Transit Time Uniformity of Two Commercial 5” Photomultiplier Tubes

Taylor Richards

Physics 492 Capstone Project Report submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Bachelor of Science

Lawrence Rees, Advisor

Department of Physics and Astronomy

Brigham Young University

August 2014

ABSTRACT

We investigated the uniformity of electron transit times across the full spatial extent of two 5” photomultiplier tubes, the Hamamatsu R1250 and the Adit B133D01. The photomultiplier tubes were translated across a localized incident light source while a portion of the incident light was simultaneously measured and recorded by a fast photodiode. Constant fraction discrimination was utilized to calculate electron transit times as the difference between the start times of the photodiode and photomultiplier traces. The Hamamatsu tube provided a uniform timing response that varied by no more than 1.7 ns. The Adit tube was extremely non-uniform with transit times that varied by as much as 57 ns, yet the symmetry of variation in transit times differed significantly when analyzed with two different algorithms to determine pulse timing. These results indicate that the Hamamatsu tube is superior to the Adit for experimental timing applications, and that the method of analysis significantly affects final timing results of the Adit tube.

Parts of this work were funded by NNSA Grant no. DE-FG52-10NA29655 and DHS Award no. 2010-DN-077-ARI039-02.

Contents

INTRODUCTION

Background and Purpose

Photomultiplier tubes, PMTs, are used in many physics experiments requiring time sensitive measurements of very small amounts of light. They convert incident photons into electrons via the photoelectric effect. This conversion occurs at the surface of the photocathode material contained within the PMT. These photoelectrons are accelerated towards the first positively charged dynode and are subsequently multiplied via a cascade of secondary emission. Upon passing through a series of dynodes and arriving at the final anode, the multiplied electrons produce a measurable signal as a sharp spike of electric current. The time it takes for the primary photoelectrons to travel and arrive at the final anode is known as the electron transit time. There is an inherent distribution in electron transit time across the spatial extent of the PMT. This distribution of transit times is attributed to the different photoelectron trajectories at the initial photocathode surface and subsequent dynode surfaces. My purpose is to investigate the uniformity of electron transit times across the full spatial extent of two commercial 5” PMTs, the Hamamatsu R1250 and the Adit B133D01.

Motivation

PMTs are used in conjunction with scintillators as an effective means to detect particle radiation. Many such experiments require precise timing information to calculate time-of-flight energies and energy spectra of the detected particles. If there is a substantial difference in electron transit times due to the location of the incident photon on the photocathode, the results of these sensitive timing measurements are made less accurate, and in extreme cases, invalid. Typically, a single large homogenous scintillator is used to illuminate the full photocathode surface simultaneously. This results in a single current spike, or signal, that is a superposition of responses from each spatial location on the PMT. Although each spatial location differs in transit time and response size, their sum typically results in a smooth Gaussian shaped response curve that minimizes the non-uniformity of the tube by essential averaging over all non-uniformities. The FWHM of this summed response is known as the transit time spread. Most manufacturers of PMTs optimized for timing applications carefully engineer their products to minimize this spread. Therefore, for a PMT operated in this manner, little importance is given to the difference in electron transit times from one location to the next. Yet, the BYU nuclear physics group currently uses a neutron detector that employs optically separated rectangular slabs of plastic scintillator. These rectangular scintillator slabs do not uniformly illuminate the full surface of the PMT simultaneously. In other words, the light created from a radiation event in a specific slab of the detector, will only illuminate a small rectangular portion of the PMT surface. If the transit time differs significantly in that region, then the sensitive timing measurements are made less accurate. Consequently, by characterizing the uniformity of transit times across the full spatial extent of the tube, we can effectively minimize timing uncertainties and improve experimental results. In addition, the experimental procedure we used to characterize the non-uniformity of transit times can also be applied to any research that utilizes a PMT for timing information. For example, the Positron Emission Tomography (PET) scan used in medical physics applications. A more accurate localization of the electron positron annihilation event is made possible when the transit time non-uniformities are known and minimized [2]. A more accurate localization results in a higher resolution image that can aid physicians in making better diagnoses.

Context

There is current research being conducted concerning the transit time characteristics of typical commercial PMTs. Lung and Arisaka at UCLA investigated the timing characteristics of a 3” Hamamatsu R11410-10 tube operated in a vacuum cryostat for Dark Matter detection experiments [3]. They found that the transit time was highly position dependent and that large deviations on the order of 5 ns did occur near the edges of the PMT surface. They also found symmetry along the X and Y axis corresponding to the location and design of the first dynode. More recently, Xu Wang has investigated many 1” and 2” PMTs for applications in cosmic ray astronomy.[4] His initial results confirm those of Lung and Arisaka. In addition, he found that a location with a slower relative transit time, like an edge point, also had greater transit time spread among the samples taken at that location.() Both of the above cited experiments, and the subject matter of this report, have similar experimental setups and procedures that will be discussed in further detail in the Methods and Materials section. Importantly, no large diameter PMT (5”) used in nuclear physics applications has yet been investigated. Additionally, by characterizing and comparing the timing properties of two very different PMTs, Adit box-and-grid and Hamamatsu linear focused, this research will provide useful information for physicists who must select which type of large PMT is best suited for their particular experiment. Finally, in conjunction with the timing information, we will also detail the uniformity of peak height and area response across the full spatial extent of the PMT. These metrics are often employed in nuclear physics experiments, and no published research has been done to quantify the uniformity across a large PMT at typical operating parameters.

METHODS AND MATERIALS

Materials

An in-depth review of the characteristics of each PMT is necessary to justify key differences in experimental design. Both the Hamamatsu R1250 and the Adit B133D01 are 5” diameter, head-on type photomultiplier tubes that use bialkali photocathodes with a minimum useful diameter of 120 mm. The Hamamatsu employs a 14-stage linear focused dynode structure with a maximum operating voltage of 2000 V. The Adit has a 10-stage box-and-grid dynode structure with a maximum operating voltage of 1500 V. We used an Ortec Model 266 short base with the Adit, and a custom base for the Hamamatsu. Although the Hamamatsu base was custom, the voltage distribution ratio matched the distribution given by the original data sheet. On average, the Hamamatsu had a larger peak voltage response than did the Adit. To compensate for this effect, and to eliminate saturation in the data acquisition oscilloscope, we attenuated the input laser pulse with two additional neutral density filters when taking data on the Hamamatsu PMT. This method of attenuation was done using a very practical algorithm. We used sufficient neutral density (ND) filters until the PMT response no longer saturated the oscilloscope, and the addition of any more ND filters would completely eliminate all signal response from the PMT. In other words, we attenuated the incident light source until we reached the minimal amount of incident photons required to produce a measureable current response from the given PMT at its maximum operating voltage. This method of attenuation also ensured that the PMT response remained linear. All other components and procedures of the experiment were held constant for each tube. As our pulsed light source, we used a Ti Sapphire laser frequency doubled to a wavelength of 400 ±10 nm. This mode of operation was ideal for our experiment because both tubes had maximum quantum efficiency and maximum absolute sensitivity near this wavelength. The laser operated in mode lock at 550 ±100 mW and a pulse length of 100 ps. At incidence on the PMT surface, it had 2 mm diameter spot size.

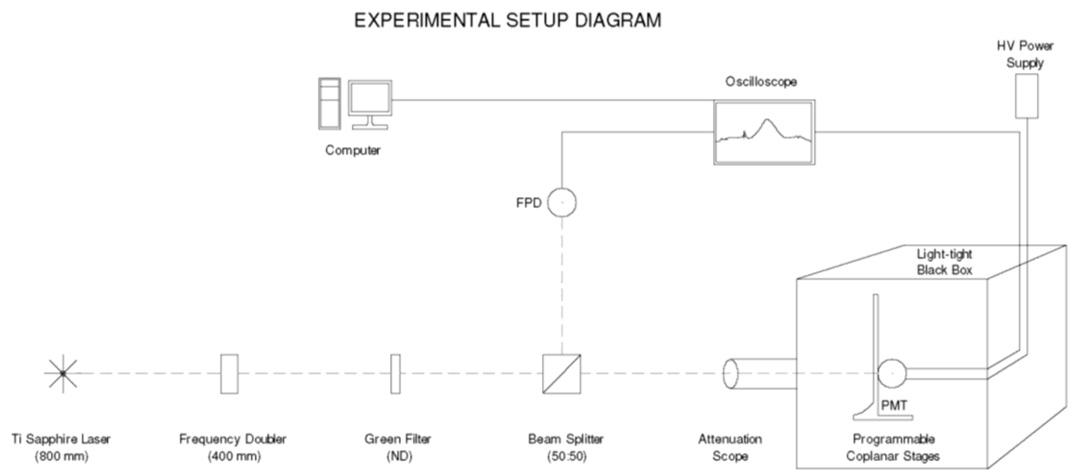


Fig. 1. A simplified diagram of the experimental design

<u>Experimental Set-Up</u>

Figure 1 concisely illustrates the basic experimental design. The Ti Sapphire laser comes from an adjacent lab by passing through a periscope and iris to reach the first component, the Quantum Technology Model D02-110R2 frequency doubler. The laser then passes through a green filter to eliminate any remnants of the original 800 nm wavelength pulse. The ThorLabs CM1-BS013 beam splitter then sends half the incident beam through a light-tight black tube to the fast photodiode, and the other half of the beam into the attenuation scope. The ThorLabs DET210 fast photodiode (FPD) has an attached NE20A filter to attenuate the incident beam to within the FPD linearity limit of 1 mW. The signal from the FPD is recorded in the Tektronix DPO7104 oscilloscope on Ch. 3. The other half of the beam passes through the attenuation scope, which is composed of three neutral density filters (NE60A, NE05A, ND40A). The attenuated beam enters the light-tight black box where it illuminates the PMT surface at normal incidence. Two programmable motorized stages translate the PMT coplanar with the front surface of the PMT and normal to the incident light pulse. The signal from the PMT is recorded in the oscilloscope on Ch. 2. Typical reflective mirrors are also utilized throughout the set-up to ensure a clear path and normal incidence at the correct location on the PMT surface.

Experimental Procedure

After placing the PMT and its base in the specially manufactured metal clamp on the motorized stages, the laser was turned on and set to operate in mode-lock at 800 nm. A measurement of the laser power was performed to ensure it was operating at approximately 550 mW. Laser power was also measured upon completion of the data acquisition session to ensure it was still functioning in the appropriate range after several hours of continuous operation. After creating a clear path for the laser beam through all parts of the experiment, a level was used to double-check for normal incidence at both the PMT and FPD surfaces. Any adjustments in mirror positioning were made as needed. Both the FPD and PMT glass surfaces were properly cleaned before every experimental run. We then used a LabView VI to operate the motorized stages and move the PMT into its initial position. It was important to correlate the physical location of the incident light with the LabView program's virtual coordinates. This ensured that our program's recorded location matched the actual location of incident light on the PMT surface. We then sealed the light-tight black box and turned off all ambient lights to ensure optimal PMT operating conditions. After we were satisfied with the light seal, we turned on the FPD, oscilloscope, and high voltage power supply. We waited a few minutes to ensure that the power supply had properly warmed up and reached steady-state operation. We operated the Hamamatsu tube at 2000 V and the Adit at 1500 V, both their respective maximum operating voltages. We choose to perform the experiment at the maximum operating voltage for two key reasons: the BYU nuclear physics group operates its PMTs in this manner for typical experiments, and the best timing performance is obtained by operating the tube at the maximum voltage permitted [1]. We then set the oscilloscope parameters: 50 Ω termination for both channels (set to match the base and cable impedances), with a sampling rate of 10 GS/s or 100 ps/pt (highest rate achievable on the oscilloscope), and 3.5 mV rising edge manual trigger on the FPD channel. The vertical scale was only set after a quick scan was performed over the entire surface of the PMT. During this scan, we checked for saturation in the oscilloscope and added more light attenuation as needed (practical attenuation algorithm). For both data acquisition sessions, the PMT channel 2 was set to 200 mV/div and the FPD channel 3 was set to 7 mV/div. At each trigger event, both the PMT and FPD wave traces were recorded on the oscilloscope and saved to the computer as csv files for later analysis. Before the final full scan was performed, we took a preliminary scan of 1000 waveforms at the center location on the PMT surface. These data were later analyzed to ensure that the laser had indeed been operating in a consistent manner. This same scan was repeated after the data acquisition session as well. Finally, we programmed the LabVIEW VI to perform a raster scan on a square grid across the full face of the PMT surface. We divided the step size in 5 mm increments in both the X and Y directions for a total of 729 spatial locations. Although increased spatial resolution was possible with our experimental design, it was time prohibitive as the regular scan already lasted more than 3 hours. The LabVIEW program saved 10 waveform pairs (PMT and FPD) at every location, along with the spatial coordinates and timescale from the oscilloscope. Once again, we choose to save only 10 waveform pairs per spatial location due to time limitations on laser operation and access to the laser lab.

Data Analysis

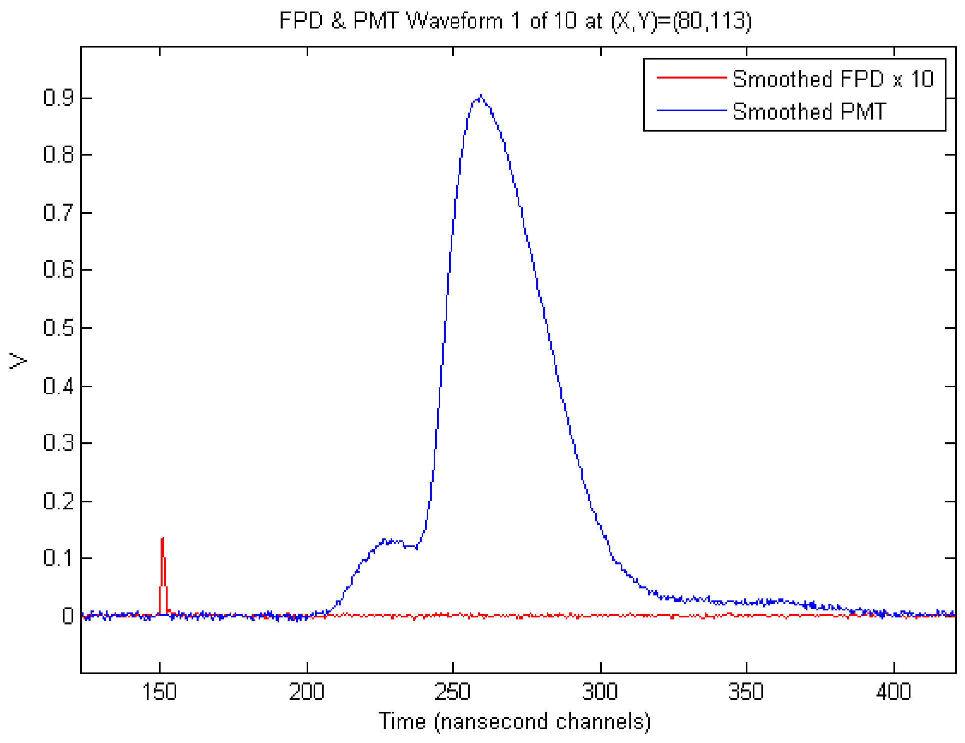


Fig. 2. Typical Adit PMT waveform example. Note the small peak before the largest Gaussian shaped peak.

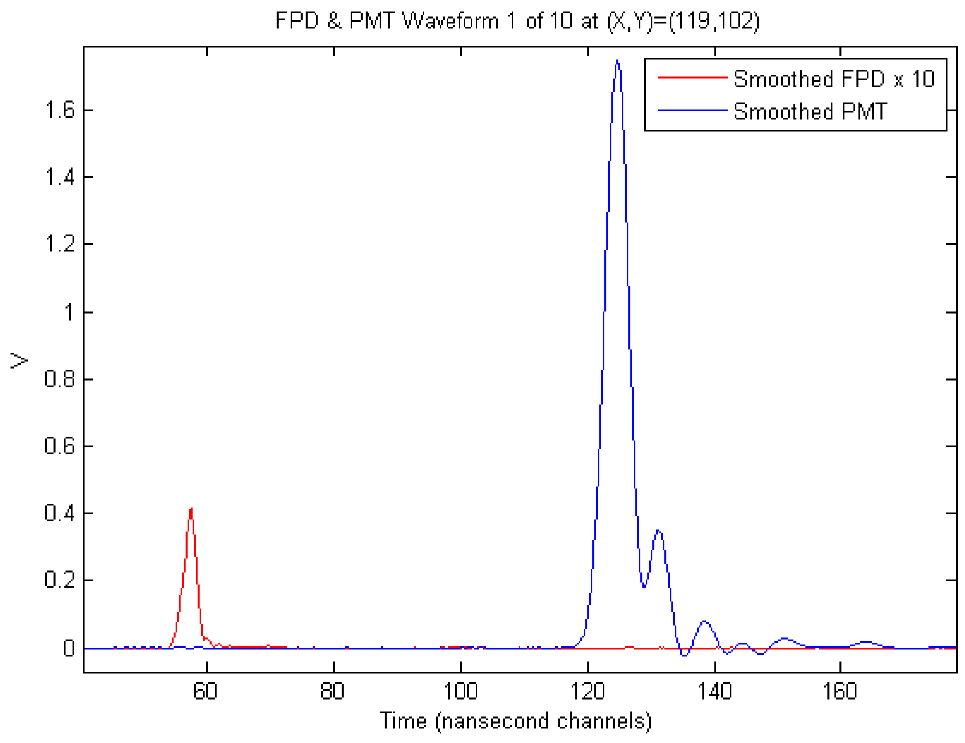


Fig. 3. Typical Hamamatsu waveform example. Note the steep rising slope and afterpulsing.

Data analysis began by using MATLAB to plot a few typical example events (both PMT and FPD waveforms) from many different locations on the PMT. Both waveforms exhibited some voltage noise and a small constant offset. The pulses were smoothed using a moving average filter of 5 adjacent data points, and a constant offset was calculated by averaging over 1000 data points of noise (Figs. 2 and 3). Visual inspection of these waveforms helped identify their common pulse characteristics. The common pulse characteristics then dictated which analysis algorithms would produce valid results. Figure 2 demonstrates that some Adit PMT responses had a small peak about 20 ns earlier in time before the main Gaussian peak. Figure 3 demonstrates that Hamamatsu response waveforms don't have any early peaks, but rather significant afterpulsing. These key pulse differences justify the use of different analysis algorithms for the two different data sets. The variables we desired to measure from the waveforms were the following: start and stop channel of the PMT and FPD waveforms, transit time (difference between start of FPD trace and the start of the PMT trace), maximum peak height, and area response. To pick-off the start time of any pulse, we used a zero-crossing algorithm. A zero-crossing algorithm first attenuates the response pulse to a constant fraction of its original amplitude. It then creates another altered response pulse by inverting and delaying the original signal. Summing these two altered waveforms results in a new waveform which crosses zero at the point in time when the leading edge of the original response pulse reaches the preset constant fraction of pulse amplitude. This algorithm for pulse timing provides consistent results across a large input dynamic range. Utilizing this algorithm required finding an ideal offset or delay time that was greater than all peak rise times but less than the sum of rise and fall times. This was accomplished by creating a histogram of all rise, fall, and sum times. After visual inspection of these histograms, we selected an offset time for each data set that matched the aforementioned criterion. A proper attenuation level was then selected which minimized the variance of start times at a given location. Interestingly, because the Adit data consisted of waveforms with two superimposed peaks, a different attenuation level produced very different results. Thus, we analyzed the Adit with one low attenuation level (10%), and one high attenuation level (90%). The Hamamatsu did not vary with respect to changes in the attenuation level, so it was only analyzed with 10% attenuation. The area response of the PMT pulse was calculated by summing the region beneath the peak from the calculated start time to the calculated stop time on the backside of the pulse. The stop time was calculated using a trailing-edge constant fraction technique set to 10% of the maximum peak height. We did try using a lower trailing-edge constant fraction, 5%, to include all afterpulses from the Hamamatsu data. This did not significantly alter the final calculated areas because all afterpulse areas were directly proportional to the original pulse area. Finally, because not all saved data locations corresponded to a physical location on the surface of the PMT (square grid scan of circular PMT), we choose a minimum voltage threshold that would define which PMT waveforms constituted a real response. This was determined by creating a histogram of the maximum voltage noise from every event. The threshold was then set above the maximum recorded noise from any one event. Near the physical edges of the PMT surface we found that only some of the 10 PMT response waveforms exceeded the minimum threshold, while others from the same location failed to meet the criterion and were therefore disregarded as noise. This variance in PMT response at a given location was due to the fact that the intensity of incident light also varied in time (as measured by the FPD peak response). Therefore, we also required that at least 7 of 10 events meet the response criterion in order for a given spatial location to be saved, averaged, and included in the analysis. We choose to require at least 7 of 10 events because all measurement variables are represented as averages at a given location. Averaging fewer than 7 events could produce skewed results given a few outliers. Additionally, by requiring 7 events, the size and geometry of our saved locations essentially matched the physical parameters of the PMT being scanned (Fig. 4). After we were satisfied that the threshold requirements were valid, we threw out all spatial locations that were not contained within the manufacturer's minimum useful diameter (120 mm). The validity of our data analysis method is brought into question by the results of the saved data locations from the Hamamatsu scan (Fig. 4). After enforcing the minimum threshold requirement on the Hamamatsu data, some spatial locations remained despite not correlating to physical locations on the surface of the PMT. Increasing the minimum threshold requirement did not eliminate these erroneous data points. Nonetheless, these same data points were not contained within the minimum useful diameter, and were therefore not included in the final data analysis. As a final check of all analysis parameters, we analyzed the before and after scans performed at the PMT center location. Because we saved 1000 waveforms at that one location, we had better counting statistics and smooth Gaussian distributions to help determine correct analysis parameters. Importantly, we checked that the transit time spread (TTS) was minimized and that the distribution of relative area responses (ratio of PMT and FPD area) was also narrow. After we were satisfied that the correct parameters had been set, we performed an analysis of all spatial points on the PMT surface. For each of the 10 waveform pairs at every location, we calculated and saved the start channel, stop channel, integrated area, and max peak voltage. We then calculated the averages and variances of these same variables at every spatial location.

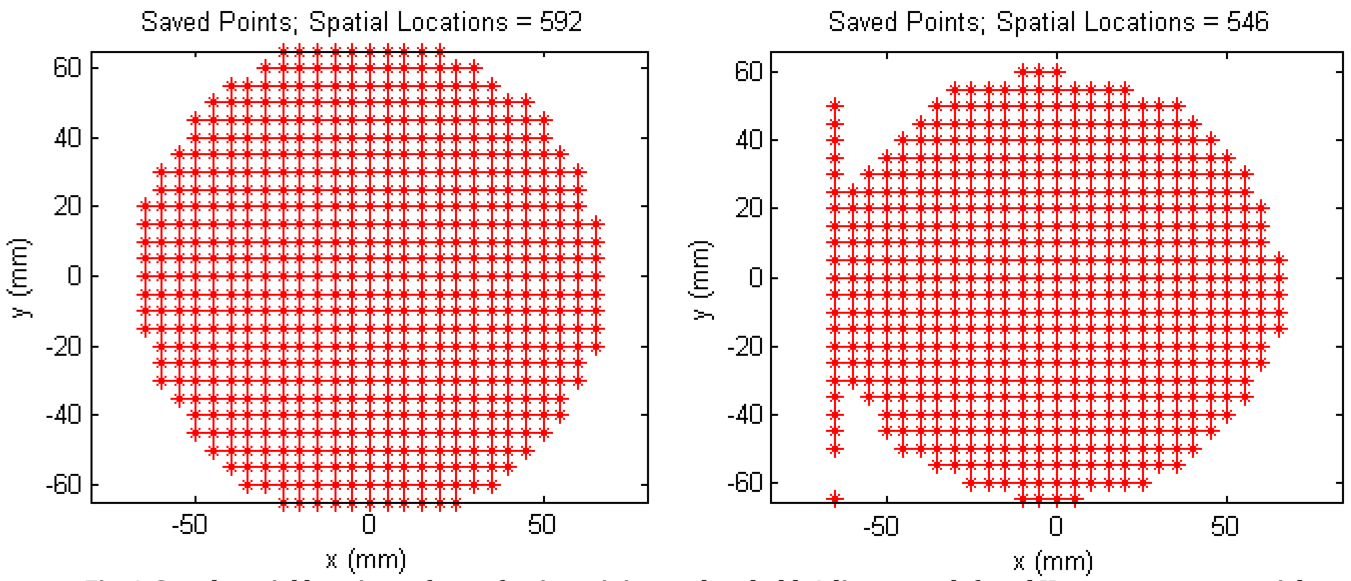


Fig. 4. Saved spatial locations after enforcing minimum threshold. Adit scan on left and Hamamatsu scan on right

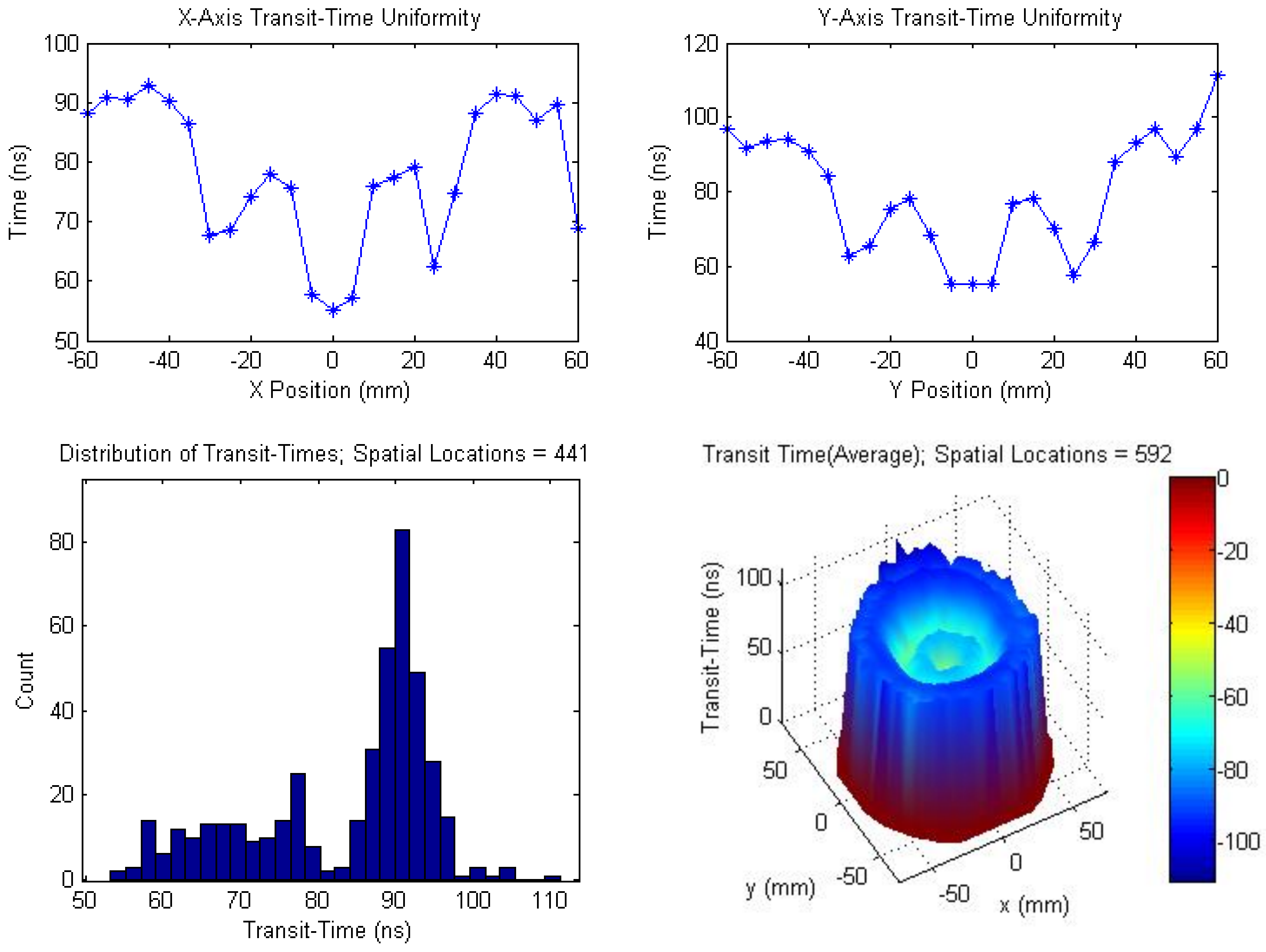


Fig. 5. Timing Characteristics of the Adit B133D01 tube with analysis attenuation level set to 10%

The Adit timing characteristics were non-uniform when analyzed with an attenuation level of 10%. Transit times varied by as much as 57.9 ns from one location to another. The distribution of transit times shows that there were two distinct groupings of transit times. These groups also correlated to separate physical regions on the tube surface (3D surface plot). Upon closer inspection of the X and Y axis uniformity plots, the fastest transit times occurred at the origin, and an azimuthally symmetric region at a radius of about 30 mm. This effect is due to the small response peak that occurs before the main Gaussian peak for points in this region (Fig. 2).

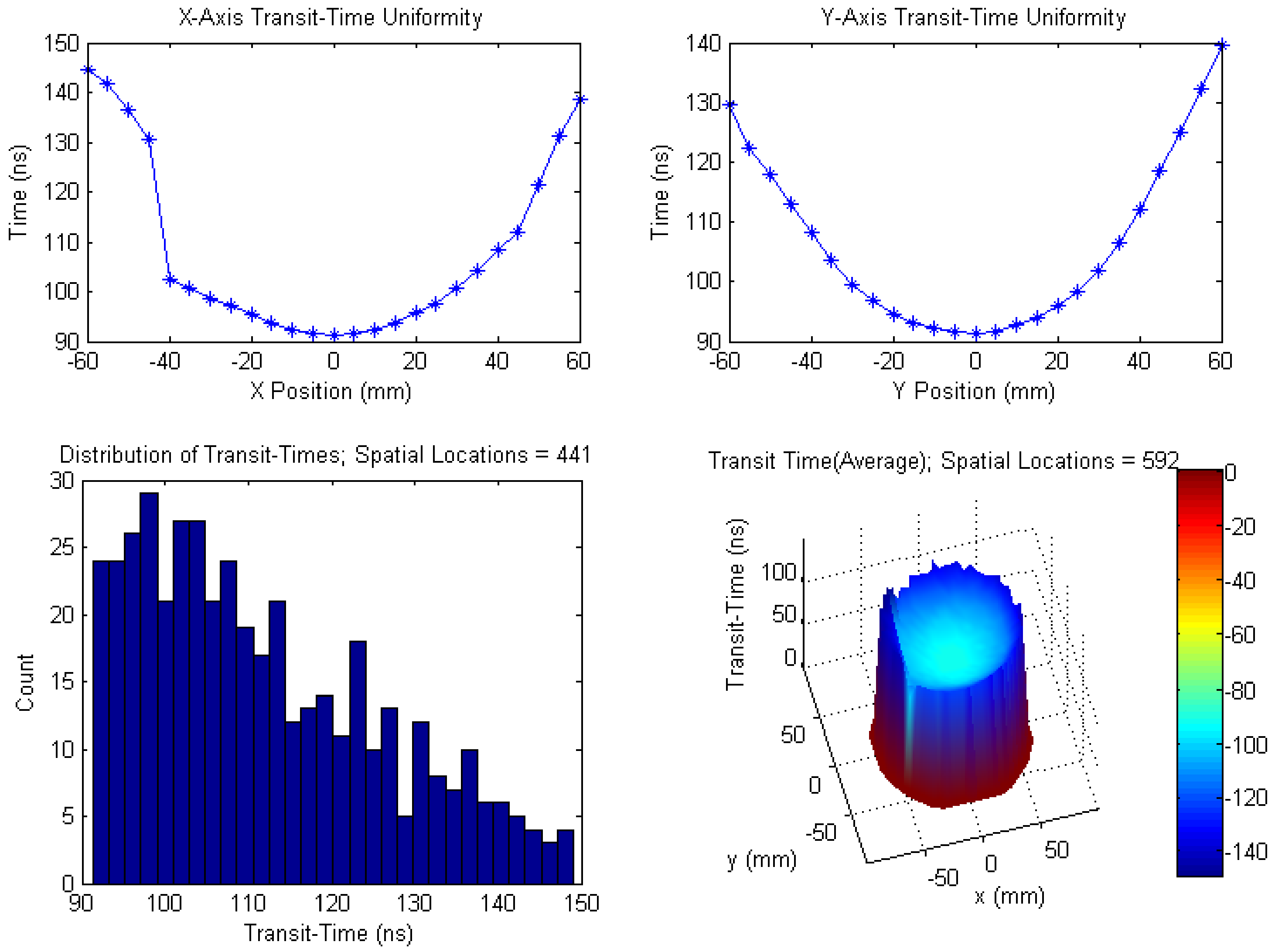
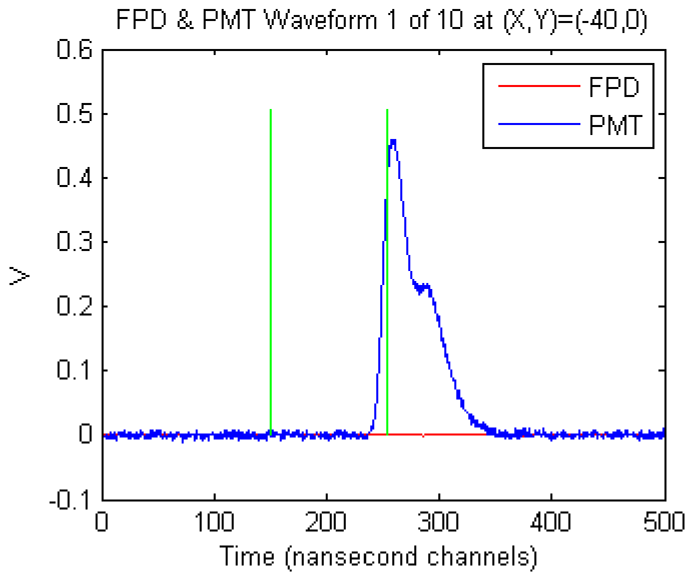


Fig. 6. Timing Characteristics of Adit tube with attenuation level set to 90%

The Adit timing characteristics were still non-uniform when analyzed with an attenuation level of 90%, yet the non-uniformities were smoother and followed a consistent pattern. Transit times still varied by as much as 57.7 ns from one location to another, but the distribution was more homogenous and uniform. The X and Y axis plots, along with the 3D surface plot, show that the transit time was fastest at the origin and slowest at the edges. The smooth rise from origin to edge roughly fits a paraboloid. The X axis plot also shows a large discontinuity 40 mm to the left of the origin. The 3D surface plot confirms that this continuity occurs at all points along the vertical line X = -40. This discontinuity is the result of a subset of PMT response pulses (Fig. 7).



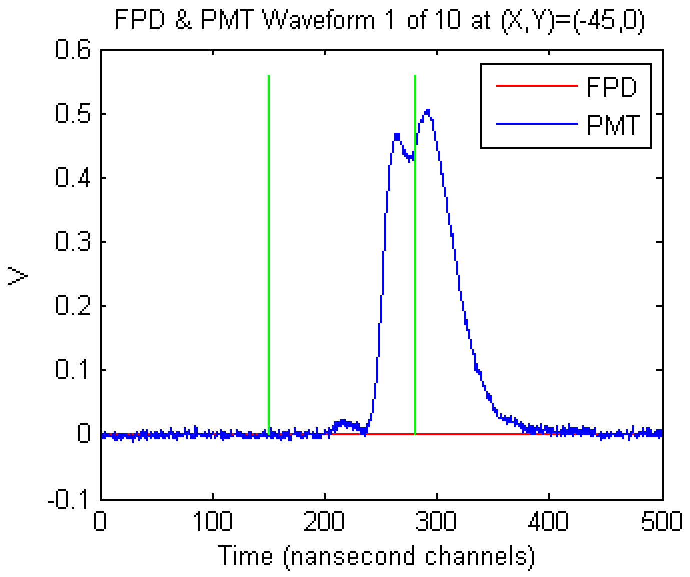


Fig. 7. Example pulses of Adit scan. First green line marks start time of FPD. Second green line marks start time of PMT.

Figure 7 demonstrates that the timing discontinuity is the result of two equally sized superimposed response pulses that occur at locations in this region. In the example on the left, $X = -40$, the pulse start time is calculated using the first superimposed pulse. In the example on the right, $X = -45$, the pulse start time is calculated using the second superimposed pulse. This dual peak effect results in a time shift of approximately 28 ns as shown in Figs. 6 and 7. Although the timing discontinuity is the direct result of our method for calculating the pulse start time, we are confident that the dual peak characteristic for points in this region is a valid response that would affect final results using any analysis method. Comparison of the Adit timing characteristics when analyzed with different attenuation levels demonstrates that Adit data from any experiment will provide different results depending on the method of analysis.

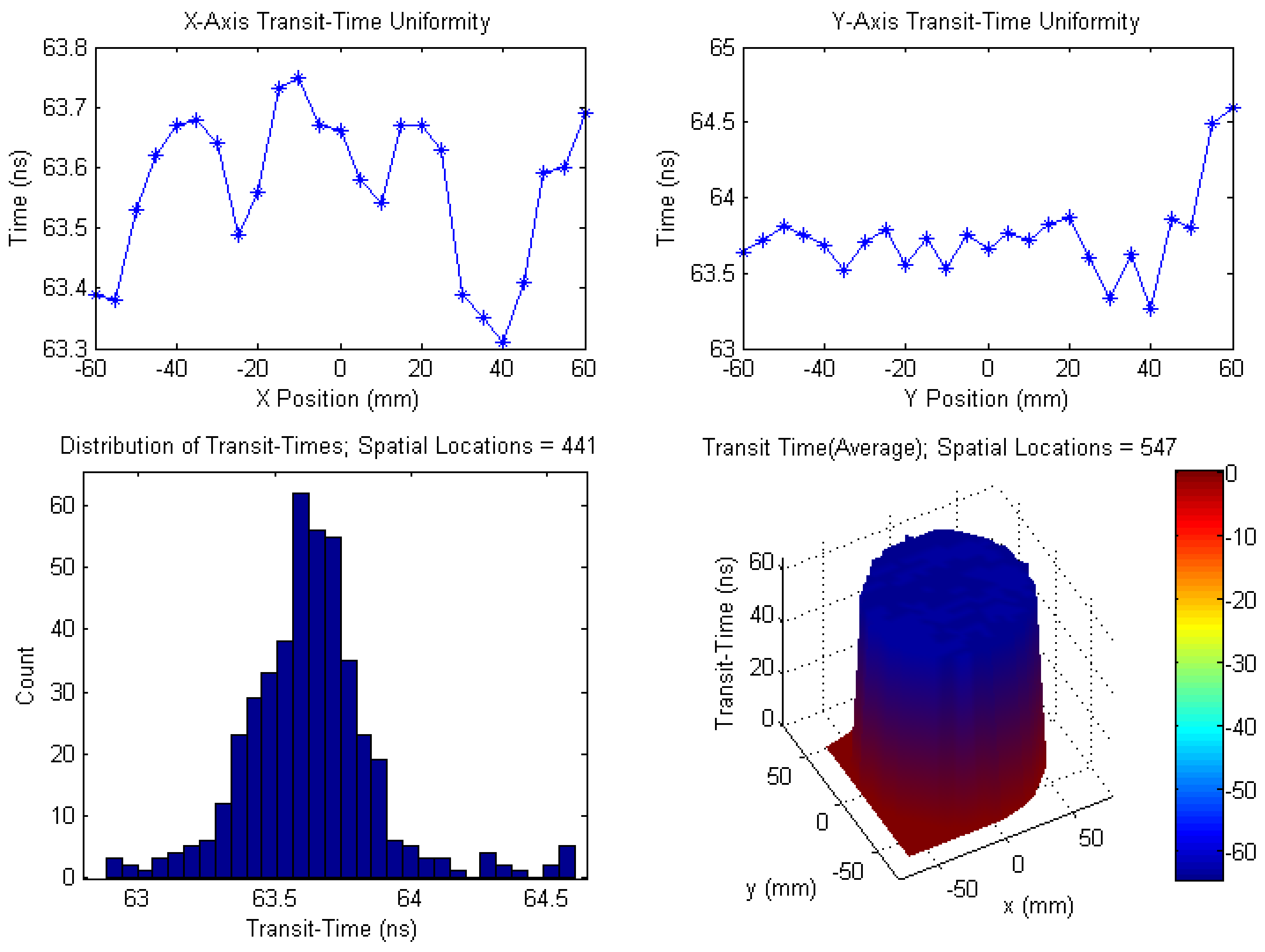


Fig. 8. Hamamatsu timing characteristics with attenuation level set to 10%

The Hamamatsu provided a very uniform timing response across the full spatial extent of the tube. The transit time varied by no more than 1.7 ns from one location to another. The Gaussian distribution of transit times had a FWHM of only 0.4 ns. Although both the X and Y axis plots show no clear pattern, the timing jitter is so minimal that it would go unnoticed in most experimental applications. In our nuclear physics experiments, we utilize a digitizer whose maximum sampling rate of 250 MHz results in 4 ns channels. Hence, the limiting factor for timing resolution in this case would be the digitizer and not the Hamamatsu PMT. The Hamamatsu is clearly far superior to the Adit for timing applications.

Although we analyzed and calculated the uniformity of area and peak response across the full spatial extent of each the Adit and the Hamamatsu tube, we are unsure of the validity of the results for two key reasons. First, we repeated a full scan of the Hamamatsu tube after rotating it 90˚ from the initial scan orientation. The timing results correctly followed the rotation, while all area and peak response results did not rotate with the tube. In other words, it appeared that all symmetry in the area and peak responses were a result of our left-to-right raster scanning method and not the tube symmetry itself. Second, we also calculated the area and peak responses of the accompanying FPD trigger pulses. The laser power randomly varied in time as measured by the FPD peak and area responses (Fig. 9). Because the PMT response is linearly proportional to the amount of incident light, the ratio of PMT area to FPD area should be correlated, and should provide a better measurement of the relative PMT response. Yet, the relative area and peak responses did not have less variance nor less overall spread than the normal area and peak responses. In other words, normalizing the PMT peak and area response by dividing by the FPD peak and area response only served to introduce more uncertainty and randomness in the relative results.

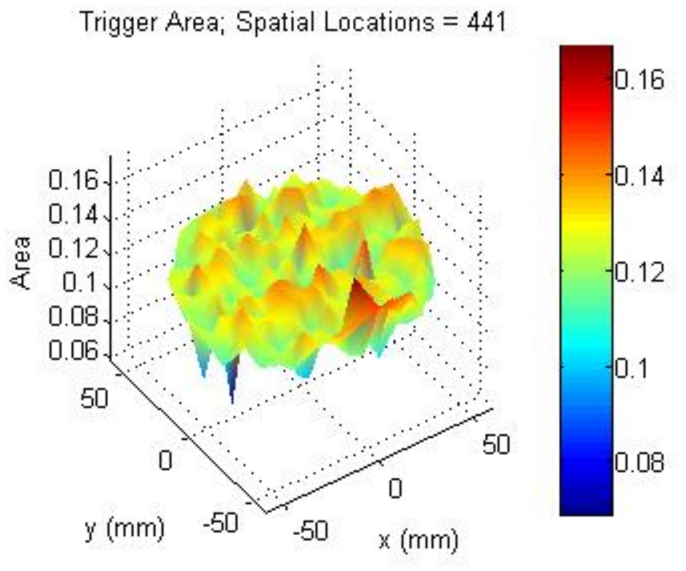


Fig. 9. Surface plot of FPD trigger pulse areas from Adit scan.

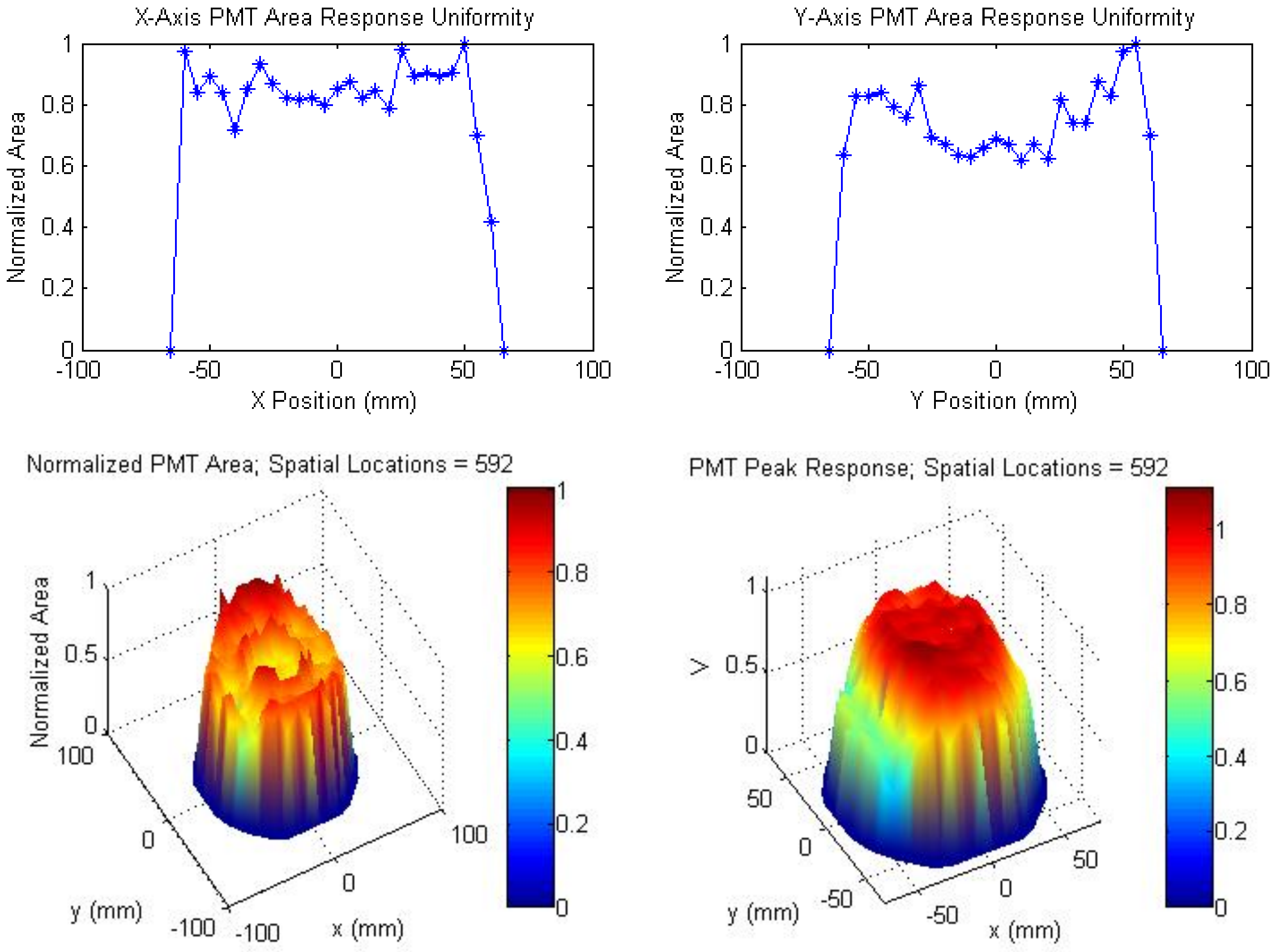


Fig. 10. Adit peak and area response characteristics with attenuation level set to 10%

The Adit was fairly uniform with respect to area response and maximum peak height for both 10% and 90% attenuation levels. Within the minimum useful diameter, the area response only varied by a factor of 3.1 for 10% attenuation, and by a factor of 4.7 for 90% attenuation. The peak height varied by a factor of 3.8 for both attenuation levels. Interestingly, the same pre-bump waveform characteristic of the Adit PMT also affected the area calculation at 10% attenuation. An azimuthally symmetric region at a radius of 30 mm had a larger area response than did the surrounding regions due to this effect. Furthermore, the largest area responses were found at the edges of the PMT where two superimposed pre-bumps occurred in the PMT response instead of the usual one. These pre-bumps increase the total area as if there had been two simultaneous responses occurring at the same point in time.

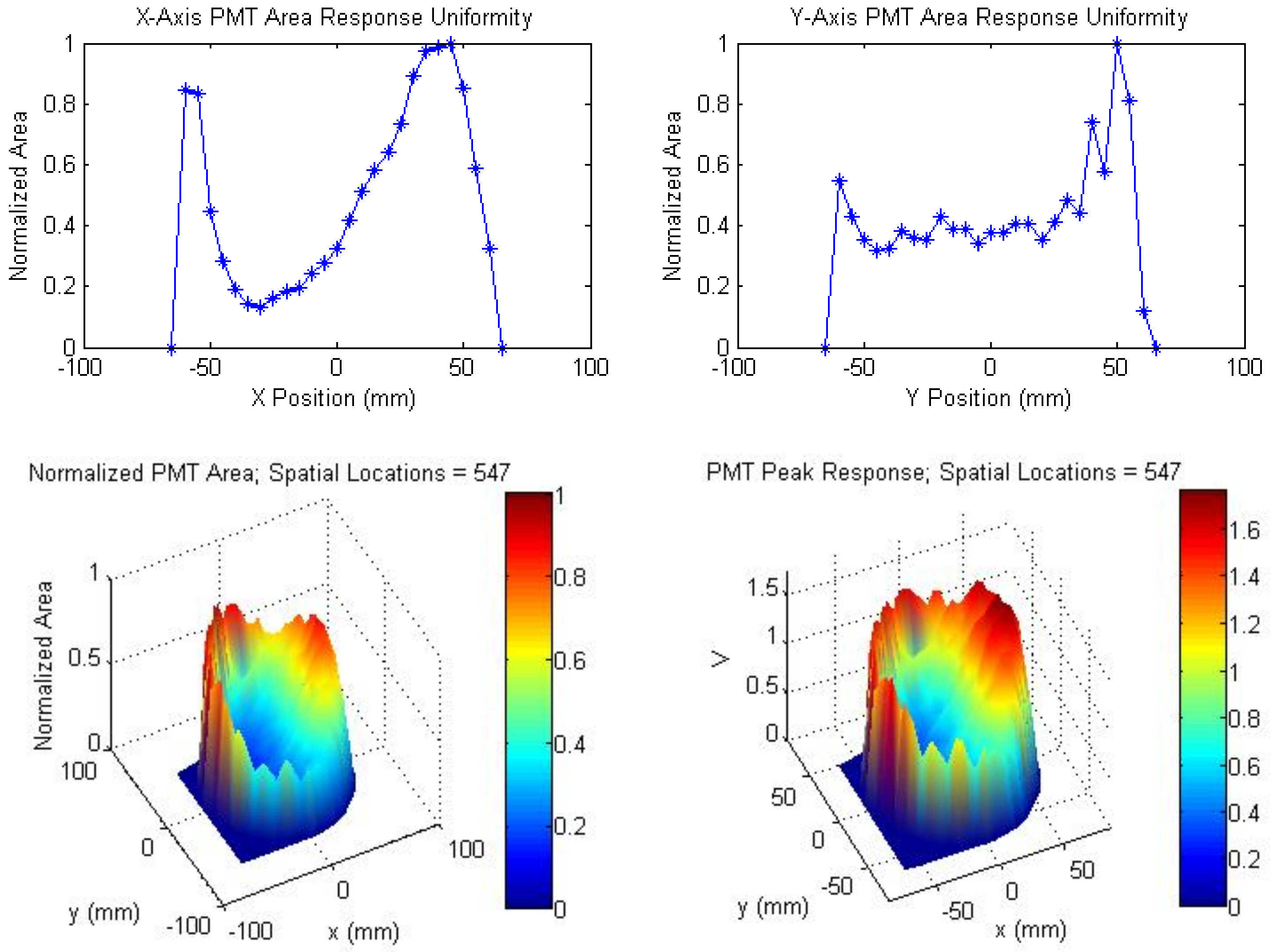


Fig. 11. Hamamatsu peak and area response characteristics with attenuation level at 10%

The Hamamatsu was not uniform with respect to maximum peak height and area response. Within the minimum useful diameter, the area response varied by a factor of 11.1 and the max peak response varied by a factor of 10.3. In other words, the peak and area responses varied by a full order of magnitude, a significant factor. There was apparent symmetry in the X axis area response; a small area response on the left rose linearly to a large area response on the right. Yet, with the tube rotated 90˚ clockwise and the scan repeated, this apparent left-to-right symmetry remained while all timing metrics correctly followed the rotation. This suggests that the apparent symmetry is only a result of our scanning methods (left-to-right). More research must be done to understand this effect. At this time, it appears that the Adit is far superior to the Hamamatsu for peak and area applications.

CONCLUSION

The investigation of the uniformity of transit times across the full spatial extent of two commercial 5” PMTs, the Hamamatsu R1250 and the Adit B133D01, revealed there are important non-uniformities and differences between the tubes that would dictate which tube is best suited for different experimental applications. For precise timing measurements, like time-of-flight energy spectra, the Hamamatsu tube provides the most uniform timing response. The transit time varied by no more than 1.7 ns from one location to another and the Gaussian distribution of transit times had a FWHM of only 0.4 ns. In other words, the limiting factor for timing resolution in a typical nuclear physics experiment would be a slow scintillator or the maximum sampling rate of the digitizer, and not the Hamamatsu tube itself. Yet, this same Hamamatsu tube simultaneously offered a very non-uniform peak height and area response. Both the area and peak response varied by more than a factor of 10 from one location to another on the surface of the PMT. Even though these peak and area results have important implications for experiments that utilize such metrics to make inferences about detected particles and their spectra, we are not wholly confident in the validity of these results. In summary, the Hamamatsu is great for timing applications, but is seemingly poor for light collection uniformity and area response applications. On the other hand, the Adit tube provided a fairly consistent peak height and area response across the full spatial extent of the tube. Within the minimum useful diameter, the area response only varied by a factor of 3.1, and the peak height varied by a factor of 3.6. Meaning, that the location of incident light could still effect the outcome of the peak height and area response, but its effect would not introduce as much uncertainty in the experimental results as did the Hamamatsu. Yet, the validity of the Adit peak and area responses are questionable. Even though the Adit fared well with respect to area response uniformity, it was extremely non-uniform with respect to transit time characteristics. Within the minimum useful diameter, transit times varied by as much as 57 ns from one location to another (when analyzed with both high and low attenuation levels). Although the transit times varied with radial and azimuthal symmetry, the symmetries were significantly different when analyzed with low versus high attenuation levels. These differences were a result of the small peak about 20 ns earlier in time before the main Gaussian response curve. Because of the radial symmetry of the regions where the pre-peaks occur, it appears that these pre-peaks are valid responses from the PMT rather than merely some artifact of our experimental design. They seem to be real responses that are superimposed with the large main Gaussian response at that location. It is possible that a fraction of the incident light is scattered or reflected towards another faster location on the photocathode surface, resulting in the superposition of the two nearly simultaneous responses. In summary, the Adit is best suited for peak height and area response experimental applications. Yet, neither the Adit nor the Hamamatsu can be considered the best option for all experimental applications. Future research could investigate: other commercial tubes that are designed as more of a compromise between timing and area response, differences between two identical tubes due to the manufacture process, different operating voltages, and isolation of the Adit pre-bump response characteristic.

REFERENCES

[1] G. F. Knoll, Radiation Detection and Measurement, 3rd ed. New York, 2000).

[2] Tomasz Szczesniak *et al.*, "Fast Photomultipliers for TOF PET," IEEE Trans.Nucl.Sci. 56 (1), 173-181 (2009).

[3] K. Lung *et al.*, “Characterization of Hamamatsu R11410-10 3-Inch Photomultiplier Tube for Liquid Xenon Dark Matter Direct Detection Experiments,” arXiv:1202.2628v2 [physics.ins-det] 23 Aug 2012

[4] Xu Wang and Ye Tian, “A study of head-on type photomultiplier tubes,” Shandong University Unpublished Apr 21 2014, URL accessed July 19 2014 no longer available.

APPENDIX A

MATLAB ANALYSIS CODE

```
%PMT Timing and Effeciency Written by Kent T, Trevor J and Taylor R

clear; clc; close all;

global fpdthreshold pmtthreshold fpdoffset fpdcfd pmtoffset
global pmtcfd req datapoints minradius xcenter ycenter smoothingfactor

%Settings for July12 Adit Data Analyzed at CFD = 0.9
fpdthreshold = .0035;
pmtthreshold = 0.03;
fpdoffset = 15;
fpdcfd = .1;
pmtoffset = 100;
pmtcfd =.9;
req = 7;
datapoints = 5000;
minradius = 60;
xcenter = 80;
ycenter = 83;
smoothingfactor = 5;

% %Settings for July12 Adit Data Analyzed at CFD = 0.1
% fpdthreshold = .0035;
% pmtthreshold = 0.03;
% fpdoffset = 15;
% fpdcfd = .1;
% pmtoffset = 570;
% pmtcfd =.1;
% req = 7;
% datapoints = 5000;
% minradius = 60;
% xcenter = 80;
% ycenter = 83;
% smoothingfactor = 5;


fprintf('What do you want to do?\n\n');
fprintf('  0 Quit\n');
fprintf('  1 Plot individual events (Fast Photodiode & PMT Traces)\n');
fprintf('  2 Write Summary file of Timing and Effeciency Information\n');
fprintf('  3 Analyze Sumfile (Zero Supressed)\n');
fprintf('  4 Plot Rise and Fall Times Distribution\n');
fprintf('  5 Create Peak Timing Information ("Pre-Bump")\n');
fprintf('  6 Create Response Histogram & Find Maximum Response\n');
fprintf('  7 Analyze SumFile\n');
fprintf('  8 Before and After Pre-Scan Analysis\n');
fprintf('  9 Analyze Peak Timing Info\n');

fprintf('\n');
choice=' ';
```

```
while choice==' '
    choice=input('Select item> ','s');
    switch choice
%__________________________________________________________________________
        case '0' %Quit
            break
%__________________________________________________________________________
        case '1';  %Plot raw spectra of Fast Photodiode & PMT Traces
            wdir=uigetdir('','Select directory for raw pmt files');
            file=uigetfile([wdir,'\*.pmt'],'Select .pmt file');
            PMT_plotRawData(wdir,file);%plot each event from .pmt file
%__________________________________________________________________________
        case '2';  %write Summary File
            wdir=uigetdir('','Select directory for raw data files');
            PMT_MakeSumFile(wdir);
%__________________________________________________________________________
        case '3';  %Analyze sumfile (New)
            wdir=uigetdir('','Select directory for sumfile');
            file1=uigetfile([wdir,'\*.csv'],'Select sumfile');
            PMT_Analysis_Zero_Supressed(wdir,file1);
%__________________________________________________________________________
        case '4'; %Make Rise and Fall Plots for CFD Offset
            wdir=uigetdir('','Select directory for raw data files');
            PMT_MakeRisePlots(wdir)
%__________________________________________________________________________
        case '5'; %Analyze Peak Timing Info
            wdir=uigetdir('','Select directory');
            PMT_PeakFinder(wdir);
%__________________________________________________________________________
        case '6'; %Response Histogram and Max Response
            wdir=uigetdir('','Select directory for raw data files');
            PMT_MakeResponse_Histogram(wdir);
%__________________________________________________________________________
        case '7'; %SumFile Analysis
            wdir=uigetdir('','Select directory for SumFile');
            file1=uigetfile([wdir,'\*.csv'],'Select sumfile');
            PMT_Analysis(wdir,file1)
%__________________________________________________________________________
        case '8';  %Before and After Pre-Scan Info
            wdir=uigetdir('','Select directory for pmtdata');
            file1=uigetfile([wdir,'\*.pmt'],'Select data point');
            PMT_PreScan(wdir,file1)
%__________________________________________________________________________
        case '9';  %Analyze Peak Timing Info
            wdir=uigetdir('','Select directory for PMT Timing Info File');
            file1=uigetfile([wdir,'\*.csv'],'Select PMT Timing sumfile');
            Peak_Analysis(wdir,file1)
%__________________________________________________________________________

        otherwise
            choice=' ';
    end
end

fclose('all');
```

```
function [offset]=FPD_Offset(data)

latetime = 1000; %Choosen by reviewing typical events (very safe margin)
offset = mean(data(latetime:end));




function [area,start,stop,maxpeak]=FPD_PeakSub(smoothdata)

global fpdoffset fpdcfd

[maxpeak,peakch]=max(smoothdata);
full=length(smoothdata);

while peakch > full-1000 %the peak can't be in last 1000 channels. If it is,
    smoothdata(peakch:full)=0; %zero it out and look for another peak
    [maxpeak,peakch]=max(smoothdata);
end

%Insert peak time offset
timedelay=fpdoffset;%Time Delay in Number of Channels
delay=zeros(timedelay,1);%Zeros because already subtracted offset
delayed=[delay.',smoothdata.'];%Add Time Delay at start of wave trace

%Create inverted delayed peak
invert_delayed=-1.*delayed;

%Create Attenuated peak
atten = [fpdcfd.*(smoothdata.'),(delay.')];

data = invert_delayed + atten; %final data set for calculating timing

start=500; %arbitrary start value for zero crossing point
for i = (peakch + timedelay):-1:10

    if data(i) < 0; %zero crossing

        start=i;%Chanel of Zero Crossing
        break;

    end

end

stop=900;%Arbitrary end point in timing
for i = peakch:+1:full

    if smoothdata(i)< fpdcfd*maxpeak;%Arbitrary level for end time

        stop=i;
        break;
    end

end


%Subtracting back time delay added to pulse at start of pulse analysis
start = start - timedelay;%This subtraction should be the same for all
area=sum(smoothdata(start:stop)); %total area

end






function [sum,rise,fall,start,stop,peakch]=FPD_RiseTime(smoothdata)

global fpdcfd

[peakmax,peakch]=max(smoothdata);
full=length(smoothdata);

while peakch>full-1000 %the peak can't be in the last 1000 channels so if it
is,
    smoothdata(peakch:full)=0; %it zeros it out and looks for another peak
    [peakmax,peakch]=max(smoothdata);
end


start=300; %arbitrary start channel
for i=peakch:-1:5 %Finding the start time of the rising edge

    if smoothdata(i) < fpdcfd*peakmax;

        start=i;
        break;

    end

end

rise=peakch-start;

stop=1500; %arbitrary stop channel
    for i=peakch:+1:full

        if smoothdata(i)< fpdcfd*peakmax;

            stop=i;
            break;

        end
    end

fall=stop-peakch;
sum = stop-start;
```

```
function Peak_Analysis(wdir,file1)

global minradius xcenter ycenter

%load data file
filename=[wdir,'/',file1];
fprintf(1,'\n Analyzing %s\n',filename);
data=load(filename);%load the .sum file chosen earlier

%Arr =
[x,y,tstart,tstop,dt,atstart,atstop,adt,adiff,ax,ay,vtstart,vtstop,vdt,vdiff,
vx,vy];

atstart=[];atstop=[];adiff=[];
adt=[];ax=[];ay=[];

vtstart=[];vtstop=[];vdiff=[];
vdt=[];vx=[];vy=[];

i=1;
while data(i,10)~= -1
    distance = sqrt((data(i,10)-xcenter)^2 + (data(i,11)-ycenter)^2);
    if distance > 12 && distance <35
        %loading the averages at each spatial location saved
        atstart =[atstart;data(i,6)];
        atstop = [atstop;data(i,7)];

        adt = [adt;data(i,8)];
        adiff = [adiff;data(i,9)];

        ax = [ax;data(i,10)];
        ay = [ay;data(i,11)];

        %loading the variances at each spatial location saved
        vtstart =[vtstart;data(i,12)];
        vtstop = [vtstop;data(i,13)];
        vdt = [vdt;data(i,14)];
        vdiff = [vdiff;data(i,15)];

        vx = [vx;data(i,16)];
        vy = [vy;data(i,17)];
    end
    i=i+1;
end

%Number of Spatial Locations NOT thrown out
num = length(ax);

ax = ax - xcenter;
ay = ay - ycenter;

%triagulate grid points (x,y)
tri = delaunay(ax,ay);

%Plot Saved Location Points
plot(ax,ay,'*r')
xlabel('x (mm)')
ylabel('y (mm)')
s=sprintf('Saved DOUBLE ONLY Peak Points; Spatial Locations = %g ',num);
title(s)
axis equal
pause


%Plot of Pre-Bump Peak Times
hist(atstart)
xlabel('Mean Pre-Bump Peak Location (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Pre-Bump Peak Location; Spatial Locations = %g
',num);
title(s)
pause

hist(vtstart)
xlabel('Variance of Pre-Bump Peak Location (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of Pre-Bump Peak Location; Spatial Locations
= %g ',num);
title(s)
pause

%Plot of Peak Channel Location
hist(atstop)
xlabel('Mean Peak Channel Location (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Peak Channel Location; Spatial Locations = %g ',num);
title(s)
pause

hist(vtstop)
xlabel('Variance of Peak Channel Location (100 ps Channels)')
```

```
ylabel('Count')
s=sprintf('Histogram of Variance of Peak Channel Location; Spatial Locations
= %g ',num);
title(s)
pause


%Scatter Plot of Transit-Time Variance vs Transit-Time
plot(adt,vdt,'*')
xlabel('Mean Transit-Time (100 ps Channels)')
ylabel('Transit-Time Variance')
s=sprintf('Scatter Plot of Transit-Time Variance vs Transit-Time; Total
Points = %g ',num);
title(s)
pause

%Distribution of Transit Times
hist(adt)
xlabel('Mean Transit-Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Distribution of Transit-Times; Spatial Locations = %g ',num);
title(s)
pause

%Distribution of Peak Time Difference
hist(adiff)
xlabel('Mean Peak Time Difference (100 ps Channels)')
ylabel('Count')
s=sprintf('Distribution of Peak Time Difference; Spatial Locations = %g
',num);
title(s)
pause

hist(vdiff)
xlabel('Variance of Peak Time Difference (100 ps Channels)')
ylabel('Count')
s=sprintf('Distribution of Variance of Peak Time Difference; Spatial
Locations = %g ',num);
title(s)
pause


%Plot Transit Time
dtnano = adt .* (.1); %nanosecond scale
hh=figure(1);
set(hh,'Units','normalized');
set(hh,'OuterPosition',[.3 0 .6 .9])
trisurf(tri, ax, ay, dtnano,-dtnano)%(-dtnano is to make red fast and blue
slow)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('dt (ns)')
s=sprintf('Mean Transit Time; Spatial Locations = %g ',num);
title(s)
```

```
shading interp
colorbar EastOutside


%Plot Peak Differnce
h2=figure(2);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .6 .9])
trisurf(tri, ax, ay, adiff)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Mean Peak Time Difference (100 ps Channels)')
t=sprintf('Peak Time Difference; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside


%Plot Pre-bump Start Channel
h2=figure(3);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .6 .9])
trisurf(tri, ax, ay, atstart)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Mean Pre-Bump Peak Location (100 ps Channels)')
t=sprintf('Pre-Bump Peak Location; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

pause

end








function PMT_Analysis(wdir,file1)

global minradius xcenter ycenter

%load data file
filename=[wdir,'/',file1];
fprintf(1,'\n Analyzing %s\n',filename);
data=load(filename);%load the .sum file chosen earlier

% Arr =
%[x,y,tstart,tstop,tarea,tpeak,start,stop,lstop,area,peak,dt,timestep,larea,
% atstart,atstop,atarea,atpeak,astart,astop,alstop,aarea,apeak,adt,alarea,
%
ax,ay,vtstart,vtstop,vtarea,vtpeak,vstart,vstop,vlstop,varea,vpeak,vdt,vlarea
,vx,vy];

%full illumination = sum of all individual transit times within useful
diameter
alldt = [];

%Overall Averages
atstart=[];atstop=[];atarea=[];atpeak=[];astart=[];astop=[];alstop=[];aarea=[
];apeak=[];
adt=[];alarea=[];ax=[];ay=[];

%Overall Variances
vtstart=[];vtstop=[];vtarea=[];vtpeak=[];vstart=[];vstop=[];vlstop=[];varea=[
];vpeak=[];
vdt=[];vlarea=[];vx=[];vy=[];

xtstart=[];xtstop=[];xtarea=[];xtpeak=[];xstart=[];xstop=[];xlstop=[];xarea=[
];xpeak=[];
xdt=[];xlarea=[];xx=[];xx=[];

ytstart=[];ytstop=[];ytarea=[];ytpeak=[];ystart=[];ystop=[];ylstop=[];yarea=[
];ypeak=[];
ydt=[];ylarea=[];yy=[];yy=[];

index=1;
while index <= length(data)

    if sqrt((data(index,1)-xcenter)^2 + (data(index,2)-ycenter)^2) <=
minradius

        alldt = [alldt;data(index,12)];

    end
    index = index + 1;
end

i=1;
while data(i,26)~= -1
    if sqrt((data(i,26)-xcenter)^2 + (data(i,27)-ycenter)^2)<=minradius
        %loading the averages at each spatial location saved
        atstart =[atstart;data(i,15)];
        atstop = [atstop;data(i,16)];
        atarea = [atarea;data(i,17)];
        atpeak = [atpeak;data(i,18)];

        astart = [astart;data(i,19)];
        astop = [astop;data(i,20)];
        alstop = [alstop;data(i,21)];
        aarea = [aarea;data(i,22)];
        apeak = [apeak;data(i,23)];
```

```
adt = [adt;data(i,24)];
alarea = [alarea;data(i,25)];

ax = [ax;data(i,26)];
ay = [ay;data(i,27)];

%loading the variances at each spatial location saved
vtstart = [vtstart;data(i,28)];
vtstop = [vtstop;data(i,29)];
vtarea = [vtarea;data(i,30)];
vtpeak = [vtpeak;data(i,31)];

vstart = [vstart;data(i,32)];
vstop = [vstop;data(i,33)];
vlstop = [vlstop;data(i,34)];
varea = [varea;data(i,35)];
vpeak = [vpeak;data(i,36)];

vdt = [vdt;data(i,37)];
vlarea = [vlarea;data(i,38)];

vx = [vx;data(i,39)];
vy = [vy;data(i,40)];

%load x-axis information
if (data(i,27)-ycenter) == 0

    xtstart =[xtstart;data(i,15)];
    xtstop = [xtstop;data(i,16)];
    xtarea = [xtarea;data(i,17)];
    xtpeak = [xtpeak;data(i,18)];

    xstart = [xstart;data(i,19)];
    xstop = [xstop;data(i,20)];
    xlstop = [xlstop;data(i,21)];
    xarea = [xarea;data(i,22)];
    xpeak = [xpeak;data(i,23)];

    xdt = [xdt;data(i,24)];
    xlarea = [xlarea;data(i,25)];

    xx = [xx;data(i,26)];

end

%load y-axis information
if (data(i,26)-xcenter) == 0

    ytstart =[ytstart;data(i,15)];
    ytstop = [ytstop;data(i,16)];
    ytarea = [ytarea;data(i,17)];
    ytpeak = [ytpeak;data(i,18)];

    ystart = [ystart;data(i,19)];
```

```
            ystop = [ystop;data(i,20)];
            ylstop = [ylstop;data(i,21)];
            yarea = [yarea;data(i,22)];
            ypeak = [ypeak;data(i,23)];

            ydt = [ydt;data(i,24)];
            ylarea = [ylarea;data(i,25)];

            yy = [yy;data(i,27)];

        end

    else
        %loading the zeros at all other spatial locations saved

        atarea = [atarea;0.0000001];
        atpeak = [atpeak;0.0000001];


        aarea = [aarea;0];
        apeak = [apeak;0];

        adt = [adt;0];
        alarea = [alarea;0];

        ax = [ax;data(i,26)];
        ay = [ay;data(i,27)];



        vtarea = [vtarea;0];
        vtpeak = [vtpeak;0];


        varea = [varea;0];
        vpeak = [vpeak;0];

        vdt = [vdt;0];
        vlarea = [vlarea;0];

        vx = [vx;data(i,39)];
        vy = [vy;data(i,40)];

        %load x-axis information
        if (data(i,27)-ycenter) == 0

            xtstart =[xtstart;0];
            xtstop = [xtstop;0];
            xtarea = [xtarea;0.0000001];
            xtpeak = [xtpeak;0.0000001];

            xstart = [xstart;0];
            xstop = [xstop;0];
            xlstop = [xlstop;0];
            xarea = [xarea;0];
            xpeak = [xpeak;0];

            xdt = [xdt;0];
            xlarea = [xlarea;0];

            xx = [xx;data(i,26)];

        end

        %load y-axis information
        if (data(i,26)-xcenter) == 0

            ytstart =[ytstart;0];
            ytstop = [ytstop;0];
            ytarea = [ytarea;0.0000001];
            ytpeak = [ytpeak;0.0000001];

            ystart = [ystart;0];
            ystop = [ystop;0];
            ylstop = [ylstop;0];
            yarea = [yarea;0];
            ypeak = [ypeak;0];

            ydt = [ydt;0];
            ylarea = [ylarea;0];

            yy = [yy;data(i,27)];

        end

    end
    i=i+1;
end


%Number of Spatial Locations Saved
num = length(ax);

%Area Response
relativearea = aarea./atarea;
relativearea = relativearea./max(relativearea);
lrelativearea = alarea./atarea;
relpeak = apeak./atpeak;
relpeak = relpeak./max(relpeak);

%triagulate grid points (x,y)
ax = ax-xcenter;
ay = ay-ycenter;
tri = delaunay(ax,ay);

%Plot Saved Location Points
hh=figure;
```

```
set(hh,'Units','normalized');
set(hh,'OuterPosition',[.3 0 .2 .35])
plot(ax,ay,'*r')
xlabel('x (mm)')
ylabel('y (mm)')
s=sprintf('Saved Points; Spatial Locations = %g ',num);
title(s)
axis equal
pause


%Plot of Trigger Start Times
hist(atstart)
xlabel('Trigger Start Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Trigger Start Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vtstart)
xlabel('Variance of Trigger Start Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of Trigger Start Times; Spatial Locations =
%g ',num);
title(s)
pause

%Plot of Trigger Stop Times
hist(atstop)
xlabel('Trigger Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Trigger Stop Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vtstop)
xlabel('Variance of Trigger Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of Trigger Stop Times; Spatial Locations =
%g ',num);
title(s)
pause

%Plot of Start Times
hist(astart)
xlabel('PMT Start Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of PMT Start Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vstart)
xlabel('Variance of PMT Start Time (100 ps Channels)')
ylabel('Count')
```

```
s=sprintf('Histogram of Variance of PMT Start Times; Spatial Locations = %g
',num);
title(s)
pause

%Plot of Stop Times
hist(astop)
xlabel('PMT Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of PMT Stop Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vstop)
xlabel('Variance of PMT Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of PMT Stop Times; Spatial Locations = %g
',num);
title(s)
pause

%Plot of Late Stop Times
hist(alstop)
xlabel('PMT Late Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Late PMT Stop Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vlstop)
xlabel('Variance of PMT Late Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of Late PMT Stop Times; Spatial Locations =
%g ',num);
title(s)
pause

%Scatter Plot of Transit-Time Variance vs Transit-Time
plot(adt,vdt,'*')
xlabel('Transit-Time (100 ps Channels)')
ylabel('Transit-Time Variance')
s=sprintf('Scatter Plot of Transit-Time Variance vs Transit-Time; Total
Points = %g ',num);
title(s)
pause

%Distribution of Transit Times
dtnano = adt .* (.1); %nanosecond scale
hist(dtnano)
xlabel('Transit-Time (ns)')
ylabel('Count')
s=sprintf('Distribution of Transit-Times; Spatial Locations = %g ',num);
title(s)
pause
```

```
%Plot Transit Time
dtnano = adt .* (.1); %nanosecond scale
hh=figure(1);
set(hh,'Units','normalized');
set(hh,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, dtnano,-dtnano)%(-dtnano is to make red fast and blue
slow)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Transit-Time (ns)')
s=sprintf('Transit Time(Average); Spatial Locations = %g ',num);
title(s)
shading interp
colorbar EastOutside
%colorbar('YTickLabel',{'Very Slow','Slow','Average','Fast','VeryFast'})


%Plot Relative Area Response
h2=figure(2);
set(h2,'Units','normalized');
set(hh,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, relativearea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Normalized Area Ratio')
t=sprintf('Normalized Relative Area(PMT Area/FPD Area); Spatial Locations =
%g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Area (Late) Response
h3=figure(3);
set(h3,'Units','normalized');
set(h3,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, lrelativearea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Area Ratio (PMT Area/FPD Area)')
t=sprintf('Relative Area(Late); Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Area Response
aarea = aarea./max(aarea);
h2=figure(4);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, aarea)
axis vis3d
xlabel('x (mm)')
```

```
ylabel('y (mm)')
zlabel('Normalized Area')
t=sprintf('Normalized PMT Area; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Late Area Response
h2=figure(9);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, alarea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Area')
t=sprintf('Late PMT Area; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Trigger Area Response
h2=figure(5);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, atarea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Area')
t=sprintf('Trigger Area; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Trigger Peak Response
h2=figure(6);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, atpeak)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('V')
t=sprintf('Trigger Peaks; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot PMT Peak Response
h2=figure(7);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, apeak)
axis vis3d
xlabel('x (mm)')
```

```
ylabel('y (mm)')
zlabel('V')
t=sprintf('PMT Peak Response; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Relative PMT Peak Response
h2=figure(8);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, relpeak)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Normalized Peak Ratio (PMT Peak/FPD Peak)')
t=sprintf('Normalized Relative PMT Peaks (PMT Peak/FPD Peak); Spatial
Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Axis Timing Uniformity

[xx,index1] = sort(xx);
xdt = xdt(index1,:).*.1;%sort and covert to ns;
xx = xx-xcenter;%centering values

h2=figure(9);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .6 .9])
plot(xx,xdt,'*-')
xlabel('X Position (mm)')
ylabel('Time (ns)')
t=sprintf('X-Axis Transit-Time Uniformity');
title(t)

[yy,index2] = sort(yy);
ydt = ydt(index2,:).*.1;%sort and covert to ns;
yy = yy-ycenter;

%Plot Axis Timing Uniformity
h2=figure(10);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .6 .9])
plot(yy,ydt,'*-')
xlabel('Y Position (mm)')
ylabel('Time (ns)')
t=sprintf('Y-Axis Transit-Time Uniformity');
title(t)

%Plot Axis Area Uniformity
xrelarea = xarea./xtarea;
xarea = xarea(index1,:);
xarea = xarea./max(xarea);
```

```
h2=figure(11);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xarea,'*-')
xlabel('X Position (mm)')
ylabel('Normalized Area')
t=sprintf('X-Axis PMT Area Response Uniformity');
title(t)

yrelarea = yarea./ytarea;
yarea = yarea(index2,:);
yarea = yarea./max(yarea);

%Plot Axis Area Uniformity
h2=figure(12);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,yarea,'*-')
xlabel('Y Position (mm)')
ylabel('Normalized Area')
t=sprintf('Y-Axis PMT Area Response Uniformity');
title(t)


%Plot Axis Peak Uniformity

xpeak = xpeak(index1,:);

h2=figure(13);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xpeak,'*-')
xlabel('X Position (mm)')
ylabel('V')
t=sprintf('X-Axis PMT Peak Response Uniformity');
title(t)

ypeak = ypeak(index2,:);

%Plot Axis Peak Uniformity
h2=figure(14);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,ypeak,'*-')
xlabel('Y Position (mm)')
ylabel('V')
t=sprintf('Y-Axis PMT Peak Response Uniformity');
title(t)


%Plot Axis Relative Area Uniformity
xrelarea = xrelarea(index1,:);
xrelarea = xrelarea./max(xrelarea);
```

```
h2=figure(15);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xrelarea,'*-')
xlabel('X Position (mm)')
ylabel('Normalized Relative Area (PMT/FPD)')
t=sprintf('X-Axis Relative Area Uniformity');
title(t)

%Plot Axis Relative Area Uniformity
yrelarea = yrelarea(index2,:);
yrelarea = yrelarea./max(yrelarea);

h2=figure(16);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,yrelarea,'*-')
xlabel('Y Position (mm)')
ylabel('Normalized Relative Area (PMT/FPD)')
t=sprintf('Y-Axis Relative Area Uniformity ');
title(t)


%Plot Full Ilumination
h2=figure(17);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
hist(alldt,50)
xlabel('Transit-Time (100 ps Channels)')
ylabel('Count')
t=sprintf('Transit-Time Spread (TTS)');
title(t)

pause

end






function PMT_Analysis_Zero_Supressed(wdir,file1)

global minradius xcenter ycenter

%load data file
filename=[wdir,'/',file1];
fprintf(1,'\n Analyzing %s\n',filename);
data=load(filename);%load the .sum file chosen earlier

% Arr =
%[x,y,tstart,tstop,tarea,tpeak,start,stop,lstop,area,peak,dt,timestep,larea,
% atstart,atstop,atarea,atpeak,astart,astop,alstop,aarea,apeak,adt,alarea,
%
ax,ay,vtstart,vtstop,vtarea,vtpeak,vstart,vstop,vlstop,varea,vpeak,vdt,vlarea
,vx,vy];

%full illumination = sum of all individual transit times within useful
diameter
alldt = [];

%Overall Averages
atstart=[];atstop=[];atarea=[];atpeak=[];astart=[];astop=[];alstop=[];aarea=[
];apeak=[];
adt=[];alarea=[];ax=[];ay=[];

%Overall Variances
vtstart=[];vtstop=[];vtarea=[];vtpeak=[];vstart=[];vstop=[];vlstop=[];varea=[
];vpeak=[];
vdt=[];vlarea=[];vx=[];vy=[];

xtstart=[];xtstop=[];xtarea=[];xtpeak=[];xstart=[];xstop=[];xlstop=[];xarea=[
];xpeak=[];
xdt=[];xlarea=[];xx=[];xx=[];

ytstart=[];ytstop=[];ytarea=[];ytpeak=[];ystart=[];ystop=[];ylstop=[];yarea=[
];ypeak=[];
ydt=[];ylarea=[];yy=[];yy=[];

index=1;
while index <= length(data)

    if sqrt((data(index,1)-xcenter)^2 + (data(index,2)-ycenter)^2) <=
minradius

       alldt = [alldt;data(index,12)];

    end
    index = index + 1;
end

i=1;
while data(i,26)~= -1
    if sqrt((data(i,26)-xcenter)^2 + (data(i,27)-ycenter)^2)<=minradius
        %loading the averages at each spatial location saved
        atstart =[atstart;data(i,15)];
        atstop = [atstop;data(i,16)];
        atarea = [atarea;data(i,17)];
        atpeak = [atpeak;data(i,18)];

        astart = [astart;data(i,19)];
        astop = [astop;data(i,20)];
        alstop = [alstop;data(i,21)];
        aarea = [aarea;data(i,22)];
        apeak = [apeak;data(i,23)];
```

```
adt = [adt;data(i,24)];
alarea = [alarea;data(i,25)];

ax = [ax;data(i,26)];
ay = [ay;data(i,27)];

%loading the variances at each spatial location saved
vtstart = [vtstart;data(i,28)];
vtstop = [vtstop;data(i,29)];
vtarea = [vtarea;data(i,30)];
vtpeak = [vtpeak;data(i,31)];

vstart = [vstart;data(i,32)];
vstop = [vstop;data(i,33)];
vlstop = [vlstop;data(i,34)];
varea = [varea;data(i,35)];
vpeak = [vpeak;data(i,36)];

vdt = [vdt;data(i,37)];
vlarea = [vlarea;data(i,38)];

vx = [vx;data(i,39)];
vy = [vy;data(i,40)];

%load x-axis information
if (data(i,27)-ycenter) == 0

    xtstart =[xtstart;data(i,15)];
    xtstop = [xtstop;data(i,16)];
    xtarea = [xtarea;data(i,17)];
    xtpeak = [xtpeak;data(i,18)];

    xstart = [xstart;data(i,19)];
    xstop = [xstop;data(i,20)];
    xlstop = [xlstop;data(i,21)];
    xarea = [xarea;data(i,22)];
    xpeak = [xpeak;data(i,23)];

    xdt = [xdt;data(i,24)];
    xlarea = [xlarea;data(i,25)];

    xx = [xx;data(i,26)];

end

%load y-axis information
if (data(i,26)-xcenter) == 0

    ytstart =[ytstart;data(i,15)];
    ytstop = [ytstop;data(i,16)];
    ytarea = [ytarea;data(i,17)];
    ytpeak = [ytpeak;data(i,18)];

    ystart = [ystart;data(i,19)];
```

```
            ystop = [ystop;data(i,20)];
            ylstop = [ylstop;data(i,21)];
            yarea = [yarea;data(i,22)];
            ypeak = [ypeak;data(i,23)];

            ydt = [ydt;data(i,24)];
            ylarea = [ylarea;data(i,25)];

            yy = [yy;data(i,27)];

        end

    end
    i=i+1;
end

%Number of Spatial Locations NOT thrown out within useful diameter
num = length(ax);

%Area Response
relativearea = aarea./atarea;
relativearea = relativearea./max(relativearea);
lrelativearea = alarea./atarea;
relpeak = apeak./atpeak;
relpeak = relpeak./max(relpeak);

%triagulate grid points (x,y)
ax = ax-xcenter;
ay = ay-ycenter;
tri = delaunay(ax,ay);

%Plot Saved Location Points
hh=figure;
set(hh,'Units','inches');
set(hh,'OuterPosition',[2 2 4 4])
plot(ax,ay,'*r')
xlabel('x (mm)')
ylabel('y (mm)')
s=sprintf('Saved Points within Minimum Useful Diameter; Spatial Locations =
%g ',num);
title(s)
axis equal
pause


%Plot of Trigger Start Times
hist(atstart)
xlabel('Trigger Start Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Trigger Start Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vtstart)
xlabel('Variance of Trigger Start Time (100 ps Channels)')
```

```
ylabel('Count')
s=sprintf('Histogram of Variance of Trigger Start Times; Spatial Locations =
%g ',num);
title(s)
pause

%Plot of Trigger Stop Times
hist(atstop)
xlabel('Trigger Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Trigger Stop Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vtstop)
xlabel('Variance of Trigger Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of Trigger Stop Times; Spatial Locations =
%g ',num);
title(s)
pause

%Plot of Start Times
hist(astart)
xlabel('PMT Start Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of PMT Start Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vstart)
xlabel('Variance of PMT Start Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of PMT Start Times; Spatial Locations = %g
',num);
title(s)
pause

%Plot of Stop Times
hist(astop)
xlabel('PMT Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of PMT Stop Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vstop)
xlabel('Variance of PMT Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of PMT Stop Times; Spatial Locations = %g
',num);
title(s)
pause

%Plot of Late Stop Times
```

```
hist(alstop)
xlabel('PMT Late Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Late PMT Stop Times; Spatial Locations = %g ',num);
title(s)
pause

hist(vlstop)
xlabel('Variance of PMT Late Stop Time (100 ps Channels)')
ylabel('Count')
s=sprintf('Histogram of Variance of Late PMT Stop Times; Spatial Locations =
%g ',num);
title(s)
pause

%Scatter Plot of Transit-Time Variance vs Transit-Time
plot(adt,vdt,'*')
xlabel('Transit-Time (100 ps Channels)')
ylabel('Transit-Time Variance')
s=sprintf('Scatter Plot of Transit-Time Variance vs Transit-Time; Total
Points = %g ',num);
title(s)
pause

%Distribution of Transit Times
dtnano = adt .* (.1); %nanosecond scale
hist(dtnano,30)
xlabel('Transit-Time (ns)')
ylabel('Count')
s=sprintf('Distribution of Transit-Times; Spatial Locations = %g ',num);
title(s)
pause



%Plot Transit Time
dtnano = adt .* (.1); %nanosecond scale
hh=figure(1);
set(hh,'Units','normalized');
set(hh,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, dtnano,-dtnano)%(-dtnano is to make red fast and blue
slow)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Transit-Time (ns)')
s=sprintf('Transit Time(Average); Spatial Locations = %g ',num);
title(s)
shading interp
colorbar EastOutside
%colorbar('YTickLabel',{'Very Slow','Slow','Average','Fast','VeryFast'})


%Plot Relative Area Response
h2=figure(2);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, relativearea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Normalized Area Ratio')
t=sprintf('Normalized Relative Area(PMT Area/FPD Area); Spatial Locations =
%g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Area (Late) Response
h3=figure(3);
set(h3,'Units','normalized');
set(h3,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, lrelativearea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Area Ratio (PMT Area/FPD Area)')
t=sprintf('Relative Area(Late); Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Area Response
aarea = aarea./max(aarea);
h2=figure(4);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, aarea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Normalized Area')
t=sprintf('Normalized PMT Area; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Late Area Response
h2=figure(9);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, alarea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Area')
t=sprintf('Late PMT Area; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Trigger Area Response
```

```
h2=figure(5);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, atarea)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Area')
t=sprintf('Trigger Area; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Trigger Peak Response
h2=figure(6);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, atpeak)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('(V)')
t=sprintf('Trigger Peaks; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot PMT Peak Response
h2=figure(7);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, apeak)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('(V)')
t=sprintf('PMT Peaks; Spatial Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside

%Plot Relative PMT Peak Response
h2=figure(8);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .35])
trisurf(tri, ax, ay, relpeak)
axis vis3d
xlabel('x (mm)')
ylabel('y (mm)')
zlabel('Normalized Peak Ratio (PMT Peak/FPD Peak)')
t=sprintf('Normalized Relative PMT Peaks (PMT Peak/FPD Peak); Spatial
Locations = %g ',num);
title(t)
shading interp
colorbar EastOutside
```

```
%Plot Axis Timing Uniformity

[xx,index1] = sort(xx);
xdt = xdt(index1,:).*.1;%sort and covert to ns;
xx = xx-xcenter;%centering values

h2=figure(9);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xdt,'*-')
xlabel('X Position (mm)')
ylabel('Time (ns)')
t=sprintf('X-Axis Transit-Time Uniformity');
title(t)

[yy,index2] = sort(yy);
ydt = ydt(index2,:).*.1;%sort and covert to ns;
yy = yy-ycenter;

%Plot Axis Timing Uniformity
h2=figure(10);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,ydt,'*-')
xlabel('Y Position (mm)')
ylabel('Time (ns)')
t=sprintf('Y-Axis Transit-Time Uniformity');
title(t)

%Plot Axis Area Uniformity
xrelarea = xarea./xtarea;
xarea = xarea(index1,:);
xarea = xarea./max(xarea);

h2=figure(11);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xarea,'*-')
xlabel('X Position (mm)')
ylabel('Normalized Area')
t=sprintf('X-Axis PMT Area Response Uniformity');
title(t)

yrelarea = yarea./ytarea;
yarea = yarea(index2,:);
yarea = yarea./max(yarea);

%Plot Axis Area Uniformity
h2=figure(12);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,yarea,'*-')
xlabel('Y Position (mm)')
ylabel('Normalized Area')
t=sprintf('Y-Axis PMT Area Response Uniformity');
```

```
title(t)


%Plot Axis Peak Uniformity

xpeak = xpeak(index1,:);

h2=figure(13);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xpeak,'*-')
xlabel('X Position (mm)')
ylabel('V')
t=sprintf('X-Axis PMT Peak Response Uniformity');
title(t)

ypeak = ypeak(index2,:);

%Plot Axis Peak Uniformity
h2=figure(14);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,ypeak,'*-')
xlabel('Y Position (mm)')
ylabel('V')
t=sprintf('Y-Axis PMT Peak Response Uniformity');
title(t)


%Plot Axis Relative Area Uniformity
xrelarea = xrelarea(index1,:);
xrelarea = xrelarea./max(xrelarea);

h2=figure(15);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(xx,xrelarea,'*-')
xlabel('X Position (mm)')
ylabel('Normalized Relative Area (PMT/FPD)')
t=sprintf('X-Axis Relative Area Uniformity');
title(t)

%Plot Axis Relative Area Uniformity
yrelarea = yrelarea(index2,:);
yrelarea = yrelarea./max(yrelarea);

h2=figure(16);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
plot(yy,yrelarea,'*-')
xlabel('Y Position (mm)')
ylabel('Normalized Relative Area (PMT/FPD)')
t=sprintf('Y-Axis Relative Area Uniformity ');
title(t)
```

```
%Plot Full Ilumination
h2=figure(17);
set(h2,'Units','normalized');
set(h2,'OuterPosition',[.3 0 .2 .3])
hist(alldt,50)
xlabel('Transit-Time (100 ps Channels)')
ylabel('Count')
t=sprintf('Transit-Time Spread (TTS)');
title(t)

pause

end





function PMT_MakeResponse_Histogram(wdir)

global smoothingfactor


savename='Response';%name of save file
wdir2=dir(wdir);
numberfiles=length(wdir2)-2;%two hidden unecessary files in every folder

%set arrays and response variables
maxrespeak=[];maxrelpeak=[];maxnoise=[];noiseratio=[];MAX=1;xmax=0;ymax=0;
trigpeak=[];

%loop over each .pmt file and make a rise file for each one
fileindex=0;
while fileindex < numberfiles
    fileindex=fileindex+1;

    %find the next file and load it
    file=wdir2(fileindex+2).name;
    if ~strcmp(file(end-2:end),'pmt')
        continue;
    end
    filename=[wdir,'/',file];
    fprintf(1,'Analyzing %s\n',filename);
    data=load(filename);%load the .pmt file chosen earlier

    %number of data points per trace
    [numtraces,~] = size(data);

    %create temporary bins
    maxrespeak_1 = [];
    maxrelpeak_1 = [];
    noiseratio_1 = [];
    maxnoise_1 = [];
    trigpeak_1 = [];

    event = 0; % comes in pairs of traces
    while event < numtraces

        event=event+2;
        pmt=data(event-1,3:end);%get pmt pulse
        pmt = pmt*-1; %invert signal

        fpd=data(event,3:end);
        offset2=FPD_Offset(fpd);
        fpd=fpd-offset2;

        %calulates offset for the anodes (PMT)
        offset = PMT_Offset(pmt);
        pmt=pmt-offset;%apply the offset

        %All remaining calculations done using smoothed data
        pmt = smooth(pmt,smoothingfactor);
        fpd = smooth(fpd,smoothingfactor);

        x1=data(event-1,1);%get the grid position
        y1=data(event,1);%get the grid position

        maxpmt = max(pmt);
        maxfpd = max(fpd);
        rel=maxpmt/maxfpd;

        noise = -min(pmt);%already subtracted offset
        signalvsnoise = maxpmt/noise;

        if maxpmt>MAX
            MAX = maxpmt;
            xmax = x1;
            ymax = y1;
        end

        maxrespeak_1=[maxrespeak_1;maxpmt];
        maxrelpeak_1=[maxrelpeak_1;rel];
        noiseratio_1=[noiseratio_1;signalvsnoise];
        maxnoise_1=[maxnoise_1;noise];
        trigpeak_1=[trigpeak_1;maxfpd];

    end

    maxrespeak=[maxrespeak;maxrespeak_1];
    maxrelpeak=[maxrelpeak;maxrelpeak_1];
    noiseratio=[noiseratio;noiseratio_1];
    maxnoise=[maxnoise;maxnoise_1];
    trigpeak=[trigpeak;trigpeak_1];

end

of=[savename,'.response'];
```

```
Arr = [maxrespeak,maxrelpeak,noiseratio,maxnoise];
dlmwrite(of,Arr,'precision', 7);
fclose('all');

%Make Plots
hist(maxrespeak,30)
title('Max PMT Response Peak Histogram')
ylabel('Count')
xlabel('(V)')
pause

hist(maxrelpeak,30)
title('Max Relative Response Peak Histogram')
ylabel('Count')
xlabel('(Max PMT/Max FPD)')
pause

hist(noiseratio,100)
title('Signal to Noise Ratio Histogram')
ylabel('Count')
xlabel('(Max PMT Response/Max PMT Noise)')
pause

hist(maxnoise,100)
title('Maximum Noise Histogram')
ylabel('Count')
xlabel('(V)')
pause

hist(trigpeak,100)
title('Trigger Peak Histogram')
ylabel('Count')
xlabel('(V)')
pause

fprintf('Max Response = %g at X=%g Y=%g',MAX,xmax,ymax)

close all;







function PMT_MakeRisePlots(wdir)

global fpdthreshold pmtthreshold req datapoints smoothingfactor

savename=input('Enter PMT Rise File Name - ','s');
wdir2=dir(wdir);
numberfiles=length(wdir2)-2;%two hidden unecessary files in every folder
```

```
%Creating a summary file of slow rise events
res = zeros(1,datapoints);
dlmwrite('slowriseevents.csv',res);

%set arrays for complete Risefile
rise=[];fall=[];sum=[];
trise=[];tfall=[];tsum=[];trigch=[];
trigpeakch=[];
start =[];
stop =[];
tstop =[];

notenough=zeros(1,2)-4;
dlmwrite('notenoughrise.csv',notenough);

%loop over each .pmt file and make a rise file for each one
fileindex=0;
while fileindex < numberfiles
    fileindex=fileindex+1;

    %find the next file and load it
    file=wdir2(fileindex+2).name;
    if ~strcmp(file(end-2:end),'pmt')
        continue;
    end
    filename=[wdir,'/',file];
    fprintf(1,'Analyzing %s\n',filename);
    data=load(filename);%load the .pmt file chosen earlier

    %number of data points per trace
    [numtraces,~] = size(data);

    %create temporary bins
    rise_1 = [];
    fall_1 = [];
    sum_1 = [];
    start_1 =[];
    stop_1 =[];
    tstop_1 =[];
    trise_1 = [];
    tfall_1 = [];
    tsum_1 = [];
    trigch_1 = [];
    trigpeakch_1=[];

    x1=0;
    y1=0;

    event = 0; % comes in pairs of traces
    while event < numtraces

        event=event+2;

        pmt=data(event-1,3:end);%get pmt pulse
        fpd=data(event,3:end);%photodiode trace
        pmt=pmt*-1;%invert the pmt pulse

        %calulates offset for the anodes (PMT)
        offset = PMT_Offset(pmt);
        pmt=pmt-offset;%apply the offset

        %calulates offset for the anodes (FPD)
        offset = FPD_Offset(fpd);
        fpd=fpd-offset;%apply the offset

        x1=data(event-1,1);%get the grid position
        y1=data(event,1);%get the grid position

        %All remaining calculations done using smoothed data
        pmt = smooth(pmt,smoothingfactor);
        fpd = smooth(fpd,smoothingfactor);

        %check for "bad" corner data points
        if max(pmt)<pmtthreshold || isnan(max(pmt)) == 1 ||
max(fpd)<fpdthreshold || isnan(max(fpd))==1
            continue;
        end


        [sum1,rise1,fall1,start1,stop1]=PMT_RiseTime(pmt);
        [tsum1,trise1,tfall1,trigch1,tstop1,trigpeakch1]=FPD_RiseTime(fpd);

        rise_1=[rise_1;rise1'];
        fall_1=[fall_1;fall1'];
        sum_1 = [sum_1;sum1'];
        trise_1=[trise_1;trise1'];
        tfall_1=[tfall_1;tfall1'];
        tsum_1 = [tsum_1;tsum1'];
        trigch_1=[trigch_1;trigch1'];
        trigpeakch_1=[trigpeakch_1;trigpeakch1'];
        start_1 =[start_1;start1];
        stop_1 =[stop_1;stop1];
        tstop_1 =[tstop_1;tstop1];

    end

    if isempty(rise_1)==1 || length(rise_1)<req %require that at least 7/10
events valid
        xy = [x1,y1];
        dlmwrite('notenoughrise.csv',xy,'-append')
        continue
    end

    rise=[rise;rise_1];
    fall=[fall;fall_1];
    sum = [sum;sum_1];
    start = [start;start_1];
    stop = [stop;stop_1];
    tstop = [tstop;tstop_1];
    trise=[trise;trise_1];
    tfall=[tfall;tfall_1];
    tsum = [tsum;tsum_1];
    trigch = [trigch;trigch_1];
    trigpeakch = [trigpeakch;trigpeakch_1];

end

of=[savename,'.rise'];
Arr = [rise,fall,sum,start,stop,trise,tfall,tsum,trigch,trigpeakch,tstop];
dlmwrite(of,Arr,'precision', 7);
fclose('all');

%Make Plots
hist(rise,30)
title('Rise Times Histogram in 100 ps Channels (PMT)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(start,30)
title('Start Times Histogram in 100 ps Channels (PMT)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(stop,30)
title('Stop Times Histogram in 100 ps Channels (PMT)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(fall,30)
title('Fall Times Histogram in 100 ps Channels (PMT)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(sum,30)
title('Sum Times Histogram in 100 ps Channels (PMT)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(trise,30)
title('Rise Times Histogram in 100 ps Channels (FPD)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(tfall,30)
title('Fall Times Histogram in 100 ps Channels (FPD)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause
```

```
hist(tsum,30)
title('Sum Times Histogram in 100 ps Channels (FPD)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(trigch,30)
title('Trigger Channel Location Start Times Histogram (FPD)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(tstop,30)
title('Trigger Channel Location Stop Times Histogram (FPD)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

hist(trigpeakch,30)
title('Trigger PEAK Channel Location Times Histogram (FPD)')
ylabel('Count')
xlabel('Time (100 ps channels)')
pause

close all;

end





function PMT_MakeSumFile(wdir)

global fpdthreshold pmtthreshold req datapoints smoothingfactor

savename=input('Enter PMT Summary Name - ','s');
wdir2=dir(wdir);
numberfiles=length(wdir2)-2;%two hidden unecessary files

%set arrays for complete sum file
tstart=[];tstop=[];tarea=[];tpeak=[];
start=[];stop=[];lstop=[];area=[];peak=[];dt=[];
x=[];y=[];timestep=[];larea=[];

%averages
atstart=[];atstop=[];atarea=[];atpeak=[];
astart=[];astop=[];alstop=[];aarea=[];apeak=[];adt=[];
alarea=[];ax=[];ay=[];

%variances
vtstart=[];vtstop=[];vtarea=[];vtpeak=[];
vstart=[];vstop=[];vlstop=[];varea=[];vpeak=[];vdt=[];
```

```
vlarea=[];vx=[];vy=[];

%Creating a summary file of events thrown out
res=zeros(1,datapoints);notenough=zeros(1,2)-2;
dlmwrite('noresponse.csv',res);
dlmwrite('notenough.csv',notenough);

%loop over each .pmt file and make a temp array of values
fileindex = 0;
while fileindex < numberfiles
    fileindex=fileindex+1;

    %find the next file and load it
    file=wdir2(fileindex+2).name;
    if ~strcmp(file(end-2:end),'pmt')%check for correct file type
        continue;
    end
    filename=[wdir,'/',file];
    fprintf(1,'\n Analyzing %s\n',filename);
    data=load(filename);%load the .pmt file chosen earlier

    %number of traces taken at each spatial location
    [numtraces,~] = size(data);

    %create temporary bins
    tstart_1 = [];
    tstop_1 = [];
    tarea_1 =[];
    tpeak_1 = [];
    start_1 =[];
    stop_1 = [];
    lstop_1= [];
    area_1 = [];
    peak_1 =[];
    dt_1 = [];
    x_1 = [];
    y_1 = [];
    timestep_1 = [];
    larea_1 =[];

    x1=0;
    y1=0;

    event = 0; % come in pairs of traces
    while event < numtraces

        event=event+2;

        pmt=data(event-1,3:end);%get pmt pulse
        fpd=data(event,3:end);%photodiode trace
        pmt=pmt*-1;%invert the pmt pulse

        %calulates offset for the anodes (PMT)
        offset = PMT_Offset(pmt);
        pmt=pmt-offset;%apply the offset
        %calulates offset for the anodes (FPD)
        offset = FPD_Offset(fpd);
        fpd=fpd-offset;%apply the offset

        x1=data(event-1,1);%get the grid position
        y1=data(event,1);%get the grid position
        timestep1=data(event-1,2);%get time interval spacing


        %All remaining calculations done using smoothed data
        pmt = smooth(pmt,smoothingfactor);
        fpd = smooth(fpd,smoothingfactor);


        %check for "bad" corner data points
        if max(pmt) < pmtthreshold|| isnan(max(pmt)) == 1 ||
max(fpd)<fpdthreshold || isnan(max(fpd))==1
            xpmt = [x1,event/2,pmt'];
            yfpd = [y1,event/2,fpd'];
            dlmwrite('noresponse.csv',xpmt,'-append')
            dlmwrite('noresponse.csv',yfpd,'-append')
            continue;
        end



        [area1,larea1,start1,stop1,lstop1,maxpeak1] = PMT_PeakSub(pmt);
        [tarea1,tstart1,tstop1,tmaxpeak1] = FPD_PeakSub(fpd);

        dt1 = start1 - tstart1;

        tstart_1=[tstart_1;tstart1];
        tstop_1 = [tstop_1;tstop1];
        tarea_1=[tarea_1;tarea1];
        tpeak_1=[tpeak_1;tmaxpeak1];
        start_1=[start_1;start1];
        stop_1=[stop_1;stop1];
        lstop_1=[lstop_1;lstop1];
        area_1=[area_1;area1];
        peak_1=[peak_1;maxpeak1];
        dt_1=[dt_1;dt1];
        x_1 = [x_1;x1];
        y_1 = [y_1;y1];
        timestep_1 =[timestep_1;timestep1];
        larea_1 = [larea_1;larea1];

    end

    if isempty(dt_1)==1 || length(dt_1)<req %require that at least 7/10
events valid
        xy = [x1,y1];
        dlmwrite('notenough.csv',xy,'-append')
        continue
    end
    tstart=[tstart;tstart_1];
    tstop = [tstop;tstop_1];
    tarea=[tarea;tarea_1];
    tpeak=[tpeak;tpeak_1];
    start=[start;start_1];
    stop=[stop;stop_1];
    lstop=[lstop;lstop_1];
    area=[area;area_1];
    peak=[peak;peak_1];
    dt=[dt;dt_1];
    x = [x;x_1];
    y = [y;y_1];
    timestep =[timestep;timestep_1];
    larea = [larea;larea_1];


    atstart=[atstart;mean(tstart_1)];
    atstop = [atstop;mean(tstop_1)];
    atarea=[atarea;mean(tarea_1)];
    atpeak=[atpeak;mean(tpeak_1)];
    astart=[astart;mean(start_1)];
    astop=[astop;mean(stop_1)];
    alstop=[alstop;mean(lstop_1)];
    aarea=[aarea;mean(area_1)];
    apeak=[apeak;mean(peak_1)];
    adt=[adt;mean(dt_1)];
    alarea = [alarea;mean(larea_1)];
    ax =[ax;mean(x_1)];
    ay =[ay;mean(y_1)];

    vtstart=[vtstart;var(tstart_1)];
    vtstop = [vtstop;var(tstop_1)];
    vtarea=[vtarea;var(tarea_1)];
    vtpeak=[vtpeak;var(tpeak_1)];
    vstart=[vstart;var(start_1)];
    vstop=[vstop;var(stop_1)];
    vlstop=[vlstop;var(lstop_1)];
    varea=[varea;var(area_1)];
    vpeak=[vpeak;var(peak_1)];
    vdt=[vdt;var(dt_1)];
    vlarea = [vlarea;var(larea_1)];
    vx =[vx;var(x_1)];
    vy =[vy;var(y_1)];

end

%Making two different types of vectors the same length for writing to file
kk = length(x)-length(atstart);
addstuff = zeros(kk,1)-1;

atstart=[atstart;addstuff];
atstop = [atstop;addstuff];
atarea=[atarea;addstuff];
atpeak=[atpeak;addstuff];
```

```
astart=[astart;addstuff];
astop=[astop;addstuff];
alstop=[alstop;addstuff];
aarea=[aarea;addstuff];
apeak=[apeak;addstuff];
adt=[adt;addstuff];
alarea = [alarea;addstuff];
ax = [ax;addstuff];
ay = [ay;addstuff];

vtstart=[vtstart;addstuff];
vtstop = [vtstop;addstuff];
vtarea=[vtarea;addstuff];
vtpeak=[vtpeak;addstuff];
vstart=[vstart;addstuff];
vstop=[vstop;addstuff];
vlstop=[vlstop;addstuff];
varea=[varea;addstuff];
vpeak=[vpeak;addstuff];
vdt=[vdt;addstuff];
vlarea = [vlarea;addstuff];
vx = [vx;addstuff];
vy = [vy;addstuff];

%writing to file
of=[savename,'.csv'];
Arr =
[x,y,tstart,tstop,tarea,tpeak,start,stop,lstop,area,peak,dt,timestep,larea,at
start,atstop,atarea,atpeak,astart,astop,alstop,aarea,apeak,adt,alarea,ax,ay,v
tstart,vtstop,vtarea,vtpeak,vstart,vstop,vlstop,varea,vpeak,vdt,vlarea,vx,vy]
;
dlmwrite(of,Arr,'precision', 7);
fclose('all');

end







function [offset]=PMT_Offset(data)

earlytime = 1000; %Choosen by reviewing typical events (very safe margin)
offset = mean(data(1:earlytime));
```

```
function PMT_PeakFinder(wdir)

global fpdthreshold pmtthreshold req smoothingfactor

savename=input('Enter PeakTime Summary Name - ','s');
wdir2=dir(wdir);
numberfiles=length(wdir2)-2;%two hidden unecessary files

%set arrays for complete sum file
tstart=[];tstop=[];dt=[];diff=[];
x=[];y=[];

%averages
atstart=[];atstop=[];adt=[];adiff=[];
ax=[];ay=[];

%variances
vtstart=[];vtstop=[];vdt=[];vdiff=[];
vx=[];vy=[];

%loop over each .pmt file and make a temp array of values
fileindex = 0;
while fileindex < numberfiles
    fileindex=fileindex+1;

    %find the next file and load it
    file=wdir2(fileindex+2).name;
    if ~strcmp(file(end-2:end),'pmt')%check for correct file type
        continue;
    end
    filename=[wdir,'/',file];
    fprintf(1,'\n Analyzing %s\n',filename);
    data=load(filename);%load the .pmt file chosen earlier

    %number of traces taken at each spatial location
    [numtraces,~] = size(data);

    %create temporary bins
    tstart_1 = [];
    tstop_1 = [];
    dt_1 = [];
    diff_1=[];

    x_1 = [];
    y_1 = [];


    event = 0; % come in pairs of traces
    while event < numtraces

        event=event+2;

        pmt=data(event-1,3:end);%get pmt pulse
        fpd=data(event,3:end);%photodiode trace
        pmt=pmt*-1;%invert the pmt pulse

        %calulates offset for the anodes (PMT)
        offset = PMT_Offset(pmt);
        pmt=pmt-offset;%apply the offset

        %calulates offset for the anodes (FPD)
        offset = FPD_Offset(fpd);
        fpd=fpd-offset;%apply the offset

        x1=data(event-1,1);%get the grid position
        y1=data(event,1);%get the grid position


        %All remaining calculations done using smoothed data
        pmt = smooth(pmt,smoothingfactor);
        fpd = smooth(fpd,smoothingfactor);

        %check for "bad" corner data points
        if max(pmt)< pmtthreshold || isnan(max(pmt)) == 1 ||
max(fpd)<fpdthreshold || isnan(max(fpd))==1
            continue;
        end



        [~,~,start1,~,~,~] = PMT_PeakSub(pmt);
        [~,tstart1dt,~,~] = FPD_PeakSub(fpd);

        dt1 = start1 - tstart1dt;



        [~,loc] =
findpeaks(pmt,'SORTSTR','descend','MINPEAKDISTANCE',100,'MINPEAKHEIGHT',pmtth
reshold);

        %ensures that is really a PRE-bump not a post bump
        if length(loc)>1
            check = loc(1)-loc(2);
            if check>0
                diff1 = loc(1)-loc(2);
            else
                continue
            end
        else
            continue %no pre-bump present
        end

        tstart_1=[tstart_1;loc(2)];
        tstop_1 = [tstop_1;loc(1)];
        diff_1=[diff_1;diff1];

        dt_1=[dt_1;dt1];
        x_1 = [x_1;x1];
        y_1 = [y_1;y1];
```

```
    end

    if isempty(dt_1)==1 || length(dt_1)<req %require that at least 7/10
events valid
        continue
    end

    tstart=[tstart;tstart_1];
    tstop = [tstop;tstop_1];
    diff = [diff;diff_1];

    dt=[dt;dt_1];
    x = [x;x_1];
    y = [y;y_1];



    atstart=[atstart;mean(tstart_1)];
    atstop = [atstop;mean(tstop_1)];
    adiff = [adiff;mean(diff_1)];

    adt=[adt;mean(dt_1)];

    ax =[ax;mean(x_1)];
    ay =[ay;mean(y_1)];

    vtstart=[vtstart;var(tstart_1)];
    vtstop = [vtstop;var(tstop_1)];
    vdiff = [vdiff;var(diff_1)];

    vdt=[vdt;var(dt_1)];

    vx =[vx;var(x_1)];
    vy =[vy;var(y_1)];

end

%Making two different types of vectors the same length for writing to file
kk = length(x)-length(atstart);
addstuff = zeros(kk,1)-1;

atstart=[atstart;addstuff];
atstop = [atstop;addstuff];
adiff = [adiff;addstuff];

adt=[adt;addstuff];

ax = [ax;addstuff];
ay = [ay;addstuff];

vtstart=[vtstart;addstuff];
```

```
vtstop = [vtstop;addstuff];
vdiff = [vdiff;addstuff];

vdt=[vdt;addstuff];

vx = [vx;addstuff];
vy = [vy;addstuff];

%writing to file
of=[savename,'.csv'];
Arr =
[x,y,tstart,tstop,dt,atstart,atstop,adt,adiff,ax,ay,vtstart,vtstop,vdt,vdiff,
vx,vy];
dlmwrite(of,Arr,'precision', 7);
fclose('all');

end





function [area,larea,start,stop,lstop,maxpeak]=PMT_PeakSub(smoothdata)

global pmtoffset pmtcfd

[maxpeak,peakch]=max(smoothdata);
full=length(smoothdata);

while peakch > full-1000 %the peak can't be in last 1000 channels. If it is,
    smoothdata(peakch:full)=0; %zero it out and look for another peak
    [maxpeak,peakch]=max(smoothdata);
end

%Insert peak time offset
timedelay=pmtoffset;%Time Delay in Number of Channels
delay=zeros(timedelay,1);%Zeros because already subtracted offset

delayed=[delay.',smoothdata.'];%Add Time Delay at start of wave trace

%Create inverted attenuated peak to add to delayed original peak
invert_atten=[(-pmtcfd).*(smoothdata.'),delay.'];

data = delayed + invert_atten; %final data set for calculating timing

start=500; %arbitrary start value for zero crossing point
for i = (peakch + timedelay):-1:10

    if data(i) < 0; %zero crossing

        start=i;%Chanel of Zero Crossing
        break;

    end

end


stop=1500;%Arbitrary end point in timing
for i = peakch:+1:full

    if smoothdata(i)< pmtcfd*maxpeak;%Arbitrary level for end time

        stop=i;
        break;

    end

end

lstop=1900;%Arbitrary end point in timing
for i = peakch:+1:full

    if smoothdata(i)< 0.05*maxpeak;%Arbitrary level for end time

        lstop=i;
        break;

    end

end

%Subtracting back time delay added to pulse at start of pulse analysis
start = start - timedelay;%This subtraction should be the same for all
area=sum(smoothdata(start:stop)); %total area
larea=sum(smoothdata(start:lstop)); %late area

end







function PMT_plotRawData(wdir,file1)

global fpdthreshold pmtthreshold smoothingfactor xcenter ycenter

data=load([wdir,'\',file1]);%load the .pmt file chosen earlier

%The data files are saved in the following format
% x-pos | dt | PMT data points
% y-pos | dt | FPD data points
```

```
%number of data points per trace
datapoints = length(data)-2; %(first index position; second is timestep)
[numtraces,~] = size(data);  %total number of traces at each spatial loc.

time=1:datapoints;% in 100 ps channels
timens=time.*(0.1);% in nanosecond channels

event = 2; % comes in pairs of traces
while event < numtraces + 1 %plot each trace combination

    pmt=data(event-1,3:end);%pmt trace
    fpd=data(event,3:end);%photodiode trace
    pmt=pmt*-1;%invert the pmt pulse

    %calulates offset for the anodes (PMT)
    offset = PMT_Offset(pmt);
    pmt=pmt-offset;%apply the offset

    %calulates offset for the fast photodiode (FPD)
    offset = FPD_Offset(fpd);
    fpd=fpd-offset;%apply the offset

    %smooth data set
    spmt = smooth(pmt,smoothingfactor);
    sfpd = smooth(fpd,smoothingfactor);

    x=data(event-1,1);%get the grid position
    y=data(event,1);%get the grid position

    %check for "bad" corner data points
    if max(spmt)< pmtthreshold || isnan(max(spmt)) == 1 ||
max(sfpd)<fpdthreshold || isnan(max(sfpd))==1

        close all;%close previous figures
        hh=figure;
        set(hh,'Units','normalized');
        set(hh,'OuterPosition',[.5 0 .5 .9])
        subplot(2,1,1)
        plot(timens,fpd,'r',timens,sfpd,'b')
        legend('FPD','SmoothFPD')
        ylabel('V')
        xlabel('time (nansecond channels)')
        t=sprintf('FPD Trace %g of %g at
(X,Y)=(%g,%g)',event/2,numtraces/2,x-xcenter,y-ycenter);
        title(t)
        subplot(2,1,2)
        plot(timens,pmt,'m',timens,spmt,'c')
        legend('PMT','SmoothPMT')
        ylabel('V')
        xlabel('time (nansecond channels)')
        t=sprintf('PMT Response %g of %g at
(X,Y)=(%g,%g)',event/2,numtraces/2,x,y);
        title(t)
    else

        [area,larea,start,stop,~,maxpeak]= PMT_PeakSub(spmt);
        [tarea,trigch,tstop,trigamp] = FPD_PeakSub(sfpd);
        dt = (start - trigch)*.1;%nanosecond scale
        rarea = area/tarea;%relative area response

        close all;%close previous figures
        hh=figure;
        set(hh,'Units','normalized');
        set(hh,'OuterPosition',[.5 0 .5 .9])
        plot(timens,sfpd,'r',timens,spmt,'b')
        legend('FPD','PMT')
        ylabel('V')
        xlabel('Time (nansecond channels)')
        t=sprintf('FPD & PMT Waveform %g of %g at
(X,Y)=(%g,%g)',event/2,numtraces/2,x-xcenter,y-ycenter);
        title(t)

        hold on;
        plot([start*.1,start*.1],[0 maxpeak*1.1],'g')
        plot([trigch*.1,trigch*.1],[0 maxpeak*1.1],'g')
        hold off;

        fprintf('Transit time (ns) = %g \n Relative Area = %g \n',dt,rarea)
        fprintf('Start = %g \n Stop = %g \n',start,stop)
        fprintf('Area = %g \n Late Area = %g Max = %g \n',area,larea,maxpeak)
        fprintf('Triger Channel Start = %g \n Triger Channel Stop = %g \n
Trigger Peak = %g \n',trigch,tstop,trigamp)
    end

    %iterate & check if user wants to continue
    event = event + 2;
    ostr2='Press Enter to continue or enter any character to end> ';%pause
    resp=input(ostr2,'s');%get user input to continue
    if strcmp(resp,''); continue;%continue or quit depending on user input
    else break; end;

end
clc;%clear command window to make it look nice again
```

```
function PMT_PreScan(wdir,file1)

global fpdthreshold pmtthreshold datapoints smoothingfactor

%set arrays for specific data to plot
dt=[];trigloc=[];trigpeak=[];trigarea=[];peak=[];area=[];rel=[];
```

```
filename=[wdir,'/',file1];
fprintf(1,'\n Analyzing %s\n',filename);
data=load(filename);%load the .pmt file chosen earlier

%number of traces at each spatial location
[numtraces,~] = size(data);

%Creating a summary file of events thrown out
res = 2 + zeros(1,datapoints); %2 used as an identifier
dlmwrite('thrownoutres.csv',res);

event = 0; % come in pairs of traces
while event < numtraces

    event=event+2;

    pmt=data(event-1,3:end);%get pmt pulse
    fpd=data(event,3:end);%photodiode trace
    pmt=pmt*-1;%invert the pmt pulse

    %calulates offset for the anodes (PMT)
    offset = PMT_Offset(pmt);
    pmt=pmt-offset;%apply the offset

    %calulates offset for the anodes (FPD)
    offset = FPD_Offset(fpd);
    fpd=fpd-offset;%apply the offset

    x1=data(event-1,1);%get the grid position
    y1=data(event,1);%get the grid position


    %All remaining calculations done using smoothed data
    pmt = smooth(pmt,smoothingfactor);
    fpd = smooth(fpd,smoothingfactor);

    %check for "bad" corner data points
    if max(pmt)< pmtthreshold || isnan(max(pmt)) == 1 ||
max(fpd)<fpdthreshold|| isnan(max(fpd))==1
        xpmt = [x1,event/2,pmt'];
        yfpd = [y1,event/2,fpd'];
        dlmwrite('thrownoutres.csv',xpmt,'-append')
        dlmwrite('thrownoutres.csv',yfpd,'-append')
        continue;
    end


    [area1,~,start1,~,~,peak1] = PMT_PeakSub(pmt);
    [trigarea1,tstart1,~,trigpeak1] = FPD_PeakSub(fpd);

    dt1 = start1 - tstart1;

    rel1=peak1/trigpeak1;
    dt = [dt;dt1];
    trigloc = [trigloc;tstart1];
    trigpeak = [trigpeak;trigpeak1];
    trigarea = [trigarea;trigarea1];
    peak = [peak;peak1];
    area = [area;area1];
    rel = [rel;rel1];


end

relarea = area./trigarea;

%transit time plot
ave = mean(dt);
timestep1=data(event-2,2);%get time interval spacing
saved = length(dt);
total = (event)/2;
h=hist(dt,20);
hh=figure (1);
set(hh,'Units','normalized');
set(hh,'OuterPosition',[.3 .3 .6 .6])
hist(dt,20)
hold on;
t=sprintf('Timing Spread (Jitter). X=%g Y=%g dt=%g Saved=%g/%g
Mean=%g',x1,y1,timestep1,saved,total,ave);
title(t)
ylabel('Count')
xlabel('Time (100 ps channels)')
height = max(h)*1.1;
plot([ave,ave],[0 height],'g')
hold off

%trigger location plot
hhh=figure(2);
set(hhh,'Units','normalized');
set(hhh,'OuterPosition',[.3 .3 .6 .6])
hist(trigloc,100)
tt=sprintf('Trigger Location X=%g Y=%g dt=%g
Saved=%g/%g',x1,y1,timestep1,saved,total);
title(tt)
ylabel('Count')
xlabel('Time (100 ps channels)')

%trigger peak histogram
ha=figure(3);
set(ha,'Units','normalized');
set(ha,'OuterPosition',[.3 .3 .6 .6])
hist(trigpeak,100)
tta=sprintf('Trigger Peak Height X=%g Y=%g dt=%g
Saved=%g/%g',x1,y1,timestep1,saved,total);
title(tta)
ylabel('Count')
xlabel('(V)')
```

```
%response peak histogram
hat=figure(4);
set(hat,'Units','normalized');
set(hat,'OuterPosition',[.3 .3 .6 .6])
hist(peak,100)
ttat=sprintf('PMT Peak Height X=%g Y=%g dt=%g
Saved=%g/%g',x1,y1,timestep1,saved,total);
title(ttat)
ylabel('Count')
xlabel('(V)')


%relative response histogram
hata=figure(5);
set(hata,'Units','normalized');
set(hata,'OuterPosition',[.3 .3 .6 .6])
hist(rel,100)
ttata=sprintf('Relative Response Peak Height (PMT/FPD) X=%g Y=%g dt=%g
Saved=%g/%g',x1,y1,timestep1,saved,total);
title(ttata)
ylabel('Count')


%relative response histogram
hata=figure(6);
set(hata,'Units','normalized');
set(hata,'OuterPosition',[.3 .3 .6 .6])
hist(relarea,100)
ttata=sprintf('Relative Response Area(PMT/FPD) X=%g Y=%g dt=%g
Saved=%g/%g',x1,y1,timestep1,saved,total);
title(ttata)
ylabel('Count')


pause

end







function [sum,rise,fall,start,stop]=PMT_RiseTime(smoothdata)

global pmtcfd

[peakmax,ch]=max(smoothdata);
full=length(smoothdata);

while ch>full-1000 %the peak can't be in the last 1000 channels so if it is,
    smoothdata(ch:full)=0; %it zeros it out and looks for another peak
    [peakmax,ch]=max(smoothdata);
end

start=10; %arbitrary start channel
for i=ch:-1:5 %Finding the start time of the rising edge

    if smoothdata(i) < pmtcfd*peakmax;

        start=i;
        break;

    end

end

rise=ch-start;

stop=1500; %arbitrary stop channel
    for i=ch:+1:full

        if smoothdata(i)< pmtcfd*peakmax;

            stop=i;
            break;

        end

    end

fall=stop-ch;
sum = stop-start;
```

ACKNOWLEDGMENTS

I would like to acknowledge all students who worked previously on this project, along with all student who have directly worked with me on this project: Nirdosh Champaign, Kent Talbert, Trevor Jex, Nicholas Harrison, and John Peterson. I would also like to thank John Ellsworth and Bart Czirr for their technical help with experimental design and oscilloscope operation.

A special acknowledgement to Dr. Michael Ware for the use of his lab and all lab equipment, along with Dr. Justin Peatross for allowing the frequent use of his laser. This project would have been impossible without their generous contributions.

Lastly, I would like to acknowledge Dr. Lawrence Rees for his highly valuable feedback, direction, and mentorship. He taught me how to approach any experimental problem, produce a valid result, and then question that result for continued learning and insight.