信号与系统大作业《语音碎片我来拼拼听》
题目1_c 实验报告

1.实现思路

解决本问题要解决三个关键问题。

1)如何消除回声问题。考虑最简单的回声：y(t) = x(t) + Ax(t-T)，利用本学期学过的拓展知识DFT的时移性质，在频域对它进行处理再反变换到时域即可解决此问题，且效果应该很好。

2)如何解决限幅问题。查阅文献发现此问题没有很好的解决办法，基本思路也是用小波或者自适应滤波解决，但是恢复效果不理想，解决此问题我认为需要主要依靠筛选算法，即是尽可能减少拼凑限幅信号来解决。

3)对这四种(或三种)信号的排序和拼接。我打算采取优先级算法来实现，即是增加一个tag，对于回声信号记为2，限幅信号记为0，其余为1，按照优先级进行拼接。

2.实现方式

2.1回声信号去除

```
function eco = echo_audio(y,w)
%model: y(t)=x(t)+Ax(t-delay);
%利用DFT的性质解决此问题
N = length(y);
z = xcorr(y);
m1 = max(z);
m = find(z == max(z));
width = w;
z(m-width:m+width-1)= 0 ;
m2 = max(z);
pos = find(z == max(z));
delay = abs(m-pos(1));  %回响延时
A = m2/m1;  %回响峰值
W = exp(-2i*pi/N);
tmp = 1:N;
tmp = A*W.^(-tmp*delay)+1;
y1 = fft(y);
z1 = y1./tmp';
eco = real(ifft(z1));
```

确定A和T的方法是找自相关函数的局部最大值。观察回响信号的自相关函数可以明显看到有三个峰值，考虑一般语音信号自相关函数除去最大值部分后其他最大值应该远小于峰值。依旧此原理可以检测出延时，考虑自相关函数的定义可以发现A的值就是自相关函数局部最大值和最大值的比。之后用离散傅立叶变换的性质来解决。

离散傅立叶变换原理参考：https://wenku.baidu.com/view/f89fb321dd36a32d73758125.html 离散傅立叶变化的性质。

2.2其余信号去噪

限幅信号：先过低通再过小波。

加性噪声：同p1b，用语音增强算法。

2.3判断标签

见文件detect_tag.m

思路是首先初始化tag为全部1，先判断回响，方法同去除回响，如果有tag为2。之后判断有无限幅，方法为比较max值集合长度与语音信号长度的关系。超过阈值tag为0。

```
function tag = detect_tag(y,w)
z = xcorr(y);
m1 = max(z);
m = find(z == max(z));
width = w;
tmp = z;
tmp(m-width:m+width-1)= 0 ;
m2 = max(tmp);
len1 = length(y);
m = find(y == max(y));
len2 = length(m);
if(m2/m1>0.15)
   tag = 2;
elseif(len2/len1 > 0.1)
    tag = 0;
else
    tag = 1;
end
```

2.4 片段拼接

见文件select_c.m。实现方法与其他无异，首先选择一个回响信号加入正确集，之后判断衔接时增加了优先级标志”first”由于每部都是最优操作，所以默认正确集合优先级为1.8(略小于2)，只有当遇到回响信号时才用回响信号覆盖原信号，其余是用out_file(正确集)信号覆盖衔接信号。

实际上，正确集优先集应该为1，对于first标记为1的信号可以考虑取均值，代码也有设计如下，但是长度似乎取值会出错，输出信号有交叠，时间不够debug，最终舍弃。

```
  %elseif(first == 1)
      %slice_1 = tmp(tmp_l-size+1:tmp_l);
      %slice_2 = out_file(1:size);
      %mix = (slice_1+slice_2)/2;
      %out_file = [tmp(1:tmp_l-size);mix;out_file(size+1:len)];
  else
      tmp = mean(abs(out_file))/mean(abs(tmp))*tmp;
      out_file = [tmp(1:tmp_l-size);out_file];
  end
else
   size = tmp_l-max_pos+rate;
   if(first>1)
       out_file = [out_file(1:len-size);tmp];
   %elseif(first==1)
       %slice_1 = out_file(len-size+1:len);
       %slice_2 = tmp(1:size);
       %mix = (slice_1+slice_2)/2;
       %out_file = [out_file(1:len-size);mix;tmp(size+1:tmp_l)];
   else
```

3.主函数

主程序：ex3.m

先预处理判断标签，再去噪，最后排序。

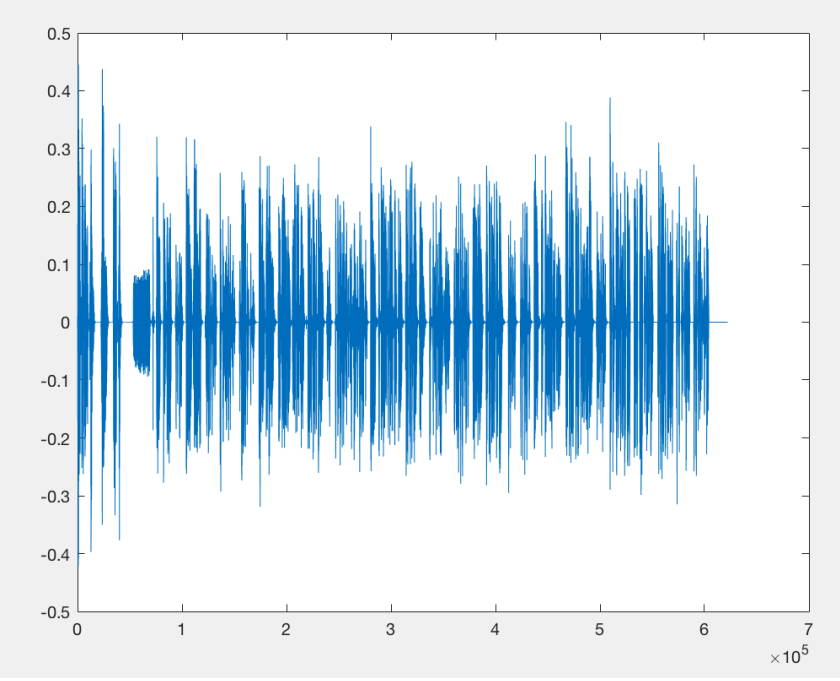
```
%预处理
for i=1:len
    na = file_list(i).name;
    y = audioread(strcat(file_path,na));
    width = 100;
    file_list(i).tag = detect_tag(y,width);
    file_list(i).use = 0;
end

%去噪处理
for i = 1:len
   tp = file_list(i).name;
   [y,fs] = audioread(strcat(file_path,tp));
   tagg = file_list(i).tag;
   if(tagg == 2)
      xxd = echo_audio(y,100);
   elseif(tagg ==1)
       xxd = enhance(y);
   else
       % not sure which fuction to use.
       t_m = ll_filter(y,2200,2500);
       xxd = wavelettt(t_m,0.6);
   end
    audiowrite(strcat(out_path,tp),xxd,8000);
end
%排序

rate = 40000;
out = select_c(out_path,file_list,rate);
audiowrite('P1c.wav',out,8000);
```

4.实验结果及分析

P1c.wav文件为实验结果，下图为处理后画出的波形。



实验效果与p1b差不多。由于回声信号处理很方便其实应该略好于前一问。改进措施就是前文提到的对于优先级的重新估计，对于加性噪声在重叠部分取平均值会有助于去噪。但是时间关系没有实现。