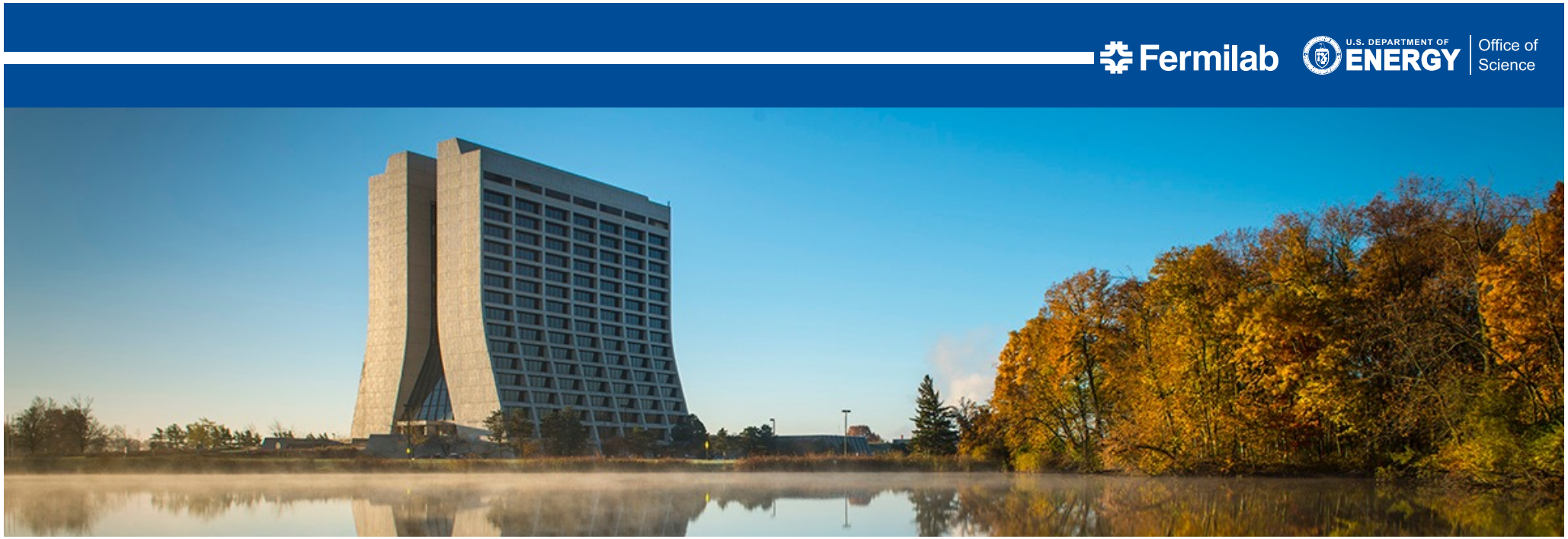


Frameworks

Andrew Norman
DUNE Computing Consortium Startup

In partnership with:

DUNE

Highlights from Frameworks

We started a “framework taskforce” way back in 2019 to assess DUNE’s needs for frameworks for data processing and analysis

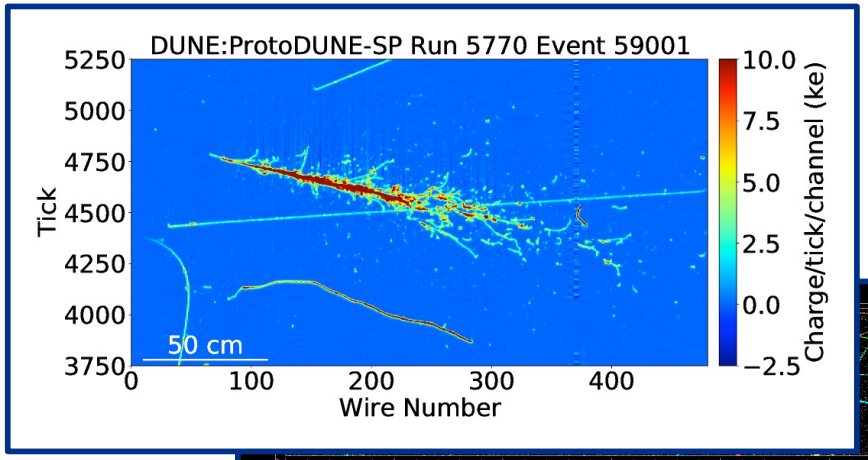
- We tried to drive this process from “physics use cases”
- Involvement from both science groups and from computing [framework] experts
- Result was a detailed report with “requirements” for a framework
- Next step was review and interaction with High Energy Software Foundation (HSF) to see how our needs aligned with the community
- HSF Review wrapped up in Summer ‘21.
  - There were major questions regarding parallelism
  - These were sorted out during a mini-workshop in Nov ‘21

Needs/Requirements in One Slide

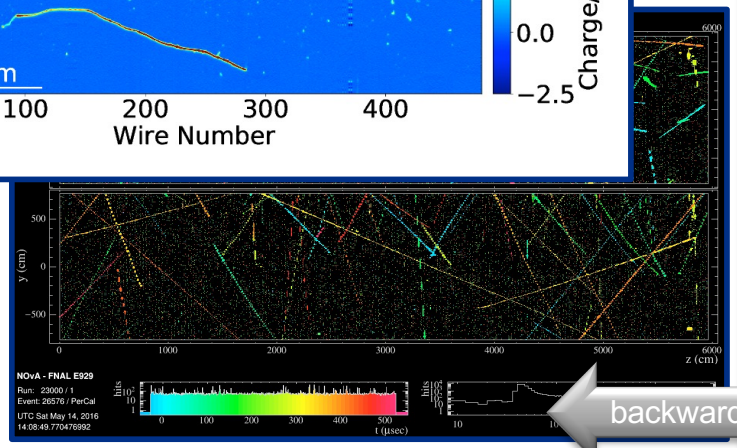
- DUNE needs:
- 2 separate frameworks
  - One for ”bulk” data processing
  - One for “analysis” level data examination and ensemble analysis

HEP Workflows @ DUNE

High Throughput Event Processing and Simulation

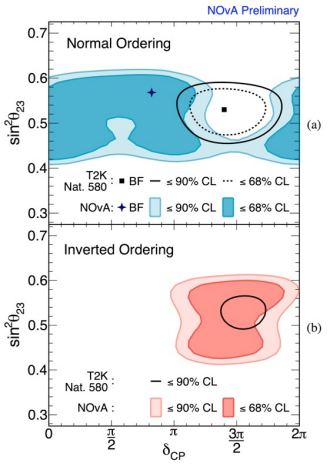


ProtoDUNE Detector Event

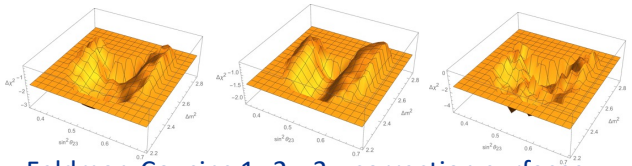


backward Mapping

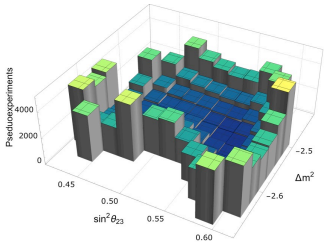
High Performance Computations for Neutrino Science



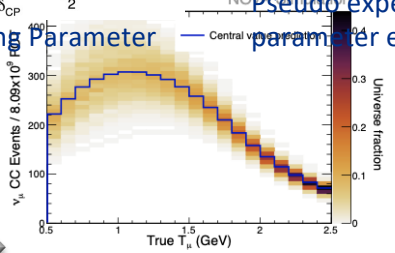
Neutrino Mixing Parameter Estimation



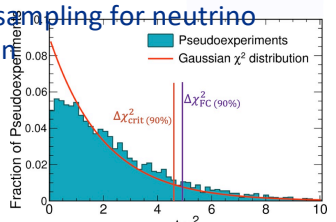
Feldman-Cousins $1\sigma 2\sigma, 3\sigma$ correction surfaces



Pseudo experiment sampling for neutrino parameter estimation



Neutrino cross section unfolding



Empirical χ^2 distribution critical value determination

BULK DATA/EVENT PROCESSING

Analysis/Selection/Computation

Fermilab

Needs/Requirements in One Slide

- DUNE needs:
- 2 separate frameworks
  - One for ”bulk” data processing
  - One for “analysis” level data examination and ensemble analysis
- Bulk data processing framework needs to have:
  - All features of todays frameworks (+LarSoft integration layers)
  - New features related to memory management, accelerator offloading/balancing, event sub-setting, reproducibility, context shifts (trigger record vs. interaction candidate looping)
  - Exposure and other accounting facilities*
  - I/O support for new backends (RNTuple, HDF5, etc…)

Needs/Requirements in ~~One~~ Two Slides

- Analysis/Ensemble framework needs to have:
  - All features of todays analysis/selection frameworks
  - Backwards indexing into full event records (e.g. to see the full events after selection)
  - Provisions for AI/ML external calls
  - Multi-node and data parallelism
  - Systematics facilities
  - Exposure and other accounting facilities*
  - I/O support for new backends (RNTuple, HDF5, etc…)

Where Today's Frameworks Measure Up

- Data Processing:
  - No framework provides everything we need currently.
  - Major holes are memory management, event context switching, concurrency locking w/ accelerators*, whole node scheduling*
  - I/O compatibility
  - Accounting facilities*
- Analysis
  - No framework provides everything we need currently.
  - Major issue is columnar analysis techniques. ← **We want to do this**
  - Multi-node scalability

Work Breakdown Options

Moving forward we have some options:

Bulk Data framework:

1. Modify existing framework w/ retrofits for DUNE needs
2. Greenfield framework w/ DUNE needs

Analysis Framework:

1. Greenfield framework based on current data representations
    1. Develop in tandem w/ bulk framework
    2. Retrofit w/ interfaces later when bulk framework is done
2. Develop translation tools for data representations
3. Develop visualization tools (event displays etc…)

Options

The options carry advantages and risks

Bulk Data framework:

1. Modify existing framework w/ retrofits for DUNE needs
   a) Base can be art+LarSoft or can be CMSSW
      i. Art+LarSoft route has major technical challenges w/ scheduling and memory management.
      ii. CMSSW route has major technical challenges w/ memory management (due to scheduling model) and exposure accounting. Requires rework/retrofit for LarSoft.
      iii. Both need GPU concurrency bolt-on's
2. Greenfield framework
   a) No wholesale code reuse (this may be a con or a pro?) – Need fuller effort profile
   b) Can retain LarSoft (and other Legacy) compatible interfaces
   c) Cleaner long term for support and continued development

Options Aside

Based on domain expert's assessments:

- Modifying existing framework or developing new framework are probably similar in actual required effort
  - Large enough gaps in functionality that bolt-on's will require some core reworks of the frameworks (art or CMSSW)
  - New framework would be designs w/ these in a native fashion (and take advantage of what was learned with art/CMSSW)
    - Art/CMSSW are "old" in terms of technology backends (example: both use TBB threading which is circa 2010 and not compatible with OpenMP5 memory model and is being deprecated for oneAPI™ which is highly Intel centric)
    - Newer GPU Memory models may be awkward to work with and vendor specific (so incompatible with current libs/tech)
    - New portability techniques can be applied in a native fashion (i.e. Kokkos, SyCL, etc…)

Options Aside

Physics code porting will need to happen regardless of path

- Porting existing code/algorithms to conform to a *different* [legacy] **interface** and **data representations** will be a challenge (i.e. what do you do when your algorithm needs a bi-direction mapping between data products but the underlying framework doesn't have that or specifically prevents that)
  - Porting LarSoft to a greenfield framework should be easier (i.e. our LarSoft interfaces become native requirements on the framework)
  - New portability techniques can be applied @porting time (i.e. kokkos, OpenMP etc…) without conflict w/ framework
- Physicist effort is needed here to retain physics integrity (vs. Comp Sci. experts for frameworks)
- **Need modern event display w/ framework integration and dynamic module runtime system**

Options

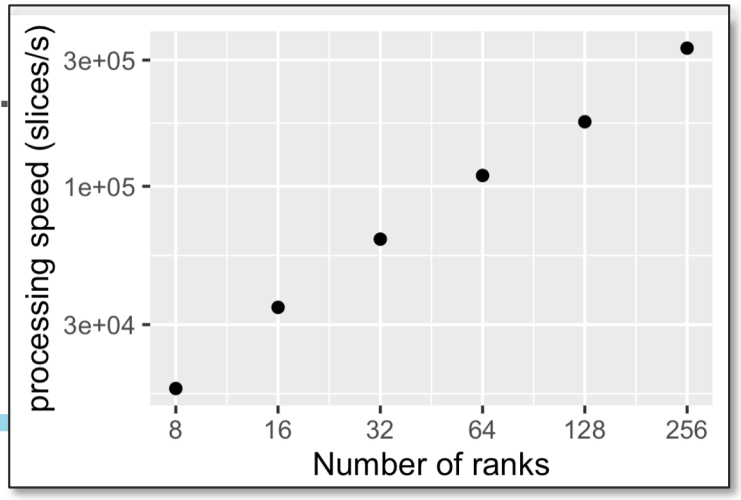
Analysis Data framework:

1. Language support for C++ and Python (or programming language de jour)
2. Current “CAFAna” is heavily tied to TTrees. Doesn’t map well into Columnar tools (but the ideas internal too CAFAna are solid). Probably mandates a rework into columnar backends.
3. Really want an analysis framework that has hooks back into bulk data representations
    a) Means we either need to wait till the bulk framework is in place and then develop analysis or establish interfaces between the two (i.e. analysis framework that can “call” the bulk framework retrieve data, to do a task, or recompute a value)
    b) Systematics…..
    c) AI/ML interfaces (these change rapidly)
4. Thousands of blooming flowers problem
    a) Issue for reproducibility and data integrity

Parallel Analysis Facilities (Pandana-2)

- With the ability to perform parallel compressed writes we are now able to generate “single file” datasets analysis (@multi-terabyte scales)
  - This replaces older models of analysis over hundreds of thousands of individual files
- Working with a SciDAC-4 project (HEP on HPC), we have developed an easy-to-use environment for fast and scalable analysis.
  - easy-to-use: Python, and Python data science tools (e.g. numpy, pandas);
  - fast: natively data-parallel and taking advantage of HPC features;
  - scalable: same code works on laptops, clusters, HPC systems.
  - **Interest in this model from Atlas, NOvA, DUNE…**
- In-memory data processing code scales perfectly.
- We are struggling to make reading scalable.
- Current working in collaboration with Northwestern U. to address parallel IO performance issues.

Neutrino Interactions Analysis Scaling (Pandana-2)



Generalized Effort Numbers

- Coming from other experiments (CMS, Atlas, Alice, NOvA, etc…) we have estimates for the effort that is needed to transition an experiment from one core/bulk framework to a new framework
  - This is the situation we should consider ourselves to be in (i.e. we are using art+LarSoft and CAFAna today)
  - There are two separate parts to the estimate:
    - Domain [Framework] Expertise development (core framework)
    - Scientific Software porting (physics code and algorithms)
- General estimate for DUNE-like code base is: ~6 FTE for O(18 months)
  - Effort is front loaded w/ the core framework devel (12 months)
  - Backend is then existing software porting needs (3-6 months of physicists porting code)