 years of TTree experience: provides a leap in performance and access to upcoming hardware choices





RNTuple Integration with HPC Object Stores

- Traditional distributed file systems limit scalability due to strong consistency guarantees
 - Cloud storage providers moved to HTTP based object stores
 - HPC is moving to high-throughput, low-latency flash storage based object stores
- Joint EP R&D, CERN openlab, Intel and HPE activity on optimal use of object stores
 - Using openlab testbed of Intel DAOS HPC object store
 - Exploring native object store capabilities compared to file system emulation
 - Order of magnitude better throughput with dedicated support in the ROOT I/O layer
- Experience can be in large parts transferred to cloud object stores (S3)



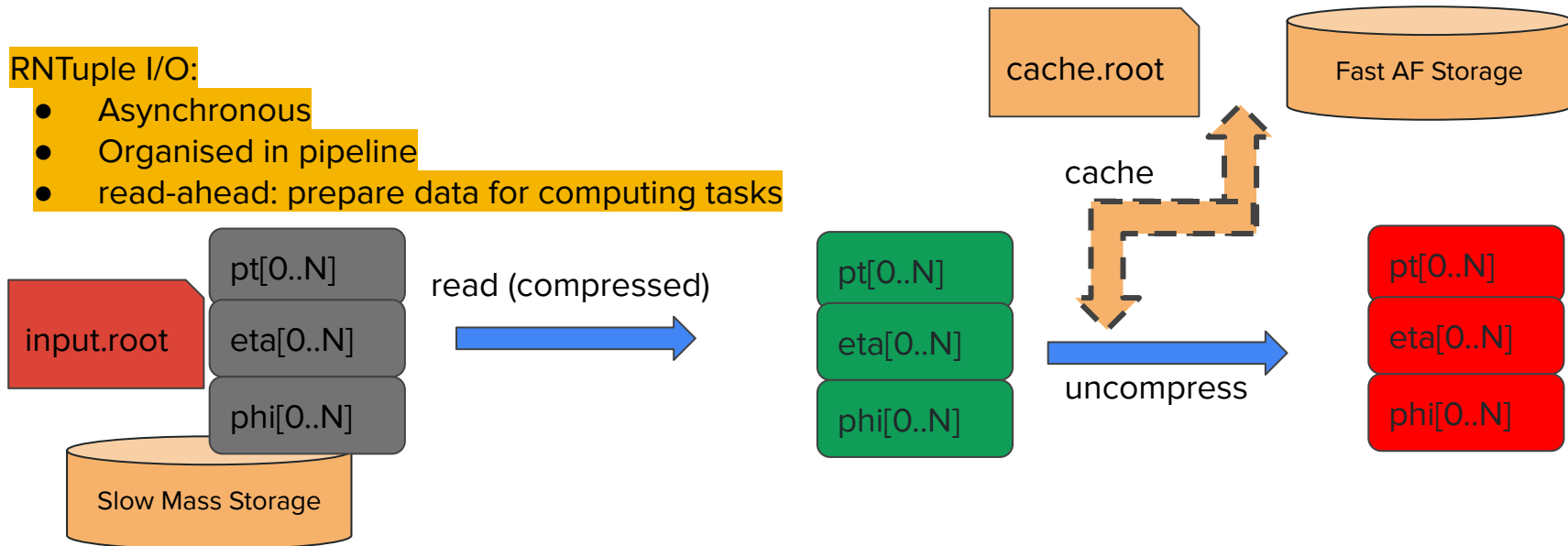


Transparent Caching for Analysis Facilities

- Targets **iterative data exploration** on the same dataset in an analysis facility
- Plugged into the RNTuple IO pipeline, cache on the fly. No extra user code required.
- Caching system **independent** of the storage **backend**: e.g. read from XRootD, write to HPC object store
- Application-defined: caches precisely what is needed by a given analysis

RNTuple I/O:

- Asynchronous
- Organised in pipeline
- read-ahead: prepare data for computing tasks



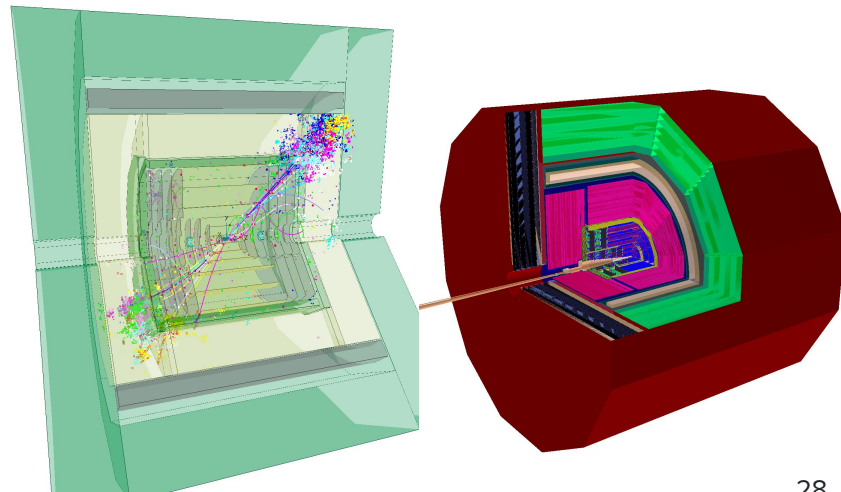
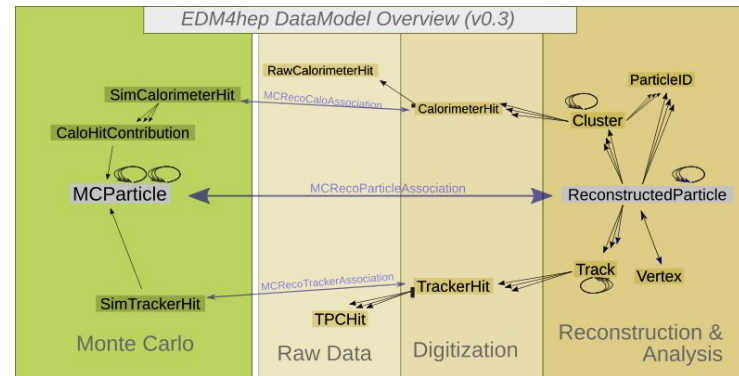
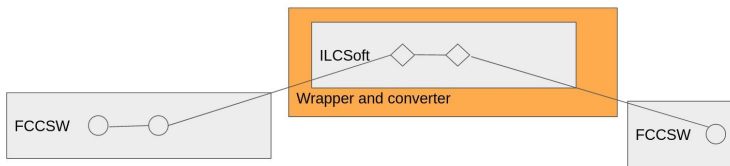
Turnkey Software Stack

Key4hep: Turnkey Stack

- EP RnD Goals:
 - Provide a single software stack for FCC and CLIC detector studies
 - Integrate functionality from iLCSoft used for CLIC and FCCSW
 - Make software stack usable by other detector groups
 - Testbed for other software developed in WP7

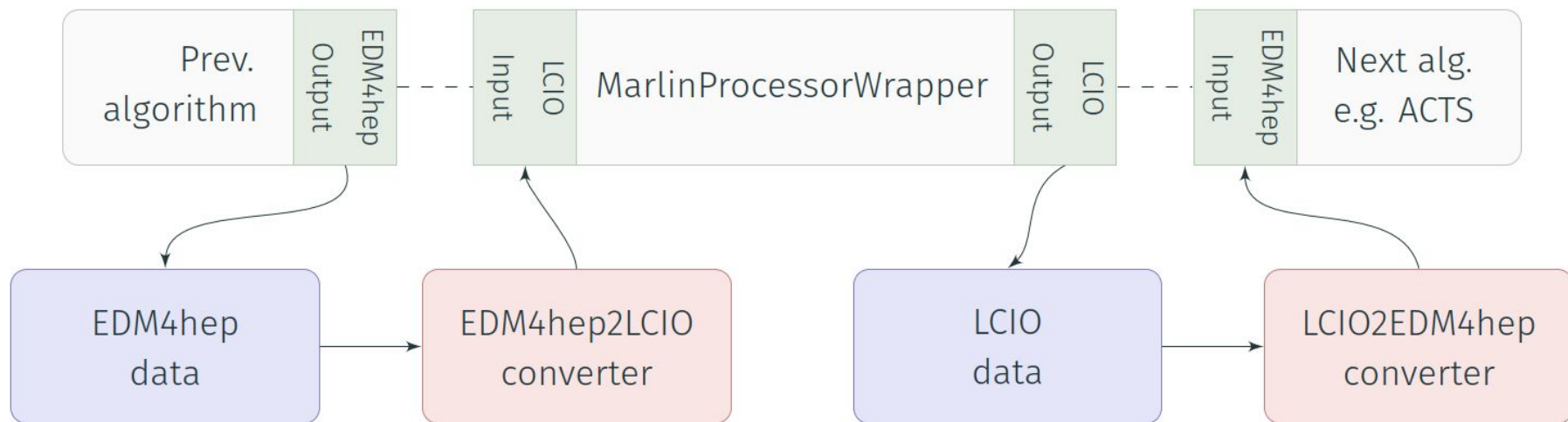
Common Software

- Gaudi Processing Framework
- EDM4hep Event Data Model using Podio
- DD4hep Geometry Toolkit
- Standard pieces, e.g., Geant4
 - Allows Fast Simulation developments to be integrated



k4MarlinWrapper and Key4hep

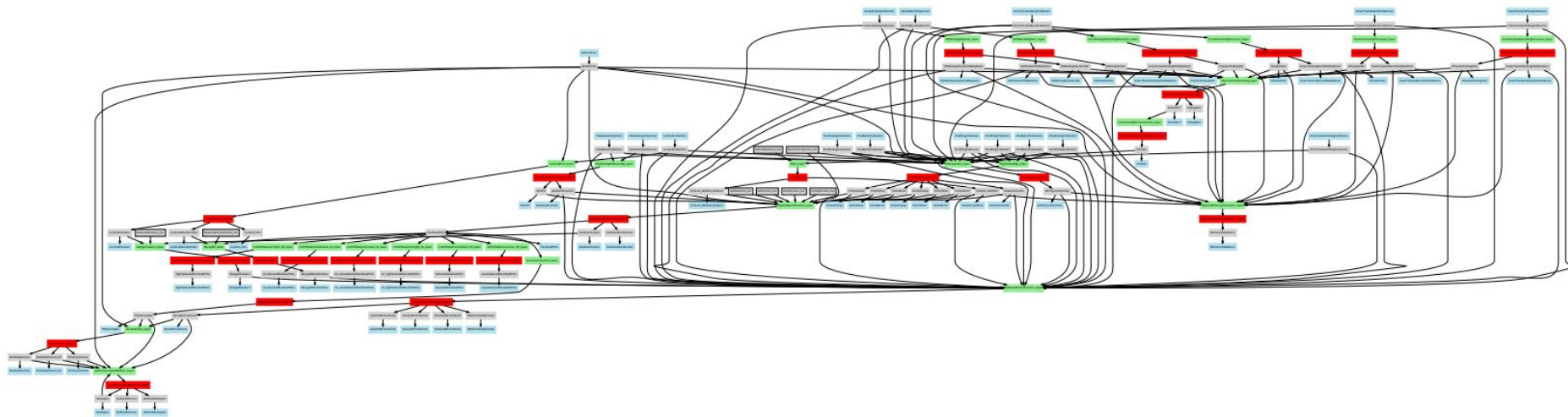
- Using the k4MarlinWrapper, all iLCSoft functionality is brought to Gaudi
- In memory conversion of from EDM4hep to LCIO before calling unmodified iLCSoft processors, convert LCIO to EDM4hep after the processor finished



CLIC Reconstruction

- Implemented CLIC reconstruction workflow with k4MarlinWrapper
 - Validation Pending

- Blue: EDM4hep collections
- Grey: LCIO collections
- Green: Marlin Processors inputs
- Red: Marlin Processor output



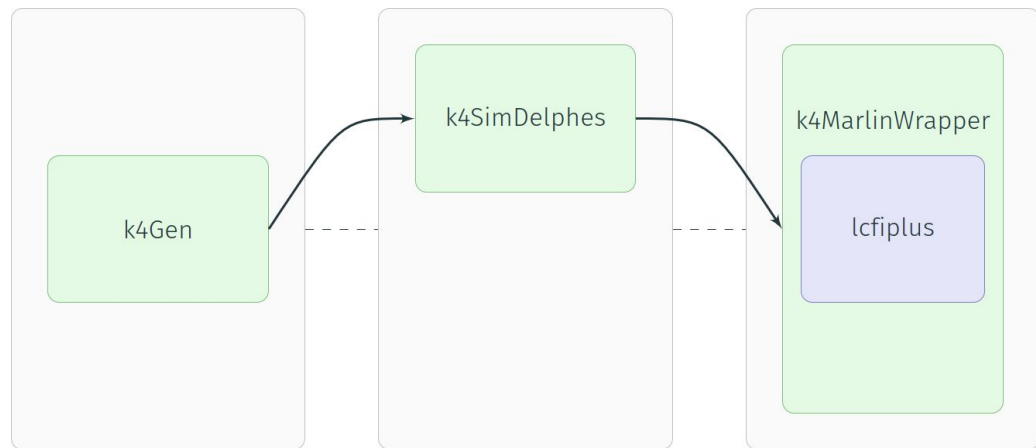
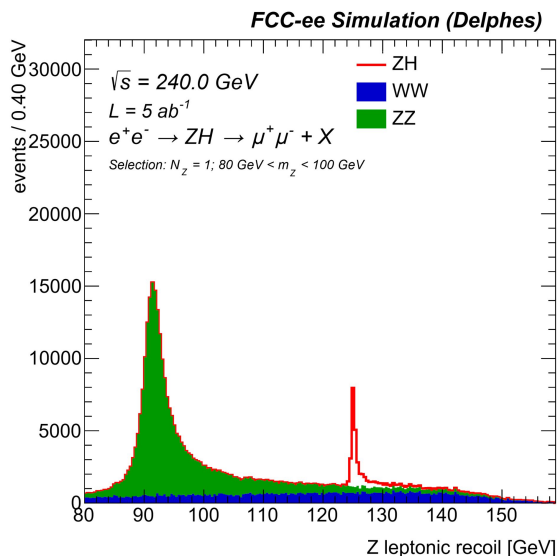
FCCSW - Reorganization for Key4hep

- FCCSW was based on Gaudi and PODIO (fcc-edm) from the start
 - Less effort to transition, switch to EDM4hep already finished
- hep-fcc/fccsw mono-repository was split up to allow use and re-use of common components in Key4hep
 - **k4FWCore** (Framework Core): PodioData services and event overlay
 - **k4Gen**: generator interfaces and particle guns
 - **k4SimDelphes**: for delphes fast simulation with EDM4hep output

| old FCCSW (version ≤ 0.16) | Key4hep | new FCCSW | status | migration |
|------------------------------------|------------------|------------------|--------|------------------|
| FWCore | k4FWCore | | done | yes |
| Sim/SimDelphesInterface | k4SimDelphes | | done | yes |
| Generation | | k4Gen | done | under evaluation |
| Sim | | k4SimGeant4 | done | under evaluation |
| Reconstruction/Rec[...]Calorimeter | | k4RecCalorimeter | done | under evaluation |
| Reconstruction/RecDriftChamber | | to be determined | | |
| Detector | | FCCDetectors | done | no, FCC specific |
| | to be determined | dual-readout | | under evaluation |

FCCSW - Fast Simulation with k4SimDelphes

- k4SimDelphes integrates Delphes in Key4hep ecosystem
 - Standalone executables and seamless integration in Gaudi workflows
 - Many existing FCC Fast Sim studies now possible in EDM4hep
- Outputs EDM4hep in a way that is interchangeable with full sim output



Explore CLUE integration in key4hep

Combined CLUE and Key4hep work



- ✓ Compile CLUE in cmake
- ✓ Source key4hep and introduce EDM4HEP & PODIO as external libraries
- ✓ Create simulated EDM4HEP events
- ✓ Reading input / creating output in EDM4HEP format
- ✓ Create CLUE Gaudi Algorithm Wrapper
- ✓ Run CLUE in CLIC reconstruction
- Validation & use CLUE Clusters in CLIC reco



Key4hep International Community and Infrastructure

- Regular weekly meetings with CEPC/CLIC/FCC/ILC communities about Key4hep and EDM4hep
 - <https://indico.cern.ch/category/11461/>
- Source code repository and Continuous Integration on Github:
<https://github.com/key4hep>
- Regular software deployment to *CVMFS* using *Spack*, nightlies and releases
 - `/cvmfs/sw.hsf.org/key4hep/setup.sh`
 - Contributing to spack developments and LCG software stacks based on spack
- Documentation: <https://key4hep.web.cern.ch>

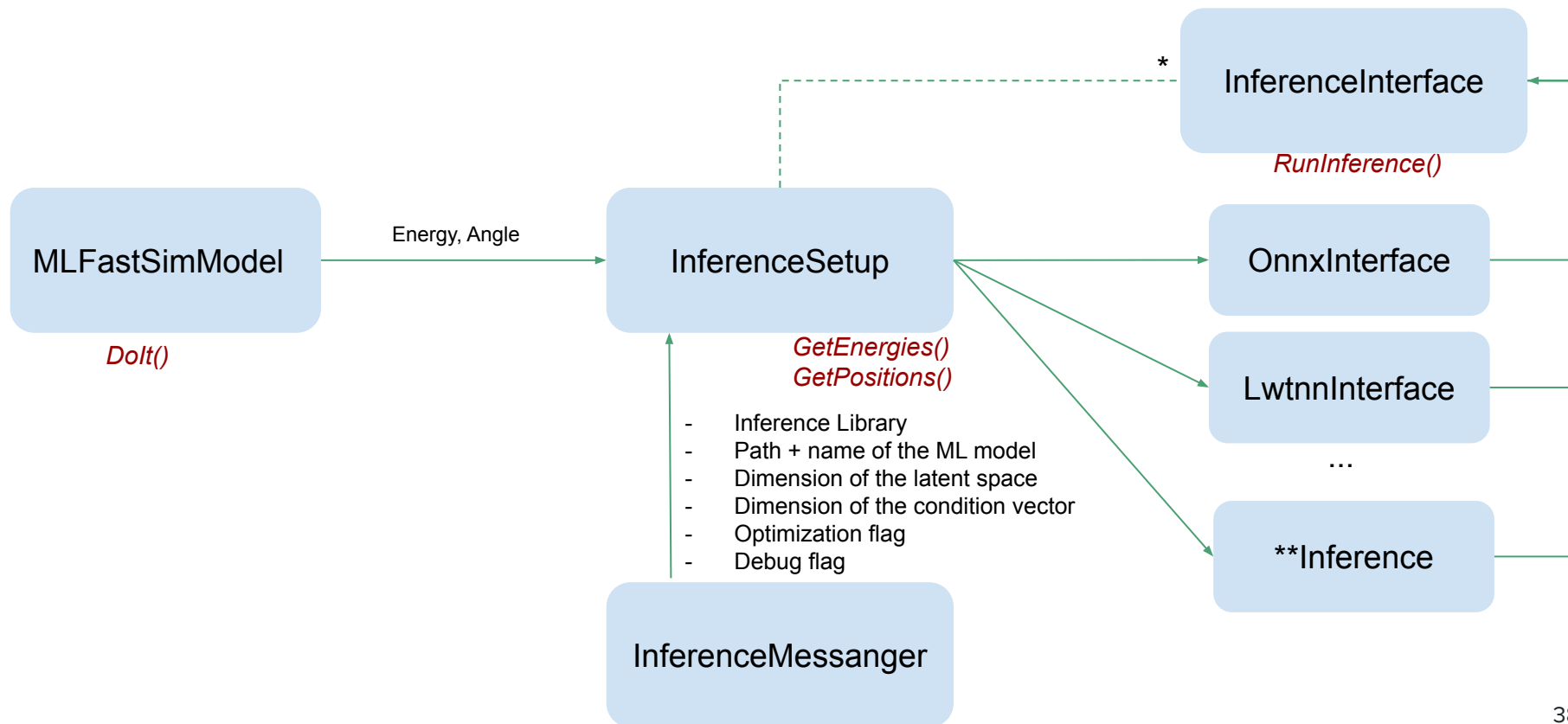
Conclusions

Conclusions

- WP7 is up and running at full speed
- Substantial progress being made in all tasks
- Dissemination of results is ongoing
 - ACAT (next major HEP software and computing conference, 29 Nov - 3 Dec)
 - CLUE: a clustering algorithm for current and future experiments
 - ACTS vecmem: Managing Heterogeneous Device Memory using C++17 Memory Resources
 - ACTS detrack: A compile-time polymorphic tracking geometry description
 - RNTuple Performance: Status and Outlook
 - Key4hep software Stack for Detector Studies
 - Fast Simulation: Meta-learning for multiple detector geometry modelling
- Strong links to AIDAinnova, IRIS-HEP, HSF

Backup Slides

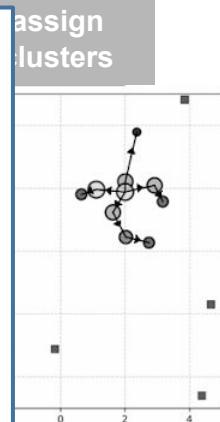
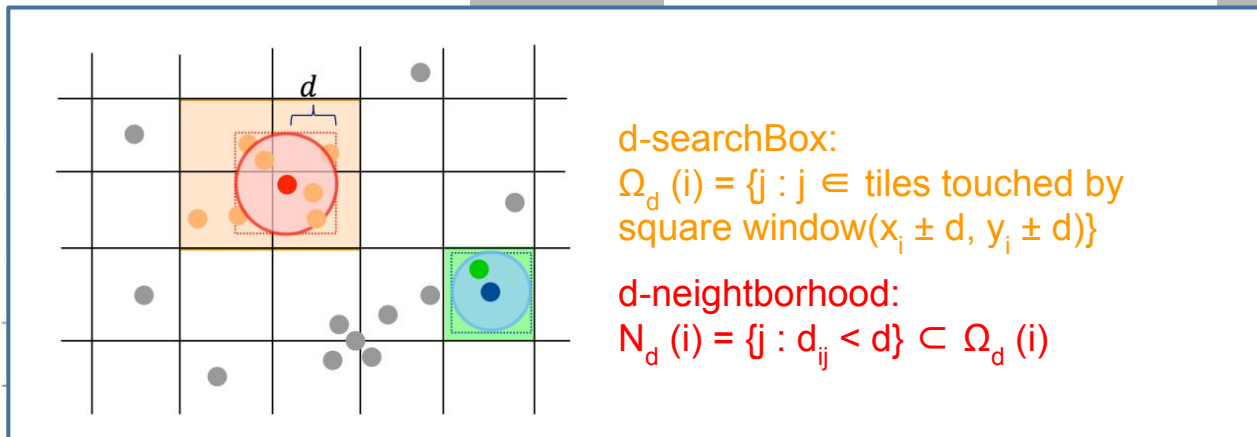
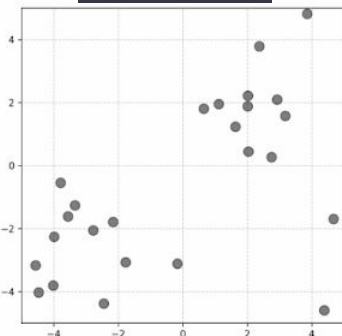
Geant4 example : ML-Inference with ONNX & LWTNN



Cluster Algorithm: CLUE

- Querying neighborhood is a frequent operation in density-based clustering → **fast!**
- Build **Fixed-Grid Spatial Index** for hits on each layer (η , ϕ space)
 - Grid tiles are small compared to the size of HGCal layer
 - Each tile in the grid hosts indices of hits inside it and has a fixed length of memory to store the hosted indices
- To find the neighborhood hits $N_d(i)$ of i -hit, we only need to loop over hits in $\Omega_d(i)$

build data structure



Cluster Algorithm: CLUE

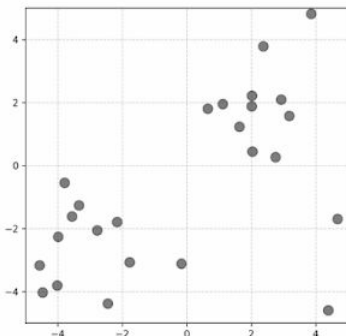
$\rho_c, d_c, \bar{\delta}_c, \bar{\delta}_0$ are tunable parameters chosen with purity vs fake studies

- Calculate local energy density (ρ) in a distance (d_c)
 - Each hit j weighted by the deposited energy (E_j)
 - For each hit, calculate ρ_i
 - Individual d_c values considered in HGCal Silicon and Scintillator sections

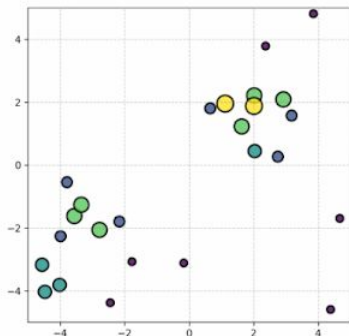
convolution kernel

$$\rho_i = \sum_{j \in N_d(i)} E_j \times f(d_{ij}); \quad f(d_{ij}) = \begin{cases} 1, & \text{if } i = j \\ k, & \text{if } 0 < d_{ij} \leq d_c \\ 0, & \text{if } d_{ij} > d_c \end{cases}$$

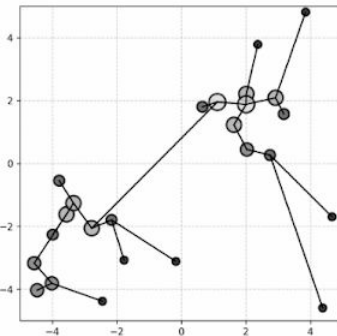
build data structure



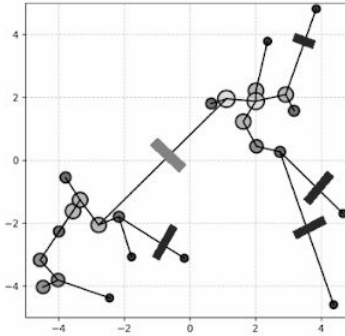
density



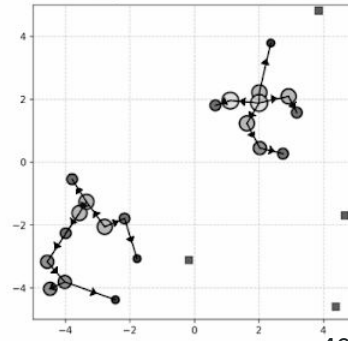
nearest higher



find seed



assign clusters



Cluster Algorithm: CLUE

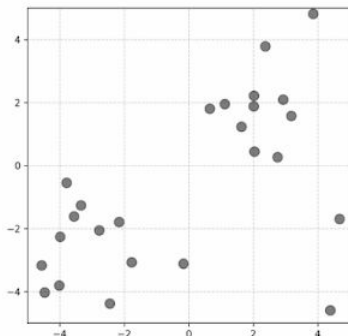
$\rho_c, d_c, \bar{\delta}_c, \bar{\delta}_0$ are tunable parameters chosen with purity vs fake studies

- Calculate “Nearest-Higher” hit within $N_{dm}(i)$
 - Define $d_m = \max(\bar{\delta}_0, \bar{\delta}_c)$, $\bar{\delta}_0$ and $\bar{\delta}_c$ parameters for outlier demotion and seed promotion
 - Find the closest hit with higher local energy density, nh_i

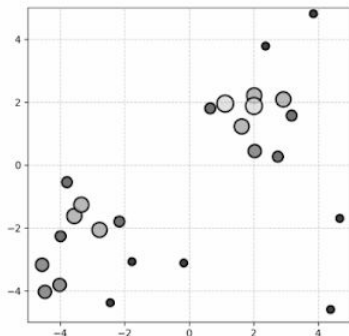
$$nh_i = \begin{cases} \underset{j \in \hat{N}_{dm}(i)}{\operatorname{argmin}} d_{ij}, & \text{if } |\hat{N}_{dm}| \neq 0, \hat{N}_{dm}(i) = \{j : j \in N_{dm}(i), \rho_j > \rho_i\} \\ -1, & \text{otherwise} \end{cases}$$

- Calculate the separation distance $\bar{\delta}_i = \operatorname{dist}(i, nh_i)$

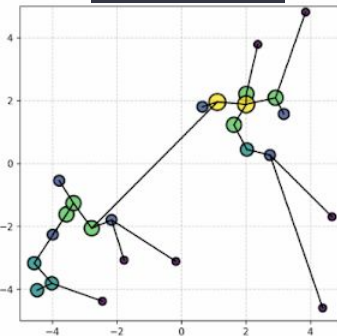
build data structure



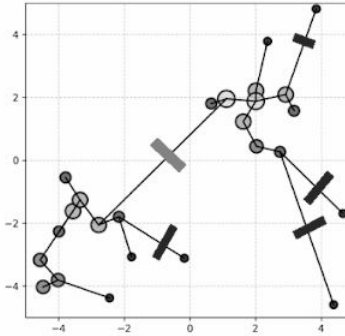
density



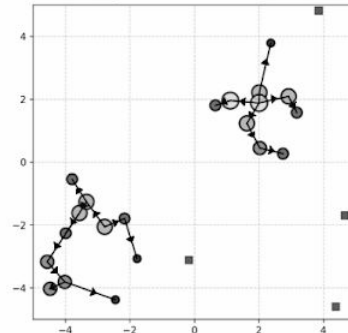
nearest higher



find seed



assign clusters



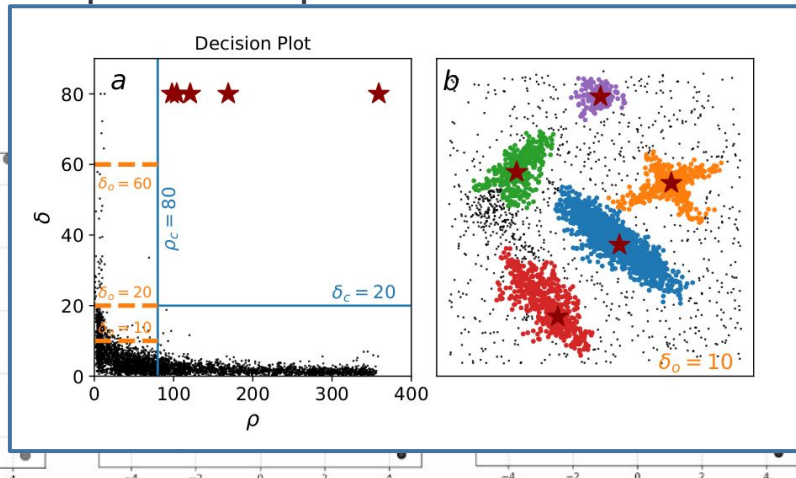
Cluster Algorithm: CLUE

$\rho_c, d_c, \delta_c, \delta_o$ are tunable parameters chosen with purity vs fake studies

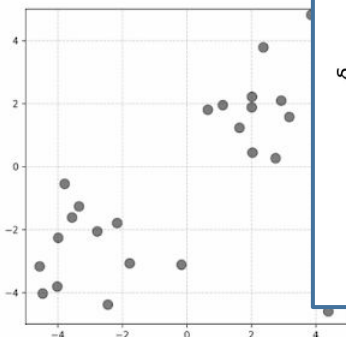
- Promote as **seed** if $\rho_i > \rho_c, \delta_i > \delta_c$
- Demote as **outlier** if $\rho_i < \rho_c, \delta_i > \delta_o$
- Assign unique, progressive clusterID to each cluster
 - **Followers** are defined and associated to their closest seed

→ Rock solid against noise
→ Clustering almost all energy
→ Tested successfully also on test beam data

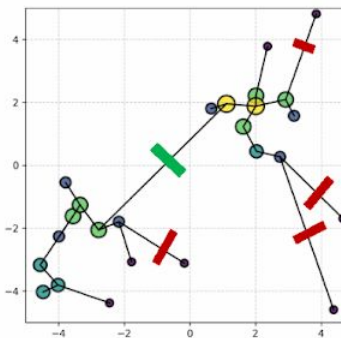
Example of decision plot



build data structure



find seed



assign clusters

