## 实验1、双音频按键识别

**任务：**

使用两部手机录制按键声音文件，导出到PC 机上。对录音中的按键音进行识别，检验算法的准确性。

1. 查阅文献或教材，找到Goertzel 算法，用Matlab 实现之，并测试其识别性能（精度、速度）。

2. 使用Matlab 的FFT 函数，将音频数据变换到频域，根据DTMF 的频率组成，检测按键。

**实现方法：**

1. Goertzel算法

使用该算法前需要把声音转化成8000采样率单声道。

这样我们可以把整个音频分成许多等长的部分，每个部分长度均为205，对每个部分，使用goertzel算法求出18、20、22、24、31、34、38、42的八点DFT，求出前四个和后四个中值最大的那个，其中前四个对应四个较小的频率，后四个对应四个较大的频率。

这样就可以得到两个频率，从而得到识别结果了。

1. FFT方法

使用该算法前需要把声音转化成8000采样率单声道。

整体做一次fft，并求出fft得到数列中每个值的模，从而得到频谱图。然后找到两个合适的峰值即可。合适的峰值需要间隔一定的距离，且一个在前四个频率附近一个在后四个频率附近，满足这个条件最大的两个峰值即是我们所需的。找到这两个峰值对应的两个频率，我们就可以输出识别结果了。

**实现结果：**

可以全部正确识别audio1和audio2文件夹中的所有音频文件，说明识别算法正确且识别率比较高。

测试时，执行test.m，会输出两个ans，可以与文件名进行比较从而判断结果是否是正确的。

## 实验二、卷积计算方法的性能比较

**任务：**

根据公式计算、FFT计算、Overlap-Save、Overlap-add 4种卷积计算方法的原理，使用*Matlab* 分别实现上述几种算法，并以不同长度的序列卷积来对比它们的计算效率。其中，*FFT* 函数直接调用*Matlab* 内部函数，不必重新实现。

**效率分析：**

1. 公式计算：

设两个序列的长度分别为N和M，则效率为O（NM），故在N和M都很大的时候计算会很慢。

1. FFT计算：

设两个序列中较长的序列长度为N，由于FFT的效率为O（NlogN），故效率应为O（NlogN）。显然比公式法稳定，但在某些情况（比如一个序列长度特别长一个序列长度特别短）的时候效果不如公式法。而且FFT需要进行大量的复数乘法加法的操作，所以常数很大。

1. Overlap-Save和Overlap-add算法：

这两种方法主要是优化了FFT的复数乘法进行的次数，都是采用分段的思路利用性质使复数乘法操作变成复数加法操作，所以使效率大大提高。

**测试结果：**

1. 公式计算：

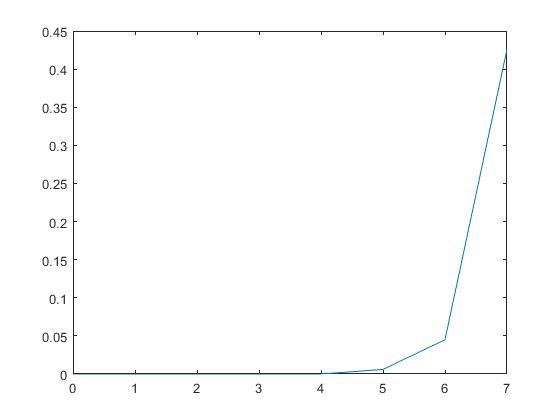
从上到下依次为：

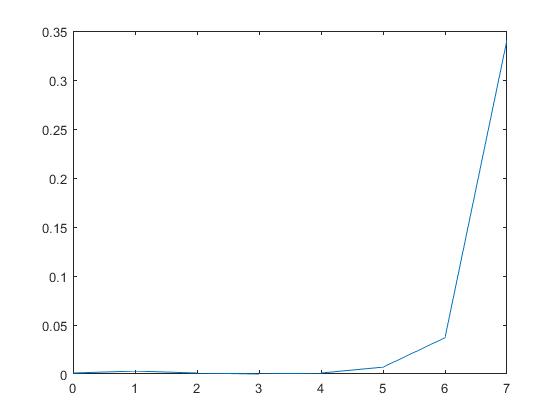
X的长度为L，h的长度为3

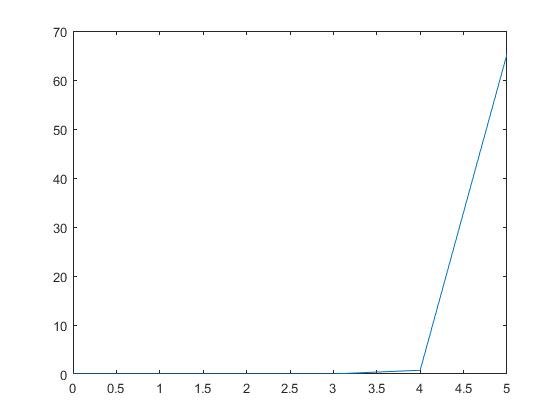
X的长度为3，h的长度为L

X的长度为L，h的长度为L

其中，横坐标为L大小，L=10^x；纵坐标为进行操作所需的时间。







观察后我们可以发现，公式法不适合计算两个序列均很长的情况，与效率分析结果一致。

1. FFT计算

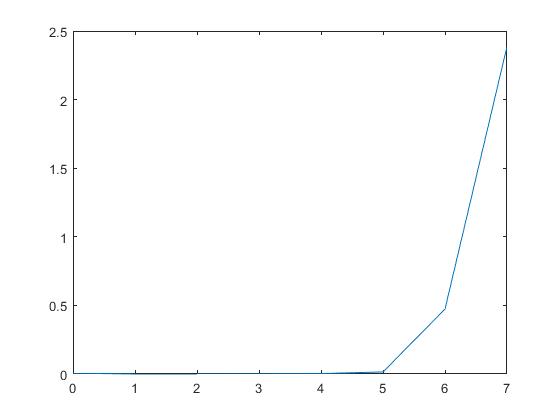
从上到下依次为：

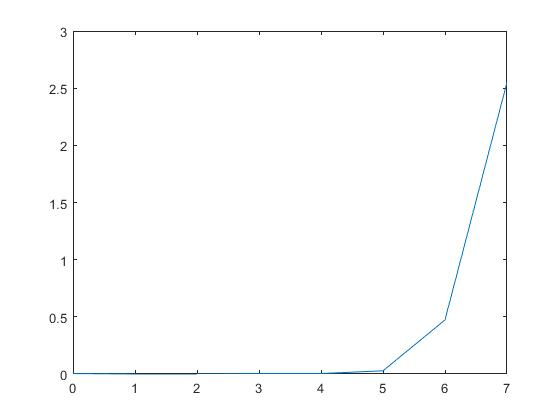
X的长度为L，h的长度为3

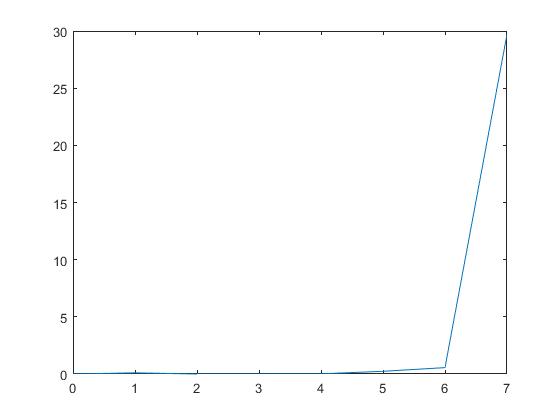
X的长度为3，h的长度为L

X的长度为L，h的长度为L

其中，横坐标为L大小，L=10^x；纵坐标为进行操作所需的时间。







通过观察我们可以发现，FFT优化的确大幅提高了速度，但是由于复数乘法次数过多，所以在两个序列长度均很长时时间仍然不能让人满意。

1. Overlap-Save算法：

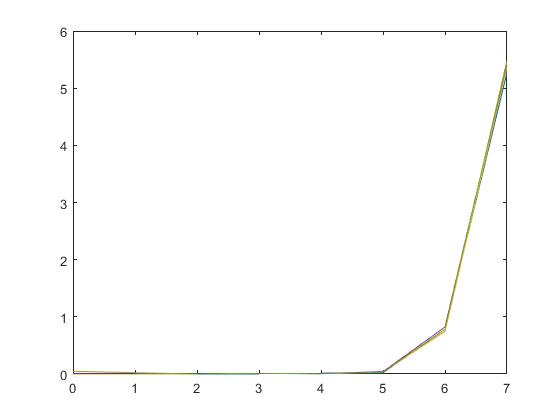
从上到下依次为：

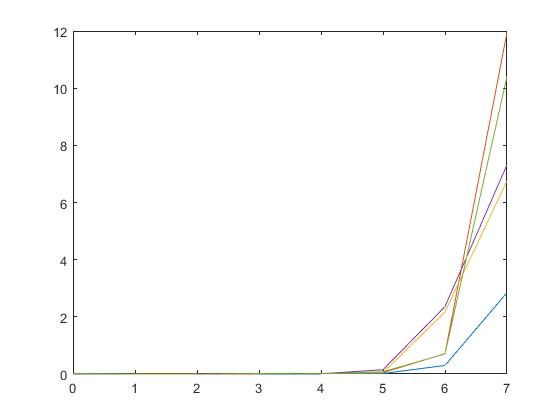
X的长度为L，h的长度为3

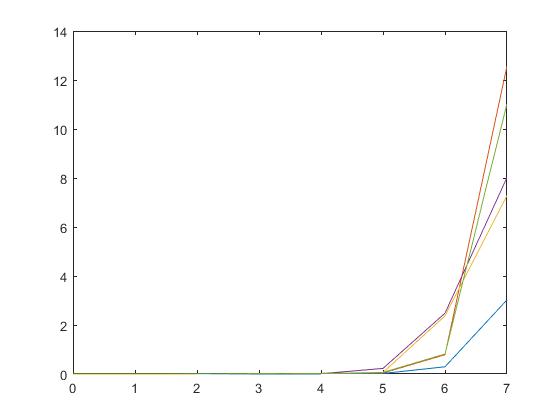
X的长度为3，h的长度为L

X的长度为L，h的长度为L

其中，横坐标为L大小，L=10^x；纵坐标为进行操作所需的时间，不同直线代表不同的划分长度。







通过观察我们可以发现在两个序列长度均很长时，改进明显。在两个序列长度均在10000000以内时，时间可以稳定在8s之内，效率很稳定。

1. Overlap-add算法：

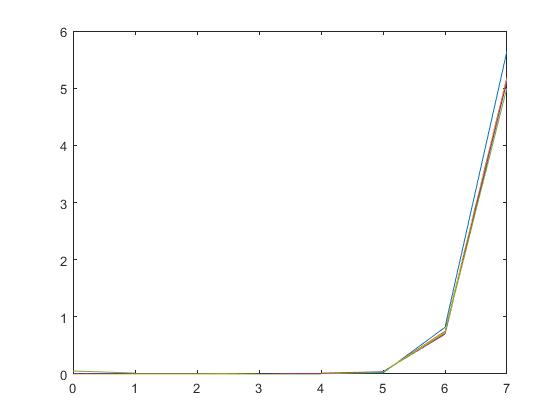
从上到下依次为：

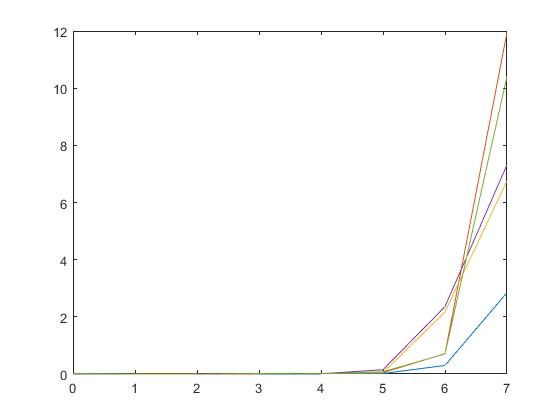
X的长度为L，h的长度为3

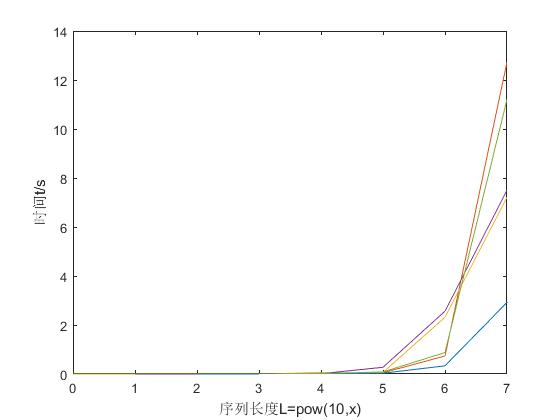
X的长度为3，h的长度为L

X的长度为L，h的长度为L

其中，横坐标为L大小，L=10^x；纵坐标为进行操作所需的时间，不同直线代表不同的划分长度。







与Overlap-save方法相似，均可以在两个序列长度在10000000以内时将时间压缩到8s以内，也是一种比较好的方法。

**实验结论：**

无论是公式法还是FFT优化的方法，对于两个长度都比较长的序列时都比较无力，而overlap-save和overlap-save方法可以很好地解决这个问题。

## 实验三、语音信号的频分复用

**任务：**

找内容不同的多段语音，要求时间长度相等，采样频率也相同。用*Matlab* 编程来验证通信系统中的“频分复用”原理。仅要求模拟和验证通信系统中的调制（相当于编码）和解调（相当于解码）的原理。

编码处理：对多个语音片段分别进行处理，然后将处理结果数据叠加起来，要求叠加后的数据长度和原始语音长度一致。

解码处理：对编码处理后的数据进行解码处理，要求能将叠加在一起的多路语音分离开，分离得到的多路语音相对于原始的语音，应保证失真尽可能小。

**实现方法：**

FFT得到序列的下标与频率有一定的关系，利用这个关系即可实现最基本的“滤波“操作和“移动”操作。这个关系为下标值=频率值\*总时间（单位为秒）。

因此，滤波时，我们需要找到对应频率范围的下标范围，将在这个下标范围之外的值设为0即可达到一个简单滤波的效果。

“移动”时，为了将a-bHz变为c-dHz，我们只需将a-bHz对应的下标范围的全部数值移动到c-dHz对应的下标范围即可。

为了达到较好效果，虽然语音一般在300-3400Hz之间，但如果只去这些失真会比较严重。故我们取0-5000Hz这段区间。当我们有三段语音时，我们将其移动到0-7000,7000-14000,14000-21000范围内，采样率为44100Hz，这样刚好满足采样定理，也可以防止频谱出现明显的混杂现象。

**实现结果：**

保真度较高，输入文件为sound1.wav、sound2.wav、sound3.wav，输出文件为output1.wav，output2.wav，output3.wav，编码后生成的文件为output.wav。运行test.m即可执行全部流程。