Pulser-based Calibration using PDVD Data

Karla Téllez Girón Flores

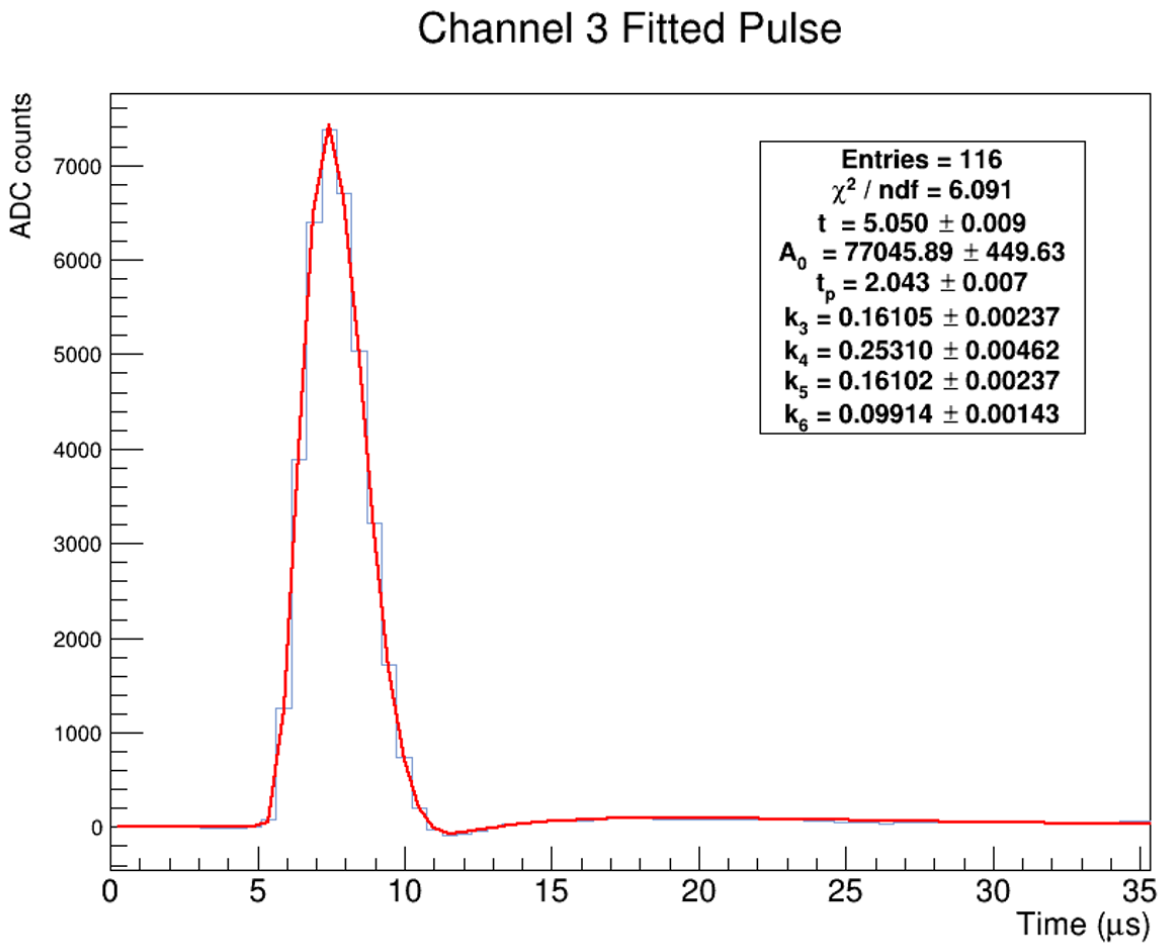
Local BNL ProtoDUNE Meeting

06/02/2026

A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

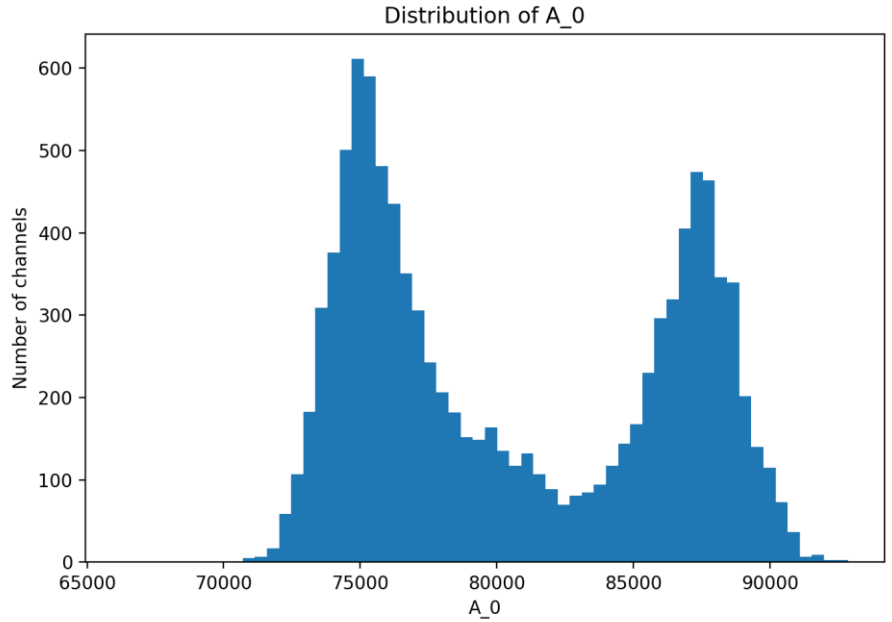
1. **Analysis starts by fitting per-channel pulses.**

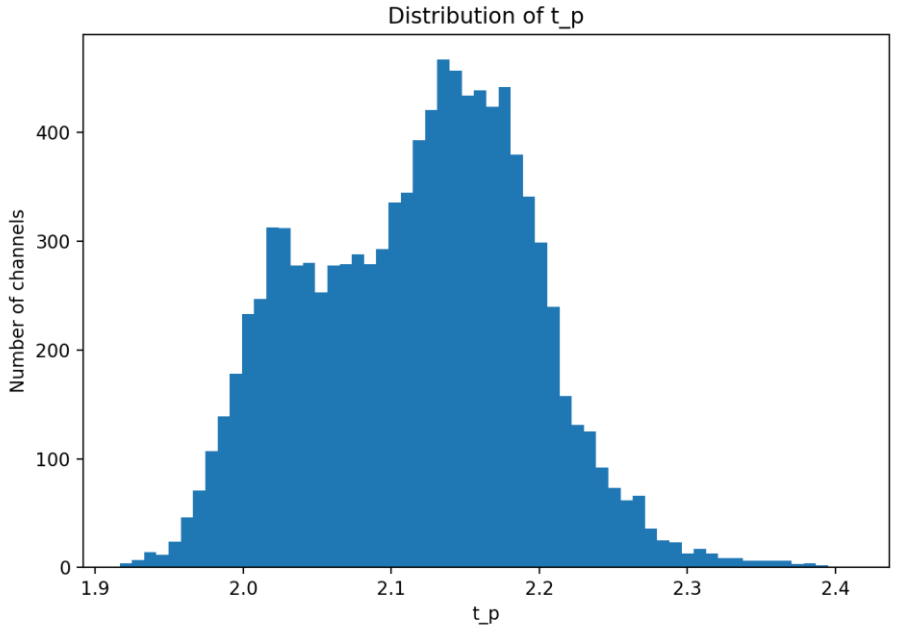


A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**

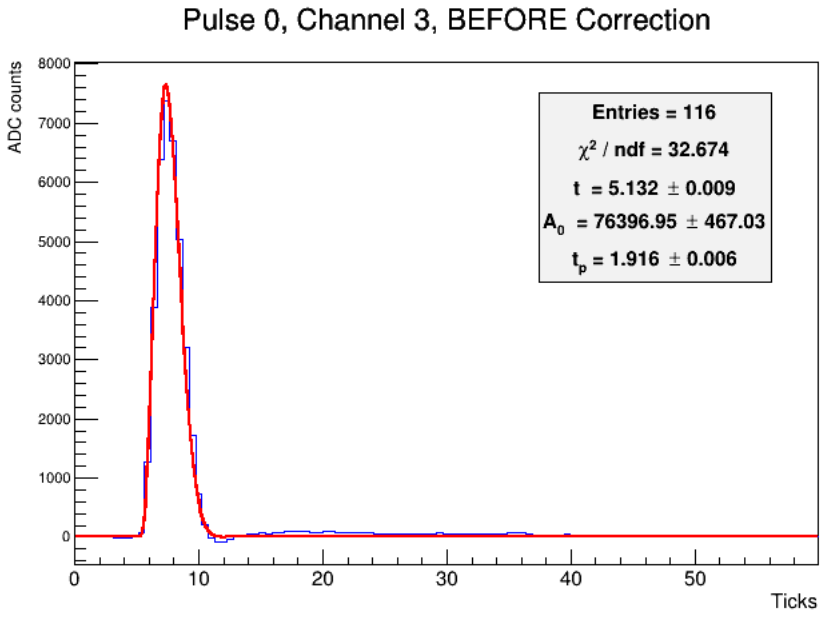


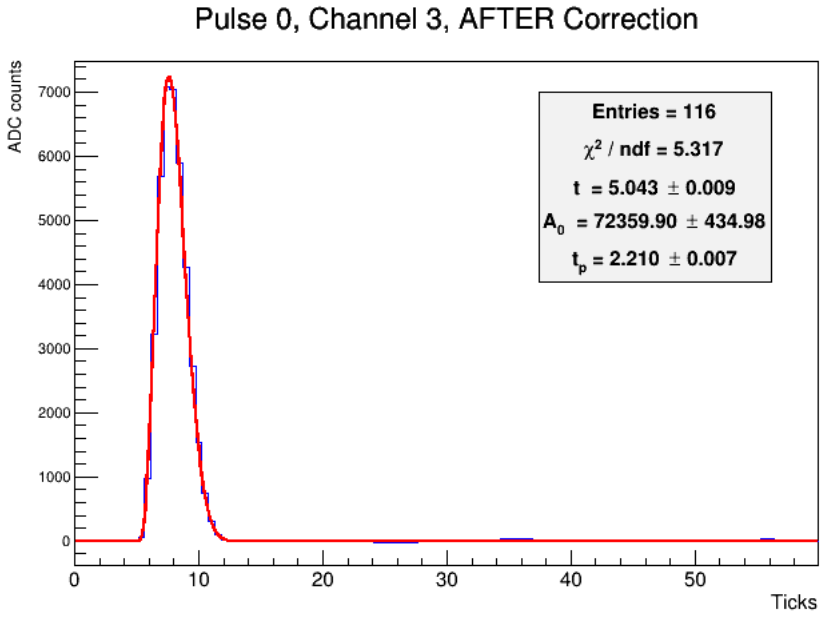


A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**
4. **We can now retrieve *corrected* waveforms.**



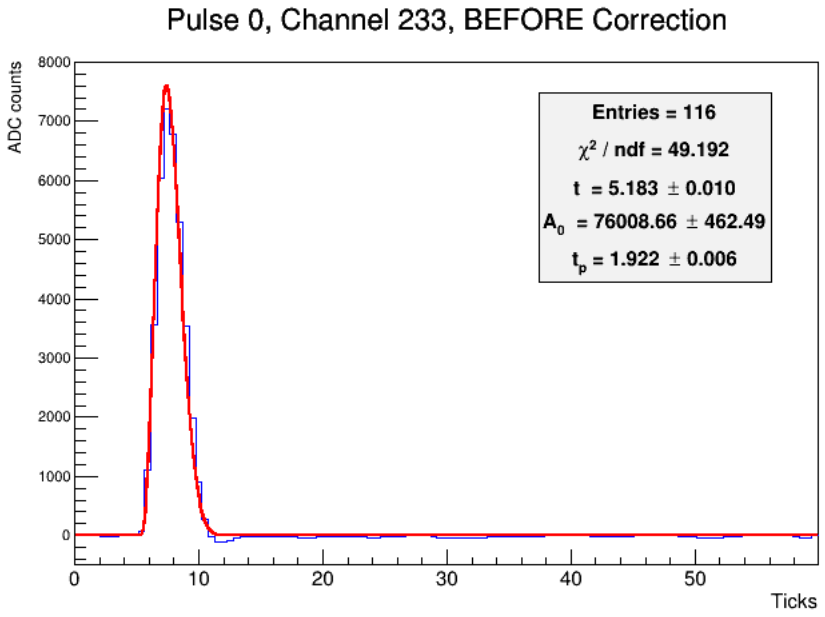


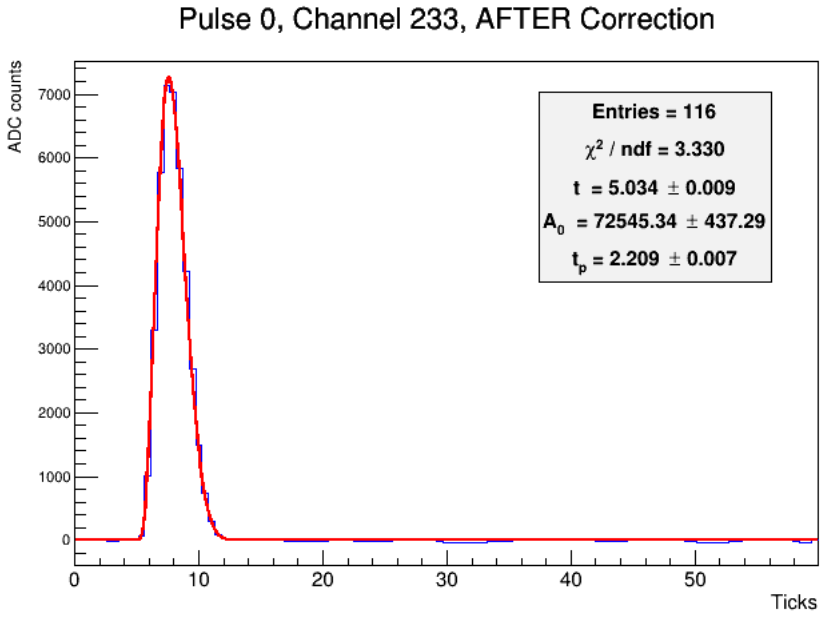
Example 1

A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**
4. **We can now retrieve *corrected* waveforms.**



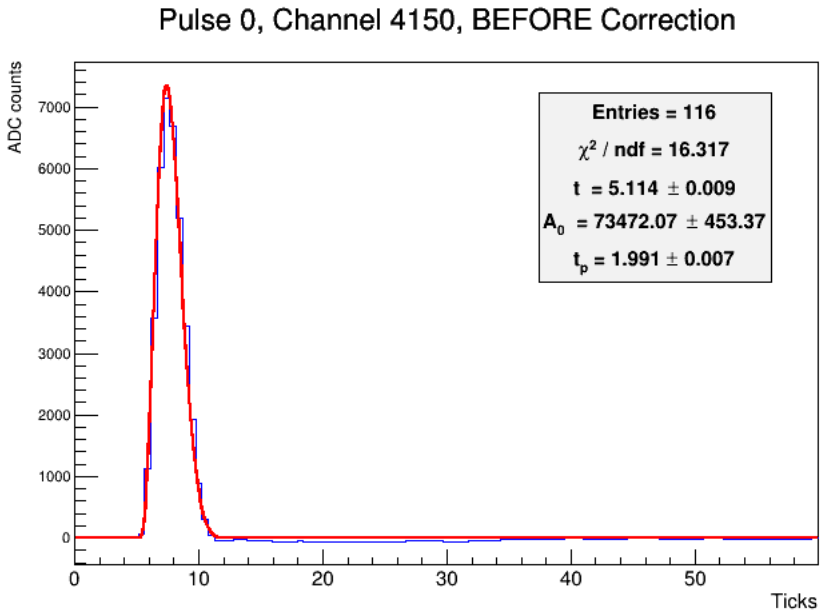


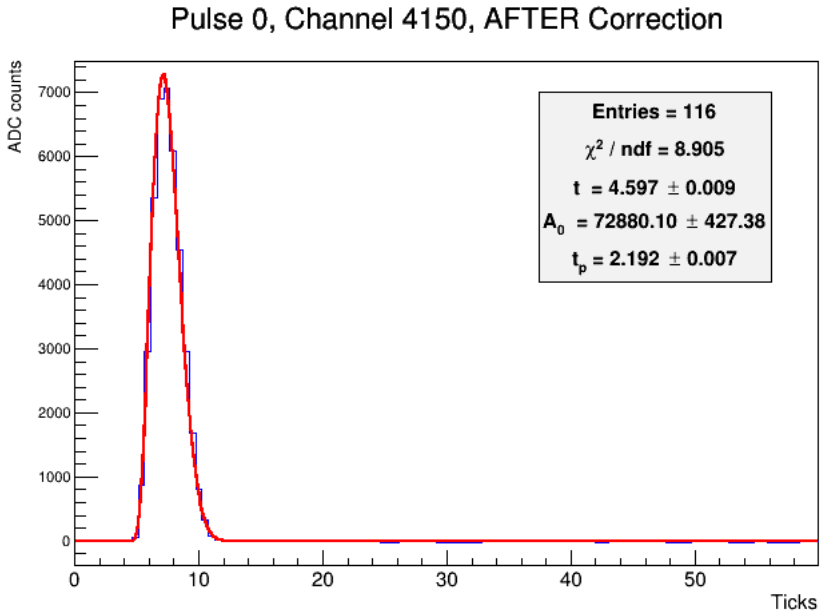
Example 2

A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**
4. **We can now retrieve *corrected* waveforms.**



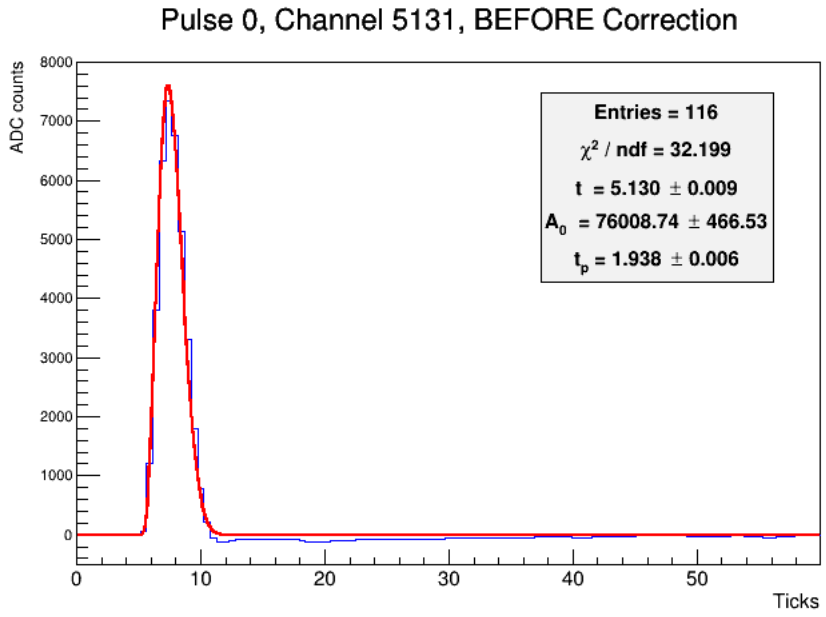


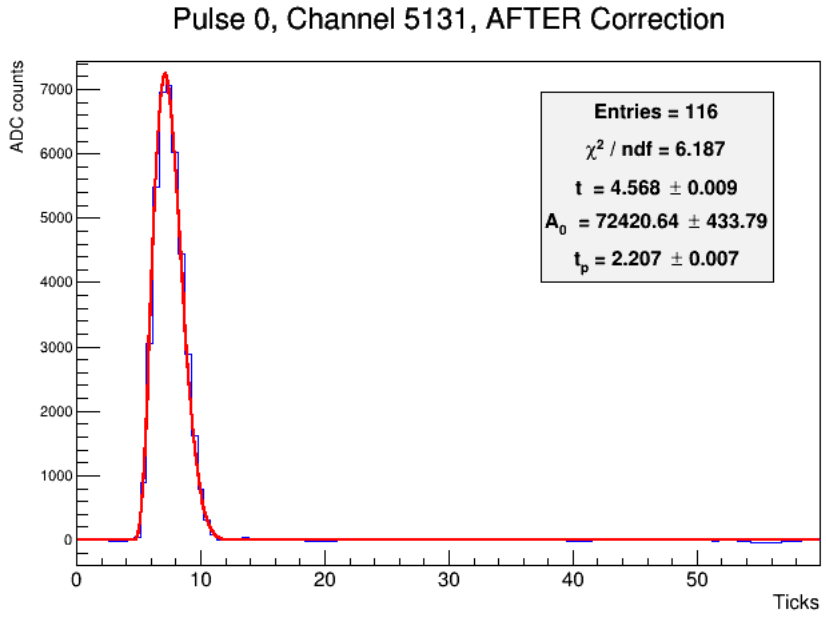
Example 3

A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**
4. **We can now retrieve *corrected* waveforms.**



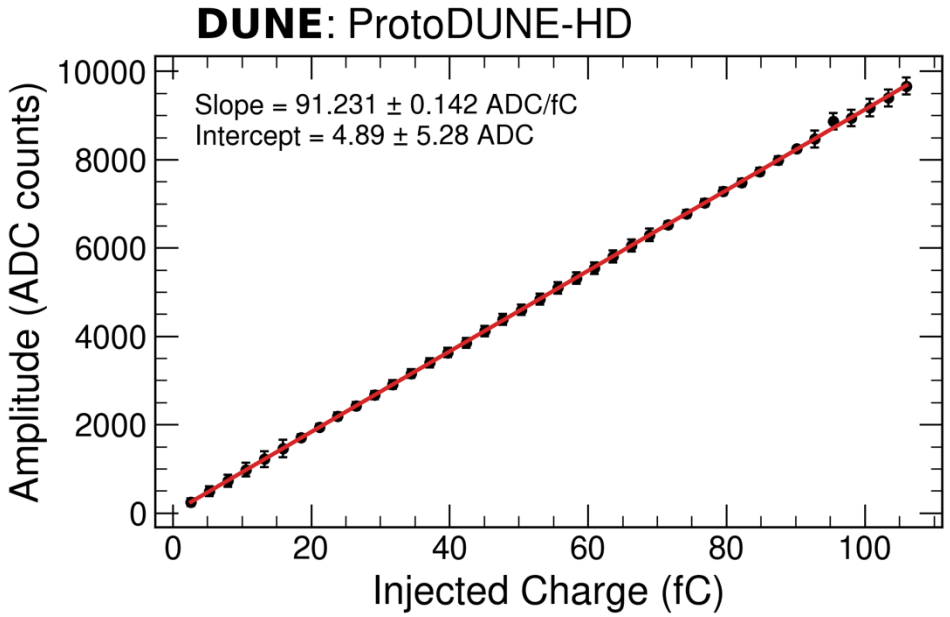


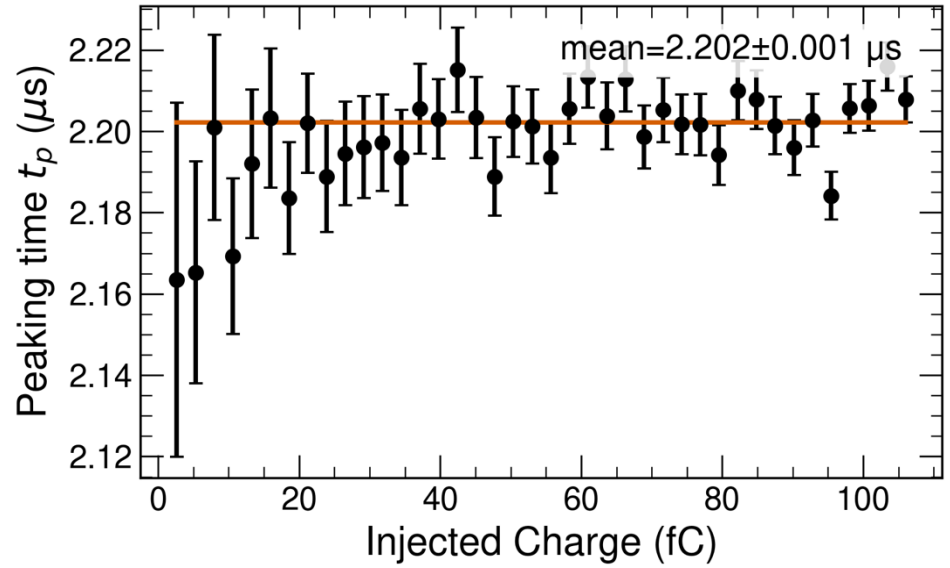
Example 4

A bit of context before jumping to VD

Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**
4. **We can now retrieve *corrected* waveforms.**
5. **If we repeat for the available range of injected charges, we get**





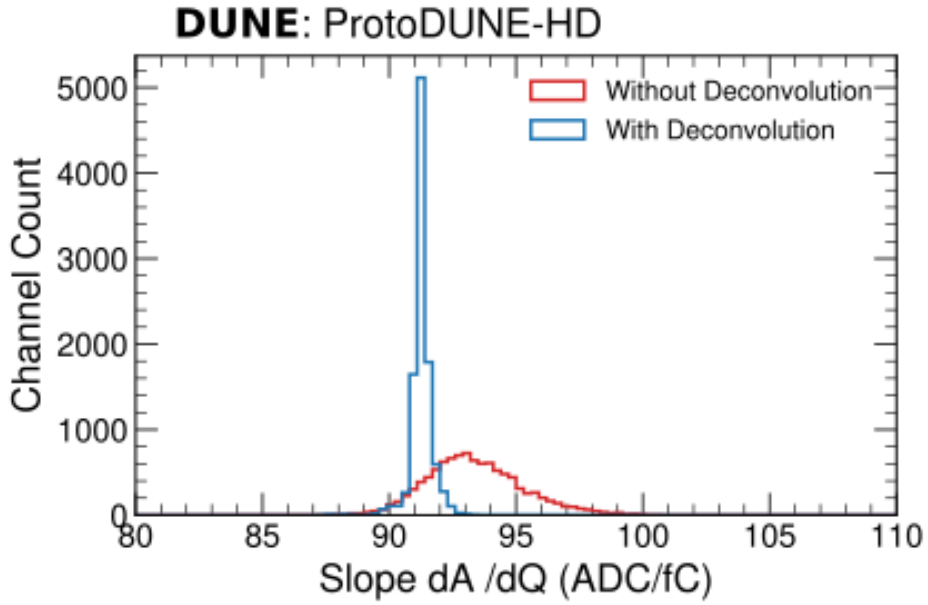
A bit of context before jumping to VD

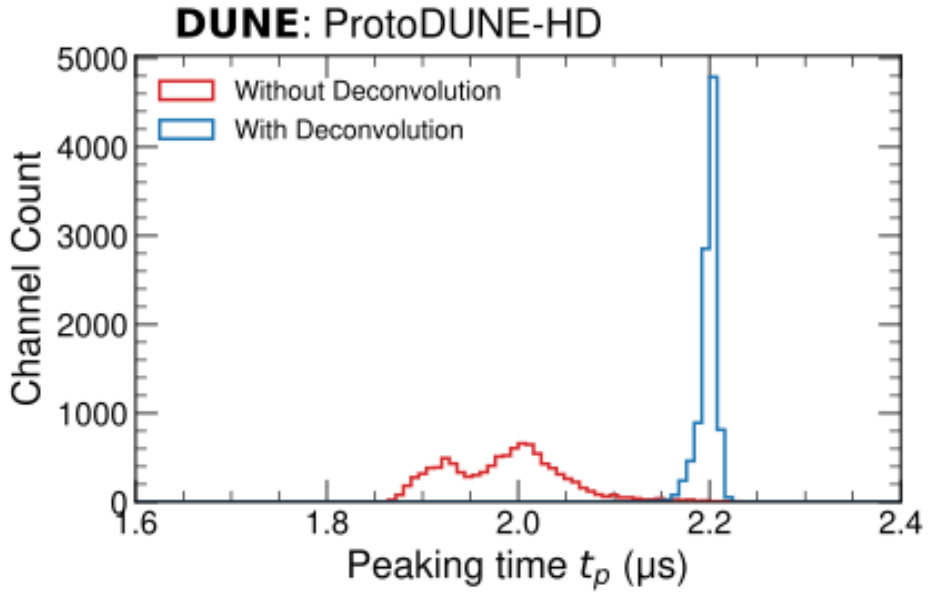
Pulser-based calibration analysis for PDHD was successfully completed.

1. **Analysis starts by fitting per-channel pulses.**
2. **Retrieve fit parameters.**
3. **Input these parameters to the WireCell-based workflow.**
4. **We can now retrieve *corrected* waveforms.**
5. **If we repeat for the available range of injected charges, we get**

- **Gain before: 7.978 mV/fC**
- **Gain after: 7.814 mV/fC**

- **Peaking Time before: 1.986123 μs**
- **Peaking Time after: 2.198 μs**





Wire-Cell Workflow for PDHD

1. Once I have the results from the electronics response fitting, I extract the parameters in `.json.bz2` format for each run.
2. Then I copy this file in my work directory in `dunegpvm`. This is `/exp/dune/data/users/ktellez/wf_correction/wc-params`
3. Then, in my `dunereco` copy, I modify the file `dunereco/dunereco/DUNEWireCell/v10_20_06d00/pdhd/params.jsonnet` to make the variable `chresp` point to the `.json.bz2` file I want to process.
4. Then I go back to my work directory (`wf_correction`) and define my wirecell path and such:
   a) `$ export WIRECELL_PATH=`pwd`/wc-params:$WIRECELL_PATH`
   b) `$ export LD_PRELOAD=$XROOTD_LIB/libXrdPosixPreload.so`
5. Then I process my run with the appropriate `.fcl` file: `$ lar -n1 -c standard_reco_stage1_protodunehd_keepup.fcl [path_to_hdf5_file_to_process.hdf5]`
6. If everything goes well, this will produce a `*_reco_stage1.root`
7. …which is where I extract *before* and *after correction* per-channel waveforms from.

Wire-Cell Workflow for PDHD

- The next natural step was to repeat these steps to achieve the same results for PDVD.
- Before anything, we tested the previous workflow for PDHD using the most recent version of `dunereco` and the most up-to-date dune software (went from `dunesw v10_17_02d00` to `v10_20_06d00`)
- After trying, we realized that we needed to implement the per-channel electronics response in the newer versions of `dunereco`.
- After some modifications (see next slide), we were able to replicate the same results for PDHD (*old wc* vs. *new wc*).
- Then we tried this for PDVD data.

Incorporating the per-channel response

To incorporate the per-channel electronics response into the workflow, we must modify 3 files from here

1 nf.jsonnet

1. Changed from `function(params, anode, chndbobj, n, name='', dft=default_dft)` to `function(params, anode, tools, chndbobj, n, name='', dft=default_dft)`
2. Added `local pc` right after `local grouped`:

```
local pc = tools.perchanresp_nameuses,
local shaper = {
  type: 'perChannelShaper',
  name: name,
  data: {
    noisedb: wc.tn(chndbobj),
    anode: wc.tn(anode),
    elecresponse: wc.tn(tools.elec_resp),
  },
},
```

3. Added the use of `shaper` in `local obnf`, in `channel_filters: wc.tn(shaper)` (commented `wc.tn(single)` out).
4. Changed the definition of `uses` right after `local obnf` to incorporate `shaper` and `tools.elec_resp`: From `uses=[chndbobj, anode, single, grouped], nin=1, nout=1)` to `uses=[chndbobj, anode, single, shaper, grouped, tools.elec_resp] + pc.uses, nin=1, nout=1`

Incorporating the per-channel response

To incorporate the per-channel electronics response into the workflow, we must modify 3 files from here

2 wcls-nf.jsonnet:

1. Changed the way we call `nf_maker`:

   From

```
nf_maker(params, tools.anodes[n], chndb[n], n, name='nf%d' % n) for n in
std.range(0, std.length(tools.anodes) - 1)
```

   to

```
nf_maker(params, tools.anodes[n], tools, chndb[n], n, name='nf%d' % n) for n in
std.range(0, std.length(tools.anodes) - 1)
```

Incorporating the per-channel response

To incorporate the per-channel electronics response into the workflow, we must modify 3 files from here

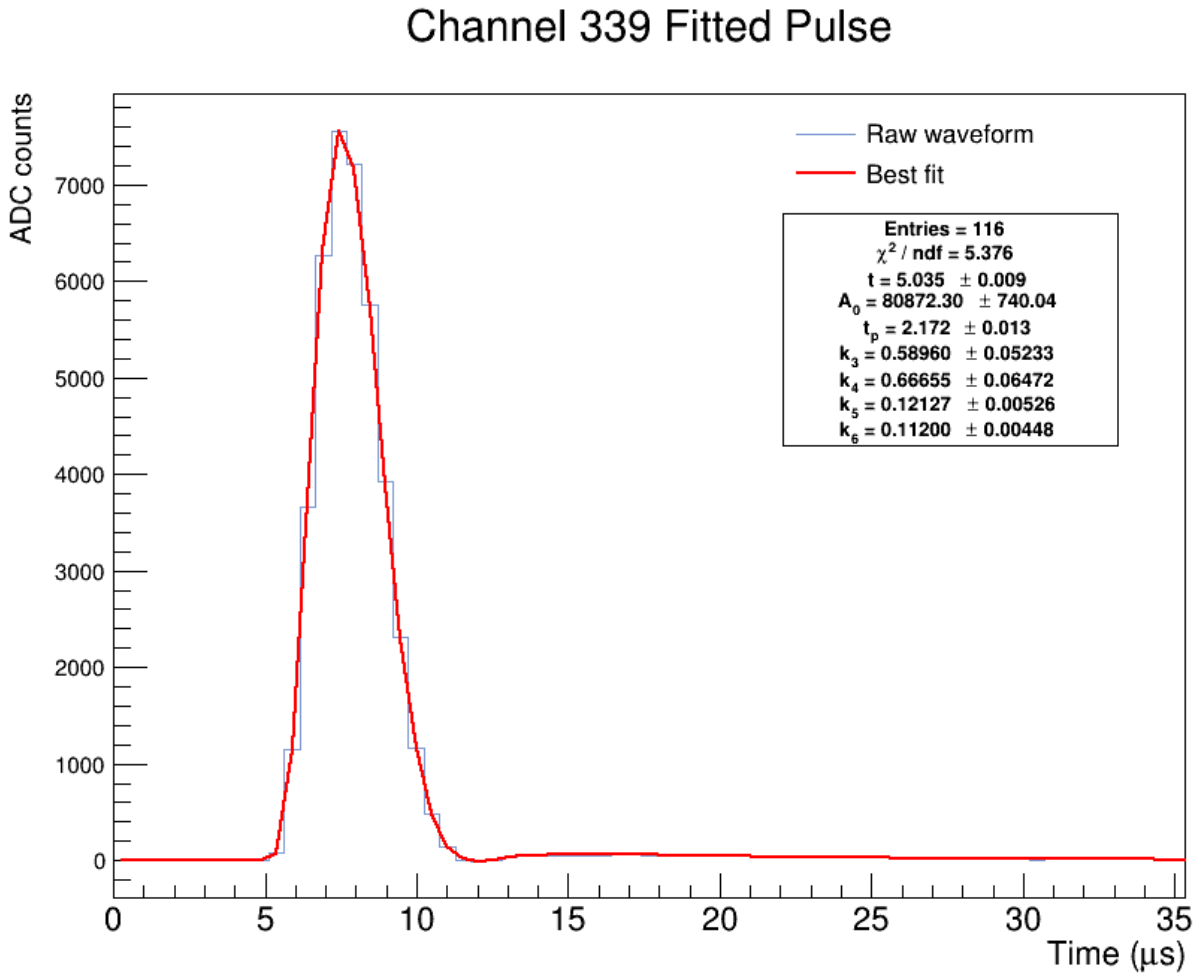
3 params.jsonnet:

1. Changed the channel response from `null` to the `.bz2` file:

   From
   `chresp: null`
   to
   `chresp: "protodunehd-params-channel-response-30413-no_avg_3.json.bz2"`

Trying this workflow with PDVD Data

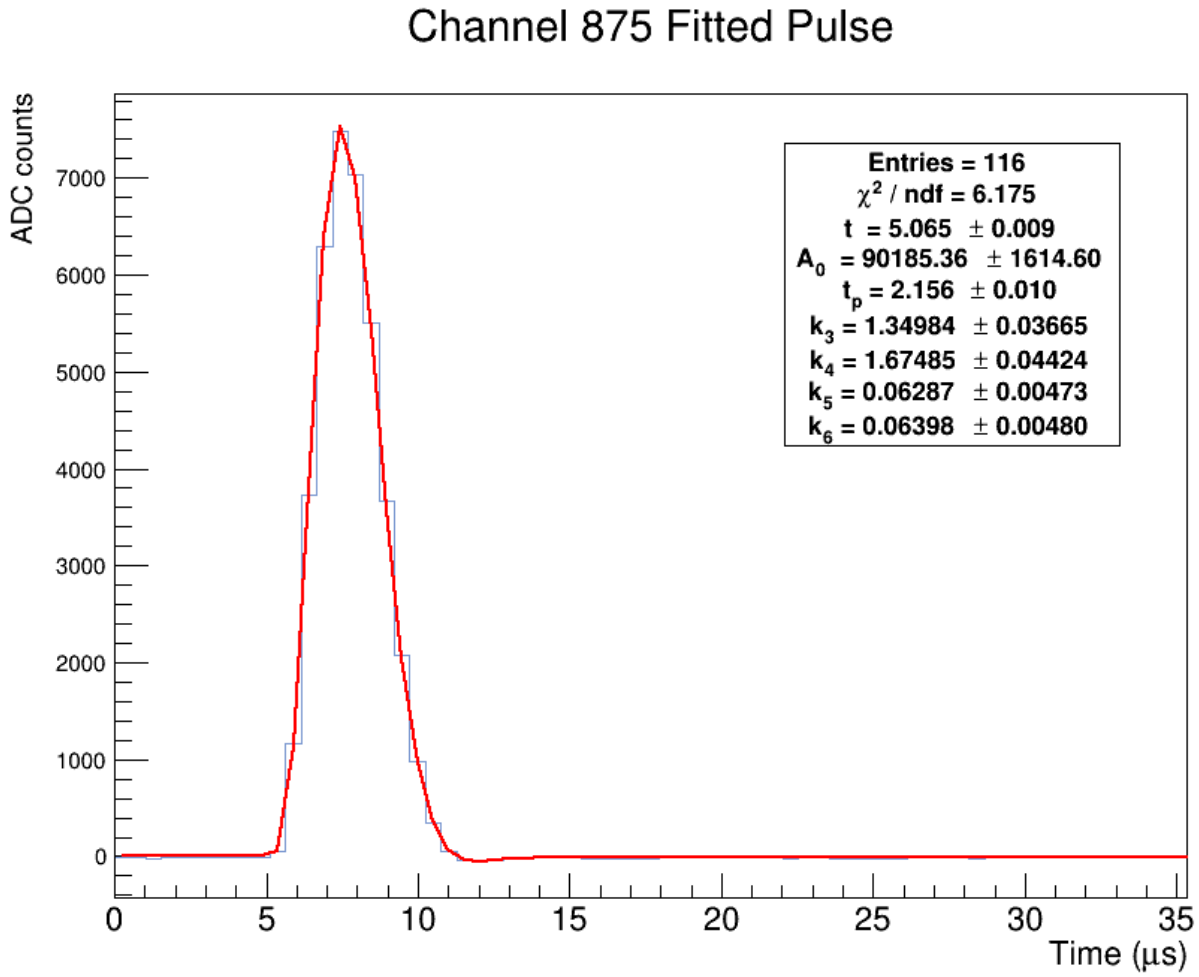
- ✓ **Fitting per-channel pulses.**



Example 1

Trying this workflow with PDVD Data

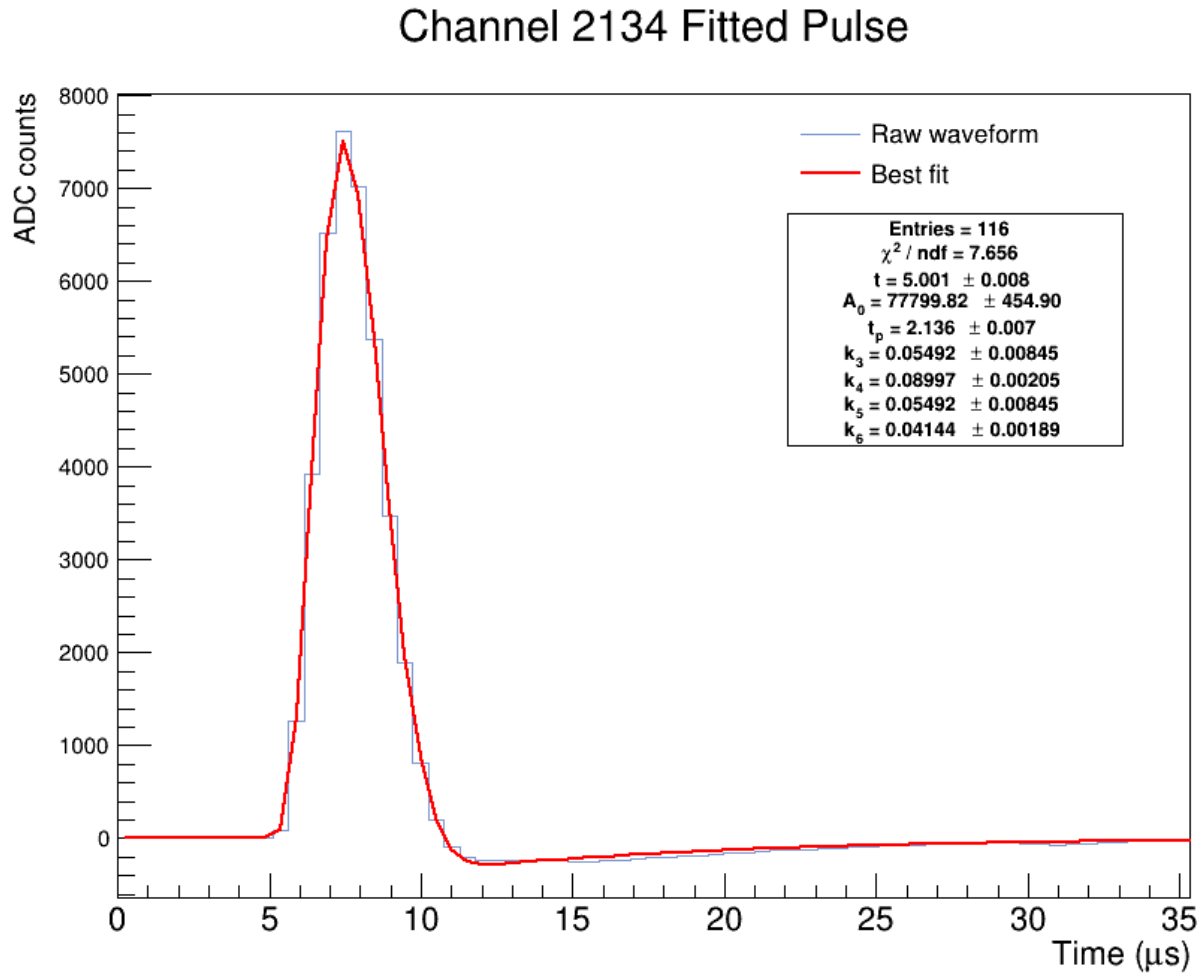
- ✓ **Fitting per-channel pulses.**



Example 2

Trying this workflow with PDVD Data

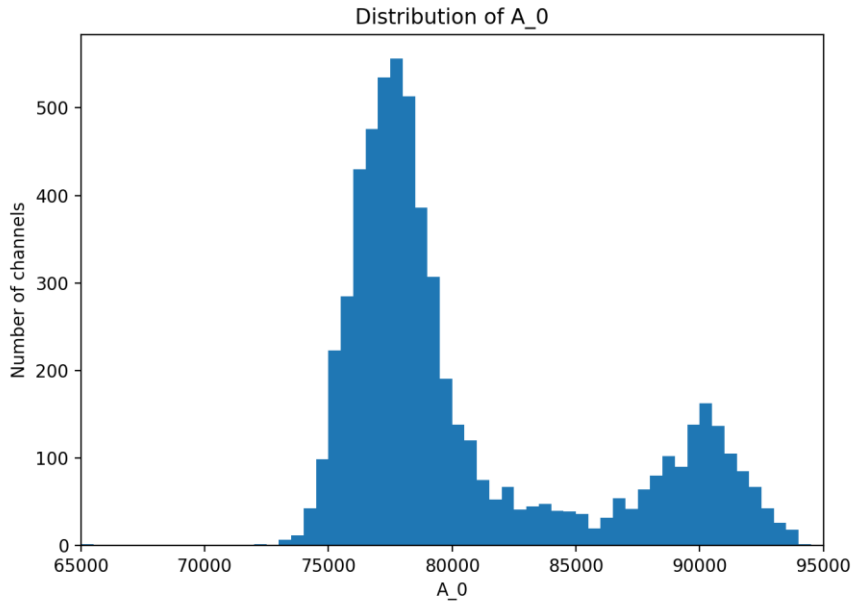
- ✓ **Fitting per-channel pulses.**

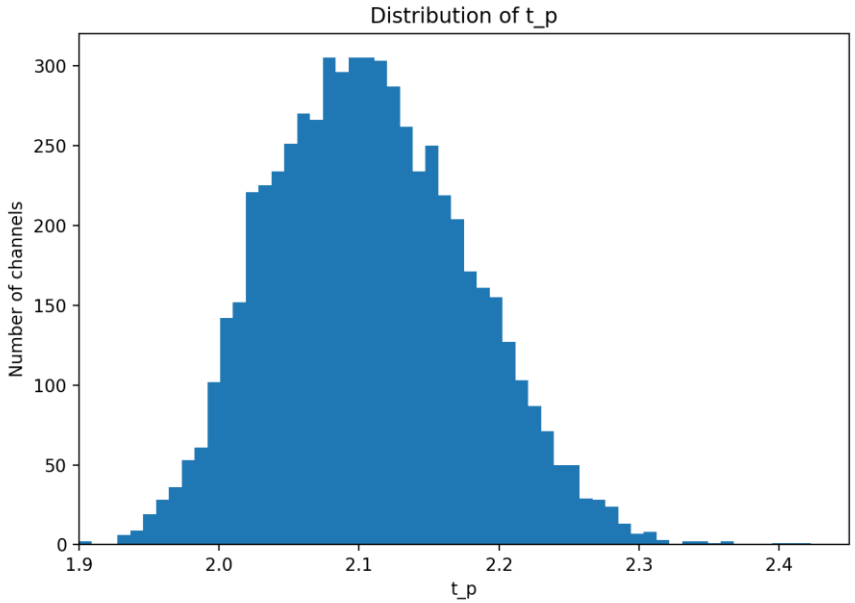


Example 3

Trying this workflow with PDVD Data

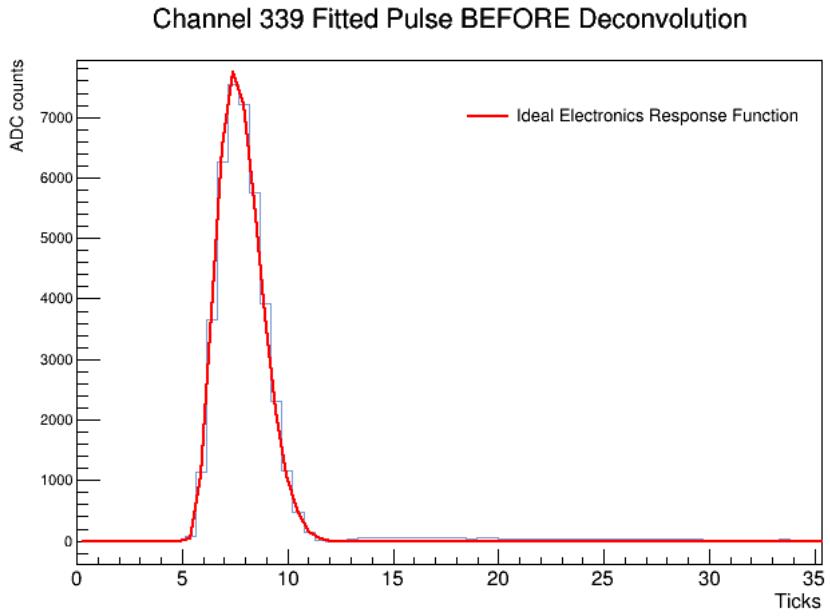
- ✓ **Fitting per-channel pulses.**
- ✓ **Retrieve fit parameters.**

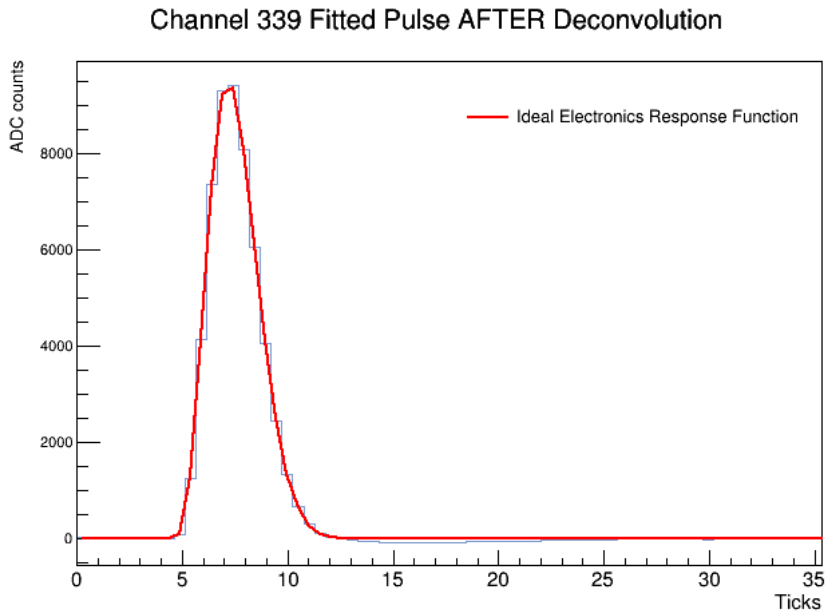




Trying this workflow with PDVD Data

- ✓ Fitting per-channel pulses.
- ✓ Retrieve fit parameters.
- ✓ Input these parameters to the WireCell-based workflow.
- 🥲 *Retrieve corrected* waveforms.

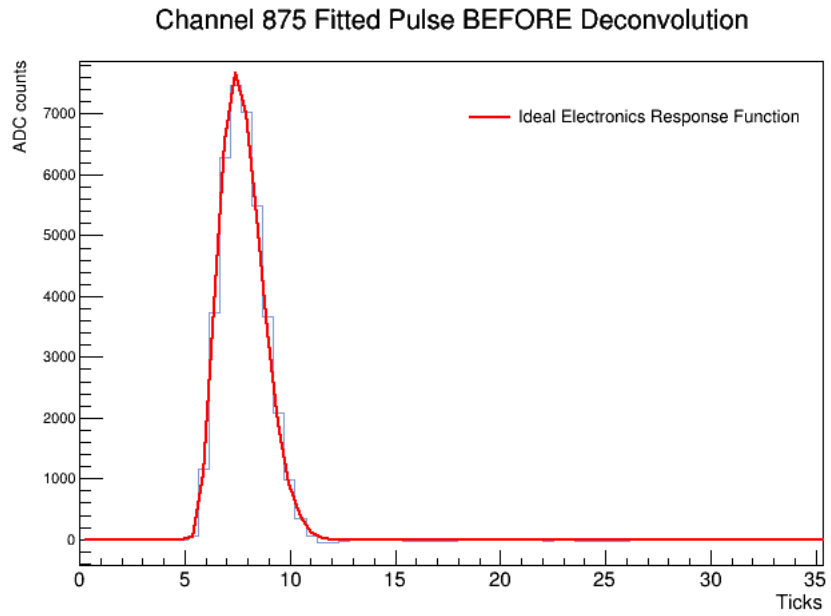


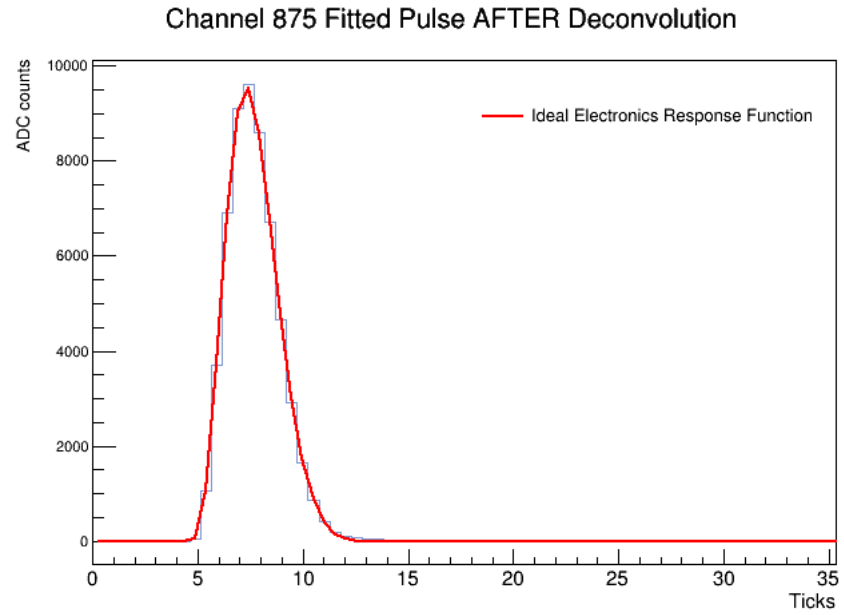


Example 2

Trying this workflow with PDVD Data

- ✓ Fitting per-channel pulses.
- ✓ Retrieve fit parameters.
- ✓ Input these parameters to the WireCell-based workflow.
- 🥲 *Retrieve corrected* waveforms.

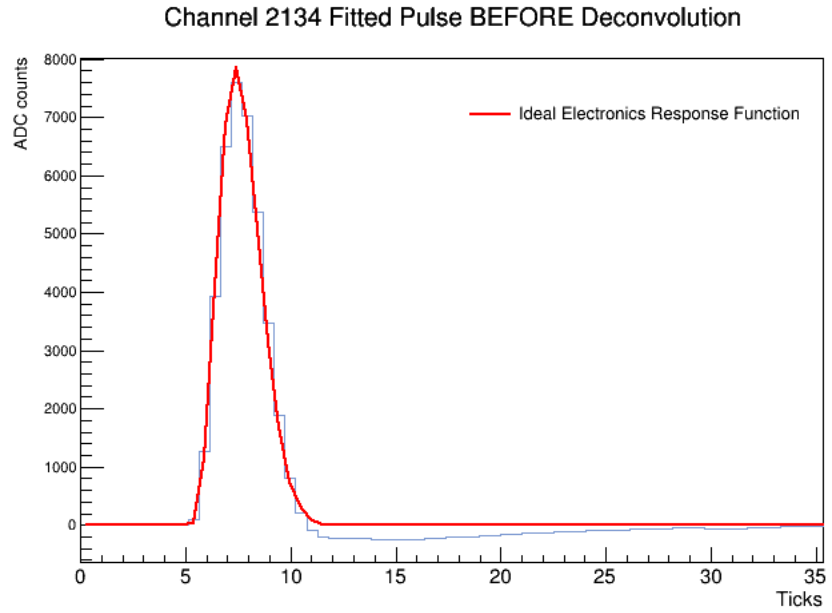


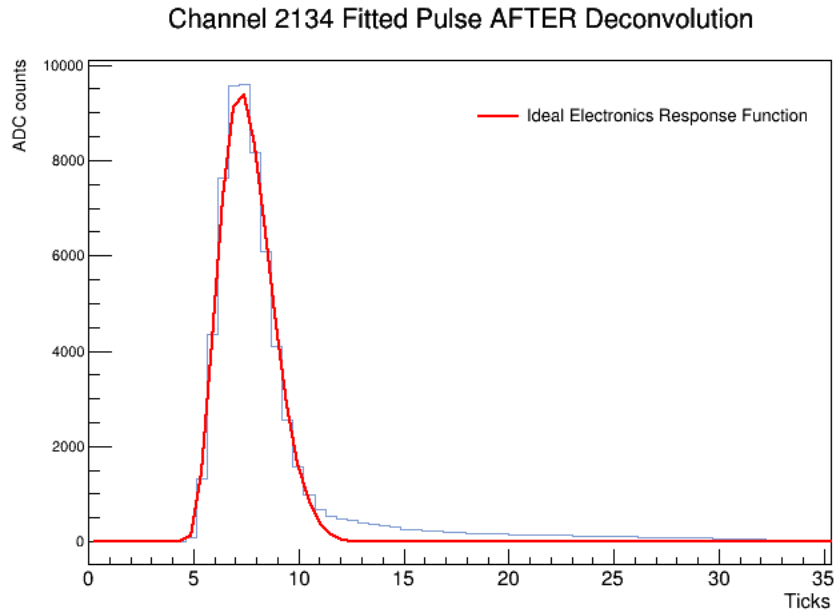


Example 2

Trying this workflow with PDVD Data

- ✓ Fitting per-channel pulses.
- ✓ Retrieve fit parameters.
- ✓ Input these parameters to the WireCell-based workflow.
- 🥲 *Retrieve corrected* waveforms.





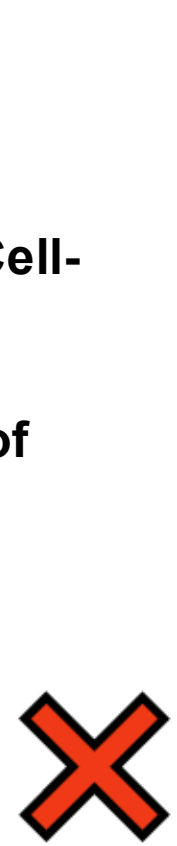
Example 3

Trying this workflow with PDVD Data

- ✓ Fitting per-channel pulses.
- ✓ Retrieve fit parameters.
- ✓ Input these parameters to the WireCell-based workflow.
- 🥲 *Retrieve corrected* waveforms.
- 💀 If we repeat for the available range of injected charges, we get

  - Gain before: 7.62 mV/fC
  - Gain after: 9.95 mV/fC

  - Peaking Time before: 2.04 μs
  - Peaking Time after: 2.33 μs



In the works: Where is the failure?