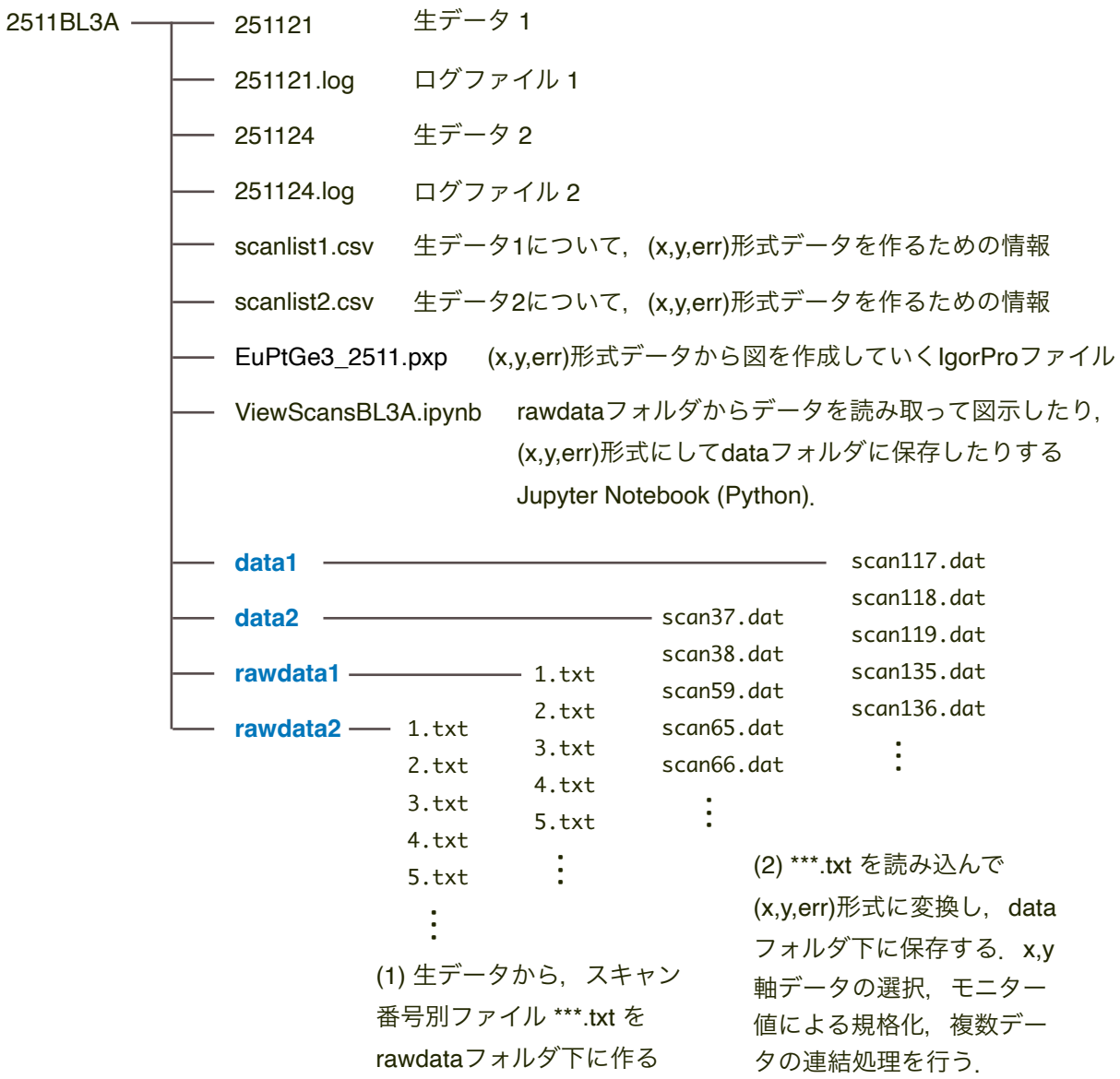
作業フォルダ構成例



(3) 作成した(x,y,err)形式のデータを図示して，測定の状況を確認したり，
様々な解析を行ったりする．IgorProを使うのがお薦め．

(1)(2)はルーチンワークで，測定中に速やかに行うもの．(3)はそれぞれの実験，状況に応じて，試行錯誤し，考えながらの作業が必要．作業工程として(1)(2)と(3)を切り分けることが肝要．

SPECの生データから解析用の(x,y,err)形式データを作る

(1) rawdataファイルの生成

まず，SPECの生データをスキャン番号別のファイル***.txtとして書き出す.
以下の内容のファイルspec2txtを作り，**.txtを作りたいrawdataディレクトリに置く.

```
BEGIN{FLAG=0}
$1=="#S" {FLAG=1; NAME=$2; printf("") > (NAME".txt")};
$0=="" {FLAG=0};
{if(FLAG==1) print $0 >> (NAME".txt")}
{if(FLAG==0) close(NAME".txt")}
```

rawdataディレクトリで次を実行．生データは1つ上にあるとする.

```
>awk -f spec2txt ../150928
```

awkの処理をWindowsでどうやるのかはわかりません（検索するといろいろ方法はあるようです）.
Linux(Mac)ではターミナルからコマンドを打つだけです.

のようなファイルが生成される.
次のデータ処理ではこれを使う.

(2) rawdataファイルを読み込んでデータを表示したり，(x,y,err)形式にしてdataフォルダに保存したりする Jupyter Notebook (Pythonコード)．[ViewScansBL3A.ipynb]

> 以前はこの作業もIgorProで行っていましたが，データのまとめや解析作業のIgorProと重複してしまうので，全く同じ作業をするPythonコードを作り，独立させました．Python やJupyter Notebookのインストールはよくわからないので各自で行って下さい．

(x,y,err)形式データを作るための情報が入った，以下のようなcsvファイルを作成しておく．これをJupyter NotebookのPythonで読みこんで，図示したり，処理して(x,y,err)形式データにしてdataフォルダに保存したりする．

1列目は列の名前．2列目から記入．

	A	B	C	D	E	F	G	H	I	J	K	L
1	scan1	scan2	scan3	scan4	scan5	scan6	scan7	xcol	moncol	datcol	mon1s	bindx
2	234	0	0	0	0	0	0	1	-2	-1	60000	1.00E-03
3	235	0	0	0	0	0	0	1	-2	-1	60000	1.00E-03
4	236	0	0	0	0	0	0	1	-2	-1	60000	1.00E-03
5	237	0	0	0	0	0	0	1	-2	-1	60000	1.00E-03
6												
7	258	259	0	0	0	0	0	1	-2	-1	55800	1.00E-03
8	264	265	266	0	0	0	0	1	-2	-1	75000	1.00E-03
9	267	268	269	0	0	0	0	1	-2	-1	75000	1.00E-03
10	270	271	272	0	0	0	0	1	-2	-1	75000	1.00E-03
11	273	274	275	0	0	0	0	1	-2	-1	75000	1.00E-03
12	276	277	278	0	0	0	0	1	-2	-1	75000	1.00E-03
13	279	280	281	0	0	0	0	1	-2	-1	75000	1.00E-03

A~G: スキャン番号．複数のスキャンを連結させるときはB,C, … にも記入．0で読み込み終了．
例えば8行目では 264, 265, 266 の3つのスキャンを連結して，一つの(x,y,err)形式データ Scan264.dat として保存する．
H: x値にする列番号（左から何列目か）．普通は 1 だが，L-scanなどでは 3 になる．
I: モニター値にする列番号（右から何列目か）．マイナスをつける．普通は -2．
J: y値にする列番号（右から何列目か）．マイナスをつける．普通は -1．
K: 単位をcpsで揃えるために使うモニター値．測定時の1秒間のおよそのモニター値．入射ビームの条件が変わるとここも変わる．同じ条件のときは統一した数値にしておき，細かい変動を規格化する．
L: 複数スキャンを連結するとき，x値の差がこの値以下なら同じx値とみなして積算する．例えば，7行目のscan258のデータでx=2.34512あり，scan259のデータでx=2.34511があったとすると，記録上の数値は違うが，実際は同一点での測定とみなしてカウント数を積算する．

ViewScansBL3A.ipynb で

```
# 最初に実行
import numpy as np
import os
import csv
import matplotlib.pyplot as plt
import time
from IPython.display import display, clear_output

raw_dir = "rawdata3"    # rawdataフォルダ名          ←
save_dir = "data3"      # data 保存フォルダ名        ←
os.makedirs(save_dir, exist_ok=True)
```

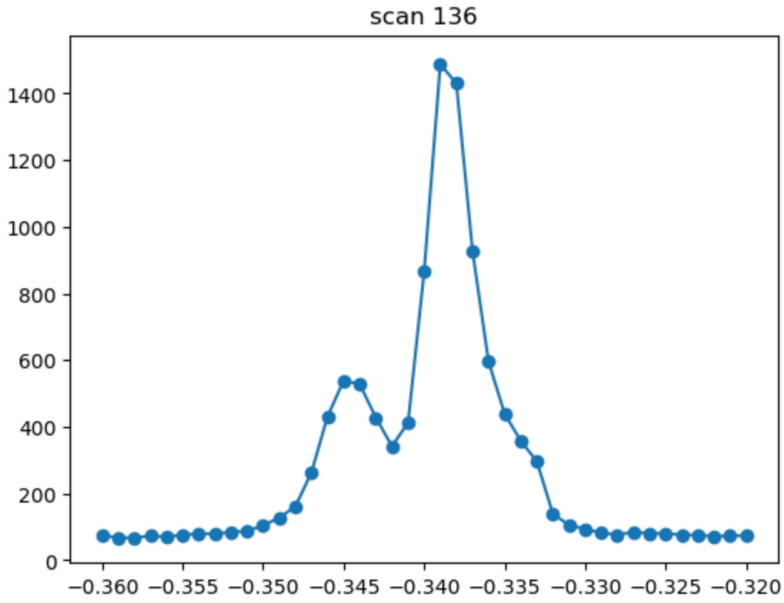
まずこれを実行．

ここも状況に合わせて記入しておく．

csvファイルの何行目から何行目までを処理するか

```
start_line = 74   # CSVの処理範囲．最初の行番号（1はカラムの名前）
end_line   = 74 # 最後の行番号

plt.ion()
#fig, ax = plt.subplots()
plt.close('all')   # ← 余計な空図を全部消す
                        読み込むcsvファイルを指定
with open("scanlist3.csv", newline='', encoding='utf-8-sig') as f:  # CSV filename
    reader = csv.reader(f)
    header = [h.strip() for h in next(reader)]
```



実行すると1つ1つ図が標示され，(x,y,err)形式のデータが保存されていく．
Enterを押していき，最後の行まで行ったら終わる．

```
Press Enter to continue (Ctrl+C to quit)↑↓ for history. Search history with c-↑/c-↓
```

指定したスキャン番号の結果を1スキャンごとに標示する．rawdataからデータを読みこんでただ標示するだけ．csvファイルは不要．

```
# 指定したスキャン番号のrawdata表示（1つずつ）
import numpy as np
import os
import ipywidgets as widgets
import matplotlib.pyplot as plt
from IPython.display import display, clear_output

raw_dir = "rawdata1"   # rawdata フォルダ名
```

読み込むrawdataフォルダを指定

Scan: 428

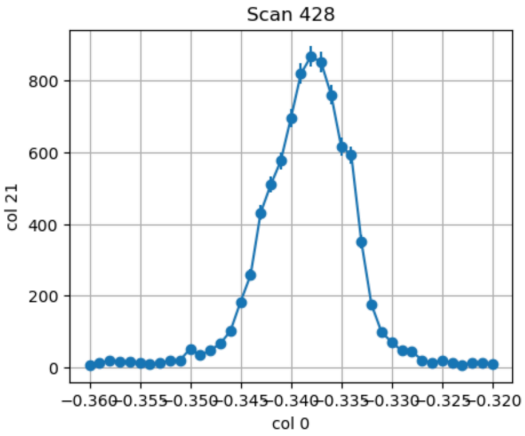
スキャン番号を指定．

X col: 0

x軸の列番号を指定（左から何列目か）．

Y col: -1

y軸の列番号を指定（右から何列目か，マイナスをつける）．



簡単な情報も表示される

```
#S 428  hklscan  -0.32 -0.36  -0.32 -0.36  8.5 8.5  40 10
Sun Oct 11 10:05:22 2026
H  K  L  Epoch  Seconds  Detector  I1  PA0  PA1  Ch2  Ch3  Ch4  TB  TC  Att  RingCurr.  Timepix2  Timepix3
Timepix4  TP_max  Monitor  Timepix1
Q=0.360001 0.360001 8.5,lambda=1.562694732, E=7.933999997639974
lattice params = 4.16508 4.16508 9.81118 90 90 90
#Tail1 MAX -0.33800 CEN -0.33815 COM -0.33837 FWHM 0.00961 ( -0.33335 - -0.34296 )
#Tail2 MAX-y 868 SUM 8482 MIN 6
# Slit1(v*h): 1.00 * 1.00 -Att 0.0 (mm)-Slit2: 2.00 * 2.00 -Slit3: 10.00 * 10.00
```

指定したスキャン番号の結果を標示する．最大3スキャンまで同時表示可能．
rawdataからデータを読みこんでただ標示するだけ．csvファイルは不要．

```
# 指定したスキャン番号のrawdata表示（3つまで重ねて表示可能）
import numpy as np
import os
import ipywidgets as widgets
import matplotlib.pyplot as plt
from IPython.display import display, clear_output
raw_dir = "rawdata3"  # rawdata フォルダ名
```

読み込むrawdataフォルダを指定

Scan1: 145 スキャン番号を指定．1個目．

Use 2 Scan2: 161 2個目．

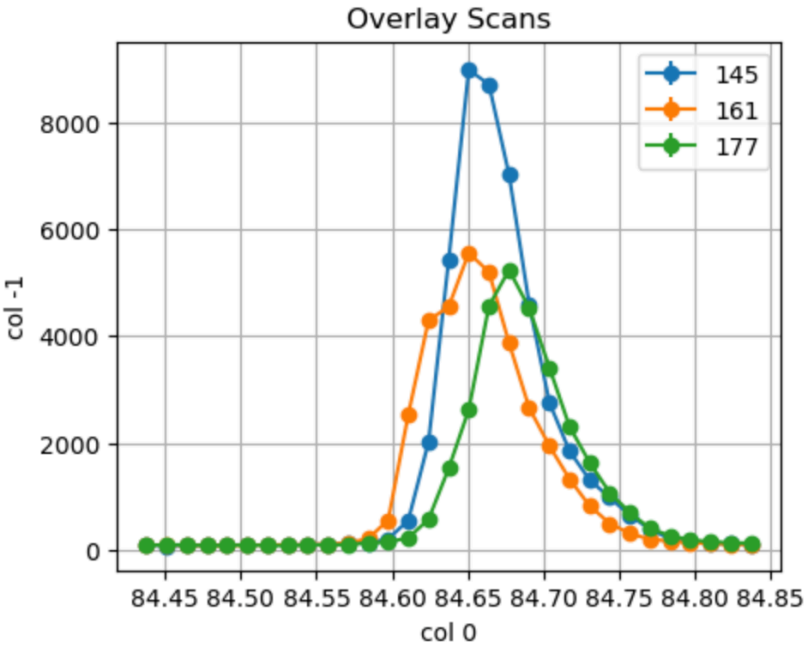
Use 3 Scan3: 177 3個目．

X col: 0 x軸の列番号を指定（左から何列目か）．

Y col: -1 y軸の列番号を指定（右から何列目か，マイナスをつける）．

Mon col: -2 モニター値の列番号（右から何列目か，マイナスをつける）．

Scale: 25000 1秒間のモニター値．cpsに直すためのスケール因子．




```
#S 145  ascan  th 84.4375 84.8375  30 1
Theta  H  K  L  Epoch  Seconds  I1  PA0  PA1  Ch2  Ch3  Ch4  TB  TC  Att  RingCu
ix2  Timepix3  Timepix4  TP_max  Monitor  Detector
84.4375 -0.496539 4.46093 10.5166 56928.8 1 0 3.70508e-07 4.87431e-12 0 0 0 4.49
```