Table of Contents

Project 2 Analysis 1
Usefull Constants 1
Given Data 1
Import Collected Data 2
Plots 2
Solve for Each RPM Observation 2
Plot Stagnation Temp vs Spool Speed 3
Plot Stagnation Pressure vs Spool Speed 4
Mach Number vs Spool Speed 5
Velocity vs Spool Speed 6
Mass Flow Rate vs Spool Speed 7
Fuel Flow Rate vs Spool Speed 8
Air/Fuel Ratio vs. Spool Speed 9
Calculate Thrust versus Experimental 10

Project 2 Analysis

This script is the main controller for the SR30 analysis in Project 2 of ME140. It should import data, and call the component functions for each of the individual parts of the project.

Authors: Jean-Christophe Perrin, Beck Goodloe, Richard Randall, Jason Trinidad

Created: 2018-04-11 Edited: 2018-04-11

```
clear all;
clc;
```

Usefull Constants

These are mostly conversion factors so that we can convert from imperial units collected into metric.

```
const.insqToMsq = 0.00064516; % [m^2/in^2]
const.lbfToN = 4.44822; % [N/lbf]
const.R = 287; % [kJ/kg/k] Gas constant of air
const.KCdiff = 273; % [deg]
const.kPaToPa = 1e3; % [Pa/kPa]
const.kJkmol2Jkg = 1e3/28.97; %[J/kg * kmol/kJ]
const.airCoefs = const.kJkmol2Jkg .* ...
    fliplr([28.11, 0.1967e-2, 0.4802e-5,-1.966e-9]); % [J/kg/k] cp air
```

Given Data

The following values were supplied to us in the original project specifications and should not be changed.

```
given.pitotEffectiveArea = 6.4*const.insqToMsq; % [m^2]
given.jetAHeating = 42.8e6; % [J/kg/K]
```

```
given.A = [27.3, 6.4, 9.0, 7.2, 4.7, 3.87]'.*const.insqToMsq; % [m^2]
```

Import Collected Data

We import the collected data into an table for use throughout our function. This table should never be changed, compromising the validity of our collected data. Instead, all deried values should be copied out the table into a seperate vector or table.

```
fname = 'data.txt';
collectedData = readtable(fname);
collectedData.Properties.VariableUnits =
 {'', 'C', 'C', 'C', 'C', 'C', ...
    'C', 'kPa', 'kPa', 'kPa', 'kPa', 'kPa', 'kg/s', 'lbs'};

nObservations = height(collectedData); % Number of observations
% disp(collectedData);
```

```
Warning: Variable names were modified to make them valid MATLAB
 identifiers. The
original names are saved in the VariableDescriptions property.
```

We also measured the following values seperately.

```
T1 = 23; % [C] room temp
P1 = 101.4e3; % [Pa] atmospheric pressue
```

Plots

Use your performance data and area measurements (listed in Appendix) to construct a series of plots showing how the following quantities vary with spool speed:

- station stagnation temperature (K),
- station stagnation pressure (kPa, absolute),
- station Mach number
- station velocity (m/s).

Make sure to include Station 1 in all of your plots.

```
krpm = collectedData.RPM/1000;
```

Solve for Each RPM Observation

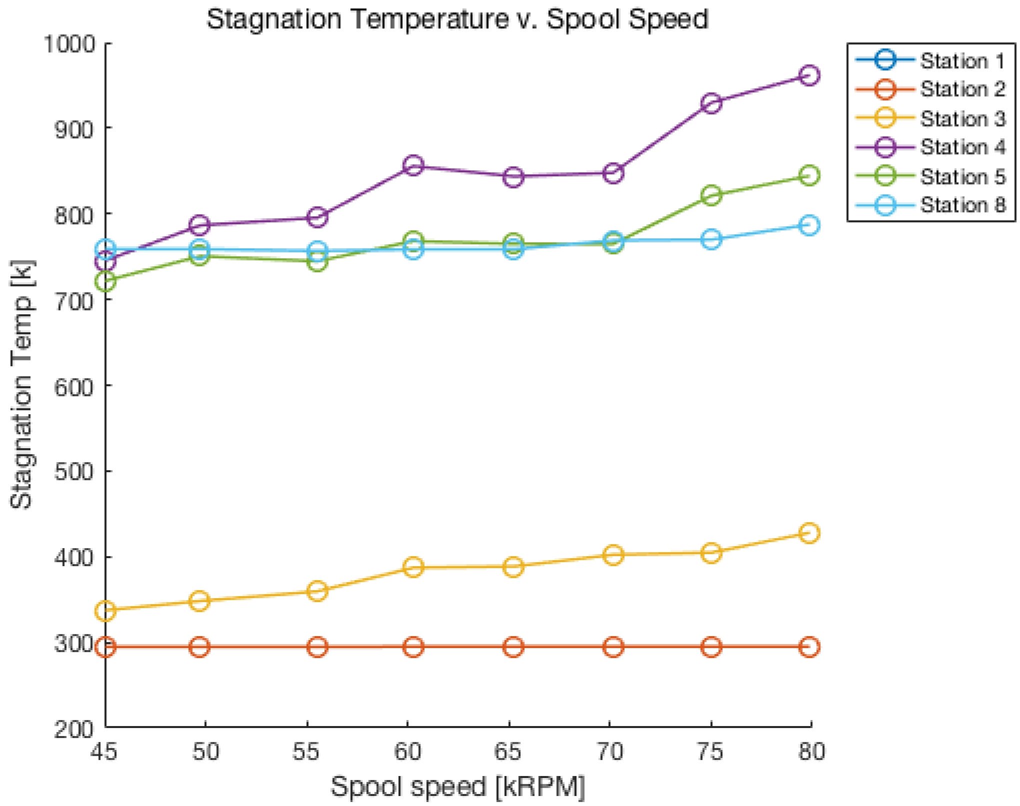
Loop over every observation taken. For each set of Tm or Pm (the measured values of Temp and Pressure) call Richard's Analysis to calculated the values of interest (T0, P0, M, V). These are stored in a 2D array. Each row of the array corresponds to an observation (i.e. one RPM). Each column is a location of interest.

```
nLocations = 6; % locations of interest: 1-5, 8
T0 = NaN(nObservations, nLocations);
```

```
P0 = NaN(nObservations, nLocations);
Ma = NaN(nObservations, nLocations);
V = NaN(nObservations, nLocations);
mdotAir = NaN(nObservations, 1);

for iObservation = 1:nObservations
    Tm = collectedData{iObservation, 2:6};
    Pm = collectedData{iObservation, 8:12};
    mdot = collectedData{iObservation, 13};
    thrust = collectedData{iObservation, 14};
    [calculatedValues, thisMdotAir] = solveEachLocation(Tm, Pm, mdot,
 thrust);
    mdotAir(iObservation) = thisMdotAir;
    calculatedValues([6, 7], :) = []; % deletes rows 6/7 (no
 instruments)
    T0(iObservation, :) = cell2mat(calculatedValues.T0)';
    P0(iObservation, :) = cell2mat(calculatedValues.P0)';
    Ma(iObservation, :) = cell2mat(calculatedValues.M)';
    V(iObservation, :) = cell2mat(calculatedValues.V)';
end
```

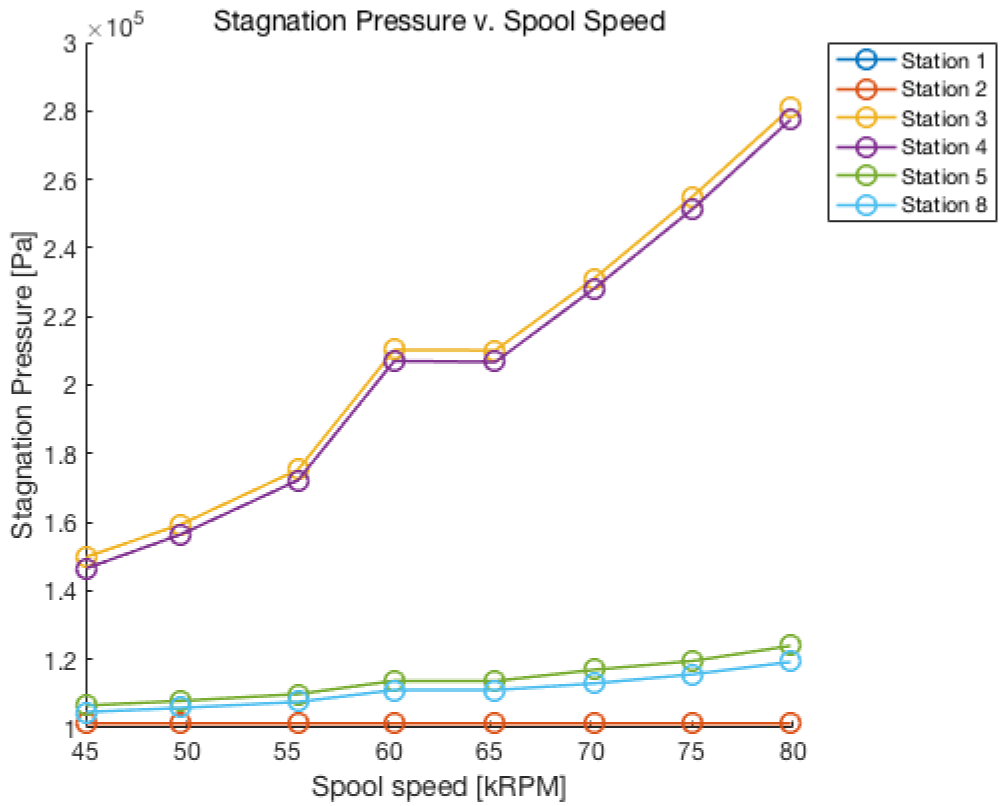
Plot Stagnation Temp vs Spool Speed

```
legendString = {'Station 1', 'Station 2', 'Station 3', 'Station
 4', ...
    'Station 5', 'Station 8'};

plot(krpm, T0, '-o');
xlabel('Spool speed [kRPM]');
ylabel('Stagnation Temp [k]');
legend(legendString, 'Location', 'bestoutside');
title('Stagnation Temperature v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/stagTVsRpm');
```



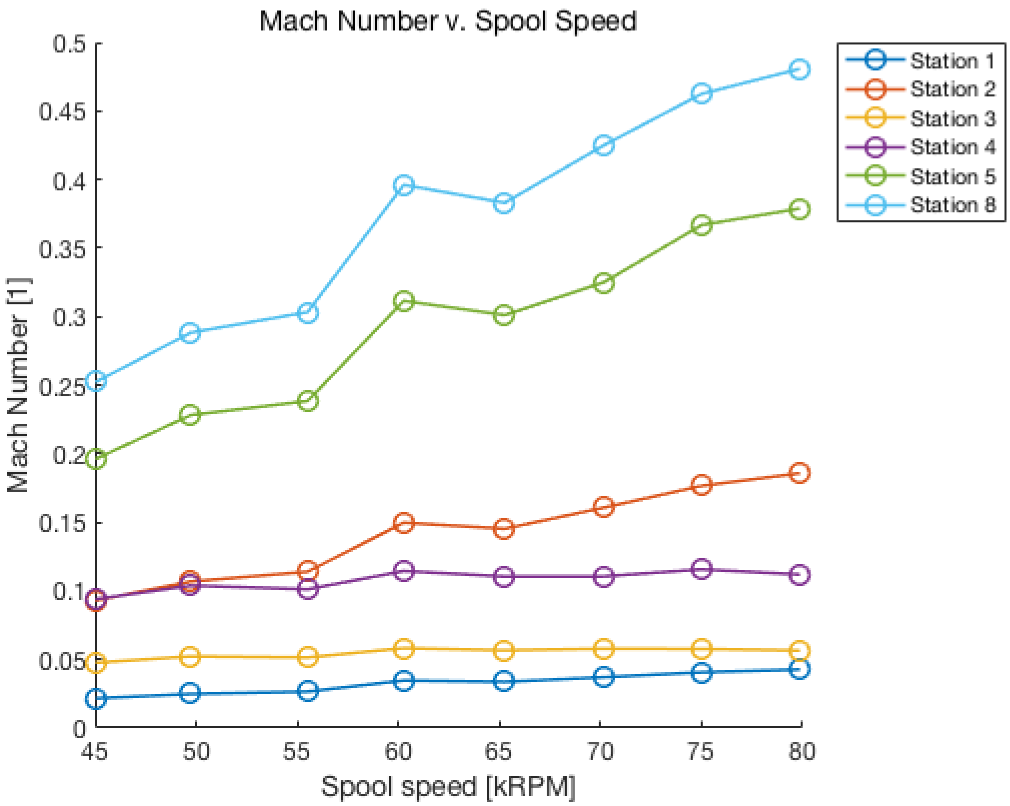
Plot Stagnation Pressure vs Spool Speed

```
plot(krpm, P0, '-o');
xlabel('Spool speed [kRPM]');
ylabel('Stagnation Pressure [Pa]');
legend(legendString, 'Location', 'bestoutside');
title('Stagnation Pressure v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/stagPVsRpm');
```



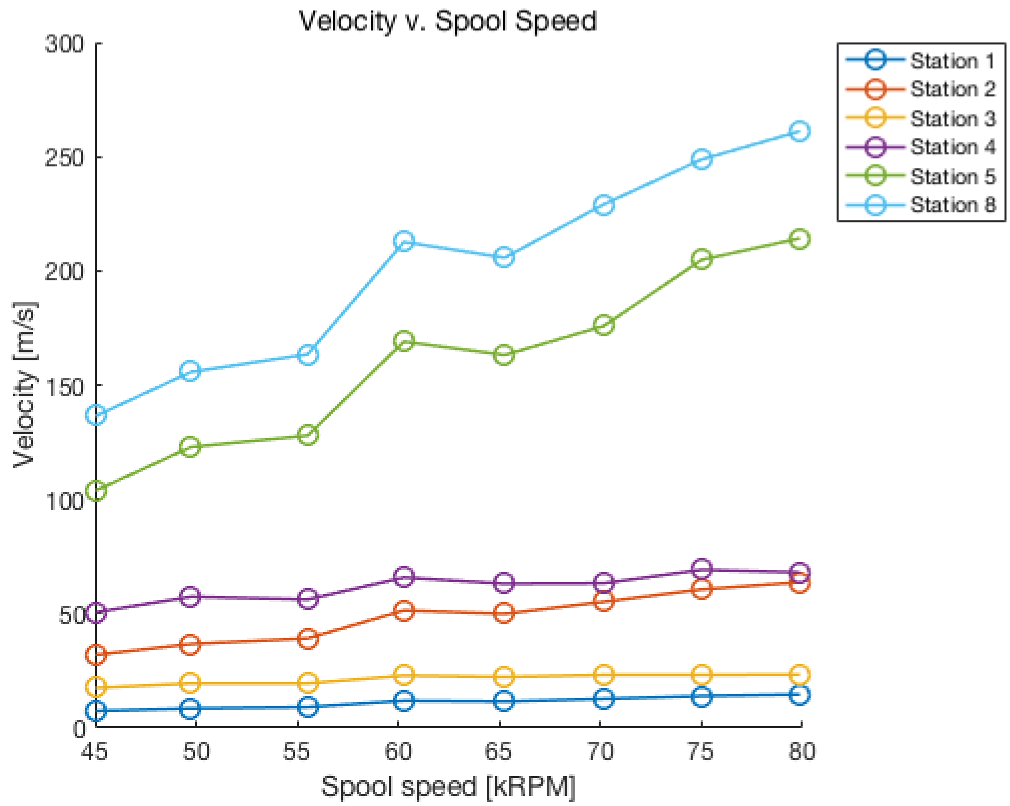
Mach Number vs Spool Speed

```
plot(krpm, Ma, '-o');
xlabel('Spool speed [kRPM]');
ylabel('Mach Number [1]');
legend(legendString, 'Location', 'bestoutside');
title('Mach Number v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/MaVsRpm');
```



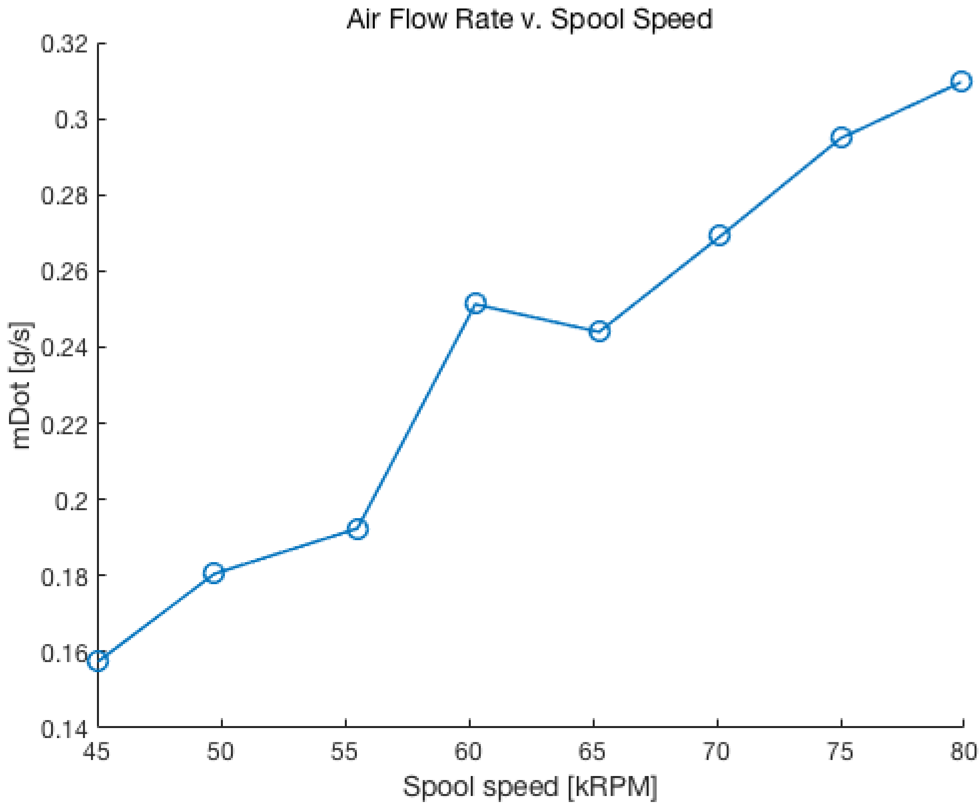
Velocity vs Spool Speed

```
plot(krpm, V, '-o');
xlabel('Spool speed [kRPM]');
ylabel('Velocity [m/s]');
legend(legendString, 'Location', 'bestoutside');
title('Velocity v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/velVsRpm');
```



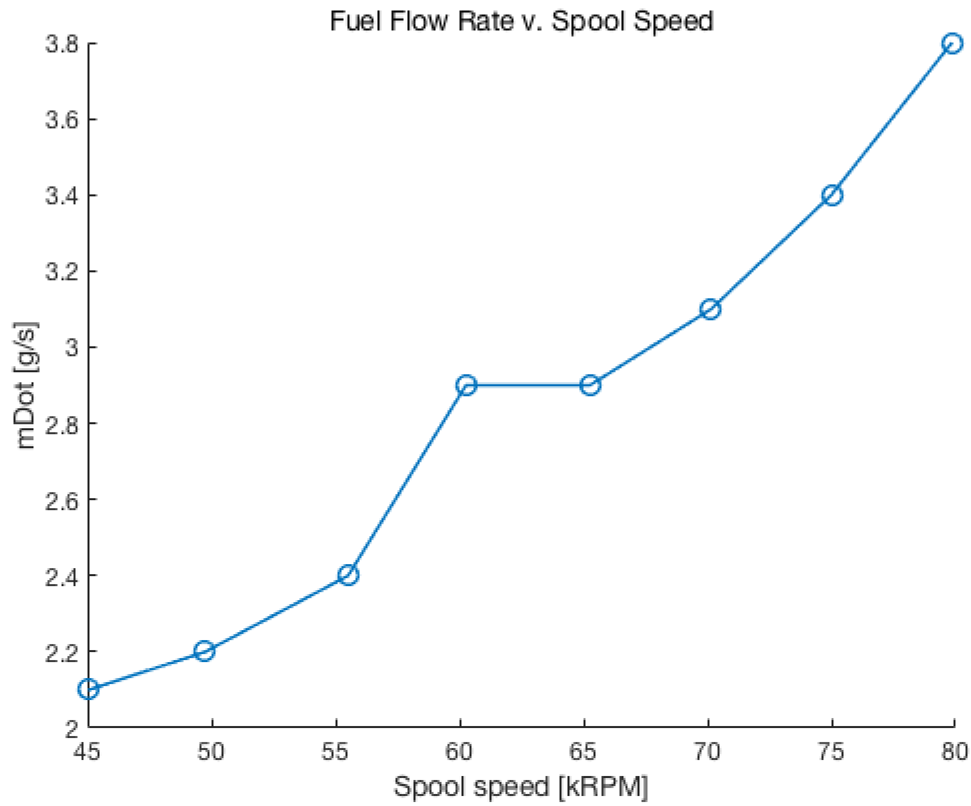
Mass Flow Rate vs Spool Speed

```
plot(krpm, mdotAir, '-o');
xlabel('Spool speed [kRPM]');
ylabel('mDot [g/s]');
title('Air Flow Rate v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/mdotAirVsRpm');
```



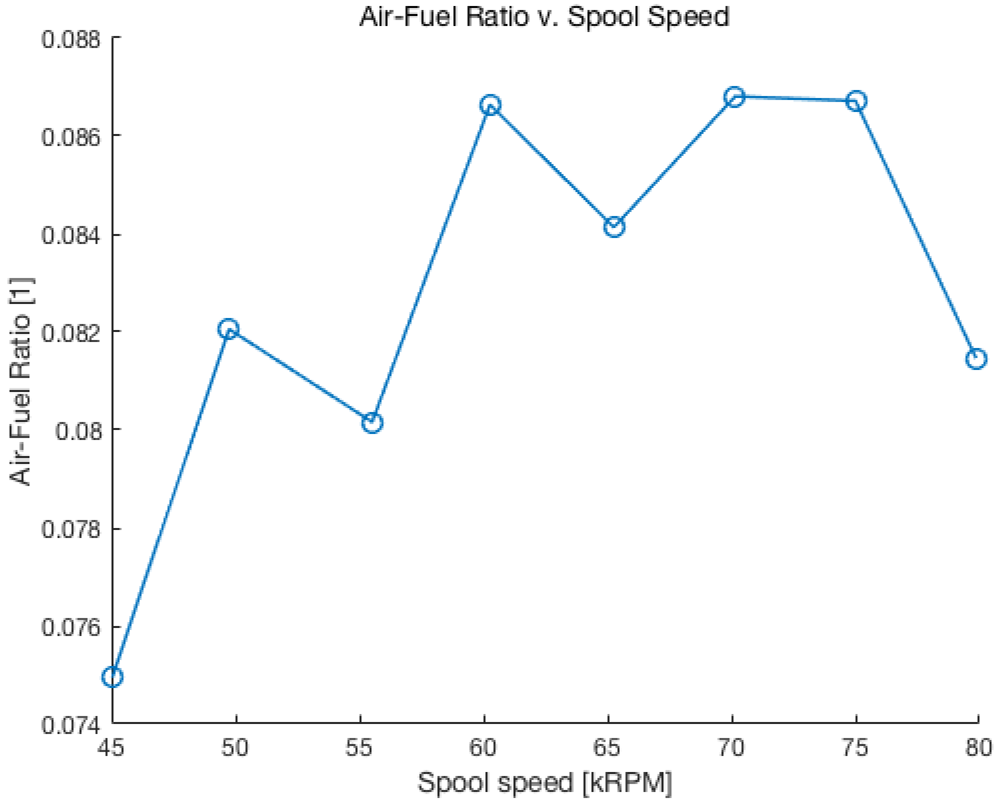
Fuel Flow Rate vs Spool Speed

```
mdotFuel = collectedData.FuelFlow * 1e3; % [g/s]

plot(krpm, mdotFuel, '-o');
xlabel('Spool speed [kRPM]');
ylabel('mDot [g/s]');
title('Fuel Flow Rate v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/mdotFuelVsRpm');
```



Air/Fuel Ratio vs. Spool Speed

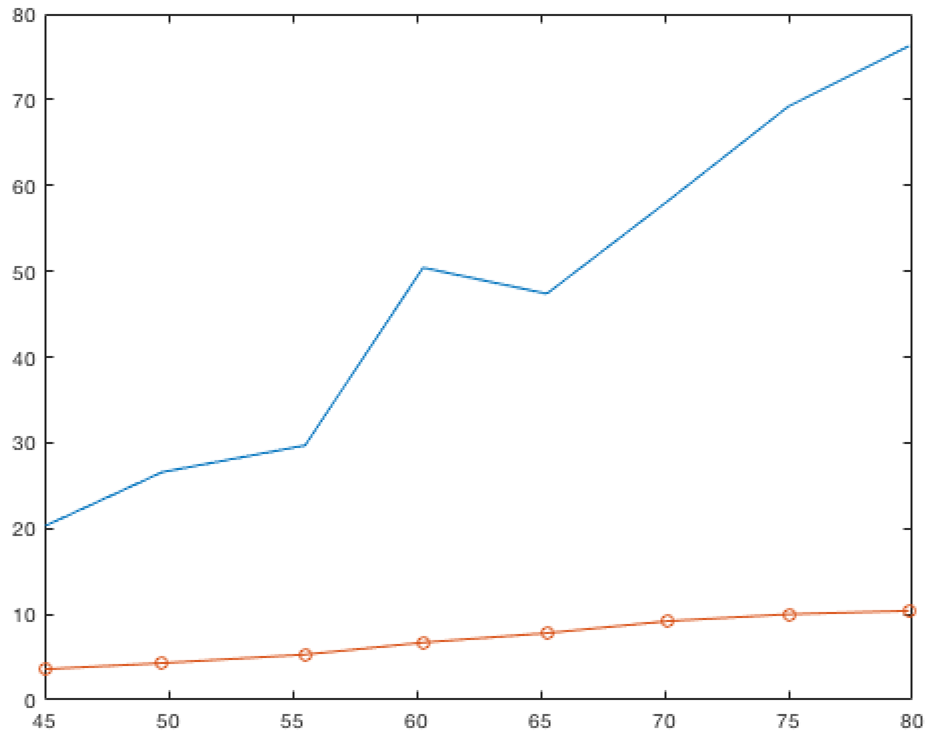
```
airFuelRatio = mdotAir ./ mdotFuel;

plot(krpm, airFuelRatio, '-o');
xlabel('Spool speed [kRPM]');
ylabel('Air-Fuel Ratio [1]');
title('Air-Fuel Ratio v. Spool Speed');
plotFixer();
print('-depsc','-tiff','-r300','plots/fuelAirRatioVsRpm');
```



Calculate Thrust versus Experimental

TODO: COMPLETE

```
idealThrust = mdotAir.*(V(:, end) - V(:, 1));
plot(krpm, idealThrust, krpm, collectedData.Thrust, '-o');
print('-depsc','-tiff','-r300','plots/thrustsVsRpm');
```



Published with MATLAB® R2017b