DX e-learning 公開講座
（リチュームイオン電池劣化試験装置）

DXを導入したリチュームイオン電池劣化試験装置の作成法を解説する自習用講座です

・・・この講座を理解することにより以下のことが行えるようになると期待しています・・・

(1)制御用ラズベリーパイと計測用PCの遠隔操作

(2)計測の自動化（デジタル化）

(3)計測データの自動データ処理（一種のRPA）

(4)以上から得られたデータから、与えられた充放電条件での寿命予測（データの価値化）

1　リチュームイオン電池劣化試験装置の部品等一覧

No	品名	税抜価格	備考
1	Raspberry Piスターターセット（ラズベリーパイ3B＋）	15,900	本体だけなら5,000円台
2	10.1インチタブレットPC	29,900	計測用
3	Raspberry用リレー4回路	2,390	2回路でもOK、値段の差小
4	高精度アナログ入力ターミナルUSB対応(AI-1608AY-USB)	50,310	PCとあわせて電圧計測、EXCELデータで記録
5	ミニインキュベータ	50,900	
6	直流安定化電源AD8723D	13,900	
7	15.6型モバイルディスプレイ	29,900	遠隔操作するなら不要
8	リチュームイオン電池	829	
9	有線超薄型ミニキーボード	2,690	遠隔操作するなら不要
10	マウス(2個)	1,420	遠隔操作するなら不要
11	小型ブレッドボード	569	
12	金属皮膜抵抗47Ω、0.4W	469	一袋25個入りの値段
13	ブレッドボード用ジャンプワイヤーセット	546	
14	配線コード30m巻（赤と黒）	1,158	
15	PHコネクターベース付ポスト	89	一袋10個入りの値段
16	スタンダードシャーレ	340	

合計　201,310円

装置全体の写真

上面部の写真

この他に遠隔操作用のPCが必要です。

2. 説明

2.1　制御用ラズベリーパイと計測用PCのWi-Fiを用いた遠隔操作

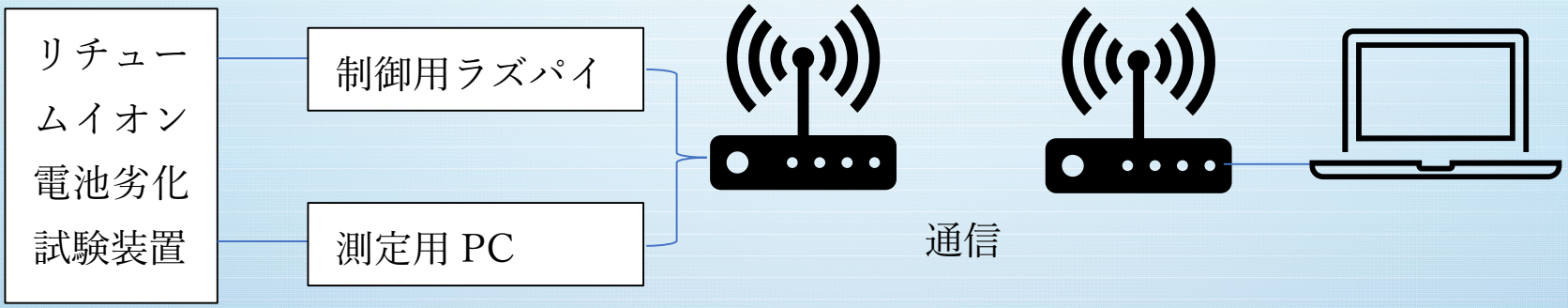


図　遠隔操作システム（「RealVNC」が提供するVNC Connect（５台までフリー）使用）

ラズベリーパイおよびPCのリモートコントロール
（「RealVNC」が提供するVNC Connect（５台までフリー）使用）

1. 前提
　遠隔操作するデバイスに「VNC Viewer」、遠隔操作されるラズパイおよびPCに「VNC Server」がインストールされているとする（インストールの仕方は、（「VNC Connect」簡単で手軽なリモートコントロールソフト）で検索すると解説があります）。

2. ラズパイ、PCのPCによるリモートコントロール
(1)接続元PCのウインドーズスタートからRealVNCをクリック、表示されたVNCViewerをクリック。
(2)表示された入力枠に、VNC Server addressを入力（今回は、ラズパイはIPアドレス、PCは接続先のPCのIDを入力としている）
(3)ラズパイはその後、IDとPWを聞いてくるので入力する。PCはPWのみ聞いてきた。

3. 参考
(1) ラズパイのIPアドレスの確認方法
　無線のマークにマウスをもっていくと表示される。
(1) PCのIDの確認
　「windous」キーを押しながらｘを入力、Windows PowerShellをクリック、入力プロンプト（＊＊＊＊＞）が出たら、hostnameと入力すると表示される。（IPアドレスはipconfigと入力すると表示される（IPv4のところ））。

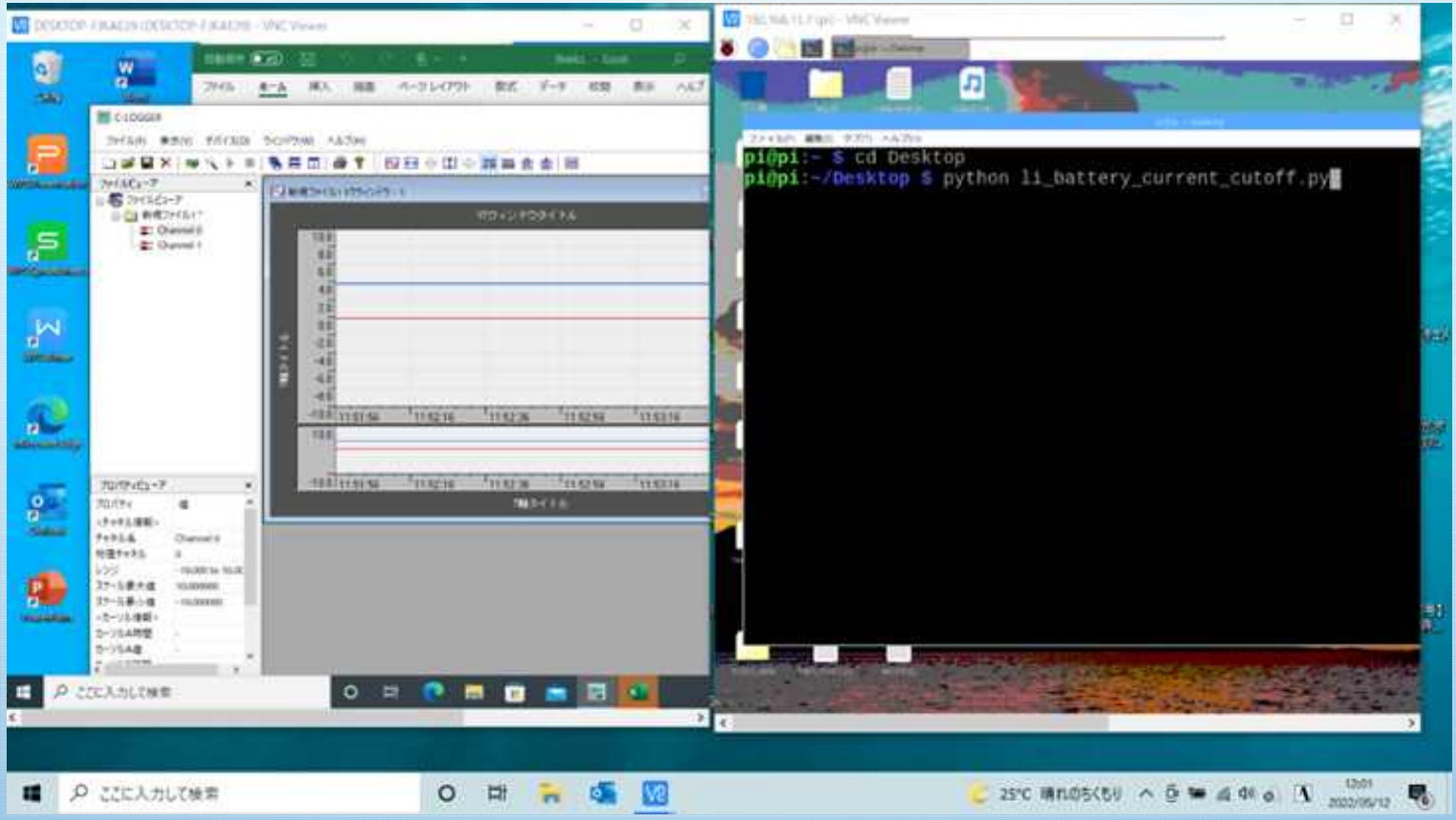


図　遠隔操作するPCの画面（左：遠隔操作されているPC画面、右：ラズパイ画面）

（マウスをタスクバーの所におき右クリック、ウインドウを左右に並べて表示を選択）

2.2　自動化したリチュームイオン電池の充放電とその時の電圧・電流測定（デジタル化）

ラズパイによりリチュームイオン電池の充放電を自動制御し（プログラムli_battery_test.py）、電圧（V1、V2）を「高精度アナログ入力ターミナル（ソフト付き）」とPCで自動計測する。

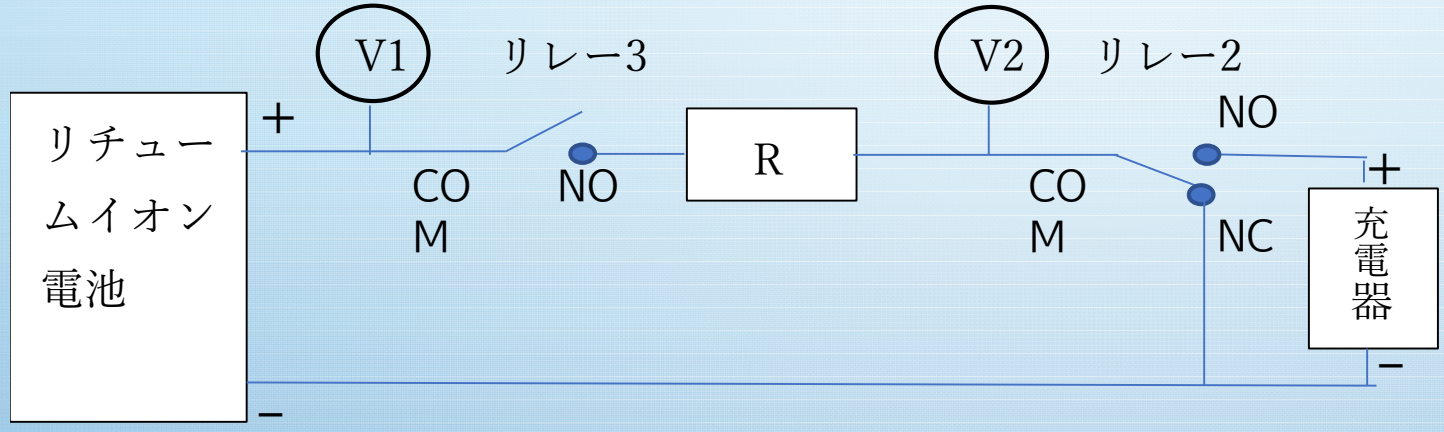


図　充放電測定システム

表　充放電自動測定結果の例

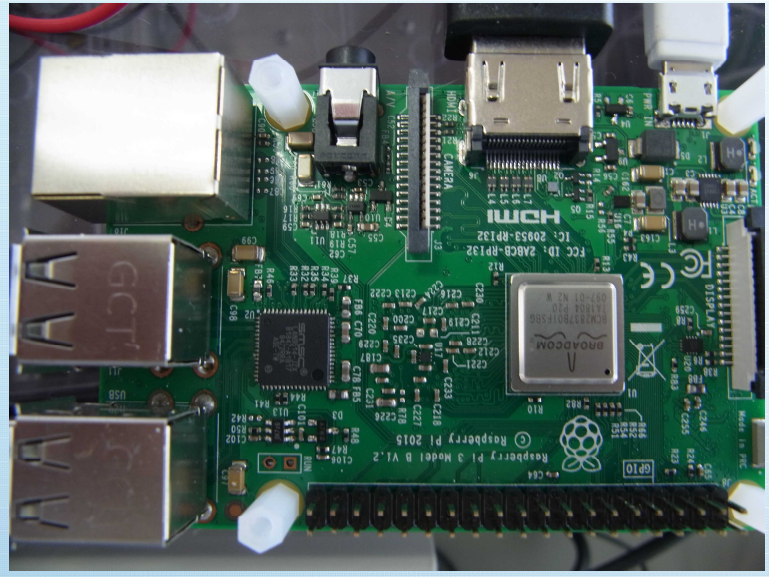
時間	V1	V2
2022/04/07 16:50:07'000"000	7.146606	7.145081
2022/04/07 16:50:17'000"000	3.651123	0.067749
2022/04/07 16:50:27'000"000	3.518982	0.065613
2022/04/07 16:50:37'000"000	3.421021	0.064087
2022/04/07 16:50:47'000"000	3.367615	0.062561
2022/04/07 16:50:57'000"000	3.337708	0.061646
2022/04/07 16:51:07'000"000	3.320923	0.061951
2022/04/07 16:51:17'000"000	3.306274	0.061035

ラズベリーパイ3BのOSインストール

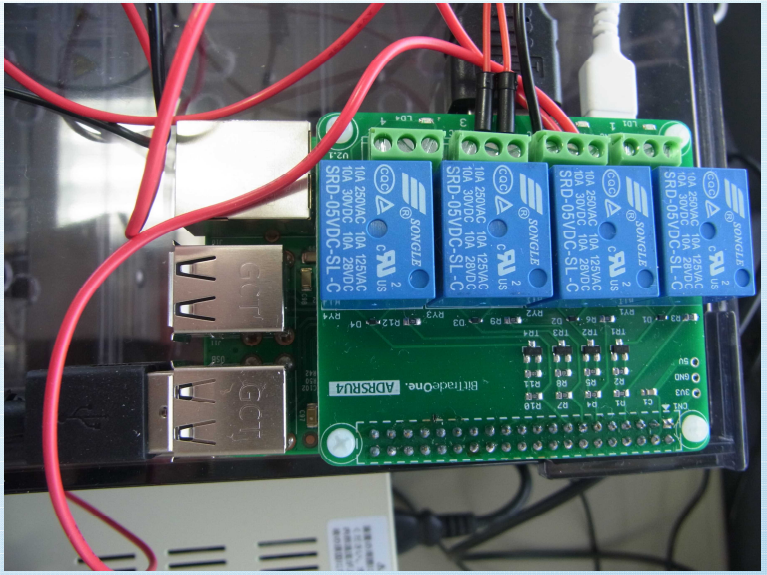
ラズベリーパイではOSをmicroSD（16GB以上推奨、FAT32形式でフォーマット済み）に書き込んで使います。ここではRaspberry Pi Imagerを使ってOS　Raspbian（Linuxの一種）をmicroSDに書き込みインストールする方法を紹介します(2021年 3月5日時点での方法)。なお、microSDカードはPCに挿入されているものとします。

(1)PCでwww.raspberrypi.orgを開き、上方のSoftwareをクリックします。
(2)表示された画面の左側のRaspberry Pi imagerをクリックし、表示された画面のDownload for Windowsをクリックするとimager_1.5.exeがダウンロードされます。
(3)ダウンロードされたimager_1.5.exeというファイルをダブルクリックし、Installをクリックします。終了したら、Run Raspberry Pi Imagerの前の□にチェックが入っていることを確認し、下のFinishをクリックします。
(4)Raspberry Pi Imagerが起動したら、CHOOSE OSをクリックしまう。
(5)次の画面が表示されたら、Raspberry Pi OS(32bit)をクリックします。
(6)その後、（4）と同じ画面が表示されますので、ここでCHOOSE SD CARDをクリックし、画面に挿入したSDカードの容量等が表示されます。これををクリックします。
(7)表示された画面でWRITEをクリックし、YESをクリックするとインストールが始まります。
(8)インストールが終了するとYou can remove the SD card from the readerと表示されますので、SDカードを取り出し、CONTINUEをクリックし、次に右上の×をクリックしてSDへの書き込みは終了です。
(9)書き込みが済んだSDをラズベリーパイに挿入、モニター、キーボード、マウスを接続し、USB電源（これがスイッチとなります）を接続すればラズベリーパイはON状態となります。ここで、国名等の初期設定を聞いてくるので設定してください。終了は画面右上のメニューアイコン（ラズベリーのマーク）をクリックし、ログアウトをクリック、出てきた表示のShutdownをクリックすれば電源がoffとなります。なお、最初の時に、国名等を聞いてくるのでそのセットアップをしてください（しないとデフォルトの値が入ります）。

制御について



ラズベリーパイ



ラズベリーパイにraspberry用リレー回路をセット

図　ラズベリーパイとRaspberry用リレー4回路をセットした写真

今回は、リレー2とリレー3を使用しています（これを使う理由は特にありません）。6枚目のスライドのように配線します。抵抗は25Ωを今回は使用しています。これで制御系の配線は終了です。

li_battery_test.py

```
# -*- coding: utf-8 -*-
# li_ion battery charge discharge test 500回
# li_battery_test.py
#GPIO用のモジュールをインポート
import RPi.GPIO as GPIO
# timeライブラリーからsleepをインポート
from time import sleep
#GPIOのモードをピン番号で指定するGPIO.BOARとする
GPIO.setmode(GPIO.BOARD)
#ピン番号13を出力番号としそこの電圧をhigh（放電ができるようになる）
GPIO.setup(13, GPIO.OUT)
GPIO.output(13, GPIO.HIGH)
# NT(予定は600)秒間放電、7*NT/3秒間充電(5V)をN回繰り返す
N = 500
NT = 600
NT2=NT/100
NT3=7*NT/3
GPIO.setup(11, GPIO.OUT)
sleep(NT2)
for i in range(N):
# 充電開始
    GPIO.output(11, GPIO.HIGH)
    sleep(NT3)
#  放電開始
    GPIO.output(11, GPIO.LOW)
    sleep(NT)sleep(4)
GPIO.cleanup()
```

動作のさせ方

ラズベリーパイにこのプログラムをおき（ここではデスクトップ）、
ラズベリーパイのLX Terminalを起動する（デスクトップ画面左上のLX Terminalのアイコンをクリック）。次にCDディレクトリでDESKTOP
に移ります。そこで、
python
li_battery_test.py
と入力しリターンすれば装置が動きます。パワーポイントの5枚目参照

参考
リレー番号1がピン番号7、リレー番号2がピン番号11、リレー番号3がピン番号13、リレー番号4がピン番号15です。

測定について

最初に、高精度アナログ入力ターミナルと測定用PCをUSBケーブルで繋いでください。
次に、測定用PCに、高精度アナログ入力ターミナルUSB対応(AI-1608AY-USB)に添付のファーストステップガイドに従って、ソフトウェア（付属のCDに入っている）をインストールしてください。その後初期設定をして下さい。
続いて、高精度アナログ入力ターミナルのアナログ入力と測定する電圧箇所を繋いで下さい（6枚目のパワーポイント参照）

以上で測定システムは完成です。

動かし方は、マニュアル（C-LOGGER%20ユーザーズガイドで検索）
を見て下さい。長いマニュアルですが、全部読まなくても大丈夫です。

測定について・・・つづき

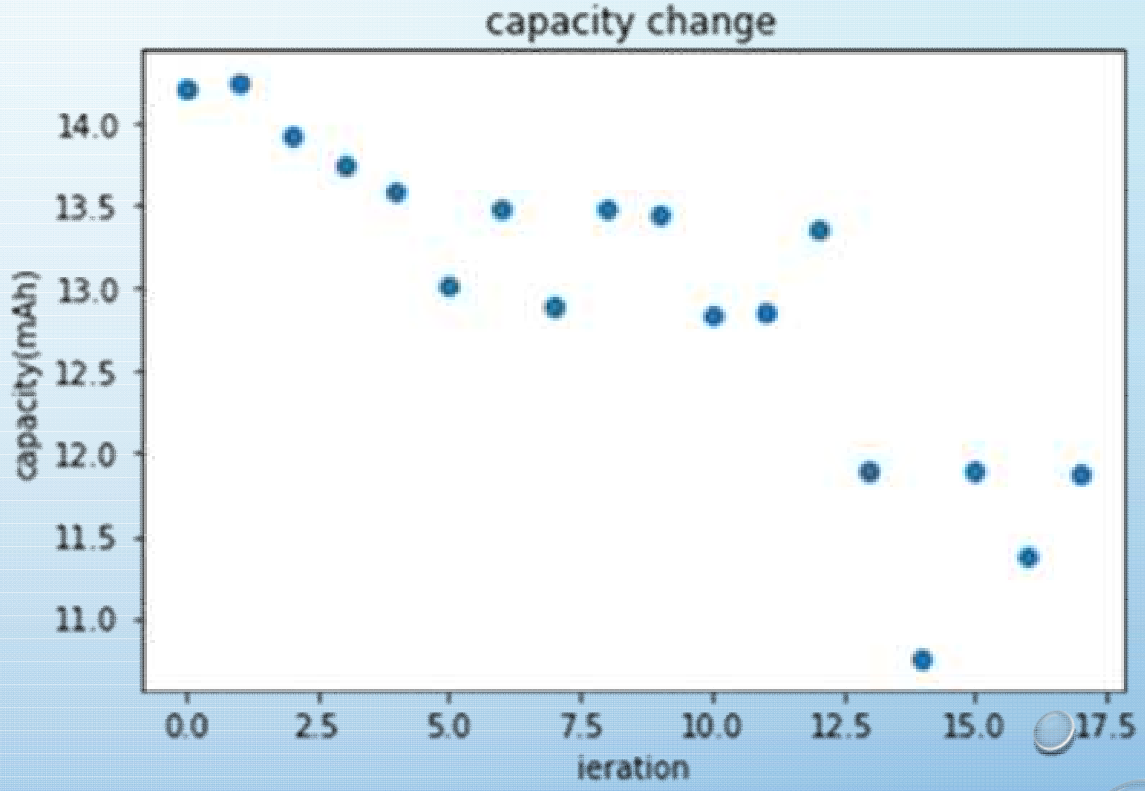
測定結果は、excelファイルに記録できます。その例が下表です。ただし、1行目はわかりやすくするため追加しています。

年/月/日/時刻	V1	V2
2022/04/07 16:50:07'000″000	7.146606	7.145081
2022/04/07 16:50:17'000″000	3.651123	0.067749
2022/04/07 16:50:27'000″000	3.518982	0.065613
2022/04/07 16:50:37'000″000	3.421021	0.064087
2022/04/07 16:50:47'000″000	3.367615	0.062561
2022/04/07 16:50:57'000″000	3.337708	0.061646
2022/04/07 16:51:07'000″000	3.320923	0.061951
2022/04/07 16:51:17'000″000	3.306274	0.061035

2.3　上記デジタルデータから放電容量の充放電回数依存性を求める自動データ処理（一種のRPA）

測定値を基に放電容量の充放電回数依存性
を「RPA」（プログラム
li_battery_data_processing.py）
で求める。

右図は500回以上充放電（寿命を過ぎている）
を行った後の例です。
参考　新品の電池の容量は40mAH



li_battery_data_processing.py

```
# -*- coding: utf-8 -*-
#Li_ion_battery劣化データのRPAもどき処理
# li_battery_data_processing.py

import openpyxl as excel
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

#book = excel.workbook()
book =
excel.load_workbook("4_7_504_1_7.xlsx")

d = pd.read_excel("4_7_504_1_7.xlsx")
#a = d[B].values

#a = book[B].values
num = len(d)

#num = len(book)
```

```
sheet = book.worksheets[0]
#output = book.worksheets[1]

i_ignore = 2
i_discharge = 20
i_charge = 20

iter = (num-i_ignore)//(i_discharge+i_charge)



x = []
y = []

for j in range(iter):
    sum1 = 0.0
    sum2 = 0.0
    i_plus = j*(i_discharge+i_charge)+i_ignore
```

li_battery_data_processing.py　のつづき

```
    for i in range(i_discharge):
        num = i + i_plus
        B1 = "B" + str(num)
        C1 = "C" + str(num)
        Bval = sheet[B1]
        Cval = sheet[C1]
        sum1 = sum1 + Bval.value
        sum2 = sum2 + Cval.value


    capacity = 10.0*4.0/50.0*(sum1-
sum2)*1000.0/60.0/60.0

    y.append(capacity)
    x.append(j)

np.savetxt("out.csv",y,delimiter=",")
```

```
fig = plt.figure()
ax = fig.add_subplot(1,1,1)
ax.scatter(x,y)
ax.set_xlabel('ieration')
ax.set_ylabel('capacity(mAh)')
ax.set_title('capacity change')

fig.show()

Y = np.array(y.T)
Y.to_csv('./out.csv'
```

動作のさせ方

X Terminalでディレクトリーを
DESKTOPに移します。そこで、
python
li_battery_data_processing.py
と入力しリターンすれば
プログラムが動きます。計算結果の
例がパワーポイントの12枚目の図で
す。

2.4　電流遮断試験と電圧測定の自動化

電池の簡単な等価回路は一般的に、起電力と抵抗および抵抗とコンデンサーの並列回を直列接続した回路として表現される。この等価回路の素子の値を求める方法の一つが電流遮断試験（制御プログラムli_battery_current_cutoff.py）である。これは、放電中に電流を遮断し（下図のようにリレー3をOFFとする）、その前後の電圧（下図のV1）の時間変化を測定するものである。なお、劣化が進むと等価回路の抵抗値とコンデンサー容量に変化が生じるため、劣化の目安を得ることができる。

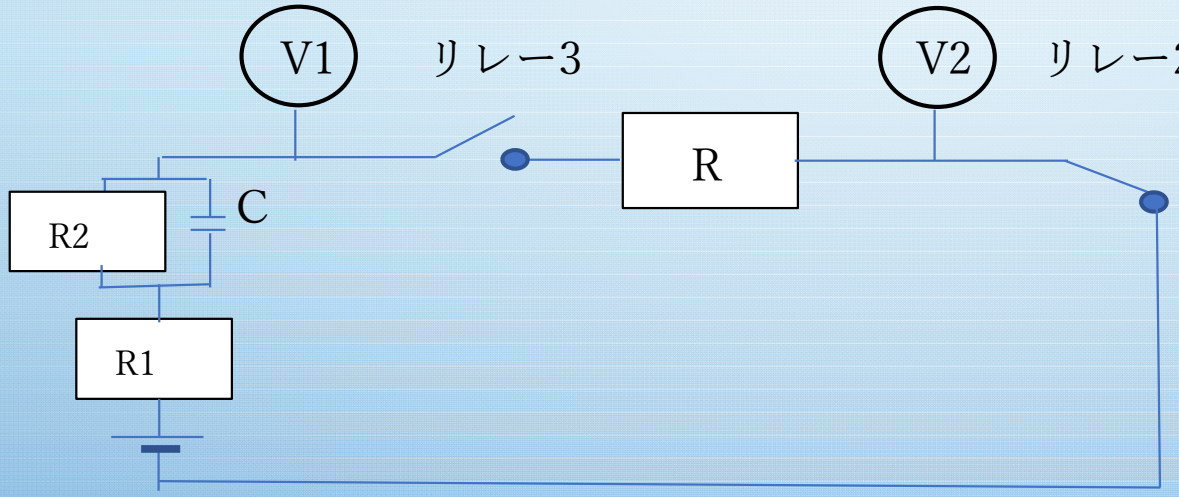


図　電流遮断実験の測定図

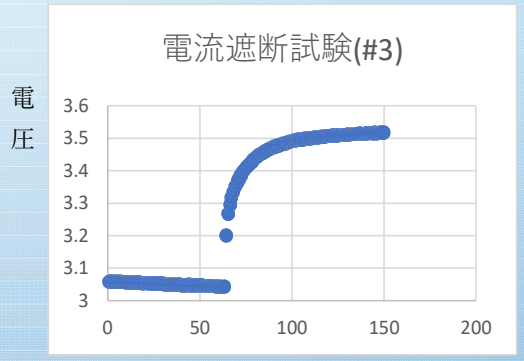


図　測定結果

電流遮断試験制御プログラム
li_battery_current_cutoff.py

```
# -*- coding: utf-8 -*-
# li_ion battery current cutoff measurement
# li_battery_current_cutoff.py
#GPIO用のモジュールをインポート
import RPi.GPIO as GPIO
# timeライブラリーからsleepをインポート
from time import sleep
#GPIOのモードをピン番号で指定するGPIO.BOARとする
GPIO.setmode(GPIO.BOARD)
# NT(予定は10)秒間充電、NT/2秒間放電しその後電流遮
断NT = 10
NT2=NT/2
# 充電開始
GPIO.setup(11, GPIO.OUT)
GPIO.output(11, GPIO.HIGH)
GPIO.setup(13, GPIO.OUT)
GPIO.output(13, GPIO.HIGH)sleep(NT)
#  放電開始
GPIO.output(11, GPIO.LOW)sleep(NT2)
# 測定開始
GPIO.output(13, GPIO.LOW)
sleep(4)
GPIO.cleanup()
```

動作のさせ方

X Terminalでディレクトリーを
DESKTOPに移します。そこで、
python
li_battery_current_cutoff.py
と入力しリターンすれば
プログラムが動きます。計算結果の
例がパワーポイントの15枚目の図で
す。

2.5　電流遮断試験のデジタルデータからリチュームイオン電池の簡易型等価モデルの各素子（RとC）を求める自動データ処理（一種のRPA、プログラムli_battery_eq_circuit.py）

表　求めた各電池の抵抗値とコンデンサー容量

セル番号	充放電回数	放電容量	R1(Ω)	R2(Ω)	C（F)
#1（古い）	不明	不明	0.94	1.1	0.10
#3（古い）	500 回以上	約 10mAH	0.65	1.3	0.07
#4（新品）	0 回	40mAH？	0.70	0.77	0.08

今回の実験では、R2の値に比較的大きな変化が見られた。

li_battery_eq_circuit.py

```
# -*- coding: utf-8 -*-
# Li_ion_battery等価回路データのRPAもどき処理
# li_battery_eq_circuit.py

import openpyxl as excel
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import math

book =
excel.load_workbook("current_test_3.xlsx")

d = pd.read_excel("current_test_3.xlsx")

num = len(d)
sheet = book.worksheets[0]

R = 12.5
i_ignore = 1000
BB = "B" + str(i_ignore)
BBval = sheet[BB]
```

```
for i in range(i_ignore,num):
    inum = i
    BI = "B" + str(inum)
    Bval = sheet[BI]
    ch = Bval.value-BBval.value

    if ch < 0.1 :
        BBval= Bval

    else :
        VV1 = BBval
        VV2 = Bval
        njump = inum
        break

V1 = VV1.value
V2 = VV2.value

ns = njump + 100
BF ="B" + str(ns)
```

li_battery_eq_circuit.pyのつづき

```
VV3 = sheet[BF]
V3 = VV3.value

current = V1/R

R1R2 = (V3-V1)/current

R2 = (V3-V2)/current
R1 = R1R2 - R2

e = 2.71828
V4 =(V3-V2)/e

for j in range(njump,num):
    BE = "B" + str(j)
    BBE = sheet[BE]

    if (V3-BBE.value) < V4 :
        jtconst = j -njump
        VE = BBE.value
        break
```

```
Tconst = jtconst*0.01
CR2 = Tconst

C = CR2/R2

print("V1= ",V1, "(V)")
print("V2= ",V2, "(V)")
print("V3= ",V3, "(V)")
print("R1= ",R1, "(Ω)")
print("R2= ",R2, "(Ω)")
print("C= ",C, "(F)")
print("CR2=", CR2," time constant")
print("(V3-V2)/(V3-VE)=", (V3-V2)/(V3-VE),"about e (=2.718)")
```

動作のさせ方

X Terminalでディレクトリーを DESKTOPに移します。
そこで、python li_battery_eq_circuit.py
と入力しリターンすればプログラムが動きます。
計算結果をパワーポイントの17枚目の表にまとめました。

2.6　以上から求めたリチュームイオン電池の充放電条件とリチュームイオン電池の

寿命の表から、与えられた充放電条件での寿命予測（データの価値化、プログラム　li_battery_lifespan_regression_prediction.py）

表　学習用データの一部（注意－これは正確なデータではありません）

No	放電時間	充電電圧	負荷抵抗	室温	寿命
1	200	7	12.5	30	300
2	250	7	12.5	30	250
3	300	7	12.5	30	200
4	150	7	12.5	30	350
5	100	7	12.5	30	400

表　推測用データ

No	放電時間	充電電圧	負荷抵抗	室温	寿命
1	210	6.5	10	30	?

AIは322回で寿命が来ると推測した。

li_battery_lifespan_regression_prediction.py

AI開発環境の構築：Jupiter notebookの使い方.docxを参考にPCにanacondaをインストールし、続いてKerasとtensorflowをインストールして下さい。これで完成です。ネット上の無料AI開発環境google colabを使用すること可能です。

temp_data.csvとtest_data.csvを入力としてAIプログラム
li_battery_lifespan_regression_prediction.pyを動かします（ Jupiter notebookの使い方.docx を参考にして下さい）

li_battery_lifespan_regression_prediction.py　の内容は添付ファイルをご覧下さい。

以上です。

なお、プログラムはこのパワーポイントと同じ
場所で入手できます。