

Q1

1. Discuss how you determine the filters.

我選擇 low-pass filter、high-pass filter、bandpass filter。

因為 low-pass filter 只留下比某頻率低的部分，high-pass filter 只留下比某頻率高的部分，

bandpass filter 只留下在某頻率範圍內的部分。適合用來分離 audio signals。

<pre>N = 1001; f1 = 400; f2 = 780; band=[400 ,780]; [low_pass_signal, low_pass_filter] = myFilter(y_input, fs, N, 'Blackman', 'low-pass', f1); [band_pass_signal, band_pass_filter] = myFilter(y_input, fs, N, 'Blackman', 'bandpass', band); [high_pass_signal, high_pass_filter] = myFilter(y_input, fs, N, 'Blackman', 'high-pass', f2);</pre>	<p>在 HW2_Q1.m 呼叫 myFilter。 其中 f1,f2,band 是 fcutoff(cut-off frequency or band frequencies)</p>
---	---

2. How you implement the filter and convolutions to separate the mixed song and one/multiple fold echo.

(1)在 myFilter.m 裡實做 3 種 filters。

low-pass filter	<pre>if strcmp(filterName, 'low-pass')==1 for n = -middle:middle if n == 0 outputFilter(n+middle+1)=2*f_c; else outputFilter(n+middle+1)=sin(2*pi*f_c*n)/(pi*n); end end end</pre>	<p>參考講義 76 頁的 algorithm 及 72 頁的 ideal impulse responses (1)outputFilter 是代完公式產生的。 (2)$f_c = \text{fcutoff} / \text{fsample}$ (3)$\text{middle} = \text{floor}(N/2)$</p>
high-pass filter	<pre>elseif strcmp(filterName, 'high-pass')==1 for n = -middle:middle if n == 0 outputFilter(n+middle+1)= 1-2*f_c; else outputFilter(n+middle+1)= -sin(2*pi*f_c*n)/(pi*n); end end end</pre>	<p>跟 low-pass filter 做法差不多，只是代的 ideal impulse responses 不同。</p>
bandpass filter	<pre>elseif strcmp(filterName, 'bandpass')==1 for n = -middle:middle if n == 0 outputFilter(n+middle+1)= 2*(f_c(2)-f_c(1)); else outputFilter(n+middle+1)= sin(2*pi*f_c(2)*n)/(pi*n)- sin(2*pi*f_c(1)*n)/(pi*n); end end end</pre>	<p>跟 low-pass filter 做法差不多，只是代的 ideal impulse responses 不同。</p>

(2)在 myFilter.m 裡實做 windowing function。

<pre>if strcmp(windowName, 'Blackman')==1 for n = -middle:middle outputFilter(n+middle+1)= outputFilter(n+middle+1)*(0.42+0.5*cos((2*pi*n)/(N-1))+0.08*cos((4*pi*n)/(N-1))); end end</pre>	
--	--

根據講義 75 頁的 windowing function。代入公式，須注意的是:矩陣 index 要大於 0，所以要調整。

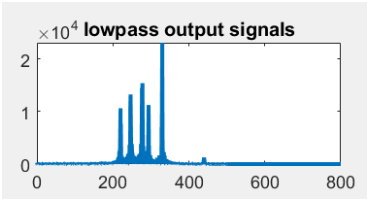
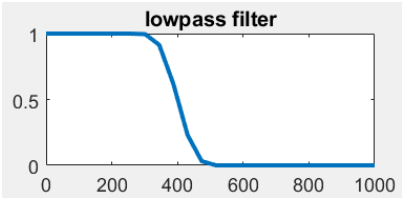
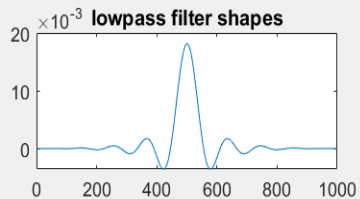
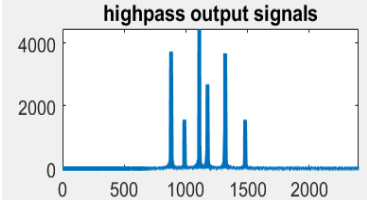
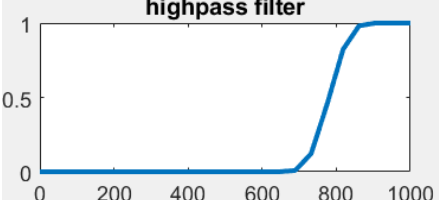
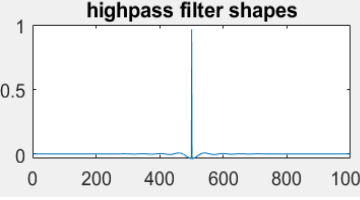
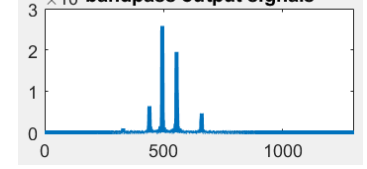
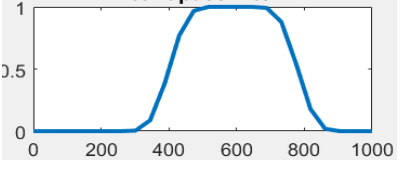
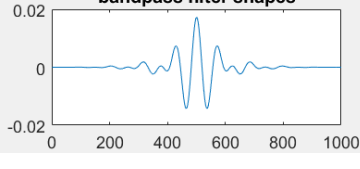
(3)convolutions

<pre> s= size(inputSignal); outputSignal = zeros(1, s(1)); for n = 1:s(1) for h = 1:N if n - h < 1 outputSignal(n)=outputSignal(n)+outputFilter(h)*0; else outputSignal(n)=outputSignal(n)+outputFilter(h)*inputSignal(n-h); end end end end </pre>	<p>參考講義 60 頁。</p> <p>判斷 $n-h$ 是否小於 1，</p> <p>若是 \rightarrow inputSignal=0</p> <p>若不是 \rightarrow inputSignal= inputSignal(n-h)</p> <p>然後 outputSignal 一直累加。</p>
--	--

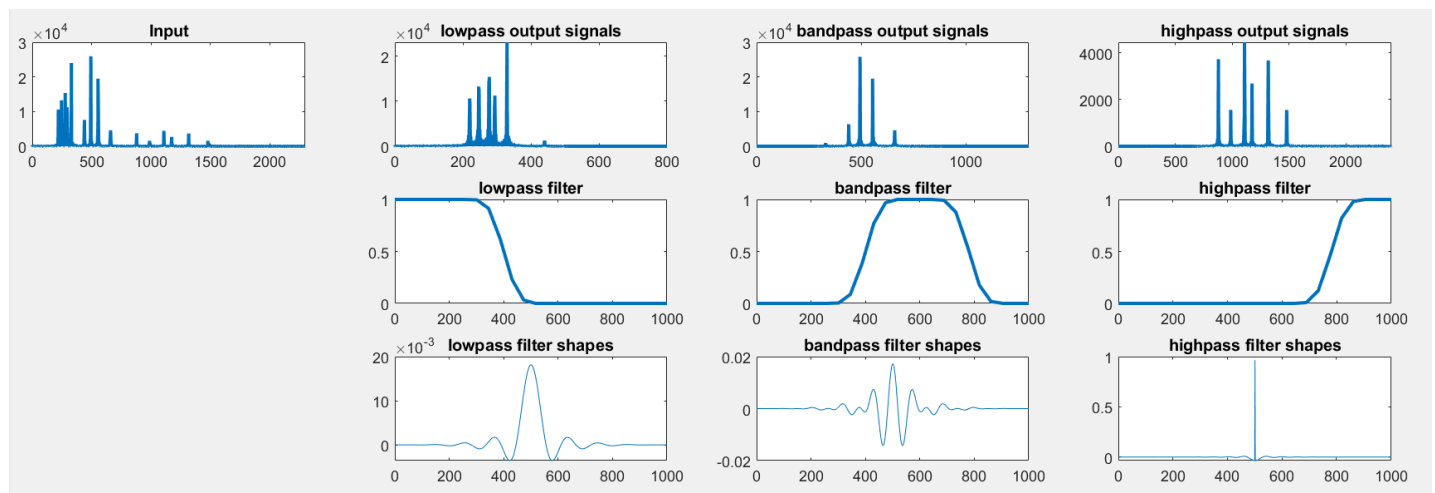
(4) one/multiple fold echo

one-fold echo	<pre> s=size(y_input); one_fold_echo = zeros(1, s(1)); multiple_fold_echo= zeros(1, s(1)); for i=1:s(1) if(i<=3200) one_fold_echo(i)=low_pass_signal(i); else one_fold_echo(i)=low_pass_signal(i)+0.8*low_pass_signal(i-3200); end end </pre>	<p>參考講義 65 頁，代入</p> $y[n]=x[n]+0.8*x[n-3200]$ <p>的公式</p> <p>其中 0.8 跟 3200 是可以調整的。</p>
multiple-fold echo	<pre> for i=1:s(1) if(i<=3200) multiple_fold_echo(i)=low_pass_signal(i); else multiple_fold_echo(i)=low_pass_signal(i)+0.4*multiple_fold_echo(i-3200); end end </pre>	<p>參考講義 65 頁，代入</p> $y[n]=x[n]+0.8*y[n-3200]$ <p>的公式</p> <p>其中 0.8 跟 3200 是可以調整的。</p> <p>我一開始用 0.8 但是有些聲音會破音，所以把 0.8 改成較小的 0.4。</p>

3. Compare spectrum and shape of the filters.

	spectrums of the output signal	spectrums of the filter	shapes of the filter
low-pass filter	 <p>圖 1</p>	 <p>圖 2</p>	 <p>圖 3</p>
high-pass filter	 <p>圖 4</p>	 <p>圖 5</p>	 <p>圖 6</p>
bandpass filter	 <p>圖 7</p>	 <p>圖 8</p>	 <p>圖 9</p>

- (1)因為 low-pass filter 的 fcutoff 取 400 所以(圖 1)signal 集中在 400 以下。
而 high-pass filter 的 fcutoff 取 780 所以(圖 4)signal 集中在 780 以上。
而 bandpass filter 的 fcutoff 取 400~780 所以(圖 7)signal 集中在 400~780 範圍內。
- (2) spectrums of the filter→即 (圖 2) 、(圖 5) (圖 8)
大致符合講義 64 頁 low-pass filter 、high-pass filter 、bandpass filter 的圖
low-pass filter 讓 f 低於 400 的通過，high-pass filter 讓 f 高於 780 的通過，bandpass filter 讓 f 介於 400~780 的通過。
- (3) shapes of the filter：觀察 3 種 filters 可以知道，頻率越高的 filter，output 出來的 shape 波形越集中。
- #### 4.結果



Q2

1.How you implement the bit reduction, audio dithering, noise shaping, low-pass filter, audio limiting and normalization.

(1)bit reduction

<pre> input=input*2^15; s=size(input); out=zeros(s(1),s(2)); for i=1:s(1) for j=1:s(2) %out(i,j)=bitsra(input(i,j),8); out(i,j)=round(input(i,j)/128)*128; end end </pre>	<p>先把 input 乘以 2^{15}。 再把 input/128 取四捨五入，再乘以 128。 就做好 bit reduction 了。</p>
--	--

(2)audio dithering

<pre> f_in=out; pd = makedist('Uniform',-1,1); addnoise = pdf(pd,f_in); for i = 1:s(1) for j = 1:s(2) if i ~= 1 f_in(i, j) = floor(f_in(i, j) + addnoise(i,j)) ; end end end </pre>	<p>參考講義 18 頁及課本。 *input 是使用 bit reduction 之後的音訊來做。 (1)對每個 f_in sample 加一個 random dithering value(取 uniform distribution)，再取 floor。 (2)取得 random: Get a random number between -1 and 1 from some probability density function。</p>
--	---

<p>課本:</p> <p>The assignment statement $F_in_i = F_in_i + D_i + cE_{i-1}$ dithers and noise shapes the sample. Subsequently, $F_out_i = \lfloor F_in_i \rfloor$ quantizes the sample.</p>	

(3) noise shaping

<pre>fin=round(f_in/128)*128; c = 1.2; f_out = fin; for i = 1:s(1) for j = 1:s(2) if i ~= 1 f_out(i, j) = floor(fin(i, j) + c *(fin(i, j)-f_out(i, j))); end end end</pre>	<p>參考講義 18 頁及課本。</p> <p>(1)先做 quantization 講 f_in 除以 128 取 round 再乘以 128。</p> <p>(2)把做完 quantization 的 fin 代入課本的式子 做 noise shaping。</p> <p>(3)c 值是自己試出來的。</p>
--	---

<p>課本:</p> <p>The assignment statement $F_in_i = F_in_i + D_i + cE_{i-1}$ dithers and noise shapes the sample. Subsequently, $F_out_i = \lfloor F_in_i \rfloor$ quantizes the sample.</p> <p>E_i is the error resulting from quantizing the ith sample after dithering and noise shaping.</p> <p>For $i = -1$, $E_i = 0$. Otherwise, $E_i = F_in_i - F_out_i$.</p>	
---	--

(4) low-pass filter

<pre>[f_out(:,1), ~] = myFilter(f_out(:,1) , fs, 1001, 'Blackman', 'low-pass', 1300); [f_out(:,2), ~] = myFilter(f_out(:,2) , fs, 1001, 'Blackman', 'low-pass', 1300);</pre>	
<p>呼叫 myFilter。其中 1300 是 fcutoff(cut-off frequency or band frequencies) 是自己試出來的，太低出來的聲音會很低沉。</p>	

(5) audio limiting

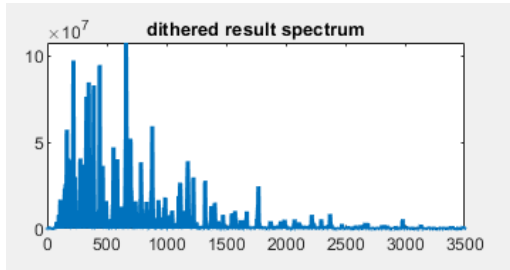
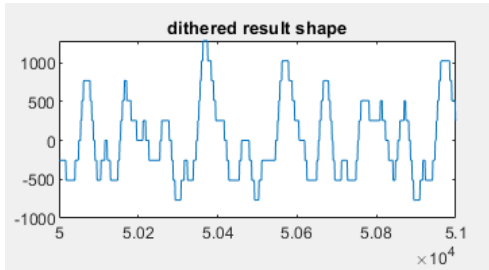
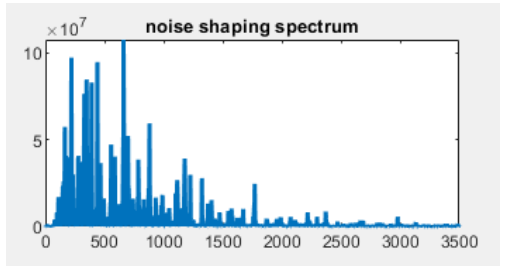
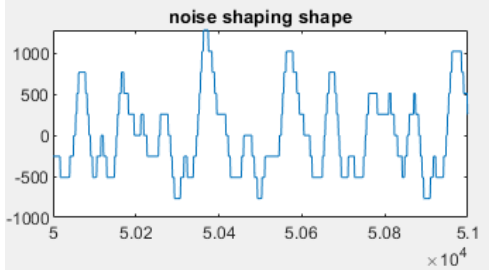
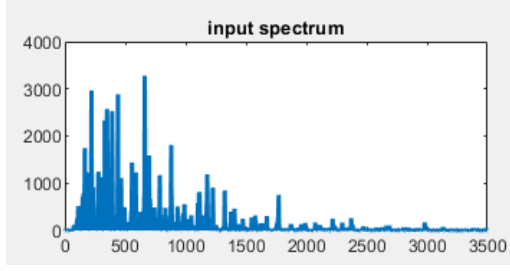
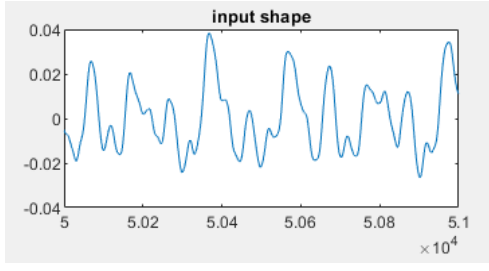
<pre>m1=-1.9*10^4; m2=1.9*10^4; for i = 1:s(1) for j = 1:s(2) if f_out(i,j)<=m1 f_out(i,j)=m1; end if f_out(i,j)>=m2 f_out(i,j)=m2; end end end</pre>	<p>參考講義 48 頁，使用 hard clipping，把超過 amplitudes 的濾掉(設成 max or min)。</p> <p>m1=-1.9*10^4 m2=1.9*10^4 是觀察 output shape 得來的值。</p>
---	---

(6) normalization

<pre>in_max = max(max(input)); out_max = max(max(f_out)); gain = in_max/out_max ; f_out =f_out*gain;</pre>	<p>參考講義 50 頁</p> <p>(1)先找到最高的 amplitude sample(input 和 output 都要)</p> <p>(2)決定 gain needed in amplitude to 提高 the highest amplitude to max amplitude</p>
--	--

(執行後 gain=0.9820)想讓音量跟原本的差不多。
(3)raise all samples by this amount

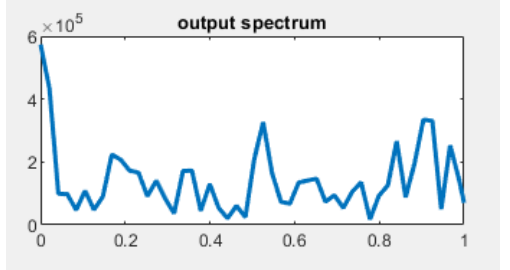
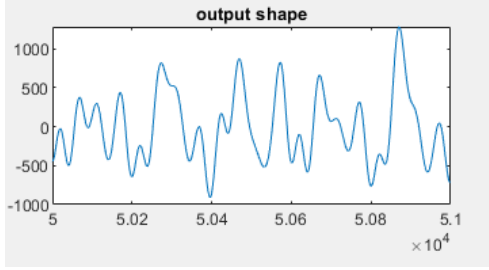
2. Discuss the effect of dithering and noise shaping according to the spectrums and shapes you plot

	spectrum	Shape(擷取部分)
dithering		
noise shaping		
input		

經過 Audio dithering 後，原本的 signals 加上 noise，noise 也會 follow the same pattern as 原本的 shape 的樣子。所以 shape 波形會跟 input 差不多，但起伏比較精細(較密集)。

經過 noise shaping，high frequency 的部分會稍微提高一點。

然後再呼叫 myFilter(使用 low-pass filter)來排除較高的 frequency。

output		
--------	---	--

3. 結果

