信号与系统大作业

——语音碎片我来拼拼听

无 56 班　　柴士佳　　2015011141

一、 题目要求：

现实中总是存在各种非理想因素，第 d 个设备真正记录到的信号应该表示为

$$y_d(t) = \begin{cases} F_d(x)(t + B_d), & 0 < t \leq E_d; \\ 0, & \text{elsewhere}, \end{cases} \tag{3}$$

其中 $F_d(x)(t)$ 表示 $x(t)$ 发生某种污染之后，在第 d 个设备真正记录到的信号。这种污染可能是如下四种常见类型之一，即

$$F_d(\cdot) \in \{G_1(\cdot), G_2(\cdot), G_3(\cdot), G_4(\cdot)\}, \quad 1 \leq d \leq D, \tag{4}$$

其中

- **加性噪声干扰**

  $$G_1(x)(t) = x(t) + n(t), \tag{5}$$

  其中 $n(t)$ 是高斯白噪声。

- **其他语音干扰**

  $$G_2(x)(t) = x(t) + v(t), \tag{6}$$

  其中 $v(t)$ 是一个独立于 $x(t)$ 的语音。

- **混响和回声**

  $$G_3(x)(t) = x(t) * h(t), \tag{7}$$

  其中 $h(t)$ 表示混响或回声系统的冲激响应，$h(t)$ 是因果系统。

- **限幅（削波）**

  $$G_4(x)(t) = \max(\min(x(t), A_U), A_L), \tag{8}$$

  其中 A_L 和 A_U 分别表示限幅的下界和上界。

综上，实际上发生的污染可以表示为

$$F_d(\cdot) \in \{G_1(\cdot), G_2(\cdot), G_3(\cdot), G_4(\cdot)\} \cup \{H_i(\cdot)\}_{i\in\mathcal{I}}, \quad 1 \le d \le D. \tag{9}$$

考虑到这些非理想因素，我们的目标调整为：设计一个最佳的算法 $R^*(\cdot)$，它可以由全部 D 个设备记录的信号 $\{y_d(t)\}_{1\le d\le D}$ 生成一个原始信号的估计 $\hat{x}(t)$，且力求估计准确，即

$$R^*(\cdot) = \arg\min_{R(\cdot)} \varepsilon(\hat{x}(t), x(t)), \tag{10}$$

其中

$$\varepsilon(\hat{x}(t), x(t)) = \int_0^T |\hat{x}(t) - x(t)|^2 \mathrm{d}t \tag{11}$$

$$\hat{x}(t) = R(\{y_d(t)\}_{1\le d\le D})(t), \quad 0 < t \le T. \tag{12}$$

总之，用浅显的话说：就是给你一堆受到各种污染的语音文件（碎片），设计算法使得不但能够正确拼接以还原出原始信号，而且能够使污染的影响尽可能小，处理后的信号尽可能接近真值，方差尽可能小。

二、 解决思路：

本次题目要求是相当高的。要求在正确拼接的同时还要尽可能的消除污染。我的整体思路是：首先，对于所列出的四种污染类型，分别采用如下解决方案：

① 加性噪声干扰：

注意到干扰信号是高斯白噪声。因此我们要充分利用高斯白噪声的特性。一开始我尝试使用较为常用的中值滤波的方法滤除白噪声，但是效果并不理想。中值滤波会使信号变得更加平滑，因而背景白噪声干扰将下降。但同时注意到：更加平滑意味着损失信息，白噪声下降得越多，语音也就越不清晰。后来，我注意到了高斯白噪声的特性：其功率谱在频带内为常数，频带外为零。自然想到如果能够得到一段空白引导段（无话帧），从而得到噪声的平均功率。由于高斯白噪声的性质，从而可以推到整个语音部分。因而引出常用的一个语音处理

方法：谱减法。

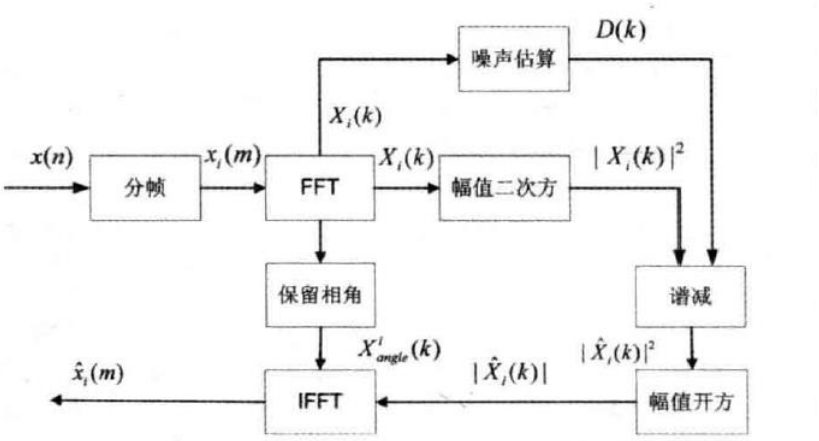


图 7－2－1　基本谱减法原理图

② 混响与回声：

我解决这一块污染充分体现了学以致用的思想，完全利用了课上所学知识：考虑设计逆系统并利用卷积来解决。

- 对第一次的假设 $h_1(t)$ 进行修正，得到新的逆系统 $h_2(t)$

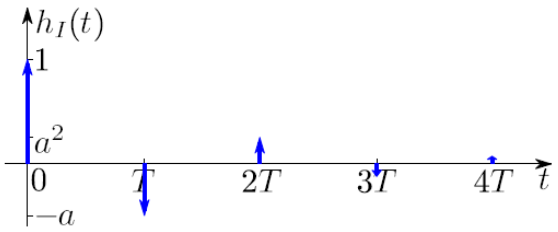
$$h_2(t) = \delta(t) - a\delta(t-T) + a^2\delta(t-2T)$$

- 级联结果为

$$h(t) * h_2(t) = \delta(t) + a^3\delta(t-3T)$$

- 依此类推，可导出 $h_I(t)$ 的最终结果

$$h_I(t) = \sum_{k=0}^{\infty}(-a)^k\delta(t-kT)$$



对于这类噪声，我想到这样的加噪信号做自相关一定会出现除了一个主峰（延时为 0）之外，还会有一个次大值峰。而这个峰的形成正是由于混响与回声的存在。同时，我们通过自相关的波形图还可以得到一重要的量：回声与主声之间的延时 delay。通过 delay 可以设计出符合要求的逆系统。

实现代码：

```
    for i=[n:-1:0]
        dx2(i*delay+1)=(-a)^(n-i);
    end
    y3=xcorr(dx2,signal);
    y3=y3(end:-1:end-L+1);
    final=y3;
end
```

③ 其他语音干扰：

对于其他语音和背景音乐的干扰，我发现可以通过①中的谱切法中适当的参数（a,b）的选取，可以削弱（甚至消除）大部分其他语音的干扰。但是，同时注意到，这种消除不是没有代价的。当其他语音干扰消除的越彻底，我们所需要记录的语音所受到的影响越大，会逐渐发生畸变。因此，我在这一部分的调参平衡上花了不少的时间。最终平衡两方面影响的结果还不错。

④ 限幅（削波）：

这一块内容受到了老师所讲的内容的启发。老师提到了信号拼接过程中的挑选问题：对某些时间段，必然有两个信号包含相同的语音，那么对应同一个时间段，应该选择信号质量最好的语音。那么，何为信号质量最好？我在代码中使用了一个量化判据：信噪比。通过比较信噪比，我们就可以较为科学的比较信号质量的好坏。从而优先挑选好的质量的信号来填充。这样，就尽可能避免选择到限幅（削波）的信号来填充到生成语音中。

计算信噪比部分：

```
    s{i}=(conj(fft(new1{i})).*fft(new1{i}));
    ss{i}=s{i}(1:floor(L(i)/16000*200));
    snr(i)=mean(s{i})/mean(ss{i});
    snr1(i)=mean(s{i})/mean(ss{i});  %计算信号的信噪比衡量信号质量
end
```

按信噪比填充部分：

```
while (any(snr))
    [maxs,ii]=max((snr));
    for jj=1:length(new1{ii})
        if (output(index(ii,1)+jj-1)==0)
            output(jj+index(ii,1)-1)=new1{ii}(jj);
        end
    end
    snr(ii)=0;
end
```

三、 遇到的问题：

在上面说了那么多，仿佛问题已经解决了。但是在实际操作过程中，却遇到了很多的问题，以至于最近两个星期我一直都在不停地Debug。具体来说，主要问题有：

① 高斯白噪声滤除过程中，我尝试了多种算法，但是遗憾的是没找到一种算法可以彻底地、完全地去掉高斯白噪声。即便使用谱减法，也会在去除白噪声的同时引入新的噪声。我在谱减法之后又让信号通过一个带通滤波器，但是仍不能完全消除。无奈之下，引用了一份开源的谱减法的代码（唯一的引用），以提升效果。引用代码部分我放在文末。

② 在语音拼接过程中，开始的数目比较少，尚可以使用手动拼接。但是后面数目越来越多，必须设计一个可以自动完成拼接的算法。其实找到拼接点并不难，我的思路是两两语音做互相关，然后找到尖峰位置的横坐标，从而得到相对偏移量。但是，对于如此多的信号，必须找到所有语音之间的先后时序关系，外加上重叠部分，一时难住了我。后来我终于想到了用伴随数组的形式记录与第i段语音相邻的段落，每一次只计算与相邻的上一段的相对偏移，其中伴随数组随着循环进行而不断生长，最终覆盖所有数据。相对偏移相加得到绝对偏移。最终，对绝对偏移来一次从小到大的排序即可得到首尾以及中间的时序关系。

相关代码：

```
adj=1;
index=zeros(50,2);
index(1,1)=1;
index(1,2)=length(new1{1});
for i=1:50
    line=adj(i);
    for j=1:50
        if(A(line,j)~=0)
            if(isempty(find(adj==j)))
                adj=[adj,j];
            end
            index(j,1)=A(line,j)+index(line,1);
            index(j,2)=index(j,1)+length(new1{j})-1;
        end
    end
end
index=index-min(min(index))+1;
result=sort(index);
```

③ 在利用xcorr函数时，判定相关与否主要看是否存在尖峰。但是尖峰的判定却又着实是一个大坑。我在这一块卡住了整整两天。后来发现，尖峰的阈值的设定实在是很讲究。如果阈值太大，则会因为尖峰峰值不到阈值而漏判相关。但是如果阈值过小，则会多判一些原本不相关的信号。这里一旦出错，则后面的拼接过程就会完全错误。最后，所有的阈值都经过我的精心设计，满足要求，例如：

```
yuzhi=100*mean(abs(rm));
if (maxium>yuzhi)
    A(i,j)=t(zz);
end
end
```

④ 在做自相关找消除多径失真的参数a时，开始图方便省事，就直接画图，用“人工智能”（人直接看）的方法估一个a了事。后来发现如果不同片段a稍有不同的话，就会导致结果比之前的更差。没办法，后来我设计了一个类似于“自适应”的办法，无需手动输入a值，算法会根据信号自动先算出一个a值，然后再继续后面的操作。具体可以通过二分法实现：

```
x1=0.1;x2=0.9;
m1=1;m2=-1;
e=abs(x2-x1);
while(e>0.00001)
    x=(x1+x2)/2;
    e=abs(x2-x1);
    for i=[n:-1:0]
        dx2(i*delay+1)=(-x)^(n-i);
    end
    y3=xcorr(dx2,signal);
    y3=y3(end:-1:end-L+1);
    temp=xcorr(y3,y3);
    [M,vv]=max(temp);
    m=temp(vv-delay);
    if(m*m1<0)
        x2=x;
        m2=m;
    else
        x1=x;
        m1=m;
    end
end
final=y3;
a=x;
```

四、 实验总结：

这次实验真的让我收获特别大。这次大作业，题目十分新颖，但是却不是遥不可及的。利用《信号与系统》课上所学习到的知识可以解决。真可以说是学以致用的典范题目。编写期间也遇到了很多问题，也曾经通宵熬夜不睡觉过。做的时候当然是觉得很辛苦的，但是完成之后的自豪感和满足感以及这满满的收获让我觉得我所付出的一切都是值得的。

谢谢老师，谢谢助教，让我在《信号与系统》这门课上收获了这么多。

五、 参考及引用：

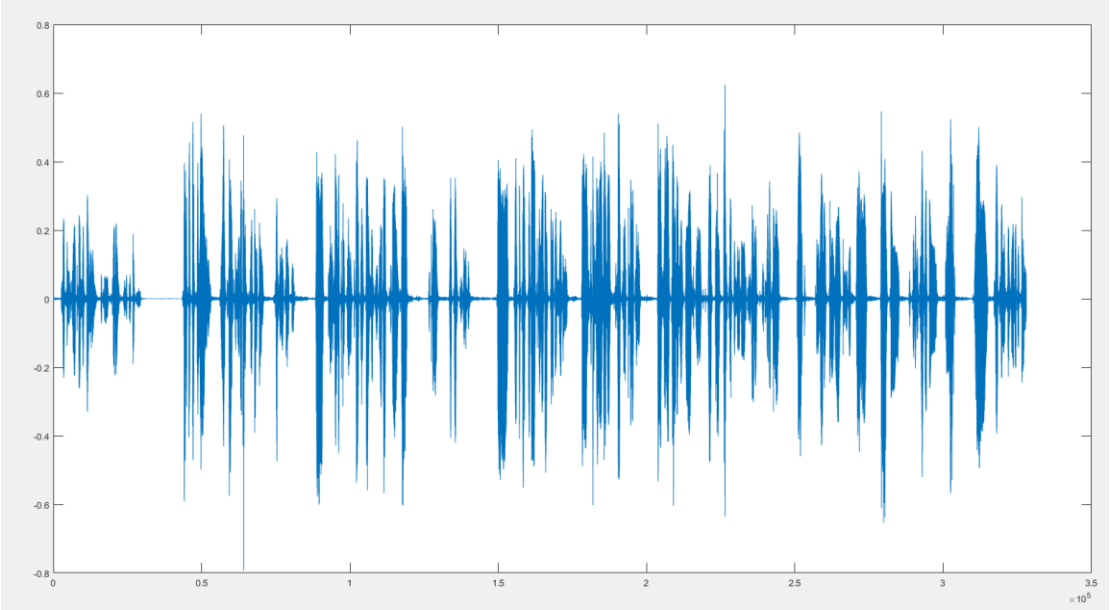
文件名：myfilter.m

内容：

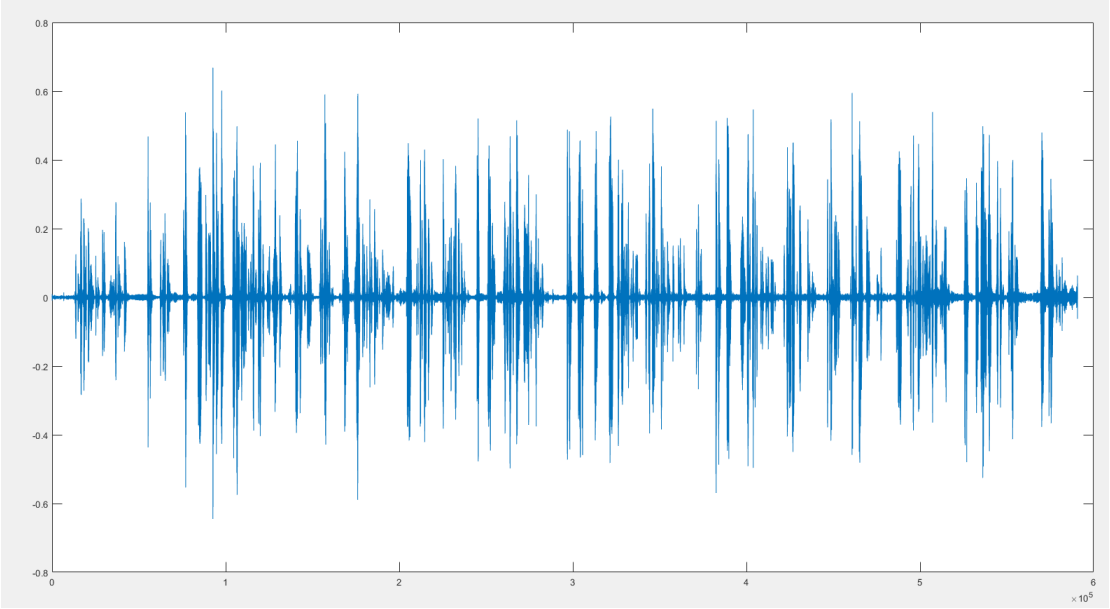
```
function improved=myfilter(winsize,signal,a,b)
%winsize=窗长
%a,b=修正系数
input=audioread(signal);%读入wav文件
size=length(input);%语音长度
numofwin=floor(size/winsize);%窗数
%定义汉明窗
ham=hamming(winsize);
hamwin=zeros(1,size);
improved=zeros(1,size);
ytemp=audioread('P1bSeg-2.wav');
noisy=ytemp(33001:33000+winsize);
N=fft(noisy);
npow=abs(N);
Ps=zeros(winsize,1);
for q=1:2*numofwin-1
    yframe=input(1+(q-1)*winsize/2:winsize+(q-1)*winsize/2);%分帧
    hamwin(1+(q-1)*winsize/2:winsize+(q-1)*winsize/2)=hamwin(1+(q-1)*winsize/2:winsize+(q-1)*winsize/2)+ham';
    y1=fft(yframe.*ham);%加噪信号FFT
    ypow=abs(y1);%加噪信号幅度
    yangle=angle(y1);%相位
    %计算功率谱密度
    Py=ypow.^2;
    Pn=npow.^2;
    %谱减
    for i=1:winsize
            if Py(i)-a*Pn(i)>0
                Ps(i)=Py(i)-a*Pn(i);
            else
                Ps(i)=b*Pn(i);
            end
        end
        %重构语音
        s=sqrt(Ps).*exp(1i*yangle);
        %去噪语音IFFT
        improved(1+(q-1)*winsize/2:winsize+(q-1)*winsize/2)=improved(1+(q-1)*winsize/2:winsize+(q-1)*winsize/2)+real(ifft(s))';
    end
    for i=1:size  %去除汉明窗所带来的增益
        if hamwin(i)==0
            improved(i)=0;
        else
            improved(i)=improved(i)/hamwin(i);
        end
    end
```

六、 处理结果：

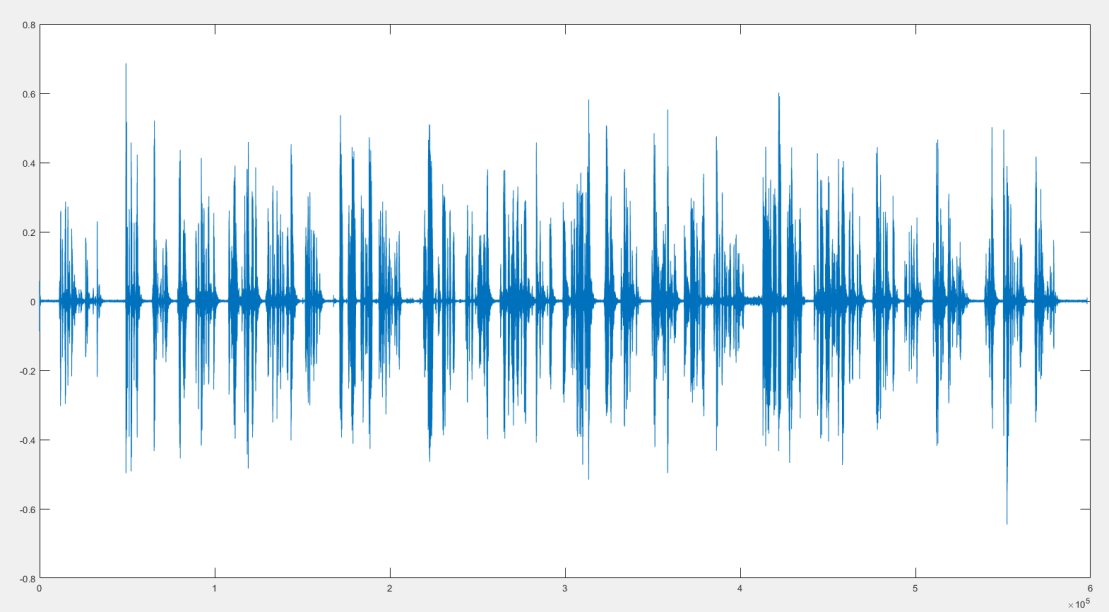
A 题


0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
0.5
1
1.5
2
2.5
3
3.5
×10^5

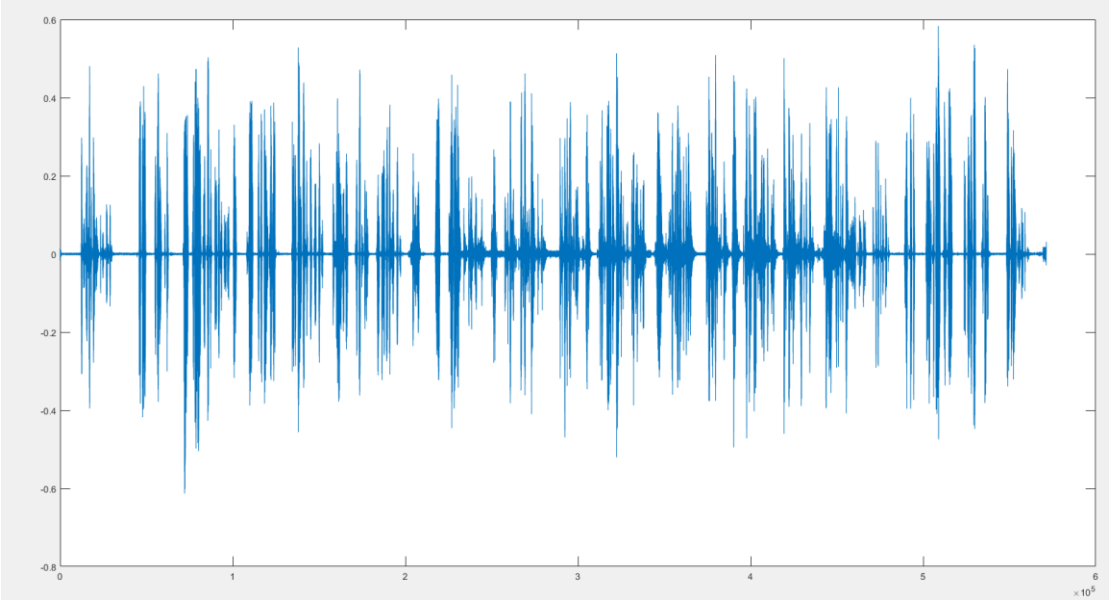
B 题：


0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1
2
3
4
5
6
×10^5

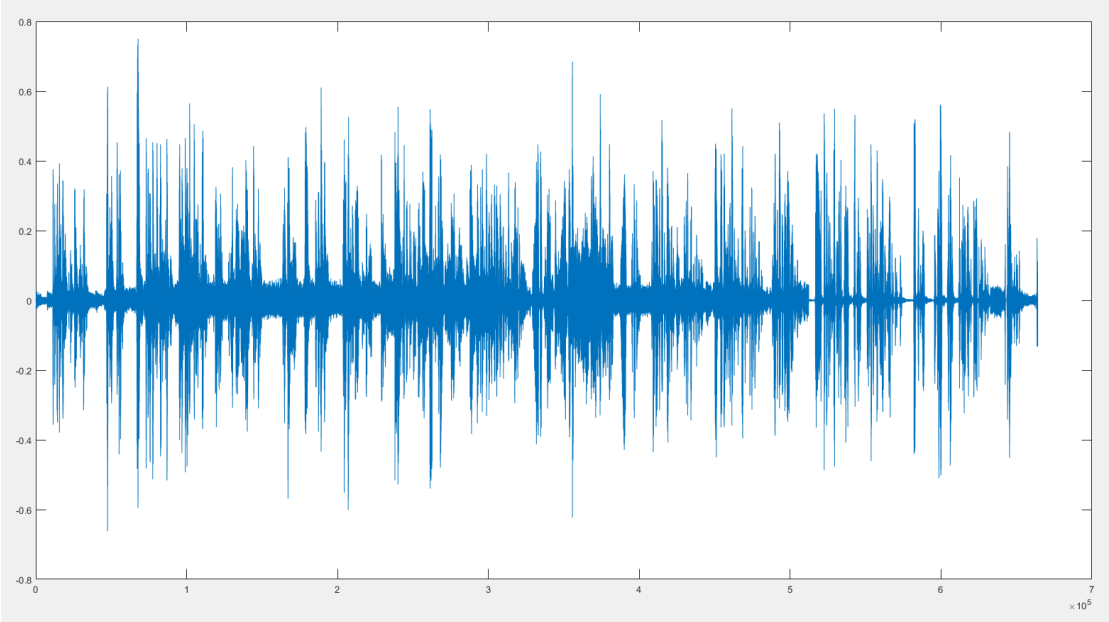
C 题：


0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1
2
3
4
5
6
×10^5

D 题：


0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1
2
3
4
5
6
×10^5

E 题：



可以看到，对于被污染得不是特别严重的信号，都能够较好的完成任务。但对于像 E 这种质量较差的信号，恢复就遇到了困难。此处无奈只得放弃对背景音乐做处理，否则语音部分将受到严重影响。

所以，我还需努力、学习的地方还有很多，以后继续加油！