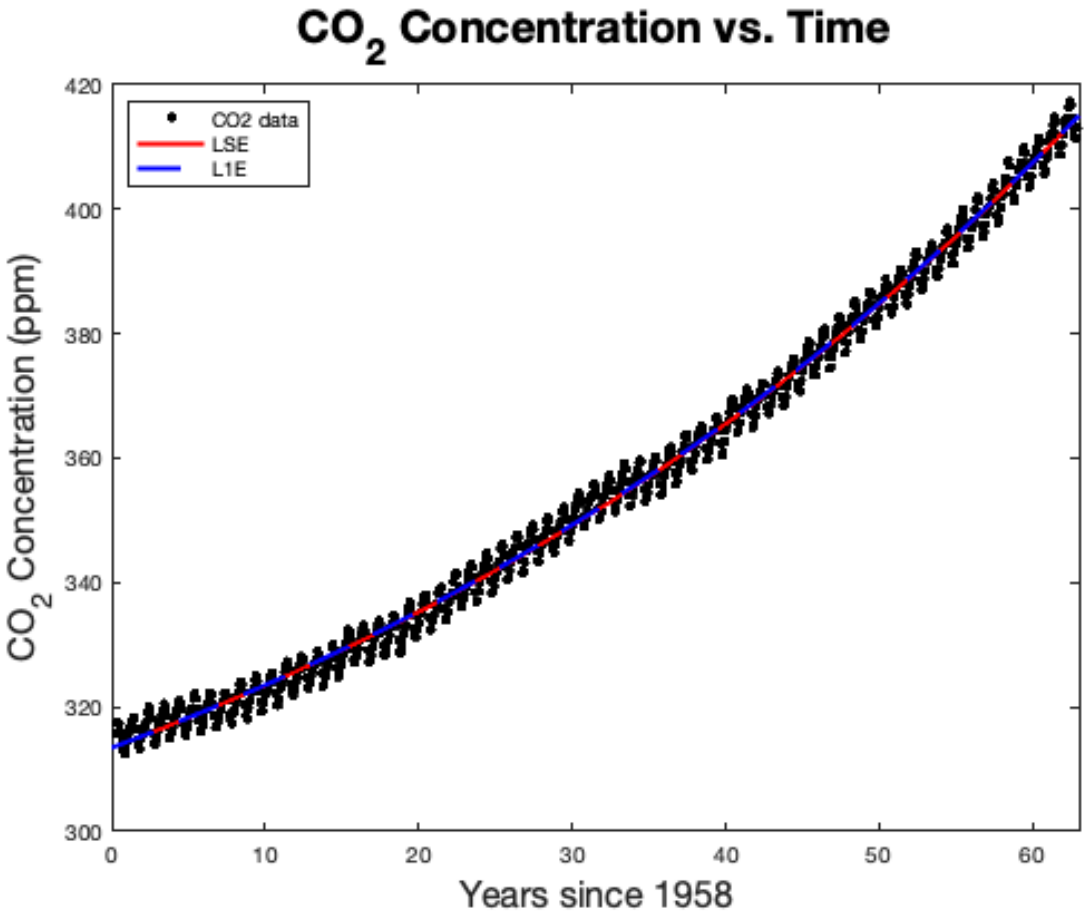
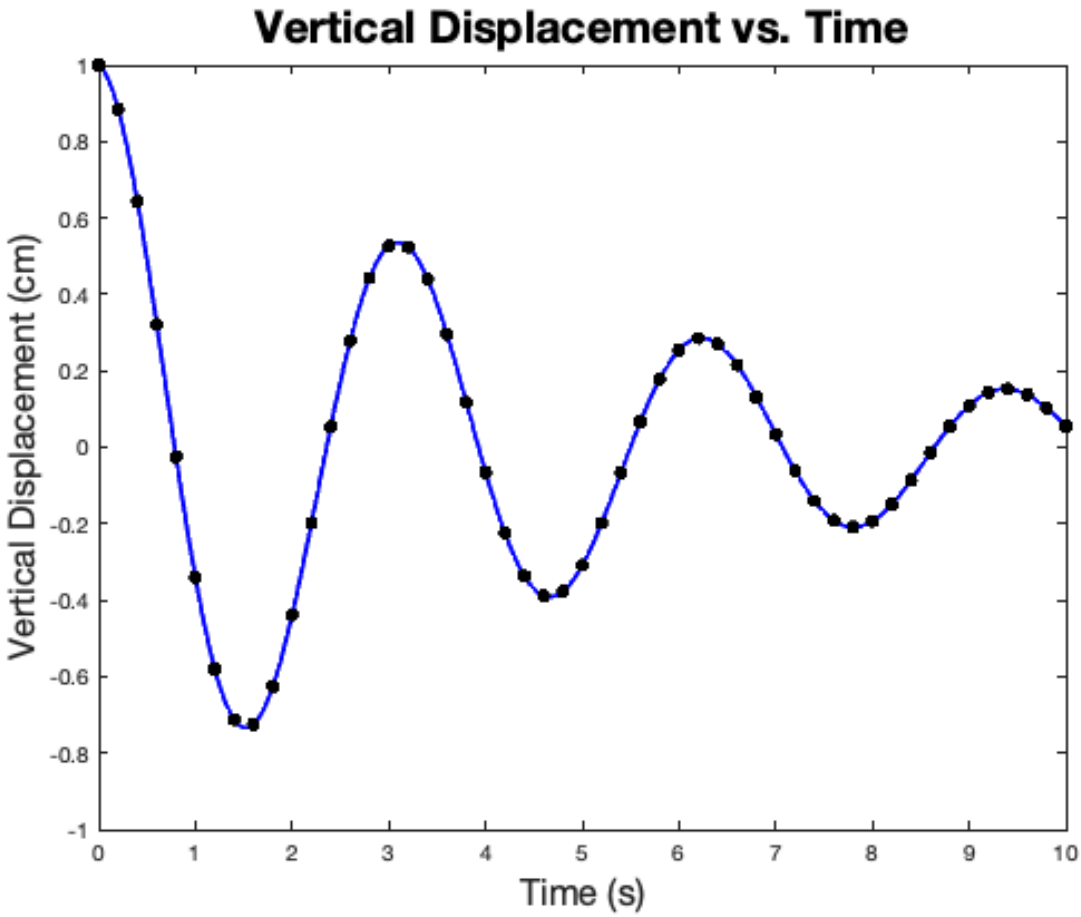
```
% Mason Wheeler
% HW5 Code
% Exercise 1 (Component Skill 5.3) - Part a

% Load the salmon_data.mat file
load('salmon_data.mat');

% first-degree polynomial that best fits the salmon data in the least-squares
 sense
A1 = polyfit(year, salmon, 1);

% Display the coefficients of the first-degree polynomial
disp('Coefficients of the first-degree polynomial:');
disp(A1);

% Exercise 1 (Component Skill 5.3) - Part b

% Use the MATLAB built-in function 'polyfit' to find the coefficients of the
% third-degree polynomial that best fits the salmon data in the least-squares
 sense
% Assign the row vector of coefficients to the variable A2
A2 = polyfit(year, salmon, 3);

% Display the coefficients of the third-degree polynomial
disp('Coefficients of the third-degree polynomial:');
disp(A2);

% Exercise 1 (Component Skill 5.3) - Part c

% Use the MATLAB built-in function 'polyfit' to find the coefficients of the
% fifth-degree polynomial that best fits the salmon data in the least-squares
 sense
% Assign the row vector of coefficients to the variable A3
A3 = polyfit(year, salmon, 5);

% Display the coefficients of the fifth-degree polynomial
disp('Coefficients of the fifth-degree polynomial:');
disp(A3);

% Exercise 1 (Component Skill 5.3) - Part d

% True salmon count in 2021
true_salmon_2021 = 489523;

% Calculate the predicted salmon count in 2021 using the polynomials p1, p3,
 and p5
p1_2021 = polyval(A1, 2021);
p3_2021 = polyval(A2, 2021);
p5_2021 = polyval(A3, 2021);

% Calculate the relative errors (epsilon1, epsilon2, epsilon3)
epsilon1 = abs(p1_2021 - true_salmon_2021) / true_salmon_2021;
```

```
epsilon2 = abs(p3_2021 - true_salmon_2021) / true_salmon_2021;
epsilon3 = abs(p5_2021 - true_salmon_2021) / true_salmon_2021;

% Assign the variable A4 to the 3x1 row vector [epsilon1, epsilon2, epsilon3]
A4 = [epsilon1; epsilon2; epsilon3];

% Display the relative errors
disp('Relative errors:');
disp(A4);

% Exercise 2 (Component Skill 5.3)

% Load CO2_data.mat file
load('CO2_data.mat');

% Part a
% Find the values of a, r, and b that minimize the least-squares error
initial_guess = [30; 0.03; 300];
A5 = fminsearch(@(params) lse(params, year, CO2), initial_guess);

% Part b
% Find the values of a, r, and b that minimize the L1 error
A6 = fminsearch(@(params) l1e(params, year, CO2), initial_guess);

% Plot the results
plot_results(year, CO2, A5, A6);

% Exercise 3 (Component Skill 5.3)

% Load spring_data.mat file
load('spring_data.mat');

% Part a
% Estimate the value of the toy displacement at t = pi/2 seconds using cubic
 spline interpolation
A7 = interp1(time, deltaz, pi/2, 'spline');

% Part b
% Plot the data and cubic spline interpolation
plot_spline(time, deltaz);



% Create separate functions for least-squares error and L1 error
function error = lse(params, year, CO2)
    error = sum((CO2 - (params(1) * exp(params(2) * year) + params(3))).^2);
end

function error = l1e(params, year, CO2)
    error = sum(abs(CO2 - (params(1) * exp(params(2) * year) + params(3))));
end


function plot_results(year, CO2, A5, A6)
```

```
    % Plot CO2 data in black
    figure;
    plot(year, CO2, '.', 'MarkerSize', 10, 'Color', 'k', 'LineStyle', 'none');
    hold on;

    % Plot least-squares error exponential curve in red
    x = linspace(0, 63, 1000);
    y_lse = A5(1) * exp(A5(2) * x) + A5(3);
    plot(x, y_lse, 'r', 'LineWidth', 2);

    % Plot L1 error exponential curve in blue
    y_l1e = A6(1) * exp(A6(2) * x) + A6(3);
    plot(x, y_l1e, 'b--', 'LineWidth', 2);

    % Set axis limits
    xlim([0, 63]);
    ylim([300, 420]);

    % Add legend, labels, and title
    legend('CO2 data', 'LSE', 'L1E', 'Location', 'NorthWest');
    xlabel('Years since 1958', 'FontSize', 16);
    ylabel('CO_2 Concentration (ppm)', 'FontSize', 16);
    title('CO_2 Concentration vs. Time', 'FontSize', 20);

    % Hold off
    hold off;
end

function plot_spline(time, deltaz)
    % Create cubic spline interpolation with sample points t = 0:0.01:10
    t_interp = 0:0.01:10;
    z_interp = interp1(time, deltaz, t_interp, 'spline');

    % Plot cubic spline interpolation in blue
    figure;
    plot(t_interp, z_interp, 'b', 'LineWidth', 2);
    hold on;

    % Superimpose the displacement data in black
    plot(time, deltaz, 'k.', 'MarkerSize', 15, 'LineStyle', 'none');

    % Set axis limits
    xlim([0, 10]);
    ylim([-1, 1]);

    % Add labels and title
    xlabel('Time (s)', 'FontSize', 16);
    ylabel('Vertical Displacement (cm)', 'FontSize', 16);
    title('Vertical Displacement vs. Time', 'FontSize', 20);

    % Hold off
    hold off;
end
```

```
Coefficients of the first-degree polynomial:
   1.0e+06 *

    0.0042   -7.8654

Warning: Polynomial is badly conditioned. Add points
with distinct X values, reduce the degree of the
polynomial, or try centering and scaling as
described in HELP POLYFIT.
Coefficients of the third-degree polynomial:
   1.0e+10 *

    0.0000   -0.0000    0.0016   -1.0637

Warning: Polynomial is badly conditioned. Add points
with distinct X values, reduce the degree of the
polynomial, or try centering and scaling as
described in HELP POLYFIT.
Coefficients of the fifth-degree polynomial:
   1.0e+14 *

   -0.0000    0.0000   -0.0000    0.0000   -0.0103    4.0754

Relative errors:
    0.2233
    0.6722
    0.1778
```


CO_2 Concentration vs. Time
CO2 data
LSE
L1E
CO_2 Concentration (ppm)
Years since 1958
420
400
380
360
340
320
300
0
10
20
30
40
50
60



Published with MATLAB® R2022b