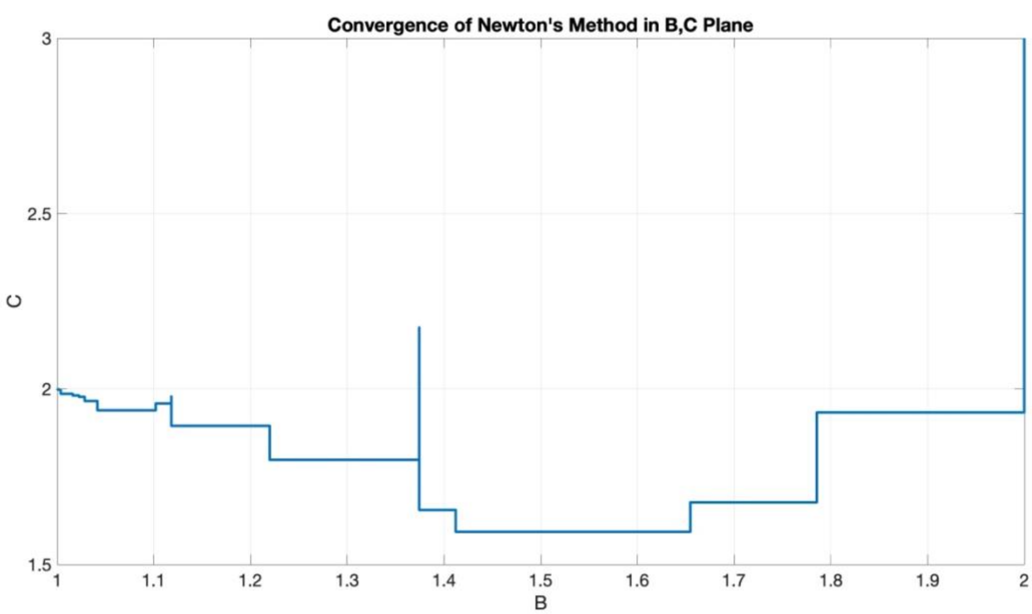
Problem 1

```
%%

clear, close all

%% Finding values a,b,c using system of linear equations

x = [-4, 0, 4];
y = [30, 2, 6];

A = [x.^2', x', ones(3, 1)];

d = y';

coefficients = A \ d;

a = coefficients(1);
b = coefficients(2);
c = coefficients(3);

fprintf('Coefficients [a, b, c]: %f, %f, %f\n', a, b, c);

%%
d = [30;2;6];

%% Project Matrix

P = A * inv(A' * A) * A';

disp("The projection matrix P:");

disp(P)
```

```
%% Predicted

predicted = P * d;

disp("The predicted values are:");

disp(predicted)

%% Error of vector e

e = d - predicted;

disp("The error vector e is:");

disp(e)

%% Newton's Method

% Initial iteration
A = 1;
B = 2;
C = 3;

% taylor series
f = @(A, B, C) sum((y - A*(x - B).*(x - C)).^2);
f_A = @(A, B, C) -2*sum((x - B).*(x - C).*(y - A*(x - B).*(x - C)));
f_B = @(A, B, C) 2*A*sum((x - B).*(y - A*(x - B).*(x - C))) + 2*A*sum((x - C).*(y - A*(x - B).*(x - C)));
f_C = @(A, B, C) 2*A*sum((x - B).*(y - A*(x - B).*(x - C))) + 2*A*sum((x - B).*(y - A*(x - B).*(x - C)));

tol = 1e-6;
iter = 100000;

%newton's method
```

```
for i = 1:iter
    df = [f_A(A, B, C), f_B(A, B, C), f_C(A, B, C)];
    delta = df \ -f(A, B, C);
    A = A + delta(1);
    B = B + delta(2);
    C = C + delta(3);
    if max(abs(delta)) < tol
        break;
    end
end

fprintf('Estimated values:\n');
fprintf('A = %f\n', A);
fprintf('B = %f\n', B);
fprintf('C = %f\n', C);

%% Plotting

% Importing initial values
A = 1;
B = 2;
C = 3;

con_B = [];
con_C = [];

% Newton's method
for i = 1:iter
    df = [f_A(A, B, C) f_B(A, B, C) f_C(A, B, C)]; %Jacobian
    delta = df \ -f(A, B, C); %solve
    A = A + delta(1);
    B = B + delta(2);
    C = C + delta(3);
```

```
    con_B(end+1) = B;
    con_C(end+1) = C;

    if max(abs(delta)) < tol
        break;
    end
end

% Plotting convergence in the B,C plane
figure(1)
plot(con_B, con_C, '-','LineWidth',3);
xlabel('B');
ylabel('C');
title('Convergence of Newton''s Method in B,C Plane');
grid on;
set(gca,'FontSize',20)

%%
```



What happens if you choose A = 0 as the initial iterate? Why can you rule out the value of A = 0 as a possible value?

- I think you cannot have A=0. My code gives me errors when doing the calculations. Yes, we can rule out A=0.

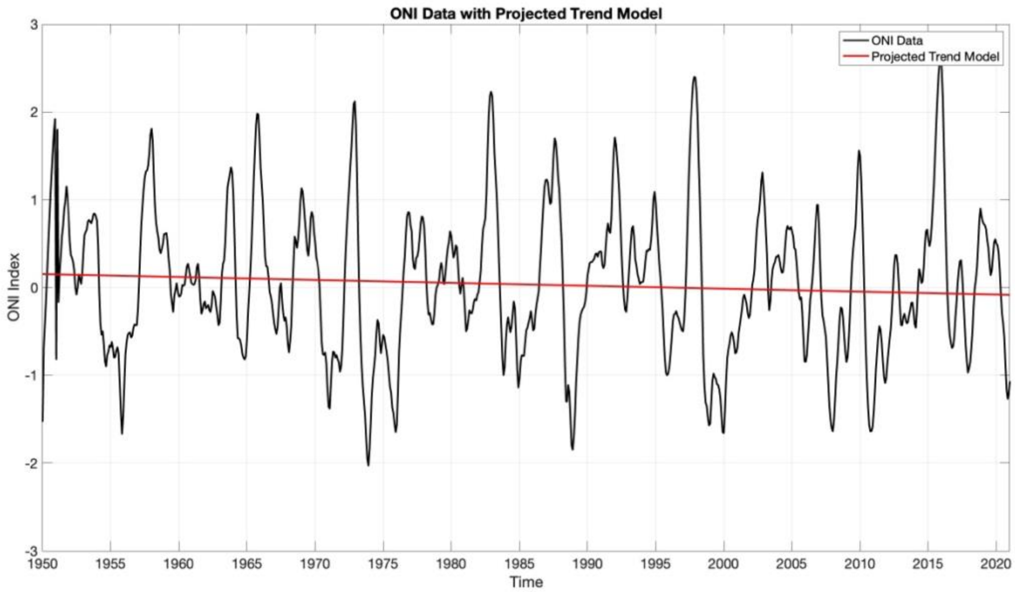
Problem 2

```
%%

clear, close all

%% Download ONI Index

clear, close all

%%

url = 'https://psl.noaa.gov/data/correlation/oni.data';

ONI_table = readtable(url, 'HeaderLines', 0, 'ReadVariableNames', true, 'FileType', 'text');

ONI_table = ONI_table(1:74, :);

%%

index = 1;

year = [];

%%
for i = 1:74
```

```
    %loop per time
    year_num = ONI_table.x1950(i);

    year = [year, repmat(year_num, 1, 12)];

    for b = 2: 13

        ONI(index) = ONI_table{i, [b]};

        month(index) = b-1;

        dayssince(index) = daysact(datetime(year_num,b-1,1));

        index = index + 1;

    end

end

%%

day = ones(888,1);

dayssince = dayssince';

month = month';

year = year';

ONI = ONI';

time = table(day, month, year, dayssince, 'VariableNames', {'day', 'month', 'year', 'Days Since
Beginning'});
```

```
clear year month dayssince day url year_num ONI_table i b index

%%

%di = μ + (ti − t)m.

time2 = 1:888;

time2 = time2';

t = ONI; % ONI data



%%

trend_model = @(params, t) params(1) + (t - mean(t)) * params(2);

% Initial guess for parameters
initial_guess = [0, 0]; % [mu, m]

% Fit the trend model using lsqcurvefit
param = lsqcurvefit(@(params, t) trend_model(params, t), initial_guess, time2, t);

% Extract parameters
mu = param(1);
m = param(2);

% Calculate the trend line
projected_trend = trend_model(param, time2);

%%
time3 = time.("Days Since Beginning");
```

```
figure(1)

plot(time3, t, 'k', 'LineWidth', 2);
hold on
plot(time3,projected_trend, 'r', 'LineWidth', 2);

xlim([datenum('January 1, 1950') datenum('dec 25, 2020')])
datetick('x','keeplimits')
set(gca, 'FontSize', 16);
xlabel('Time');
ylabel('ONI Index');
title('ONI Data with Projected Trend Model');
legend('ONI Data', 'Projected Trend Model');
grid on;
hold off;

%%


% Calculate residuals
est = ONI - projected_trend; %actual data minus trend

std_est = std(est);

%% Printing

% Output
disp(['Estimated value of mu: ' num2str(mu)]);
disp(['Estimated value of m: ' num2str(m)]);
disp(['Standard deviation of the components of the e vector: ' num2str(std_est)]);

%%
```



Problem 3

```
%%

clear, close all

%% drafting

% 16 different combination of co2 and h2o

%d = [d1; d2; d3; d4; d5; d6; d7; d8; d9; d10; d11; d12; d13; d14; d15; d16]; %data

% A matrix with all possible combinations
A = ones(16, 1);

CO2_labels = [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]; % CO2
H2O_labels = [0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1]; % H2O
```

```
%model - di = µ + β1ΔCO2,i + β2ΔH2O,i + β3ΔCO2,iΔH2O,i + ei (error unknown)
A = [A, CO2_labels.', H2O_labels.', CO2_labels.' .* H2O_labels.'];

%B = [mu; B1; B2; B3];



% Calculate A^TA
ATA = A.' * A;


% Calculate A^Td
%ATd = A.' * d;


%d= AB


%%
```

```
A =

     1     0     0     0
     1     1     0     0
     1     0     1     0
     1     1     1     1
     1     0     0     0
     1     1     0     0
     1     0     1     0
     1     1     1     1
     1     0     0     0
     1     1     0     0
     1     0     1     0
     1     1     1     1
     1     0     0     0
     1     1     0     0
     1     0     1     0
     1     1     1     1

ATA =

    16     8     8     4
     8     8     4     4
```

8	4	8	4
4	4	4	4