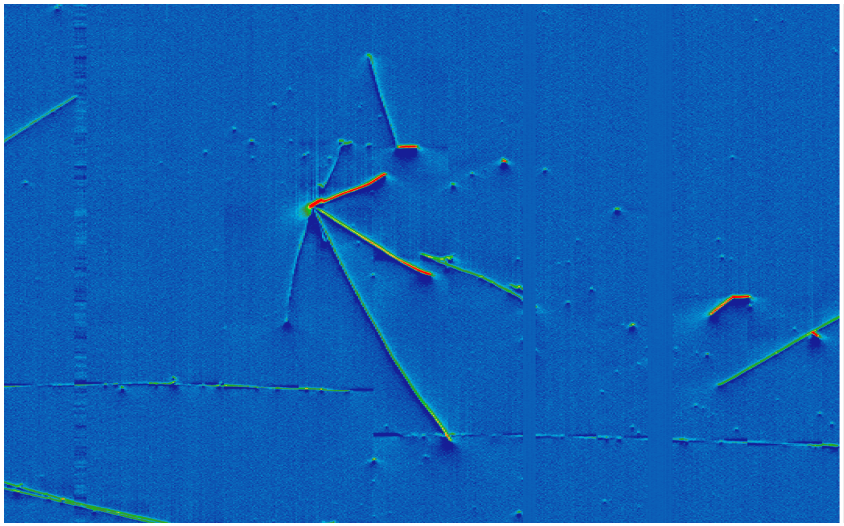
Vectorizing and Parallelizing the Gaus-Hit Finder

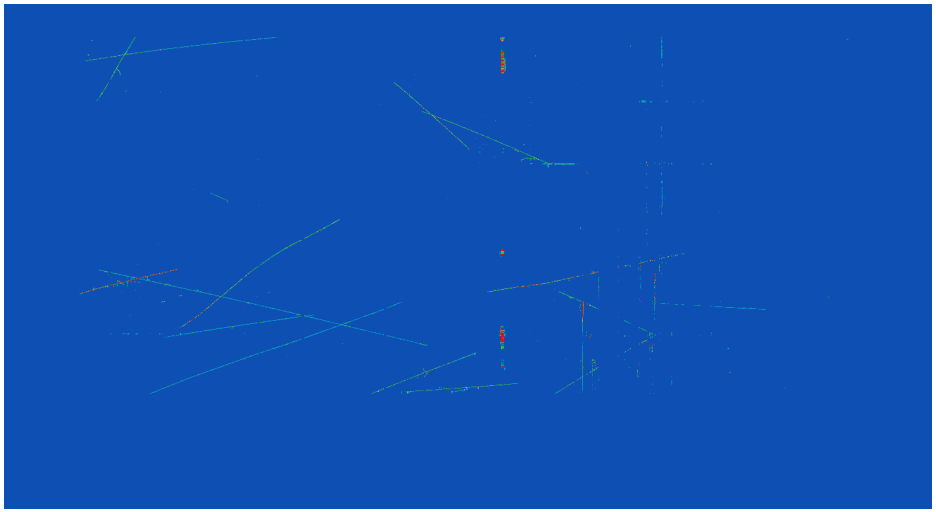


Sophie Berkman
for the SciDAC HEP Reco Group
(G. Cerati, B. Gravelle, A. Hall, B. Norris, M. Wang)

LArSoft Meeting
July 16, 2019

SciDaC Project: HEP Event Reconstruction

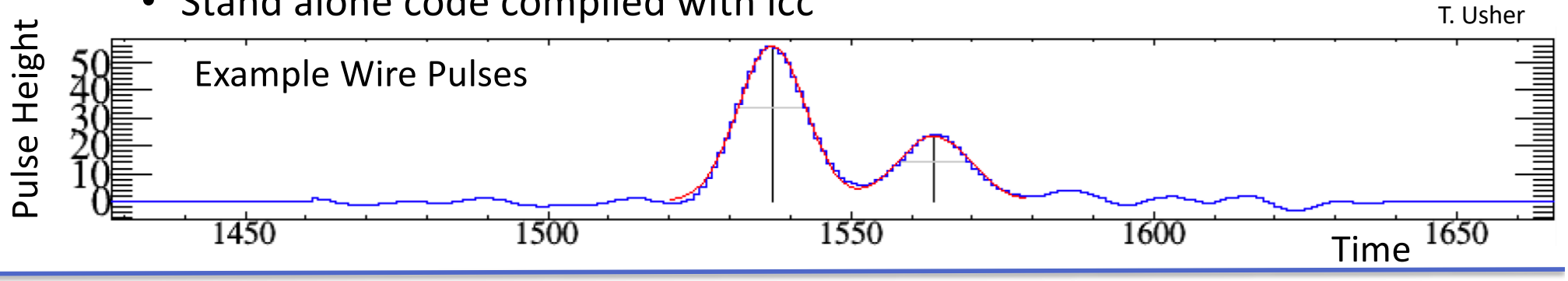
- Study improvements to HEP event reconstruction using vectorization and modern computing architectures
- Liquid Argon:
  - Took O(100 s) to process a μBooNE event (8,256 wires)
    - MCC8 reconstruction
  - Improvements necessary for a larger scale experiment like DUNE (384,000 wires/ 10 kTon cryostat)
  - Focus on vectorizing and parallelizing low level signal processing and event reconstruction
- CMS: vectorize and parallelize tracking code



Fermilab UNIVERSITY OF OREGON

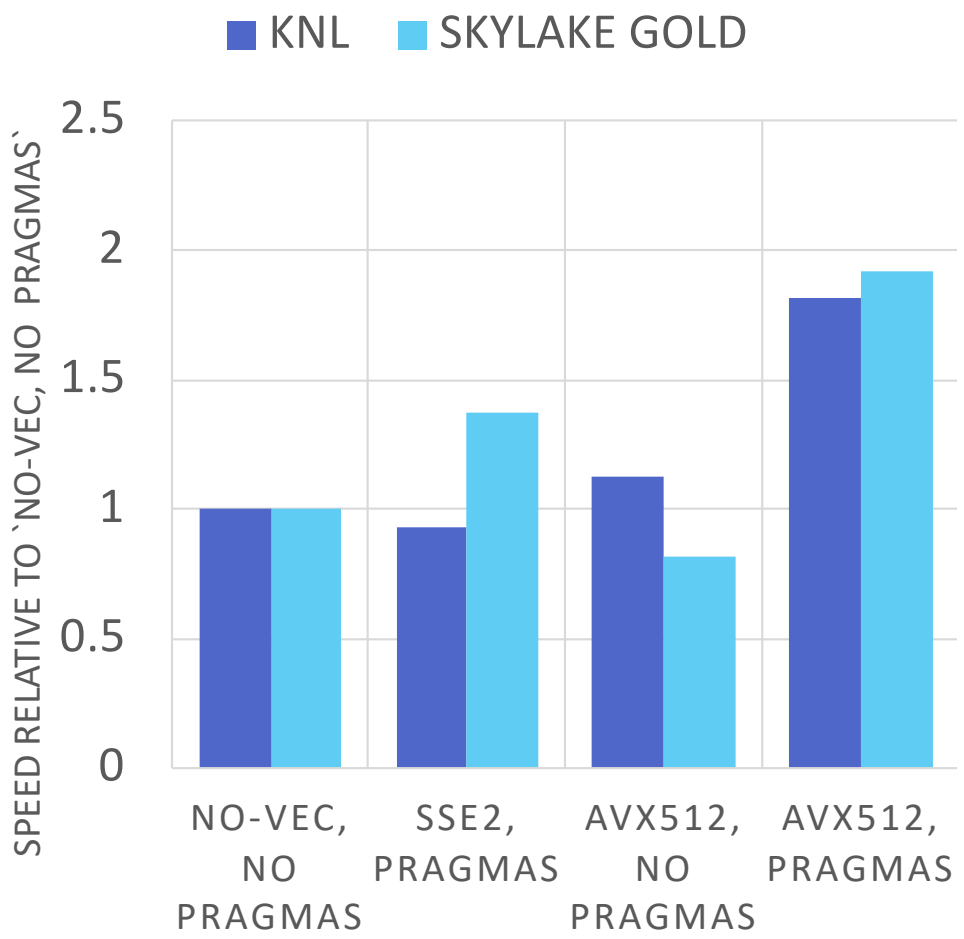
Feasibility study: GausHitFinder

- Feasibility study: GausHitFinder
  - Charged particles produce pulses on wires. Identify and extract parameters associated with pulses (position, amplitude, width).
  - Wires are independent; can be processed independently
  - Few percent to few tens of percent of reconstruction depending on the experiment
- Vectorization and parallelization developments were done within a stand-alone version of the GausHitFinder developed by M. Wang, G. Cerati, B. Norris
  - Implements the Levenberg-Marquardt algorithm to do the fitting
  - ROOT/ Minuit not suitable for parallelization - global memory management
  - Stand-alone code is faster than the ROOT version even before vectorization and parallelization.
  - Will discuss results on stand-alone code, and then LArSoft integration
    - All results are on overlay neutrino events simulated in MicroBooNE
    - Stand alone code compiled with icc



Vectorization of Stand-Alone GausHitFinder

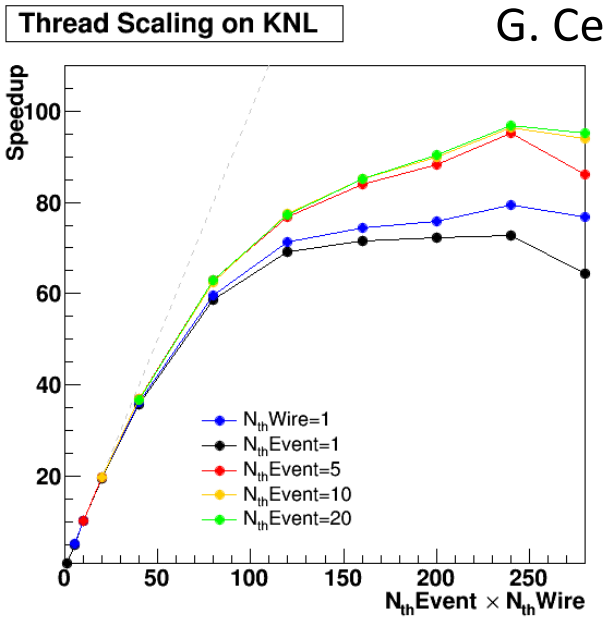
- Vectorization challenges:
  - Minimization difficult because fits converge in different numbers of iterations
  - Cannot fit multiple hits at the same time
  - Vectorize the most time consuming loop, but this is not all of the code
- Vectorization Strategies:
  - Compiler vectorization: use avx512
  - Explicit vectorization on the most time consuming loops
  - Loops determined by profiling the code
  - #pragma omp simd, #pragma ivdep
- **Speed increases**
  - **Explicit vectorization:** ~70% faster on KNL, ~90% faster on Skylake
  - **Compiler and explicit vectorization:** 2 times faster on KNL and Skylake than with no vectorization

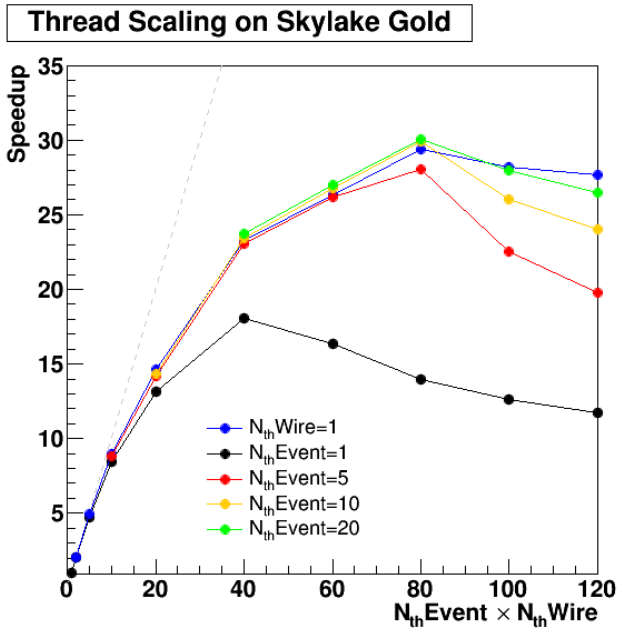


Parallelization of Stand-Alone GausHitFinder

- Using OpenMP
  1. Parallel for loop over events
  2. Parallel region with OMP for + critical (to synchronize output) over regions of interest (ROI) on the wires
  - Fastest with “dynamic” thread scheduling
- Parallelization challenges:
  - Algorithm has a relatively small amount of work.
  - Thread overhead may limit speed up
- **Speed increases with parallelization:**
  - KNL: up to 100 times faster
  - Skylake: up to 30 times faster
- The speed improvements from parallelization are not yet included in LArSoft

G. Cerati



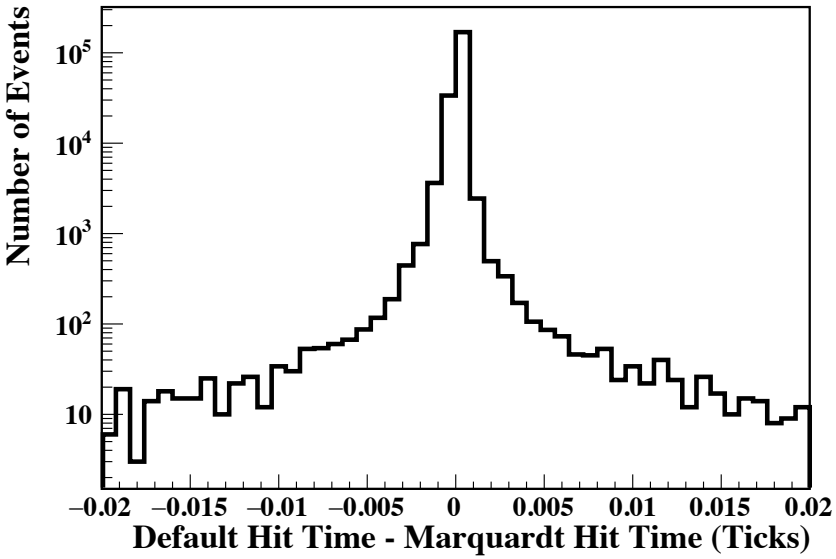


LArSoft Integration

- Integrated a version of the stand-alone code with the Marquardt fitter into LArSoft
  - **Branch of larreco:** *feature/cerati_gshf-larsoft*
  - Marquardt fitting is implemented as a class called MarqFitAlg
  - Does not depend on any external libraries
- New tool "PeakFitterMrqdt_tool.cc" does the fit using the same Marquardt fitter as implemented in the stand alone code.
- Can call this new tool instead of the default "PeakFitterGaussian_tool.cc" in the GausHitFinder_module.cc
  - Does the fitting in "findPeakParameters" function
- None of the current functionality was changed in this branch, just has the option to use the new fitter
- Mike is also using this Levenberg-Marquardt fitter in LArSoft.

LArSoft Validation

- Initial validation done on uboonebuild01.fnal.gov, with overlay neutrino events in MicroBooNE
- **Results:**
  - Hit finder is **12 times** faster on average than the current LArSoft version.
  - Physics results are nearly identical.
    - Difference in number of hits at 0.02% level
    - 2% of hits with a difference in peak time larger than 0.02 ticks
- Does not yet include all of the vectorization and parallelization improvements.
  - No parallelization
  - Uses sse instead of avx512
- Validation ongoing for ICARUS



Pending LArSoft Integration Issues

- Parallelization over ROIs: Implement TBB parallel for within PeakFitterMrqdt_tool.cc
- Vectorization:
  - GCC in stand-alone version:
    - Slower than icc in all cases
    - Almost no increase in speed with explicit vectorization using SSE or AVX512
  - Issues compiling #pragma simd and #pragma ivdep simultaneously over a loop using CMake
- Possible solution: compile Marquardt fitter with icc AVX-512 and link it to LArSoft as a library?
  - Encourage experiments and grid to allow selection of nodes with specific vector extensions

Conclusions & Future Work

- GausHitFinder has been vectorized and parallelized:
  - Up to 100 times faster with parallelization
  - Up to 2 times faster with vectorization
- Levenberg-Marquardt algorithm has been implemented to do the fitting in the GausHitFinder algorithm instead of ROOT
  - Fitter implementation performs well when compared to MKL
- New version of the GausHitFinder integrated into LArSoft:
  - 12 times faster than the current implementation on MicroBooNE overlay events, work ongoing for ICARUS.
  - Physics results nearly identical to current LArSoft version.
  - Not yet taking advantage of all of the potential vectorization and parallelization improvements, which are further independent speed-ups.
- Future directions:
  - GPUs: work has started on the CMS side of the SciDAC project and plan to test similar techniques with liquid argon code.
  - Plan to start working with other signal processing algorithms next.