2011

Determination of a solid propellant's burn rate using a new digital burn rate analysis method

Ashley F. Penton

DETERMINATION OF A SOLID PROPELLANT'S BURN RATE USING A NEW DIGITAL BURN RATE ANALYSIS METHOD

by

ASHLEY F. PENTON

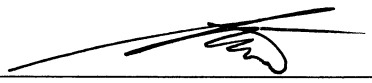
A THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in
The Department of Mechanical and Aerospace Engineering
to
The School of Graduate Studies
of
The University of Alabama in Huntsville

HUNTSVILLE, ALABAMA

2011

In presenting this thesis in partial fulfillment of the requirements for a master's degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/ her absence, by the Chair of the Department or the Dean of the School of Graduate Studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.



10-31-11

Ashley F. Penton

Date

THESIS APPROVAL FORM

Submitted by Ashley F. Penton in partial fulfillment of the requirements for the degree of Master of Science in Engineering in Mechanical and Aerospace Engineering and accepted on behalf of the Faculty of the School of Graduate Studies by the thesis committee.

We, the undersigned members of the Graduate Faculty of The Univeristy of Alabama in Huntsville, certify that we have advised and/ or supervised the candidate of the work described in this thesis. We further certify that we have reviewed the thesis manuscript and approve it in partial fulfillment of the requirements for the degree of Master of Science in Engineering in Mechanical and Aerospace Engineering.

Robert A Frederick 10/31/11 Committee Chair

Dr. Robert A. Frederick *Date*

H W Coleman 11/1/11

Dr. Hugh W. Coleman *Date*

10/31/11

Dr. David M. Lineberry *Date*

11/1/11 Department Chair

Date

11/01/11 College Dean

Dr. Shankar Mahalingam *Date*

Rhonda Kay Gaede 12/2/11 Graduate Dean

Date

ABSTRACT

School of Graduate Studies
The University of Alabama in Huntsville

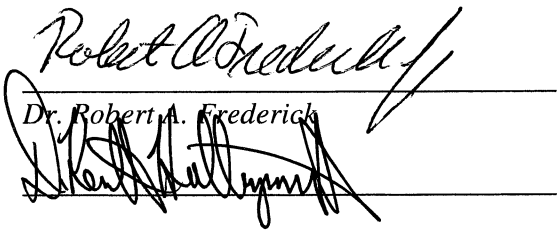
Degree Master of Science in Engineering College/ Dept. Engineering/ Mechanical and Aerospace Engineeing

Name of Candidate Ashley F. Penton

Title Determination of a Solid Propellant's Burn Rate Using a New Digital Burn Rate Analysis Method

The work presented here is a study of three solid propellant formulations tested at two temperatures to determine the burn rates at varying pressures using a new digital posttest analysis method, the New Zero Crossing Method. The same tests were also analyzed by two previously developed digital posttest analysis methods, the Original Zero Crossing Method and the Cross Correlation Method. It was found that, due to changes in the data collection system, the Original Zero Crossing Method is no longer a valid method, while the New Zero Crossing Method, along with the Cross Correlation Method, are valid and comparable to one another. Uncertainty analysis is performed on each test. The relative uncertainty of each test is below 10%.

Abstract Approval:

Committee Chair 
Dr. Robert A. Frederick

Department Chair

Graduate Dean 

ACKNOWLEDGMENTS

I would like to thank the NASA Constellation Virtual Institute Project for the funding of this project. I would also like to thank Seth Thompson for lending his Matlab expertise to my aid and Chrystine Gorman for volunteering her time so that I could conduct these experiments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AMRDEC	=	Aviation and Missile Research Development and Engineering Center
AP	=	Ammonium Perchlorate
EDUM	=	Electronic Device for Ultrasonic Measurement
HTPB	=	Hydroxl- Terminated Polybutadiene

LIST OF SYMBOLS

a	=	temperature coefficient
a_0	=	temperature coefficient at initial temperature
a_c	=	slope from regression of coupling material
a_p	=	slope from regression of propellant
b_c	=	y-intercept from regression of coupling material
b_p	=	y-intercept from regression of the propellant
c	=	speed of sound
E_p	=	instantaneous thickness of propellant
E_{p0}	=	initial thickness of propellant
$°F$	=	Fahrenheit
in	=	inches
ips	=	inches per second
n	=	burn rate exponent
p	=	chamber pressure
P	=	pressure
psi	=	pounds per square inch
r	=	burn rate of propellant
r_b	=	burn rate of propellant
s	=	seconds
t	=	thickness
T	=	temperature
T_0	=	initial temperature

V	=	volts
σ_p	=	temperature sensitivity of burn rate at constant pressure
τ	=	propagation time

To my Mother and Father, for always telling me that I could be whatever I wanted. To my sister, Rachel, for always blazing the trail ahead of me. And to my love, Zach, for inspiring me to finish what I start.

CHAPTER I

INTRODUCTION

The objective of this work is to investigate a new digital burn rate analysis method in conjunction with the ultrasonic pulse-echo technique.

1.1 Description of Burn Rate of Solid Propellants

The burn rate of solid propellants is one of its most important ballistic characteristics. It greatly affects the performance of a solid motor using that propellant. In designing a solid motor the burn rate must be known under a variety of conditions. Chamber pressure is one of these conditions with significant influence over the burn rate characteristics. The burn rate as a function of pressure can be approximated by the equation,

$$r = ap^n, \tag{1.1}$$

where r is the burn rate, p is the chamber pressure, a is the temperature coefficient, and n is the burn rate exponent. The temperature coefficient and pressure exponent are empirical constants that are generally controlled by changing the propellant formulation or ingredient particle diameters [1].

The burn rate of solid propellants depends greatly on the formulation of the propellant. Two ingredients of note are Ammonium Perchlorate (AP) and Iron Oxide. AP, a commonly used oxidizer, influences burn rate depending on the AP particle size and shape. Iron Oxide, a burn-rate modifier, is known to increase the burn rate when present in a solid propellant [1].

1.2 Description of Temperature Sensitivity of Solid Propellants

The temperature of a solid propellant also affects its burn rate. With a higher initial temperature, there is more energy to release and therefore a higher burn rate. The temperature sensitivity, σ_p, of a solid propellant is the change in burn rate per degree temperature. The equation for the temperature sensitivity of burn rate is

$$\sigma_p = \left(\frac{\partial \ln r}{\partial T} \right) \Bigg|_p , \tag{1.2}$$

where r is the burn rate, T is the temperature, and p is chamber pressure.

The complete burn rate equation including temperature sensitivity, σ_p, is

$$r = a_0 {}^{\sigma_p \cdot (T - T_0)} p^n , \tag{1.3}$$

where r is the burn rate, p is the chamber pressure, a is the temperature coefficient, n is the burn rate exponent, and T is the temperature.

1.3 Description of the Ultrasonic Pulse-Echo Technique

The ultrasonic pulse-echo technique has been used extensively for the determination of burn rates of solid propellants. This technique was first conceived by Osborn and Ho at Virginia Tech in 1965 [2]. It was then developed and used by ONERA in France [3-6]. Since then it has been applied at the University of Alabama in Huntsville [3, 7-13], Pennsylvania State University [14], and the University of Illinois at Urbana-Champaign [15-19]. Figure 1.1 shows a schematic of the ultrasonic technique.

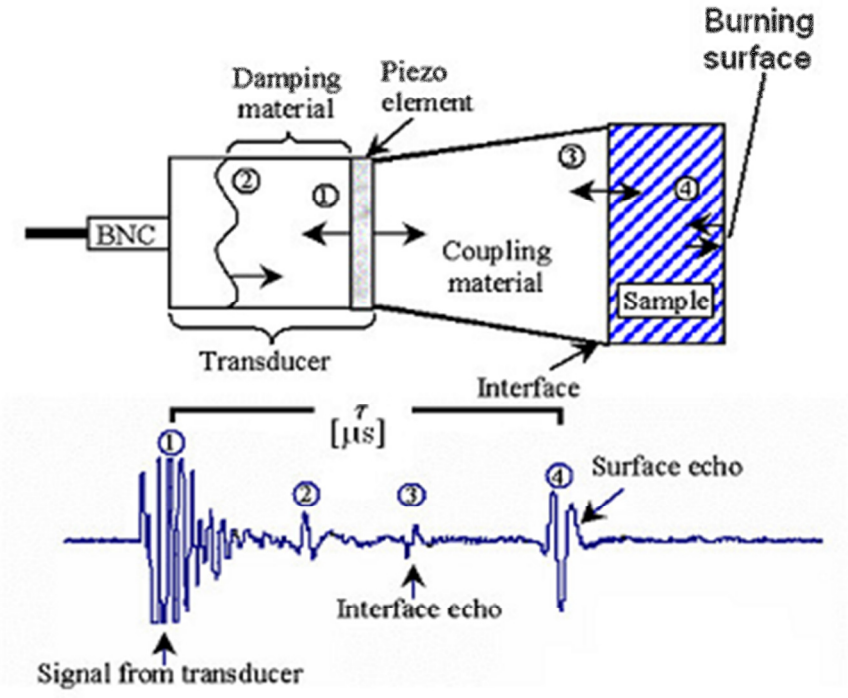


Figure 1.1: Ultrasonic Pulse-Echo Schematic and Sample Waveform

An ultrasonic transducer emits an acoustic pulse which propagates through the setup reflecting whenever it encounters an acoustic impedance, which happens when there is a change in material. The transducer then detects these reflected waves. An example of a waveform produced by this propagating wave can be seen below the ultrasonic setup in Figure 1.1. Peak 1 is a reflection of the pulse from the back on the transducer. Peak 2 is a reflection of the pulse from the interface of the transducer and the coupling material used. Peak 3 is a reflection of the pulse from the interface of the coupling material and the propellant sample. Finally, peak 4, is a reflection of the pulse from the sample surface. The propagation time for the propellant is the time taken for the wave to travel through the propellant. The first y-intercept after the first peak of peak 4 corresponds to the surface of the sample [20]. As the sample is burned, the surface peak, peak 4, moves to the left in successive waveforms because the surface of the sample is getting closer to the transducer. This trend can be seen in the following Figures 1.2 through 1.8.

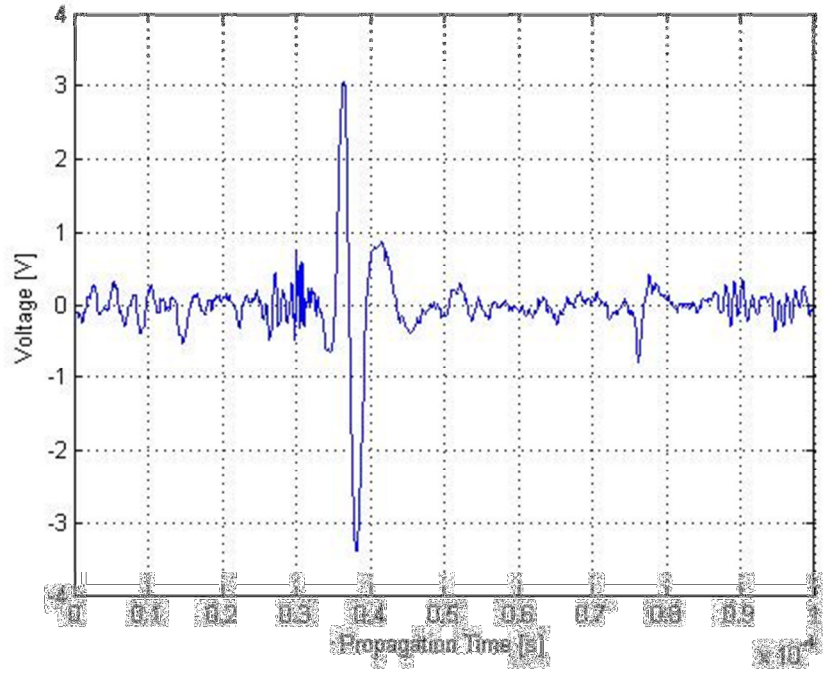


Figure 1.2: Test #1, GM08-20 @ 75°F, at 0.0 Seconds

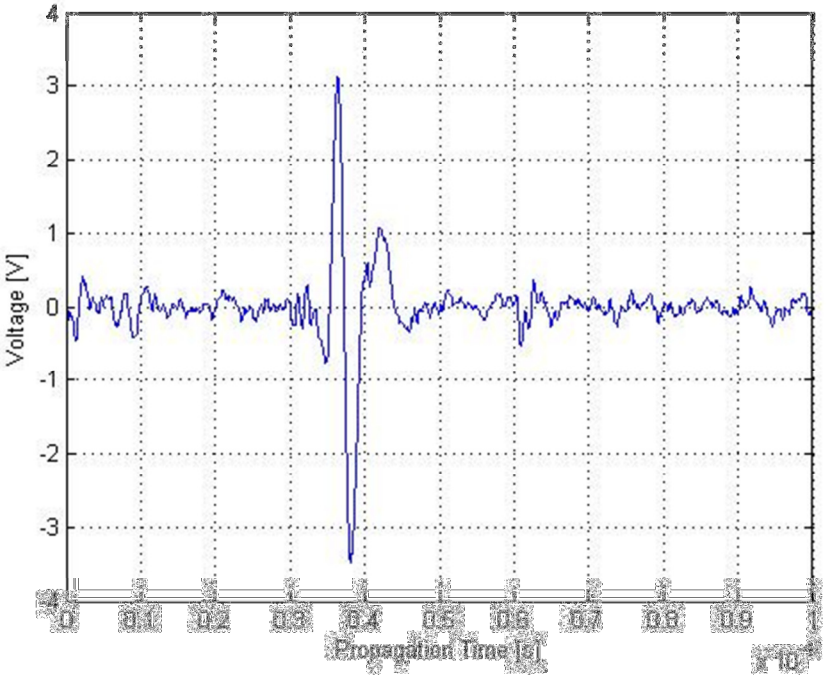


Figure 1.3: Test #1, GM08-20 @ 75°F, at 0.5 Seconds

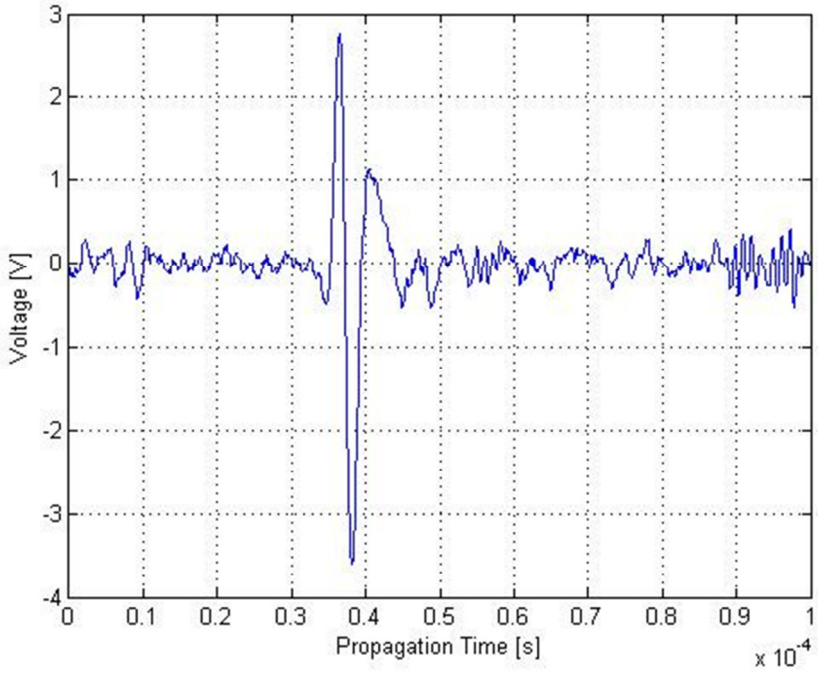


Figure 1.4: Test #1, GM08-20 @ 75°F, at 1.0 Seconds

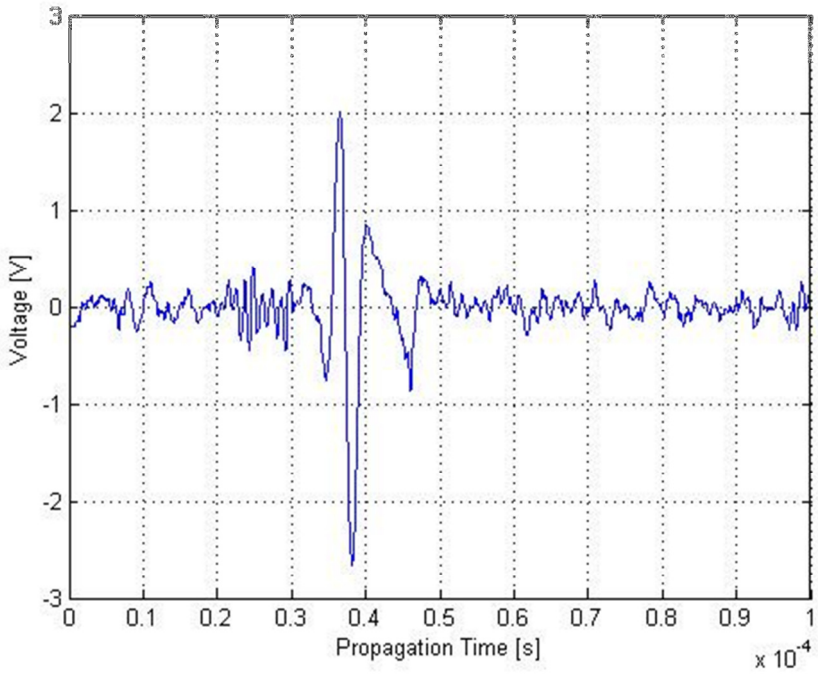


Figure 1.5: Test #1, GM08-20 @ 75°F, at 1.5 Seconds

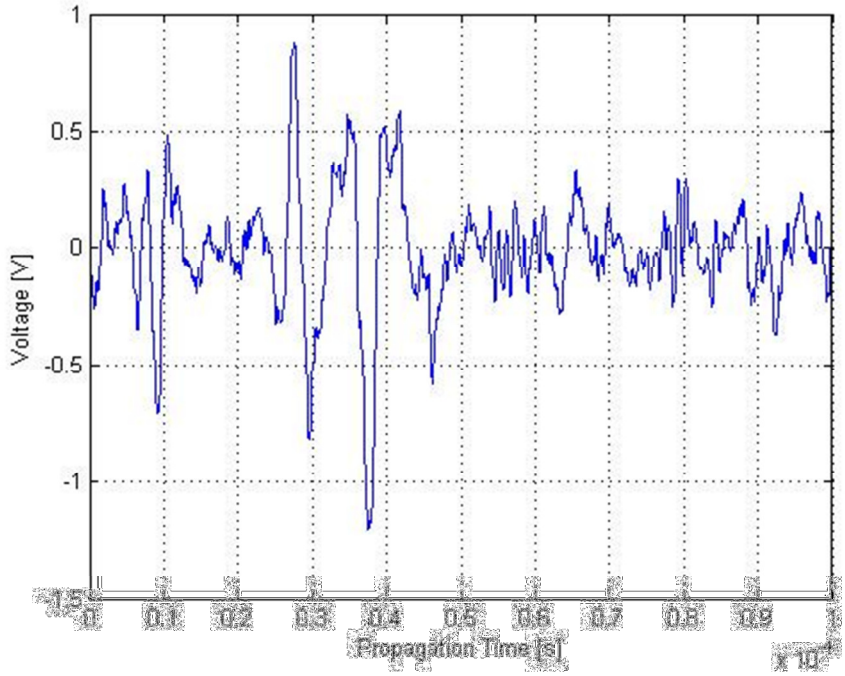


Figure 1.6: Test #1, GM08-20 @ 75°F, at 2.0 Seconds

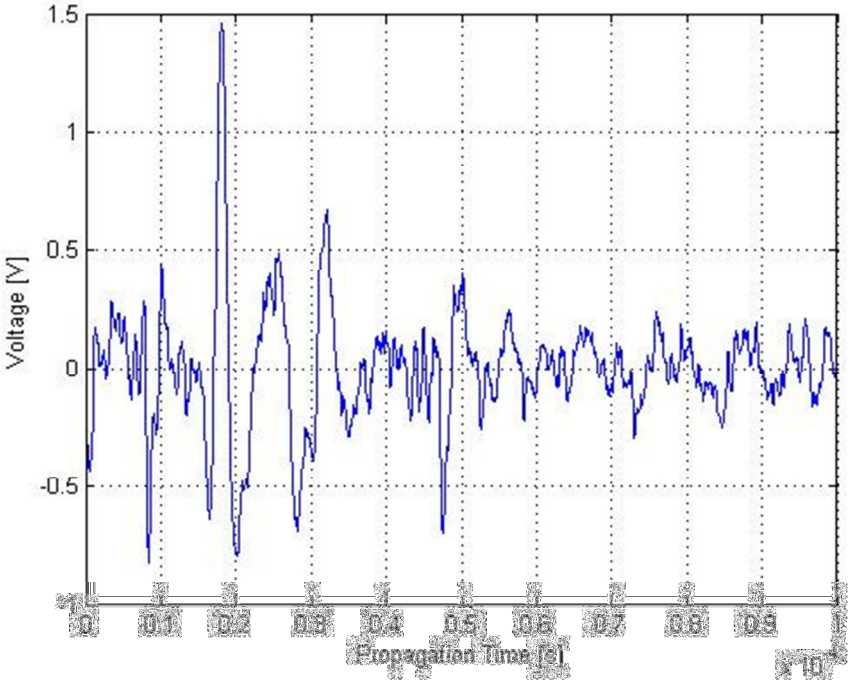


Figure 1.7: Test #1, GM08-20 @ 75°F, at 2.5 Seconds

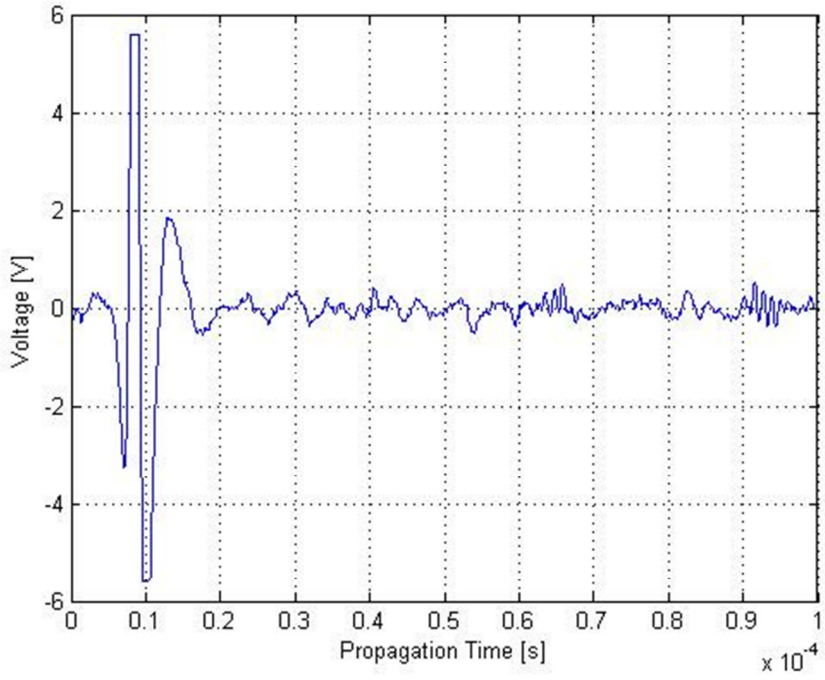


Figure 1.8: Test #1, GM08-20 @ 75°F, at 3.0 Seconds

The propagation time for a single material is related to the thickness and speed of sound by the equation,

$$t = \frac{c \cdot \tau}{2}, \tag{1.4}$$

where t is the thickness, c is the speed of sound, and τ is the propagation time.

The speed of sound in the propellant and coupling material is a function of both pressure and temperature. The effect of pressure is determined in pretest and posttest calibrations. The effect of temperature is neglected. Equation 1.4 can be solved for the thickness of the propellant then allowing the instantaneous burn rate of the propellant to be determined, as seen in Equation 1.5,

$$E_p = \frac{E_{p0}}{a_p \cdot P + b_p}\left[\tau - \left(a_c \cdot P + b_c\right)\right], \tag{1.5}$$

where E_p is the instantaneous thickness, E_{p0} is the initial thickness, a_p and b_p are the slope and y-intercept, respectively, from the regression of the propellant, P is the pressure, and a_c and b_c are the slope and y-intercept, respectively, from the regression of the coupling material. The derivative of this curve with respect to time is the instantaneous burn rate.

1.4 Previous Work on Solid Propellant Burn Rates at The University of Alabama in Huntsville

Just prior to and during this research, related research on solid propellant burn rates took place at The University of Alabama in Huntsville. Marshall [21] conducted experiments on burn rate determination comparing the older EDUM method with two new digital methods, Zero Crossing (henceforth known as the Original Zero Crossing Method) and Cross Correlation. He tested solid propellant samples and analyzed the same experiments with the three methods then compared the results. His results showed that the two new digital methods' results, which were near identical to each other, were within 5% of the EDUM method results, thereby validating the use of the digital methods. His results also suggested that the new digital methods are more reliable than the EDUM method as the EDUM periodically loses its signal, thereby losing data. His Original Zero Crossing method is the basis for the method tested in this investigation. Changes were made to the algorithm to make the code more robust, but the concept remains the same.

In addition to Marshall's work, Evans [23] did in-depth uncertainty analysis on the pulse-echo technique used at The University of Alabama in Huntsville. He investigated the uncertainty of the three methods Marshall used, the EDUM method, the Original Zero Crossing method, and the Cross Correlation method. His analysis concluded that all three methods produced similar burn rates and relative uncertainties.

The results of the tests performed in this investigation will be compared to both the Original Zero Crossing Method as well as the Cross Correlation Method. As these methods were validated by comparing them to the analog EDUM method, validation of the New Zero Crossing Method can be gained by comparison to these methods.

CHAPTER II

EXPERIMENTAL SETUP

2.1 Propellant

The composite propellant used was a bimodal AP, 200 micron and 16 micron, propellant with HTPB as a binder. The propellant was made in a one gallon mix and cast into 3 1.25 inch diameter tubes. Each tube was cut into 0.5 in samples, approximately 20 grams, and labeled with the tube, position, and orientation. The propellant was mixed and prepared by AMRDEC.

The three propellant formulations used in these six tests can be seen in Table 2.1. The testing matrix is shown in Table 2.2.

Table 2.1: Sample Propellant Formulations

Sample	AP Distribution (200 micron/ 16 micron)	% Iron Oxide
GM08-20	70/30	1
GM08-21	65/35	0.75
GM08-22	60/40	0.75

Table 2.2: Test Matrix

Test #	Sample	Temperature
1	GM08-20	75°F
2	GM08-21	75°F
3	GM08-22	75°F
4	GM08-20	145°F
5	GM08-21	145°F
6	GM08-22	145°F

2.2 Combustion Vessel

A schematic of the closed combustion vessel used can be seen in Figure 2.1.

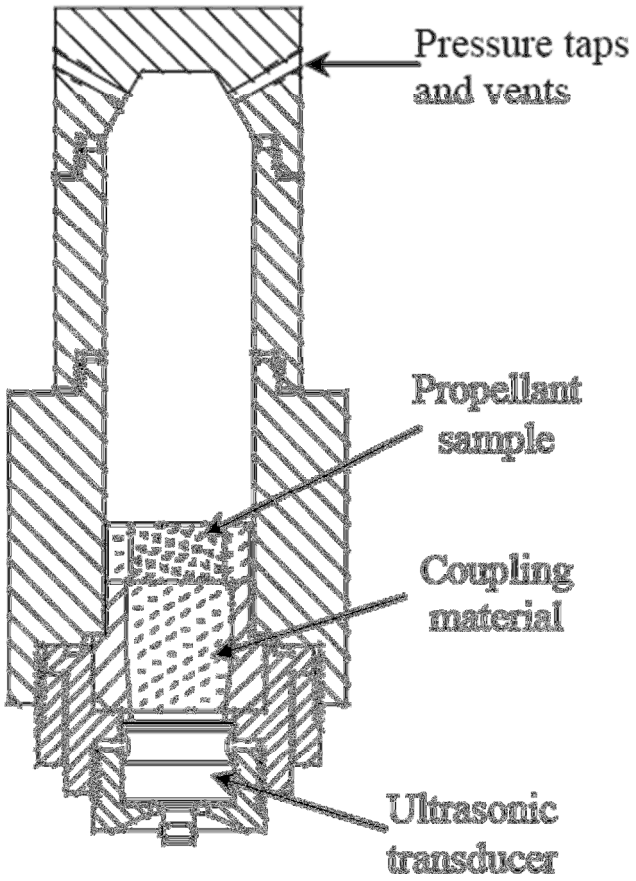


Figure 2.1: Combustion Vessel Schematic

The system consists of a fixed volume pressure vessel, a sample holder, and a nitrogen pressurization system. The pressure vessel is fabricated from thick walled stainless steel. Two electrical connectors provide an electrical circuit through the vessel to a pyrogen igniter. The upper cylindrical section has attachments for the pressure transducers, one located inside the temperature conditioning box and one located outside the temperature conditioning box. Prior to the test the pressure transducers are calibrated using a dead weight tester. The upper cylindrical section also has the inlet for the

nitrogen pressurization and purge system. The upper section also includes attachments for the 5000psi burst disk and nozzle assembly. The sample holder is made from stainless steel and has a cylindrical cavity filled with epoxy resin. The sample is fixed to the sample holder and cast in epoxy resin. Silicone grease is put between the ultrasonic transducer and the coupling material, the epoxy resin, at the bottom of the sample holder. A picture of this setup can be seen in Figure 2.2.

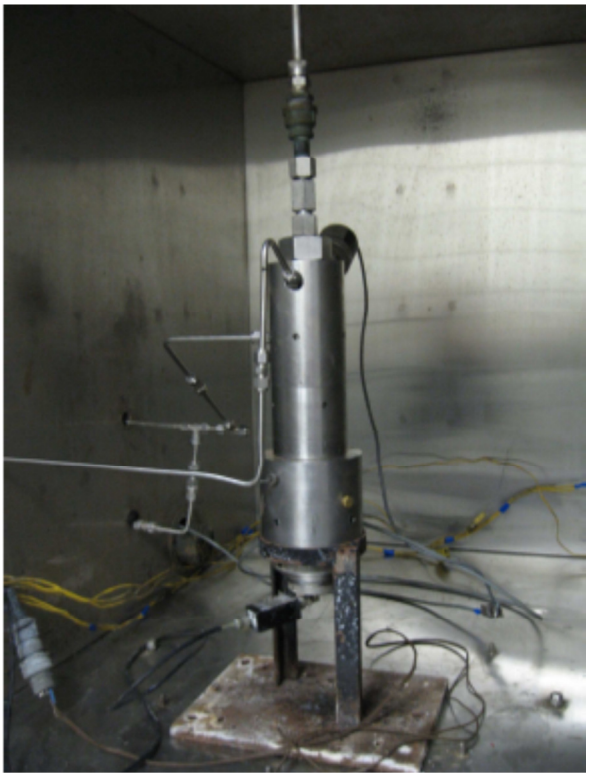


Figure 2.2: Closed Combustion Vessel Inside the Temperature Conditioning Box

2.3 Ultrasonic Apparatus

The hardware used in the ultrasonic technique includes a signal generator and converter manufactured by ONERA called the EDUM, a Panametrics V102 1.0/1.0 ultrasonic transducer, a HP 54602B oscilloscope, National Instruments BNC-2090

adapter chassis, and a National Instruments AT-MIO 16F 5 A/D board with LabVIEW data acquisition software.

2.3.1 Electronic Device for Ultrasonic Measurement

The Electronic Device for Ultrasonic Measurement (EDUM) drives the ultrasonic transducer, measures the propagation time, and outputs a voltage proportional to the propagation time. In the previous research done at the University of Alabama in Huntsville, the EDUM also amplified the signal, but currently that attribute of the EDUM is no longer available. When initiated, the EDUM sends a signal to the ultrasonic transducer which then sends a wave normal to the face of the sample. The EDUM then perceives the returning waves that bounce off of the material interfaces and displays the waveform on the oscilloscope. The EDUM then generates a clocking signal that is used by the data acquisition system. The ultrasonic pulses are initiated at 1000 samples per second, therefore sampling the propagation time at this rate.

2.3.2 Data Acquisition Card and Software

The data are then fed through the adapter to the A/D board in the computer and are acquired by LabVIEW. The LabVIEW software collects the waveforms, pressure data, and propagation time from the afore mentioned hardware.

2.4 Control Center

The control center, Figure 2.3, controls the ventilation, pressurization and ignition systems. It also holds the BNC adapter chassis.

Figure 2.3: Control Center

The valves on the control center allow for the pressurization and ventilation of the combustion vessel. For a test, the pressure in the combustion vessel is raised to 1600psi. The ignition system on the control center allows for the disabling, arming, and ignition of the system. When armed a battery is connected to the system. When fired the igniter is ignited and a signal is sent to the data acquisition system.

2.5 Temperature Conditioning System

Two boxes make up the temperature conditioning system. The first is housed in a chamber inside the lab where the combustion vessel is located during a test. This box is connected by pipes to the second box, outside the lab. The second box controls the temperature of the air circulated through the pipes to the box inside the lab. For the tests in this investigation the high temperature of 145°F and ambient temperature of 75°F were used. This temperature conditioning system allows for the test system to be thermally soaked prior to the test.

2.6 Pressurization/ Exhaust System

The pressurization/ exhaust system consists of tubes running to and from the combustion vessel. Nitrogen bottles are hooked up the combustion vessel by these tubes for pressurization. For the exhaust system, the tubes first run to a surge tank, in order to decrease the pressure in the system; then through a chemical scrubber, to clean the exhaust before venting into the atmosphere. The chemical scrubber functions by bubbling the exhaust fumes through a mixture of water and baking soda.

CHAPTER III

APPROACH

3.1 EDUM Method

The analogue method used, the EDUM method, automatically determines the propagation time as the test is being performed after calibrating it to look at the two peaks already indicated. The problem with this method is that it is not entirely reliable. At times the EDUM does not record the time value between these two pulses and that data is lost. A post test digital analysis method would appear to be the answer to this problem.

3.2 Cross Correlation Method

A previously developed digital method, called Cross Correlation, finds the time shift between two sequential waveforms, and uses this to find the propagation time of the second waveform. The time shift between waveforms is found by shifting the second

waveform until it overlaps the first waveform to the greatest extent possible. This method was written in Matlab and presented in the thesis of Marshall [21].

3.3 Zero Crossing Method

Another previously developed digital method, called the Original Zero Crossing Method, finds the first y-intercept, or zero crossing, of the first peak of the sample to determine the propagation time. This method was written in Matlab and presented in the thesis of Marshall [21]. His algorithm for determining the zero crossing of the sample peak starts with setting a horizontal threshold that the sample peak goes over and a vertical mask that the interface peak, the peak at the interface of the coupling material and sample, does not cross, as seen in Figure 3.1. The algorithm then follows the threshold starting at the mask until a peak is encountered. Next, the algorithm follows the peak until the zero crossing is found. This is achieved by multiplying consecutive points until a zero or negative product is reached. The algorithm then interpolates between these two points to find the zero crossing. This is done for each waveform recorded for the entire test.

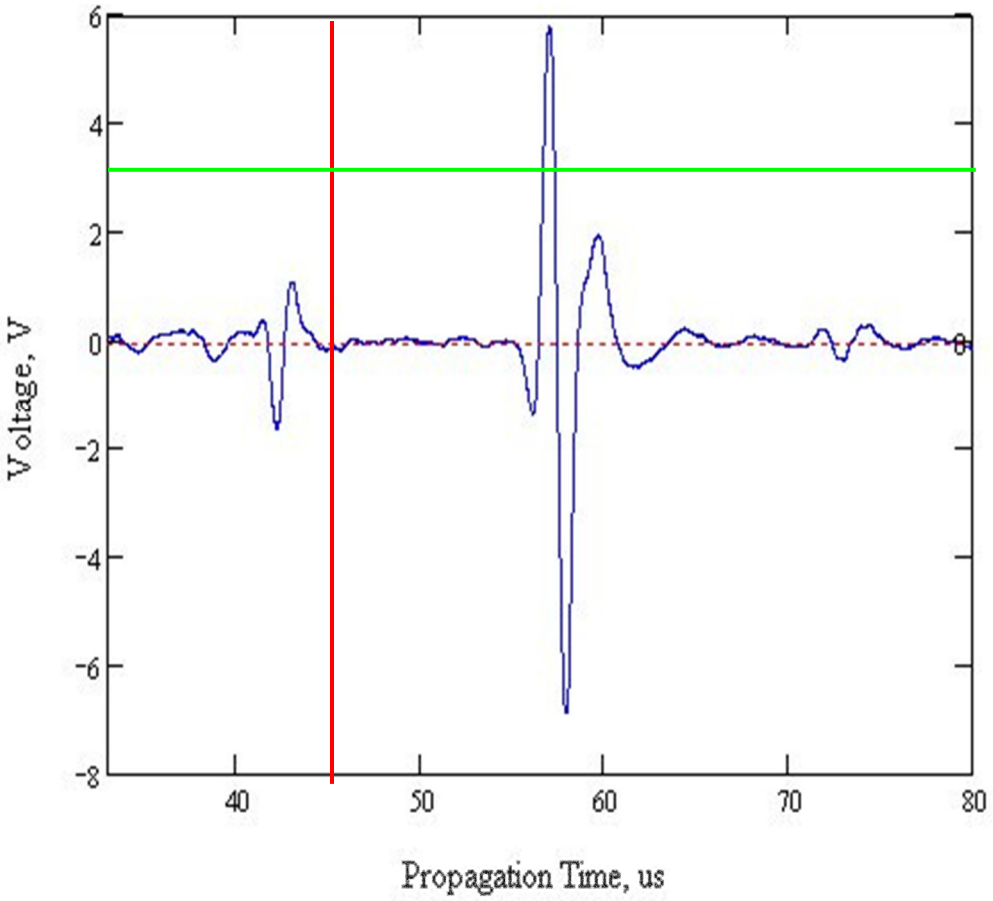


Figure 3.1: Waveform with Mask and Threshold

Though this algorithm appears solid, there are times when it fails. Such instances include when the sample peak is too small to cross the threshold or other interference peaks are large enough to pass the threshold. An example of a waveform that was not analyzed properly by this method can be seen in Figure 3.2. With this in mind, an alteration to the Original Zero Crossing Method was made.

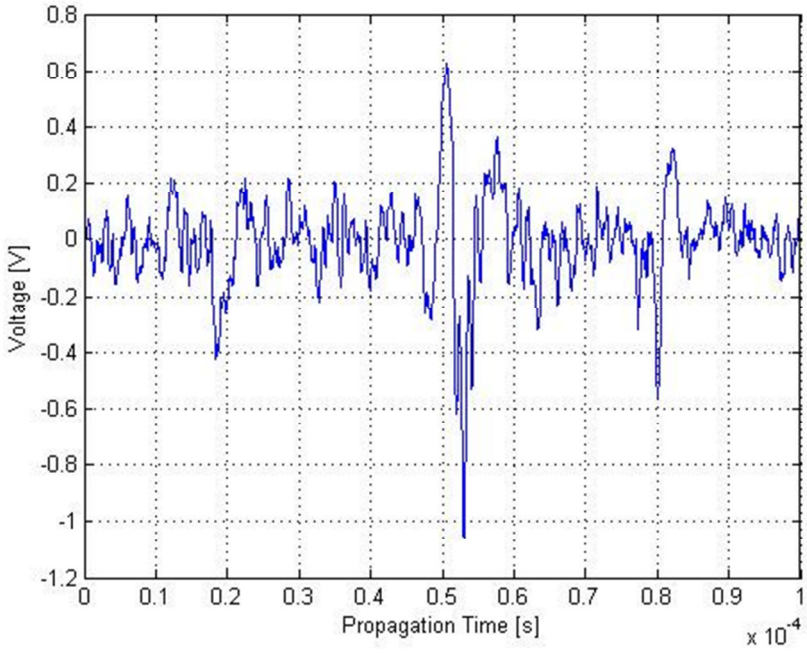


Figure 3.2: Example Waveform Not Properly Analyzed by Original Algorithm

In the New Zero Crossing Method, for the first waveform of the test, the previous method of determining the zero crossing is the same. A threshold and mask are set for this first waveform only. Then the algorithm follows the threshold starting at the mask until a peak is found at the threshold and follows the peak until the zero crossing is found. For the second waveform, and each additional waveform, the algorithm changes. Starting at the x-position of the previous waveform's zero crossing, the new algorithm then moves a set number of points in the negative direction of the x-axis. The set number of points used for this investigation was 50 points. 50 points is 1.5 times the approximate expected movement of the surface peak between consecutive waveforms. Moving the algorithm back 50 points, rather than starting again at the mask and threshold, allows it to already lie on the peak of the surface reflection wave without being concerned about whether that peak is large enough to cross the threshold. The algorithm then follows the

peak in the same manner as before until the zero crossing is found. This is repeated for all waveforms collected from the test after the first waveform. The waveform shown in Figure 3.2 was properly analyzed by this algorithm after the Original Zero Crossing Method algorithm failed. Figure 3.3 shows this algorithm on the second waveform of the series where the zero crossing has already been found by the previous method on the first waveform.

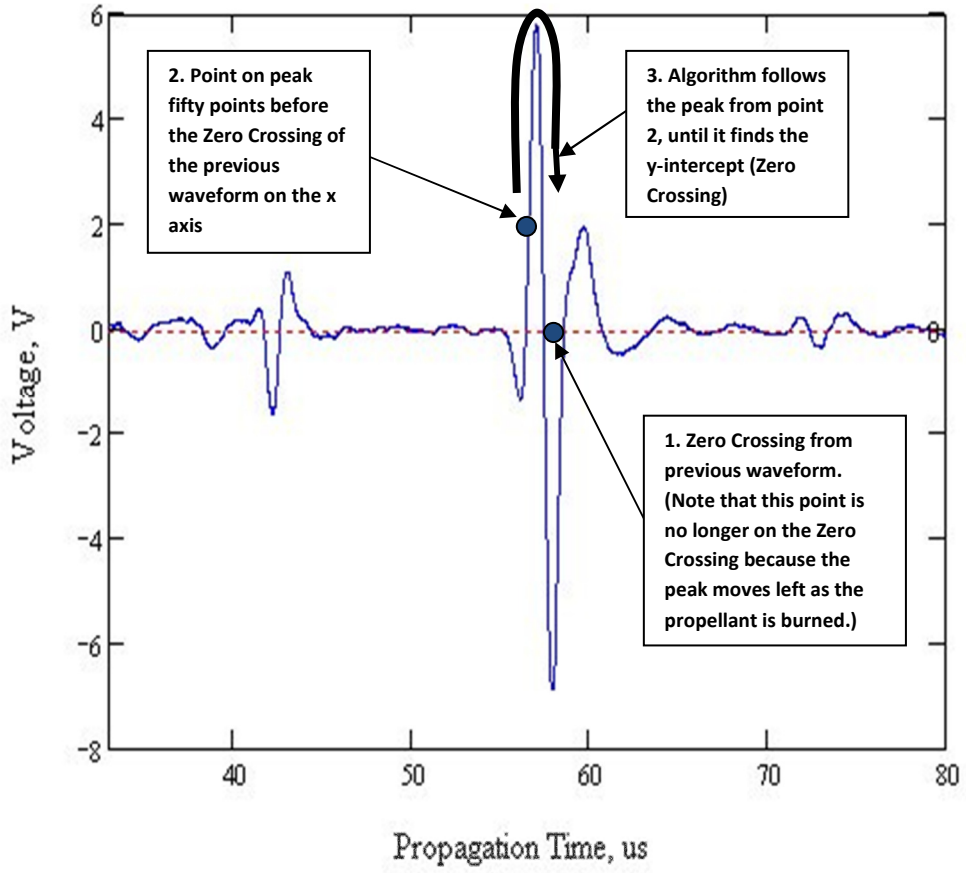


Figure 3.3: New Zero Crossing Algorithm

After the zero crossings for all waveforms are found, this information is imported into a MathCAD data reduction code to determine the propagation time, instantaneous thickness, and burn rate.

3.4 Data Analysis

For each test a pre- and post-test are performed to determine the constants in the system. The pressure vessel with the sample installed is pressurized to approximately 1600 psi and then vented. Linear regression is performed and the constants for the coupling material are found. Figures 3.4 through 3.6 show the pressure versus time plots for the pre-test, test, and post-test of Test #1.

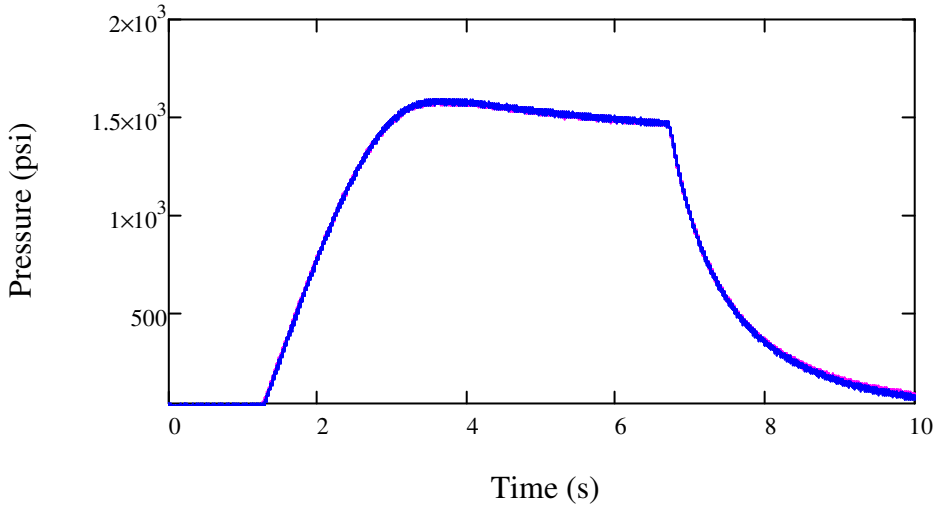


Figure 3.4: Test #1, Pre-Test, GM08-20 @ 75°F, Pressure Data

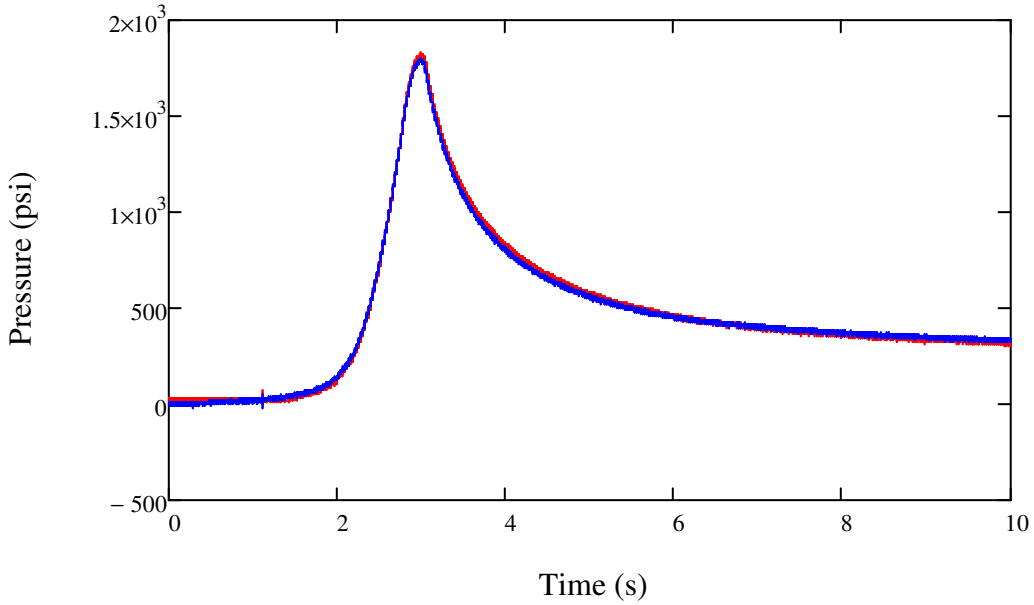


Figure 3.5: Test #1, Test, GM08-20 @ 75°F, Pressure Data

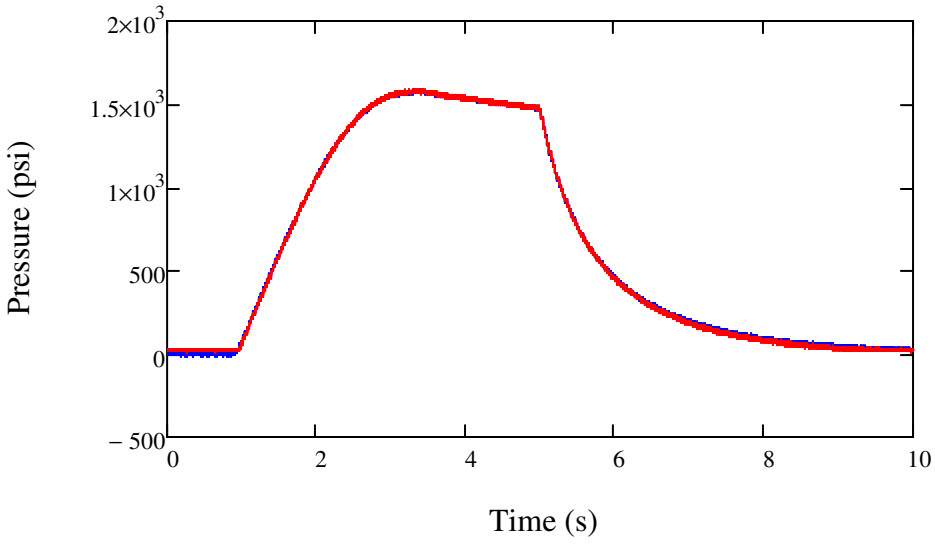


Figure 3.6: Test #1, Post-Test, GM08-20 @ 75°F, Pressure Data

CHAPTER IV

RESULTS

4.1 Propellant Testing Matrix

Six tests were run using three different propellant formulations. Each propellant was tested at two temperatures, 75°F and 145°F. In the following section the propagation time and burn rates are presented for Test #1 found with the New Zero Crossing Method, the Original Zero Crossing Method, and the Cross Correlation Method. Charts for Tests #2 through #6 can be found in Appendix B. The burn rate, burn rate equation and temperature sensitivity of each propellant will be presented.

4.2 Propagation Time and Burn Rates

The following charts, Figures 4.1 through 4.5, display the propagation time and burn rate of Test #1 using the New Zero Crossing Method, the Original Zero Crossing Method, and the Cross Correlation Method.

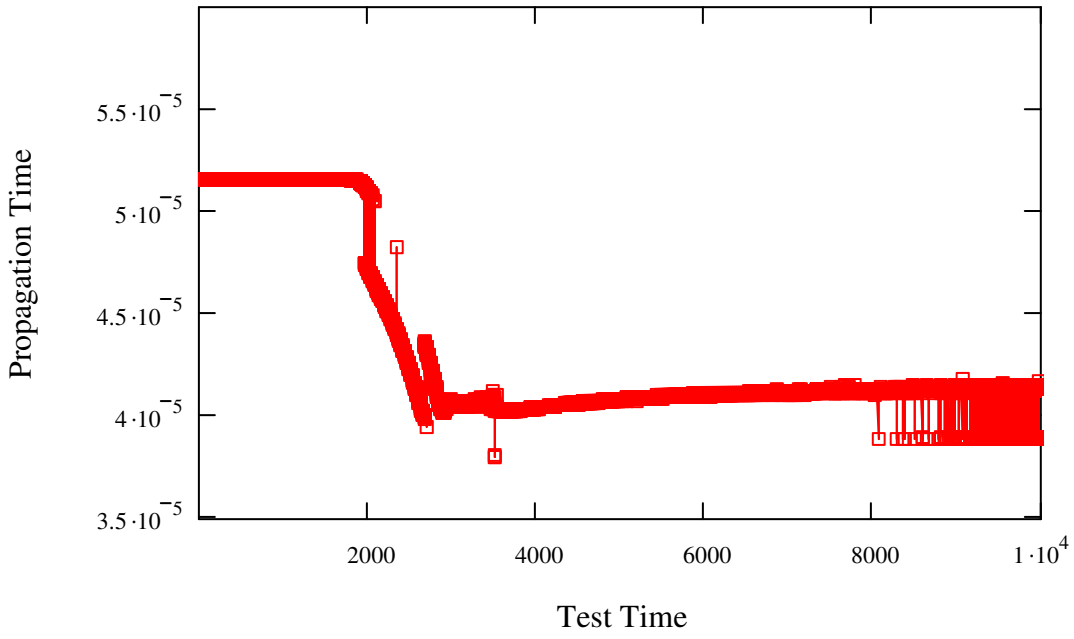


Figure 4.1: Test #1, GM08-20 @ 75°F, Propagation Time, New Zero Crossing

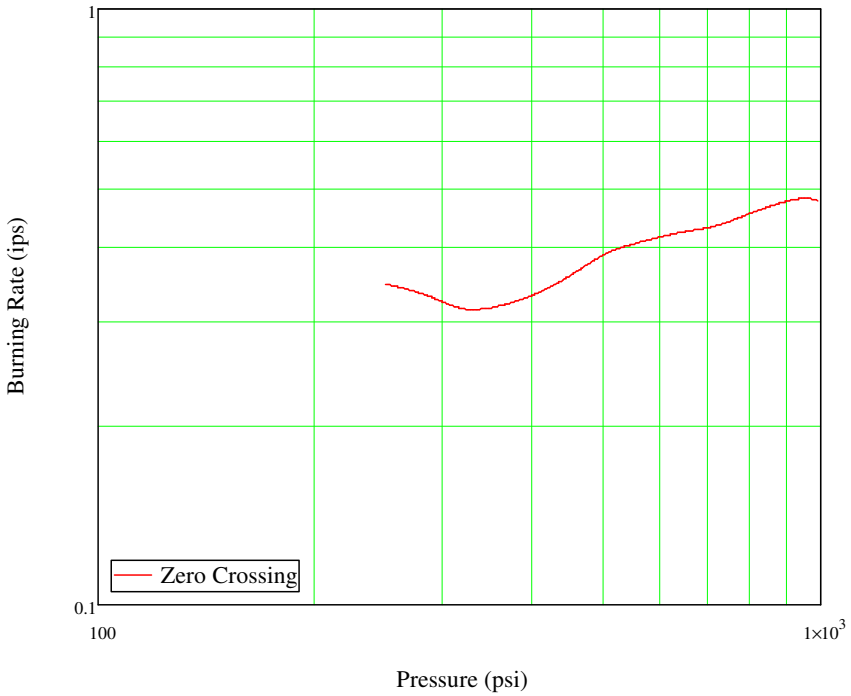


Figure 4.2: Test #1, GM08-20 @ 75°F, Burn Rate, New Zero Crossing

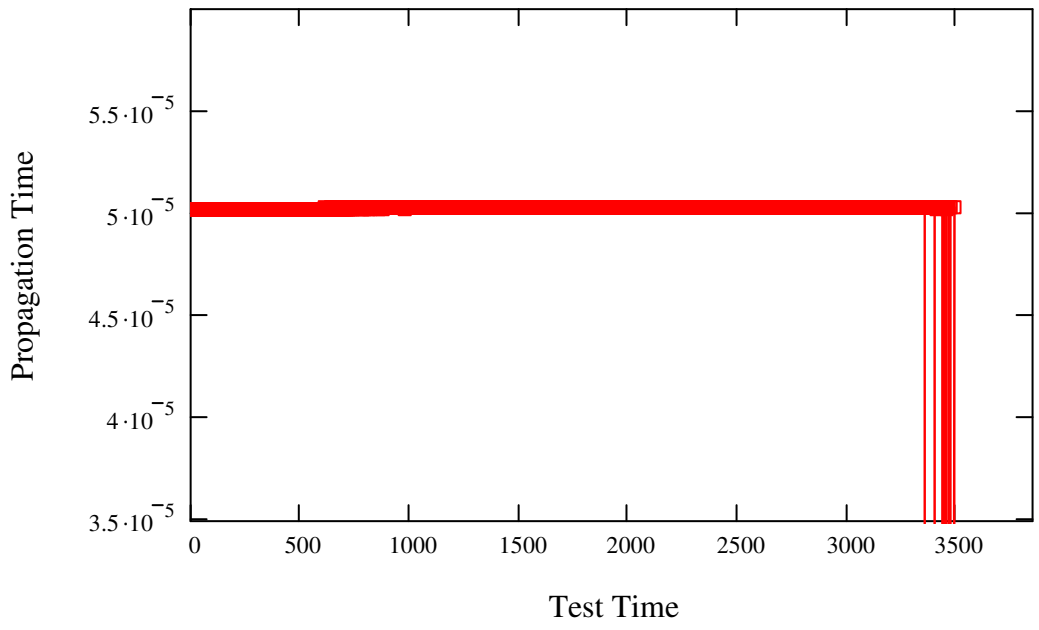


Figure 4.3: Test #1, GM08-20 @ 75°F, Propagation Time, Original Zero Crossing

Figure 4.3 shows that the propagation time could not be found by the Original Zero Crossing method for Test #1. As no propagation time was found, the burn rates could not be calculated.

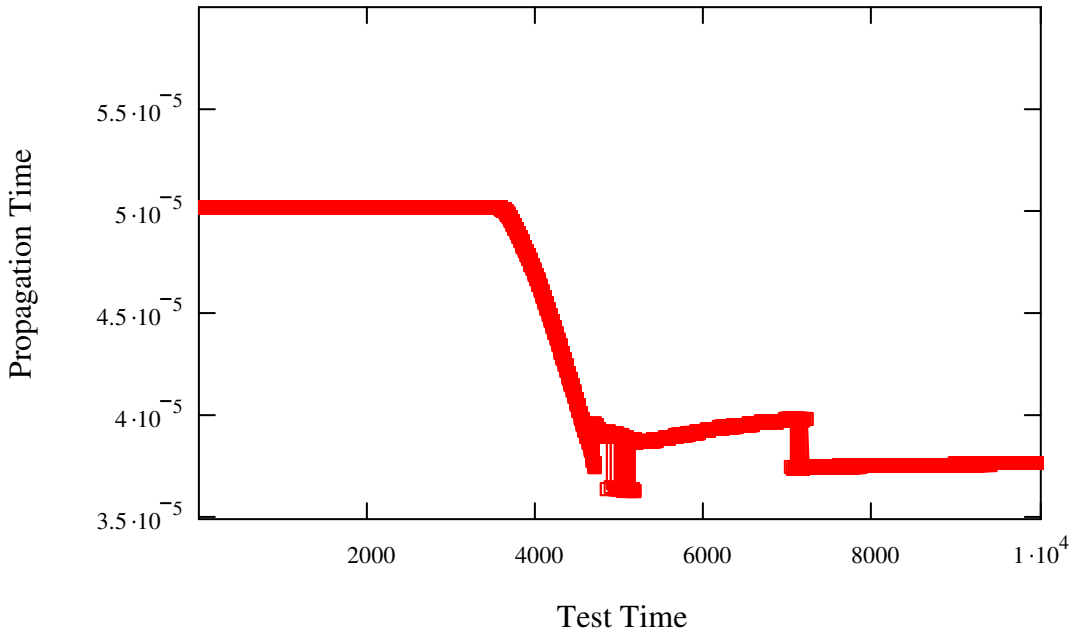


Figure 4.4: Test #1, GM08-20 @ 75°F, Propagation Time, Cross Correlation

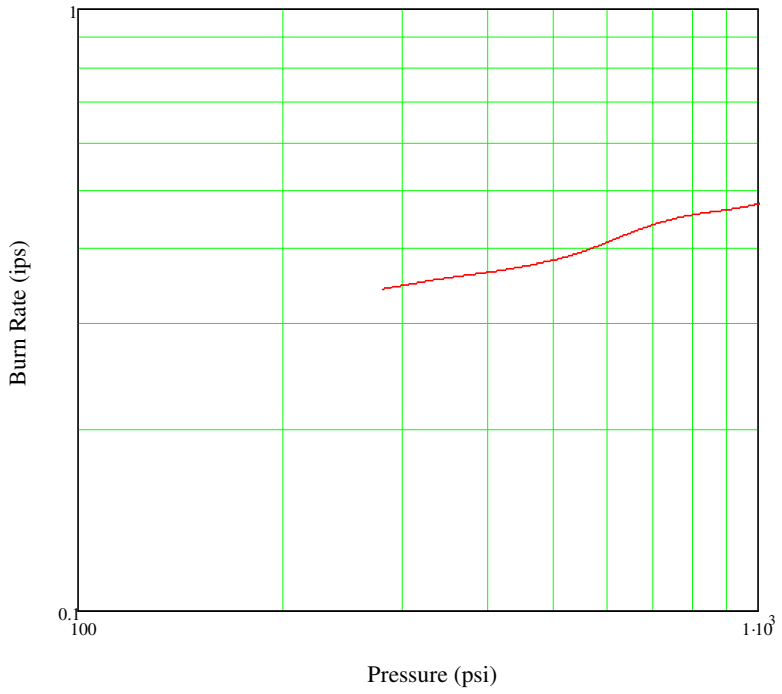


Figure 4.5: Test #1, GM08-20 @ 75°F, Burn Rate, Cross Correlation

The burn rates of the three samples at both ambient and high temperatures, at 500psi are shown in Table 4.1 below.

Table 4.1: Burn Rates of Sample Propellants at 500psi

Test	Burning Rate (ips) @ 500psi
1	0.39
2	0.40
3	0.45
4	0.48
5	0.58
6	0.62

4.3 Burn Rate Equations

The burn rate equation for the three propellants at ambient temperature is shown in Figures 4.6 through 4.8 below.

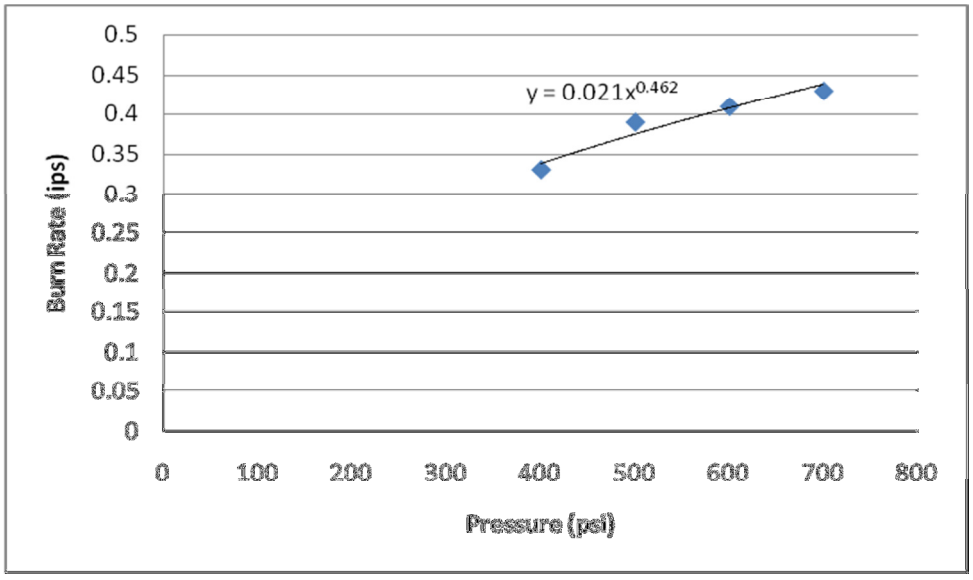


Figure 4.6: Propellant GM08-20 @ 75°F Burn Rate Equation

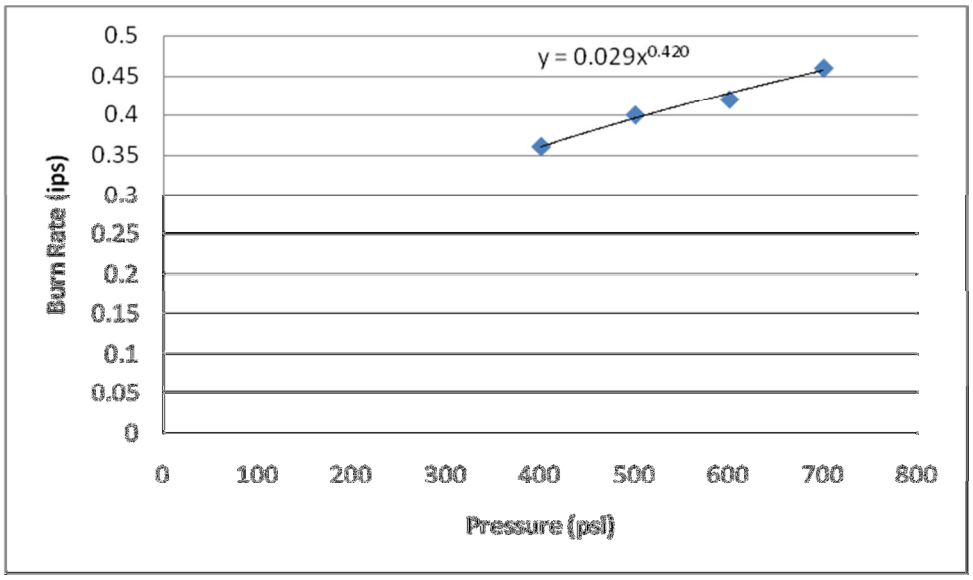


Figure 4.7: Propellant GM08-21 @ 75°F Burn Rate Equation

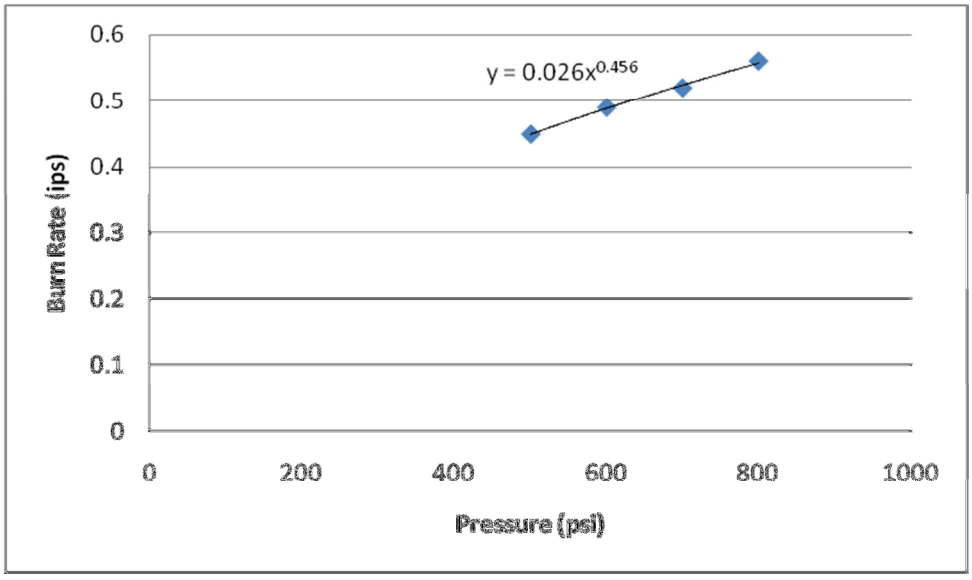


Figure 4.8: Propellant GM08-22 @ 75°F Burn Rate Equation

The burn rate equation for the three propellants at ambient temperature is shown in Table 4.2 below.

Table 4.2: Burn Rate Equations for the Three Propellants at Ambient Temperature

Propellant	Burn Rate Equation
GM08-20	$r_b = 0.0212 \cdot P^{0.4626}$
GM08-21	$r_b = 0.0291 \cdot P^{0.4202}$
GM08-22	$r_b = 0.0264 \cdot P^{0.4564}$

4.4 Temperature Sensitivity

The temperature sensitivity was calculated with Equation 1.2 for each propellant tested, using the New Zero Crossing Method, at 500psi. These results can be found in Table 4.3 below.

Table 4.3: Temperature Sensitivity at 500psi

Propellant	Temperature Sensitivity
GM08-20	0.00297
GM08-21	0.00531
GM08-22	0.00458

CHAPTER V

DISCUSSION

A comparison of burn rates determined for each test using the three digital methods, the Original Zero Crossing Method, the New Zero Crossing Method, and the Cross Correlation Method, is made. The robustness of each method is also considered.

In Tests #1, #3, #4, #5, and #6 the Original Zero Crossing Method was unable to determine propagation times, while the New Zero Crossing Method and the Cross Correlation Method were able to. So no burn rates could be computed for those tests using the Original Zero Crossing Method. For Test #2 burn rates at only a small range of pressure were found using this method.

In all Tests the New Zero Crossing Method and the Cross Correlation Method were able to compute propagation times, allowing for the calculation of burn rate. However in Test #5 the New Zero Crossing Method only allowed for the calculation of burn rates over a small range of pressures and the Cross Correlation Method seems to have failed entirely offering an almost vertical line for burn rates.

In Tests #1, #2, #3, #4, and #6, both the New Zero Crossing Method and the Cross Correlation Methods found similar burn rates.

CHAPTER VI

UNCERTAINTY ANALYSIS

Using the Direct Monte Carlo Simulation method with the values shown in Table 6.1 below and Equation 1.5, the uncertainties of each test at 500psi were found and presented in Table 6.2. The Direct Monte Carlo Simulation method was performed in MS Excel. Random uncertainties were obtained from a Guassian distribution of random numbers given a base value and standard deviation. Systematic uncertainties were drawn from Table 6.1 below, where these values were determined previously by Dauch [22] and as the system has not changed, the values have not changed either. The one uncertainty value that could have changed from previous investigations would be the Propagation Time, τ, as this value was determined from the EDUM Method of burn rate determination. Though, as the EDUM uses a similar method as the Digital Zero Crossing Method to determine the burn rates, it was decided that the same uncertainty could be used. This uncertainty value was also used by Marshall [21] in his investigation of new digital burn rate analysis methods. The data reduction equation was calculated with the uncertainties applied 10,000 times. 10,000 iterations were done in agreement with the

iterations done in previous investigations done at the University of Alabama in Huntsville in this area. The mean and standard deviation of the distribution of the results were calculated. Twice the standard deviation is the 95% confidence uncertainty, reported in Table 6.2. Further information on the Direct Monte Carlo Simulation method and other uncertainty analysis methods can be found in Reference 24.

Table 6.1: Uncertainties Used for Analysis

Variable	**Uncertainty**
Pressure, P	7 psi
Initial Thickness, E_{p0}	0.022 in
Regression Constant, a_c	4.629e-14 s/psi
Regression Constant, a_p	8.118e-14 s/psi
Regression Constant, b_c	1.0e-7 s
Regression Constant, b_p	1.0e-7 s
Propagation Time, τ	1.0e-7 s

Table 6.2: Uncertainty Results for Burn Rates at 500psi

Test	Burn Rate (ips)	Uncertainty (95% Interval) (ips)	% Relative Uncertainty
1	0.39	0.028	7.30
2	0.40	0.028	7.05
3	0.45	0.028	6.30
4	0.48	0.029	6.00
5	0.58	0.029	4.96
6	0.62	0.028	4.56

For Tests #1 through #6 the uncertainties for the 95% confidence interval are similar and the relative uncertainties are below 10%. It should be noted that these uncertainties do not capture repeatability; these are single test uncertainty values. These uncertainties are also comparable to those found by Marshall [21] and Evans [23] utilizing the same laboratory and equipment.

CHAPTER VII

CONCLUSIONS

For the six tests performed within this investigation some conclusions can be drawn. The Original Zero Crossing Method failed almost entirely. It is suspected that this is due to there no longer being an automatic amplification of the surface peak by the EDUM. The new algorithm for the New Zero Crossing method works more reliably than the previous algorithm, though improvements can still be made. The Cross Correlation Method works on each test that the New Zero Crossing works on; however it may be a more robust method as it often found burn rates in pressure ranges that the New Zero Crossing Method did not. As seen in Test #5, if the voltage of the waves is reduced, the noise becomes more prominent, sometimes causing the New Zero Crossing Method and the Cross Correlation Method to fail. Since these methods are post test analysis methods it would be possible to utilize noise reduction techniques. This may allow for a clearer surface peak for the algorithm to follow and a more robust analysis method. The uncertainties of the burn rates are less than 10%. These uncertainties come from systematic and random errors in the equipment.

APPENDICES

APPENDIX A

Testing Data

Table A.1: Testing Data from Lab Notebook

Test # in Thesis	Test # in Notebook	Sample	Date	Temp (°F)	Sample Thickness (in)	Sample Position
1	**81**	GM08-20	4/28/2009	75	0.509	1-1
2	**82**	GM08-21	5/26/2009	75	0.508	3-15
3	**83**	GM08-22	6/1/2009	75	0.5045	5-24
4	**84**	GM08-20	6/8/2009	145	0.505	1-5
5	**85**	GM08-21	6/15/2009	145	0.512	3-11
6	**91**	GM08-22	9/28/2010	145	0.5045	6-29

APPENDIX B

Results

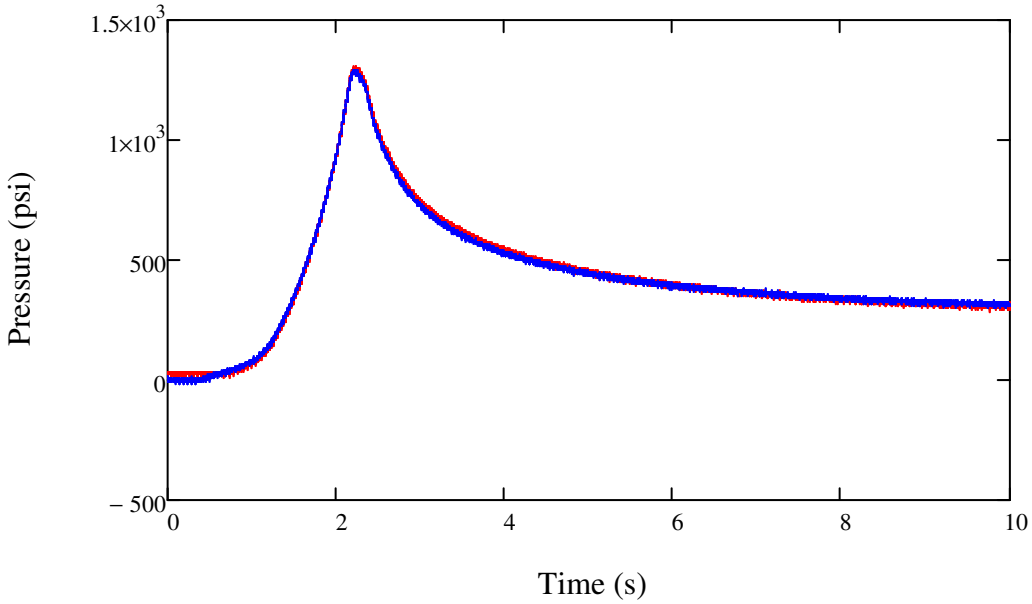


Figure B.1: Test #2, GM08-21 @ 75°F, Pressure Data

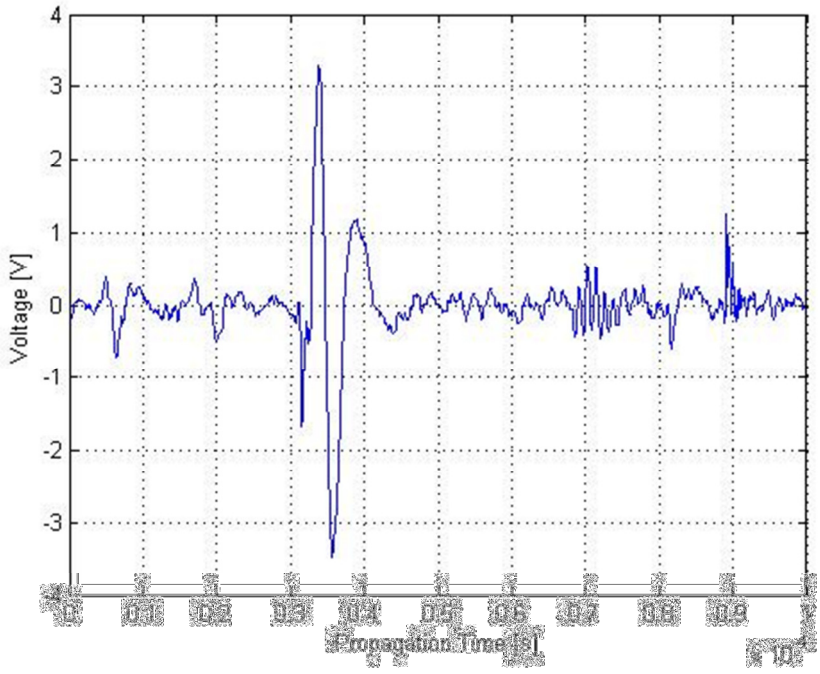


Figure B.2: Test #2, GM08-21 @ 75°F, at 0.0 Seconds

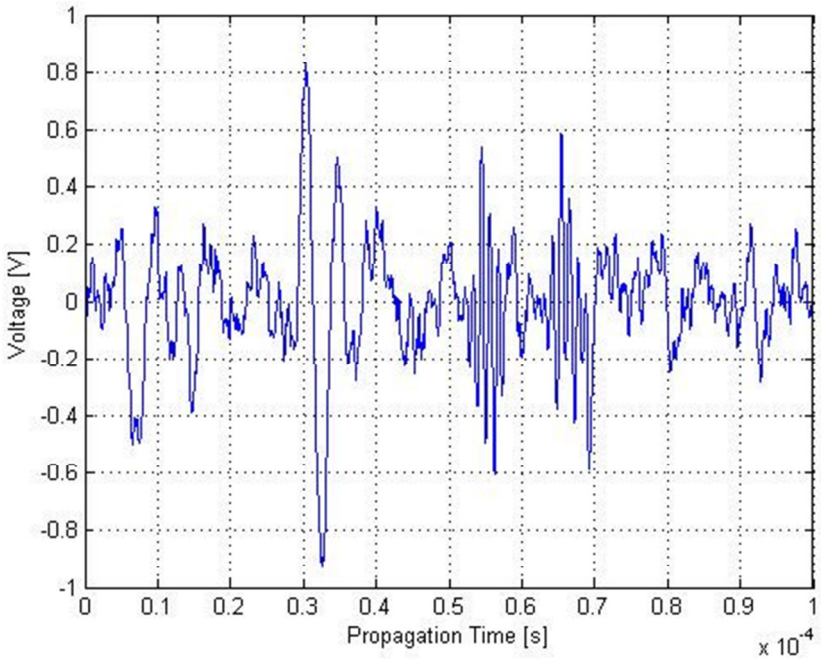


Figure B.3: Test #2, GM08-21 @ 75°F, at 1.0 Seconds

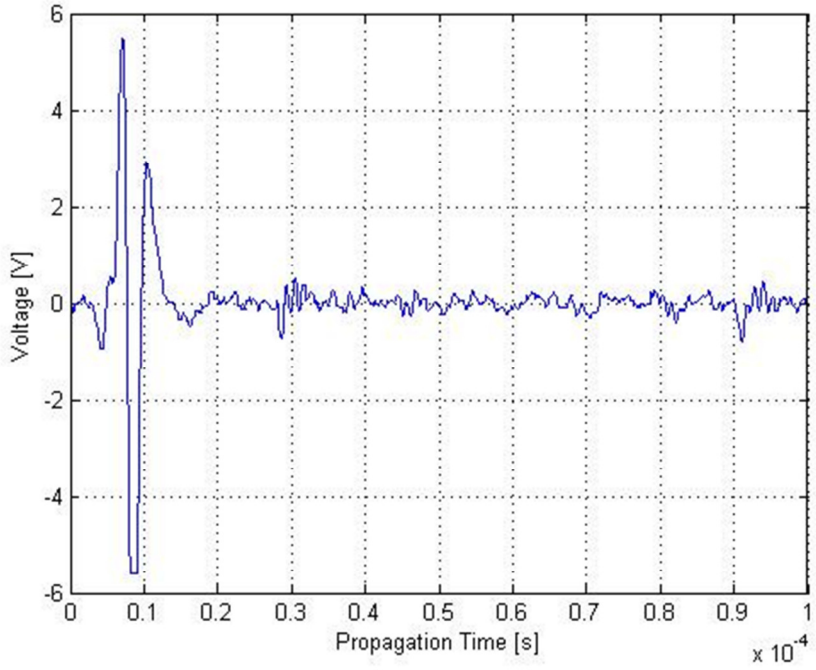


Figure B.4: Test #2, GM08-21 @ 75°F, at 2.0 Seconds

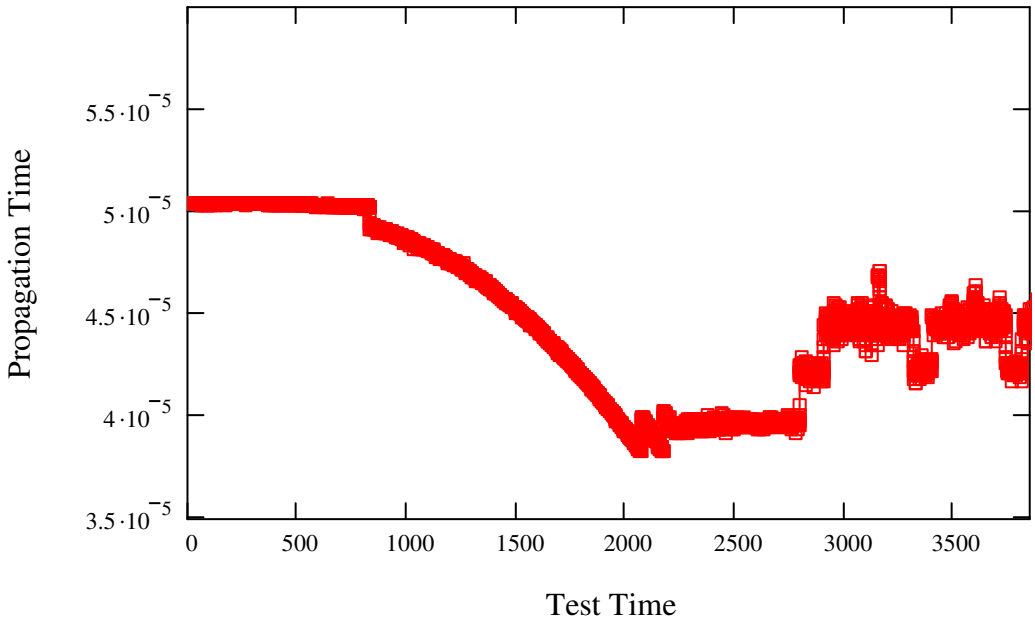


Figure B.5: Test #2, GM08-21 @ 75°F, Propagation Time, New Zero Crossing

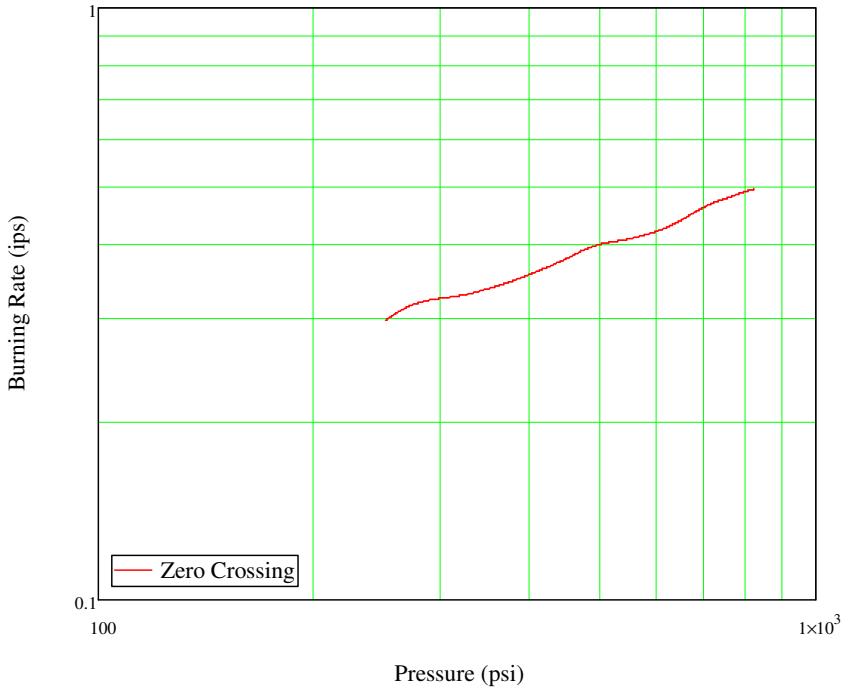


Figure B.6: Test #2, GM08-21 @ 75°F, Burn Rate, New Zero Crossing

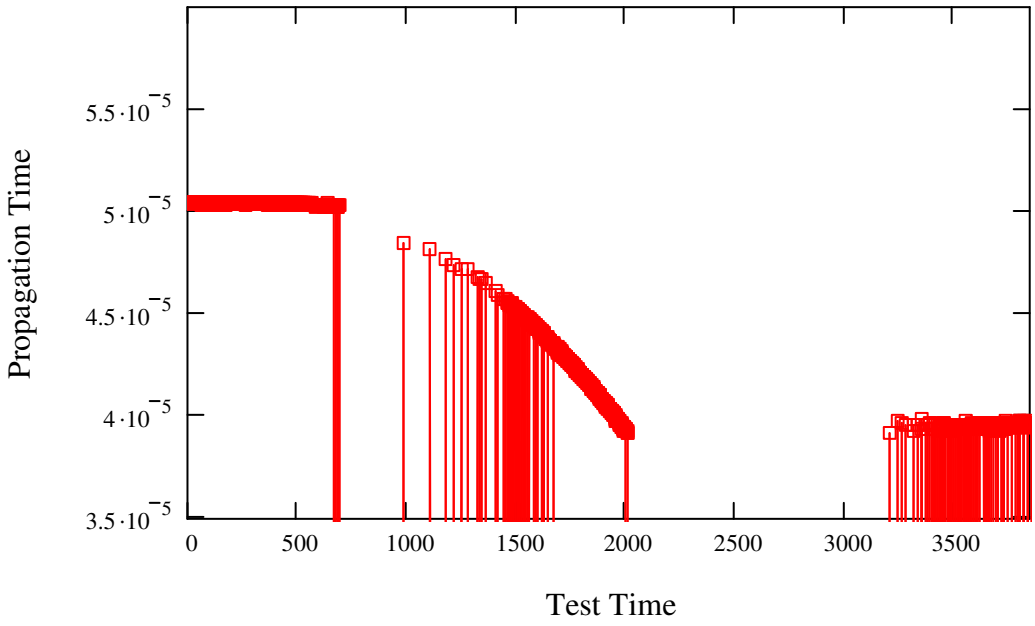


Figure B.7: Test #2, GM08-21 @ 75°F, Propagation Time, Original Zero Crossing

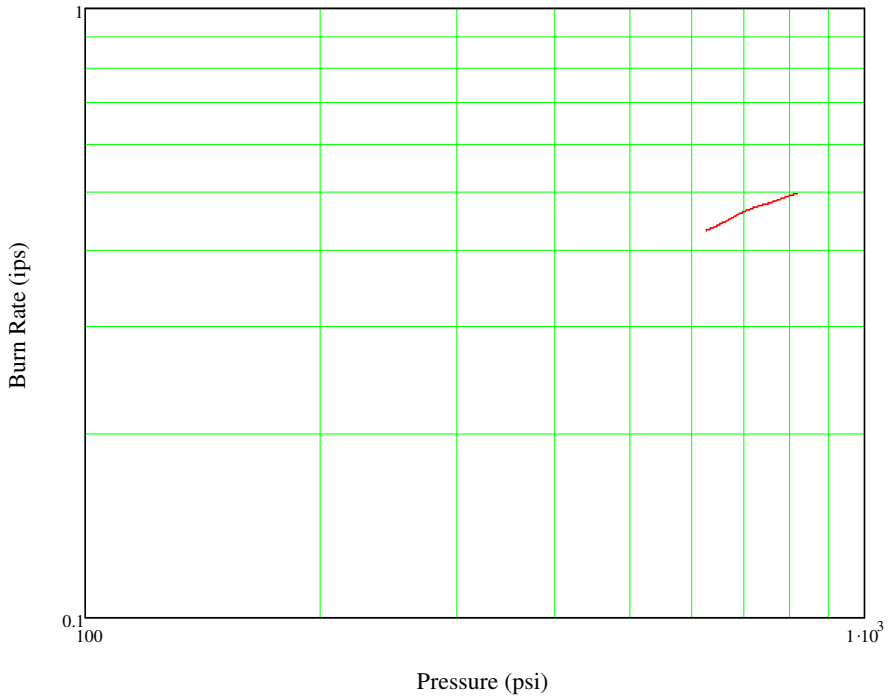


Figure B.8: Test #2, GM08-21 @ 75°F, Burn Rate, Original Zero Crossing

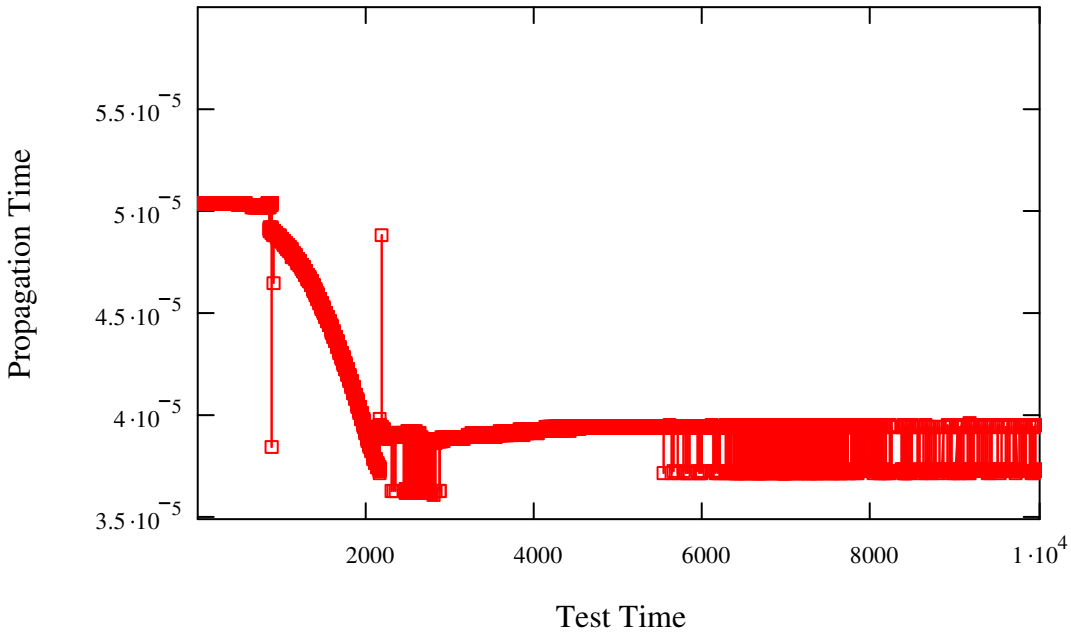


Figure B.9: Test #2, GM08-21 @ 75°F, Propagation Time, Cross Correlation

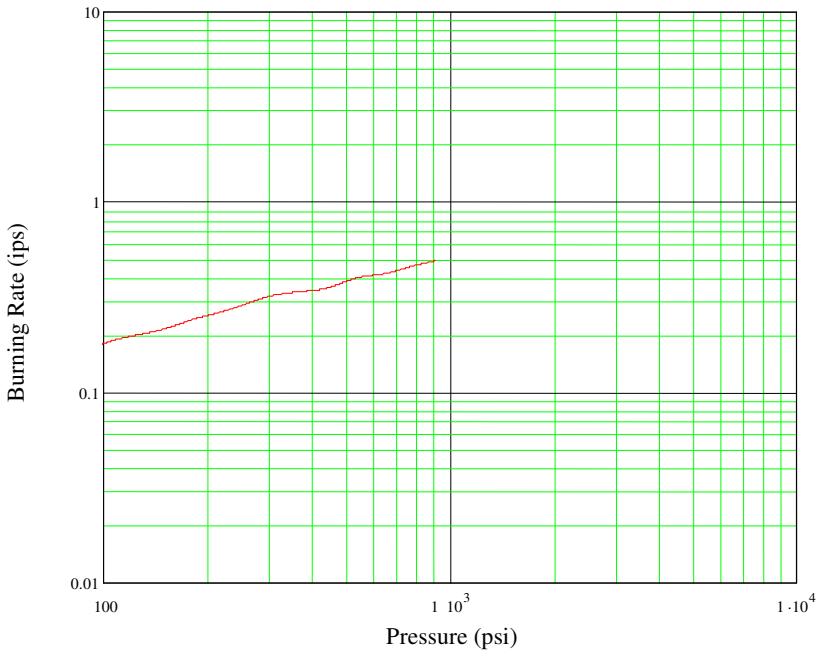


Figure B.10: Test #2, GM08-21 @ 75°F, Burn Rate, Cross Correlation

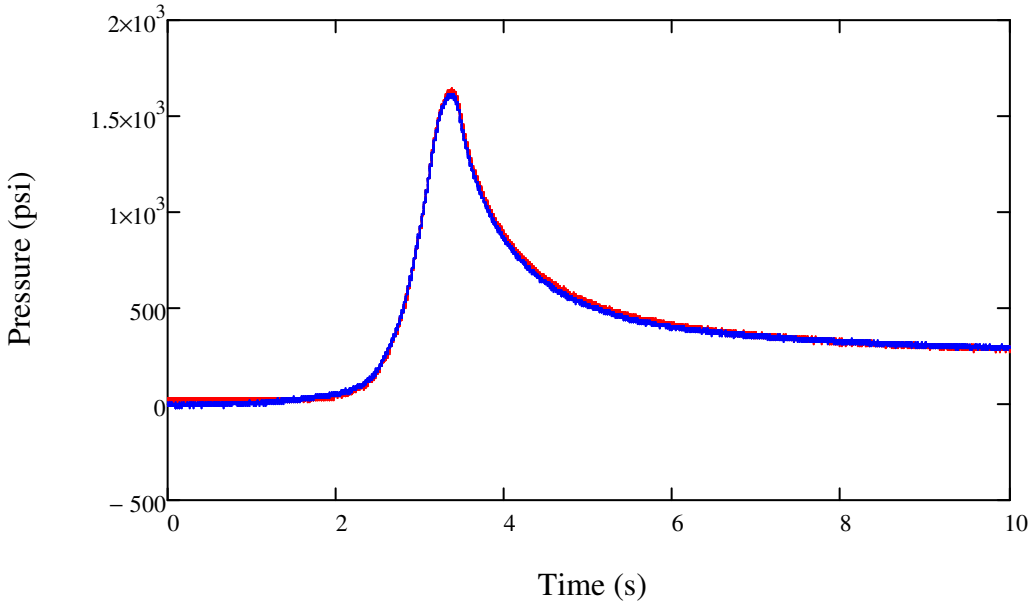


Figure B.11: Test #3, GM08-22 @ 75°F, Pressure Data

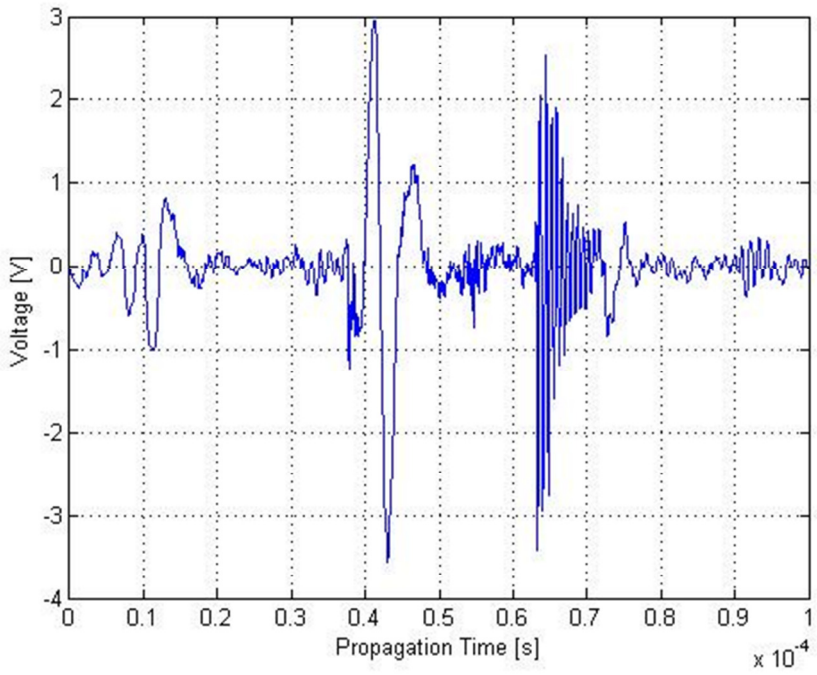


Figure B.12: Test #3, GM08-22 @ 75°F, at 0.0 Seconds

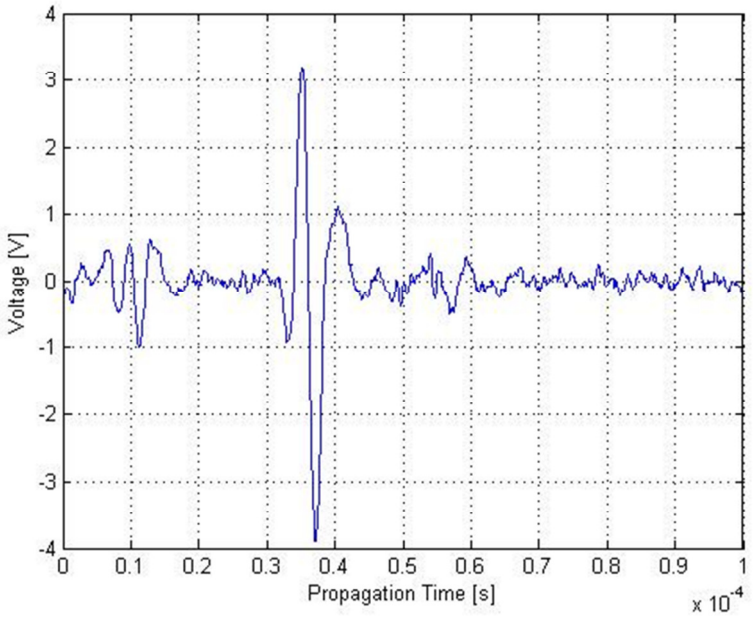


Figure B.13: Test #3, GM08-22 @ 75°F, at 1.0 Seconds

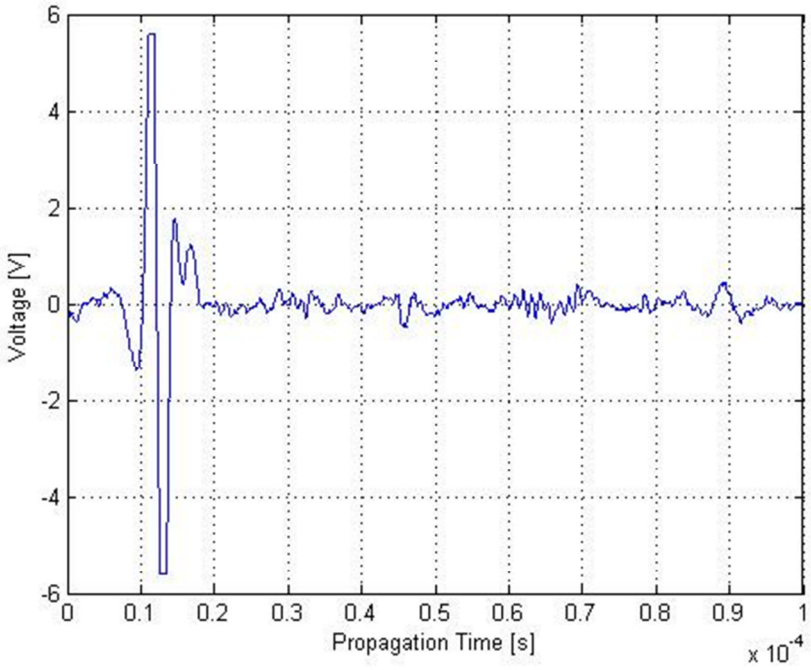


Figure B.14: Test #3, GM08-22 @ 75°F, at 2.0 Seconds

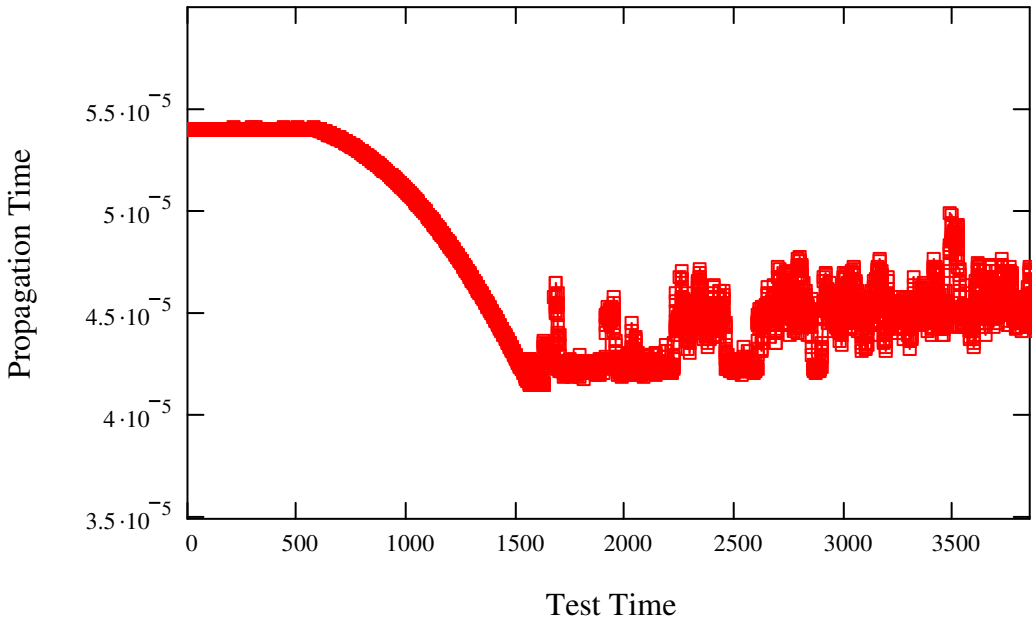


Figure B.15: Test #3, GM08-22 @ 75°F Propagation Time, New Zero Crossing

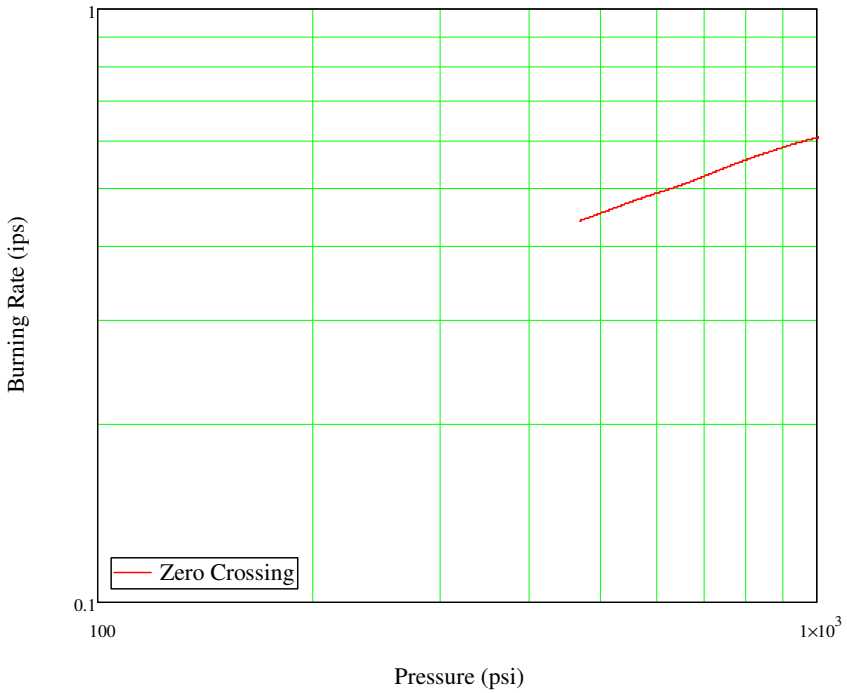


Figure B.16: Test #3, GM08-22 @ 75°F Burn Rate, New Zero Crossing

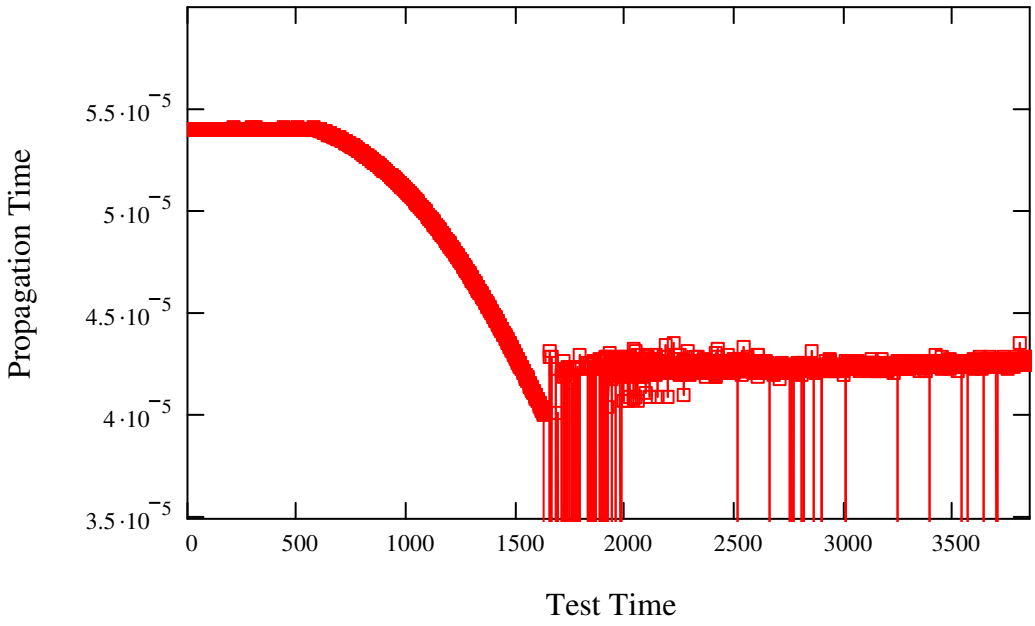


Figure B.17: Test #3, GM08-22 @ 75°F Propagation Time, Original Zero Crossing

Figure B.17 shows that the propagation time could not be found by the Original Zero Crossing method for Test #3 (look at the Test Time from about 1700 to 2000). As no propagation time was found, the burn rates could not be calculated.

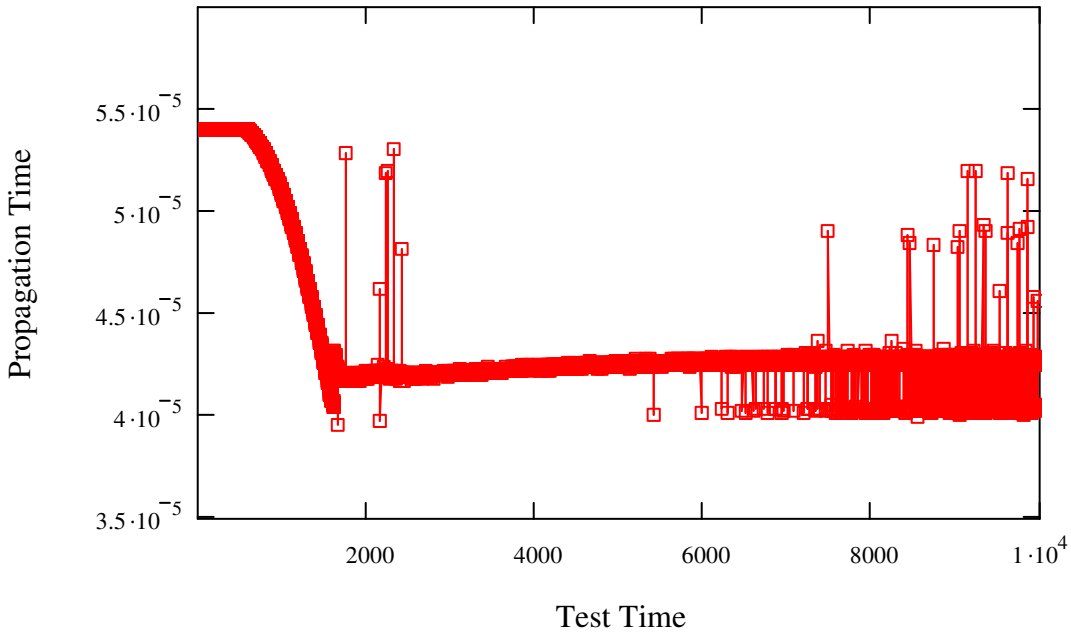


Figure B.18: Test #3, GM08-22 @ 75°F Propagation Time, Cross Correlation

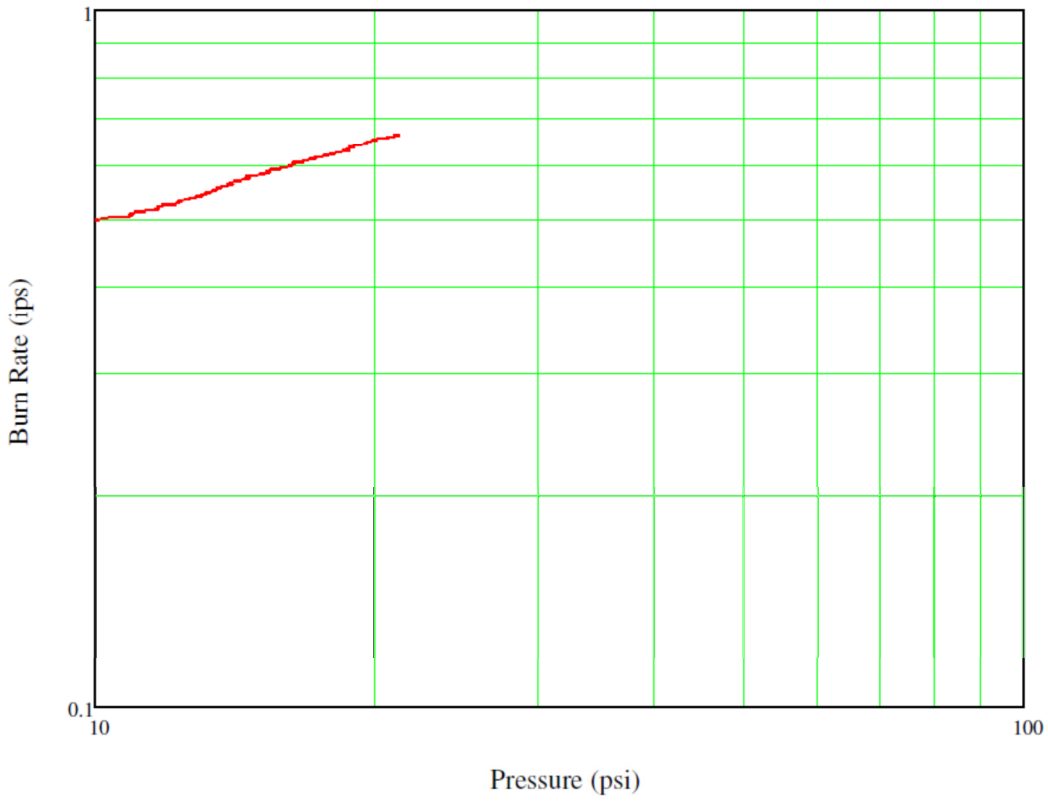


Figure B.19: Test #3, GM08-22 @ 75°F Burn Rate, Cross Correlation

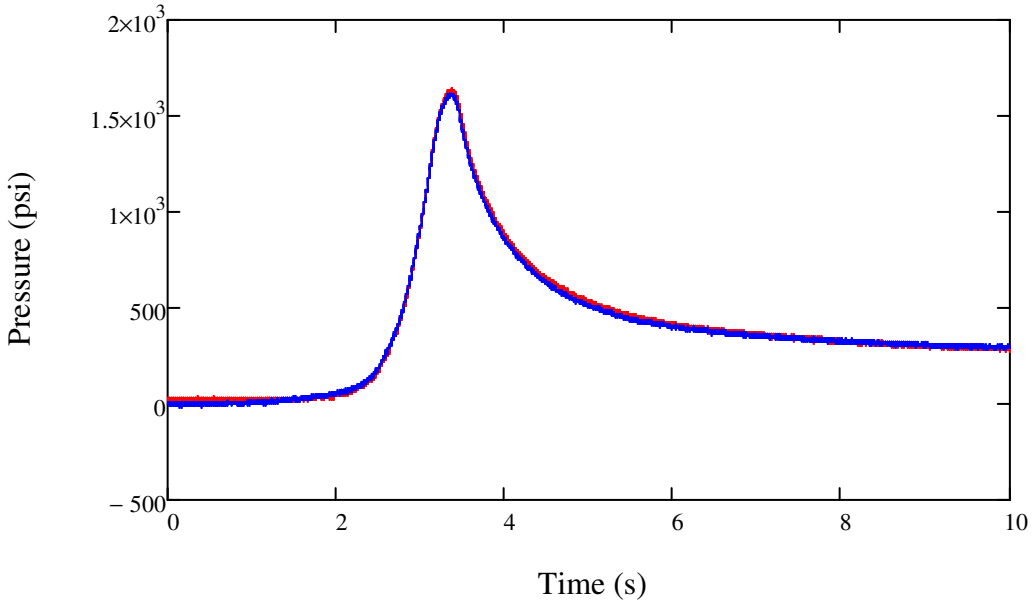


Figure B.20: Test #4, GM08-20 @ 145°F, Pressure Data

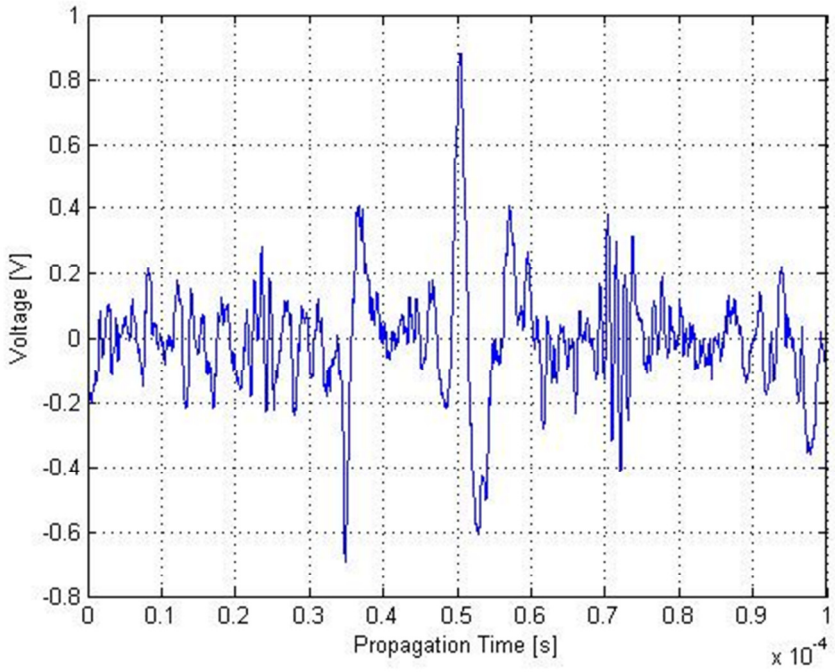


Figure B.21: Test #4, GM08-20 @145°F, at 0.0 Seconds

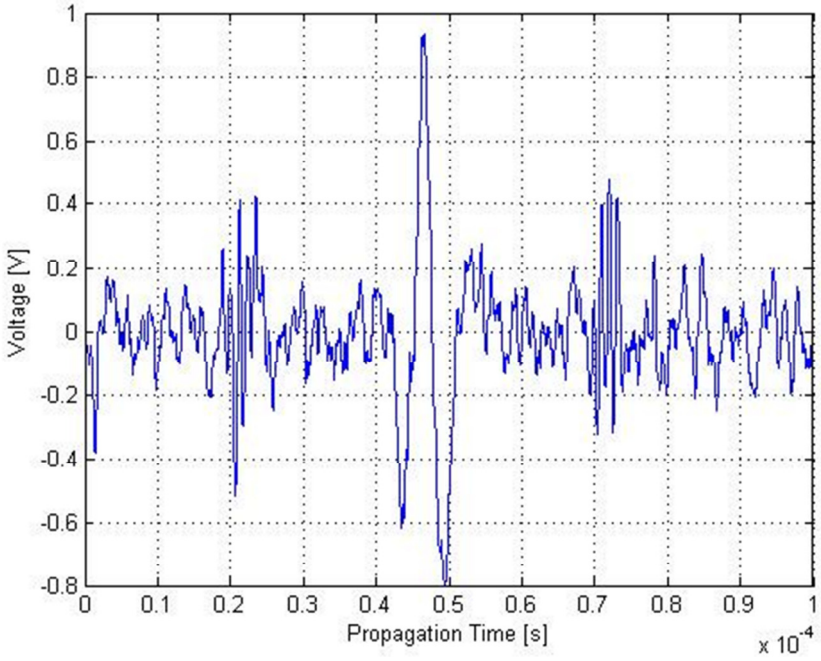


Figure B.22: Test #4, GM08-20 @145°F, at 1.0 Seconds

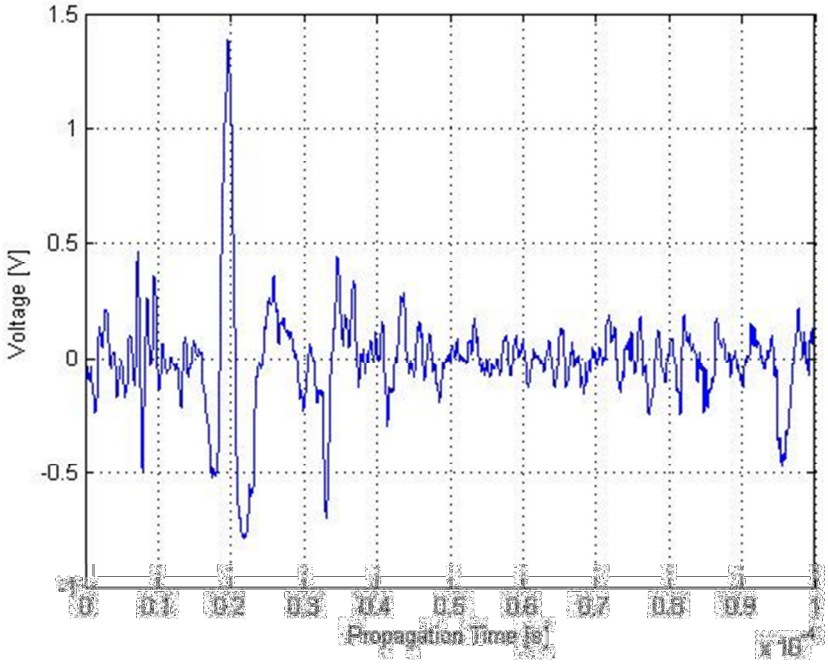


Figure B.23: Test #4, GM08-20 @145°F, at 2.0 Seconds

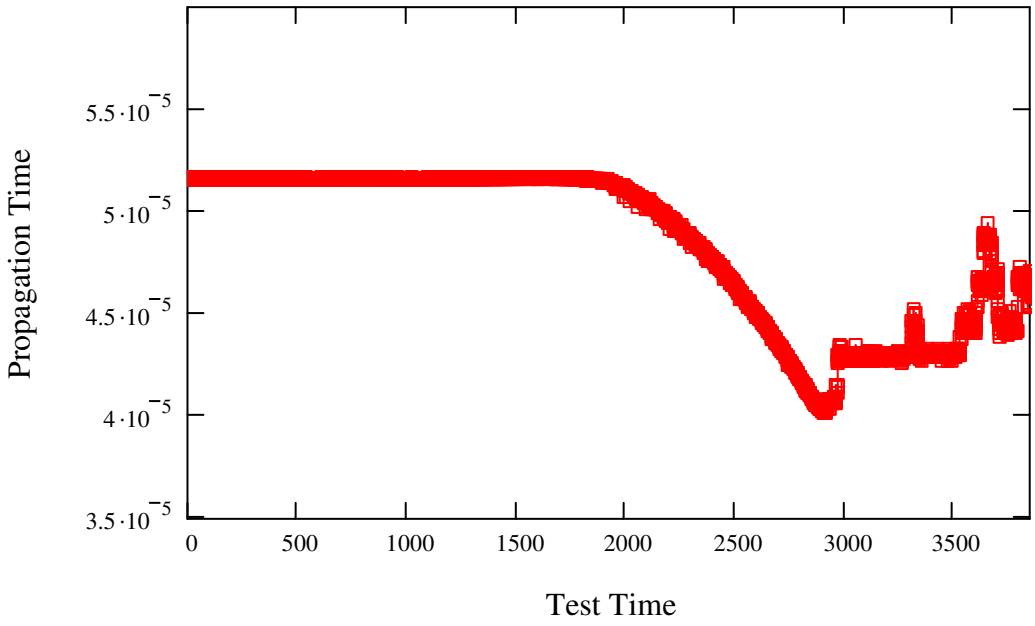


Figure B.24: Test #4, GM08-20 @ 145°F, Propagation Time, New Zero Crossing

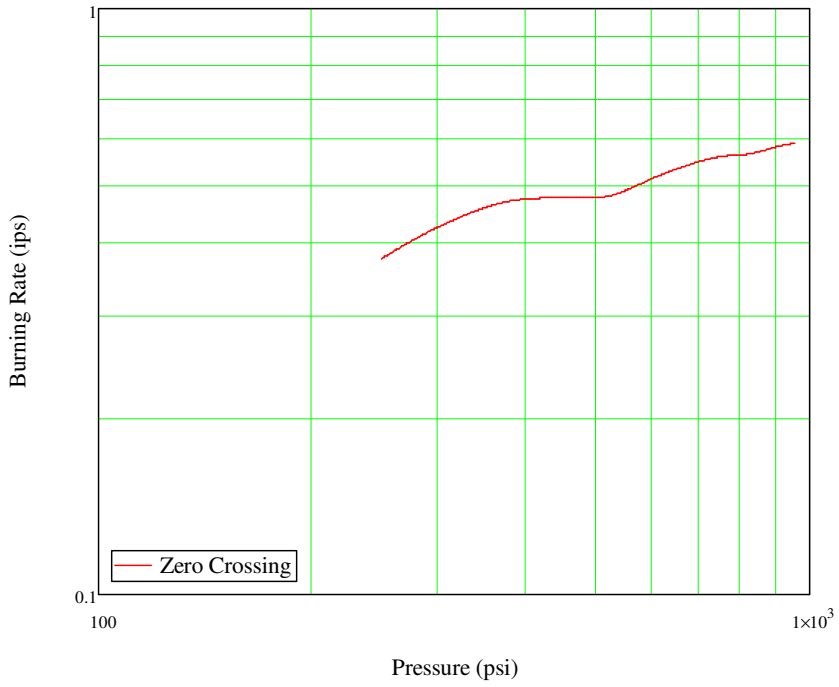


Figure B.25: Test #4, GM08-20 @ 145°F, Burn Rate, New Zero Crossing

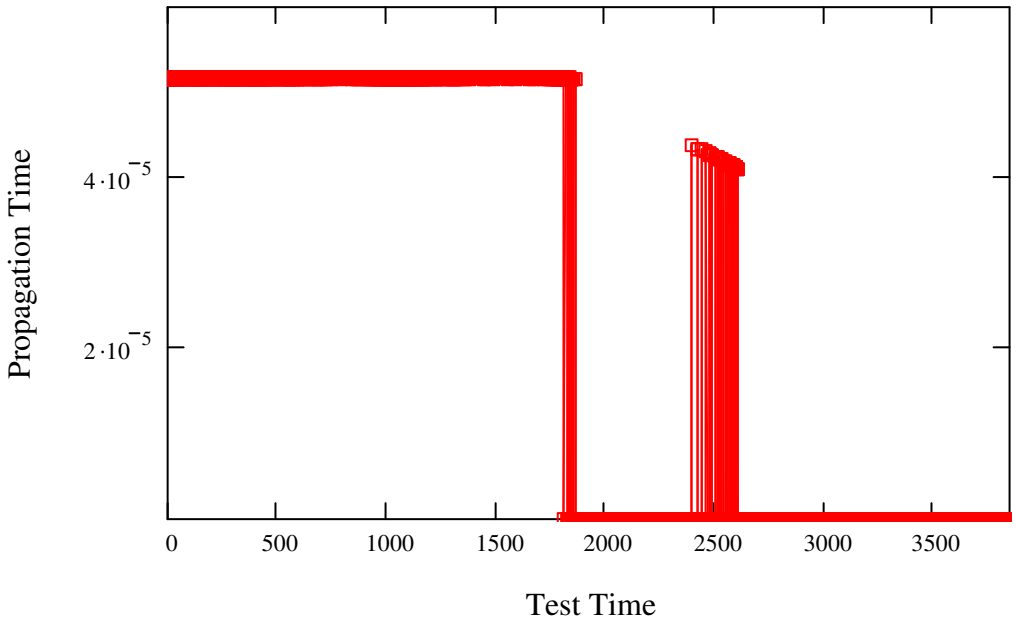


Figure B.26: Test #4, GM08-20 @ 145°F, Propagation Time, Original Zero Crossing

Figure B.26 shows that the propagation time could not be found by the Original Zero Crossing method for Test #4. As no propagation time was found, the burn rates could not be calculated.

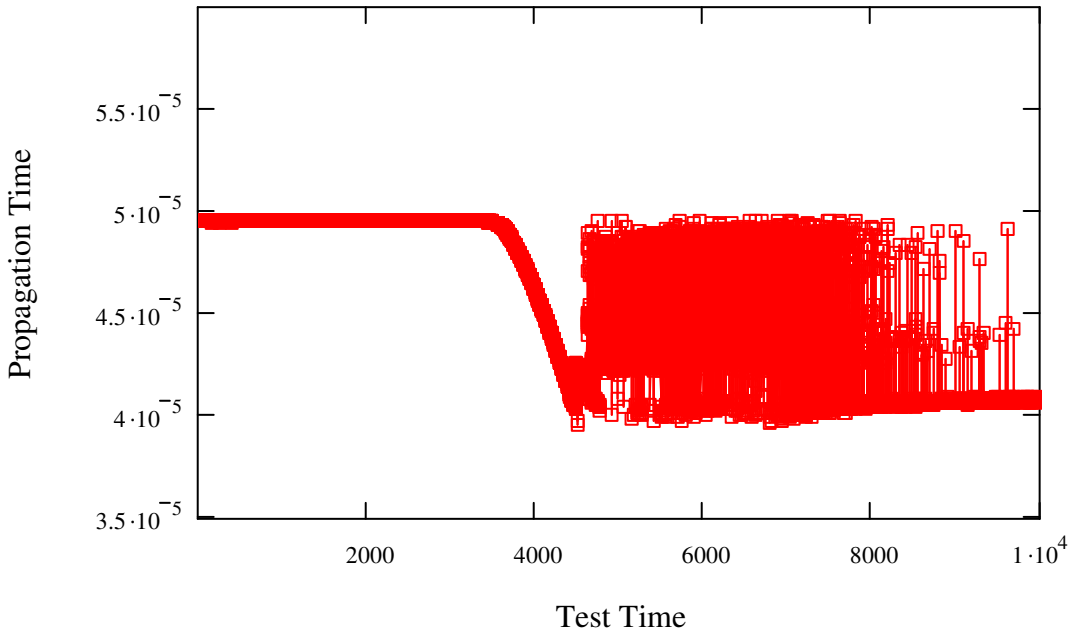


Figure B.27: Test #4, GM08-20 @ 145°F, Propagation Time, Cross Correlation

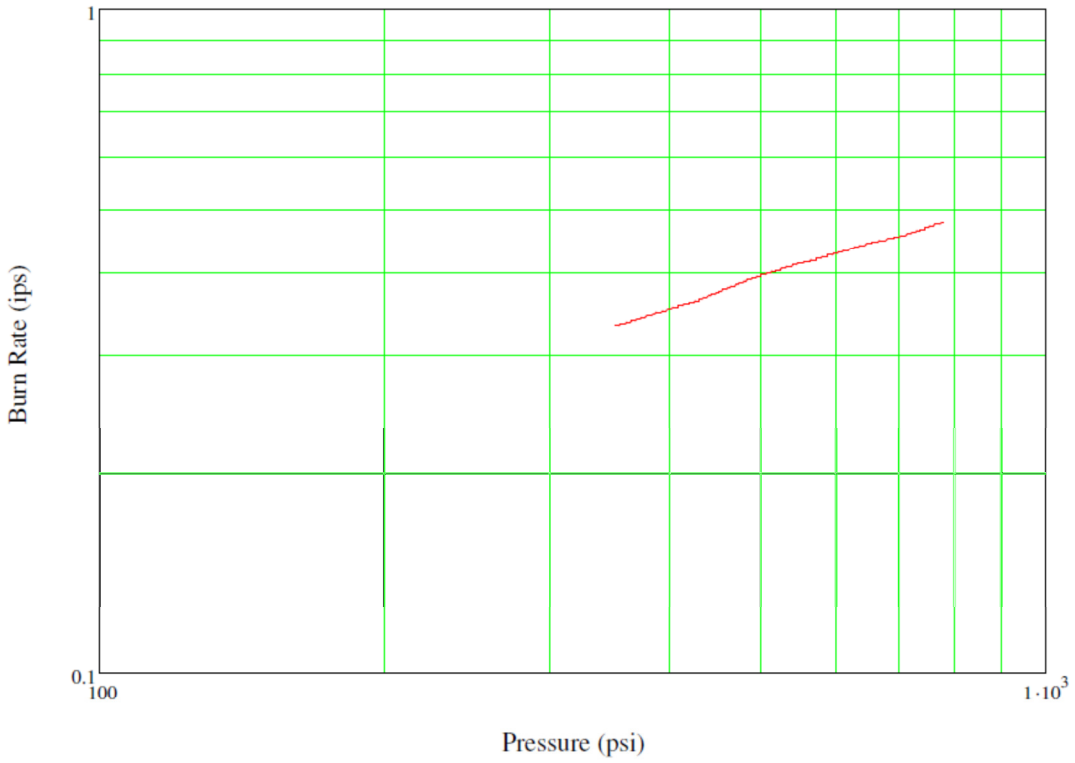


Figure B.28: Test #4, GM08-20 @ 145°F, Burn Rate, Cross Correlation

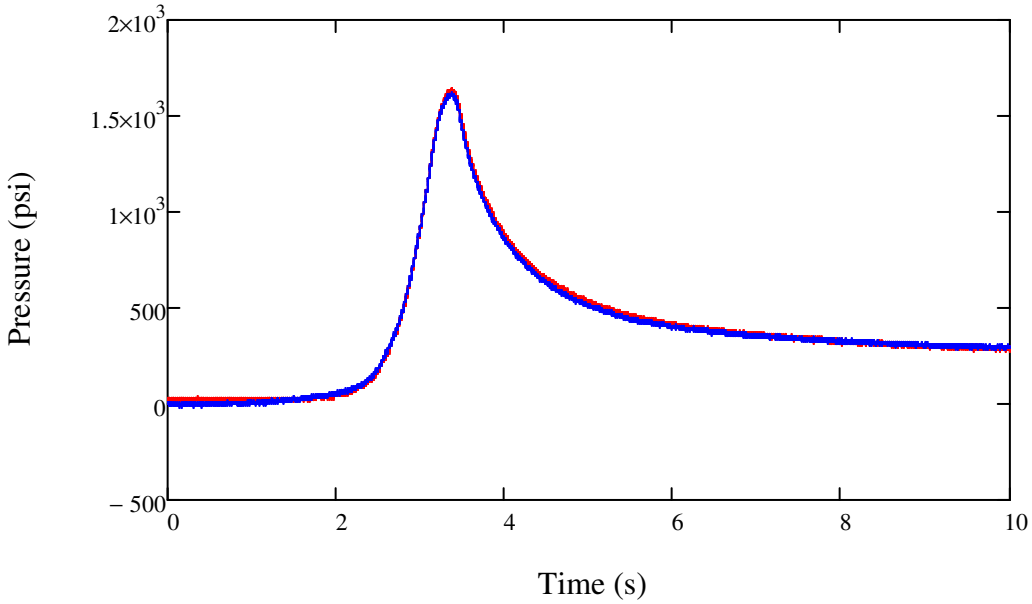


Figure B.29: Test #5, GM08-21 @ 145°F, Pressure Data

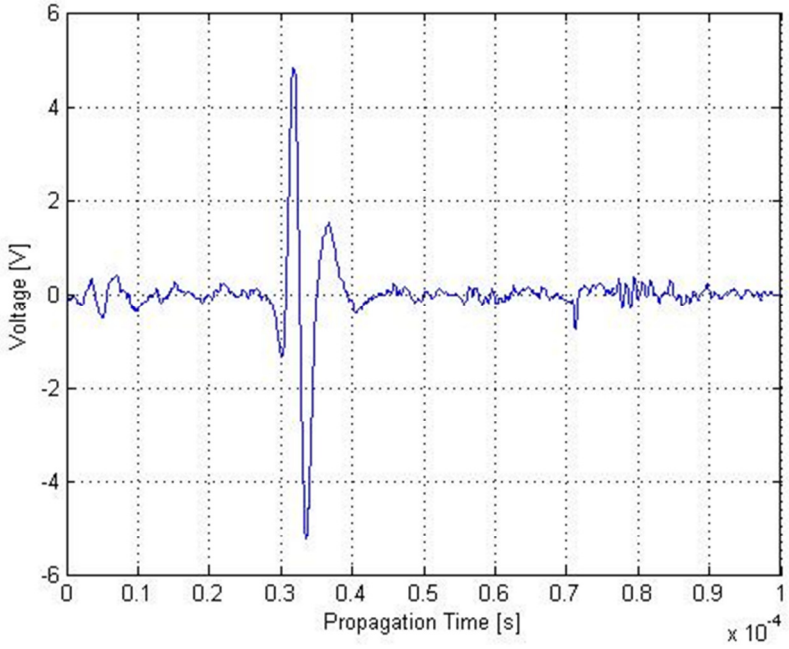


Figure B.30: Test #5, GM08-21 @ 145°F, at 0.0 Seconds

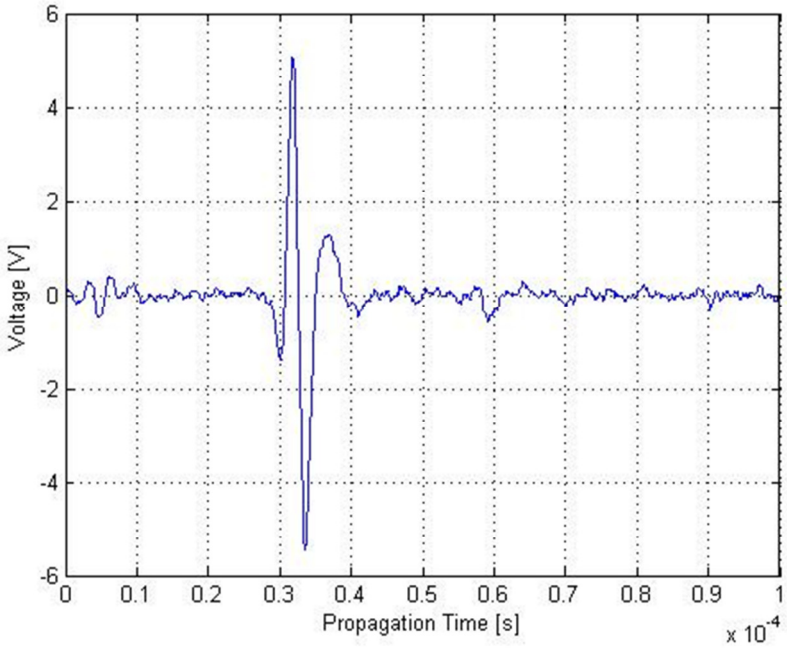


Figure B.31: Test #5, GM08-21 @ 145°F, at 1.0 Seconds

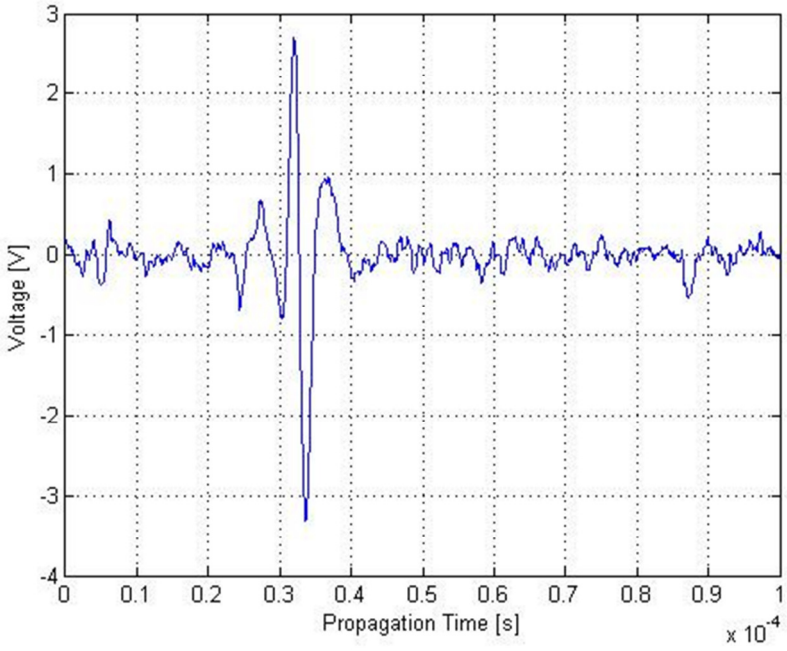


Figure B.32: Test #5, GM08-21 @ 145°F, at 2.0 Seconds

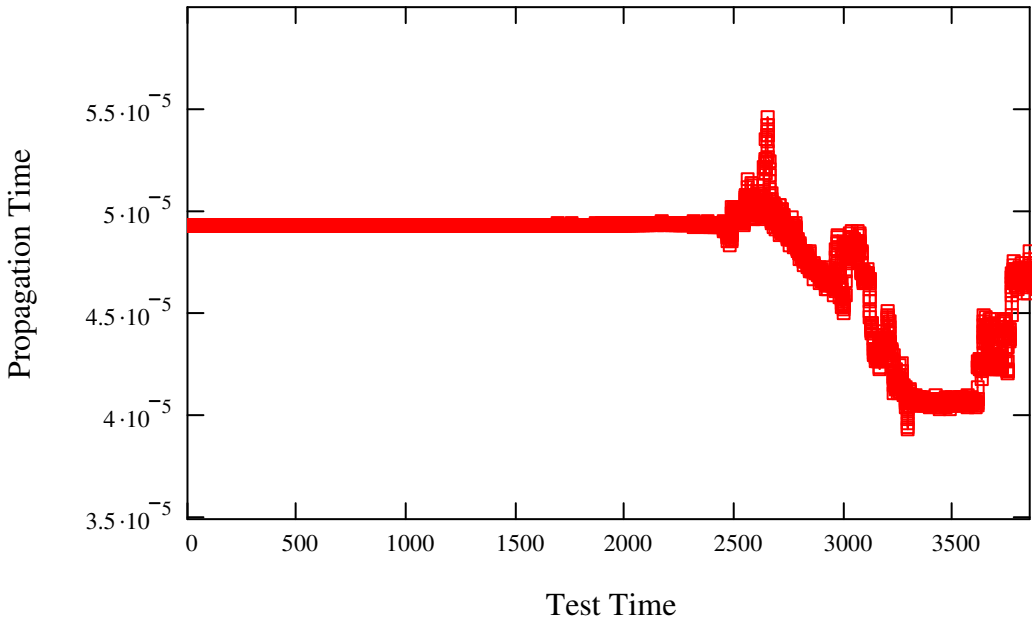


Figure B.33: Test #5, GM08-21 @ 145°F, Propagation Time, New Zero Crossing

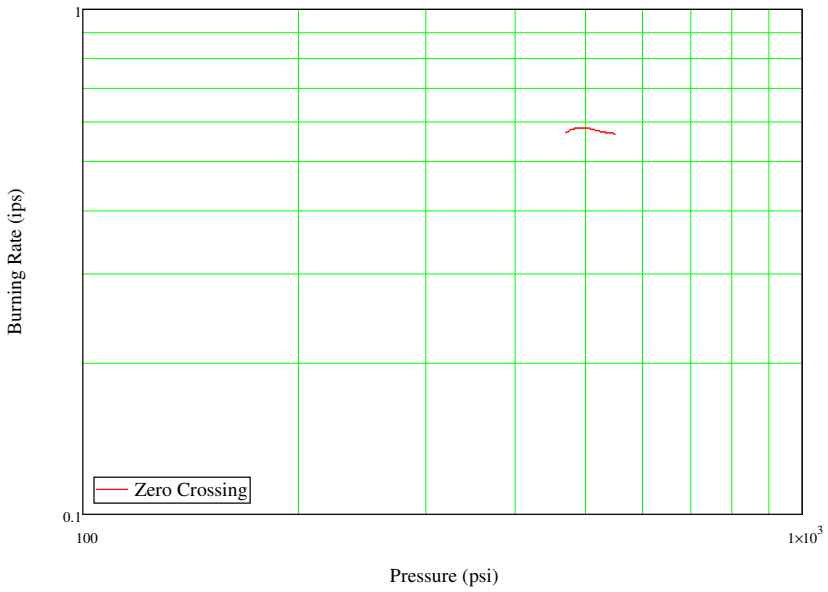


Figure B.34: Test #5, GM08-21 @ 145°F, Burn Rate, New Zero Crossing

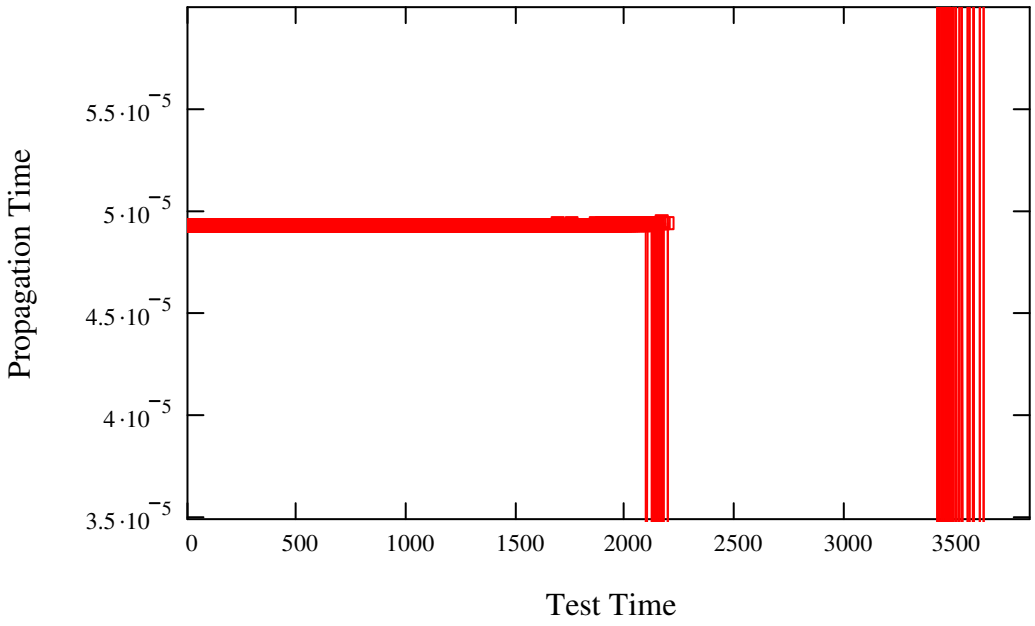


Figure B.35: Test #5, GM08-21 @ 145°F, Propagation Time, Original Zero Crossing

Figure B.35 shows that the propagation time could not be found by the Original Zero Crossing method for Test #5. As no propagation time was found, the burn rates could not be calculated.

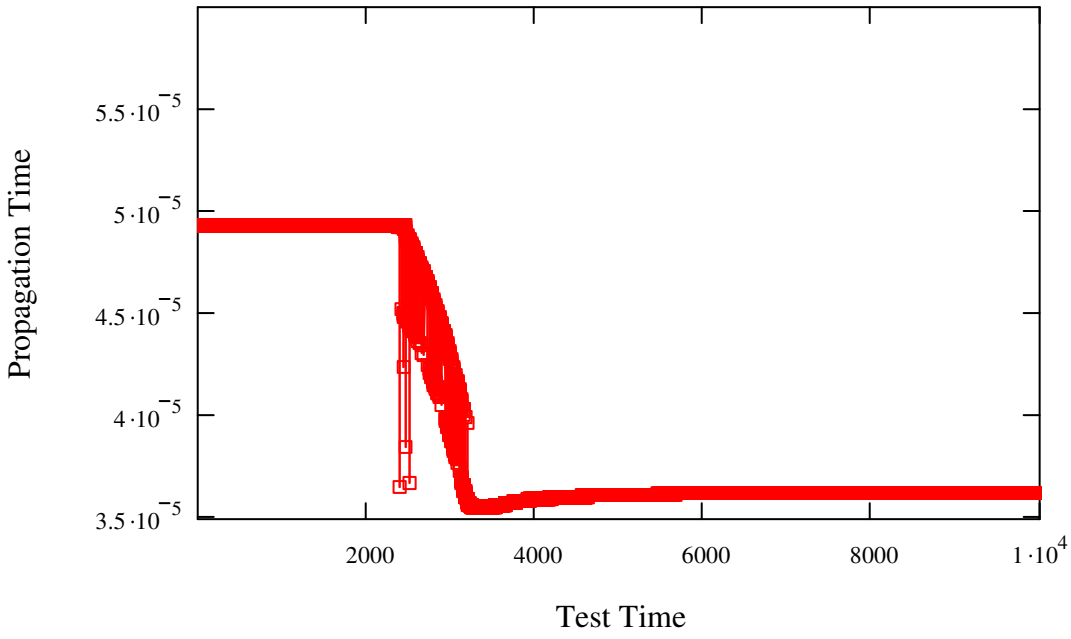


Figure B.36: Test #5, GM08-21 @ 145°F, Propagation Time, Cross Correlation

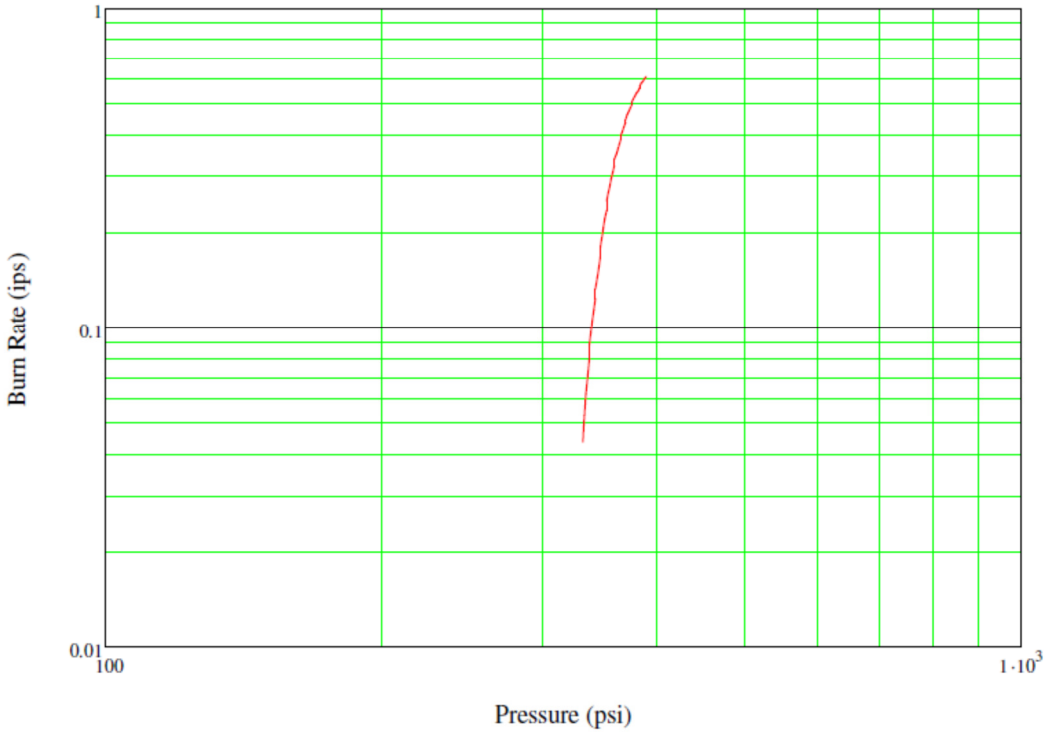


Figure B.37: Test #5, GM08-21 @ 145°F, Burn Rate, Cross Correlation

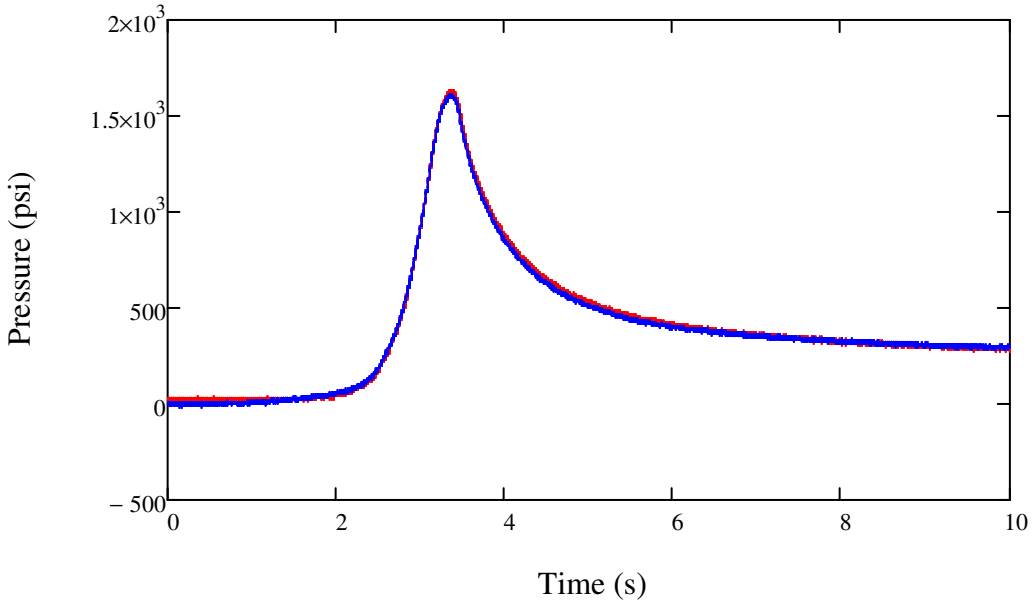


Figure B.38: Test #6, GM08-22 @ 145°F, Pressure Data

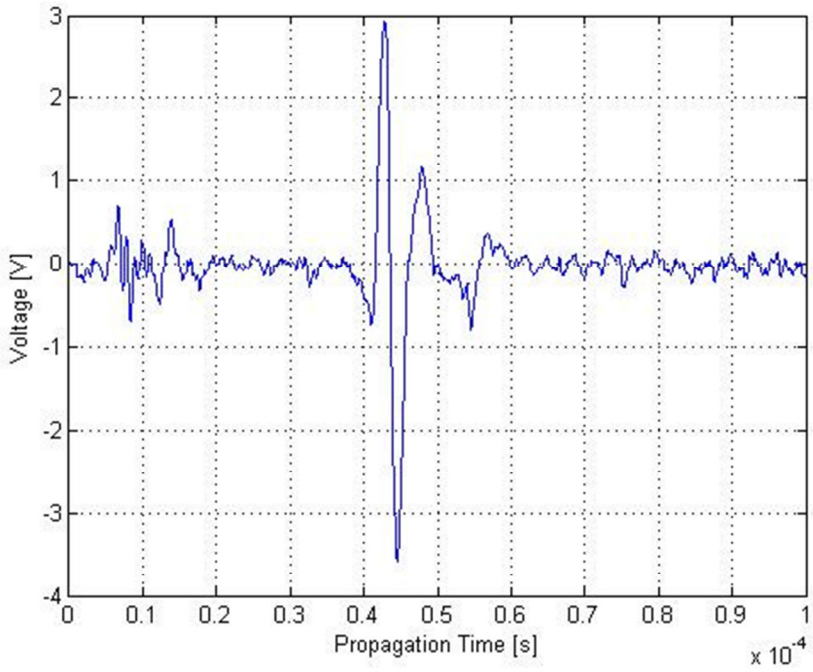


Figure B.39: Test #6, GM08-22 @ 145°F, at 0.0 Seconds

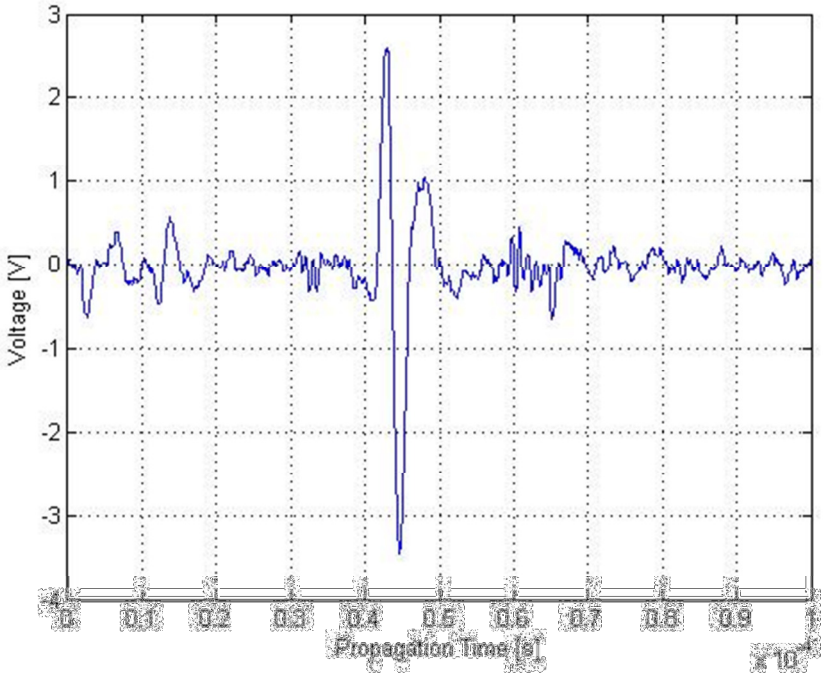


Figure B.40: Test #6, GM08-22 @ 145°F, at 1.0 Seconds

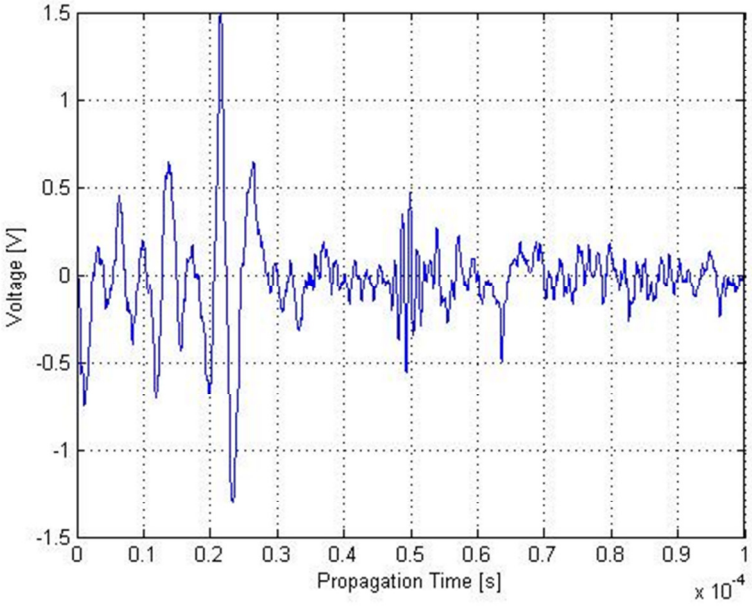


Figure B.41: Test #6, GM08-22 @ 145°F, at 2.0 Seconds

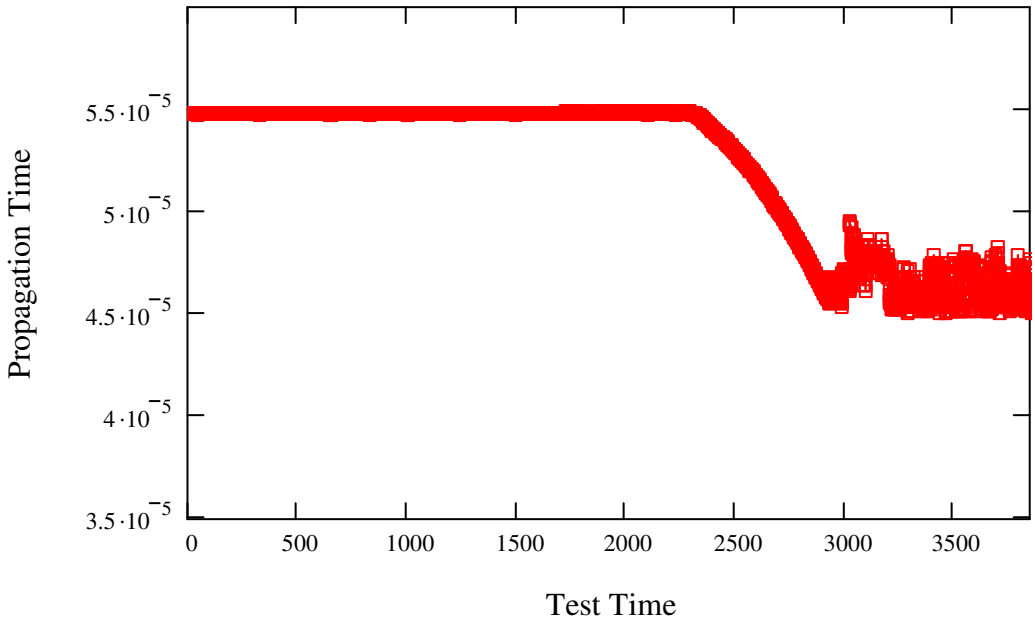


Figure B.42: Test #6, GM08-22 @ 145°F, Propagation Time, New Zero Crossing

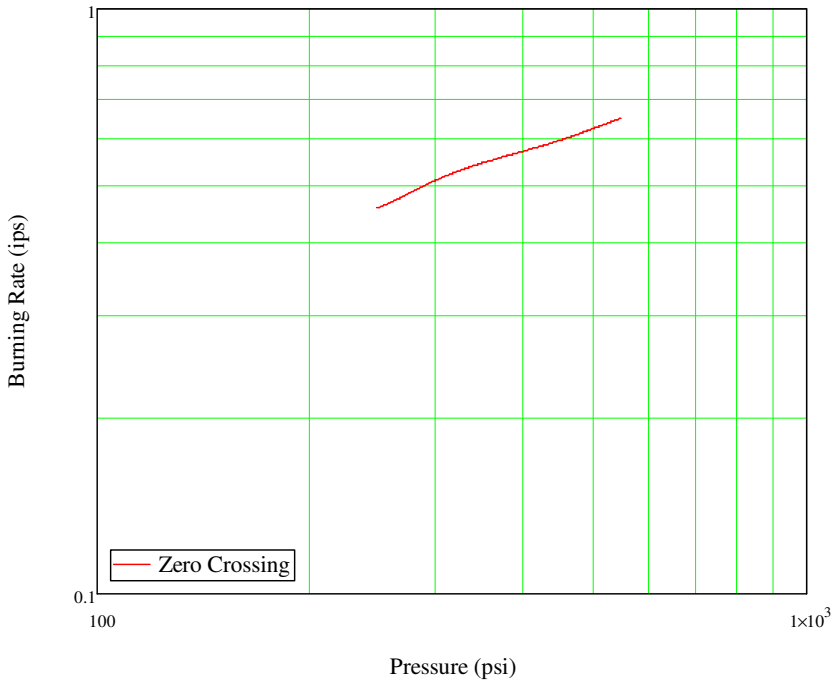


Figure B.43: Test #6, GM08-22 @ 145°F, Burn Rate, New Zero Crossing

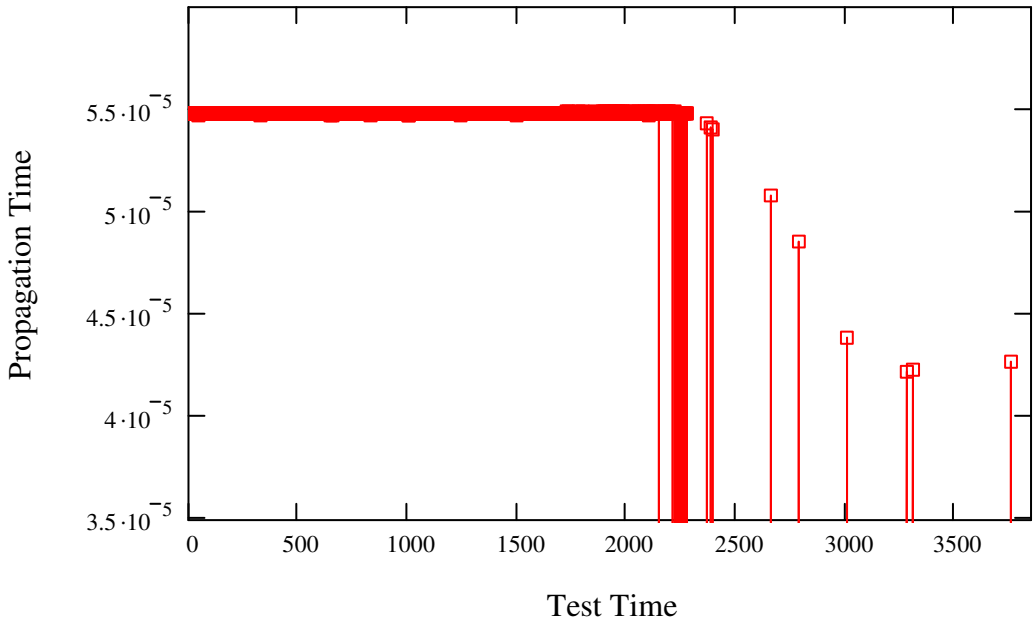


Figure B.44: Test #6, GM08-22 @ 145°F, Propagation Time, Original Zero Crossing

Figure B.44 shows that the propagation time could not be found by the Original Zero Crossing method for Test #6. As no propagation time was found, the burn rates could not be calculated.

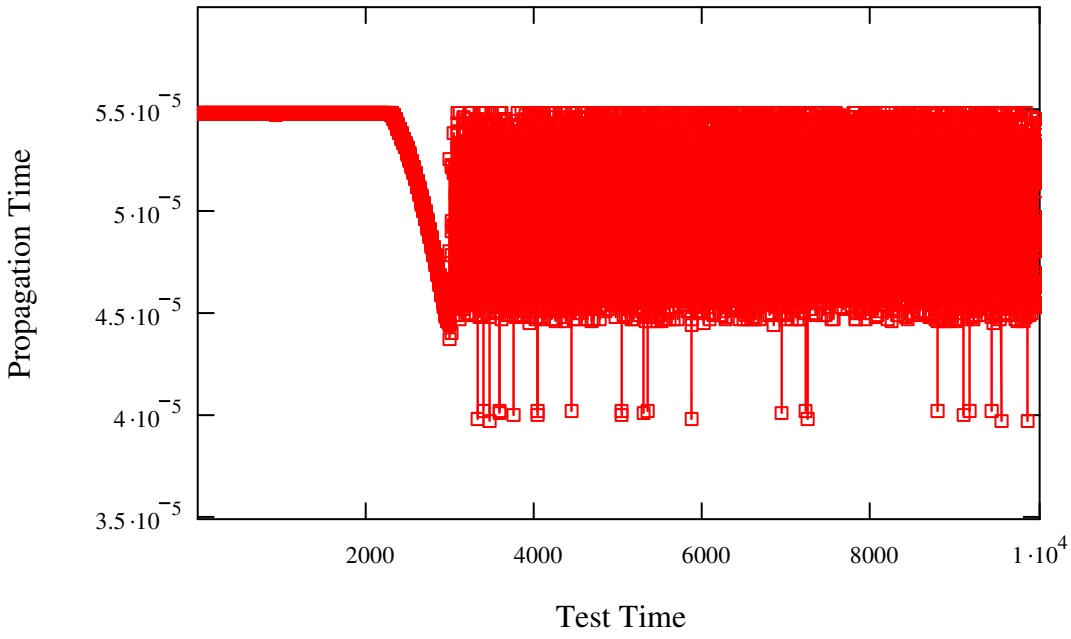


Figure B.45: Test #6, GM08-22 @ 145°F, Propagation Time, Cross Correlation

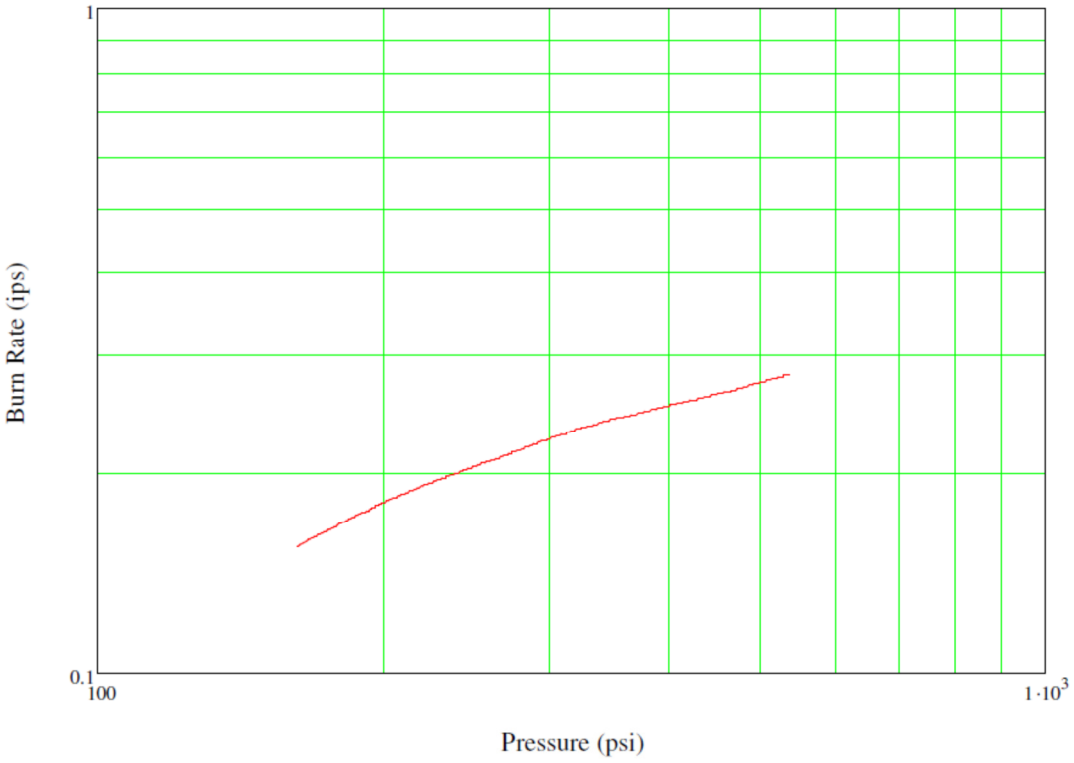


Figure B.46: Test #6, GM08-22 @ 145°F, Burn Rate, Cross Correlation

APPENDIX C

MatLab Code

The bulk of the code below was written by Marshall [21]. The highlighted code was rewritten for this investigation to include the new algorithm.

```
Function varargout = ZeroandCross(varargin)
% ZEROANDCROSS M-file for ZeroandCross.fig
%      ZEROANDCROSS, by itself, creates a new ZEROANDCROSS or raises the existing
%      singleton*.
%
%      H = ZEROANDCROSS returns the handle to a new ZEROANDCROSS or the handle to
%      the existing singleton*.
%
%      ZEROANDCROSS('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in ZEROANDCROSS.M with the given input arguments.
%
%      ZEROANDCROSS('Property','Value',...) creates a new ZEROANDCROSS or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before ZeroandCross_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to ZeroandCross_OpeningFcn via varargin.
```

```
%
%     *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ZeroandCross

% Last Modified by GUIDE v2.5 17-Jul-2008 12:42:50

% Begin initialization code – DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ZeroandCross_OpeningFcn, ...
                   'gui_OutputFcn',  @ZeroandCross_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code – DO NOT EDIT


% --- Executes just before ZeroandCross is made visible.
Function ZeroandCross_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved – to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ZeroandCross (see VARARGIN)

% Choose default command line output for ZeroandCross
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ZeroandCross wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
Function varargout = ZeroandCross_OutputFcn(hObject, eventdata, handles)
```

```
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved – to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;




%------------------Editable Text Creation Functions------------------
function endzero_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function startzero_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function startcross_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function endcross_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function samplerate_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function trigdelay_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function pop_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end


function thresh_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end


function mask_CreateFcn(hObject, eventdata, handles)


if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

%--------------------------------------------------------------------

%--------------------------------------------------------------------

%--------------------------------------------------------------------

%--------------------------------------------------------------------

%----------------DO NOT EDIT ABOVE THESE LINES!!!--------------------
```

```
%--------------------------------------------------------------------

%--------------------------------------------------------------------

%--------------------------------------------------------------------


%------------------Editable Text Callback Functions------------------


function pop_Callback(hObject, eventdata, handles)


function samplerate_Callback(hObject, eventdata, handles)


function trigdelay_Callback(hObject, eventdata, handles)


function fileload_Callback(hObject, eventdata, handles)


[filename, pathname] = uigetfile('*.dat', 'Select Waveform File')


a = [pathname filename];


wave = load (a);


%[x,y] = size(wave)

%

%i = 1;

%

%progressbar;
```

```
%autogain digitally, added 10/29/2008
%while i <= x
%
%    p = max(wave(i,☺);
%
%    q = 5/p;
%
%    newwave(i,☺ = wave(i,☺*q;
%
%    i = i +1;
%
%    progressbar(i/x)
%end

set(handles.fileload, 'UserData', wave);

axes(handles.fileloadzero_display);

cla;

plot(wave(25,☺);

grid on

hold on
```

```
axes(handles.fileloadcross_display);

cla;

plot(wave(25,☺);

grid on

hold on

set(handles.fileload_display, 'String', filename)

function thresh_Callback(hObject, eventdata, handles)
axes(handles.fileloadzero_display);

cla;

wavefile = get(handles.fileload, 'UserData');

plot(wavefile(25,☺);

hold on

threshold = str2double(get(handles.thresh, 'String'));
```

```
range = [0:5000];

plot(range , threshold ,'m');

hold on

function startzero_Callback(hObject, eventdata, handles)
axes(handles.fileloadzero_display);

cla;

Wavefile = get(handles.fileload, 'UserData');

plot(Wavefile(25,☺);

hold on

Threshold = str2double(get(handles.thresh, 'String'));

length = [0:5000];

plot(length,Threshold,'m');

hold on
```

```
startpoint = str2double(get(handles.startzero, 'String'));

range = [-10 : .025 : 10];

plot(startpoint , range , 'g');

hold on

function endzero_Callback(hObject, eventdata, handles)
axes(handles.fileloadzero_display);

cla;

Wavefile = get(handles.fileload, 'UserData');

plot(Wavefile(25,☺);

hold on

threshold = str2double(get(handles.thresh, 'String'));

range = [0:5000];

plot(range , threshold , 'm');
```

```
hold on

range = [-10 : .025 : 10];

startpoint = str2double(get(handles.startzero, 'String'));

plot(startpoint , range , 'g')

hold on

endpoint = str2double(get(handles.endzero, 'String'));

plot(endpoint , range , 'r');

hold on

function zoomzero_Callback(hObject, eventdata, handles)

axes(handles.zoomzero_graph);

cla;

start_sub_array = str2double(get(handles.startzero, 'String'));

end_sub_array = str2double(get(handles.endzero, 'String'));
```

```
wavefile = get(handles.fileload, 'UserData');

sub_array = wavefile(25 , start_sub_array : end_sub_array);

plot(sub_array);

grid on;

hold on;

function mask_Callback(hObject, eventdata, handles)
axes(handles.zoomzero_graph);

cla;

start_sub_array = str2double(get(handles.startzero, 'String'));

end_sub_array = str2double(get(handles.endzero, 'String'));

wavefile = get(handles.fileload, 'UserData');

sub_array = wavefile(25 , start_sub_array : end_sub_array);

plot(sub_array);
```

```
grid on;

hold on;

mask = str2double(get(handles.mask, 'String'));

range = [-2 : .025 : 2];

plot(mask , range , 'g');

hold on

function getzero_Callback(hObject, eventdata, handles)

axes(handles.tauzero);

cla;

startpoint = str2double(get(handles.startzero, 'String'));

mask = str2double(get(handles.mask, 'String'));

start_sub_array = startpoint + mask;
```

```
trigdelay = str2double(get(handles.trigdelay, 'String'));

wave = get(handles.fileload, 'Userdata');

samplerate = str2double(get(handles.samplerate, 'String'));

actual_samplerate = samplerate * 1000000;

time_division = 1/actual_samplerate;

trigger_delay = trigdelay * 10^-6

f = size(wave);

lastcol = f(1,2);

p = lastcol -1;

t = p *time_division;

w = [0:time_division:t];

v = w + trigger_delay;

wave = [v ; wave];
```

```
maskedarray = wave(:,start_sub_array:lastcol);

86hreshold = str2double(get(handles.thresh, 'String'));

taus = GimmieMyTau(maskedarray, 86hreshold)

set(handles.getzero, 'UserData', taus);

axes(handles.tauzero);

plot(taus)

hold on

function startcross_Callback(hObject, eventdata, handles)

axes(handles.fileloadcross_display);

cla;

Wavefile = get(handles.fileload, 'UserData');

plot(Wavefile(25,☺);
```

```
hold on

length = [0:5000];

hold on

startpoint = str2double(get(handles.startcross, 'String'));

range = [-10 : .025 : 10];

plot(startpoint , range , 'g');

hold on


function endcross_Callback(hObject, eventdata, handles)

axes(handles.fileloadcross_display);

cla;

Wavefile = get(handles.fileload, 'UserData');

plot(Wavefile(25,☺);
```

```
hold on

range = [0:5000];

hold on

range = [-10 : .025 : 10];

startpoint = str2double(get(handles.startcross, 'String'));

plot(startpoint , range , 'g')

hold on

endpoint = str2double(get(handles.endcross, 'String'));

plot(endpoint , range , 'r');

hold on

function zoomcross_Callback(hObject, eventdata, handles)

axes(handles.zoomcross_graph);

cla;
```

```
start_sub_array = str2double(get(handles.startcross, 'String'));

end_sub_array = str2double(get(handles.endcross, 'String'));

wavefile = get(handles.fileload, 'UserData');

sub_array = wavefile(25 , start_sub_array : end_sub_array);

plot(sub_array);

grid on;

hold on;



function getcross_Callback(hObject, eventdata, handles)

axes(handles.taucross);

cla;

start_sub_array = str2double(get(handles.startcross, 'String'));

end_sub_array = str2double(get(handles.endcross, 'String'));
```

```
wave = get(handles.fileload, 'Userdata');

f = size(wave);

lastcol = f(1,2);

lastrow = f(1,1);

maskedarray = wave(:,start_sub_array:lastcol);

taus = Delay(maskedarray, start_sub_array, end_sub_array, lastcol, lastrow);

samplerate = str2double(get(handles.samplerate, 'String'));

actual_samplerate = samplerate * 1000000;

%taus = taus *-1;

taus = taus -1;

artificial_wave = get(handles.getzero, 'UserData');

kept = [1:20];

initial_tau = mean(artificial_wave(kept,1))
```

```
actual_initial_tau = initial_tau;

[rows_of_taus, columns_of_taus] = size(taus);

c = 1;
while c <= rows_of_taus
   d(c,1) = actual_initial_tau;
   c = c + 1;
end

nonconfigured_taus = taus/actual_samplerate;

actual_taus = d – nonconfigured_taus;

set(handles.getcross, 'UserData', actual_taus);

axes(handles.taucross);

plot(actual_taus)

hold on

%-----------------------File Menu Callbacks-------------------------
function saveaszero_Callback(hObject, eventdata, handles)
```

```
jack = get(handles.getzero, 'UserData');
val = get(handles.pop, 'Value');
if val == 1
   jill = 'Zero_Crossing_Pre_Test';
elseif val == 2
   jill = 'Zero_Crossing_Test';
else
   jill = 'Zero_Crossing_Post_Test';
end
save (jill, 'jack', '-ASCII');

function saveascross_Callback(hObject, eventdata, handles)
jack = get(handles.getcross, 'UserData');
val = get(handles.pop, 'Value');
if val == 1
   jill = 'Cross_Correlation_Pre_Test';
elseif val == 2
   jill = 'Cross_Correlation_Test';
else
   jill = 'Cross_Correlation_Post_Test';
end
save (jill, 'jack', '-ASCII');

function close_Callback(hObject, eventdata, handles)
close
```

```
function file_Callback(hObject, eventdata, handles)

%--------------------Heart of Code-----------------------

function [tauArray] = GimmieMyTau(maskedarray, 93hreshold)

   i = 2; %rows
   j = 1; %columns
   x = size(maskedarray);  %finding number of columns
   y = x(1,2); %number of columns
   z = x(1,1); %number of rows
   n = 0; %temporarily stores tauArray

   while i <= z

      if (i == 2)

      %increment 'j' until the 93hreshold is crossed or the
```

```
%last column is reached
while (maskedarray(i,j) < 94hreshold) && (j < y – 1)
  j = j + 1;
end
end
%increment ‘j’ until a negative number or zero is encountered
%or the last colimn is reached
while (maskedarray(i,j) * maskedarray(i,j+1)) > 0 && (j < y – 1)
    j = j + 1;

end

if (j < y – 2)
    n = ((((maskedarray(1,j+1)-maskedarray(1,j))/(maskedarray(i,j+1)-
maskedarray(i,j)))*maskedarray(i,j+1))-maskedarray(1,j+1))*-1;

else
    n = 0;
end
tauArray(i – 1,1) = n;
i = i + 1;

if (j > 50)
j = j – 50;
end
```

```
    end



function [tauArray] = Delay(maskedarray, startpoint, endpoint, lastcol, lastrow)

   x = 1; % Initializes x to 1

   n = lastrow;

   progressbar

   tauArray = zeros(n,1);

   while x <= lastrow %stops incrementing x to the number of rows (x is the waveform
number)

      tauArray(x,1) = Correlation(x, startpoint, endpoint, maskedarray, lastcol); %Calls
Correlation function (for waveform x) gives it a row to do its

                  % cross correlation on

      x = x +1; %increments x or rows

      progressbar(x/n)

   end



function [m] = Correlation(x, startpoint, endpoint, maskedarray, lastcol)  %the correlation
function calculates the combined area

    j = 1; %initializes j to 1

    k = 0; %initializes k to 0

    n = 1; %initializes n to 1

    q = 1;

    mim = endpoint – startpoint; %this sets the maximum number of shifts in the cross
correlation

    c = zeros(mim,1);
```

```
    while k < mim
        i = k + 1;
        n = 1;
        while i < k + mim %this while statement basically increments the two points its multiplying and puts the value in a matrix
            c(n,1) = maskedarray(1,i)*maskedarray(x,n); %multiplies the two points to get an area
            i = i + 1;
            n = n + 1;
        end
    p(j,1) = sum©; %this sums the the matrix to get an area under the two curves and stores that value in the p matrix
    j = j + 1;
    k = k + 1; %this is basically shifting the first waveform
    end
[C,m] = max(p);
```

APPENDIX D

Programs and Data Files

Table D.1: Programs

Filename	Description
Soak Determination	Labview Code to determine thermal soak of sample in chamber
Pressure Transducer Calibration	Labview Code to determine calibration constants for pressure transducers
Ultrasonic Tony's Excellent Adventure 10	Labview Code to collect data from test system
Zero and Cross	Matlab Code to translate data from Labview into propagation time
MathCad	Used to determine and analyze burn rates from Zero and Cross program

Table D.2: Data Files

Folder Name	Description
#81 29 Apr 2009	Data from GM08-20 @ 75°F
#82 27 May 2009	Data from GM08-21 @ 75°F
#83 2 June 2009	Data from GM08-22 @ 75°F
#84 9 June 2009	Data from GM08-20 @ 145°F
#85 16 June 2009	Data from GM08-21 @ 145°F
#91 5 Oct 2010	Data from GM08-22 @ 145°F

REFERENCES

[1] Sutton, G.P. 2001 *Rocket Propulsion Elements, An Introduction to the Engineering of Rockets,* 7th ed, John Wiley & Sons, Inc., New York, 2001.

[2] Cauty, F., "The ultrasound waves: A measurement tool for energetic material characterization," *40th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit*, AIAA 2004-4057, Fort Lauderdale, FL, July 2004.

[3] Frederick, R., *Evaluation of Methods for Solid Propellant Burning Rate Measurments Chapter 5 "Non-Intrusive Techniques,"* AGARD/RTO, January 2002.

[4] Traineau, J., Kuentzmann, P., "Ultrasonic measurement of solid propellant burning rates in nozzleless rocket motors," *Journal of Propulsion,* pp215-222, 1986.

[5] Cauty, F., Demarais, J.C., Erades, C., "Determination of solid propellant burning rate sensitivity to initial temperature by the ultrasonic method," *3rd International Symposium on Special Topics in Chemical Propulsion*, 1993-69, Scheveningne, The Netherlands, May 1993.

[6] Frederick, R. A., Traineau, J. C., Popo, M., "Review of ultrasonic technique for steady state burning rate measurements," *36th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit*, AIAA 2000-3801, Huntsville, AL, July 2000.

[7] Marshall, M. A., Evans, J. A., Frederick, F. A., Moser, M. D., "UAH solid propellant characterization," *43rd AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit*, AIAA 2007-5763, Cincinnati, OH, July 2007.

[8] Frederick, R. A., Moser, M. D., "Research in solid propellant ballistics at UAH," *41st AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit,* AIAA 2005-3620, Tucson, AZ, July 2005.

[9] Smith, M. D., Moser, M. D., Frederick, F. A., "Temperature sensitivities of energetic binder propellants," *36th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit,* AIAA 2000-3319, Huntsville, AL, July 2000.

[10] DiSalvo, R., Frederick, R. A., Moser, M. D., "Direct ultrasonic measurement of solid propellant combustion transients," *35th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit,* A99-31080, Los Angeles, CA, June 1999.

[11] DiSalvo, R., Frederick, R. A., Moser, M. D., "Experimental determination of solid propellant combustion response," *AIAA Journal,* AIAA-98-3553, 1998.

[12] DiSalvo, R., Frederick, R. A., Moser, M. D., "Design and performance evaluation of components of micro solid propellant thruster," *40th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit,* AIAA 2004-3386, Fort Lauderdale, FL, July 2004.

[13] Linn, C. A. B., Jackson, D. E., Moser, M. D., "Use of cross correlation waveform analysis to determine burning rate using ultrasonic technique," *42nd AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit*, AIAA 2006-4773, Sacramento, CA, July 2006.

[14] Serin, N., Chiaverini, M. J., Harting, G. C., Kuo K. K., "Pressure correction of ultrasonic regression rate measurements of a hybrid slab motor," *35th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit,* AIAA 99-2319, Los Angeles, CA, June 1999.

[15] Murphy, J. J., Krier, H., "Ultrasound measurements of solid propellant burning rates: Theory and application," *AIAA Journal,* pg272, 1998.

[16] Murphy, J. J., Martin, A. O., Krier, H., "Precision techniques for measuing burning rates of solid propellants during pressure transients," *36th Aerospace Sciences Meeting and Exhibit,* A98-16411, Reno NV, January 1998.

[17] Murphy, J. J., Chai S., Bedar, C. R., Krier, H., "Response function measurement using an ultrasonic technique in an oscillatory burner," *36th AIAA/ ASME/ SAE/ ASEE Joint Propulsion Conference and Exhibit,* AIAA 2000-3797, Huntsville, AL, July 2000.

[18] Murphy, J. J., Krier, H., "Evaluation of ultrasonic technique for solid propellant burning rate response measurements," *Journal of Propulsion and Power,* 18(3):641-651, May-June 2002.

[19] Hafenrichter, T. J., Murphy, J. J., Krier, H., "Ultrasonic measurement of the pressure-coupled response function for composite solid propellants," *Journal of Propulsion and Power,* 20(1):110-119, January-February 2004.

[20] DiSalvo, R., "Coupling Effects in the Ultrasonic Burn Rate Technique," Thesis, The University of Alabama in Huntsville, 1996.

[21] Marshall, M. A., "Methods of Analysis of Ultrasonic Echoes to Determine Solid Propellant Burning Rates," Thesis, The University of Alabama in Huntsville, 2008.

[22] Dauch, F. T., Uncertainty Analysis of the Ultrasonic Technique Applied to Solid Propellant Burning Rate Measurement, Thesis, The University of Alabama in Huntsville, 1999.

[23] Evans, J. A., Penton, A. F., Frederick, R. A., "Uncertainty of Solid Propellant Burning Rate Measurements," AIAA2009-4975, presented at *45th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, August 2009.

[24] Coleman, H. W., Steele, W. G., "Experimentation and Uncertainty Analysis for Engineers," John Wiley & Sons, 1999.