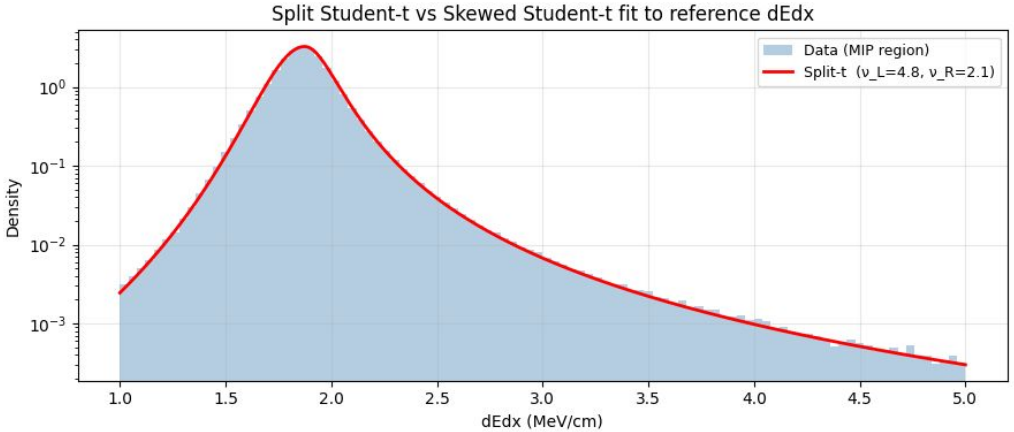
LArTPC calibration with real-like fake-data

- How to do calibration when you don't know the underlying truth?
- Through-going cosmic muons will be MIPs. Their energy deposition rate in the detector stays in average constant



- Taking only the MPV is not enough, e.g. Ab/kb degeneracies
- Randomly drawing from the distribution is even worse

→ Treating the track parameters as nuisance parameters in the fit: they are co-optimized with the calibration parameters

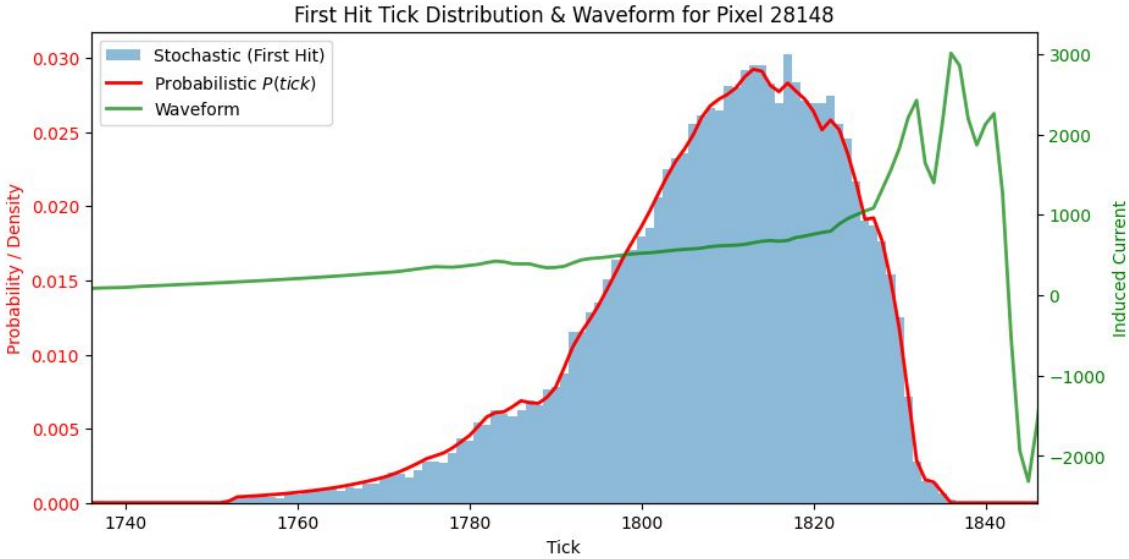
The setup

Target:

- Target calib. params
- True **dEdx** depositions
- Stochastic draw of noise

Fitted guess:

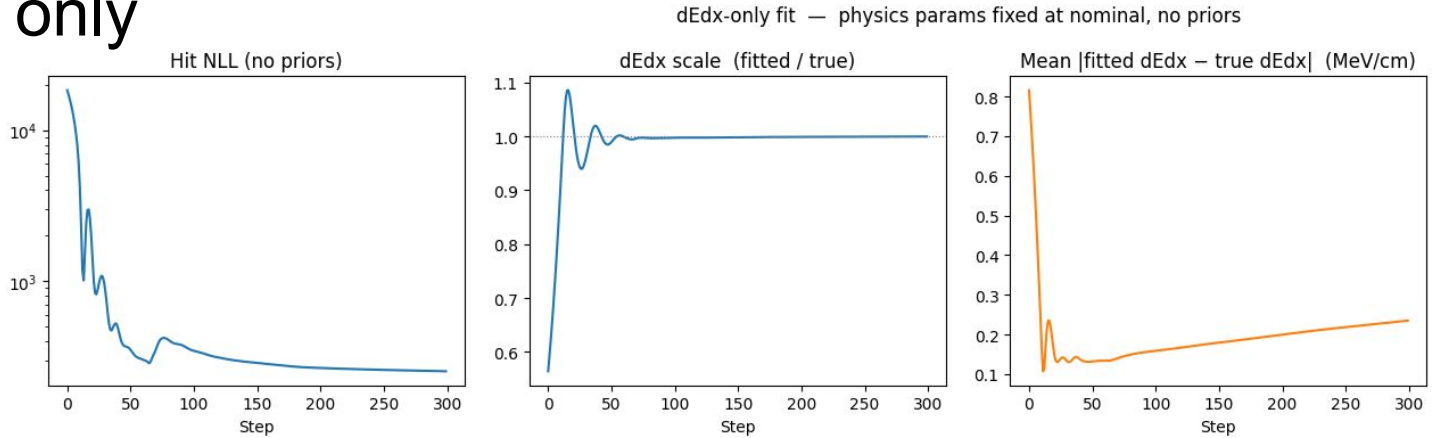
- Fitted calib. params
- Fitted **dEdx**
- Probabilistic estimate of output given noise level



Loss: Poisson Point Process (PPP)

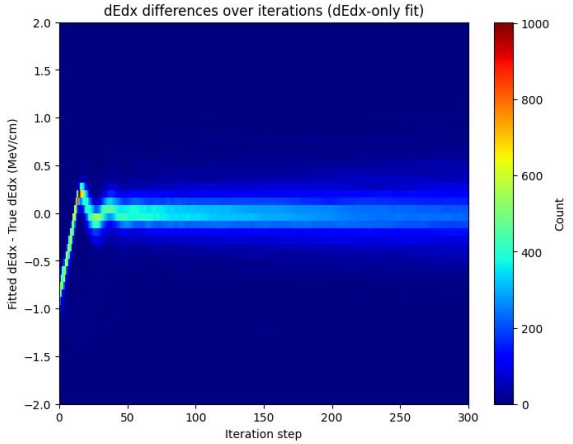
$$\mathcal{L}(\theta) = \sum_{p \in \text{pixels}} \int \lambda_p(t), dt - \sum_{i \in \text{hits}} \max \left[\log \left(\lambda_{p_i}(t_i) \cdot \frac{1}{\sqrt{2\pi\sigma_q^2}} e^{-\frac{(q_i - \hat{q}_{p_i}(t_i))^2}{2\sigma_q^2}} \right), \log \epsilon \right]$$

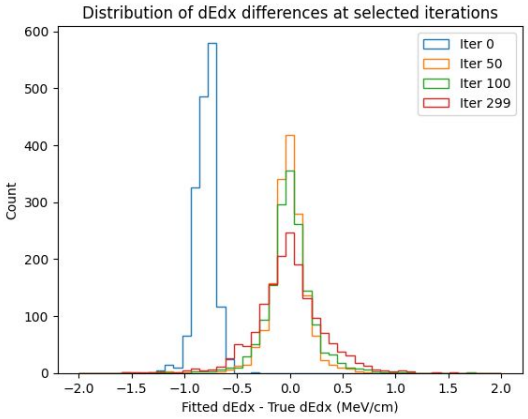
Fitting **dEdx** only



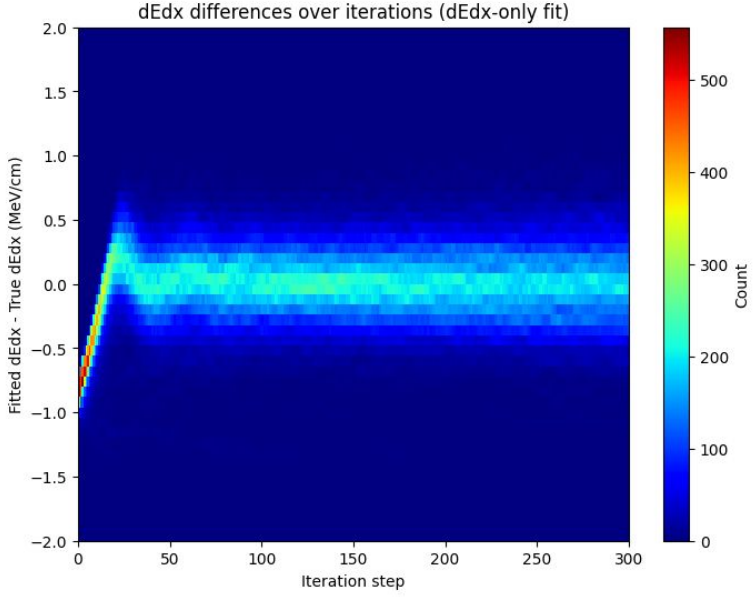
Fitting the chopped segments **dEdx**: works well but overfitting to fit noise as we continue the fit.

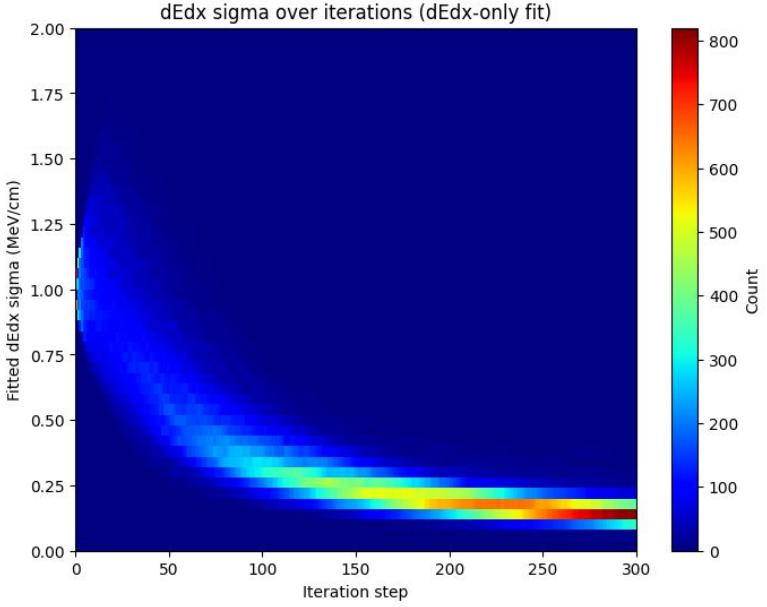
In practice cheating with lo-hi compensations on neighboring segments





Fitting **dEdx** only: adding some noise to prevent over-fitting





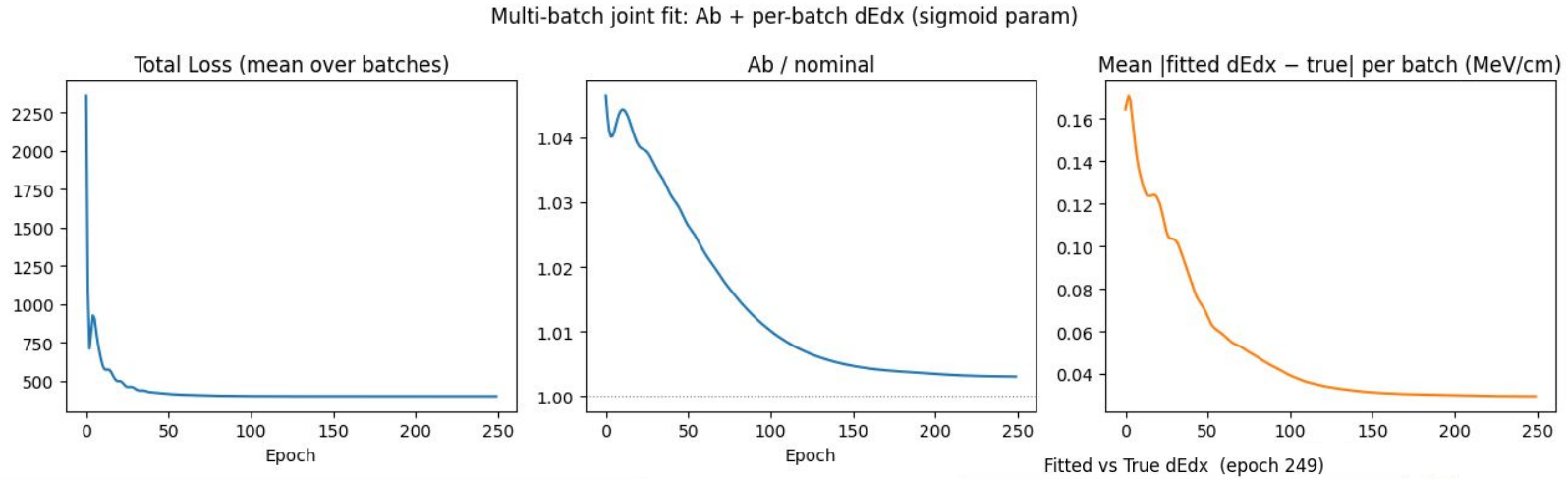
Adding random noise along the fit to prevent overfitting:

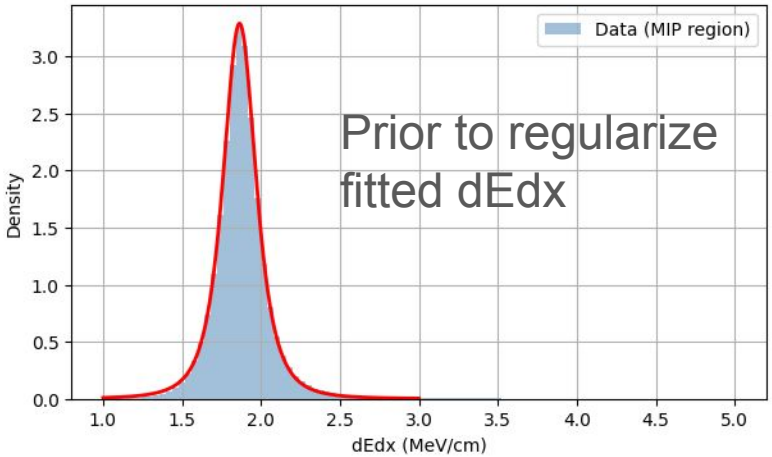
$$\frac{dE}{dx} \rightarrow \mu + \sigma\mathcal{N}(0,1)$$

$$\mathcal{L} \rightarrow \mathcal{L} - \sum \log\sigma$$

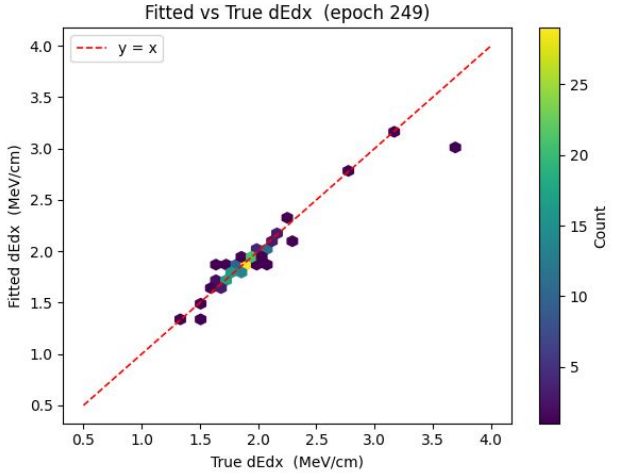
Actually, only fitting the original bigger segments also solves the issue and is easier…

Joint calibration/**dEdx** fit

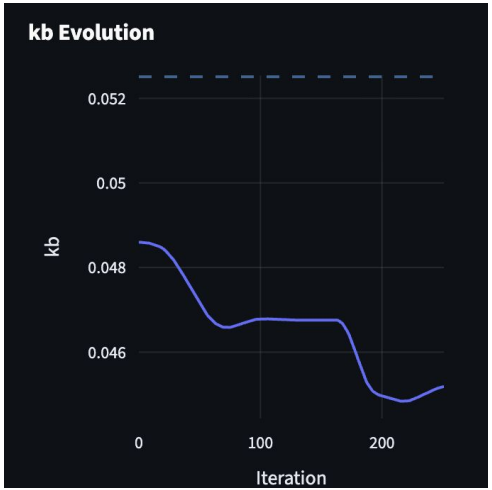


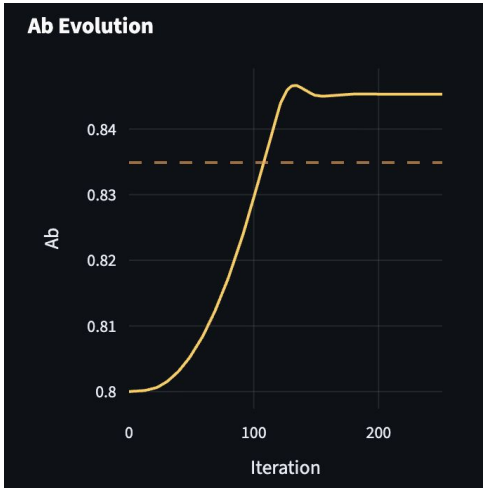


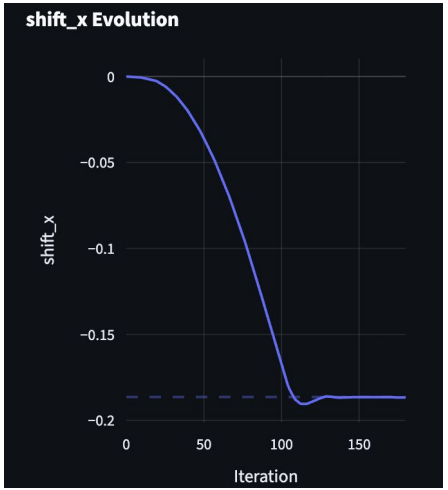
Prior to regularize fitted dEdx



In practice fits are not that good

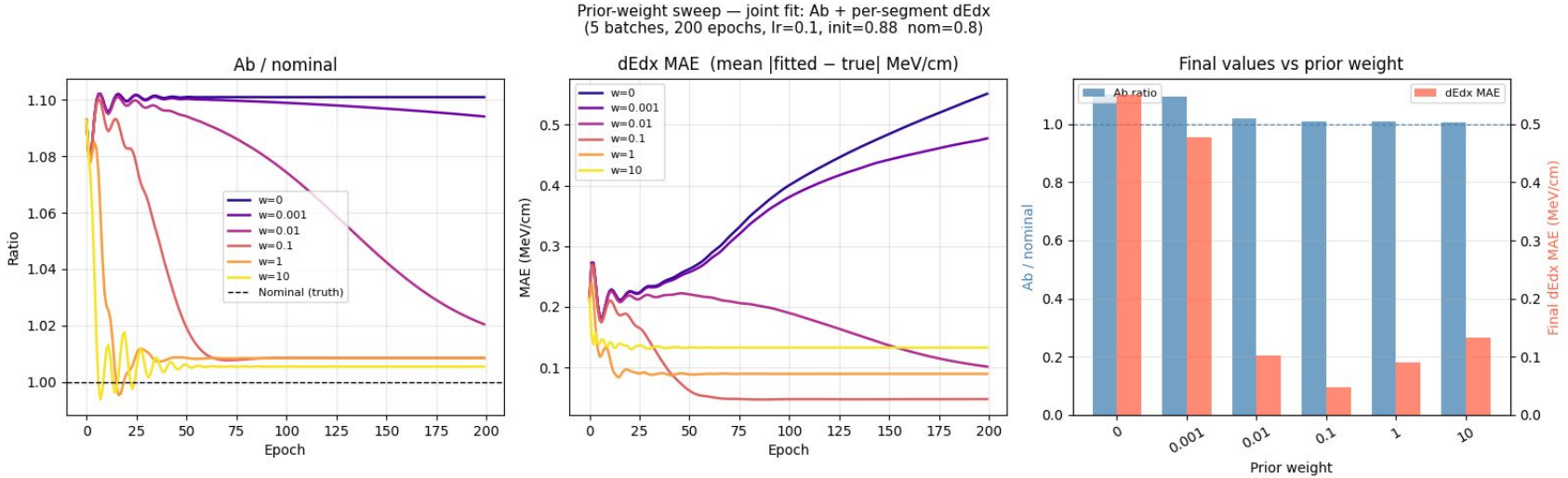






Getting some ~ 10% error on some parameters, which is not really great (especially that currently I still assume the true segment positions)

The relative weights of the loss terms matter



- Depending on the choice of **w** we either give more importance to:
  - penalizing differences in the hit output
  - penalizing differences in the **dEdx** distribution
- High-w ~ fixing **dEdx** to MPV ; Low-w ~ no physics prior on **dEdx**

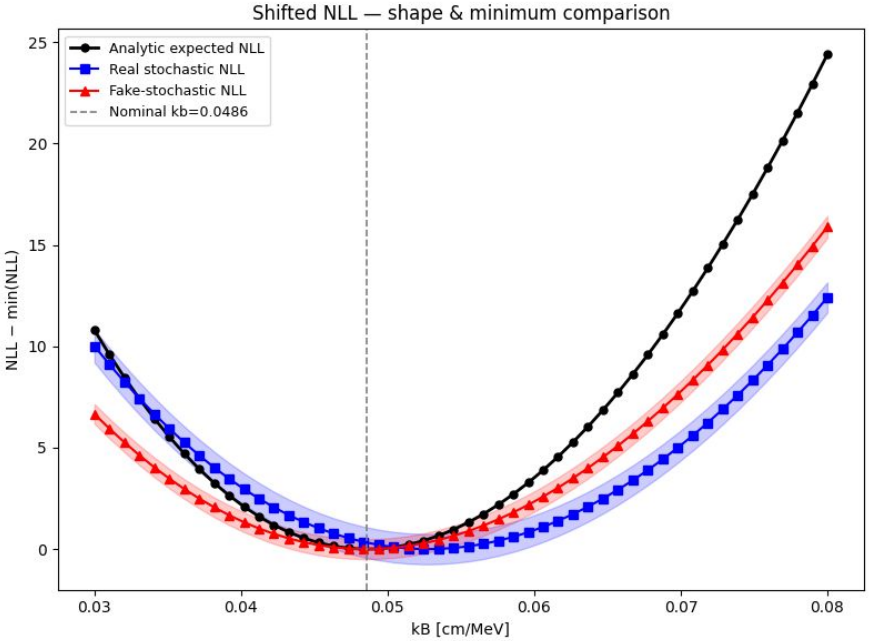
Back to basics

Studying whether the loss bias comes from:

- improper loss formulation
- lack of accuracy of the probabilistic estimate

Using a “Fake-stochastic” sample:

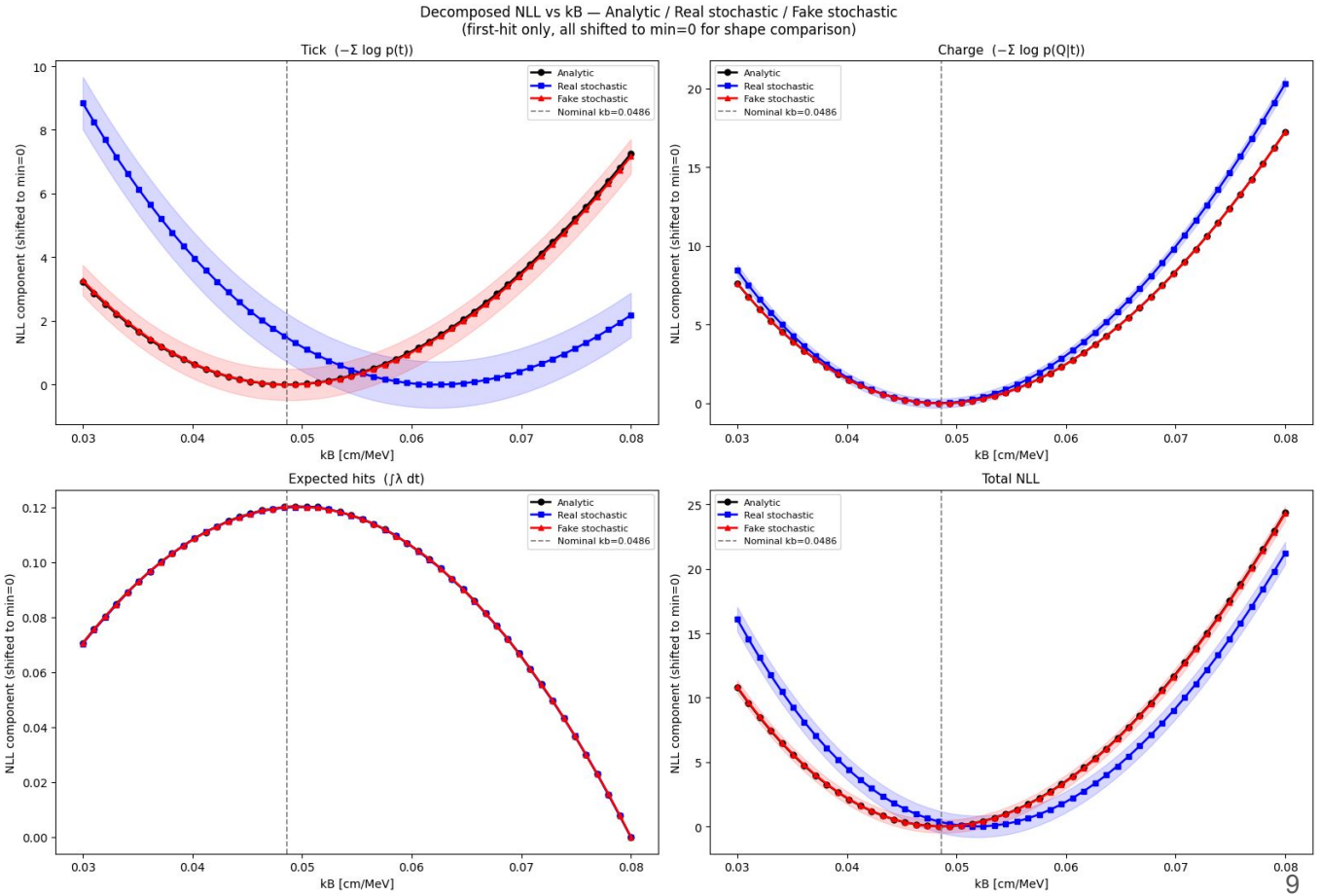
- Stochastic draws from the probabilistic estimate



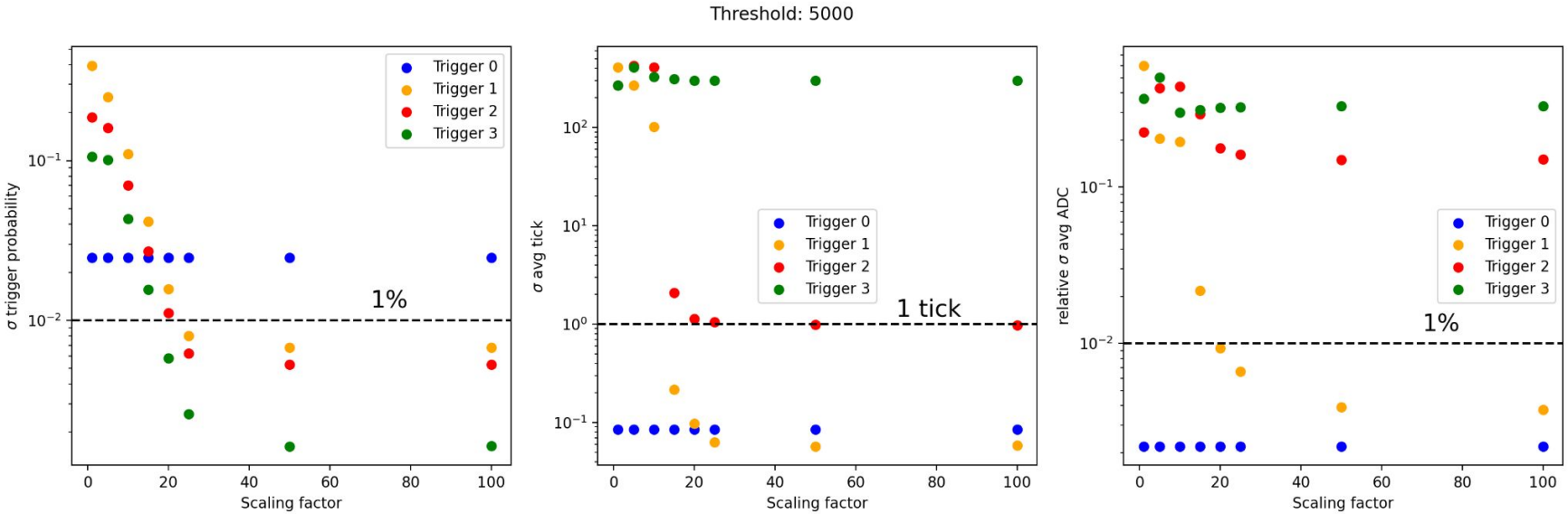
By drawing directly from the guessed distribution, we fix the bias

Back to basics

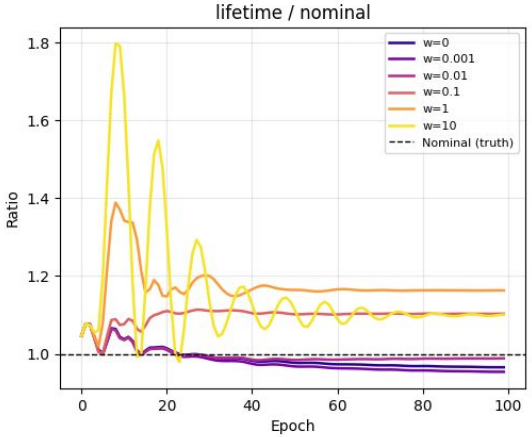
The tick probability estimate seems to be the issue in the loss → went back to reintroduce a fix in the probabilistic estimate that got lost on some function refactor…

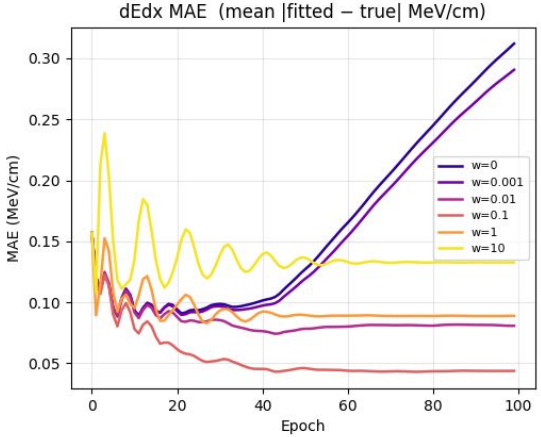


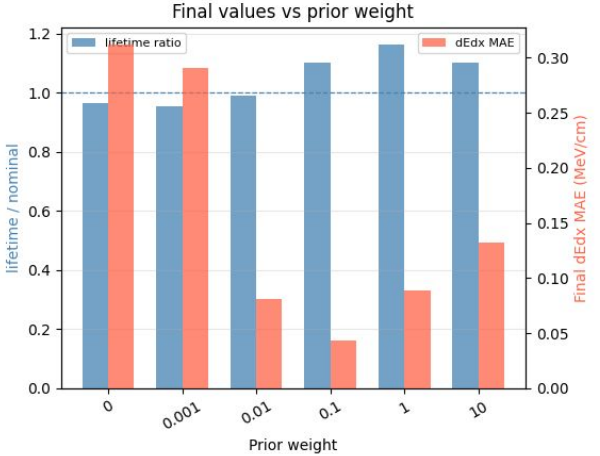
Backup


Threshold: 5000
σ trigger probability
Scaling factor
1%
Trigger 0
Trigger 1
Trigger 2
Trigger 3
σ avg tick
1 tick
relative σ avg ADC

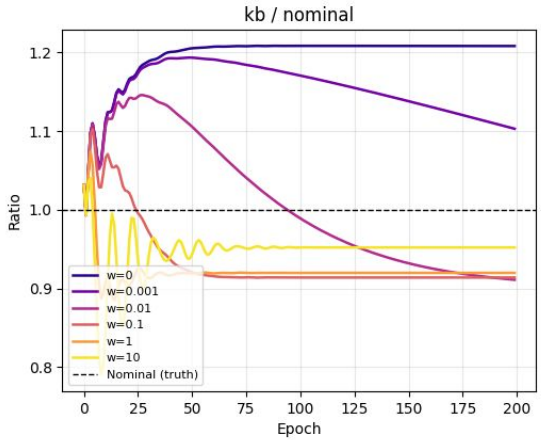
Prior-weight sweep — joint fit: lifetime + per-segment dEdx
(5 batches, 100 epochs, lr=0.1, init=2420 nom=2200)

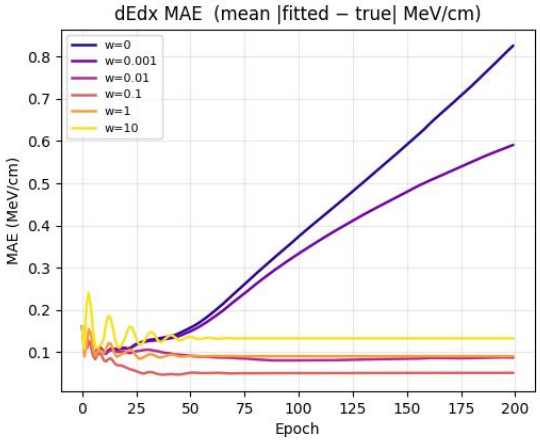


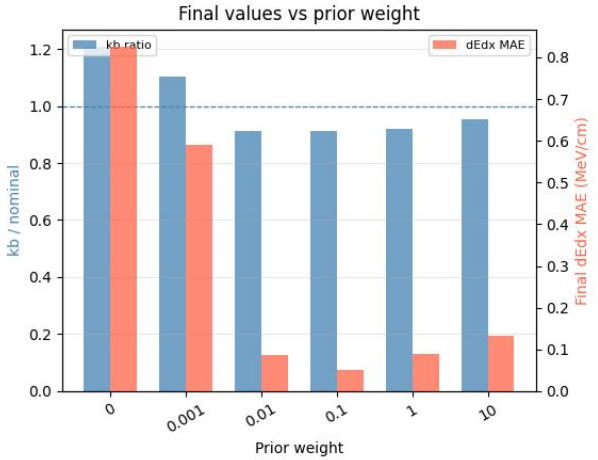




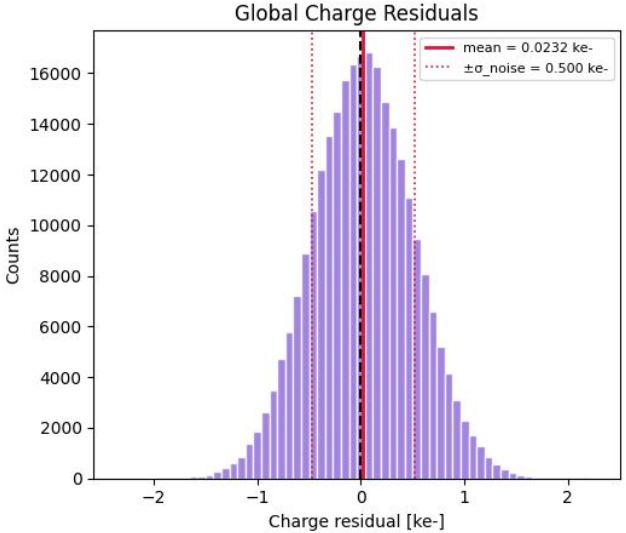
Prior-weight sweep — joint fit: kb + per-segment dEdx
(5 batches, 200 epochs, lr=0.1, init=0.05346 nom=0.0486)

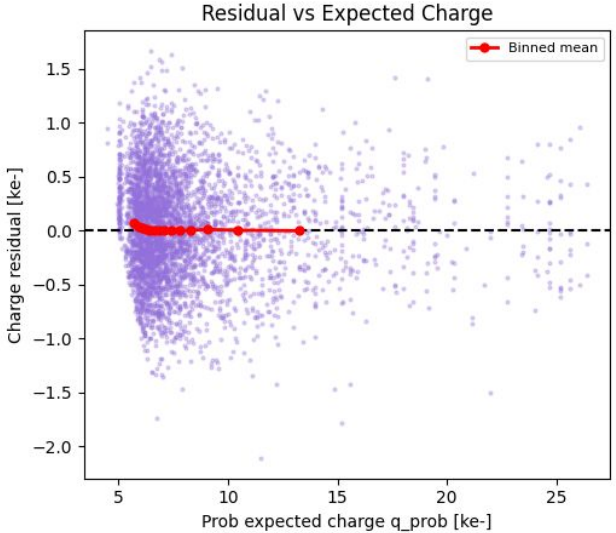


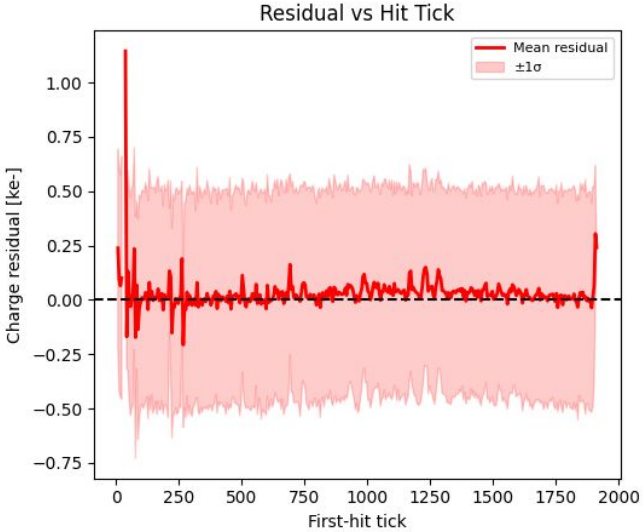


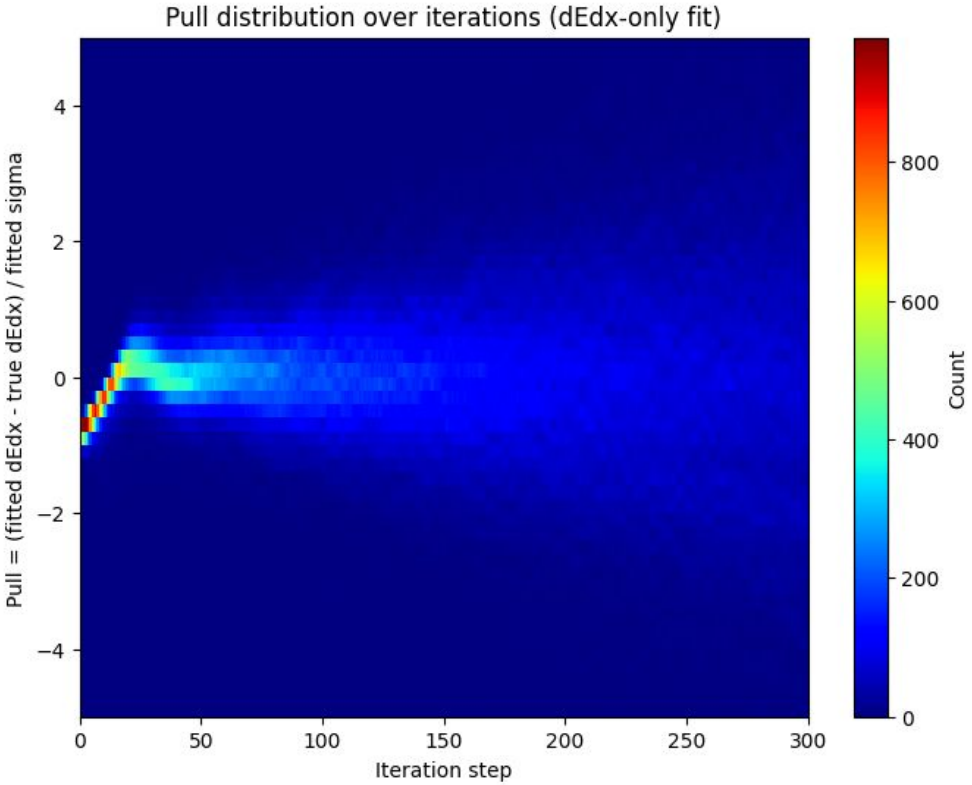


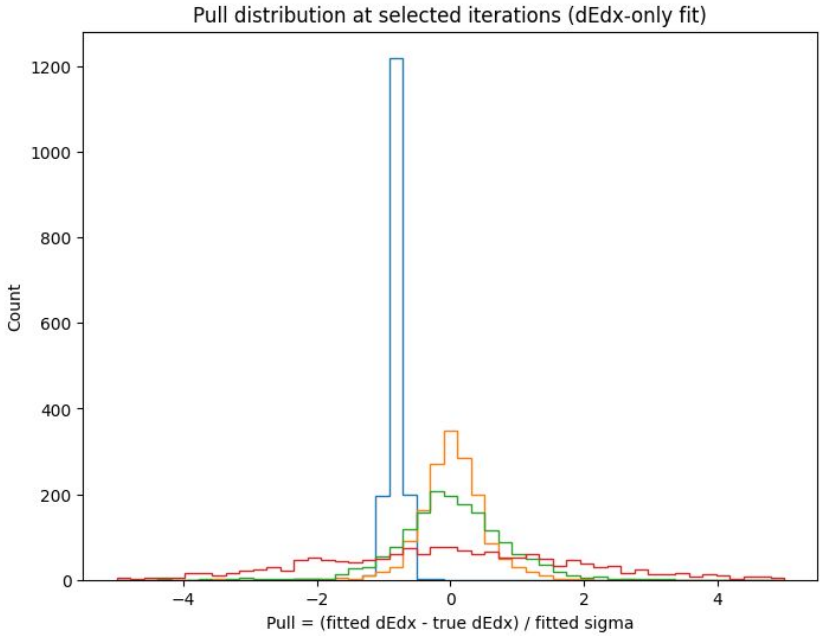
Charge Residuals: q_stoch − q_prob(t_drawn)



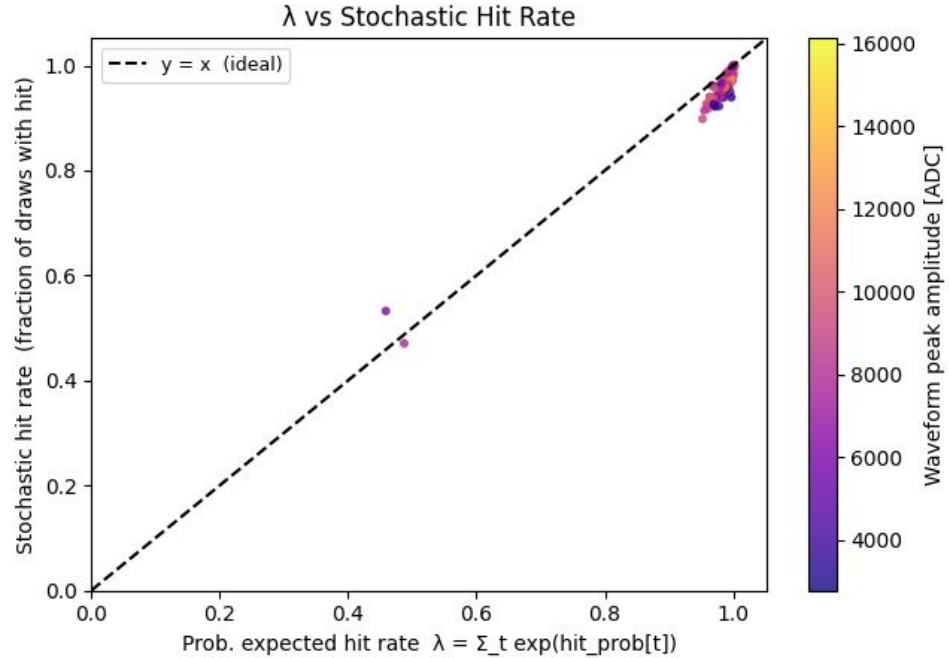


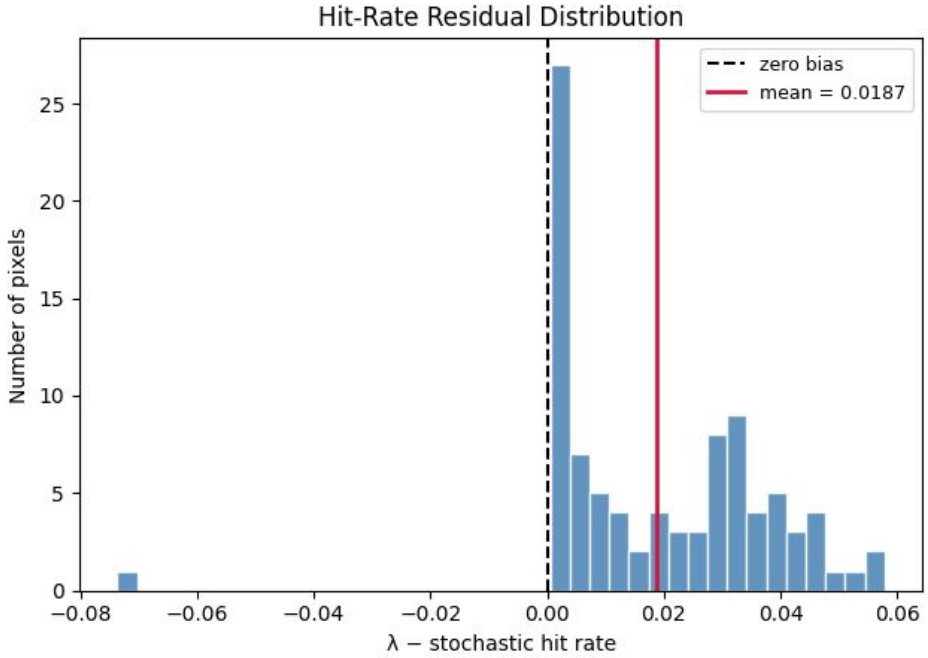


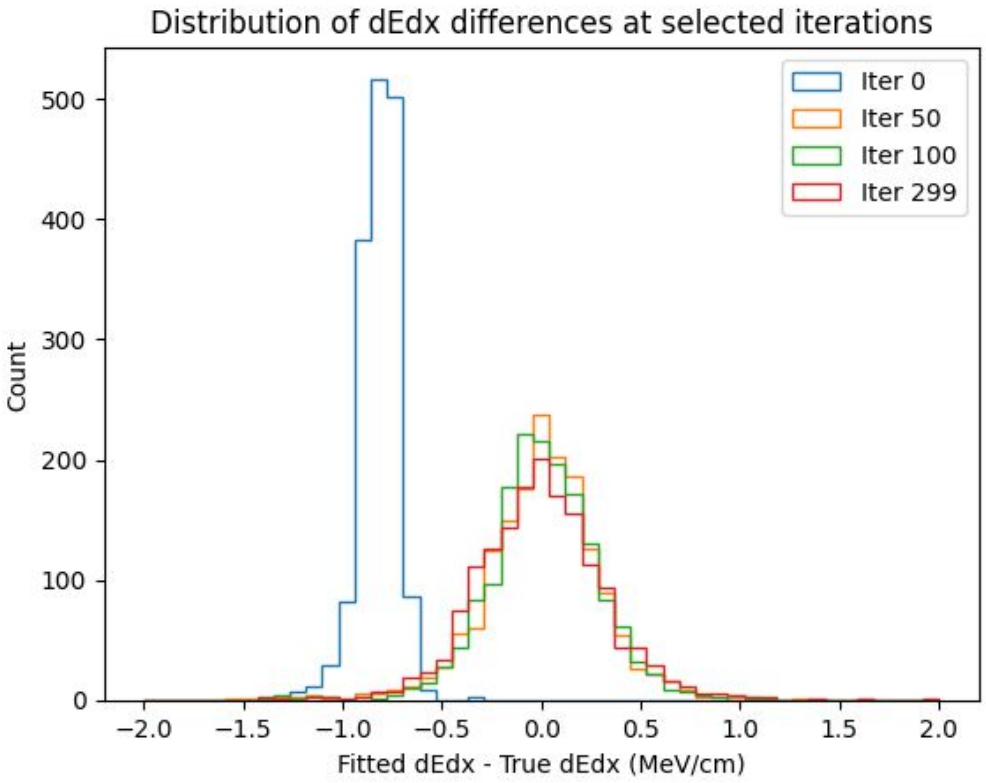

Pull distribution over iterations (dEdx-only fit)
Pull = (fitted dEdx - true dEdx) / fitted sigma
Iteration step
Count
0
50
100
150
200
250
300
4
2
0
−2
−4
800
600
400
200
0

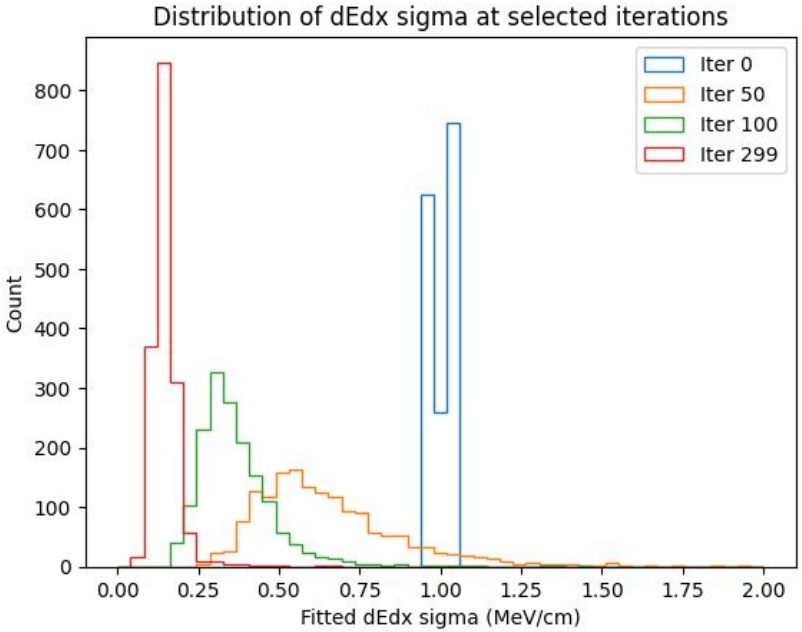

Pull distribution at selected iterations (dEdx-only fit)
Count
Pull = (fitted dEdx - true dEdx) / fitted sigma
1200
1000
800
600
400
200
0
−4
−2
0
2
4

Probabilistic λ Calibration (first-hit slot)






Distribution of dEdx differences at selected iterations
Iter 0
Iter 50
Iter 100
Iter 299
Count
Fitted dEdx - True dEdx (MeV/cm)


Distribution of dEdx sigma at selected iterations
Iter 0
Iter 50
Iter 100
Iter 299
Count
Fitted dEdx sigma (MeV/cm)