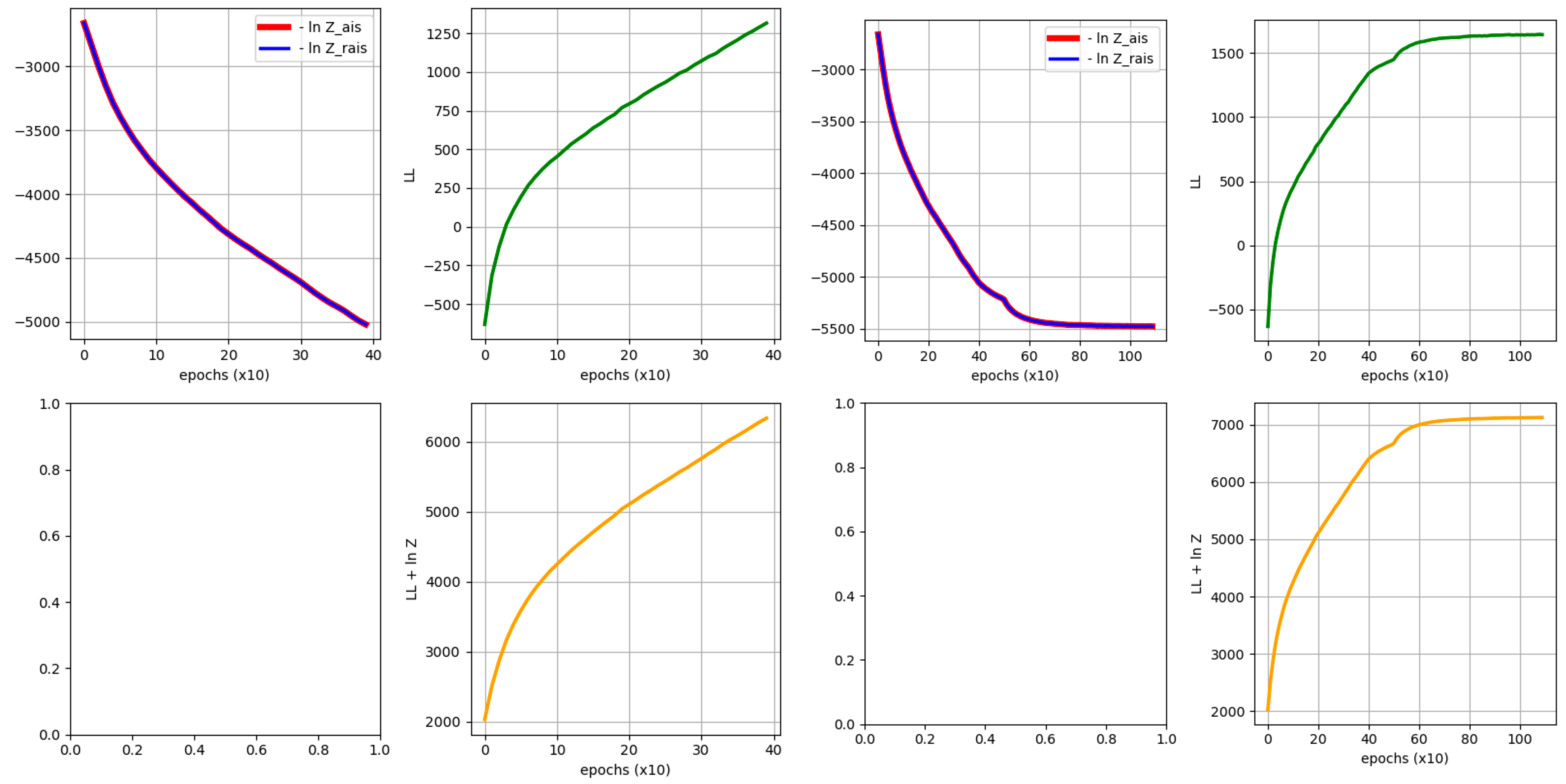
QVAE w/ Pegasus

Jan 15th

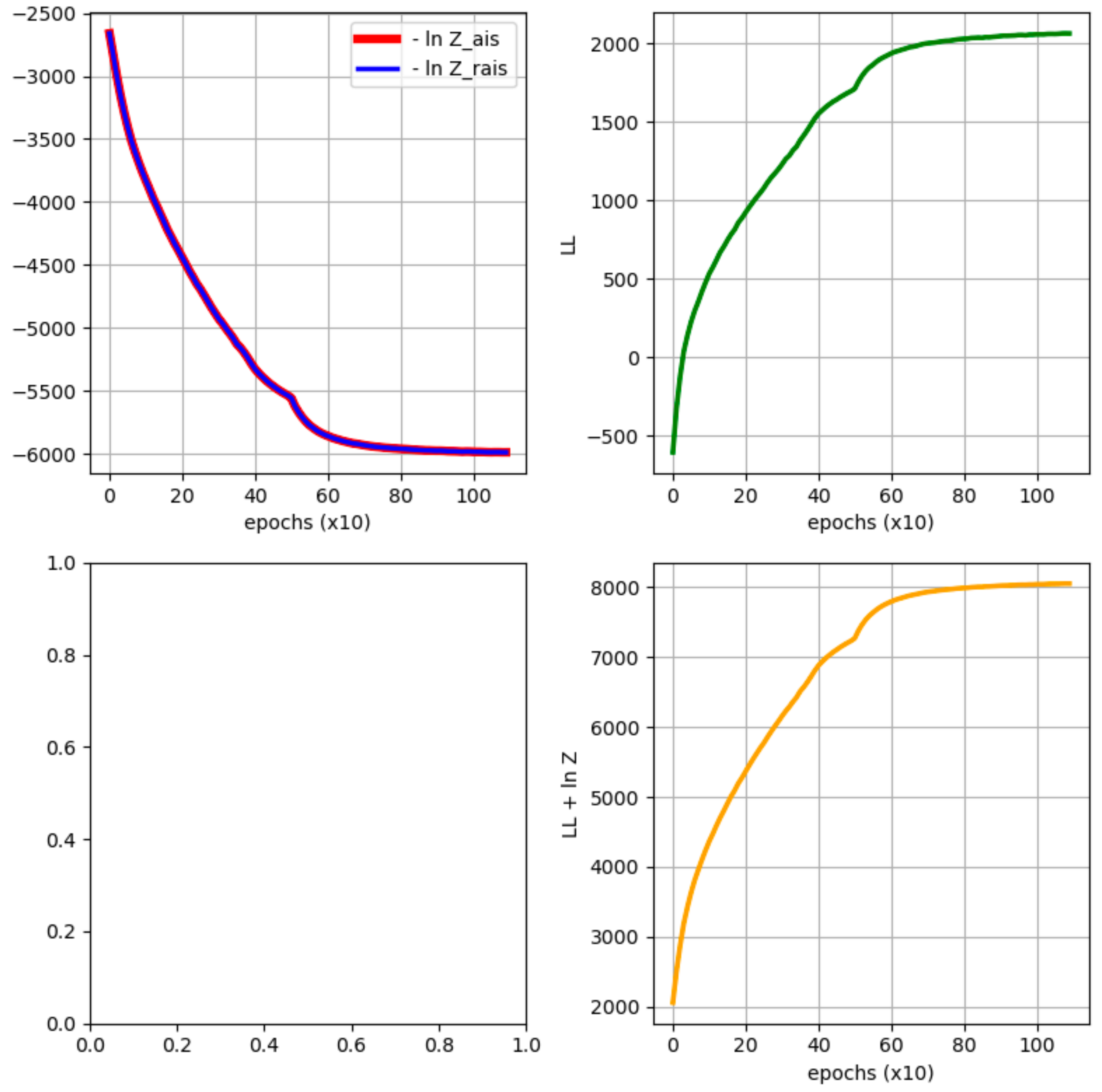
Happy-sun-270

CNN+ cond VAE+posEnc on voxels+scaled data



Drawn-cosmos— 270

CNN+ cond VAE+scaled data



LL not saturating :(

Why?

Possible reasons(?)

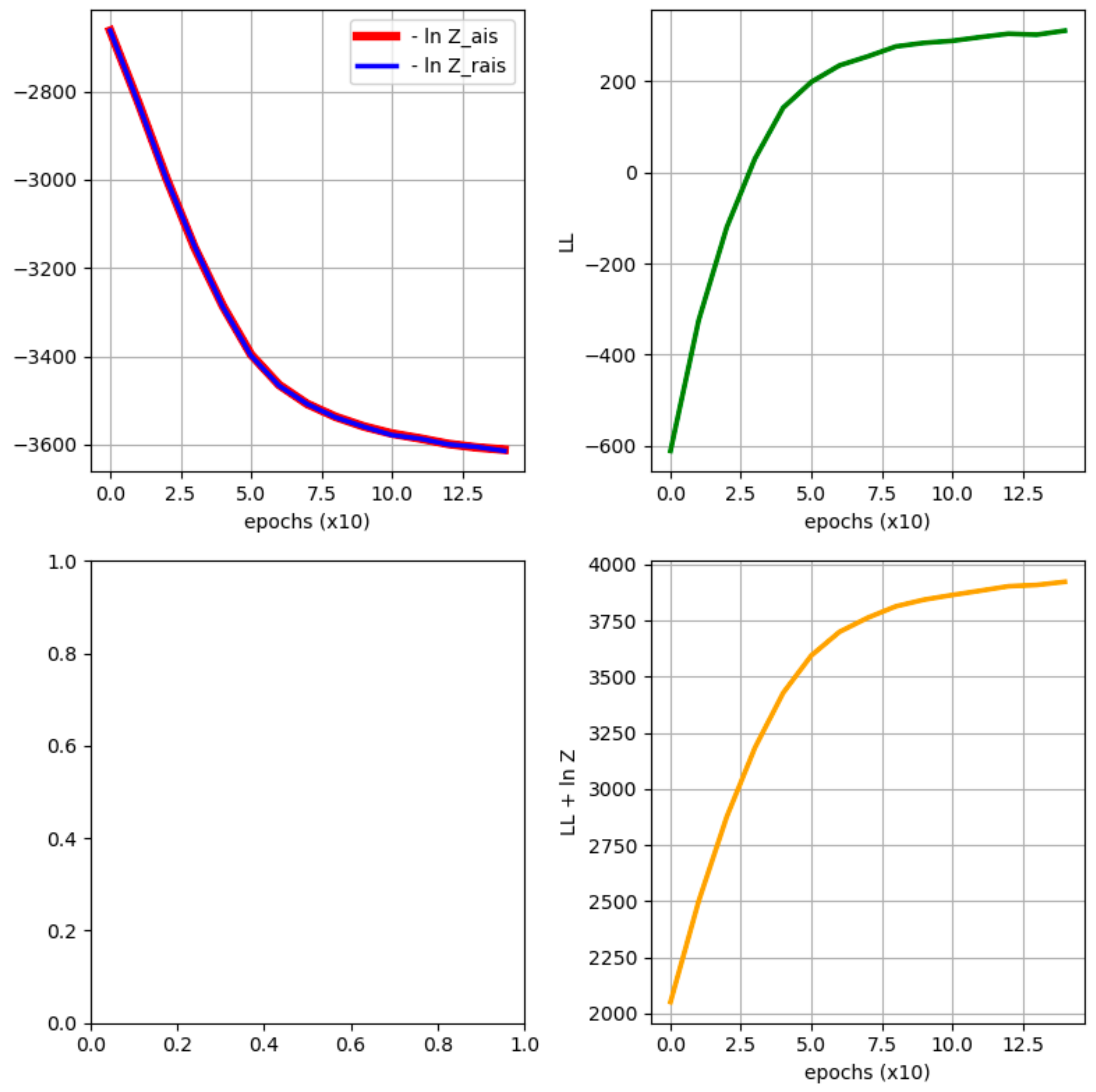
- The encoded data a moving target, perhaps leading to some kind of adversarial learning. ———> Try freezing encoder and decoder.
- Bug in code.
- If none of the above, when can we say our RBM is trained enough?

*Encoded val-set is not static. Need to compute <LL>.

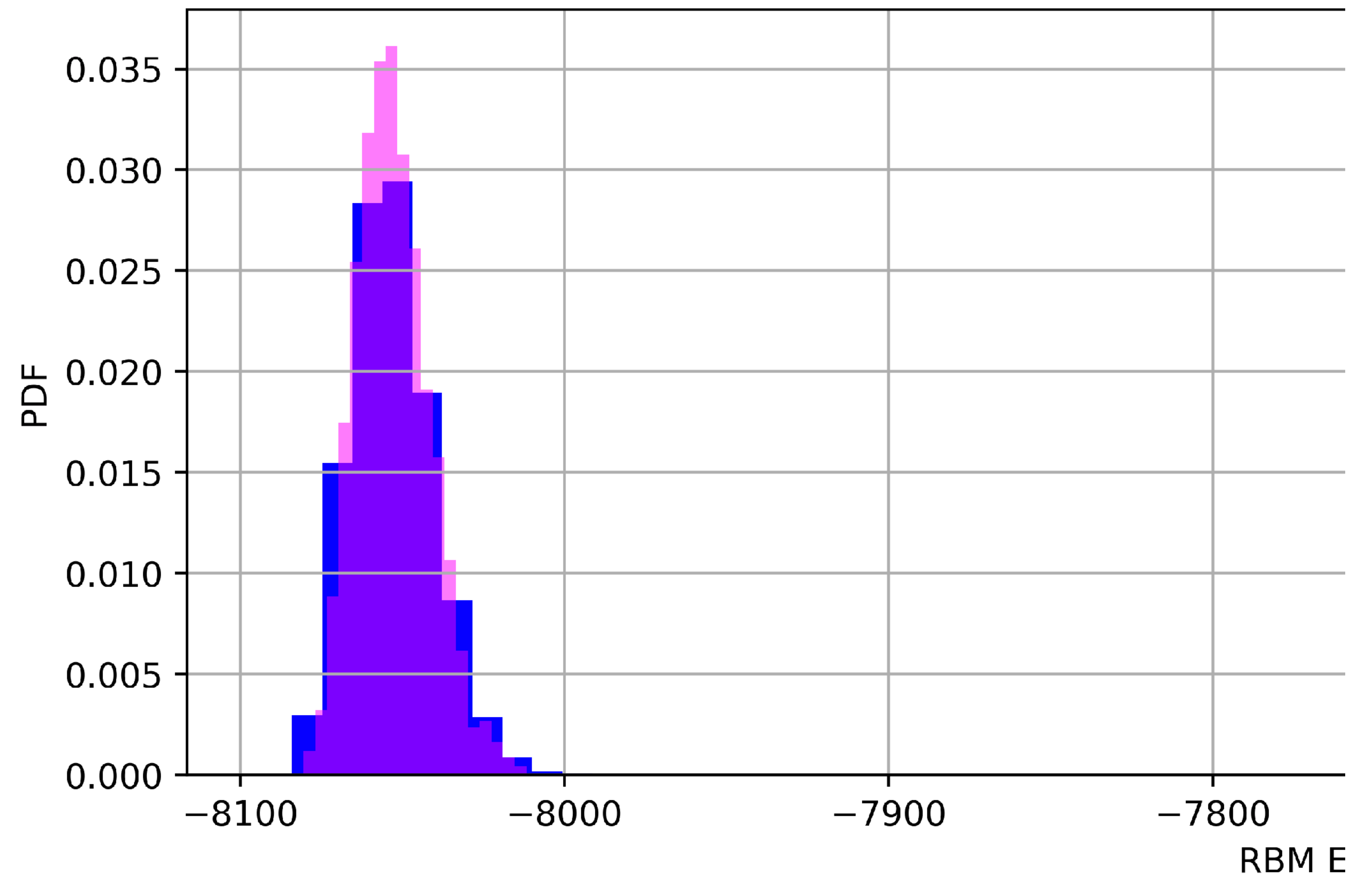
Prime-totem-280

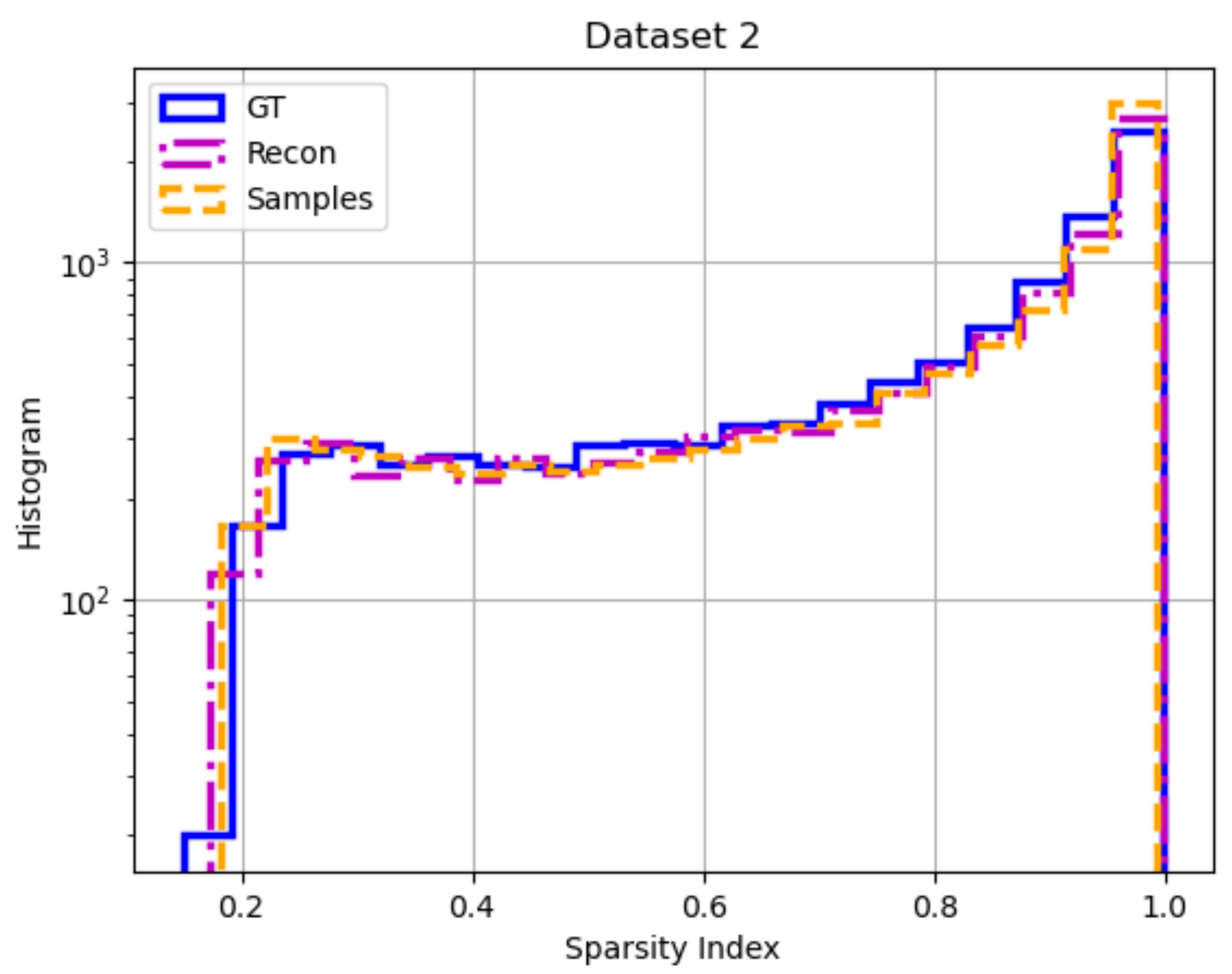
CNN+ cond
VAE+scaled data

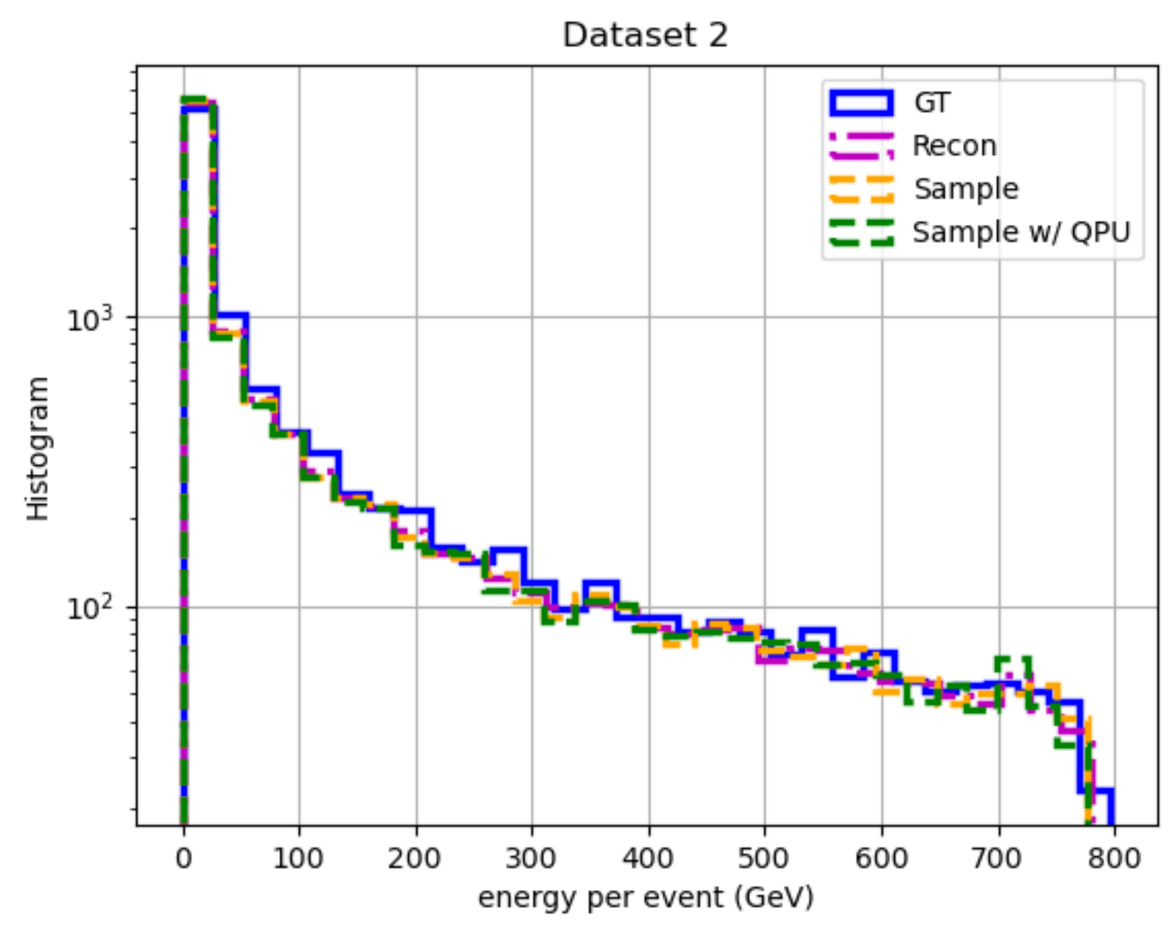
- Trained for 150 epochs.
- First 45 epochs — anneal parameters
- Next 15 epochs —continue training
- Last 90 epochs —train w/ freezed encoder and decoder



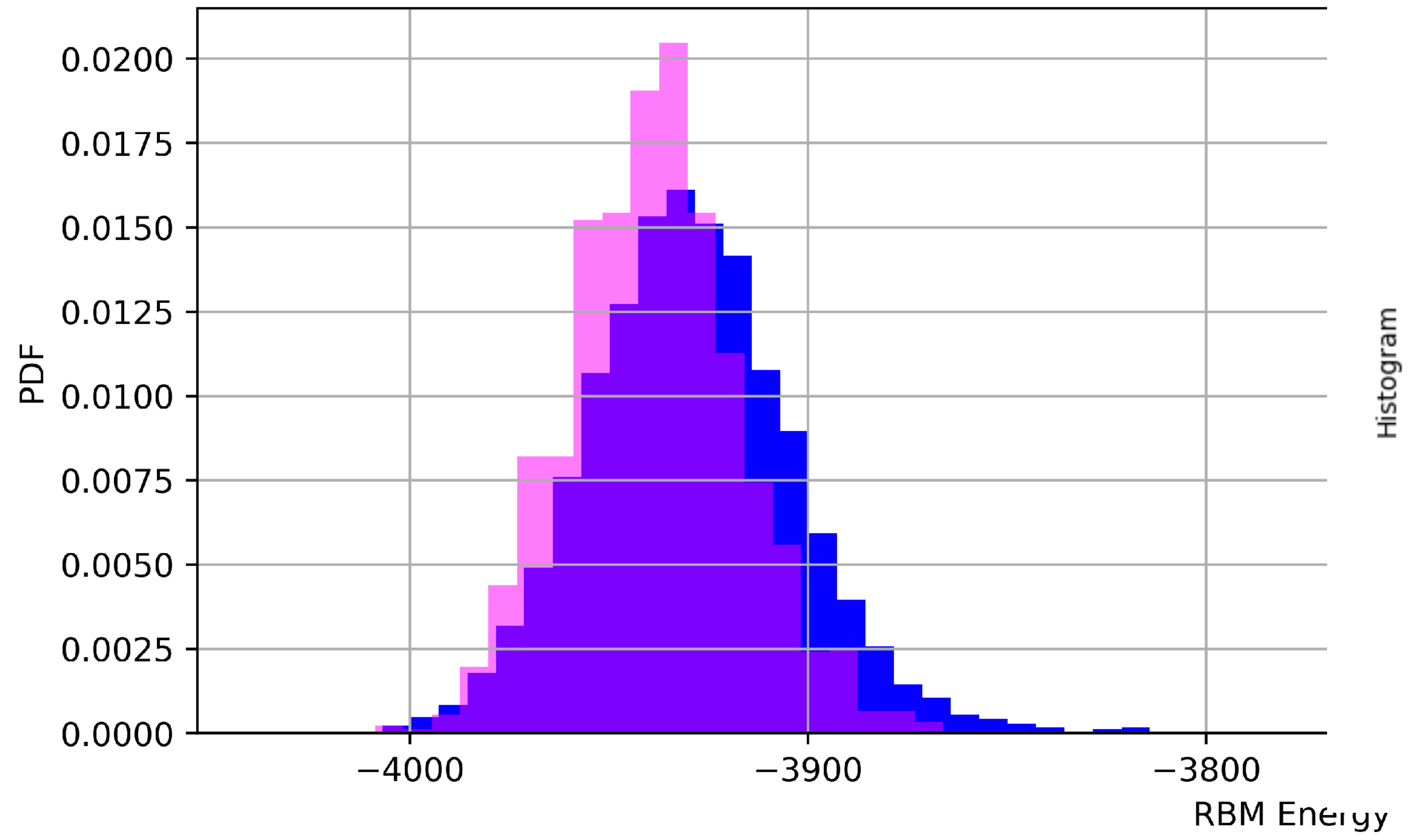
Drawn-cosmos

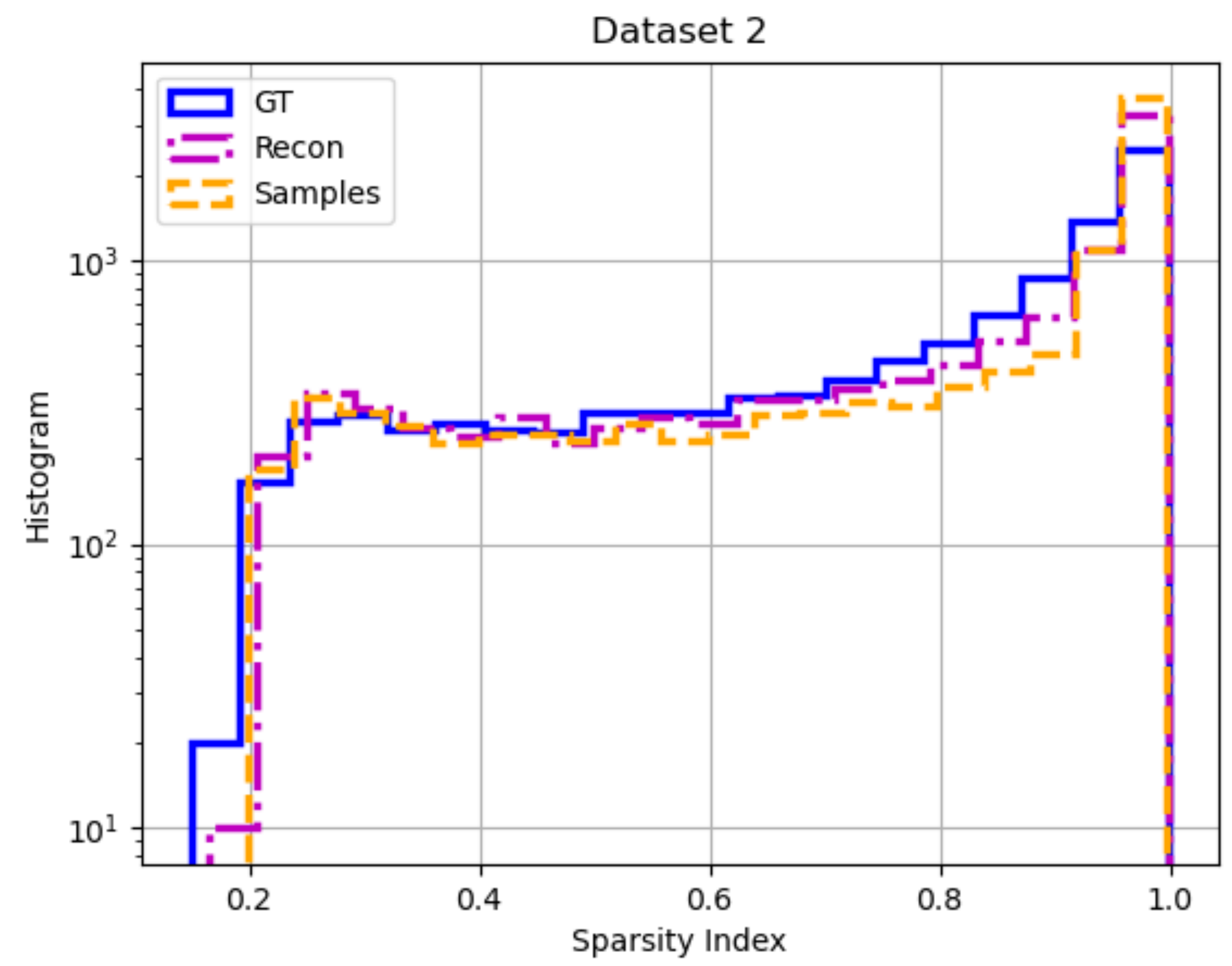


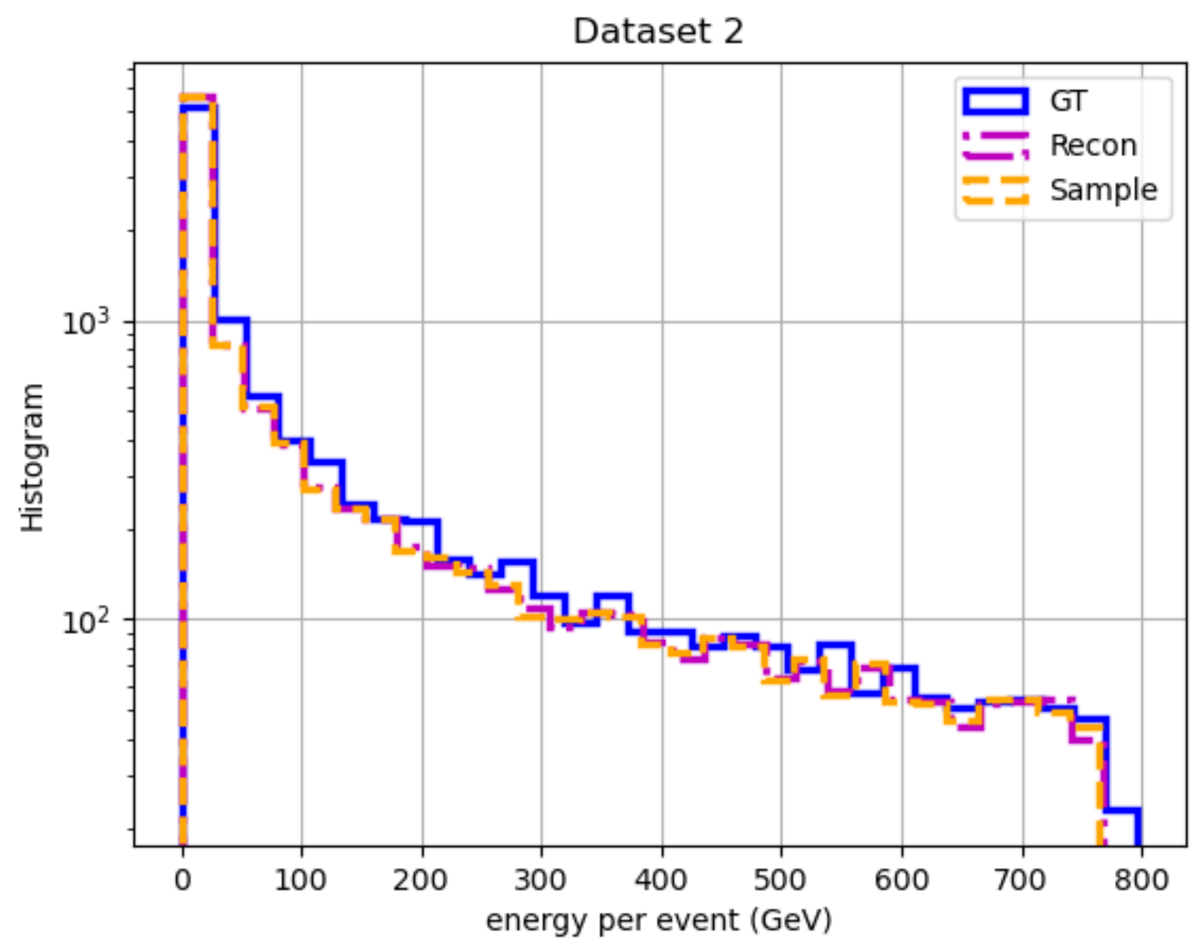




Prime-totem







and S_B as the entropy of QA and B, respectively, and assume $S_{QA} = S_B$, from which after some straightforward algebra:

$$\beta = \beta_{QA}\frac{\langle H\rangle_{QA}}{\langle H\rangle_{B(\beta)}} + \frac{\ln\frac{Z(\beta_{QA})}{Z(\beta)}}{\langle H\rangle_{B(\beta)}} . \tag{E30}$$

We can further simplify the previous expression by introducing the variable $\Delta\beta = \beta_{QA} - \beta$:

$$\beta = \beta_{QA}\frac{\langle H\rangle_{QA}}{\langle H\rangle_{B(\beta)}} + \frac{\ln\langle e^{-\Delta\beta H}\rangle_{B(\beta)}}{\langle H\rangle_{B(\beta)}} . \tag{E31}$$

Notice that the r.h.s. of Eq. (E31) has a fixed point at $\beta = \beta_{QA}$. Here on we will only keep the first term in the r.h.s. and we will show that the fixed point is stable. In addition, same as we did when deriving the previous method, we replace $H(x) \to H(x)/\beta$. Since we do not have any control over β_{QA} nor we know the value *a priori*, we replace the prefactor in the first term of the r.h.s. with β since it does not affect the fixed point value and we further introduce a stability parameter $\delta(> 0)$. After the previous considerations, we propose the following mapping:

$$\beta_{t+1} = f_\delta(\beta_t) \equiv \beta_t\left(\frac{\langle H\rangle_{QA^{(r)}}}{\langle H\rangle_{B(1)}}\right)^{\delta} \tag{E32}$$

The function f_δ has a fixed point at $\beta = \beta_{QA}$. The stability condition close to the fixed point correspond to $|f'_\delta(\beta_{QA})| < 1$. The first derivative at the fixed point yields:

$$|f'_\delta(\beta_{QA})| = \begin{cases} |1 + \frac{\sigma^2_{QA}}{\langle H\rangle_{B(1)}}|, \; \delta = 1 \\ |1 + \delta\frac{\sigma^2_{QA}}{\langle H\rangle_{QA}}|, \; \delta \neq 1 . \end{cases} \tag{E33}$$

In Fig. 3 we have plotted Eq. (E33) *vs* β for different values of δ. The values of β chosen for this plot correspond to where we typically find the fixed point. We call δ a stability parameter since we can tune it to stabilize the mapping per iteration.

From the previous it is easy to notice that the fixed point is unstable when the learning rate, η, such that $\eta > \beta_{QA}/\sigma^2_{QA}$ ($\beta_{QA}/\sigma^2_{QA} \sim 2 \cdot 10^{-2}$).

Appendix F: Data Pre-processing

Prior to training the model, the GEANT4-simulated events where transformed and re-scaled so as to improve the model's capability to learn useful representations for the data. In turn, during inference, we applied the inverse transformation and re-scaling to bring the generated events back to the original data space.

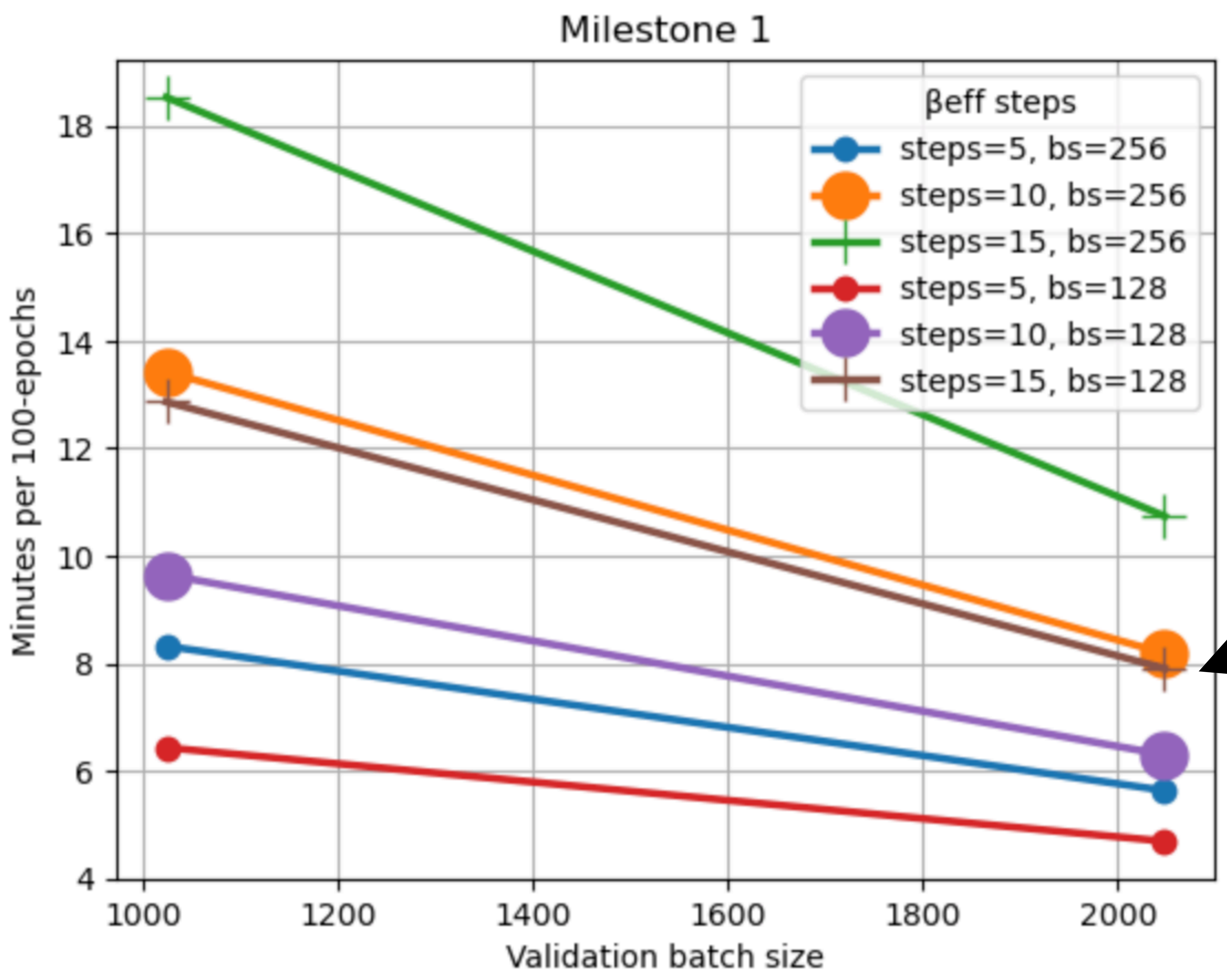
For a particular data set, the transformation was applied on a per-voxel (i.e. column) basis and utilized the per-voxel statistics for all events in the data set.

Consider a data set where each row is a GEANT4 shower-simulated event, the first column of the event represents the first voxel of the first layer, and the last column represents the last voxel of the last layer. We will now outline the steps taken to pre-process a given GEANT4 simulation data set.

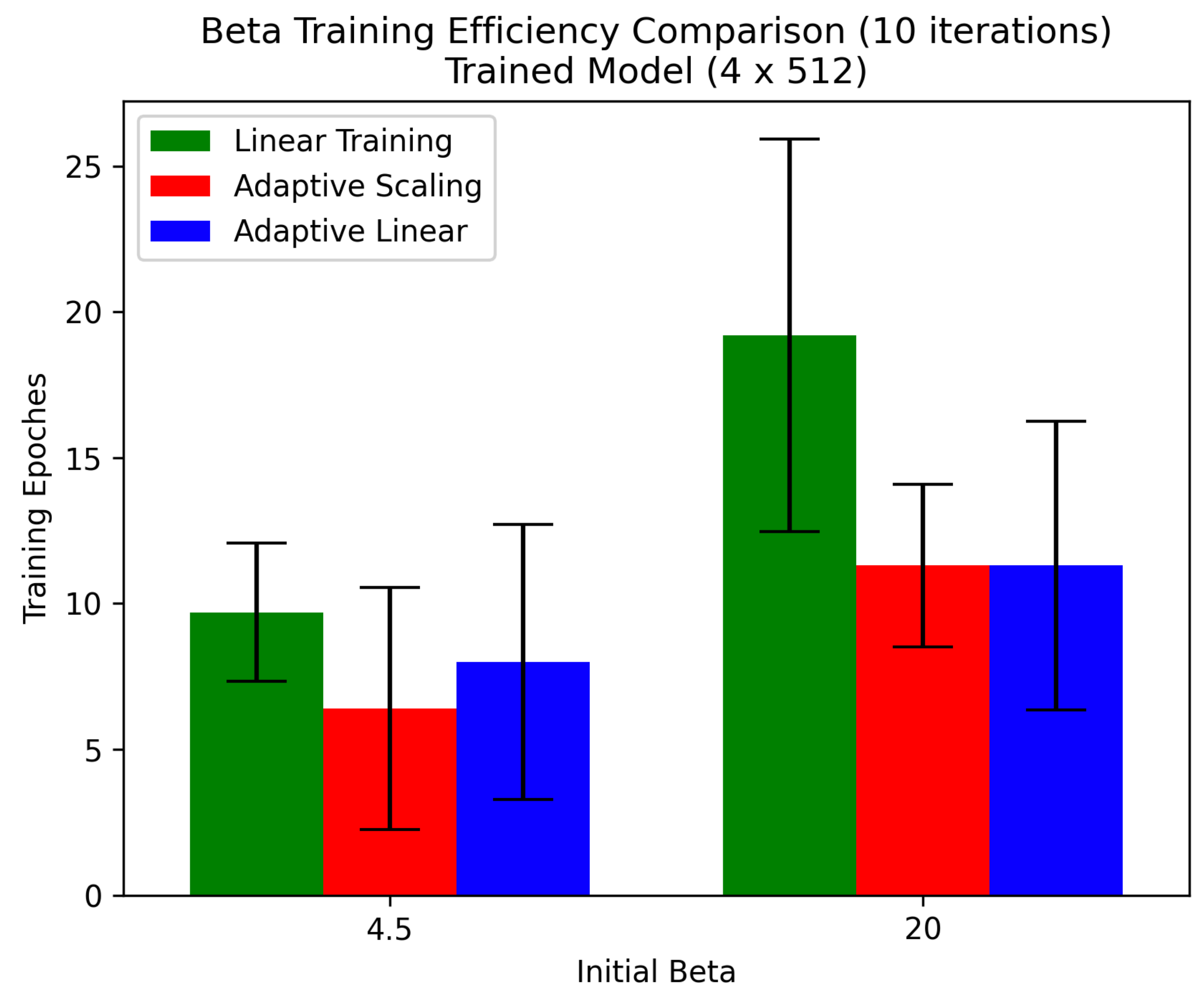
1. For each column, only consider those events where the given voxel energy is non-zero. Normalize the column by the mean and standard deviation of those non-zero events only, while also leaving zero-energy voxels unchanged.

2. Again, excluding zero-energy voxels and leaving them unchanged, find the minimum of the normalized column. Subtract the minimum from and add a small number $\epsilon = 10^{-2}$ to all non-zero energy voxels in the column.

3. Store the column-wise mean, standard deviation, and minimum as these will be needed when applying the inverse transformation to the output of the model.



Train model for 12 epochs using free dwave trial



- PRX paper highlights
  - Architectures
    - CNN
    - FCN
    - 4-partite RBM
  - Energy incidence
    - Condition on encoder and decoder via concat or positional encoding
  - Results/metrics
    - Energy histogram
    - Sparsity histogram
    - Mean energy per r,theta,z
    - Energy distribution for encoded and RBM Gibbs samples
    - Zais and Zrais estimates for partition function => log-likelihood of model
  - Dwave QPU for sampling and validation
  - Method to estimate temperature
    - Sehmi’s method
    - Hao’s method/ adaptive method