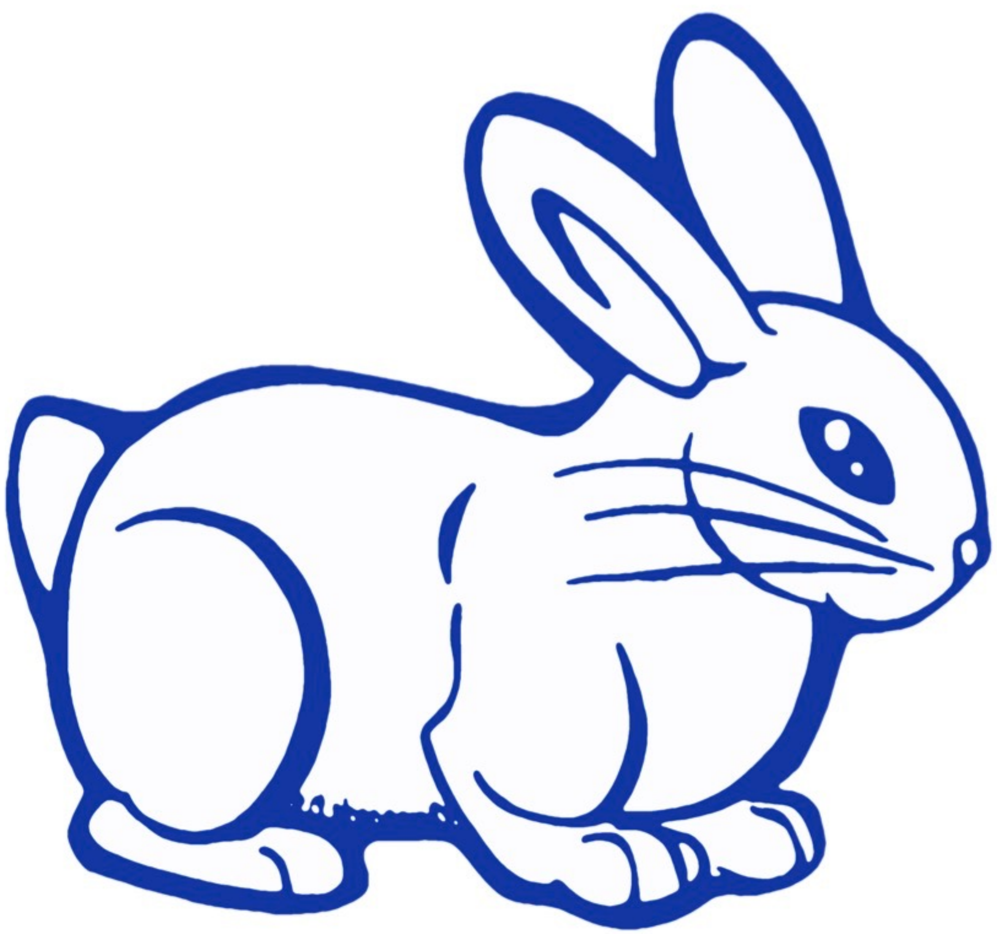


CMS FastSim overview

Sezen Sekmen (KNU)
for the CMS FastSim team

GEANTV Workshop
26-27 / 5 / 2016, CERN

Introduction

Fast simulation (FastSim) is an integral part of CMS physics studies.

- Speeds up CMS event simulation ~100 times and CMS event simulation+reconstruction ~20 times.
- Actively maintained by ~20 developers working on different aspects of the framework.
- Used mostly for large SUSY model scans. Also used for systematic studies by the top group, and in some cases, for signals by the exotica group. Further usage in private productions for MSc/PhD theses.
- Regularly validated within the official CMS software release validation framework.
- Mainly validated against FullSim. Reproduces FullSim mostly by <~10%.

3 ways to simulate in CMS

CMS FullSim

- detailed geometry
- particles tracked in small steps
- detailed material interaction model (mostly Geant4)
- detailed emulation of detector electronics and trigger
- standard event reconstruction

- O(100s) per ttbar event

CMS FastSim

- simplified geometry
- infinitely thin material layers
- simple analytical material interaction models
- detailed emulation of detector electronics and trigger, with exceptions
- standard event reconstruction, with exceptions

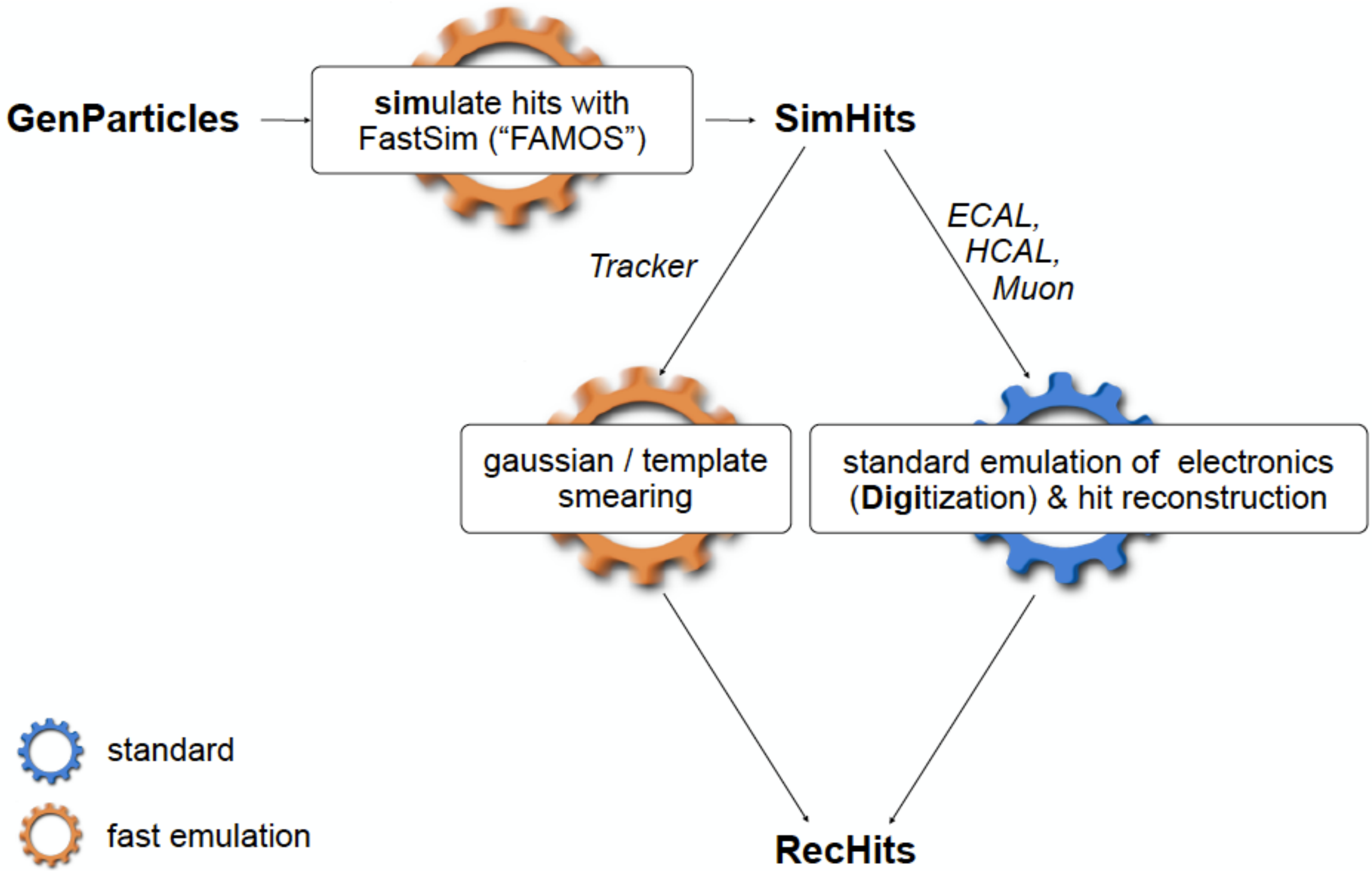
- O(5s) per ttbar event

Delphes

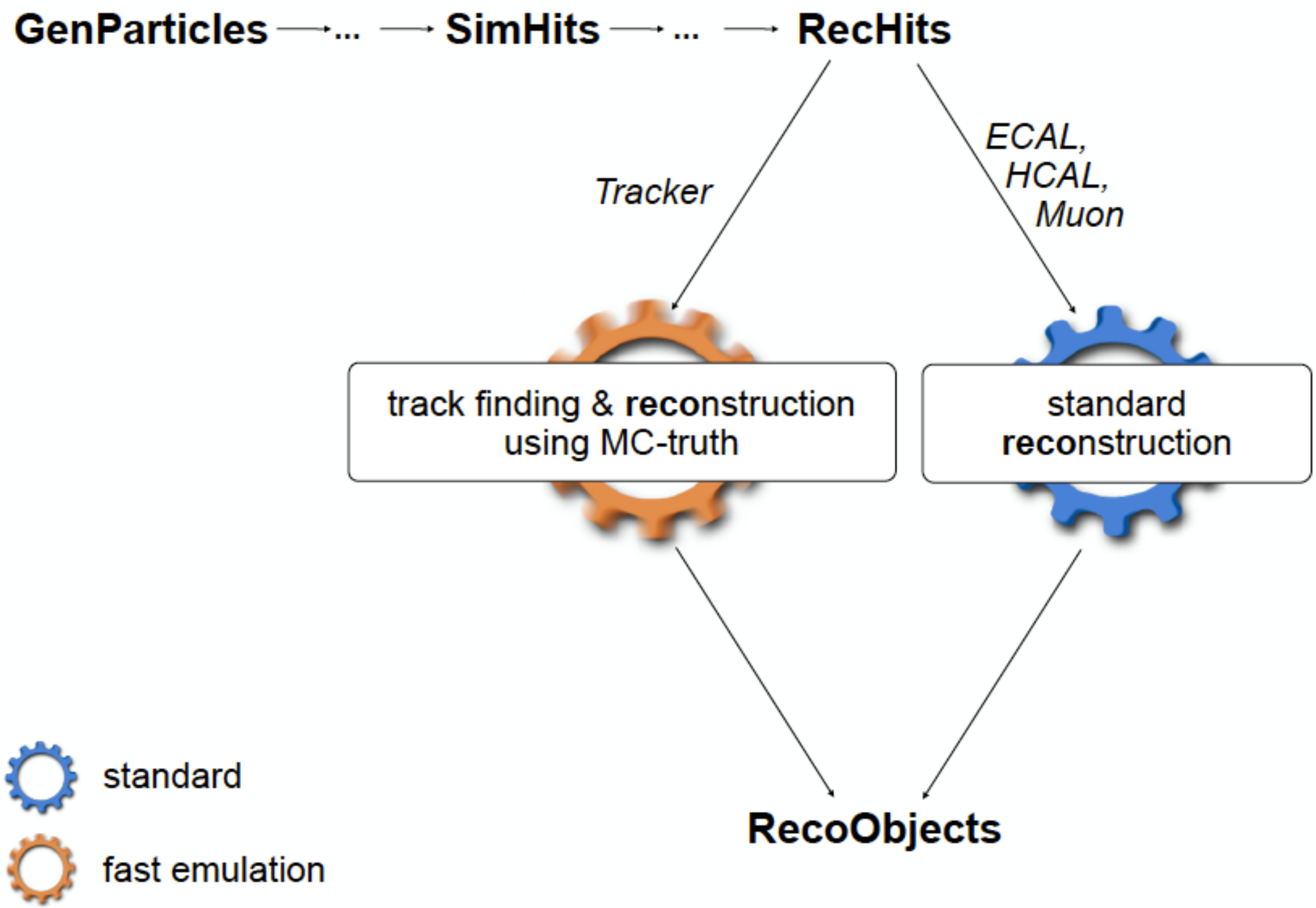
- (almost) simple 4-vector smearing

- O(.01s) per ttbar event

FastSim at a glance - I

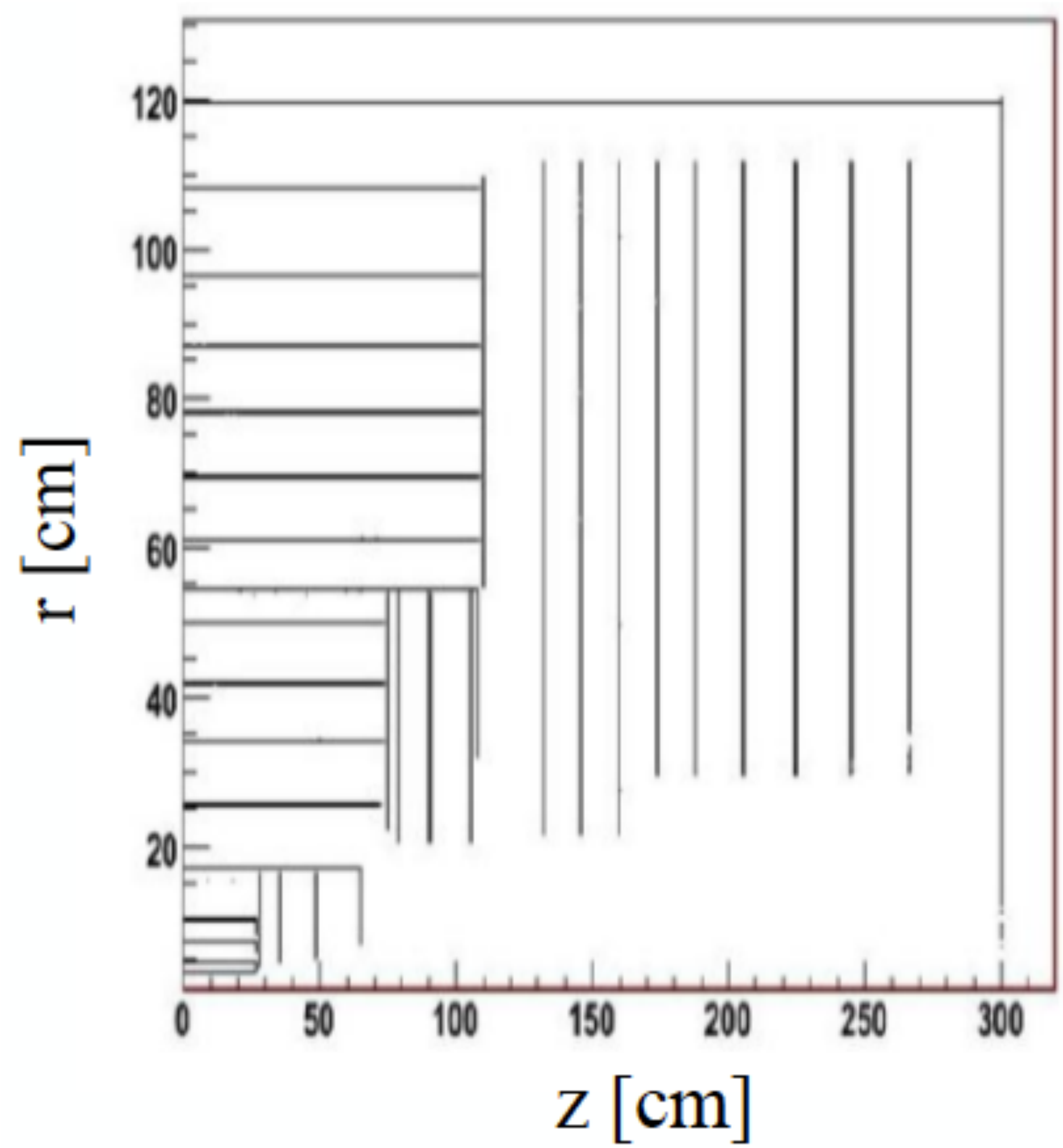


FastSim at a glance - II



Tracker: Tracker geometry

- Tracker material is modelled with 30 cylinders.
- Cylinder layers have zero spatial thickness. Material resides on the surface.
- Material emulation is done by assigning thickness in radiation lengths / interaction lengths to parts of cylinders.
- Some layers modelled as instrumented: sensitive
- Other layers model dead material: insensitive



Tracker: Particle propagation

- Particles are propagated using a simplified B field map.
- Only the B field on the cylinders is stored (binned in η).
- Simplified propagation: particles are propagated from one layer to the next layer assuming
  - the B field at the position at the first layer
  - there is no B field between the layers —> particle follows a simple helix.
- Particles may decay during propagation (using Pythia8)
  - particles are not allowed to decay outside the tracker volume
- Particle-material interactions occur when particles cross a layer:
  - Energy / direction changes
  - Particles might disappear
  - New particles might be produced

Tracker: Considered material interactions

Charged particles:

- Energy loss by ionization
- Multiple Coulomb scattering

Electrons:

- Bremstrahlung

Photons:

- e+/e- conversion

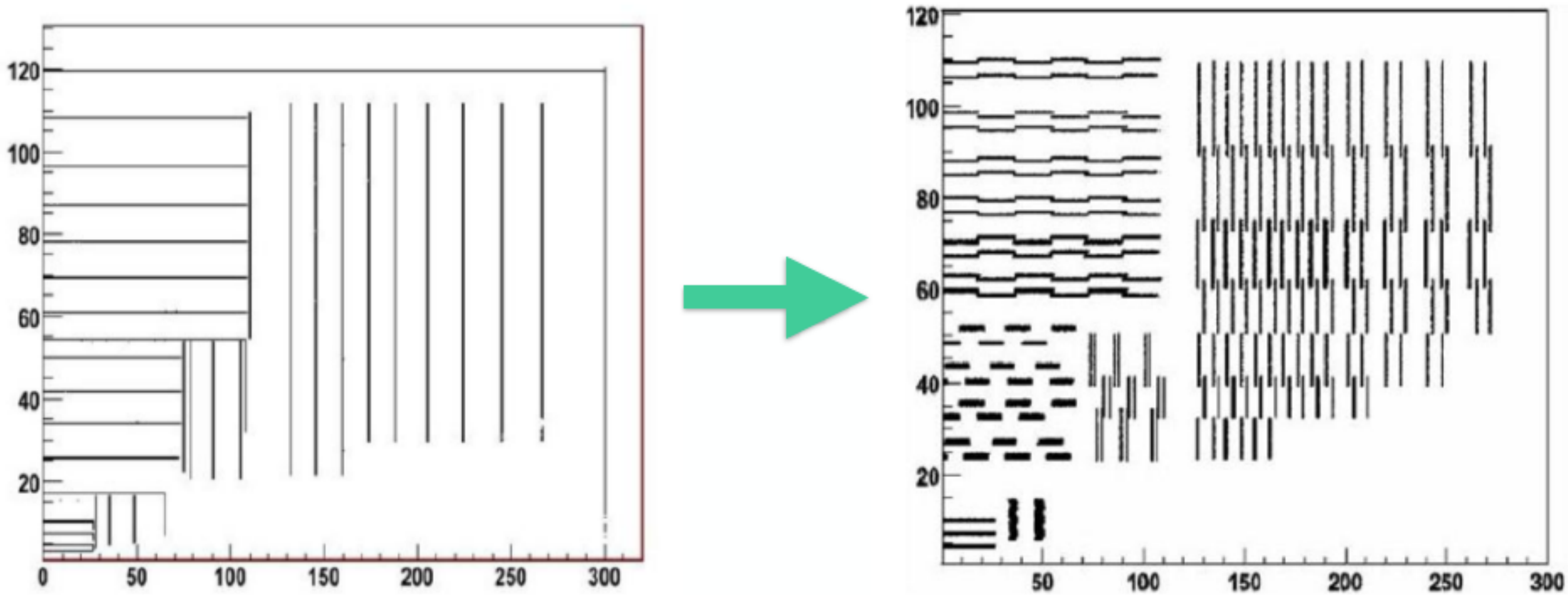
Hadrons:

- Elastic nuclear interactions
- Inelastic nuclear interactions

Simple, approximate formulae taken from PDG (applies to all interactions above except inelastic nuclear interactions)

Library of pre-generated nuclear interactions (applies to inelastic nuclear interactions)

Tracker: SimHits



- Now consider the full geometry of tracker modules,
- Find crossings between particles and modules
  —> Calculate entry and exit points
  —> SimHits

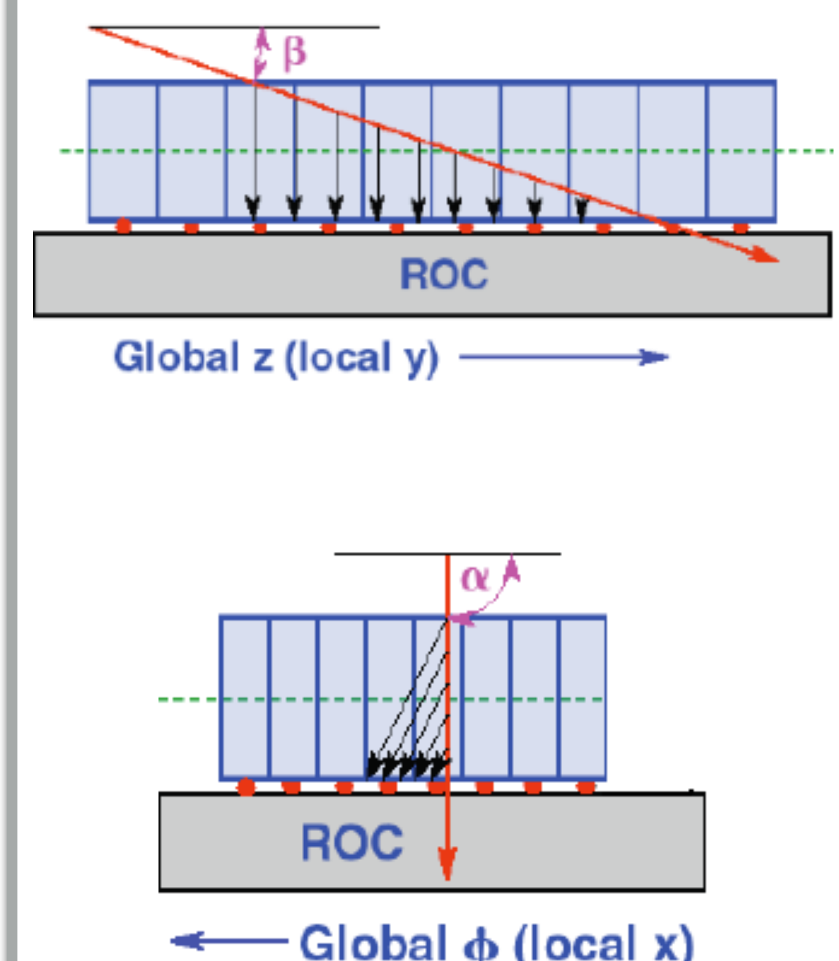
Tracker: Emulating RecHits: Pixel RecHits

RecHit position: Smear SimHit positions based on templates obtained from a detailed pixel simulation (PIXELAV). Templates depend on:

- dimension: X / Y (local coordinates)
- pixel type: normal / edge / big
- incident angle: cotα and cotη
- number of pixels that are hit

RecHit error: Extract from pixel templates used in standard pixel hit reconstruction.

per module
local coordinates



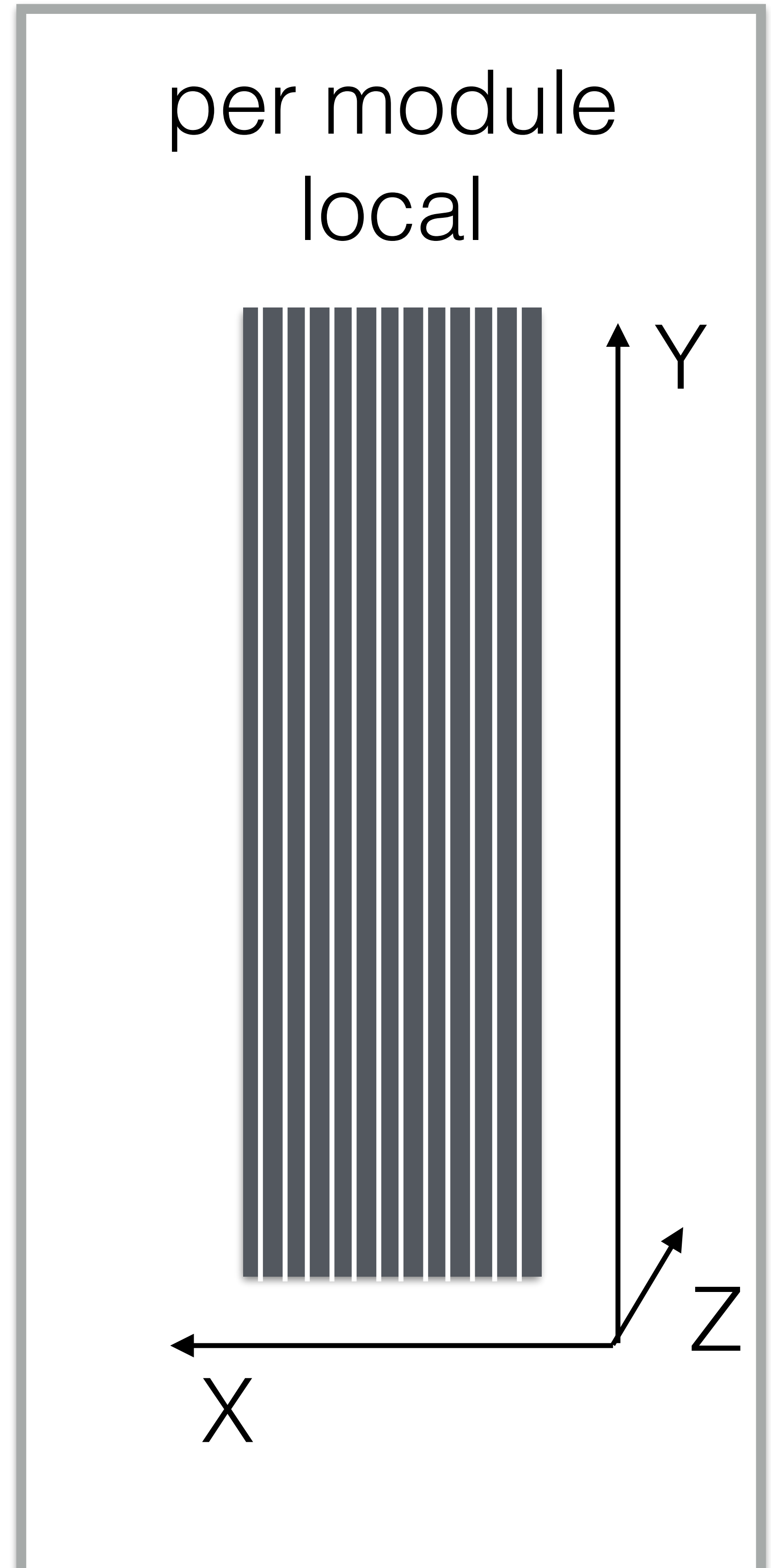
Tracker: Emulating RecHits: Strips RecHits

RecHit position:

- X: Gaussian smearing
  <— width obtained from FullSim
- Y: taken as the strip center

RecHit error:

- X: width of the Gaussian
- strip length / sqrt(12)



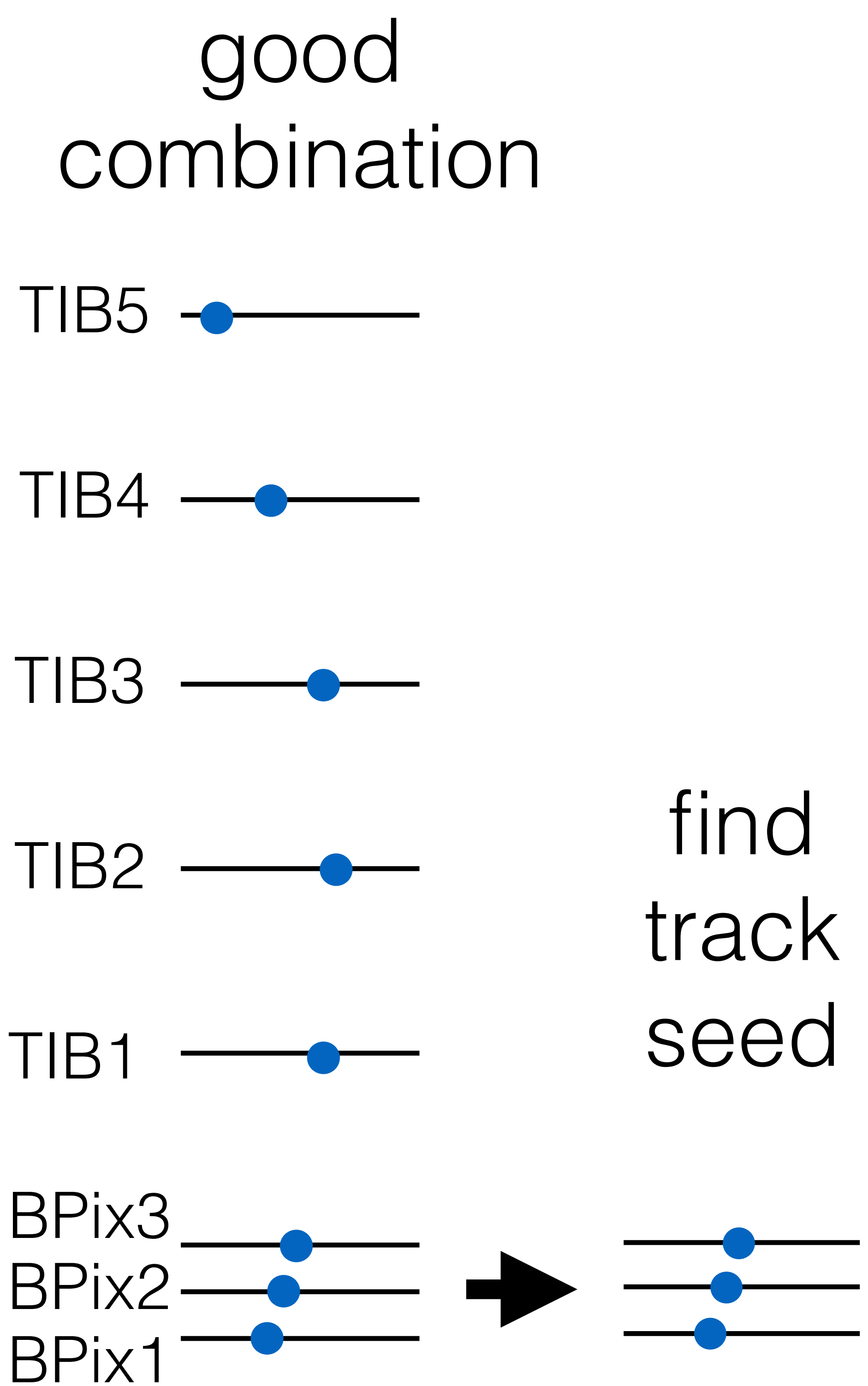
Tracker: Track RECO: Overview

- Track seeding:
  Find small combinations of (2-3) hits compatible with
  - allowed hit pattern (e.g. [BPix1, BPix2, BPix3])
  - compatibility with the beamspot / vertex
  - p_T threshold
- Track finding:
  For each seed, walk through the tracker to find compatible hits. The walk is constrained in many ways to avoid wasting time on fake combinations.
- Track fit:
  Fit through all the hits of the track to obtain a precise estimate of track momentum and vertex position.
- Iterative tracking:
  Hits used in one iteration are hidden from use in the next iteration.

Tracker: Track RECO emulation

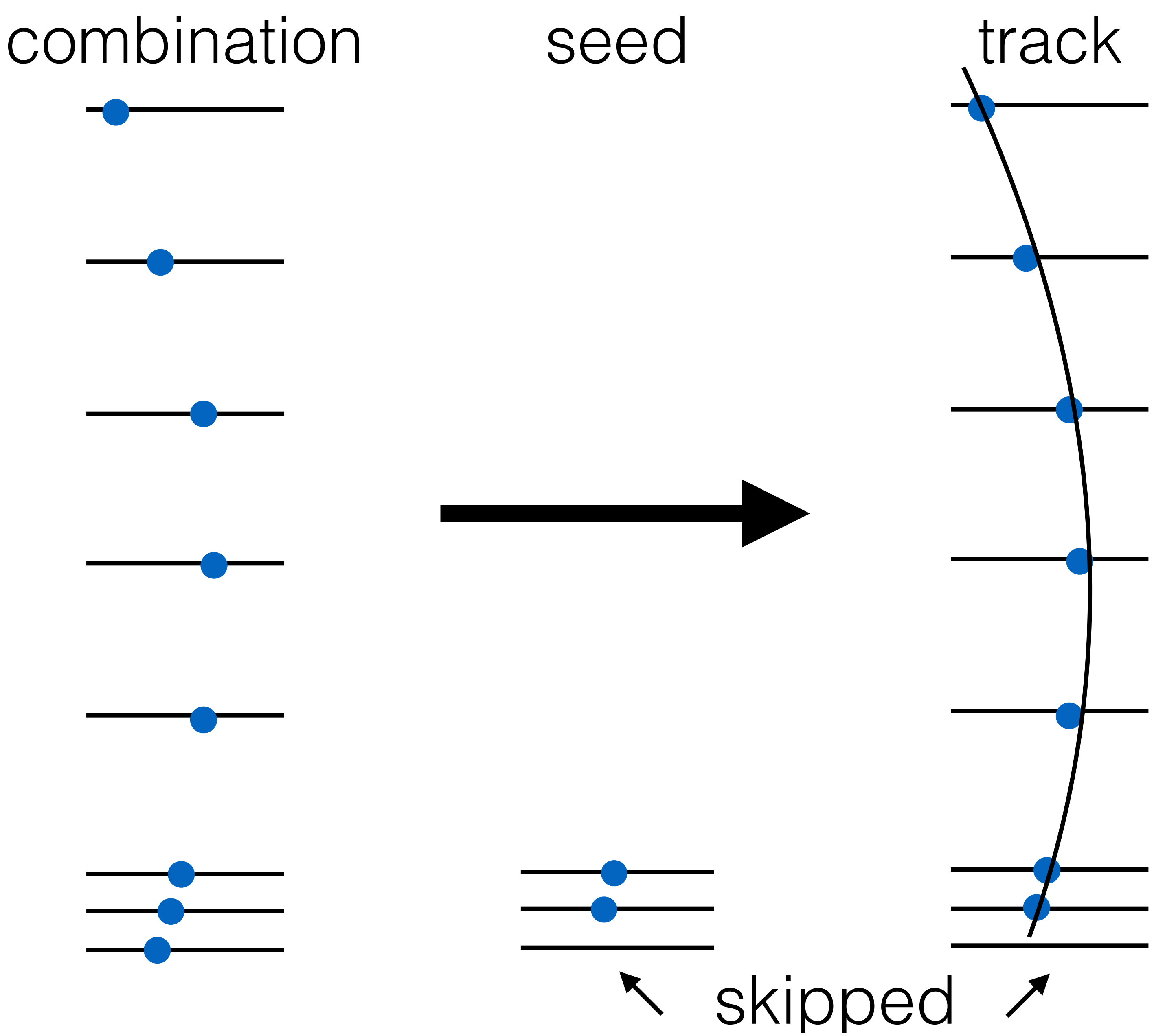
- Find good hit combinations:
  These are the hits belonging to the same simulated particle
- Track seeding:
  Emulate seeds, and reject good hit combinations for which no track seed can be reconstructed
- Track fit:
  standard track fit followed by standard track ID
- Iterative tracking:
  Standard iterative tracking except for a few missing iterations (jet core and muon seeded iterations)

Tracker: Track seed emulation



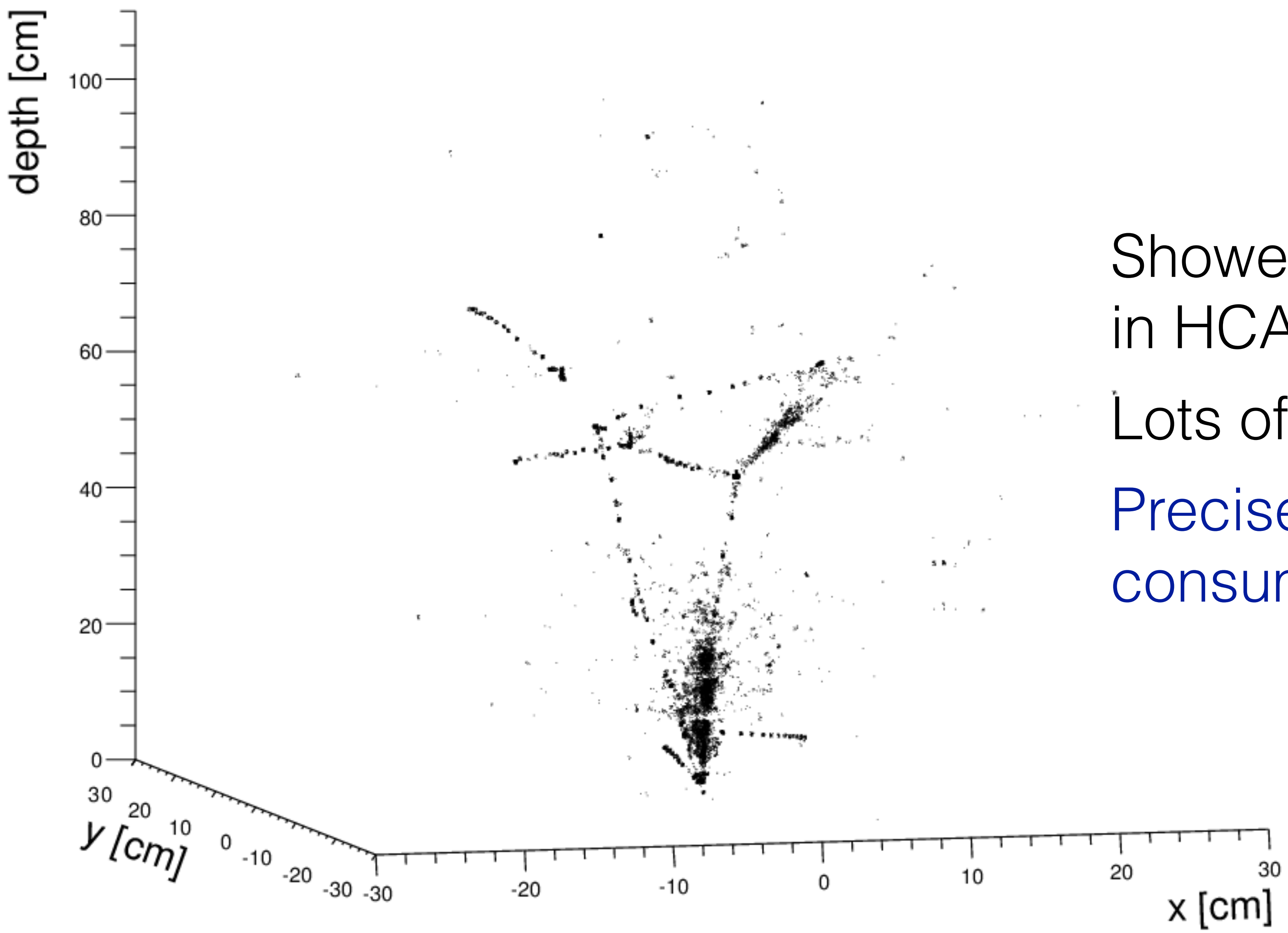
- compatibility with allowed hit pattern e.g. [BPix1, BPix2, BPix3]
- p_T / vertex constraints on inner 2 hits (uses standard track seed code)
- p_T / vertex constraints on 3 hits

Tracker: Track seed hit skipping



Sometimes the track seed skips hits. The resulting track is also made to skip these hits.

Calorimeter: Calo showers



Shower by a 20 GeV pion in HCAL by GEANT4

Lots of fluctuations.

Precise simulation is time consuming.

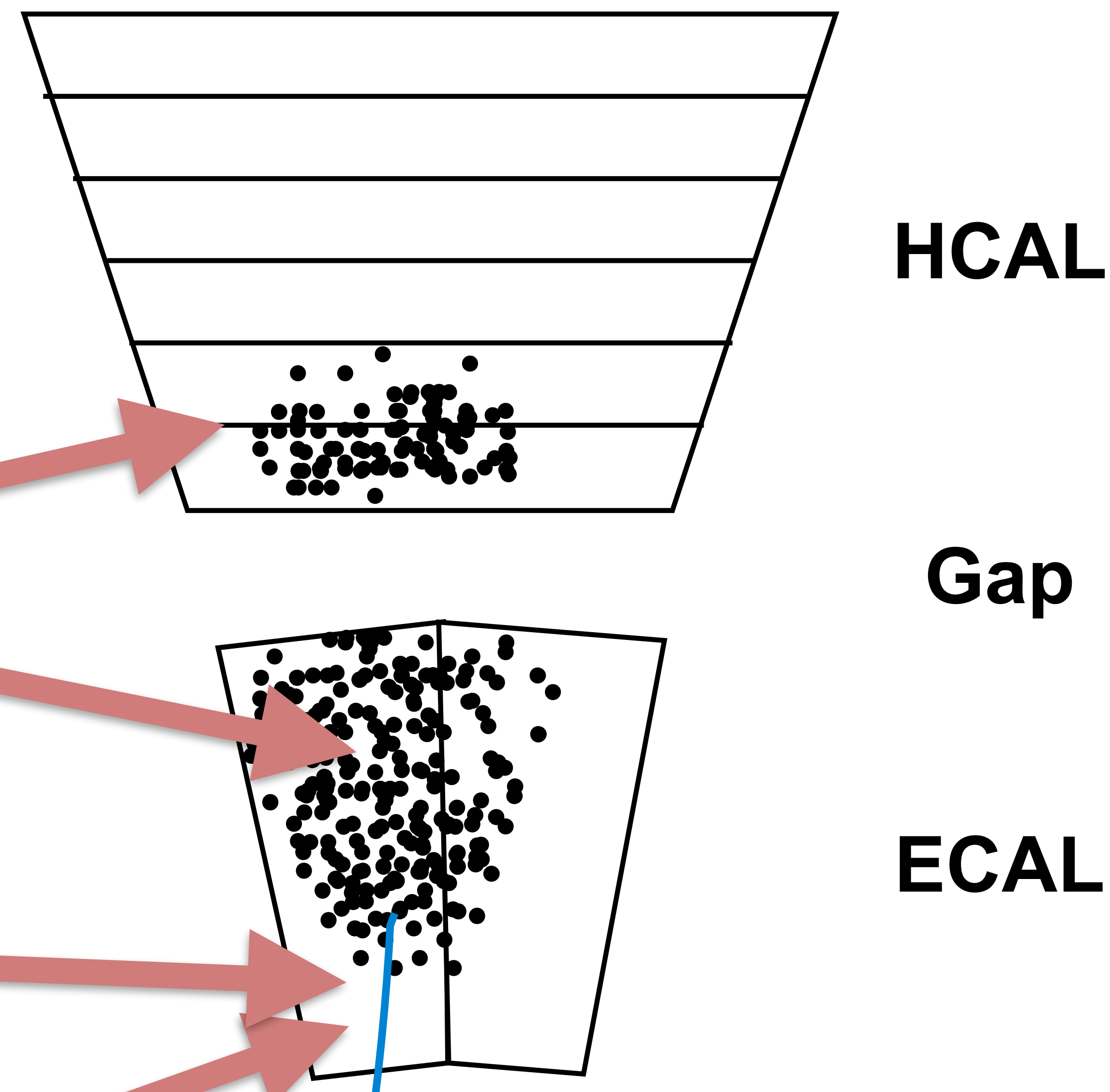
Calorimeter: Overview

GAP ignored

Shower modeled with "GFLASH"

staring point shower modeled based on radiation / interaction lenght

no energy loss before shower



photons: simulate conversion then simulate showers of e-,e+

Calorimeter: Shower simulation with GFLASH

GFLASH is a program for the fast simulation of electromagnetic and hadronic showers using parameterizations for the longitudinal and lateral shower profiles. It also takes into account fluctuations and correlations of the parameters in a consistent way. (Grindhammer, et. al., 1990)

This approach can be used for a realistic simulation of calorimetry showers, since the calorimeters do not see actual particles, but rather see energy blobs.

- Draw energy depositions from a cylindrically symmetric energy density function (“shower shape”).
- Shower shapes fluctuate largely from shower to shower.

Calorimeter: Longitudinal shower shape

Electrons:

1 gamma distribution

$$\frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$$

Hadrons:

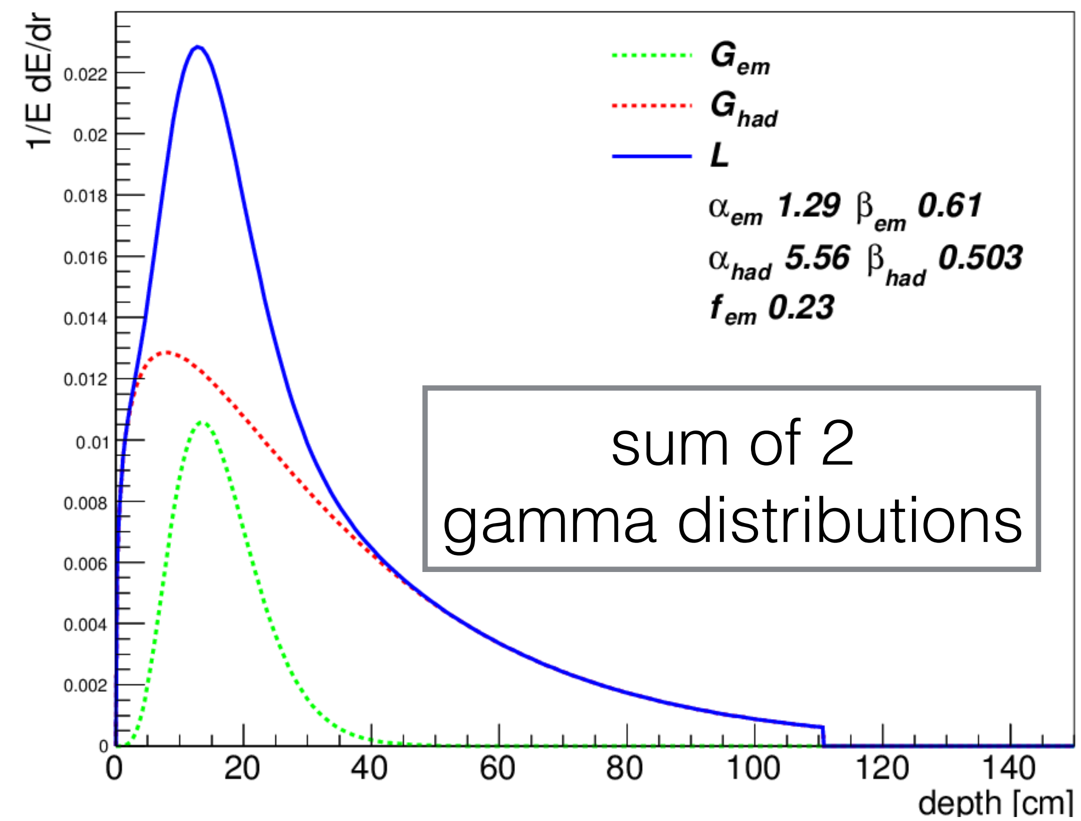
3 classes of showers:

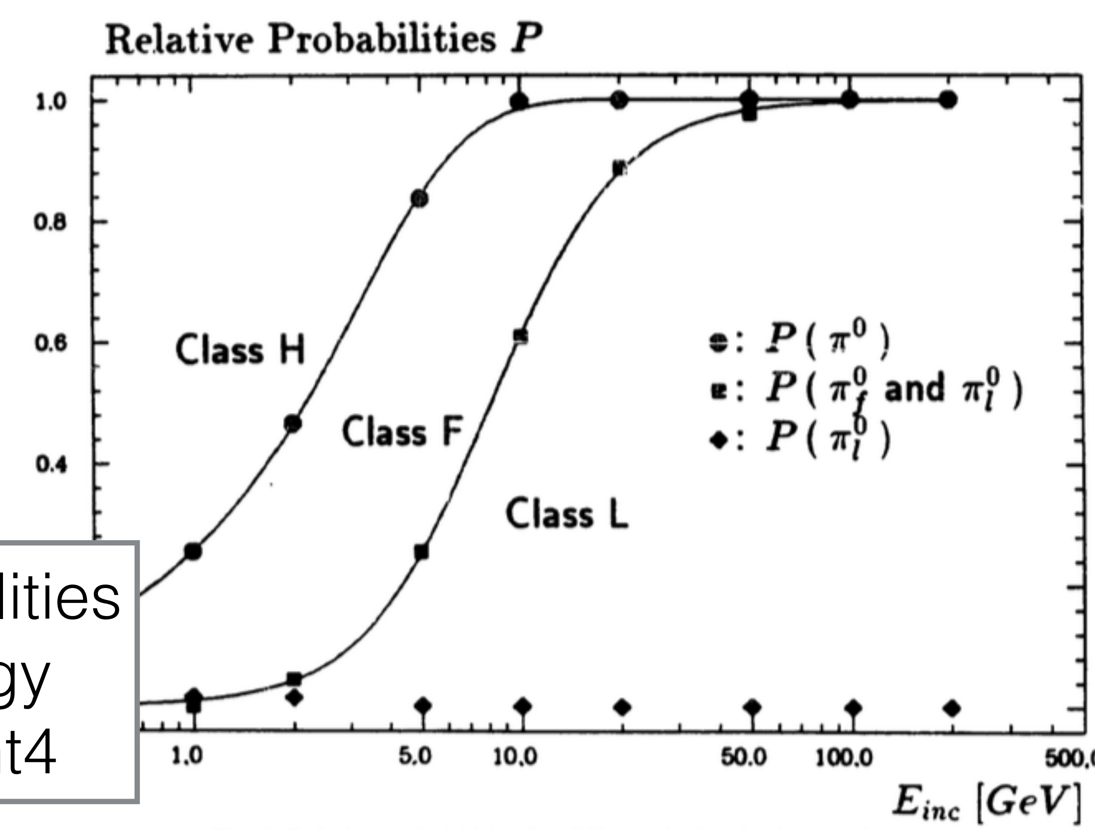
- no π^0
- early π^0
- early and late π^0

Sum of the 3 gamma distributions.

Note that π^0 -> γγ

—> EM subshowers





shower class probabilities as function of energy obtained from geant4

Calorimeter: Longitudinal shower shape - II

The longitudinal shower shape can be described by the GFLASH parametrization as follows:

$$dE_{dp} = f_{dp}\, E_{inc} \left[c_h\, H(x)dx + c_f\, F(y)dy + c_l\, L(z)dz \right]$$

- c_h: Fraction of deposited energy carried by hadrons
- c_f: Fraction of deposited energy carried by π^0 produced in first inelastic interaction
- c_l: Fraction of deposited energy carried by π^0 produced in late inelastic interactions

$$H(x) = \frac{x^{\alpha_h - 1}\, e^{-x}}{\Gamma(\alpha_h)} \qquad x = \beta_h s_h [\lambda_0]$$

$$F(y) = \frac{y^{\alpha_f - 1}\, e^{-y}}{\Gamma(\alpha_f)} \qquad y = \beta_f s_f [\chi_0]$$

$$L(Z) = \frac{z^{\alpha_l - 1}\, e^{-z}}{\Gamma(\alpha_l)} \qquad z = \beta_l s_l [\lambda_0]$$

$$E_{dp} = f_{dp}\, E_{inc}$$

Calorimeter: Longitudinal shower shape

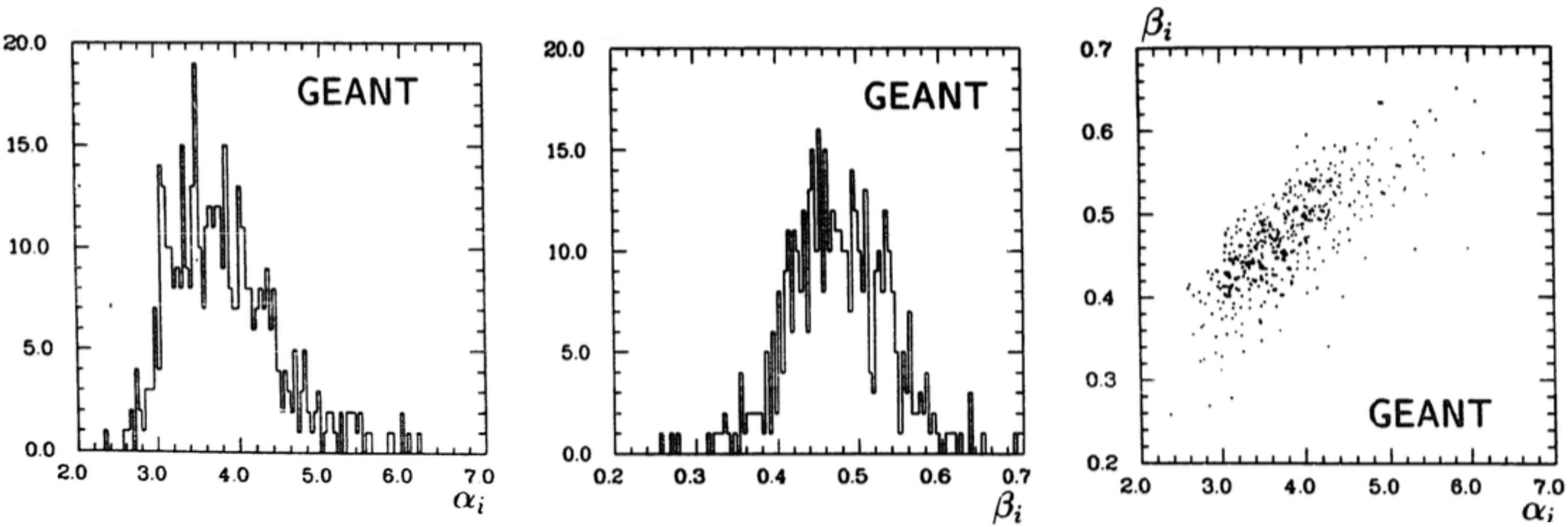
Many parameters:

α and β parameters of Gamma distributions

Relative amplitude of Gamma distributions

Parameters behave ~Gaussian:

—> Sample random shower shape parameter values from a multi-dimensional Gaussian model



Calorimeter: Transverse shower shape

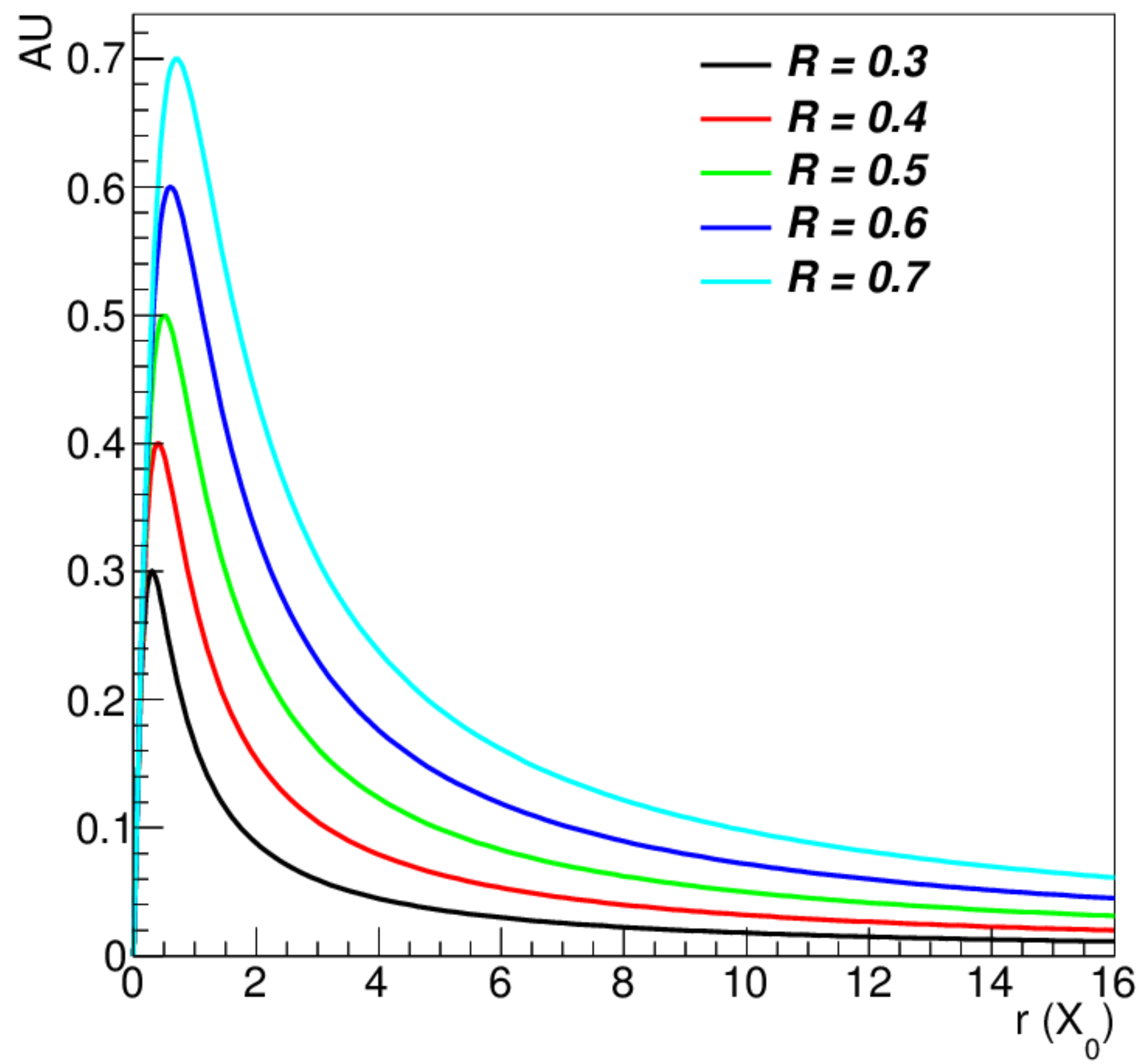
Hadrons:

$$f(r) = \frac{2rR_{50}^2}{(r^2 + R_{50}^2)^2},$$

$$\langle R_{50}(E, z)\rangle = [R_1 + (R_2 - R_3 \ln E)z]^n,$$

$$V_{R_{50}}(E, z) = [(S_1 - S_2 \ln E)(S_3 + S_4 z)\langle R_{50}(E, z)\rangle]^2.$$

Parameters obtained from GEANT.



Electrons:

Similar but more complicated.

Calorimeter: Energy response

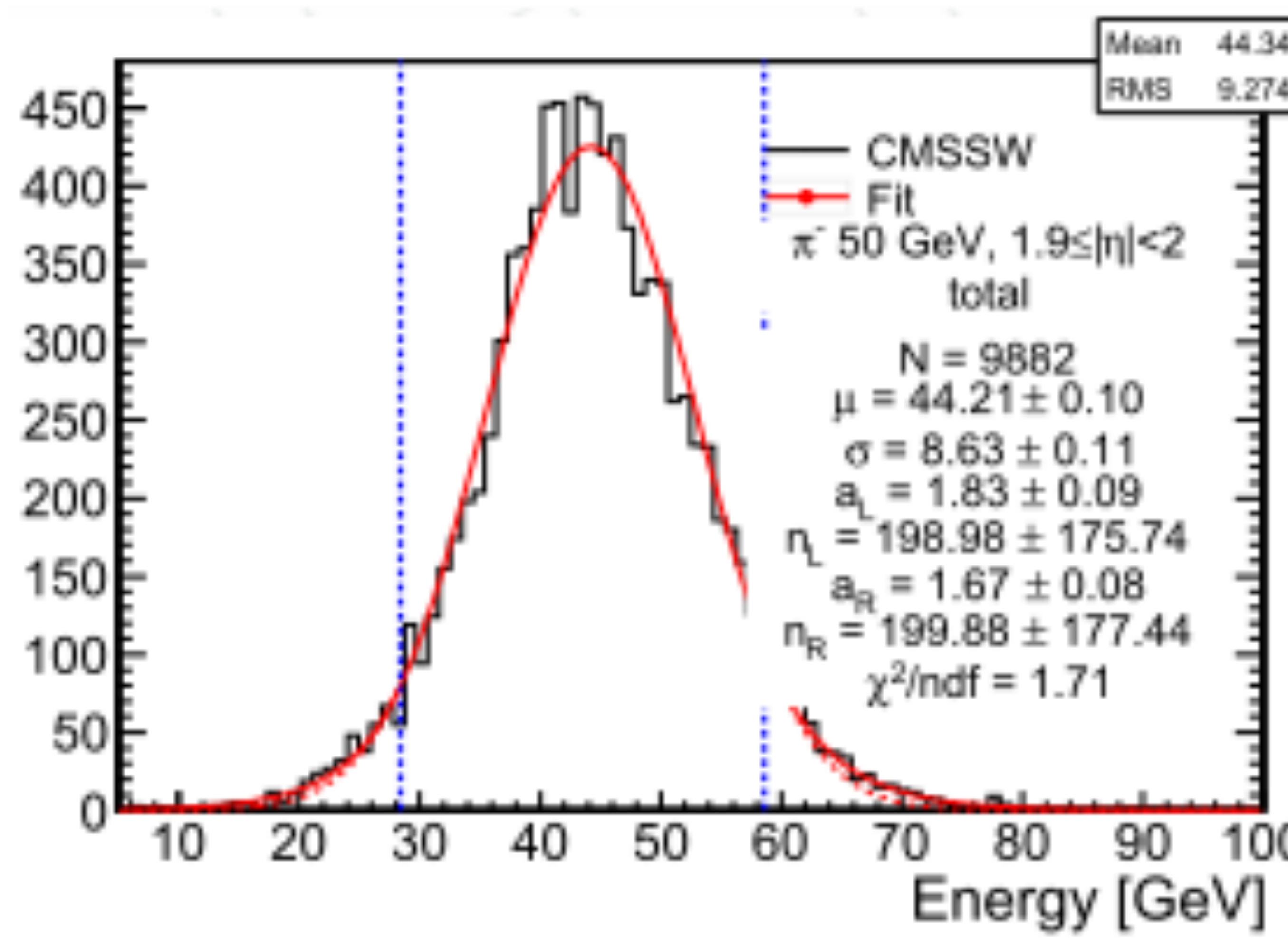
Each energy spot is scaled with a scale factor such that energy sum of all spots equals the number drawn from a function.

Hadrons:

Crystall-Ball function tuned in small energy and eta steps in FullSim.

Electrons:

gamma distribution, tuned on GEANT as function of energy, combined with a model for signal losses through inter crystal gaps in ECAL



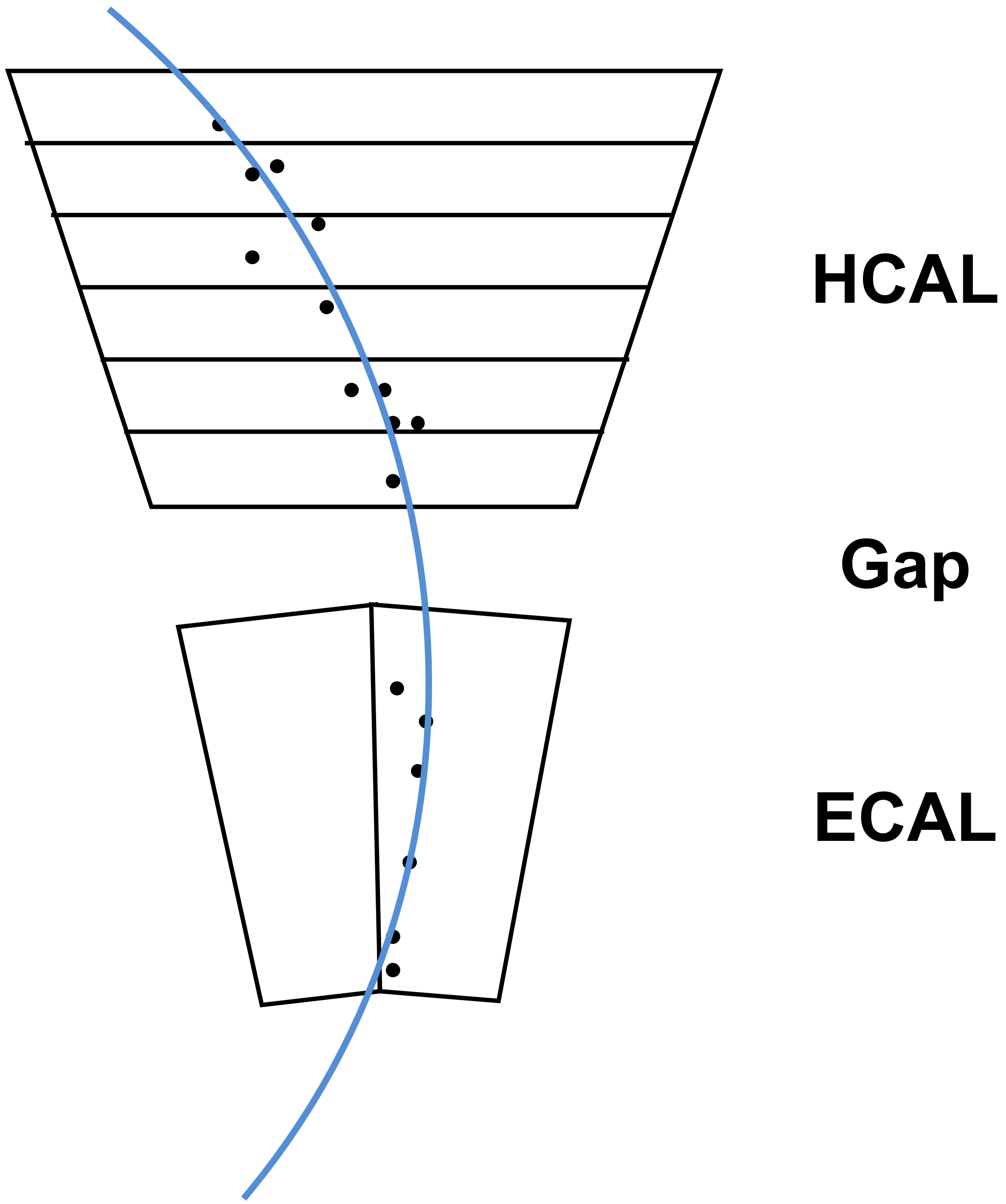
AN-14-187, Kevin Pedro et al.

Calorimeter: Hadron forward calorimeter

- We use a shower library previously simulated using GEANT4 to simulate the hadron forward calorimeter.
- Showers are classified according to particle type, η and energy.
- Material in front of the HF is not modelled. We apply FastSim-specific correction factors to correct the incomplete modelling.

Calorimeter: Muons

- Muon energy loss in calorimeters is through ionization.
- Gap between the ECAL and the HCAL is ignored.



Calorimeter: Energy spots, SimHits, RecHits

- GFLASH mechanism generates energy spots within a shower. These are then converted into SimHits.
- Calorimeter SimHits in FastSim have the same definition and structure as the SimHits in FullSim.
- Therefore we use standard emulation of electronics (i.e. digitization).

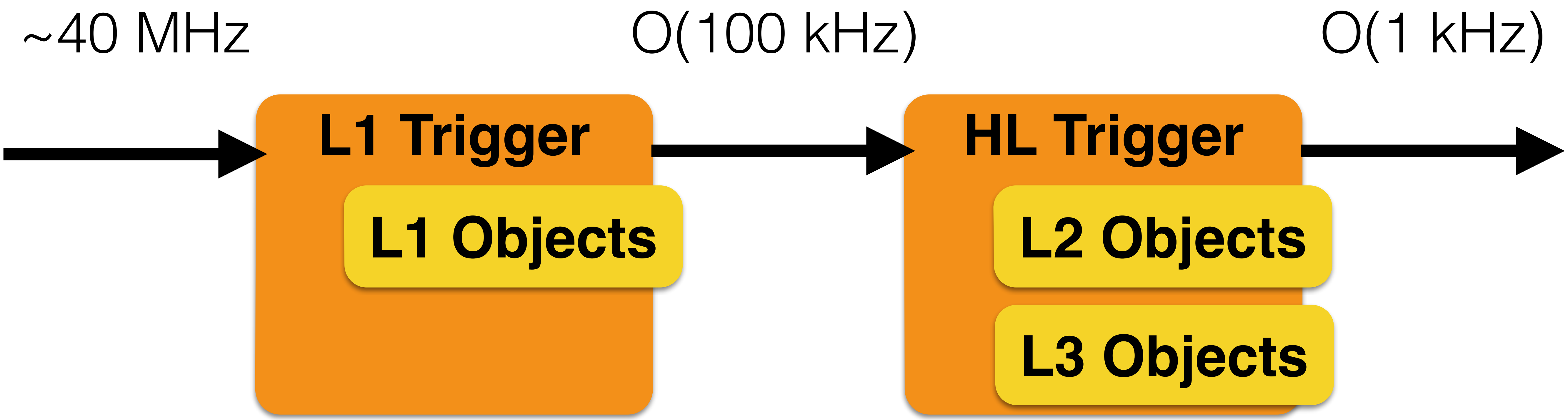
Muon system

- After traversing the tracker, muons are propagated through a rather realistic geometry including ECAL, HCAL, solenoid and iron yoke. Material in muon detectors are ignored, and propagation machinery used for muon reconstruction is employed.
- Mean energy loss through ionization is calculated and taken into account during propagation through material.
- Coulomb scattering and energy loss through ionization are simulated in big, discrete steps. Each step models effects between one ring / disk of detectors and the next (as for the tracker).

Muon system: Muon SimHits

- Interaction points between the muon path and all individual detector layers inside the muon chambers are calculated and are converted into SimHits.
- SimHits in FastSim have the same structure as those in FullSim.
—> They can be fed into the standard emulation of electronics and then into standard trigger / reconstruction.

Trigger in CMS



Trigger: L1 emulation

- Standard L1 emulation in FullSim uses as input “digis” (i.e. emulated electronic signals) from calorimeters and the muon system (and not from the tracker)
- FastSim uses this standard emulation, since it mostly uses the standard reconstruction.
- Exception: some technical triggers are skipped because they rely on detectors not included in FastSim.

Trigger: HLT

- HLT in FastSim tends to use the very same algorithms and menus as those used with real data and in FullSim.
- Exception: tracking in the tracker system
- Standard HLT has several flavors of tracking targeting specific HLT paths.
- FastSim HLT: one “general purpose” track collection: the standard track collection.

Recent developments and future plans

Recently:

- Redesigned the tracking RecHit producer in a more modular fashion
- Made track reconstruction more consistent with FullSim.
- Improving the simulation of hadron showers, with a better modelling of shower start, shower shape, shower sampling, etc. Also redesigning the tuning procedure.
- …

Ongoing and near future:

- Introduce the new upgrade pixel detector in FastSim.
- Develop a package for validation and tuning of material interactions in the tracker.
- Continue developing the hadronic shower simulation.
- Improve the material description in front of the HF.