ASEN 2004 Lab 2 Static Test Stand Report

Hannah Blanchard Obolsky, Luca Bonarrigo, Nick Larson, Jason Popich, Ryan Sievers, Mark Wilbourne
Smead Aerospace Engineering Sciences; University of Colorado - Boulder

A static test fire of a rocket can provide useful information on the propulsion phase such as the specific impulse (I_{sp}) and max thrust. This report takes given test stand data and generates a force curve plot in order to determine these values. To do this, the data must be analyzed to determine the proper start and stop times of the test and certain systematic errors of the test design must be factored out. Then the area under the force curve can be calculated to derive I_{sp} and associated values of max/average thrust with standard deviation can be found. This report found these values to be 1.58s for I_{sp} and 208 ± 20 N for peak thrust. This report finds key values of a water bottle rocket's thrust performance, which will be useful in future experimental modelling.

Nomenclature

A_e	=	nozzle exit area [m^2]
F	=	force [N]
g_0	=	sea-level gravity [$\frac{m}{s^2}$]
I_{sp}	=	specific impulse [s]
I_t	=	total impulse [N-s]
$\dot{m}$	=	mass flow rate [$\frac{kg}{s}$]
m_{prop}	=	mass of propellant [kg]
N	=	number of trials conducted
p_0	=	atmospheric pressure [Pa]
p_e	=	exit pressure [Pa]
PDF	=	Probability Distribution (Function)
σ	=	standard deviation
SEM	=	Standard Error of the Mean
T	=	thrust [N]
V_e	=	exhaust velocity [m/s]
$\bar{X}$	=	sample mean
z	=	Z-value for confidence interval

I. Introduction and Theory

All rockets are bound by Newtonian laws. These laws serve as a foundation through which engineers can understand the behavior of rockets. Newton's third law states that forces have equal and opposite reactions; this law can be combined with the definition for impulse (force applied over a certain period of time; see Eq. 1) and Newton's second law, $\Sigma F = ma$, to develop models of rocket behavior.

$$I_t = \int F(t)dt = F\Delta t \tag{1}$$

These equations can be used to model thrust, or the reaction force that launches rockets. Thrust depends on the mass of fuel or propellant, m_{prop}, its flow rate $\dot{m}$ through the engine(s), the exit (or exhaust) velocity V_e, and the pressure at the exit of the nozzle p_e [2]. The equation for thrust of a rocket is shown with Eq. 2.

$$T = \dot{m}V_e + A_e(p_e - p_0) \tag{2}$$

Normalizing the efficiency of a rocket is important due to differences between measurement systems, such as Imperial versus metric. Thus, the impulse is normalized by dividing the total impulse I_t by the weight of the propellant (i.e., m_{prop} times g_0) (see **Eqn. 4**). This normalized property is known as specific impulse, denoted by I_{sp}.

$$I_{sp} = \frac{I_t}{m_{prop}g_0} = \frac{\int F(t)dt}{m_{prop} * g_0} \tag{3}$$

Specific impulse is defined as a measure of a rocket's efficiency and is measured in seconds, which allows for easy comparison between different measurement systems. It is a vital value used in many applications, including the rocket equation. A rocket with a high momentum transfer to weight ratio is considered more efficient; thus, the higher the specific impulse, the more efficient the rocket. Depending on the situation, a high specific impulse might be less desirable than meeting other requirements.

For this lab, various static fire tests were run to determine the specific impulse of a pressurized water bottle rocket. These static fire tests measured the force of thrust of the rocket over time; after data collection, the specific impulse I_{sp} could be found using 1 and 3. However, as with all experimentation, various sources of error can easily reduce the accuracy of the calculations and estimate of I_{sp}. Thus, several tests were run to establish a desired degree of confidence in the results. The number of trials necessary was calculated to determine the qualitative uncertainty in modelling the rocket's flight path using the number of trials provided for this lab.

In addition, the data was further analysed using statistical methods including the Standard Error of the Mean (SEM), which measures the discrepancy between the experimental mean of the data and the true mean (in this case, the actual I_{sp}. Calculating and using the SEM allows for the estimation of the actual I_{sp} to be given within a smaller interval of higher confidence.

II. Materials and Methods

A. Test Set-Up

The test set up for the static test fires required a coke bottle, which served as the rocket body, a test stand, a pressurization attachment, and an air tank for pressurization of the rocket.

The coke bottle was first filled up with 1000 mL of water and attached to the pressurization system, which consisted of an air line, a rubber stopper, and a check valve. To avoid error, it was ensured there was no water in the tubing before the bottle (i.e., the rocket body) was attached to the launch plate. All four clamps were properly fastened to the launch plate to ensure the bottle wouldn't launch and the bottle was checked again for water leakage. Finally, the bottle was subsequently pressurized to 40 psi using the air tank. Figure 1 depicts the final set up of the test stand and bottle rocket.

B. Data Collection and Analysis

Data for the static fire test was collected automatically with the use of a VI console, which was connected to conditioning boxes with couplings set to DC with a bias voltage of 11V. Prior to the test, the sensors were zeroed out to avoid any erroneous collection of data. Figure 2 depicts the set up of the VI console and software used to collect data from the static fire.

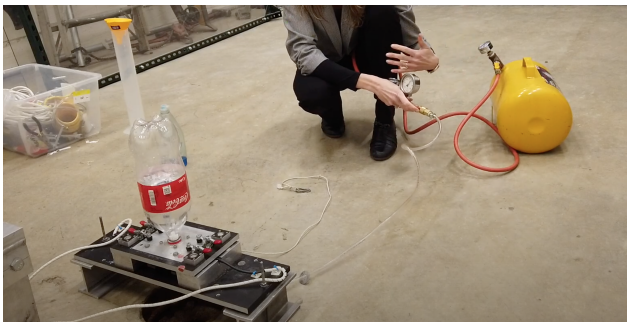

Fig. 1 Static Fire Test Set-Up

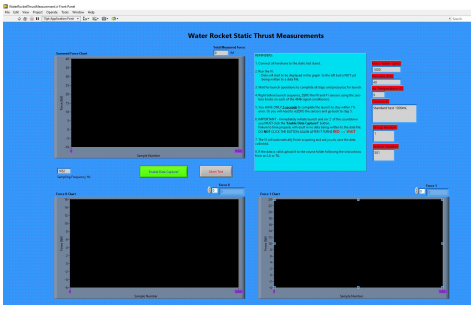

Fig. 2 VI Console Set-Up

Variations and changes in the thrust curve for the static fire test occur mainly as a product of the water and pressurized air, which act as the propellant, leaving the rocket as time increases from the start of the 'launch' (i.e., when the launch cord is pulled). The thrust curve may not return to zero by the end of the test if the sensor mistakenly and falsely records the system weight as a reading of thrust. Furthermore, the force sensor may also not return to its zero point after the test due to friction, which provides resistance to the sensor's returning motion after being compressed during the experiment. This friction would yield a low constant value of thrust at the end as the sensor is still compressed.

To determine the specific impulse from the thrust collected during the static fire tests, the datasets for each test were analysed using MATLAB (see Appendix B). The thrust measurements were converted from lbf to N and the time measurements from Hertz to seconds for continuity within SI units, and then 'cleaned'. This data cleanup of the thrust curve was done to avoid including any data from before or after the thrust test in calculations and thus reduce error in the data. The total impulse, I_t, was then calculated for each of the three data sets by integrating the area under the thrust curve through trapezoidal estimation. As outlined in Section I, the specific impulse I_{sp} for each trial was then calculated by normalizing the total impulse by the weight of the propellant.

III. Results

A. Descriptive Statistics

Table 1 outlines the peak thrusts, total thrust times, and calculated I_{sp} for each trial conducted in each of three lab sections, which were subsequently named data sets 1, 2, and 3. As can be seen in the table, two of the trials yielded outlying I_{sp} values, with trial 2.5 producing a very high I_{sp} and trial 3.8 producing a very low I_{sp}. Due to their outlying results, these two trials were removed to perform error and SEM analysis to avoid skewing of the data.

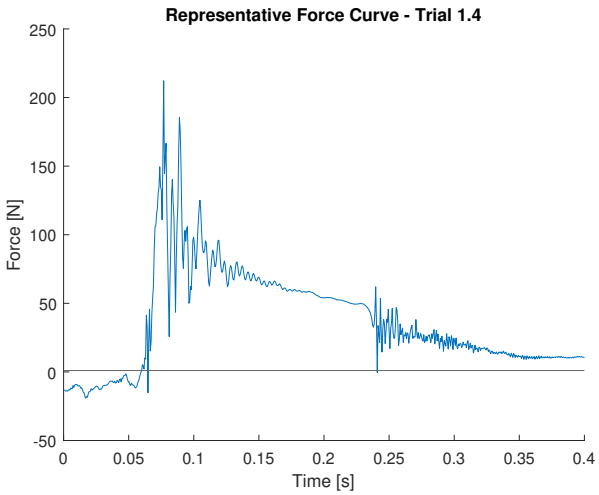

Fig. 3 Representative Force Curve

The sample mean for the I_{sp} was calculated to be 1.58 s. Additionally, the averages and standard deviations for the peak thrust and thrust time were determined to be 208 ± 20 N and 0.48 ± 0.16 seconds, respectively. A representative force curve, seen in Figure 3 was plotted using the trial that produced the closest to all three of these sample means, which was trial 1.4. A 'zero line' was added to represent the zeroing factor that was used for the entire dataset 1. The full force curves for all trials can be found in Appendix A.

Data Set	Trial	I_{sp} [s]	Peak Thrust [N]	Thrust Time [s]
1	1	1.46	221	0.45
	2	1.67	213	0.30
	3	1.23	173	0.48
	4	1.58	212	0.45
2	1	1.46	223	0.55
	2	1.64	190	0.61
	3	1.93	191	0.73
	4	1.43	212	0.73
	5	3.16	222	0.67
3	1	1.56	200	0.45
	2	1.53	201	0.45
	3	1.65	223	0.45
	4	1.84	205	0.52
	5	1.39	184	0.50
	6	1.93	195	0.45
	7	1.21	222	0.18
	8	0.122	188	0.12
	9	1.81	260	0.48

Table 1 Results by Dataset

B. SEM Analysis

The Standard Error of the Mean (SEM) is commonly used to estimate the variance between the actual population and the sample mean values. When given a number of samples, the SEM provides an indication of how accurate the sample mean is in estimating the true mean. The most accurate estimate produces the smallest SEM; thus, minimizing the SEM best estimates the probability distribution (aka PDF). The higher the deviation among the means, the higher the SEM.

The SEM number is useful for this report in order to better understand the expected I_{sp} value of the bottle rocket. The I_{sp} value in turn will help create a more accurate representation of what the bottle rocket will do once it is launched. SEM is calculated using:

$$SEM = \frac{\sigma}{\sqrt{N}} \tag{4}$$

where σ is the standard deviation and N is the number of trials.

Figure 4 shows the plot of SEM versus N, which varies the number of datasets between 1 and 16. As previously mentioned, two trials (2.5 and 3.8) were removed from the full data to prevent skewing in doing SEM analysis. As expected, the general trendline slopes downwards as the number of trials increases, due to more trials reducing deviation from the true mean.

Using Equation 4, the I_{sp} SEM was calculated to be 0.00551 s for the full dataset.

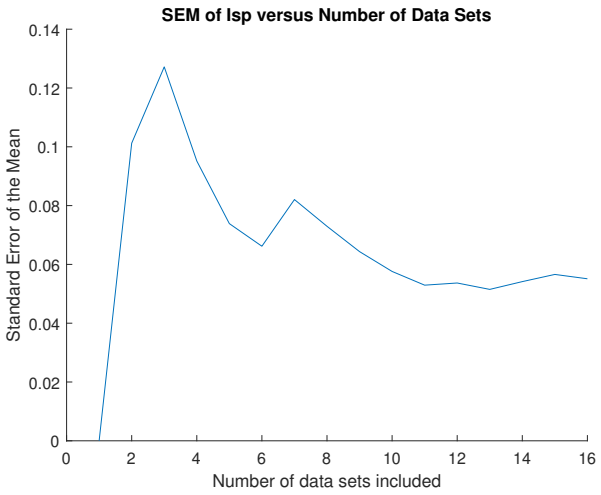

Fig. 4 Standard Error of the Mean vs Number of Trials

C. Confidence Analysis

Confidence intervals for 95%, 97.5%, and 99% confidence were calculated to analyse the uncertainty in the rocket thrust profile model, since there were many possible sources of error that meant the calculated value could not be given with 100% confidence. These confidence intervals were calculated using Equation 5, where z is the arbitrary z-values for each level of confidence:

$$\bar{X} \pm z * SEM \tag{5}$$

Given the interval 1.58 $\pm 0.11s$, the group is 95% confident that the true I_{sp} value lies within this range. Similarly, the 97% confidence interval is 1.58 ± 0.12 s, and the 99% confidence interval is 1.58 ± 0.14 s.

For further analysis, the number of test firings N necessary for the calculated mean I_{sp} value to be within 0.1s of the true I_{sp} value was calculated for each confidence interval. It was found that for the 95% confidence interval, 19 tests would need to be done to ensure a difference under 0.1s between the calculated and actual I_{sp} values. Similarly, 24 tests would be required for a 97.5% confidence interval, and 32 tests would be required for a 99% confidence interval. The number of tests required for the calculated I_{sp} to be withing 0.1s of the actual I_{sp} are fairly reasonable for the three confidence intervals, given that the 16 tests used for analysis yielded a 0.11 precision within 95% confidence. Finally, the process was repeated to find the number of test firings required for each of the three confidence intervals to ensure a precision of 0.01s. For the 95% confidence interval, 1867 tests would be required. For the 97.5% confidence interval, 2438 tests would be required, and for the 99% confidence interval, 3235 tests would be required. Such a high number of tests is largely unreasonable, and thus it can be concluded that this level of precision is unnecessary.

IV. Conclusion

As previously discussed, the computed number of test firings needed for 95% confidence is 19 for 0.1s precision, and 1867 for 0.01s precision. However, there were only 18 static test fires conducted for this lab, two of which produced I_{sp} estimates that were clearly outliers and were thus removed from the final estimate calculations. Because the total amount of trials conducted were less than the number calculated as necessary for a 0.1s precision within a 95% confidence interval, it can be reasonably concluded that the data will not be able to provide an estimate for the rocket's specific impulse within that confidence or precision. At least three more static test fires, and ideally as many more as possible, should be conducted to avoid larger uncertainty (i.e., a larger confidence interval) in modelling the rocket's flight performance.

Acknowledgments

Thank you to the ASEN 2004 instructional team and teaching assistants for their assistance on this project.

V. References

[1] Lee, Dong Kyu, Junyong In, and Sangseok Lee. "Standard deviation and standard error of the mean." Korean journal of anesthesiology 68.3 (2015): 220. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4452664/

[2] NASA. “Rocket Thrust.” Nasa.gov, 2014, www.grc.nasa.gov/www/k-12/rocket/rktth1.html.

[3] ASEN 2004, *ASEN 2004 Water Bottle Rocket Lab Deliverable 2: Static Test Stand Report*, University of Colorado at Boulder, Mar 2021.

[4] Test Stand and VI Console Set-Up Images from supplementary ASEN 2004 videos on YouTube.

Appendix

A. Extra Plots

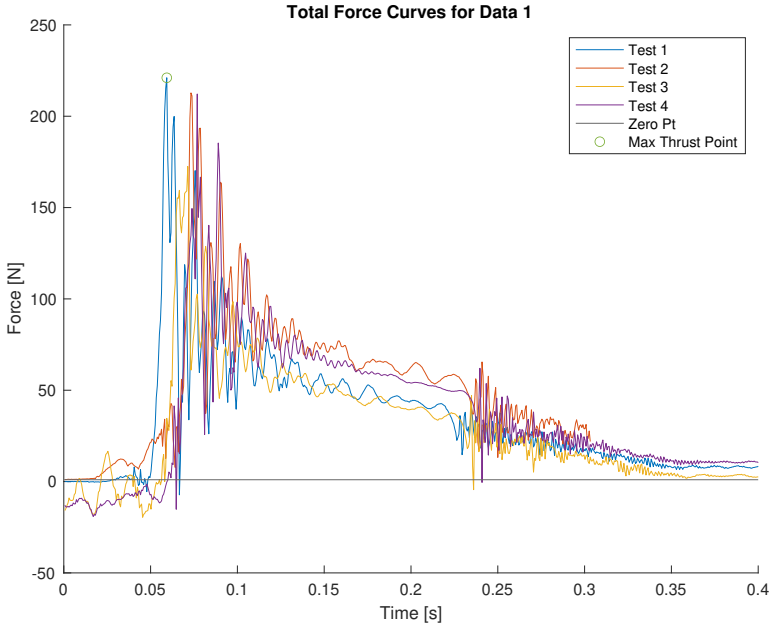

Fig. 5 S011 Test Trials

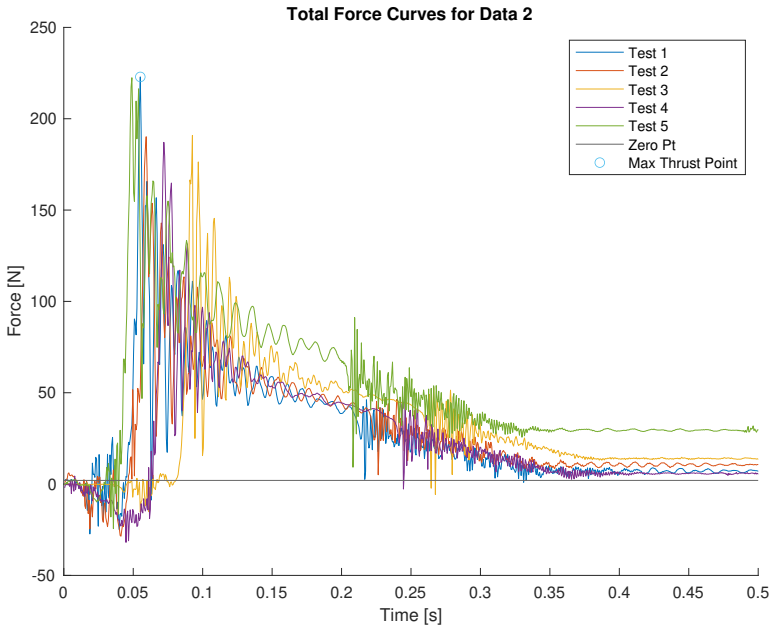

Fig. 6 S012 Test Trials

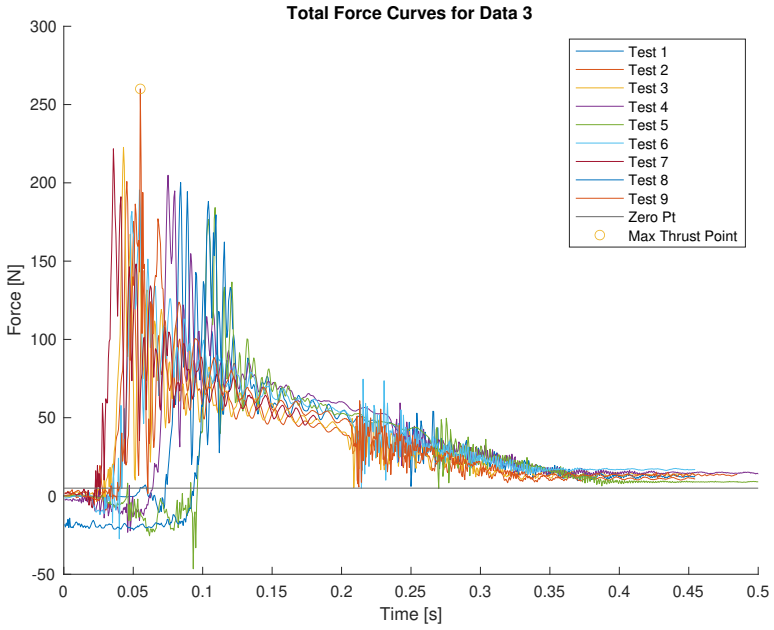

Fig. 7 S013 Test Trials

B. MATLAB Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                               %
%                          LAB 2 - ASEN 2004                                    %
%                       Static Test Fire Analysis                               %
%                                                                               %
%                                                                               %
%       The purpose of this code is to use static test                          %
%        stand data to determine the specific impulse                           %
%         for a bottle rocket and use error analysis                            %
%         techniques to determine the confidence of                             %
%                          the calculation                                      %
%                                                                               %
%                          Created: 03/17/2021                                  %
%                       Last Modified: 03/30/2021                               %
%                          Version: V9                                          %
%                                                                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Set Environment
clear
clc
close all

%% Define Variables
water_mass = 1;                                         %[kg]
pressure = 275790.3;                                    %[Pa]
g0 = 9.81;                                              %[m/s^2]


%% Load in Data
% Hard Code (iOS prevented from more effificent methodology)

% LOAD DATA 1
demo_11 = load('DATA/LA_Demo_S011_test1');
demo_12 = load('DATA/LA_Demo_S011_test2');
demo_13 = load('DATA/LA_Demo_S011_test3');
demo_14 = load('DATA/LA_Demo_S011_test4');

% Convert the Data
demo_11 = demo_11*4.44822162;                           %[lbf] to [N]
demo_12 = demo_12*4.44822162;                           %[lbf] to [N]
demo_13 = demo_13*4.44822162;                           %[lbf] to [N]
demo_14 = demo_14*4.44822162;                           %[lbf] to [N]

% LOAD 2
demo_21 = load('DATA/LA_Demo_S012_test1');
demo_22 = load('DATA/LA_Demo_S012_test2');
demo_23 = load('DATA/LA_Demo_S012_test3');
demo_24 = load('DATA/LA_Demo_S012_test4');
demo_25 = load('DATA/LA_Demo_S012_test5');

% Convert the Data
demo_21 = demo_21*4.44822162;                           %[lbf] to [N]
```

```
demo_22 = demo_22*4.44822162;                                   %[lbf] to [N]
demo_23 = demo_23*4.44822162;                                   %[lbf] to [N]
demo_24 = demo_24*4.44822162;                                   %[lbf] to [N]
demo_25 = demo_25*4.44822162;                                   %[lbf] to [N]

% LOAD 3
demo_31 = load('DATA/LA_Demo_S013_test1');
demo_32 = load('DATA/LA_Demo_S013_test2');
demo_33 = load('DATA/LA_Demo_S013_test3');
demo_34 = load('DATA/LA_Demo_S013_test4');
demo_35 = load('DATA/LA_Demo_S013_test5');
demo_36 = load('DATA/LA_Demo_S013_test6');
demo_37 = load('DATA/LA_Demo_S013_test7');
demo_38 = load('DATA/LA_Demo_S013_test8');
demo_39 = load('DATA/LA_Demo_S013_test9');

% Convert the Data
demo_31 = demo_31*4.44822162;                                   %[lbf] to [N]
demo_32 = demo_32*4.44822162;                                   %[lbf] to [N]
demo_33 = demo_33*4.44822162;                                   %[lbf] to [N]
demo_34 = demo_34*4.44822162;                                   %[lbf] to [N]
demo_35 = demo_35*4.44822162;                                   %[lbf] to [N]
demo_36 = demo_36*4.44822162;                                   %[lbf] to [N]
demo_37 = demo_37*4.44822162;                                   %[lbf] to [N]
demo_38 = demo_38*4.44822162;                                   %[lbf] to [N]
demo_39 = demo_39*4.44822162;                                   %[lbf] to [N]

%% Clean up the data
demo_11 = demo_11(3000:3750,3);
demo_12 = demo_12(2500:3000,3);
demo_13 = demo_13(2200:3000,3);
demo_14 = demo_14(3000:3750,3);

demo_21 = demo_21(2850:3750,3);
demo_22 = demo_22(2750:3750,3);
demo_23 = demo_23(800:2000,3);
demo_24 = demo_24(1550:2750,3);
demo_25 = demo_25(2650:3750,3);

demo_31 = demo_31(2750:3500,3);
demo_32 = demo_32(2250:3000,3);
demo_33 = demo_33(3000:3750,3);
demo_34 = demo_34(2900:3750,3);
demo_35 = demo_35(2175:3000,3);
demo_36 = demo_36(4500:5250,3);
demo_37 = demo_37(3200:3500,3);
demo_38 = demo_38(3300:3500,3);
demo_39 = demo_39(2700:3500,3);

%% Translate from Samples to Time
    % Preallocate
time_11 = zeros(size(demo_11));
time_12 = zeros(size(demo_12));
time_13 = zeros(size(demo_13));
```

```
time_14 = zeros(size(demo_14));
time_21 = zeros(size(demo_21));
time_22 = zeros(size(demo_22));
time_23 = zeros(size(demo_23));
time_24 = zeros(size(demo_24));
time_25 = zeros(size(demo_25));
time_31 = zeros(size(demo_31));
time_32 = zeros(size(demo_32));
time_33 = zeros(size(demo_33));
time_34 = zeros(size(demo_34));
time_35 = zeros(size(demo_35));
time_36 = zeros(size(demo_36));
time_37 = zeros(size(demo_37));
time_38 = zeros(size(demo_38));
time_39 = zeros(size(demo_39));


for i = 1:length(demo_11)
    time_11(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_12)
    time_12(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_13)
    time_13(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_14)
    time_14(i,1) = i/1652;                          %[Hz] to [s]
end


for i = 1:length(demo_21)
    time_21(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_22)
    time_22(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_23)
    time_23(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_24)
    time_24(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_25)
    time_25(i,1) = i/1652;                          %[Hz] to [s]
end


for i = 1:length(demo_31)
    time_31(i,1) = i/1652;                          %[Hz] to [s]
end
for i = 1:length(demo_32)
    time_32(i,1) = i/1652;                          %[Hz] to [s]
end
```

```
for i = 1:length(demo_33)
    time_33(i,1) = i/1652;                                  %[Hz] to [s]
end
for i = 1:length(demo_34)
    time_34(i,1) = i/1652;                                  %[Hz] to [s]
end
for i = 1:length(demo_35)
    time_35(i,1) = i/1652;                                  %[Hz] to [s]
end
for i = 1:length(demo_36)
    time_36(i,1) = i/1652;                                  %[Hz] to [s]
end
for i = 1:length(demo_37)
    time_37(i,1) = i/1652;                                  %[Hz] to [s]
end
for i = 1:length(demo_38)
    time_38(i,1) = i/1652;                                  %[Hz] to [s]
end
for i = 1:length(demo_39)
    time_39(i,1) = i/1652;                                  %[Hz] to [s]
end


%% Find Max Thrust
% Data 1
[max_data_11,idx_11] = max(demo_11);
[max_data_12,idx_12] = max(demo_12);
[max_data_13,idx_13] = max(demo_13);
[max_data_14,idx_14] = max(demo_14);

max_time_11 = time_11(idx_11);
max_time_12 = time_12(idx_12);
max_time_13 = time_13(idx_13);
max_time_14 = time_14(idx_14);

data_1_maxs = [max_data_11; max_data_12; max_data_13; max_data_14];
data_1_max_times = [max_time_11;max_time_12;max_time_13; max_time_14];
[max_data_1,max_data_1_idx] = max(data_1_maxs);
max_time_1 = data_1_max_times(max_data_1_idx);


% Data 2
[max_data_21,idx_21] = max(demo_21);
[max_data_22,idx_22] = max(demo_22);
[max_data_23,idx_23] = max(demo_23);
[max_data_24,idx_24] = max(demo_24);
[max_data_25,idx_25] = max(demo_25);

max_time_21 = time_21(idx_21);
max_time_22 = time_22(idx_22);
max_time_23 = time_23(idx_23);
max_time_24 = time_24(idx_24);
max_time_25 = time_25(idx_25);

```

```
data_2_maxs = [max_data_21; max_data_22; max_data_23; max_data_14; max_data_25
    ];
data_2_max_times = [max_time_21; max_time_22; max_time_23; max_time_24;
    max_time_25];
[max_data_2,max_data_2_idx] = max(data_2_maxs);
max_time_2 = data_2_max_times(max_data_2_idx);

% Data 3
[max_data_31,idx_31] = max(demo_31);
[max_data_32,idx_32] = max(demo_32);
[max_data_33,idx_33] = max(demo_33);
[max_data_34,idx_34] = max(demo_34);
[max_data_35,idx_35] = max(demo_35);
[max_data_36,idx_36] = max(demo_36);
[max_data_37,idx_37] = max(demo_37);
[max_data_38,idx_38] = max(demo_38);
[max_data_39,idx_39] = max(demo_39);

max_time_31 = time_31(idx_31);
max_time_32 = time_32(idx_32);
max_time_33 = time_33(idx_33);
max_time_34 = time_34(idx_34);
max_time_35 = time_35(idx_35);
max_time_36 = time_36(idx_36);
max_time_37 = time_37(idx_37);
max_time_38 = time_38(idx_38);
max_time_39 = time_39(idx_39);

data_3_maxs = [max_data_31; max_data_32; max_data_33; max_data_34; max_data_35
    ; max_data_36; max_data_37; max_data_38; max_data_39];
data_3_max_times = [max_time_31; max_time_32; max_time_33; max_time_34;
    max_time_35; max_time_36; max_time_37; max_time_38; max_time_39];
[max_data_3,max_data_3_idx] = max(data_3_maxs);
max_time_3 = data_3_max_times(max_data_3_idx);

%% Plot the Force Curves Data
% Using summed load to plot
% Plot the force for Data 1

figure;
hold on
plot(time_11,demo_11);
plot(time_12,demo_12);
plot(time_13,demo_13);
plot(time_14,demo_14);
%Zero Pt Line to Indicate Weight of Test Stand
    % Where the group zeroed the data varies with each test and so there is
    % some variation expected between the three groups. The zero line was
    % chosen based on the given graph from the lab document.
yline(1);
% Max Thrust for Entire Data 1
plot(max_time_1,max_data_1,'o');
hold off
legend('Test 1','Test 2','Test 3','Test 4','Zero Pt','Max Thrust Point');
```

```
title('Total Force Curves for Data 1');
xlabel('Time [s]');
ylabel('Force [N]');
xlim([0 0.4])

% Plot the force for Data 2
figure;
hold on
plot(time_21,demo_21);
plot(time_22,demo_22);
plot(time_23,demo_23);
plot(time_24,demo_24);
plot(time_25,demo_25);
yline(2);
% Max Thrust for Entire Data 2
plot(max_time_2,max_data_2,'o');
hold off
legend('Test 1','Test 2','Test 3','Test 4','Test 5','Zero Pt','Max Thrust
    Point');
title('Total Force Curves for Data 2');
xlabel('Time [s]');
ylabel('Force [N]');
xlim([0 0.5])

% Plot the force for Data 3
figure;
hold on
plot(time_31,demo_31);
plot(time_32,demo_32);
plot(time_33,demo_33);
plot(time_34,demo_34);
plot(time_35,demo_35);
plot(time_36,demo_36);
plot(time_37,demo_37);
plot(time_38,demo_38);
plot(time_39,demo_39);
yline(5);

% Max Thrust for Entire Data 3
plot(max_time_3,max_data_3,'o');
hold off
legend('Test 1','Test 2','Test 3','Test 4','Test 5','Test 6','Test 7','Test 8'
    ,'Test 9','Zero Pt','Max Thrust Point');
title('Total Force Curves for Data 3');
xlabel('Time [s]');
ylabel('Force [N]');
xlim([0 0.5])

% Plot Representative Force Curve (trial 1.4)
figure;
hold on
plot(time_14,demo_14);
yline(1);
hold off
```

```
318 title('Representative Force Curve - Trial 1.4');
319 xlabel('Time [s]');
320 ylabel('Force [N]');
321 xlim([0 0.4])
322
323
324 %% Calculate I for each dataset and then combine into array
325 I_demo_11 = trapz(time_11,demo_11);
326 I_demo_12 = trapz(time_12,demo_12);
327 I_demo_13 = trapz(time_13,demo_13);
328 I_demo_14 = trapz(time_14,demo_14);
329
330 % Combine into array for Data 1
331 I_data_1 = [I_demo_11 I_demo_12 I_demo_13 I_demo_14];
332
333 I_demo_21 = trapz(time_21,demo_21);
334 I_demo_22 = trapz(time_22,demo_22);
335 I_demo_23 = trapz(time_23,demo_23);
336 I_demo_24 = trapz(time_24,demo_24);
337 I_demo_25 = trapz(time_25,demo_25);
338
339 % Combine into array for Data 2
340 I_data_2 = [I_demo_21 I_demo_22 I_demo_23 I_demo_24 I_demo_25];
341
342 I_demo_31 = trapz(time_31,demo_31);
343 I_demo_32 = trapz(time_32,demo_32);
344 I_demo_33 = trapz(time_33,demo_33);
345 I_demo_34 = trapz(time_34,demo_34);
346 I_demo_35 = trapz(time_35,demo_35);
347 I_demo_36 = trapz(time_36,demo_36);
348 I_demo_37 = trapz(time_37,demo_37);
349 I_demo_38 = trapz(time_38,demo_38);
350 I_demo_39 = trapz(time_39,demo_39);
351
352 % Combine into array for Data 3
353 I_data_3 = [I_demo_31 I_demo_32 I_demo_33 I_demo_34 I_demo_35 I_demo_36
        I_demo_37 I_demo_38 I_demo_39];
354
355 %% Calculate I_sp for each Data
356 Isp_data_1 = I_data_1 / (water_mass*g0);
357 Isp_data_2 = I_data_2 / (water_mass*g0);
358 Isp_data_3 = I_data_3 / (water_mass*g0);
359
360 fprintf('The calculations for Isp produce the following: \n\n');
361 fprintf('For data 1: %f s \n', Isp_data_1);
362 fprintf('\n');
363 fprintf('For data 2: %f s \n', Isp_data_2);
364 fprintf('\n');
365 fprintf('For data 3: %f s \n', Isp_data_3);
366
367
368 %% Statistics
369 % Average and standard deviation of peak thrust and thrust times
370 thrustmaximums = [data_1_maxs;data_2_maxs;data_3_maxs];
```

```
mean_of_max = mean(thrustmaximums);
std_of_max = std(thrustmaximums);
thrusttimes = [time_11(end),time_12(end),time_13(end),time_14(end),...
    time_21(end),time_22(end),time_23(end),time_24(end),time_25(end),...
    time_31(end),time_32(end),time_33(end),time_34(end),time_35(end),...
    time_36(end),time_37(end),time_38(end),time_39(end)];
mean_of_times = mean(thrusttimes);
std_of_times = std(thrusttimes);

fprintf('\nAverage peak thrust: %.2f    %.2f N\n',mean_of_max,std_of_max);
fprintf('Average thrust time: %.2f    %.2f s\n',mean_of_times,std_of_times);

%% Error Calculations
% Isp SEM analysis for varying N
Isps1 = [Isp_data_1,Isp_data_2,Isp_data_3]; %(OUTLIERS NOT REMOVED)
Isps = [Isp_data_1,Isp_data_2(1:4),Isp_data_3(1:7),Isp_data_3(9)]; % OUTLIERS
    REMOVED
stds_varied = zeros(1,length(Isps));
SEMS_varied = zeros(1,length(Isps));
N = linspace(1,length(Isps),length(Isps));
for i=1:length(Isps)
    stds_varied(i) = std(Isps(1:i));
    SEMS_varied(i) = stds_varied(i)/sqrt(i);
end

figure;
hold on
plot(N,SEMS_varied);
xlabel('Number of data sets included');
ylabel('Standard Error of the Mean');
title('SEM of Isp versus Number of Data Sets');

% Isp SEM for Full Data Sets
std_data_1 = std(Isp_data_1);
SEM_data_1 = std_data_1/sqrt(length(Isp_data_1));

std_data_2 = std(Isp_data_2);
SEM_data_2 = std_data_2/sqrt(length(Isp_data_2));

std_data_3 = std(Isp_data_3);
SEM_data_3 = std_data_3/sqrt(length(Isp_data_3));

std_full = std(Isps);
SEM_full = std_full/sqrt(length(Isps));

%plot(length(Isps),SEM_full,'ro');
hold off

fprintf('\nSEM for data 1 Isp (4 trials): %f s\n',SEM_data_1);
fprintf('SEM for data 2 Isp (5 trials): %f s\n',SEM_data_2);
fprintf('SEM for data 3 Isp (9 trials): %f s\n',SEM_data_3);
fprintf('SEM for full data Isp (18 trials): %f s\n',SEM_full);

%% Confidence Intervals
```

```
424 % Define z values for [95%,97.5%,99%] confidence intervals
425 zvals = [1.96,2.24,2.58];
426 mean_full = mean(Isps); % final estimate for Isp (sample mean of full dataset)
427 fprintf('\nIsp estimate (full sample mean): %f s\n',mean_full);
428 CI95 = zvals(1)*SEM_full;
429 CI975 = zvals(2)*SEM_full;
430 CI99 = zvals(3)*SEM_full;
431
432 fprintf('\nIsp Estimate Confidence Intervals:\n');
433 fprintf('95%%: %.2f    %.2f s\n',mean_full,CI95);
434 fprintf('97.5%%: %.2f    %.2f s\n',mean_full,CI975);
435 fprintf('99%%: %.2f    %.2f s\n',mean_full,CI99);
436
437 % How many tests needed for each CI to have a precision of 0.1s? 0.01s?
438 SEM_req_95_01 = 0.1/zvals(1);
439 SEM_req_975_01 = 0.1/zvals(2);
440 SEM_req_99_01 = 0.1/zvals(3);
441
442 N_req_95_01 = (std_full/SEM_req_95_01)^2;
443 N_req_975_01 = (std_full/SEM_req_975_01)^2;
444 N_req_99_01 = (std_full/SEM_req_99_01)^2;
445
446 SEM_req_95_001 = 0.01/zvals(1);
447 SEM_req_975_001 = 0.01/zvals(2);
448 SEM_req_99_001 = 0.01/zvals(3);
449
450 N_req_95_001 = (std_full/SEM_req_95_001)^2;
451 N_req_975_001 = (std_full/SEM_req_975_001)^2;
452 N_req_99_001 = (std_full/SEM_req_99_001)^2;
453
454 fprintf('\nNumber of tests required for 0.1s precision:\n');
455 fprintf('95%%: %.0f \n',N_req_95_01);
456 fprintf('97.5%%: %.0f \n',N_req_975_01);
457 fprintf('99%%: %.0f \n',N_req_99_01);
458
459 fprintf('\nNumber of tests required for 0.01s precision:\n');
460 fprintf('95%%: %.0f \n',N_req_95_001);
461 fprintf('97.5%%: %.0f \n',N_req_975_001);
462 fprintf('99%%: %.0f \n',N_req_99_001);
463
464 %% Final Results in Table
465 N1 = linspace(1,length(Isps1),length(Isps1));
466 T = table(N1',Isps1',thrustmaximums,thrusttimes','VariableNames',...
467     {'Trial','Isp [s]','Peak Thrust [N]','Thrust Time [s]'});
468 display(T);
```