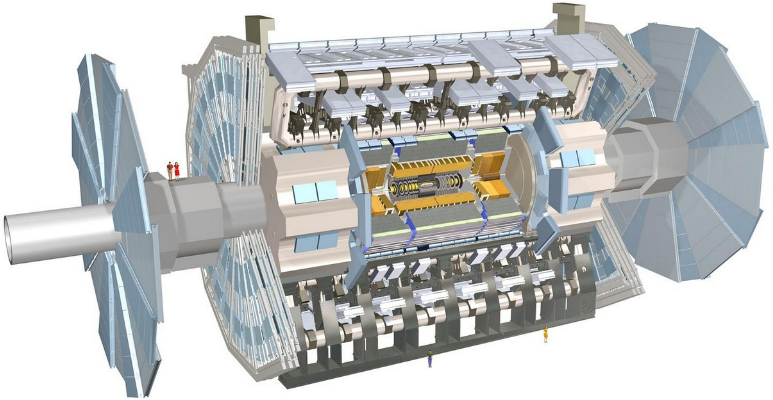
ML for hadronic jets in ATLAS DNN and GNN

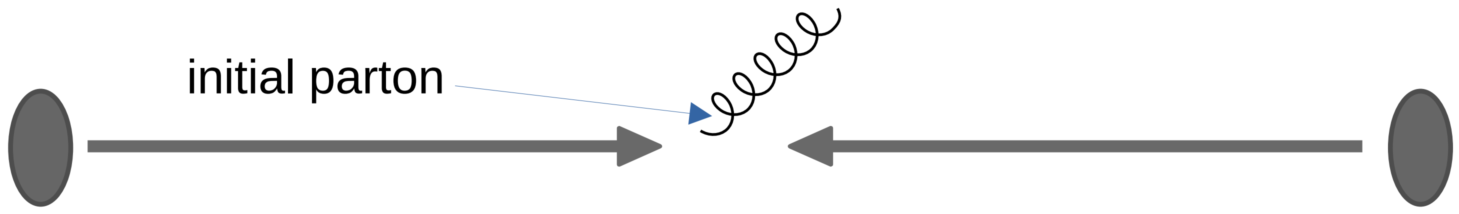
ATLAS ML Team at LPSC:
- Guillaume Albouy
- Ana Peixoto
- Thomas Wojtkowski
- P-A D

Experimental context

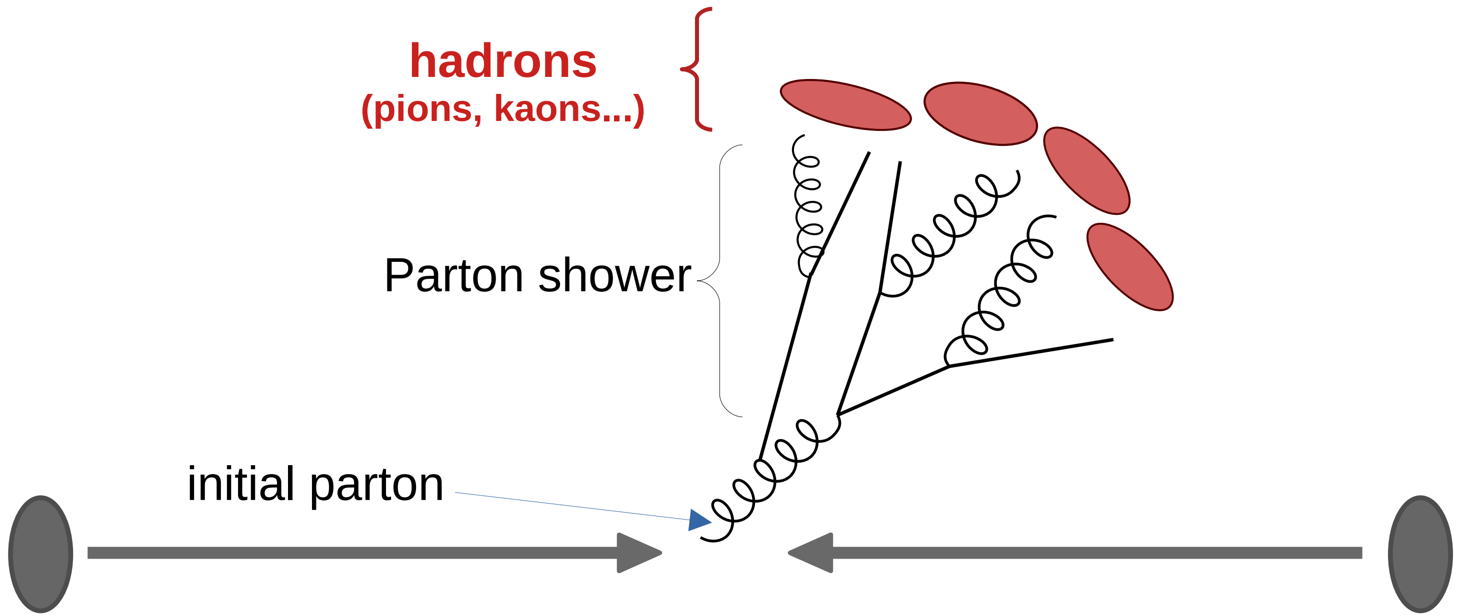
- Proton collision at LHC



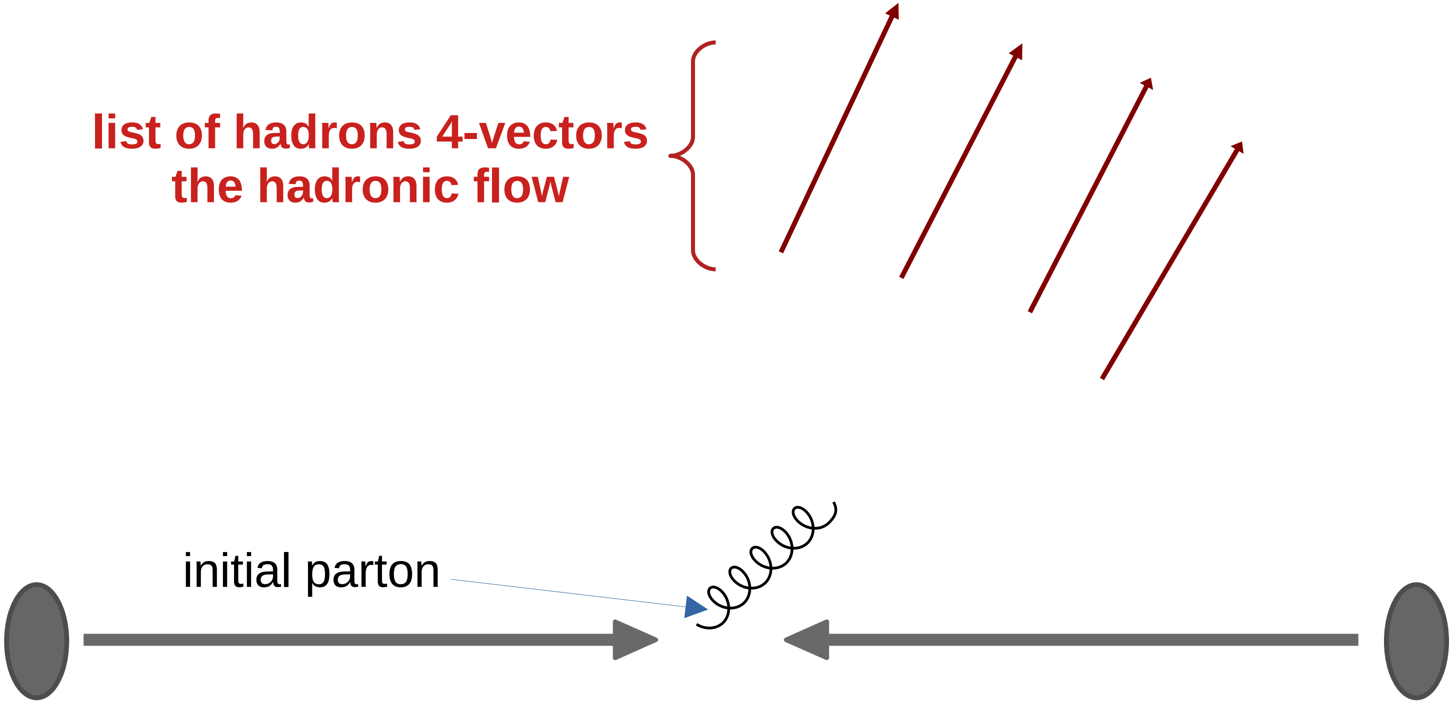
Hadronic jets



Hadronic jets



Hadronic jets



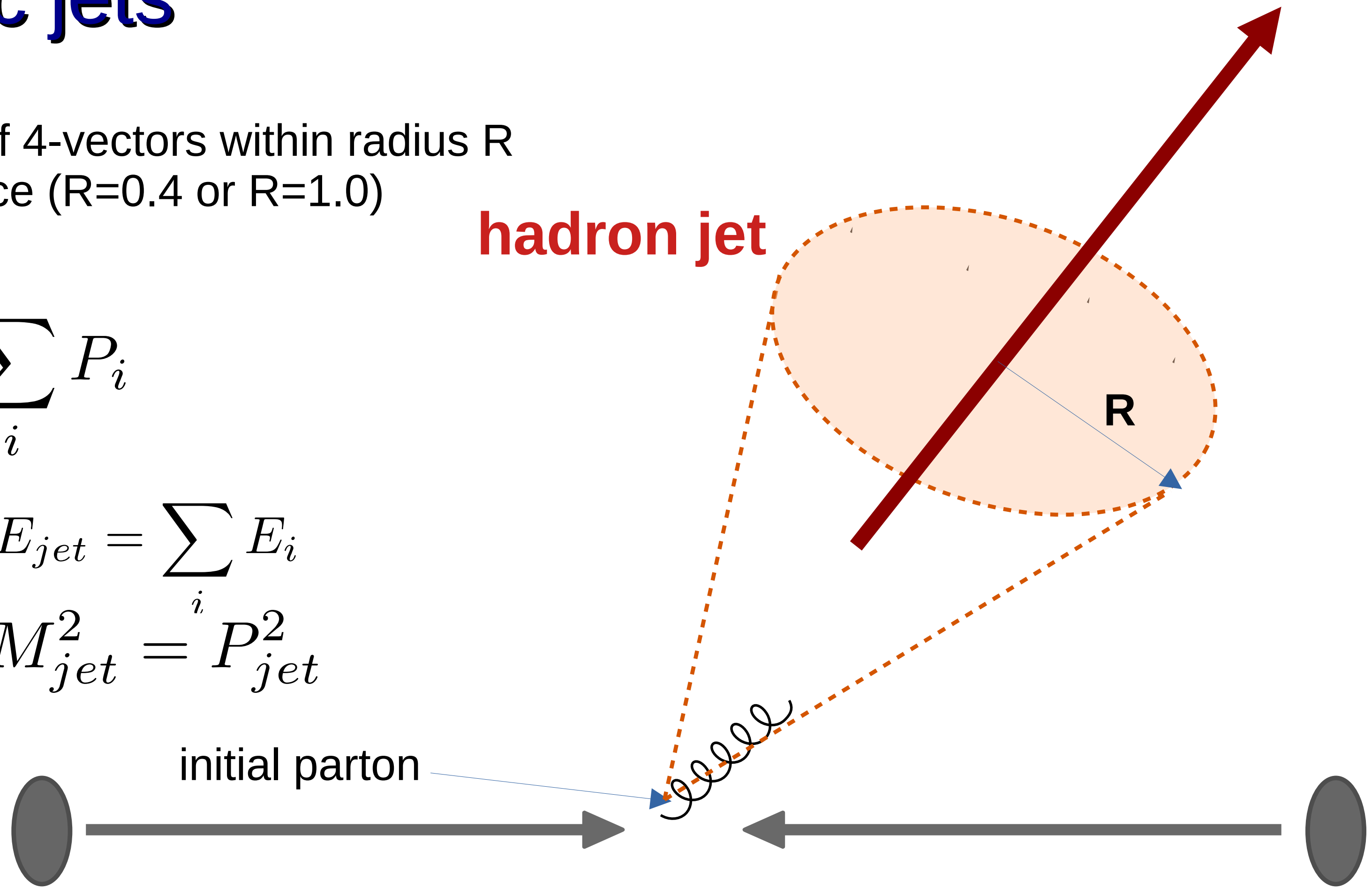
Hadronic jets

Form groups of 4-vectors within radius R in angular space (R=0.4 or R=1.0)

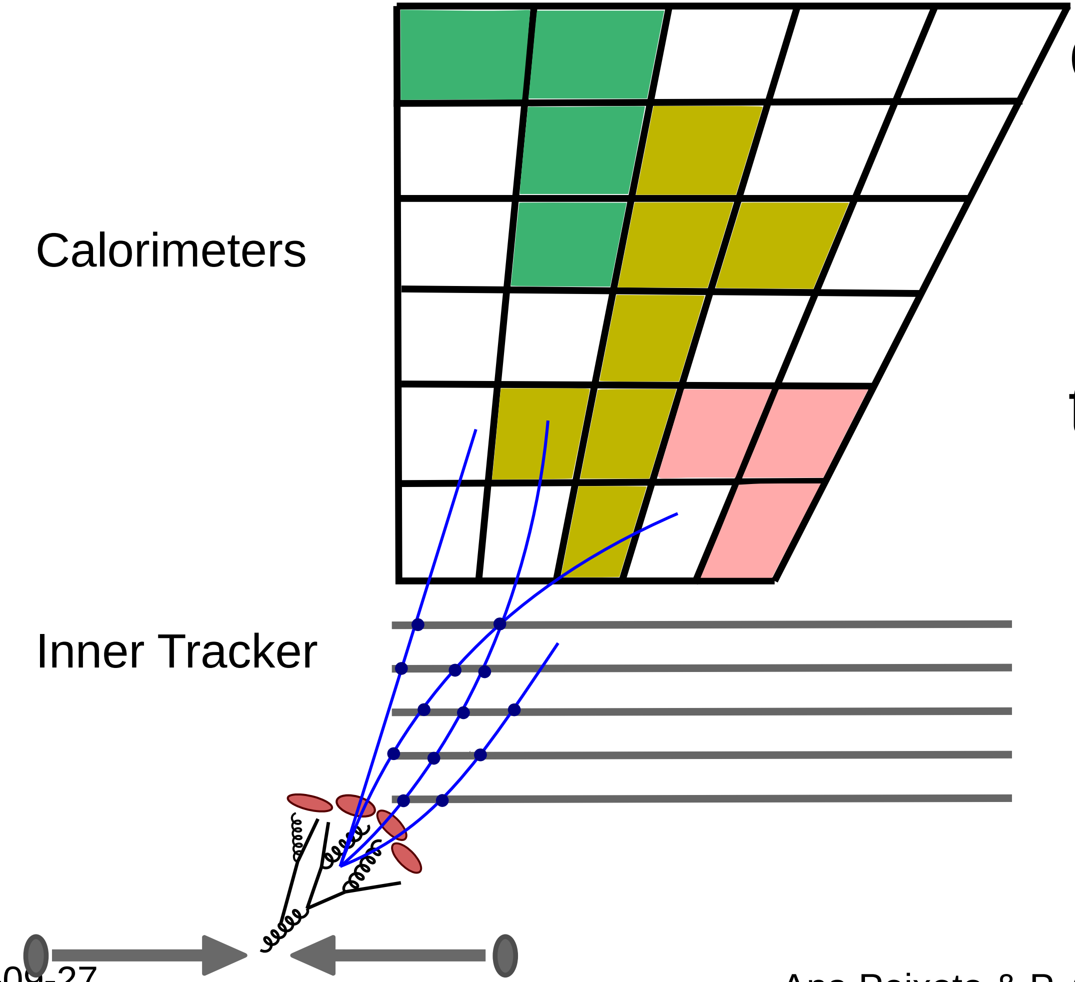
$$P_{jet} = \sum_i P_i$$

$$E_{jet} = \sum_i E_i$$

$$M^2_{jet} = P^2_{jet}$$



Hadronic jets



Combine

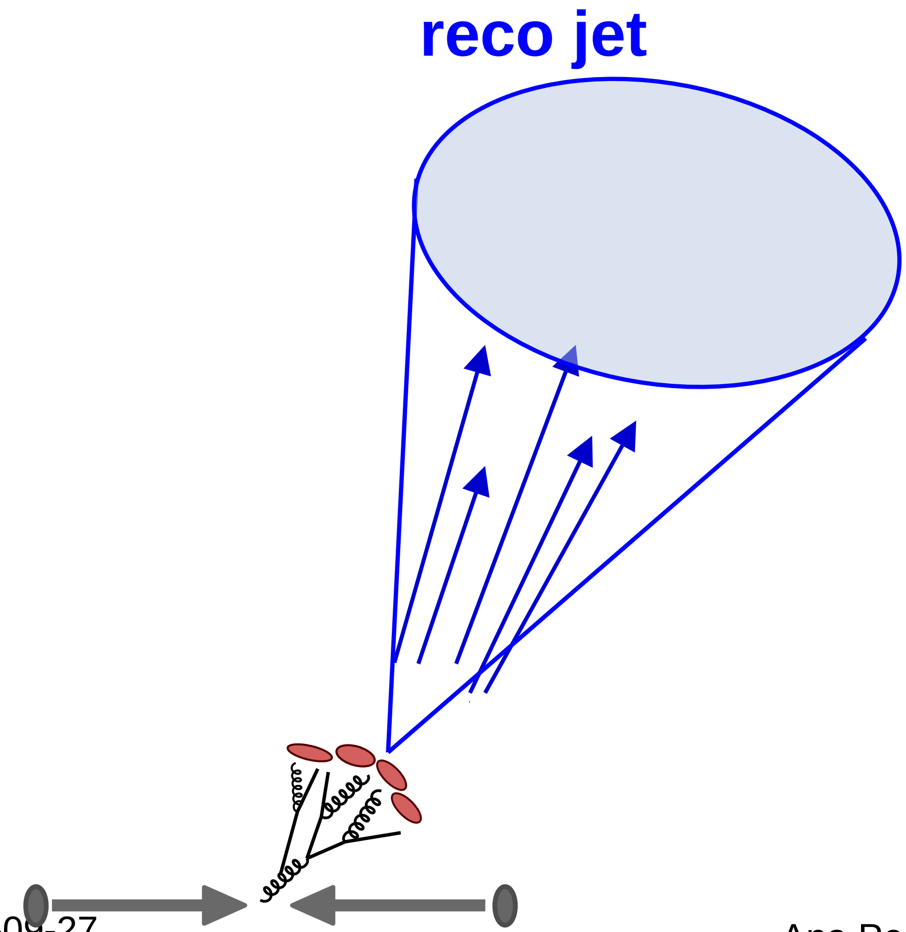
- Tracks
- Calorimeter E clusters

to form

reconstructed 4-vectors

("reco constituent")

Hadronic jets



- No 1-to-1 correspondence between (true) hadrons and reco constituents
  - calorimeter is too coarse
- 1-to-1 correspondence between hadron jets and reco jets
  - Simulation can have reference quantities for reco jets

Measuring jets

- We MUST calibrate jet level quantities : E, mass, angles
- But this is not enough !
  - need precise substructure variables
    - better jet type identification
  - hadron composition matters
    - big source of uncertainties

Crucial for many physics analysis at LHC

Must optimize the energy flow within the jets : constituent calibration

Jet constituent calibration with GNN

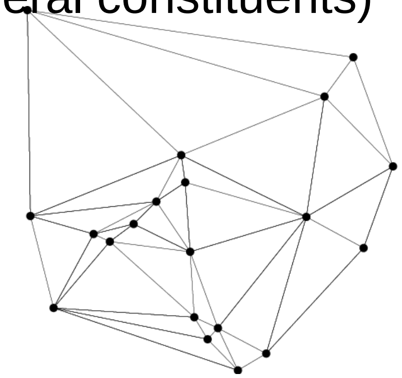
Calibrating jet constituents

- There is no 1-to-1 correspondence between reco constituents and truth hadrons

What reference to calibrate against ?

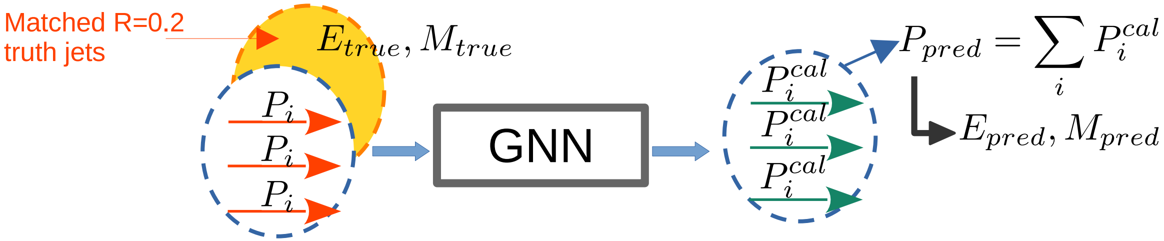
⇒ **jets** : physical objects with a truth reference from MC

- Build very small (R=0.2) jets
  - small enough (fine angular resolution) & big enough (contains several constituents)
- Put jet constituents on a graph
  - nodes == constituents angular position
  - account for spatial proximity between constituents
  - (graph from Delauney triangulation)



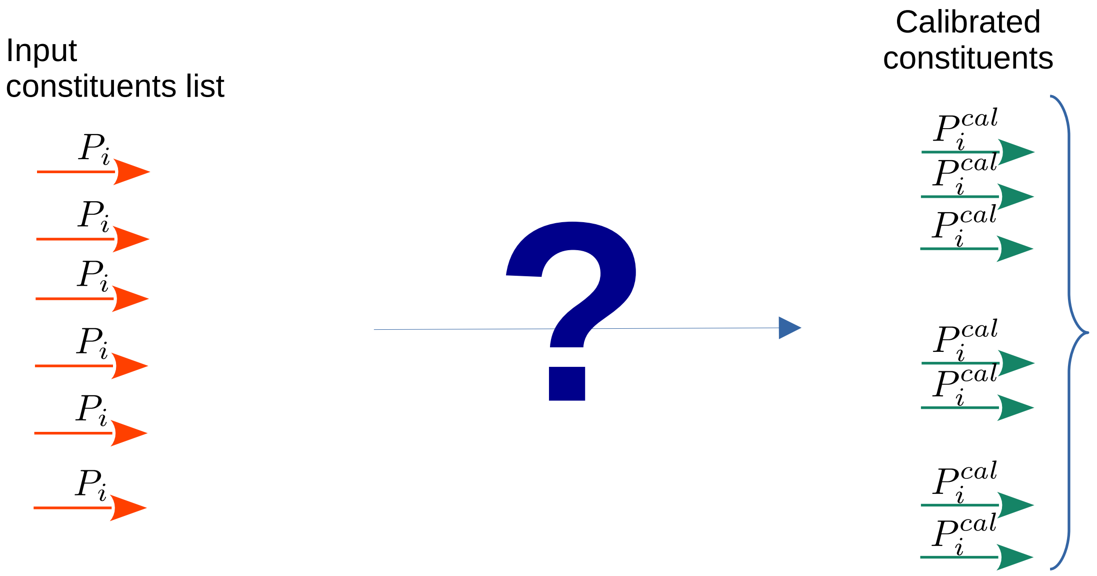
Graph Neural Network

- Predict node-level quantities
  - constituent correction factors
- Training on graph-level constraints
  - Loss depends on Jet energy,mass, angles



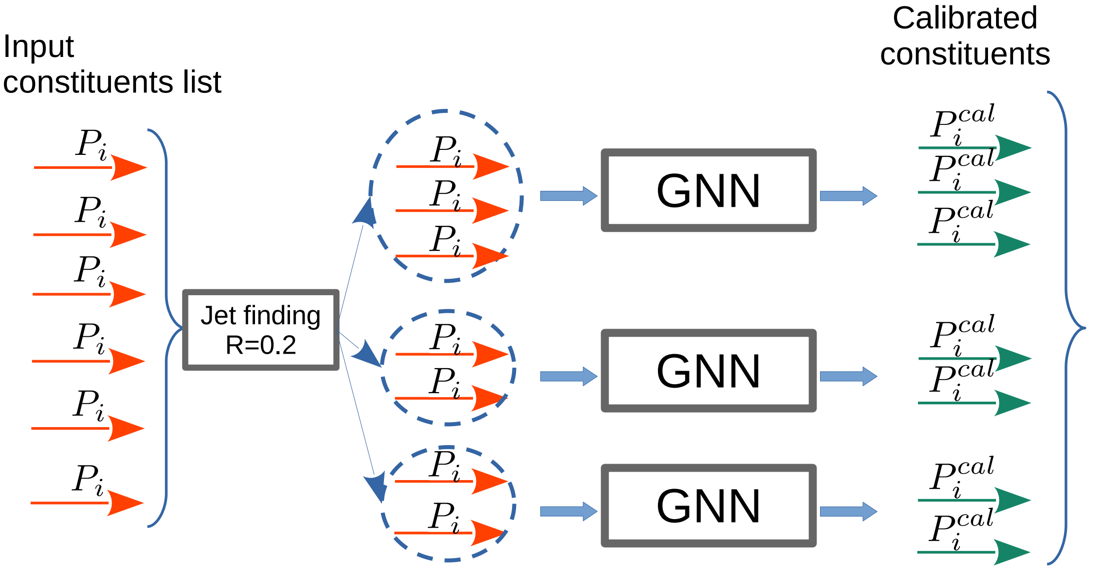
$$\text{Loss} = \lambda_1(E_{pred} - E_{true})^2 + \lambda_2(M_{pred} - M_{true})^2$$

GNN application at collision event level



Use as **input to physic**
small-R or large-R
jet finding...

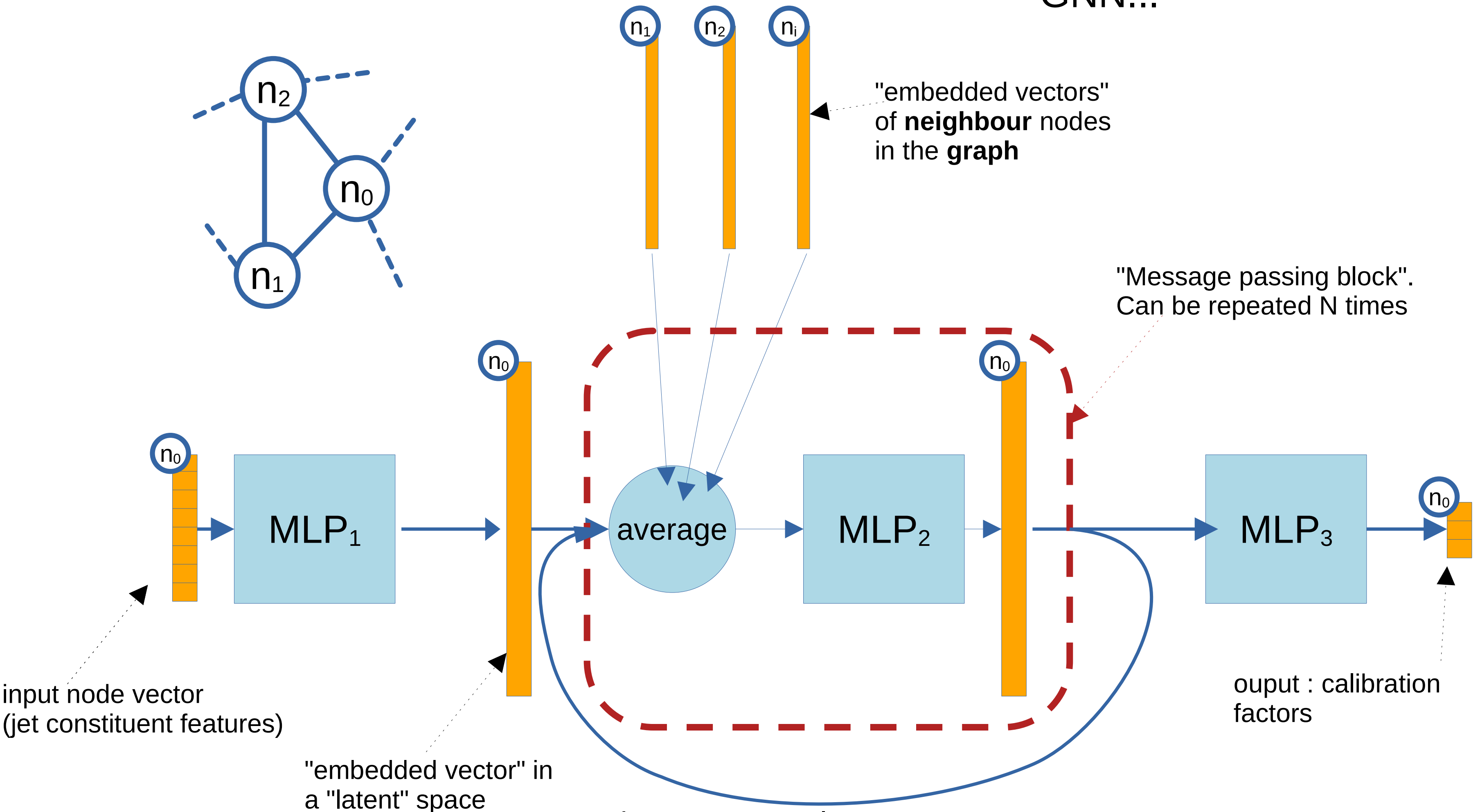
GNN application at collision event level



Use as **input to physic**
small-R or large-R
jet finding...

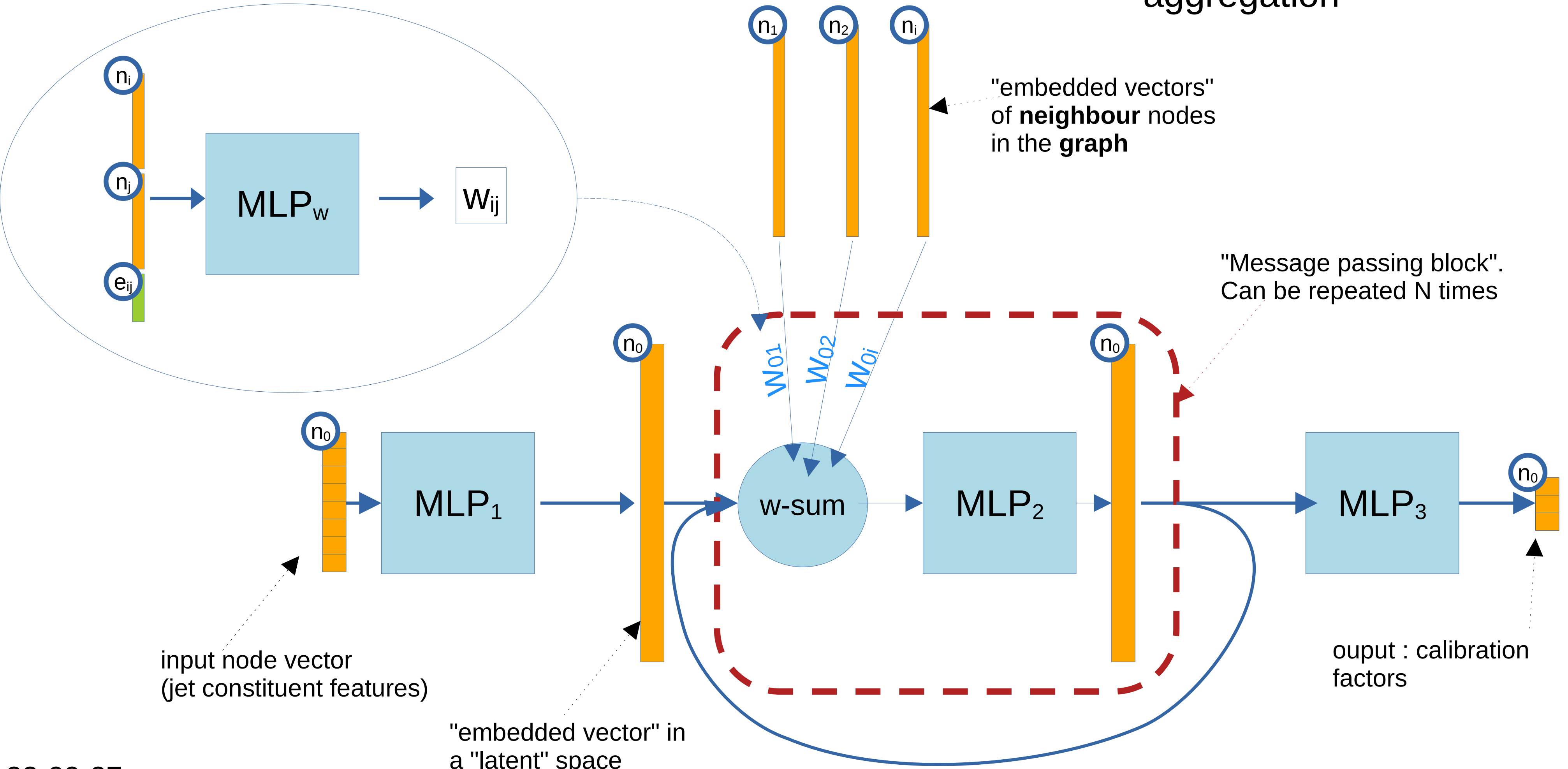
GNN structure

Classical Message Passing GNN...



GNN structure

...with "attention"-like weighted aggregation



GNN setup

- Use QCD di-jet full simulation events
  - include constituents & jet features + truth jet reference
  - O(500M) graphs available for training
- Graph sizes vary greatly : from 1 to ~50 nodes
- Features :
  - ~15 node features : constituent kinematics+detector info
  - ~10 graph features : jet kinematics & variables+ evt info
  - (1 edge features : angular distance)

GNN setup, technicalities

- Framework : own GNN code build on keras/tensorflow
  - graph structure represented by arrays of indices of edges&Nodes
  - using TF's "segment" functions
    - ex: tf.unsorted_segment_sum(data, sumIndices, N)
- Data flow : custom solution
  - O(100M) examples x N features > available memory
  - ROOT ntuple → read by uproot → numpy array → tensorflow
  - Other better technical solutions ?
- Computing : using CC-IN2P3 GPU farm
  - NN convergence in ~few hours

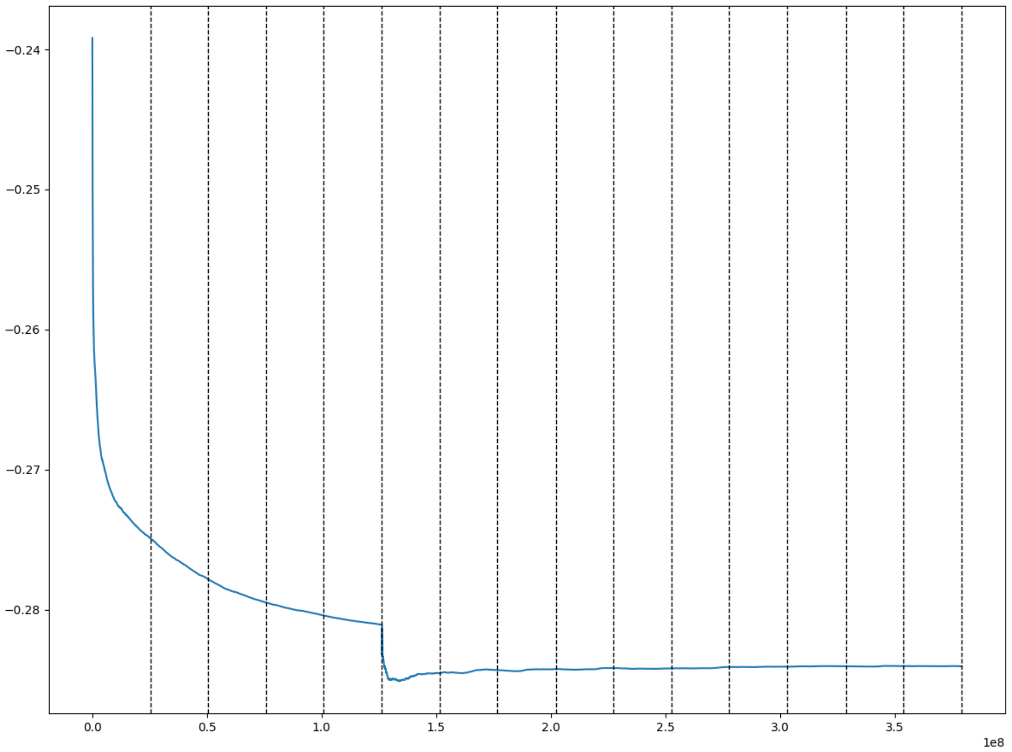
First Results – Loss function

For now, only **Energy** and **rapidity** corrections considered

- Smooth convergence with **LGK** loss

$$L = \beta \left|1 - \frac{E_{pred}}{E_{true}}\right| + e^{-(1-\frac{E_{pred}}{E_{true}})^2/2\alpha}$$

- plus rapidity term
- **LGK** loss seems better than classic MSE to fit values from a distribution

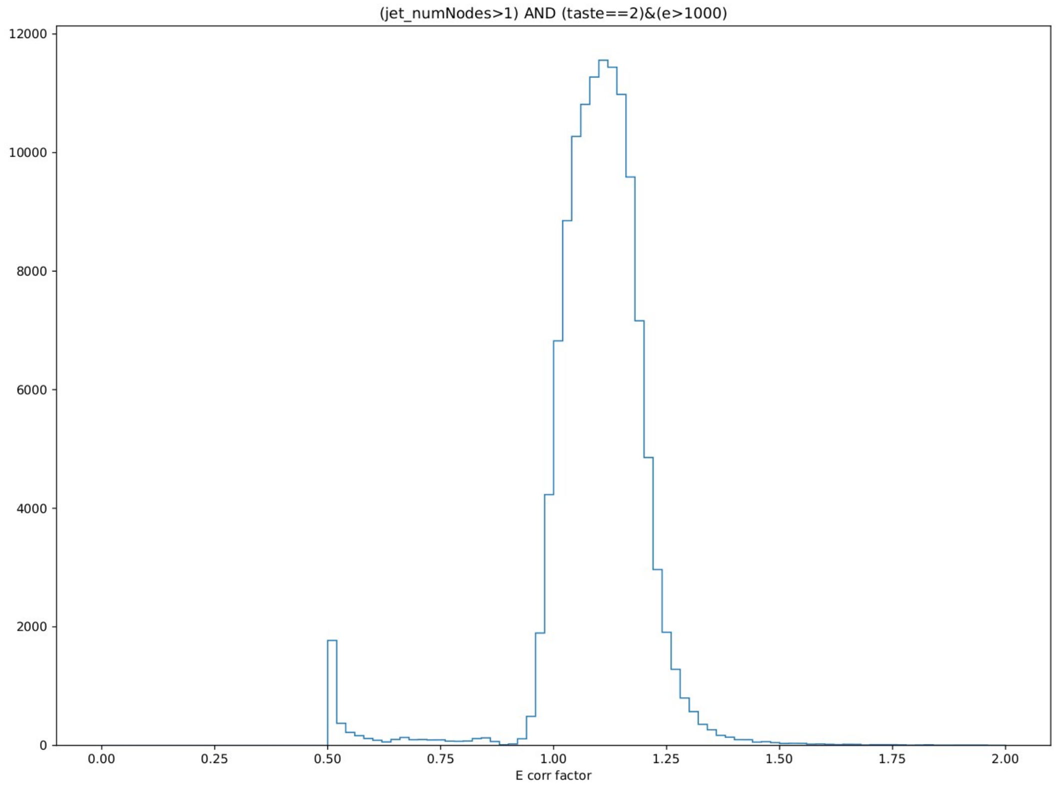


First results – Correction factors

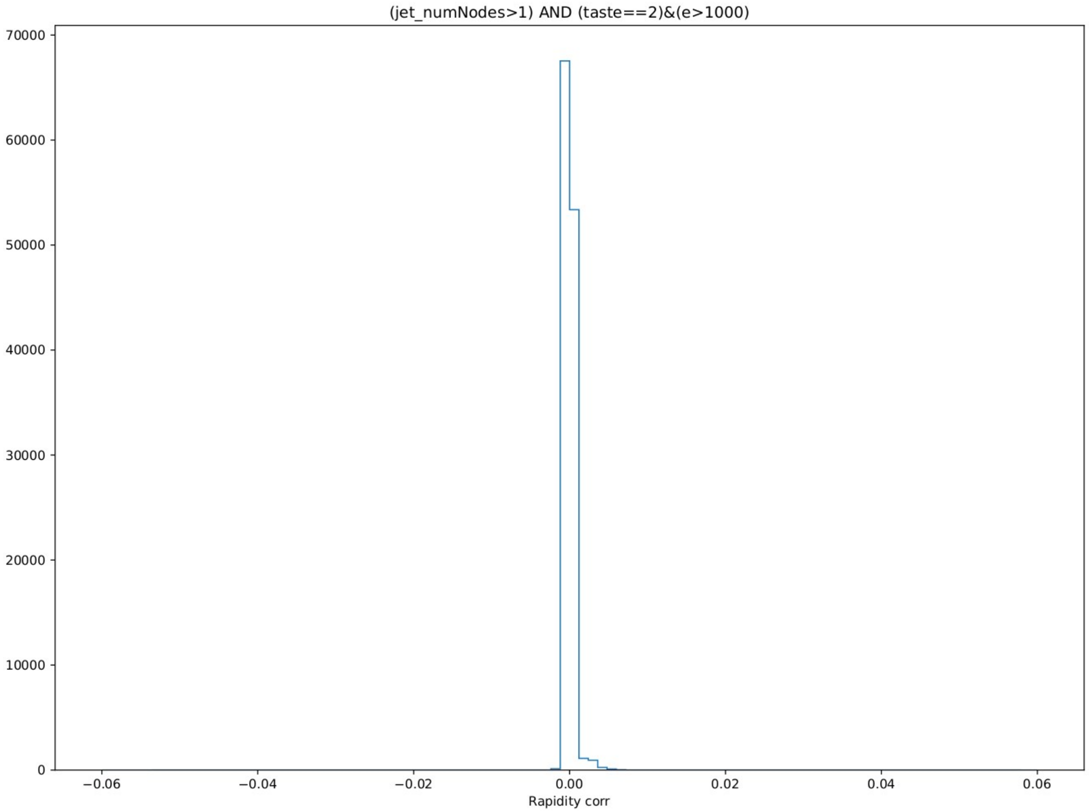
Basic checks performed:

Predicted E scale factors

- Energy correction as expected:
  - Majority around 1.1
- Unexpected fraction at 0.5 (=enforced minimum correction)
  - mostly related with jets with low number of edges or far from truth reference
  - Is the NN correctly suppressing Pile-Up noises ?



First results – Correction factors

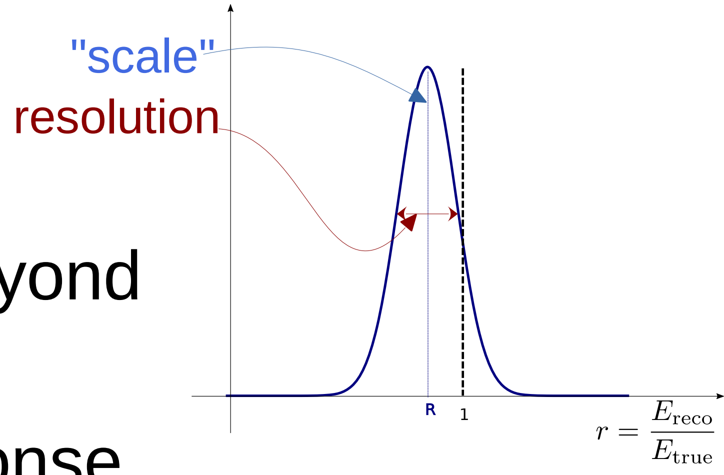


Basic checks performed:

Predicted rapidity corrections

- Low width peak around 0 as expected
  - More epochs -> Narrower peak for rapidity

Performance evaluation

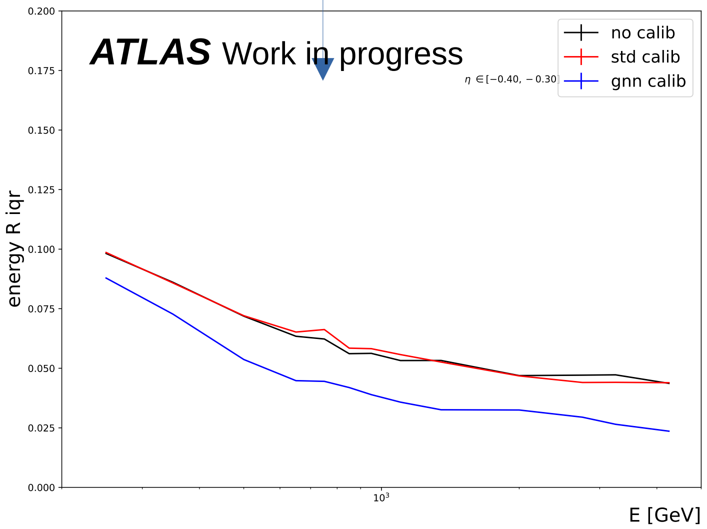


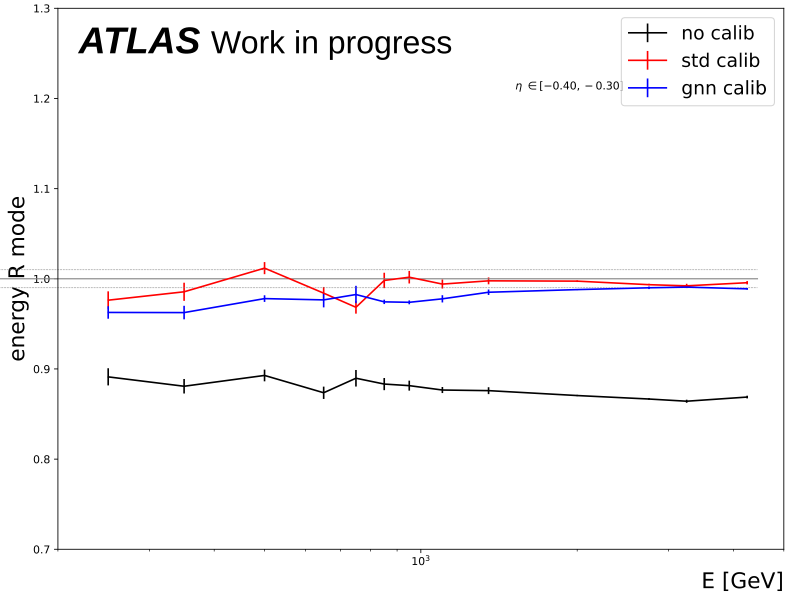
How to evaluate the calib performances beyond the loss ?

- Check physics jets energy & mass response
  - rebuild jets with GNN-calibrated constituents
  - distribution of ratios E_{calib}/E_{true} and M_{calib}/M_{true}
  - consider scale and resolution
- Do this in many E and/or M bins
  - then plot scale vs bin center

E calibration in physics R=1.0 jets

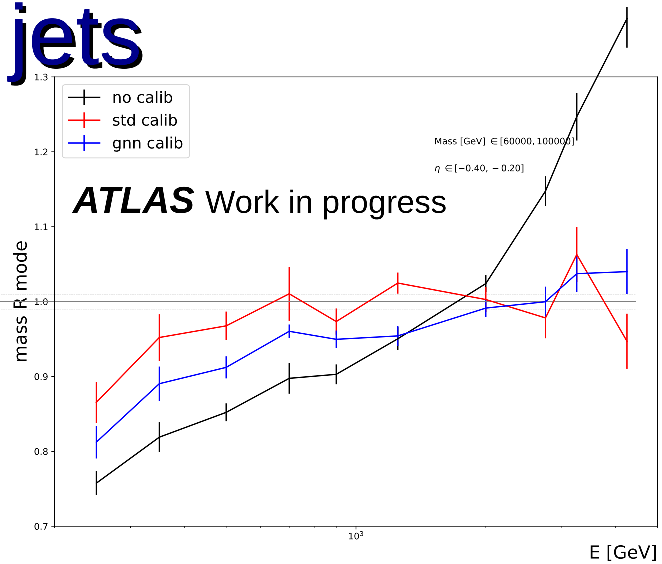
- Energy scale is well reconstructed
  - almost as well as standard ATLAS calib
- E resolution is improved
- some other rapidity (~η) bins are more difficult



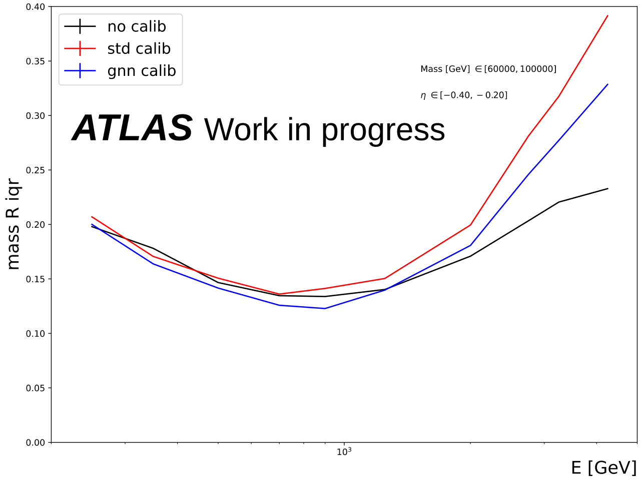


Mass calibration in physics R=1.0 jets

- Mass scale is improved w.r.t no calib
  - NOT just due to E scaling !
  - GNN learnt more



- M resolution is improved w.r.t std ATLAS calib
  - specially at high mass



Conclusions

- Hadronic calibration based on graphs of constituents from small jets
- GNN trained to do node-level regression from graph-level constraints
- Promising results : physics performance comparable to jet-level, dedicated jet calibration
  - just a beginning : many other performance metrics to monitor
- Difficulties & challenges
  - technical ones mostly solved, maybe far from optimal
  - how to improve calib in particular region of phase space ?
  - how to disentangle GNN/ML effects from physics effects ?

Technical details & difficulties

Back-up

Same loss, different behaviour

Same GNNs

- training stopped at different epochs
  - gnn2 has ~15 more epochs
- Loss identical : diff ~0.025%
- YET : response diff ~ 2%