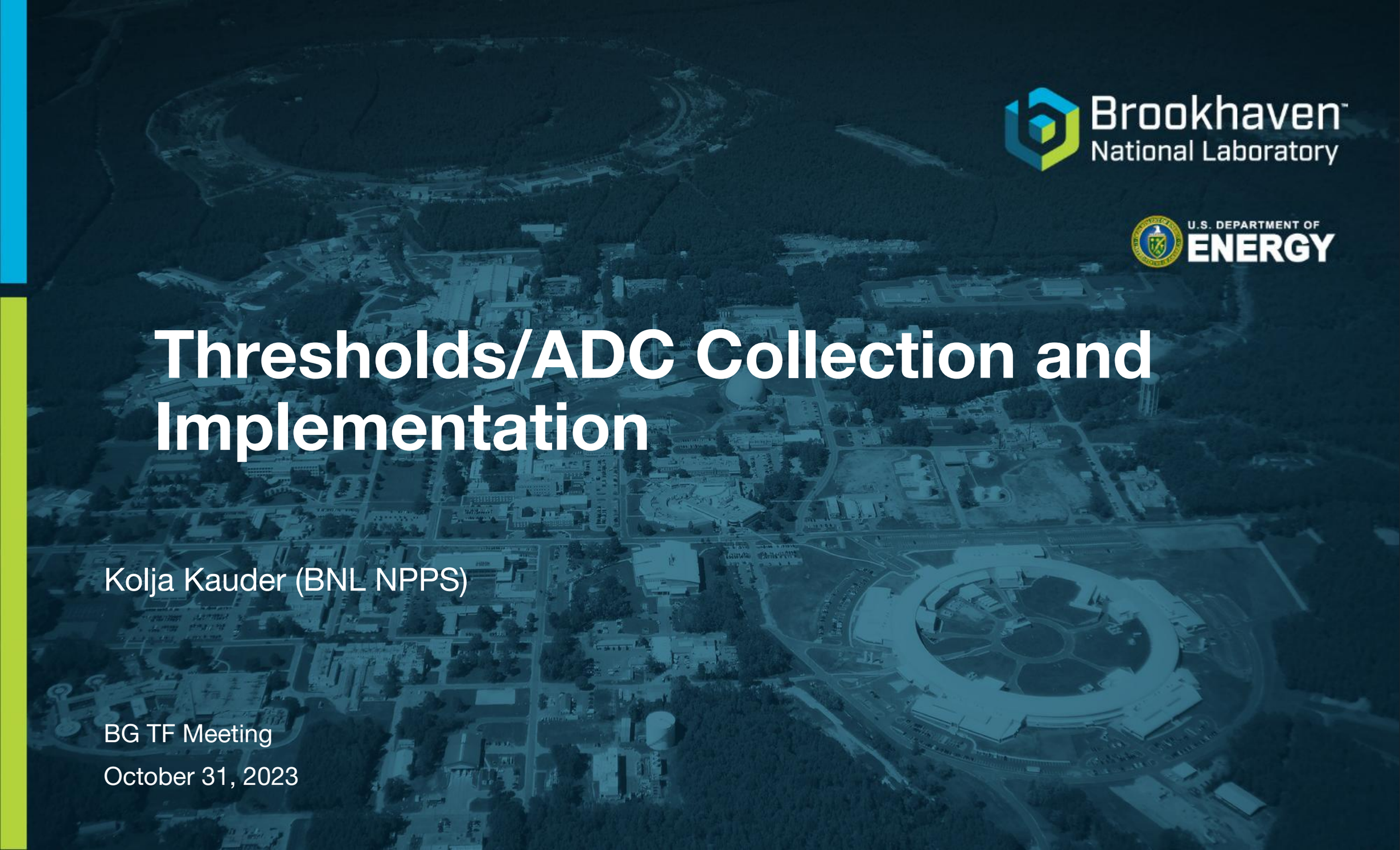


Thresholds/ADC Collection and Implementation

Kolja Kauder (BNL NPPS)

BG TF Meeting

October 31, 2023

Energy cut

if (edep > m_cfg.threshold)

eResRel = [smeared edep] // depends on m_cfg.eRes parameter set

- Cut before sampling fraction correction

- eRes: I believe used to parameterize costly simulation
  - Used in: B0ECAL, FEMC, LUMISPECCAL, EEMC, BEMC (AstroPix)

ADC conversion

```
ped    = m_cfg.pedMeanADC + m_normDist(generator) * m_cfg.pedSigmaADC;
adc    = round(ped + edep * (m_cfg.corrMeanScale + eResRel) /
                  / m_cfg.dyRangeADC * m_cfg.capADC);
```

- Also no sampling fraction applied; that happens in Reco
- m_cfg.corrMeanScale is not totally clear to me. FEMC: corrMeanScale=0.03?
- Double-check that dynamic range and ADC cap (bit-ness) is correct

ADC cut

```
thresholdADC = m_cfg.thresholdFactor * m_cfg.pedSigmaADC + m_cfg.thresholdValue;
for (const auto &rh: rawhits) {
    if (rh.getAmplitude() < m_cfg.pedMeanADC + thresholdADC) continue;
```

- First step in Reco
- EITHER cut on thresholdValue OR on nSigma * Sigma
- Sampling fraction correction only now:

```
energy = ( rh.getAmplitude() - m_cfg.pedMeanADC) / m_cfg.capADC *
m_cfg.dyRangeADC / m_cfg.sampFrac;
```

- Double-check that m_cfg.sampFrac is correct

Supplementary slides

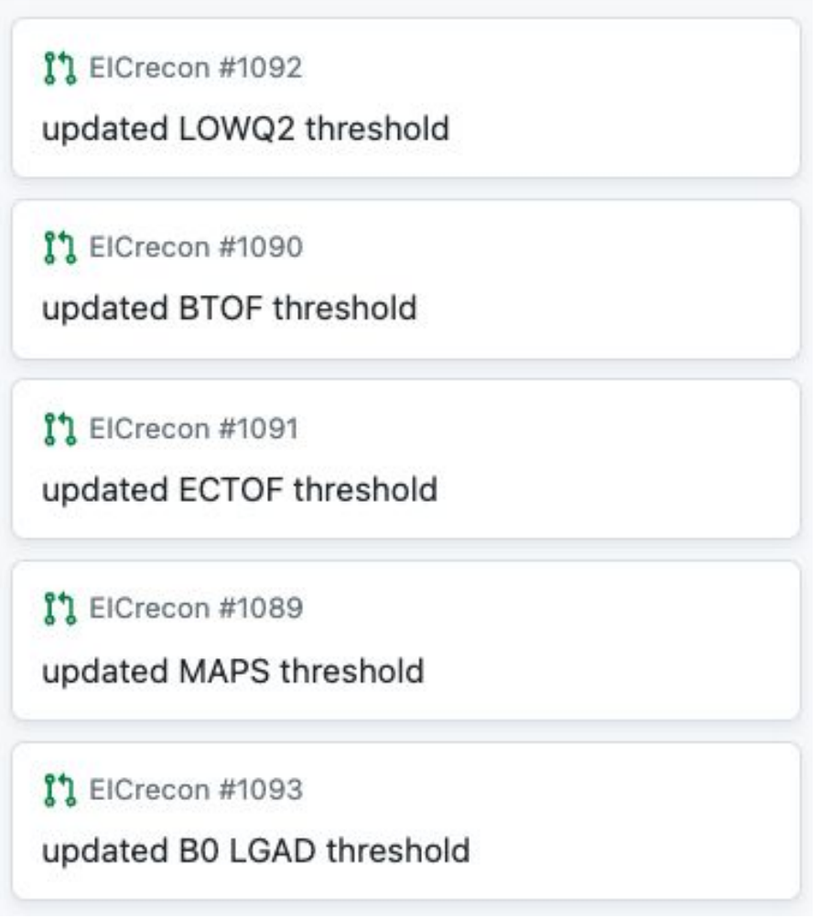
Contacts and Values

- List of github contacts is all but complete
  - (missing: Low-Q2 Tagger (CAL), ZDC)

- Values are pretty much complete - but see later slides
  - (missing: Low-Q2 Tagger (CAL), ZDC (have values, need the technology choice), Roman Pots (no implementation)

Silicon-style detectors

- All PRs are created



Calorimeters

- Created one PR for FEMC, to see if it answers questions from Gerard
- In the process, found that it's actually more complicated than I thought, and we'd actually need more parameters still
- Also, logic is not right, but there's no complete agreement on what would be correct → active discussion, considering inviting developers and DSCs to Thursday Simulation meeting

Calorimeters - Current digitization

Energy Summation:

```
// Make merge map, then
    for (const auto &[id, ixs] : merge_map) {
        for (size_t i = 0; i < ixs.size(); ++i) {
            auto hit = simhits[ixs[i]];
            edep += hit.getEnergy();
            }
        }
```

Now energy cut: m_cfg.threshold;

And energy smearing: m_cfg.eRes;

```
        if ( edep > m_cfg.threshold)
                eResRel = [smeared edep]
```

ADC conversion: m_cfg.corrMeanScale, m_cfg.dyRangeADC, m_cfg.capADC, m_cfg.pedMeanADC;

Also smeared: m_cfg.pedSigmaADC

```
ped     = m_cfg.pedMeanADC + m_normDist(generator) * m_cfg.pedSigmaADC;
adc     = std::llround(ped + edep * (m_cfg.corrMeanScale + eResRel) / m_cfg.dyRangeADC * m_cfg.capADC);
adc     = min(adc, m_cfg.capADC)
```

Calorimeters - Current ADC

Zero suppression:
m_cfg.pedMeanADC,
m_cfg.thresholdFactor,
m_cfg.pedSigmaADC,
m_cfg.thresholdValue;

thresholdADC =
m_cfg.thresholdFactor *
m_cfg.pedSigmaADC +
m_cfg.thresholdValue;

```
    for (const auto &rh: rawhits) {
        if (rh.getAmplitude() <
m_cfg.pedMeanADC + thresholdADC)
continue;
```

ADC to energy:
m_cfg.pedMeanADC,
m_cfg.capADC,
m_cfg.dyRangeADC,
m_cfg.sampFrac;

```
        energy = ( rh.getAmplitude() -
m_cfg.pedMeanADC) /
m_cfg.capADC *
m_cfg.dyRangeADC /
m_cfg.sampFrac;
```

Calorimeters - What I think the logic should be

- convert to ADC
- smear
- apply adc cut

- In the reco class maybe have an optional additional energy cut

Intermediate Solution

- Implement all energy thresholds
- Set all ADC thresholds zero-suppression to 0

- This sets thresholds as indicated at first by the DSCs, but it does not reflect ADC fluctuation bringing a value above or below threshold