Machine learning with FAST

Evaluating the performance of a FAST only reconstruction

Fraser Bradfield
Osaka Metropolitan University, Graduate School of Science

2025/10/15

Contents

① The Fluorescence detector Array of Single-pixel Telescopes (FAST)

② Training a ML model to predict shower parameters with FAST
- Model training & performance in simulations

③ Applying the trained model to real events detected by FAST
- Issues/challenges
- Degeneracy in reconstructed solutions

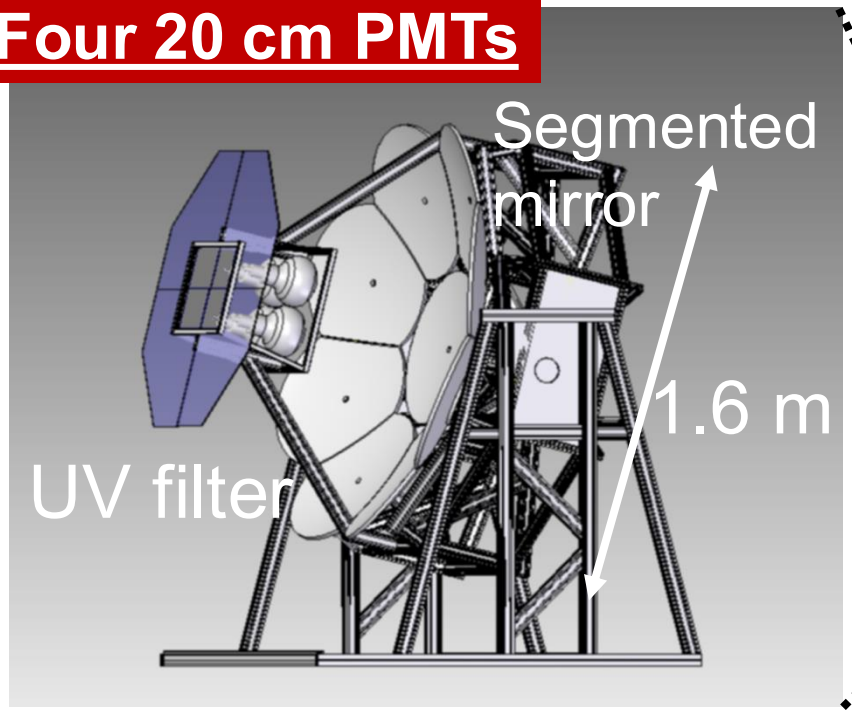
Contents

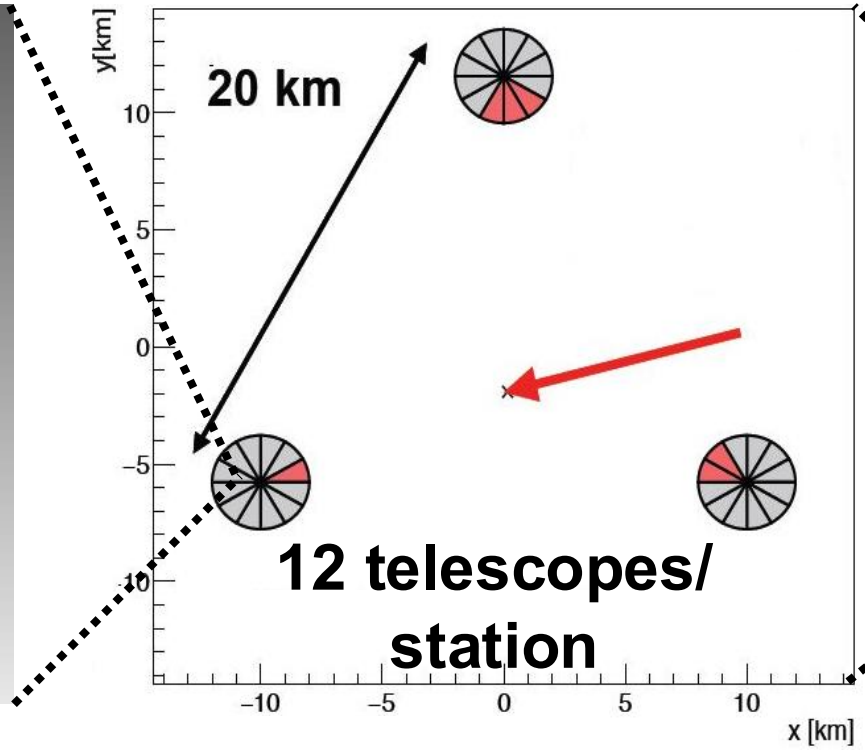
The Fluorescence detector Array of Single-pixel Telescopes (FAST)

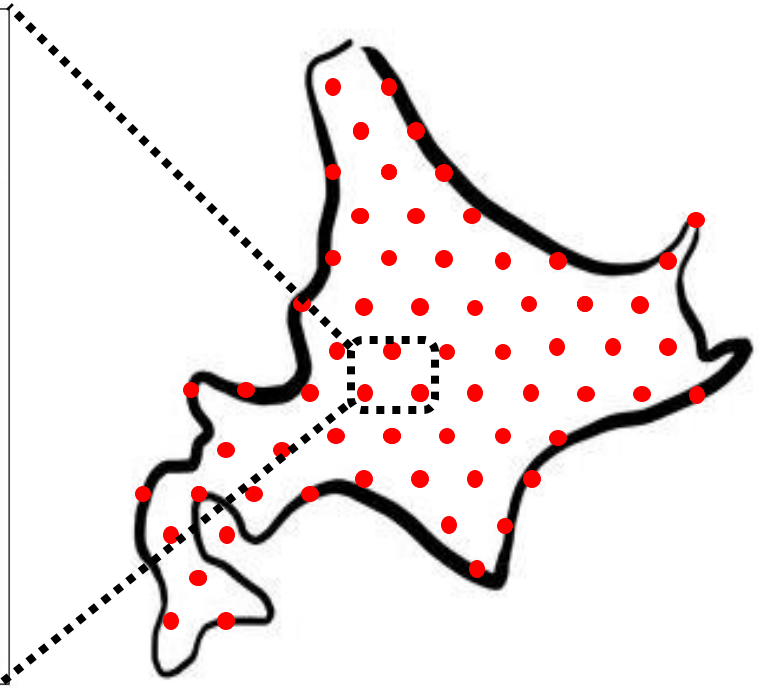
- Next generation cosmic ray experiment
- Sites in both hemispheres
- Array of fluorescence telescopes ~ 60,000 km^2

- **Just 4 pixels!**
- **Ideally want same resolutions as Auger/TA**

Four 20 cm PMTs



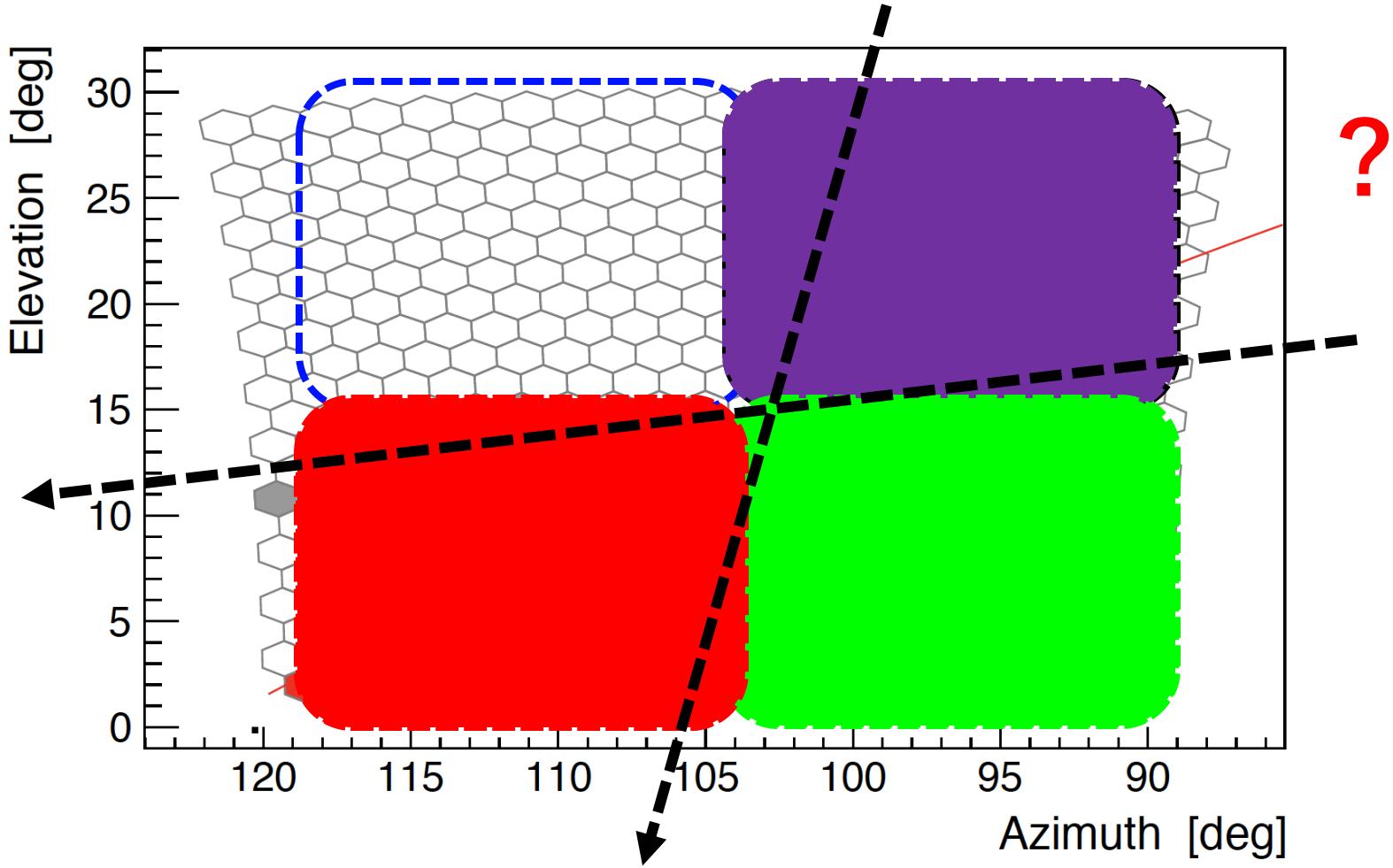




FAST aims to measure an unprecedented number of UHECRs above 10^{20} eV

FAST: Top-Down Reconstruction

With just 4 pixels – can't utilize same geometrical reconstruction procedures as Auger/TA!



FAST: Top-Down Reconstruction

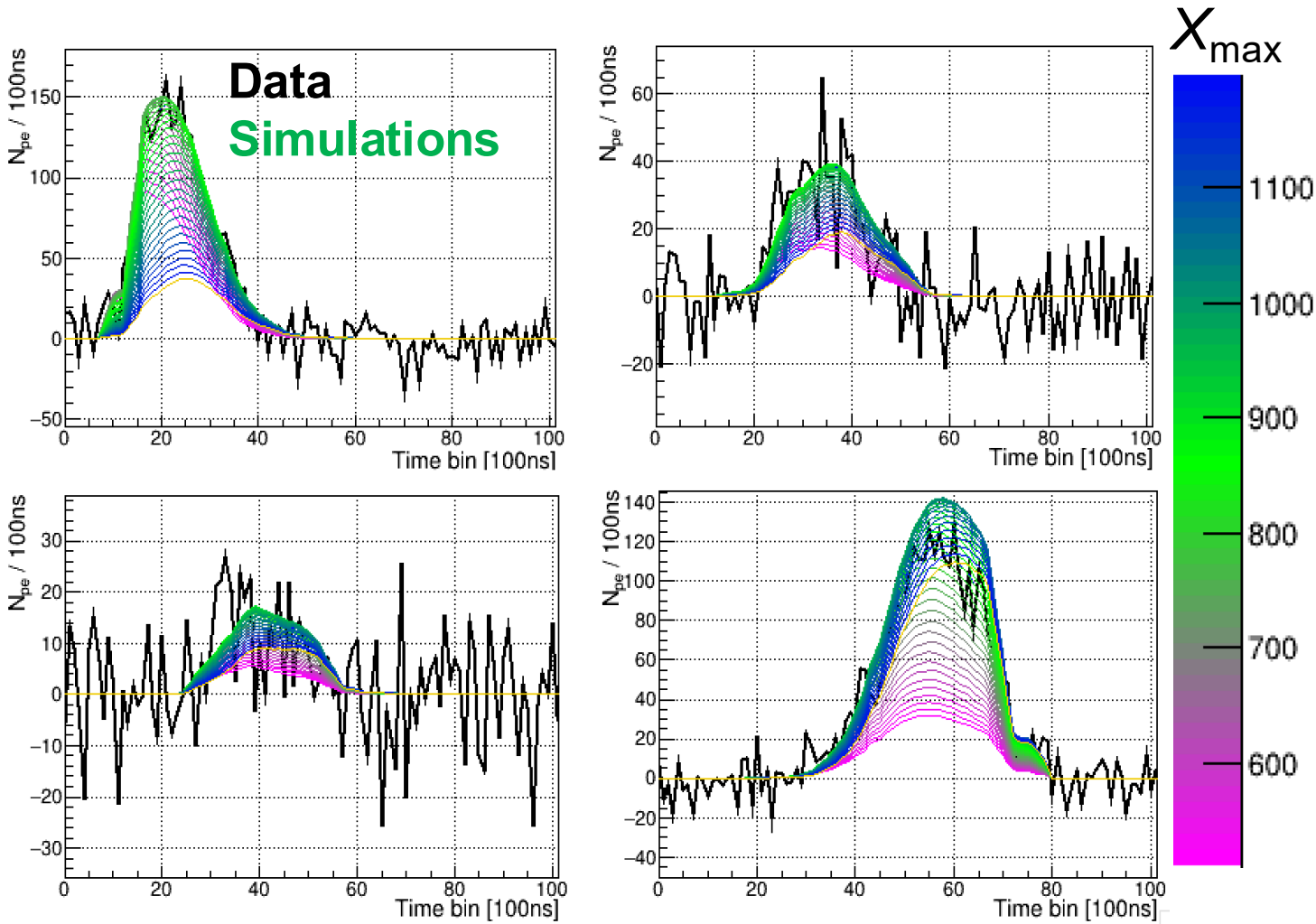
Waveform matching:

- Compare data directly to simulations
- Use minimizer to select the simulated parameters to test

Maximize log-likelihood:

$$\ln \mathcal{L}(\vec{x}|\vec{a}) = \sum_{k}^{N_{\text{pix}}} \sum_{i}^{N_{\text{bins}}} \ln\left(P_k(x_i|\vec{a})\right)$$

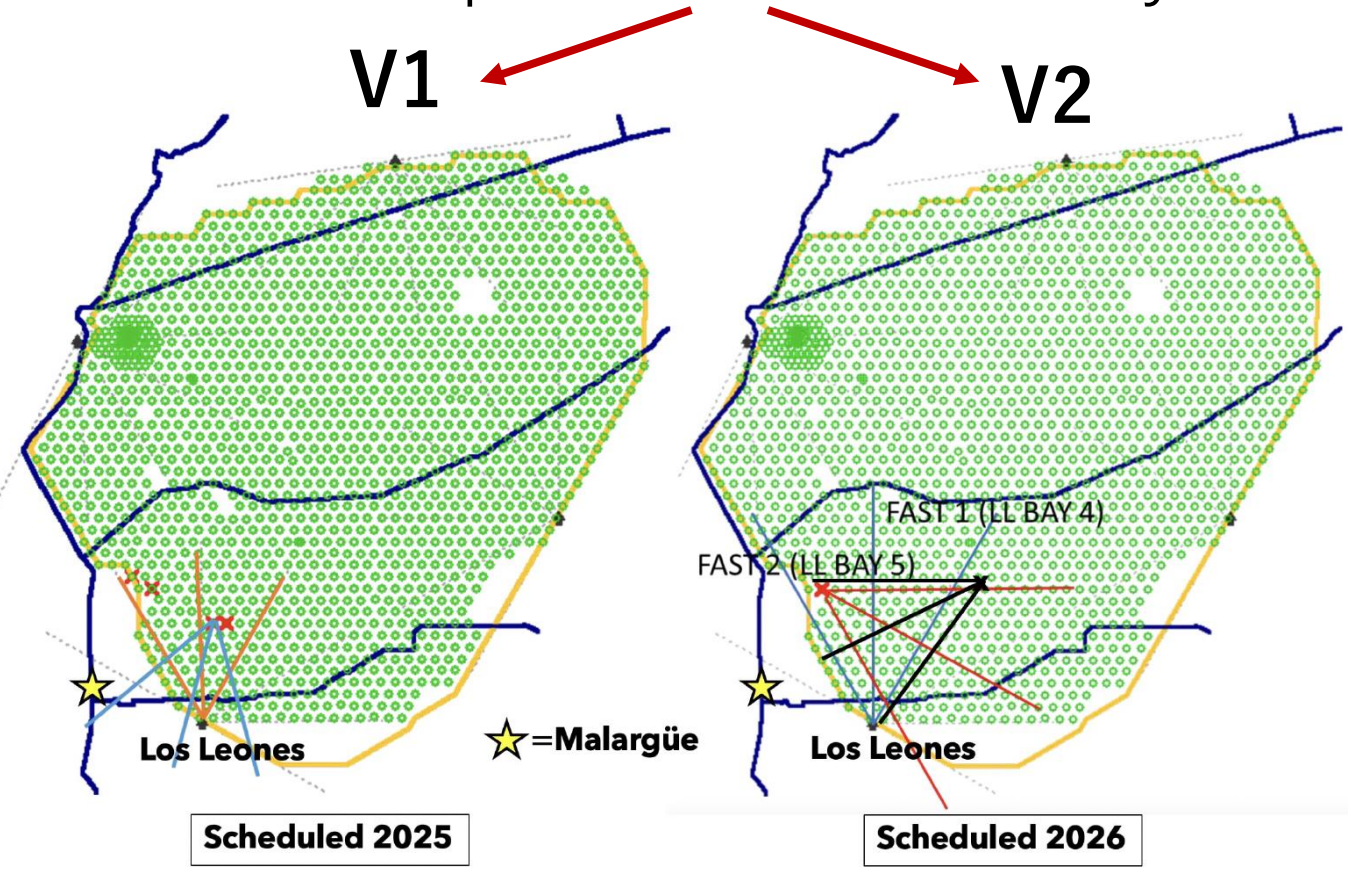
Probability of observing signal x_i in bin i of PMT k given shower parameters $\vec{a} = (E, X_{\text{max}}, \theta, \phi, x, y)$

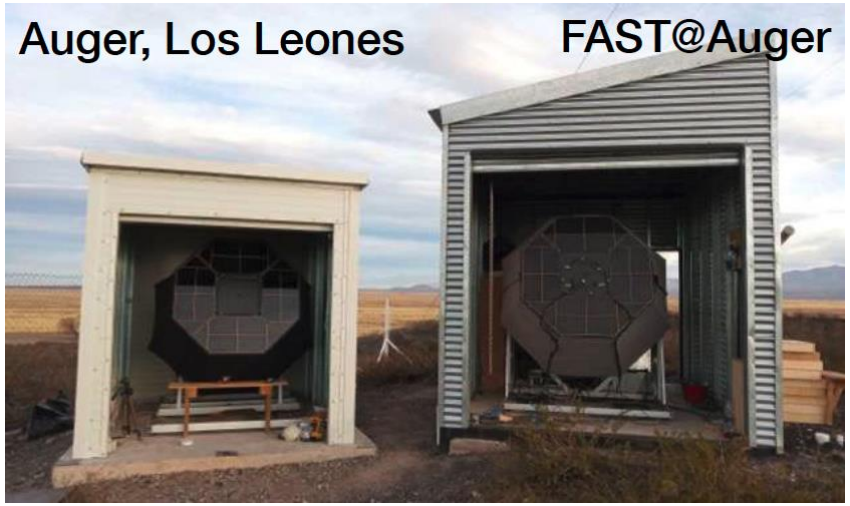


The result of the TDR depends heavily on the "first guess"

FAST: Current / future prototypes

- Two sets of prototypes currently in operation
- Next step – “FAST mini-array”!



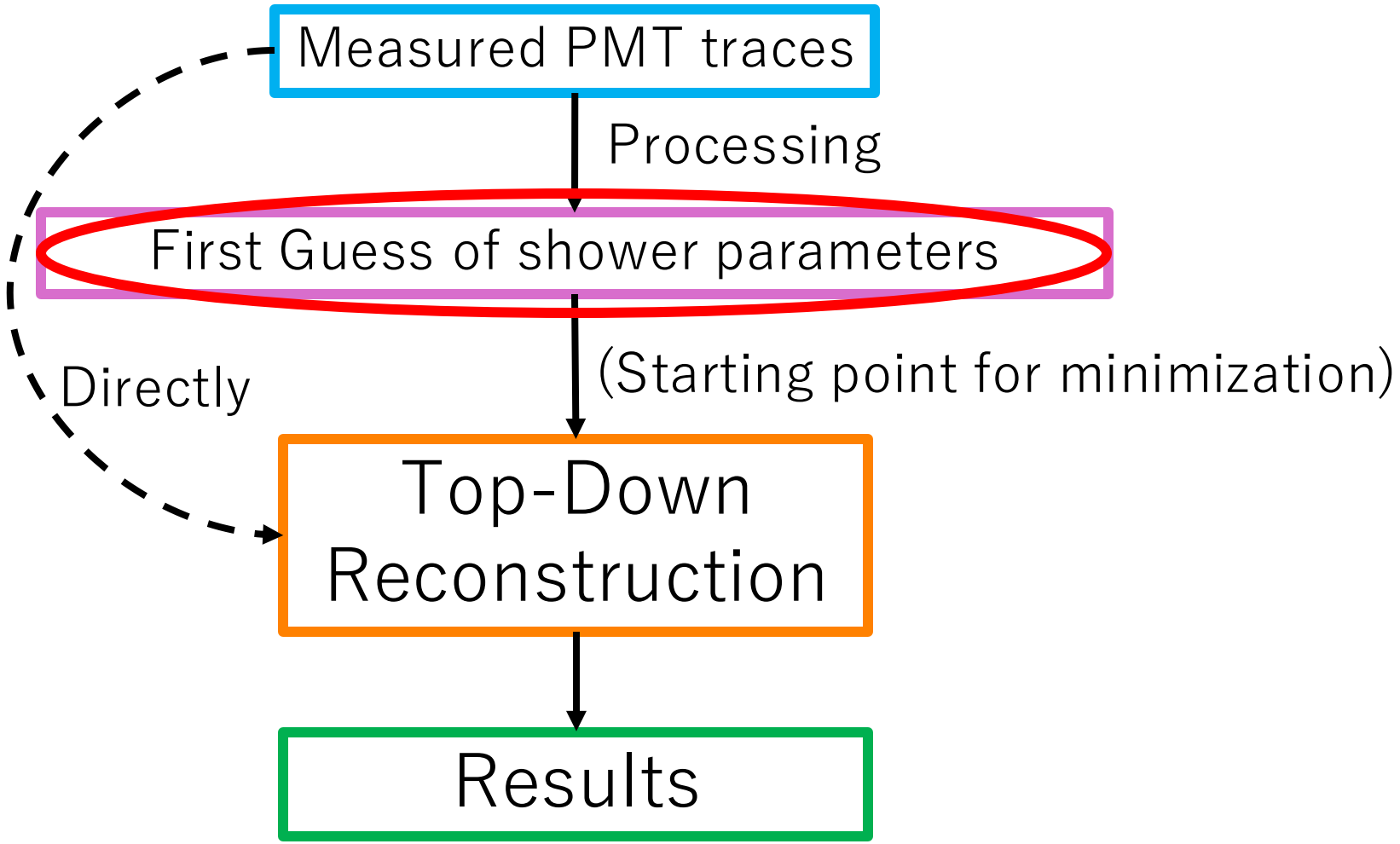


Fujii, 2023

Contents

FAST Reconstruction Overview

- Flow of the FAST reconstruction



First Guess Estimation: Why machine learning?

- Can't apply standard fitting procedures to determine geometry due to low resolution camera (2 x 2)
  - How about trying to parameterize the relationship between pixel timing/s, relative signal size/s and geometry?

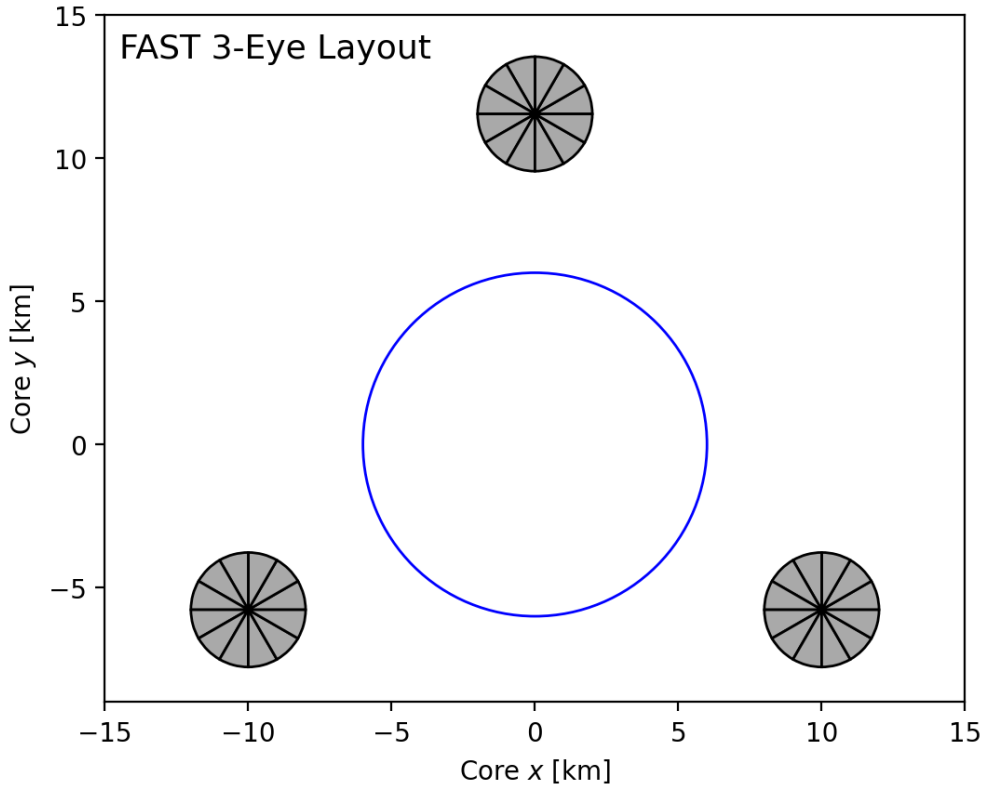
- Would have to consider different time/signal orderings, how to combine info from separated telescopes, and fitting some likely non-trivial function
  - Difficult, time consuming, almost certainly degenerate for low number of triggered pixels, and may not gain much insight into relationship between shower parameters and timing/signal anyway

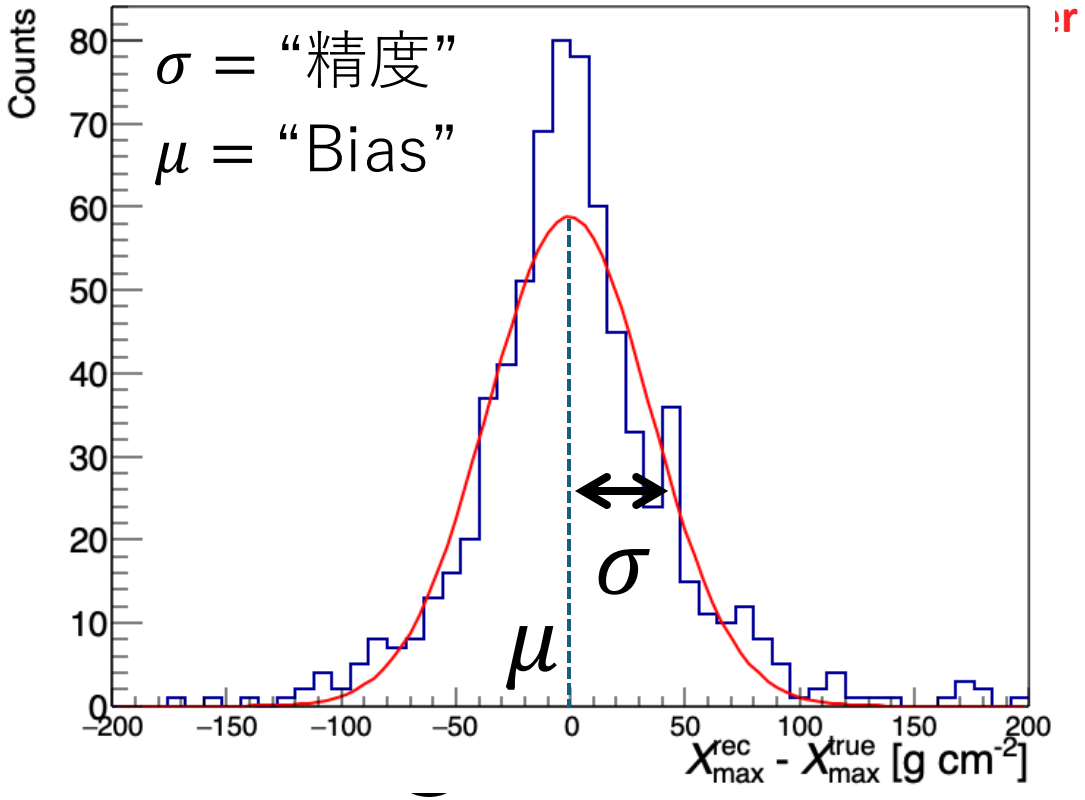
- Moreover, just want a first guess → will be optimized with top-down reconstruction anyway using time-dep. info

So, we try machine learning

Previous Machine Learning Studies with FAST

- Albury and later Fujii showed that a feed-forward, deep neural network can predict the shower parameters with a full-sized FAST array well
  - **Resolutions** - X_{max}: 30 g cm^{-2}, Energy: 8%, Arrival direction: 4.2°, Core: 460 m



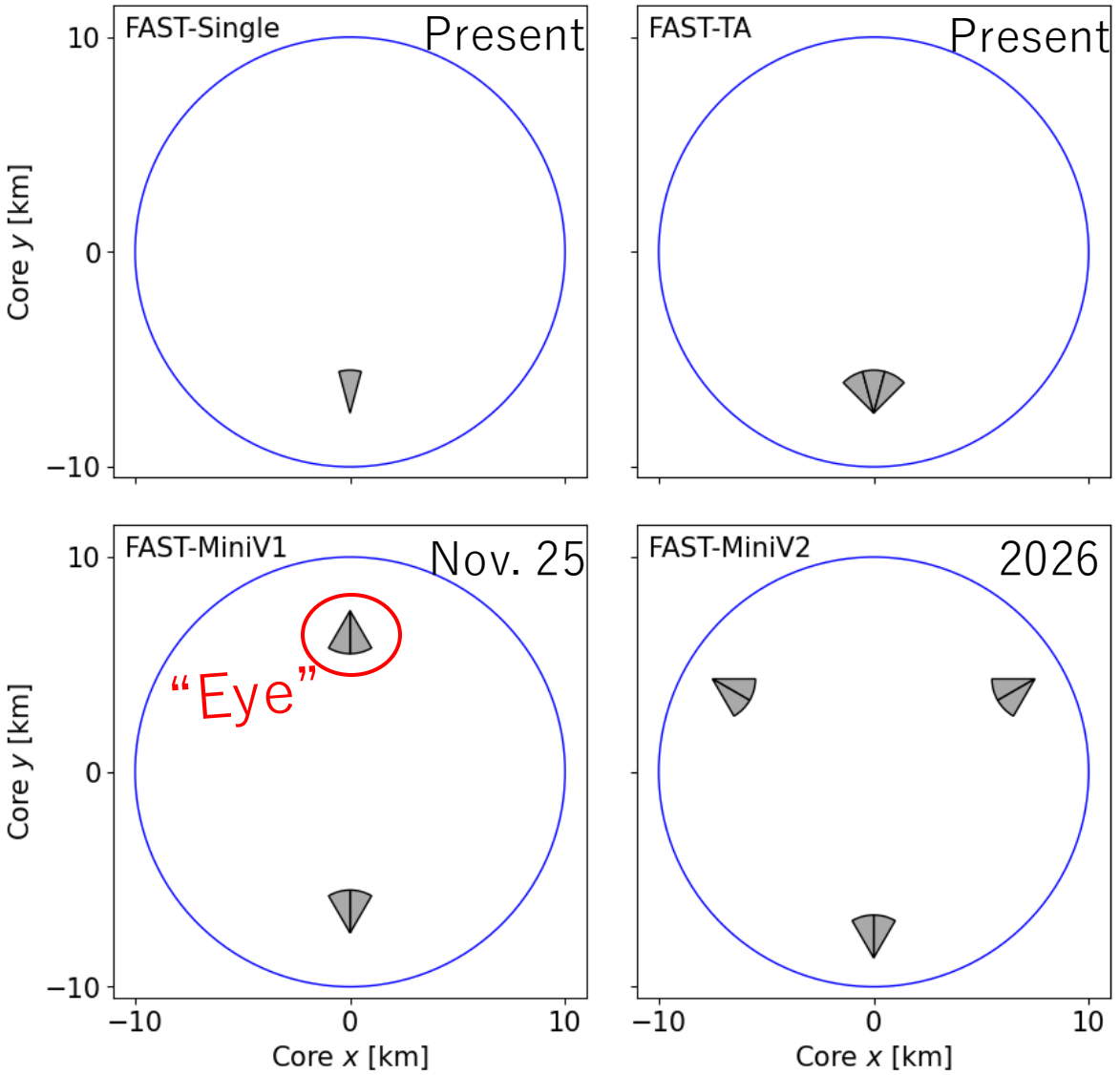


Can a similar model work for current layouts? Different architectures?

Dataset

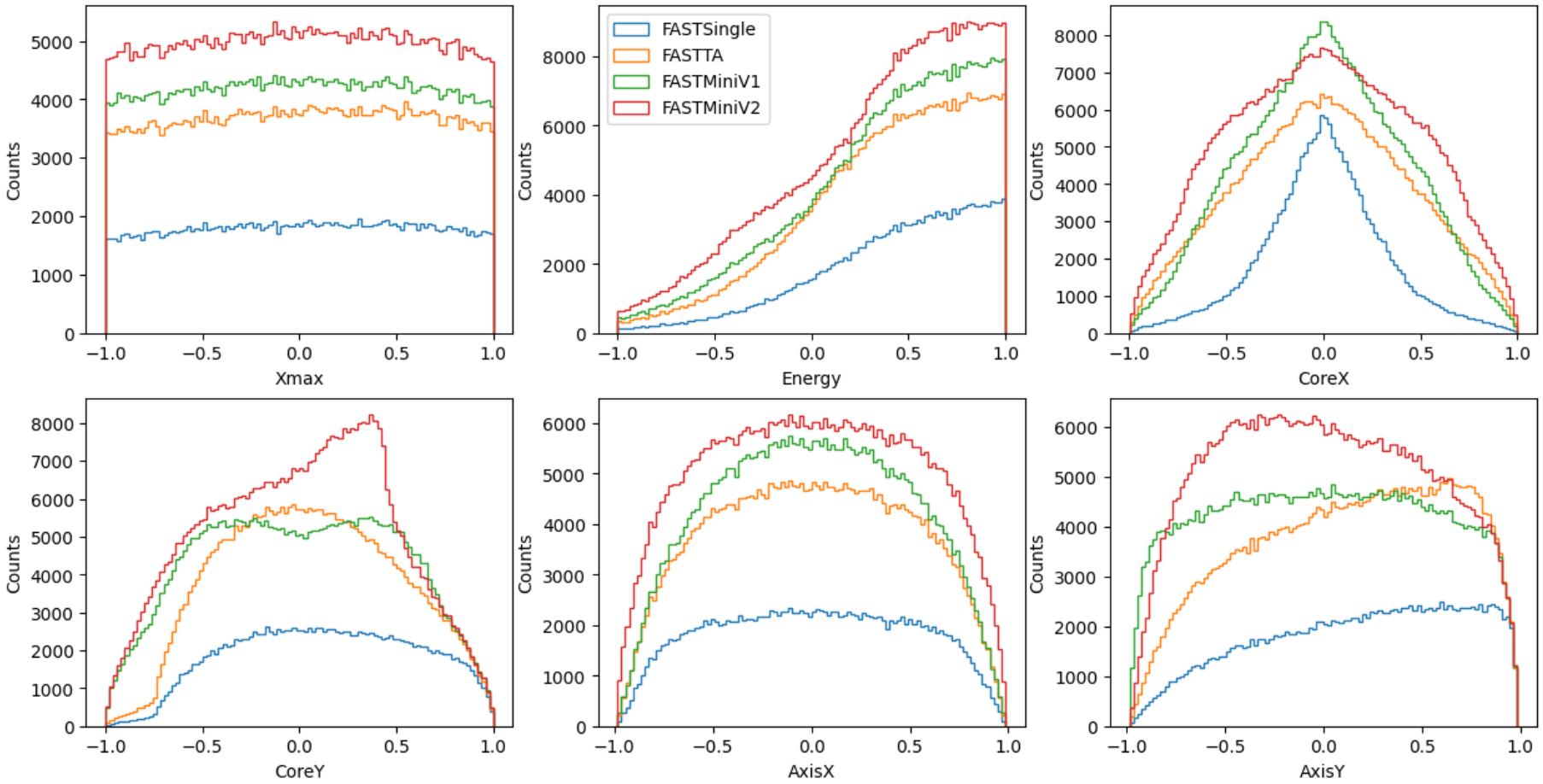
- 1,000,000 showers
- Four layouts
- Showers with significant signal in ≥ 1 PMT
  - FAST-Single ~ 2×10^5
  - FAST-TA ~ 4.1×10^5
  - FAST-MiniV1 ~ 4.7×10^5
  - FAST-MiniV2 ~ 5.5×10^5

$X_{\max}$	Uniform (500 - 1200 g cm^{-2})
Energy	Uniform in $\log(E/\mathrm{eV})$ (17.3 - 20)
θ	$\sin\theta\cos\theta$ (0 - 80°)
ϕ	Uniform (0 - 360°)
Core	Uniform (in circle centred at (0,0), $r = 10$ km)



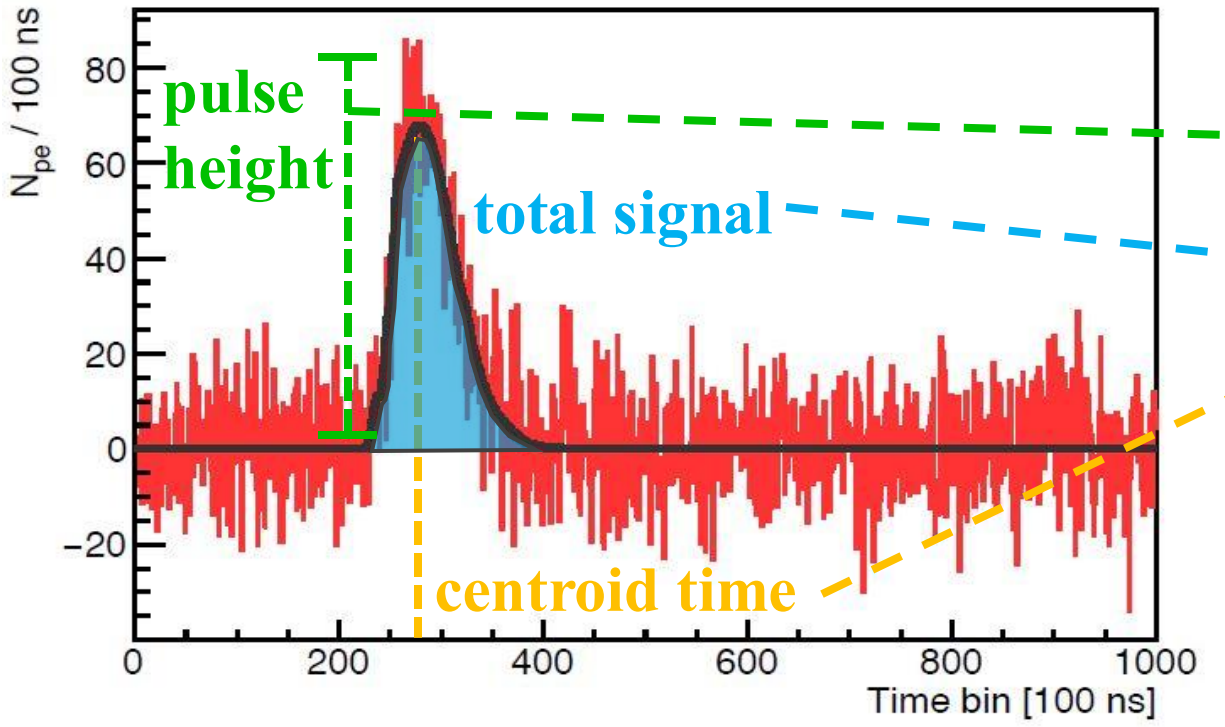
Output Parameters

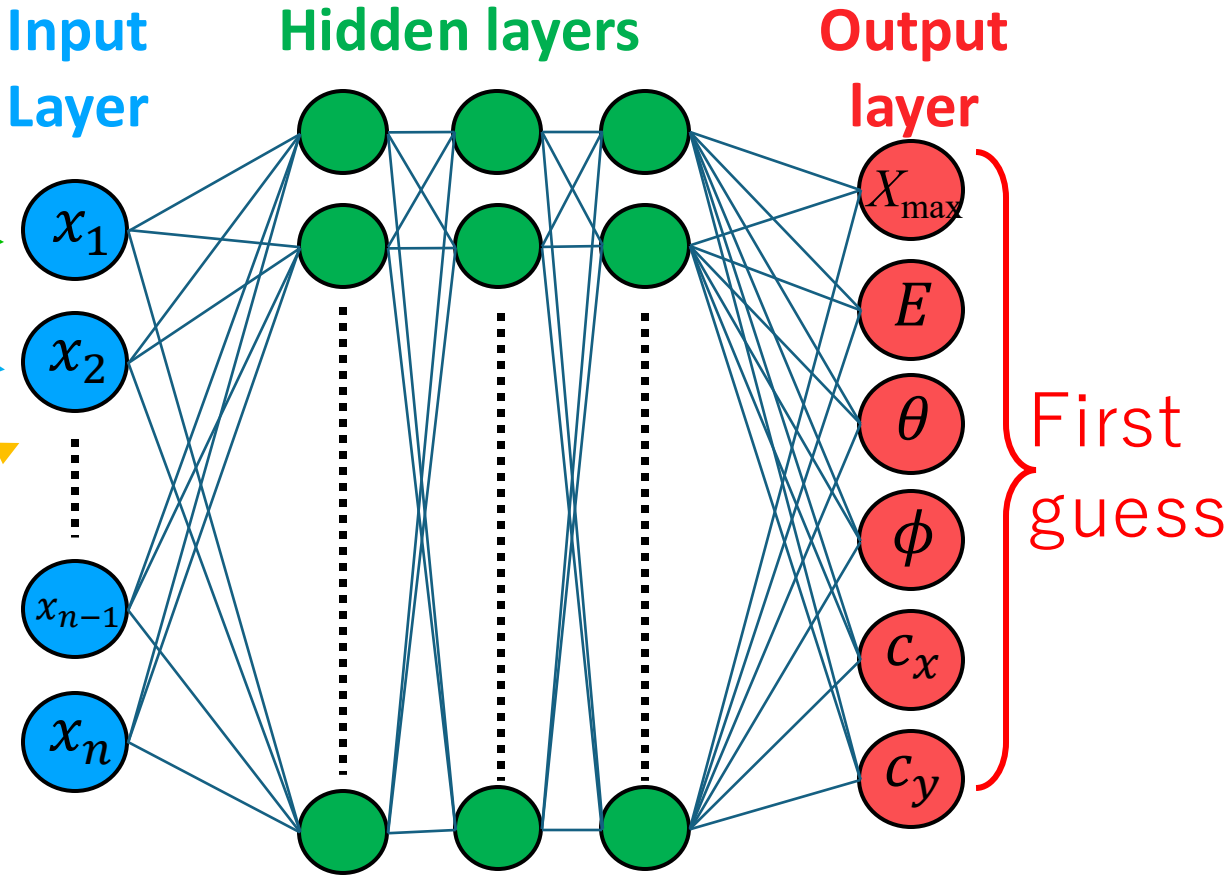
- Distributions of output (shower) parameters we want the models to learn – shapes of core/axis histograms due to layout geometry



Model 1: Basic DNN

- Identical architecture to previous studies
- Use height of PMT pulse, total signal and timing from each triggered PMT as inputs (use log of these values → more stable training)



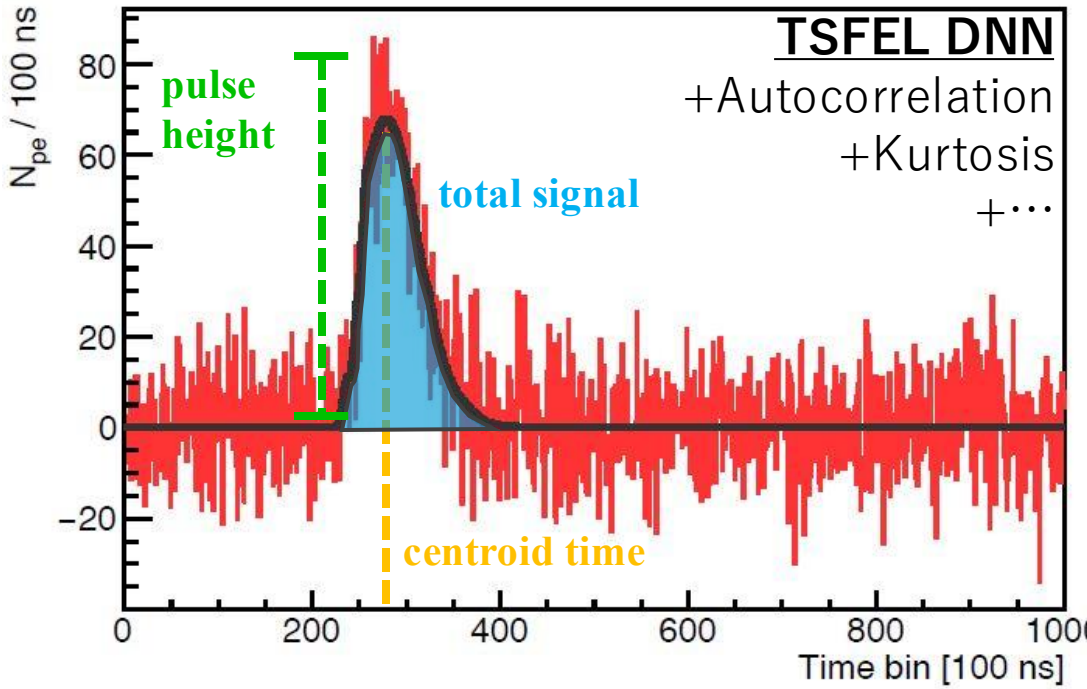


Model 2: TSFEL DNN

- Extension of the Basic DNN - same feedforward architecture

- Use the TSFEL python library to extract additional trace features
  - 45 statistical / temporal features per PMT
  - Remove features which don't vary between traces
  - Apply log transformation to remaining features
  - Add features from Basic DNN
  - Calculate correlation between features, remove one if correlation with another > 0.95

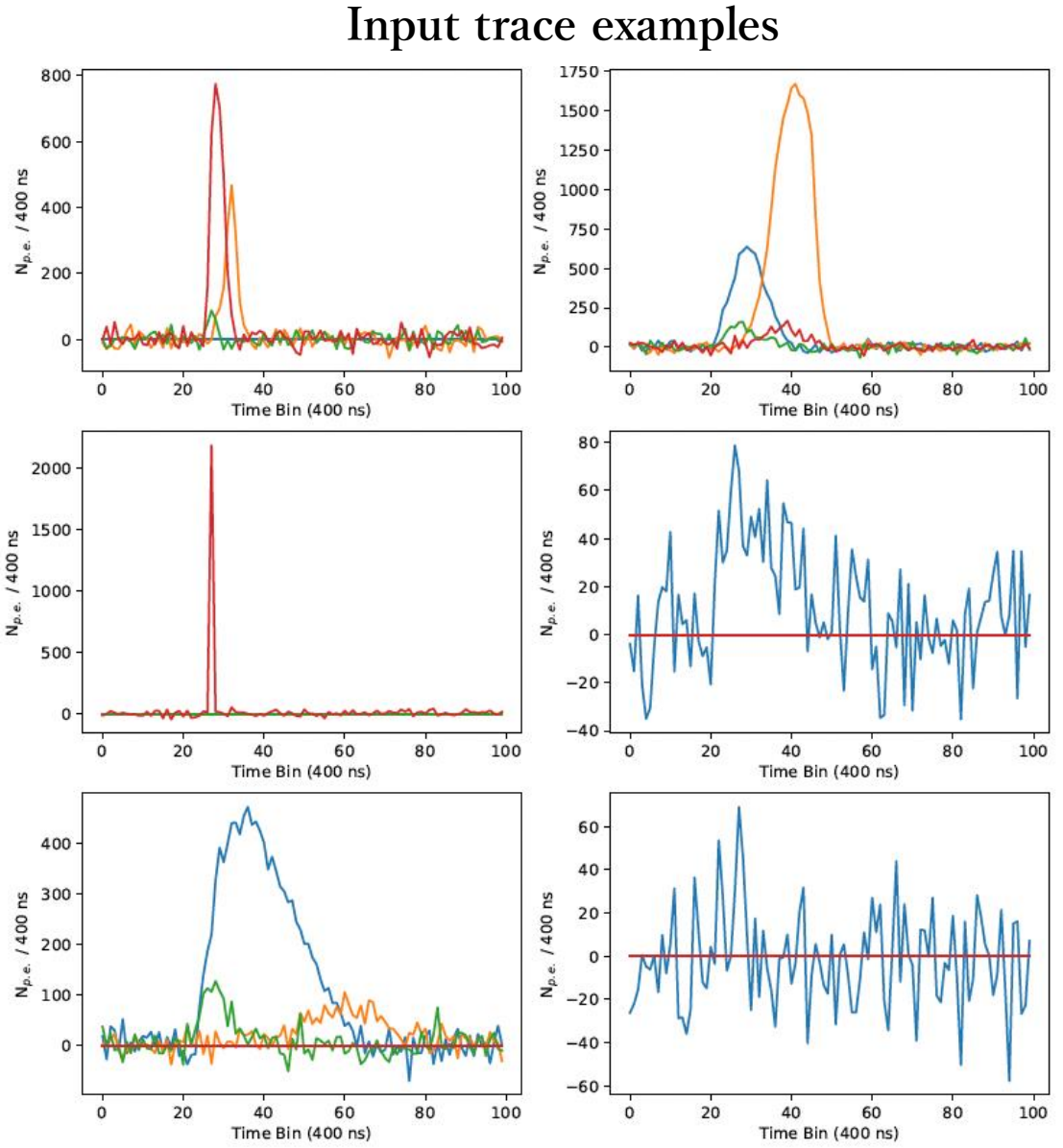
$$\hat{x} = \begin{cases} \log_{10}(|x|) & x \neq 0 \\ 0 & x = 0 \end{cases}$$



- Left with 11 features per PMT

Model 3: Long-short term memory (LSTM)

- Type of recurrent neural network used for analyzing time series data
  - Input traces directly to network
- Each PMT trace processed by same LSTM layer
  - Extract same features – 64/PMT
  - For PMTs with no signal, set the output of the LSTM layer for that PMT to all 0's
- Structure is
  - nPMTs trace inputs → LSTM layer → (64*nPMTs) nodes → 64 nodes → output layer



Model Comparison

- Use “hyper-parameter optimized” Basic DNN and TSFEL DNN
  - Both with hidden layer node structure 512/256/128/64/32
  - Learning rate 0.0003

- No hyper-parameter search for LSTM (time constraints)

- Train on full dataset and compare validation losses (MSE)

	FAST-Single	FAST-TA	FAST-MiniV1	FAST-MiniV2
Basic	0.1144	0.0827	0.0765	0.0685
TSFEL	0.1006	0.0689	0.0649	0.0570
LSTM	0.0906	0.0672	0.0632	0.0565

Better than Basic DNN

TSFEL DNN faster and more interpretability, so proceed with it

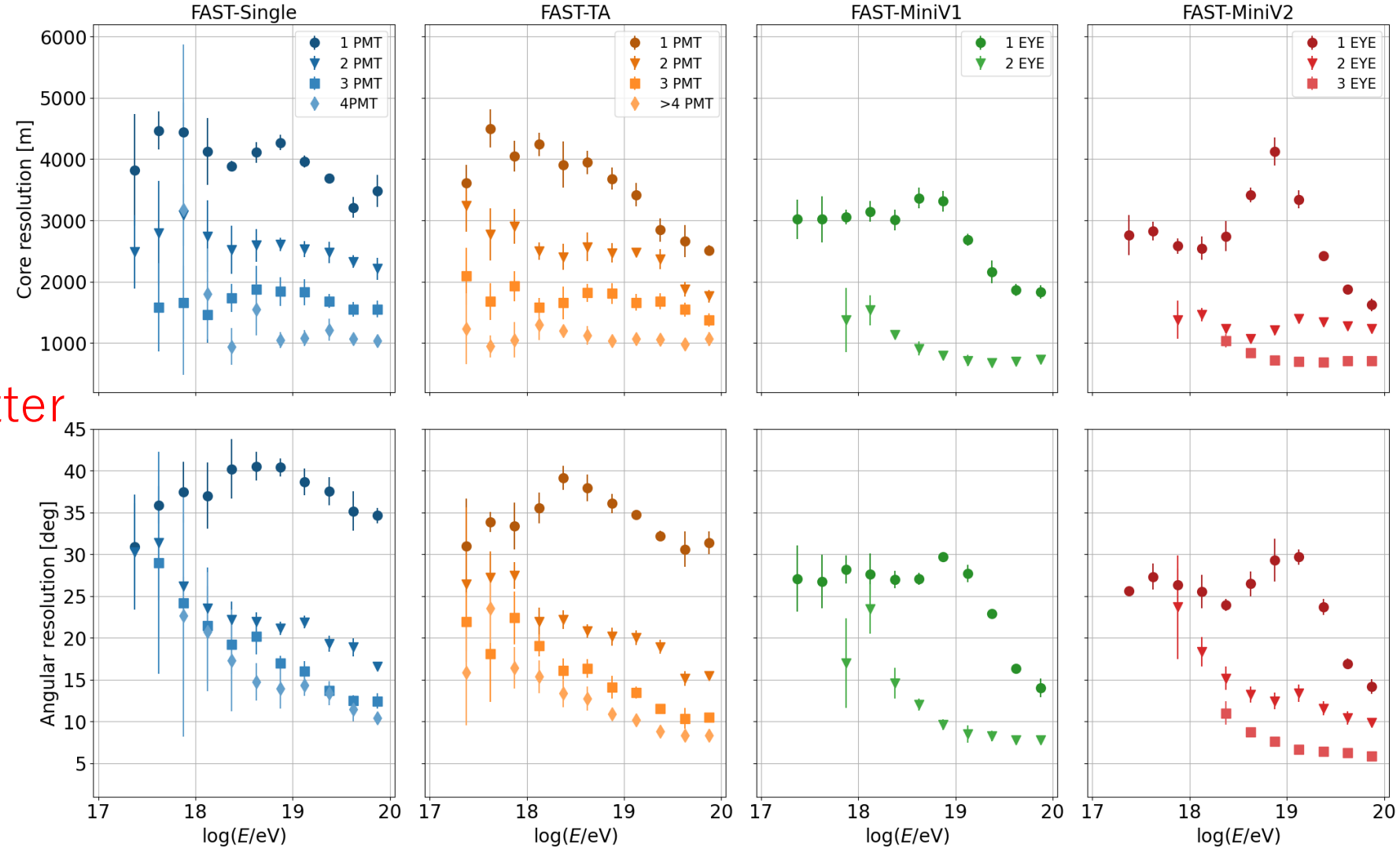
TSFEL DNN Performance

Core and angular resolutions*

Core

More PMTs & eyes is better

Angular



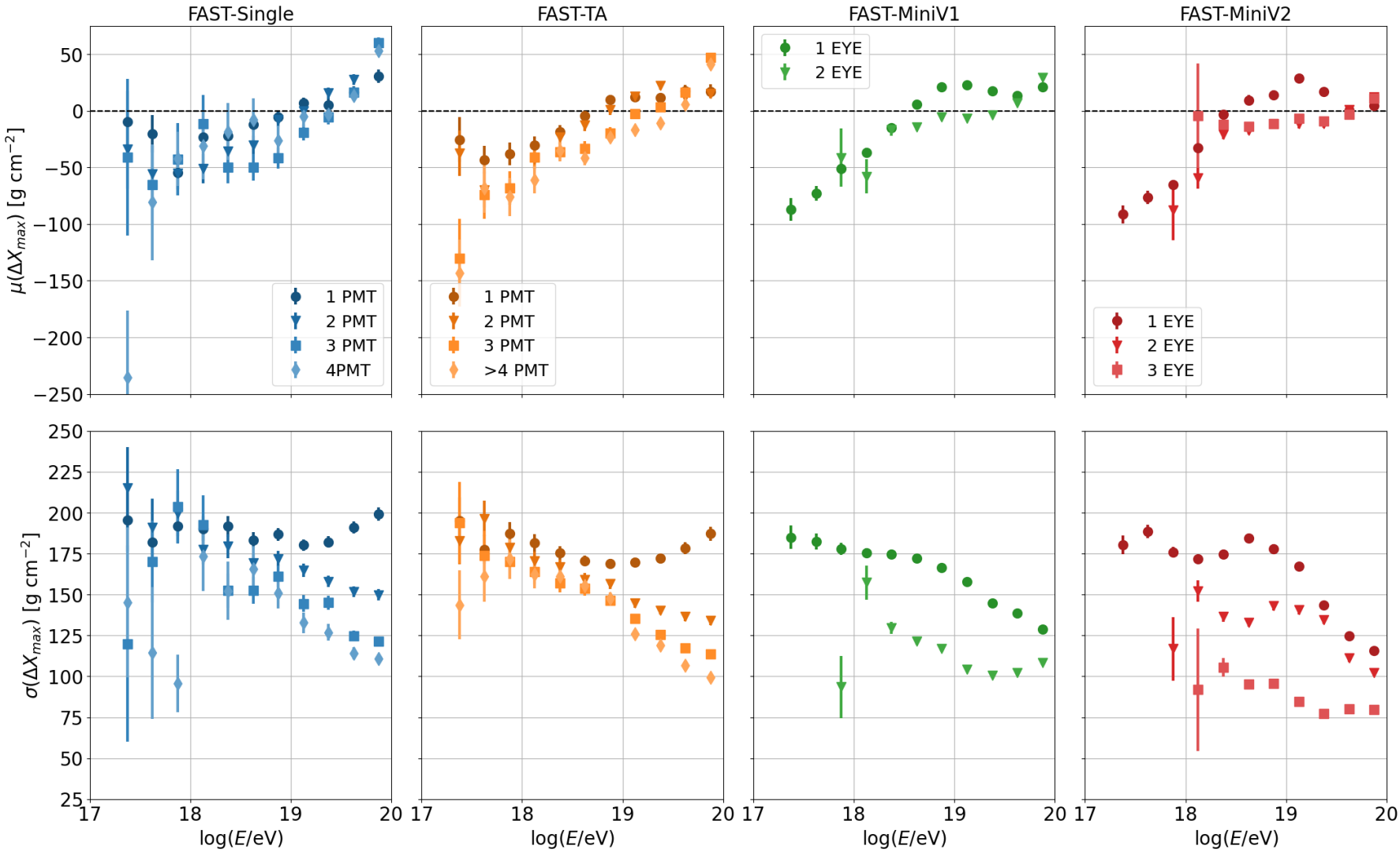
*Core distance / opening angle within which 68% of reconstructed events lie

TSFEL DNN Performance

Biases and resolutions: $X_{\max}$

Biases

Resolutions



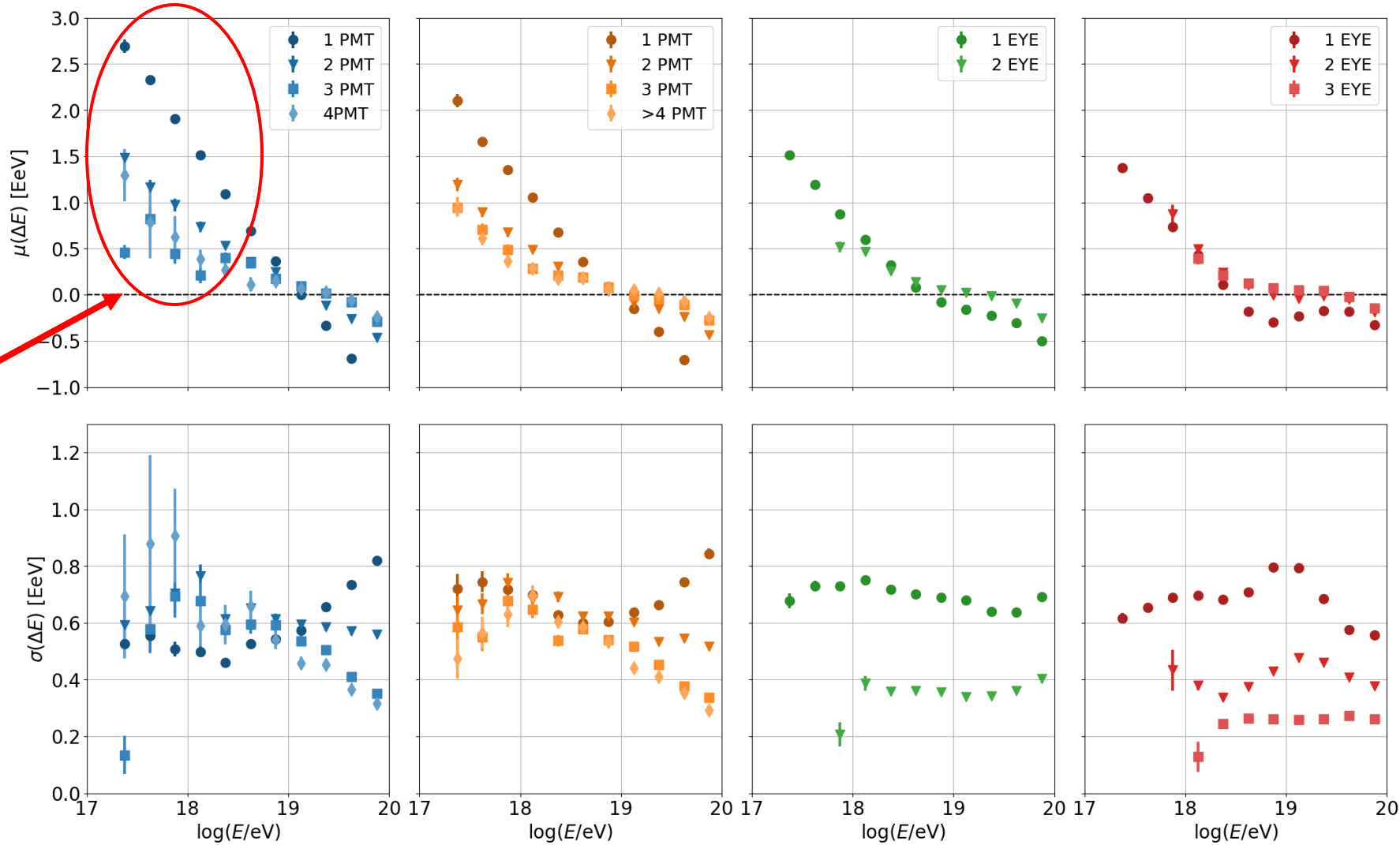
TSFEL DNN Performance

Biases and resolutions: **Energy**

Biases

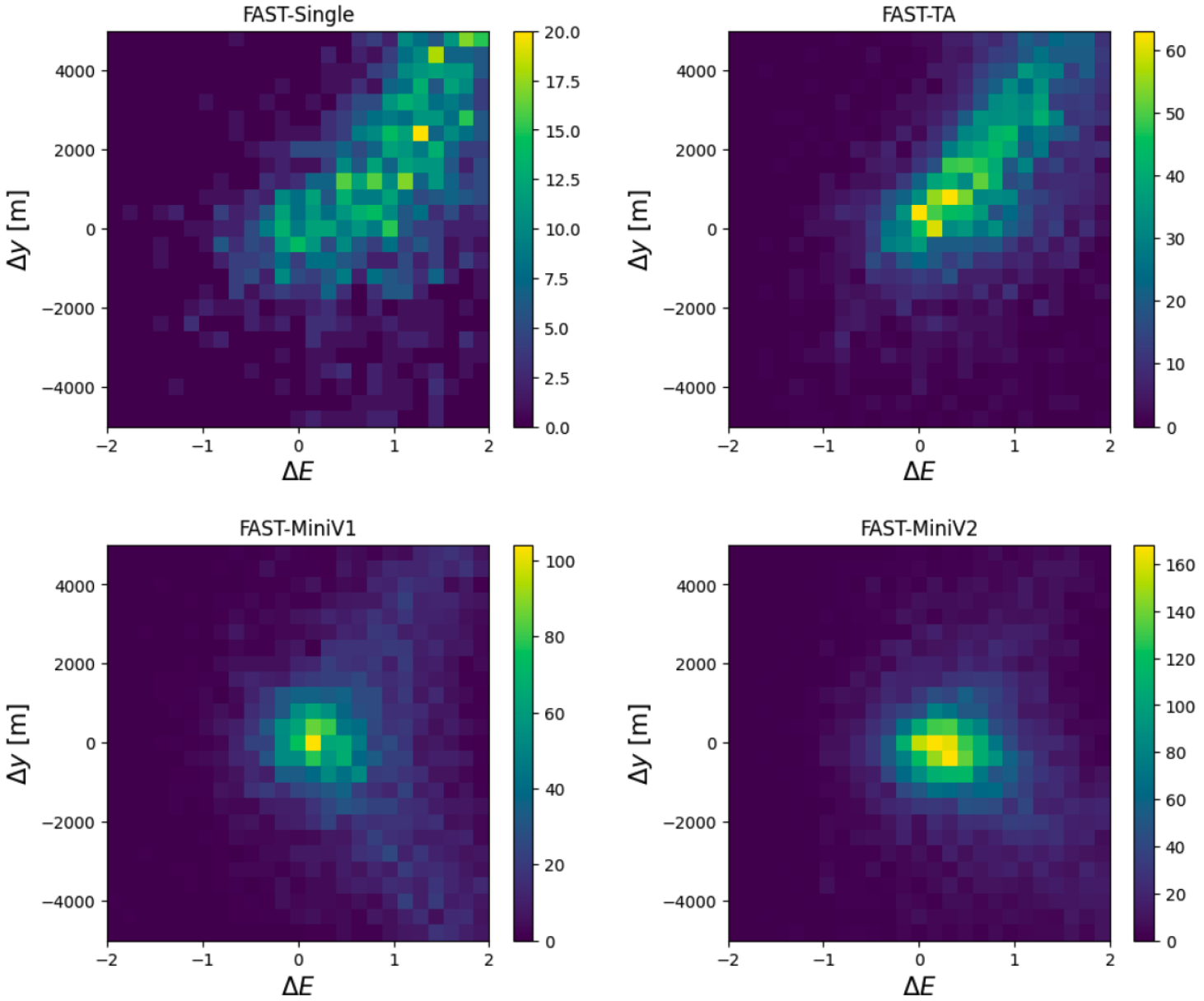
Degeneracy between energy and core position

Resolutions



Energy / Core position estimation

$17.5 < \log(E_{sim}/eV) < 18.5$



Full Reconstruction: Setup

TSFEL DNN First guess + TDR

- Use first guess from TSFEL DNN as input to TDR
  - Only use first guesses with values inside the range of the training data

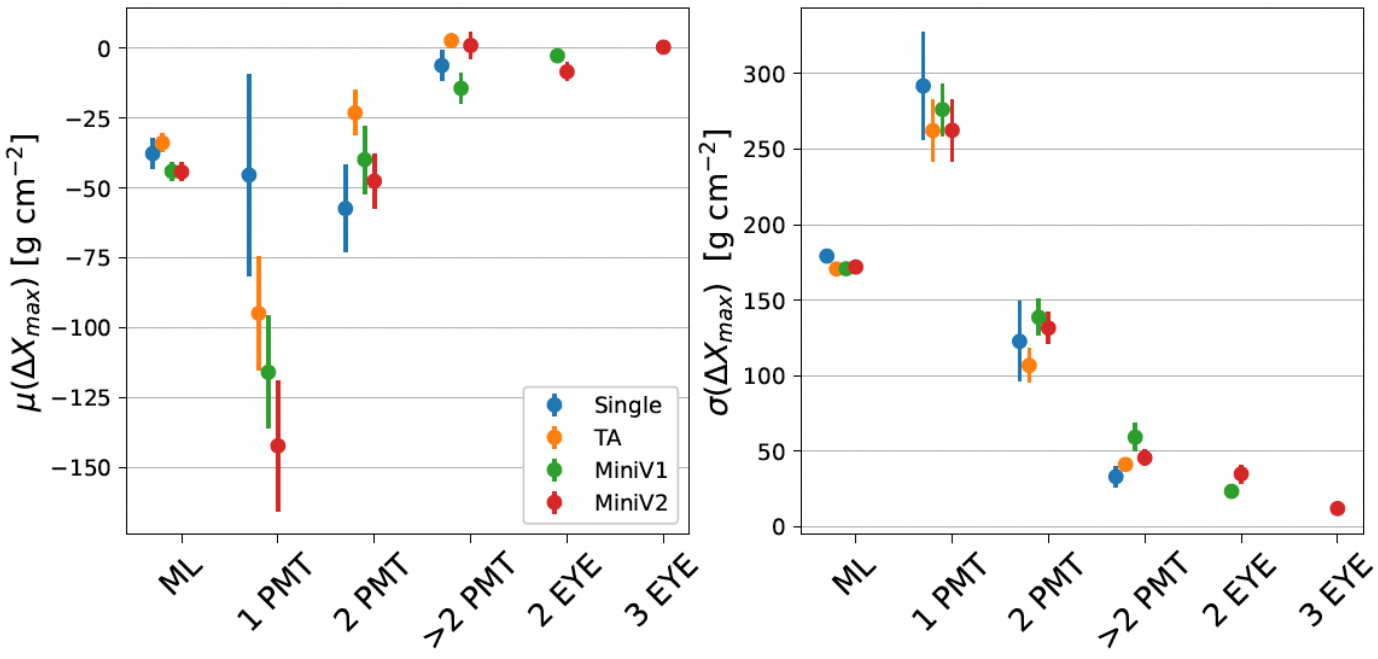
- TDR cuts
  - Successful minimization
  - X_{max} in FOV
  - Relative uncertainty in X_{max} and energy both < 0.5
  - Absolute uncertainty in core x and core y both < 1000 m

Full Reconstruction: Summary

- 1 or 2 PMTs - poor resolutions

- 3 or more PMTs
  - X_{max} ~ 40 – 50 g cm^{-2}
    Energy ~ 10%
    Core res ~ 700 m
    Angular res ~ 7 deg

- Stereo performs even better!
  - Slightly unrealistic results…
    - No shower-to-shower fluctuations
    - No atmospheric/calibration uncertainties
    - No simulation of electronic response (e.g. saturated signals)

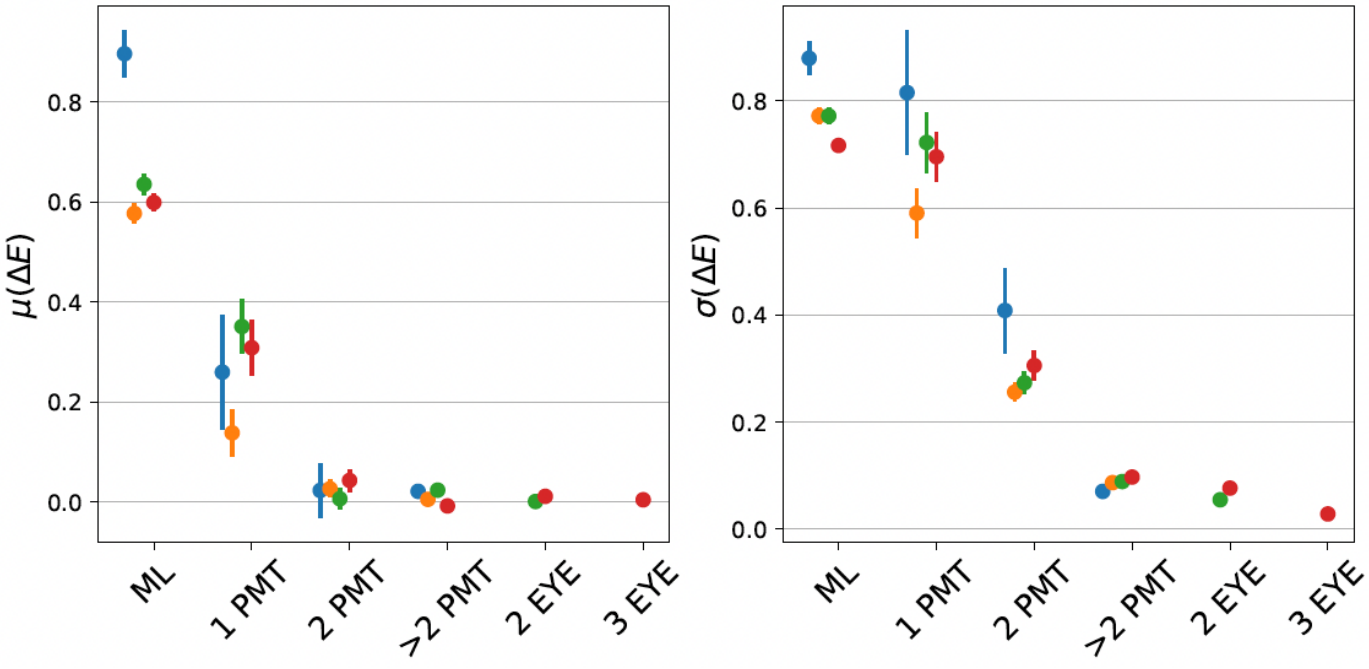


Full Reconstruction: Summary

- 1 or 2 PMTs - poor resolutions

- 3 or more PMTs
  - X_{max} ~ 40 – 50 g cm^{-2}
    Energy ~ 10%
    Core res ~ 700 m
    Angular res ~ 7 deg

- Stereo performs even better!
  - Slightly unrealistic results…
    - No shower-to-shower fluctuations
    - No atmospheric/calibration uncertainties
    - No simulation of electronic response (e.g. saturated signals)

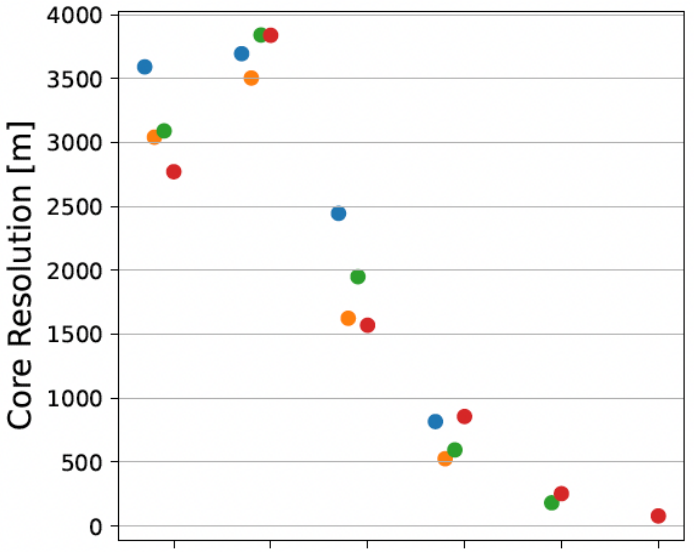


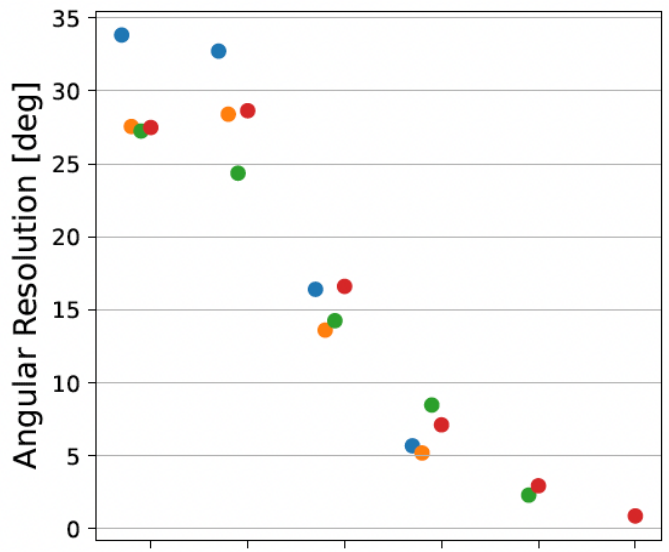
Full Reconstruction: Summary

- 1 or 2 PMTs - poor resolutions

- 3 or more PMTs
  - X_{max} ~ 40 – 50 g cm^{-2}
    Energy ~ 10%
    Core res ~ 700 m
    Angular res ~ 7 deg

- Stereo performs even better!
  - Slightly unrealistic results…
    - No shower-to-shower fluctuations
    - No atmospheric/calibration uncertainties
    - No simulation of electronic response (e.g. saturated signals)





Contents

① The Fluorescence detector Array of Single-pixel Telescopes (FAST)

② Training a ML model to predict shower parameters with FAST
- Model training & performance in simulations

③ Applying the trained model to real events detected by FAST
- **Issues/challenges**
- **Degeneracy in reconstructed solutions**

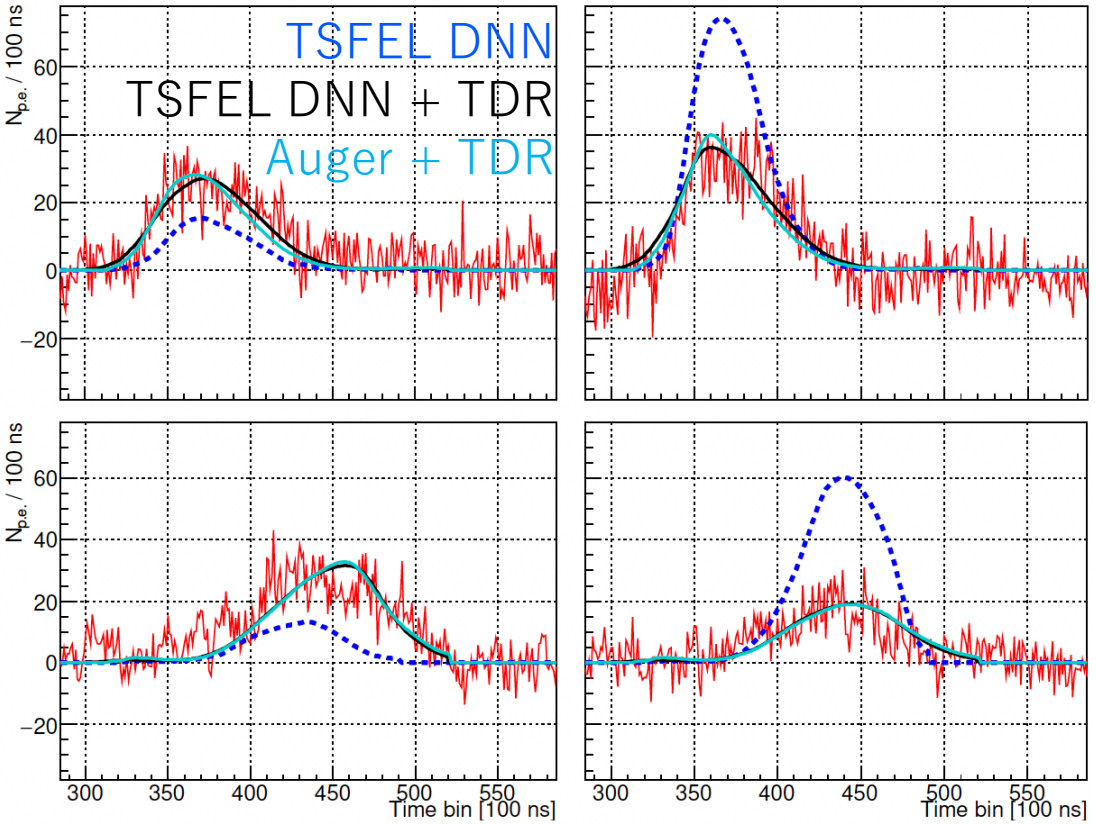
TSFEL DNN + TDR: Real data

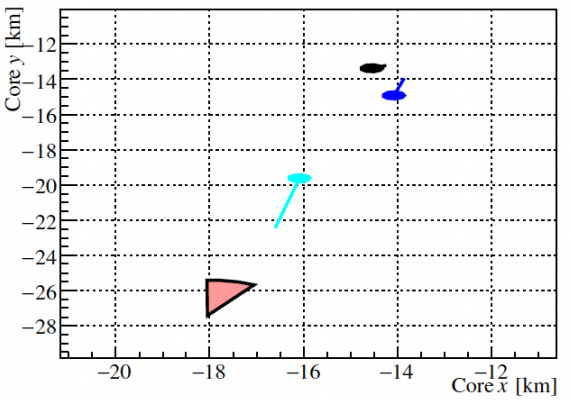
Machine learning first guess

- From previous analysis, need > 2 triggered pixels for reasonable guess

- Number of events with > 2 triggered pixels
  - **FAST@TA:** 70 (out of ~440)
  - **FAST@Auger:** 75 (out of ~230)

- After reconstruction cuts (no X_{max} in FOV cut) number of fits which "reasonably" match data
  - **FAST@TA:** ~50 events
  - **FAST@Auger:** ~30 events





RECONTRUCTION RESULTS

X_{max}: 516 ± 16 (537.73) {863} [g cm^{-2}]

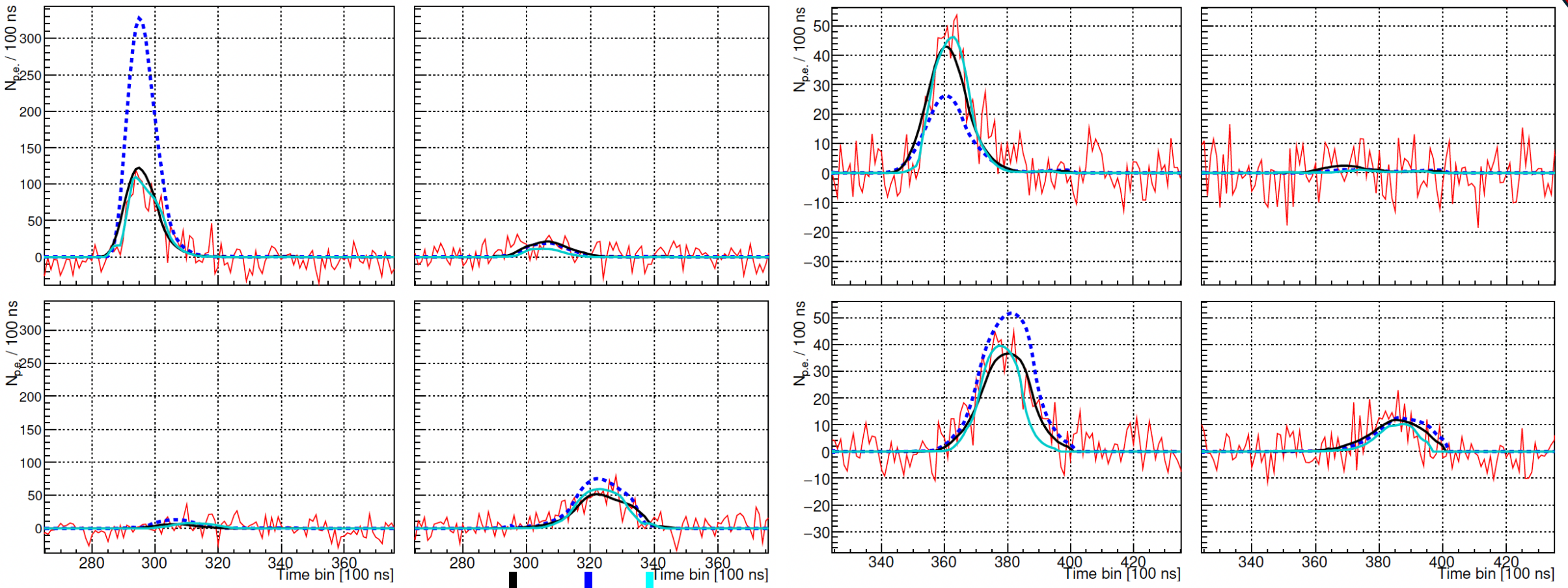
E: 90.92 ± 3.91 (67.39) {19.26} [EeV]

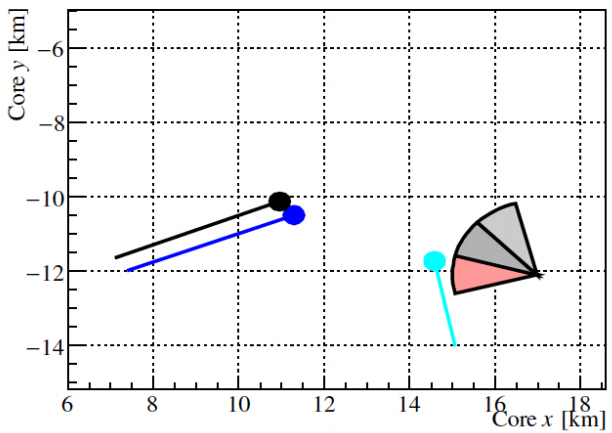
θ: 4.1 ± 0.5 (11.74) {35.3} [deg]

ϕ: 28.7 ± 4.2 (77.35) {-100.2} [deg]

x: -14551 ± 23 (-14080) {-16089} [m]

y: -13378 ± 141 (-14920) {-19611} [m]





RECONTRUCTION RESULTS

X_{max}: 815 ± 25 (716.34) {605} [g cm^{-2}]

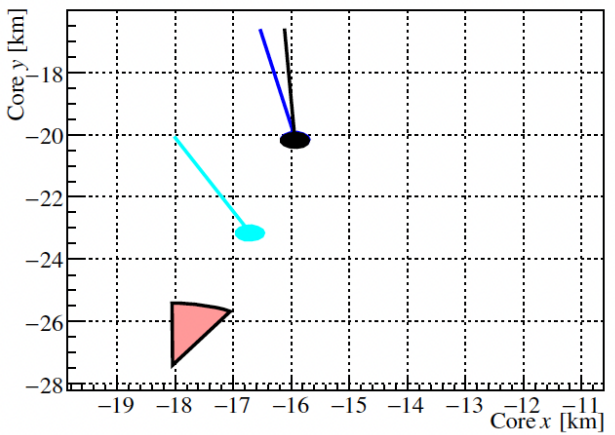
***E*:** 4.62 ± 0.22 (7.2) {1.41} [EeV]

θ: 55.9 ± 0.4 (56.71) {28} [deg]

ϕ: -158.6 ± 0.4 (-158.98) {-78.1} [deg]

***x*:** 10957 ± 56 (11290) {14583} [m]

***y*:** -10132 ± 50 (-10500) {-11739} [m]



RECONTRUCTION RESULTS

X_{max}: 904 ± 40 (1079.17) {1003} [g cm^{-2}]

***E*:** 3.36 ± 0.41 (4.03) {1.07} [EeV]

θ: 46.2 ± 0.6 (45.89) {42.4} [deg]

ϕ: 92.9 ± 2.3 (99.69) {112.8} [deg]

***x*:** -15937 ± 58 (-15930) {-16713} [m]

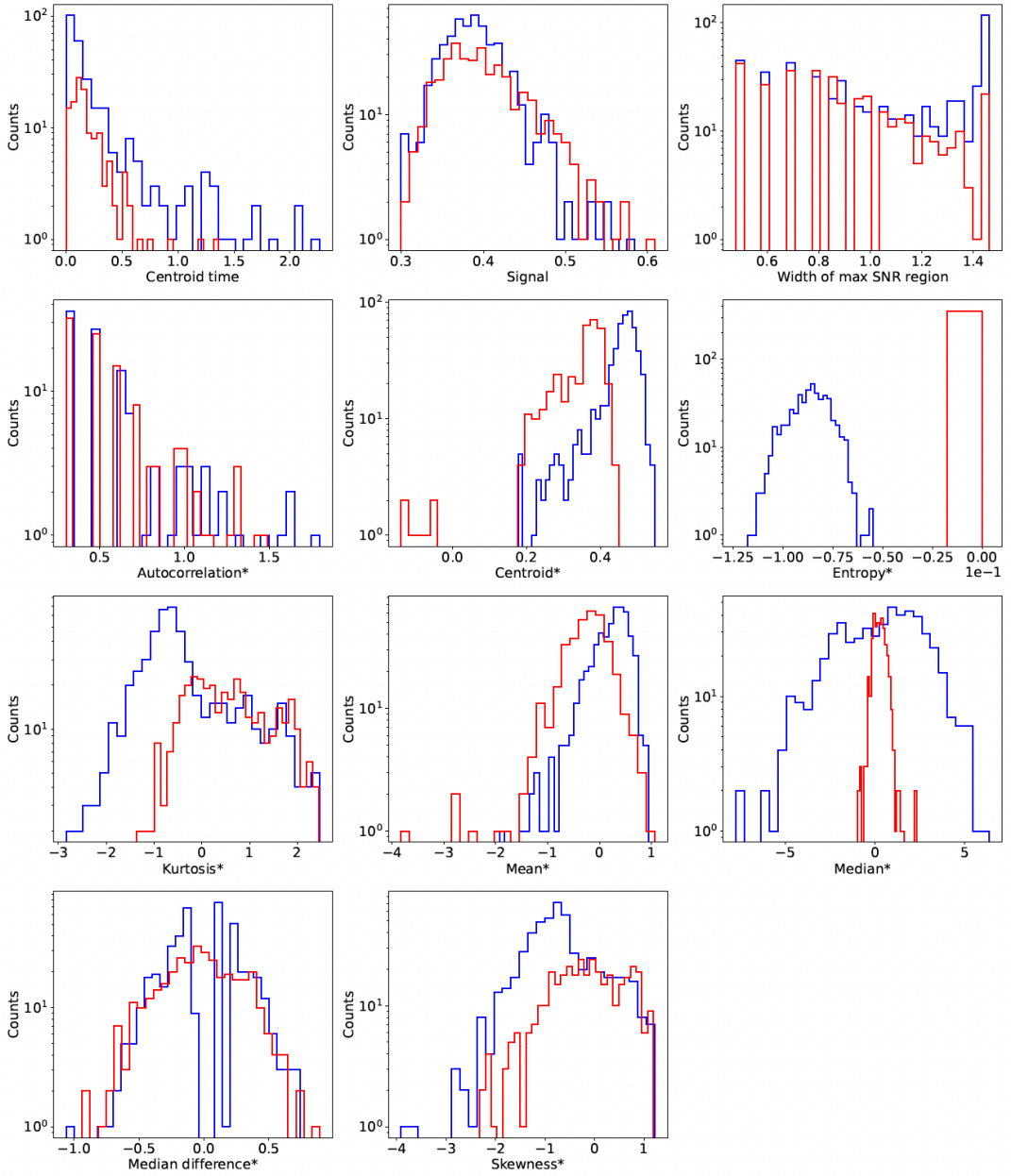
***y*:** -20182 ± 130 (-20150) {-23163} [m]

TSFEL DNN + TDR: Real data

Inputs to machine learning model:

- Red – Take Auger reconstructed values for coincidence events, perform FAST simulation with these values (+ nominal background noise) and calculate inputs
- Blue – Calculate inputs for coincidence data directly

- See that for a few parameters there are some systematic differences → will impact ML recon!
  - Must ensure that trace features are consistent between data and simulations



Summary and Future

Summary

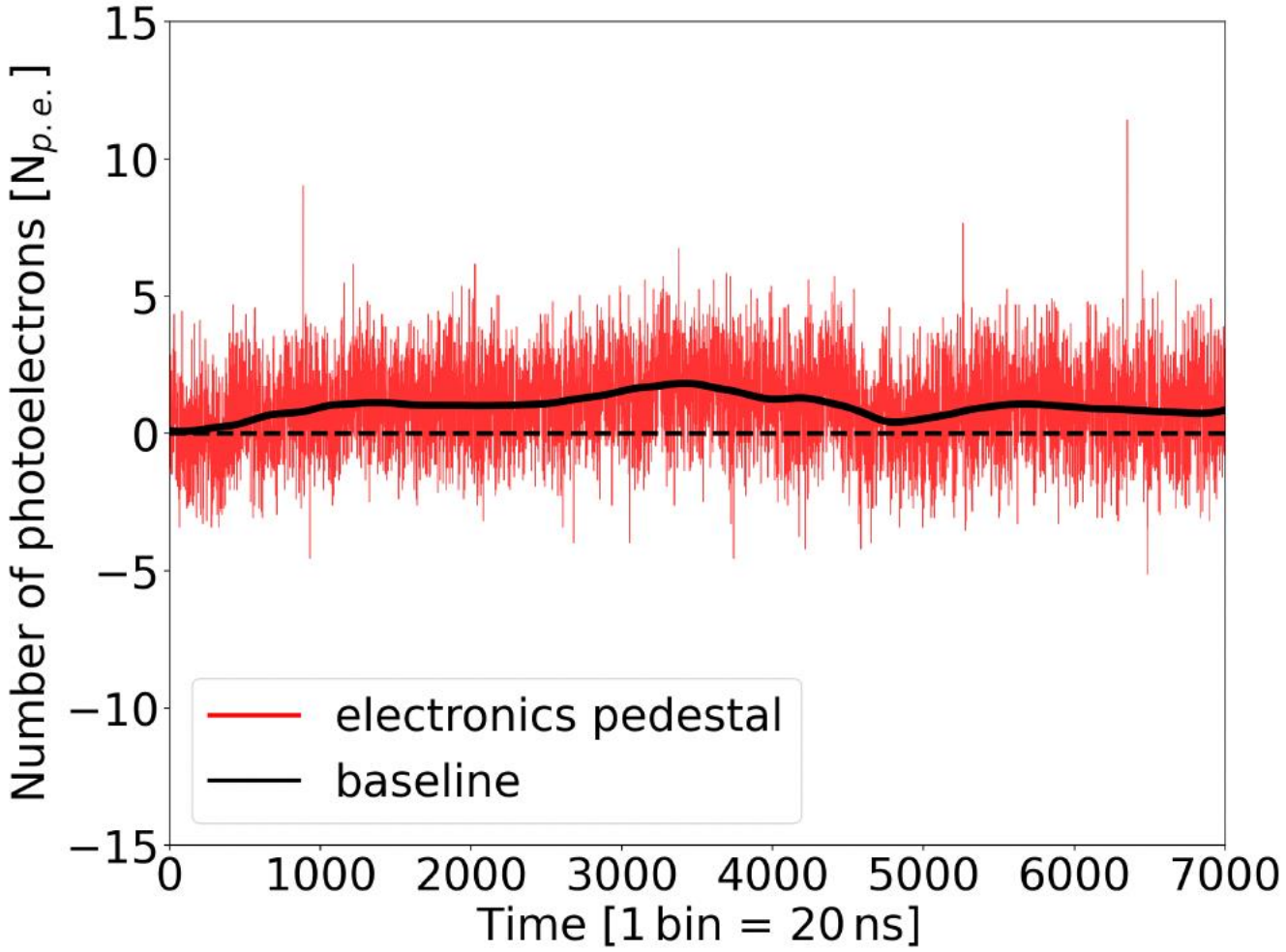
- With only 1 or 2 pixels, can't predict parameters very well
- Need 3 or more pixels in one eye or stereo observation
- ML + TDR **w. FAST-MiniV2** shows ideal resolutions in X_{max}~30 g cm^{-2}, $E <$ 10% - promising! But need to include additional reality…
- Real data analysis: degeneracies with monocular reconstruction, some inconsistencies between simulations / data traces.

Future

- Focus on stereo observation, ensure consistency between trace properties in data and simulations
- Add in additional uncertainties to simulations and re-evaluate expected resolutions (e.g. different atmospheres, PMT efficiency maps etc.)

Backup

Example of floating baseline



Basic DNN: Details

- Feed-forward, deep neural network
  - ReLU activation function for hidden layers, Adam optimizer, MSE loss

- 3 inputs from each PMT with SNR > 6 (other PMT inputs set to 0):
  - Integrated signal – $S_i = \sum_{j=k_{\text{start}}}^{k_{\text{stop}}-1} s_j$ signal in jth bin
  - Centroid time – $\bar{t}_i = \frac{\sum_{j=k_{\text{start}}}^{k_{\text{stop}}-1} s_j t_j}{\sum_{j=k_{\text{start}}}^{k_{\text{stop}}-1} s_j}$ time of jth bin
  - Pulse height – $h_i = \max\left(\{s_{k_{\text{start}}}, s_{k_{\text{start}}+1}, \ldots, s_{k_{\text{stop}}-1}\}\right)$

where k_{start} and k_{stop} are determined from maximum SNR region

$$\text{SNR} = \frac{S}{N} = \frac{\sum_{i=k_{\text{start}}}^{k_{\text{stop}}-1} S_i}{\sigma_{\text{nsb}}\sqrt{k_{\text{stop}} - k_{\text{start}}}}$$

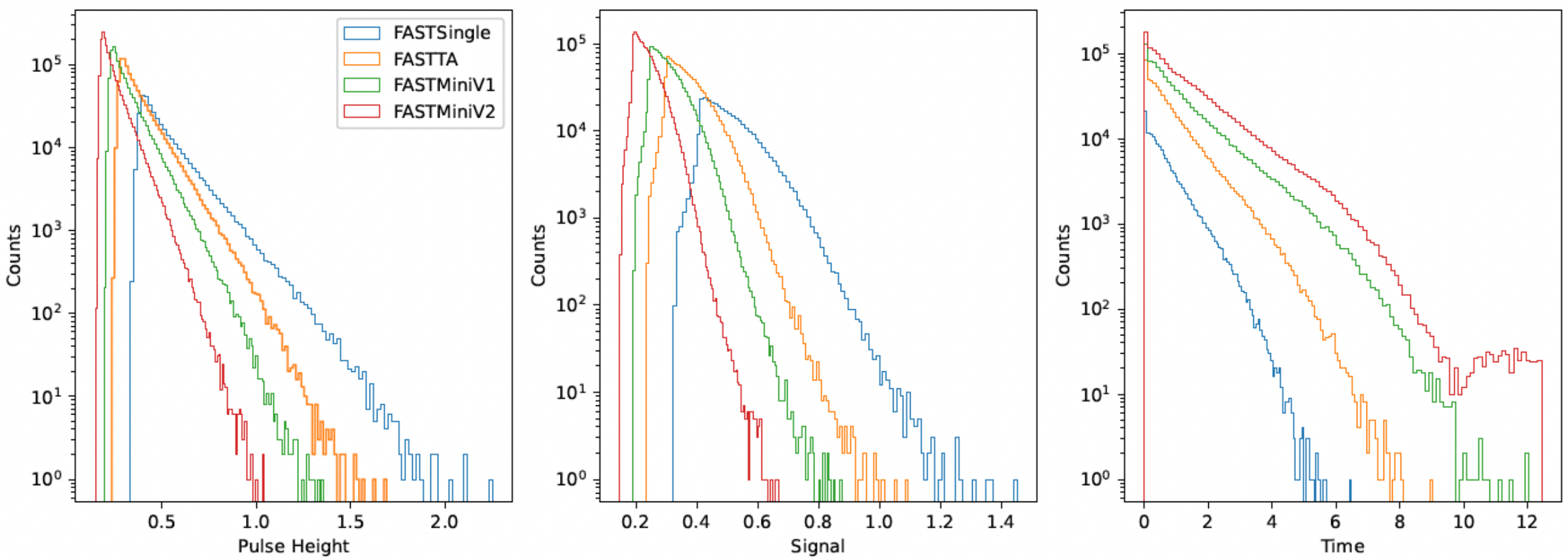
S_i

Basic DNN: Input normalisation

- Integrated signal:
  - Take log10 $\hat{S}_i = \log_{10}(S_i)$
  - Divide by average total (logarithmic) signal over all events $\hat{\hat{S}}_i = \frac{\hat{S}_i}{\hat{S}_0}$
- Centroid time
  - Subtract earliest centroid time in event $\hat{t}_i = \bar{t} - \bar{t_0}$
  - Divide by standard deviation of all centroid times in data set $\hat{\hat{t}}_i = \frac{\hat{t}_i}{\sigma_t}$
- Pulse height:
  - Take log10 $\hat{h}_i = \log_{10}(h_i)$
  - Divide by average (logarithmic) height over all events $\hat{\hat{h}}_i = \frac{\hat{h}_i}{\hat{h_0}}$

Actual inputs

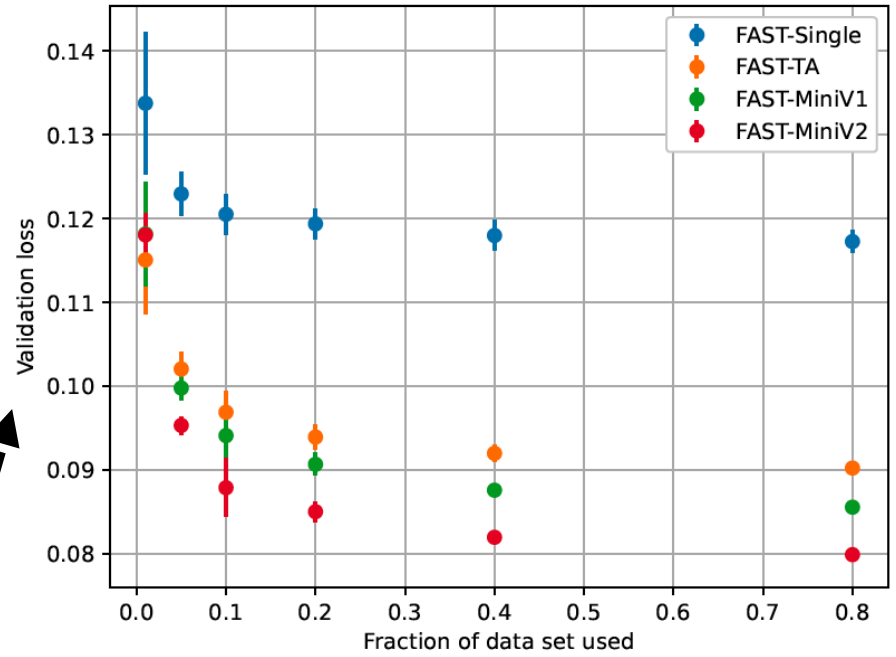
Basic DNN: Input parameter distributions

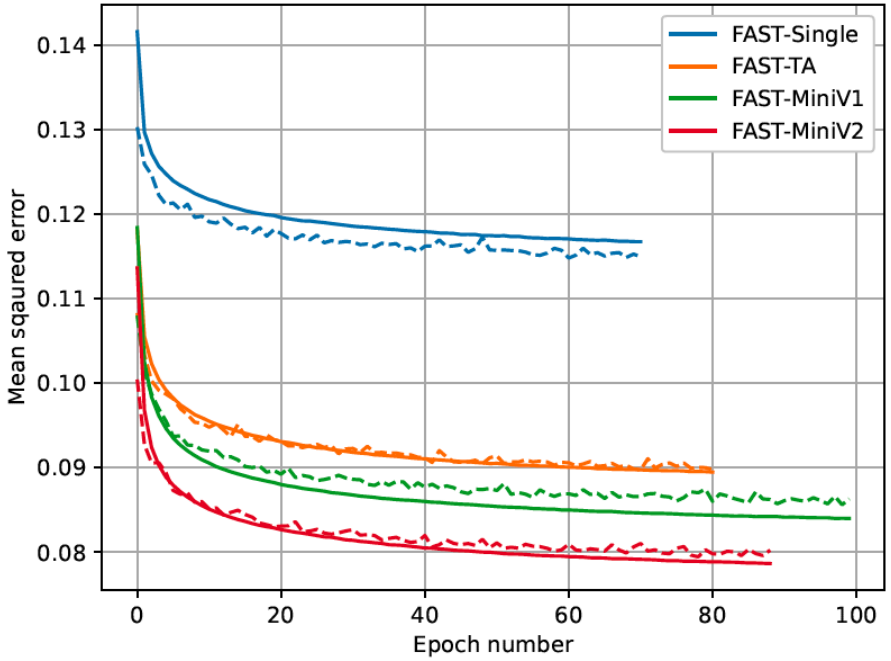


Basic DNN: Training tests

- Initial training setup
  - Learning rate 0.001
  - 90%:10% Train-Test split
  - Epochs 100 (patience of 10)
  - Batch size of 64
  - Layer structure
    (Number of PMTs x 3)/128/64/6

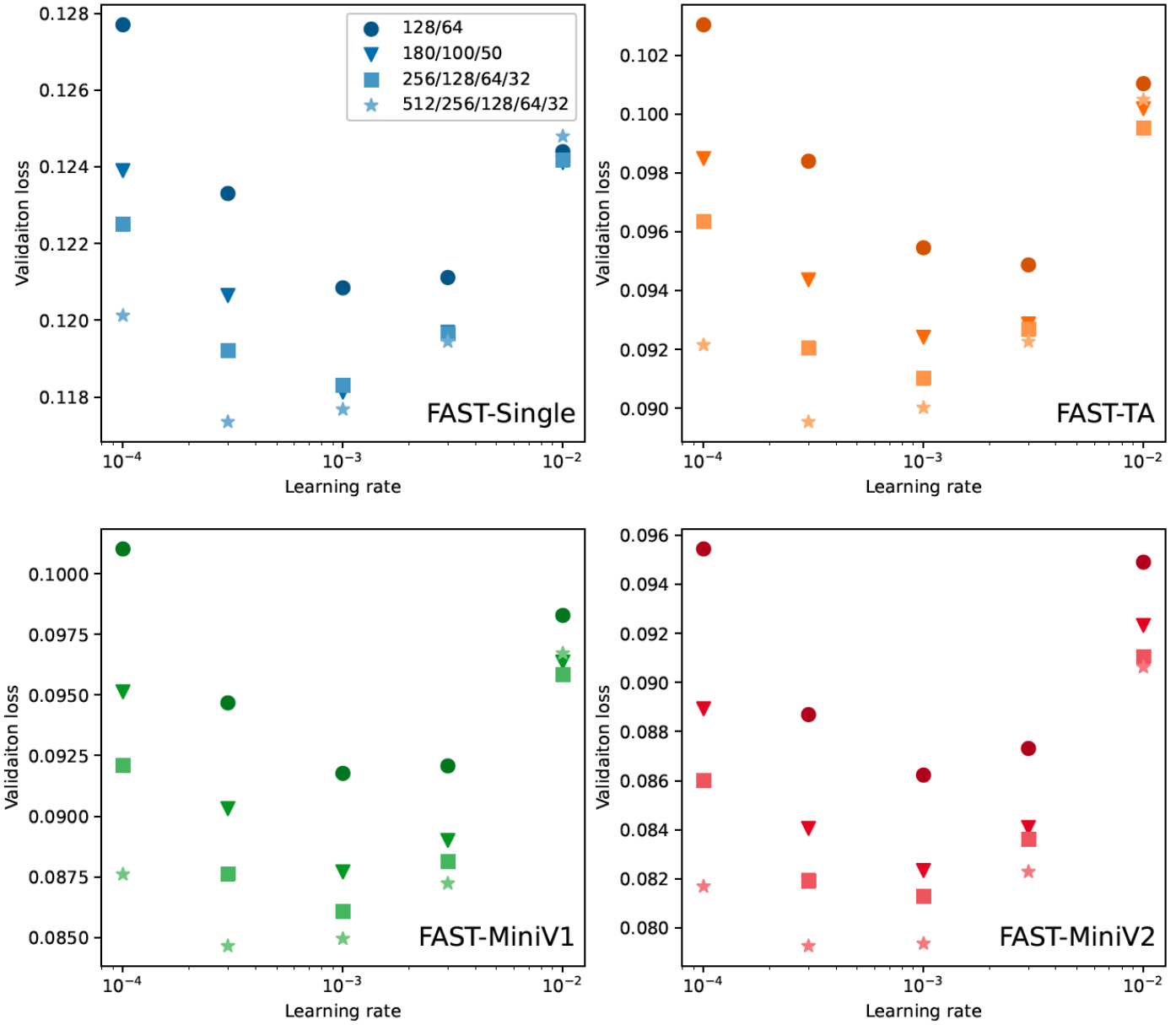
- Check learning curves, validation loss vs. % of data set used
  - Train 10 times using different 10% as validation set each time and take mean & std. of results





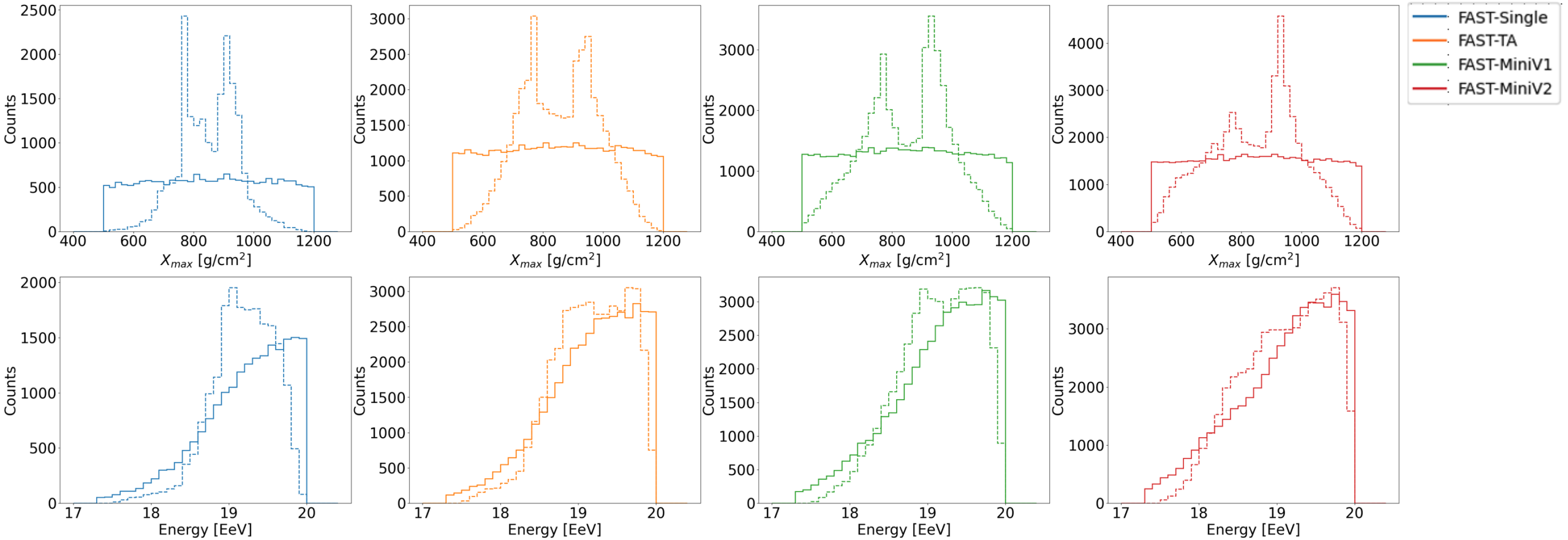
Basic DNN: Hyperparameter Tuning

- Basic hyperparameter tuning
- Varying the learning rate and the number of layers/nodes in each layer
- Best result for each layout with 5-layer structure, 0.003 learning rate
- Should experiment with “Optuna” in the future



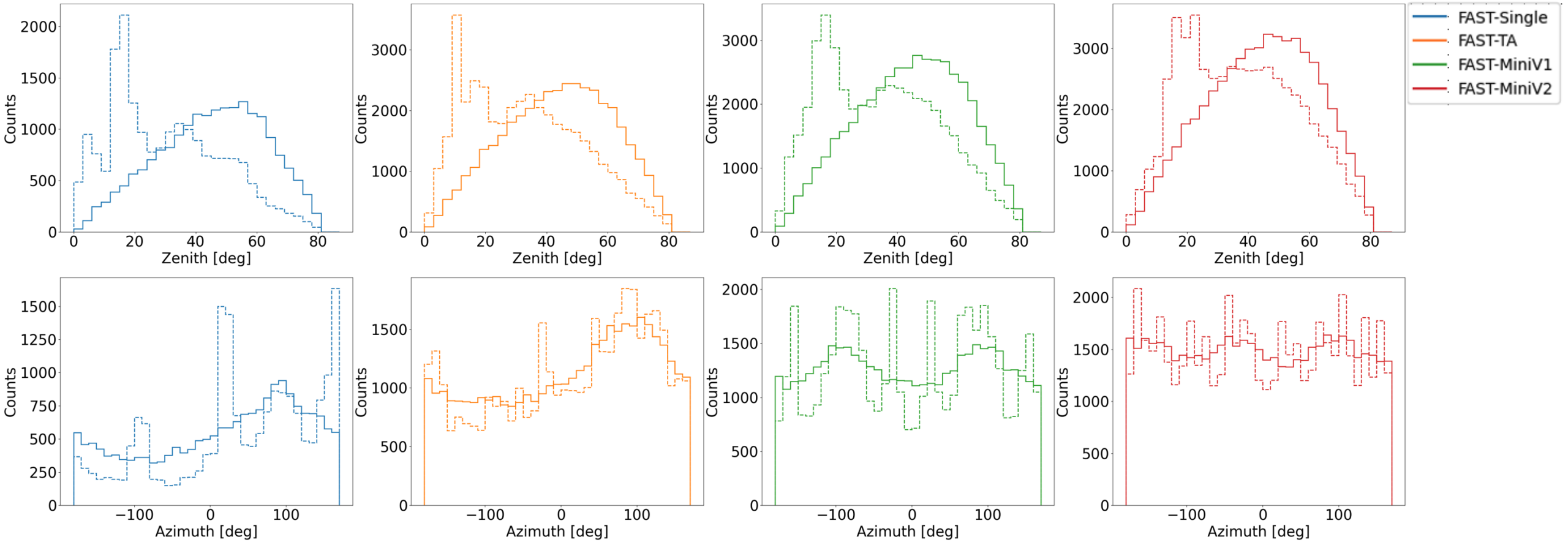
Basic DNN: Results

- Apply hyper-parameter tuned model on test data set
- True (solid line) and reconstructed (dashed line) output distributions



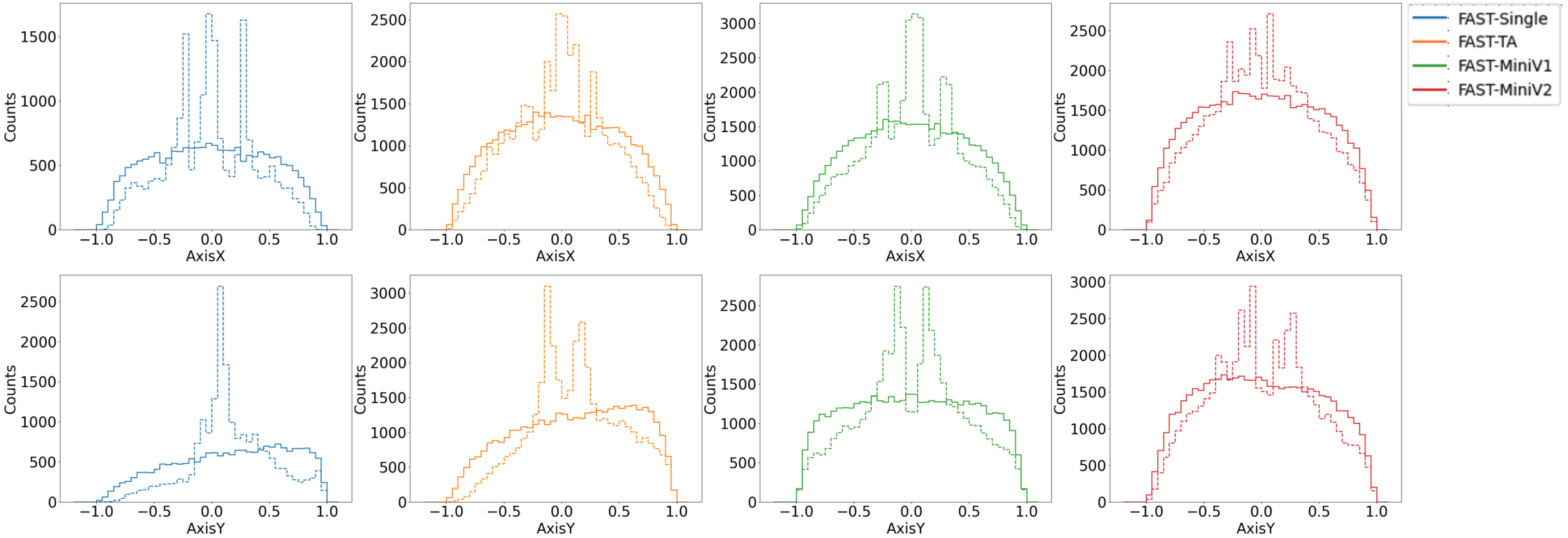
Basic DNN: Results

- Apply hyper-parameter tuned model on test data set
- True (solid line) and reconstructed (dashed line) output distributions



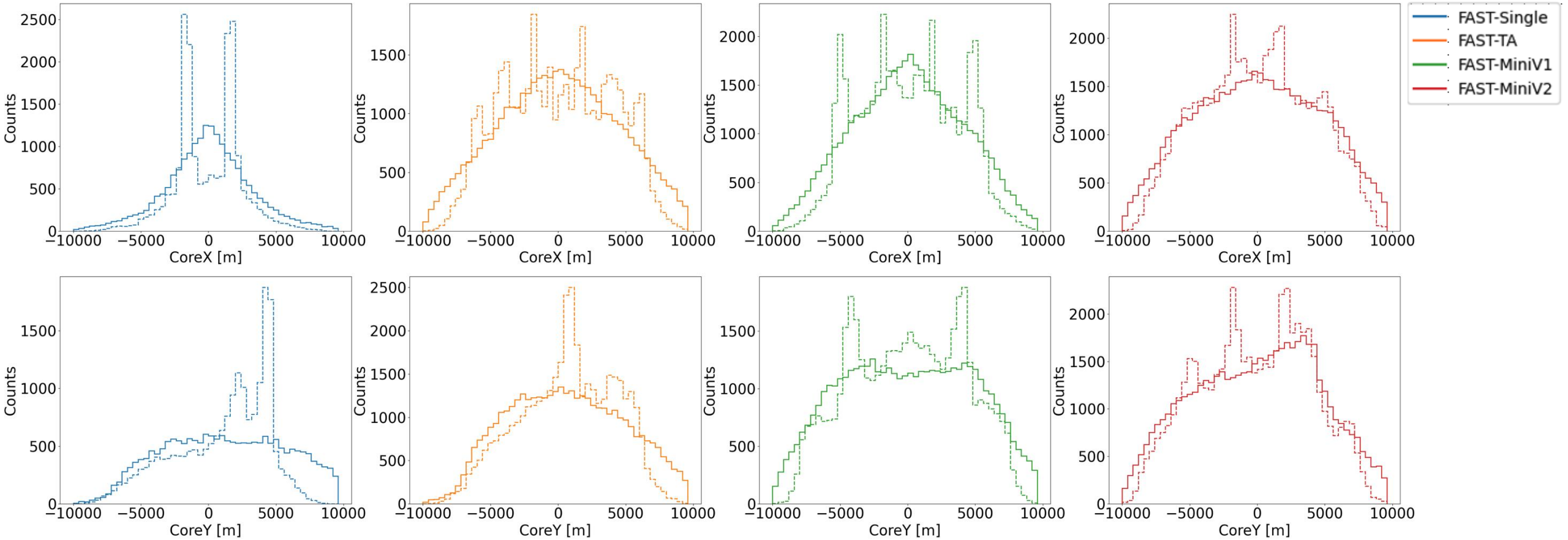
Basic DNN: Results

- Apply hyper-parameter tuned model on test data set
- True (solid line) and reconstructed (dashed line) output distributions



Basic DNN: Results

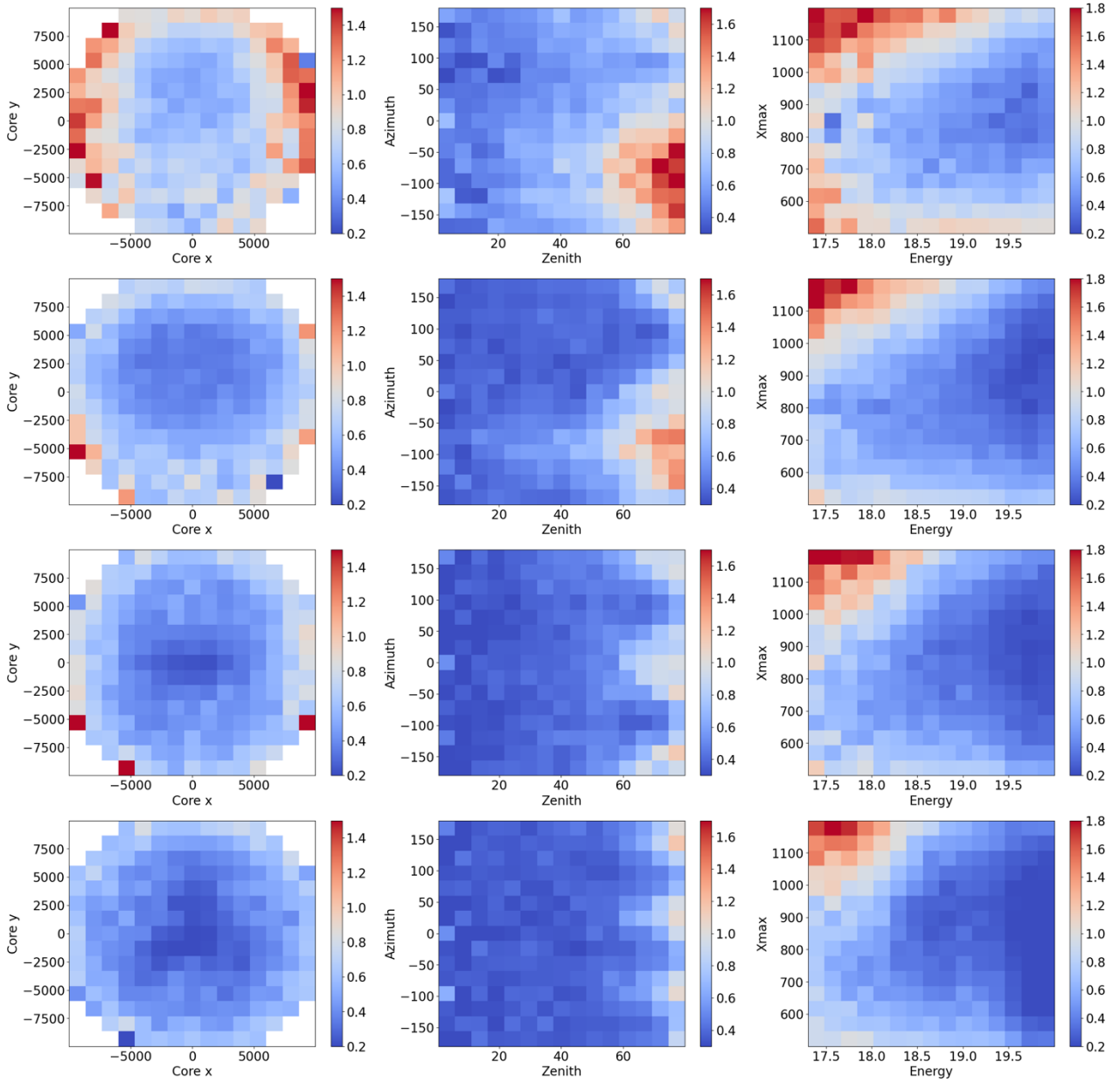
- Apply hyper-parameter tuned model on test data set
- True (solid line) and reconstructed (dashed line) output distributions



Basic DNN: Results

- Another diagnostic – 2D histograms of the average loss for slices in core location, arrival direction and X_{max}/energy

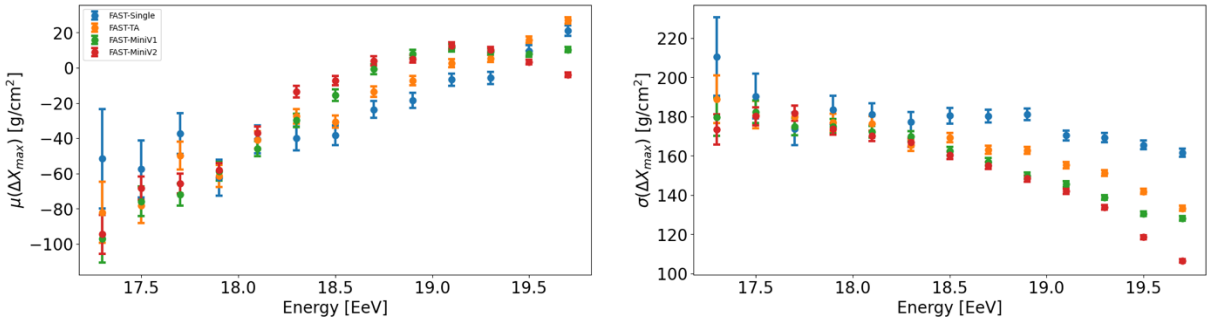
- Difficult regions
  - Low energy, high X_{max} - not present in data so OK
  - Showers coming from behind telescopes
  - Edge of array

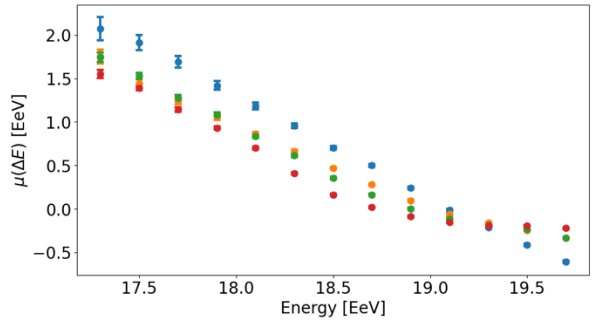


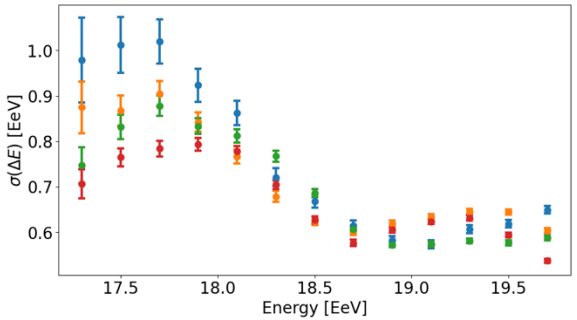
Basic DNN: Results

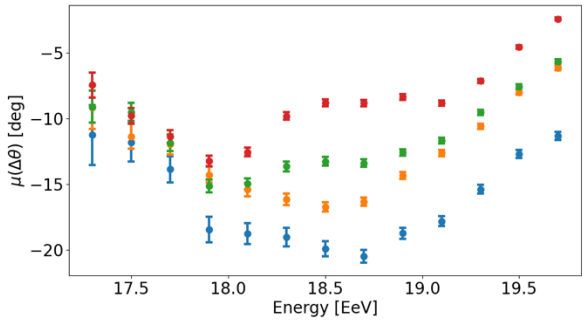
- Biases and resolutions in each parameter as a function of energy.

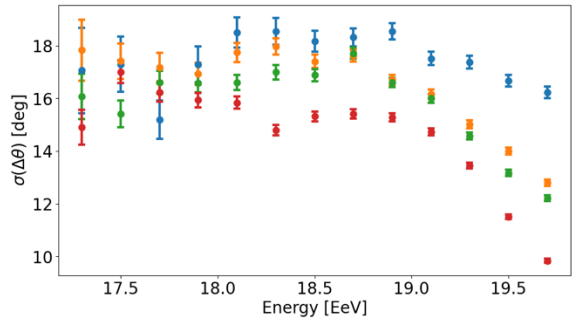
$$\Delta E = \ln(E_{rec}/E_{true})$$



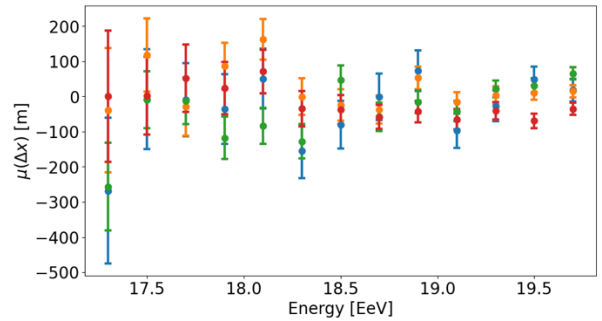


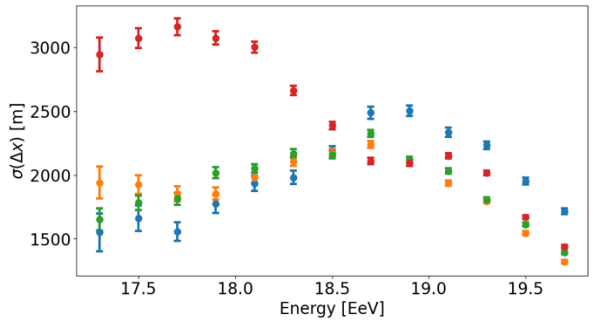


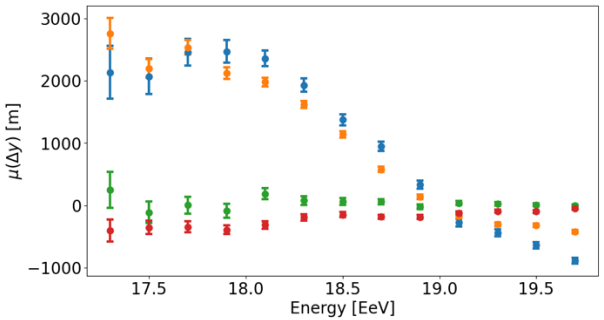


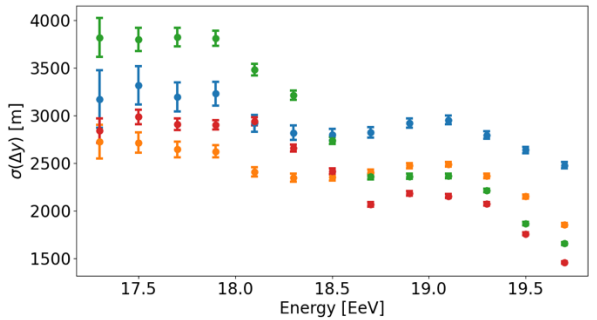


- Shape of some plots strange due to range of observable showers changing as function of energy









LSTM Network

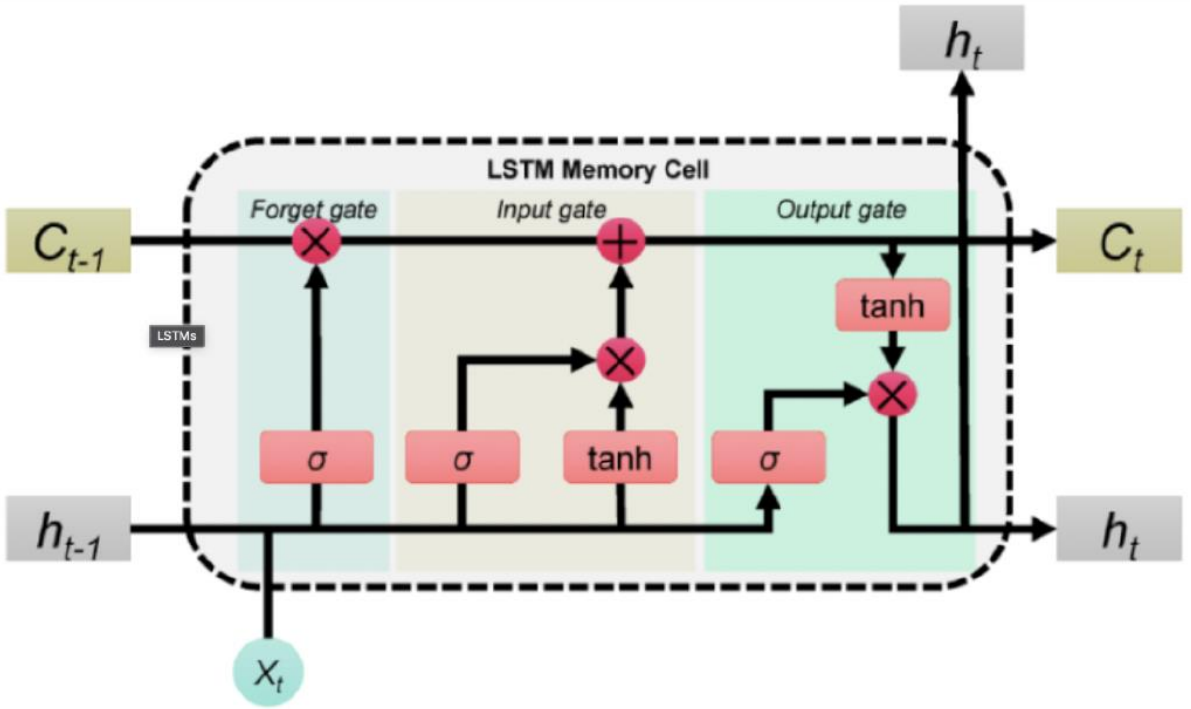
- Hidden state H_t (short term memory)
- Internal state C_t (long term memory)

Use same 600 bin segments
Cut first 10 and last 40 bins, re-bin by factor of 4, gives 100 bin traces (400 ns/bin)

- Processing trace $x = \{x_1, x_2, \cdots., x_n\}$
  - At each time step t, x_t & C_{t-1}& H_{t-1} are combined to give C_t and H_t
  - Updated using three different gates which use sigmoid/tanh activation functions. For a gate g

$$\vec{I_g} = U_g \vec{h_{t-1}} + W_g \vec{x_t} + \vec{b_g}$$

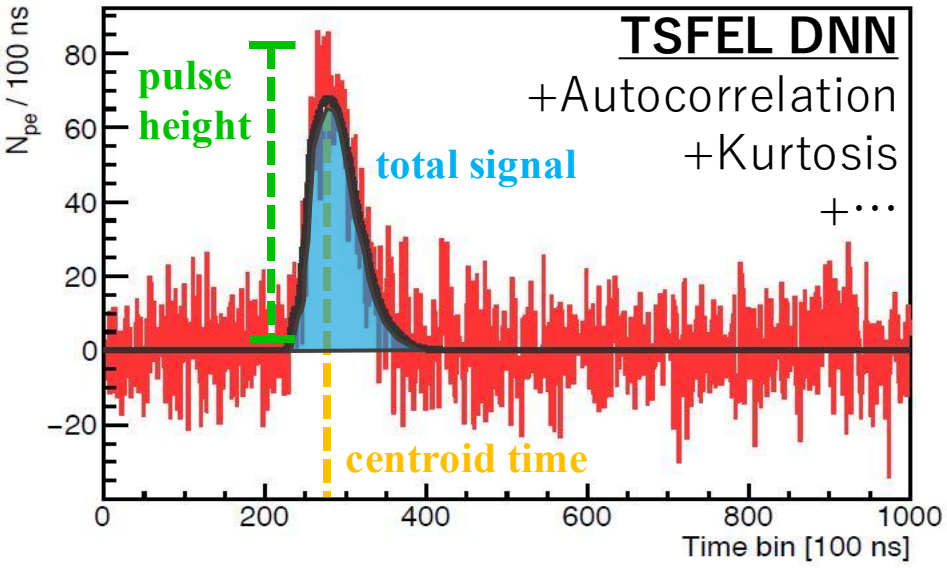
- **Goal:** Learn the matrices U_g & W_g and bias terms b_g for each gate



Models

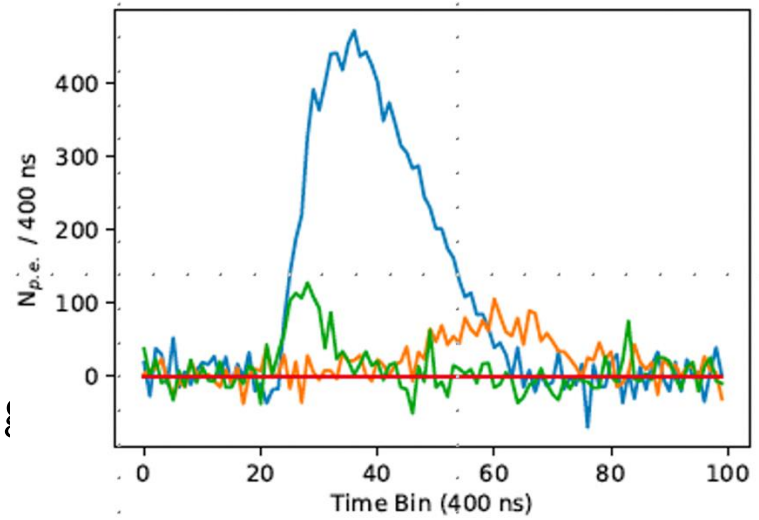
Models Tested:

- Basic DNN:
  - Identical architecture to previous studies
  - Use height of PMT pulse, total signal and timing from each PMT as inputs



- TSFEL DNN:
  - Extension of the Basic DNN - same feedfor
  - Use the TSFEL python library to extract additional trace features
  - Total of 11 inputs per PMT

- LSTM (Long-Short Term Memory):
  - Type of recurrent neural network
  - Often used for analyzing time series data
  - Extracts salient features from traces to use for learnin
  - 64 features extracted per PMT



More degeneracy examples

- Try events with ≥ 2 triggered pixels. After recon. cuts (no X_{max} in FOV cut)
  - **FAST@TA:** ~ 170 events, **FAST@Auger:** ~ 90 events
- Find "good fits" but tendency to guess larger energy / further away core
  → need stereo observation

Comparing which fit is better