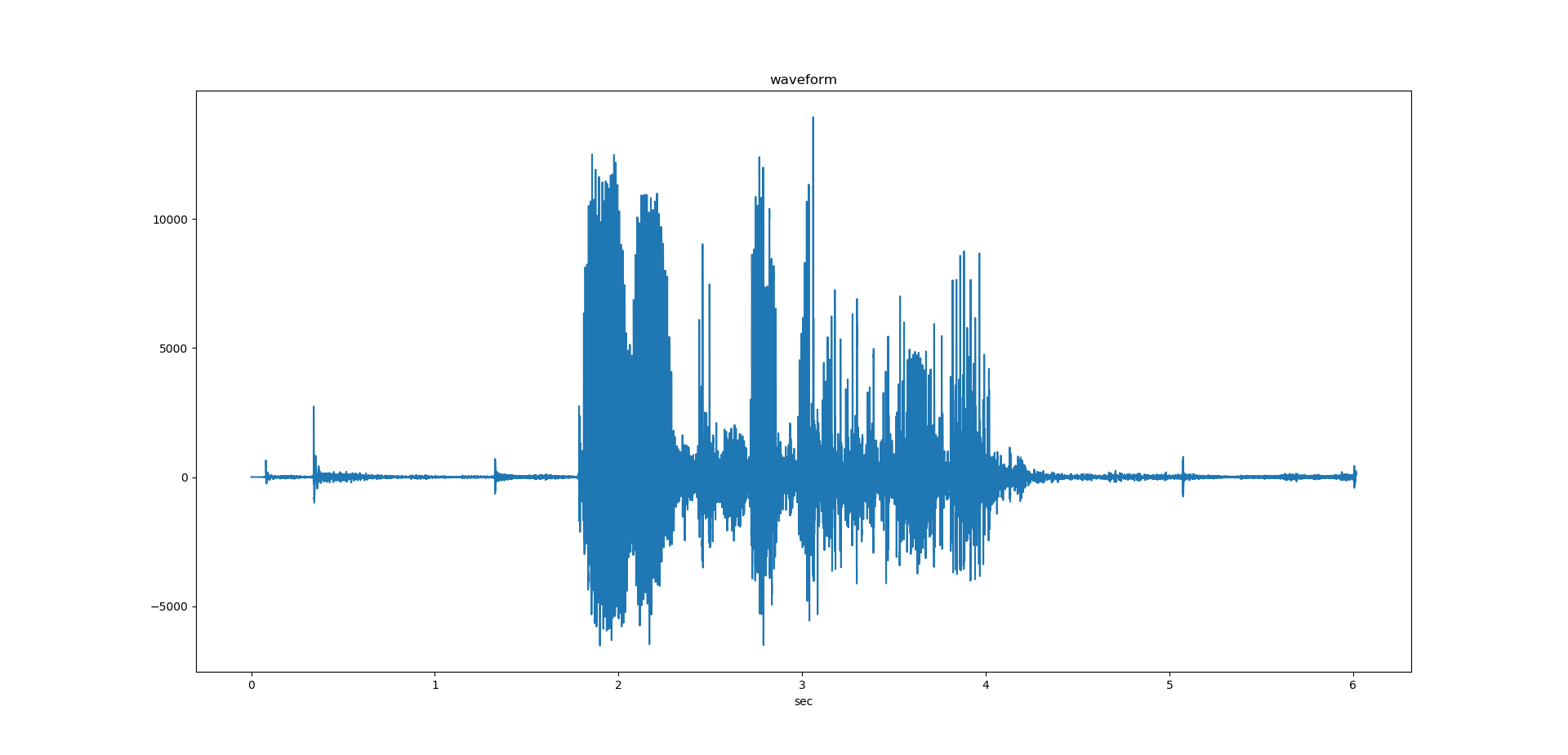
**程式與詳細註解:**

from scipy.io import wavfile  
from numpy import ndarray  
import matplotlib.pyplot as plt  
import numpy as np  
import scipy.signal as sig  
from scipy.ndimage.filters import gaussian\_filter1d  
  
data: ndarray  
# read wav file  
sample\_rate, data = wavfile.read("test5.wav")  
# select channel 1  
data = np.array(data[:, 0], dtype=np.int64)  
# remove potential dc bias  
data = data - data.mean()  
# total frame number  
frame\_num = data.shape[0]  
# 10 ms rect window  
window = int(sample\_rate\*0.01)  
print(sample\_rate, frame\_num, window)  
  
# x axis for frame number  
xax = np.arange(0, frame\_num/sample\_rate, 1/sample\_rate)  
  
# plot waveform  
plt.plot(xax, data)  
plt.title("waveform")  
plt.xlabel("sec")  
plt.show()  
  
# plot short time energy  
En = []  
norm = np.square(data)  
# use the formula of the course ppt, collect previous n frames  
for i in range(window, frame\_num):  
 En.append(norm[i-window+1: i+1].sum())  
# the 0~(window-1)-th frame use the window-th value to padding  
val = En[0]  
for i in range(0, window):  
 En.insert(0, val)  
  
plt.plot(xax, En)  
plt.title("short time energy")  
plt.xlabel("sec")  
plt.show()  
  
# calculate avg magnitude  
Mn = []  
ab = abs(data)  
# use the formula of the course ppt, collect previous n frames  
for i in range(window, frame\_num):  
 Mn.append(ab[i-window+1: i+1].sum())  
# the 0~(window-1)-th frame use the window-th value to padding  
val = Mn[0]  
for i in range(0, window):  
 Mn.insert(0, val)  
  
  
# plot zero crossing rate  
Zn = []  
# if >= 0 mark true, else false  
sgn = [(True if x >= 0 else False) for x in data]  
sgn = np.array(sgn)  
pre = [0]  
# if true->false or false->true mark 1, else 0  
for i in range(1, frame\_num):  
 pre.append(1 if sgn[i] != sgn[i-1] else 0)  
pre = np.array(pre)  
# use the formula of the course ppt, collect previous n frames  
for i in range(window, frame\_num):  
 Zn.append(pre[i - window+1: i+1].sum()/window)  
# the 0~(window-1)-th frame use the window-th value to padding  
val = Zn[0]  
for i in range(0, window):  
 Zn.insert(0, val)  
  
plt.plot(xax, Zn)  
plt.title("zero crossing rate")  
plt.xlabel("sec")  
plt.ylabel("times/frame")  
plt.show()  
  
# end point detection  
# assume first 100ms is silent  
pre = 10\*window  
Mn = np.asarray(Mn)  
En = np.asarray(En)  
Zn = np.asarray(Zn)  
# determine itu, itl, izct  
itu = np.sort(Mn[0:pre]).max()\*2  
itl = np.sort(Mn[0:pre]).max()  
izct = np.sort(Zn[0:pre]).mean()  
# find start point  
# 1.from first frame to find first point(ieu) >= itu  
# 2.from ieu look back to find first point(n1) < itl  
# 3.if zero crossing rate of n1 >= 3\*izct then look back to find first point(n1) >= itl  
# 4. n1 will be start point  
ieu = np.where(Mn >= itu)[0][0]  
for i in range(ieu, -1, -1):  
 if Mn[i] < itl:  
 n1 = i  
 break  
if int(Zn[n1])/izct >= 3:  
 n1 = np.where(Zn >= izct)[0][0]  
 pass  
print("start at frame "+str(n1))  
# find end point  
# 1.from last frame to find first point(ieu) >= itu  
# 2.from ieu look back to find first point(n1) < itl  
# 3.if zero crossing rate of n1 >= 3\*izct then look back to find first point(n1) >= itl  
# 4. n1 will be end point  
ieu = np.where(Mn >= itu)[0][-1]  
for i in range(ieu, -1, -1):  
 if Mn[i] < itl:  
 n1 = i  
 break  
if int(Zn[n1])/izct >= 3:  
 n1 = np.where(Zn >= izct)[0][0]  
 pass  
print("end at frame "+str(n1))  
  
  
def fill(fl):  
 *"""  
 use pre 3 and post 3 frames's mean to fill all <=0 entries in f1* ***:type*** *fl: ndarray* ***:param*** *fl: input data  
 """* for i in range(0, len(fl)):  
 if fl[i] <= 0 and i >= 3 and i+3 < len(fl):  
 fl[i] = (fl[i-3:i].sum()+fl[i+1:i+4].sum())/6  
 pass  
  
  
# plot pitch  
freq = []  
# voice threshold  
threshold = 1/10\*Mn.max()  
# reference: pitch function in matlab  
# https://www.mathworks.com/help/audio/ref/pitch.html#mw\_245063d9-5bf9-4930-8fbc-2659faa9b551  
# set window size and overlap point number (refer to above link's suggested value)  
window = int(sample\_rate\*0.052)  
overlap = int(sample\_rate\*0.042)  
for i in range(0, frame\_num, window-overlap):  
 # if mean magnitude < threshold(sound is too small) => set frequency to 0 directly  
 if Mn[i:i+window].mean() < threshold:  
 freq.append(0)  
 continue  
 pass  
 x = data[i:i+window]  
 # use auto correlation and center-clipping  
 u=x.mean()+0.5\*x.std()  
 l=x.mean()-0.5\*x.std()

m=x.mean()  
 for i in range(0,len(x)):  
 if x[i]>u:  
 x[i]-=m  
 elif x[i]<l:  
 x[i]-=m  
 else:  
 x[i]=0  
 r = np.correlate(x, x, mode='full')[len(x)-1:] # the autocorrelation produces a symmetric signal,only care about the right half  
 # set range to 50 - 400 hz due to human speaking  
 r[0:int(sample\_rate/400)] = 0  
 r[int(sample\_rate/50):] = 0  
 peak = sig.find\_peaks(r, height=0)[0]  
 if peak.shape == (0,):  
 # if can not find any peak in Rn then set frequency to lower bound 50  
 freq.append(50)  
 else:  
 # calculate frequency by peak of Rn  
 freq.append(sample\_rate/peak[0])  
freq = np.array(freq)  
# fill <= 0 value in frequency  
fill(freq)  
  
plt.subplot(2, 1, 1)  
# apply median filter  
plt.plot(np.arange(0, frame\_num/sample\_rate, 1/sample\_rate\*(window-overlap)), sig.medfilt(freq))  
plt.title("pitch with median filter by acf and center-clip")  
plt.ylabel("hz")  
plt.xlabel("sec")  
  
plt.subplot(2, 1, 2)  
# apply gaussian filter  
plt.plot(np.arange(0, frame\_num/sample\_rate, 1/sample\_rate\*(window-overlap)), gaussian\_filter1d(freq, sigma=3))  
plt.title("pitch with gaussian filter by ncf")  
plt.ylabel("hz")  
plt.xlabel("sec")  
  
plt.tight\_layout()  
plt.show()

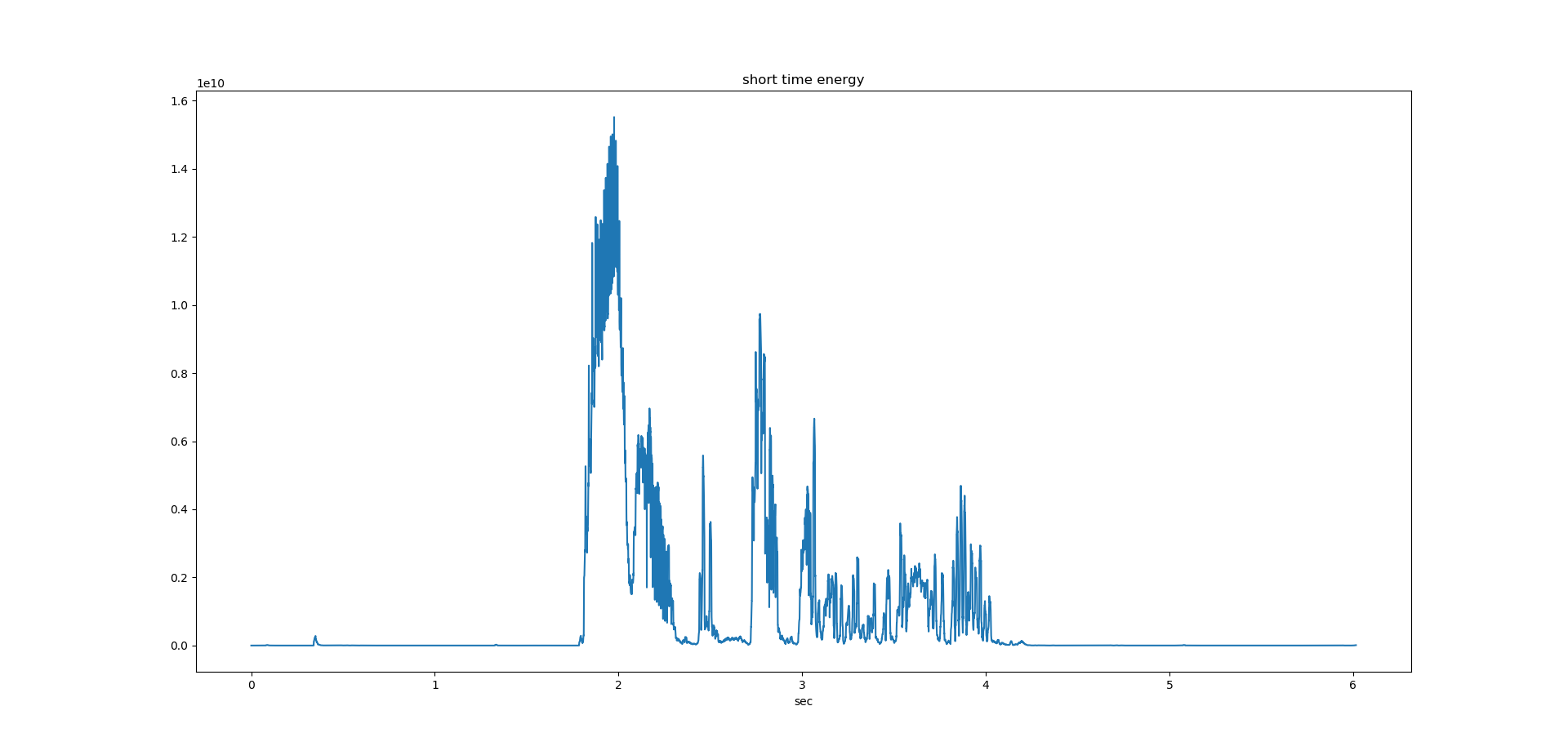
**輸出與簡介:**

1.waveform(waveform.png)



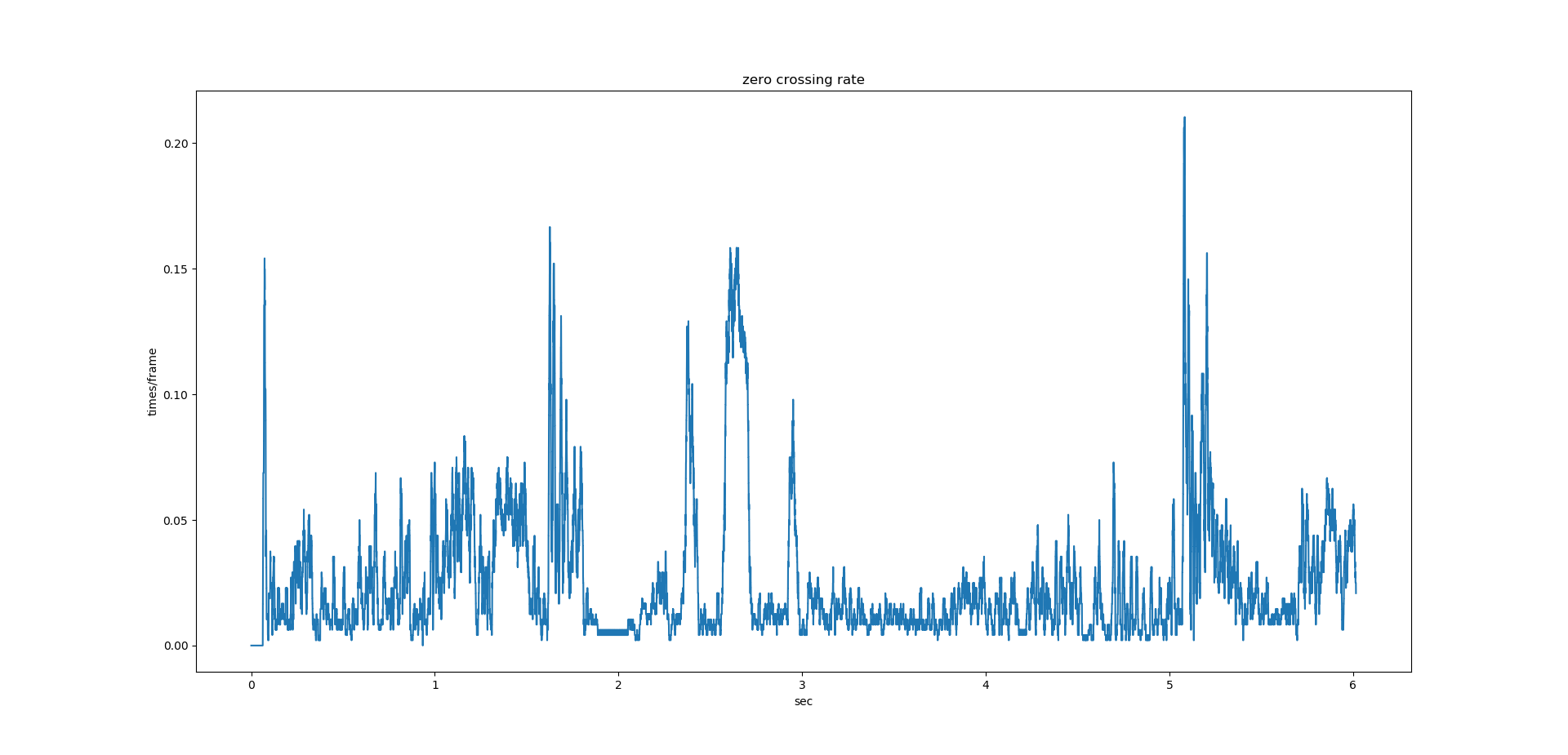
讀出來後取單聲道畫圖

2. short time energy(energy.png)



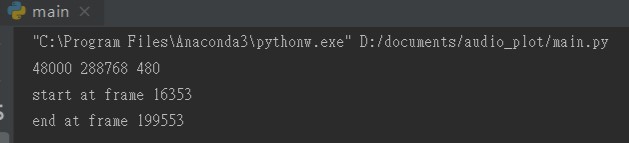
依照上課投影片公式取前n frames(含自己)做平方和計算，但因第0到window-1個frame取前n frames會超過原點(若超過的都當0會與其他點計算方式不同會有偏差)，故用第window個frame的計算結果來填充前方的值。

3.zero crossind rate(zcr.png)



依照上課投影片公式取前n frames(含自己)做計算，由>=0變化到<0視為一次cross，最後除以window size得 次數/frame為過零率，但因第0到window-1個frame取前n frames會超過原點(若超過的都當0會與其他點計算方式不同會有偏差)，故用第window個frame的計算結果來填充前方的值。

4.end point detection(endpoint.png)



依照上課投影片之方法，前100ms先算itu, itl, izct

itu 取前100ms中max average magnitude \*2

itl =取前100ms中max average magnitude

izct =取前100ms中zero crossing rate平均

再來由以下步驟

# find start point

1.from first frame to find first point(ieu) >= itu

2.from ieu look back to find first point(n1) < itl

3.if zero crossing rate of n1 >= 3\*izct then look back to find first point(n1) >= itl

4. n1 will be start point

# find end point

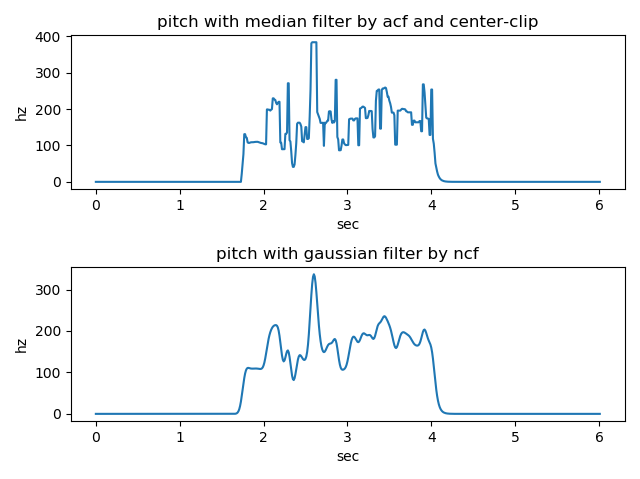
1.from last frame to find first point(ieu) >= itu

2.from ieu look back to find first point(n1) < itl

3.if zero crossing rate of n1 >= 3\*izct then look back to find first point(n1) >= itl

4. n1 will be end point

5.pitch(pitch.png)



一開始我本來是用上課投影片上的center clip和auto correlation公式來做，也就是每個frame每次回頭取n frames來算，但是做出來頻率的震盪範圍非常大(0-1k多hz)，劃出的曲線也很混亂的上下跳動，後來我在matlab的函示庫中發現有現成的pitch function，我就進去看他是如何運作的，原來他是採用一個個window再加上window間的overlap來計算auto correlation function

( 參考https://www.mathworks.com/help/audio/ref/pitch.html#mw\_245063d9-5bf9-4930-8fbc-2659faa9b551)，我就把他的概念改寫道我的python code中，結果畫出來的頻率圖比之前好多了，但是前後還是有一些上上下下的震盪，於是我再把average magnitude太小的當成雜音不列入頻率計算(先直接當成0，之後再用前後點平均值填補)，最後再用median filter得上圖之上半，下半圖是因我覺得曲線還是有點坑坑疤疤於是再用更強力的gaussian filter來更進一步smooth，結果曲線變得非常平滑了，但是缺點是gaussian filter可能會造成部分值失真。