Table of Contents


```
close all; clear; clc;
```

Changeable parameters for different cases

```
%Change these values to analyze different data

% Test numbers (case 2- testrun11.mat - testrun20.mat)
first_test=11;
last_test=20;
cal_case = 2; %Static Test Stand Calibration Case #.xlsx, where # is cal_case
```

Inputting Calibration data

```
%Read in calibration data
calibration_data = readmatrix(sprintf('Static Test Stand Calibration Case
 %g.xlsx',cal_case));

%Storing calibration data in arrays
weight = calibration_data(:,1);
f_0_offset = calibration_data(:,2);
f_1_offset = calibration_data(:,3);
f_0_mv = calibration_data(:,4);
f_1_mv = calibration_data(:,5);
```

Line of Best Fit, 95% conifdence interval

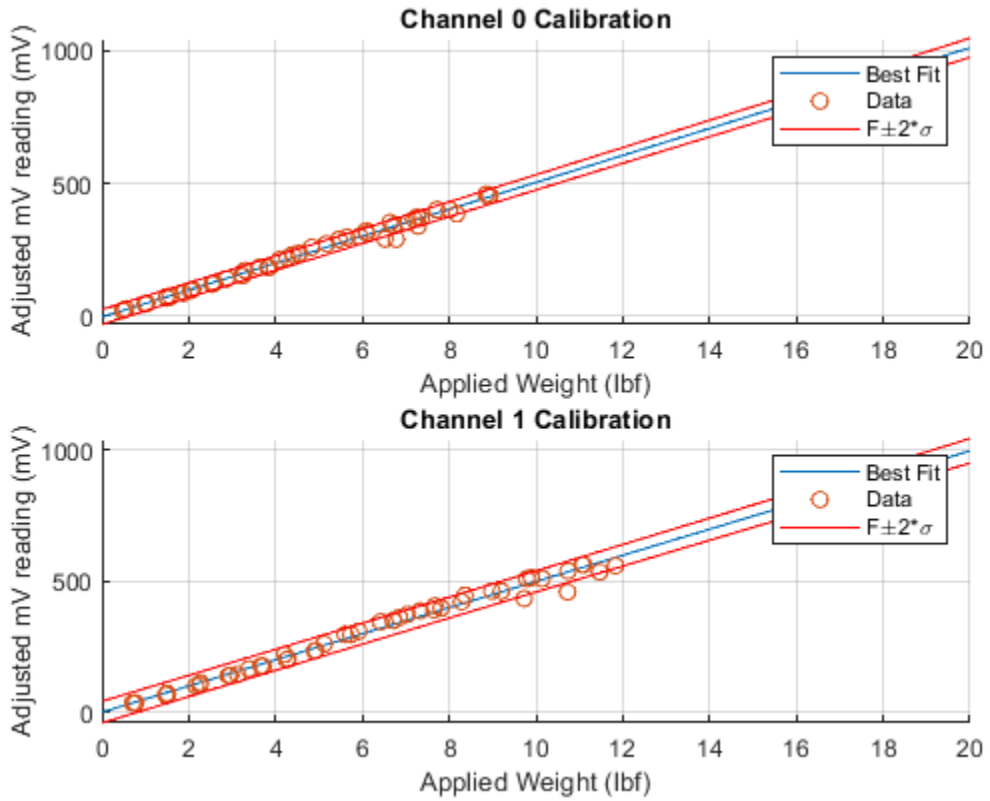
```
%Subtracting channel offset
f_0_mv = f_0_mv - f_0_offset;
f_1_mv = f_1_mv - f_1_offset;

%Since the channels are uneven, the total force is the sum
ftot_mv = f_0_mv+f_1_mv;

%Weighted averages of mV reading, converting to adjusted weight
f_adjusted_0 = weight.*(f_0_mv./(f_0_mv+f_1_mv));
f_adjusted_1 = weight.*(f_1_mv./(f_0_mv+f_1_mv));
```

```
f_adjusted_tot = f_adjusted_0+f_adjusted_1;

%calculating line of best fit for the data
[p0,b0] = polyfit(f_adjusted_0,f_0_mv,1);
[p1,b1] = polyfit(f_adjusted_1,f_1_mv,1);

%Sum of data line of best fit
[ptot,btot] = polyfit(f_adjusted_tot,ftot_mv,1);

% Arbitrary array for domain of calibration plot for extrapolation
x = 0:.1:60;
x1 = 0:.1:20;

%Finding sigma for the line of best fit
[y0,delta0] = polyval(p0,x1,b0);
[y1,delta1] = polyval(p1,x1,b1);
[ytot,deltatot] = polyval(ptot,x,btot);

%Renaming constants for simplicity
a0 = p0(1);
b00 = p0(2);
a1 = p1(1);
b11 = p1(2);
a0tot = ptot(1);
a1tot = ptot(2);

%Function handles for either channel for calibration
calibrate_1 = @(fv) (1/a1)*(fv-b11);
calibrate_0 = @(fv) (1/a0)*(fv-b00);

calibrate_tot = @(fv) (1/a0tot)*(fv-a1tot);
```

Plotting LBF for seperate channels

```
figure (1);
subplot(2,1,1);
hold on;
plot(x1, y0);
scatter(f_adjusted_0,f_0_mv);
grid on;
xlabel('Applied Weight (lbf)');
ylabel('Adjusted mV reading (mV)');
plot(x1,y0+2*delta0,'r');
plot(x1,y0-2*delta0,'r');
legend('Best Fit','Data','F\pm2*\sigma');
title('Channel 0 Calibration');
hold off;

subplot(2,1,2);
hold on;
plot(x1, y1);
scatter(f_adjusted_1,f_1_mv);
```

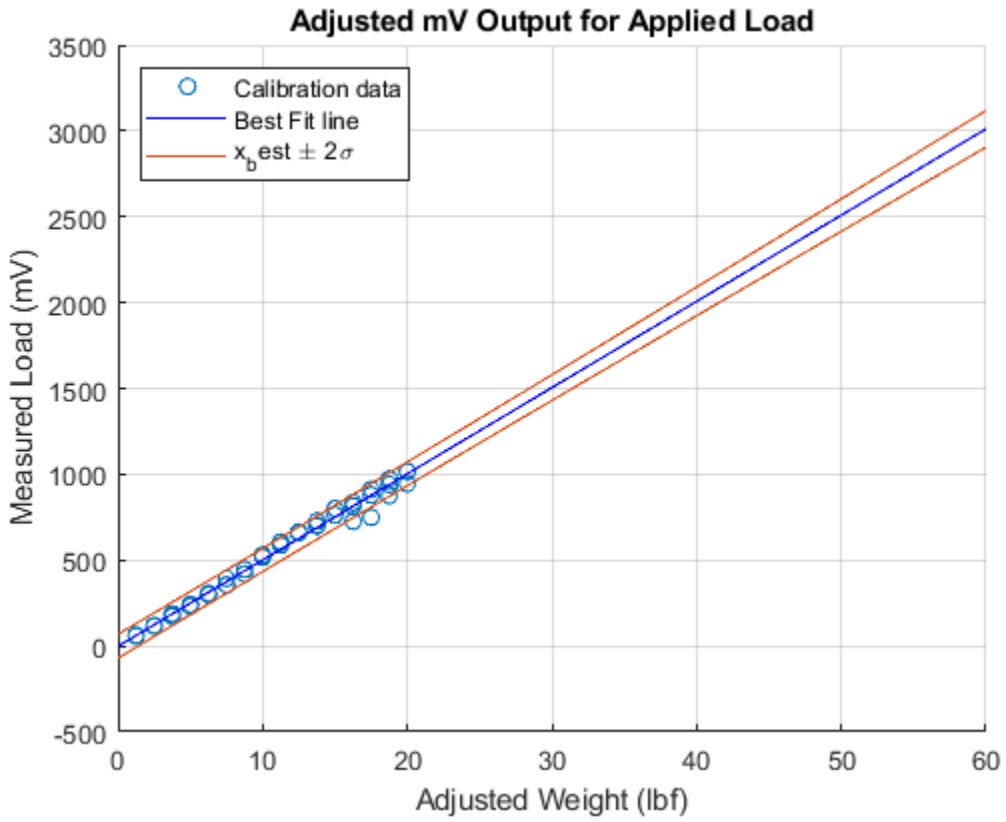
```
grid on;
xlabel('Applied Weight (lbf)');
ylabel('Adjusted mV reading (mV)');
plot(x1,y1+2*delta1,'r');
plot(x1,y1-2*delta1,'r');
legend('Best Fit','Data','F\pm2*\sigma');
title('Channel 1 Calibration');
hold off;

saveas(gcf,'seperate_channels_lbf.png');
```



Plotting calibration data with LBF, Error

```
figure(2);
hold on;
scatter(f_adjusted_tot,ftot_mv); %Caliration data scatter plot
plot(x,ytot,'color','blue'); %Line of best fit
plot(x,ytot+2*deltatot,'color','#D95319'); %+/- 2 sigma
plot(x,ytot-2*deltatot,'color',"#D95319");
grid on;
xlabel('Adjusted Weight (lbf)');
ylabel('Measured Load (mV)');
title('Adjusted mV Output for Applied Load');
legend('Calibration data','Best Fit line','x_best \pm
 2\sigma','location','Northwest');
```

```
saveas(gcf,'combined_lbf.png');
```



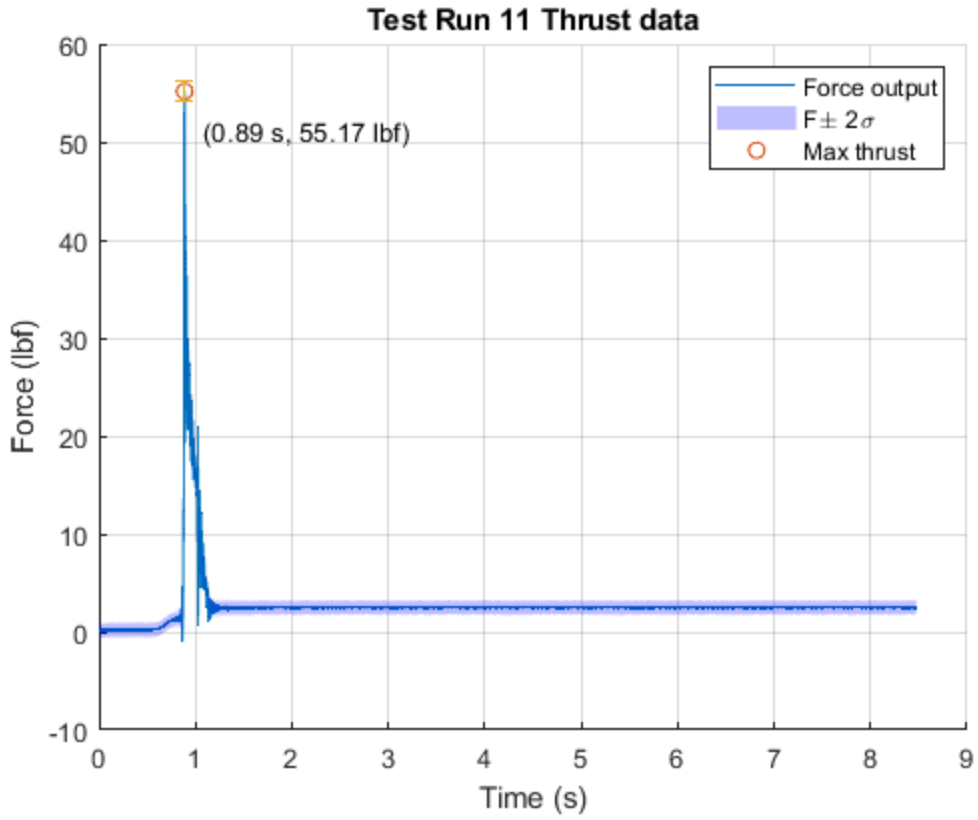
Converting mV inputs to Force for thrust analysis

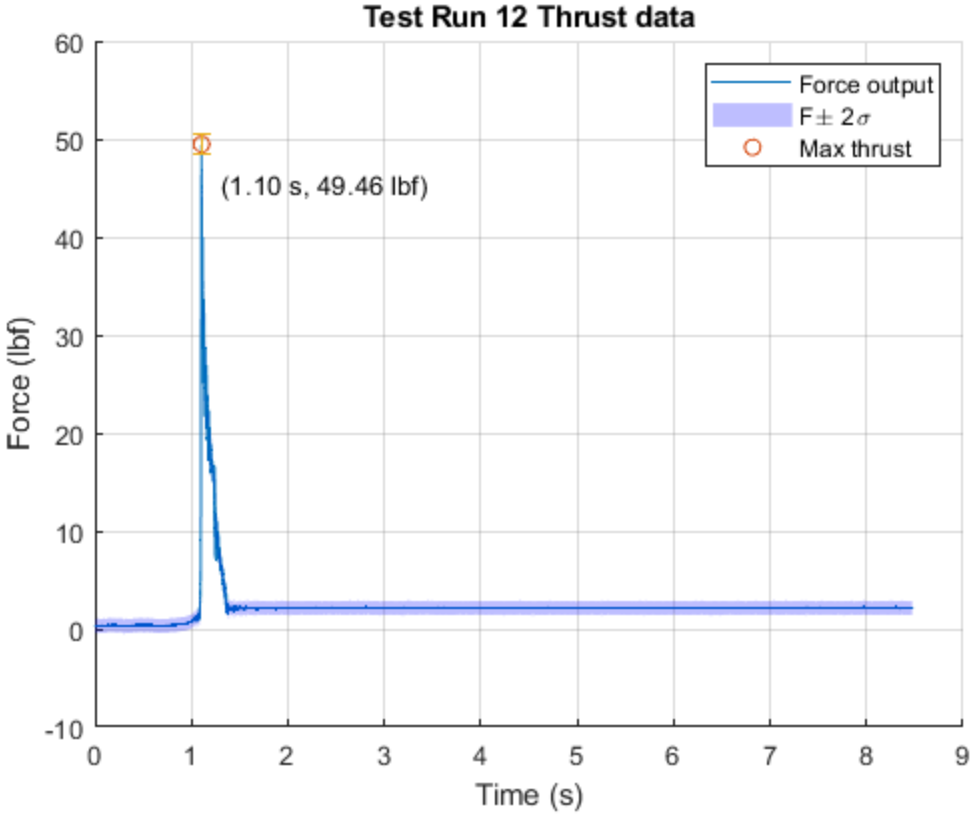
```
% Preallocation of cells and structs before for loop
dimension = last_test;
s = strings(dimension,1);
data = struct('time',cell(1,dimension), 'mV',cell(1,dimension));
mv= cell(1,dimension);
time = cell(1,dimension);
f_c_1 = cell(1,dimension);
f_c_2 = cell(1,dimension);
f_c_tot = cell(1,dimension);
delta = cell(1,dimension);
y = cell(1,dimension);
delta_cal = cell(1,dimension);
max_force = zeros(1,dimension);
max_when = zeros(1,dimension);
max_error = zeros(1,dimension);

%All conversions and calculations done in for loop
for i = first_test:last_test

    %name of file to string (testruni), loading data splitting structs
```

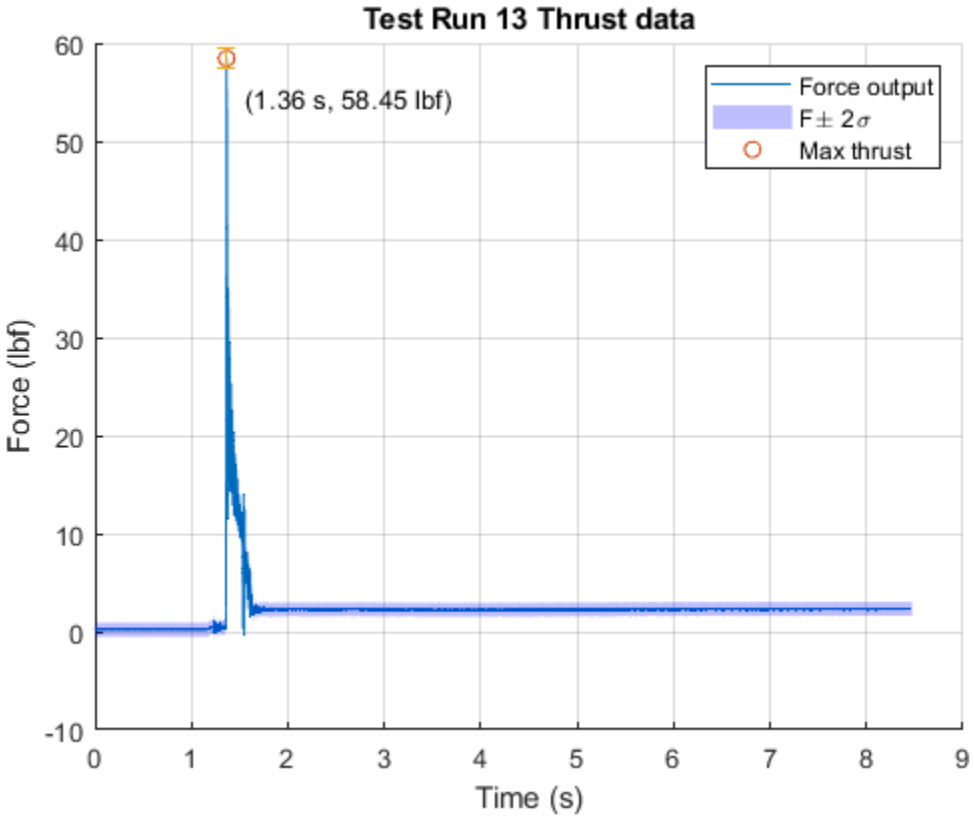
```
    s(i) = "testrun"+i+".mat";
    data(i) = load(s(i));

    %Storing struct data in cells
    mv{i} = data(i).mV;
    time{i}=data(i).time;

    %converting the mv readings to force using calibration line above
    for j = 1:length(mv{1,i})
        f_c_1{i}(j) = calibrate_0(mv{1,i}(j,1));
    end

    for j = 1:length(mv{1,i})
        f_c_2{i}(j) = calibrate_1(mv{1,i}(j,2));
    end

     % f_tot = f_0+f_1
     f_c_tot{i} = f_c_1{i}+f_c_2{i};

     %Calculating delta for the given mV ranges using polyval
    [y{i},delta{i}] = polyval(ptot,f_c_tot{i},btot);
    delta_cal{i} = calibrate_tot(delta{i});

    %Plotting the results
    figure(i);
    hold on;
    plot(time{1,i},f_c_tot{i}); %Thrust data
    title(sprintf('Test Run %g Thrust data',i));  %Changing title for each
iteration
    ylabel('Force (lbf)');
    xlabel('Time (s)');

    %Creating a cloud for +/- 2 sigma interval using patch command
    patch([time{i} flip(time{i})],[(f_c_tot{i}-delta_cal{i})
flip(f_c_tot{i}+delta_cal{i})],'b', 'FaceAlpha',0.25, 'EdgeColor','none');
    grid on;

    %Calculating the max force and associated error
    [max_force(i),max_when(i)] = max(f_c_tot{i});
    max_error(i) = delta_cal{i}(max_when(i));

    %converting index to time
    timestep = time{i}(3)-time{i}(2);
    max_when(i) = max_when(i)*timestep;

    %Plotting max value and text label
    scatter(max_when(i),max_force(i));
    text(max_when(i)+.2,max_force(i)-4,sprintf('(%2.2f s, %2.2f
lbf)',max_when(i),max_force(i)));

    %Putting an error bar at the peak thrust point
    errorbar(max_when(i),max_force(i),max_error(i));

    legend('Force output','F\pm 2\sigma','Max thrust')
```

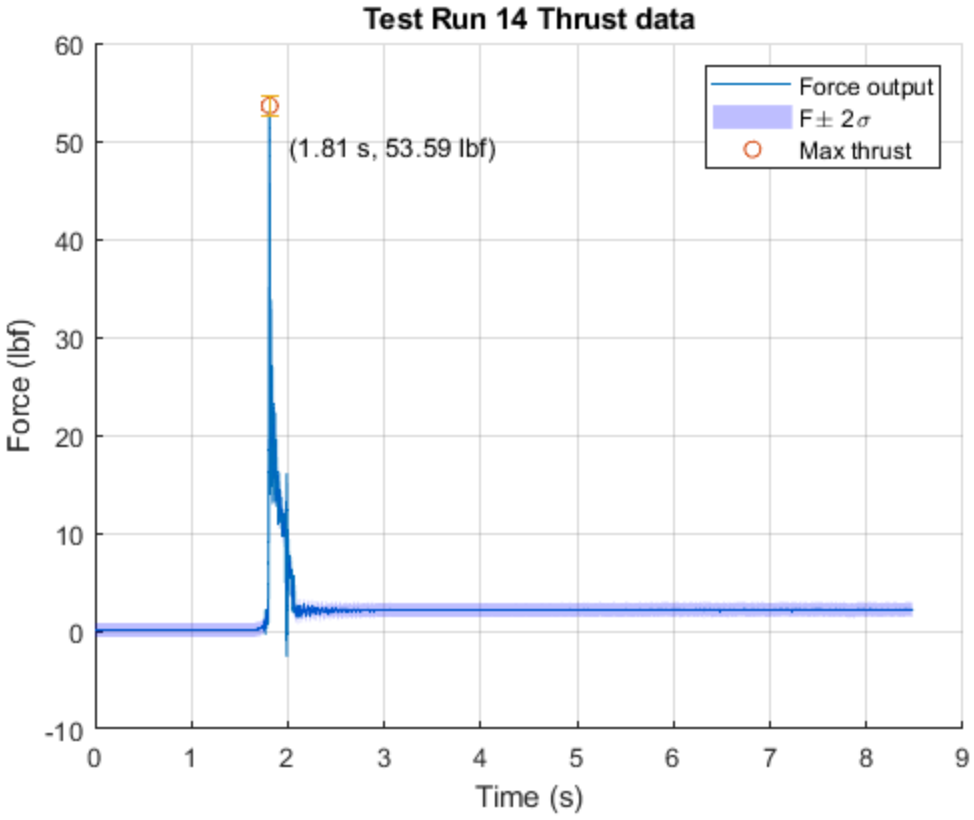
```
    hold off;

    saveas(gcf,sprintf('testrun%ggraph.png',i));

    %printing thrust and associated error
    fprintf('Max thrust for case %g is %2.2f \x00B1 %2.2f lbf
 \n',i,max_force(i),2*max_error(i));



end
```

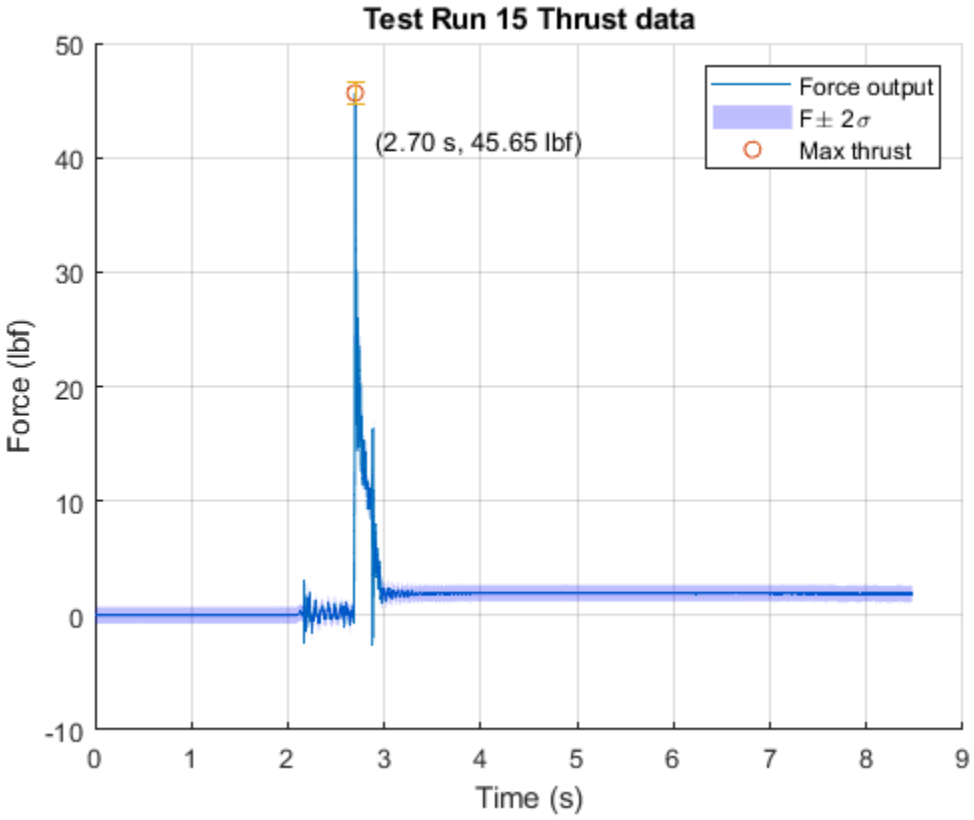
```
Max thrust for case 11 is 55.17 ± 2.04 lbf
Max thrust for case 12 is 49.46 ± 1.91 lbf
Max thrust for case 13 is 58.45 ± 2.13 lbf
Max thrust for case 14 is 53.59 ± 2.01 lbf
Max thrust for case 15 is 45.65 ± 1.82 lbf
Max thrust for case 16 is 51.76 ± 1.96 lbf
Max thrust for case 17 is 47.58 ± 1.86 lbf
Max thrust for case 18 is 47.38 ± 1.86 lbf
Max thrust for case 19 is 49.06 ± 1.90 lbf
Max thrust for case 20 is 50.24 ± 1.93 lbf
```

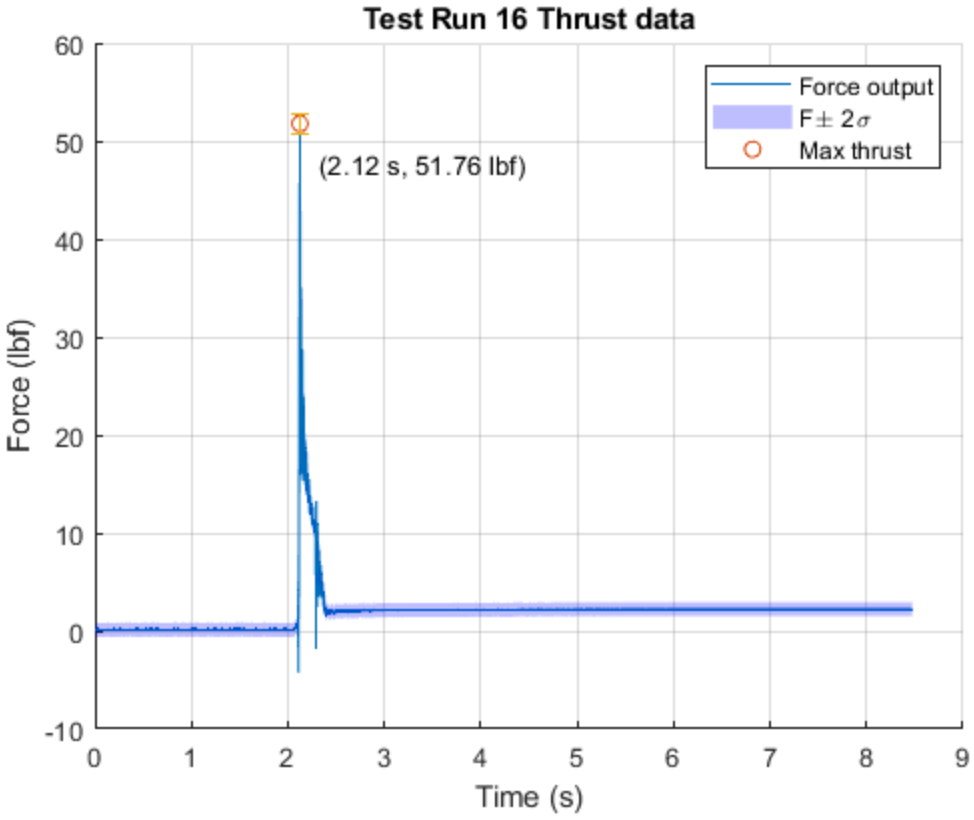


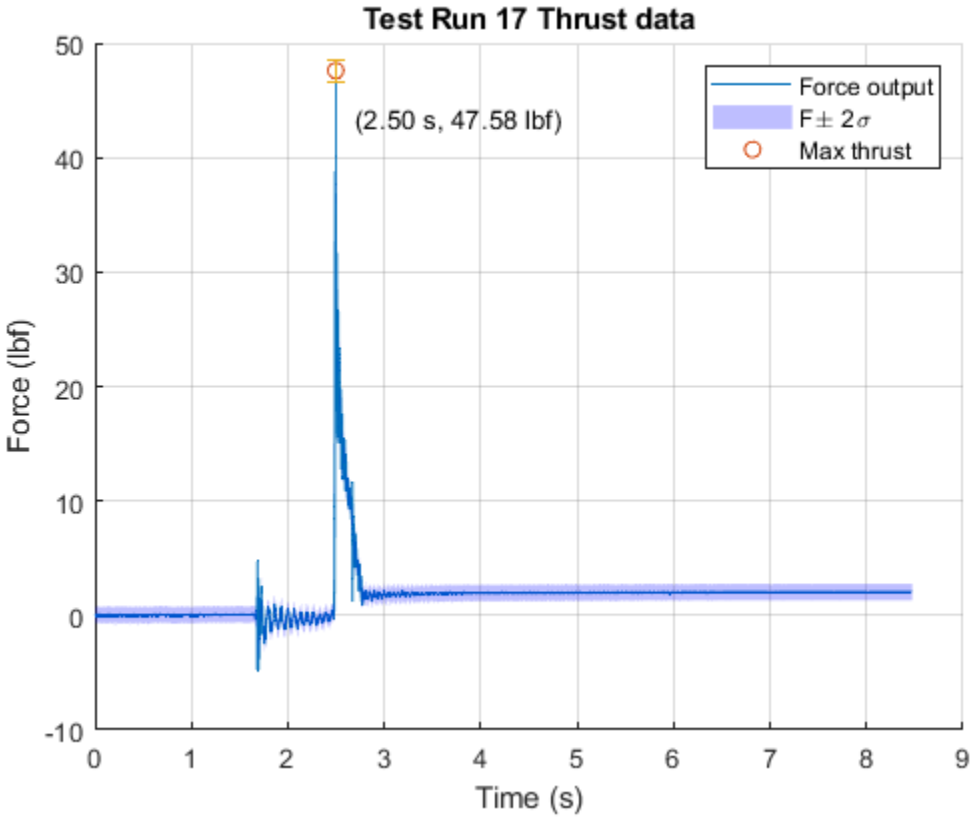

Test Run 12 Thrust data
Force output
F± 2σ
Max thrust
(1.10 s, 49.46 lbf)
Force (lbf)
Time (s)
60
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9

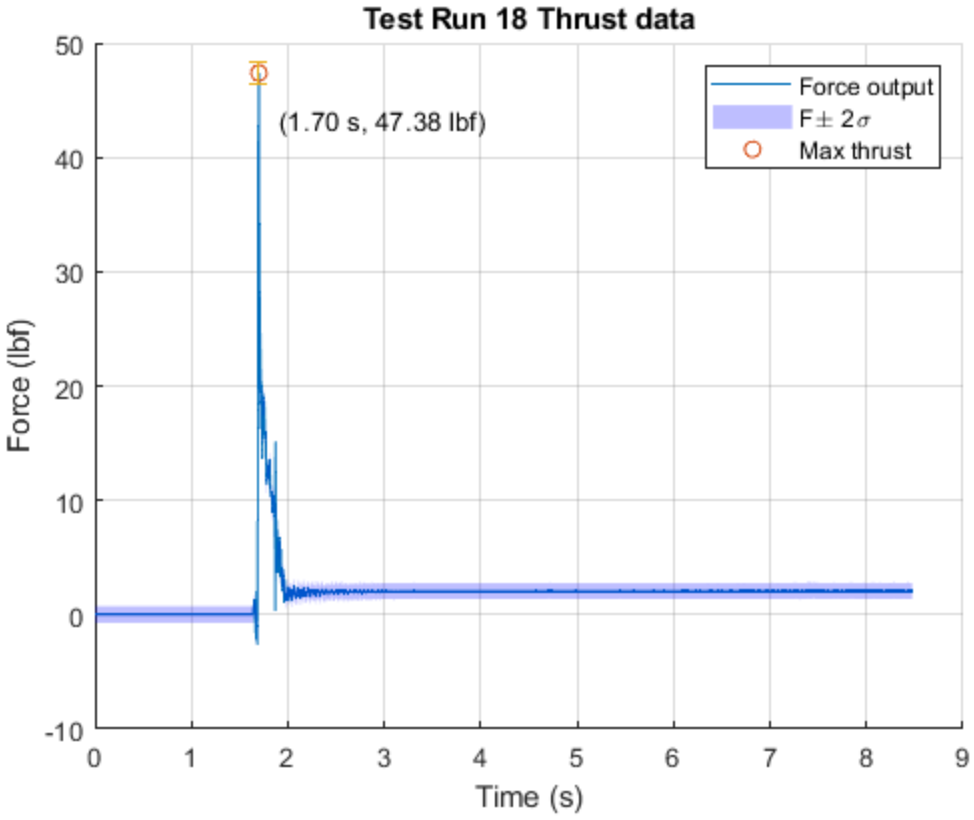

Test Run 13 Thrust data
Force output
F± 2σ
Max thrust
(1.36 s, 58.45 lbf)
Force (lbf)
Time (s)
60
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9

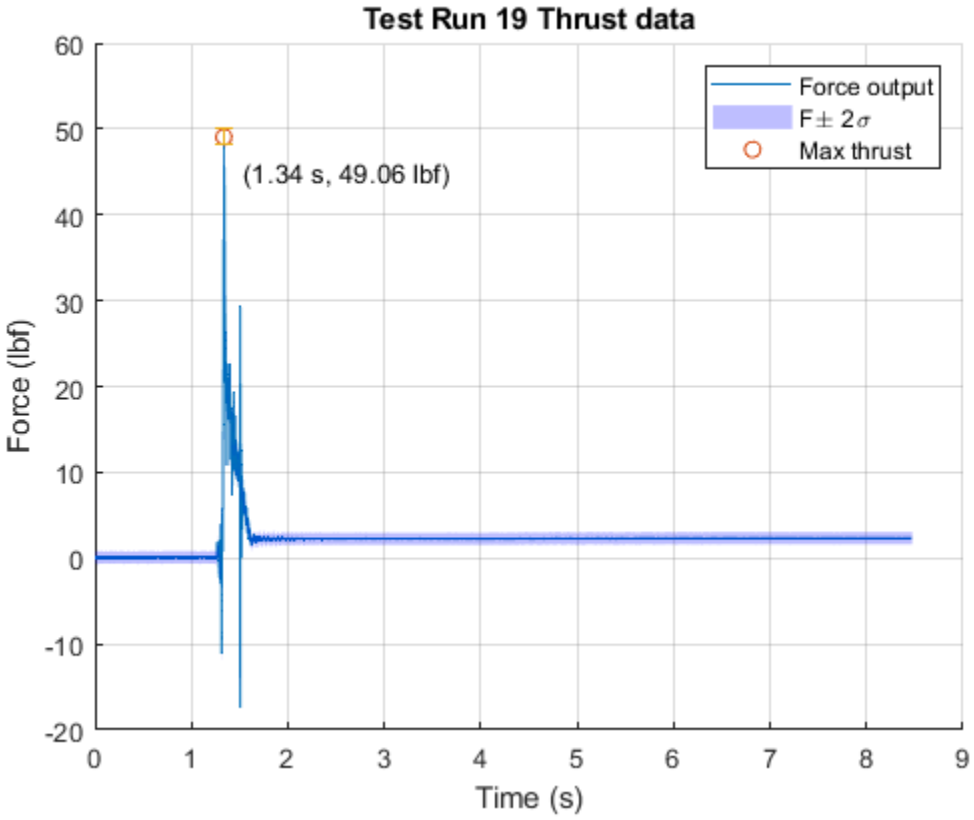

Test Run 14 Thrust data
Force output
F± 2σ
Max thrust
(1.81 s, 53.59 lbf)
Force (lbf)
Time (s)
60
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9

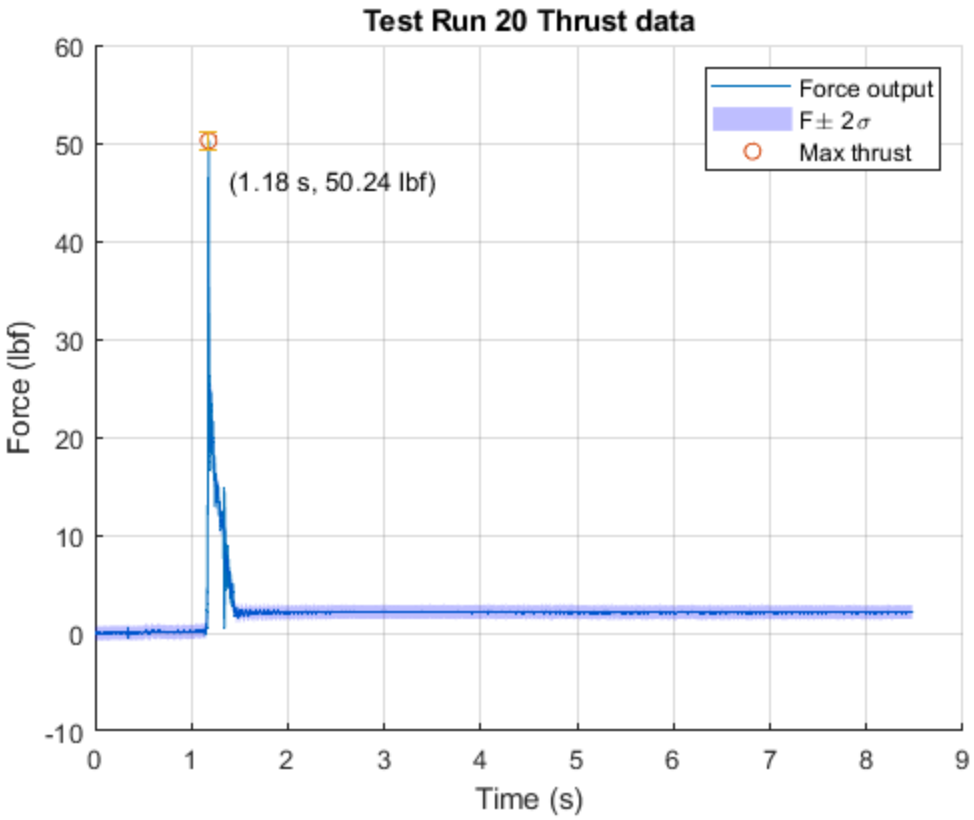

Test Run 15 Thrust data
Force output
F± 2σ
Max thrust
(2.70 s, 45.65 lbf)
Force (lbf)
Time (s)
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9


Test Run 16 Thrust data
Force output
F± 2σ
Max thrust
(2.12 s, 51.76 lbf)
Force (lbf)
Time (s)
60
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9


Test Run 17 Thrust data
Force output
F± 2σ
Max thrust
(2.50 s, 47.58 lbf)
Force (lbf)
Time (s)
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9


Test Run 18 Thrust data
(1.70 s, 47.38 lbf)
Force output
F± 2σ
Max thrust
Force (lbf)
Time (s)
50
40
30
20
10
0
-10
0
1
2
3
4
5
6
7
8
9


Test Run 19 Thrust data
(1.34 s, 49.06 lbf)
Force output
F± 2σ
Max thrust
Force (lbf)
Time (s)
60
50
40
30
20
10
0
-10
-20
0
1
2
3
4
5
6
7
8
9



Calculating Weighted average of all the tests (max only)

```
omega = 1./(max_error(first_test:last_test).^2);

Weighted_average = sum(omega.*max_force(first_test:last_test))/sum(omega);

average_uncertainty = 1/sqrt(sum(omega));

fprintf('Weighted average of all tests: %2.2f \x00B1 %2.2f lbf
 \n',Weighted_average,average_uncertainty);
```

```
Weighted average of all tests: 50.51 ± 0.31 lbf
```

Published with MATLAB® R2022a