Particle Tracking Inefficiencies in the Qweak Experiment

A thesis submitted in partial fulfillment of the requirement
for the degree of Bachelor of Science with Honors in
Physics from the College of William and Mary in Virginia,

by

Zach Hutcheson

Advisor: Prof. David S. Armstrong

Senior Research Coordinator: Gina L. Hoatson

Williamsburg, Virginia
May 2017

Contents

Acknowledgments

I would like Doctor Armstrong without who I would have been able to do absolutely none of this.

List of Figures

Abstract

The Q_{weak} experiment was run at JLab between 2010 and 2012 with the purpose of precisely deriving the weak charge of the proton, Q_W^p. This quantity is predicted in the standard model, so the experiment serves to test the standard model by comparing an experimentally derived value of Q_W^p with the theoretical value of Q_W^p=0.0710 $\pm$ 0.0007 [1]. The experiment was done by shooting a polarized electron beam at a liquid helium target and looking at scattering with parity violating asymmetry. Vertical and Horizontal drift chambers were used in order to determine the momentum of the scattered electrons after hitting the target . The beam was set at a lower current in order to not damage the drift chambers[2]. This momentum value is used to reduce the error on the final measured Q_W^p value. The drift chambers consist of a crosshatch of wires in a gas which ionizes when electrons traverse the gas. This ionization causes the electrons to drift to the charged wires. This system requires that the inefficiencies must be small in order to have the VDC (vertical drift chamber) construct an accurate trajectory for each electron. It was determined that there were significant inefficiencies in some locations causing an unexpected momentum distribution. Previous work has been to determine from which wires the inefficiencies arose and in doing so reducing the inefficiencies by determining what caused them and correcting for the inefficiencies in the data. By differences in hit time we were able to examine one of the sources of the inefficiencies and take steps toward better understanding the cause of the inefficiencies.

Chapter 1

Introduction

The Standard Model is a theory in physics describing the elementary particles of the universe as well as how they interact via the fundamental forces of Electromagnetism, Strong, and Weak Nuclear force. Since its inception in the 1970's [5] it has proven extremely accurate in how it predicts experimental results but it is known to have a large gaps of understanding proving that it is not complete, for example, gravity is not included. Due to this, many physicists attempt experiments with the purpose of finding results that indicate a different of physics beyond the standard model. The Q_{weak} experiment at Jefferson Lab was one of these such experiments, a low-energy high-precision experiment whose purpose was to precisely measure the weak charge of the proton through parity-violating electron scattering. The purpose of the research described in this paper is to reduce wire inefficiencies in the vertical drift chambers used in that experiment, by identifying which wires caused them. A second motivation was to discover the reason behind these issues so that these can be accounted for in the data analysis giving a more accurate momentum distribution.

Chapter 2

The Qweak Experiment

2.1 Overview

The Q_{weak} experiment was done in Thomas Jefferson National Laboratory's (Jlab's) Hall C and used parity-violating electron-proton scattering in order to determine the weak charge of the proton Q_W^p [1]. Within the experiment a polarized beam of electrons was scattered off of a nonpolarized liquid hydrogen target. The weak charge of the proton can be calculated by examining the parity-violating scattering between the target and the electrons. The experiment measured the smallest and most precise asymmetry of electron-proton scattering and thus provides us with the most precise calculation of the weak charge of the proton [2].

2.2 Tracking System

The setup of the experiment can be seen in figure 2.1. In the experiment a 35 cm container of liquid hydrogen was used as a target for a 180 microamp longitudinally-polarized 1.16 GeV electron beam. A large torodial magnet was then used to analyze and select those electrons that scattered elastically at angles between 6 and 12 degrees. The magnetic field made it so that electrons with different momentum would bend differently making it possible for one to determine momentum based on where

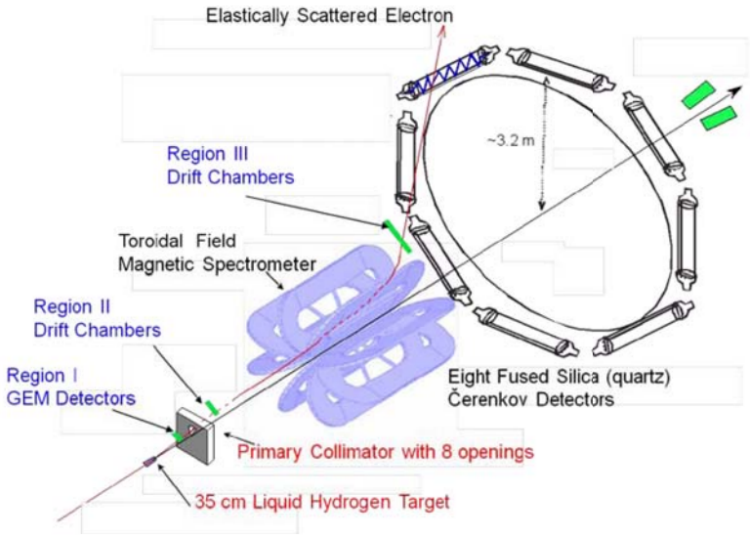

Figure 2.1: The layout of the Q_{weak} experiment's experimental apparatus [2]

the electron would hit the main detector which consisted of 8 Cerenkov detectors. The scattered electrons are then directed into the Cerenkov detectors. In order to determine the trajectories and thus the momentum more accurately, at lower currents wherein they would not be destroyed, horizontal drift chambers (Region II Chambers in Fig. 2.1) were placed in front of the magnet and vertical drift chambers (Region III chambers in Fig 2.1) were placed after the magnet.

2.3 Q_{weak} results

A preliminary result of the experiment was published dealing with 4% of the data collected in the experiment[1]. It measured the parity-violating asymmetry in elastic ep scattering at low momentum transfer, Q^2=0.025 GeV^2. At that point the value of Q_W^p was calculated at Q_W^p=0.064 $\pm$ 0.012, agreeing very well with the standard model prediction of Q_W^p(SM)=0.0710 $\pm$ 0.0007 [1].

2.4 Drift Chambers

In order to more accurately determine the momentum distribution after the parity violating scattering two drift chambers were used, the Horizontal and the Vertical Drift Chambers. A drift chamber is a type of wire chamber used to detect particles in which there are multiple wire planes contained in a gas mixture, with a drift electric field. When an electron passes through a wire plane the gas is ionized and the released electrons drift towards nearby wires. When the drift electrons collide with a wire a signal is sent through the wire from which information about the how long and far the electron drifted can be reconstructed. The Horizontal and Vertical Drift Chambers are made slightly differently but collect information in similar ways [3].

2.5 Drift Chamber Construction

The Drift Chambers consisted of two packages with two chambers in each and in each chamber two planes of wires. Each package was oriented 180^o with respect to the other and was manually rotated. The Vertical Drift Chamber (VDC), the focus of this project, was designed and constructed at William and Mary. Package one of the VDC contains the chambers Vader and Leia, where Vader is in front of Leia. In Package 2 we have the chambers Yoda and Han, where Yoda is in front of Han. Within each chamber there are two wire planes, V and U wherein the V plane is in front of the U plane. Each wire plane consists of 280 gold-plated tungsten wires with diameter 25 micrometers suspended at 26.5 degrees with respect to the long side of the chamber and the chamber is filled with a gas mixture of Argon and Ethane. The distance between each wire is 0.5 cm and the two planes form a crosshatch.

2.5.1 Tracking

The signals caused by particle hits give us information from which we can reconstruct the time of hit, which combined with combined with knowledge of the velocity can be used to extract information on the time and location. These may be used to make a "tree line" in the plane, that connects hits together to make a plausible path. Combining constructed tree lines it is possible to construct a three dimensional trajectory through the VDC chambers called a partial track. If a partial track corresponds to another in the horizontal drift chambers a complete track can be constructed.

2.6 Inefficiencies

When examining data it was found that some wires seemed to be hit much less frequently then surrounding wires. Incongruous wires giving bad information would distort the measured momentum distribution. A combination of noisy wires, i.e. picking up radio signals, and dead wires might account for the vast majority of the inefficiencies. By attempting to correct for these and form a more accurate trajectory we hope to correct the calculated momentum distribution.

2.7 Previous work

We began building on Computer programs written in a summer REU Project in order to analyze the data[4]. We can see the plots of the wire hits for raw wire hits (Figure 2.2), partial tracks, tracks (Figure 3), and number of wire hits (Figure 2.3) for each plane. These data can be manipulated through use of programs such as Q2inefficiency2 and Q2remove_wires2. Q2inefficiency can be used to produce plots of wire residuals (figure 2.4), the distance drifted to each wire, for each chamber as well as various other histograms. We also are utilizing Q2removewires2 which was used

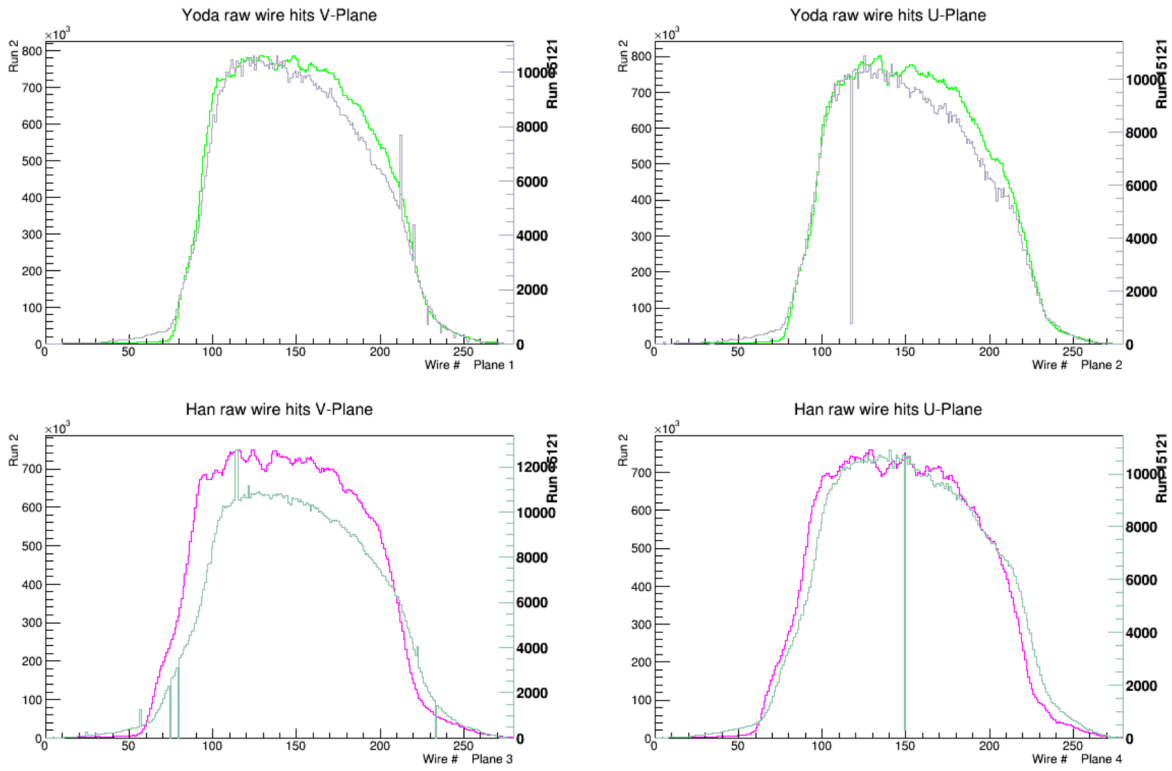

Figure 2.2: The relative probabilities of raw wire hits in Package 2 plotted vs wire numbers with simulated data in green and real data in grey for those in Yoda and simulated data in purple and real data in green for those in Han. Note the bell shape due to the beam being aimed at the center and the sudden changes in some wires do most likely to dead wires and noisy wires. These graphs were generated with the program Q2Inefficiency2 written by Jennica LeClerc [4].

to look at various wires to determine where the Vertical Drift Chamber inefficiencies were occurring and attempt to figure out what causes them. The data from the program can be used to to see if tracks can be reconstructed.

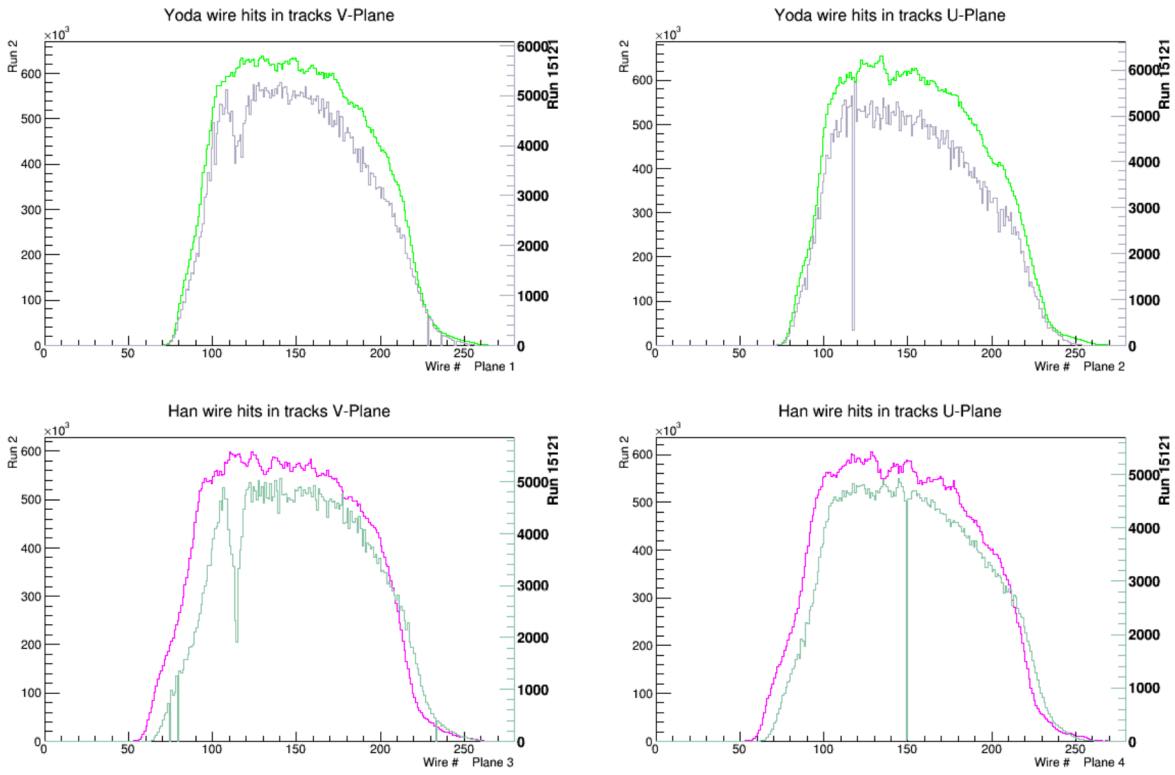

Figure 2.3: The distribution of wire hits per event in Package 2 with with simulated data in green and real data in grey for those in Yoda and simulated data in purple and real data in green for those in Han [4].

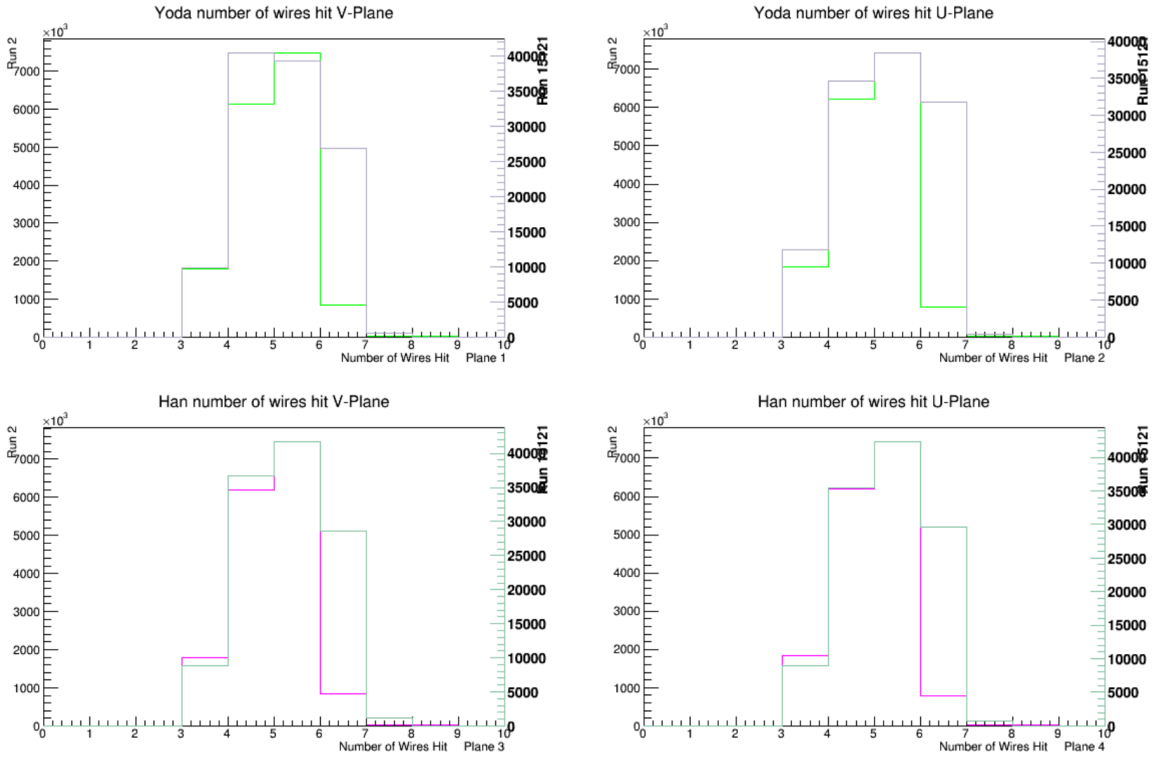

Figure 2.4: The number of wires hit in one event in given wire plane in Package 2 with simulated data in green and real data in grey for those in Yoda and simulated data in purple and real data in green for those in Han [4].

Chapter 3

Results and Discussion

Starting midyear, the focus of investigation shifted to investigation of possible crosstalk between the VDC wires. Which is to say we began to investigate if inefficiencies could have been caused by the signal of one wire being picked up by one or more adjacent wires, causing the system to register a hit when in fact the actual hit was on its neighbor. Utilizing Dr. Armstrong's program VDC_Crosstalk we investigated the mode of the time difference between adjacent wires (Figure 3.1) , which is to say the bin with the most entries in the histogram of time differences for a wire. We then marked any wires with a peak time difference of ten nanoseconds or less as of interest, choosing an arbitrary cut off point higher then is probably needed. Using this methodology we were able to identify wires that had potential crosstalk to further investigate. Histograms were then constructed for the wires of interest. These histograms depicted the number of hits per time difference for a certain wire. A wire with no indication of crosstalk would have a peak at some nonzero value making a sort of "bell shape" around this peak value (Figure 3.2). Histograms with clear peaks at zero conversely indicated the presence of crosstalk (Figure 3.3). In a situation other then crosstalk it would be very unlikely for this kind of behavior to occur requiring a statistical near impossibility. These crosstalk indicating wires tended to appear in bands in package one plane one from wires 140-146 and package one plane 2 wires 185-190, irrespective

of run number. This indicates that these bands of wires had an issue causing crosstalk which persisted throughout the experiment. Further cuts were made examining only the first hit per event, the only one actually used in the track reconstruction software, in order to attempt to ascertain if the apparent crosstalk was an artifact of subsequent hits. Examining the difference between the histograms utilizing all hits in an event and those utilizing only the first hit per event it becomes clear that, for most wires, utilizing only the first hit does very little to effect the distribution (Figure 3.4). The exception to this appears to be "noisy" wires such as Package One Plane 4 Wires 147-149 of run 18482. In such cases removing the hits beside the first in an event result in a much more normal set of time differences (Figure 3.5) In an attempt to see if there were any wires experiencing anomalous numbers of hits a graphs were also generated of the mean number of hits per wire (Figure 3.6). These graphs indicated that, for the vast majority of runs, there are very few hits beyond the first hit the exception to this being noisy runs in which case we can see noisy wires having an extremely large number of extra hits (Figure 3.7). An interesting byproduct of the mean hit graph was noticed the distinctive bowl shape, preserved in all runs and planes. This might be due to a steep angle at the edges of the planes.

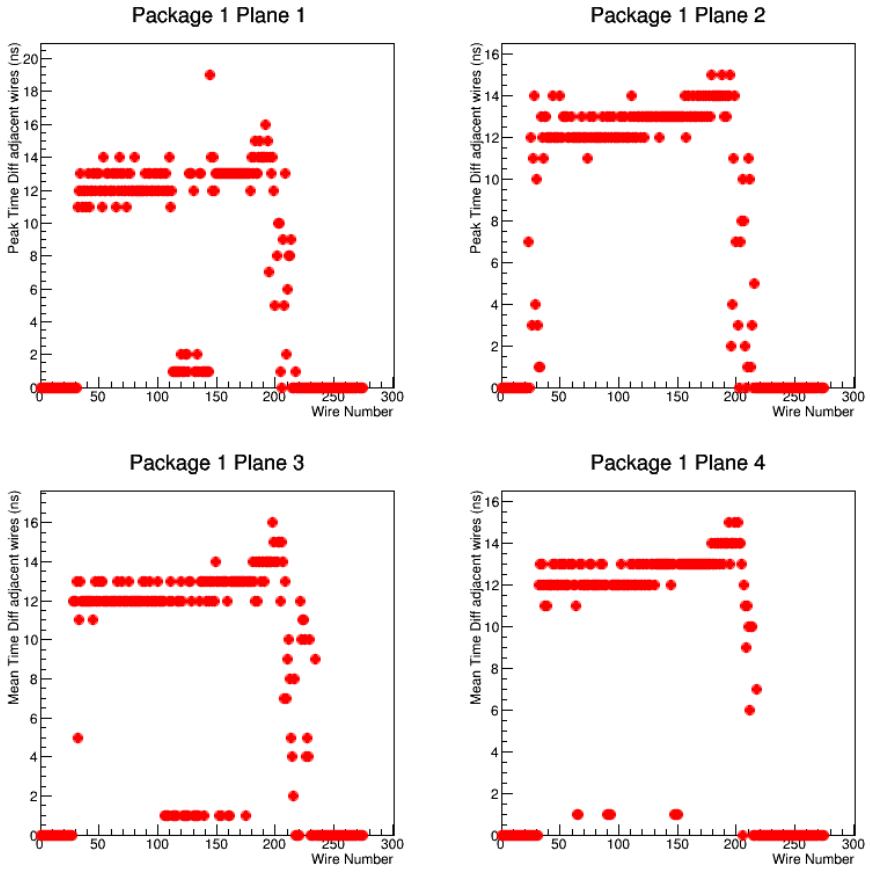

Figure 3.1: The Mode of the time difference in nanoseconds between adjacent wires in Package One Run 18482. This was used to determine wires of interest for crosstalk analysis.

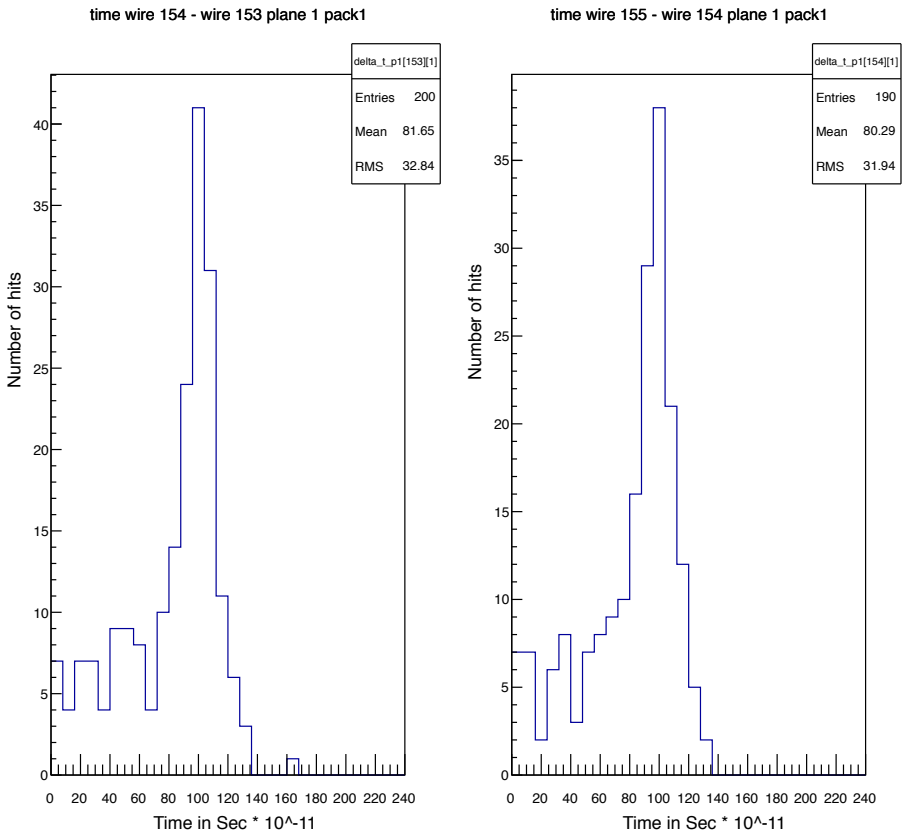

Figure 3.2: Example histograms of time differences between adjacent wires for wires not indicative of crosstalk, in run 13681. Note that the peaks are not at zero showing a typical time difference between the wires, as expected.

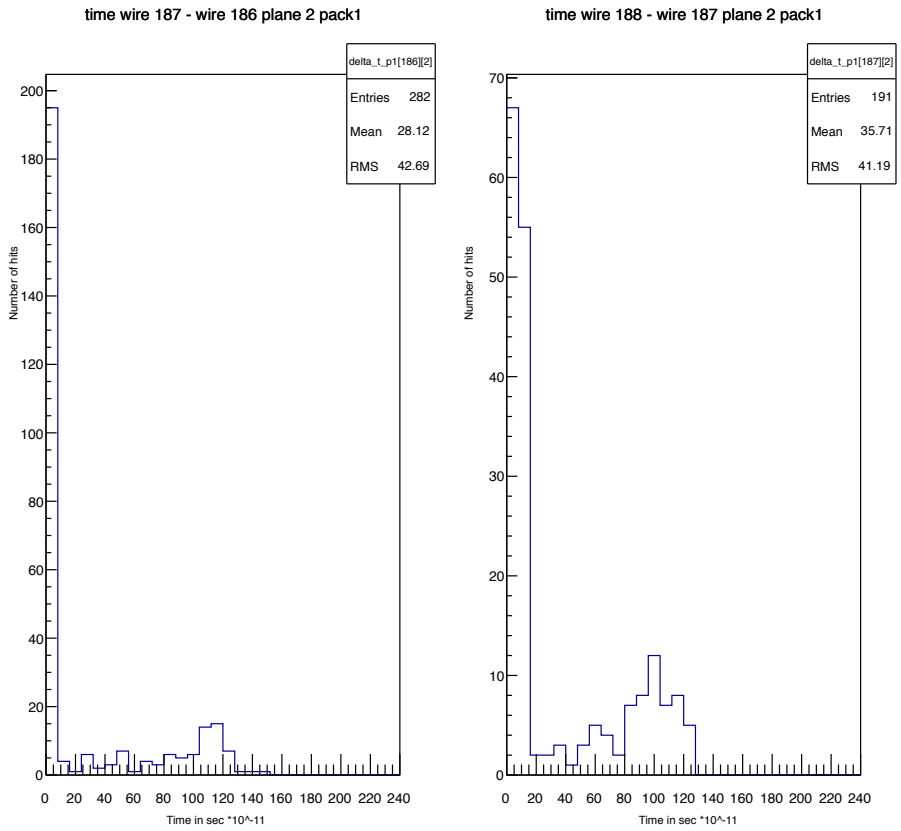

Figure 3.3: Example histograms of time differences between adjacent wires for wires indicative of crosstalk in run 13653. Note that while there are peaks at nonzero values they are overshadowed by the spikes at zero.

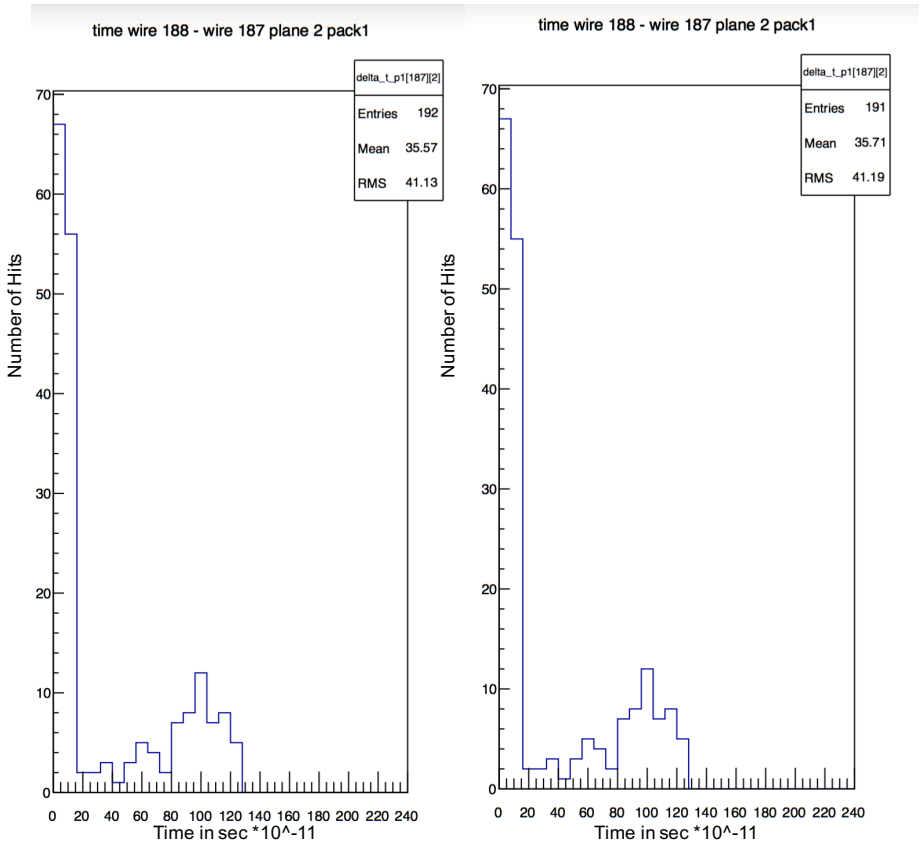

Figure 3.4: As can be seen here, with histograms all hits per event on left and histograms of only first hits per event on the right, for a wire where crosstalk is evident in most cases excluding all hits but the first has very little effect resulting in almost identical histograms. This is due to the very low number of hits past the first hit for most events.

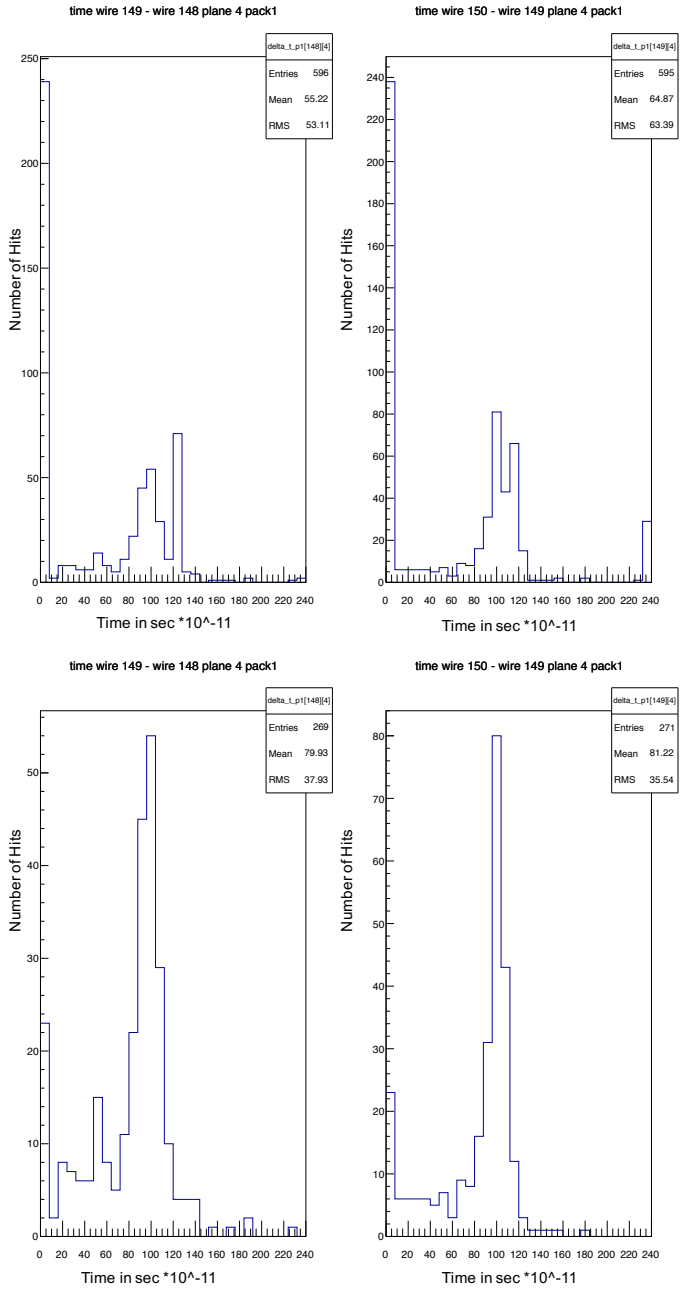

Figure 3.5: As can be seen here, with histograms all hits per event on top and histograms of only first hits per event on the bottom, for a noisy wire excluding all hits but the first in an event removes much of the inefficiency

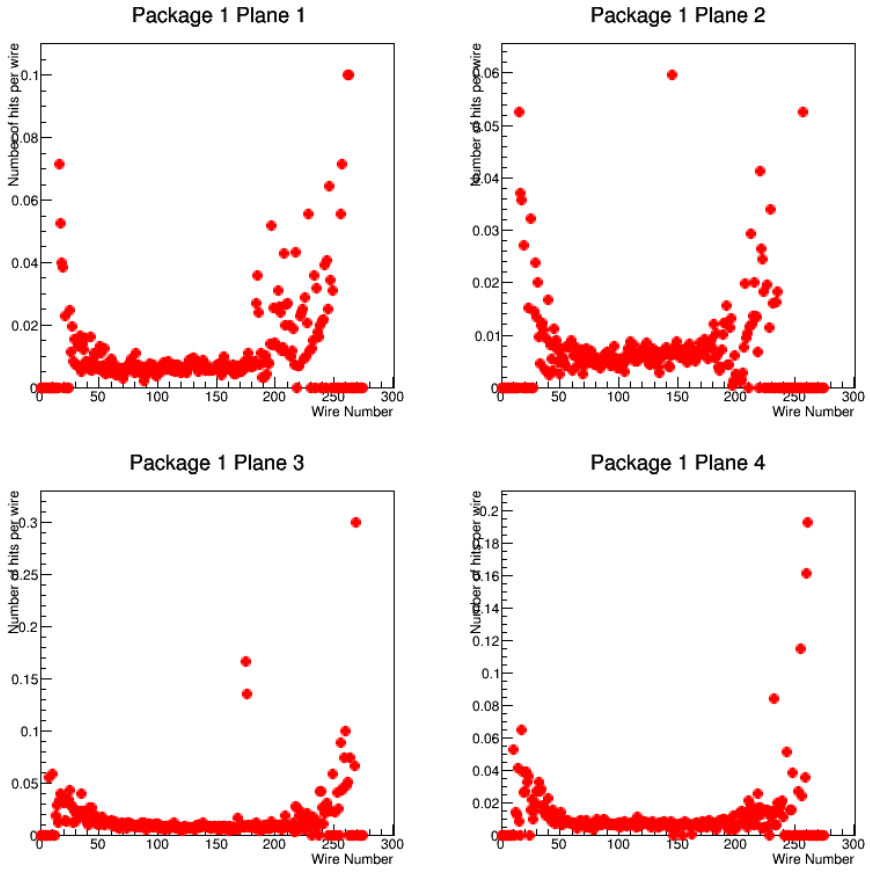

Figure 3.6: The mean hit number on a wire per event vs. wire number, for planes in Package 1, for run 13653 note the bowl shape possibly due to the steep angle of the beam at that edges of the plane.

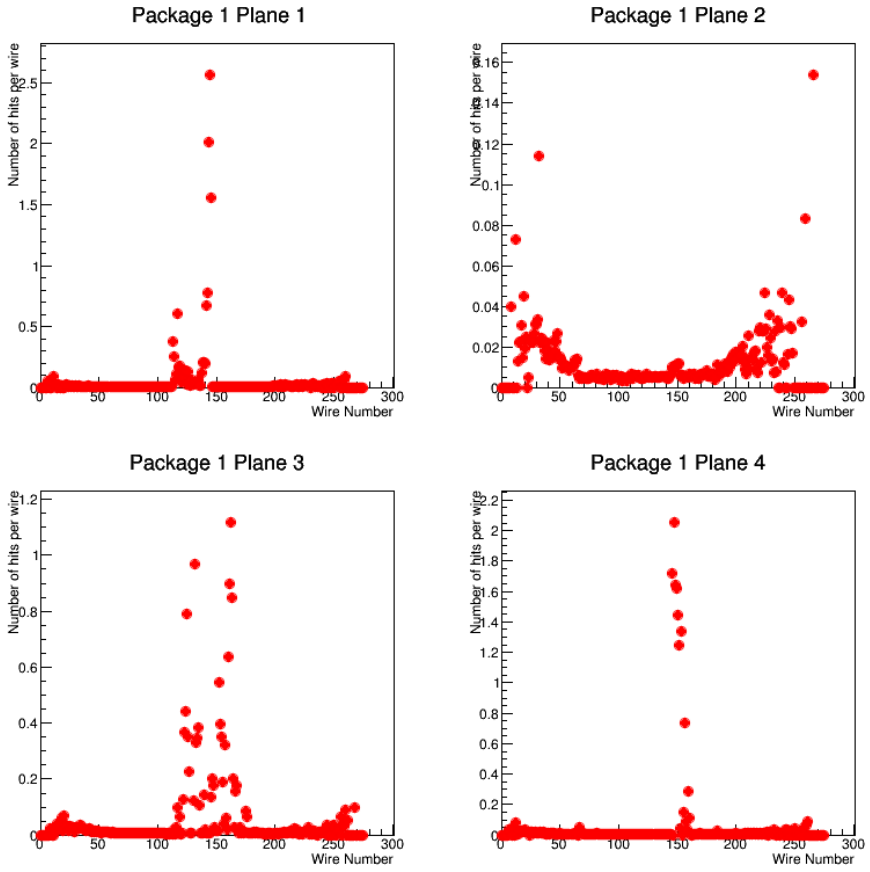

Figure 3.7: The mean hit number on a wire per event vs. wire number, for planes in Package 1, for an apparently noisy run (run 18482). Note that mean hit numbers of apparently noisy wires far eclipse mean hit numbers of a normal run, with values ranging from 5 to 25 times as large.

Chapter 4

Future Research

The focus of our research was on crosstalk between adjacent wires, future research could be done with a focus more upon noisy and dead wires as well as pinpointing specific wires. From there the main step that needs to be taken would be to account for the wires within the data. One manner in which this might be done for the crosstalk would be, upon recognizing a string of crosstalk selecting one of the crosstalking wires and excluding the others firing at the same time. This selection would be done by examining which wire hit best fit the calculated trajectory.

Appendix A

VDC_Multiplicity

A.1 Code sample

The following is the C code which generates the mean hit number graphs as well as the histograms of hit number per wire. It is based on the VDC_crosstalk code of Dr. Armstrong

```
/*************************************************************/

/*   Zach Hutcheson   */

// Name:   VDC_crosstalk_Multest.C
//Based on the code VDC_Crosstalk.C by David Armstrong
// root[0] .L Extensions/Macros/Tracking/VDC_crosstalk.C
// root[1] vdc_crosstalk(1,100000,13653)
//             ->  analyzes first 100K events of run 13653
//
// root[1] q2_ineff(1,-1,13653)
//             -> looks at all events for Run 13653
//

#include "TSystem.h"
#include "TStyle.h"
#include "TChain.h"
#include "TCanvas.h"
#include "TROOT.h"
#include "TGraph.h"
#include "TFile.h"
#include "TH1D.h"
#include "TF1.h"
#include "TH1F.h"
#include "TLeaf.h"
#include <fstream>
#include <iostream>
#include <cstring>
#include <cstdlib>
#include <sstream>
```

```
#include <iomanip>
#include <vector>

void vdc_crosstalk(int event_start = -1,int event_end = -1,int run = 8658, bool opt = false, TSt
{

  int n=275;

  gROOT->SetBatch(kTRUE);   // keep Canvases from writing to screen

  std::vector <vector<TH1D*> >  numhits_p1; // array of histograms [wire][plane] package 1
  numhits_p1.resize(n);
  for (size_t k = 0; k<delta_t_p1.size(); k++)
    {
     numhits_p1[k].resize(5);
    }
  std::vector <vector<TH1D*> > numhits_p2; // array of histograms [wire][plane] package 2
  numhits_p2.resize(n);
  for (size_t k = 0; k<delta_t_p2.size(); k++)
    {
     numhits_p2[k].resize(5);
    }

  for (int ii=0; ii<n; ii++){
    for (int ij=0; ij<5; ij++){
     numhits_p1[ii][ij] = new TH1D(Form("numhits_p1[%d][%d]",ii,ij),Form("time wire %d - wire %d
      numhits_p2[ii][ij] = new TH1D(Form("numhits_p2[%d][%d]",ii,ij),Form("time wire %d - wire %
    }
  }

  Float_t time_diff_mean_plane1_pack1[275];
  Float_t time_diff_errors_plane1_pack1[275];
  Float_t time_diff_mean_plane2_pack1[275];
  Float_t time_diff_errors_plane2_pack1[275];
  Float_t time_diff_mean_plane3_pack1[275];
  Float_t time_diff_errors_plane3_pack1[275];
  Float_t time_diff_mean_plane4_pack1[275];
  Float_t time_diff_errors_plane4_pack1[275];
  Float_t time_diff_mean_plane1_pack2[275];
  Float_t time_diff_errors_plane1_pack2[275];
  Float_t time_diff_mean_plane2_pack2[275];
  Float_t time_diff_errors_plane2_pack2[275];
  Float_t time_diff_mean_plane3_pack2[275];
  Float_t time_diff_errors_plane3_pack2[275];
  Float_t time_diff_mean_plane4_pack2[275];
  Float_t time_diff_errors_plane4_pack2[275];
  Float_t wires[275];
  Float_t wire_errors[275];
  for (int i=0; i<275; i++){
    wires[i]=i;
    wire_errors[i]=0.01;
```

```
}

// since plane and package start counting from 1, not zero, size of arrays increased by one for

float time[11][5][3];
int wirehit[11][5][3];
int nhit[5][3];
int hitnum[11][5][3];

string folder = gSystem -> Getenv("QW_ROOTFILES");
ostringstream ss;
ss << folder << "/";
ss << stem.Data();
ss << run << suffix;
ss << ".root";
string file_name = ss.str();
cout << file_name << endl;
TFile* file = new TFile(file_name.c_str());
TString outputPrefix(Form("VDC_crosstalk_run%d_",run));
TString fileName(Form("Q2_tracking_%d.root",run));

QwEvent* fEvent = 0;
QwTrack* track = 0;
TTree* event_tree = (TTree*) file -> Get ("event_tree");

TBranch* event_branch = event_tree -> GetBranch("events");
if (event_branch) event_tree -> SetBranchStatus("events",1);
event_branch -> SetAddress(&fEvent);

 // How many events are in this rootfile?
Int_t nevents = event_tree -> GetEntries();
cout << "\nTotal events: " << nevents << "\n" << endl;

int start = (event_start == -1)? 0:event_start;
int end = (event_end == -1)? nevents:event_end;

// Loops over all events
for (int i = start;i < end;++i)
{
  if (i%10000 == 0)
    cout << "Events processed so far: " << i << "  Run:  " << run << endl; // Announces every 1

  if (event_branch) event_branch -> GetEntry(i);

  float time_113=0;
  float time_114=0;
  float time_111=0;
  float time_112=0;
  int plane=0;
  int package=0;
  int hitnumber=0;
```

```
    int wire=0;
    int drifttime=0;
    int nhits = fEvent -> GetNumberOfHits();

    for(int ij=0; ij<5; ij++){
      for (int ik=0; ik<3; ik++){
nhit[ij][ik]=0;
for (int ii=0; ii<11; ii++){
  wirehit[ii][ij][ik]=0;
  time[ii][ij][ik]=0;
}
      }
    }

    for (int j = 0; j < nhits; j++)
      {
const QwHit* hit = fEvent -> GetHit(j);
if(hit->GetRegion()==3){
wire = hit -> GetElement();
plane = hit -> GetPlane();
package = hit -> GetPackage();
drifttime = hit -> fTimeNs;
hitnumber = hit -> GetHitNumber();

if (nhit[plane][package] < 10 && wire < 275){
  nhit[plane][package]++;
  int ihit = nhit[plane][package];
  wirehit[ihit][plane][package] = wire;
  time[ihit][plane][package] = drifttime;
  hitnum[ihit][plane][package] = hitnumber;

  }

} // Region 3 hits only
      } // ends loop over nHits



    for(int ij=1; ij<5; ij++){
      for (int ik=1; ik<3; ik++){
int nmax = nhit[ij][ik];
for (int ii=1; ii<nmax-1; ii++){
    float hnumb = hitnum[ii][ij][ik];
    int wirenum = wirehit[ii][ij][ik];
if(ik==1)  numhits_p1[wirenum][ij] -> Fill(hnumb);
if(ik==2)  numhits_p2[wirenum][ij] -> Fill(hnumb);
}
      }
    }
  } // Ends for (int i = start;i < end;++i)
```

```
   for(int ii=1; ii<275; ii++){
     if(numhits_p1[ii][1]->GetEntries() >= 10){
      hit_number_mean_plane1_pack1[ii] = delta_t_p1[ii][1]->GetMean();
     }
     if(numhits_p1[ii][2]->GetEntries() >= 10){
       hit_number_mean_plane2_pack1[ii] = delta_t_p1[ii][2]->GetMean();
     }
     if(numhits_p1[ii][3]->GetEntries() >= 10){
       hit_number_mean_plane3_pack1[ii] = delta_t_p1[ii][3]->GetMean();
     }
     if(numhits_p1[ii][1]->GetEntries() >= 10){
      hit_number_mean_plane4_pack1[ii] = delta_t_p1[ii][4]->GetMean();
     }
     if(numhits_p2[ii][1]->GetEntries() >= 10){
       hit_number_mean_plane1_pack2[ii] = delta_t_p2[ii][1]->GetMean();
     }
     if(numhits_p2[ii][2]->GetEntries() >= 10){
       hit_number_mean_plane2_pack2[ii] = delta_t_p2[ii][2]->GetMean();
     }
     if(numhits_p2[ii][3]->GetEntries() >= 10){
       hit_number_mean_plane3_pack2[ii] = delta_t_p2[ii][3]->GetMean();
     }
     if(numhits_p2[ii][4]->GetEntries() >= 10){
       hit_number_mean_plane4_pack2[ii] = delta_t_p2[ii][4]->GetMean();
     }
  }
   //Mean Hit Number per wire 0 is first hit

cout << "Done" << endl;



  TCanvas* testing = new TCanvas("testing","testing",800,800);
  testing->Divide(4,2);
  testing->cd(1);
  numhits_p1[137][1]->Draw();
  testing->cd(2);
  numhits_p1[160][3]->Draw();
  testing->cd(3);
  numhits_p1[184][1]->Draw();
  testing->cd(4);
  numhits_p1[185][1]->Draw();
  testing->cd(5);
  numhits_p1[186][1]->Draw();
  testing->cd(6);
  numhits_p1[188][1]->Draw();
  testing->cd(7);
  numhits_p1[189][1]->Draw();
  testing->cd(8);
```

```
  numhits_p1[190][1]->Draw();
  testing -> SaveAs(outputPrefix+"hittesting.png");
  return;


  TCanvas* pack1 = new TCanvas("pack1","pack1",800,800);
  pack1->Divide(2,2);
  pack1->cd(1);
  TGraph *gr11 = new TGraph(275,wires,hit_number_mean_plane1_pack1);
  //  TGraphErrors *gr11 = new TGraphErrors(275,wires,time_diff_mean_plane1_pack1,
  //wire_errors,time_diff_errors_plane1_pack1);
  gr11->SetMarkerStyle(20);
  gr11->SetMarkerColor(2);
  gr11->GetXaxis()->SetTitle("Wire Number");
  gr11->GetYaxis()->SetTitle("Number of hits per wire");
  gr11->SetTitle("Package 1 Plane 1");
  gr11->Draw("AP");
  pack1->cd(2);
  TGraph *gr12 = new TGraph(275,wires,hit_number_mean_plane2_pack1);
  // TGraphErrors *gr12 = new TGraphErrors(275,wires,hit_number_mean_plane2_pack1,
//wire_errors,hit_number_errors_plane2_pack1);
  gr12->SetMarkerStyle(20);
  gr12->SetMarkerColor(2);
  gr12->GetXaxis()->SetTitle("Wire Number");
  gr12->GetYaxis()->SetTitle("Mean hit number per wire");
  gr12->SetTitle("Package 1 Plane 2");
  gr12->Draw("AP");
  pack1->cd(3);
  TGraph *gr13 = new TGraph(275,wires,hit_number_mean_plane3_pack1);
  //  TGraphErrors *gr13 = new TGraphErrors(275,wires,hit_number_mean_plane3_pack1,
  //wire_errors,hit_number_errors_plane3_pack1);
  gr13->SetMarkerStyle(20);
  gr13->SetMarkerColor(2);
  gr13->GetXaxis()->SetTitle("Wire Number");
  gr13->GetYaxis()->SetTitle("Mean hit number per wire");
  gr13->SetTitle("Package 1 Plane 3");
  gr13->Draw("AP");
  pack1->cd(4);
  TGraph *gr14 = new TGraph(275,wires,hit_number_mean_plane4_pack1);
  //  TGraphErrors *gr14 = new TGraphErrors(275,wires,hit_number_mean_plane4_pack1,
  //wire_errors,hit_number_errors_plane4_pack1);
  gr14->SetMarkerStyle(20);
  gr14->SetMarkerColor(2);
  gr14->GetXaxis()->SetTitle("Wire Number");
  gr14->GetYaxis()->SetTitle("Mean hit number per wire");
  gr14->SetTitle("Package 1 Plane 4");
  gr14->Draw("AP");

  pack1 -> SaveAs(outputPrefix+"Multpack1.png");
  pack1 -> SaveAs(outputPrefix+"Multpack1.C");
```

```
  TCanvas* pack2 = new TCanvas("pack2","pack2",800,800);
  pack2->Divide(2,2);
  pack2->cd(1);
  TGraph *gr21 = new TGraph(275,wires,hit_number_mean_plane1_pack2);
  //  TGraphErrors *gr21 = new TGraphErrors(275,wires,hit_number_mean_plane1_pack2,
  //wire_errors,hit_number_errors_plane1_pack2);
  gr21->SetMarkerStyle(21);
  gr21->SetMarkerColor(4);
  gr21->GetXaxis()->SetTitle("Wire Number");
  gr21->GetYaxis()->SetTitle("Mean hit number per wire");
  gr21->SetTitle("Package 2 Plane 1");
  gr21->Draw("AP");
  pack2->cd(2);
  TGraph *gr22 = new TGraph(275,wires,hit_number_mean_plane2_pack2);
  //  TGraphErrors *gr22 = new TGraphErrors(275,wires,hit_number_mean_plane2_pack2,
  //wire_errors,hit_number_errors_plane2_pack2);
  gr22->SetMarkerStyle(21);
  gr22->SetMarkerColor(4);
  gr22->GetXaxis()->SetTitle("Wire Number");
  gr22->GetYaxis()->SetTitle("Mean hit number per wire");
  gr22->SetTitle("Package 2 Plane 2");
  gr22->Draw("AP");
  pack2->cd(3);
  TGraph *gr23 = new TGraph(275,wires,hit_number_mean_plane3_pack2);
  //  TGraphErrors *gr23 = new TGraphErrors(275,wires,hit_number_mean_plane3_pack2,
  //wire_errors,hit_number_errors_plane3_pack2);
  gr23->SetMarkerStyle(21);
  gr23->SetMarkerColor(4);
  gr23->GetXaxis()->SetTitle("Wire Number");
  gr23->GetYaxis()->SetTitle("Mean hit number per wire");
  gr23->SetTitle("Package 2 Plane 3");
  gr23->Draw("AP");
  pack2->cd(4);
  TGraph *gr24 = new TGraph(275,wires,hit_number_mean_plane4_pack2);
  //  TGraphErrors *gr24 = new TGraphErrors(275,wires,hit_number_mean_plane4_pack2,
  //wire_errors,hit_number_errors_plane4_pack2);
  gr24->SetMarkerStyle(21);
  gr24->SetMarkerColor(4);
  gr24->GetXaxis()->SetTitle("Wire Number");
  gr24->GetYaxis()->SetTitle("Mean hit number per wire");
  gr24->SetTitle("Package 2 Plane 4");
  gr24->Draw("AP");

  pack2 -> SaveAs(outputPrefix+"Multpack2.png");
  pack2 -> SaveAs(outputPrefix+"Multpack2.C");


  return;

} // Ends void
```

Bibliography

[1] Armstrong, David S. **First result from $\mathbf{Q}_{weak}$**. EPJ Web of Conferences, 73 (2014) 07008 DOI: http://dx.doi.org/10.1051/epjconf/20147307008

[2] Leckey, John P. "Qweak-A Search For New Physics." AIP Conference Proceedings 1377.1 (2011): 380-382. Academic Search Complete.

[3] Charpak, et al. Nuclear Instructions and Methods, 126 (1975), p. 381-386

[4] LeClerc, Jennica N. "Particle Tracking Inefficiencies in the Qweak Experiment." William & Mary REU. 2016

[5] Perkins, Donald H. Introduction to High Energy Physics 2000: Cambridge University Press, N.Y.