181081907 Pooja V. Patil
181081904 Sejal Jain
181081909 Mitali Wade
181081905 Rajashree Gavhane
181081910 Amruta Bansode
TY B.Tech[IT]

Analysis of the data set and comparative study

In [144]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.svm import SVR
%matplotlib inline

from mat2json import loadMat
from util import getBatteryCapacity, getChargingValues, getDischargingValues, getDataframe, series_to_supe
rvised, rollingAverage
```

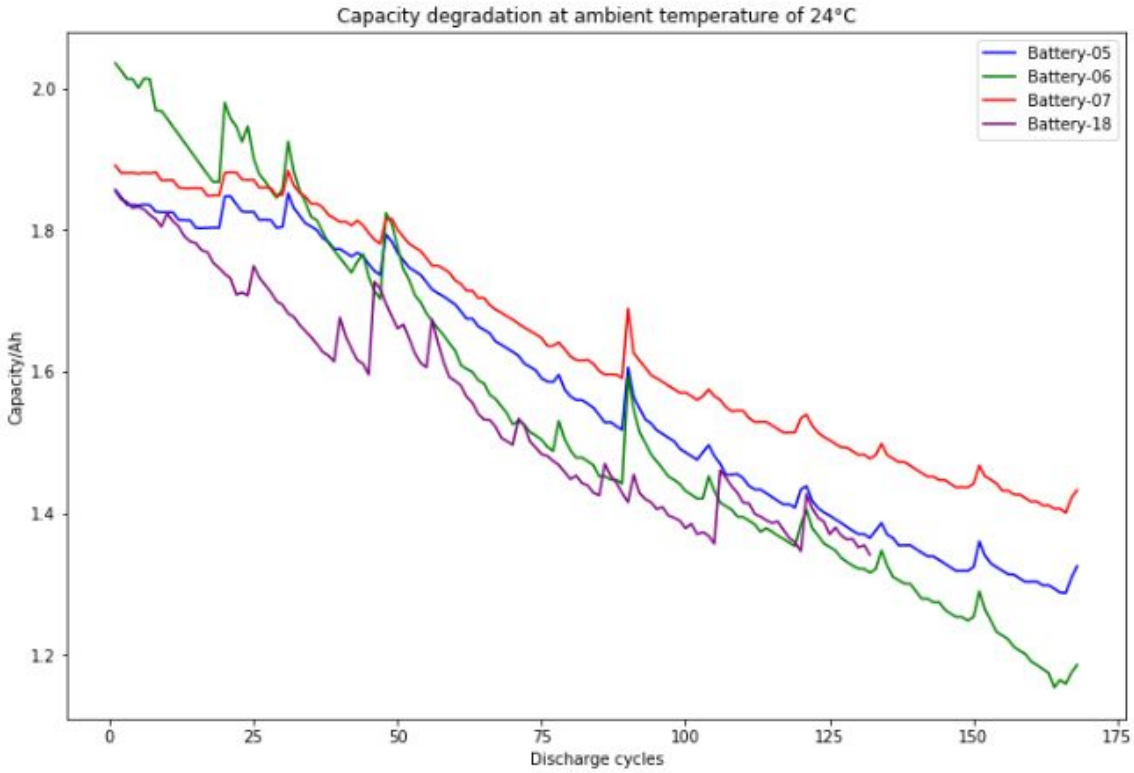
In [3]:
```
#Ambient temp 24
B0005 = loadMat('B0005.mat')
B0006 = loadMat('B0006.mat')
B0007 = loadMat('B0007.mat')
B0018 = loadMat('B0018.mat')
```

In [4]:
```
B0005_capacity = getBatteryCapacity(B0005)
B0006_capacity = getBatteryCapacity(B0006)
B0007_capacity = getBatteryCapacity(B0007)
B0018_capacity = getBatteryCapacity(B0018)
```

Capacity degradation at certain temperature:

In [6]:
```
fig, ax = plt.subplots(1, figsize=(12, 8))

ax.plot(B0005_capacity[0], B0005_capacity[1], color='blue', label='Battery-05')
ax.plot(B0006_capacity[0], B0006_capacity[1], color='green', label='Battery-06')
ax.plot(B0007_capacity[0], B0007_capacity[1], color='red', label='Battery-07')
ax.plot(B0018_capacity[0], B0018_capacity[1], color='purple', label='Battery-18')
ax.set(xlabel='Discharge cycles', ylabel='Capacity/Ah', title='Capacity degradation at ambient temperature
of 24°C')
plt.legend()
```

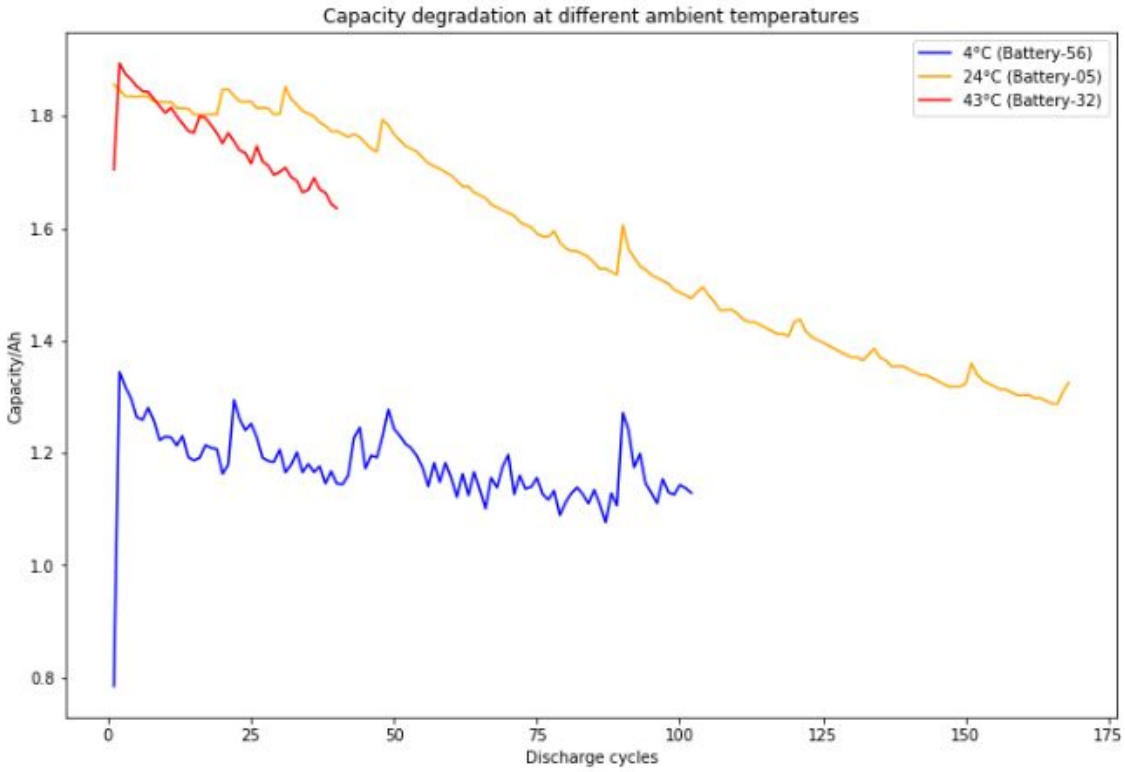
Out[6]:
```
<matplotlib.legend.Legend at 0x7f314a8e5400>
```



Capacity degradation at different temperatures:

In [8]:
```
fig, ax = plt.subplots(1, figsize=(12, 8))

ax.plot(B0056_capacity[0], B0056_capacity[1], color='blue', label='4°C (Battery-56)')
ax.plot(B0005_capacity[0], B0005_capacity[1], color='orange', label='24°C (Battery-05)')
ax.plot(B0032_capacity[0], B0032_capacity[1], color='red', label='43°C (Battery-32)')
ax.set(xlabel='Discharge cycles', ylabel='Capacity/Ah', title='Capacity degradation at different ambient t
emperatures')
plt.legend()
```

Out[8]:
```
<matplotlib.legend.Legend at 0x7f314905ff28>
```



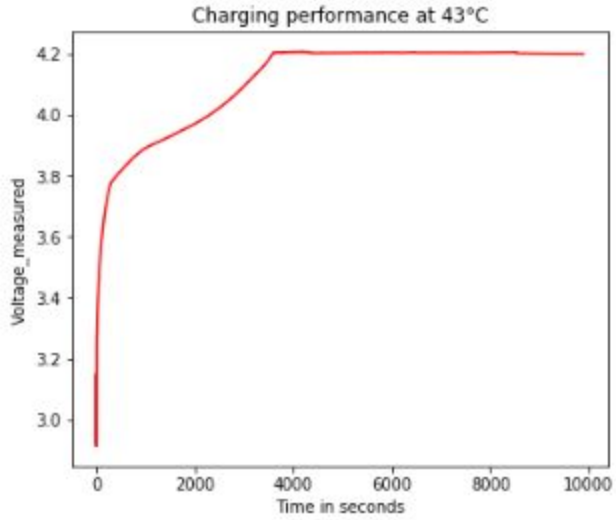
Charging performance at various temperatures:

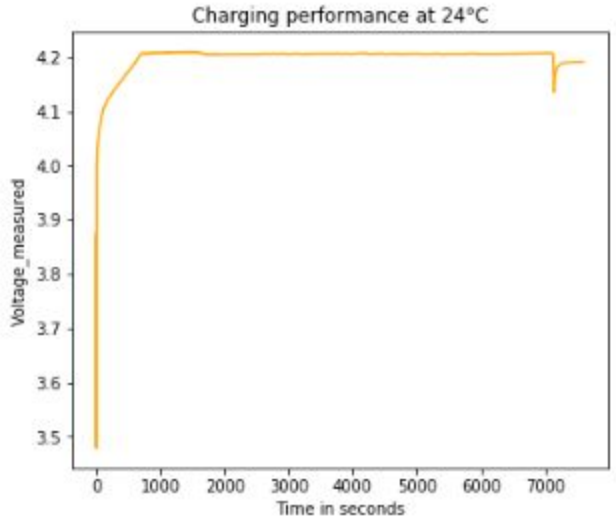
In [13]:
```
B0005_charging = getChargingValues(B0005, 0)
B0029_charging = getChargingValues(B0029, 2)
B0054_charging = getChargingValues(B0054, 5)
```

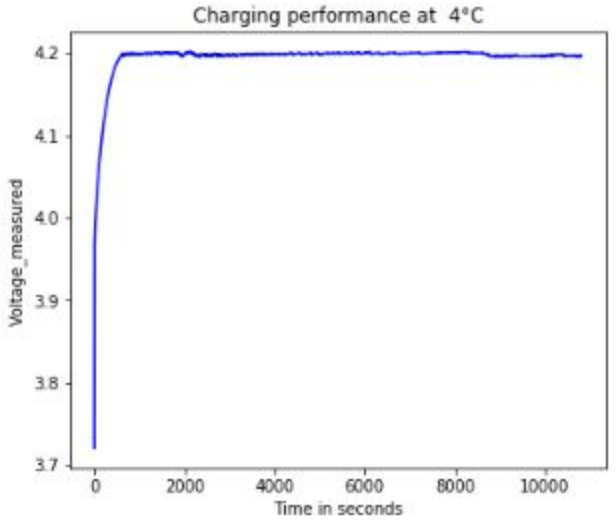
In [14]:
```
charging_labels = ['Voltage_measured','Current_measured','Temperature_measured']
```

In [15]:
```
indx = 1
for label in charging_labels:
    fig, ax = plt.subplots(1, figsize=(6, 5))
    fig1, ax1 = plt.subplots(1, figsize=(6, 5))
    fig2, ax2 = plt.subplots(1, figsize=(6, 5))

    ax.plot(B0029_charging[5], B0029_charging[indx], color='red')
    ax1.plot(B0005_charging[5], B0005_charging[indx], color='orange')
    ax2.plot(B0054_charging[5], B0054_charging[indx], color='blue')

    ax.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at 43°C')
    ax1.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at 24°C')
    ax2.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at  4°C')

    indx += 1
```

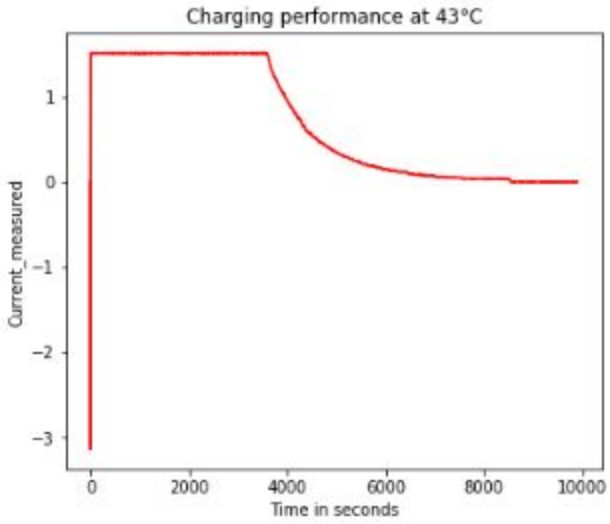
Voltage measured:

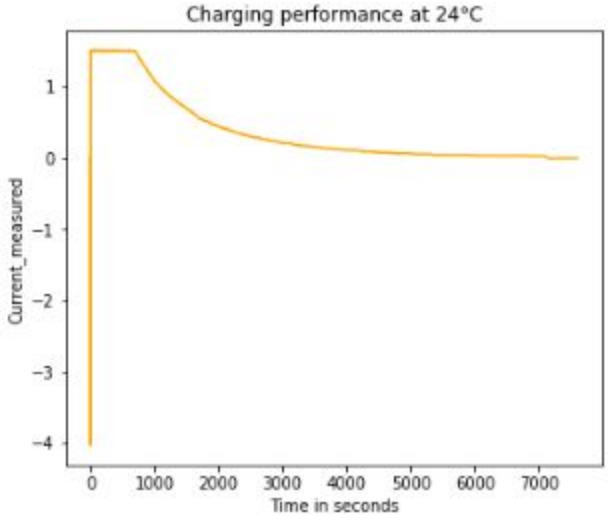


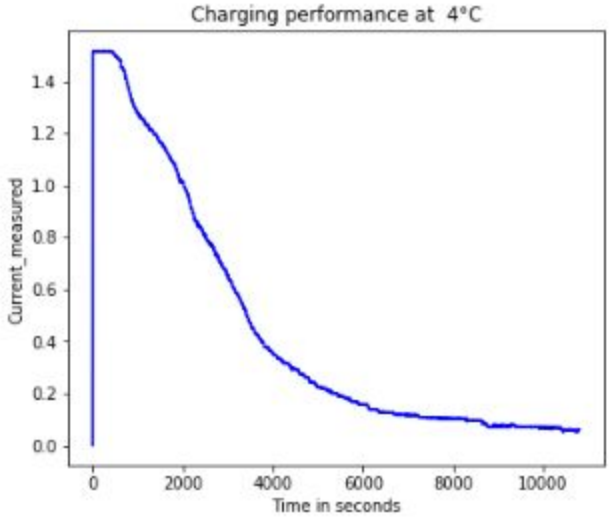




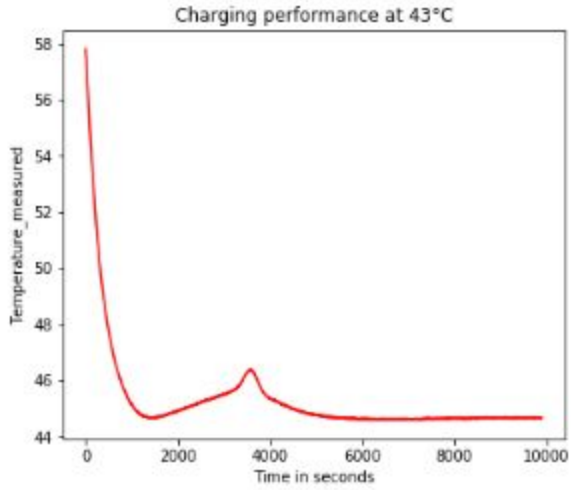
Current measured:

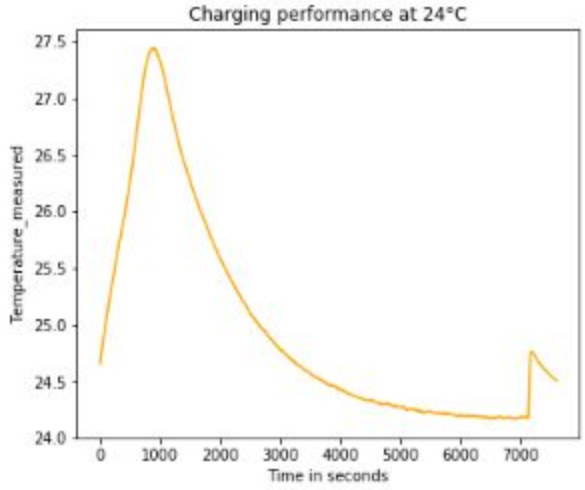

Charging performance at 43°C
Current_measured
Time in seconds
1
0
−1
−2
−3
0
2000
4000
6000
8000
10000

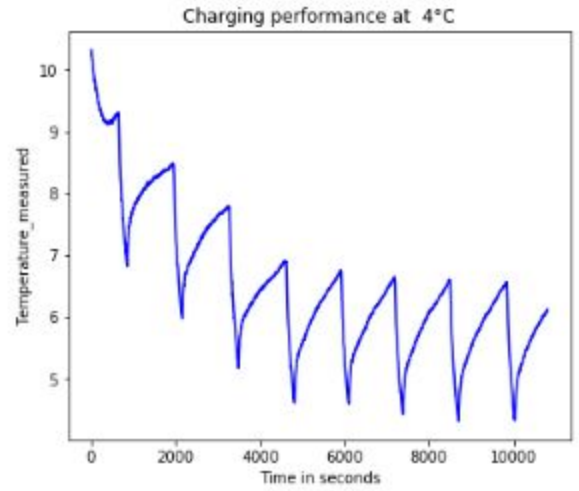

Charging performance at 24°C
Current_measured
Time in seconds
1
0
−1
−2
−3
−4
0
1000
2000
3000
4000
5000
6000
7000


Charging performance at 4°C
Current_measured
Time in seconds
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
2000
4000
6000
8000
10000

Temperature measured:







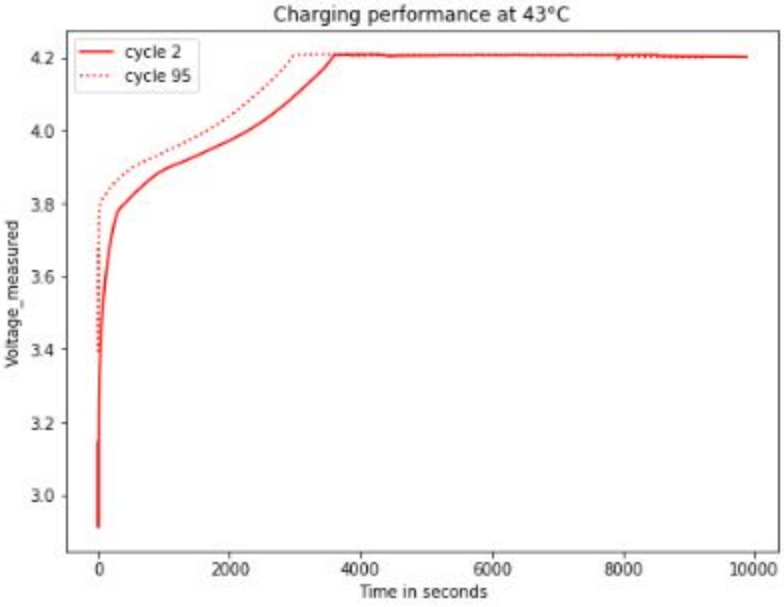
In [16]:
```
B0005[400]['cycle']
```

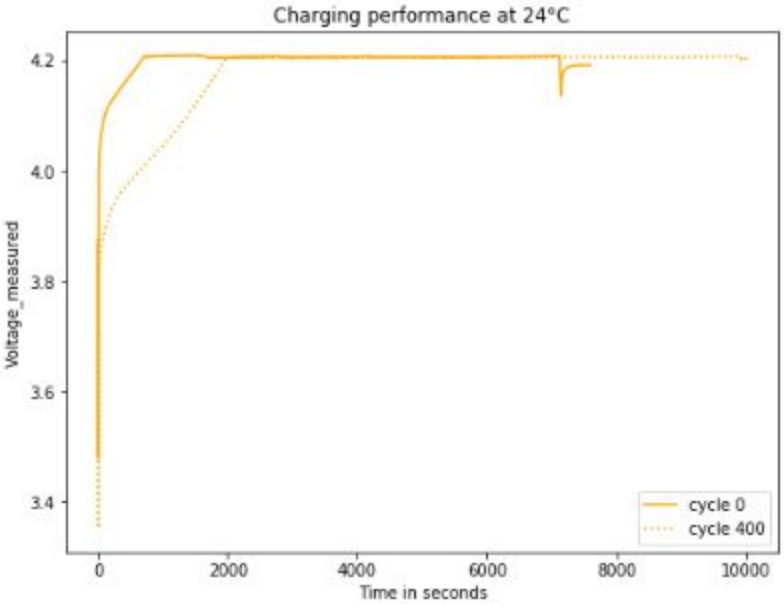
Out[16]:
```
'charge'
```

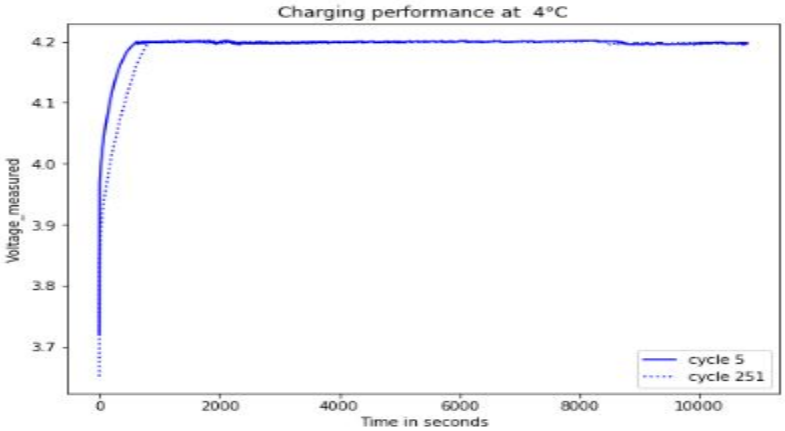
In [17]:
```
B0005_charging_400 = getChargingValues(B0005, 400)
B0029_charging_95 = getChargingValues(B0029, 95)
B0054_charging_251 = getChargingValues(B0054, 251)
```

In [18]:
```
indx = 1
for label in charging_labels:
    fig, ax = plt.subplots(1, figsize=(8, 6))
    fig1, ax1 = plt.subplots(1, figsize=(8, 6))
    fig2, ax2 = plt.subplots(1, figsize=(8, 6))

    ax.plot(B0029_charging[5], B0029_charging[indx], color='red', label='cycle 2')
    ax.plot(B0029_charging_95[5], B0029_charging_95[indx], linestyle=':', color='red', label='cycle 95')
    ax1.plot(B0005_charging[5], B0005_charging[indx], color='orange', label='cycle 0')
    ax1.plot(B0005_charging_400[5], B0005_charging_400[indx], linestyle=':', color='orange', label='cycle
400')
    ax2.plot(B0054_charging[5], B0054_charging[indx], color='blue', label='cycle 5')
    ax2.plot(B0054_charging_251[5], B0054_charging_251[indx], linestyle=':', color='blue', label='cycle 25
1')

    ax.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at 43°C')
    ax1.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at 24°C')
    ax2.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at  4°C')
    ax.legend()
    ax1.legend()
    ax2.legend()

    indx += 1
```

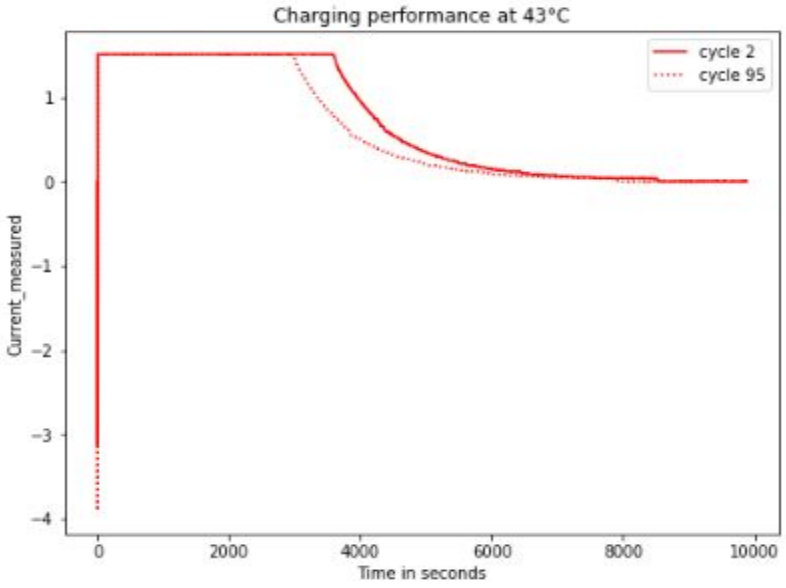
Voltage:

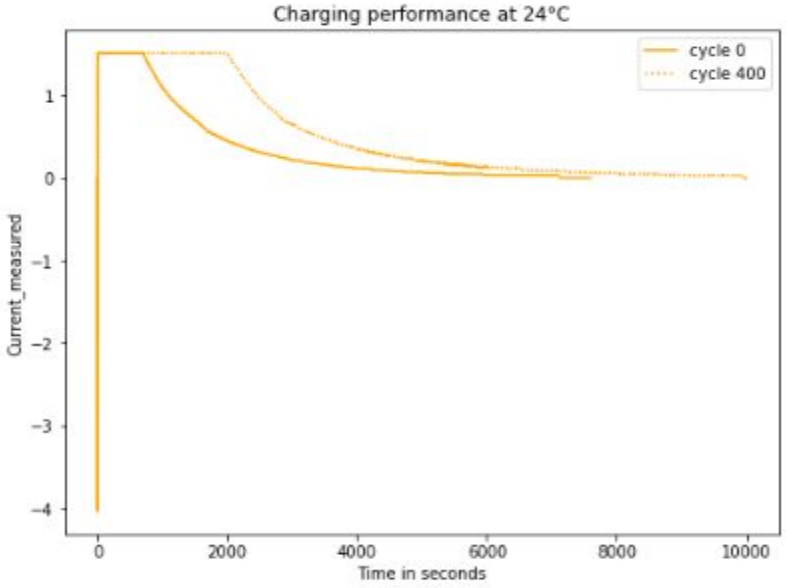

Charging performance at 43°C
cycle 2
cycle 95
Voltage_measured
Time in seconds

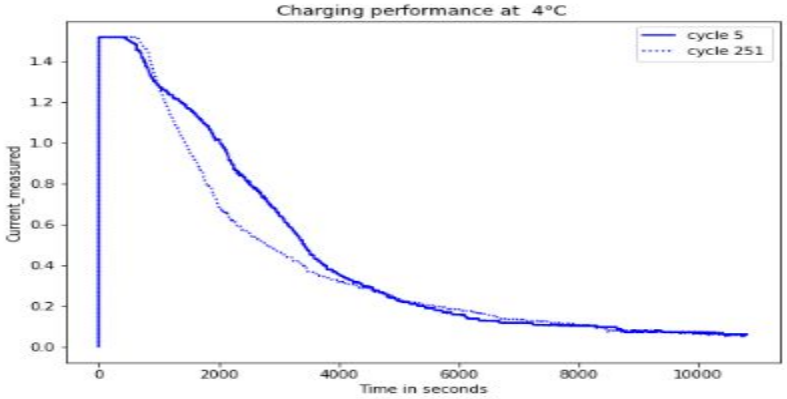

Charging performance at 24°C
cycle 0
cycle 400
Voltage_measured
Time in seconds


Charging performance at 4°C
cycle 5
cycle 251
Voltage_measured
Time in seconds

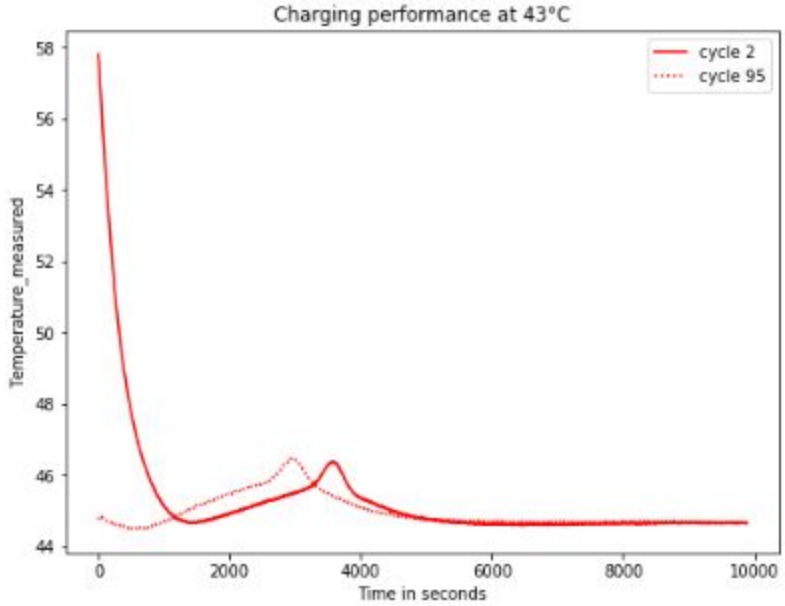
Current:

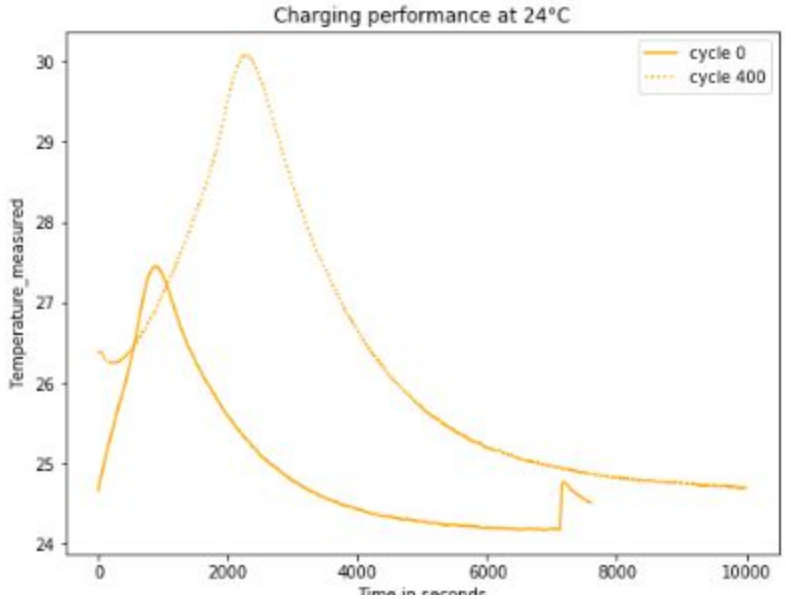

Charging performance at 43°C
cycle 2
cycle 95
Current_measured
Time in seconds

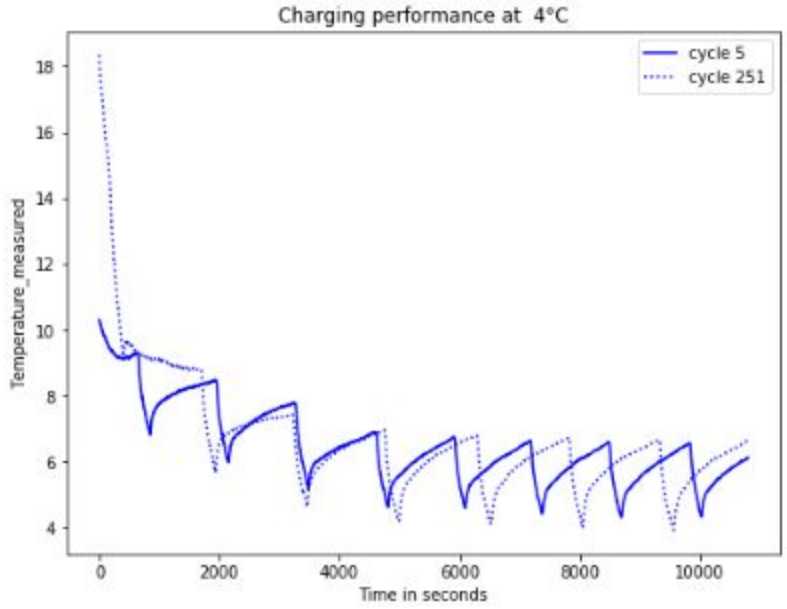

Charging performance at 24°C
cycle 0
cycle 400
Current_measured
Time in seconds


Charging performance at 4°C
cycle 5
cycle 251
Current_measured
Time in seconds

Temperature:

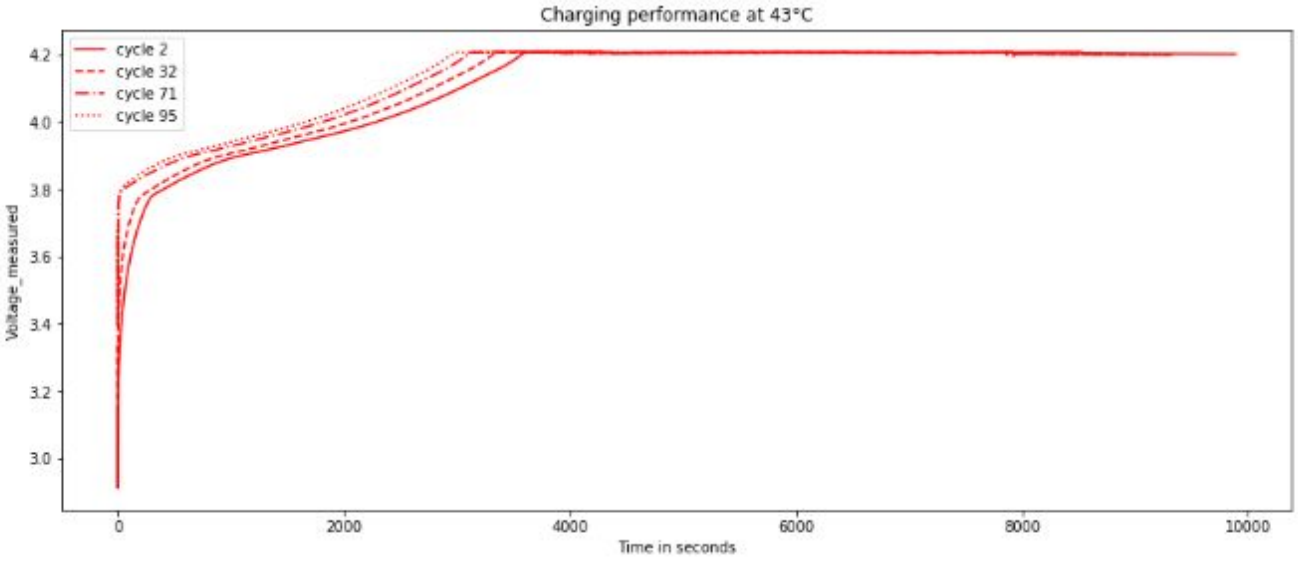


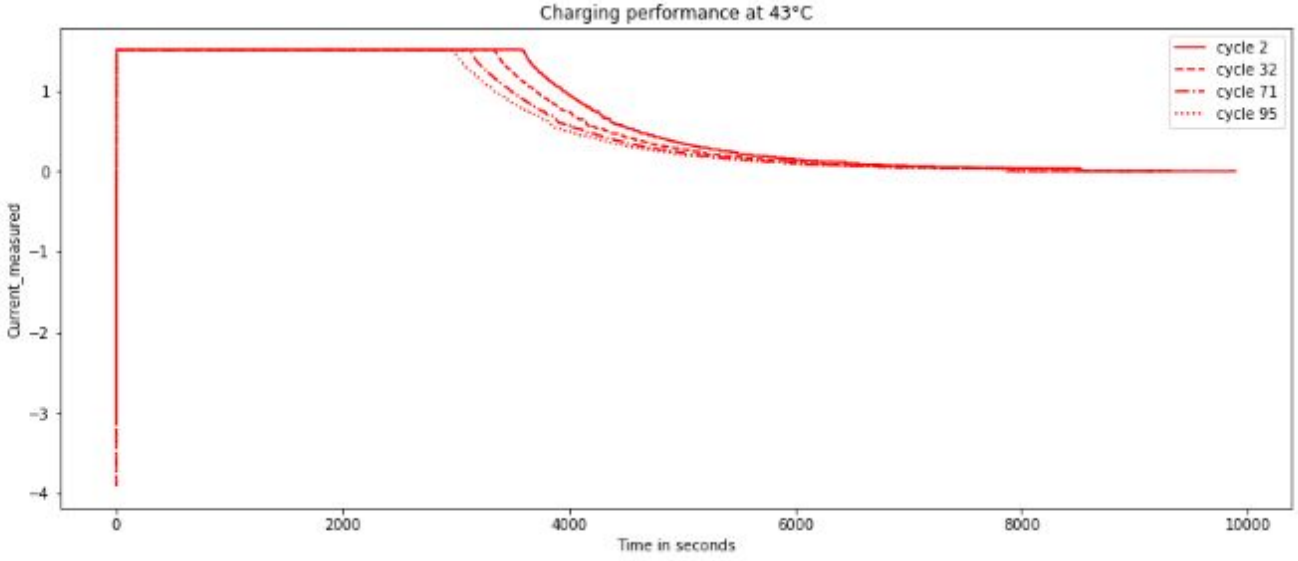


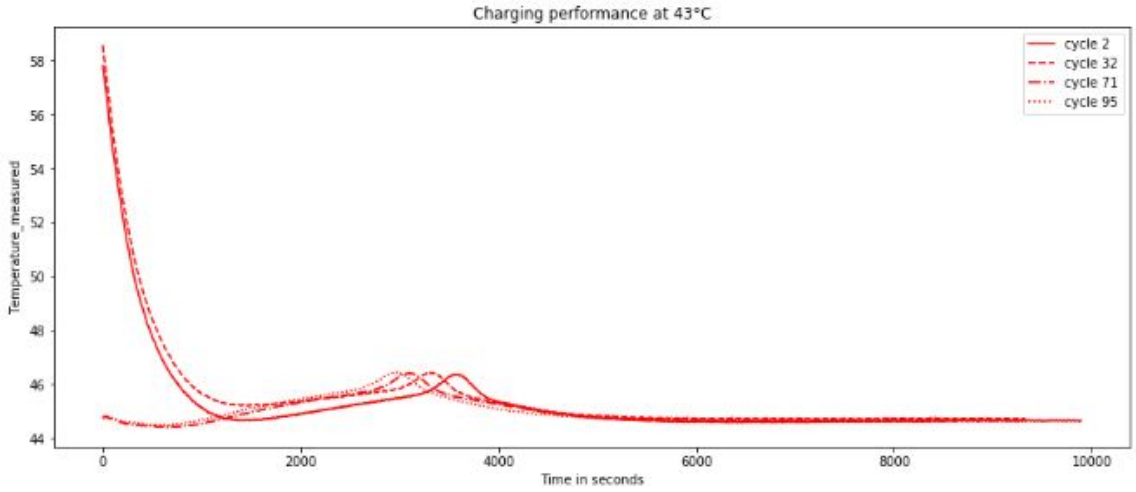


In [19]:
```
B0029_charging_71 = getChargingValues(B0029, 71)
B0029_charging_32 = getChargingValues(B0029, 32)
```

In [20]:
```
indx = 1
for label in charging_labels:
    fig, ax = plt.subplots(1, figsize=(15, 6))

    ax.plot(B0029_charging[5], B0029_charging[indx], color='red', label='cycle 2')
    ax.plot(B0029_charging_32[5], B0029_charging_32[indx], linestyle='--', color='red', label='cycle 32')
    ax.plot(B0029_charging_71[5], B0029_charging_71[indx], linestyle='-.', color='red', label='cycle 71')
    ax.plot(B0029_charging_95[5], B0029_charging_95[indx], linestyle=':', color='red', label='cycle 95')

    ax.set(xlabel='Time in seconds', ylabel=label, title='Charging performance at 43°C')
    ax.legend()

    indx += 1
```


Charging performance at 43°C
cycle 2
cycle 32
cycle 71
cycle 95
Voltage_measured
Time in seconds


Charging performance at 43°C
cycle 2
cycle 32
cycle 71
cycle 95
Current_measured
Time in seconds



Discharging performance:

```
In [21]: B0054[0]['cycle']
```

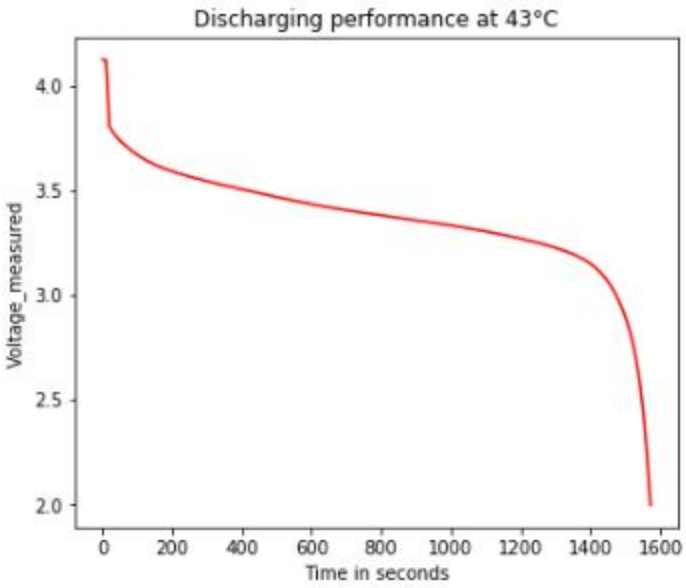
```
Out[21]: 'discharge'
```

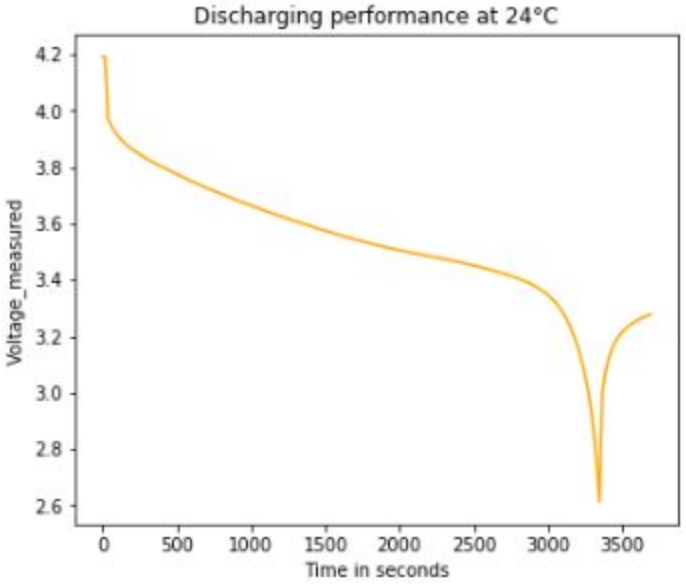
```
In [22]: B0005_discharging = getDischargingValues(B0005, 1)
         B0029_discharging = getDischargingValues(B0029, 1)
         B0054_discharging = getDischargingValues(B0054, 0)
```

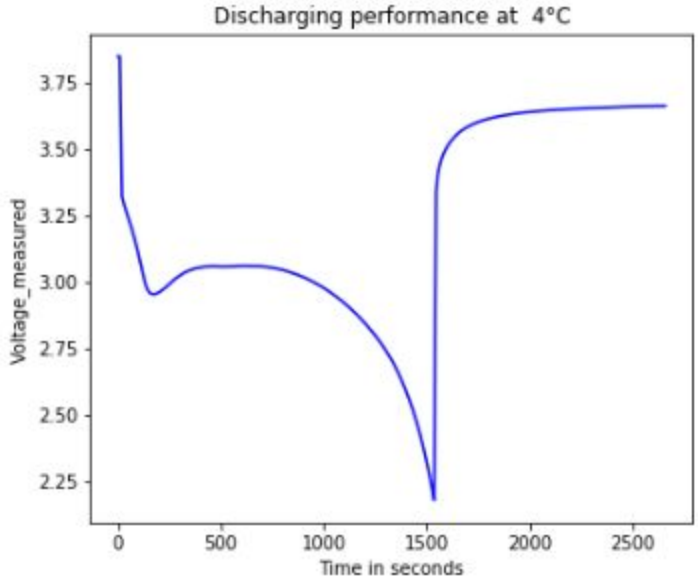
```
In [23]: discharging_labels = ['Voltage_measured','Current_measured','Temperature_measured']
```

```
In [24]: indx = 1
         for label in discharging_labels:
             fig, ax = plt.subplots(1, figsize=(6, 5))
             fig1, ax1 = plt.subplots(1, figsize=(6, 5))
             fig2, ax2 = plt.subplots(1, figsize=(6, 5))

             ax.plot(B0029_discharging[5], B0029_discharging[indx], color='red')
             ax1.plot(B0005_discharging[5], B0005_discharging[indx], color='orange')
             ax2.plot(B0054_discharging[5], B0054_discharging[indx], color='blue')

             ax.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at 43°C')
             ax1.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at 24°C')
             ax2.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at  4°C')

             indx += 1
```

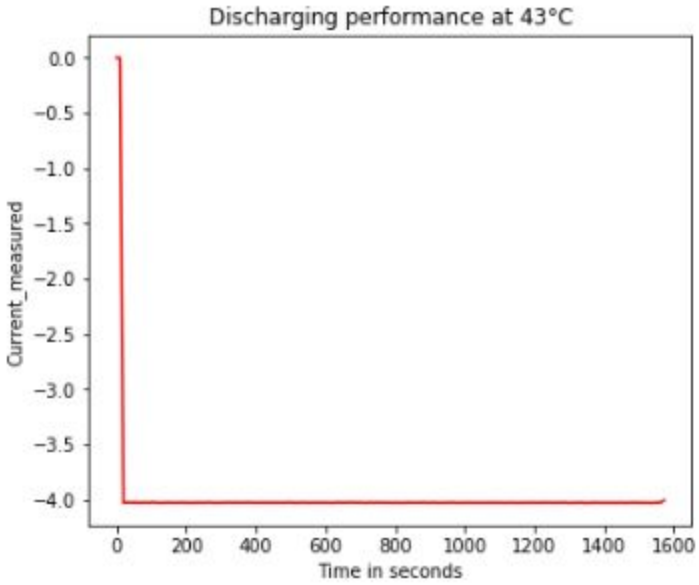
Voltage:

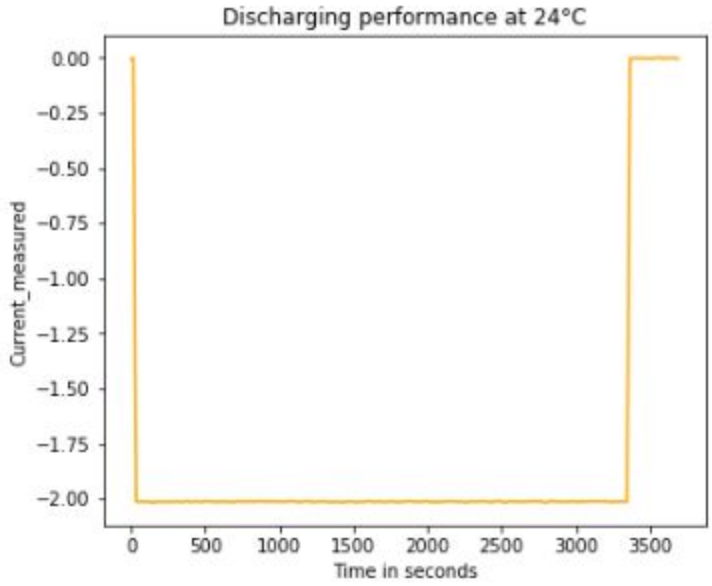

Discharging performance at 43°C
Voltage_measured
4.0
3.5
3.0
2.5
2.0
0
200
400
600
800
1000
1200
1400
1600
Time in seconds

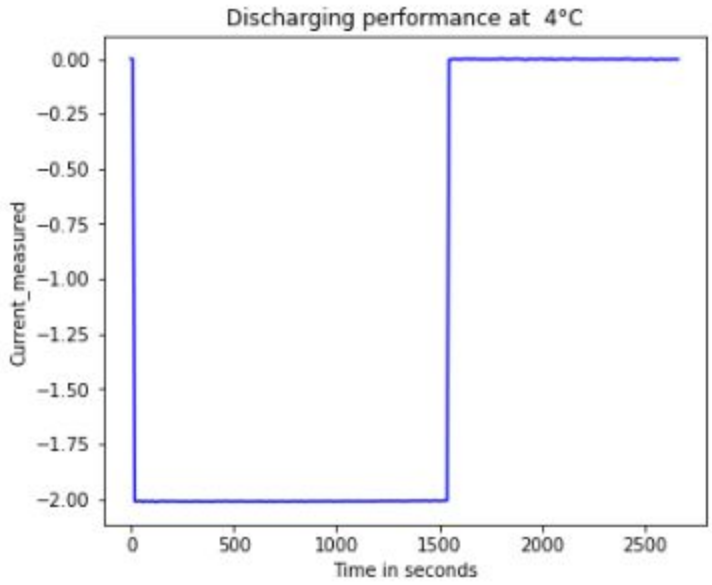

Discharging performance at 24°C
Voltage_measured
4.2
4.0
3.8
3.6
3.4
3.2
3.0
2.8
2.6
0
500
1000
1500
2000
2500
3000
3500
Time in seconds


Discharging performance at 4°C
Voltage_measured
3.75
3.50
3.25
3.00
2.75
2.50
2.25
0
500
1000
1500
2000
2500
Time in seconds

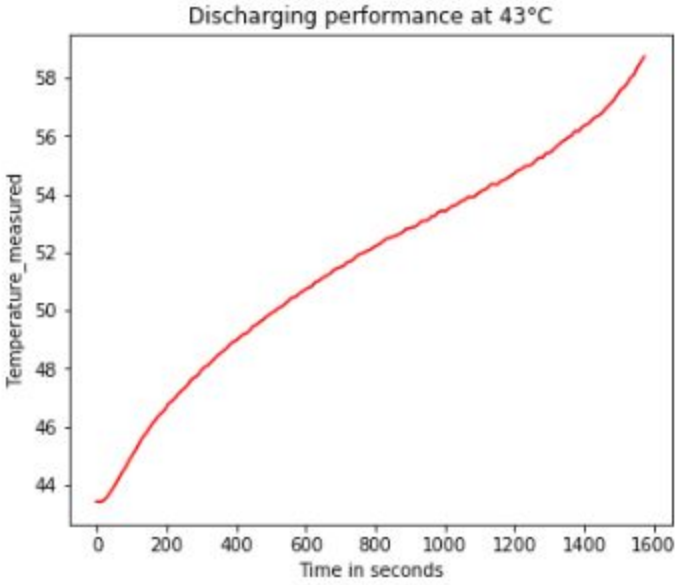
Current:

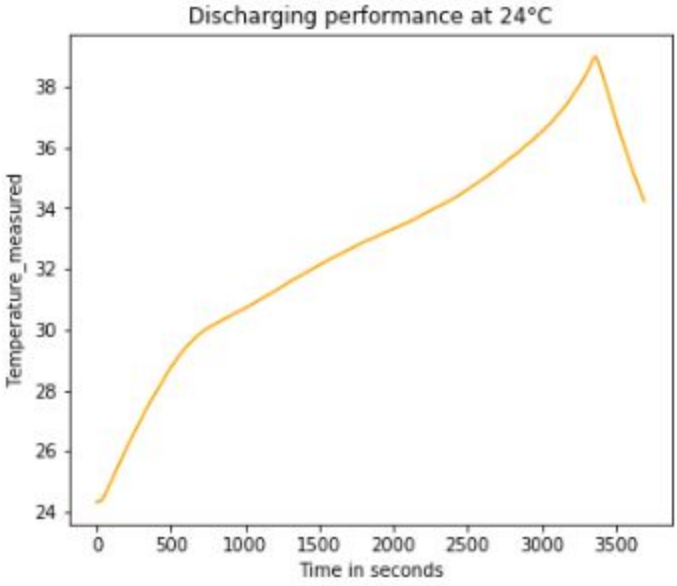

Discharging performance at 43°C
Current_measured
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
−3.5
−4.0
0
200
400
600
800
1000
1200
1400
1600
Time in seconds

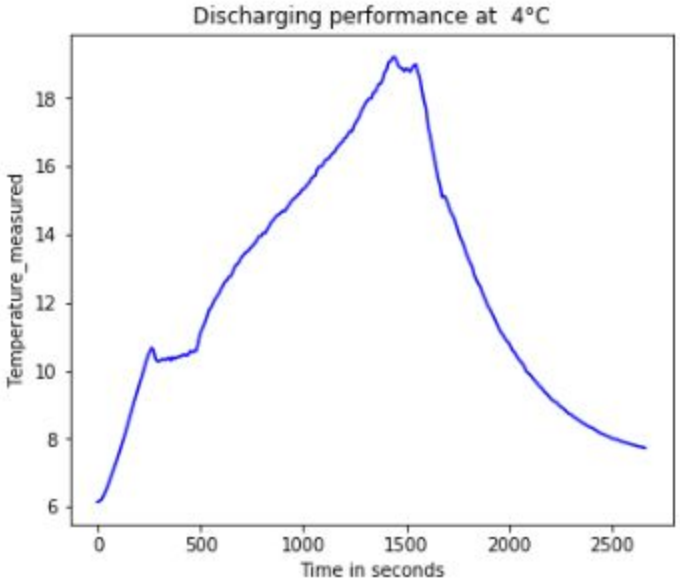




Temperature:


Discharging performance at 43°C
Temperature_measured
Time in seconds
58
56
54
52
50
48
46
44
0
200
400
600
800
1000
1200
1400
1600


Discharging performance at 24°C
Temperature_measured
Time in seconds
38
36
34
32
30
28
26
24
0
500
1000
1500
2000
2500
3000
3500

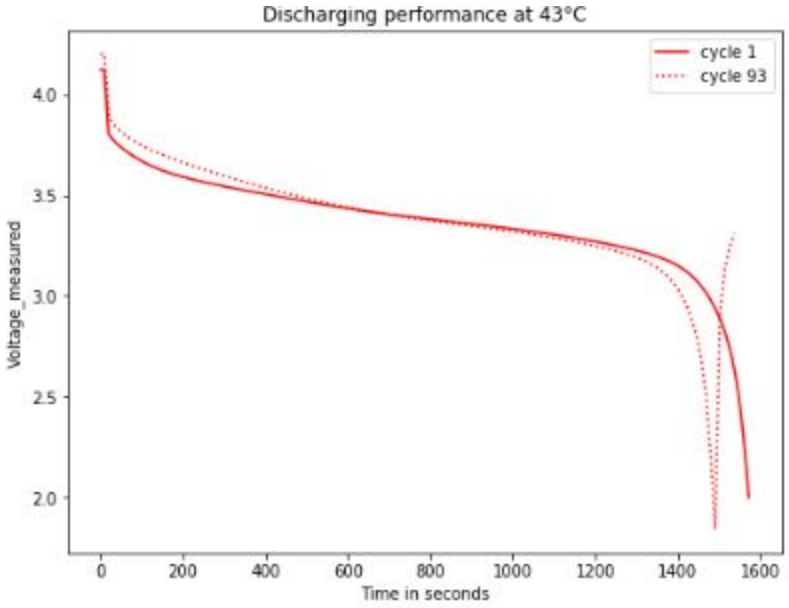


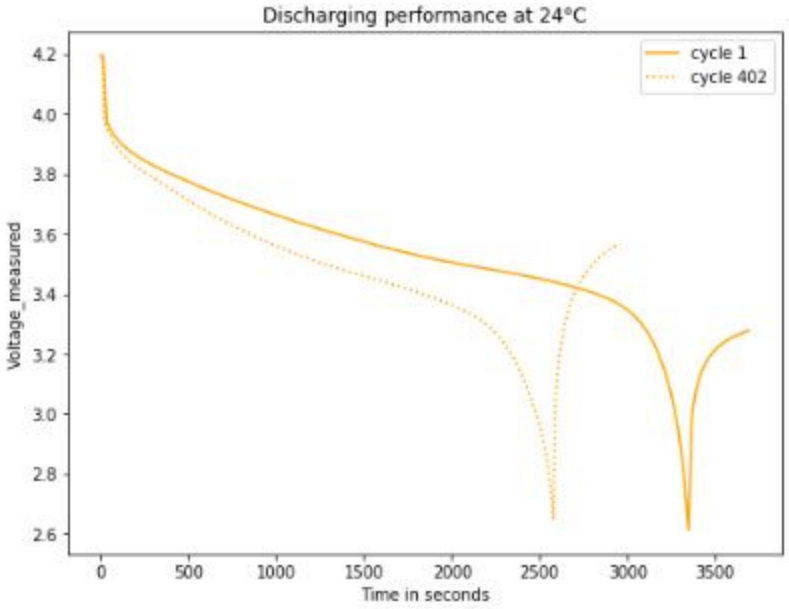
Comparative discharging performance:

In [25]:
```
B0005_discharging_402 = getDischargingValues(B0005, 402)
B0029_discharging_93 = getDischargingValues(B0029, 93)
B0054_discharging_250 = getDischargingValues(B0054, 250)
```

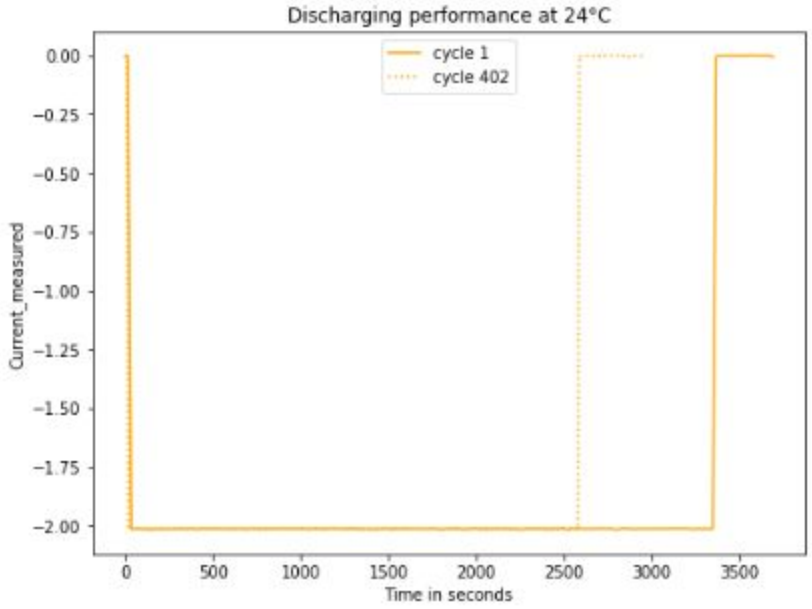
In [26]:
```
indx = 1
for label in charging_labels:
    fig, ax = plt.subplots(1, figsize=(8, 6))
    fig1, ax1 = plt.subplots(1, figsize=(8, 6))
    fig2, ax2 = plt.subplots(1, figsize=(8, 6))

    ax.plot(B0029_discharging[5], B0029_discharging[indx], color='red', label='cycle 1')
    ax.plot(B0029_discharging_93[5], B0029_discharging_93[indx], linestyle=':', color='red', label='cycle
93')
    ax1.plot(B0005_discharging[5], B0005_discharging[indx], color='orange', label='cycle 1')
    ax1.plot(B0005_discharging_402[5], B0005_discharging_402[indx], linestyle=':', color='orange', label
='cycle 402')
    ax2.plot(B0054_discharging[5], B0054_discharging[indx], color='blue', label='cycle 0')
    ax2.plot(B0054_discharging_250[5], B0054_discharging_250[indx], linestyle=':', color='blue', label='cy
cle 250')

    ax.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at 43°C')
    ax1.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at 24°C')
    ax2.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at  4°C')
    ax.legend()
    ax1.legend()
    ax2.legend()

    indx += 1
```

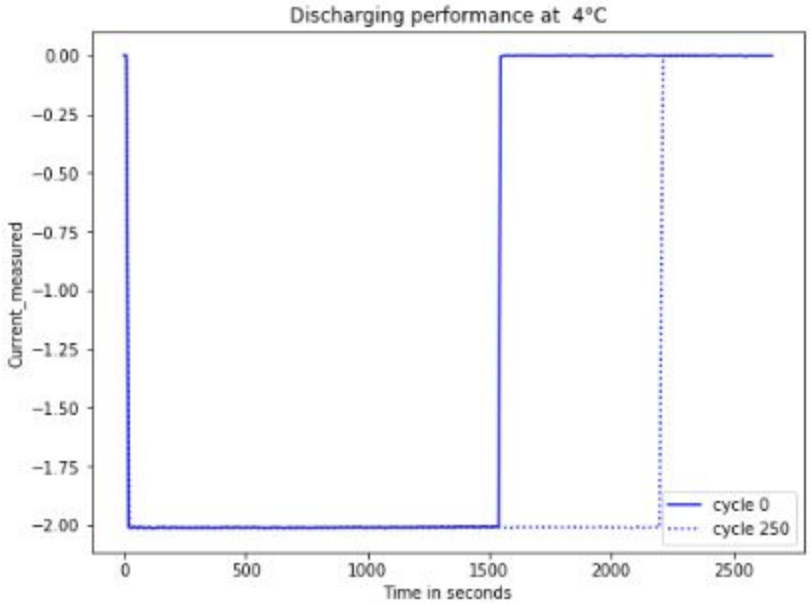
Voltage:


Discharging performance at 43°C
cycle 1
cycle 93
Voltage_measured
Time in seconds

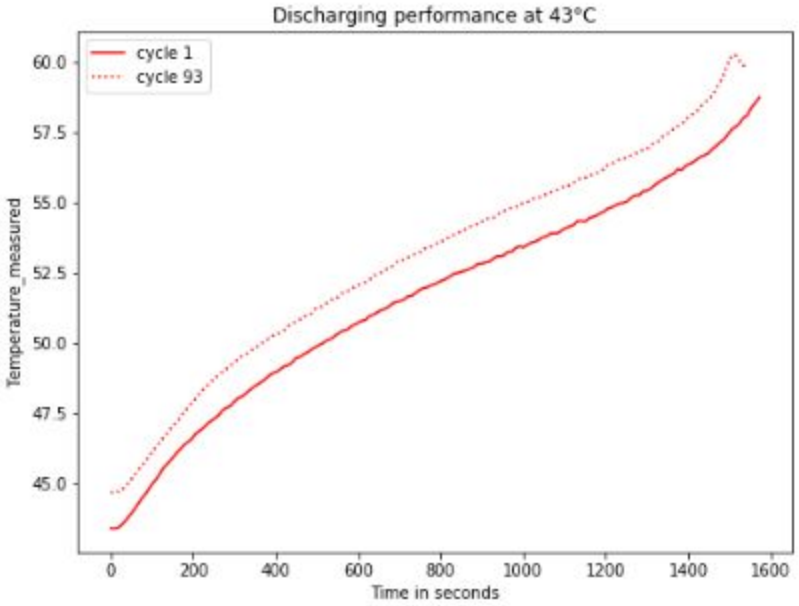

Discharging performance at 24°C
cycle 1
cycle 402
Voltage_measured
Time in seconds

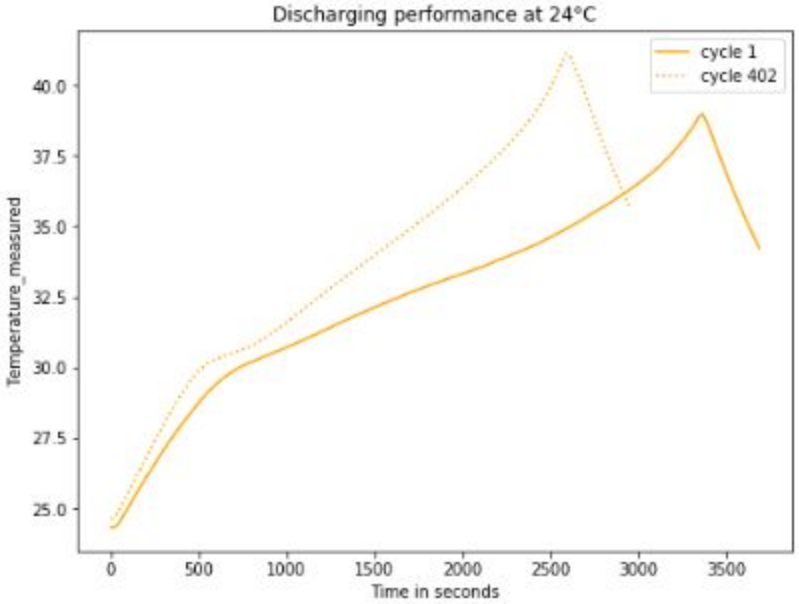
Current:

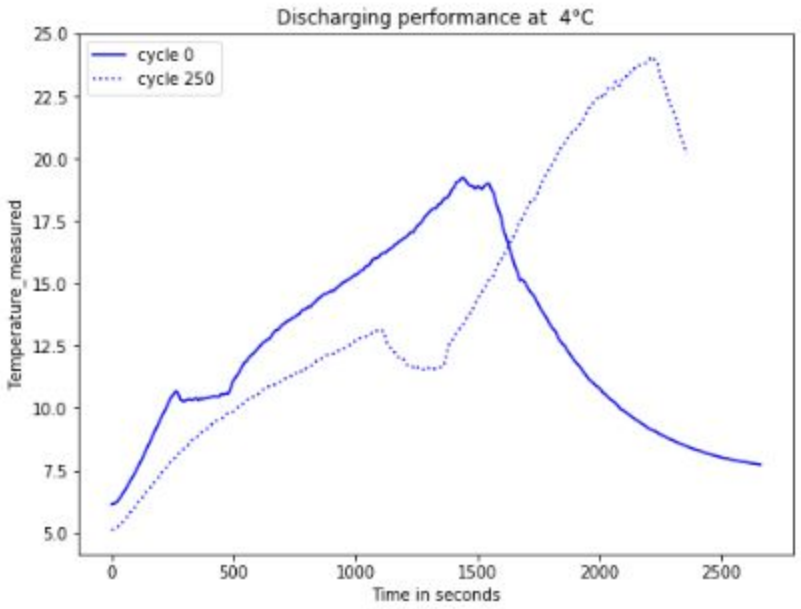




Temperature:

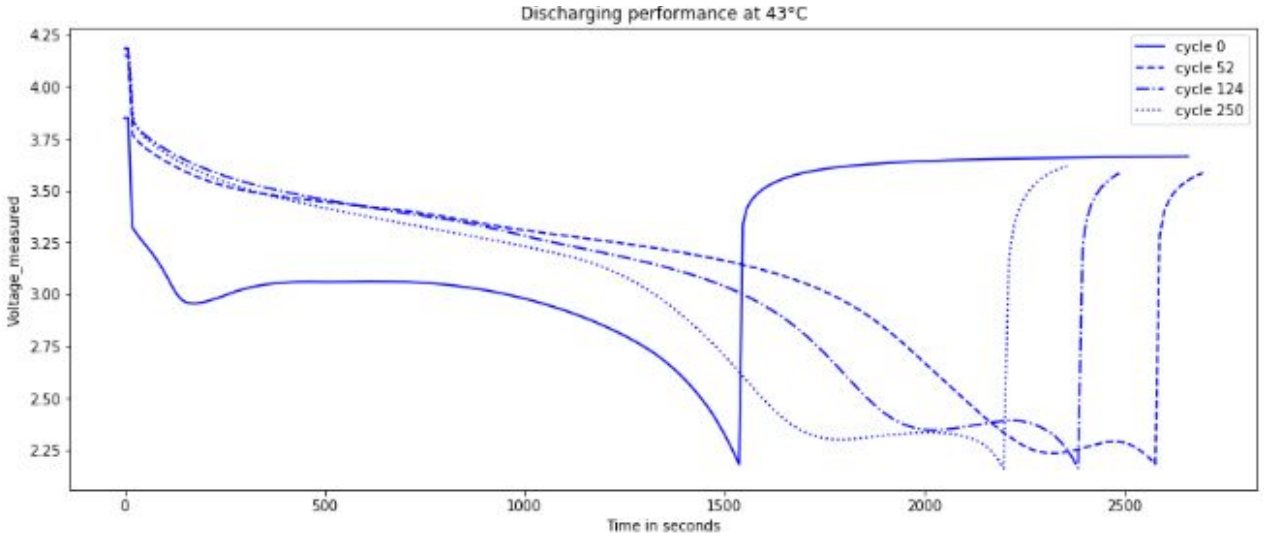

Discharging performance at 43°C
cycle 1
cycle 93
Temperature_measured
60.0
57.5
55.0
52.5
50.0
47.5
45.0
0
200
400
600
800
1000
1200
1400
1600
Time in seconds

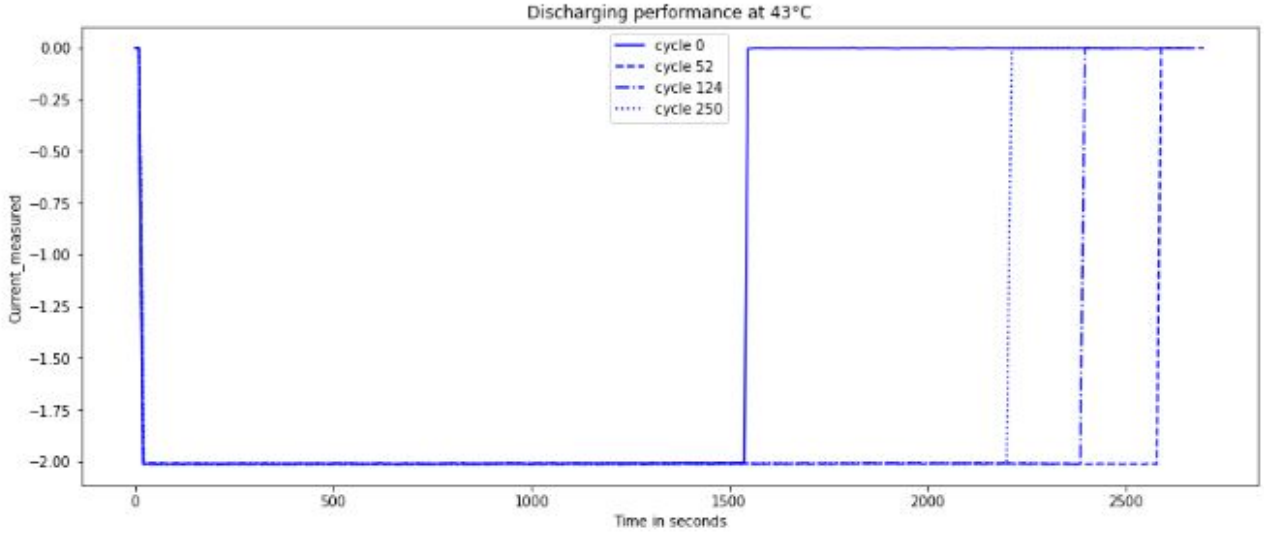

Discharging performance at 24°C
cycle 1
cycle 402
Temperature_measured
40.0
37.5
35.0
32.5
30.0
27.5
25.0
0
500
1000
1500
2000
2500
3000
3500
Time in seconds

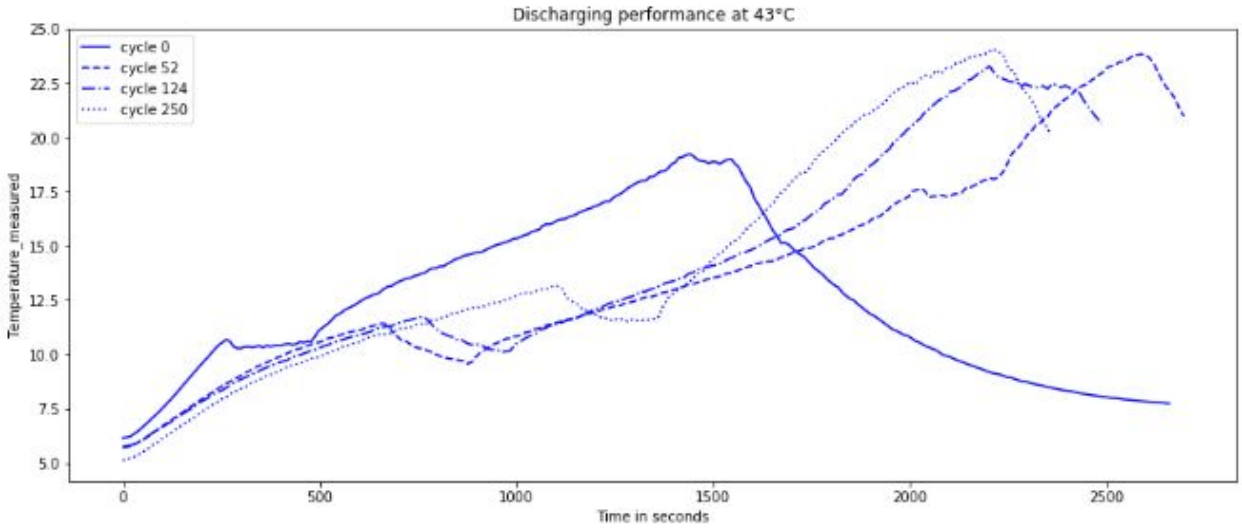


In [27]:
```
B0054_discharging_124 = getDischargingValues(B0054, 124)
B0054_discharging_52 = getDischargingValues(B0054, 52)
```

In [28]:
```
indx = 1
for label in charging_labels:
    fig, ax = plt.subplots(1, figsize=(15, 6))

    ax.plot(B0054_discharging[5], B0054_discharging[indx], color='blue', label='cycle 0')
    ax.plot(B0054_discharging_52[5], B0054_discharging_52[indx], linestyle='--', color='blue', label='cycl
e 52')
    ax.plot(B0054_discharging_124[5], B0054_discharging_124[indx], linestyle='-.', color='blue', label='cy
cle 124')
    ax.plot(B0054_discharging_250[5], B0054_discharging_250[indx], linestyle=':', color='blue', label='cyc
le 250')

    ax.set(xlabel='Time in seconds', ylabel=label, title='Discharging performance at 43°C')
    ax.legend()

    indx += 1
```


Discharging performance at 43°C
cycle 0
cycle 52
cycle 124
cycle 250
Voltage_measured
Time in seconds


Discharging performance at 43°C
cycle 0
cycle 52
cycle 124
cycle 250
Current_measured
Time in seconds


Discharging performance at 43°C
cycle 0
cycle 52
cycle 124
cycle 250
Temperature_measured
Time in seconds

In [29]:
```
dfB0005 = getDataframe(B0005)
```

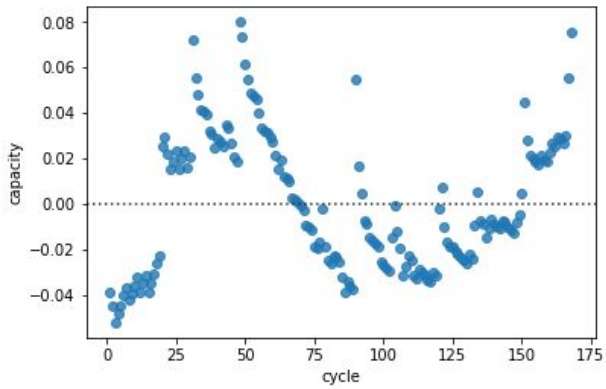
In [30]:
```
dfB0005.head(5)
```

Out[30]:

	cycle	capacity	max_discharge_temp	max_charge_temp
0	1	1.856487	38.982181	27.445134
1	2	1.846327	39.033398	29.341949
2	3	1.835349	38.818797	29.553301
3	4	1.835263	38.762305	29.456340
4	5	1.834646	38.665393	29.481334

In [31]:
```
sns.residplot(dfB0005['cycle'], dfB0005['capacity'])
```

Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3148a3b160>
```



In [32]:
```
dfB0006 = getDataframe(B0006)
dfB0007 = getDataframe(B0007)
dfB0018 = getDataframe(B0018)
```

Model building:

In [33]:
```
dfB0006.head(10)
```

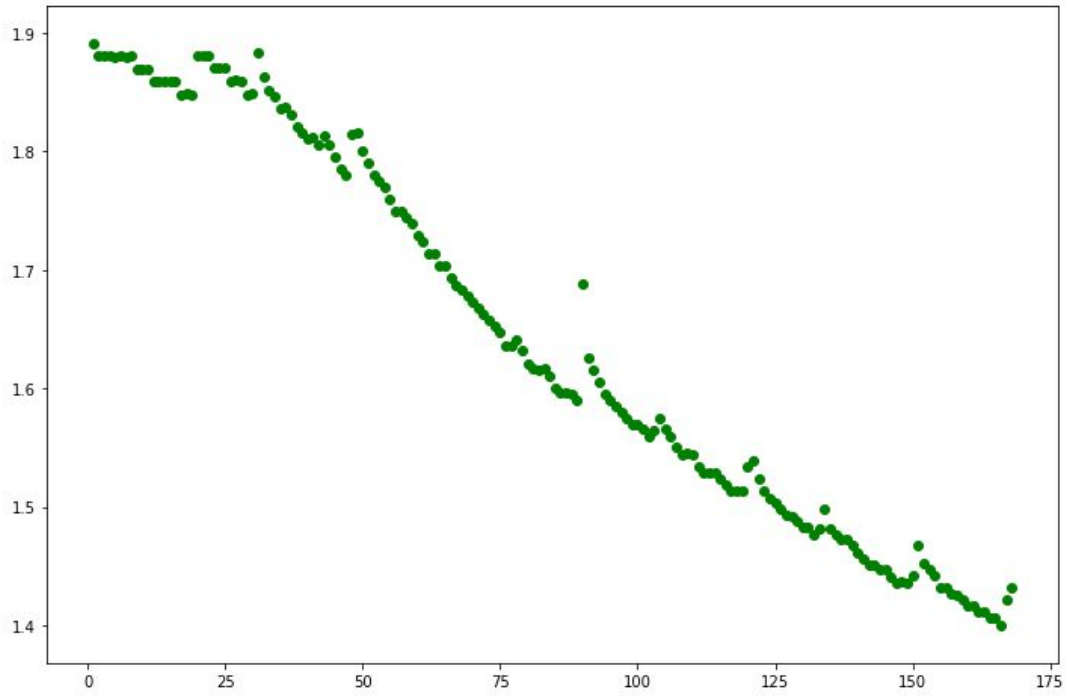
Out[33]:

	cycle	capacity	max_discharge_temp	max_charge_temp
0	1	2.035338	39.162987	27.355870
1	2	2.025140	39.246203	31.874973
2	3	2.013326	38.999202	32.149173
3	4	2.013285	38.843628	32.017074
4	5	2.000528	38.977989	31.921070
5	6	2.013899	38.839899	32.032402
6	7	2.013101	39.046108	32.002633
7	8	1.968790	38.875075	32.204361
8	9	1.968166	38.726054	32.089091
9	10	1.957231	38.986297	31.997731

In [197]:
```
X = dfB0007['cycle']
Y = dfB0007['capacity']
```

In [198]:
```
fig, ax = plt.subplots(1, figsize=(12, 8))

ax.scatter(X, Y, color='green', label='Battery')
```

Out[198]: <matplotlib.collections.PathCollection at 0x7f31480c5438>



In [199]:
```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.20, shuffle=False)
```

In [200]:
```
lst_x, lst_y = rollingAverage(X_train, y_train)
d = {'X_train':X_train.values,'y_train':y_train.values}
d = pd.DataFrame(d)
d = d[~d['X_train'].isin(lst_x)]
```

In [201]:
```
d = {'X_train':X_train.values,'y_train':y_train.values}
```

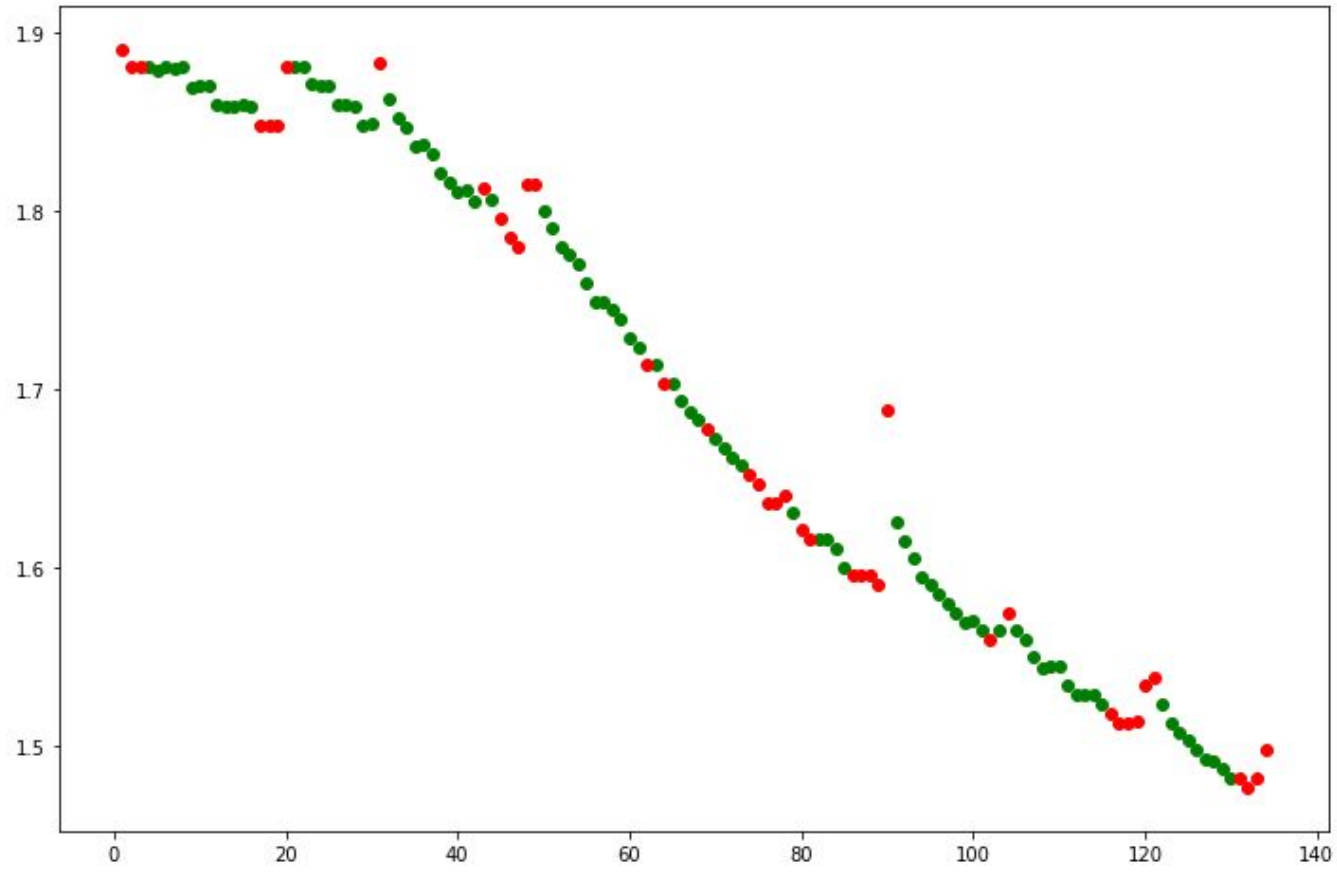
In [202]:
```
d = pd.DataFrame(d)
```

In [203]:
```
d = d[~d['X_train'].isin(lst_x)]
```

In [204]:
```
X_train = d['X_train']
y_train = d['y_train']
```

In [205]:
```
fig, ax = plt.subplots(1, figsize=(12, 8))

ax.scatter(X_train, y_train, color='green', label='Battery')
ax.scatter(lst_x, lst_y, color='red', label='Battery')
```

Out[205]:
```
<matplotlib.collections.PathCollection at 0x7f31480a32e8>
```



In [206]:
```
X_train = X_train.values.reshape(-1, 1)
y_train = y_train.values.reshape(-1, 1)
```

In [207]:
```
best_svr = SVR(C=20, epsilon=0.0001, gamma=0.00001, cache_size=200,
  kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

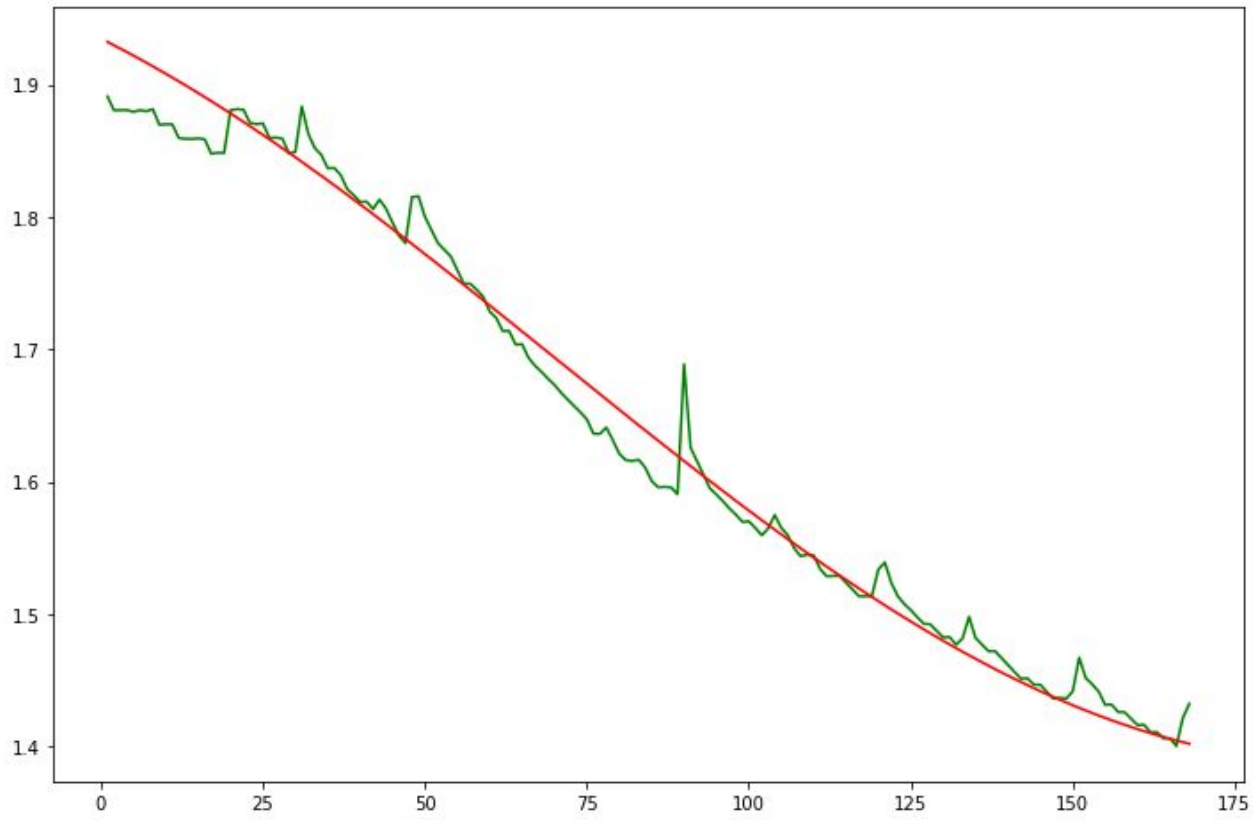
In [208]:
```
best_svr.fit(X_train,y_train)
```

Out[208]:
```
SVR(C=20, cache_size=200, coef0=0.0, degree=3, epsilon=0.0001, gamma=1e-05,
  kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

In [209]:
```
y_pred = best_svr.predict(X.values.reshape(-1, 1))
```

In [210]:
```
fig, ax = plt.subplots(1, figsize=(12, 8))

ax.plot(X, Y, color='green', label='Battery')
ax.plot(X, y_pred, color='red', label='Battery')
```

Out[210]:
```
[<matplotlib.lines.Line2D at 0x7f31385a7d30>]
```



In [181]:
```
X = dfB0018['cycle']
Y = dfB0018['capacity']
```

In [190]:
```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.10, shuffle=False)
```

In [191]:
```
lst_x, lst_y = rollingAverage(X_train, y_train)
d = {'X_train':X_train.values,'y_train':y_train.values}
d = pd.DataFrame(d)
d = d[~d['X_train'].isin(lst_x)]
X_train = d['X_train']
y_train = d['y_train']
```

In [192]:
```
X_train = X_train.values.reshape(-1, 1)
y_train = y_train.values.reshape(-1, 1)
```

In [193]:
```
best_svr = SVR(C=20, epsilon=0.0001, gamma=0.00001, cache_size=200,
  kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

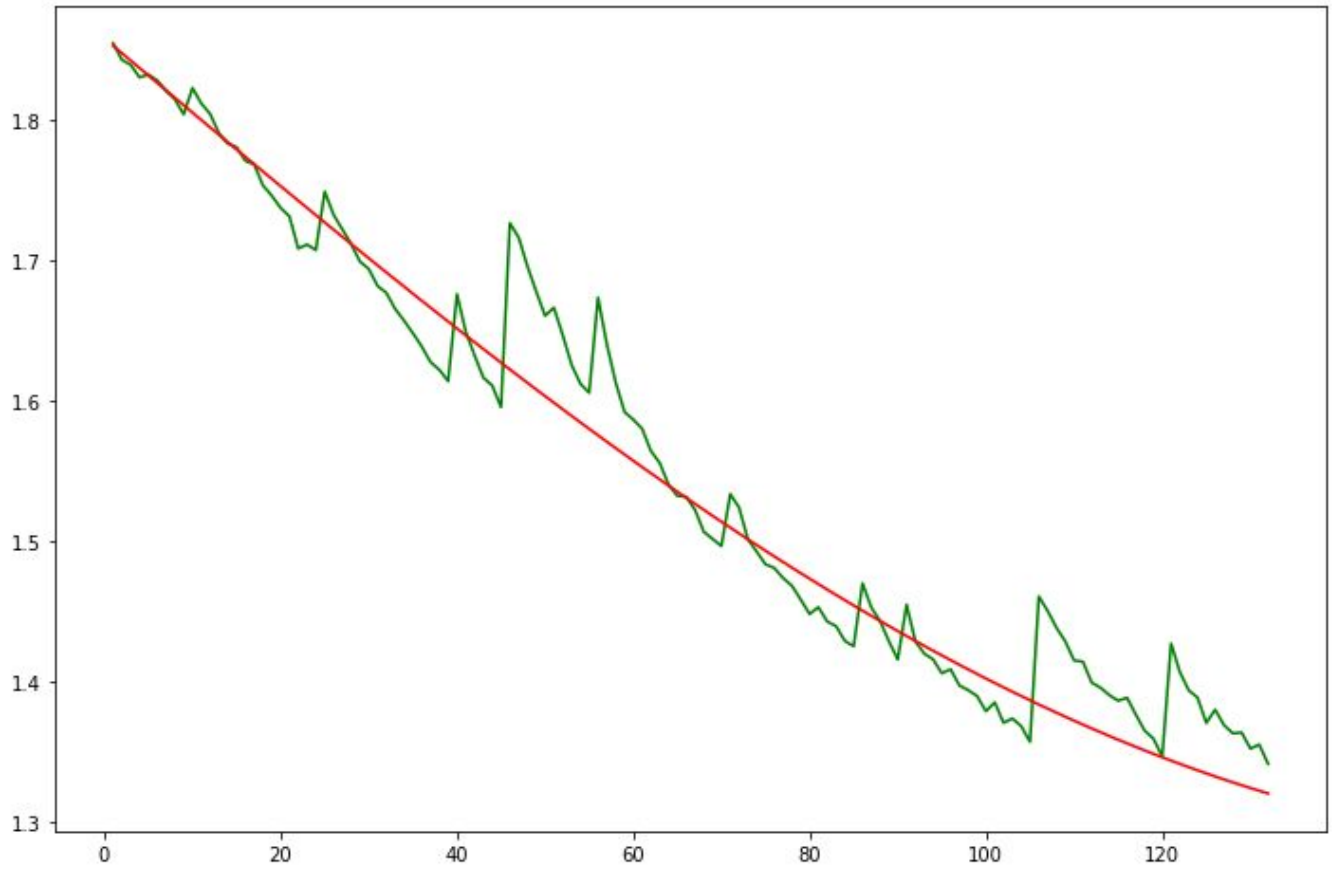
In [194]:
```
best_svr.fit(X_train,y_train)
```

Out[194]:
```
SVR(C=20, cache_size=200, coef0=0.0, degree=3, epsilon=0.0001, gamma=1e-05,
  kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

In [195]:
```
y_pred = best_svr.predict(X.values.reshape(-1, 1))
```

In [196]:
```
fig, ax = plt.subplots(1, figsize=(12, 8))

ax.plot(X, Y, color='green', label='Battery')
ax.plot(X, y_pred, color='red', label='Battery')
```

Out[196]:
```
[<matplotlib.lines.Line2D at 0x7f3148164e10>]
```


1.8
1.7
1.6
1.5
1.4
1.3
0
20
40
60
80
100
120