Sam Kramer
ENERGY 294 Homework 2
4/26/18

Problem 1.1)

C Rate	Q at T=25˚C (Ah)	Q at T=45˚C (Ah)
0.05	2.22	2.20
1	1.97	2.10
2	1.90	2.05
5	1.84	1.98
15	1.26	1.76

Problem 1.3) Observations

For this problem, I utilized the `fit()` function in MATLAB, using the `power1` fit type. This function found the best-fit curve of the form $Q = k * I^n$, where Q is the capacity in Amp-hours and I is the current in Amps. However, the desired form of the equation is $Q = k * I^{1-n}$, so the values reported below for the parameter n were obtained by multiplying the value obtained from MATLAB by -1 and adding 1.

Peukert Equation Coefficients		
	T = 25˚C	T = 45˚C
k	1.968	2.088
n	1.070	1.033
RMSE	0.216	0.087

The values of k are close to the nominal capacity of the battery, which implies that the nominal capacity of the battery is based on a C-rate which corresponds to roughly a 1A constant discharge current. The value of n for the 25˚C trials was higher than that of the 45˚C trials, which suggests that the effects modeled by the Peukert equation are not as strong at higher temperatures – at higher temperatures, battery capacity is diminished less by a higher discharge current. The RMSE for each of these curve fits seemed relatively high. Upon further examination, it seemed that the 0.05 C-rate trials yielded results that did not quite match the Peukert equation, so the fit was performed again while leaving out the 0.05 C-rate trials. The results can be seen below.

Peukert Equation Coefficients – Excluding 0.05C trial		
	T = 25˚C	T = 45˚C
k	2.02	2.11
n	1.041	1.021
RMSE	0.005	0.023

Problem 1.5) Observations

These plots make it apparent that the data from the 25˚C trial appears to be missing most of the charging cycle. However, the 45˚C trial clearly shows both the constant-current and constant-voltage portions of the charging cycle. There is also a long period of time (at least 1 hour) between charging and discharging for each battery. The voltage and current profiles are similar for each trial over time, though the discharge for the 45˚C trial had a slightly longer duration due to the increased capacity of the battery with temperature. The voltage fell quite quickly in both trials due to the high current needed to discharge the battery in 12 minutes.

Problem 1.6)

Coulometric Efficiencies		
C Rate	T=25˚C	T=45˚C
1	43.6%	55.4%
2	35.2%	43.6%
15	0.0%	0.0%

Energy Efficiencies		
C Rate	T=25˚C	T=45˚C
1	43.1%	55.0%
2	34.5%	43.1%
15	0.0%	0.0%

The first thing that struck me about these results were that the efficiencies were all incredibly low. This is likely because most of the charging occurs at or above 3.65V, while a significant amount of the battery capacity cannot be discharged without the voltage dropping below 3.65V. The efficiencies decreased as the C-rate increased. This can be understood by considering a simple equivalent circuit model for the battery with a voltage source and series resistor – the higher the discharge current, the higher the resistive losses and the greater the voltage drop across the resistor, which results in a lower voltage at the terminals. This is why the 15C discharge has efficiencies of 0 - the battery cannot be discharged at a current this high without the voltage dropping below 3.65V. The efficiencies were higher for the higher temperature trials, which is interesting.

Table of Contents


```
clearvars
close all
clc
```

Read in Data for Problem 1

```
% Create variables for file name components
path = 'Experimental Data Sets/NMC_Cell_M2_';
temp_str = string({'T25_', 'T45_'});
temp_val = [25, 45];
Crate_str = string({'0_05C_', '1C_', '2C_', '5C_', '15C_'});
Crate_val = [0.05, 1, 2, 5, 15];
N_MAX_OBS = 94695;

% Initialize arrays for results
times = NaN(length(temp_val), length(Crate_val), N_MAX_OBS);
currents = NaN(length(temp_val), length(Crate_val), N_MAX_OBS);
voltages = NaN(length(temp_val), length(Crate_val), N_MAX_OBS);
I_discharge = NaN(length(temp_val), length(Crate_val));

% Iterate through files
for i = 1:length(temp_str)
    for j = 1:length(Crate_str)

            % Read data file
            name = strcat(path, temp_str(i),
 Crate_str(j), 'CTID.xlsx');
            curr = xlsread(char(name));
            % Read second sheet of file if it is for the 0.05 C-rate
            if j==1
                curr2 = xlsread(char(name), 2);
                curr = [curr; curr2];
            end
```

```
            % Extract the desired series
            times(i,j,1:length(curr)) = curr(:,2);
            currents(i,j,1:length(curr)) = curr(:,3);
            voltages(i,j,1:length(curr)) = curr(:,4);

    end
end
```

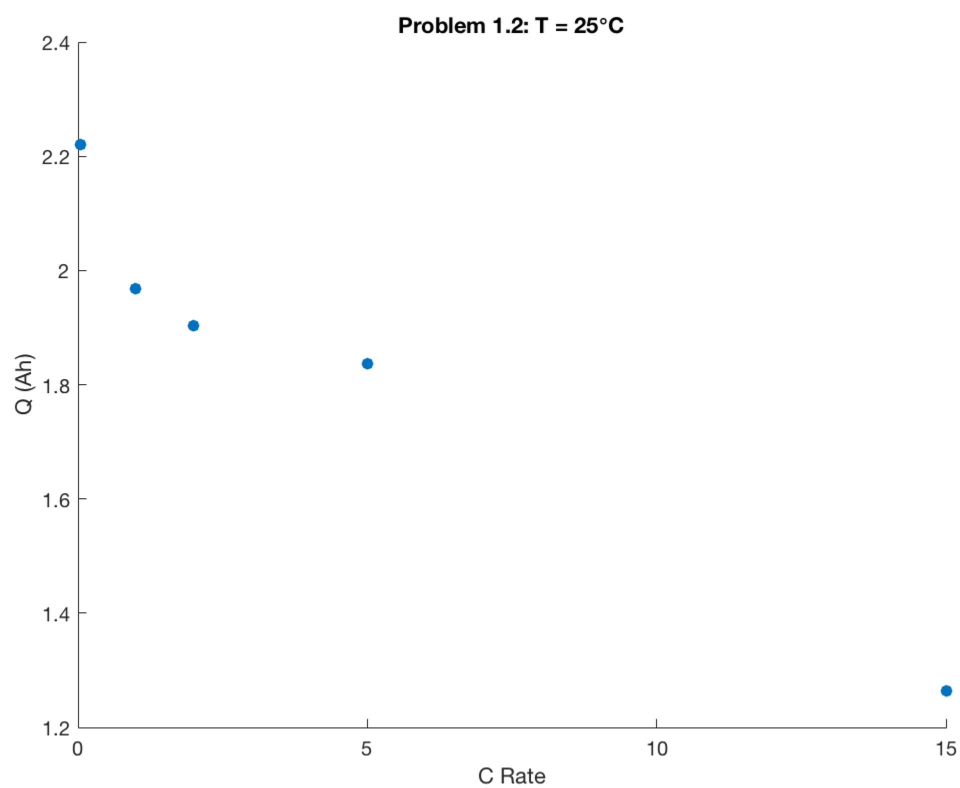
1.1) Calculate capacity

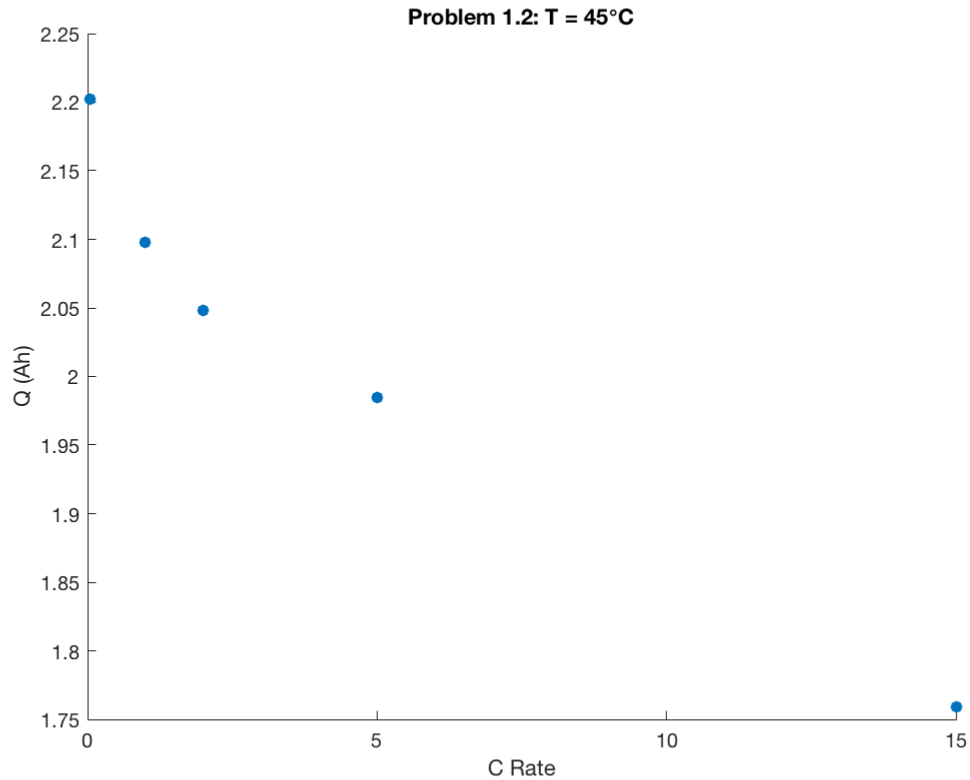
```
% Initialize array for results
capacities = zeros(length(temp_val), length(Crate_val));

% Iterate through the trials
for i = 1:length(temp_str)
    for j = 1:length(Crate_str)

        % Pull out the current time and current vectors
        t = times(i,j,:);
        I = currents(i,j,:);

        % Find discharge data by finding the indexes of negative
 current
        t = squeeze(t(I<0));
        I = squeeze(I(I<0));

        % Store discharge current for each C rate (for Peukert
 fitting)
        I_discharge(i,j) = -1 * mean(I);

        % Calculate battery capacity
        capacities(i,j) = -1 * trapz(t, I) / 3600;

    end
end
```

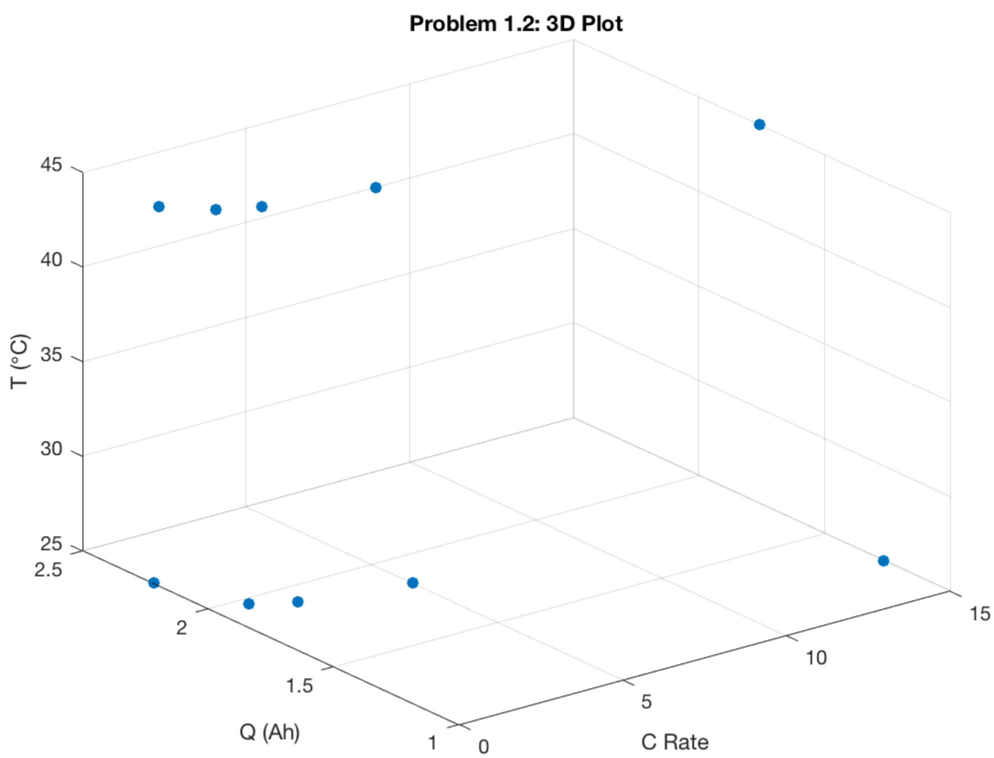
1.2) Plots

```
% 2D plot for 25C
figure
hold on
scatter(Crate_val, capacities(1,:), 'filled')
xlabel('C Rate')
ylabel('Q (Ah)')
title('Problem 1.2: T = 25\circC')

% 2D Plot for 45C
figure
hold on
scatter(Crate_val, capacities(2,:), 'filled')
xlabel('C Rate')
```

```
ylabel('Q (Ah)')
title('Problem 1.2: T = 45\circC')

% Concatenate arrays into 1-row vectors for 3D plotting
capacity3 = [capacities(1,:), capacities(2,:)];
Crate3 = repmat(Crate_val, 1, 2);
temp3 = [repmat(25, 1, 5), repmat(45, 1, 5)];

% Make 3D Plot
figure
scatter3(Crate3, capacity3, temp3, 'filled')
xlabel('C Rate')
ylabel('Q (Ah)')
zlabel('T (\circC)')
title('Problem 1.2: 3D Plot')
```




Problem 1.2: T = 45°C
Q (Ah)
C Rate
2.25
2.2
2.15
2.1
2.05
2
1.95
1.9
1.85
1.8
1.75
0
5
10
15


Problem 1.2: 3D Plot
T (°C)
Q (Ah)
C Rate
45
40
35
30
25
2.5
2
1.5
1
0
5
10
15

1.3) Fitting Peukert Equation

```
% Calculate Peukert Equation fit for each temperature
[fit1, out1] =
 fit(I_discharge(1,:)',capacities(1,:)','power1'); %T=25C
[fit2, out2] =
 fit(I_discharge(2,:)',capacities(2,:)','power1'); %T=45C

fit1
out1.rmse
fit2
out2.rmse

% Calculate Peukert Equation fit - ignore 0.05C trial
[fit1, out1] = fit(I_discharge(1,:)',capacities(1,:)','power1',...
                    'Exclude', 5);
[fit2, out2] = fit(I_discharge(2,:)',capacities(2,:)','power1',...
                    'Exclude', 5);

fit1
out1.rmse
fit2
out2.rmse
```

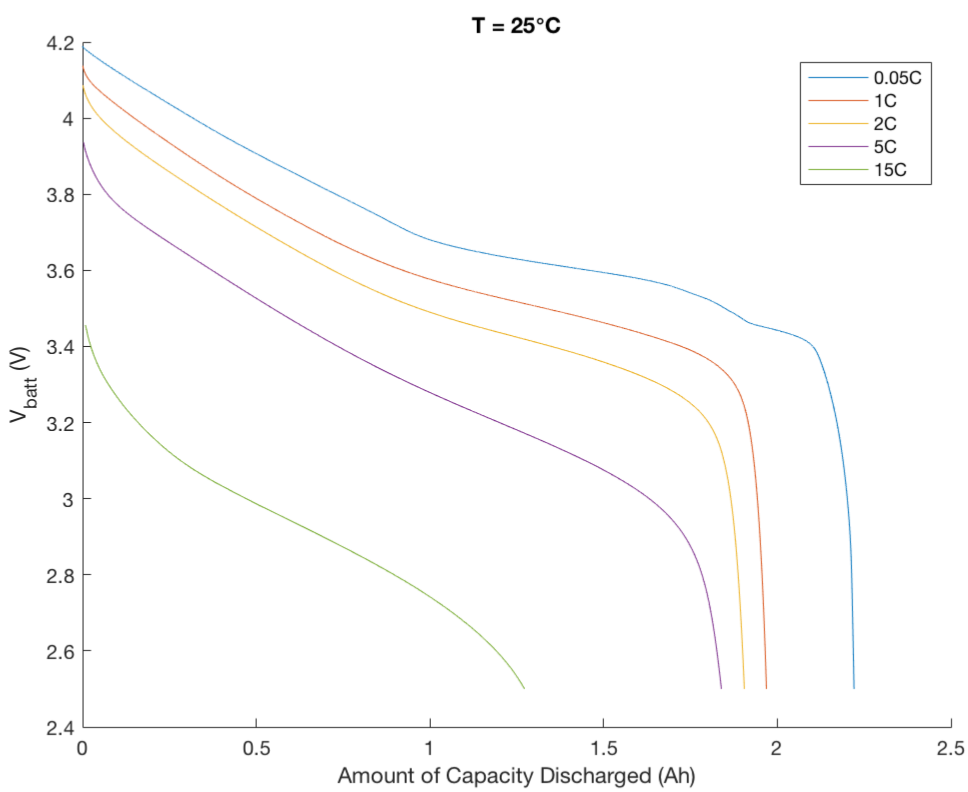
```
fit1 =

     General model Power1:
     fit1(x) = a*x^b
     Coefficients (with 95% confidence bounds):
       a =       1.968  (1.633, 2.303)
       b =    -0.06993  (-0.1497, 0.009826)

ans =

    0.2155


fit2 =

     General model Power1:
     fit2(x) = a*x^b
     Coefficients (with 95% confidence bounds):
       a =       2.088  (1.949, 2.226)
       b =    -0.03268  (-0.06318, -0.002182)

ans =

    0.0869


fit1 =
```

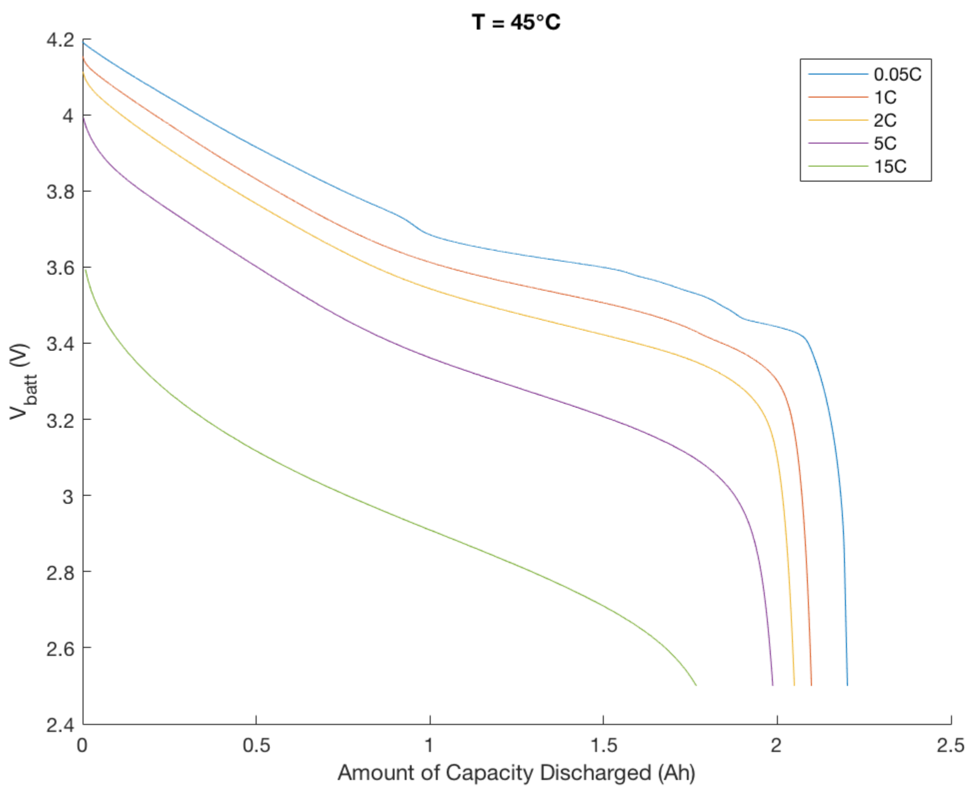
```
     General model Power1:
     fit1(x) = a*x^b
     Coefficients (with 95% confidence bounds):
       a =        2.02  (2.01, 2.03)
       b =    -0.04135  (-0.04404, -0.03865)

ans =

    0.0045


fit2 =

     General model Power1:
     fit2(x) = a*x^b
     Coefficients (with 95% confidence bounds):
       a =       2.105  (2.054, 2.156)
       b =    -0.02129  (-0.03465, -0.007941)

ans =

    0.0229
```

1.4a) Plotting OCV Curves

```
% Nominal capacity (Ah)
Qnom = 2.1;

% Iterate through files
for i = 1:length(temp_str)

    figure
    hold on

    for j = 1:length(Crate_str)

        % Extract time, current, and voltage vectors
        t = squeeze(times(i,j,:));
        I = squeeze(currents(i,j,:));
        V = squeeze(voltages(i,j,:));

        % Calculate time duration of each measurement
        t_lag = t((find(I<0,1)-1):(find(I<0,1,'last')-1));
        delta_t = t(I<0) - t_lag;

        % Calculate capacity decrease for each timestep
        delta_Q = I(I<0) .* delta_t / 3600;

        % Calculate capacity vector
        Q = Qnom + cumsum(delta_Q);

        % Plot data
```

```
        plot(Qnom - Q, V(I<0))

    end

    % Add axis labels and legend
    xlabel('Amount of Capacity Discharged (Ah)')
    ylabel('V_{batt} (V)')
    legend('0.05C', '1C', '2C', '5C', '15C')
    if i == 1
        title('T = 25\circC')
    else
        title('T = 45\circC')
    end

end
```

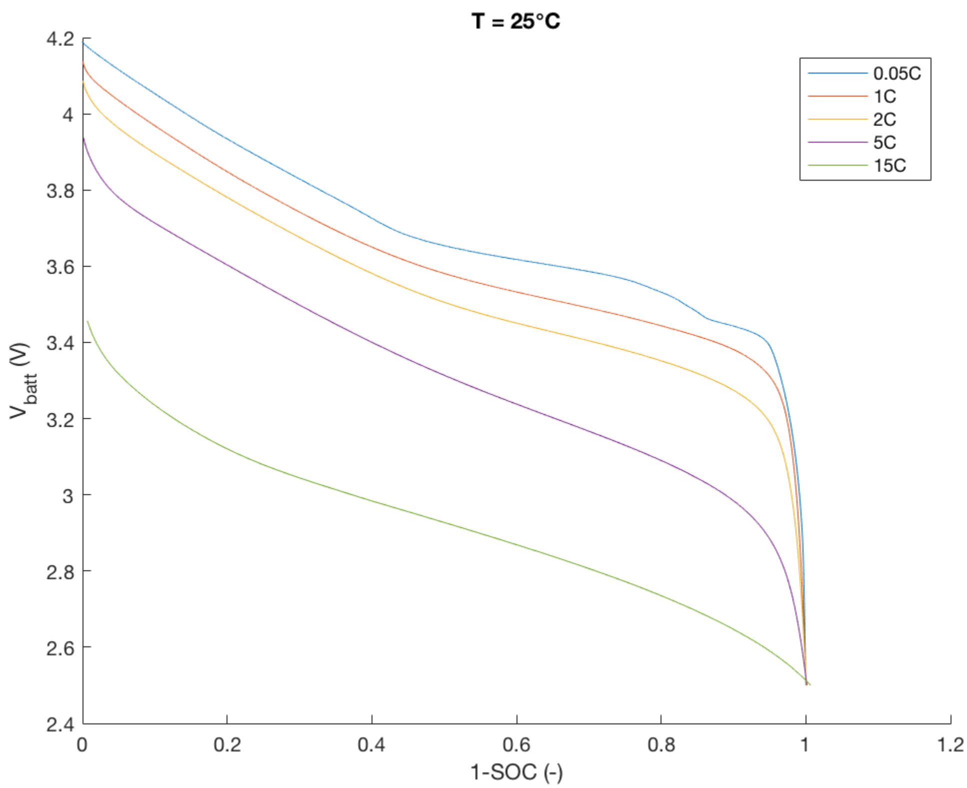


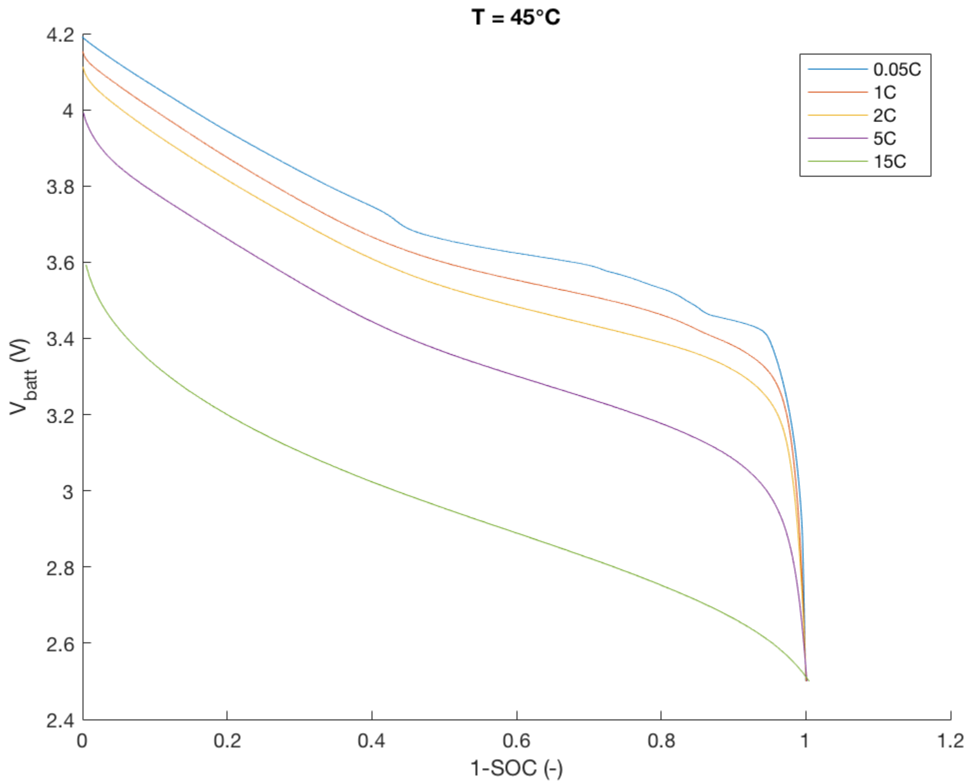


1.4b) Plotting OCV Curves - Normalized Capacity

```
% Iterate through files
for i = 1:length(temp_str)

    figure
    hold on

    for j = 1:length(Crate_str)

        % Extract time, current, and voltage vectors
        t = squeeze(times(i,j,:));
        I = squeeze(currents(i,j,:));
        V = squeeze(voltages(i,j,:));

        % Calculate time duration of each measurement
        t_lag = t((find(I<0,1)-1):(find(I<0,1,'last')-1));
        delta_t = t(I<0) - t_lag;

        % Calculate capacity decrease for each timestep
        delta_Q = I(I<0) .* delta_t / 3600;

        % Get full capacity
```

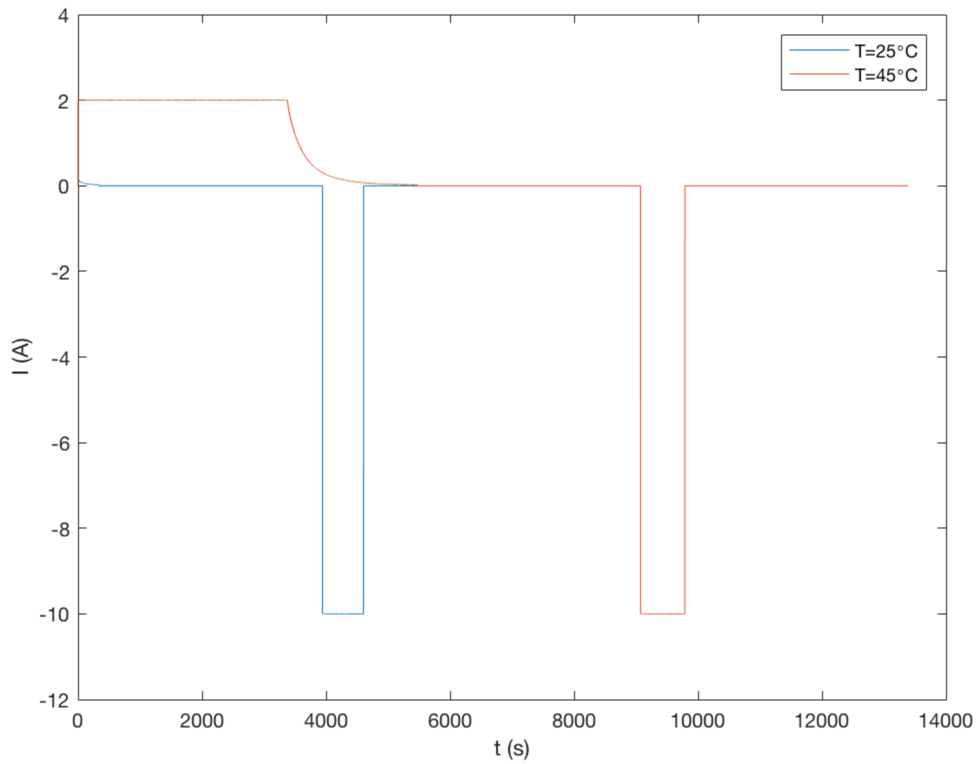
```
        Qnom = capacities(i,j);

        % Calculate capacity vector
        Q = Qnom + cumsum(delta_Q);

        % Plot data
        plot((Qnom - Q)/Qnom, V(I<0))

    end

    % Add axis labels and legend
    xlabel('1-SOC (-)')
    ylabel('V_{batt} (V)')
    legend('0.05C', '1C', '2C', '5C', '15C')
    if i == 1
        title('T = 25\circC')
    else
        title('T = 45\circC')
    end

end
```



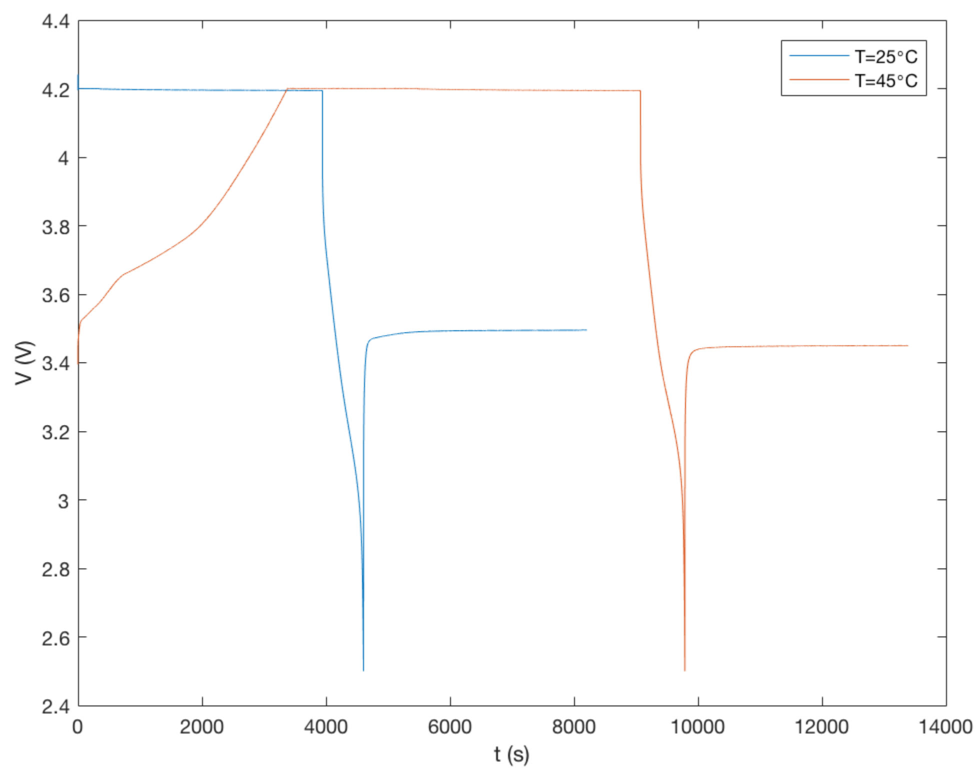


1.5) Time Series Plots

```
% Find discharge times, currents and voltages
I1 = squeeze(currents(1,4,:));
I2 = squeeze(currents(2,4,:));
t1 = squeeze(times(1,4,:));
t2 = squeeze(times(2,4,:));
V1 = squeeze(voltages(1,4,:));
V2 = squeeze(voltages(2,4,:));

figure
plot(t1, I1, t2, I2);
xlabel('t (s)')
ylabel('I (A)')
legend('T=25\circC', 'T=45\circC')

figure
plot(t1, V1, t2, V2);
xlabel('t (s)')
ylabel('V (V)')
legend('T=25\circC', 'T=45\circC')
```



T=25°C
T=45°C
I (A)
t (s)



T=25°C
T=45°C
V (V)
t (s)

1.6) Calculate efficiencies

```
Q_charge = zeros(2,3);
Q_dis = zeros(2,3);
E_charge = zeros(2,3);
E_dis = zeros(2,3);
eta_coul = zeros(2, 3);
eta_energy = zeros(2, 3);

Crates_use = [2 3 5];

for i = 1:2
    for j = 1:length(Crates_use)

        j2 = Crates_use(j);

        t = squeeze(times(i,j2,:));
        I = squeeze(currents(i,j2,:));
        V = squeeze(voltages(i,j2,:));

        charge_indices = I>0 & V>=3.65 & V<=4.20;
        dis_indices = I<0 & V>=3.65 & V<=4.20;

        t_charge = squeeze(times(i,j2, charge_indices));
        I_charge = squeeze(currents(i,j2, charge_indices));
        V_charge = squeeze(voltages(i,j2, charge_indices));

        t_dis = squeeze(times(i,j2, dis_indices));
        I_dis = squeeze(currents(i,j2, dis_indices));
        V_dis = squeeze(voltages(i,j2, dis_indices));

        Q_charge(i,j) = trapz(t_charge, I_charge);
        Q_dis(i,j) = trapz(t_dis, I_dis);
        eta_coul(i,j) = -1 * Q_dis(i,j) / Q_charge(i,j);

        E_charge(i,j) = trapz(t_charge, I_charge .* V_charge);
        E_dis(i,j) = trapz(t_dis, I_dis .* V_dis);
        eta_energy(i,j) = -1 * E_dis(i,j) / E_charge(i,j);

    end
end

eta_coul
eta_energy


eta_coul =

    0.4362    0.3520         0
    0.5537    0.4356         0


eta_energy =
```

```
    0.4310    0.3452         0
    0.5496    0.4310         0
```

Read in Data for Problem 2

```
clearvars
close all
clc

% Variables for file name components
path = 'Experimental Data Sets/NMC_Cell_H1_';
temp_str = string({'T05_', 'T23_', 'T40_', 'T45_', 'T52_'});
temp_val = [5, 23, 40, 45, 52]';
N_MAX_OBS = 30931;

% Initialize arrays for results
times = NaN(length(temp_val), N_MAX_OBS);
currents = NaN(length(temp_val), N_MAX_OBS);
voltages = NaN(length(temp_val), N_MAX_OBS);

% Iterate through files
for i = 1:length(temp_str)

    % Read data file
    name = strcat(path, temp_str(i), '1C_CTID.xlsx');
    curr = xlsread(char(name));

    % Extract series
    times(i, 1:length(curr)) = curr(:,2);
    currents(i, 1:length(curr)) = curr(:,3);
    voltages(i, 1:length(curr)) = curr(:,4);

end
```

2.2a) Calculate Capacity

```
capacities = zeros(length(temp_val),1);
I_discharge = zeros(length(temp_val),1);

for i = 1:length(temp_str)

    t = times(i,:);
    I = currents(i,:);

    % Find discharge data by negative current data points
    t = squeeze(t(I<0));
    I = squeeze(I(I<0));

    % Calculate battery capacity
    capacities(i) = -1 * trapz(t, I) / 3600;
```

```
end

capacities
```

```
capacities =

    1.7423
    1.9561
    2.0628
    2.1337
    2.1559
```

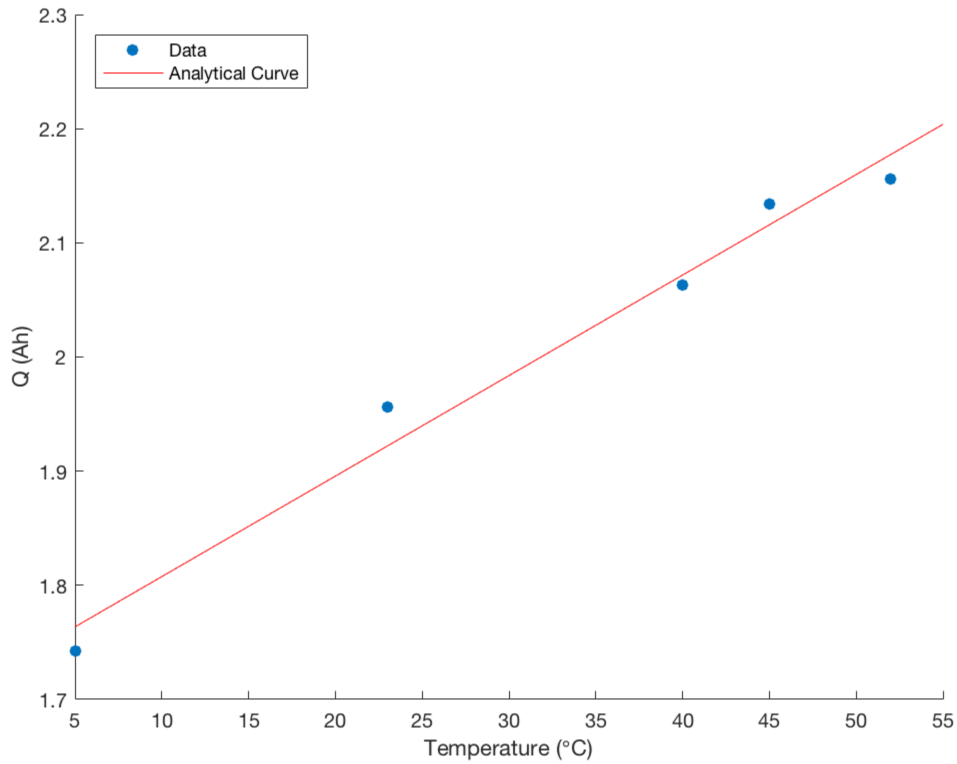
2.2a) Analytical Expression

```
fit1 = fit(temp_val, capacities, 'poly1')
```

```
fit1 =

     Linear model Poly1:
     fit1(x) = p1*x + p2
     Coefficients (with 95% confidence bounds):
       p1 =    0.008805  (0.006395, 0.01122)
       p2 =        1.72  (1.63, 1.809)
```

2.2b) Plot Results

```
figure
hold on
scatter(temp_val, capacities, 'filled')
plot(fit1)
legend('Data', 'Analytical Curve','Location', 'Northwest')
xlabel('Temperature (\circC)')
ylabel('Q (Ah)')
```



Published with MATLAB® R2016b