生醫實驗期末專題報告

測謊原理與實作

第八組

電機四 賴慶旻 B02901046

電機四 林裕洲 B02901064

電機四 林子翔 B02901069

1. 動機

我們常常在電視影集或相關的新聞中看到測謊橋段，因此我們希望除了能夠了解測謊的相關原理與潛在問題之外，也能運用所學實際做出測謊模組來進行測試。

1. 原理

測謊可藉由以下幾種生理訊號的變化進行判讀。

1. 皮膚電阻：在緊張、焦慮與恐懼的情境之下，腎上腺素分泌增加，促使汗腺分泌，造成皮膚電阻下降。
2. 心血管：腎上腺素分泌增加造成心跳加速、血壓上升。
3. 呼吸：焦慮使得受測者呼吸變得急促、呼吸深度變淺。
4. 腦波：在高度警覺與焦慮的情況之下，某些頻段的腦波會增強。例如快速α波代表高度警覺，而快速β波則代表焦慮。所以當受測者說謊時，高α跟高β波會有明顯的波峰。

資料來源：維基百科 -- 腦波

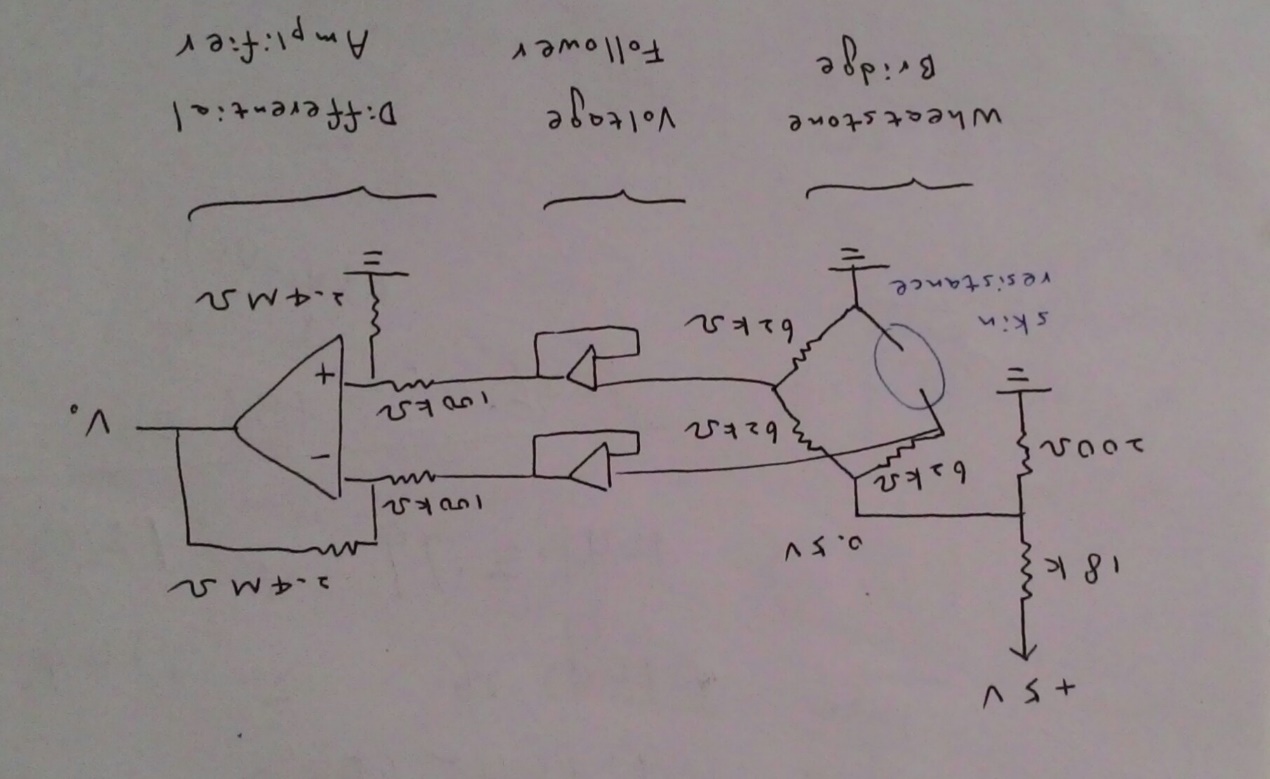
1. 實驗設計

我們從1到10分別製作10張數字卡。測謊方式可分為相關-無關測試法（relevant-irrelevant test)、對照問題測試法（control question tests)、罪知感問題測試（concealed information tests（CIT）與緊張高點法（peak-of-tension test）。本實驗採用緊張高點法，使用者的生理反應會隨著正確答案的逼近而漸強，並隨著正確答案的遠離而逐漸平復。我們先使受測者在心中想一個1到10之間的數字，接著訪問者會從1、2、….、10的順序進行猜測，受測者必須從頭到尾否認。我們將藉由判讀所量測到的生理訊號，推測受測者何時說謊，也就是說，必須判讀出受測者心目中的數字。

1. 量測模組

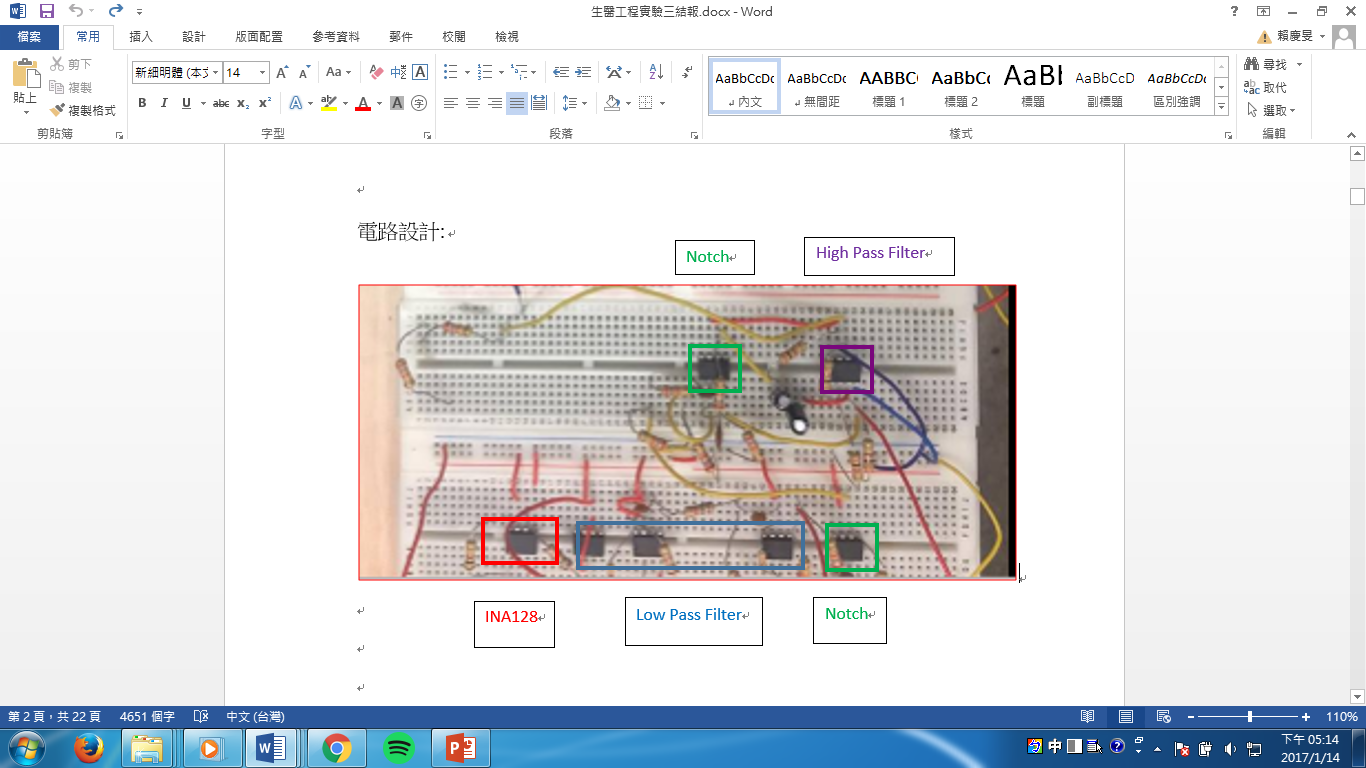
1. Galvanic skin response

電路設計如下圖所示。為了得到手指的皮膚電阻，我們將生物貼片分別黏貼於左手的食指與無名指上，並將所得到的皮膚電阻由下圖所標註的skin resistance區域輸入。在此Wheatstone Bridge的功能為得到differential voltage。接著我們將differential voltage輸入voltage follower，如此一來便可隔絕前後級的影響並減少crosstalk。最後再將訊號輸入differential amplifier放大24倍以方便觀察跟分析，最後分析output voltage。理論上說謊時皮膚電阻下降會使得觀察到的output voltage下降。



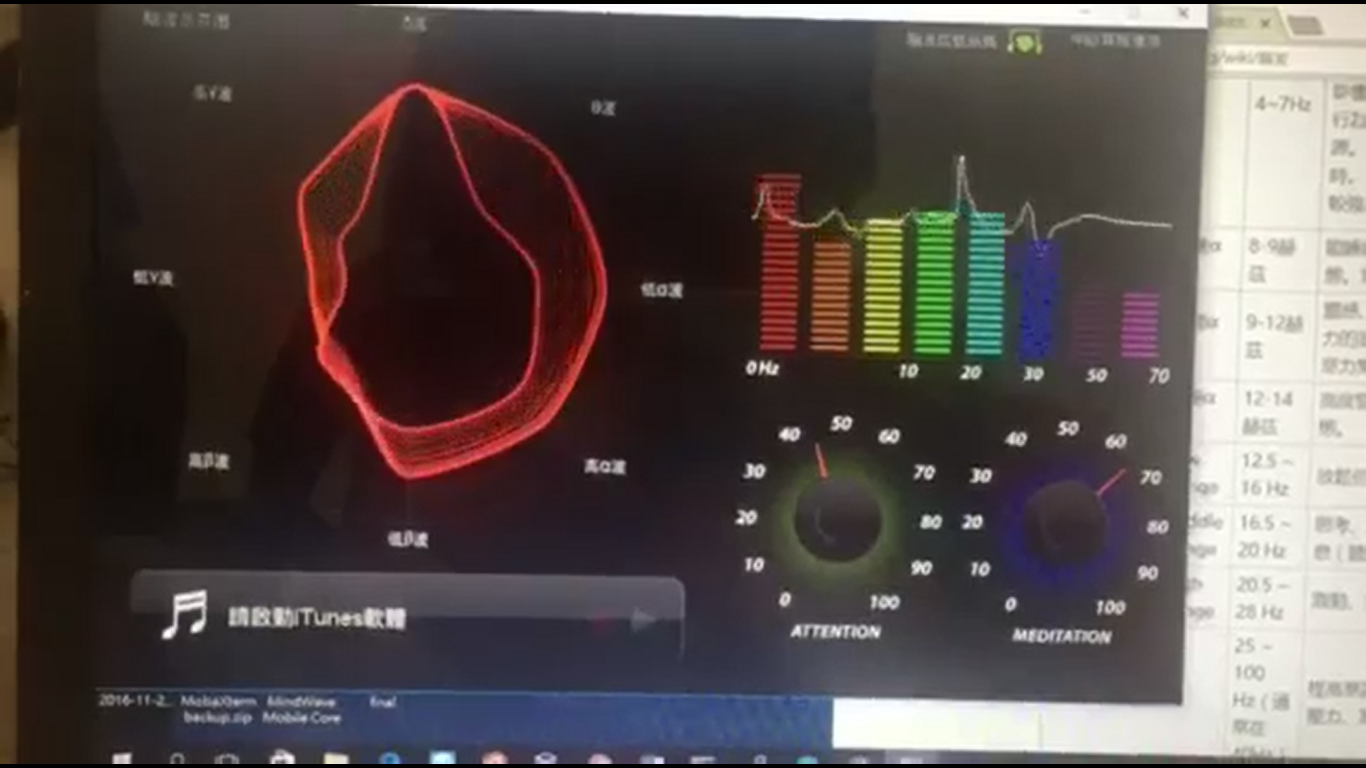
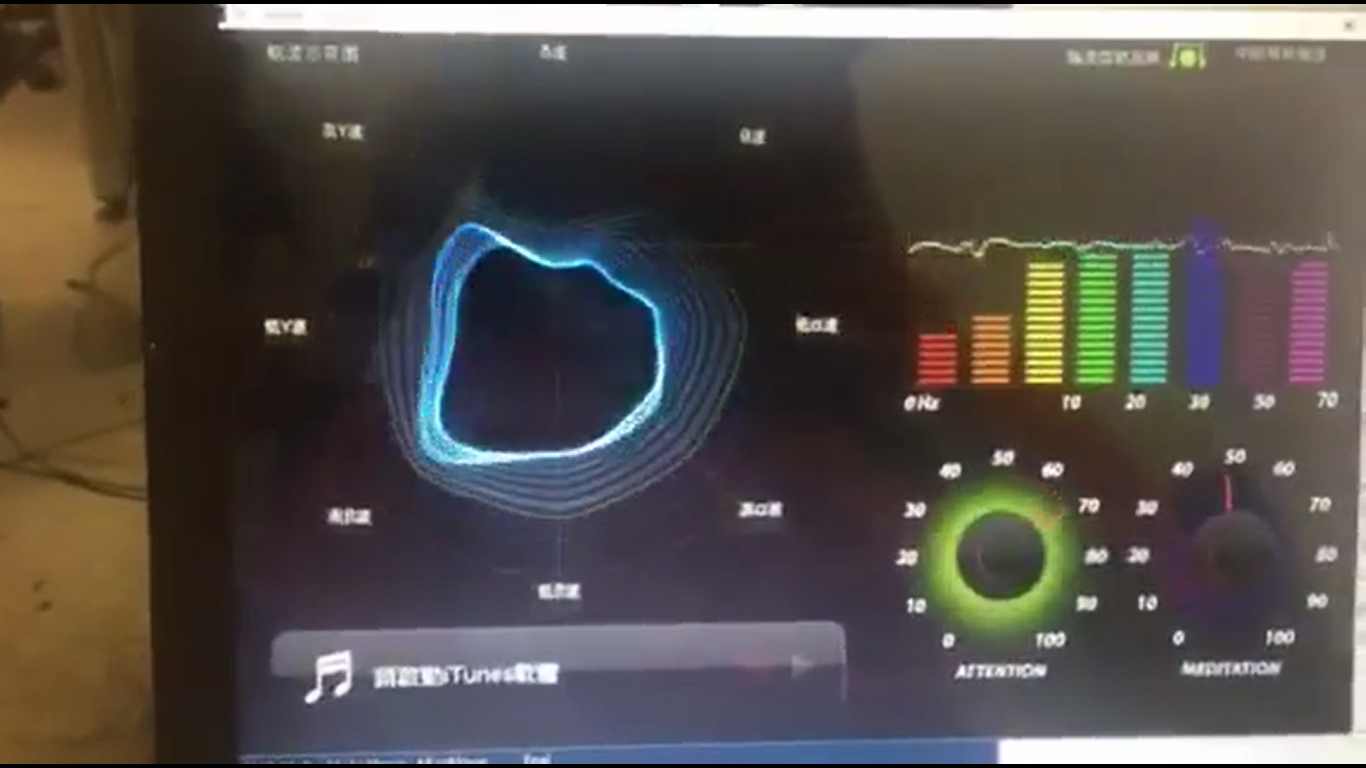
1. ECG 電路

我們使用實驗三的ECG電路，並利用NI Elvis將心電訊號輸出進行分析。ECG電路設計如下圖所示。訊號先輸入LNA128 進行放大並減少系統的noise figure，再分別經由LPF、Notch與HPF以擷取出我們要的心電訊號。



1. 腦波

我們使用助教提供的NeuroSky brainwave: Brainwave starter kit套件來量測腦波。這款商用儀器會將所測得的腦波透過藍芽，將結果傳輸至電腦。可以看出不同頻率的波相對大小值以及專注和放鬆程度，如下圖。

1. 呼吸頻率量測

在這個主題上我們查詢了非常多的資料跟作法，呼吸產生的生理訊號主要可分為這四大類：

|  |  |  |
| --- | --- | --- |
| 生理訊號 | 量測儀器、方式 | 主要的難題 |
| 熱能 | 熱敏電阻、溫度感測器 | 1. 準確度，需要校準環境的室溫。 2. 散熱需要夠好，成年人呼吸頻率約每分鐘12到18次，因此需在4秒內完成散熱。 |
| 胸腔、腹腔的位移 | 接觸式：加速度計  非接觸式：光學、超聲波 | 加速度計：需要設計能附著在胸腔上的設施(如：一個腰帶)，機械結構的穩定度。  光學、超聲波：儀器不易取得，且要排除環境中的干擾。 |
| 呼吸聲 | 將麥克風貼在喉嚨上，若直接在胸腔測量容易被心音干擾 | 本實驗中因為受測者需要透過講話來回答問題，因此透過喉嚨收取呼吸聲的訊號會嚴重的被干擾，如果更改實驗形式為透過紙筆進行，可能會因為心理因素而讓受測者比較不緊張 |
| 鼻孔外的空氣氣流 | 壓力計 | 環境干擾，如人走動、風扇、冷氣等。 |

我們查了商用的測謊機(<http://polygraph.com.tw/faq.php>)，他們是透過綁兩條感測器在胸腔跟腹腔上，同時測量胸呼吸跟腹呼吸。整體來說呼吸聲跟遙測胸腔的位移是最可以準確量測的訊號(呼吸聲有查到實作出來的論文，準確度非常高)，但考量到本實驗的性質，最後決定到光華商場購買三軸加速度計，但來不及設計出可以附著在胸腔上的機械構造，非常可惜。

呼吸部分的參考資料：

心電圖中的低頻載波會等於呼吸的頻率，因此可透過分析心電圖來量測呼吸頻率，但本實驗中因為無法量測出精確的心電圖，因此沒採用。

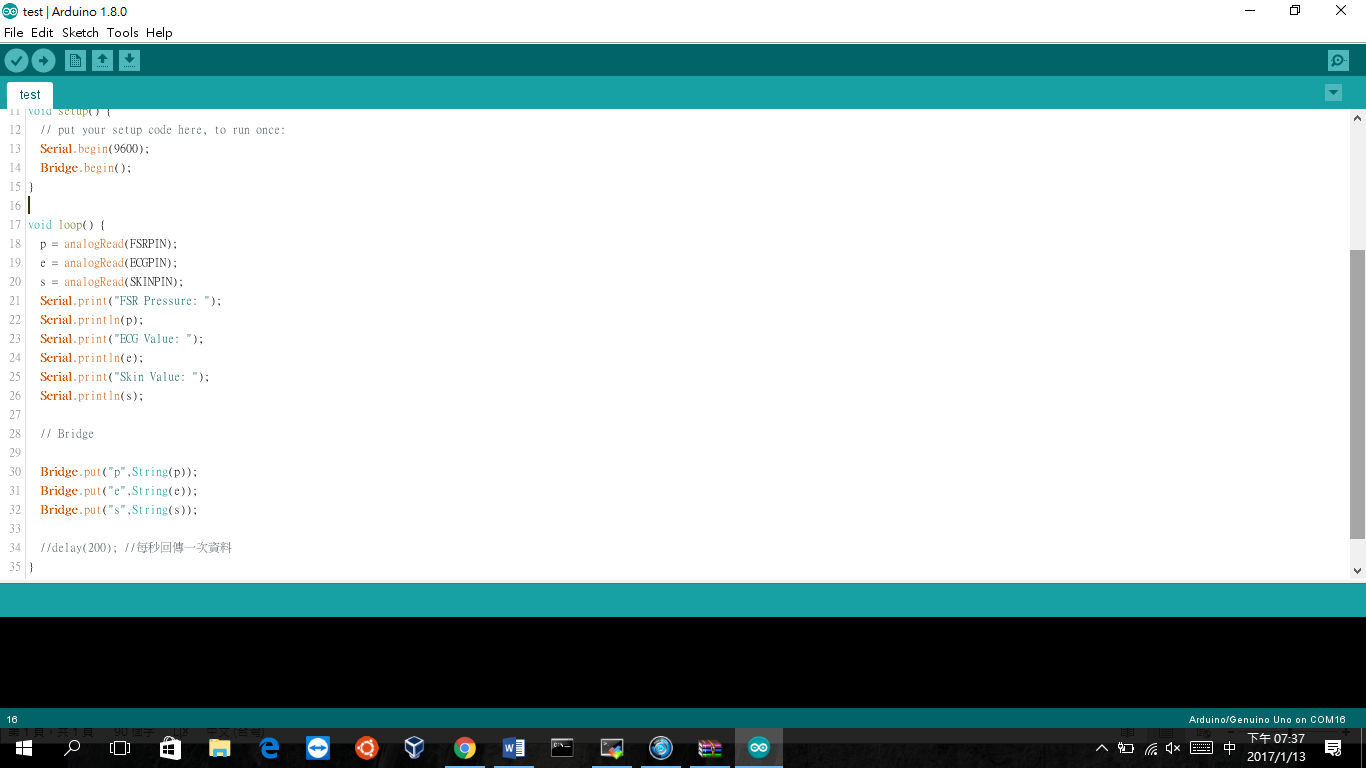
<http://www.auto.fcu.edu.tw/wSite/publicfile/Attachment/f1254204690020.pdf>

可攜式呼吸速率測量儀之設計(即透過量測喉嚨處的呼吸聲，以推算呼吸頻率的相關論文，實驗結果顯示可以非常精確)

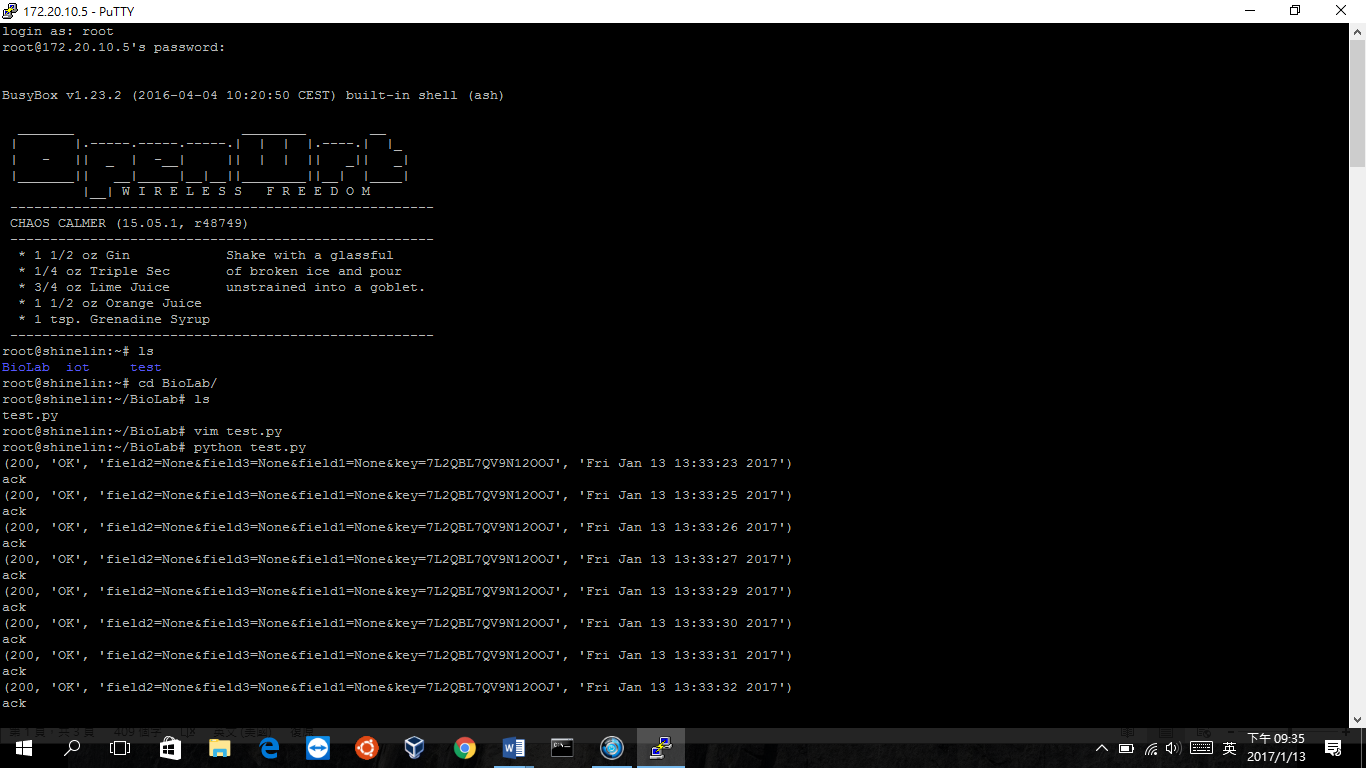
<http://eshare.stust.edu.tw/EshareFile/2011_12/2011_12_c803f5c4.pdf>

1. 數據視覺化：

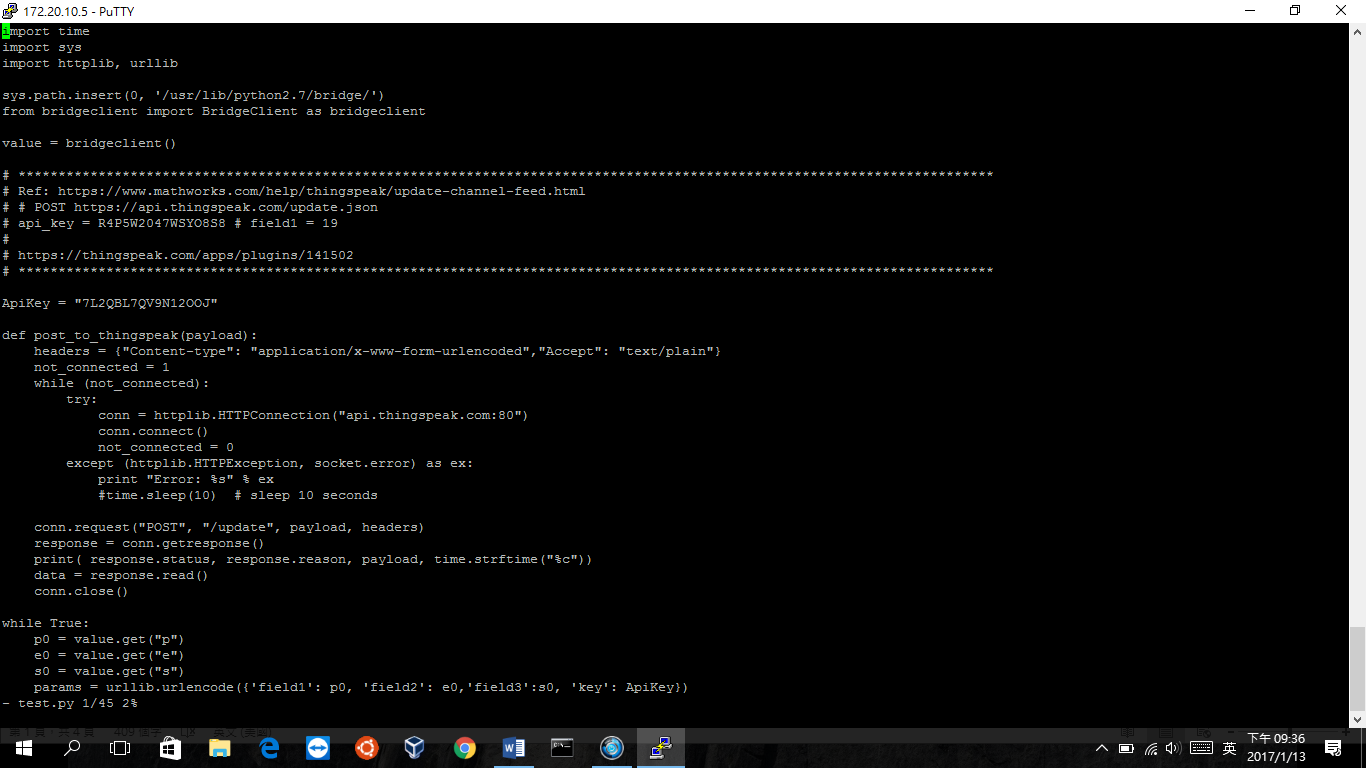
**Arduino 端：**利用 MCU 收集在arduino上sensor收集到的數據，再利用duo中的linux openUART系統，寫python code以HTTP來完成與thingspeak的連結。註：Code放在附錄



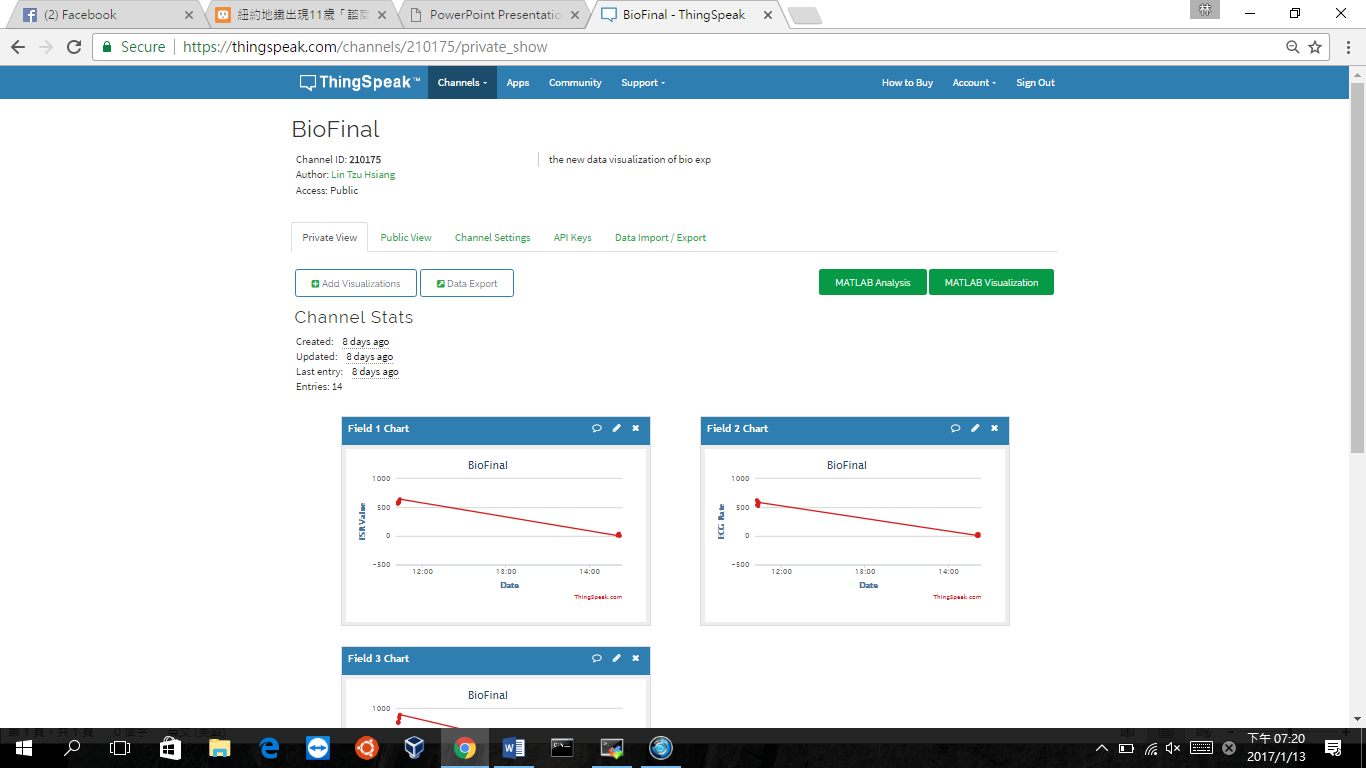
**Linux 端：**



**Python code：**



**Thingspeak：**

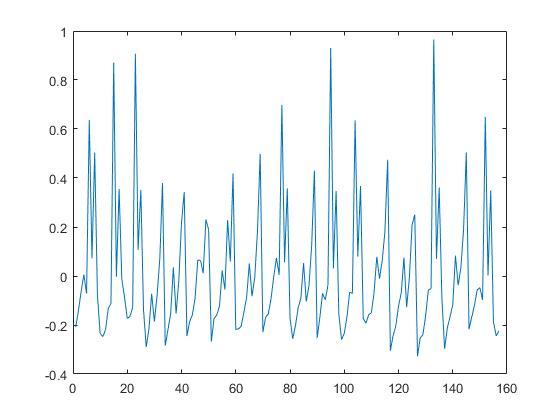
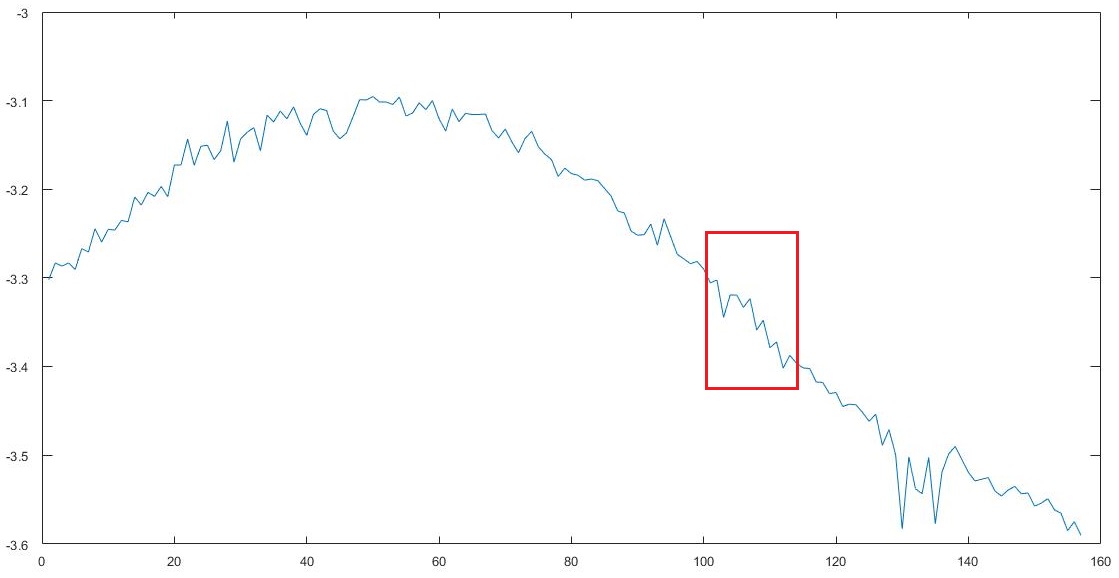


目的：利用duo7688將arduino收集到的sensor上傳到thingspeak，希望及時觀察上傳的data來判斷受測者是否說謊。

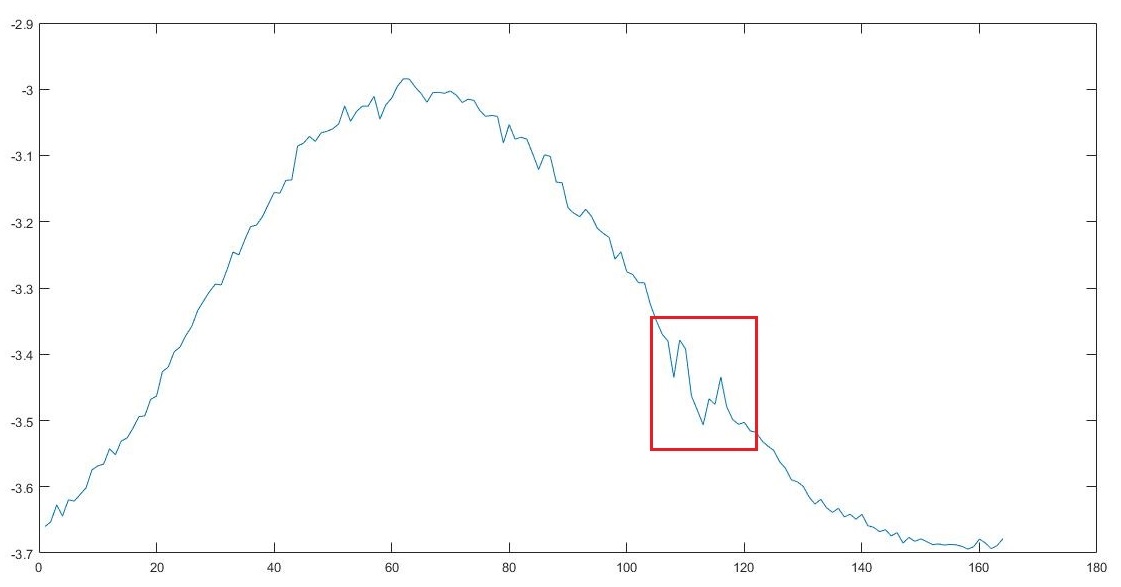
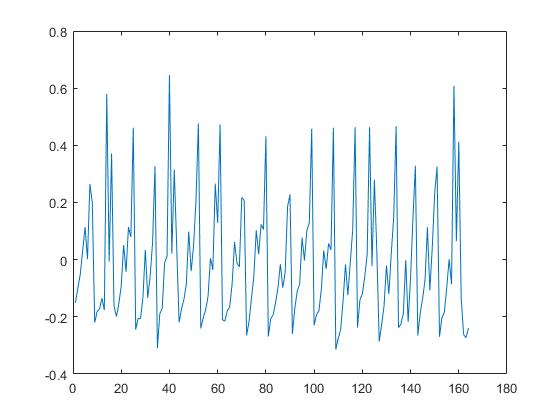
失敗原因：因為在windows系統的上傳速率最快是15s/次，linux可以改成自己想要的速率，但我們太晚發現，以至於無法使用此功能。

1. 數據分析

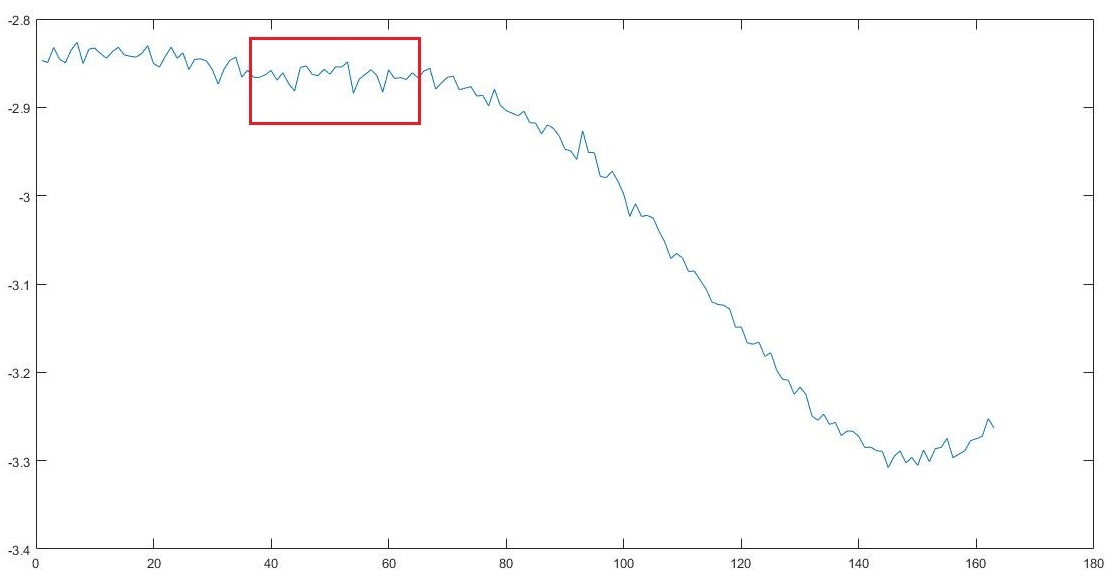
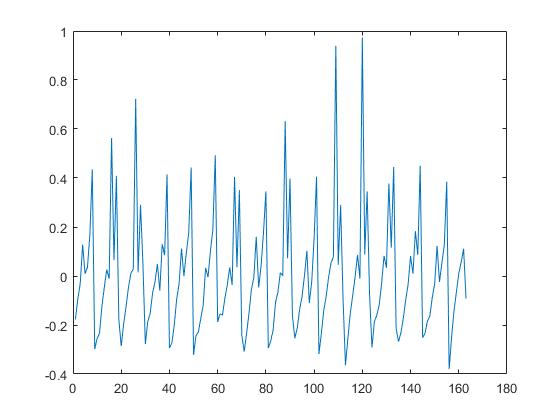
我們總共有效的實驗有五輪，逐次分析如下：

**第一輪：**受測者在橫軸t=98(代表9.8秒)左右說謊。

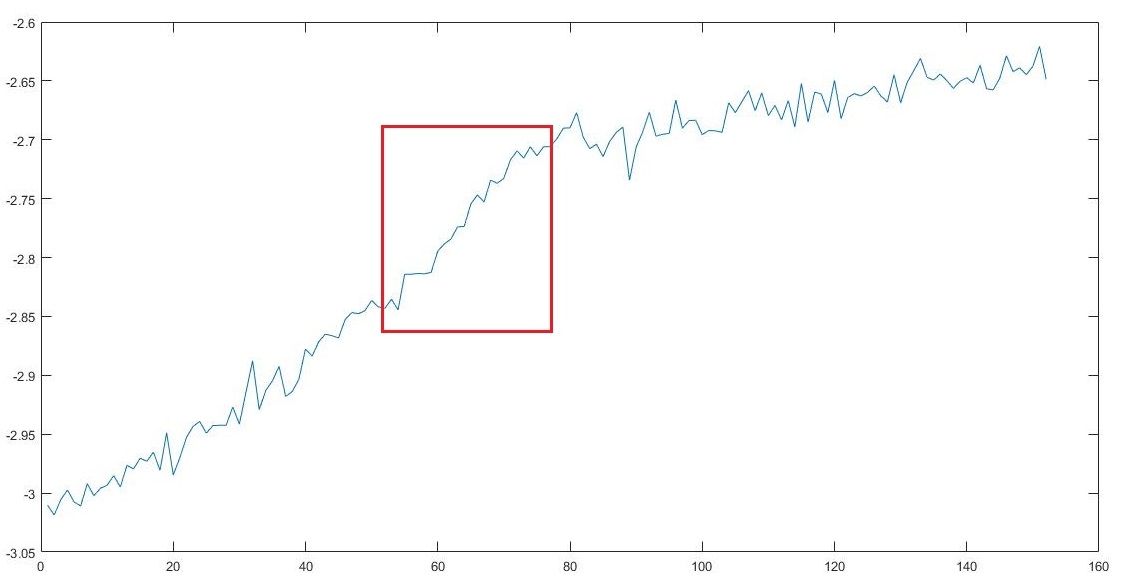
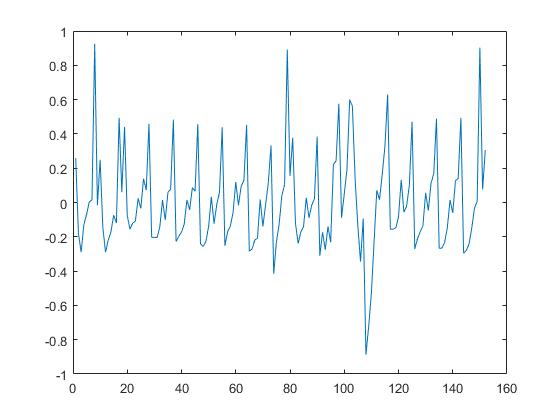
**第二輪：**受測者在橫軸t=134(代表13.4秒)左右說謊。



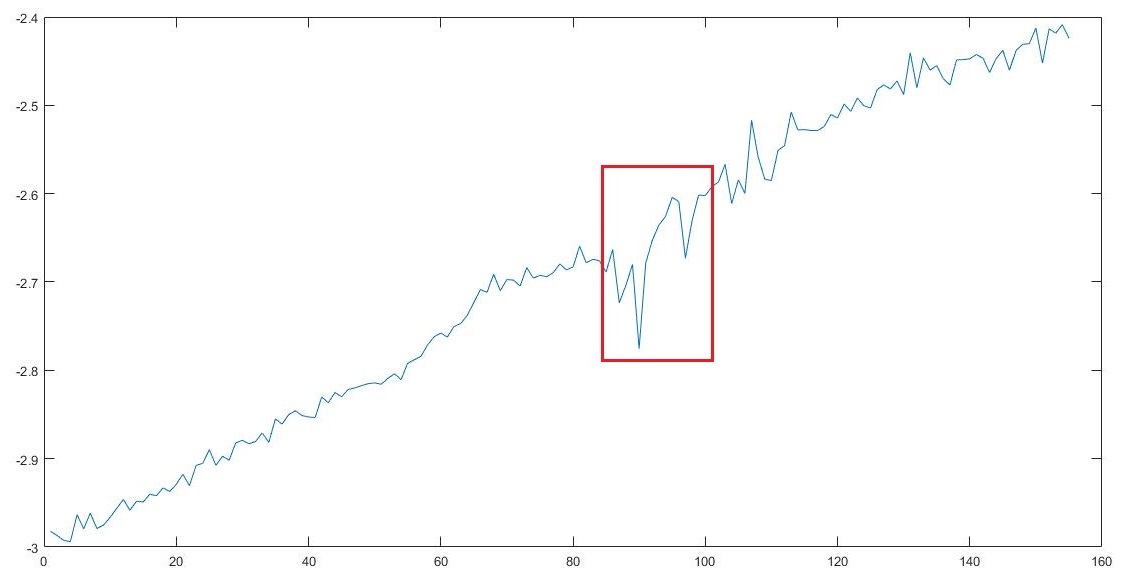
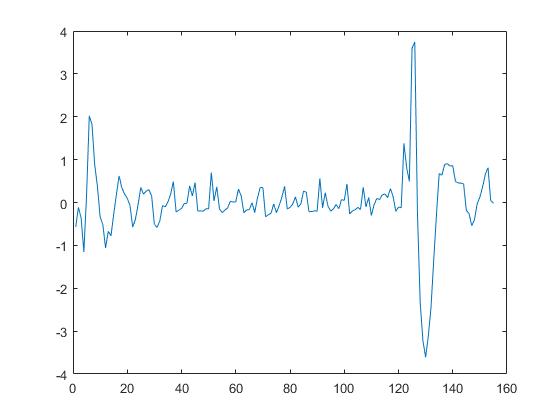
**第三輪：**受測者在橫軸t=42(代表4.2秒)左右說謊。



**第四輪：**受測者在橫軸t=57 (代表5.7秒)左右說謊。



第五輪：受測者在橫軸t=95 (代表9.5秒)左右說謊。



數據分析：首先因為實驗執行的有點太快，平均一個問題才1.6秒左右，所以數據普遍不甚理想。這點助教有提醒過，卻到了最後補做數據還沒改進。

ECG的變化相對不明顯，比較難用於判斷。膚電阻的部分在第二輪的10秒以後，即使在下降的趨勢中，還是看得出一個提早了一到兩秒反應的電阻下降。第五輪實驗則非常完美，提早一秒到說謊的當下的膚電阻都有明顯偏離趨勢的變化。但其他三輪應該是因為實驗方式不夠仔細，因此算是實驗失敗。

1. 問題討論

我們有找幾個受測者來進行分析，發現到幾個問題。

1. Galvanic skin response適用在比較會冒手汗的人身上。對於不容易冒手汗的人，就算緊張也無法藉由汗腺分泌來減少皮膚電阻，這也造成該電路的使用限制。
2. 我們發現在說謊時，男性的快速α波與快速β波會明顯增強，然而女性受測者並未呈現如此顯著的變化。儘管我們知道男女的腦部結構是有差異的，不過是否會因此導致說謊當下腦波的差異仍有待商榷。推測也有可能是因為女性的頭髮量較多，使其雜訊相對男生大很多，而使儀器無法準確接收到數據進而影響我們對資料的解讀。
3. 量測腦波的儀器一開始連接電腦時已經成功安裝驅動軟體也成功配對卻一直連不上，後來查詢才了解是因為美國供電的頻率是50Hz，在台灣則是使用60Hz的交流電，要改成60Hz。設定完成後即可成功配對。(但DEMO當天，不知道為什麼和筆電又無法成功配對，試了兩個小時還是無法成功。)
4. 實驗過程中，我們發現我們所預估的數字與受測者會相差1到2個數字不等。例如受測者想的可能是5，結果我們會猜4或6。這是因為生理訊號會有”delay”，也就是說皮膚的電阻值下降需要時間，可藉由放慢測試速度來做改善。
5. 最後是我們所問的內容。由於我們只是單純的猜數字，並不是問像犯罪如此嚴重的問題，因此有些受測者並不會感到緊張或壓力。這使得所量測到生理訊號不夠明顯，進而造成誤差。
6. 由目前的實驗結果顯示離網路上查到的猜數字可達精確度80%以上還有一段距離，可能要逐次實驗後檢討，並多跟受測者討論每個當下的心理狀態，是否會因為短時間內大量測試而疲勞、反應不明顯等，也可能需要控制實驗環境讓受測者更不易分心。
7. 未來展望

我們認為我們所設計的模組除了能應用在測謊方面，也能用於相關的情緒感測。像是我們有請懼高的受測者觀賞高空彈跳或極限自拍的影片，我們發現Galvanic skin response的output voltage會快速下降，這代表受測者當下處於相當緊繃的情緒。由於現在流行VR therapy，也就是利用VR幫助使用者克服相關的恐懼(例如蛇、創傷症候群等)。我們可將VR裝置與我們所設計的模組結合，當使用者的恐懼超過臨界threshold時，就將VR裝置暫停以確保安全。另外情緒辨識也是相關的發展方向。

1. 心得

林子翔：

這次期末專題是一個很崩潰的時候，我當天下午也有另一個期末demo，其他組員下午也都有期末考。但在實作上學到很多東西，比如試著用arduino完成把sensing data上傳雲端並視覺化，原本想更輕易以圖形資料判斷受測者有沒有說謊，也試著以樹梅派實作，但後來樹梅派不知道出了甚麼問題，兩塊板子都無法順利登入，也寫信去台灣官方網站詢問問題卻仍沒有解決。而腦波的儀器在當天demo也一直出現問題，認真認為硬體的問題比軟體多非常多，而且無法預測。

這門課的總心得是很棒的，可惜的是前面loading太輕，平均兩個禮拜做一次實驗，但第三次實驗等於每天都要花兩三個小時泡在實驗室，期末專題亦是如此。但學到了很多東西，包括如何設計電路偵測心電、肌電、腦電訊號並過濾分析，包括如何測謊，用電路以及儀器解讀身體的生理反應。

賴慶旻：

我們本來希望能夠加上Arduino與三軸加速器將期末專題做得更完整，不過因為來不及處理相關的硬體問題只好作罷，而demo當天助教提供的Neurosky無法運作也相當可惜。不過藉由專題讓我重新複習快被忘光的電子學，也學到Arduino與三軸加速器的操作，讓我覺得非常值得。

林裕洲：

這次期末專題成果雖然不甚理想，但收穫很多。

1. 在膚電阻量測上，我們發現不論有無使用電極貼片，膚電都蠻不穩定的，很容易有長時間(持續三十秒以上)的上升或下降(且變化前跟一分鐘後可差20%以上)，或是在小範圍內不斷的變化震盪，且若是接上一個可變電阻，只要沒改動電阻值最後量測到的電壓就非常穩定，因此我們確定不是儀器的問題。可能還需要多了解相關的生理機制才能設計出更好的信號處理的方法。
2. 在呼吸頻率上，認真查了很多資料後才發現，原來一個簡單的數字就有這麼多的切入角度，也學到如何分析、推測各個方法哪些可行、哪些容易準確，並因此推翻原本打算和學長姐一樣都使用熱敏電阻的想法。可惜自己的很多知識跟技巧都不夠純熟，沒辦法在這次實作出來。
3. 在數據處理上，因為matlab只能接受很整齊的資料格式(Matrix)，每一行內要用逗點隔開，如 1.215765, 5, 7, 2

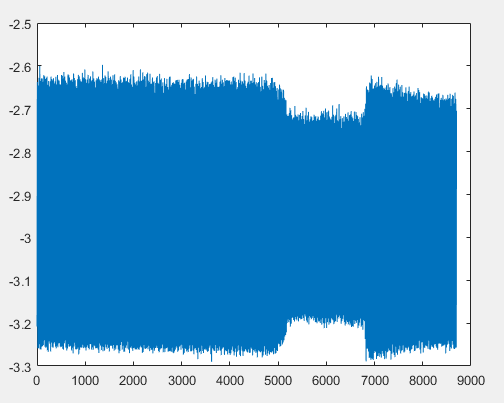
但我們從NI ELVIS上得到的資料檔案的前兩行是：

Time Dev2 (NI ELVIS II+)/ai0 Time Dev2 (NI ELVIS II+)/ai1

2017/1/13 13:39:20.267471 -3.207755E+0 2017/1/13 13:39:20.267482 -2.037608E-2

有煩瑣沒必要且matlab無法接受的日期格式，還有指數被化成E+或E-，加上資料點有上千個，只能透過程式大量處理，因此寫了一個code處理，程式碼附在最後方。

1. 資料sample頻率高不一定好：處理過後的數據，前幾次實驗因為取樣的頻率調得太高(1000Hz)，導致matlab膚電阻的輸出電壓如下，因為連續五次都這樣，因此應該不是實驗時的操作錯誤，推測可能是取樣頻率過高會把許多微小的擾動納入，但若將取樣頻率適度調低，電腦會自動作running average自動排除這些雜訊。



1. 參考資料
2. <http://www.stat.cmu.edu/tr/tr766/tr766.pdf>
3. <http://highscope.ch.ntu.edu.tw/wordpress/?p=6899>
4. <https://zh.wikipedia.org/wiki/%E6%B8%AC%E8%AC%8A%E6%A9%9F>
5. <http://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/f2012/htq2_mg573/htq2_mg573/>

十、 附錄

* 1. 測謊膚電阻與MCG資料格式處理程式碼

#include <iostream>

#include <fstream>

#include <vector>

#include <stdlib.h>

#include <sstream>

using namespace std;

double timeconvert (string str){

string min;

string sec;

string subsec;

int minute;

int second;

double subsecond;

double timesum;

min = str.substr(0,2);

sec = str.substr(3,2);

subsec = str.substr(6,6);

minute = atoi(min.c\_str());

second = atoi(sec.c\_str());

subsecond = atoi(subsec.c\_str());

timesum = minute\*60+second+subsecond/1000000;

cout.precision(10);

return timesum;

}

double dataconvert (string str){

string numint;

string numsub;

string numexp;

int numberint;

double numbersub;

double numbersum;

bool posi;

int exp;

double expnumber=1;

bool expposi;

if (str.substr(0,1)=="-"){

posi = false;

str=str.substr(1);

}

else

posi = true;

numint = str.substr(0,str.find("."));

numsub = str.substr(str.find(".")+1,6);

numexp = str.substr(str.find("E")+2);

if (str.substr(str.find("E")+1,1)=="+"){

expposi=true; //cout <<"haha posi";

}

else if (str.substr(str.find("E")+1,1)=="-"){

expposi=false; //cout <<"haha nega";

}

numberint = atoi(numint.c\_str());

numbersub = atoi(numsub.c\_str());

exp = atoi(numexp.c\_str());

while (exp!=0){

if (expposi)

expnumber=expnumber\*10;

else

expnumber=expnumber/10;

exp--; //cout <<"ha";

}

if (posi)

numbersum = expnumber\*(numberint+numbersub/1000000);

else

numbersum = (-1)\*expnumber\*(numberint+numbersub/1000000);

return numbersum;

}

int

main(int argc, char \*argv[])

{

string str;

string time1;

double time\_sum1;

string num1;

double number1;

string time2;

double time\_sum2;

string num2;

double number2;

ifstream fin(argv[1]);

ofstream fout(argv[2]);

int count =0;

fout.precision(15);

while (!fin.eof()){

getline(fin,str);

if (str.substr(0,1)!="T"){

if (count ==1){

str=str.substr(str.find(":")+1);//delete anything before first :

//cout << str << endl;

time1 = str.substr(0,12);// get time 1,

time\_sum1 = timeconvert(time1);

fout <<time\_sum1<<", ";

str=str.substr(13); //remove the part of time1

num1=str.substr(0,str.find(' '));

number1 = dataconvert(num1);

fout << number1 <<", ";

str=str.substr(str.find(":")+1);

time2=str.substr(0,12);

time\_sum2 = timeconvert(time2);

fout <<time\_sum2<<", ";

str=str.substr(13);

num2=str.substr(0,str.find(' '));

number2 = dataconvert(num2);

fout << number2 <<", ";

fout<<endl;}

}

else {

count +=1;

if (count ==2){

cout <<" incontinue!!";

break;

}

}

}

}

* 1. Arduino 收集sensor data code

#define FSRPIN A3

#define ECGPIN A4

#define SKINPIN A5

// Bridge

#include <Bridge.h>

float p = 0;

float e = 0;

float s = 0;

void setup() {

// put your setup code here, to run once:

Serial.begin(9600);

Bridge.begin();

}

void loop() {

p = analogRead(FSRPIN);

e = analogRead(ECGPIN);

s = analogRead(SKINPIN);

Serial.print("FSR Pressure: ");

Serial.println(p);

Serial.print("ECG Value: ");

Serial.println(e);

Serial.print("Skin Value: ");

Serial.println(s);

// Bridge

Bridge.put("p",String(p));

Bridge.put("e",String(e));

Bridge.put("s",String(s));

//delay(200); //每秒回傳一次資料

}

* 1. Linux上傳thingspeak code

import time

import sys

import httplib, urllib

sys.path.insert(0, '/usr/lib/python2.7/bridge/')

from bridgeclient import BridgeClient as bridgeclient

value = bridgeclient()

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Ref: https://www.mathworks.com/help/thingspeak/update-channel-feed.html

# # POST https://api.thingspeak.com/update.json

# api\_key = R4P5W2047WSYO8S8 # field1 = 19

#

# https://thingspeak.com/apps/plugins/141502

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ApiKey = "7L2QBL7QV9N12OOJ"

def post\_to\_thingspeak(payload):

headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}

not\_connected = 1

while (not\_connected):

try:

conn = httplib.HTTPConnection("api.thingspeak.com:80")

conn.connect()

not\_connected = 0

except (httplib.HTTPException, socket.error) as ex:

print "Error: %s" % ex

#time.sleep(10) # sleep 10 seconds

conn.request("POST", "/update", payload, headers)

response = conn.getresponse()

print( response.status, response.reason, payload, time.strftime("%c"))

data = response.read()

conn.close()

while True:

p0 = value.get("p")

e0 = value.get("e")

s0 = value.get("s")

params = urllib.urlencode({'field1': p0, 'field2': e0,'field3':s0, 'key': ApiKey})

post\_to\_thingspeak(params)

print "ack"