Assignment 8 ENGR 220

Table of Contents

Ethan Vosburg

Problem 1

The following are experimental data from a tensile test of aluminum to determine an important elastic property, E (Elastic Modulus):

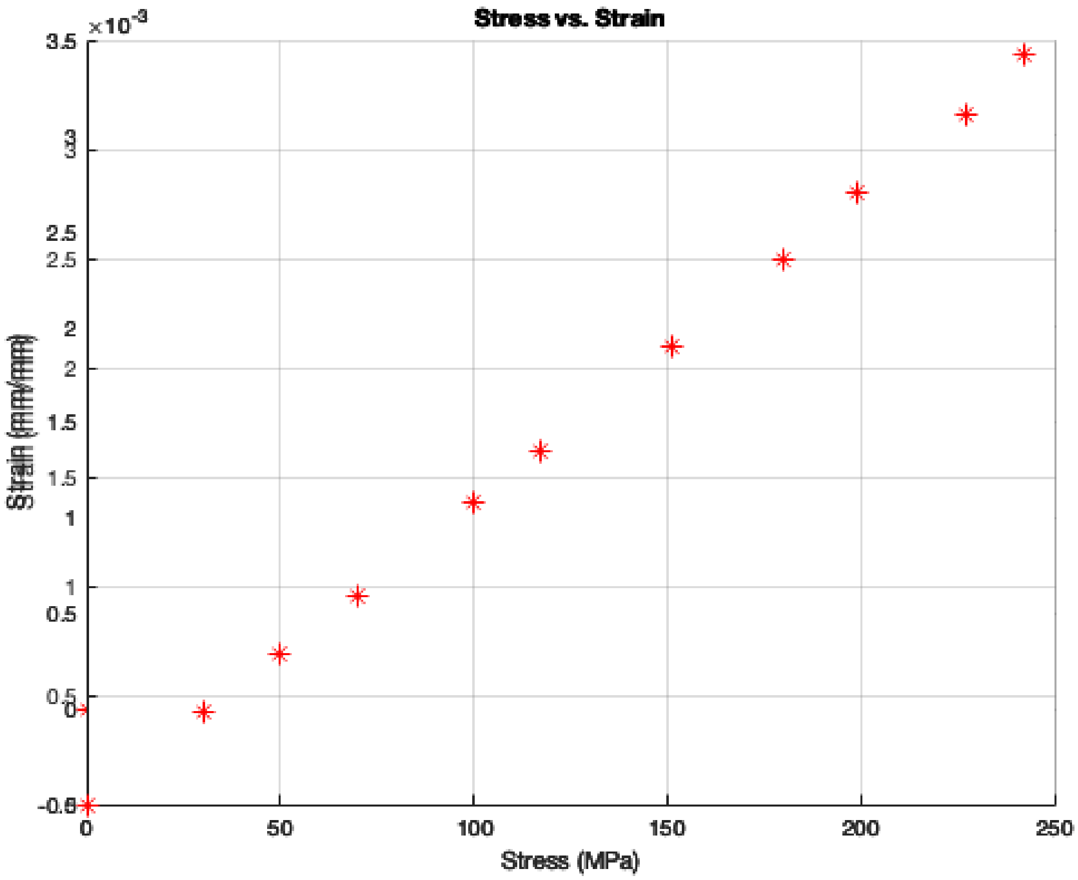
```
clear
clc

problem1Data(1, :) = [0, 30, 50, 70, 100, 117, 151, 180, 199, 227, 242]; % Stress (MPa)
problem1Data(2, :) = [0, 0.00042, 0.00069, 0.00095, 0.00138, 0.00162, 0.00210, 0.00250, 0.00280, 0.00316, 0.00343]; % Strain (mm/mm)
```

Problem 1a

Plot the data points with “Strain” on the x-axis and “Stress” on the y-axis. Be sure to show the data on the plot as points and not as a line.

```
figure(10);
problem1Plot = axes();
hold(problem1Plot, 'on');

plot(problem1Data(1, :), problem1Data(2, :), "r*", "Parent", problem1Plot);
```

```
% Create title
title(problem1Plot, {'Stress vs. Strain'});

% Create ylabel
ylabel(problem1Plot, {'Strain (mm/mm)'});

% Create xlabel
xlabel(problem1Plot, {'Stress (MPa)'});

% Set the remaining axes properties
grid(problem1Plot, 'on');
```



Problem 1b

Use the polyfit function to determine the coefficients of the best-fit line. Note: the slope here is the material property "Elastic Modulus".

```
problem1Fit = polyfit(problem1Data(1, :), problem1Data(2, :), 1);
```

Problem 1c

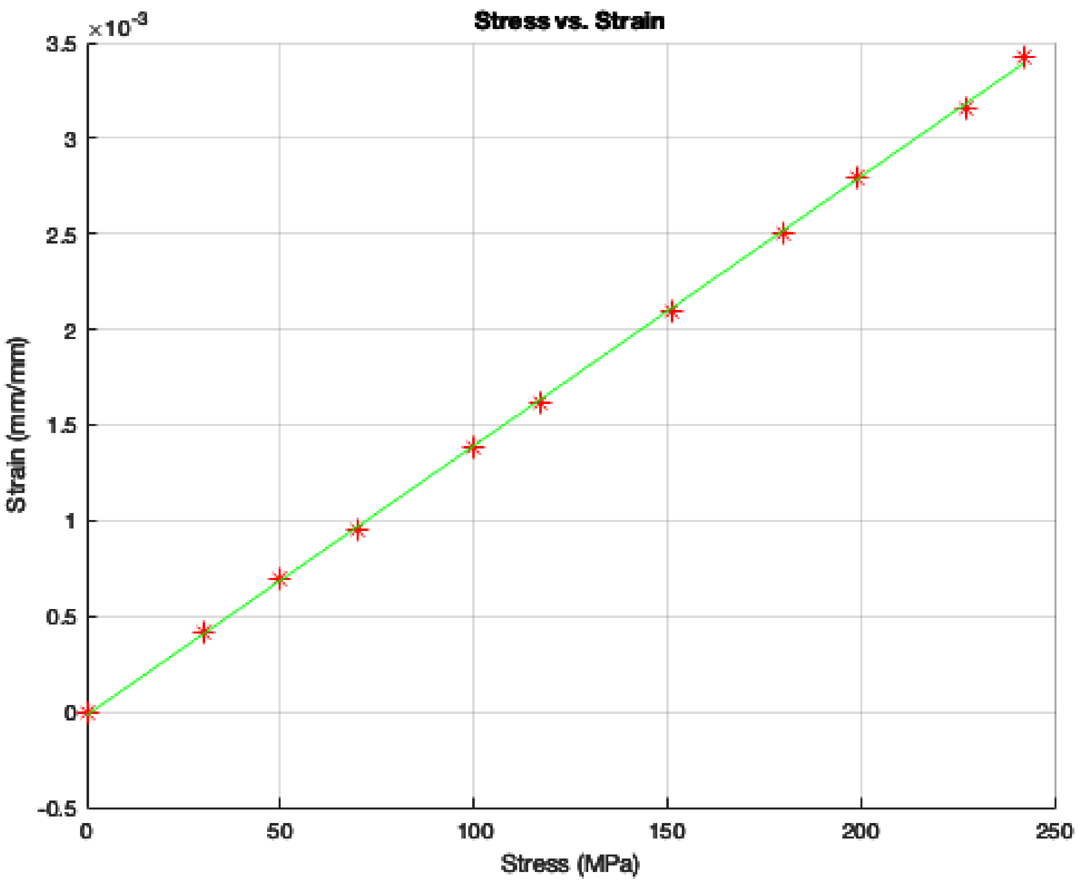
Create "Strain" data to apply to the equation of the best-fit line and determine "Stress" values over the range of "Strain" values (minimum to maximum strain).

```
problem1BestFitData(1, :) = linspace(0, max(problem1Data(1, :)), 1000);
problem1BestFitData(2, :) = problem1Fit(1) .* problem1BestFitData(1, :) +
 problem1Fit(2);
```

Problem 1d

Plot the best-fit line through the data from a.

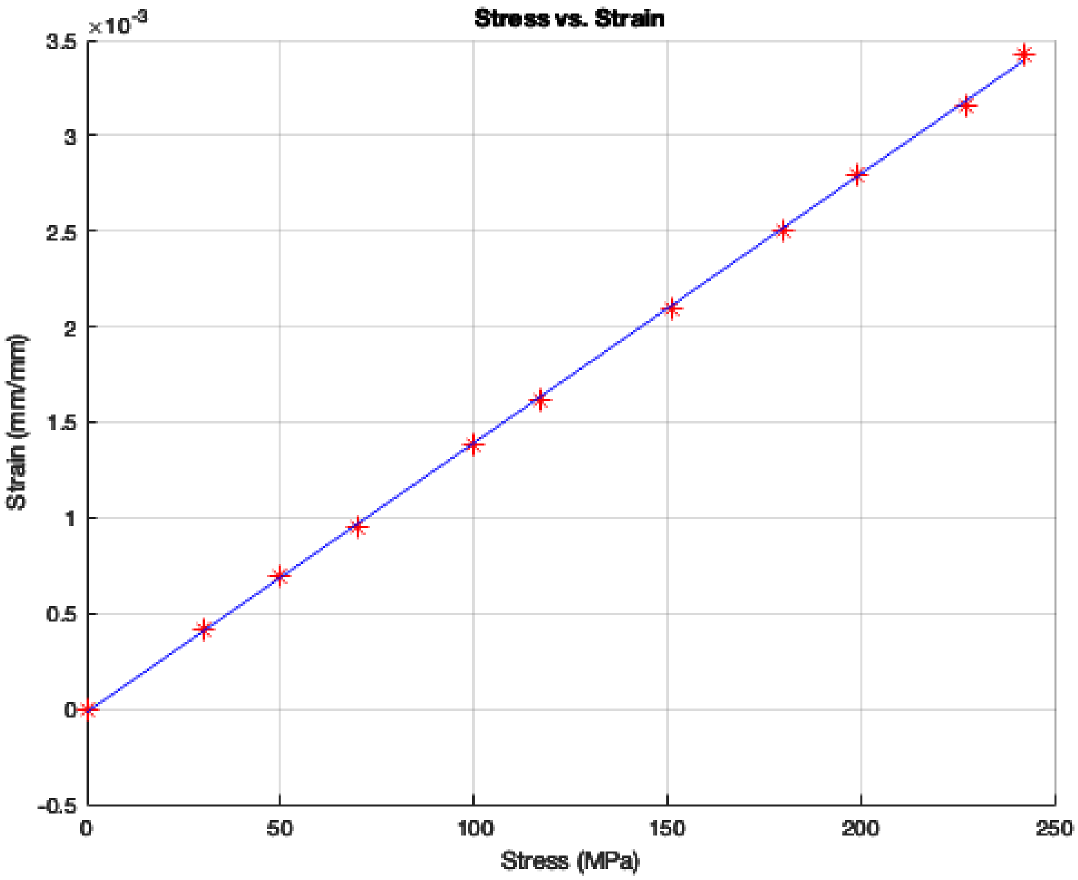
```
plot(problem1BestFitData(1, :), problem1BestFitData(2, :), "-g", "Parent",
 problem1Plot);
```



Problem 1e

Complete parts c. and d. again using the “polyval” function.

```
fplot(@(x) polyval(problem1Fit, x), [0, max(problem1Data(1, :))], "-
b", "Parent", problem1Plot);
```



Problem 1f

Report the results of this experiment by creating code that does the following:

Problem 1fi

Use the “table” function to create a table of Stress & Strain values in columnar form. Use the help section in MATLAB to figure out how to add column titles, and see if you can show units for the “Stress” column.

```
tableNames = ["Stress (MPa)", "Strain (mm/mm)"];

table(problem1Data(1, :)', problem1Data(2, :)', 'VariableNames', tableNames)
```

```
ans =

  11x2 table

    Stress (MPa)    Strain (mm/mm)
    ____________    ______________

          0                0
```

```
   30          0.00042
   50          0.00069
   70          0.00095
  100          0.00138
  117          0.00162
  151           0.0021
  180           0.0025
  199           0.0028
  227          0.00316
  242          0.00343
```

Problem 1fii

Use the fprintf function to create a table of the experimental data as shown above (yes, you will have TWO tables in your output). The table should have the main title “Experimental Data” centered above the column subtitles shown. Show the units as above under the column subtitles. Then show the data in the columns as shown above.

```
% Print Table
fprintf('%-12s %15s\r\n', tableNames(1), tableNames(2));
fprintf('%12s %15s\r\n', '------------', '--------------');
fprintf('% 12.0f % 15.5f\r\n', problem1Data);
```

```
Stress (MPa)  Strain (mm/mm)
------------  --------------
           0         0.00000
          30         0.00042
          50         0.00069
          70         0.00095
         100         0.00138
         117         0.00162
         151         0.00210
         180         0.00250
         199         0.00280
         227         0.00316
         242         0.00343
```

Problem 1fiii

Use the fprintf function to report the Elastic Modulus just under the experimental data, starting with “The Elastic Modulus of Aluminum is:”, then report the value with correct units.

```
fprintf("\nThe Elastic Modulus of Aluminum is: %f MPa^-1\n\n",
 problem1Fit(1));
```

```
The Elastic Modulus of Aluminum is: 0.000014 MPa^-1
```

Problem 2

The following are experimental data from a fatigue test of a steel alloy to determine fitting constants (fatigue properties) The data should fit the well-used design equation:

$$\sigma_a = AN_f^B$$

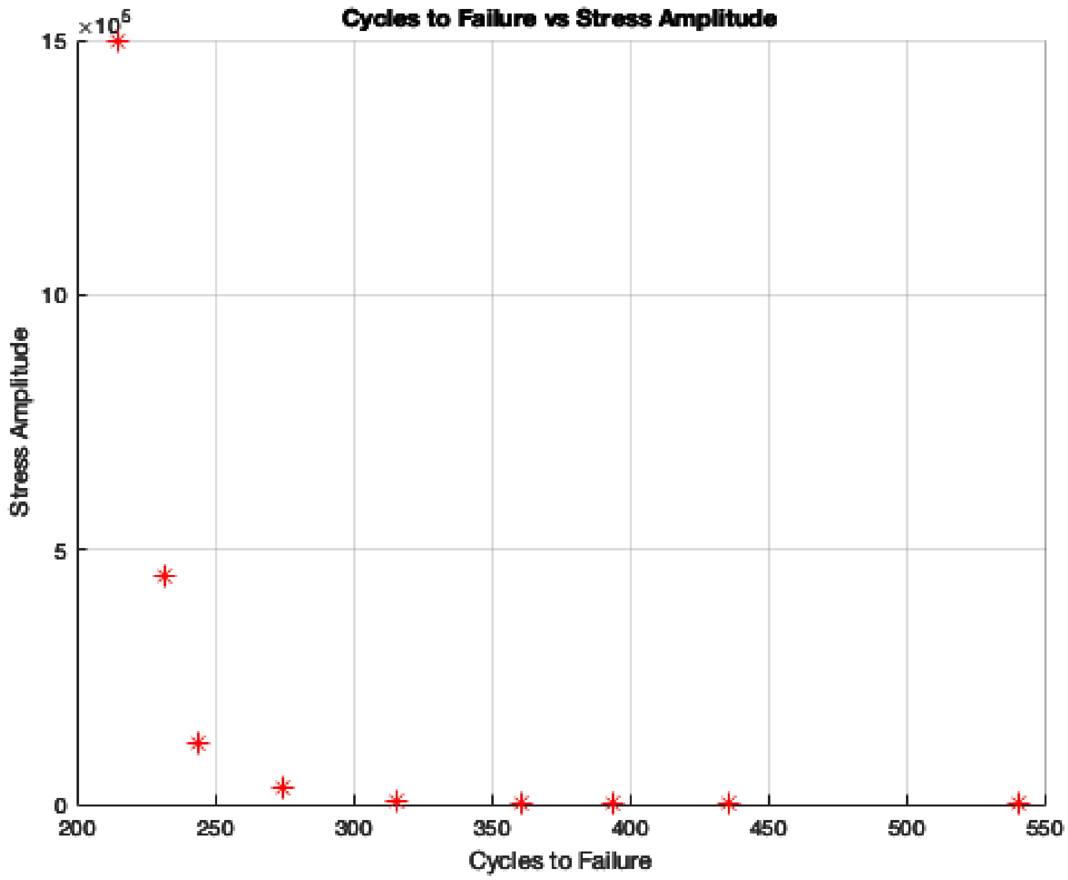
where σ_a is the stress amplitude, N_f is the number of cycles to failure, and A & B are the fitting constants (material properties), determined from the fatigue test.

```
problem2Data(1, :) = [541, 436, 394, 361, 316, 275, 244, 232, 215];
problem2Data(2, :) = [15, 50, 200, 2080, 5900, 34100, 121000, 450000,
 1500000];
problem2Data(3, :) = log(problem2Data(2, :));
```

Problem 2a

Plot the data points with “Cycles to Failure” on the x-axis and “Stress Amplitude” on the y-axis.

```
figure(20);
problem2Plot = axes();
hold(problem2Plot, 'on');

plot(problem2Data(1, :), problem2Data(2, :), "r*", "Parent", problem2Plot);

% Create title
title(problem2Plot, {'Cycles to Failure vs Stress Amplitude'});

% Create ylabel
ylabel(problem2Plot, {'Stress Amplitude'});

% Create xlabel
xlabel(problem2Plot, {'Cycles to Failure'});

% Set the remaining axes properties
grid(problem2Plot, 'on');
```



Problem 2b

Use the polyfit function to determine the coefficients of the function fitting the data (should be a power curve here, so be sure your A & B fitting constants are appropriately determined). A & B are the fatigue constants (material properties).

```
problem2Fit = polyfit(log(problem2Data(1, :)), problem2Data(3, :), 1);
```

Problem 2c

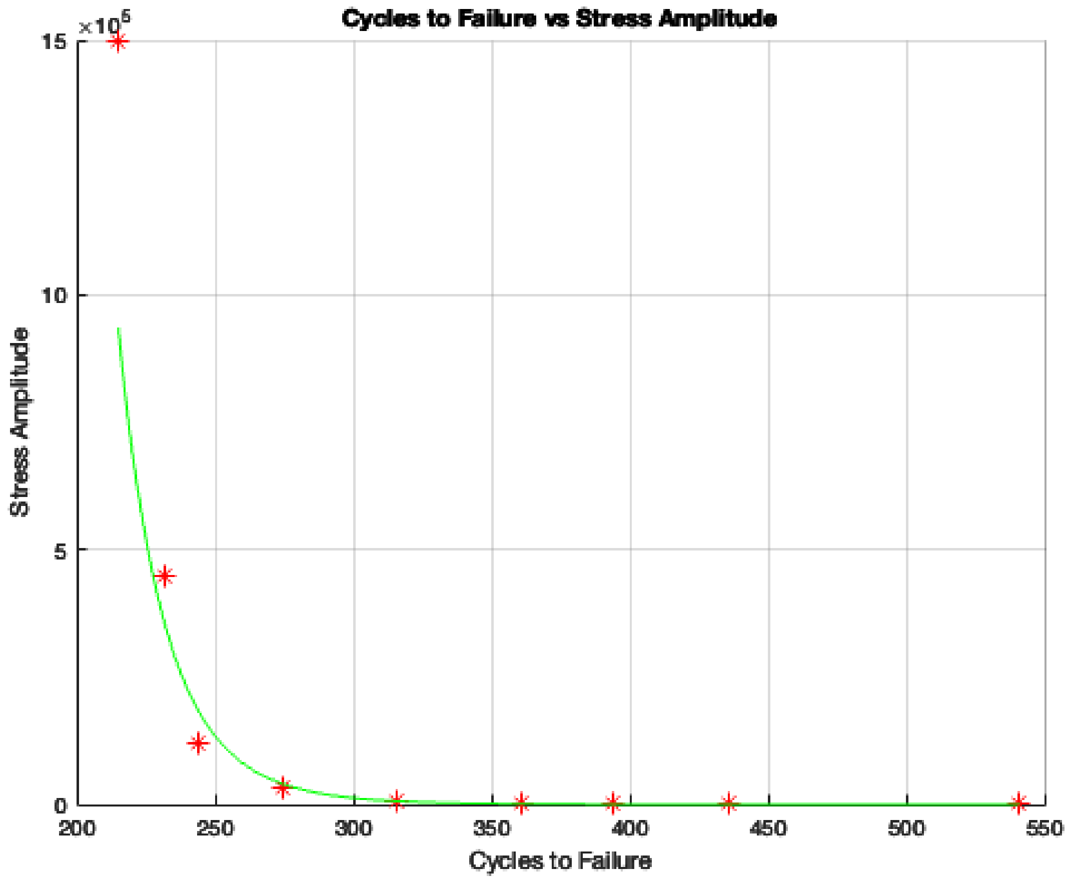
Create "Cycles to Failure" data to apply to the function fitting the data and determine "Stress Amplitude" values over the range of "Cycles to Failure" values (minimum to maximum cycles).

```
problem2BestFitData(1, :) = min(problem2Data(1, :)):max(problem2Data(1, :));
problem2BestFitData(2, :) = exp(problem2Fit(2)) .*
 (problem2BestFitData(1, :) .^ (problem2Fit(1)));
```

Problem 2d

Plot the function as a best-fit curve through the data from a.

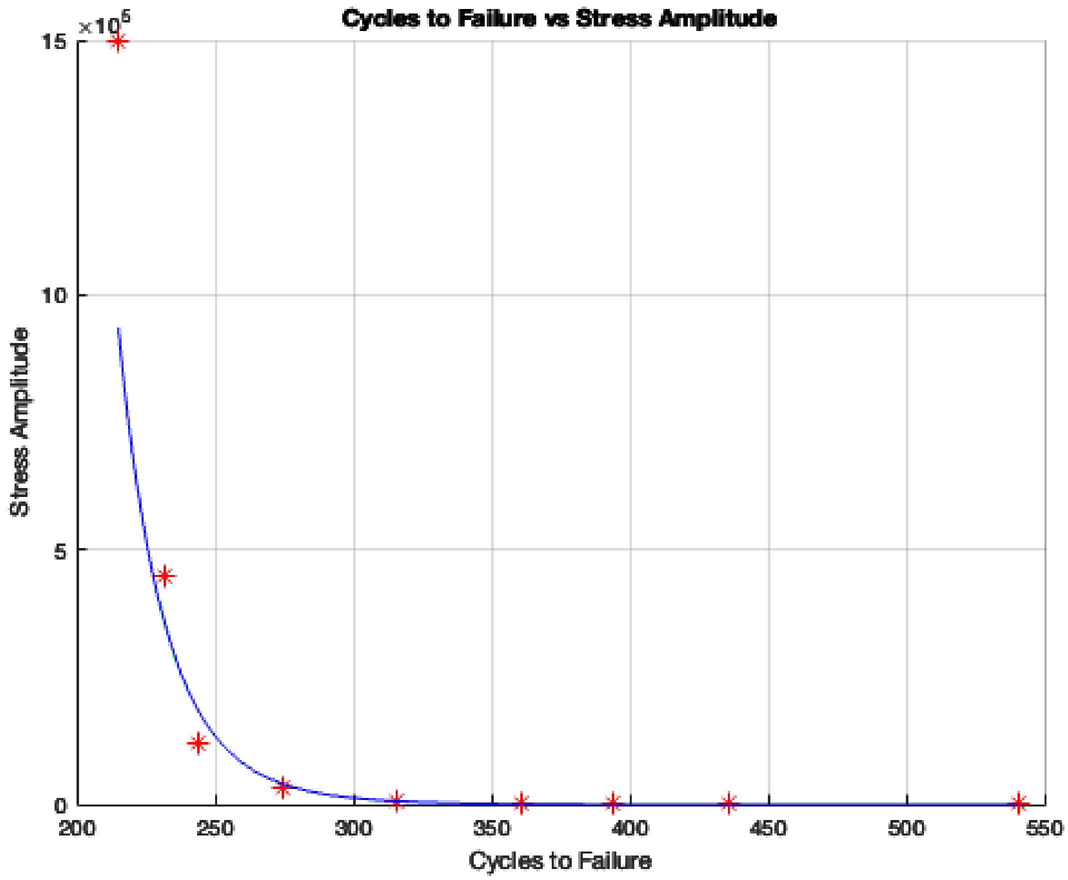
```
plot(problem2BestFitData(1, :), problem2BestFitData(2, :), "-g", "Parent",
 problem2Plot);
```



Problem 2e

Complete parts c. and d. again using the “polyval” function. fplot(@(x) exp(polyval([problem2Fit(1) exp(problem2Fit(2))], (x))), [min(problem2Data(1, :))-1, max(problem2Data(1, :)+1)], "-b", "Parent", problem2Plot);

```
problem2Function = @(x) (exp(problem2Fit(2)) .* (x .^ (problem2Fit(1))));

fplot(problem2Function, [min(problem2Data(1, :)), max(problem2Data(1, :))], "-b", "Parent", problem2Plot);
```



Problem 2f

Report the results of this experiment by creating code that does the following:

Problem 2fi

Use the “table” function to create a table of “Stress Amplitude” & “Cycles to Failure” values in columnar form. Use the help section in MATLAB to figure out how to add column titles, and see if you can show units for the “Stress Amplitude” column.

```
tableNames2 = ["Stress Amplitude", "Cycles to Failure"];

table(problem2Data(1, :)', problem2Data(2, :)', 'VariableNames', tableNames2)
```

```
ans =

  9x2 table

    Stress Amplitude    Cycles to Failure
    ________________    _________________
```

```
541                      15
436                      50
394                     200
361                    2080
316                    5900
275                   34100
244                1.21e+05
232                 4.5e+05
215                 1.5e+06
```

Problem 2fii

Use the fprintf function to create a table of the experimental data as shown above (yes, you will have TWO tables in your output). The table should have the main title “Experimental Fatigue Data” centered above the column subtitles shown. Show the units as above under the column subtitles. Then show the data in the columns as shown above.

```
fprintf('%-16s %17s\r\n', tableNames2(1), tableNames2(2));
fprintf('%16s %17s\r\n', '----------------', '-----------------');
fprintf('% 16.0f % 17.0f\r\n', problem2Data);
```

```
Stress Amplitude Cycles to Failure
---------------- -----------------
             541                15
               3               436
              50                 4
             394               200
               5               361
            2080                 8
             316              5900
               9               275
           34100                10
             244            121000
              12               232
          450000                13
             215           1500000
              14
```

Problem 2fiii

Use the fprintf function to report the fitting constants just under the experimental data, starting with “The fatigue fitting constants of this steel alloy are:”, then report the values with correct units (the exponent B is unit-less).

```
fprintf("\nThe fatigue fitting constants of this steel alloy are: %f\n\n",
 problem2Fit(1));
```

```
The fatigue fitting constants of this steel alloy are: -12.856423
```

Published with MATLAB® R2022b