Study of the MFT standalone tracking

Optimisation of the LTF radius

- Check the impact of the search radius and its geometry
- Looking at simulations obtained with BoxGen

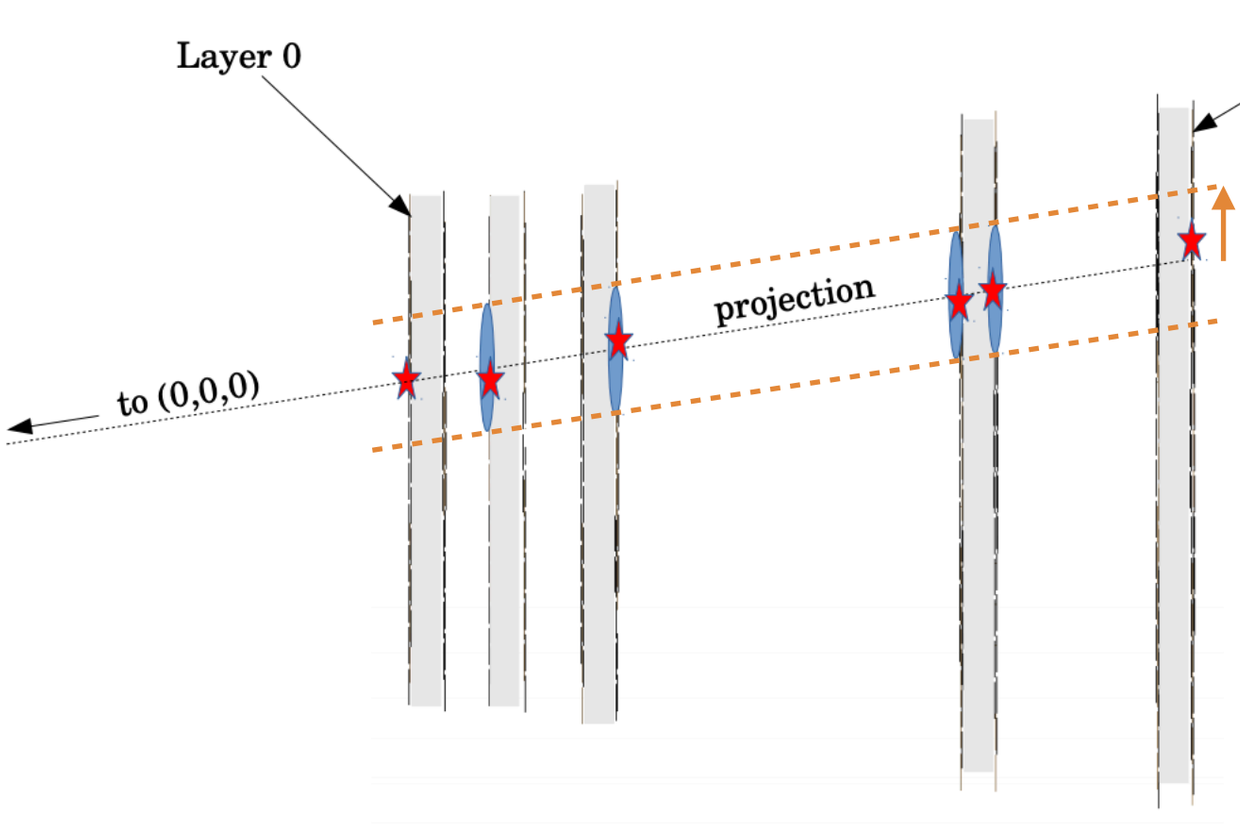
Audrey Francisco, LPC

Basics of the MFT standalone tracking

2 algorithms : **Linear Track Finder** (LTF) + Cellular automaton (CA)

The **more tracks the LTF finds the better** as the CA is more refined and therefore slower

Simplified principle of the LTF method



Radius R = maximum distance for a cluster to be attached to a seed line

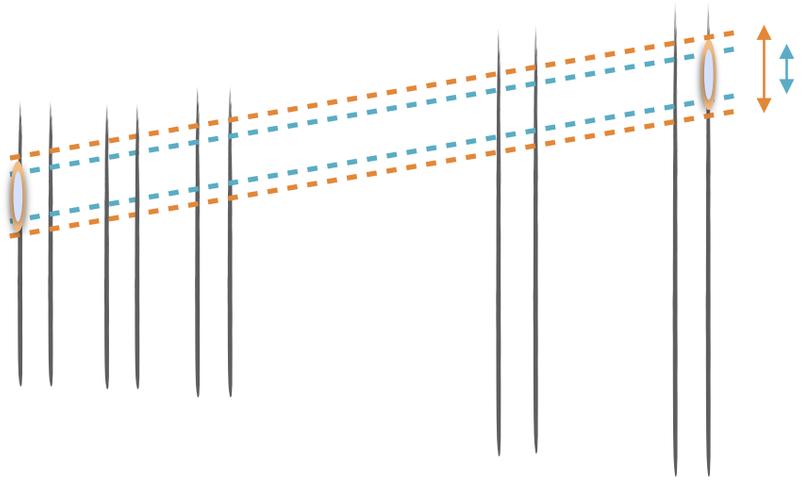
Reference : R = 100 µm

Search road = cylinder → same radius for all plane

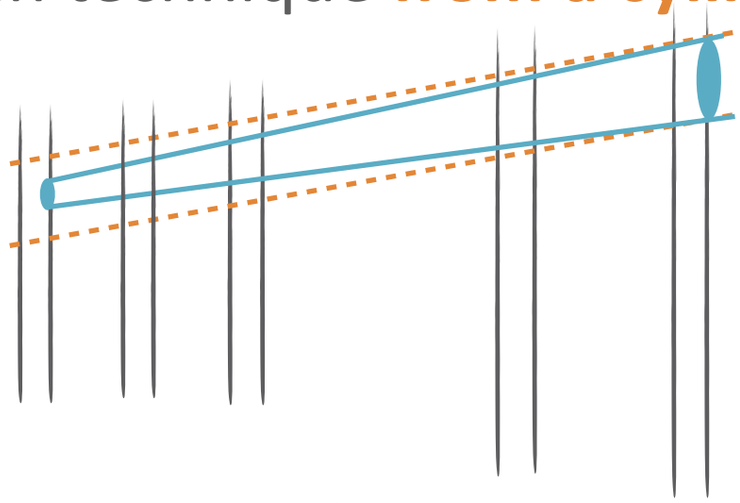
Optimisation of the LTF for the standalone tracking

Focusing at low-p_T

1. Change main reconstruction parameter: **reduce the search radius**



2. Change the search technique **from a cylinder to a cone**



$$R_i = R_0 \left(1 + \frac{z_i - z_0}{z_0}\right)$$

3. Impact of the z-vertex distribution range

   - Using boxgen simulations : muon only

4. *Longer term : check for more realistic simulations with full event*
   + Perform similar studies for the CA algorithm

Reducing the LTF radius

- 10000 events ran locally with Geant3 (boxgen)
  10 muons per event up to p=10GeV/c
  -5<**η**< 0.
  Default cuts and parameters

- How to run this?

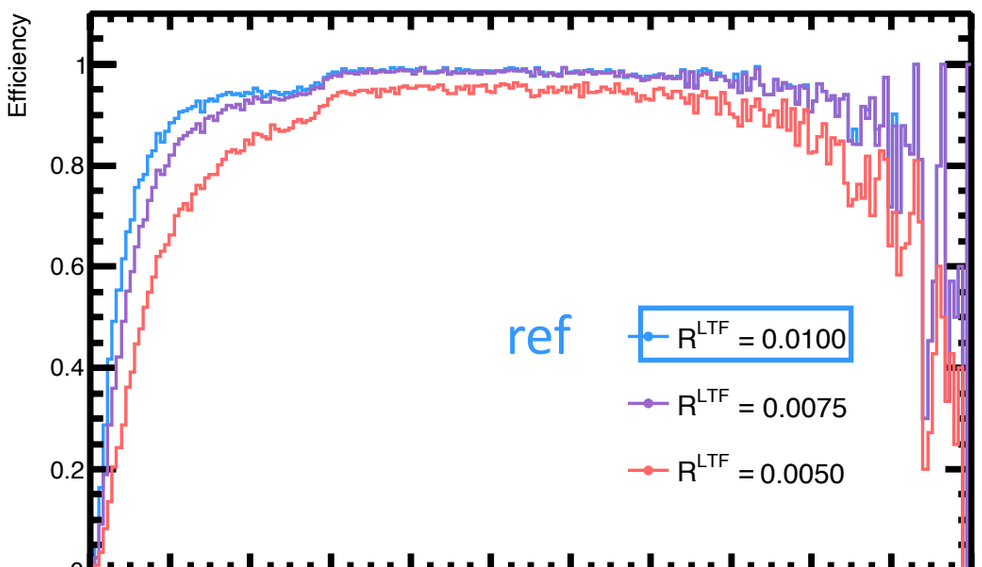
```
>o2-sim-serial -m MFT -e TGeant3 -n 10000 -g boxgen --configKeyValues 'BoxGun.pdg=13;
BoxGun.eta[0]=-3.6; BoxGun.eta[1]=-2.5; BoxGun.prange[0]=0.0; BoxGun.prange[1]=10.0;
BoxGun.number=10’

>o2-sim-digitizer-workflow -b

>o2-mft-reco-workflow -b
```

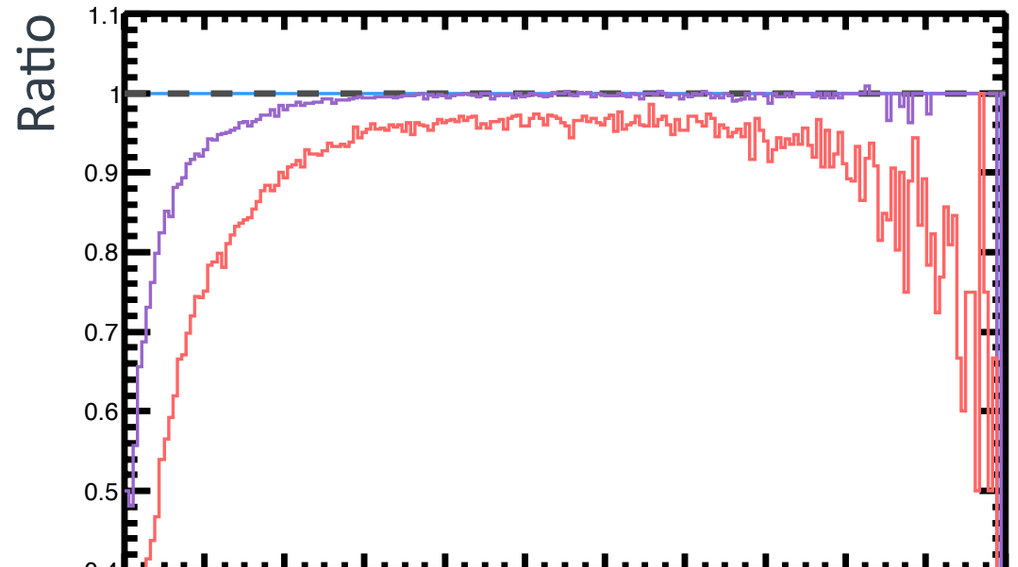
Reducing the LTF radius

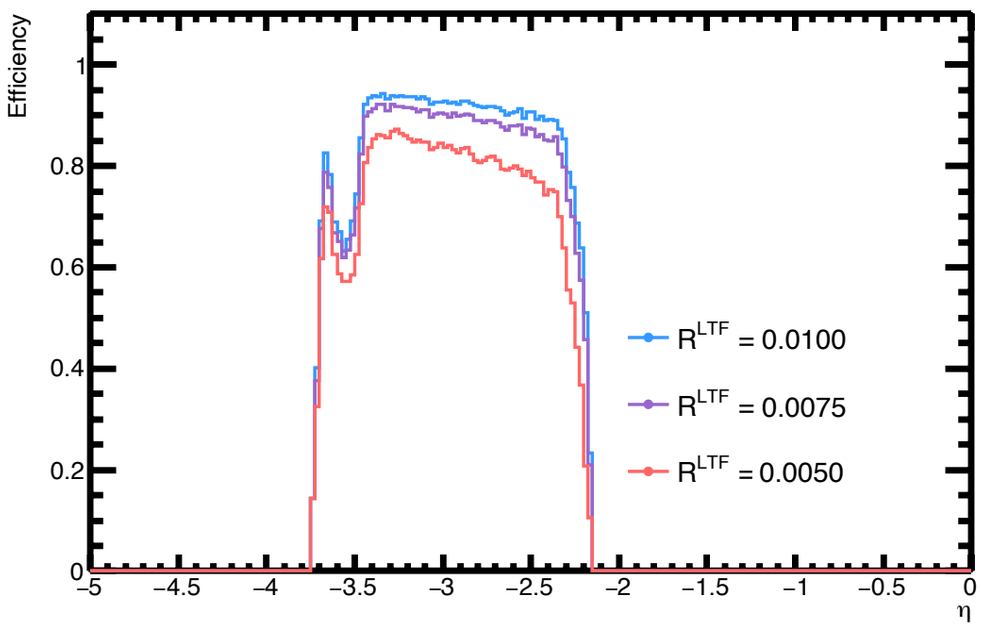
BoxGen 10muons

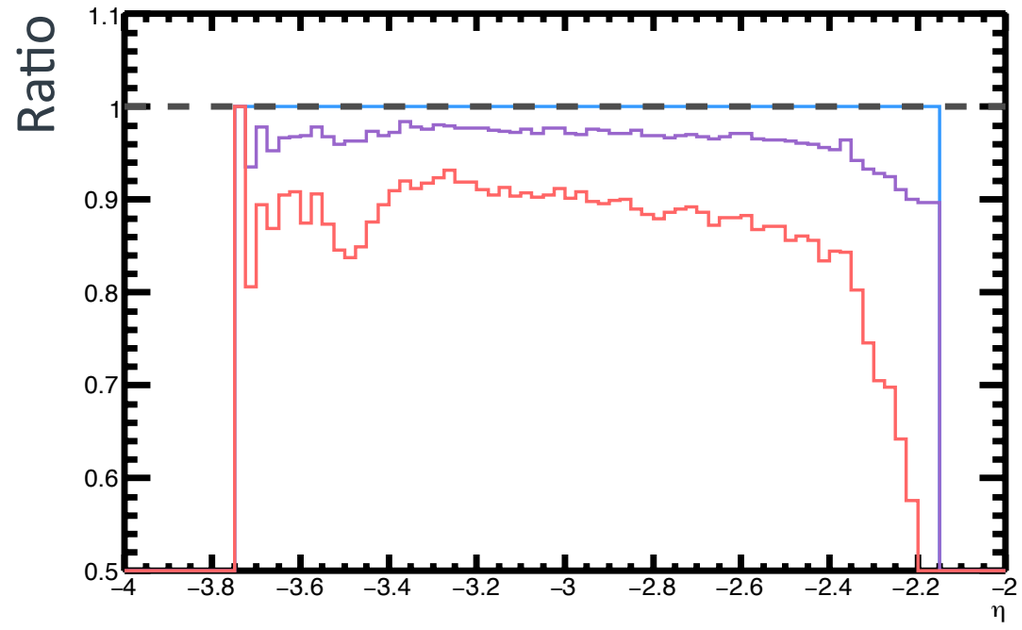
Efficiency of the LTF algo

$$\text{Efficiency} = \frac{\text{Reconstructed tracks}}{\text{Trackable tracks}}$$







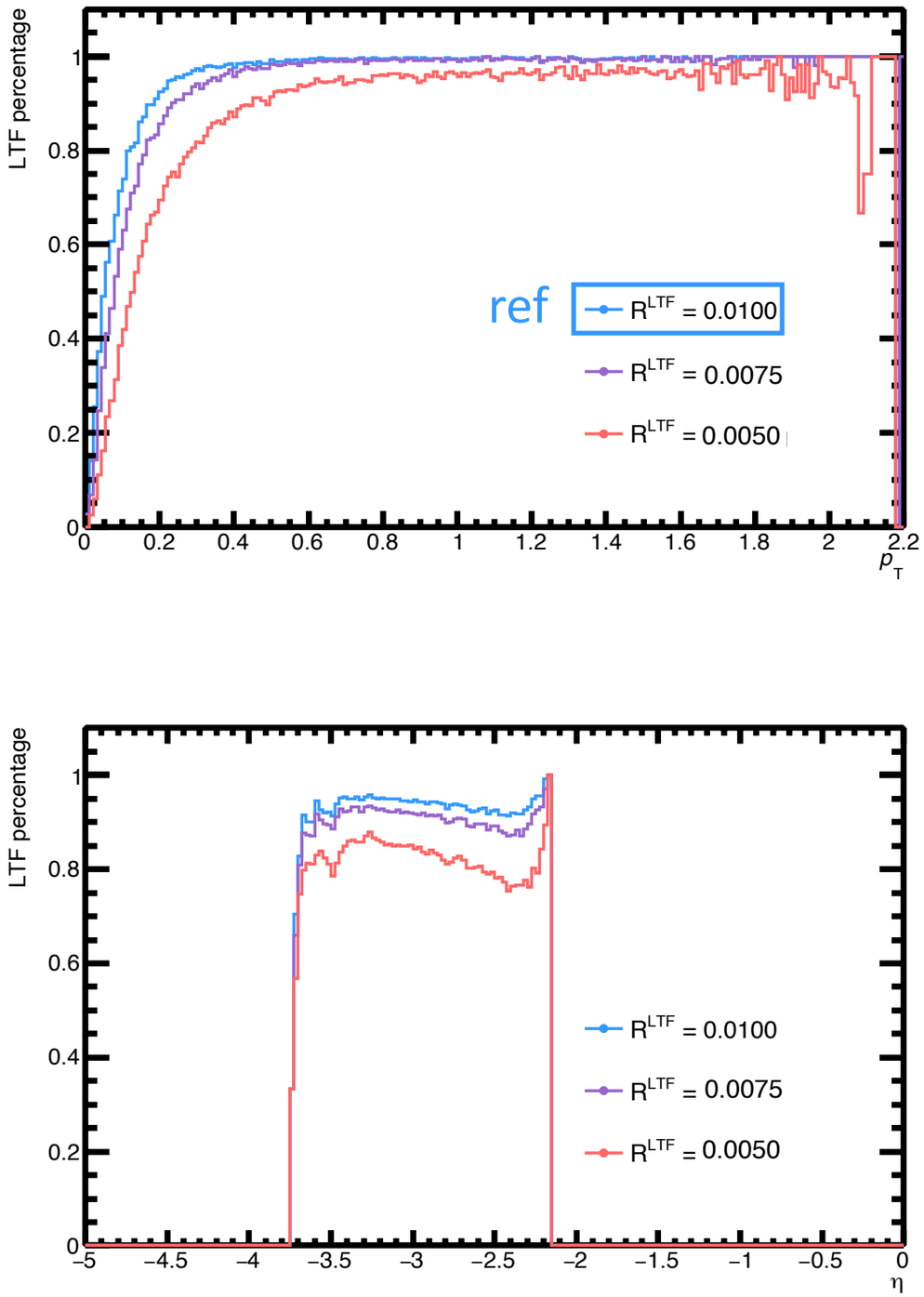


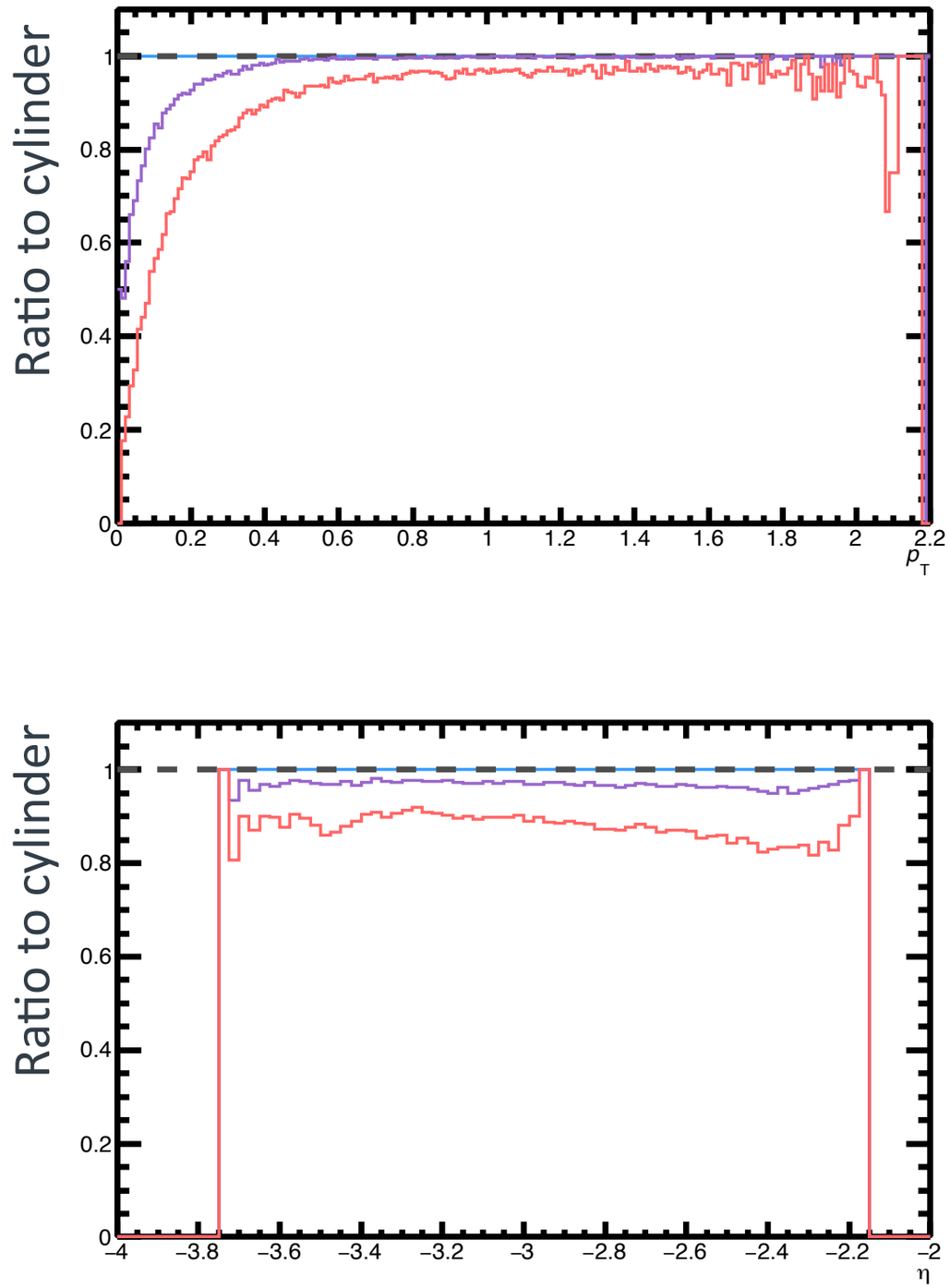
Reducing the LTF radius

BoxGen 10muons

Percentage of tracks found by the LTF

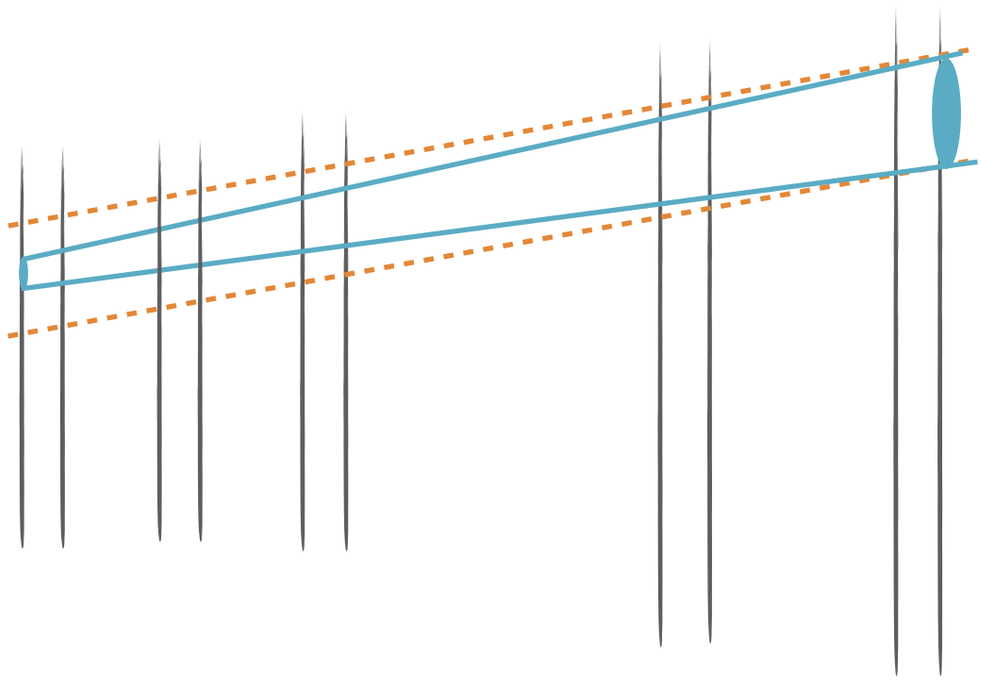
$$\text{Percentage} = \frac{\text{LTF tracks}}{\text{LTF + CA tracks}}$$





Changing the LTF road from a tube to a cone

- Same simulation file : 10000 events ran locally with Geant3 (boxgen)
  10 muons per event up to p=10GeV/c
- **Radius increase for the LTF algo as a function of the plane z**
- CA road is still a cylinder



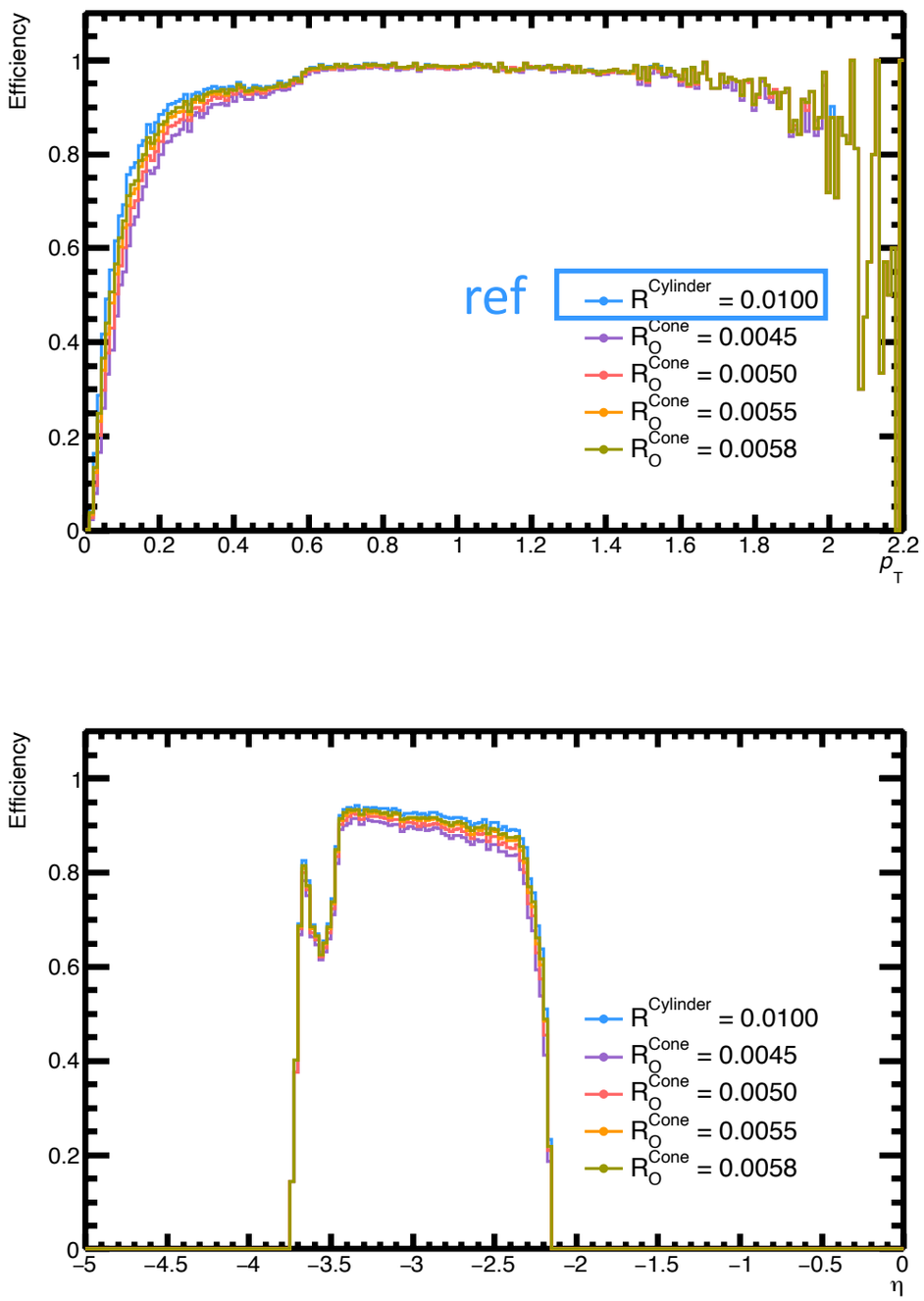
$$R_i = R_0 \left(1 + \frac{z_i - z_0}{z_0}\right)$$

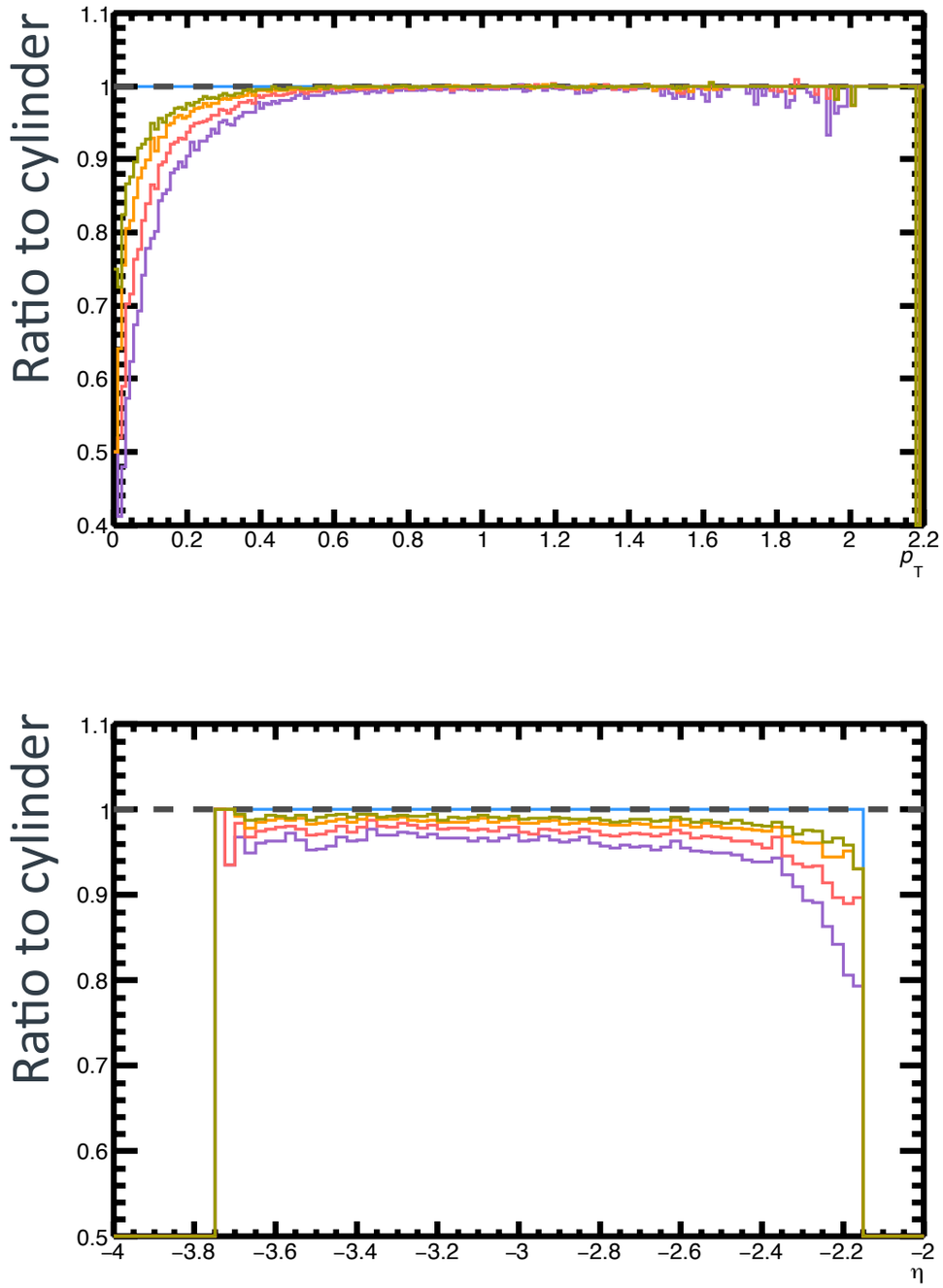
From cylinder to cone

BoxGen 10muons

Efficiency of the LTF algo

$$\text{Efficiency} = \frac{\text{Reconstructed tracks}}{\text{Trackable tracks}}$$



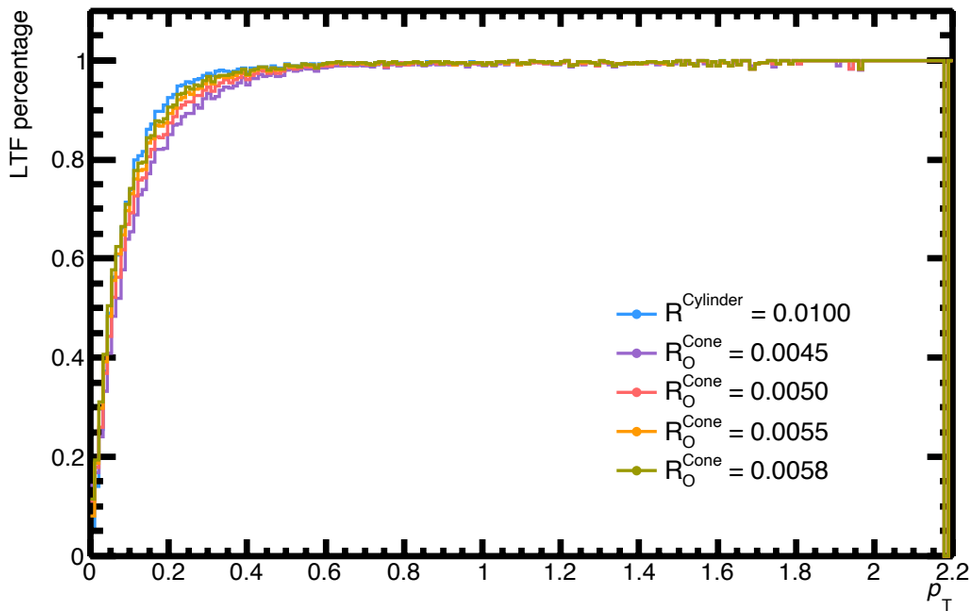


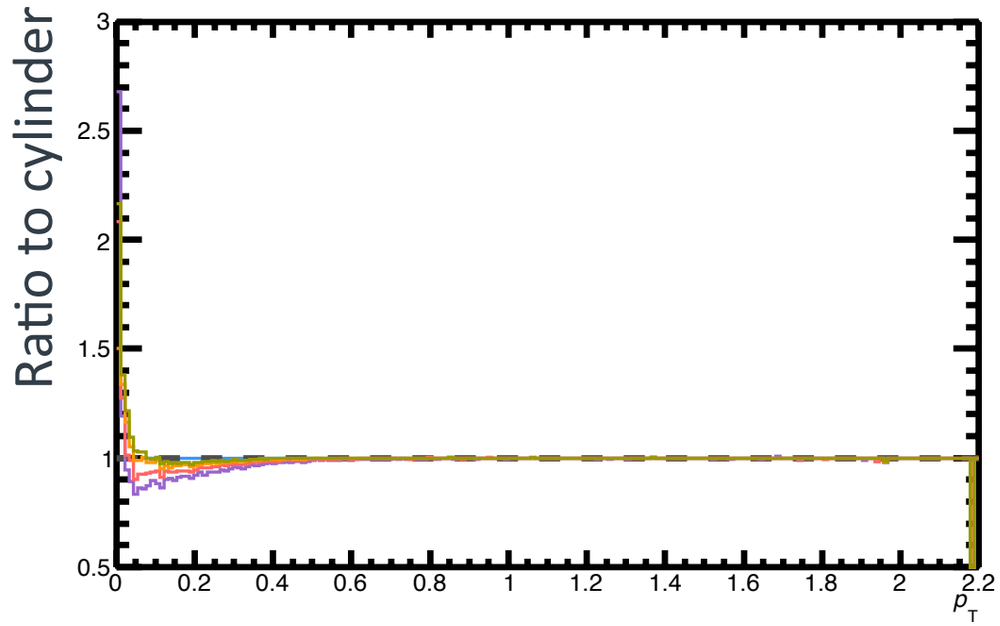
From cylinder to cone

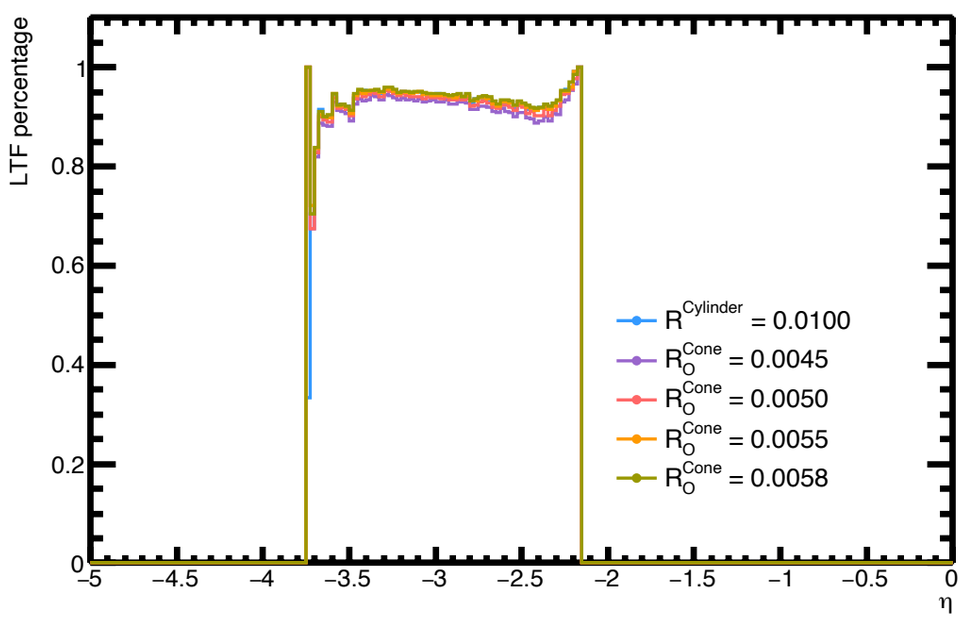
BoxGen 10muons

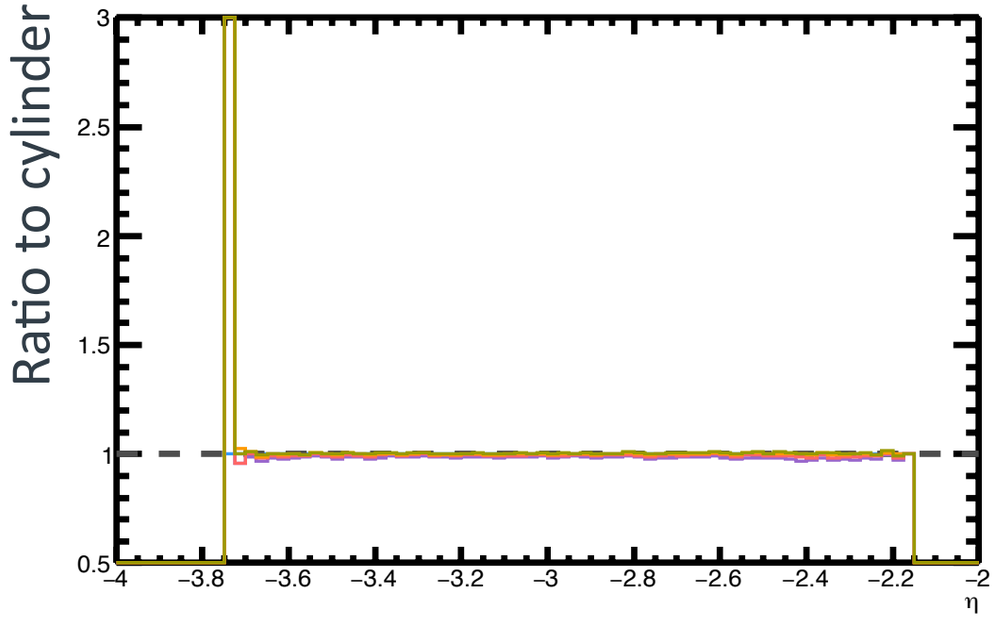
Percentage of tracks found by the LTF

$$\text{Percentage} = \frac{\text{LTF tracks}}{\text{LTF + CA tracks}}$$



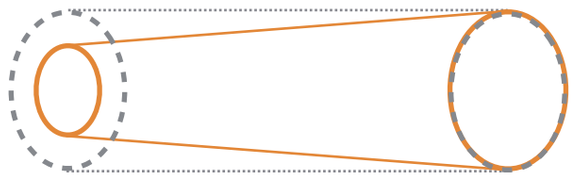






Effect of a larger z-vertex distribution

- Similar simulation files : 10000 events ran locally with Geant3 (boxgen)
  10 muons per event up to p=10GeV/c
- **V_z at +/- 6cm, 15cm and 30cm**
- **Checking the radius increase for the LTF algo as a function of the plane z**
- CA road is still a tube



$$R_i = R_0 \left(1 + \frac{z_i - z_0}{z_0}\right)$$

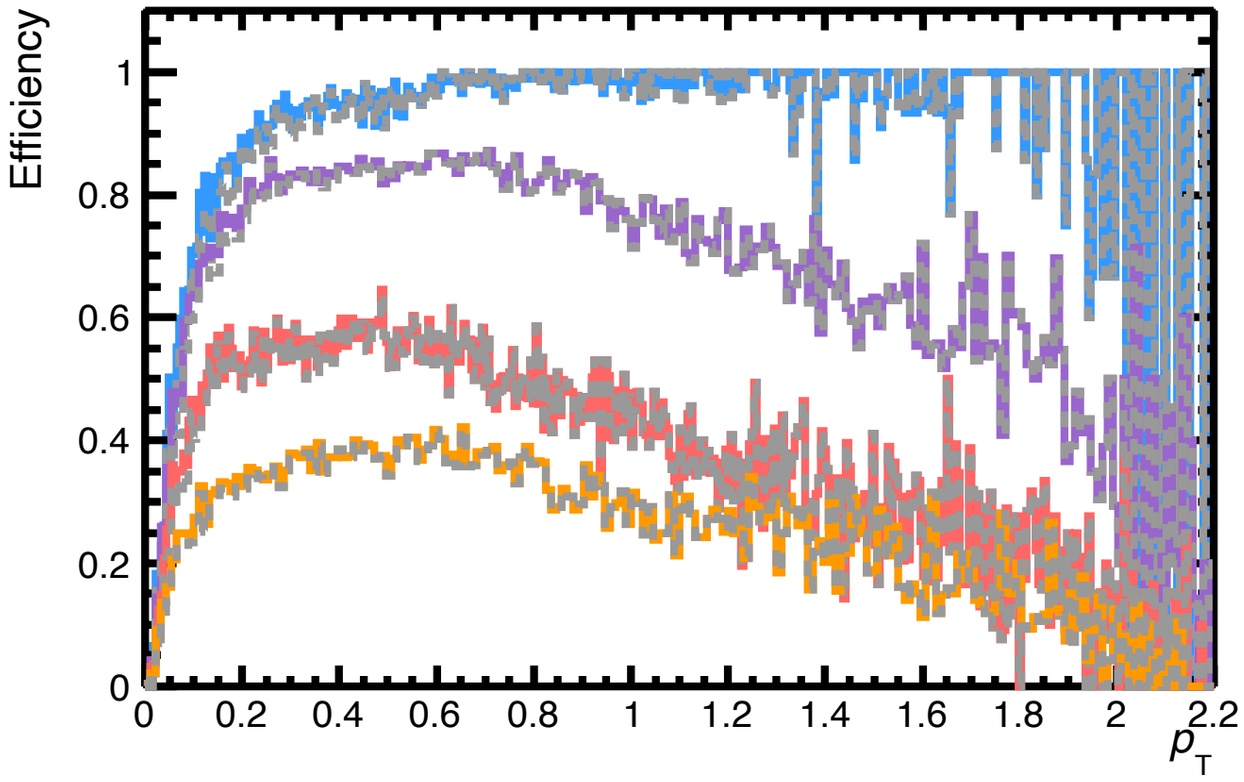
- How to run this?

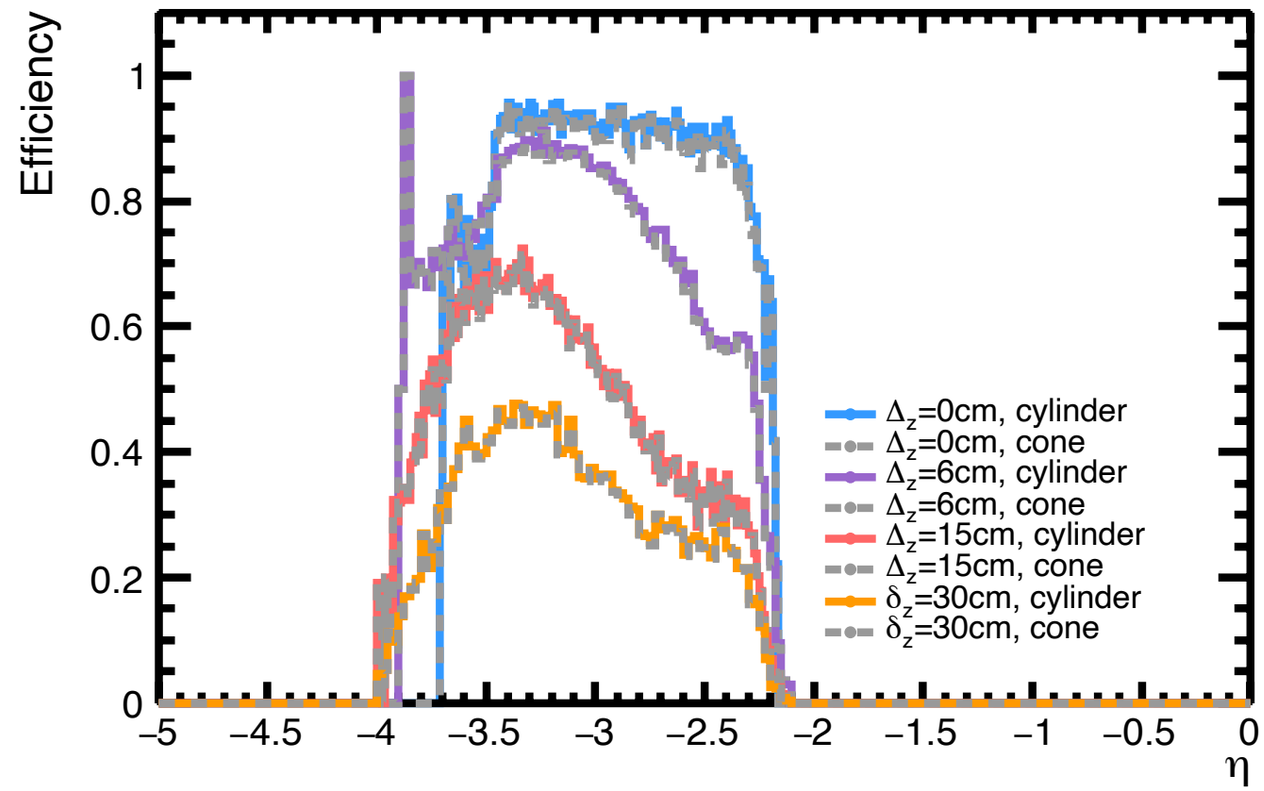
```
>o2-sim-serial -m MFT -e TGeant3 -n 10000 -g boxgen --configKeyValues 'BoxGun.pdg=13; BoxGun.eta[0]=-3.6; BoxGun.eta[1]=-2.5;
BoxGun.prange[0]=0.0; BoxGun.prange[1]=10.0; BoxGun.number=10;Diamond.width[2]=6.'

>o2-sim-digitizer-workflow -b

>o2-mft-reco-workflow -b
```

Effect of the z-vertex distribution

BoxGen 10muons

Efficiency of the LTF algo





No real loss of efficiency moving from a cone to a cylinder

Conclusions

Investigation of **the LTF search radius** by looking at simulations obtained with BoxGen : how to enhance the track reconstruction quality with the LTF?

Keeping the radius constant:
current value (0.0100 cm) already close to an optimum

- Changing the search area **from a tube to a cone seems to provide interesting results**

Next steps

- Look at more realistic events from Pythia simulations
  Pull request in preparation (next days) to run on lxplus
  Check the gain on time computation

- Perform similar study for the radius of the Cellular Automaton algorithm