Update on PMT analysis

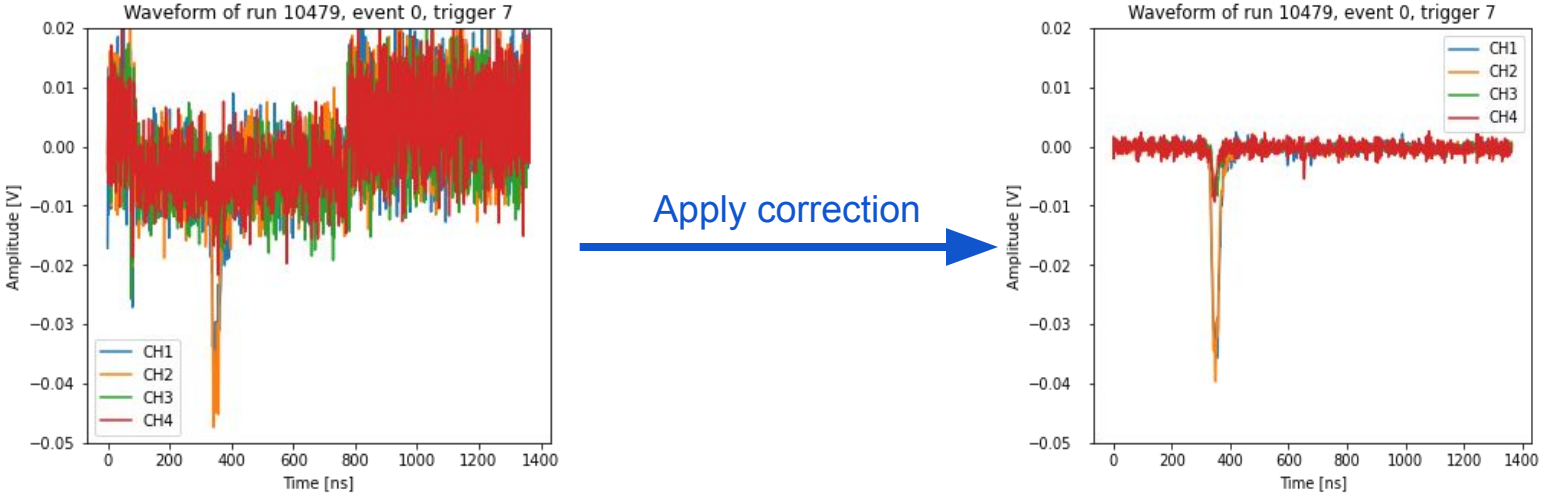
Reconstruction & analysis meeting

2023/03/02

Borra F., Folcarelli M., Marques D., Messina A., Piacentini S.

DRS correction

- implemented the **Domino Ring Sampler** correction:
  - **RMS** of ~ 1.4 ADC counts (< 1mV) aka 0.65 pC uncertainty on a 200 ms integration window
  - The correction is **satisfactory**



- Future plans:
  - generalize the correction,
  - **include** the functions in the **CYGNO LIB**

PMT reco

- implemented PMT variables in **the reco software**

```
waveform_test = dav_simplePeak(0, waveform_f[ch], name)
```

From data banks, using ***Cygno libs***

```
class dav_simplePeak:
    def __init__(self,length,y_array, name):

        self.name = name

        self.length = length
        self.y_array = y_array

        self.freq = 0.75
        self.x_array = np.linspace(0,1024/self.freq,1024)


        self.baseline = self.getBaseline(0,100)
        self.invert_and_center_WF(self.baseline)


    def getBaseline(self,n_offset,n_samples):
        bl = np.mean(self.y_array[n_offset:n_offset+n_samples])
        return bl

    def getRMS(self, n_offset, n_samples):
        rms = np.std(self.y_array[n_offset:n_offset+n_samples])
        return rms

    def invert_and_center_WF(self, baseline):

        demo_y = list(self.y_array)
        for i in range(len(demo_y)):

            demo_y[i] -= baseline
            demo_y[i] *= (-1.)

        self.y_array = tuple(demo_y)

    def getTotalIntegral(self, begin=None,end=None):

        if begin is not None:
            return np.sum(self.y_array[begin:end])
        else:
            return np.sum(self.y_array)
```

```
def create_dav_PMTVariables(self):

    ## Better test
    self.outTree.branch('dav_max_ampl', 'F', title = 'David\'s max voltage')
    self.outTree.branch('dav_last_elem', 'F' , title = 'David\'s last element')
    self.outTree.branch('dav_first_last_ele_list', 'F',lenVar= 'whatsthis' , title = 'David\'s

    self.outTree.branch('dav_nPeaks', 'I', title = 'David\'s number of peaks')
    self.outTree.branch('dav_peak_position', 'F', lenVar = 'whatsthis')

    ## Real variables
    self.outTree.branch('pmt_wf_run', 'I')
    self.outTree.branch('pmt_wf_event', 'I')
    self.outTree.branch('pmt_wf_channel', 'I')


    ### 2nd batch
    self.outTree.branch('pmt_baseline', 'F')
    self.outTree.branch('pmt_RMS', 'F')
    self.outTree.branch('pmt_tot_integral', 'F')
    self.outTree.branch('pmt_tot_charge', 'F')


def fill_dav_PMTVariables(self,wf_info,wf):

    self.outTree.fillBranch('pmt_wf_run', wf_info[0])
    self.outTree.fillBranch('pmt_wf_event',  wf_info[1])
    self.outTree.fillBranch('pmt_wf_channel',  wf_info[2])

    self.outTree.fillBranch('dav_max_ampl', wf.getMaxAmpl())
    self.outTree.fillBranch('dav_last_elem', wf.getLastElem())
    self.outTree.fillBranch('dav_first_last_ele_list', [fl for fl in wf.getLastFirstElem()])
    # self.outTree.fillBranch('dav_nPeaks', len(wf.getPeaksPosition()))
    self.outTree.fillBranch('dav_peak_position', [pp for pp in wf.getPeaksPosition() ] )

    ### 2nd batch

    self.outTree.fillBranch('pmt_baseline', wf.getBaseline(0,100))
    self.outTree.fillBranch('pmt_RMS', wf.getRMS(0,100))
    self.outTree.fillBranch('pmt_tot_integral', wf.getTotalIntegral())
    self.outTree.fillBranch('pmt_tot_charge', wf.voltageToCharge(wf.getTotalIntegral()))
```

```
## Read PMT waveforms
if name.startswith('DGH0'):
    # print('pmt waveform here')
    header = cy.daq_dgz_full2header(mevent.banks[key], verbose=False)
    waveform_f, waveform_s = cy.daq_dgz_full2array(mevent.banks['DIG0'], header)
    pmt = True
else:
    pmt = False
```

Missing:

★ Add peak analysis

★ Test combination with camera reco and see if both recos are working stand-alone and together

Position reconstruction

- Trying to reconstruct **slices of waveforms**, in order to obtain the positions at different times:
  - working on ^{55}Fe events for simplicity, tried **10** ADC samples → **too short**
  - first attempt on **z reconstruction** by means of the temporal information of the waveform.

- Future plans:
  - test on wider temporal samples → using **1^{55}Fe signal** as reference?
  - test on long signals, try to evaluate temporal evolution of the waveform