Advanced Digital Signal Processing Report

109511207 蔡宗儒

Beamforming for Ultrasound Imaging

QUESTION 1.1: Complete the delay-and-sum beamforming code involving image formation of the image pixels in the polar coordinates. (See below)

(a) the distance between each transmit (Tx) channel and the target (The variable is called Tx_dist), and the distance between each receive (Rx) channel and the target (called Rx_dist).

```
%%%% Delay and Sum (DAS) to realize the beamforming
% Rx distance  (You should calculate the distance between each image pixel to ALL receive channels)
Rx_dist = zeros(Nr,Nx);           % initialization of the Rx distance
% Rx_dist =                                          ------------------------------------------------>   You  need complete the code here
chx_rx = repmat(chx,Nr,1);
Rx_dist = sqrt((chx_rx - range.'*sin_th).^2 + (range.'*cos_th).^2);  % Calculate Rx distance

% Tx distance (You should calculate the distance between each image pixel to ii-th transmit channel)
Tx_dist = zeros(Nr,Nx);           % initialization of the Tx distance
% Tx_dist =                                          ------------------------------------------------>   You  need complete the code here
chx_tx = repmat(chx(:,ii),Nr,Nx);
Tx_dist = sqrt((chx_tx - range.'*sin_th).^2 + (range.'*cos_th).^2);  % Calculate Tx distance
```

(b) Summing up all sub-images (called subbf) corresponding to all Tx-Rx combinations to obtain the final beamformed image (called bf).

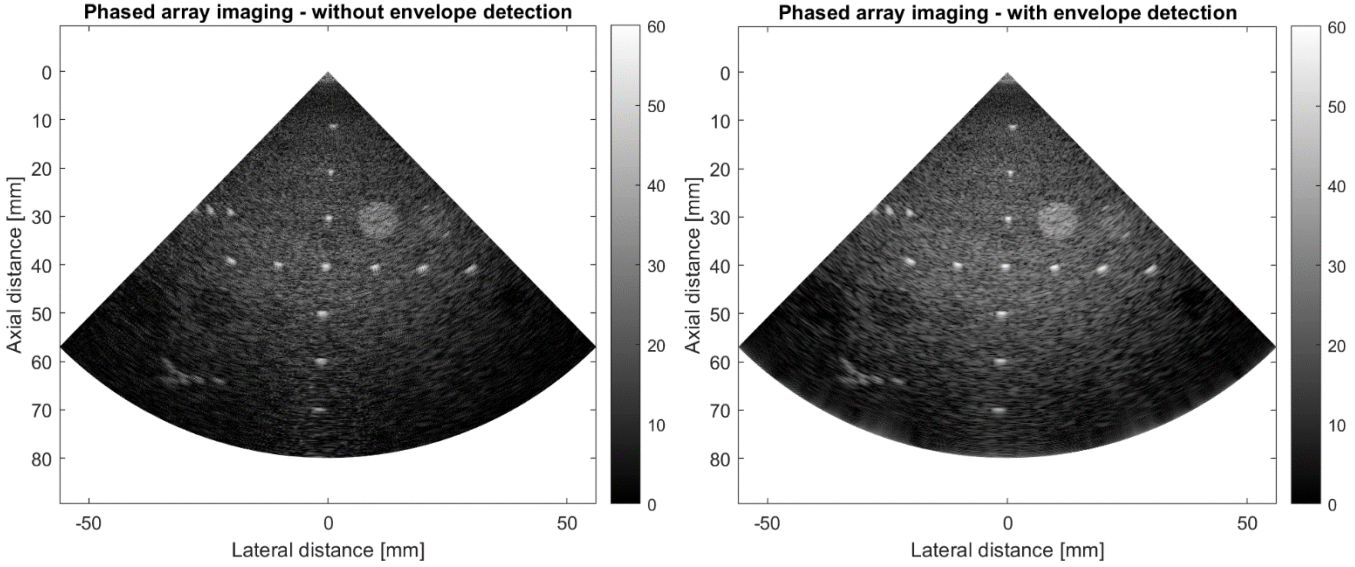
```
    %%% Find the sample according to "delayindx" and do the summation
    % subbf(:,jj) =   % sub-image for ii-th Tx emission       ------------------------------------------------>   You  need complete the code here
    for kk = 1 : Nx
        for mm = 1 : Nr
           if delayindx(mm,kk) > 0 && delayindx(mm,kk) <= size(bb_data,1)
               subbf(mm,jj) = subbf(mm,jj) + bb_data(delayindx(mm,kk),kk);
           end
        end
    end
end
bf = bf + subbf;   % summation among all sub-images.
```

(c) Envelope detection of the beamformed image (called bf_env). Envelope detection can be done by one of the following two methods: Hilbert transform and base-band demodulation.

```
%%% Envelope detection
bf_env = zeros(Nr,no_lines); % initialization of envelope detected result
%bf_env =                    % envelope detected result              ------------------------------------------------>   You  need complete the code here
bf_env = abs(hilbert(bf));  % with doing envelope detection
```

利用 Hilbert transform 得到 analytic signal，再用絕對值來得到其 magnitude。

(d) Once you complete (a)-(c), the code will generate the final image.



What is the beamformed image without doing envelope detection (i.e., skipping step (c))?

A: 不做 envelope detection 的 beamformed image 表示累加後的射頻訊號。Image 保留了原始的 RF 訊號的特性，並因為 RF 訊號的震盪性，所以包含了正負值。

What is the beamformed image after step (d)?

A: 做 envelope detection 的 beamformed image(即 bf_env)表示 RF 包絡的 amplitude。Image 強調了反射訊號的 magnitude，提供了清晰且可解釋的結構表示。因此 Envelope detection 將 RF 訊號轉換成 unipolar 的 form，強調了反射特性。

Can you tell the difference between the two images? Please explain your findings.

A:

Without envelope detection:

- Image 顯示了原始的 RF 訊號，其中包含正負震盪。
- 由於像素強度變化迅速，圖像難以解釋，夾帶大量高頻雜訊。
- Noise 較大，不適用於直接診斷目的。

With envelope detection:

- Image 顯示了原始的 RF 訊號的 magnitude，提供更清晰地反射特性。
- 圖像較為 smooth，更加強調和顯示了細節。
- 有效地去除了高頻震盪，可以擷取訊號的低頻成分，較適用於診斷目的。

總結：

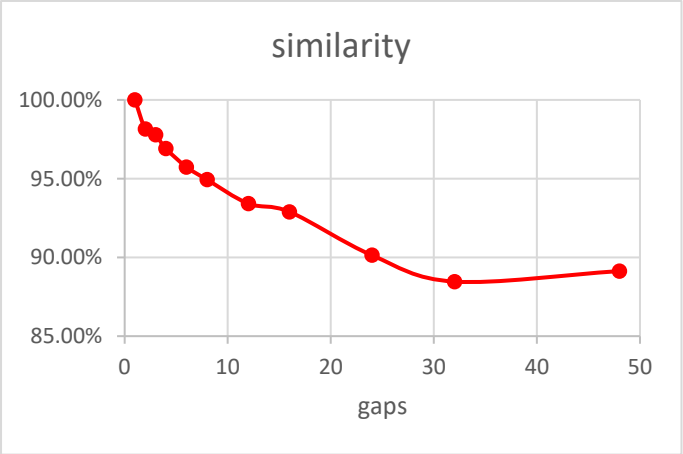
可觀察出，有做 envelope detection 的圖(上面右圖)，較清楚地顯示了結構的邊界跟特徵，圖片較為明亮、對比度較高，影像較為 smooth。這是因為 envelope detection 的主要作用是可以將實部訊號拓展至負數平面，從而獲得 magnitude 或是 phase 等資訊，已進行更進一步的分析、計算和成像等等。這可以抑制高頻雜訊、減少 noise，並可以提高清晰地，從而使醫務人員更容易分析和診斷 B-mode 的 image。

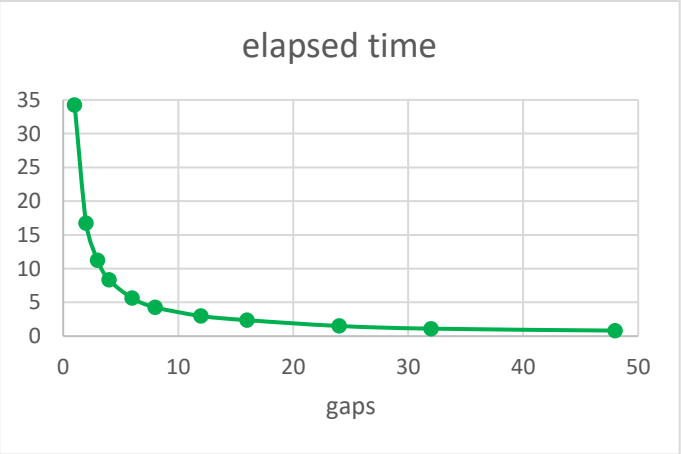
QUESTION 1.2: The image frame rate using full 96 Tx events is slow because it requires 96 emissions. As explained in Appendix A, Tx events are possibly reduced while maintaining similar image quality. Can you design your own Tx event sequence (i.e., specify which Tx channels are emitted) to speed up the frame rate at the expense of slight image quality degradation? **Explore your own Tx sequence and answer how fast your frame rate could be. Justify that the image quality of your design is comparable to that of the full 96 Tx events.**

我在 Quiz1_1.m 裡將有做 envelope detection 且沒有做 reduce emissions 的影像存入 ori_img.mat 中，在 Quiz1_2.m 裡將有做 envelope detection 且有 reduce emissions 的影像存入 emi_redu_img.mat 中，並在 Quiz1_2.m 裡用了 tic、toc 來計算 elapsed time 以及用了 function ssim()來計算兩張影像的 similarity。

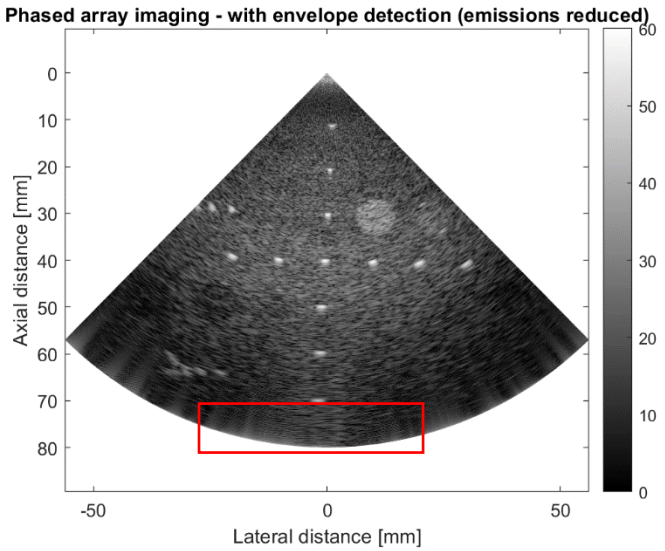
首先我設置了 gaps 大小，利用 for ii= 1: gaps :Nt 來手動調整 Tx event sequence，讓 sequence 為等距的，而結果如下表格所示。

gaps	# of emissions	similarity	elapsed time
1	96	100.000 %	34.217038 s
2	48	98.155 %	16.738005 s
3	32	97.775 %	11.228331 s
4	24	96.906 %	8.322062 s
6	16	95.736 %	5.649631 s
8	12	94.933 %	4.273529 s
12	8	93.413 %	2.985416 s
16	6	92.893 %	2.364771 s
24	4	90.133 %	1.504010 s
32	3	88.463 %	1.108744 s
48	2	89.130 %	0.820187 s
96	1	81.224 %	0.411604 s





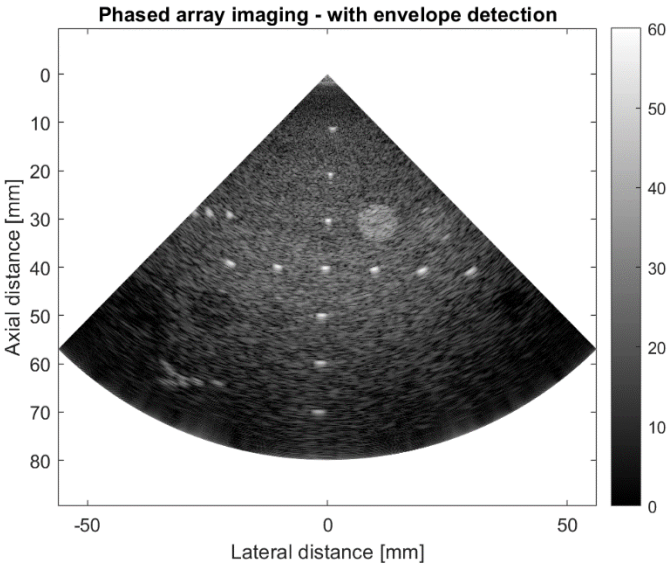
可以看出當 gaps 越大時，similarity 會越小，影像的解析度也會越差。這是因為 gaps 越小就代表等效孔徑內涵蓋的 emitted channels 就越多，可以捕捉到的角度也就越大，所以就會有更佳的解析度。然而這樣產生的 image 會有較為嚴重的亮暗條紋干涉產生，gaps 越大時干涉狀況也會越嚴重，如下圖為將 gaps 設為 6 產生的 image。

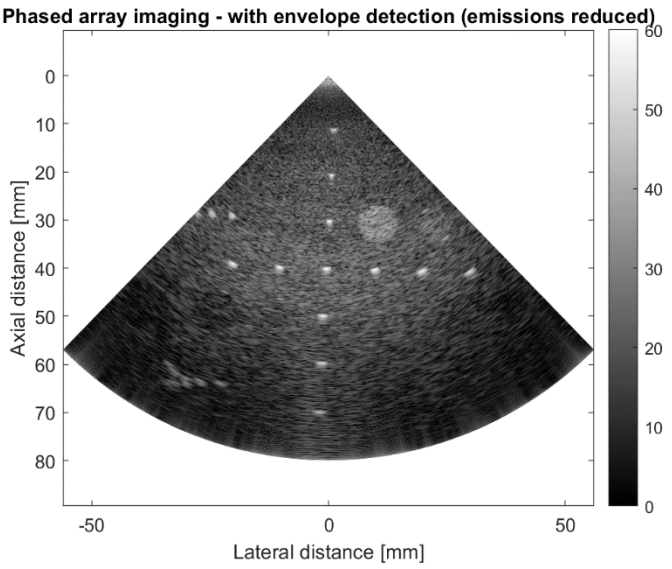


因此我選擇將固定 gaps = 6 的 Tx event sequence 做微調，在維持# of emissions 不變的情況下，調整 emissions 間的 gaps，降低規律性並進而降低 noise 間的相關性。

而選擇 gaps = 6 的原因是因為能讓 similarity 維持在接近 96%以上，並讓計算量減少至 1/6，elapsed time 降低 6~7 倍左右。

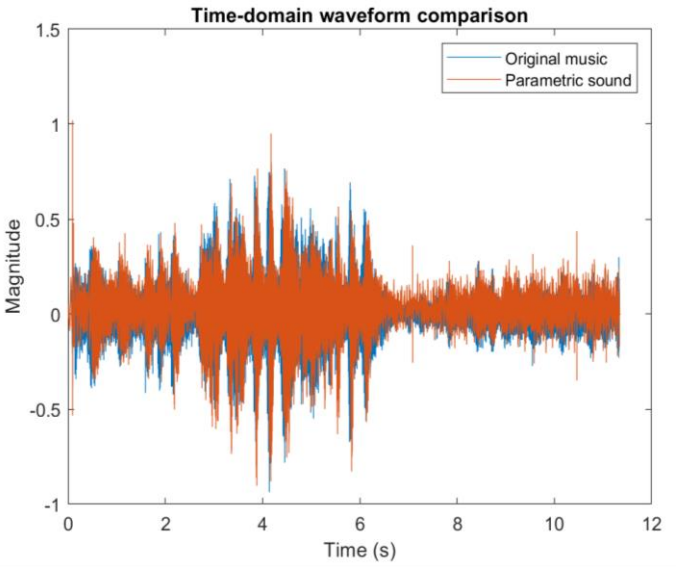
因此最後調整 Tx event sequence 為[1:gaps:13, 20:gaps+1:48, 49:gaps+1:77, 84:gaps:96]，其中 gaps 設為 6，即 Tx event sequence 為[1, 7, 13, 20, 27, 34, 41, 48, 49, 56, 63, 70, 77, 84, 90, 96]。如此一來 similarity 可以達到 96.286%，並讓 elapsed time 維持在 6s 以內。最後生成的 image 如下右圖，而下左圖為未調整 Tx event sequence 的原 image。

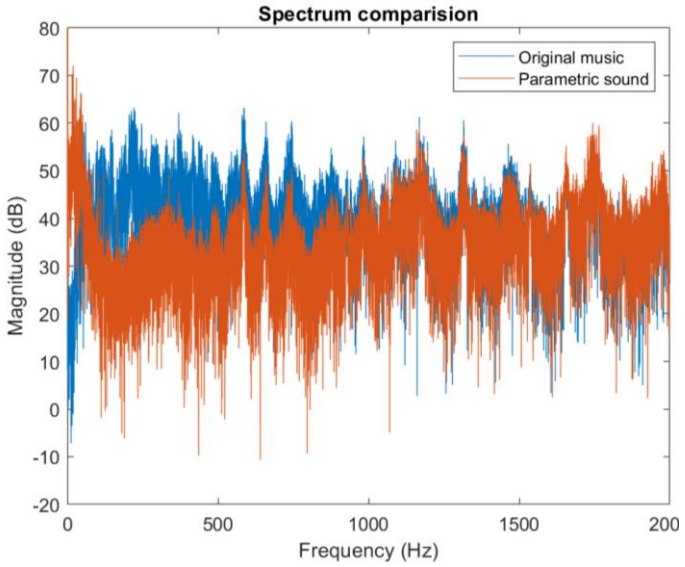




Time-frequency analysis (TFA) for music

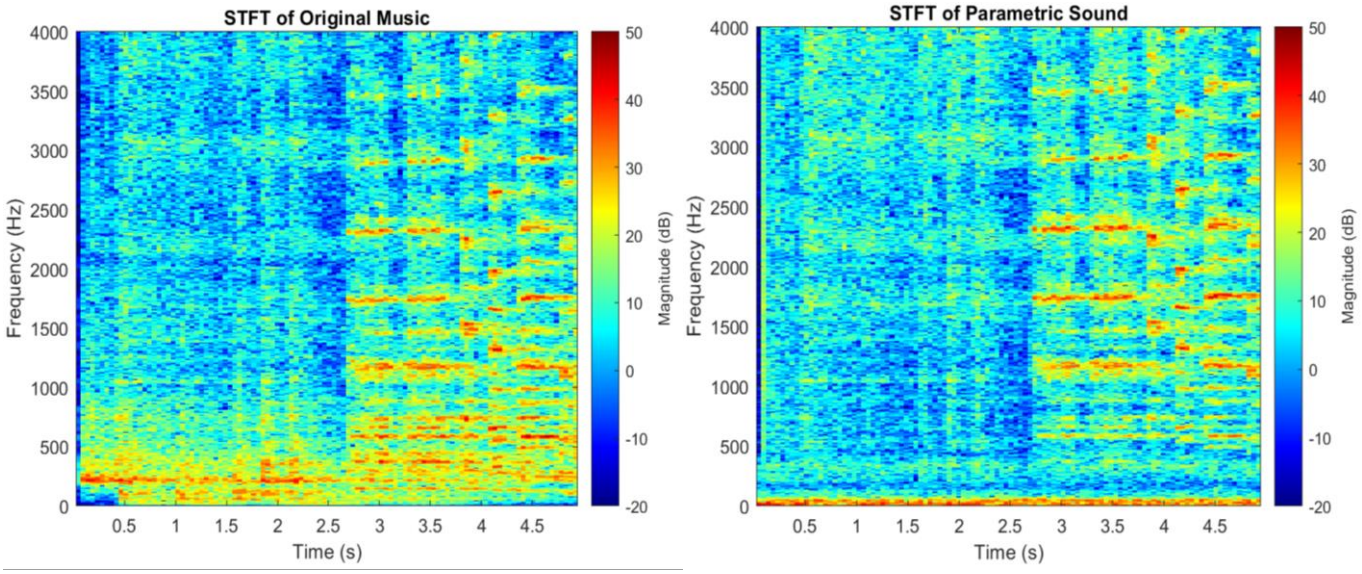
QUESTION 2.1: Run the code and you can see two figures. The first figure is the waveform comparison between the original sound and the parametric sound. The second figure is the spectrum comparison. **From these two figures, is there any difference between two sounds? Explain your answer.**





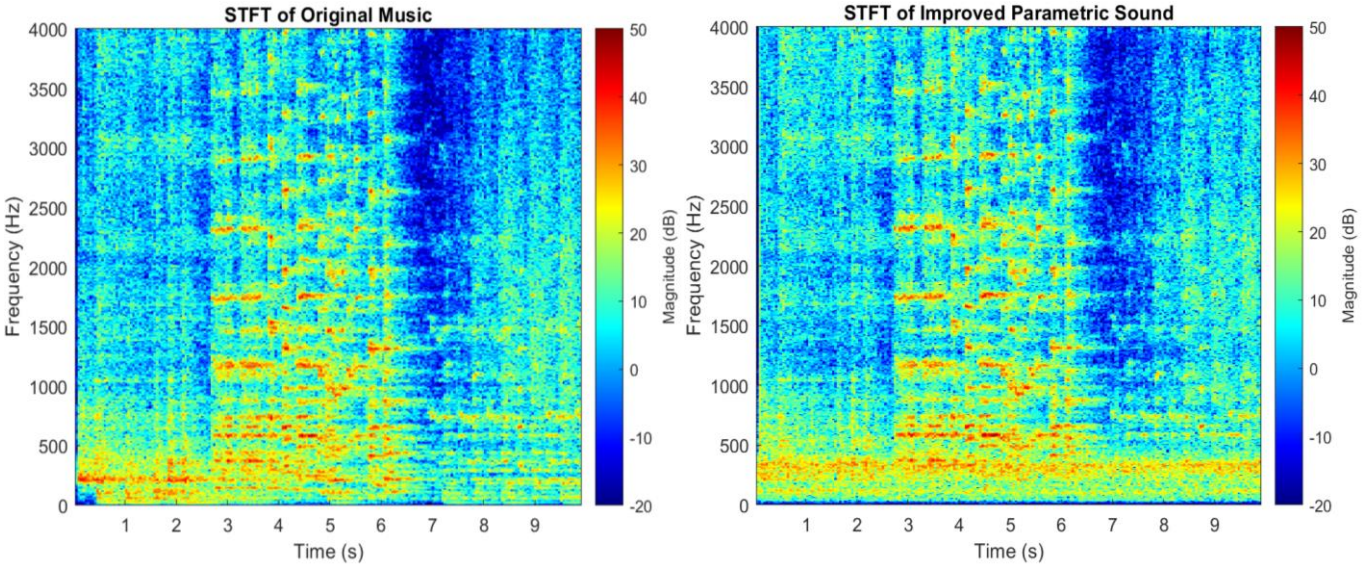
在時域上，兩者的 waveform 看起來非常接近。但在頻譜上可以觀察到，parametric sound 的頻譜在 100Hz 以內的頻段有比起 original music 更多的 noise，這表示 parametric sound 存在較多的低頻噪聲。而在差不多 150-900 Hz 的頻段處，parametric sound 的頻譜的 magnitude 明顯要小於 original music，顯示出了較為嚴重的衰減，這表示 parametric sound 可能失去原本較為明顯的低頻訊號。這是因為 parametric sound 在生成過程中使用了 nonlinear effects，導致低頻成分不在麥克風的收音方向，生成效率較低，進而導致低頻訊號損失並產生不同程度的噪聲。

QUESTION 2.2: The spectrum shown in Question 2.1 has no timing information. So your job is to provide STFT result for two sounds. Complete the code (you can use MATLAB function "stft" or write by yourself, or other internet/ChatGPT resources), but you need to specify the four parameters: Window size, Windowing function, FFT length, and Overlapping window length between two spectrum calculations. **Show your result in dB within the time range of 0-5 seconds and the frequency range of 0-4000Hz. What is the difference between two STFT results?**



對於時間軸的前 2.5s，original music 集中在 1000Hz 以內，而 parametric sound 的低頻段則幾乎都衰減掉了，僅剩下低頻的噪聲而已。對於時間軸的 2.5s 之後，parametric sound 依然保留了 1000Hz 至 4000Hz 的頻段，所以失真情況較沒有前 2.5s 嚴重。而從整個 STFT 的結果可以發現，parametric sound 在頻段 1000Hz 以下的訊號都衰減掉了，且 parameter sound 在 100Hz 以下的噪聲明顯多於 original music。STFT 的結果更能觀察出 parametric sound 在生成過程中因為使用了 nonlinear effects，導致低頻成分生成效率較低，進而導致低頻訊號損失並產生不同程度的噪聲。

QUESTION 2.3: Based on the difference you find in Questions 2.1 and 2.2, can you design a filter (may be low-pass, high-pass, or band-pass filter) to improve the parametric sound so that the sound quality is comparable to the original sound? Explain your design and OUTPUT your sound as a mp4 file.



我將頻段拆成 4 段來看，分別是 0-100Hz、100-500Hz、500-1000Hz 跟 1000Hz 以上。這樣拆的原因是觀察 QUESTION 2.2 的 STFT 可以發現，parametric sound 在 0-100Hz 的噪聲最多最嚴重，而在 100-500Hz 和 500-1000Hz 的訊號成分都有一定的衰減，且 100-500Hz 衰減地更為嚴重，而在 1000Hz 以上則沒有過於明顯的衰減，因此我將頻段拆成這 4 段來做不同的設計。

首先對於 0-100Hz 來說，低頻噪音過為嚴重，所以我設計了一個 high-pass filter 來移除低頻的噪音。再來對於 100-1000Hz 來說，我用了兩個 band-pass filter 來分別擷取 100-500Hz 跟 500-1000Hz 兩個頻段的訊號，並對這兩個頻段做不同幅度的放大。又因為 100-500Hz 衰減明顯更為嚴重，所以我將這段訊號放大 8 倍，而將 500-1000Hz 放大 2 倍。最後對於 1000Hz 以上的訊號來說，因為高頻部分稍微有點刺耳，所以我再用一個 band-pass filter 擷取這段訊號，並乘上 0.9 稍微降低高頻訊號的 magnitude。最後將這些訊號疊加在一起，得到 improved parametric sound。

觀察 STFT 可以發現比起原本的 parametric sound，已經有盡可能地還原大部分訊號，但在 500Hz 以下的低頻訊號依然沒有還原地非常好。再來可以發現因為將 100-500Hz 的頻段放大 8 倍的原因，所以在 6s 後可以聽到有一陣不應該出現的低頻聲(如下圖)，而這也是我無法有效解決的問題之一。