

**Advanced Energy Storage Systems**

**MECH8290-60-R-2024F**

**Group 41**

**“Project - 01”**

**Submitted by**

Vijayarangan Pandurengadurai Raju 110128279

Jaiharini Mani 110131837

Bharath Suresh 110128856

Mageshwari Palanisamy 110142170

**Submitted to:**

Dr. Balakumar Balasingam

**Submitted on:**

13th November 2024

Describing the data

load('C1209.mat');

h=figure;

subplot(211);plot(Time,V, 'LineWidth',2); grid on

xlabel('Time(hr)');

ylabel('Meas. Voltage(V)');

subplot(212);plot(Time,I, 'LineWidth',2); grid on

xlabel('Time(hr)');

ylabel('Meas. Current(A)');

sgtitle('Batt:C1209');

A graph of a graph of a graph

Description automatically generated with medium confidence



Matlab Code

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Identify the indices where the current I is positive (charging) and negative (discharging)

charging\_indices = find(I > 0);

discharging\_indices = find(I < 0);

% Calculate the total charging and discharging time

charging\_time = Time(charging\_indices(end)) - Time(charging\_indices(1));

discharging\_time = Time(discharging\_indices(end)) - Time(discharging\_indices(1));

% Compute the average absolute current during charging and discharging

average\_charging\_current = mean(abs(I(charging\_indices(1):charging\_indices(end))));

average\_discharging\_current = mean(abs(I(discharging\_indices(1):discharging\_indices(end))));

% Calculate the total charge for charging and discharging phases

Qc = charging\_time \* average\_charging\_current;

Qd = discharging\_time \* average\_discharging\_current;

% Display results

disp('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

fprintf('Charge Capacity (Qc): %.4f Ah\n', Qc);

fprintf('Discharge Capacity (Qd): %.4f Ah\n', Qd);

Output

A black text on a white background

Description automatically generated

A math equations on a white background

Description automatically generated

Matlab Code

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SOC = zeros(length(I), 1);

SOC(1) = 1;

for i = 2:length(I)

delta = Time(i) - Time(i-1);

if(I(i))<0

SOC(i) = SOC(i-1) + (delta \* I(i))/Qd;

else

SOC(i) = SOC(i-1) + (delta \* I(i))/Qc;

end

if (SOC(i) < 0)

SOC(i) = 0;

elseif(SOC(i) > 1)

SOC(i) = 1;

end

end

% Plot Question 2 SOC vs. Time

figure;

plot(Time, SOC, 'LineWidth', 2);

xlabel('Time (hr)');

ylabel('State of Charge (SOC)');

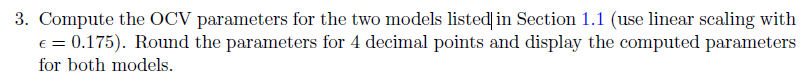
title('Question 2 SOC vs. Time');

grid on;

Output

A graph with a blue line

Description automatically generated



Matlab Code

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Define epsilon for scaling

epsilon = 0.175;

s\_scaled = epsilon + (1 - 2 \* epsilon) \* SOC; % Apply linear scaling

% Define the matrix P for both models

% Combined model

P\_combined = [ones(size(s\_scaled)), 1 ./ s\_scaled, s\_scaled, log(s\_scaled), log(1 - s\_scaled), I];

% Combined+3 model

P\_combined3 = [ones(size(s\_scaled)), 1 ./ s\_scaled, 1 ./ (s\_scaled.^2), 1 ./ (s\_scaled.^3), ...

1 ./ (s\_scaled.^4), s\_scaled, log(s\_scaled), log(1 - s\_scaled), I];

% Calculate the least squares solution for both models

% Combined model

c\_combined = (P\_combined' \* P\_combined) \ (P\_combined' \* V);

c\_combined = round(c\_combined, 4); % Round to 4 decimal places

% Combined+3 model

c\_combined3 = (P\_combined3' \* P\_combined3) \ (P\_combined3' \* V);

c\_combined3 = round(c\_combined3, 4); % Round to 4 decimal places

disp('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

% Display the parameters

disp('Combined model parameters:');

disp(c\_combined);

disp('Combined+3 model parameters:');

disp(c\_combined3);

Output

A screenshot of a computer

Description automatically generated



Matlab Code

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 4\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Number of data points

N = length(V);

% Model parameters count

M\_combined = length(c\_combined);

M\_combined3 = length(c\_combined3);

% Predicted voltage for Combined model

V\_combined\_hat = P\_combined \* c\_combined;

% Calculate Error Metrics for Combined model

V\_mean = mean(V);

R2\_combined = (1 - norm(V - V\_combined\_hat)^2 / norm(V - V\_mean)^2) \* 100;

MaxError\_combined = max(abs(V - V\_combined\_hat));

RMS\_combined = norm(V - V\_combined\_hat) / sqrt(N - M\_combined);

% Display Results for Combined model

disp("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 4\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

fprintf('--- Combined Model ---\n');

fprintf('R² Fit: %.2f%%\n', R2\_combined);

fprintf('Max Error: %.4f V\n', MaxError\_combined);

fprintf('Root-Mean Square Error (RMS): %.4f V\n', RMS\_combined);

% Predicted voltage for Combined+3 model

V\_combined3\_hat = P\_combined3 \* c\_combined3;

% Calculate Error Metrics for Combined+3 model

R2\_combined3 = (1 - norm(V - V\_combined3\_hat)^2 / norm(V - V\_mean)^2) \* 100;

MaxError\_combined3 = max(abs(V - V\_combined3\_hat));

RMS\_combined3 = norm(V - V\_combined3\_hat) / sqrt(N - M\_combined3);

% Display Results for Combined+3 model

fprintf('\n--- Combined+3 Model ---\n');

fprintf('R² Fit: %.2f%%\n', R2\_combined3);

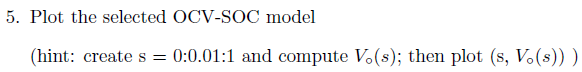
fprintf('Max Error: %.4f V\n', MaxError\_combined3);

fprintf('Root-Mean Square Error (RMS): %.4f V\n', RMS\_combined3);

Output

A white paper with black text

Description automatically generated



Matlab Code

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 5\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Define SOC range from 0 to 1 with a step of 0.01

s\_raw = (0:0.01:1);

s = epsilon + (1 - 2 \* epsilon) \* s\_raw;

k = c\_combined3;

% Combined+3 Model Calculation using the array values

V\_combined3 = k(1) + k(2)./s + k(3)./s.^2 + k(4)./s.^3 + k(5)./s.^4 + ...

k(6) \* s + k(7) \* log(s) + k(8) \* log(1 - s);

% Plot the Combined+3 Model

figure; hold on; box on;

plot(s\_raw, V\_combined3, 'r-', 'LineWidth', 2);

% Add labels and title

xlabel('State of Charge (SOC)');

ylabel('Open Circuit Voltage (V\_{o})');

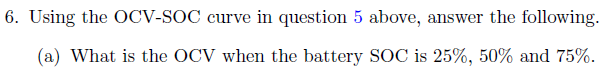
title('Question 5 OCV-SOC Curve for Combined+3 Model');

grid on;

Output

A graph with a red line

Description automatically generated



Matlab Code

fprintf('\n\*\*\*\*\*\*\*\*\*Question 6a\*\*\*\*\*\*\*\*\*\n')

SOC\_levels = [0.25, 0.50, 0.75];

OCV\_values = interp1(s\_raw, V\_combined3, SOC\_levels, 'linear');

fprintf('OCV when the battery SOC is 25%%: %.4f V\n', OCV\_values(1));

fprintf('OCV when the battery SOC is 50%%: %.4f V\n', OCV\_values(2));

fprintf('OCV when the battery SOC is 75%%: %.4f V\n', OCV\_values(3));

Output

A group of black text

Description automatically generated



Matlab Code

fprintf('\*\*\*\*\*\*\*\*\*Question 6b\*\*\*\*\*\*\*\*\*\n')

% Target OCV value

target\_OCV = 4.0;

% Interpolating SOC at 4V

SOC\_at\_4V = interp1(V\_combined3, s\_raw, target\_OCV, 'linear');

fprintf('SOC at 4V (zero current): %.4f\n', SOC\_at\_4V);

Output

A close up of words

Description automatically generated



Matlab Code

fprintf('\*\*\*\*\*\*\*\*\*Question 6c\*\*\*\*\*\*\*\*\*\n')

% Target OCV value with a discharge current of 1A

Vo\_OCV\_3\_8V = 3.8;

Reff = c\_combined3(end);

discharge\_current6c = -1;

target\_OCV\_3\_8V = Vo\_OCV\_3\_8V - discharge\_current6c \* Reff;

% Interpolating SOC at 3.8V

SOC\_at\_3\_8V = interp1(V\_combined3, s\_raw, target\_OCV\_3\_8V, 'linear');

fprintf('SOC at %.4fV with 1A discharge: %.4f\n', target\_OCV\_3\_8V, SOC\_at\_3\_8V);

Output





Matlab Code

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Question 7\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% from question 4

fprintf('\n\*\*\*\*\*\*\*\*\*Question 7a\*\*\*\*\*\*\*\*\*\n')

% Check for duplicates in the SOC array

[SOC\_unique, idx\_unique] = unique(SOC, 'stable'); % Get unique SOC values and their indices

V\_combined3\_hat\_unique = V\_combined3\_hat(idx\_unique); % Corresponding OCV values without duplicates

OCV\_values\_7a = interp1(SOC\_unique, V\_combined3\_hat\_unique, SOC\_levels, 'linear');

fprintf('OCV when the battery SOC is 25%%: %.4f V\n', OCV\_values\_7a(1));

fprintf('OCV when the battery SOC is 50%%: %.4f V\n', OCV\_values\_7a(2));

fprintf('OCV when the battery SOC is 75%%: %.4f V\n', OCV\_values\_7a(3));

fprintf('\*\*\*\*\*\*\*\*\*Question 7b\*\*\*\*\*\*\*\*\*\n')

% Target OCV value

target\_OCV = 4.0;

% Interpolating SOC at 4V

SOC\_at\_4V = interp1(V\_combined3\_hat\_unique, SOC\_unique, target\_OCV, 'linear');

fprintf('SOC at 4V (zero current): %.4f\n', SOC\_at\_4V);

fprintf('\*\*\*\*\*\*\*\*\*Question 7c\*\*\*\*\*\*\*\*\*\n')

% Target OCV value with a discharge current of 1A

Vo\_OCV\_3\_8V = 3.8;

Reff = c\_combined3(end);

discharge\_current6c = -1;

target\_OCV\_3\_8V = Vo\_OCV\_3\_8V - discharge\_current6c \* Reff;

% Interpolating SOC at 3.8V

SOC\_at\_3\_8V = interp1(V\_combined3\_hat\_unique, SOC\_unique, target\_OCV\_3\_8V, 'linear');

fprintf('SOC at %.4fV with 1A discharge: %.4f\n', target\_OCV\_3\_8V, SOC\_at\_3\_8V);

Output

A screenshot of a computer code

Description automatically generated

Differences from Question 4 and Question 5 (using range 0:0.01:1)

Differences:

|  |  |  |  |
| --- | --- | --- | --- |
| Question 7a | | | |
| SOC | OCV (V) from Question 5 | OCV (V) from Question 4 | Differences |
| 25% | 3.5638 | 3.5587 | 0.0051 |
| 50% | 3.7389 | 3.7456 | 0.0067 |
| 75% | 3.9861 | 3.9998 | 0.0137 |
| Question 7b | | | |
| V | SOC from Question 5 | SOC from Question 4 | Differences |
| 4V | 0.7672 | 0.7852 | 0.018 |
| Question 7c | | | |
| V | SOC from question 5 | Soc from Question 4 | Differences |
| 3.9935 | 0.7591 | 0.7839 | 0.0248 |