Battery Analysis using NASA Battery Dataset

Submitted by: Shreerang Kolhe
Date: December 8, 2024
Submission for: ThinkClock Battery Labs, London, United Kingdom

1. Introduction

The **NASA Battery Dataset** consists of data related to the behavior of Lithium-Ion batteries under various operational profiles. These profiles include **charge, discharge**, and **impedance** measurements conducted at different temperatures. The dataset contains key parameters like **battery impedance**, **electrolyte resistance (Re)**, and **charge transfer resistance (Rct)**, which change as the batteries age during charge/discharge cycles.

The aim of this project is to analyze the changes in battery impedance and resistance parameters (**Re** and **Rct**) over aging cycles to understand how they evolve as the battery undergoes usage. The analysis is visualized using **Plotly**, a powerful plotting library in Python.

2. Methodology

2.1 Data Preprocessing

The analysis starts with loading and exploring the dataset. The dataset consists of a large number of measurements from different Li-ion batteries. Each measurement contains details about the battery's impedance, electrolyte resistance, charge transfer resistance, and other parameters. We focus on three specific parameters for this analysis:

- **Battery Impedance** (Battery_impedance)
- **Electrolyte Resistance** (Re)
- **Charge Transfer Resistance** (Rct)

We perform the following steps to prepare the data for analysis:

1. **Data Loading**: Load the dataset from a CSV file using pandas.
2. **Data Cleaning**: Remove any rows containing missing values for the relevant columns: Battery_impedance, Re, and Rct.
3. **Data Filtering**: Select only the columns necessary for this analysis, specifically cycle, Battery_impedance, Re, and Rct.

2.2 Data Analysis

The data is visualized in **scatter plots** to show the relationship between the aging cycle and the three battery parameters. This allows us to observe how each parameter evolves over time.

2.3 Visualizations

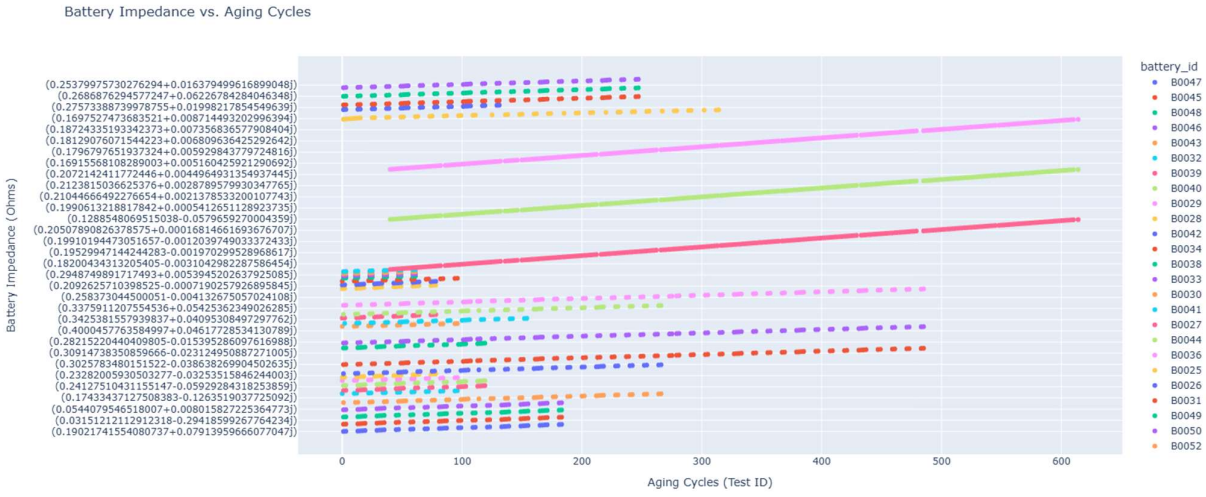
We use **Plotly** to create the following plots:

- **Scatter plot** for Battery_impedance vs. cycle (showing the change in impedance with aging cycles).
- **Scatter plot** for Re vs. cycle (showing the change in electrolyte resistance).
- **Scatter plot** for Rct vs. cycle (showing the change in charge transfer resistance).

3. Analysis & Results

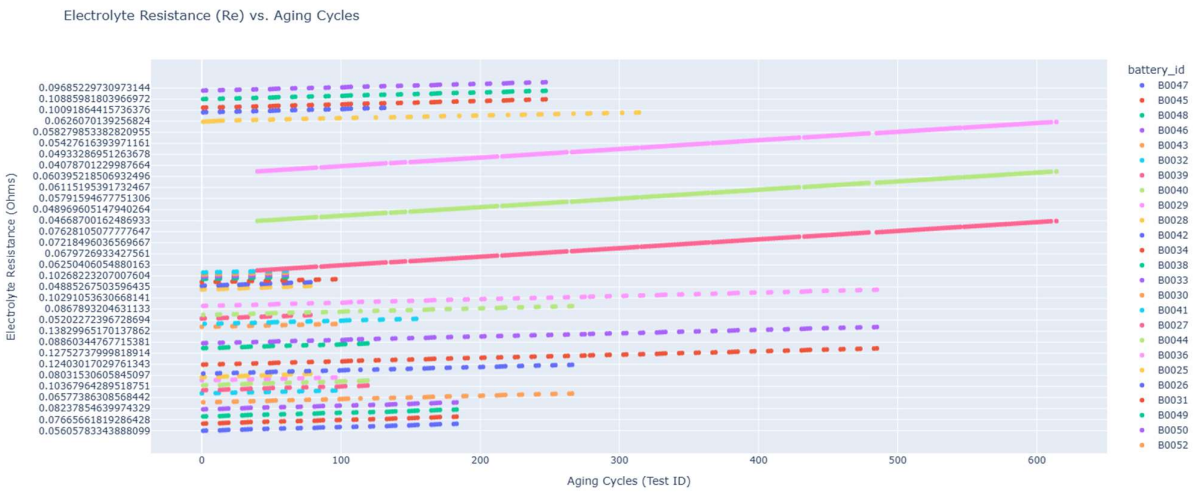
3.1 Scatter Plot 1: Battery Impedance vs. Aging Cycle

- **Description**: This plot shows how Battery_impedance changes as the battery goes through charge/discharge cycles.
- **Observation**: We observe an increase in the impedance as the battery ages, which indicates the degradation of the battery's internal components.



3.2 Scatter Plot 2: Electrolyte Resistance (Re) vs. Aging Cycle

- **Description**: This plot shows how Re (electrolyte resistance) changes with the battery cycles.
- **Observation**: The resistance increases over time, which could indicate aging or degradation of the electrolyte.



3.3 Scatter Plot 3: Charge Transfer Resistance (Rct) vs. Aging Cycle

- **Description**: This plot demonstrates how Rct (charge transfer resistance) increases as the battery undergoes charge/discharge cycles.
- **Observation**: We notice an increase in charge transfer resistance with aging, which is typically associated with internal battery degradation.

4. Conclusion

The analysis of the **NASA Battery Dataset** reveals that key battery parameters, such as **battery impedance**, **electrolyte resistance (Re)**, and **charge transfer resistance (Rct)**, increase over time as the battery undergoes aging cycles. These results suggest that the internal components of the battery degrade with repeated usage. This type of analysis can be useful in predicting the **end-of-life (EOL)** of batteries and helps improve battery management systems (BMS) by providing insights into the behavior of batteries over their lifespan.

5. Code

The code for the analysis is provided as a **Python script** (main.py). The code utilizes **Pandas** for data processing and **Plotly** for visualizations.

```
import os
import pandas as pd
import plotly.express as px

# Define paths
data_folder = r"cleaned_dataset\data"  # Update as per your dataset location
metadata_path = r"cleaned_dataset\metadata.csv"  # Update with the metadata
file path

# Load metadata
metadata = pd.read_csv(metadata_path)

# Filter for "impedance" data type
impedance_data = metadata[metadata['type'] == 'impedance']

# Initialize a list to store data
combined_data = []

# Loop through the CSV files listed in metadata
for _, row in impedance_data.iterrows():
    file_path = os.path.join(data_folder, row['filename'])
    if os.path.exists(file_path):
        # Load the data
        temp_data = pd.read_csv(file_path)
        # Add metadata columns to the data
        temp_data['test_id'] = row['test_id']
        temp_data['battery_id'] = row['battery_id']
        temp_data['ambient_temperature'] = row['ambient_temperature']
        temp_data['Re'] = row['Re']
        temp_data['Rct'] = row['Rct']
        temp_data['Battery_impedance'] = temp_data['Battery_impedance']  #
Ensure battery impedance is included
        combined_data.append(temp_data)

# Combine all data into a single DataFrame
if combined_data:
    combined_df = pd.concat(combined_data, ignore_index=True)
else:
    raise ValueError("No valid data files were loaded.")

# Display sample of the combined data
print("Combined Data Sample:")
print(combined_df.head())
```

```
# Drop rows where `Battery_impedance`, `Re`, or `Rct` is NaN
filtered_data = combined_df.dropna(subset=['Battery_impedance', 'Re', 'Rct'])

# Plot `Battery_impedance` vs. aging cycles
fig_impedance = px.scatter(filtered_data, x='test_id', y='Battery_impedance',
color='battery_id',
                           title='Battery Impedance vs. Aging Cycles',
                           labels={'test_id': 'Aging Cycles (Test ID)',
'Battery_impedance': 'Battery Impedance (Ohms)'})
fig_impedance.show()

# Plot `Re` vs. aging cycles
fig_re = px.scatter(filtered_data, x='test_id', y='Re', color='battery_id',
                    title='Electrolyte Resistance (Re) vs. Aging Cycles',
                    labels={'test_id': 'Aging Cycles (Test ID)', 'Re':
'Electrolyte Resistance (Ohms)'})
fig_re.show()

# Plot `Rct` vs. aging cycles
fig_rct = px.scatter(filtered_data, x='test_id', y='Rct', color='battery_id',
                     title='Charge Transfer Resistance (Rct) vs. Aging
Cycles',
                     labels={'test_id': 'Aging Cycles (Test ID)', 'Rct':
'Charge Transfer Resistance (Ohms)'})
fig_rct.show()
```

6. Environment Setup

The analysis was conducted in a Python virtual environment named **`test`** with the following dependencies:

Steps to Set Up the Environment:

1. **Create a virtual environment:**
   - `python -m venv test`
2. **Activate the `test` Virtual Environment**:
   - On Windows:

     ```
     .\test\Scripts\activate
     ```

   - On macOS/Linux:

     ```
     source test/bin/activate
     ```

3. **Install the Required Dependencies**: After activating the environment, install the necessary Python libraries:

   ```
   pip install -r requirements.txt
   ```

4. **Run the Python Script**: Once the environment is set up and dependencies are installed, you can run the Python script to generate the analysis:

   ```
   python main.py
   ```

7. Submission Files

The following files are included in the submission:

1. **main.py**: Python script for analysis and plotting.
2. **Battery_analysis_report_Shreerang.pdf**: This report document.
3. **requirements.txt**: List of Python dependencies required to run the code.
4. **cleaned_dataset:** Folder containing metadata and datasets.