MAT 343 Lab 5 - Your name

Exercise 1

Load the file co2.dat

```
% Exercise 1
% Load the CO2 data file
co2 = load('co2.dat');      % assumes co2.dat is in the same folder

% Columns: year, concentration, (third column is not used here)
yeat = co2(:, 1);           % years
conc = co2(:, 2);           % CO2 concentration
```

Create the data vectors yeat and conc and plot the data points.

```
% Create the data vectors and plot the data points
figure
plot(yeat, conc, 'o')
xlabel('Year')
ylabel('CO2 concentration (ppm)')
title('CO2 data')
grid on
```

(a)

Write the code for finding and plotting the best linear fit. Make sure the vector c is displayed.

```
% Linear least squares fit: conc ≈ c1 + c2 * year

A_lin = [ones(size(yeat)) yeat];   % design matrix for linear model
c_lin = A_lin \ conc               % c_lin is displayed because there is no
semicolon
```

```
c_lin = 2×1
10^3 ×
   -2.9393
    0.0017
```

```

% Plot the linear fit on a fine grid
year_fine = linspace(min(yeat), max(yeat), 300).';
conc_lin_fine = c_lin(1) + c_lin(2) * year_fine;

hold on
plot(year_fine, conc_lin_fine, '-')
legend('Data', 'Best linear fit', 'Location', 'best')
```

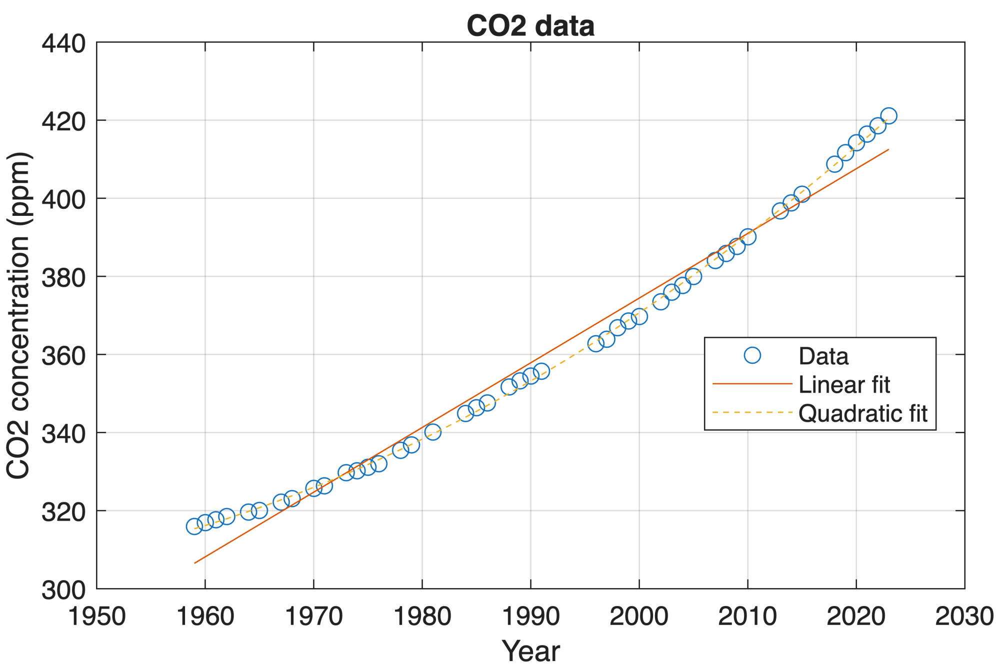
(b)

Write the code for finding the quadratic fit and plot the quadratic and linear fit in the same figure. Make sure the vector c is displayed.

```
% Quadratic least squares fit: conc ≈ c1 + c2 * year + c3 * year^2

A_quad = [ones(size(yeat)) yeat yeat.^2];  % design matrix for quadratic
model
c_quad = A_quad \ conc                     % c_quad is displayed
```

```
c_quad = 3×1
10^4 ×
    4.7846
   -0.0049
    0.0000
```

```

% Quadratic fit on the same fine grid
conc_quad_fine = c_quad(1) + c_quad(2)*year_fine + c_quad(3)*year_fine.^2;

plot(year_fine, conc_quad_fine, '--')
legend('Data', 'Linear fit', 'Quadratic fit', 'Location', 'best')
hold off
```



Question 2

(a)

Enter the vector t (use the colon operator ":") and the vector Y *(note the upper case) as given in the statement of the problem. Plot the data points* (t,Y), *find the best linear fit and plot it together with the points. Make sure the vector c is displayed.*

```
% Question 2(a)
% Enter t and g as COLUMN vectors

t = (0:2:20)';         % Example — replace with YOUR t values
g = [5; 9; 14; 22; 35; 52; 77; 110; 160; 230; 330];   % Replace with YOUR g data
```

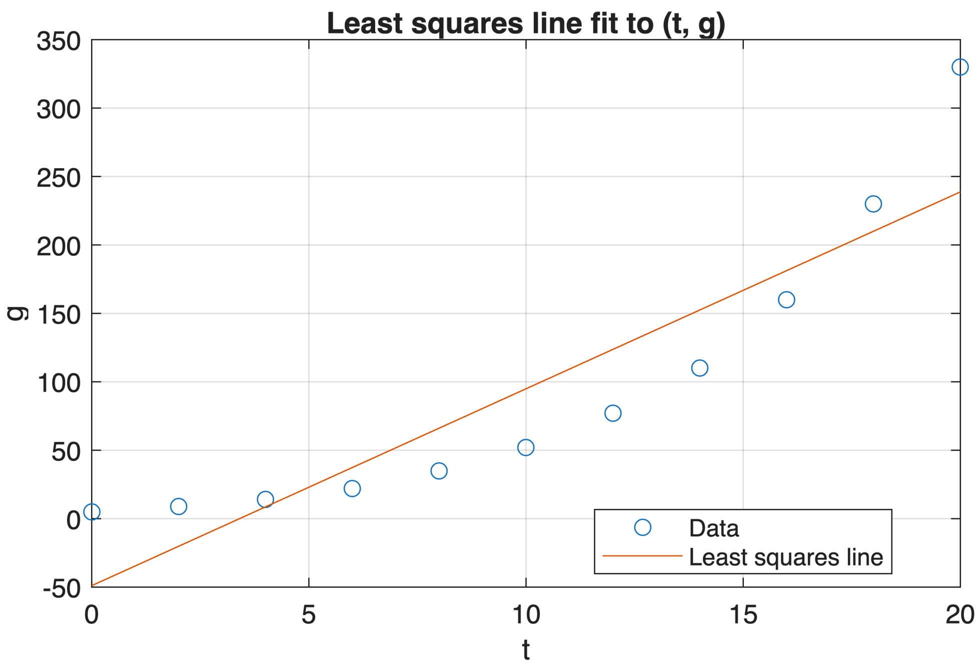
(b)

Define y (lower case) in terms of Y (upper case), and a and b in terms of c(1) and c(2) where c is the vector from part (a)

```
% Question 2(b)
% Least squares fit g ≈ c1 + c2 * t

A2 = [ones(size(t)) t];
c = A2 \ g               % display c
```

```
c = 2×1
  -48.9545
   14.3864
```

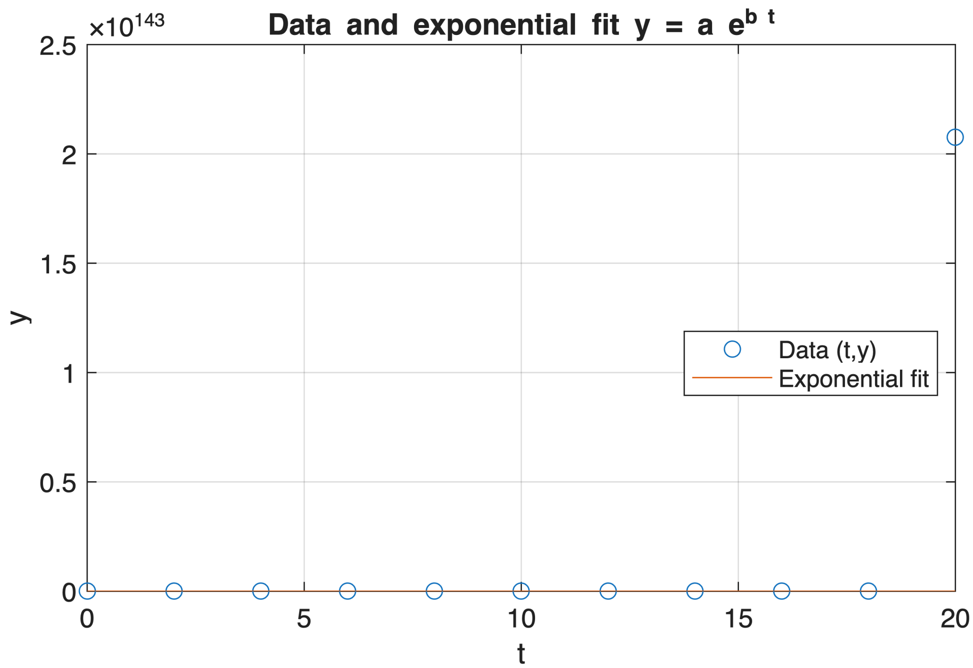
```
% Plot data and least squares line
t_fine = linspace(min(t), max(t), 200).';
g_fit = c(1) + c(2)*t_fine;

figure
plot(t, g, 'o', t_fine, g_fit, '-')
xlabel('t')
ylabel('g')
title('Least squares line fit to (t, g)')
legend('Data', 'Least squares line', 'Location', 'best')
grid on
```




```
y = exp(g);   % convert Y = log(y) back to y
```

plot the data points (t,y) together with the exponential fit $y = ae^{bt}$. Use an appropriate vector q to graph the exponential fit.

```
% Using c from part (a), where Y = log(y) and Y ≈ c(1) + c(2)*t
% So: a = e^{c(1)},  b = c(2)

a = exp(c(1));
b = c(2);

% Recover y from Y (which in this lab is stored in g)
y = exp(g);

% Vector for smooth exponential curve
q = linspace(min(t), max(t), 200).';   % column vector
y_exp = a * exp(b*q);

% Plot data and exponential fit
figure
plot(t, y, 'o', q, y_exp, '-')
xlabel('t')
ylabel('y')
title('Data and exponential fit y = a e^{b t}')
legend('Data (t,y)', 'Exponential fit', 'Location', 'best')
grid on
```



(c)

Enter the time at which the balance will reach the given amount (make sure you explain how you find the value of t)

Question 3

Enter the vector m (use the colon operator :), and enter the average temperature Y. Both vectors must be **column** vectors.

```
% Question 3
% Enter m and Y as COLUMN vectors

m = (1:12)';            % months 1 through 12
Y = [                   % replace with your temperature data
    32;
    35;
    41;
    50;
    60;
    68;
    73;
    71;
    64;
    54;
```

```
    45;
    35
];
```

(a)

```
% Question 3(a)
% Least squares fit Y ≈ c1 + c2 * m using normal equations

A3 = [ones(size(m)) m];

% Normal equations: (A3' * A3) * c = A3' * Y
c = (A3.' * A3) \ (A3.' * Y)    % display c
```

```
c = 2×1
   45.0152
    1.1259
```

```
m_fine = linspace(min(m), max(m), 200).';
Y_fit = c(1) + c(2) * m_fine;

figure
plot(m, Y, 'o', m_fine, Y_fit, '-')
xlabel('m')
ylabel('Y')
title('Least squares line from normal equations')
legend('Data', 'Normal equation fit', 'Location', 'best')
grid on
```

(b)

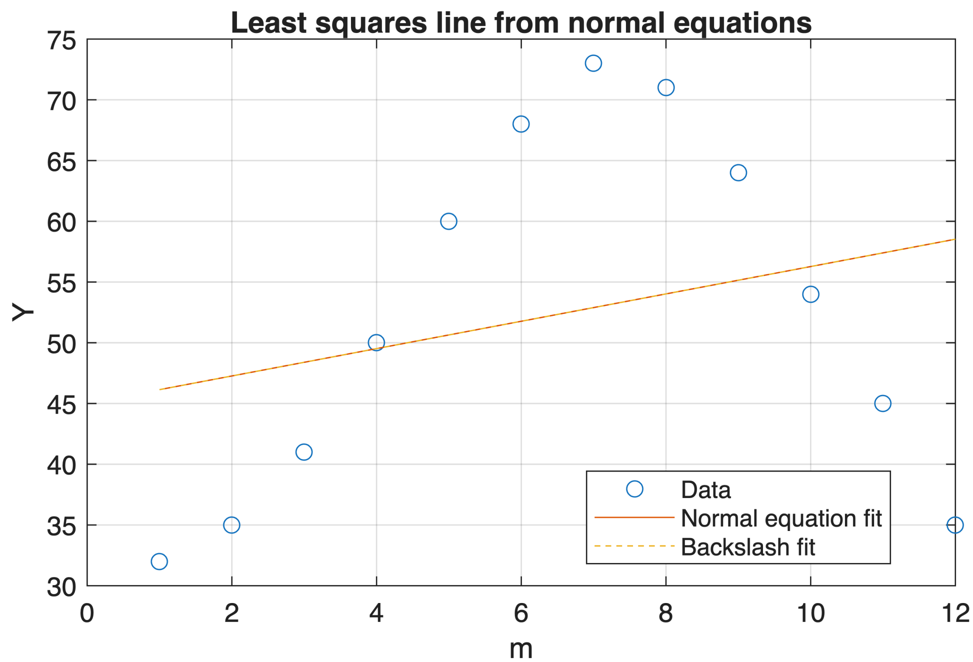
Enter the given commands

```
% Question 3(b)
% Same model using MATLAB backslash instead of explicit normal equations

A3 = [ones(size(m)) m];
c2 = A3 \ Y            % display c2
```

```
c2 = 2×1
   45.0152
    1.1259
```

```
Y_fit2 = c2(1) + c2(2) * m_fine;

hold on
plot(m_fine, Y_fit2, '--')
legend('Data', 'Normal equation fit', 'Backslash fit', 'Location', 'best')
```

```
hold off
```



How do the values of c compare to the ones you found in part (a)?

Answer: **both methods produce the same best-fit line**, and any difference is too small to matter.

How does the plot compare to the one you found in part (a)?

Answer: The plots lie **directly on top of each other**, showing the same straight line.