uBooNE SimWireMicroBooNE and CalWireMicroBooNE: code/physics/EE walk-through

Eric Church, 28-Sep-2011

Introduction

- This was to be a well-organized pedagogic talk, full of mathematical details and code-snippets.
- It will be instead a broad-brush set of assertions. We had hoped for a tech note. That might still be coming. In lieu of that, I will make this presentation.
- I hope my anguish in reading undocumented (but rich and important) code and this talk will serve to decipher SimWireXYZ.cxx and CalWireXYZ.cxx.

Schematic

- There are 2 convolutions needed to produce a simulated signal on a wire.
- The first convolution mimics what happens when a cloud of electrons drifts toward a wire plane . So, this is a pulse of microsecondish width convoluted with a sawtooth (Collection plane) or square wave (Induction plane) which is of a width of few-ish usecs. Namely, on order of the "approach" distance, ~cm -- when the induction/collection begins -- divided by the drift velocity. This *is* a convolution in time space.
- The second convolution is the resulting signal from the above, convolved with the electronics response. Again, by fiat of the model. This is how one characterizes the way any circuit operates on an input. In uBooNE this is a ~microsecond rise-time pulse.

Schematic 2

- We can do the second convolution first. They commute! Doesn't matter the order.
- Hence we do the convolution of the electronics response with the field response in the beginJob() exactly one time per job.
- We do the convolution with the electron pulse evt-by-evt, with each track of drifting ionization e's as G4 gives 'em to us.

Schematic 3

- Now.... I refer you to wikipedia to recall the following facts. The punch line of which is that a convolution in time space is equivalent to a product in Fourier Transform space.

- In super natural units .... (2pi = i = -1 = sqrt(2) = 1), and "*" is convultion and "." is ordinary multiplication

  h(t) = f(t)*g (t) = Int dt' f(t).g(t-t')

  h(w)=f(w).g(w)

  h(t) = Int dw exp(-iwt) h(w),

Schematic 4

- The field and electronics convolution in beginJob() is done in time space. The resulting "kernel" -- the FFT of this is saved out to a root TFile.
- The 2nd convolution is done in fourier space (evt-by-evt, as stated).
- I don't know historical reasons why.

Schematic 5: CalWire

- The Fourier transform “kernel” is read back in CalData/CalWireMicroBooNE.cxx’s beginJob().
- We read up the simulated (or data) wire signals and divide in Fourier space (de-convolve!) by this kernel.
- **This recovers the electron pulse**, which we feed to FFTHitFinder (sic).

And that’s it ...

- except rather than BNL handing us the time pulse electronics response, they’ve given us the Laplace transform.
- I also don’t know this history either.
- Fortunately, wikipedia reminds us how to invert this and recover the timespace response.

Laplace inversion

- 1/(s+p0) * 1/((s+k1)^2+p1^2) ...
- inverts to something (very roughly) like
- exp(-p0t)*u(t) - cos(k1t)exp(-p1t)u(t) ... a pulse of ~1musec width, I presume.

uBooNE electronics simulation To Do

- Georgia will invert Laplace transform
- She and I will go through steps outlined here with 198->500nsec sampling time, getting normalizations and FFT nuance all right (need 2x the time buffer, e.g.). And the full drift time of 1.6 msec's instead of ArgoNeuT's 0.8msec.
- Watch this space.