Fast simulation(s) at LHCb

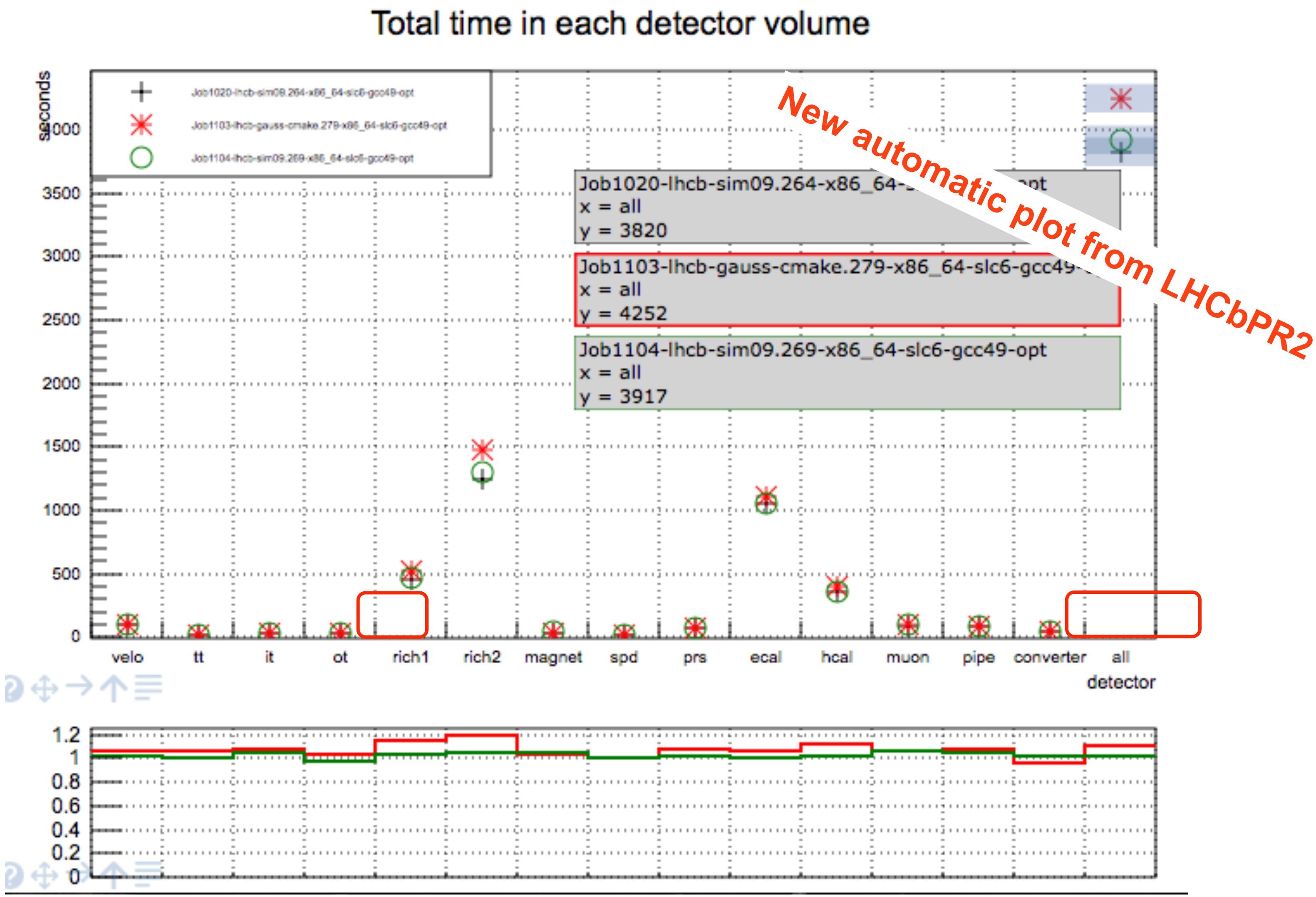
Riccardo Cenci[1], Gloria Corti[2], Matteo Rama[3]
[1]INFN and SNS Pisa [2]CERN [3]INFN Pisa

LHCb computing workshop, 16 May 2017

Strategies reminder

- Two general approaches:
  - ***Simulate less detector***
    - Simulate partial detector by turning off or fast-simulating components with large contribution e.g. RICHLess, Calo with shower library
  - ***Simulate less particles***
    - Particle gun: simulate only particles from signal decay
    - Re-use of the underlying event: simulate one underlying event for N signal events
  - … Or any combination of them

Where CPU time is spent in Gauss v49r6



Where CPU time is spent in Gauss v49r6

Existing fast simulation options

Simulating without RICH physics (Richless)

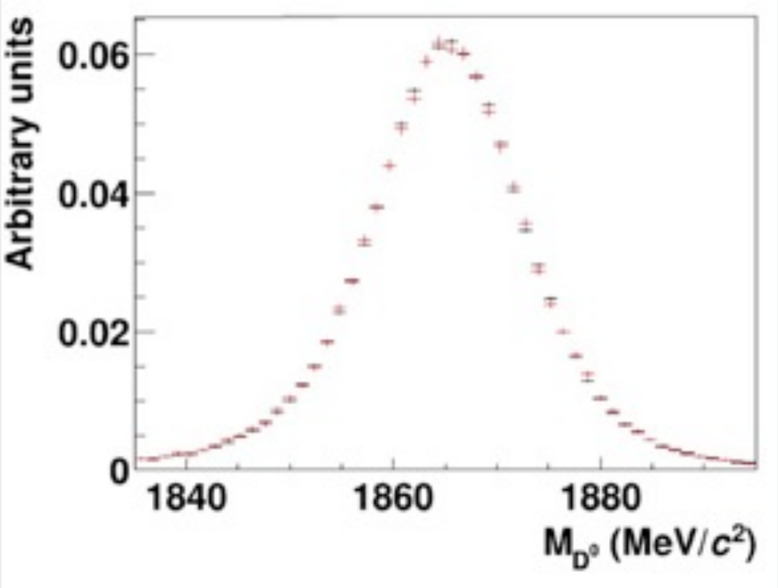
- PID in simulation not needed in all analyses
  - Switch off optical γ simulation in RICH
  - RICH material still fully simulated
    → no impact on downstream detectors

✔ Saves O(40-50%) of simulation time
✔ Size of data slightly reduced

✗ No PID information in MC

- Model for 2012 already available and used for analysis
- Request for 2015-6 conditions, but trigger configuration is not trivial, under discussion

$B \to D^{*}\mu\nu X$



Validated on 0.5M sample of $B \to D^{*}\mu\nu X$ MC events
All relevant distributions in perfect agreement

Used in production to reduce large systematic uncertainty from MC for R(D)

G. Ciezarek, D. Muller

Optimization of RICH simulation

- In Gauss v49 RICH2 took ~3x CPU time than RICH1 (~15% and ~40% of overall event time, respectively). Why? In principle they should be similar.
- Problem investigated by Sajan Easo:
  - Origin identified in simulation of scintillation photons in RICH2
  - Implemented tricks which bring the RICH2 CPU time back to the level of RICH1
  - No expected degradation in accuracy
- A number of additional changes being applied to improve the RICH1/2 simulation speed without sacrificing accuracy. For examples:
  - Use max quantum efficiency in HPDs and max reflectivity of mirrors
  - Deactivate some step actions not needed for standard production

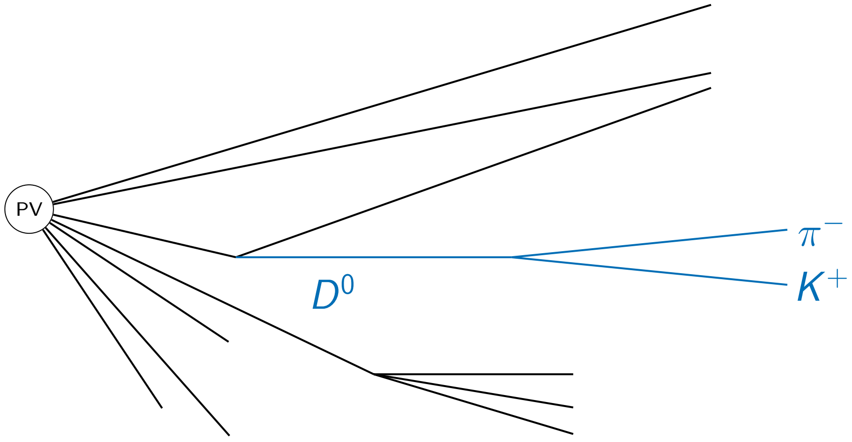
All together, these changes could bring to O(30%) reduction of overall event CPU time. To be released in v50r1 soon, then validated before being available in production

S. Easo

Reuse of the underlying event

D. Muller and M. Gersabeck

- Combine full simulation and particle gun approach
- Generate a full event but reuse it *N* times, each time replacing the signal decay
- Hadronisaton stays the same.
- Kinematics of the redecayed particle stay the same. Correct correlation with the underlying event.
- Same efficiencies and resolution as nominal simulation. Example use-cases: efficiencies in high dimensional amplitude analyses, templates for
- R (something). Large number of redecays: almost the same speed as particle gun.



For more details see <u>link</u>

✔ Substantial increase in speed, depending on number of redecays
✔ Complexity of full event

✗ Stat fluctuations must be handled with care
✗ No disk space saving

Reuse of the underlying event

D. Muller and M. Gersabeck

- Main guideline: fully integrated in Gauss, i.e. use mostly existing algorithms and services and does not break Gauss
- Option to redecay signal only (faster) or anything at least as heavy (slower, but deals with multiple true candidates)
- One service added: GaussRedecay

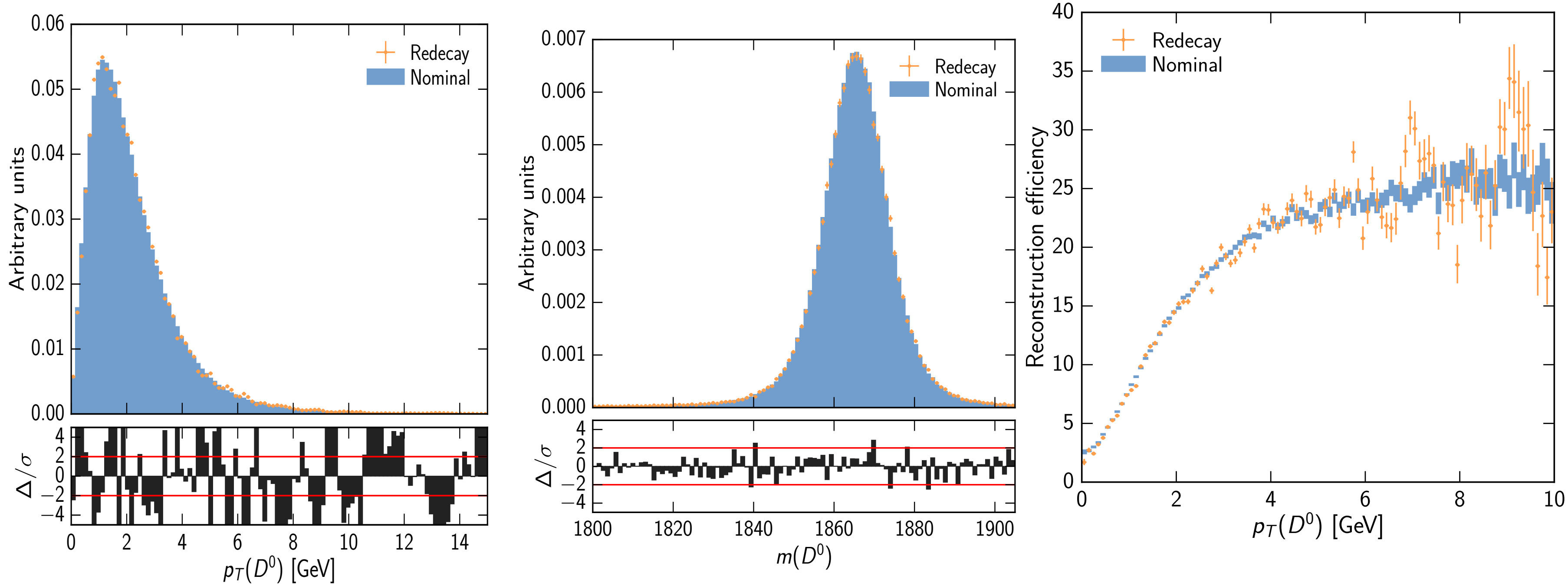
Usage: add this do your options files

```
from Configurables import Gauss
Gauss().Redecay['active'] = True
Gauss().Redecay['N'] = 100  # 100 is the default.
Gauss().Redecay['rd_mode'] = 0  # Redecay signal instead of everything heavier.
```

- Two step bootstrap to avoid autocorrelation:
  - Generate original events, sample with replacement sets of events with same original event
  - Redecay the original event, sample with replacement events from each drawn set.

Reuse of the underlying event

$D^{*+} \to D^0\pi^+, D^0 \to K^-\pi^+$ Sim09a, 5 TeV, spillover, Truth-matched candidates

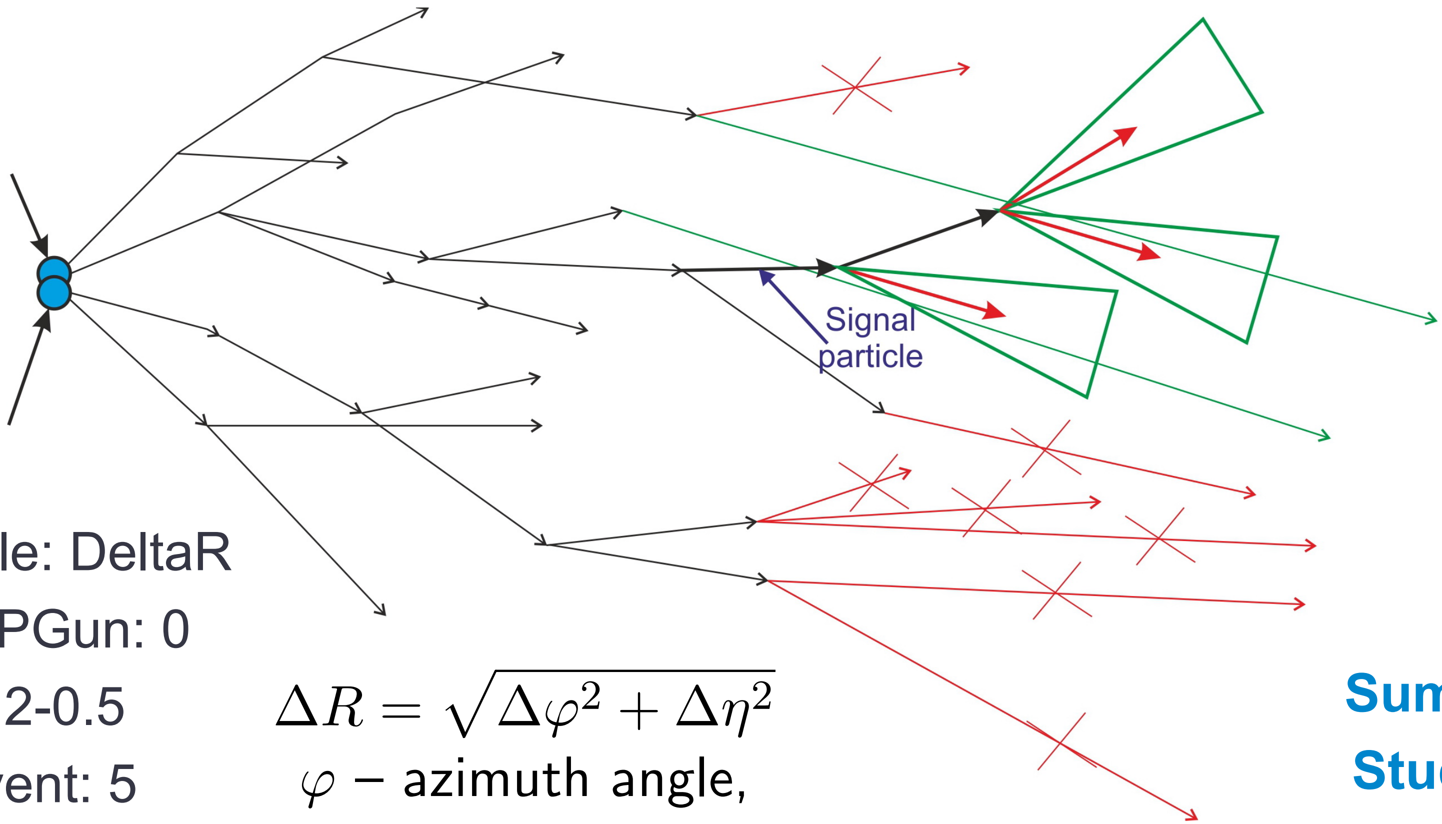


- Already available in production
- Production for final test pending in the request queue, to be used also for an analysis
- More requests already planned after the validation

D. Muller and M. Gersabeck

Extended Signal Particle Gun

- Goal: generate the whole events, but kill selectively all the particles apart from signal product and stable “neighbors” (configurable cuts)
-



- Example: DeltaR
- Signal PGun: 0
- Jets: 0.2-0.5
- One event: 5
- Full event: 42

$$\Delta R = \sqrt{\Delta\varphi^2 + \Delta\eta^2}$$

φ – azimuth angle,

η – pseudorapidity

Summer Student

L. Olifer, D. Muller, M. Gersabeck

Extended Signal Particle Gun

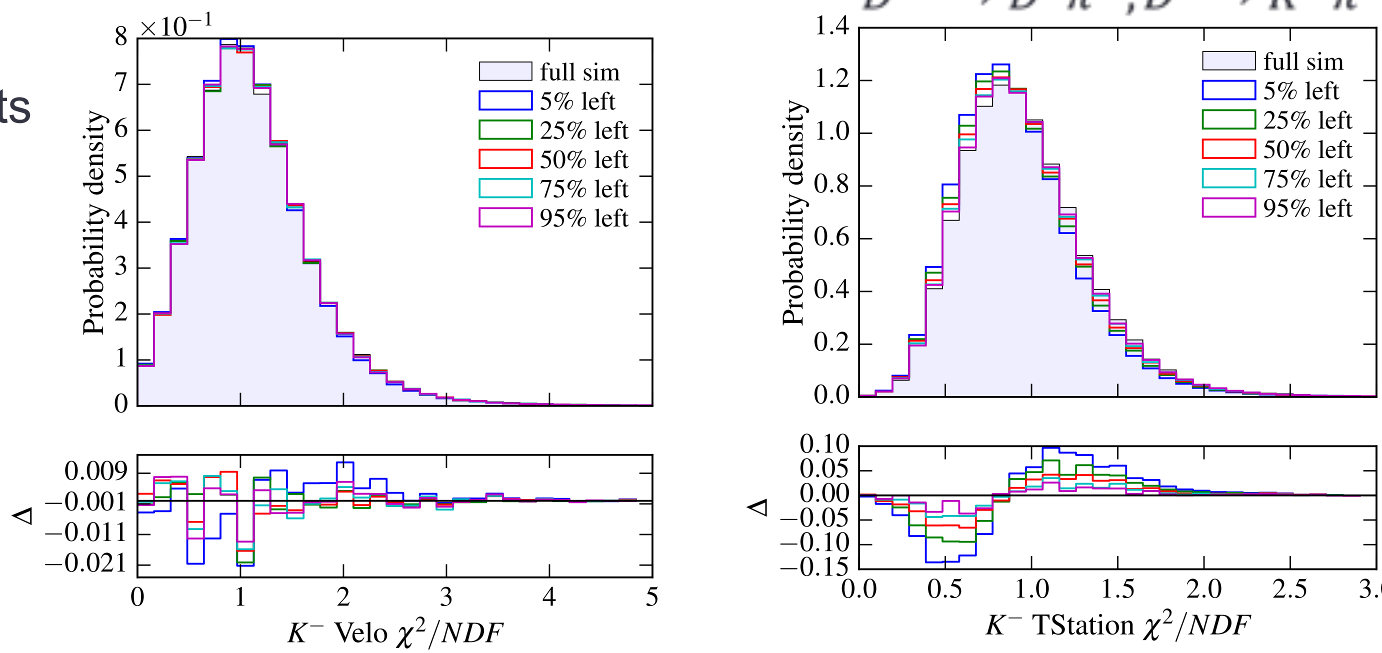
Summer Student

L. Olifer, D. Muller, M. Gersabeck

- Signal products analyzed down to T-stations to find tracks to be kept
- Other particles can be also rejected before the magnet kick based on acceptance and momentum

Possible cuts based on cumulative distribution tracks to be kept

$D^{*+} \rightarrow D^0\pi^+, D^0 \rightarrow K^-\pi^+$



✔ Good description of kinematics
✔ Fine tuning based on analysis requirements

✗ Missing final performance analysis
✗ Currently on hold for lack of manpower

New Options under development

Fast simulation of the Calo system

- Different possible approaches, picked shower libraries to simulate the calorimeter hits
  - Faster than Geant4 and accurate (after proper tuning)
  - Potential issue with punch-through to muon detector not simulated
  - Q: How fast can it be?
- Multiple solutions available, but need to satisfy some requirements
  - Use of common tool to kill electrons and photons at the entrance of the calorimeter and add the energy deposits in each cell
  - Able to be run in a centralized environment
  - Gain at least a factor 10 for simulating the calorimeter response wrt the full simulation
  - Performances within 10 of full simulation, to be useful for most of the analysis.

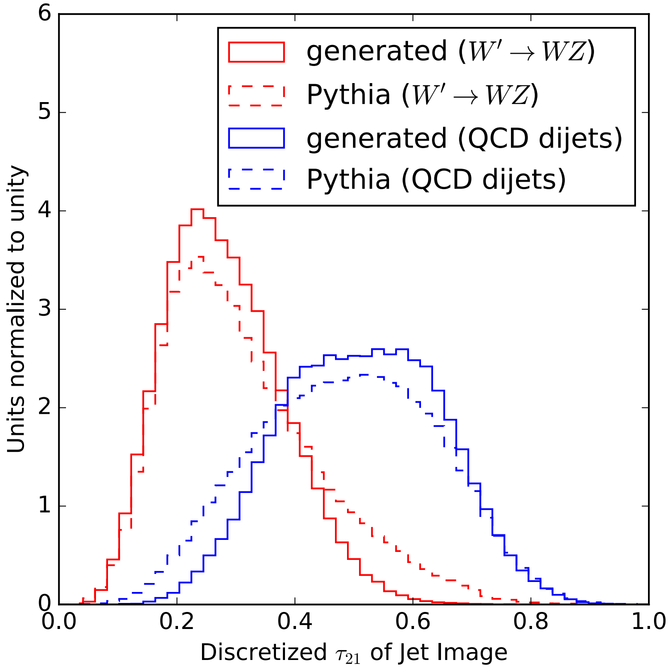
Traditional shower library

- Extract energy deposits from full simulation
- Definition of shower library format to keep size small
  - Characterization of p,x distribution for particles entering the calorimeter to determine initial reasonable binning
  - The cell hit cluster depends significantly on the angle θ -> It is necessary to bin the shower library as a function of the angle of incidence θ
  - There is a perfect symmetry for $\varphi \in [0,\pi]$ and $\varphi \in [\pi,2\pi]$ for the X and Y cluster shift.
- Each particle entering the calorimeter is killed in Geant4
- Output is MCCaloHits, to be processed by the digitization
- First prototype of library is ready
- Need also extensive work on tuning and validation
- Work started last summer, but slow due to lack of personpower

J-F Marchand, M. Rama

Alternative shower library

- Use ML techniques, specifically Location-Aware Generative Adversarial Networks (arXiv: 1701.05927v1) to generate realistic distributions of Calo hits from training samples.
  - An hybrid solution where the interpolation between binned showers would be done with ML techniques would also be possible.
- Developed for image classification, the approach has been already applied to jets generation

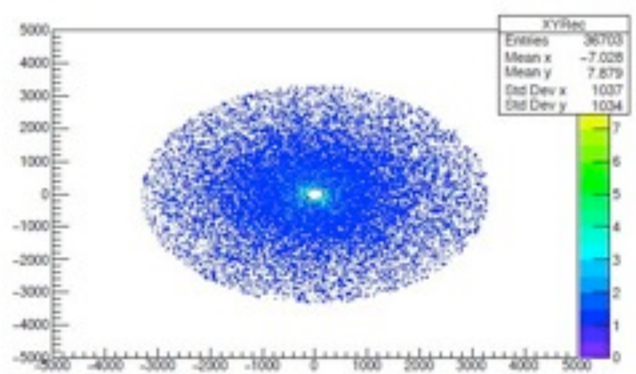


A. Ustyuzhanin, F. Ratnikov, D. Derkach

Fully parametric super fast simulation

B. Siddi

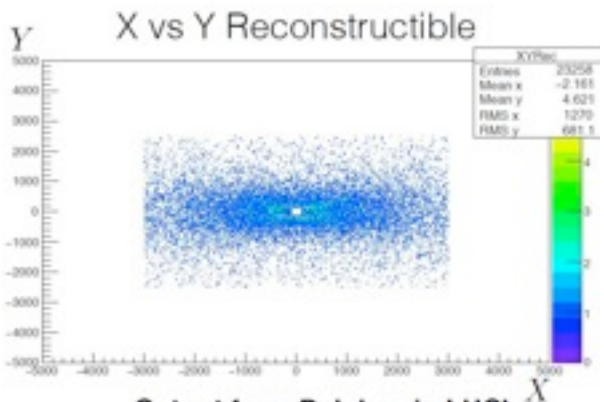
- Work in progress on a fully parametric ultra-fast simulation based on DELPHES package
  - Proof of principle of DELPHES adaptor in Gauss in 2015 (P. Robbe)
  - Identified and implementing extensions necessary in DELPHES to be used for physics studies in LHCb
- Original DELPHES particle propagator module has been rewritten



Original Delphes output shows a cylindrical symmetry not suitble for LHCb

From cylindrical to LHCb geometry

Implemented:
LHCb acceptance, pT kick
Implementing:
tracking efficiency and resolution



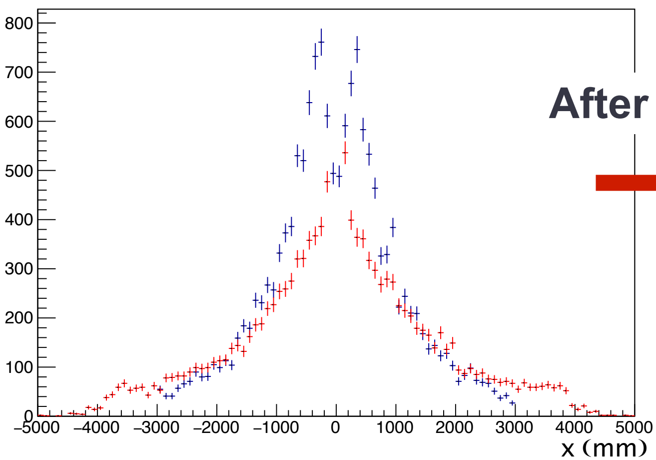
Output from Delphes in LHCb acceptance, TES location

- Output is (will be) directly the LHCb reconstructed high level objects compatible with DaVinci tools

Fully parametric super fast simulation

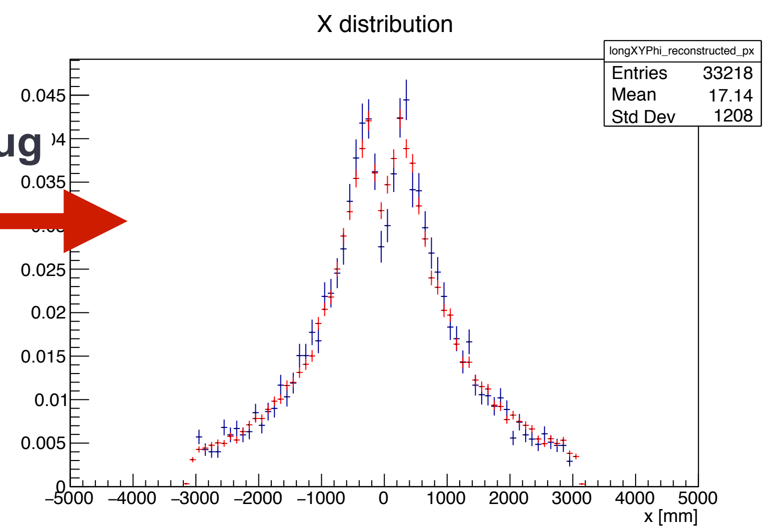
B. Siddi

- The **problem in the efficiency module has been solved**
  - Plot for x, similar for other variables used to parametrize efficiency and resolution (y, phi, tx, ty, p)
- Currently doing a private production of about 10000 minBias events with the same Delphes generator conditions.
- Last thing to do with efficiency is to rewrite our tool in Brunel in order to match perfectly the same Delphes efficiency definition



Distribution of x variable, blue Delphes, red full Simulation

After fixing the bug



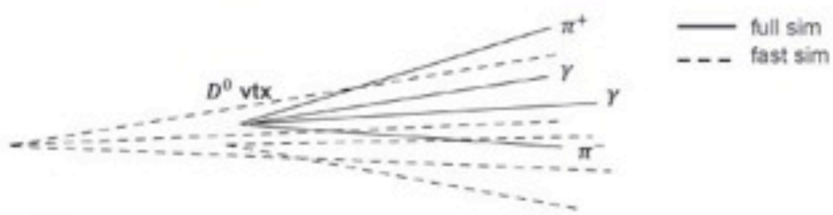
Towards an Integrated Simulation Framework

G. Corti

- Towards a flexible framework to mix fast and full simulated particles in the same event – similar to ATLAS Integrated Simulation Framework
  - Muons always fully simulated
  - Possibility to select full/fast mode according to particle type
  - But could have more complex criteria
  - Treat differently out-of-time events
  - and in-time pileup
  - … we can even consider mixing all
- User configuration has to be as simple as possible and coherent for all fast options

Example

Simulation of decay $D^0 \to \pi^+\pi^-\pi^0[\to \gamma\gamma]$:
full simulation for $\pi^+, \pi^-, \gamma, \gamma$ from signal
fast simulation of calo and/or RICH for the other particles of the event



- Speed close to that of particle gun
- Accuracy close to that of fully simulated event. In particular:
  - reconstruction of PVs
  - track reco degradation
  - possible to implement track isolation

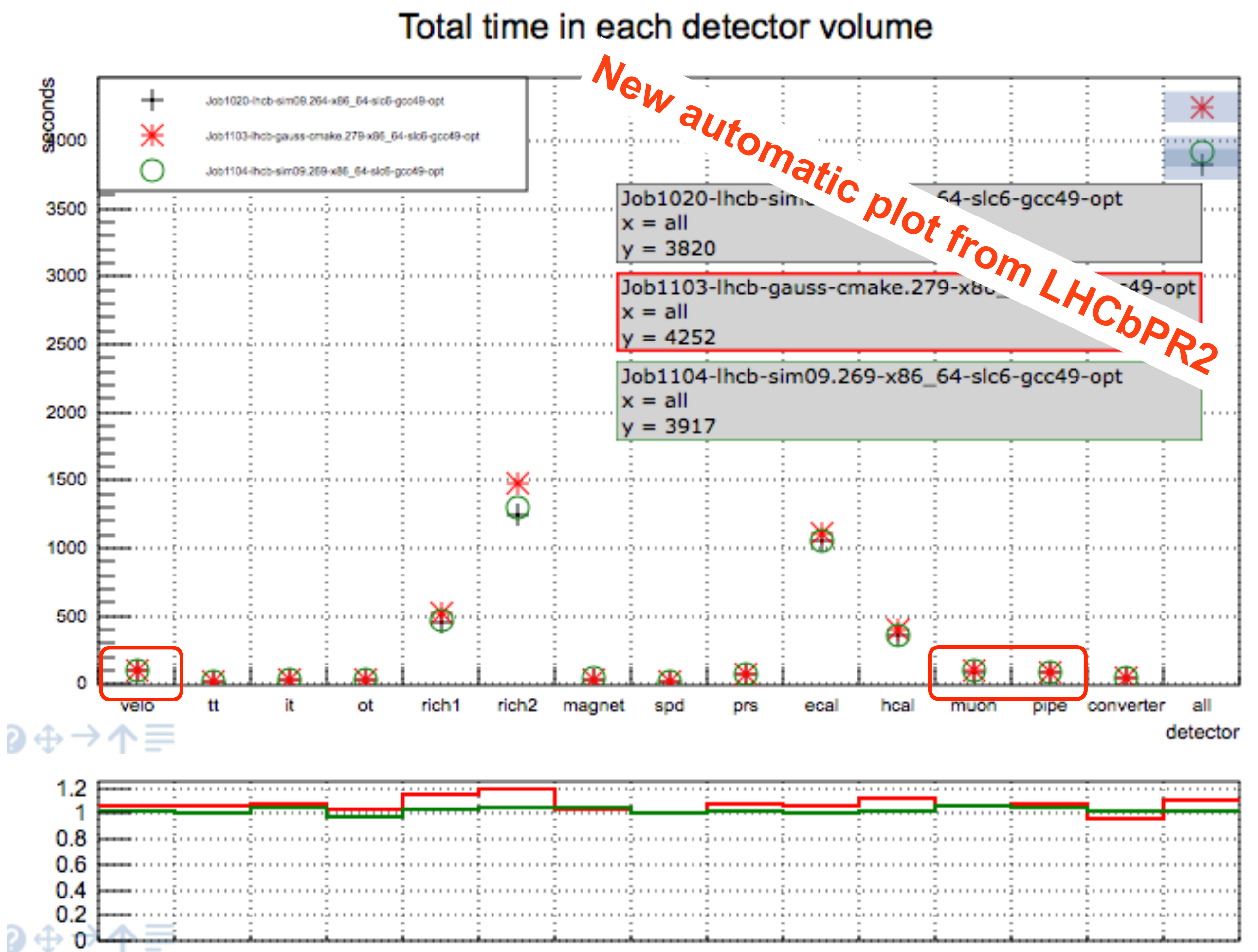
Under investigation

NB: the fast sim mode might even be "RICH-less and calo-less"

M. Rama

Where CPU time is spent in Gauss v49r6

- What to improve after removing RICH Physics and use a shower library for Calo?
- Not clear answer
- Using a simplified geometry may be bring some benefits for tracking



Summary and plans

- **ReDecay**: finally in production, waiting for final validation sample that will also be used for analysis
- **Fully parametric** fast simulation (DELPHES): fixed problem with efficiency module, needed additional fixes to match the efficiency definition used in Brunel
- **Optimization of RICH Full simulation**: available in Gauss v50r1, to be quickly validated before putting in production
- **Fast simulation of Calo information**: parallel ongoing development of traditional library and alternative based on machine learning approach
- Still targeting end of 2017 for a significant speedup, but quite a lot of work to release a full **Integrated Simulation Framework**