数字信号处理第二次课程设计

课程设计报告

2019011008

无 92 刘雪枫

一、实验平台

实验采用 Windows 上 MATLAB 2021a (64-bit)进行实验。

二、实验过程

（一） DTMF 信号的产生

根据中文教材 5.10.1 节课，如果令：

$$\begin{cases} y_c[n] = (\cos\omega_0)y_c[n-1] - (\sin\omega_0)y_s[n-1] \\ y_s[n] = (\sin\omega_0)y_c[n-1] + (\cos\omega_0)y_s[n-1] \end{cases}$$

并取初值：$y_c[-1] = \cos\omega_0, y_s[-1] = -\sin\omega_0$，则得到的$y_c$和$y_s$分别为：

$$\begin{cases} y_c[n] = \cos\omega_0 n \\ y_s[n] = \sin\omega_0 n \end{cases}$$

由此，根据此公式迭代即可得到离散的单频正弦波，取y_c即可。据此写出下面的 MATLAB 代码：

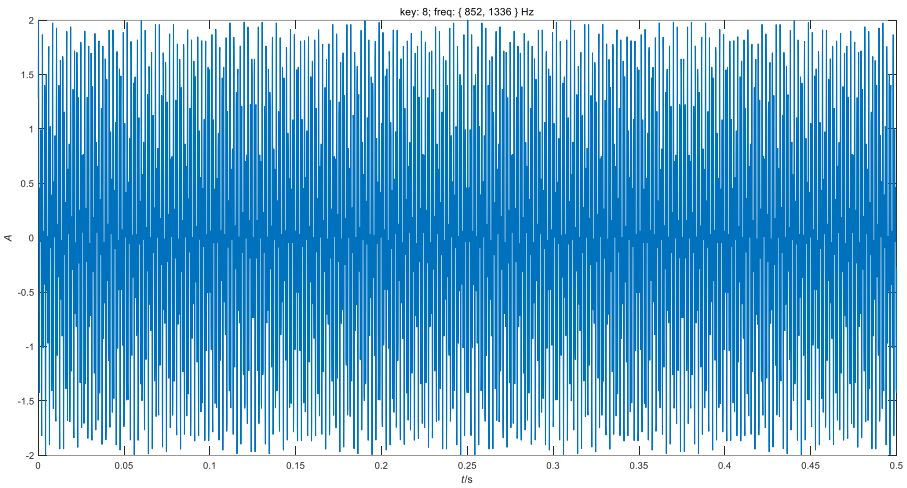
```
cw = cos(omega);
sw = sin(omega);
yc(1) = cw;
ys(1) = -sw;
for i = 2 : 1 : n_points
    yc(i) = cw * yc(i - 1) - sw * ys(i - 1);
    ys(i) = sw * yc(i - 1) + cw * ys(i - 1);
end
```

```
y = yc;
```

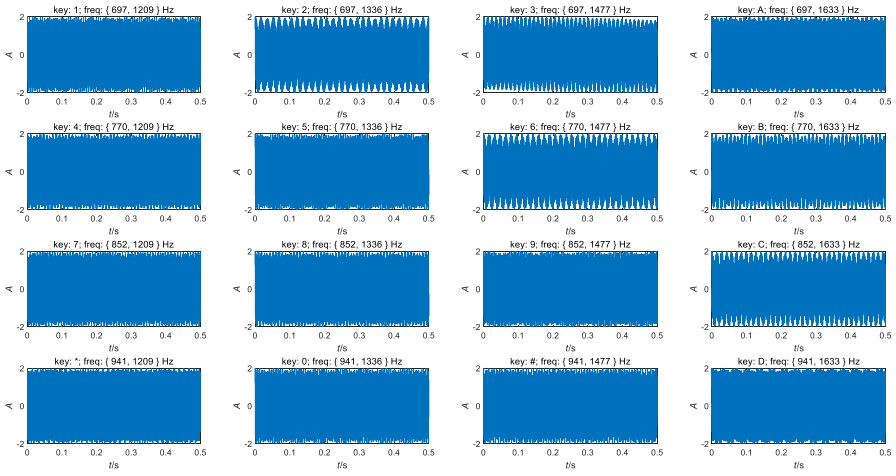
其中 n_points 为要产生的离散正弦信号的点数，得到的 y 记为得到的离散正弦信号。根据模拟频率与数字频率的关系，由于采样率为 8000Hz，因此将模拟角频率除以采样率即可得到数字角频率。将该段代码封装为函数 get_sin，储存在函数文件 get_sin.m 中。

再根据各个按键产生的声音的两个模拟频率产生正弦信号并相加即可。

将启动脚本命名为 hw_3_1_1.m，运行脚本，可以看到输入按键时可以获得相应的波形，下图为按键 8 时的波形：



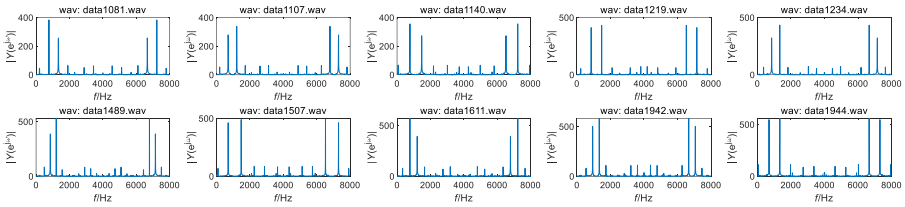
下图为所有按键产生的波形：



（二） DTMF 信号的检测和识别

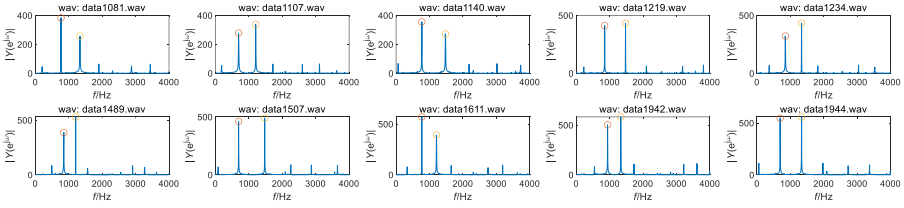
I) 使用 MATLAB 自带函数分析音频

对给定的十个按键音进行频谱分析。使用 `audioread` 函数读入音频得到离散的声音序列。使用 MATLAB 自带的 `fft` 函数对声音序列做 DFT 变换，并绘制频谱图。得到是个音频文件的频谱如下：



由于实信号的幅频响应是对称的，因此只考察其半边频谱。分别在行、列频率的频点周围正负 10Hz 范围内取最大值作为该频点的幅度值，然后分别在行、列频点中取幅度值最大的频点，作为该按键的行、列频率。

在频谱图的左半边绘制除提取到的最大频点如下图：



可以看到最大频点均被提取出来。根据频点将识别到的按键输出如下：

```
命令行窗口
The key of file: data1081.wav is: 5
The key of file: data1107.wav is: 1
The key of file: data1140.wav is: 6
The key of file: data1219.wav is: 9
The key of file: data1234.wav is: 8
The key of file: data1489.wav is: 7
The key of file: data1507.wav is: 3
The key of file: data1611.wav is: 4
The key of file: data1942.wav is: 0
The key of file: data1944.wav is: 2
fx >>
```

由此可以看到，十个音频文件的按键依次为（以音频文件名称中的数字从小到大排序）：5、1、6、9、8、7、3、4、0、2。

本题的启动脚本为 hw_3_2_1.m。

II) 使用 Goertzel 算法分析音频

使用 Goertzel 算法计算按键音的频点上音频序列的幅值。对于频点f_0，其数字角频率为 $\widetilde{\omega_0} = 2\pi\frac{f_0}{f_s}$（$f_s$为采样率），则选取$k = \text{round}\left(\frac{N\omega_0}{2\pi}\right) = \text{round}\left(\frac{Nf_0}{f_s}\right)$，则计算所用数字角频率为$\omega_0 = \frac{2\pi k}{N}$。考察下面的差分方程：

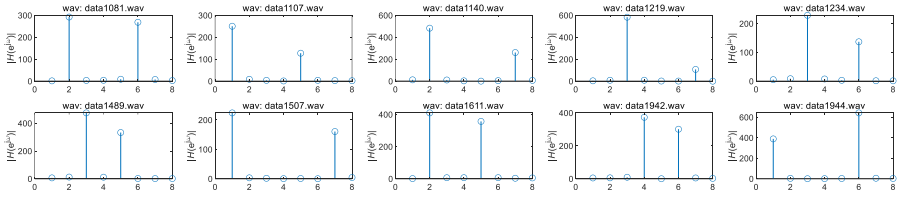
$$v_k[n] = 2(\cos\omega_k)v_k[n-1] - v_k[n-2] + x[n]$$

$$y_k[n] = v_k[n] - W_N^k v_k[n-1]$$

计算$X[k] = y_k[N]$即可得到频点上的幅值。由于要计算$y_k[N]$，而$x[n]$的长度为$(N-1)$，因此需要将$x[n]$后面补一个零进行延拓。因此使用 MATLAB 代码计算$v[n]$的代码即为：

```
b = [1, -2 * cos(omega), 1];
a = [1];
v = filter(a, b, [x; 0]);
```

据此，依次对十个音频文件进行频谱分析，并绘制出图像得到：



其中，从左到右依次为 697、770、852、941、1209、1336、1477、1633（单位：Hz）频点的幅度值。

对每个音频文件，分别取前四个和后四个频点中的最大值，即可得到按键的行频率和列频率，并判断按键，结果如下：

```
命令行窗口
  The key of file: data1081.wav is: 5
  The key of file: data1107.wav is: 1
  The key of file: data1140.wav is: 6
  The key of file: data1219.wav is: 9
  The key of file: data1234.wav is: 8
  The key of file: data1489.wav is: 7
  The key of file: data1507.wav is: 3
  The key of file: data1611.wav is: 4
  The key of file: data1942.wav is: 0
  The key of file: data1944.wav is: 2
fx >>
```

可以看到，和使用 FFT 得到的结果一致。

本题的启动脚本为 hw_3_2_2.m。

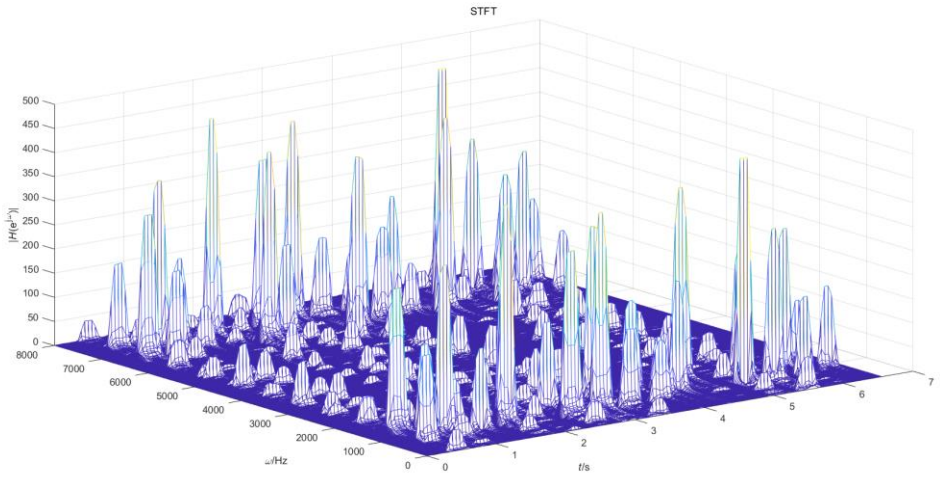
III) 识别长音频信号

对于给定的的长音频信号，需要计算其每个时间点附近的频谱。为此，对给定的长音频信号做短时傅里叶变换，计算每个时刻附近的频谱情况。

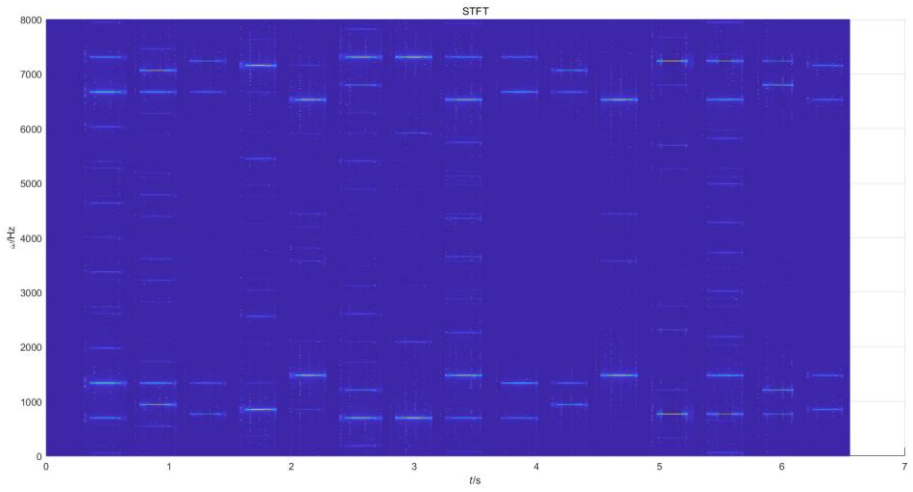
考虑到电话按键时，每次按键的时间一般都比 0.1s 大很多，因此取短时傅里叶变换的长度为 0.1s，由于采样率为 8000，因此变换长度为 800 个点。注意到音频文件的长度为 53082 个点，若以步长为 1 做短时傅里叶变换则计算量很大，因此取步长为 400 个点，减少计算次数。步长为 400 个点时，对应连续时间为 0.05 秒，根据生活中打电话的按键经验应该足够小。

a) 使用 FFT 进行短时傅里叶变换

使用 MATLAB 内置的函数 FFT 进行短时傅里叶变换，得到的三维频谱图像如下：



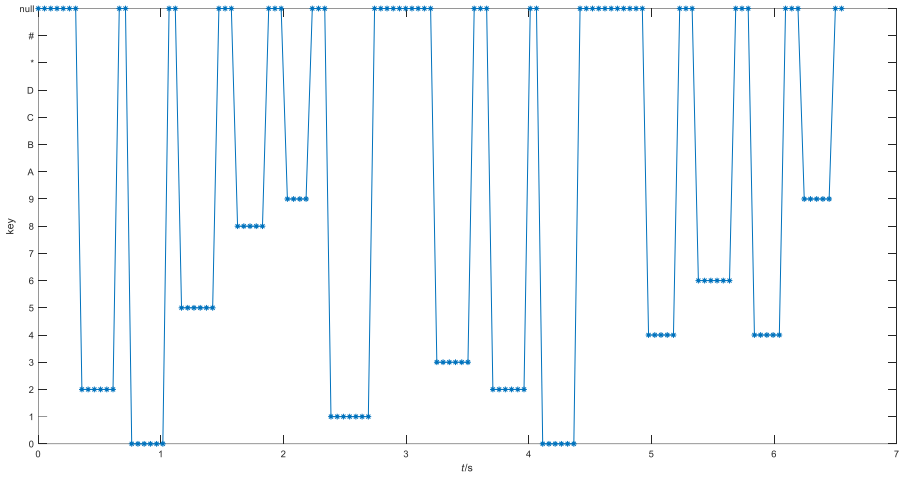
三维图像不够直观，查看其俯视图：



其中较亮的点为频谱较强的点，可以看到短时傅里叶变换成功分析出了各个时间点附近的频谱。

从频谱图中还可以看出，有一些点是在按键的间隙当中，即不代表任何按键，因此在判断按键时还需要考虑是否真的是按键音。为此，在根据各个按键频点的幅值判断按键音频率时，可以加上约束：所有频点的幅度最大值与所有频点的幅度最小值之差需要超过最大值的 80%，并且得到的幅度值必须不小于 50；同时由于变换点数的减少，将每个频点周围正负 20Hz 的范围内幅度的最大值作为该频点的幅度以提高准确度。

最后绘制出识别出的按键与时间的关系如下图：

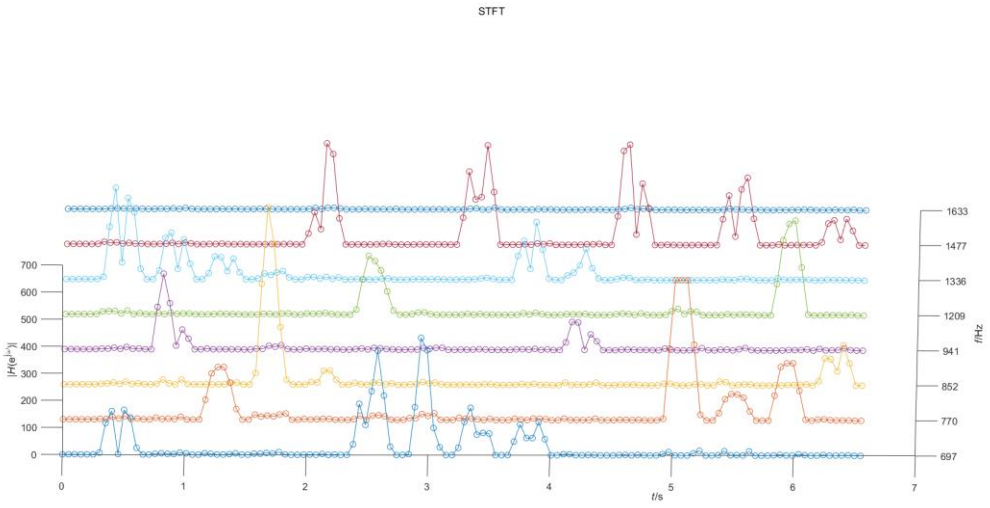


根据上图可以看到，识别结果为依次按键 2、0、5、8、9、1、？、3、2、0、？、4、6、4、9。其中，问号处可以明显看到有两处按键并没有被识别。观察之前得到的短时傅里叶变换得到的频谱的 4.5~4.8 秒之间的部分，可以看到该处按键音的频谱只有一个明显的峰，即在 1477Hz 处左右，即，FFT 只分析出了列频率，并没有分析出行频率；而在 2.8~3.1 秒之间的部分，按键音只在 697Hz 处左右有一个列频率的峰，没有分析出行频率。推测原因可能是在按键是行频率的声音没有发出，或是在录制时出现了失真等等原因导致无法识别。

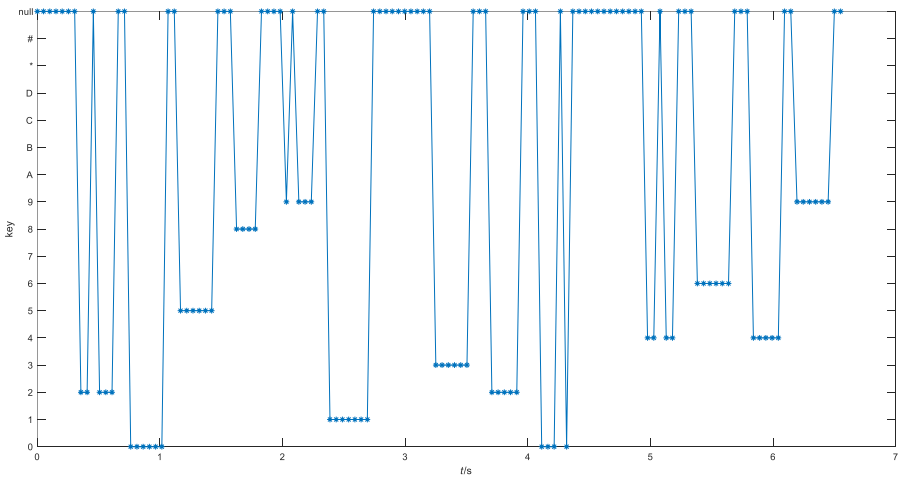
本题启动脚本为 hw_3_2_3_1.m。

b) 使用 Goertzel 算法计算

现在使用 Goertzel 算法算法分析音频。类似于之前使用 FFT 进行操作，得到各个时间点的各个按键音频点上的频谱如下：



然后对其每个时间点获取按键，和使用 FFT 算法获取的流程相同。得到如下的图像：



可以看到，得到的结果基本上与使用 FFT 的结果一致，即依次识别出了 2、0、5、8、9、1、？、3、2、0、？、4、6、4、9 几个按键。其中问号处没有识别出的原因在之前已经分析过，值得注意的是，在使用 Goertzel 算法分析的过程中，与使用 FFT 有一处不同。在第 1、5、10 个音处各存在一个时间点没有被识别出来。观察原来的频谱图可以发现，在该时间点，频谱的值几乎接近于零。仔细考察原因，可能是该算法在将一个系统函数拆解为两个滤波器时，引入了新的零点和极点，因此输入信号在迭代计算过程中发生了共振而有较大程度的精度损失，

因此会出现这种状况。

三、实验小结

经过不懈的努力，我比较完整地完成了本次课程设计。总地来说，过程还算是比较顺利的。其中遇到的一些困难就是我对 DFT 的运算可能还不是特别纯熟，在模拟频率与时间和数字频率与采样点之间的转换关系还是需要自己慢慢计算。通过这次实验，我对这种转换关系也有了进一步的熟悉。

最后，感谢老师的辛勤讲授和助教的辛苦付出，没有老师和助教的悉心指导我也很难学到知识去完成本次课程设计，在此表达由衷的感谢。