HCal Update

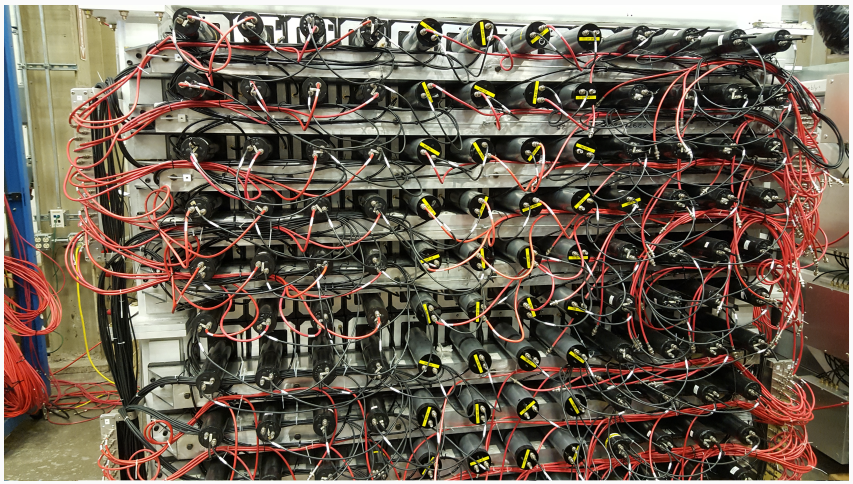
Scott Barcus and Juan Carlos Cornejo

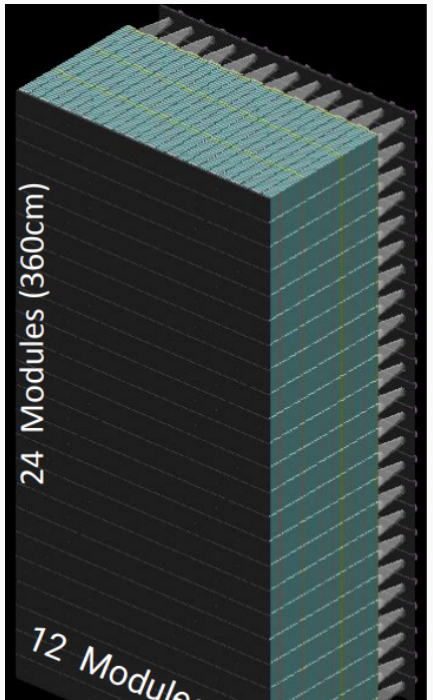
July 14th 2020

Jefferson Lab

HCal Overview

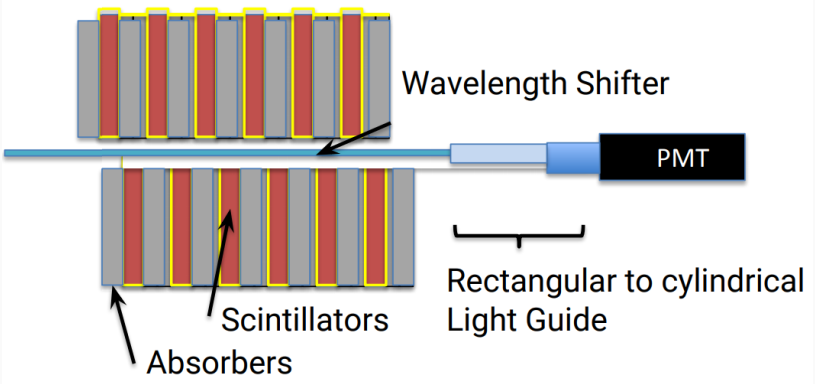
- Segmented calorimeter designed to detect multiple GeV protons and neutrons.
  - 288 PMT modules (12×24).
  - Four craneable subassemblies.
  - Weighs ≈40 tons.
  - Wavelength shifter.
  - Custom light guides.
  - LED fiber optics system.
- SBS dipole magnet separates scattered hadrons by charge on HCal's surface.
- Designed for good time resolution (goal 0.5 ns).
- Energy resolution ≈30%.

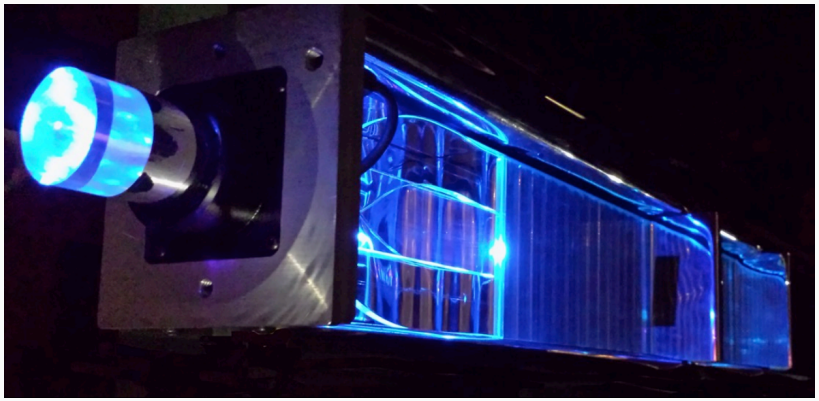




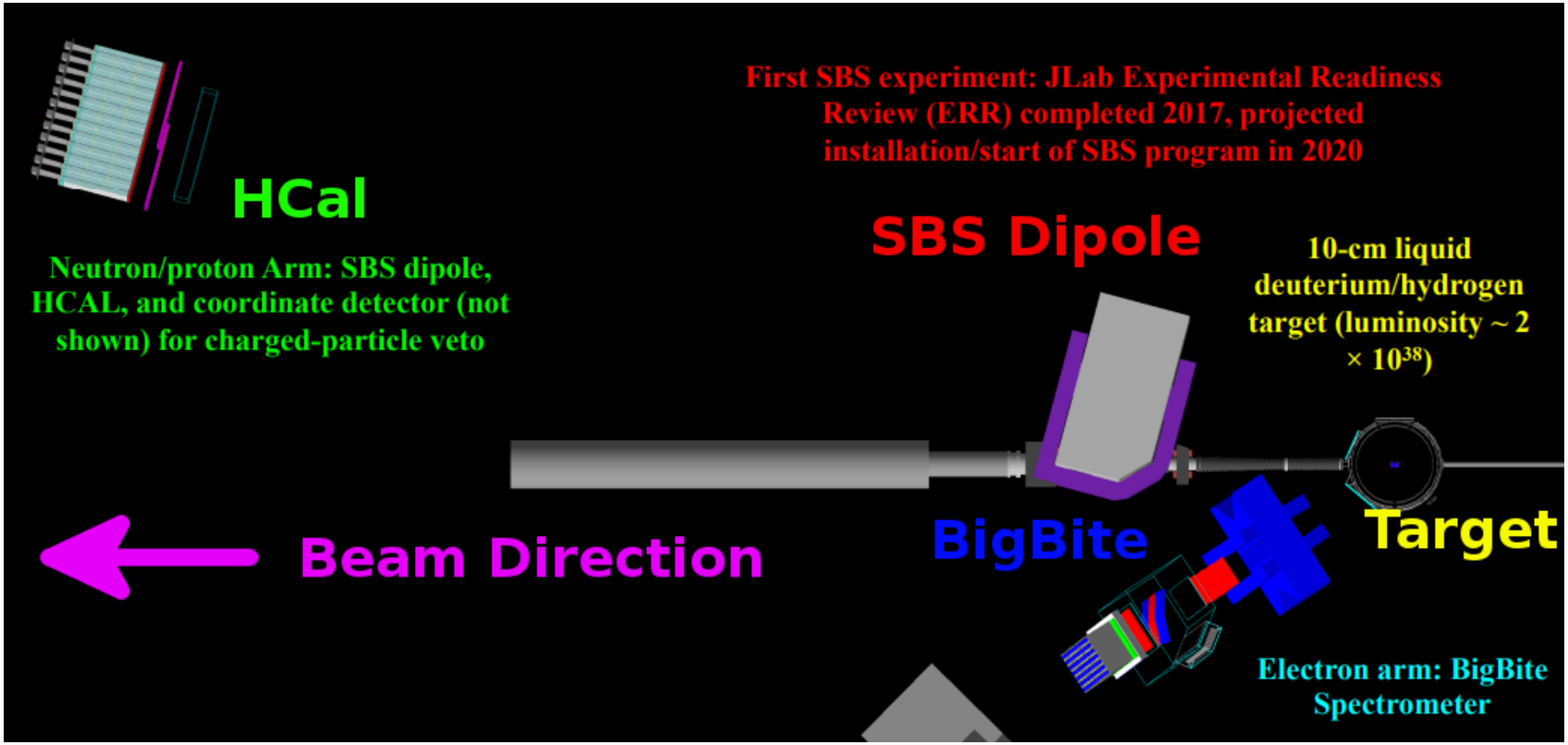
HCal Interior (288 Individual PMT Modules)

- 40 layers of iron absorbers alternate with 40 layers of scintillator.
- Iron layers cause the hadrons to shower.
- Scintillator layers sample the energy.
- Photons pass through a wavelength shifter increasing detection efficiency.
- Custom light guides transport photons to PMTs.
  - 192 12 stage 2" Photonis XP2262 PMTs.
  - 96 8 stage 2" Photonis XP2282 PMTs.

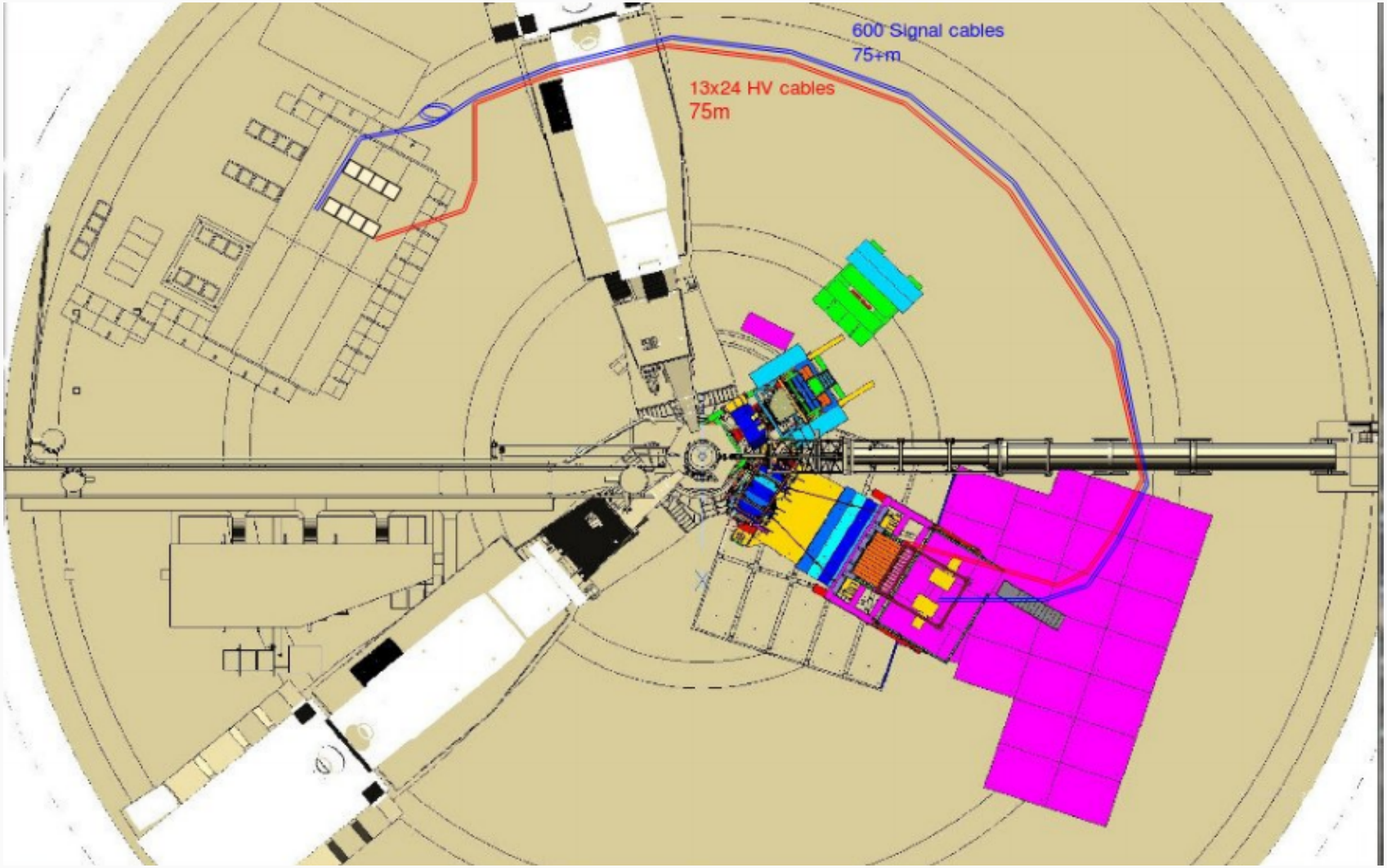




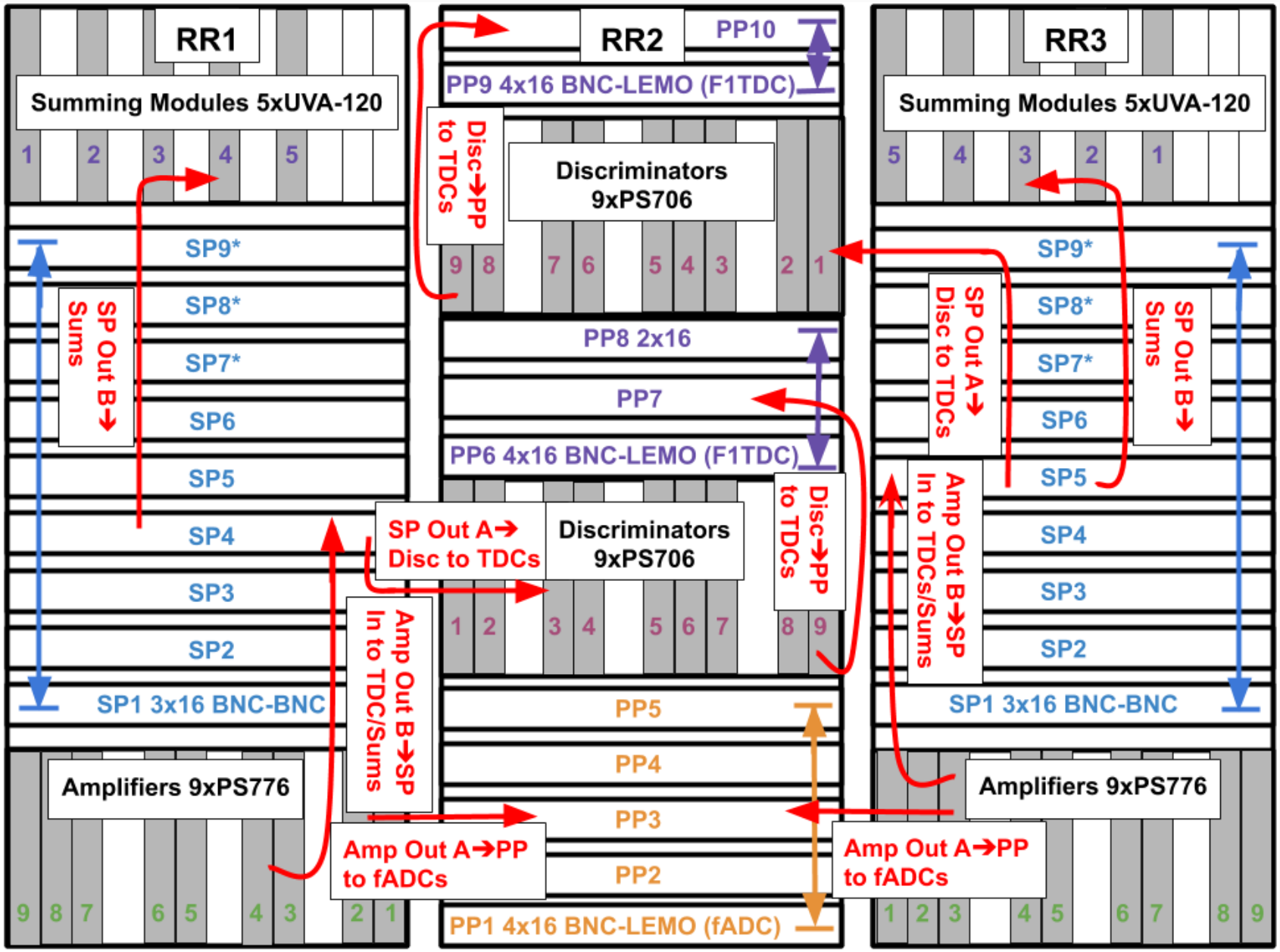
G_M^n Experimental Setup



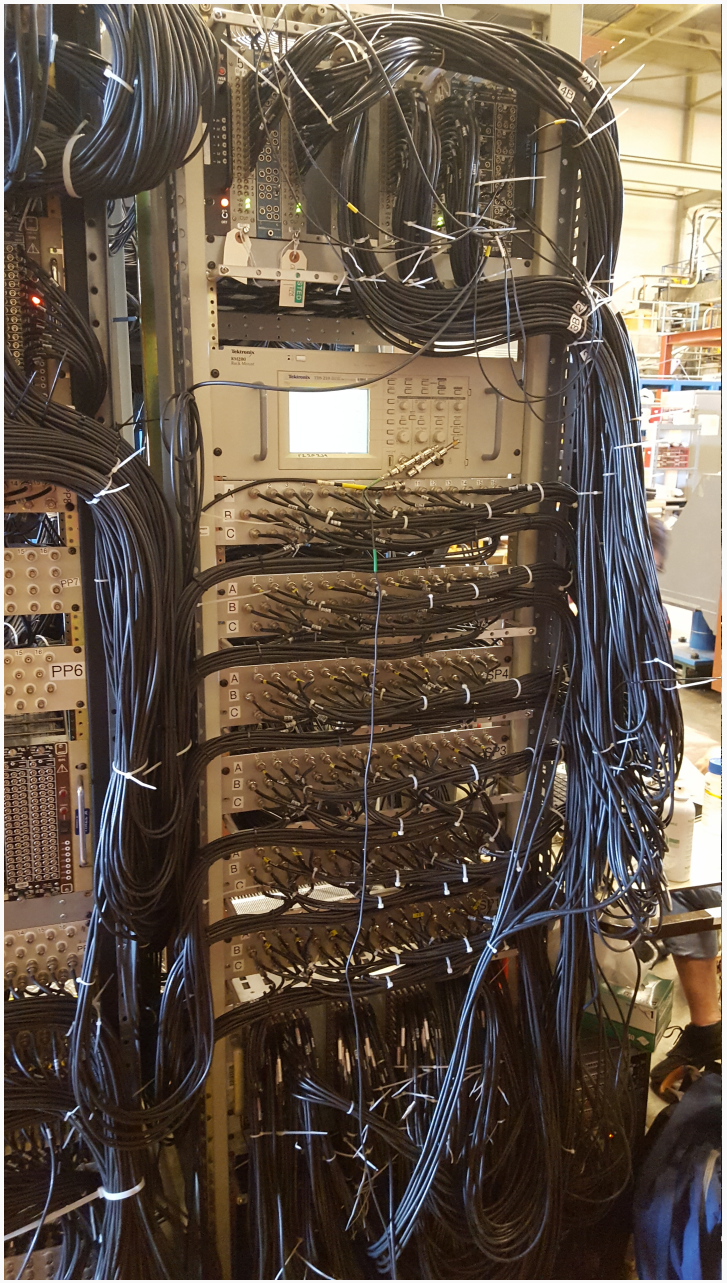
HCal Hall Layout



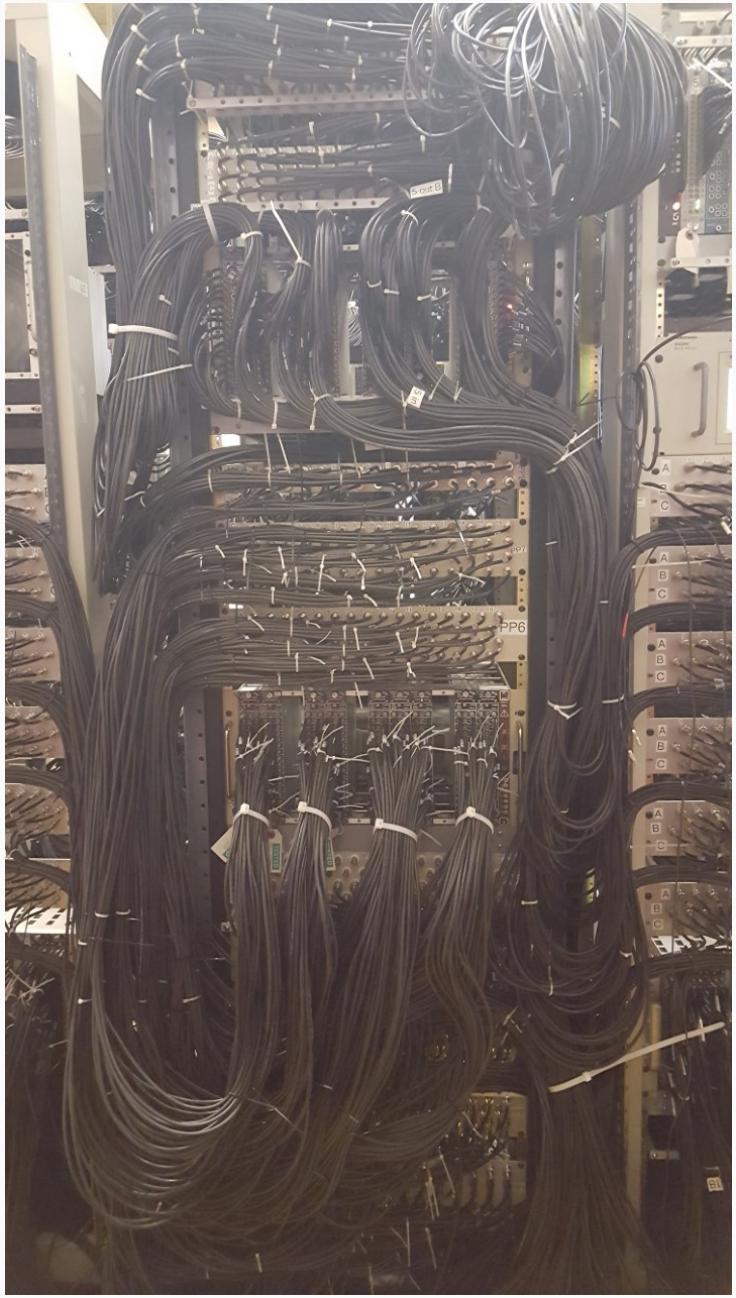
HCal Front End



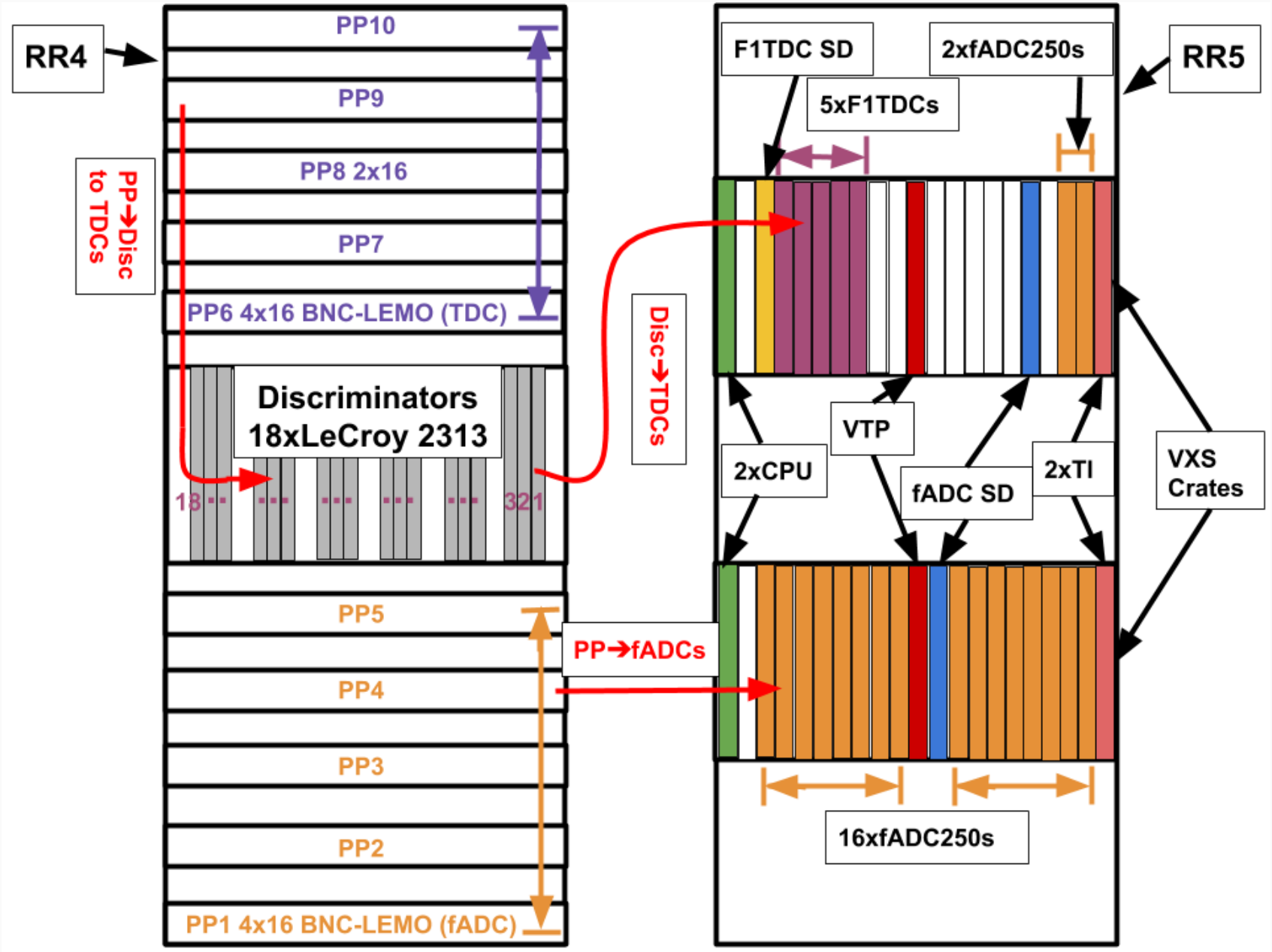
HCal Front End Cont.



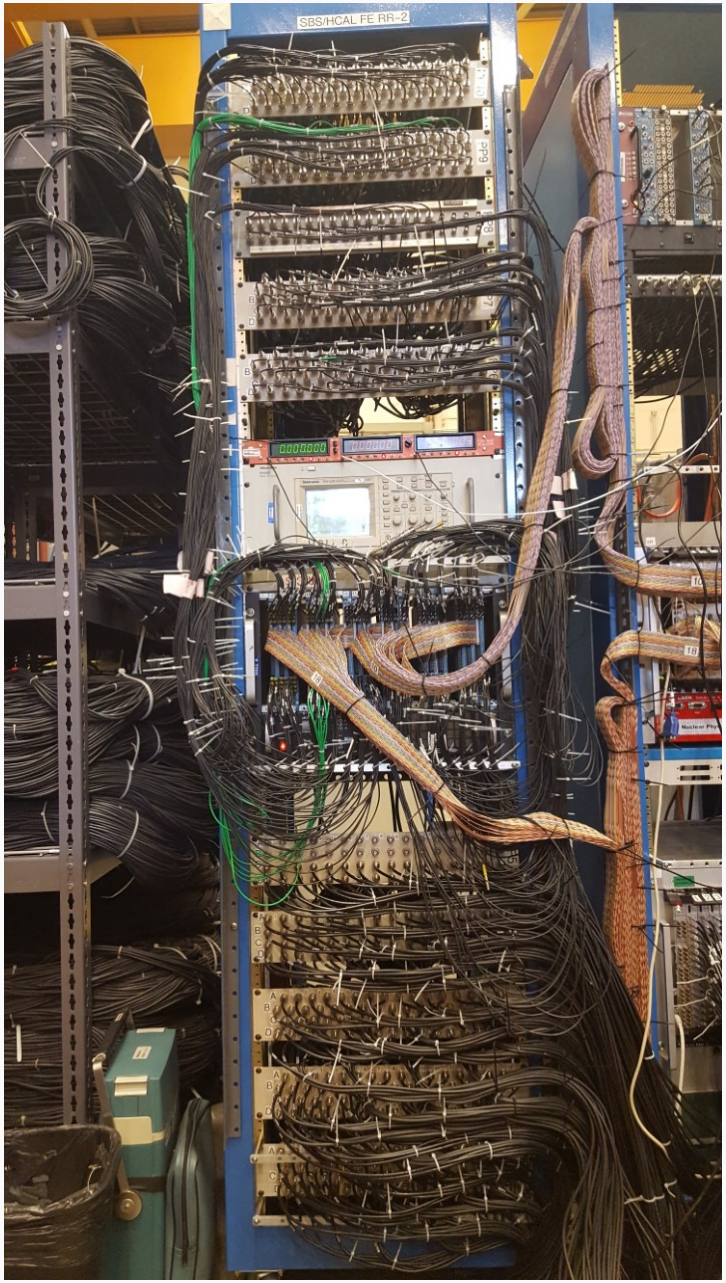
HCal Front End Cont.

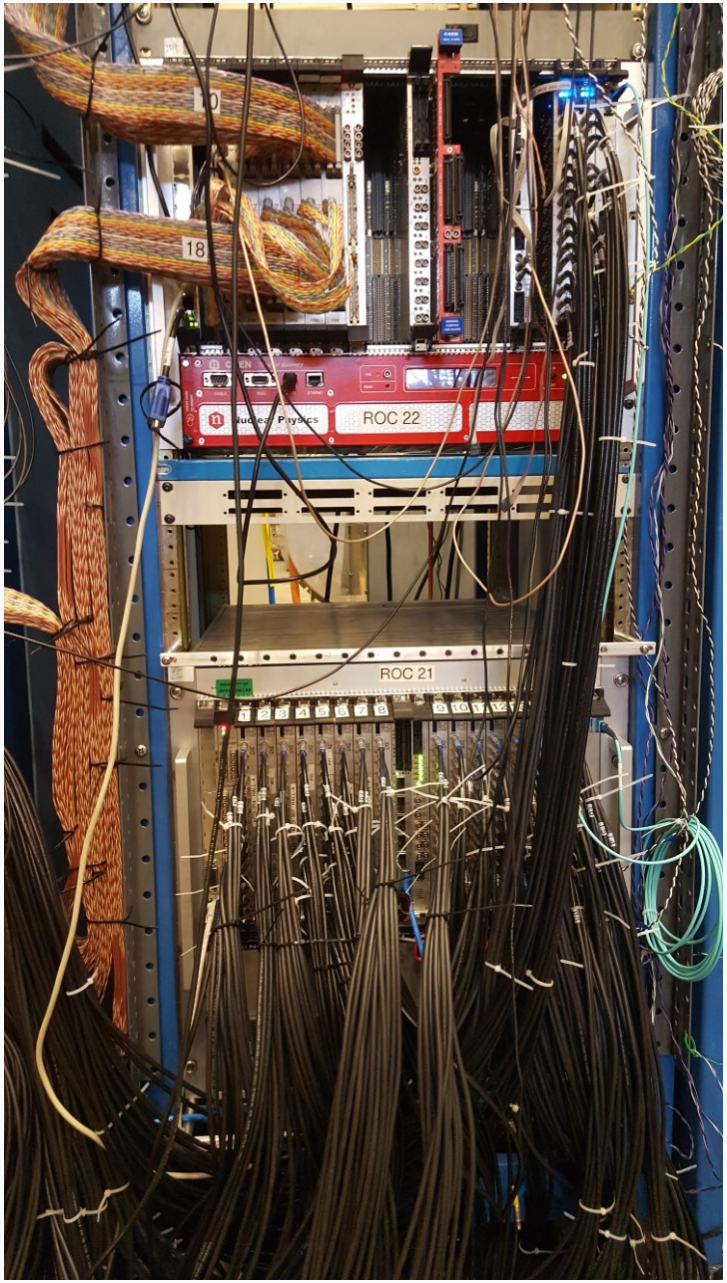


HCal DAQ Side



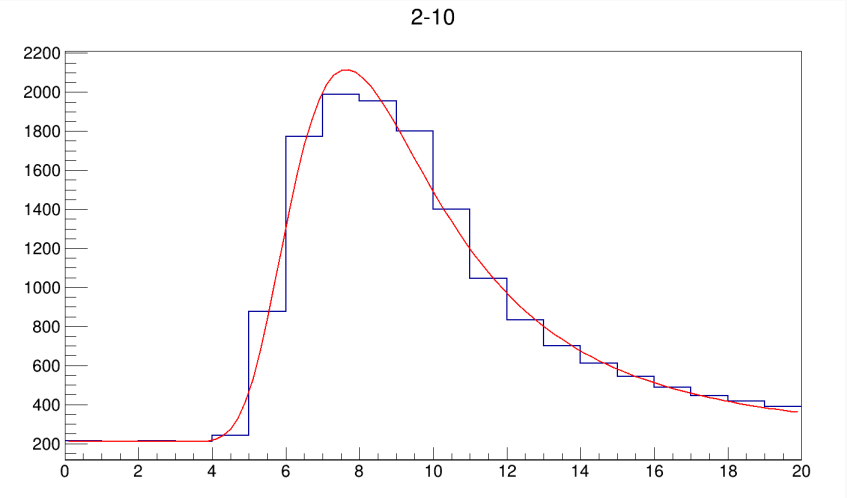
HCal DAQ Side Cont.

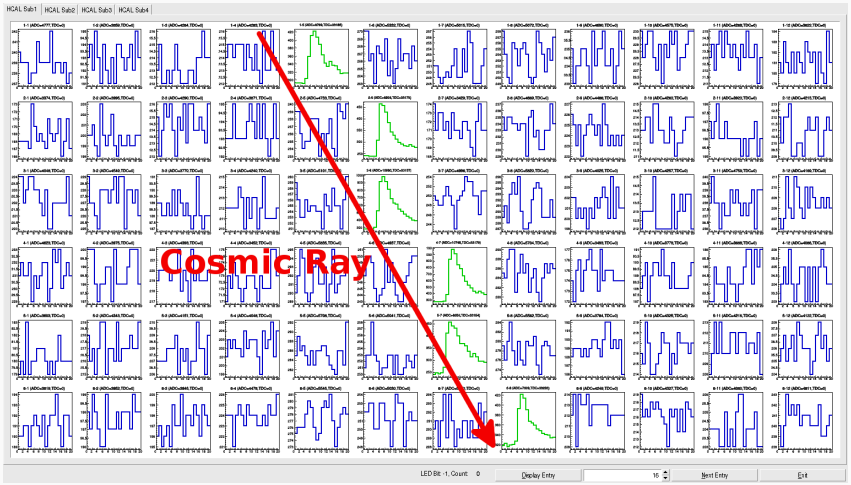




Data Acquisition System

- One VME (temp.) and one VXS crate.
- 18 16-channel fADC250 flash ADCs measure energy.
  - Takes numerous samples (250 MHz, 4ns).
  - Time over threshold measurements extract timing (CFD removes time walk).
  - PMT traces fit by Landau.
- 5 64-channel F1TDCs measure timing.
- Triggers:
  - Scintillator paddle (cosmics).
  - Summing module trigger.
  - LED pulser trigger.
  - BigBite trigger.





TDC Timing Resolution

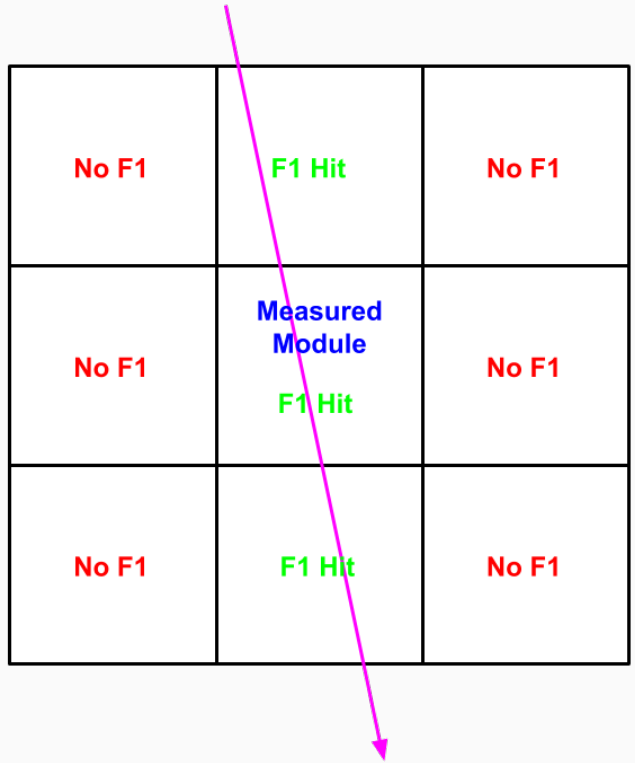
- Require cosmic to be nearly 'vertical'.
  - Vertical F1 signals.
  - No surrounding F1 signals.
- TDC time:

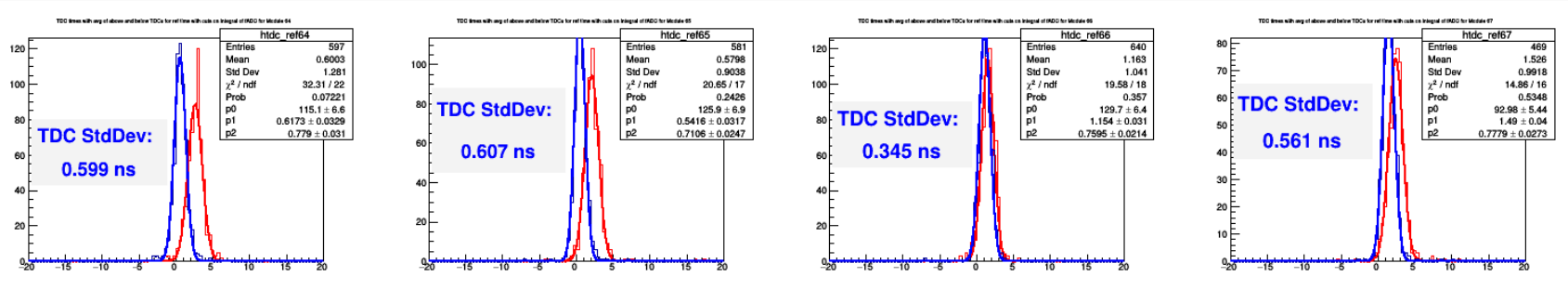
$$T_{cor} = T_{PMT} - T_{ref},$$

$$T_{ref} = \frac{\text{TDC 1} + \text{TDC 2}}{2}.$$

- Extract standard deviation of single PMT.

$$\sigma_{PMT} = \sqrt{|\sigma_{cor}^2 - \sigma_{ref}^2|}.$$



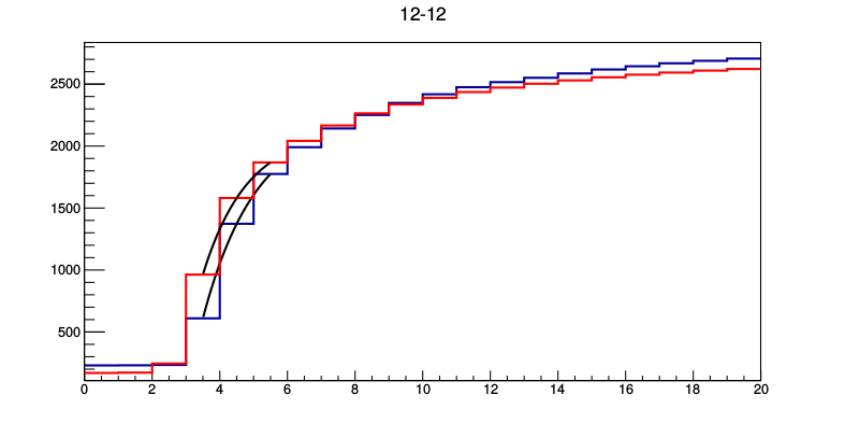


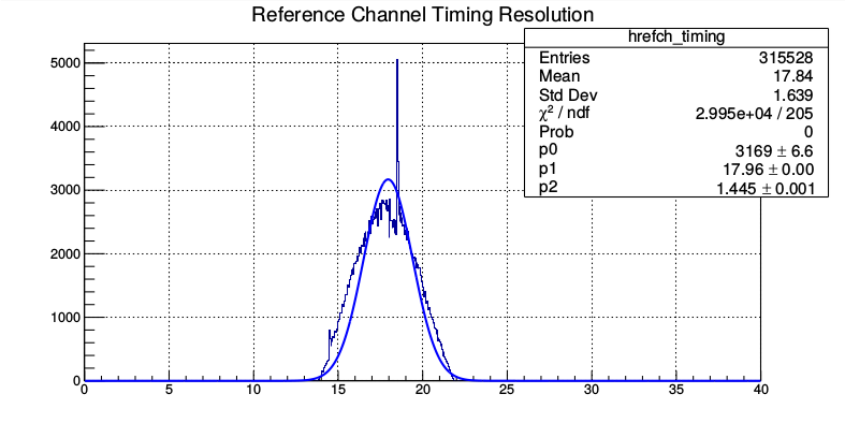
fADC Timing Resolution

- Calculated in same manner as TDC timing except ref. channel is a copy of cosmic paddle trigger.

  - Fit the leading edge of trigger copies with an exponential.
  - Two channels to check relative timing of trigger (good).
  - Standard deviation of ref. channel time not quite Gaussian.

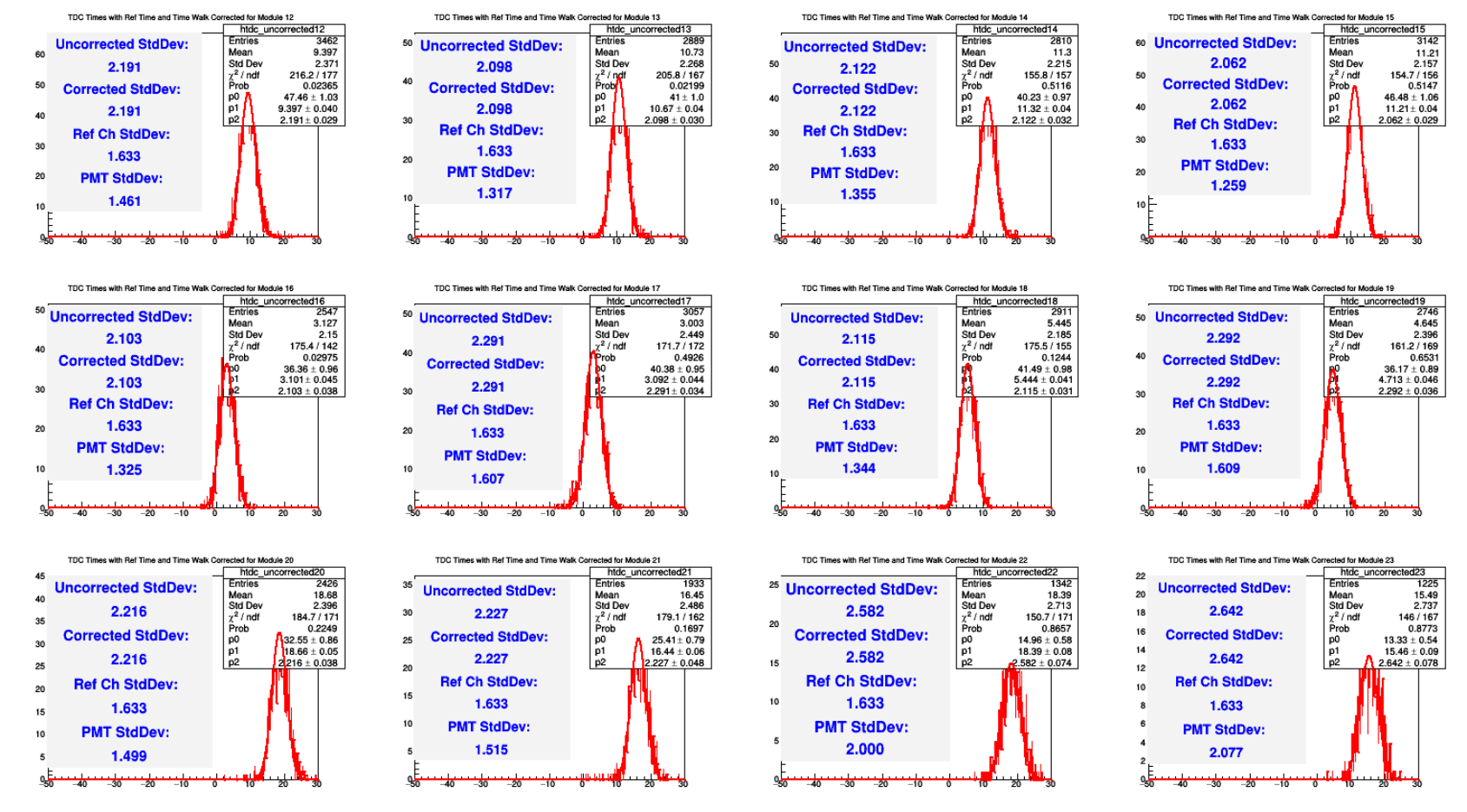
$$\sigma_{PMT} = \sqrt{|\sigma_{cor}^2 - \sigma_{ref}^2|}$$





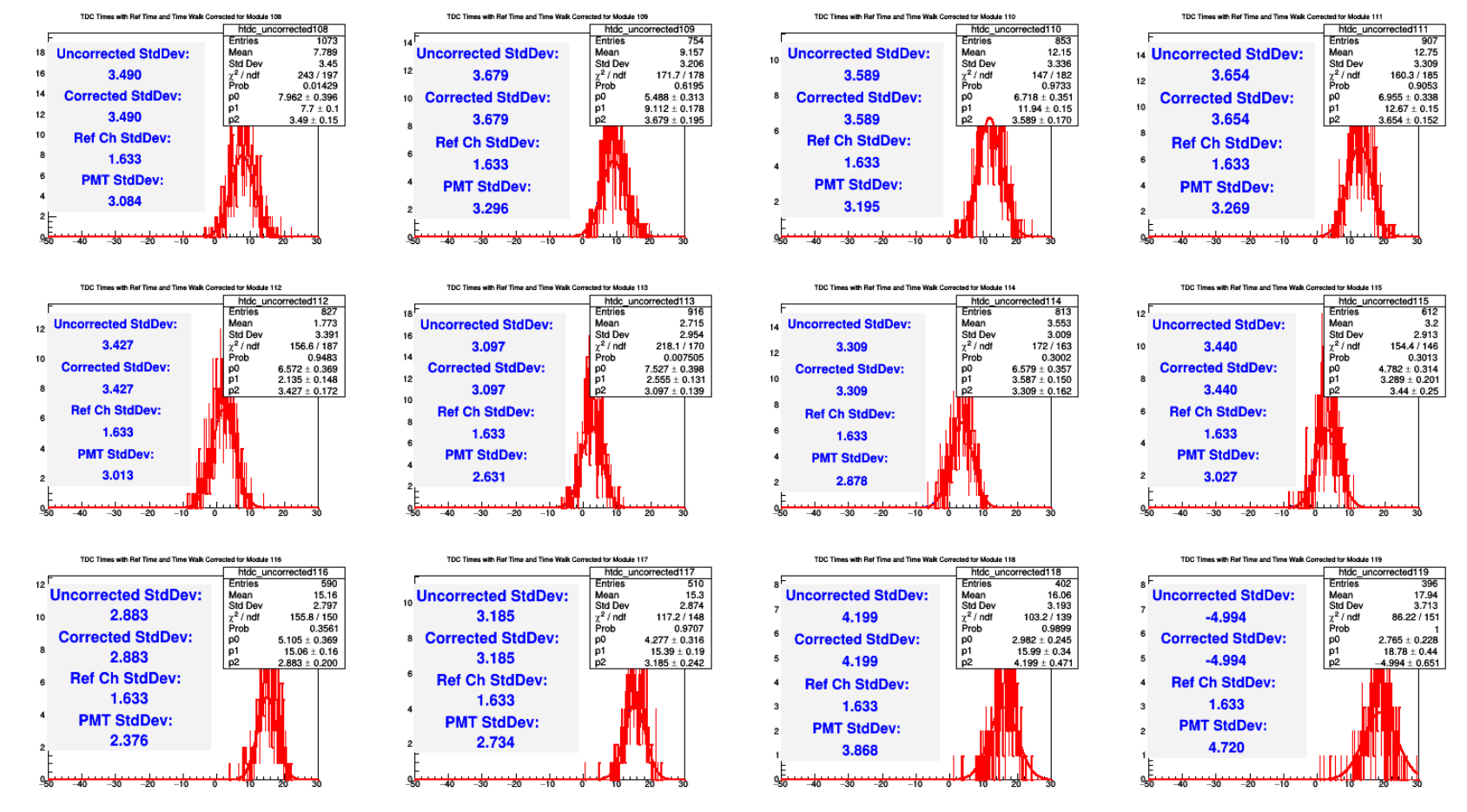
fADC Timing Resolution

- Timing gets worse towards the bottom of the detector.
  - This is due to the ref. time being the cosmic paddle on top of the detector unlike with the TDC.



fADC Timing Resolution

- Timing gets worse towards the bottom of the detector.
  - This is due to the ref. time being the cosmic paddle on top of the detector unlike with the TDC.



Current Status

- DAQ operational and detector fully cabled.
- Cosmics/calibrations will resume when test lab access is restored.
- To do: `https://docs.google.com/document/d/1S--OKOlQLOgP-EP-2nf8LSBLx6Y6d6TAFrWK1UkRxBE/`
  - Grease remaining PMTs.
  - Calibrate relative PMT QEs.
  - Calibrate PMTs with LED pulser/cosmics.
  - Voltage scans.
  - Simulation cosmics vs. real.
  - Upgrade to CODA 3.
  - Online replay.
  - Analysis scripts.
  - Assemble remaining pulser boxes.
  - Fabricate shims.
  - Move to Hall A.
  - Install dry air supply.
- Personnel:
  - 2 postdocs: Scott Barcus and Juan Carlos Cornejo.
  - 2 students: Vanessa Brio and Dimitrii Nikolaev.
  - Brian Quinn and Bogdan Wojtsekhowski.
  - New collaborators: Jim Napolitano, Donald Jones, and Kent Paschke.
  - Prospective collaborators please contact skbarcus@jlab.org.

Acknowledgments

Thanks to Gregg Franklin for his many dedicated years designing and overseeing the construction of HCal. Thanks to Universitá di Catania for their major financial contributions. Many other people and institutions were involved in making HCal possible, including, but not limited to:

- Thanks to the many students who have worked on HCal including Alexis Ortega, So Young Jeon, Jorge Peña, and Carly Wever.
- Thanks to Alexandre Camsonne for helping us get the DAQ working and finding all the modules for us.
- Thanks to Chuck Long for all his help fixing and acquiring things.
- Thanks to Bryan Moffit for DAQ help.
- Thanks also to Brian Quinn and Bogdan Wojtsekhowski.
- Thanks to Vanessa Brio, Cattia Petta, and Vincenzo Bellini for their cosmic commissioning efforts last summer.
- Finally welcome to Dimitrii Nikolaev, Jim Napolitano, Donald Jones, and Kent Paschke.

Machine Learning Detector Trigger for HCal LDRD Proposal

- Motivation:
  - High background rates obscure physics signals.
- Traditional Solutions:
  - Energy threshold cuts.
  - Prescaling the data.
  - Decreasing the beam current.
- Machine Learning Solution:
  - Train a neural network to classify detector events (e.g. p, n, π).
  - Use data from G4SBS converted to detector output to train NN.
  - Load trained NN onto VTP FPGA (fast) to use as HCal trigger.
- Goal:
  - Demonstrate that NNs can be loaded onto VTPs for triggering JLab detectors.
  - Allow HCal to run at higher current with a cleaner trigger.

Brief Neural Network Overview

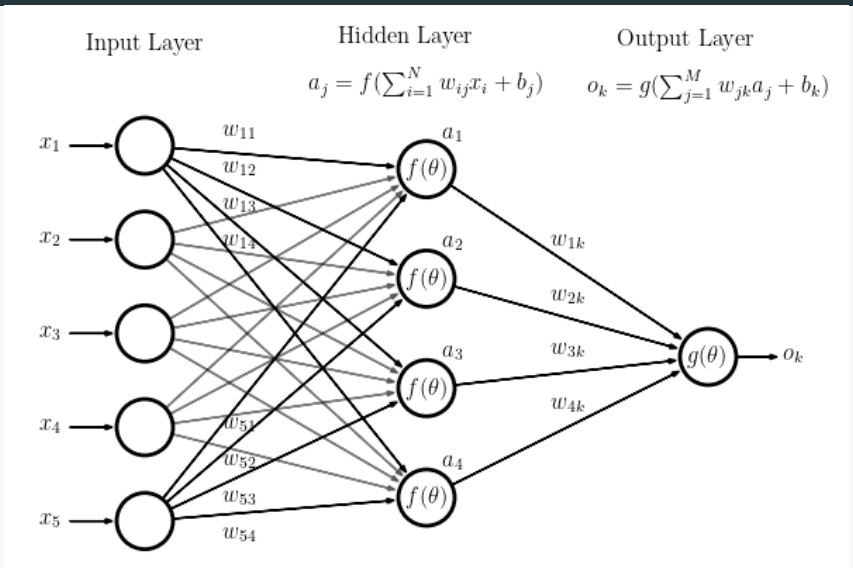


Image from https://www.astroml.org/book_figures/chapter9/fig_neural_network.html.

- Feed Forward:
  - Initialize random weights and biases.
  - Enter labeled training data to input layer.
  - Calculate activation of each neuron (did it fire?).
  - Feed forward activation until output layer reached.
- Backpropagation:
  - Evaluate loss function. How wrong is the NN's guess?
  - Apply backpropagation algorithm to step back through NN (Chain rule).
  - Adjust weights and biases along the gradient of descent (minimize loss function).
  - Repeat.

Convolutional Neural Networks

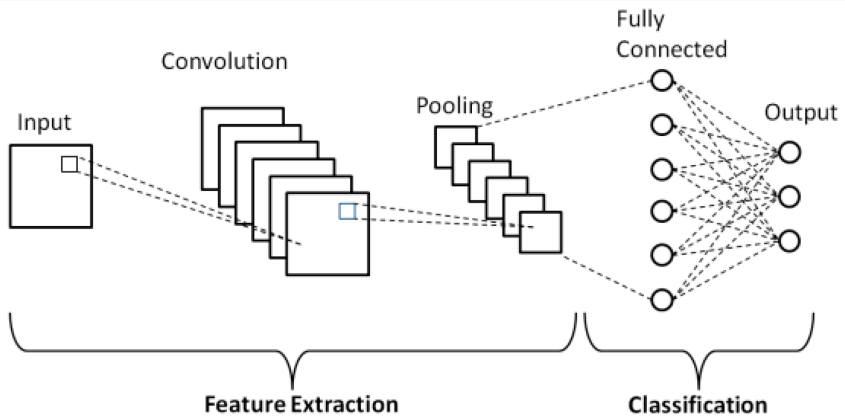


Image from https://www.mdpi.com/2076-3417/9/21/4500.

- Started with image classification.
- Detector traces are essentially images.
  - For every event each PMT has numerous fADC samples (like pixels) and a TDC value.
  - The location of the PMTs relative to one another is important!
- Dense NNs assume each neuron connection is equally important.
- CNNs scan across the image with a kernel creating filters which identify localized features.
- Pooling layers decrease the dimensionality to keep things manageable.

Overview of Research Plan

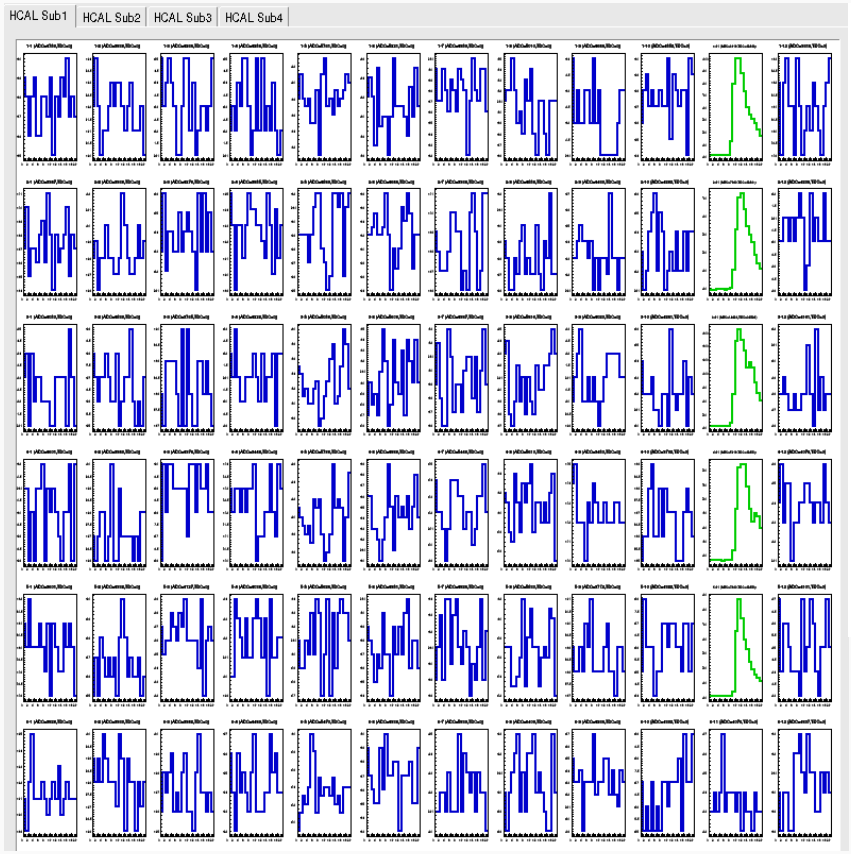
1. Produce labeled training data from G4SBS.
2. Convert G4SBS data into detector signals.
3. Design model architecture (probably CNN).
4. Train CNN on simulated detector data.
5. Optimize model hyperparameters (layers, neurons, learning rate ...).
6. Using hls4ml translate the CNN into FPGA compatible code.
7. Load CNN onto VTP FPGA where it will use fADC250 and VETROC (3 requested from LDRD) data to form HCal trigger.
8. Test the trigger under beam conditions and compare its performance with traditional methods.

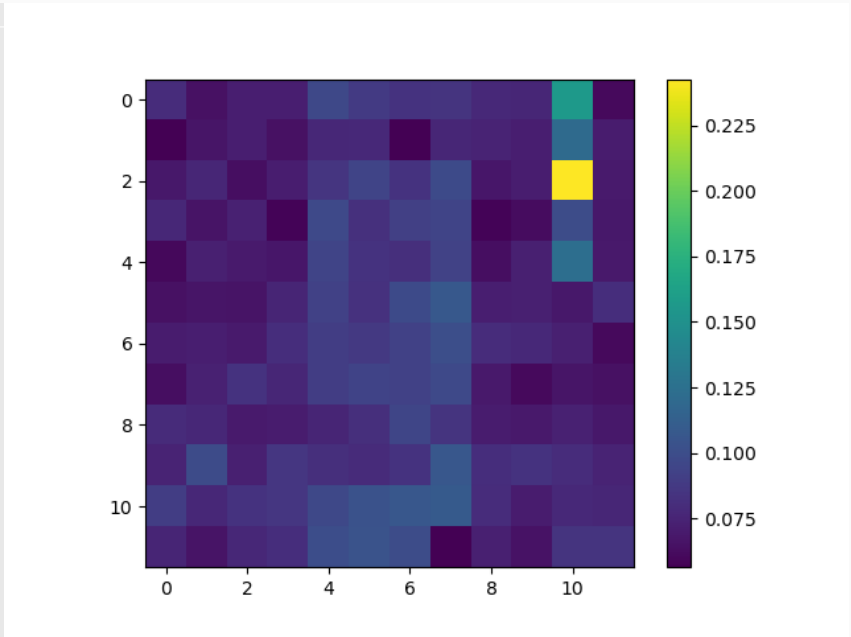
Toy Example HCal Neural Network

- Create NN to identify events with probable cosmic signals. (Note: traditional methods work better than this illustative example.)
- Tools: ROOT, Python, Numpy, Scikit-learn, Tensorflow, Keras, Google Colaboratory (GPUs).
- Steps:
  1. Format HCal data for NN compatibility.
  2. Split data into train, test, and validate.

```
306807
adc_arr shape =  (306807, 144)
tdc_arr shape =  (306807, 144)
hit_arr shape =  (306807,)
[[ 4923.  3871.  4275. ...  3891. 39091. 39055.]
 [ 4753.  3841.  4261. ...  3891. 38345. 38233.]
 [16731.  3843.  5307. ...  3880. 38570. 38497.]
 ...
 [ 4752.  3845.  4275. ...  3877. 39144. 39097.]
 [ 4767.  3855.  4269. ...  3861. 39547. 39514.]
 [ 4758.  3856.  4267. ...  3886. 37892. 37830.]]
[[0.08651978 0.06803129 0.07513143 ... 0.06838278 0.08787302 0.08787302]
 [0.08266092 0.06680004 0.0741044  ... 0.06766961 0.08695658 0.08695658]
 [0.28321155 0.06505182 0.08983346 ... 0.06567813 0.08463677 0.08463677]
 ...
 [0.07222936 0.05844316 0.06497907 ... 0.05892955 0.07599891 0.07599891]
 [0.08326621 0.06733611 0.07456754 ... 0.06744092 0.08733608 0.08733608]
 [0.08391552 0.0680072  0.07525589 ... 0.0685363  0.08818361 0.08818361]]
```

Toy Example HCal Neural Network Continued





Toy Example HCal Neural Network Continued

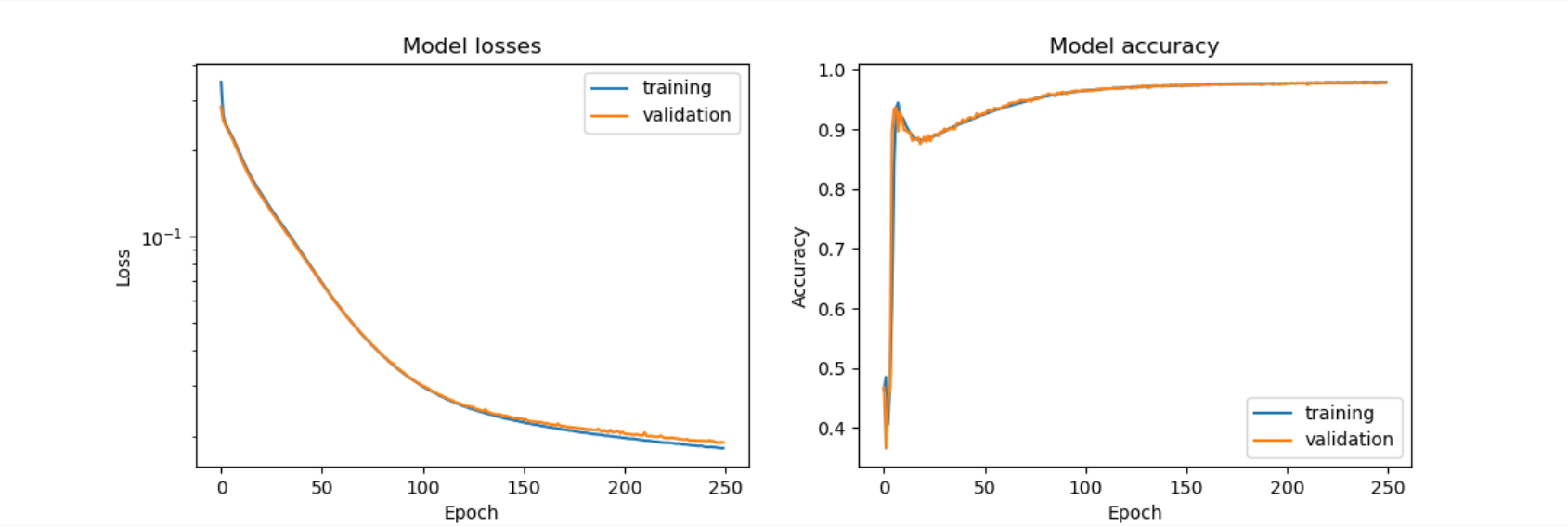
3. Build model architecture.
4. Train neural network.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 128)               18560
_________________________________________________________________
dense_1 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 65
=================================================================
Total params: 26,881
Trainable params: 26,881
Non-trainable params: 0
_________________________________________________________________
Epoch 1/250
391/391 [==============================] - 2s 5ms/step - loss: 0.3477 - accuracy: 0.4634 - val_loss: 0.2844 - val_accuracy: 0.4681
Epoch 2/250
391/391 [==============================] - 2s 6ms/step - loss: 0.2666 - accuracy: 0.4850 - val_loss: 0.2554 - val_accuracy: 0.3657
Epoch 3/250
391/391 [==============================] - 2s 5ms/step - loss: 0.2506 - accuracy: 0.4055 - val_loss: 0.2460 - val_accuracy: 0.4475
Epoch 4/250
391/391 [==============================] - 2s 6ms/step - loss: 0.2420 - accuracy: 0.4712 - val_loss: 0.2379 - val_accuracy: 0.4657
Epoch 5/250
391/391 [==============================] - 2s 6ms/step - loss: 0.2346 - accuracy: 0.6089 - val_loss: 0.2308 - val_accuracy: 0.8968
Epoch 6/250
391/391 [==============================] - 2s 6ms/step - loss: 0.2276 - accuracy: 0.8408 - val_loss: 0.2236 - val_accuracy: 0.9340
Epoch 7/250
391/391 [==============================] - 2s 5ms/step - loss: 0.2203 - accuracy: 0.9359 - val_loss: 0.2164 - val_accuracy: 0.9363
Epoch 8/250
391/391 [==============================] - 2s 5ms/step - loss: 0.2127 - accuracy: 0.9444 - val_loss: 0.2082 - val_accuracy: 0.8967
Epoch 9/250
391/391 [==============================] - 2s 5ms/step - loss: 0.2052 - accuracy: 0.9211 - val_loss: 0.2010 - val_accuracy: 0.9282
Epoch 10/250
```

```
Epoch 245/250
391/391 [==============================] - 2s 4ms/step - loss: 0.0183 - accuracy: 0.9783 - val_loss: 0.0192 - val_accuracy: 0.9762
Epoch 246/250
391/391 [==============================] - 2s 5ms/step - loss: 0.0183 - accuracy: 0.9782 - val_loss: 0.0190 - val_accuracy: 0.9778
Epoch 247/250
391/391 [==============================] - 2s 5ms/step - loss: 0.0182 - accuracy: 0.9788 - val_loss: 0.0190 - val_accuracy: 0.9774
Epoch 248/250
391/391 [==============================] - 2s 5ms/step - loss: 0.0182 - accuracy: 0.9785 - val_loss: 0.0190 - val_accuracy: 0.9778
Epoch 249/250
391/391 [==============================] - 2s 4ms/step - loss: 0.0181 - accuracy: 0.9784 - val_loss: 0.0190 - val_accuracy: 0.9775
Epoch 250/250
391/391 [==============================] - 2s 5ms/step - loss: 0.0182 - accuracy: 0.9788 - val_loss: 0.0190 - val_accuracy: 0.9778
391/391 [==============================] - 1s 2ms/step - loss: 0.0184 - accuracy: 0.9779
test loss, test acc: [0.018352365121245384, 0.9779199957847595]
```

Toy Example HCal Neural Network Continued

5. Evaluate the model's performance:
   - Is loss function decreasing?
   - Is accuracy increasing?
   - Check for overfitting.
6. Optimize hyperparameters.
   - 98%+ accuracy with some optimizing.



- Simple CNN reached 99+% accuracy in only 20 epochs.

Other Machine Learning Applications

- Triggers for other detectors.
- Data analysis:
  - Event classification.
  - Track reconstruction.
  - Cluster finding.
  - Regression.
- Simulation:
  - Autoencoder/Generative Adversarial Network to more quickly produce G4SBS and other simulation data.
- Unsupervised learning for event classification.
- Explore successful pretrained model architectures.
- Much more!

Questions?