「計数実験の手引き」：Python で計数経時変化の実験

日本科学技術振興財団　尾崎　哲

ＧM計数管を使用した、半減期の実験に適用できる

■計数実験データをパソコンに記録する

^{220}Rnの半減期の実験では、その短半減期に合わせて1秒ごとの計数を5分間以上継続する。ＧM計数管の表示を読み取って記録する方法では、2人で行う必要があるが、計数をパソコンに取り込めば、1人でも実験が可能となるほか、表計算ソフトを利用すれば、半減期の算出も容易となる。

ここでは、プログラミング言語 Python を使用して主に半減期の実験に適用するため、1 秒間の計数を繰り返してパルス波形をモニターするとともに 1 秒率の経時変化をグラフ表示するプログラムを紹介する。

■計数の経時変化を見るプログラム

以下では、プログラムの流れとポイントを解説する。

冒頭では、必要なライブラリーを呼び込む。なお、cv2 はプログラムの終了時のキー入力に使用する。また、関数の内外で使用する global 変数を初期化し、出力ファイルを用意。

```
import pyaudio #マイク入出力処理
import numpy as np
import matplotlib.pyplot as plt #グラフ処理
import cv2 #カメラ画像処理
m = 1 #経過時間
threshold = input('Threshold?') #閾値を設定
with open('cps_data.csv', mode='w') as fc: #計数値を書込むファイルを新規モードで開く
    fc.write('Time(sec)' + ',' + 'CPS' + '¥n') #タイトル

cps_data = [] #計数値の空リストを用意
```

def は関数で、1 秒間サンプリングして、グラフの描画までの処理を行う。まず、入力ストリームを開き、バッファーには CHUNK 単位で取り込む。frames は全データで、リストとして用意し、バッファーからリストに追記を繰り返す。後処理の都合で、リストからアレイに変換してからストリームを閉じる。

```
def streamandcount(): #入力からグラフ描画まで処理
    global m, threshold #グローバル変数
    CHUNK = 1024 #入出力の長さ
    FORMAT = pyaudio.paInt16 #2Byte 整数
    CHANNELS = 2 #ステレオ
    RATE = 44100 #サンプリング周波数(Hz)
    RECORD_SECONDS = 1 #録音時間(秒)

    p = pyaudio.PyAudio()
    stream = p.open(format = FORMAT,
                    channels = CHANNELS,
                    rate = RATE,
                    input = True, #入力のみ
                    frames_per_buffer = CHUNK)
    frames = [] #空の list を用意

    for i in range(int(RATE * RECORD_SECONDS / CHUNK)): #CHUNK 単位で切り捨て
        data_raw = stream.read(CHUNK)
        frames.append(data_raw) #43 回分を加算
    frames_int16 = np.array(frames)

    stream.stop_stream() #ストリームを閉じる
    stream.close()
    p.terminate()
```

バイトデータを整数に変換し、ステレオ録音の片チャンネルを抽出する。計数率は、パルス波高が閾値を超えるとフラッグを立てて、カウントを 1 増やし、閾値を下回るとフラッグを降ろす。1 秒率を CPS として画面に表示するとともに、ファイルに書き込む処理を次々と更新する。

```
pulse = 0 #計数を初期化
data_l = data[::2] #ステレオの L チャンネルを取り出す
onflag = False #閾値超過フラッグを初期化
for k in range(len(data_l)): #data_l 全体が範囲
    dat = data_l[int(k)] #data_l の k 番目の値
    if int(dat) > int(threshold) and not onflag: #閾値を初超過
        pulse = pulse + 1 #1 カウントを加算
        onflag = True #閾値超過フラッグを立てる
    if int(dat) <= int(threshold): #閾値以下になったら
        onflag = False #閾値超過フラッグを降ろす
cps = pulse #CPS に換算
print('CPS = ' + str(int(cps))) #計数値を画面に出力
cps_data.append(cps) #計数値の list に追加
with open('cps_data.csv', mode='a') as fc: #ファイルを開く
    fc.write(str(m) + ',' + str(cps) + '¥n') #経過と計数値を追記
fig = plt.figure(num = 1, figsize = (6, 7)) #グラフを定義
plt.subplot(211) #グラフの上半分を指定
plt.plot(data_l[200:1200]) #入力信号の一部 44msec 分を表示
plt.ylim(-250, 1000) #y 軸の範囲
plt.xlabel('t (msec/44)') #x 軸のラベル
plt.ylabel('PulseHeight') #y 軸のラベル
plt.subplot(212) #グラフの下半分を指定
plt.plot(cps_data) #計数値を表示
plt.xlim(0, 600) #x 軸の範囲
plt.ylim(0, 1000) #y 軸の範囲
plt.xlabel('t (sec)') #x 軸のラベル
plt.ylabel('CPS') #y 軸のラベル
plt.pause(0.3) #0.3 秒間だけ描画
plt.clf() #グラフを閉じる
```

「q」をキーボードから入力すると、プログラムは終了する。

```
while True: #キー入力を待つ
    streamandcount() #処理を反復
    m = m + 1 #経過時間 1 秒を加算
    if cv2.waitKey(0) & 0xFF == ord('q'): #q を入力
        break #処理を終了
```

■実験方法

７セグメントＬＥＤ表示の計数装置から出力信号を取り出す。イヤホン・ジャックとパソコンのマイク入力端子をケーブルで接続するが、0.022μF 程度のコンデンサを並列に入れて過大入力(<5V)を防ぐ。必要があれば、マイクのプロパティーを開き、レベルを調整する。

ＧＭ管は、密閉型ではなくクリアケース型を使用し、ランタン・マントルから抽出したラドンガスを定量のブタンガスとともにＧＭ管に注入し、すぐに測定を開始する。測定開始前に空起動しておくと、起動時の遅れが防げる。

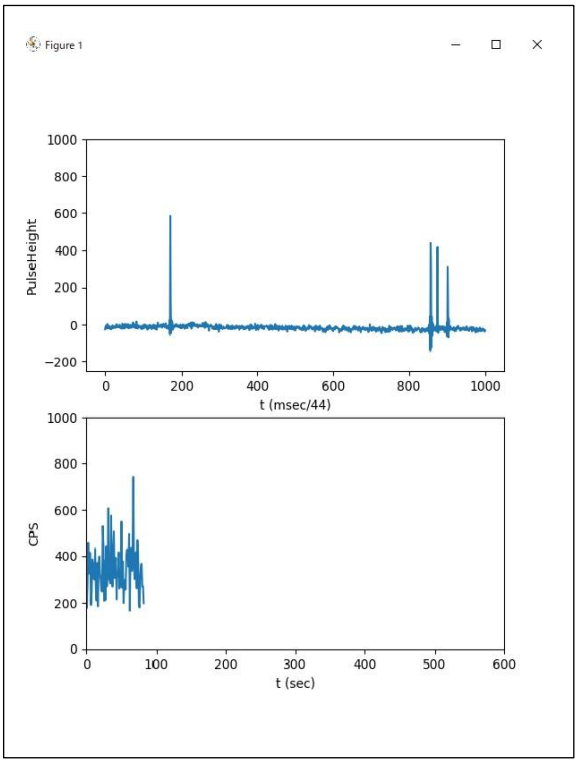

図　グラフの描画例