TOF Transformer - Update

★ R=12.0mm, T=190ps

★ R=10.3mm, T=179ps

Data

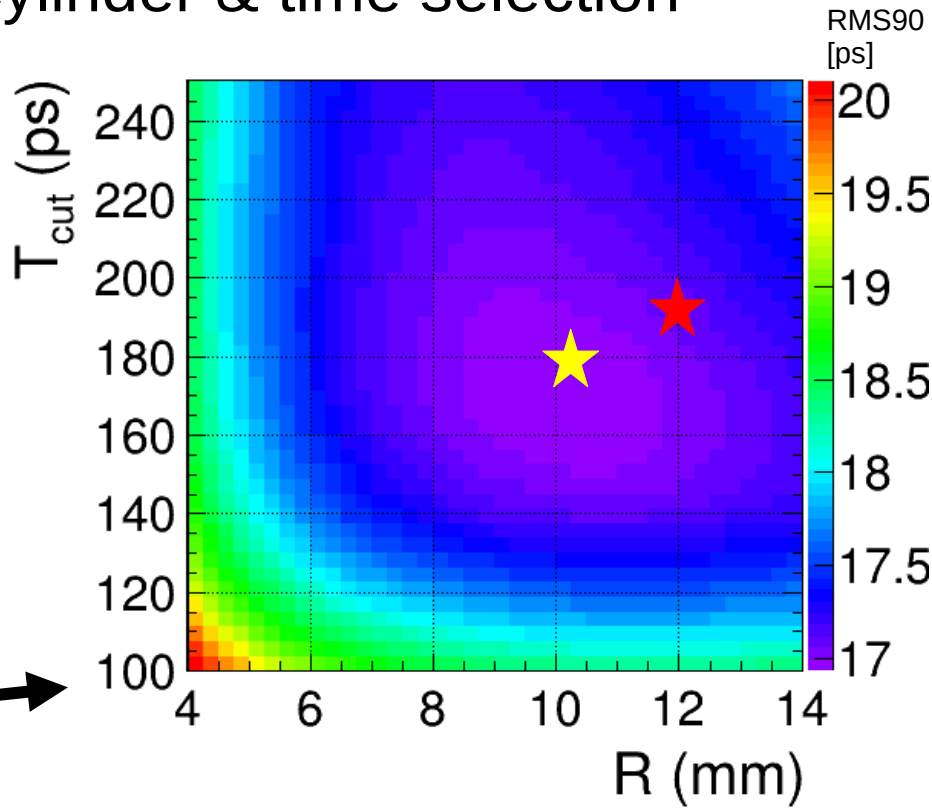
- Transformer trained on Bohdan's cylinder & time selection

★ Transformer trained here

★ Optimal values

- First 10 Ecal layers
- Bohdan's algorithm on Bohdan's data*:
  - RMS90: 16.99ps

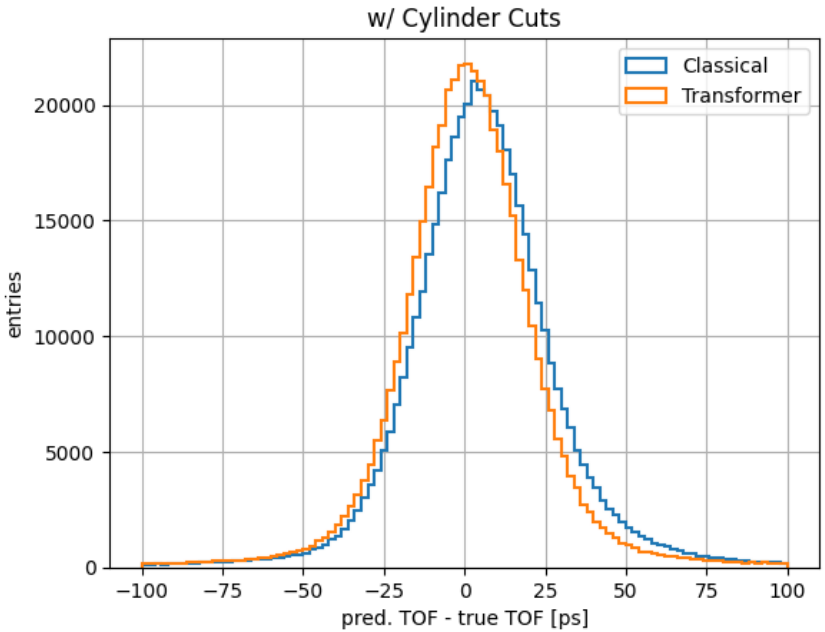
Cylinder algorithm parameter scan →



*see remarks

Transformer #1

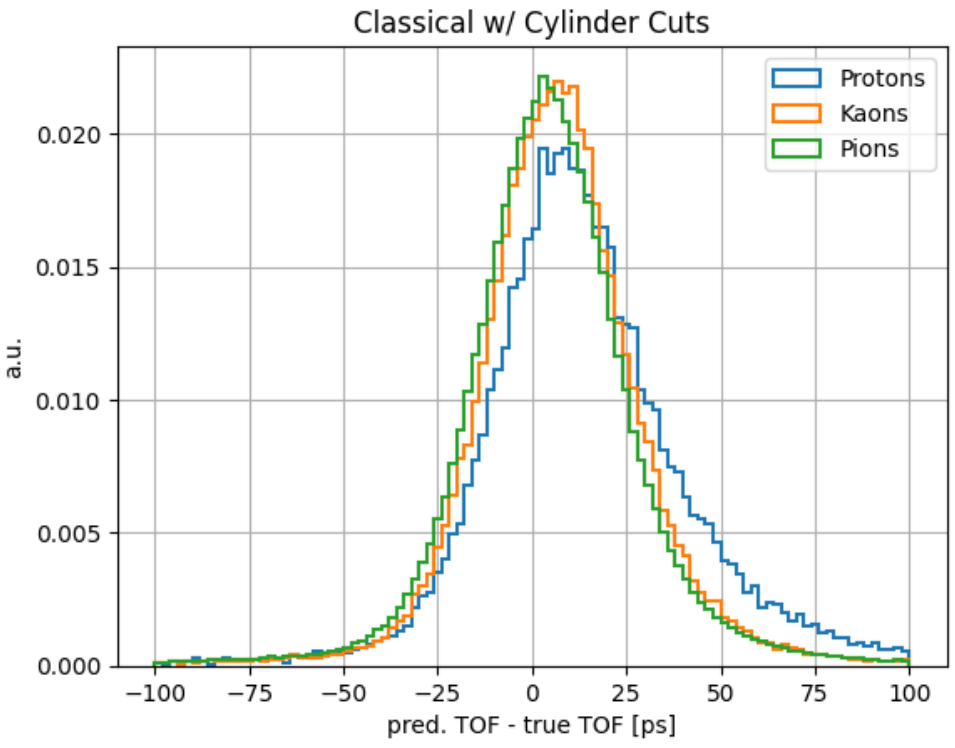
- Performance on validation dataset with 500k showers
- Transformer:
  - RMS90: (16.17+-0.02)ps
  - Mean90: (1.18+-0.02)ps
- Bohdan’s algorithm on my data*:
  - RMS90: (17.27+-0.02)ps
  - Mean90: (5.41+-0.03)ps



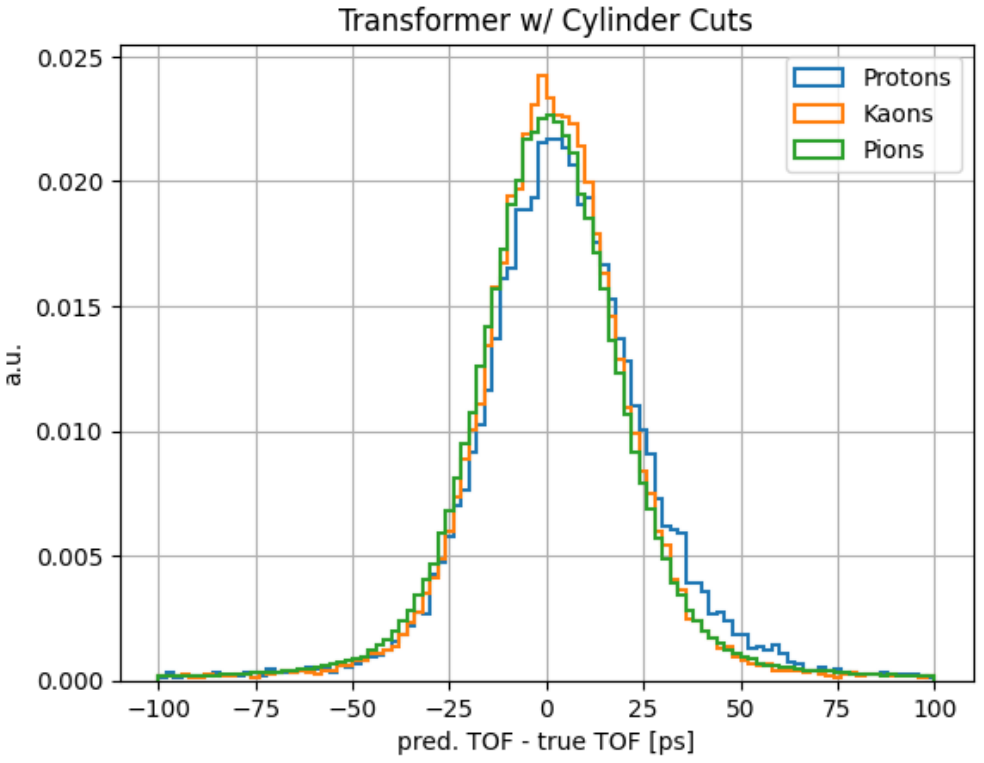
*see remarks

Transformer #2

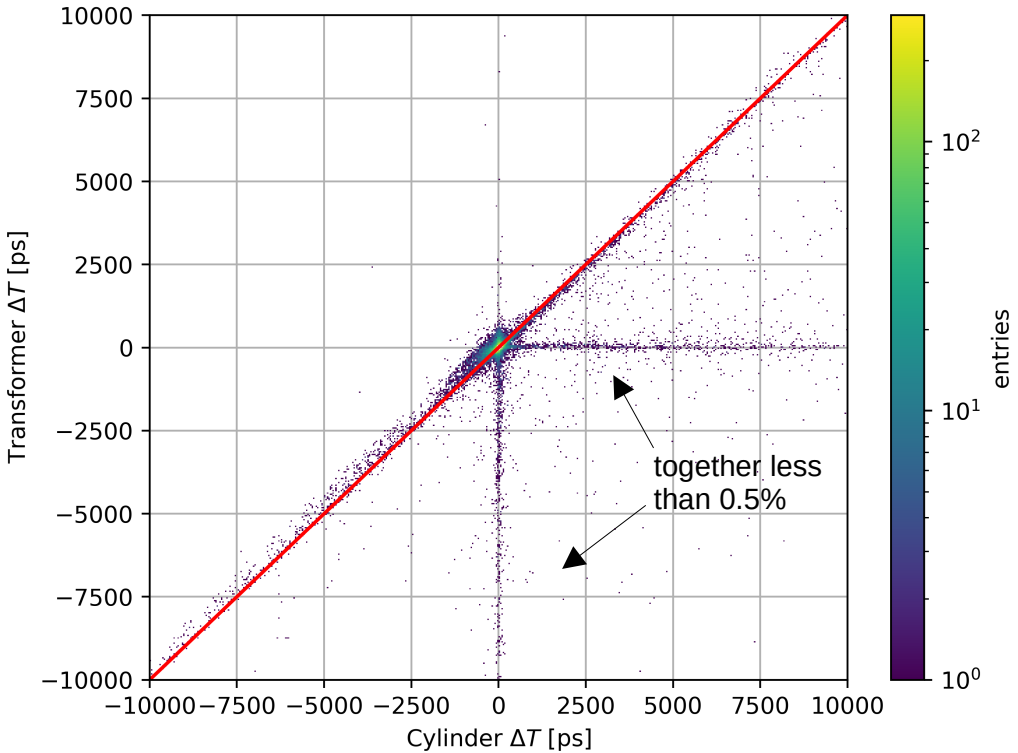
Bohdan:

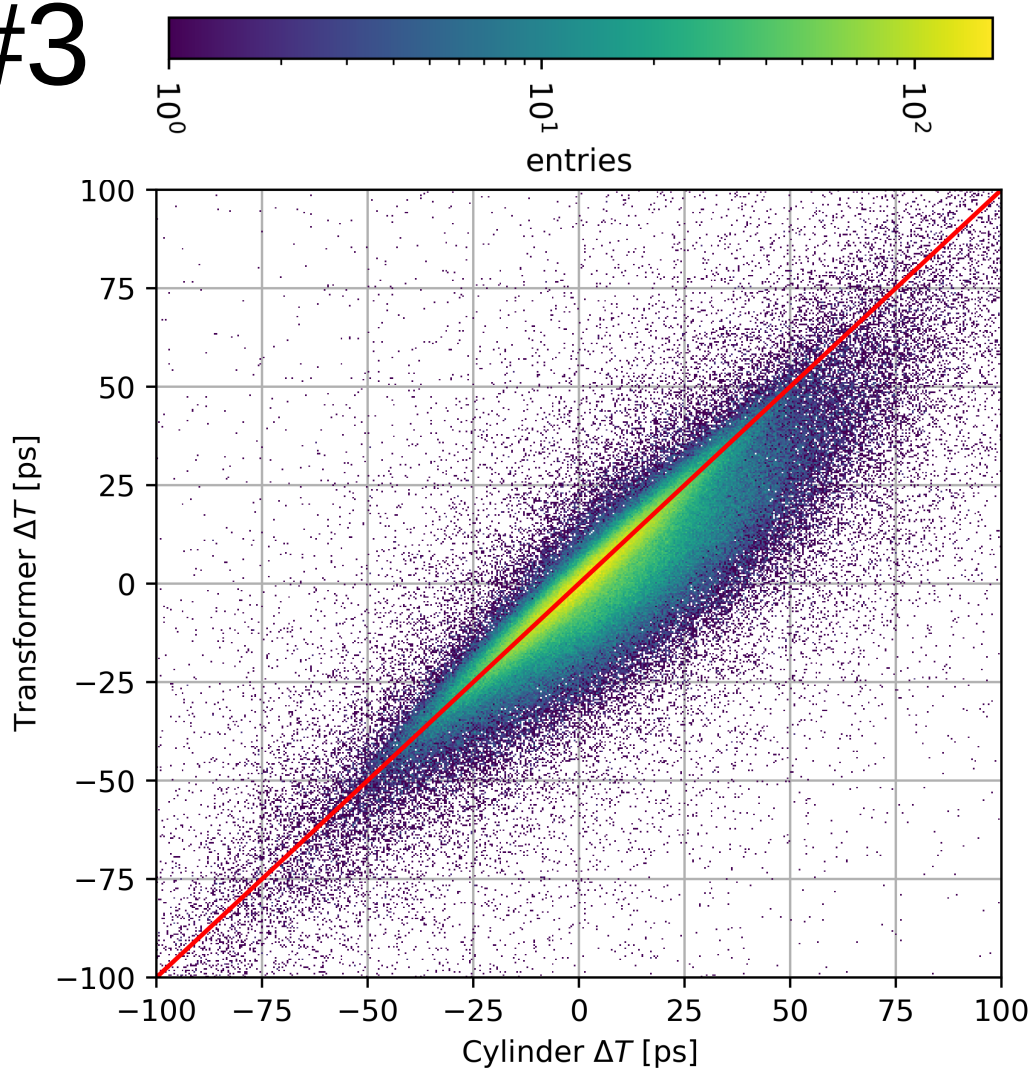


Transformer:

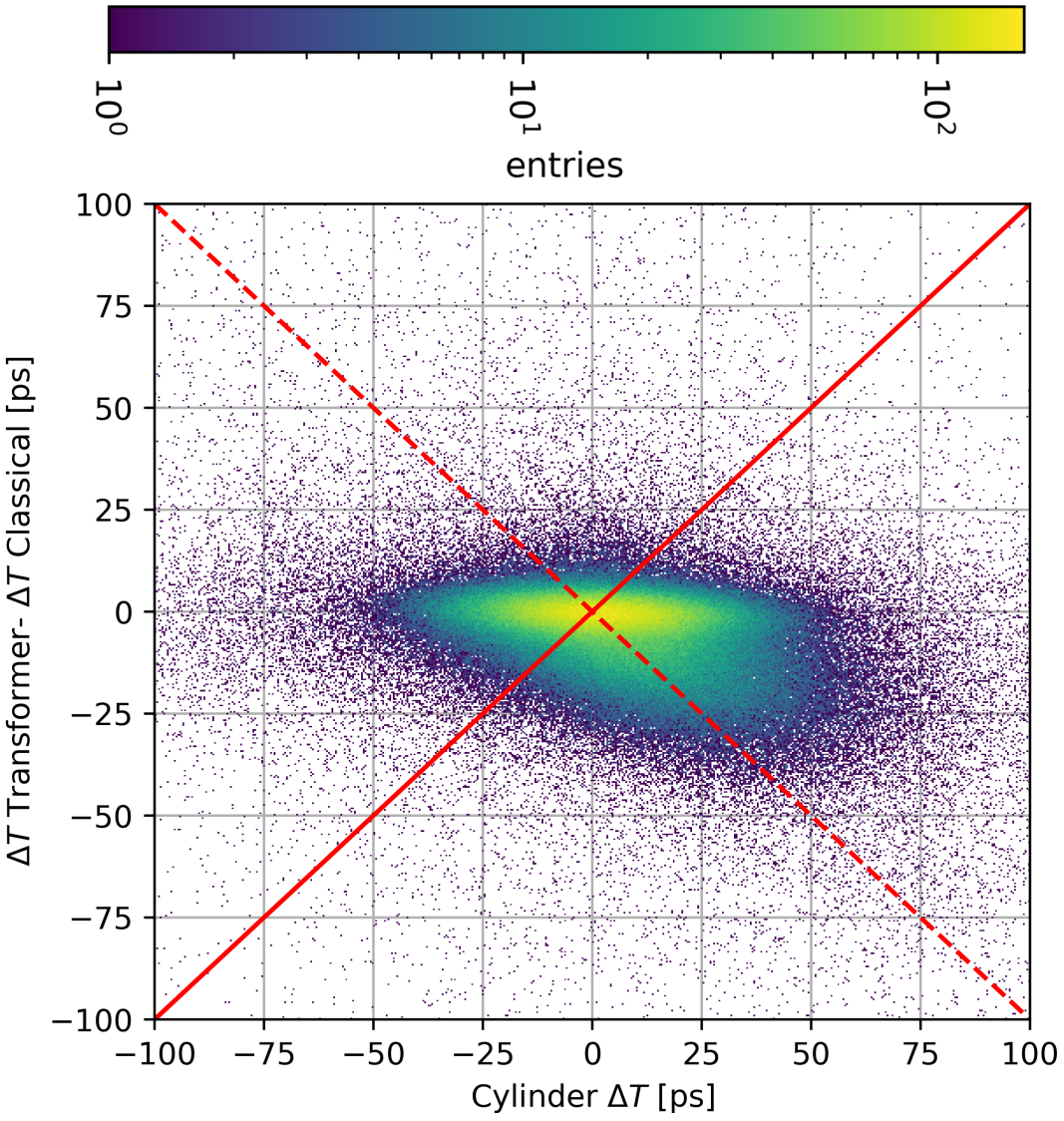


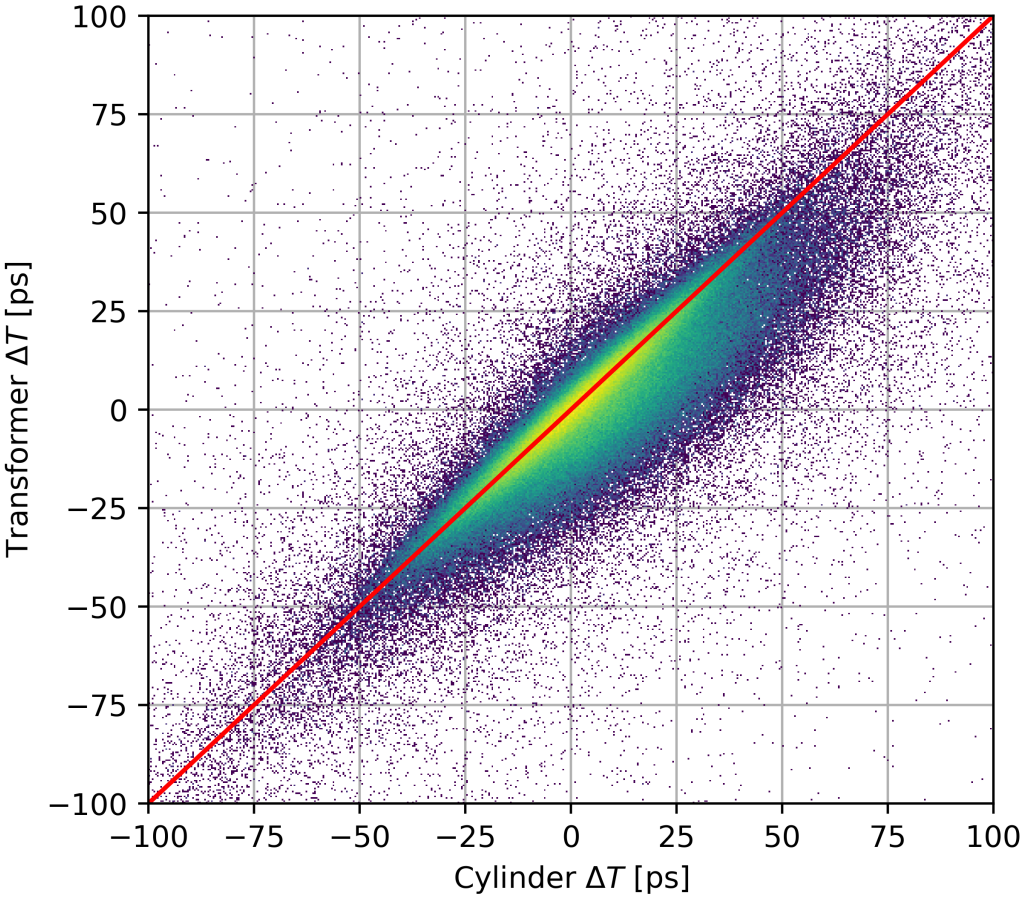
Transformer #3





Transformer #5





Transformer #6

- Transformer training on whole shower (<10 layers): currently training

- Currently using the same model for the whole shower, this (probably) has to be changed
- Model training (strongly) dependent on the loss function
- Always training on 1,000,000 showers

- Note:
  - More data = worse results bug does not appear anymore!

Transformer #7

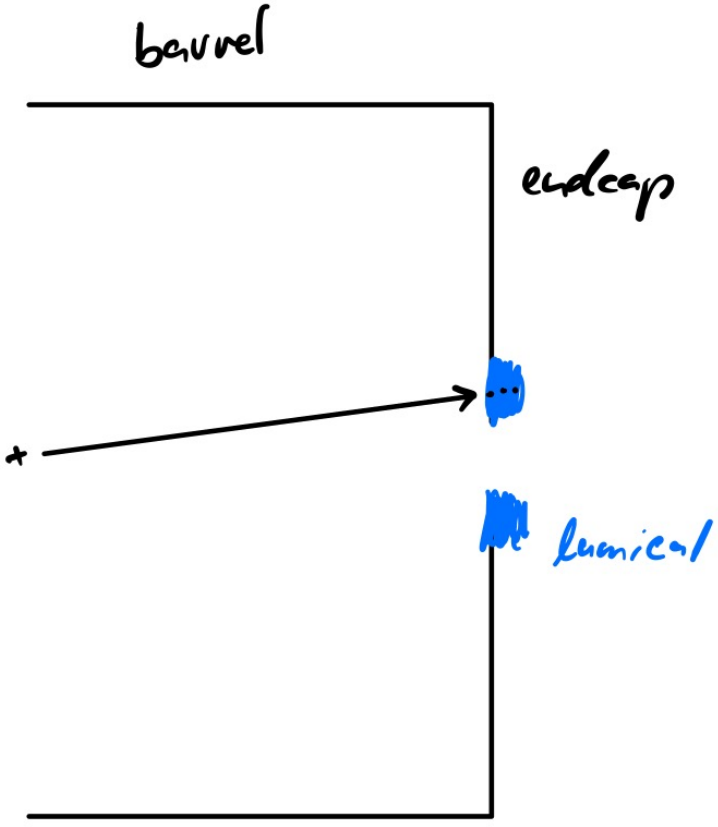
- Total number of model parameters: 606,593

```
# init. model
model = src.ml_utils.TOFTransformer(shower_dim=shower_dim,
                                    max_n_hits=max_n_hits,
                                    d_model=128,
                                    n_class_attn_layers=3,
                                    n_class_attn_heads=1,
                                    class_attn_dim_feedforward=512,
                                    dropout_value=0.,
                                    pos_encoding=False,
                                    device=device
                                   )
```

Remarks

- Yesterday:
  - discovered bug in data generation → see next slide
  - Bohdan ran his algorithm again, no (significant) changes in the results → RMS90 worse(?!) by 0.05ps
- Bohdan’s data =/= my data:
  - I have a momentum cut p<10GeV
- Whole shower = cylinder cut with 999mm cylinder radius and time median cut of 999ps

The Bug


barrel
endcap
+
luminal


only hit we see
ECal hit
used to get true TOF
track

Cylinder Algorithm

- Calculate d_perp to every hit
- Remove all hits with d_perp > R
  - If no hit has d_perp <= R, select the one with smallest d_perp and remove all other hits
- Calculate: t_i – d_i/c
- Calculate median of t_i – d_i/c, i=1,…
- Remove all hits with |median corrected time – t_i| > dT
- If no hit has |median corrected time – t_i| > dT, select the one with smallest |median corrected time – t_i| and remove all other hits