Current status of offline conditions data

DUNE Conditions DB Workshop

David Adams
BNL
July 2, 2021

Introduction

This is second talk in this session

- First outlined offline requirements for conditions data
  - Including a pattern for retrieving conditions data
- Here describe some of what is currently implemented
  - Storage in fcl files and DB tables
  - Retrieval through tools, services and modules
- List of categories from first talk is on following slide
  - Categories discussed here are marked in red

Categories

First pass at list of categories (~DB tables)

- Run configuration
- Event metadata
- Geometry
- Event data file metadata (now in SAM)
- Datasets
- Good run (event?) lists
- Electron lifetime
- LAr temperature
- Cathode and anode voltages and currents
  - Unstable voltages
- Detector status (APAs, FEMBs, …)
- Channel status
- TPC CE channel calibration
- TPC dQ/Dx calibration
- Beam status

Run configuration

Run configuration

Class to hold data for retrieval in dunetpc

- dunetpc/dune/DuneInterface/Data/RunData.h
- Retrieval methods on following page
- Schema described at
  - https://wiki.dunescience.org/wiki/ProtoDUNE_run_configuration
  - These include the preamp gain in mV/fC and shaping time in μs

Tool interface to retrieve this data is also in dunetpc

- dunetpc/dune/DuneInterface/Tool/RunDataTool.h
- Method getRunData(run, subrun =0)

Fcl-based tool implementation at

- dunetpc/dune/DuneCommon/Tool/FclRunDataTool_tool.cc
- Run data store in fcl file for each run
  - Tedious to make these by hand (~10k runs in protoDUNE)
  - Example on next-to-following page
- This is the current default tool for protoDUNE
  - Used in calibration (not production) jobs

Run configuration: Retrieval data interface

```
// RunData.h
.
.
.
// Return a value.
  Index run() const { return m_run; }
  Name cryostat() const { return m_cryostat; }
  const IndexVector& apas() const { return m_apas; }
  float gain() const { return m_gain; }
  float shaping() const { return m_shaping; }
  float baseline() const { return m_baseline; }
  float leakage() const { return m_leakage; }
  float hvfrac() const { return m_hvfrac; }
  Index pulserAmplitude() const { return m_pulserAmplitude; }
  Index pulserSource() const { return m_pulserSource; }
  Index pulserPeriod() const { return m_pulserPeriod; }
  Name phaseGroup() const { return m_phaseGroup; }
  const IndexVector& phases() const { return m_phases; }
.
.
.
```

Run configuration: Example storage fcl file

dunetpc/fcl/protodune/fcldirs/rundata/protodune/rundata006600.fcl:

```
run: 6600
cryostat: protodune
apas: [1, 2, 3, 4, 5, 6]
gain: 14.0
shaping: 2.0
pulserAmplitude: 2
pulserSource: 2
pulserPeriod: 497
```

Run configuration (cont.)

IFDH-based tool at

- https://github.com/dladams/duneifdh
- Uses SAM to find the files associated with the run and then uses the metadata associated with one of those files to fill the run configuration information
  - Sets the preamp gain, shaping, pulser source and DAC setting
  - More could be added
- Intended and tested only with Iceberg
  - Data taken with may gains and shaping
  - At present, each gain-shaping combination has its own reco configuration
  - Work in progress to make a single configuration for all that uses the new run configuration tool
- Should we copy the tool and supporting code to dunetpc?

TPC channel status

TPC channel status

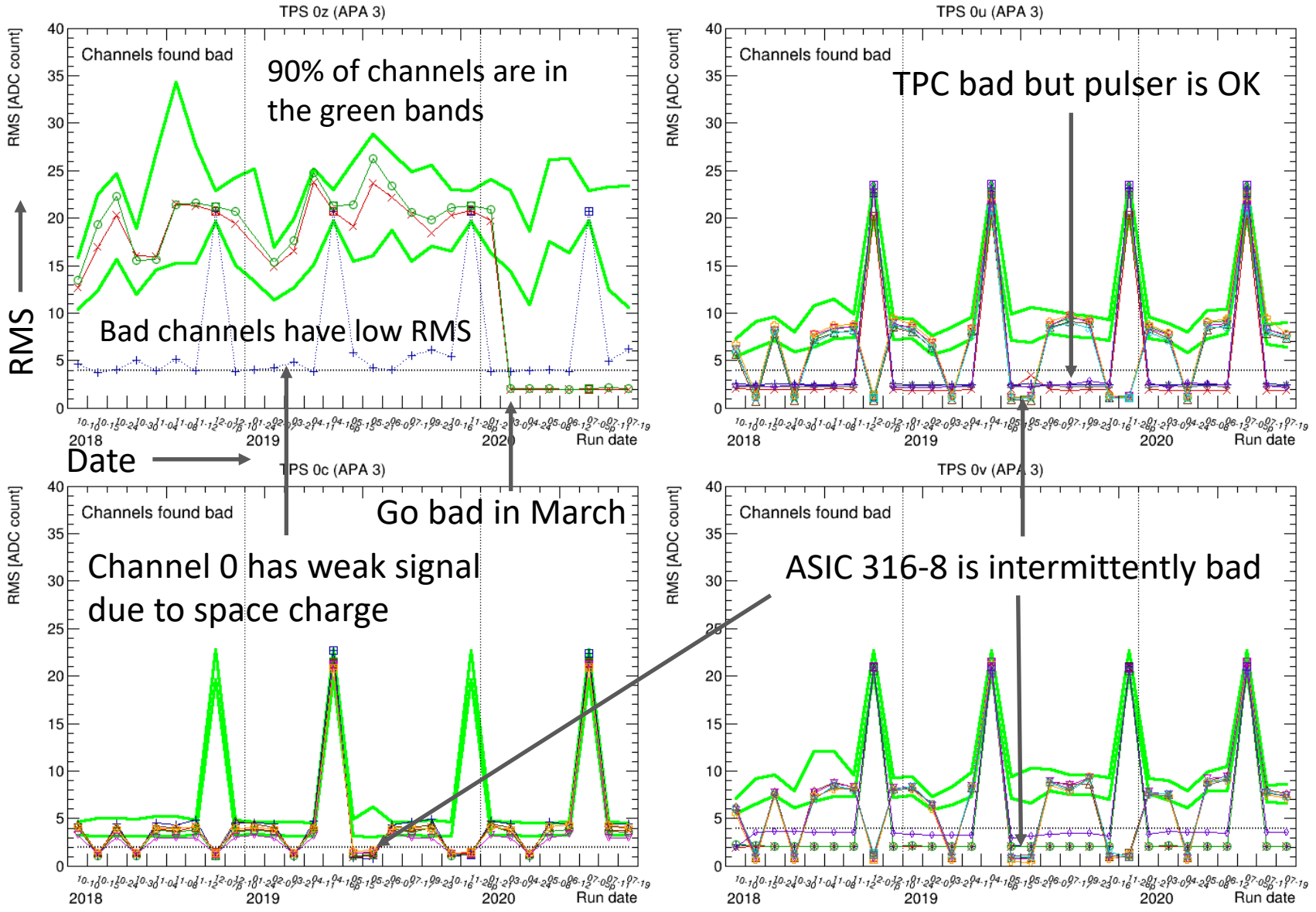
LarSoft service is the default for TPC channel status

- SimpleChannelStatusService
- Methods return if given channel is good, bad or noisy
- Storage is a fcl file listing all the bad and noisy channels
  - For beam data:
    - dunetpc/dune/Protodune/singlephase/fcl/channelstatus_pdsp_2018.fcl
  - Update in late 2019:
    - dunetpc/dune/Protodune/singlephase/fcl/channelstatus_pdsp_nov2019.fcl

But channel status can vary more rapidly

- See https://indico.fnal.gov/event/46063
  - One plot on following page
- Above treat channel that is ever bad as always bad
- Do we want something where different channels go bad (and back good) at different times?

APA 3 bad channels



TPC CE channel calibration

TPC CE channel calibration

TPC calibration based on pulser runs

- Fast signal with known amplitude varying run to run
- See DUNE-doc-15523
- Calibration code in dunetpc and https://github.com/dladams/dunececalib
- Evaluates area gain, height gain and shaping time for each channel

Storage and access

- At present, results are stored in fcl files
  - See dunetpc/ fcl/protodune/fcldirs/calib/protodune/*
- Values are accessed with tool of
  - type FclFileFloatArray
  - implementing the FloatArrayTool interface
    - Including “const FloatArray& values() const”
- Values for protoDUNE were fairly stable but we would probably like to have the option to support IOV time dependence

Unstable voltages

Unstable voltages

There were times when HV was unstable

- Most analyses want to exclude such data

Implemented as filter module for protoDUNE

- I.e., an art module that skips events which occur during times flagged as having unstable HV
- Module configuration specifies those times
  - dataprep/dune/Protodune/singlephase/DataUtils/ProtoDUNEUnstableHVFilter.fcl

Electron lifetime and dQ/dx correction

Electron lifetime and dQ/dx correction

Some protoDUNE conditions info was stored in DB tables

- Electron lifetime
- dQ/dx corrections

In both cases

- Low level interface is the nuevdb package
  - Are we encouraging or discouraging use of nuevdb for protoDUNE 2?
- Utility classes do most of the work though this interface
- Service wrappers provide configuration

Summary/comments

ProtoDUNE conditions data stored and retrieved in many ways

- Storage in DB tables and dunetpc fcl files with varying format
  - Do we want to standardize on something for protoDUNE 2?
  - I.e., the same for many or all categories?
- Retrieval though tools, services and a module
  - Pros:
    - Use what is easy and familiar to the implementer
    - Try out different ideas to see which is best
    - Different approaches my be best for different categories
  - Cons:
    - Retriever needs to learn different patterns for different categories
    - Some conditions data is not easily accessed outside art event loop
      - » Unstable voltage in module fcl
    - Some of the conditions data is not amenable to query