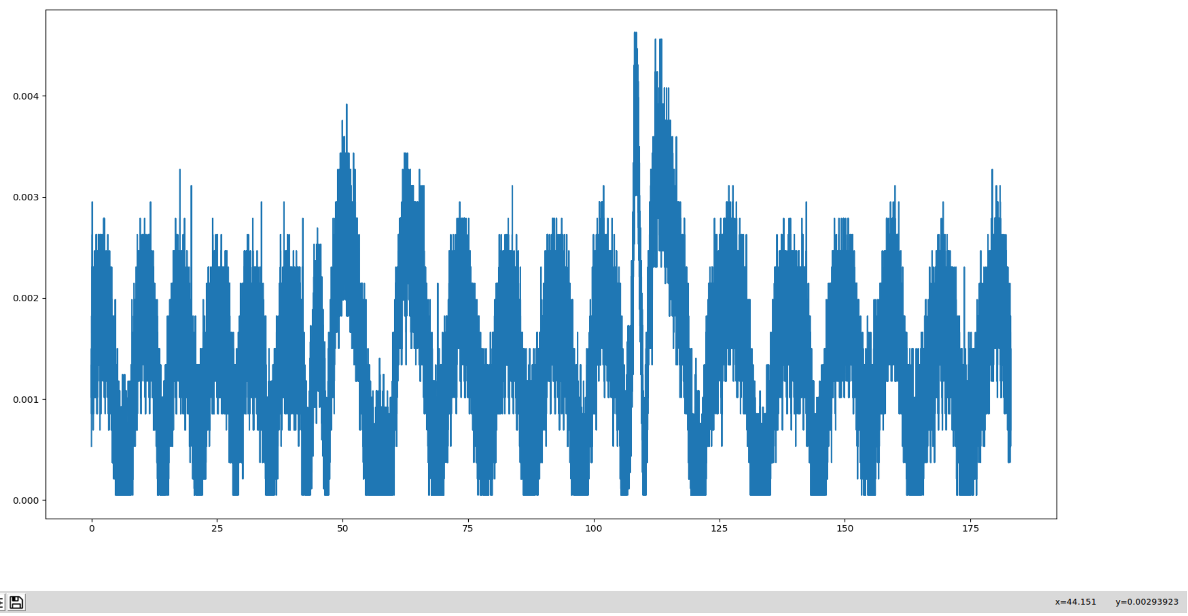
訊號與系統期末專題 0416324 胡安鳳

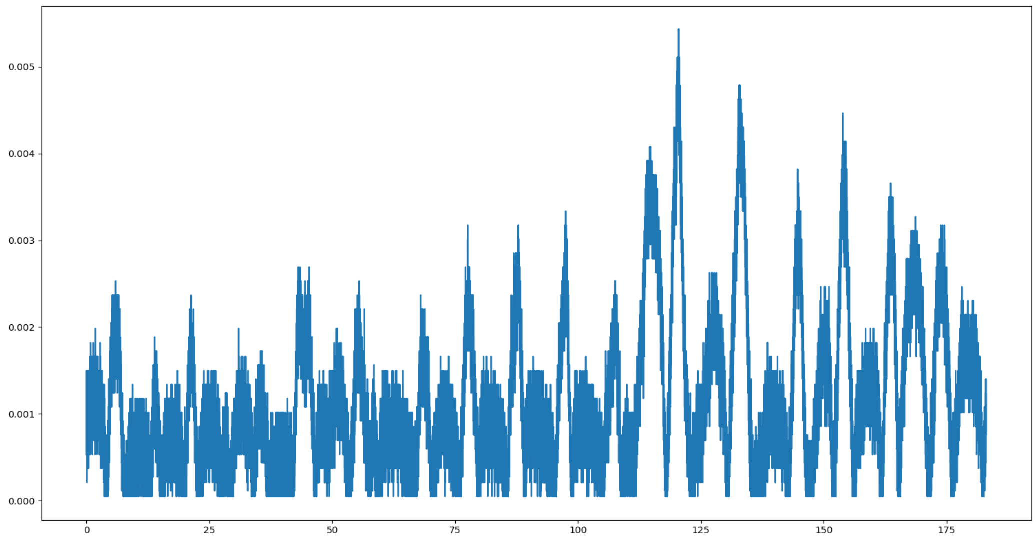
1.(a)使用 Python 搭配 numpy 函數庫做數理分析以及 matplotlib(matlab 作圖函數庫)做繪圖，解釋如程式碼中的註解。

```python
import numpy as np
import matplotlib.pyplot as plt

fptr=open("./physiology/sub019_baseline_C1.csv",'r')
read_data = fptr.read()
ts_data = read_data.split('\n')#time_splitted data , using next line as split token
len_of_tsdata = len(ts_data)-1 #get 2d vector each row's end length
print(len_of_tsdata)

for i in range(len(ts_data)):
    ts_data[i]=ts_data[i].split(',') #split each row with ',' token

#initilaization
extracted_final_data = np.append(ts_data[0][1:],ts_data[1][1:])
#concatenate altogether to match a 1d list and plot accodring to time
for i in range(2,len_of_tsdata):
    extracted_final_data = np.append(extracted_final_data,ts_data[i][1:])
    #concatenate altogether to match a 1d list and plot accodring to time
extracted_final_data = extracted_final_data.astype(float) #change the datatype to floating point version
time_axis = np.arange(0,183,0.001,float)
#from 0 ro 182 seconds, the difference between each is 0.001 (as sampling freq = 1000 Hz)
#plot as the floating type
plt.plot(time_axis,np.absolute(extracted_final_data))
#using matlab api for drawing x axis as time_axis and y axis as np.absolute(extracted_final_data)
plt.show()
```

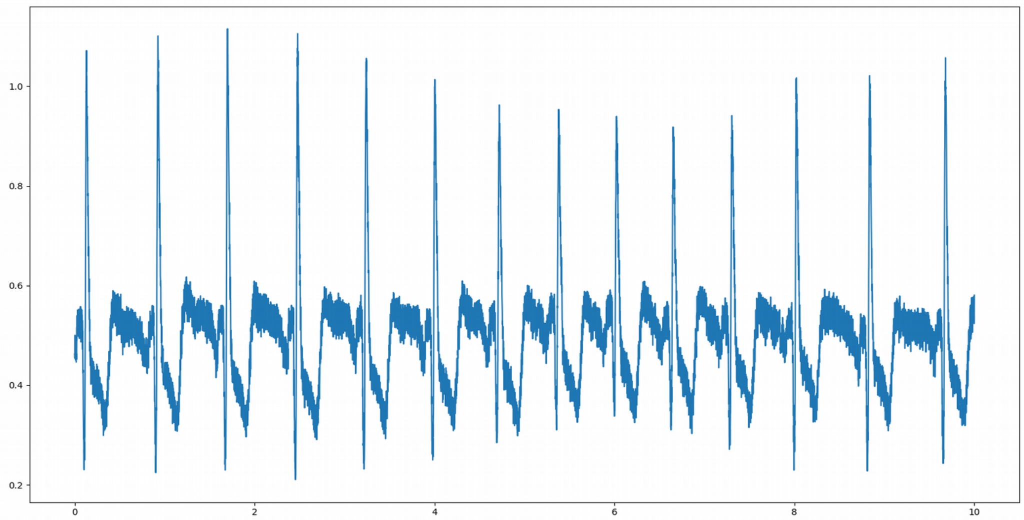
1(b)180sec x:time(sec) y:amplitude



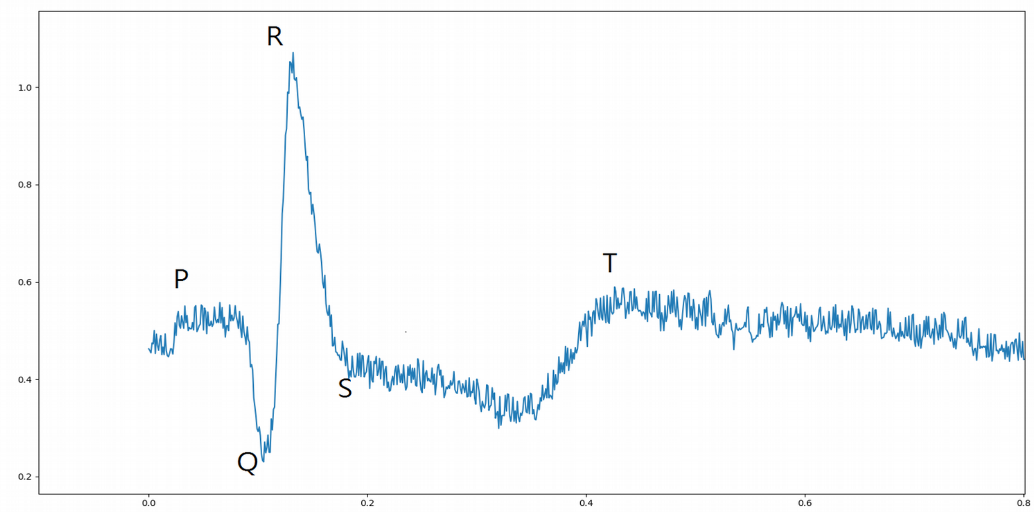
1(c)180sec x:time(sec) y:amplitude



1(d)180sec x:time(sec) y:amplitude



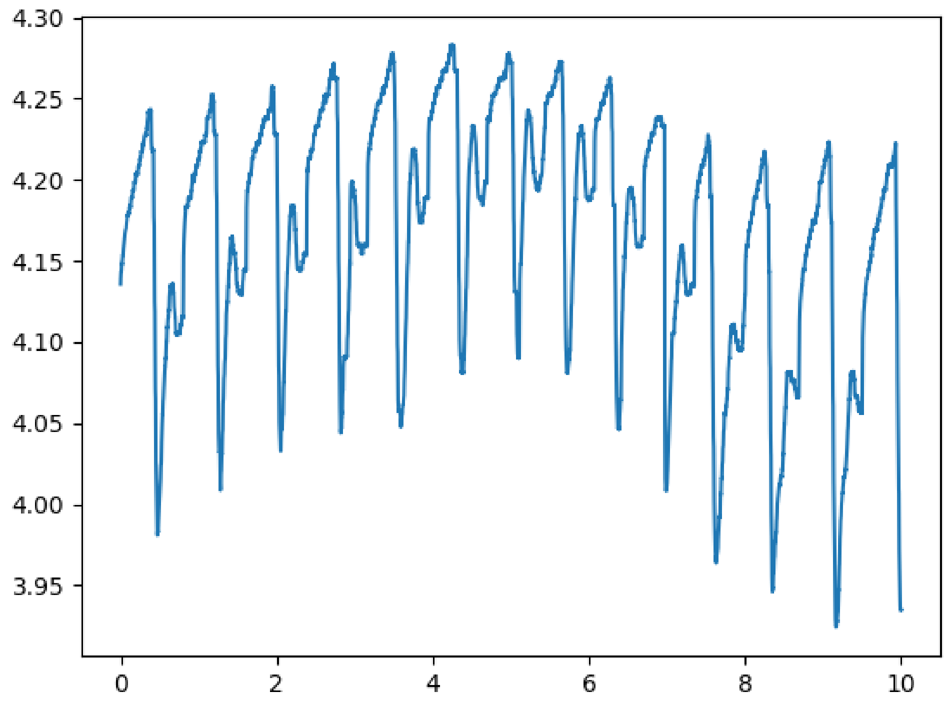
After zoom scale to see PQRST point



程式架構同 1(b)，一樣使用 Python 搭配 numpy 函數庫做數理分析以及 matplotlib(matlab 作圖函數庫)做繪圖，解釋如程式碼中的註解。

```
import numpy as np
import matplotlib.pyplot as plt

fptr=open("./physiology/sub019_baseline_ECG.csv",'r')
read_data = fptr.read()
ts_data = read_data.split('\n')
#time_splitted data , using next line as split token
len_of_tsdata = len(ts_data)-174
#get 2d vector one row's end length,originally 184 seconds,
#now down to 10 secs (so minus by 174)
print(len_of_tsdata)

for i in range(len(ts_data)):
    ts_data[i]=ts_data[i].split(',') #split each row with ',' token

#initilaization
extracted_final_data = np.append(ts_data[0][1:],ts_data[1][1:])
#concatenate altogether to match a 1d list and plot accodring to time
for i in range(2,len_of_tsdata):
    extracted_final_data = np.append(extracted_final_data,ts_data[i][1:])
    #concatenate altogether to match a 1d list
#and plot accodring to time
extracted_final_data = extracted_final_data.astype(float)
#change the datatype to floating point version
time_axis = np.arange(0,10,0.001,float)
#from 0 ro 182 seconds, the difference between each is 0.001 (as sampling freq = 1000 Hz)
#plot as the floating type
plt.plot(time_axis,np.absolute(extracted_final_data))
#using matlab api for drawing x axis as time_axis and y axis as np.absolute(extracted_final_data)
plt.show()
```

1(e)180sec x:time(sec) y:amplitude

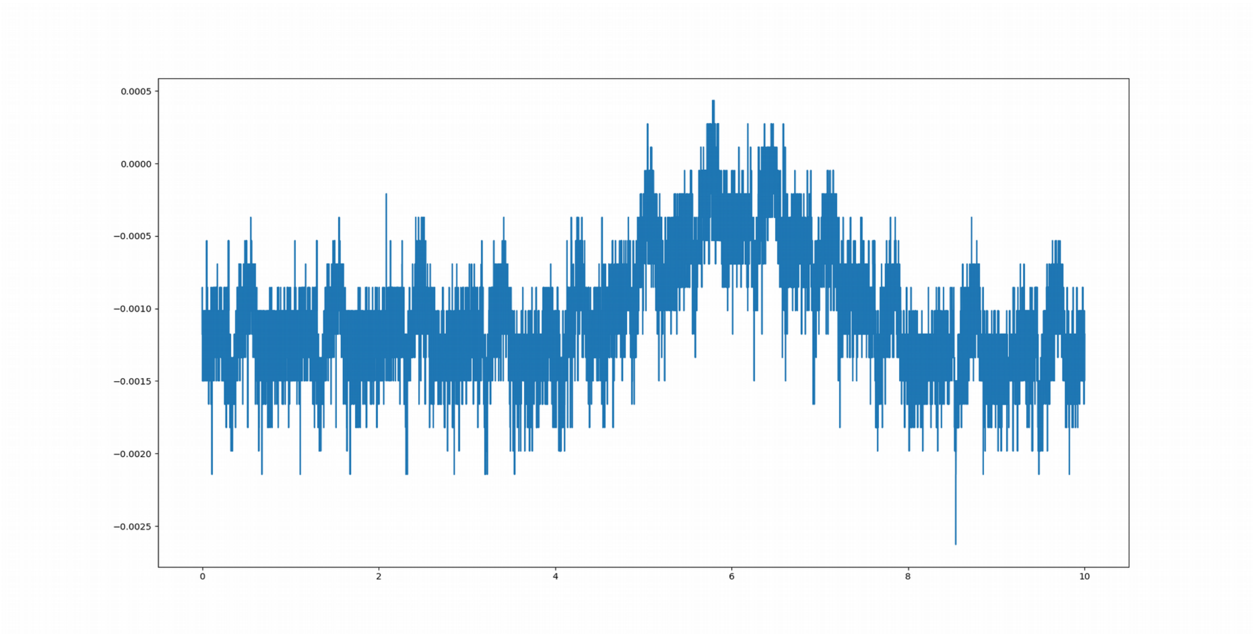


程式架構同 1(b)，一樣使用 Python 搭配 numpy 函數庫做數理分析以及 matplotlib(matlab 作圖函數庫)做繪圖，解釋如程式碼中的註解。

```python
import numpy as np
import matplotlib.pyplot as plt

fptr=open("./physiology/sub019_baseline_BloodPulse.csv",'r')
read_data = fptr.read()
ts_data = read_data.split('\n')
#time_splitted data , using next line as split token
len_of_tsdata = len(ts_data)-174
#get 2d vector one row's end length,originally 184 seconds,
#now down to 10 secs (so minus by 174)
print(len_of_tsdata)

for i in range(len(ts_data)):
    ts_data[i]=ts_data[i].split(',')
    #split each row with ',' token

#initilaization
extracted_final_data = np.append(ts_data[0][1:],ts_data[1][1:])
#concatenate altogether to match a 1d list and plot accodring to time
for i in range(2,len_of_tsdata):
    extracted_final_data = np.append(extracted_final_data,ts_data[i][1:])
    #concatenate altogether to match a 1d list
#and plot accodring to time
extracted_final_data = extracted_final_data.astype(float)
#change the datatype to floating point version
time_axis = np.arange(0,10,0.001,float)
#from 0 ro 182 seconds, the difference between each is 0.001 (as sampling freq = 1000 Hz)
#plot as the floating type
plt.plot(time_axis,np.absolute(extracted_final_data))
#using matlab api for drawing x axis as time_axis and y axis as np.absolute(extracted_final_data)
plt.show()
```

2(a) 取樣頻率同為一千，與方才 1 相同，取十秒的區間來看

- 胸部 x:time(sec) y:amplitude(>0 表示胸部前凸反之亦然)

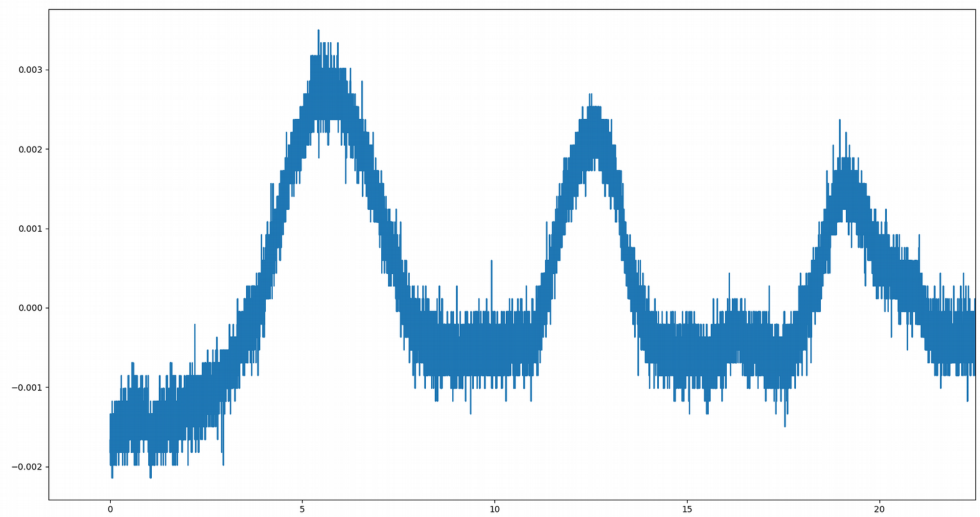


由圖中推測頻率約為 0.8~1Hz 之間，平均振幅

```
Avreage amplitude :  0.000410451612903
```

- 腹部，因為自己腹部變化較慢故取二十多秒區間

x:time(sec) y:amplitude(>0 表示腹部前凸反之亦然)

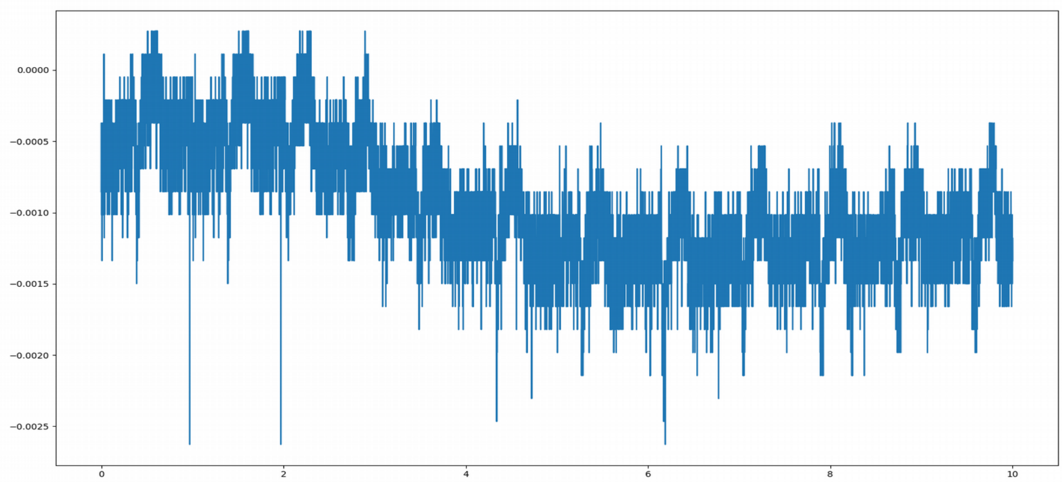


由圖中推測約為 0.1~0.125Hz，平均振幅

```
Avreage amplitude :  0.0024955
```

2(b) 取樣頻率同為一千，與方才 1 相同，取十秒的區間來看

- 胸部 x:time(sec) y:amplitude(>0 表示胸部前凸反之亦然)

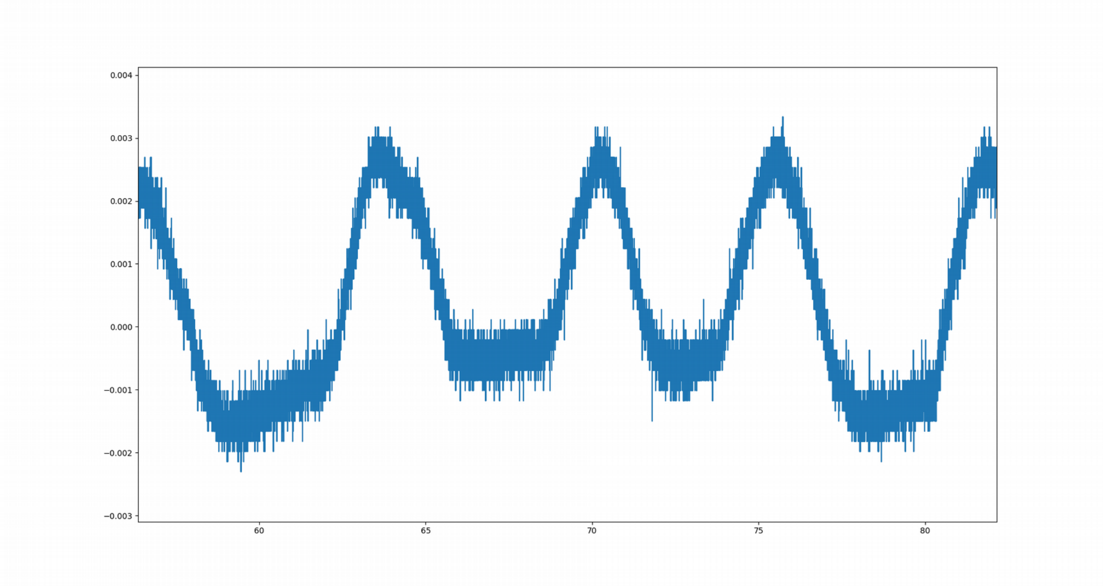


由圖中推測約為 1~1.2Hz，平均振幅

```
Avreage amplitude :  0.000582888888889
```

- 腹部，因為自己腹部變化較慢故取二十多秒區間

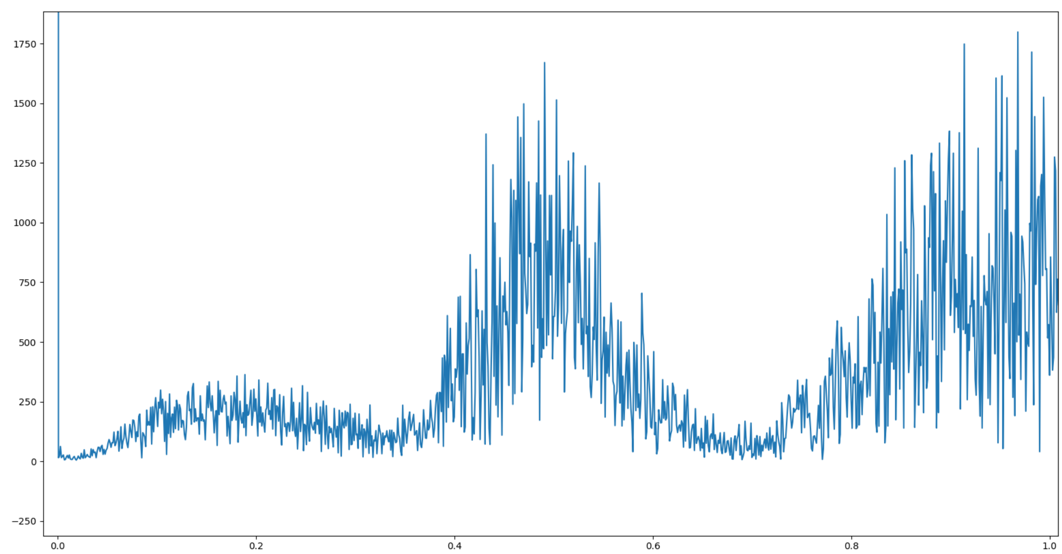
x:time(sec) y:amplitude(>0 表示腹部前凸反之亦然)



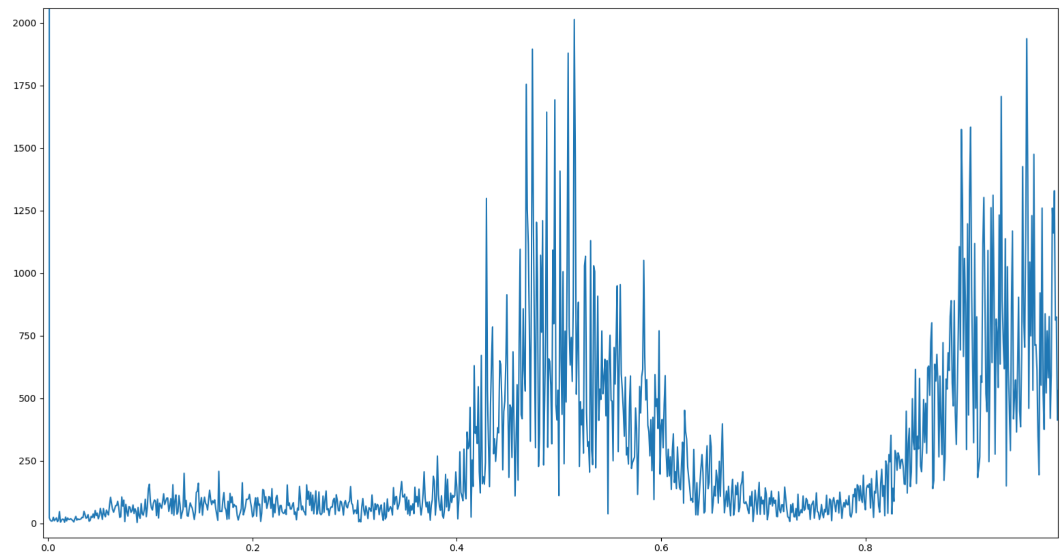
由圖中推測頻率約為 0.2Hz，平均振幅

```
Avreage amplitude :  0.00303147619048
```

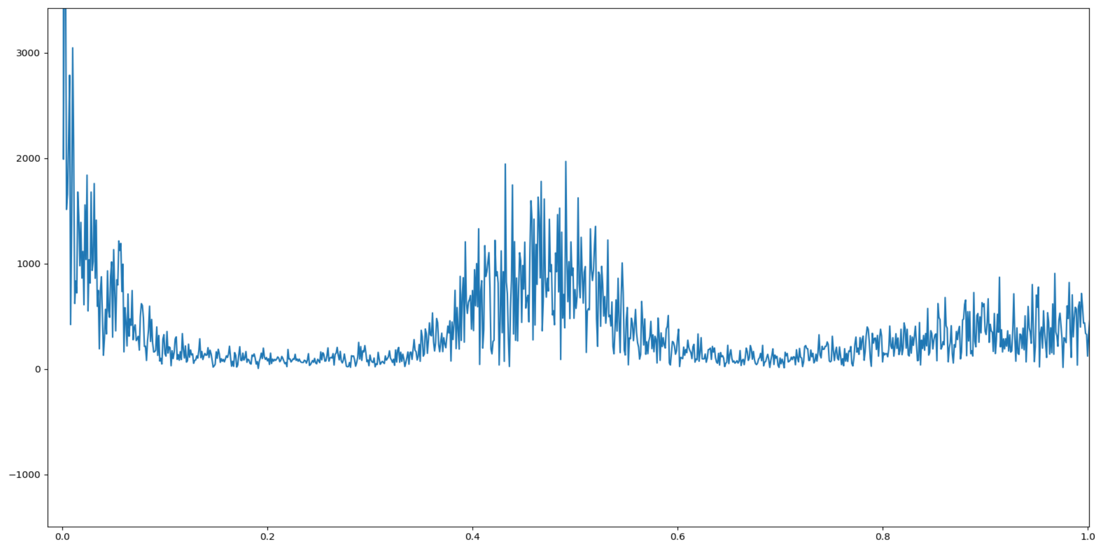
2(c) x:freq(Hz) y:amplitude



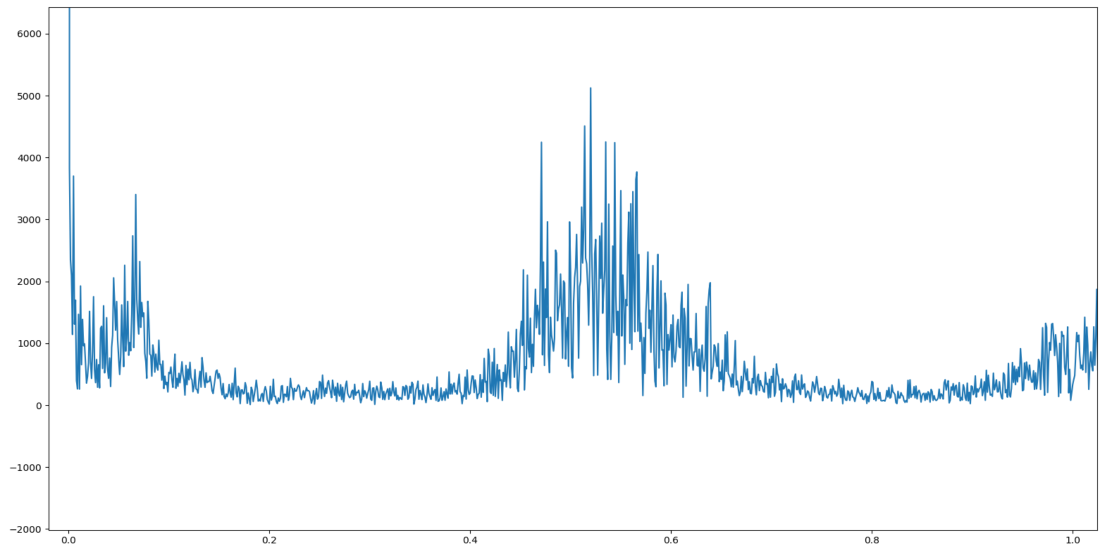
2(d) x:freq(Hz) y:amplitude



2(e) x:freq(Hz) y:amplitude



2(f) x:freq(Hz) y:amplitude



3(a)第一個影片是惡靈古堡為恐怖片，影片中有許多噁心的鬼怪對戰，強烈的視覺效果撼動人心，常會有鬼怪冒出嚇人，因此可見胸部起伏有正有負，情緒較為緊張所導致，自己緊張時呼吸起伏程度會加大，而在第六秒時突然升到更為正的地方，推測是有深呼吸動作讓胸部起伏更大了。

第二個影片是普通英雄聯盟，沒有太多情緒起伏，因此胸部的起伏程度也沒有太多變化，維持相當平均的一條線。

不過對於兩個影片胸部起伏實驗數據都在 y 軸偏負地方較多，個人推測是因為實驗時可能綁帶沒有束緊，以為胸部都在較為凹陷的程度所致。

3(b)腹部呼吸在兩個影片比較下，卻反出乎意料之外，原先以為惡靈古堡的影片令我受驚而呼吸急促，但從兩個實驗腹部起伏的圖形來看並沒有差太多，推測是在影片之前有訓練過規律的腹部呼吸法，而且是在很穩定的心情下去觀看這裡兩部影片，況且雖恐怖卻是在寧靜教室觀看並無太多氣氛的夾雜，因此不至於讓情緒起伏太大，除非是用 VR 玩類似沉默之丘的恐怖實境遊戲才會。

不過對於腹部和胸部，腹部調控的頻率遠低於胸部覺得相當奇怪，可能是自己還是比較習慣胸部呼吸的原因，且對於腹部呼吸都較為遲緩。

3(c) 從圖中來看腹部呼吸的起伏較胸部大，因為胸部的肌肉比例較腹部為多且有肋骨、橫膈、劍突等等都是硬骨組織，較不容易受內外肋間肌與橫膈肌牽引太多；而腹部軟組織較多，在腹肌的牽引下很容易就有明顯的起伏，這也是人們較容易透過腹部觀察呼吸的原因，而且自己也習慣透過腹部呼吸才有如此的實驗結果。

3(d)這個實驗結果也頗令人意外，以為可怕的影片會讓心律起伏比較大，然從上圖的結果看下，可見第二個影片的波峰較高，事實上兩個影片都有讓人情緒起伏的地方，第一個是不安，以及突如其來的驚嚇，第二個是變化萬分的賽事，

各種進攻、突圍，也在英雄聯盟影片中看到熟悉的電競選手，情緒霎時興奮了起來，個人推測**由興奮與遊戲的緊張相較於恐怖的緊張給我的情緒起伏前者是比較多的**，這也和我的 self report 對於情緒強烈程度指數的量表相同。
3(e)結果類似於 3(d)，因為心臟負責將血流打至全身各地，因此心臟和脈搏的訊號其波型走勢會類似，確實也有符合這項推測，且在第二個影片的起伏程度較第一個多，呼應了 3(d)情緒影響的想法。

GitHub backup URL:
https://github.com/Alfons0329/Signals_and_Systems_Final_Project