# Final Project

STAT 420, Summer 2020, Sara Kohtz, Xuenan Mi, Jackie Kang, Bruno Seo

---

## Li-ion Battery Health Degradation Analysis

### Introduction

Li-ion batteries are utilized in numerous day-to-day life applications, including cell phones, computers, and even electric cars. Therefore, it is important to accurate determine what causes li-ion batteries to degrade in health. "Health" can be defined in numerous ways. For this project, health is defined as the level of capacity within the battery. This is an important aspect that differs from the the state of charge (SOC), which is what is traditionally shown on the device (i.e. your phone is at 75% or 50% etc.). However, in reality 75% SOC for a brand new phone means something different than 75% for a five-year-old phone, this is due to capacity degradation. Unfortunately, there is no direct way of measuring capacity after the battery has been installed in a device. So, the goal of this project is find some relation between other available measurements and capacity, and provide a degradation model for capacity within li-ion batteries.

The data file is obtained from the NASA Prognostics Center of Excellence (PCoE) website. The link is provided in this reference [1]. On the site, it is under set number five, which is labelled "Battery Data Set." There are six datasets, each of which contains experiments for roughly four batteries. The experiment entails charging and discharging a 4.2-volt li-ion battery until the capacity degrades by 30%. There is a measurement for temperature and voltage every 15 seconds, and a measurement for resistance and capacity at the end of each cycle. A full cycle is complete when the battery is completely discharged. Essentially, the battery is discharged to a certain point, recharged completely, then discharged again in a continous fashion. For each dataset, one of three ambient temperatures (5 C, 20 C, 40 C) is set for the entire experiment. Currently, the dataset is in a .mat format, but it has been converted to .csv for easy input into R. The overall goal of the model is to provide insight as to what factors contribute to battery degradation, and provide a model that accurately models capacity degradation.

### Methods

The first step is data cleaning, most of the code for data cleaning is in the appendix. The dataset is rather large, so to reduce the size, only the "discharge" data is considered. In addition, since a capacity measurement is only available at the end of every cycle, and every cycle has around 170 datapoints corresponding to other variables (voltage measurements, temperature measurements, etc.), the data is cleaned further by having variables that summarize the cycle. The column names of the cleaned dataset is provided below:

```
colnames(cycle_data_full)
```

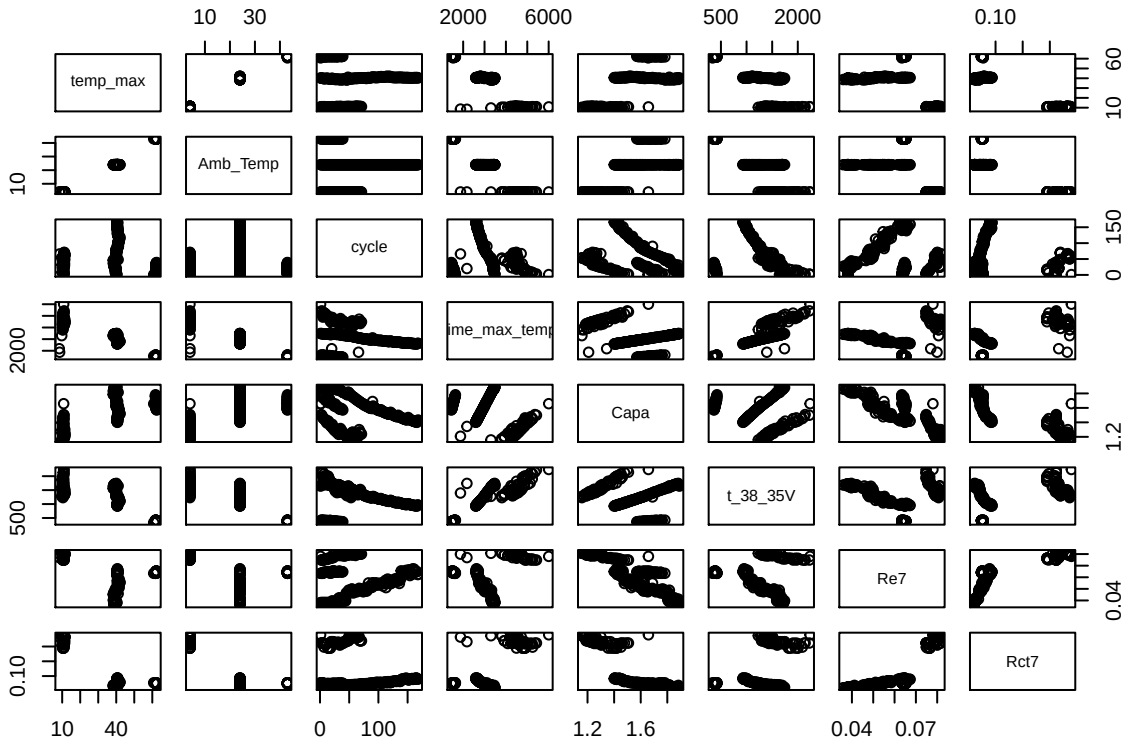
```
## [1] "temp_max"      "Amb_Temp"      "cycle"         "time_max_temp"
## [5] "Capa"          "t_38_35V"      "Re7"           "Rct7"
```

Where `temp_max` is the maximum temperature reached during the discharge cycle, `time_max_temp` is the time it took for the battery to reach that temperature (within the dataset), `Amb_Temp` is the outside temperature at which the experiment was run, it is the categorial variable in this project. An important factor is the time it takes for the battery to decrement from 3.8 V to 3.5 V. It is saved as `t_decrement_38_35V`. `Capa` is the capacity, which for the remainder of the project is the response variable. `Re` is the electrolyte resistance, `Rct` is the transfer resistance, and `cycle` is the cycle number.

Before we fit the model, we want to examine if the any predictor variables are having colinearity issues.If so, we want to identify the model that may not have significant influnce over the response variable to improve the model we select.

To visualize the relationship between the variables and get the correlation between variables

```
pairs(cycle_data_full)
```




```
round(cor(cycle_data_full),2) > 0.80
```

```
##                temp_max Amb_Temp cycle time_max_temp  Capa t_38_35V   Re7  Rct7
## temp_max           TRUE     TRUE FALSE         FALSE FALSE    FALSE FALSE FALSE
## Amb_Temp           TRUE     TRUE FALSE         FALSE FALSE    FALSE FALSE FALSE
## cycle             FALSE    FALSE  TRUE         FALSE FALSE    FALSE FALSE FALSE
## time_max_temp     FALSE    FALSE FALSE          TRUE FALSE     TRUE FALSE FALSE
## Capa              FALSE    FALSE FALSE         FALSE  TRUE    FALSE FALSE FALSE
## t_38_35V          FALSE    FALSE FALSE          TRUE FALSE     TRUE FALSE FALSE
## Re7               FALSE    FALSE FALSE         FALSE FALSE    FALSE  TRUE  TRUE
## Rct7              FALSE    FALSE FALSE         FALSE FALSE    FALSE  TRUE  TRUE
```

Based on the result, there exists collinearity between: `temp_max` & `Amb_Temp`, `t_38_35v` & `time_max_temp`, `Re7` & `Rct7`.

The code below performs partial relation with the variables mentioned above with respect to response variable `Capa`.

```
if_tm1 = lm(Capa ~. - temp_max, data = cycle_data_full)
if_tm2 = lm(temp_max ~. - Capa, data = cycle_data_full)
cor(resid(if_tm1), resid(if_tm2))
```

```
## [1] -0.04196
```

The Partial correlation between `temp_max` and `Capa` is quite small, which suggests that maximum temperature has less of an effect on capacity degradation then the other predictor variables in the model. Some other values of partial correlation coefficeints are shown below.

```
if_at1 = lm(Capa ~. - Amb_Temp, data = cycle_data_full)
if_at2 = lm(Amb_Temp ~. - Capa, data = cycle_data_full)
cor(resid(if_at1), resid(if_at2))
```

```
## [1] 0.3426
```

```
if_t3851 = lm(Capa ~ . - t_38_35V, data = cycle_data_full)
if_t3852 = lm(t_38_35V ~. - Capa, data = cycle_data_full)
cor(resid(if_t3851), resid(if_t3852))
```

```
## [1] 0.7478
```

```
if_tmt1 = lm(Capa ~. - time_max_temp, data = cycle_data_full)
if_tmt2 = lm(time_max_temp ~. - Capa, data = cycle_data_full)
cor(resid(if_tmt1), resid(if_tmt2))
```

```
## [1] 0.1947
```

```
if_re71 = lm(Capa ~ . - Re7, data = cycle_data_full)
if_re72 = lm(Re7 ~ . - Capa , data = cycle_data_full)
cor(resid(if_re71), resid(if_re72))
```

```
## [1] -0.4667
```

```
if_rct71 = lm(Capa ~ . - Rct7, data = cycle_data_full)
if_rct72 = lm(Rct7 ~ . - Capa , data = cycle_data_full)
cor(resid(if_rct71), resid(if_rct72))
```

```
## [1] 0.037
```

The partial correlation between Rct7 and Capa is 0.037, which is also quite small. This suggests that `Rct7` may not be as useful a predictor as others. This makes sense because there is anothe resistance predictor `Re`.

The next step is to create some variables. Since `Amb_Temp` can be defined as a categorical variable, and includes that includes 3 levels: 5, 20, 40, the `as.factor` function is utilized.

```r
cycle_data_full$Amb_Temp = as.factor(cycle_data_full$Amb_Temp)
```

We start with full two-way interaction model, and select model using backwards procedure based on BIC. The simplest model would be best, so utlizing BIC will hopefully provide a simple model due to its penalty function.

```r
mod <- lm(Capa ~ .^2, data = cycle_data_full )
mod_selected = step(mod, direction = 'backward', k = log(nrow(data_full)), trace = 0)
```

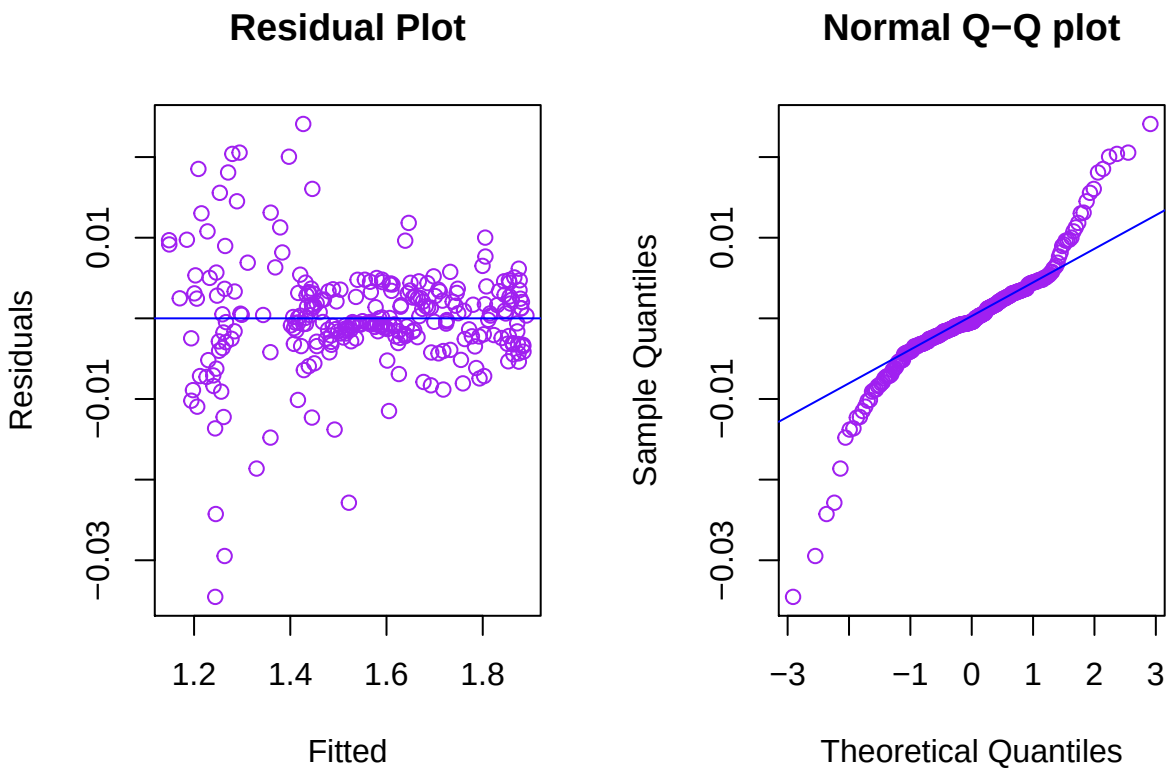
Model diagnostics are performed on this first model `mod_selected`. First, some functions are created to easily test models, and determine "good" models with respect to: leave-one-out-cross-validation root mean square error (LOOCV-RMSE), adjusted r-squared values, and number of parameters (predictor coefficients).

```r
library(lmtest)
library(boot)
#write diagnostic functions for later use
#Fitted VS Residual Test
test_FvsR = function(model){
plot(fitted(model), resid(model), xlab = "Fitted", ylab = "Residuals", main = "Residual Plot", col = "p
abline(h = 0, col = "blue")}
#QQ plot & QQline
test_qq = function(model){
qqnorm(resid(model), main = "Normal Q-Q plot", col = "purple")
qqline(resid(model), col = "blue")
}
#BP test
test_bp = function(model, alpha) {
  p_val = bptest(model)$p.value
  decision = ifelse(p_val < alpha,"Reject", "Fail to Reject")
  list1 = list(bp.p_val = p_val, bp.decision = decision)
  return(list1)
}
# Shapiro-Wilk test (sw test)
test_sw = function(model, alpha) {
  p_val = shapiro.test(resid(model))$p.value
  decision = ifelse(p_val < alpha,"Reject", "Fail to Reject")
  list1 = list(sw.p_val = p_val, sw.decision = decision)
  return(list1)
}
#Calculate loocv_rmse
get_loocv_rmse = function(model) {
  sqrt(mean((resid(model) / (1 - hatvalues(model))) ^ 2))
}
#Calculate adr2
get_adj_r2 = function(model) {
  summary(model)$adj.r.squared
}
#Get number of parameters
get_num_params = function(model) {
  length(coef(model))
}
```

For this initial model, the LOOCV-RMSE is 0.007616, the adjusted r-squared is 0.9989, and the number of parameters is 20. The LOOCV-RMSE is very low, and the adjusted r-squared is very high, however it is

clear that there some assumptions that could be violated from the Breush-Pagan (BP) test and Shapiro-Wilk (SW) test. The table below the graph shows these results. So, adjustments of this model are considered.

```
result_mod_selected = run_all_diagnostic(mod_selected)
```




| loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|
| 0.0076 | 0.9989 | 20 | 0 | Reject | 0 | Reject |

The first step is to address the normality assumption, which can be done by removing the high influence points. The datapoints to keep are found below:

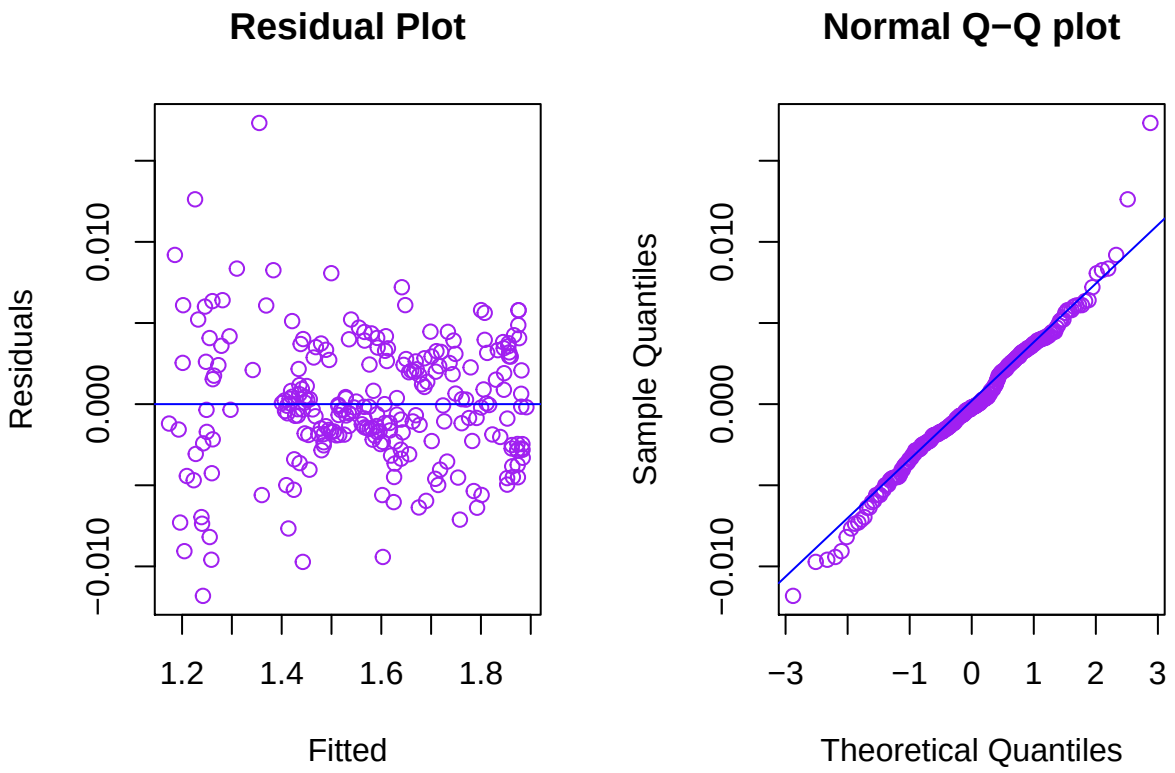
```
keep = cooks.distance(mod_selected) < 4 / length(cooks.distance(mod_selected))
```

Then, this subset is applied to the model determined by BIC.

```
mod_selected_new = lm(Capa ~ temp_max + Amb_Temp + cycle + time_max_temp +
    t_38_35V + Re7 + Rct7 + temp_max:Amb_Temp + temp_max:time_max_temp +
    temp_max:t_38_35V + temp_max:Re7 + Amb_Temp:t_38_35V + cycle:time_max_temp +
    cycle:Re7 + time_max_temp:Re7 + t_38_35V:Rct7, data = cycle_data_full, subset = keep)
```

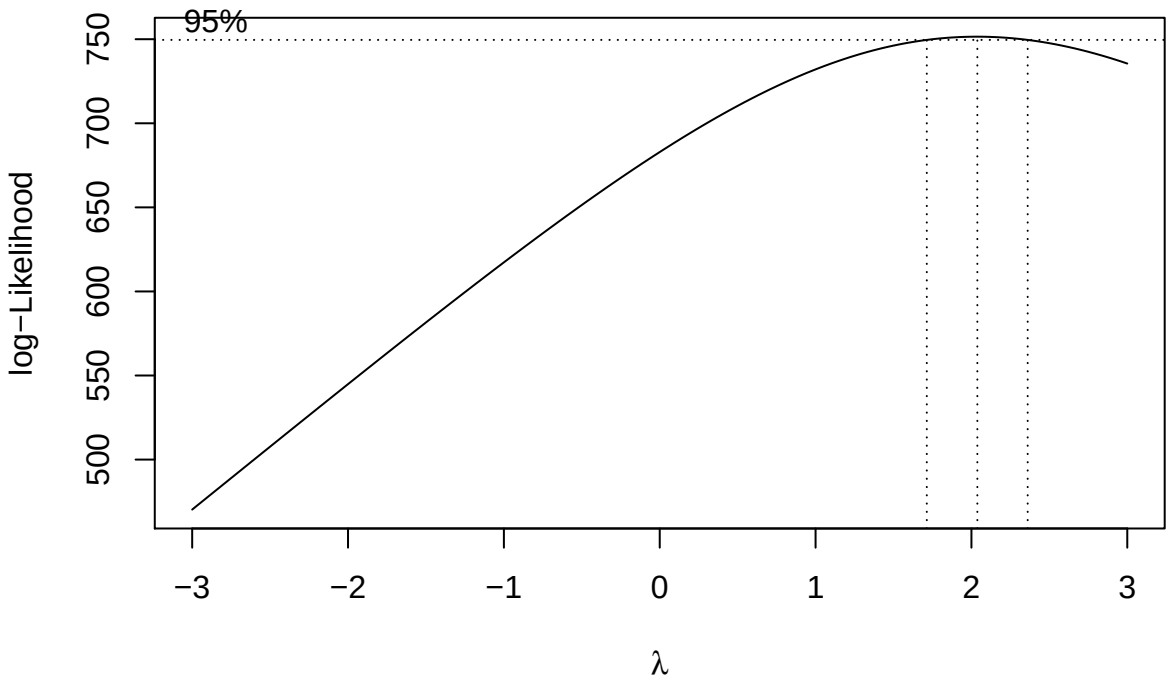
We test the model with the removed high infleunce points. The LOOCV-MSE still remains low, and the adjusted r-squred remains high. Although the model does not explicitly pass the diagnostic tests, it looks a lot more promising in both graphs. So, removing the high-influence points has improved the model in terms of lessening the suspicion of any assumption violations. However, the equal variance assumption still seems to be violated based on the BP test and the residual plot, so some box-cox methodologies are utilized to lessen the suspicion.

```
result_mod_selected_new = run_all_diagnostic(mod_selected_new)
```




| loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|
| 0.0045 | 0.9996 | 20 | 0 | Reject | 0.003 | Reject |

```
library(MASS)
boxcox(Capa ~ temp_max + Amb_Temp + cycle + time_max_temp +
    t_38_35V + Re7  + Rct7 + temp_max:Amb_Temp + temp_max:time_max_temp +
    temp_max:t_38_35V + temp_max:Re7 + Amb_Temp:t_38_35V + cycle:time_max_temp +
    cycle:Re7 + time_max_temp:Re7 + t_38_35V:Rct7, data = cycle_data_full,
       lambda = seq(-3, 3, length = 500))
```

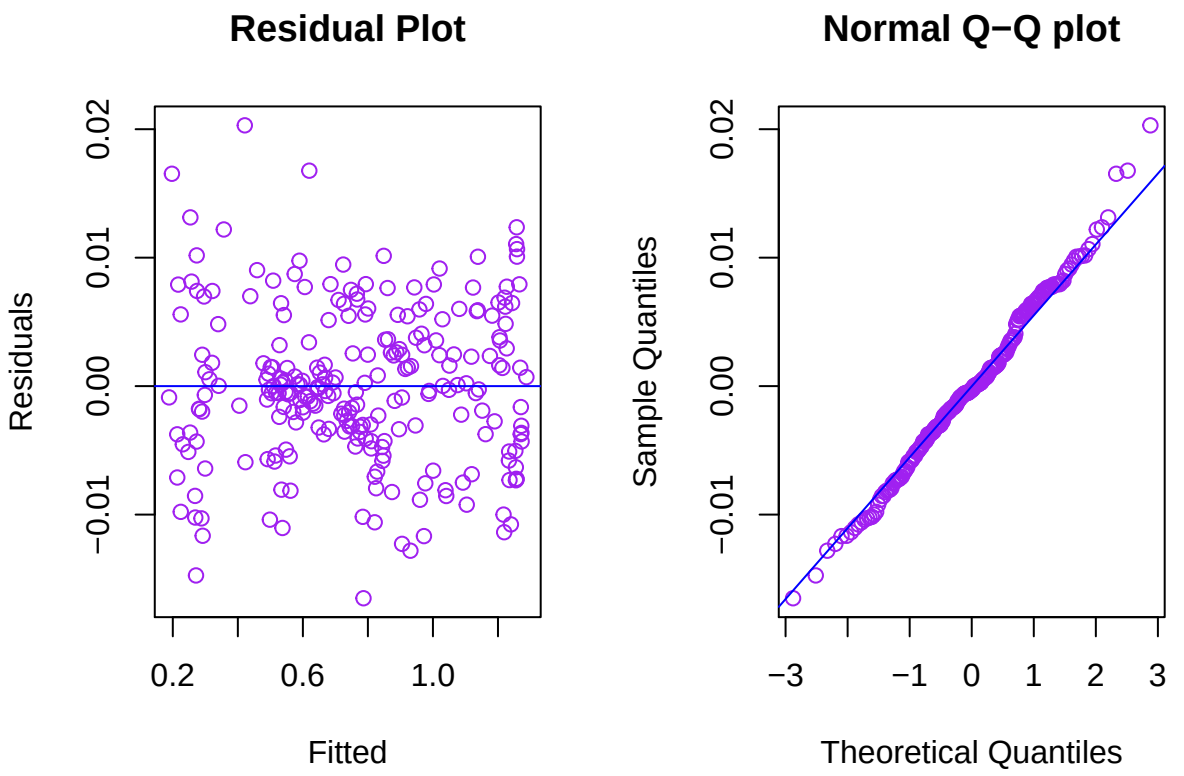



Based on the plot, $\lambda = 2$ maximizes the log-likelihood. This suggests a transformation of form:

$$\frac{y^{\lambda} - 1}{\lambda} = \frac{y^2 - 1}{2}$$

```
mod_selected_trans = lm((Capa^2-1)/2 ~ temp_max + Amb_Temp + cycle + time_max_temp +
    t_38_35V + Re7  + Rct7 + temp_max:Amb_Temp + temp_max:time_max_temp +
    temp_max:t_38_35V + temp_max:Re7 + Amb_Temp:t_38_35V + cycle:time_max_temp +
    cycle:Re7 + time_max_temp:Re7 + t_38_35V:Rct7, data = cycle_data_full, subset = keep)
```

```
result_mod_selected_trans = run_all_diagnostic(mod_selected_trans)
```



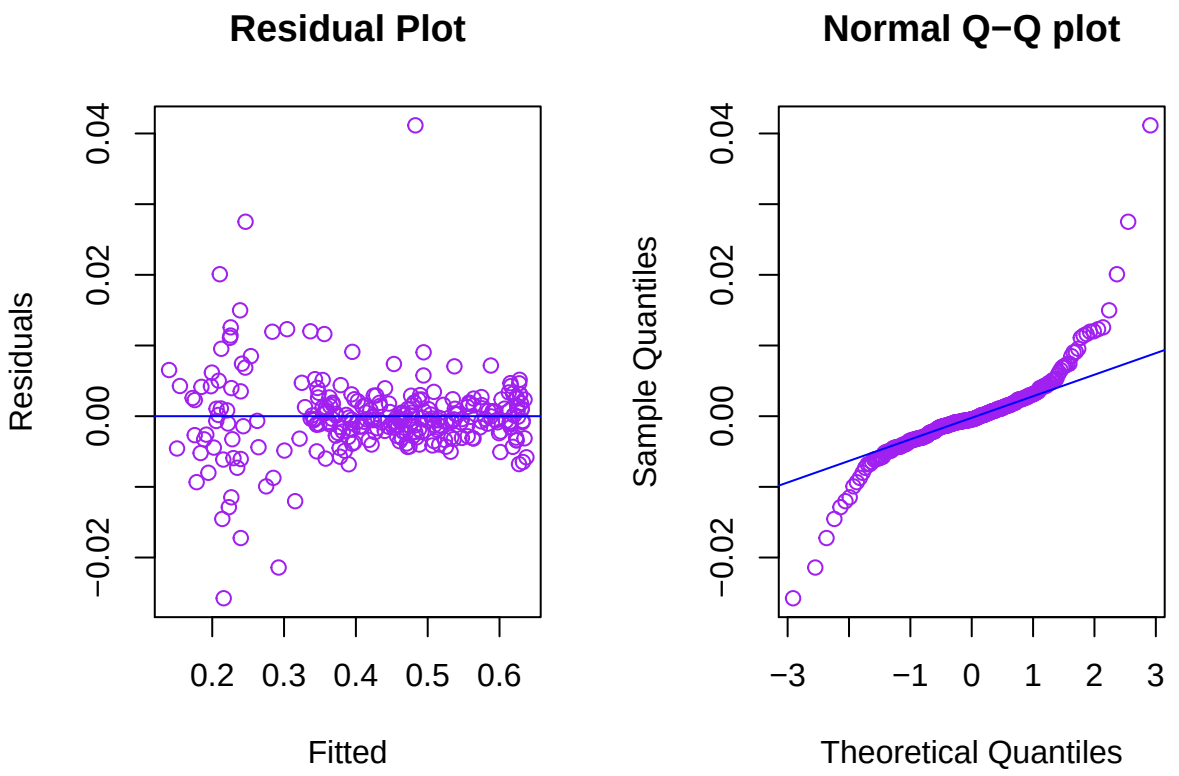

| loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|
| 0.0068 | 0.9996 | 20 | 0 | Reject | 0.3686 | Fail to Reject |

After Box-Cox transformation, the Fitted vs Residuals plot looks better, but based on the Breusch-Pagan test, we still reject the null hypothesis. The Q-Q plot looks promising, and Shapiro-Wilk test fails to reject the null hypothesis, so it is safe to assume the normality assumption is not violated.

However, if we use a different technique to try to come up with a model that surely does not violate the equal variance assumption, the model violate the bp test.

```
#This model is randomly chosen by excluding temp_max variable, which does not have significant influnce
mo1 = lm(log(Capa) ~ Amb_Temp * cycle * t_38_35V * Re7 * Rct7 + I(cycle ^2) + I(t_38_35V ^2) + I(Re7 ^2)

result_mo1 = run_all_diagnostic(mo1)
```




| loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|
| 0.0104 | 0.9978 | 57 | 0.4084 | Fail to Reject | 0 | Reject |

In this model, it passes the bp test but failed the Shapiro–Wilk test test. Therefore, there exists a trade off between the bp test and the Shapiro–Wilk test.

## Results

Our proposed model carefully selected predictors from full two-way interaction model. Then we have applied Box-Cox transformation. We have done some ablation study to prove that the proposed model is the preferred model when compared to the other models. Fully additive model and full three-way interaction model are used for the ablation study.

**Full additive model and its variations**

| Model | loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|---|
| add_selected | 0.0208 | 0.9906 | 8 | 0 | Reject | 0.0000 | Reject |
| add_selected_new | 0.0137 | 0.9956 | 8 | 0 | Reject | 0.0249 | Reject |
| add_selected_trans | 0.0103 | 0.9962 | 8 | 0 | Reject | 0.0021 | Reject |

**Full three-way interaction model and its variations**

| Model | loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|---|
| int_selected | 0.0096 | 0.9989 | 28 | 0 | Reject | 0.0000 | Reject |
| int_selected_new | 0.0042 | 0.9996 | 28 | 0 | Reject | 0.2542 | Fail to Reject |
| int_selected_trans | 0.0069 | 0.9996 | 28 | 0 | Reject | 0.5172 | Fail to Reject |

**Proposed model**

| Model | loocv_rmse | adj.r.squared | params | bp.p_val | bp.decision | sw.p_val | sw.decision |
|---|---|---|---|---|---|---|---|
| mod_selected | 0.0076 | 0.9989 | 20 | 0 | Reject | 0.0000 | Reject |
| mod_selected_new | 0.0045 | 0.9996 | 20 | 0 | Reject | 0.0030 | Reject |
| mod_selected_trans | 0.0068 | 0.9996 | 20 | 0 | Reject | 0.3686 | Fail to Reject |

While `full three-way interaction model` achieved the best accuracy as in `0.0042`, we choose to use the proposed model. That is because the number of parameters used for the proposed model is less than the three-way interaction model, while their `loocv_rmse` difference is very small. The difference is approximately `0.0026`. Therefore, we chose interpretability over accuracy.

## Discussion

First, we can take a look at the additive models for reference, which obviously have the highest error of all the possible models. The coefficients are shown below:

```
names(coef(mod_full_add))
```

```
## [1] "(Intercept)"   "temp_max"      "Amb_Temp24"    "Amb_Temp43"
## [5] "cycle"         "time_max_temp" "t_38_35V"      "Re7"
## [9] "Rct7"
```

And after utilizing the `summary` function, all of these coefficients are determined significant to the model, which makes sense because that was the purpose of utilizing BIC. The only was the electrolyte resistance `Re7`, which makes sense because it would be very similar to transfer resistance `Rct7`, so both are not needed in the model. However, it is clear there are interactions and some non-linear relationship between capacity and other predictors.

The coefficients of the chosen model are shown below:

```
names(coef(mod_full_int_trans))
```

```
##  [1] "(Intercept)"                "temp_max"
##  [3] "Amb_Temp24"                 "Amb_Temp43"
##  [5] "cycle"                      "time_max_temp"
##  [7] "t_38_35V"                   "Re7"
##  [9] "Rct7"                       "temp_max:cycle"
## [11] "temp_max:time_max_temp"     "temp_max:t_38_35V"
## [13] "temp_max:Rct7"              "cycle:time_max_temp"
## [15] "cycle:Re7"                  "cycle:Rct7"
## [17] "time_max_temp:t_38_35V"     "time_max_temp:Re7"
## [19] "time_max_temp:Rct7"         "t_38_35V:Re7"
## [21] "t_38_35V:Rct7"              "Re7:Rct7"
```

```
## [23] "temp_max:cycle:time_max_temp" "temp_max:cycle:Rct7"
## [25] "cycle:time_max_temp:Rct7"     "cycle:Re7:Rct7"
## [27] "time_max_temp:t_38_35V:Rct7"  "t_38_35V:Re7:Rct7"
```

The model essentially says that all factors affect capacity degradation models. We were hoping for a simpler model, but if it were a simple relationship, there would be probably exist a direct measurement for capacity within devices. However, the model does provide some insight, as the predictors have to do mostly with temperature, resistance, and time. This suggests from a practical point of view that the faster the internal battery temperature rises, the lower the capacity of the battery. This makes sense because it would also suggest a higher internal resistance within the battery. However, the temperature of the internal battery would be effected by the external temperature (ambient temperature) up until a certain point when the battery reaches its highest temperature. The model also suggests that cycle number has an effect on capacity degradation, ie. how many times you would charge and recharge.

Also, we realized there's a trade off between the bp test and Shapiro–Wilk test as we fitted many models but some of them can only satisfy one test. In reality, this is common for data proccessor to encounter and it should train our mindset on how should we balance the trade off and select a better model to improve the accuracy for data prediction. The model that passed the Shapiro–Wilk test (`mo1`) did not consider the maximum temperature in the model, which is known in battery systems to be an important factor for overall health estimation. In addition, `mo1` is quite a complex model, with more predictor coefficients, and it also violates the normality assumption. The proposed model considers `temp_max` and most likely does not violate any normality assumptions. Moreover, it is expected that this data would have issues with equal variance because the predictors within the dataset are highly correlated with one another. The ambient temperature `Amb_temp` has a direct effect on the internal temperature, electrolyte and transfer resistance are very similar measurements, and cycle number effects the time in the cycle to highest temperature. The capacity degradation is effected by all of these factors and the interactions between them; so it remains a challenge to find an exact model that represents the capacity degradation with equal variance. Nonetheless, the proposed model provides a solid representation of battery degradation.

## Appendix

The following code is the data cleaning code.

Read in the three batteries, and combine them.

```
library(readr)

data7 = read.csv("B7_reorganized_data.csv")
data30 = read.csv("B30_reorganized_data.csv")
data48 = read.csv("B48_reorganized_data.csv")

data_full = rbind(data7,data30,data48)
```

Right now, we can look only at the data from discharging. The following is the full dataset (all three batteries) but just with the discharge type.

```
data_full_discharge = data_full[data_full$charge_type=="Discharge",]
```

It is clear that the capacity changes at a macro rate (we see the change at the end of each cycle); so it is advantageous to set up the data in terms of cycle. In this case, we summarize the data for the cycle in the following ways: maximum temperature during the cycle, total time of the cycle

```
#extract number of cycles from each battery
cycles_7 = max(unique(data7$cycle_number))-1
cycles_30 = max(unique(data30$cycle_number))-1
cycles_48 = max(unique(data48$cycle_number))-1
```

It is easier to clean individual battery dataset, then combine them in the end.

```
data7_discharge = data7[data7$charge_type=="Discharge",]
data30_discharge = data30[data30$charge_type=="Discharge",]
data48_discharge = data48[data48$charge_type=="Discharge",]
```

Based on the voltage curve, an important factor is the time it takes for the battery to decrement from 3.8 V to 3.5 V. It is saved as `t_decrement_38_35V`.

```
temp_max_cycle = rep(0,cycles_7)
cycle_time = rep(0,cycles_7)
time_to_max_temp = rep(0,cycles_7)
capacity = rep(0,cycles_7)
t_decrement_38_35V = rep(0,cycles_7)
Re = rep(0,cycles_7)
Rct = rep(0,cycles_7)
ambient_temp = rep(0,cycles_7)
for (each in 1:cycles_7){
  df = data7_discharge[data7_discharge$cycle_number==each,]
  temp_max_cycle[each] = max(df$Temperature_measured)
  cycle_time[each] = tail(df$Time,1)
  time_to_max_temp[each] = tail(df[1:which.max(df$Temperature_measured),]$Time,1)
  condition1 = df$Voltage_measured <3.8 &df$Voltage_measured > 3.5
  t_decrement_38_35V[each] = tail(df[condition1,]$Time,1)-df[condition1,]$Time[1]
  capacity[each] = unique((df$Capacity))
  Re[each] = unique(df$Re)
  Rct[each] = unique(df$Rct)
  ambient_temp[each] = unique(df$Ambient_Temp)

}

df_7 = data.frame(temp_max = temp_max_cycle,Amb_Temp = ambient_temp,cycle = 1:cycles_7,time_max_temp = t
```

`df_7` is our dataset by cycle.

```
temp_max_cycle = rep(0,cycles_30)
cycle_time = rep(0,cycles_30)
time_to_max_temp = rep(0,cycles_30)
capacity = rep(0,cycles_30)
t_decrement_38_35V = rep(0,cycles_30)
Re = rep(0,cycles_30)
Rct = rep(0,cycles_30)
ambient_temp = rep(0,cycles_30)
for (each in 1:cycles_30){
  df = data30_discharge[data30_discharge$cycle_number==each,]
  temp_max_cycle[each] = max(df$Temperature_measured)
  cycle_time[each] = tail(df$Time,1)
```

```
  time_to_max_temp[each] = tail(df[1:which.max(df$Temperature_measured),]$Time,1)
  condition1 = df$Voltage_measured <3.8 &df$Voltage_measured > 3.5
  t_decrement_38_35V[each] = tail(df[condition1,]$Time,1)-df[condition1,]$Time[1]
  capacity[each] = unique((df$Capacity))
  Re[each] = unique(df$Re)
  Rct[each] = unique(df$Rct)
  ambient_temp[each] = unique(df$Ambient_Temp)

}

df_30 = data.frame(temp_max = temp_max_cycle,Amb_Temp = ambient_temp,cycle = 1:cycles_30,time_max_temp =
```

df_30 is our dataset by cycle

```
temp_max_cycle = rep(0,cycles_48)
cycle_time = rep(0,cycles_48)
time_to_max_temp = rep(0,cycles_48)
capacity = rep(0,cycles_48)
t_decrement_38_35V = rep(0,cycles_48)
Re = rep(0,cycles_48)
Rct = rep(0,cycles_48)
ambient_temp = rep(0,cycles_48)
for (each in 1:cycles_48){
  df = data48_discharge[data48_discharge$cycle_number==each,]
  temp_max_cycle[each] = max(df$Temperature_measured)
  cycle_time[each] = tail(df$Time,1)
  time_to_max_temp[each] = tail(df[1:which.max(df$Temperature_measured),]$Time,1)
  condition1 = df$Voltage_measured <3.8 &df$Voltage_measured > 3.5
  t_decrement_38_35V[each] = tail(df[condition1,]$Time,1)-df[condition1,]$Time[1]
  capacity[each] = unique((df$Capacity))
  Re[each] = unique(df$Re)
  Rct[each] = unique(df$Rct)
  ambient_temp[each] = unique(df$Ambient_Temp)

}

df_48 = data.frame(temp_max = temp_max_cycle,Amb_Temp = ambient_temp,cycle = 1:cycles_48,time_max_temp =
```

df_48 is our dataset by cycle

Now we combine:

```
cycle_data_full =  rbind(df_7,df_30,df_48)
```

This function to run all the tests was utilized quite a bit for analysis as well:

```
#Run All Test Diagnostic
run_all_diagnostic = function(model, visual=TRUE, alpha=0.05){
  if (visual) {
    par(mfrow = c(1,2))
    test_FvsR(model)
    test_qq(model)
  }
```

```
  loocv_rmse = get_loocv_rmse(model)
  adj.r.squared = get_adj_r2(model)
  params = get_num_params(model)
  bp = test_bp(model, alpha)
  sp = test_sp(model, alpha)
  data_frame = data.frame(loocv_rmse, adj.r.squared, params, bp, sp, row.names = '')
  return(data_frame)
}
```

Regarding BP test, hypothesis test was done under the following setting where $\alpha = 0.05$.

$$H_0 : \text{Homoscedasticity}$$

$$H_1 : \text{Hetroscedasticity}$$

Regarding Shapiro–Wilk test, hypothesis test was done under the following setting where $\alpha = 0.05$

$$H_0 : \text{Normality assumption is not suspect}$$

$$H_1 : \text{Normality assumption is suspect}$$

The following code was utilized in the results section:

```
# Fully addictive model
mod_full_add = lm(Capa~. , data=cycle_data_full)
mod_full_add_selected = step(mod_full_add, direction = 'backward', k = log(nrow(data_full)), trace = 0)
keep = cooks.distance(mod_full_add_selected) < 4 / length(cooks.distance(mod_full_add_selected))

mod_full_add_new = lm(Capa~ temp_max + Amb_Temp + cycle + time_max_temp + t_38_35V + Rct7 ,subset=keep,

boxcox(Capa~ temp_max + Amb_Temp + cycle + time_max_temp + t_38_35V + Rct7, data = cycle_data_full,
       lambda = seq(-3, 3, length = 500))

# lambda is 0.5
mod_full_add_trans = lm((Capa^.5-1)/.5 ~
 temp_max + Amb_Temp + cycle + time_max_temp + t_38_35V + Rct7, subset=keep, data=cycle_data_full)

result_add_selected = run_all_diagnostic(mod_full_add_selected, visual=FALSE)
result_add_selected_new = run_all_diagnostic(mod_full_add_new, visual=FALSE)
result_add_selected_trans = run_all_diagnostic(mod_full_add_trans, visual=FALSE)
```

**Full addictive model and its variations**

```
add_result = rbind.data.frame(
cbind.data.frame(Model='add_selected', result_add_selected),
cbind.data.frame(Model='add_selected_new', result_add_selected_new),
cbind.data.frame(Model='add_selected_trans', result_add_selected_trans)
)

kable(add_result, row.names=FALSE)
```

**Full three-way interaction model and its variations**

```
three_result = rbind.data.frame(
cbind.data.frame(Model='int_selected', result_int_selected),
cbind.data.frame(Model='int_selected_new', result_int_selected_new),
cbind.data.frame(Model='int_selected_trans', result_int_selected_trans)
)

kable(three_result, row.names=FALSE)
```

**Proposed model**

```
final_result = rbind.data.frame(
cbind.data.frame(Model='mod_selected', result_mod_selected),
cbind.data.frame(Model='mod_selected_new', result_mod_selected_new),
cbind.data.frame(Model='mod_selected_trans', result_mod_selected_trans)
)

kable(final_result, row.names=FALSE)
```

## References


1. B. Saha and K. Goebel (2007). "Battery Data Set", NASA Ames Prognostics Data Repository(https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/, NASA Ames Research Center, Moffett Field, CA