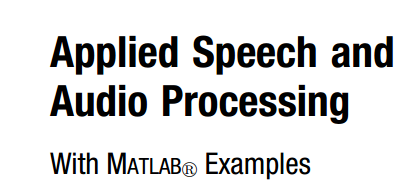
实用语音技术与声频处理：使用matlab r d的案例

《实用语音技术与声频处理》是一本基于matlab的混合语音和听力的研究的关于语音和声频处理的核心技术一站式资料书。

该面向实践的文本提供了matlab贯穿描述所讨论概念到给读者亲手实践亲历核心技术的案例。基础音频处理以及语音信号处理为基础的语言和听觉研究，是建立在之后的章节提到的音频信号处理、编码、压缩和分析技术之上的。最后一章我们将探索许多这些技术的高级应用，如心理声学建模——这是支撑mp3和相似音频格式的移门学科。

由于本书具有实践性，以及包含大量matlab案例，它将是毕业生和语音或声音系统的工程师的理想书籍。

Ian McLoughlin是新加坡南洋理工大学的计算机学院的副教授。过去的十二年，他在工业、政府和学术领域工作。她的出版物和专利包括了语音处理的清晰度，压缩，探测和口译，英文与中文普通话清晰度的听力模型，以及将声音信息隐藏的心理声学方法。







………………

前言

听与说是两个人类的紧密相连的能力。可以说，人类的语言被划到我们最能感知的频率范围，又或者说听力被优化到我们说话时所使用的频率范围。然而当我们讨论此问题的时候，工程师往往使用听与说同时在语音交流领域被考虑的语音传输和处理系统。然而，听和说在她们自己的领域已经很复杂，尤其是听力。

近些年，语音听力教科书中关于心理声学的讨论变得流行。心理声学是联合了心理学和声学的一个术语，它描述了认知层面处理声音的方式。它强调了人类接收声音的方式，并不是直接的物理的方式。有一段时间，在学术会议上使用这个词是一种对高级知识的自吹自擂，和对尖端术语的熟悉程度，尤其是这个词念出来非常顺口。

……………………

**1.介绍**

声音和语音处理系统在大部分发展中国家的人们的生活当中的地位是稳定上升的。在hifi音乐系统中，不管是无线电广播还是便携式音乐播放器，音频处理对于消费者的娱乐来说都是很重要的。数字音频技术现在在使用CD播放器，互联网电台，mp3播放器和i POD的音频传输方面占统治地位。在电视和电影工作室，以及现场演出的混音台，数字音频处理也是占优势地位的。音乐和音效在电脑游戏当中，也很重要。

随着蜂窝通信的发展（尤其是欧洲GSM标准），语音处理在世界范围内的发展是一个上升的趋势。如今，GSM几乎是无处不在的，甚至在世界上最贫穷的地区，也有广泛的采用度。

当然，数字语音信号使用的年限已经不短，比如卫星通讯连接，但如今，传统电话网络（named POTS for ‘Plain Old Telephone Services’）也在向数字化屈服。最后一公里使用几百米的铜双绞线接入消费者的家中，这样的数字技术工作方式是从来不会出现在工程师的脑海中的，光纤通信以太网和无线连接在多年来也是一直被抵制的。DSL (digital subscriber line – normally asymmetric so it is faster in one direction than the other, hence ADSL) ，即使是这种模拟的双绞线，也可以可观地高速传达数字信号。a d s l，对于互联网视频传输或者是像skype这样的数字语音传输都是足够快的。

数字音频

目前数字处理是解决音频和语言问题的选择方式：事实上，新的音频软件或系统都是占优势地位的。这种从模拟到数字的改变大多发生在过去十年，直到现在已经变成了一种默认的模式。

这样的趋势大部分发展到了语音音频和听力相关的研究中，许多相关的研究现在都是基于数字的。数字技术的一大好处是它几乎不依赖设备：一个数字处理平台上的技术，可以一直到另外一个上。一个标准在开发平台上会变的易用以及易测试，开发平台的标准可能完全不同，低功耗，小容量，或高功耗，低价格等。

从易用的层面开发，在PC上运行的matlab在这个领域的工作中广泛使用。它是专为处理数字信号而设计的，尤其是长串的音频采样。其内建的函数，允许大部分可能的常用的操作控制，也可以录音和回放，其可视化和绘图工具也很棒。本文作者常在m a c o sx和linux平台使用matlab完成它的音频工作。

尽管目前，matlab并没有提供语音音频或听觉工具箱，信号处理工具箱包含了大多需要的函数，Department of Electrical and Electronic Engineering, Imperial College, London的voicebox开源工具箱也提供了许多有用的函数。也可以使用开源的octave开发环境执行本书中的大部分音频和语音进程，但需要对本书中的某些案例作出小的变更。Octave比matlab小众一些，它缺乏高级的绘图和侦错功能，但是其她的方面很相像。

收集和转换声音

你说的所有内容都是关于声音的。不是声音被机械产生，就是声音被人或机械听到。在纯物理层面，声音是由于分子振动穿越空气的纵波（或其她媒质中的横波）。在空气中，声音是作为高低压力变化传播的，高低变化的速度决定了频率。气压变化程度决定了幅度。

麦克风通过感受薄膜偏移捕获到的声波，成比例的转换成电压或电流。作为结果的电信号，通常是由模数转换器提供的的已编码的数字信号序列。最常见的格式，脉冲编码调制，将在5.1.1部分介绍。

如果相同的编码数据序列流经一个兼容的数模转换器，通过放大器送往扬声器，便可以产生声音。这样提供给扬声器的电压信号，都是每一个瞬间由电脑提供的采样值经过数模转换后比例放大所得来的。扬声器上的电压会导致圆锥体向内或向外位移，这样的位移时刻压缩附近的空气，从而产生声波。

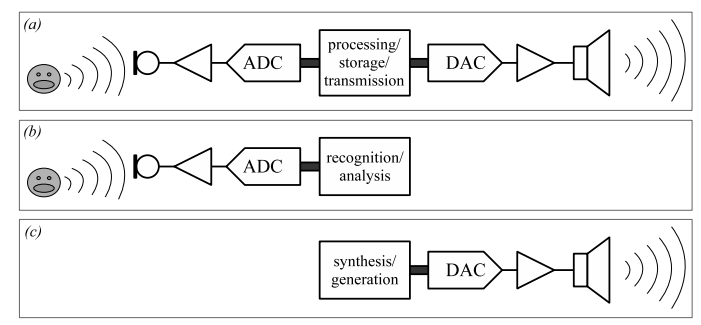


Figure 1.1 Block diagram of three classes of digital audio system showing (a) a complete digital audio processing system comprising (from left to right) an input microphone, amplifier, ADC, digital system, DAC, amplifier and loudspeaker. Variations also exist for systems recognising audio or speech (b), and systems synthesising audio (c).

上图所示的处理过程，代表了任何数字音频处理系统的主要步骤。音频，即本例中自由空气里传播的语音，被麦克风转换成电信号，然后放大以及可能会有的滤波，然后被ADC转换成数字信号。进入数字领域后，这些信号可以被很多方式处理，发送或者储存，或者被matlab试验。相反的过程，会将这些信号转换成声音。

上图中的处理、储存和传输系统都可能是串联或者并联的，各种情况下标准不一。光纤和无线传输也很受欢迎。

这个基本系统的变化如上图使用了其分析或合成声音的部分。立体声系统需要两个麦和扬声器，有些甚至更多。每个简单的放大器，图上的adc和dac模块也会隐藏某些系统中的复杂性——如模拟滤波、自动增益控制等，作为基础放大器的增强功能。

ADC个DAC也有自己的特点：采样率、技术、信噪比、动态范围等，通常由他们输出的位数决定。

采样

一段音频采样序列首先需要考虑的是连续采样点间的时间间隔，通常它们被设计成均匀的。这些间隔的频率即采样率。在上图它应该由一个周期时钟信号发出送给adc和dac，需注意两个采样率没有必要完全相同，因为数字处理本身可以更改采样率。根据著名的奈奎斯特准则，最高频率达到采样率的一半时便能被精确表达。

ADC送出的采样本身是16位深的固定点，高端系统可能将这个标准提高到20甚至24bit。在计算机中处理它们可能使用固定或浮点数表示。合理的一般化准则是，在16位输入输出的系统需要20位定点分辨率来操作和处理。

缺乏其他因素时，基本准则是n位相同的数字音频信号最好有以下动态范围（the ratio of the biggest amplitude that can be represented in the system to the smallest one）：

电话音质的语音，分辨率在相应设备上低至8-12位。GSM标准的手机上14位为标准。电话音质也叫电话相同音质，对于声音通信是可用的，但不能算是高音质。出于这个原因现代声音通信系统倾向于在实践中再增加8位的分辨率。

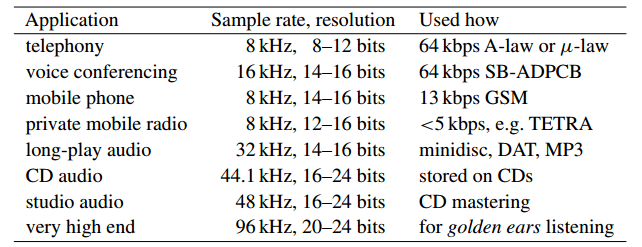
采样率从电话音质的7.2khz到8khz、cd音质的44.1khz不等。长时间播放的数字音频系统偶尔选择32khz，高音质系统选择48khz，近年也流行增加到96khz。除了对于某些金耳朵听众，是否96khz的采样率对于通常听不见高于18khz的人耳有用是可争辩的。但这样的系统也许对宠物更友好：据说狗的听力高达44khz，猫则高达80khz。

注释1：喜欢阀放大器的顽固音频爱好者花费多年薪水购买扬声器，这些人宣称自己能听见20khz，所以他们是金耳朵。

信息框1.1音频保真度

需要注意的是整个转换过程是不确定的：你听到的是声波冲击耳膜，而电脑获得的声音由空气直接送达，其间可能经过一些障碍，如被麦克风拾音、激励震动膜，被转换成电信号、放大，接着被采样。放大器会产生噪音、失真，并不是完全的线性元件。麦克风则是各种意义上的糟糕。模数转换同样饱经非线性、噪音、失真和电压采样过程的精确度导致的量化误差。所有上述原因可能是你的采样结果与希望的真实录音不同，或者裸耳回放当中信号发生的细微变化是否明显将难以预测。

Table 1.1. Sampling characteristics of common applications.



采样率及常见场合的采样精度，上表已列出。

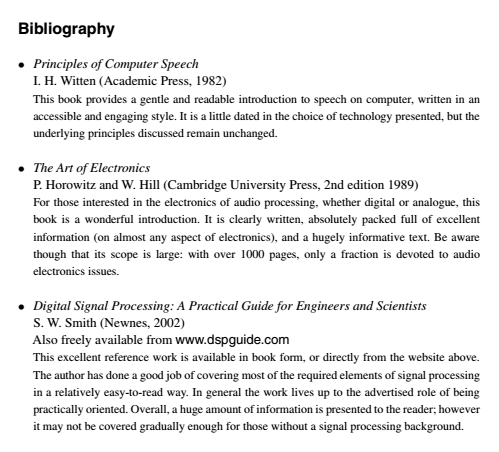
1.4 总结

大多关于转换和传播的过程的技术细节已经超出本书的范畴，相关内容请移步参考文献。通常来说，声音处理专家不需要了解声音的采集和回放过程就能完成自己工作。因此，我们讨论的范畴主要在信号的处理/保存/传输，以及图1.1上的信号辨别/分析以及合成/生成，而忽略模拟部分的技术细节。

对人类来说声音有一些特征：如时域的持续时间、节拍、启动时间和衰减时间等，也有频域的特征如声调（tone）和音高，以及定义不甚明确的音质、音色、音调（tonality）等等，通常声波承载了信息：如火警警报音、狮子的咆哮、婴儿的哭泣、雷声隆隆或国歌等。

然而正如我们所见，ADC采样（大多经过脉冲编码）所得的声音，仅仅是通过瞬时采样得到的幅度值来表达声音的容器。本书接下来将试图建立这些数字和声音的联系，以及试图理解、解释这些声音。

参考文献



1. **基本音频处理**

声音大多是matlab处理的，通过处理每个双精度的浮点数采样值。这些值和采样率这个因素结合之后便能完全表达声音的特征。通常来说平台间的采样率相去甚远，但它们都使用固定数值存储采样序列。其他系统可能难以解决这个问题，但matlab却能很好地工作，因为它不用考虑向上或向下溢出的问题。

理论上matlab在存储音频时作为一个向量容器工作。声音向量像其他matlab向量一样加载、读取、添加、绘图等。本章将就声音处理的特殊问题做讨论，作为后续章节一些处理和分析技术的基础。

在讨论其他问题（基本处理方法、后续分析和处理）之前，本章概览了matlab的输入输出部分，包括了录音和回放功能。有一节介绍了时域和频域的绘图技术。最终是生成所需声音和噪声的方法。

2.1 在matlab上处理声音

采样率足够高时，双精度采样已经几乎可以忽略量化误差，所以matlab环境是很安全的。但由于固定值格式的工作模式，matlab的输入输出环节仍可能产生其他量化误差。因此我们将讨论输入输出的问题：matlab上的录制和回放和音频文件处理的问题。

2.1.1 录制声音

直接在matlab上录制声音要求用户指定需要录制的采样数、采样率和通道数和采样格式。如：当要录制电脑麦克风送出的双精度浮点向量采样时，matlab应输入如下指令：

speech=wavrecord(16000,8000,1,’double’);

上述代码以8000khz的采样率记录了16000个采样，之后存入了16000个元素的向量，命名为speech。“1”参数表示录音为单声道。本指令仅在windows下工作，如在linux或macos下请使用audiorecorder()函数，或使用独立音频程序录制音频（如一些大胆且优秀的开源程序），然后用我们即将提到的方式加载进matlab。

【信息框2.1 音频文件格式

Wave：该格式文件通常有.wav后缀，它可以通过文件的头字段容纳许多不同的音频数据。采样率、通道数和采样精度都被wave文件声明了。相比其它不声明这些信息的文件格式，它更容易被分辨，同时matlab也是支持它的。声频工作中，wav文件包含pcm数据，是单通道，并以16bit为精度。支持的采样率从8khz到44.1khz不等，16k、24k、32k和48k也都很常见。

Pcm和raw格式包含没有头文件和gap的脉冲编码流模型数据。它们被设为单声道但文件中并不声明采样率和精度——音频研究人员请一定记住每一个自己使用的这两种文件。他们能被matlab读取和改写，但并不作为音频格式而对待。但它们由于历史原因常被研究人员采用，也许是由于早期使用c、c++等语言做针对它们的研究的原因。

A-law和u-law是对数压缩的字节格式音频。每字节代表了相等的12bit的线性pcm值。它常用于8khz采样率的产所。同时au后缀（常见于unix和linux系统）不包含关键的采样率信息，请大家记住。Matlab原生支持这种格式。

其它压缩格式还有如MP3、mp4，或特殊的乐器格式如midi以及各种专有格式。】

如果使用audiorecorder（）函数，该进程会首先创造一个录音目标文件，声明采样率、精度、通道数，然后再开始录音：

aro=audiorecorder(16000,16,1); record(aro);

此时麦克风已经开始录音，稍后可停止录音和回放：

stop(aro); play(aro);

将录音转成更常见的向量形式，则需要getaudiodata（）函数：

speech=getaudiodata(aro, ’double’);

其他指令如：pause（）暂停 和resume（）重新开始，将从录音控制处理的单元考虑，在完整的录音回放操作指令背景下，它们将共同建立良好的交互语音试验环境。

2.1.2 保存和重放声音

上述案例。“speech”向量由双精度数组成，但使用16bit精度录制。16bit格式的取值范围是-32768到+32767，但转换到双精度时被缩放到+/-1.0，且这是matlab中最常见的缩放形式，以后我们也将经常使用。这种格式下录音采样的整数值32767将被储存成+1.0的浮点数，-32768的整数值对应-1.0。

回放存成浮点数格式的声音向量一样容易：

sound(speech, 8000);

请注意声明声音向量的名字和采样率（本例的8khz将回放变调后的声音）。如果你的麦克风扬声器与pc相连，可尝试播放这些指令。录下一个简单的语音后增加/减少50%采样率然后试听回放的声音。

有时处理或其他对于音频向量的操作将导致采样值超出可表示范围，或仅有很小的值。使用sound()函数播放它们时分别会遇到削波、听不见的情况。这些情况下，另一些可选指令能自动修正音频向量值使它们优先基于最大振幅元来播放声音：

soundsc(speech, 8000);

本指令倾向于将过小的音量放大，过大的则衰减。

我们也能自行用相似的指令自己修改声音：

sound(speech/max(abs(speech)), 8000);

Matlab常用来开发将移植到定点计算架构上的音频算法，如整数型dsp或微处理器。这些情况下需要保证这些技术兼容整数运算而不是浮点数。因此双精度浮点数向整数的转换很重要。使用wavrecord和getaudio函数将双精度数转换成-32768~+32767的16位整型数。

这些音频输入和输出指令将组成声音实验的基石：图像和光谱图显示能力很强大，但甚至许多研究人员都不能通过它们辨认具体单词。听起来完美的声音，经过很小的处理后，有可能造成极低质量的结果——这种现象单绘图无法描述。人耳对于工程领域来说是相当精巧的对象，它是专门用来测试听觉的，眼睛并不能完全通过可视化声音来识别声音的特征。偶尔绘图能在理解识别声音上有好效果，但通常还是听来得更好。

用matlab绘制声音的时域图像很简单：

plot(speech);

有时x轴需要用秒为单位：

plot( [ 1: size(speech) ] / 8000, speech);

此处采样率需再次声明。

2.1.3 声音文件处理

声音研究领域，声音文件常为原始的PCM格式。文件仅仅有采样值组成——没有采样率等信息。此外多与8bit的采样可能还有字节问题，如果记录或处理的计算机更换过的话。

将pcm文件读进matlab使用fread（）函数，它其中的参数声明数据的精度和字节序（见信息框）。首先用命名打开文件：

fid=fopen(’recording.pcm’, ’r’);

然后读取整个文件，注意一次性读进同一个向量：

speech=fread(fid , inf , ’int16’ , 0, ’ieee-le’);

读入整个16位文件后（inf表示无限值），格式为IEEE小端模式，在pc中常用。不常用的一种为：

speech=fread(fid , inf , ’uint16’ , 0, ’ieee-be’);

文件将被读取成16位无符号整数，使用大端模式（大型unix主架构使用）。

记得关闭打开的文件是个好习惯：

fclose(fid) ;

知道如何在matlab中保存和加载数组也很有用。其内建的二进制格式语音数组能使用save指令保存，load指令加载，常见后缀名为.mat。

【信息框2.2：音乐文件格式

MP3，其后缀名为.mp3，德国Fraunhofer Institute 发明的标准压缩格式。它在世界范围内卷起了风暴，几乎成为了最广泛使用的音频格式。MP3其实是MPEG的三层叠放版本，它激发了大量相似或相同的其他格式。如……

Ogg Vorbis，是…………】

若两个向量分别叫speech和sppech2，它们可以被如下指令保存在当下目录：

save myspeech.mat speech speech2

然后用如下指令加载：

load myspeech.mat

这两个向量将被加载进matlab。与fread()不同，这个函数，向量名称座位存储好的文件识别。

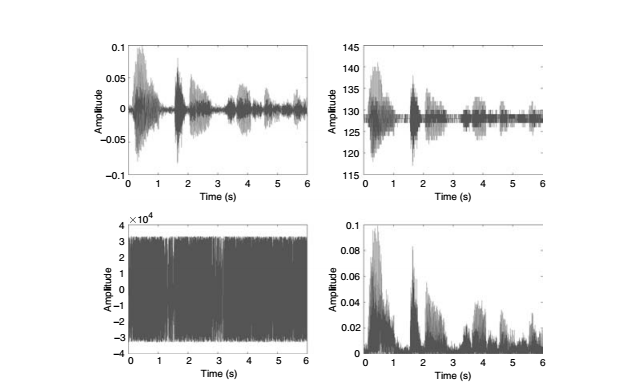
2.1.4 声频转换问题

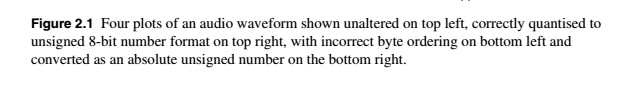
基础条件不同的声音最好用耳朵分辨。调好音量避免听力损伤。绘图用看的也可以。图2.1直接绘制了，它在上面的图将取值量化到了无符号8bit范围，，底部那张则使用错的比特数命令绘制(此处16bit为大字节序数)，无符号数。所有案例都听得清话语（同时使用swapbytes()和soundsc()）。

还有一个问题需注意，即两倍长或一半长的录音，有可能是bit数不匹配所致。

耳朵是好用的识别工具。升高采样率使声音尖锐，降低采样率使声音低沉。不匹配的位深使得声音中含有大量噪音，无符号/符号数的混合将导致失真和噪声严重的语音（尤其大音量）。

不匹配的精度播放会导致噪音，但仍能听清。





2.2 常规化

基础声音处理的最后一个过程是声音向量的常规化。之前提到这样一个避免削波的操作：

sound(speech/max(abs(speech)), 8000);

重复一遍：本操作旨在将每个声音元素向量规范到-1.0到+1.0之间的值。但我们输入的声音可能是16位有符号线性不动点格式的，意即比这大很多倍的值。即使声音在正确范围内表示，也有可能使采样超出原声的范围。

因此使matlab的估计与缩放音频在期望范围内是不错的练习，除非保持特定数字系统的精确度，或研究它们的量化问题。

两种方式缩放：

绝对尺度：考虑声音输入的格式，以相近尺度计量(将每个输入向量中的元素按32768的16bit线性符号数表达出来)。

相对尺度：以相近尺度的最大值计量，之前播放函数使用的方法（+-1之间的那种）。

量表的选择根据声音还原度的需求选择，音乐要求还原度则选绝对，只需要看频谱则选相对更方便。

【信息框2.3 字节序问题

大小端是竞争关系。大端里最重要的字节首先播放、存储，sun或hp机上用，小端则是最不重要的字节最先播放存储，桌面pc和笔记本中的intel和amd处理器上使用。有些处理器（如ARM7）可使用变字节序。



现代计算机的可变接入位深是字节序问题复杂化了。如果所有机器都是宽字节那么很好处理，但困难度在于，如果有一未知系统，检查是否为小端、归类为非大端很容易，但反过来就难了。

例子：

下例为一个储存在宽字节系统的32位音频采样（常见格式）……

**2.3声音处理**

一大难题是声音向量应切片处理还是全部采样一次性处理。这些问题伴随着实时系统的输出问题而来：切片式处理往往能更快产生输出。以及处理能力需求程度的问题（如决定是否进程在输入的一个小隆起后输出，还是所有数据处理完再播出）。

Matlab中，短录音就这样整个处理没有问题，长的则会导致内存占用问题。还有就是，声音特征常常是随时间变化的（如语音），那么同特质的声音等量分段处理总是会更好一些。稍后（2.4）将讨论这种“分割”。

很多时域操作，如数字滤波器的工作就与输入声音的长度无关，也与其分割与否无关；如一个FIR (finite impulse response)滤波器如下：

y=filter(b, 1, x);

b是滤波器施加的对象，x、y之类是与输入输出采样相关的等长度向量，y的计算公式如下：

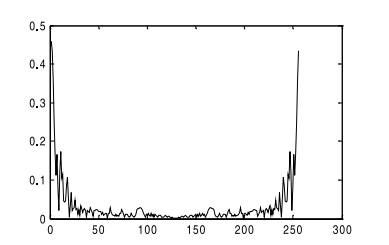
y(n) = b(1) × x(n) + b(2) × x(n - 1) + b(3) × x(n - 2) + …+ b(m + 1) × x(n - m). (2.1)

零极点滤波器（IIR）实现如下：

y=filter(b, a, x);

a、b向量都是滤波器的系数，xy同上一个公式，y的计算：

y(n) = b(1) × x(n) + b(2) × x(n - 1) + b(3) × x(n - 2) + …+b(m + 1) × x(n - m)-a(2) × y(n - 1) - a(3) × y(n - 2) … -a(m + 1) × y(n - m). (2.2)



**Figure 2.2 Absolute FFT plot for audio spectrum, with frequency index along the x-axis and  
amplitude along the y-axis.**

对于如下采样：

频域操作就相反了。因为它首先要求声音被fft转变为频域信号：

a\_spec=fft(a\_vector);

更常用的是，声音向量的长度二的若干次幂时，截断他们来满足fft的要求就可能，如下：a\_spec=fft(a\_vector, 256);

“转换多长的音频”是被引发的相关问题，以及“所需的频率分辨率”。在FFT输出频率的数量是基于样本输入的数量。2.5.2节中我们将进一步探讨，但目前说一下:

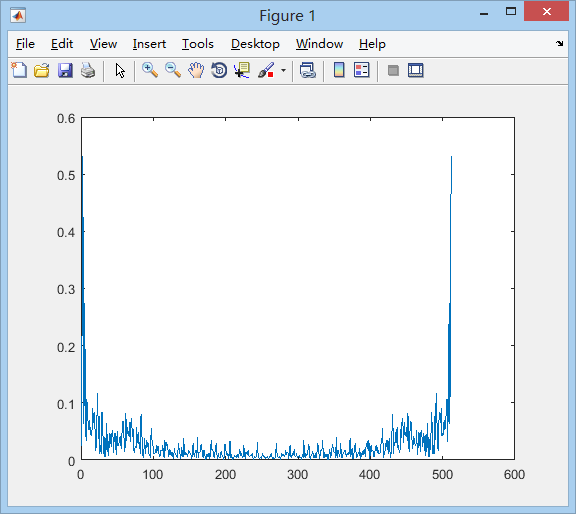
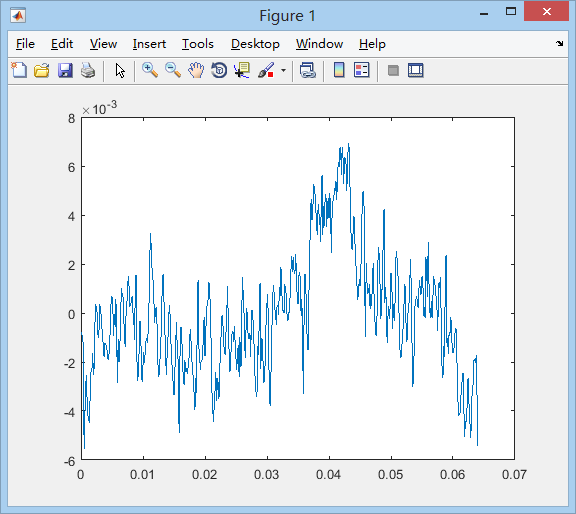
当长度为二的若干次幂时有如下便利的解决：

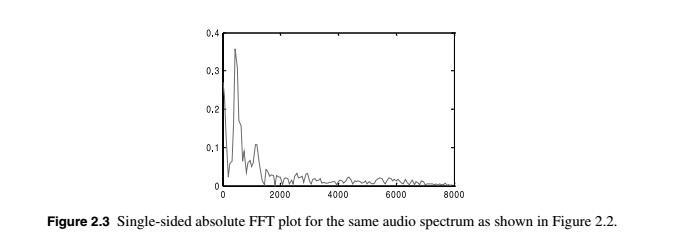
a\_spec=fft(a\_vector(1:256));

Matlab中输出的频域向量是复杂的。绘制Figure 2.2 的频率点能量双边绝对值图像使用：

plot(abs(a\_spec));

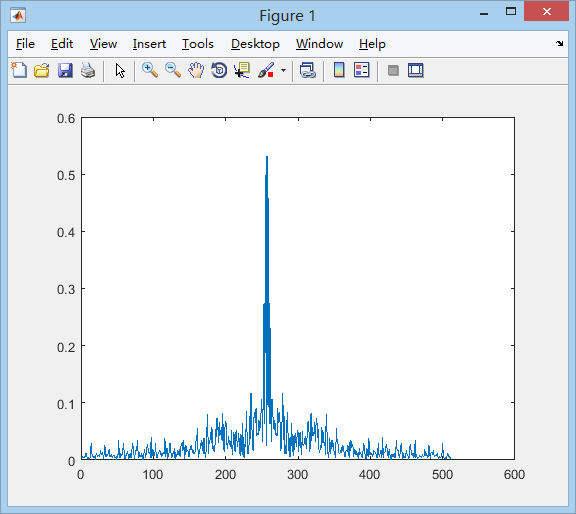
以下为同一段录音的时频域图形：





这张不寻常的图上，频率坐标轴（单位正确的话）从零开始，正中间是奈奎斯特频率，在最远端衰减至0。集中和不集中的频率都画出来了——此举效率不高。由于历史原因这点上matlab跟c或者fortran等软件不同。我们可以绘制更标准的图形（低频更加靠中间）：

plot(abs(fftshift(a\_spec)));



如果需要更有用放的单边频谱以变量Fs=8000和Ns=256分别描述fft的初始采样率和大小，以下操作：

plot( [1 : 2\*Fs/Ns : Fs], abs(a\_spec(1:Ns/2)), ’r’);

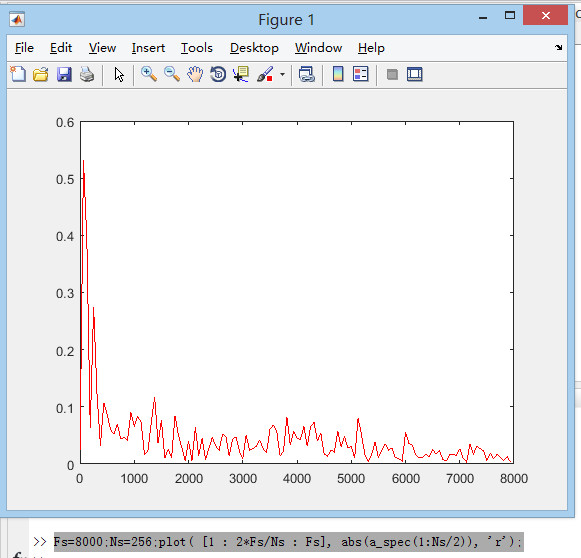


图2.3绘出了此图，它显然更有用，提供了物理上紧密相关的表示。“r”参数表示所绘图像用红色。

完成声音处理时，fft所采用的分析方式也会反映到频率向量上。这是好的，但如果采样数多于256很多怎么办？它们可以被分割哟！然后单独处理。

分割采样，不仅是因为256（打个比方）是个方便的大小，还有如下原因：

1. 声音是连续的（但想要试听的效果是实时的）
2. 声音特质不断改变，或短期特质的分析很重要（如一个不断升高的采样，较短的fft间隔可以分析出它的特点，但完整一次性做频率分析会导致整张图为白噪音）。
3. 一次性执行将导致内存受限（常见原因）
4. 想要的是长时间段里的散布式结果，而不是最终的一次性结果。
5. 延迟时间（第一个采样时钟和分析输出的延迟）最小化——这是语音交互系统的常见要求

**2.4 分割**

大多数上节提到的声音处理中分割帧可重要了，然而它本身也是一大论题。

声音的“特质”，我是说一段采样的特征。当一个采样被分析时可能出现：它的特征被分成两半，分别出现在不同的两帧里，窗口上找不到任何完整的特征，它被分析的过程本身淹没了。处于采样帧正中央的特征幸免于难，而被拦腰砍断的那些就惨了。考虑到视窗是（2.4.2），由于分析帧最后的部分将被忽略的更多，这个问题将更严重。解决丢失特征问题的方法是交叠采样。

**2.4.1 重叠**

重叠意味着并不直接将声音分割成按顺序的帧，每一帧包含一些前一帧的一部分和下一帧的一部分。交叠使得音频的特征以非连续的形式在接连交叠的帧中出现。

交叠的程度（通常以百分比出现）描述了前后帧所重复的数量。常见的百分比是25%和50%。

图2.4描绘了50%交叠。顶端图为波形全图，然后是交叠分割的帧，标记为0-6.新。帧间重复部分可以找到原始音频的所有特征。

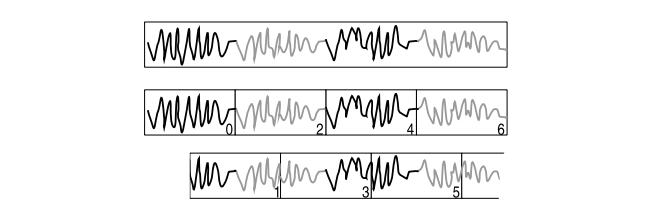


Figure 2.4 Illustration of an original audio recording (the upper waveform) divided into two  
offset sequences of analysis windows (two lower waveforms) with 50% overlapping frames so  
that no short-term auditory feature is obscured by straddling the boundary between analysis  
windows: it will appear unbroken in at least one of the two analysis streams.

而这种分割方式不好的一面就是：加倍了运算量。

分割与交叠的共同作用也导致一些分析上的困难。分析过程是一次性的，分析器的工作模式为一进一出。输入时声音将损失一次，输出时再损失一次。

每一帧进行处理时，如果是按平均音量缩放，有时称为自动归零，所有帧会一起输出平均音量。

首先是如何交叠帧的问题。它们不能一般连接，否则许多采样会直接放大两倍，直接相加也不行。

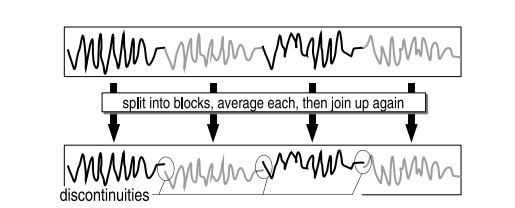
无交叠的图2.5也显示了另一个问题：帧被分离、处理、连接后，相邻帧可能会连不起来。这种问题很容易被听成“咔哒”声。其实几乎任何有意义的处理都会这样。

Figure 2.5 Illustration of an original audio recording (upper waveform) being split into equal  
length analysis windows, each of which is normalised, and then rejoined (lower waveform). The  
rejoined audio exhibits discontinuities between frames which would result in significant audio  
distortion.

**2.4.2 加窗**

许多分割与交叠问题的处理可用加窗变通解决。选择合适的窗函数可以缓解交叠时的诸多问题（(see, for example, Chapter 18 of [1] ）。窗函数也是避免吉布斯现象的好方法。

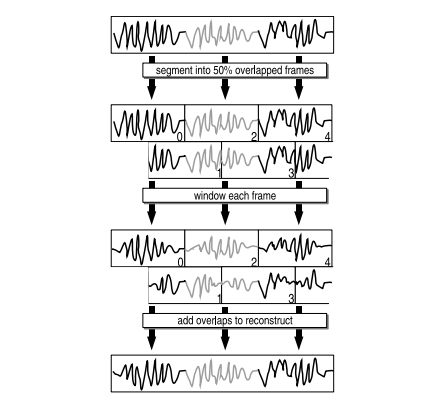
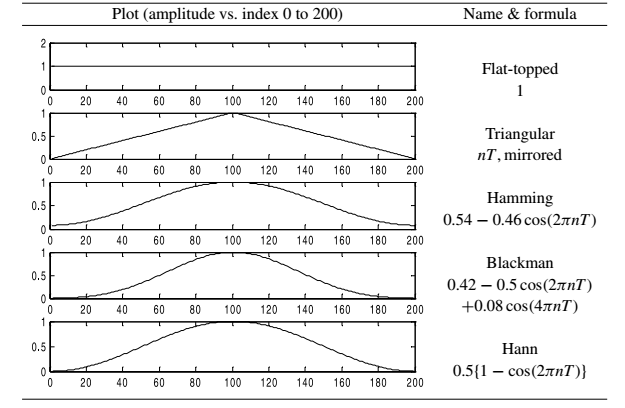


Figure 2.6 Illustration of an original audio recording (upper waveform) being split into 50%  
overlapped analysis frames, each of which are windowed before being summed together to  
reconstruct output audio which does not exhibit discontinuities (lower waveform).

图2.6显示了50%交叠分割的加窗帧重建的结果。

常用的预定义窗类型，他们的特征不同（许多信号处理书籍上有详细介绍）。如有疑问使用汉明窗即可（虽然它两边都不趋近于零）。表2.1给了几个常用窗函数。

Table 2.1. Common window functions.



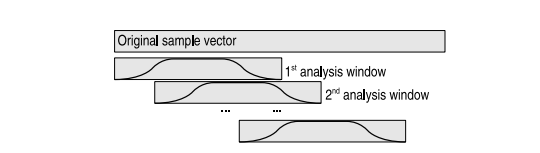


Figure 2.7 Illustration of overlap analysis employing a custom window shape.

还有许多其他窗函数，以及大家可以自己发明一个。以及，每个窗函数可以被各种扭曲放入一个分析帧，或分成两个并削平顶部——此举在25%交叠中很管用（图2.7）。

**2.4.3 连续滤波器**

2.4.1和2.4.2 我们讨论了如何分割-加窗-交叠语音以使不同的处理范畴过度平滑。

研究表明，相邻帧的处理分析方式不同时需要这些过程，若通过相同的滤波器，不谨慎考虑执行滤波的方式也会导致连续性问题。

来演示一个matlab可滤波和回放的语音。首先，录下几秒的声音，采样率8khz，命名为序列s。试听它：soundsc(s);

效果：正常录音，音量有放大

接着定义一个小数字滤波器在其上工作，为了测试方便就定义了h = (1 - 0.9375z-1) 这个变换。第五章将遇到它：语音的预增强函数。

实践过程：创建滤波器，应用在录音上，试听结果。

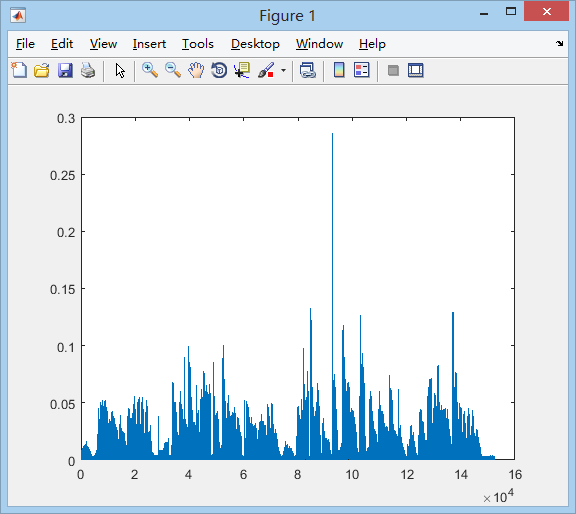
h=[1, -0.9375];  
y=filter(h, 1, s);  
soundsc(y);

听到的声音应该是“紧绷”或者“鼻音重”但也清晰的声音。接下来再次滤波，但分了采样帧，每帧单独滤波，再叠加起来。

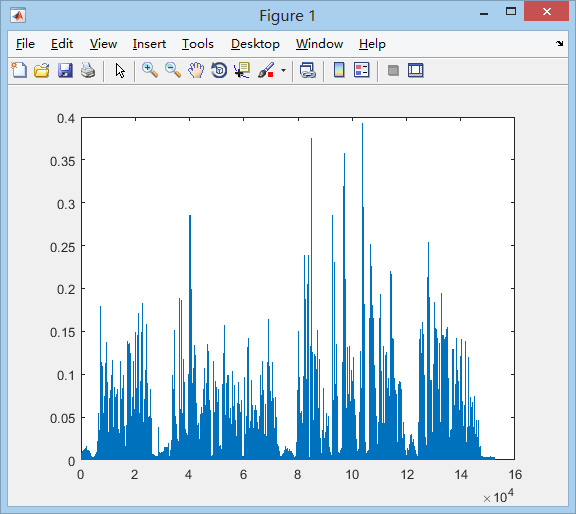
w=240;  
n=floor(length(s)/w);  
for k=1:n  
 seg=s(1+(k-1)\*w:k\*w);  
 segf=filter(h, 1, seg);  
 outsp(1+(k-1)\*w:k\*w)=segf;  
end  
soundsc(outsp);

效果与前一个相似，但有咔哒声。如果我们绘制了该滤波输出信号y和outsp，他们可能会无法区分(其实不是)。

plot(abs(y));



plot(abs(outsp));



只有用光谱法(参见2.6节)，这些咔哒声或不连续性才可见。

所以现在仍然存在两个问题：首先，是什么导致不连续性，其次我们可以怎样防止它们发生？如果考虑滤波过程，原因可能显而易见。实际上，FIR滤波器是描述一个“当前输出采样值等于当前输入采样值加上先前P输入采样值，每个值都是这种共同作用的结果，”的过程，P是滤波器阶数。因此第一级滤波器即将输入信号减去上一个信号的0.9375 倍：

y[n] = s[n] - 0.9375 × s[n - 1] (2.3)

参数n取值为0到语音向量的长度。问题发生于每帧起始处，也就是n=0处[1]。当第二帧被滤波时，该帧计算y[n]时，n=0，则y[0]由s[-1],其值为0.而不分割来滤波时这个参数为240（在本例中）。由给定y[240] = s[240] - 0.9375 × s[239] ，因为s[239]已知，输出会与分割时计算的不同。

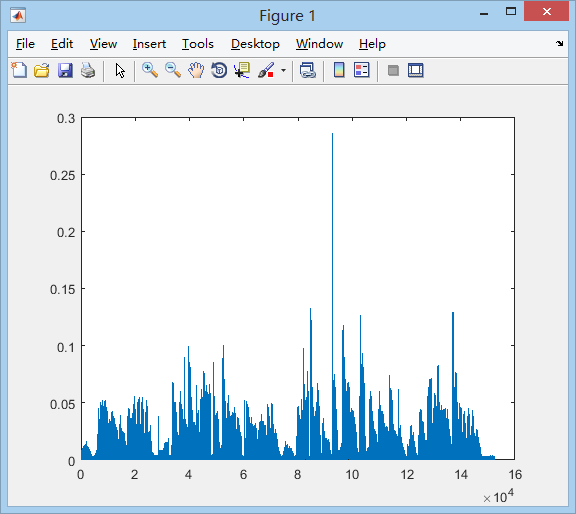
这只是简单的一阶滤波器而已，如果阶数更多，每个帧起始处受到的影响就更大，不连续性就更明显。当到达十阶时，将严重影响输出的音质。

数字滤波器发生的这些现象术语为滤波器的“历史状态”，或“内部状态”，对一个数列进行滤波时，历史状态自动在采样间更新。然而在序列初始处滤波时，历史状态一定要声明，否则将默认为零。在语音分割的情况下，历史状态实际上是被重置为零在每个帧的边界–难怪输出是扭曲的。

[Matlab](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[实际上](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[提供了](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[一种](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[方便](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)的[方法](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[来](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[设置](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[和](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[存储](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)的[内部](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[历史记录](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[使用](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[filter](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[(](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[)](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[命令](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)[。](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Matlab%20actually%20provides%20a%20convenient%20way%20to%20set%20and%20store%20internal%20history%20using%20the%20filter()%20command." \t "https://cn.bing.com/dict/_blank)我们将说明这通过编辑和重复分割上面的示例：

w=240;  
hst=[];%定义空序列  
n=floor(length(s)/w);%向负无穷去整  
for k=1:n  
 seg=s(1+(k-1)\*w:k\*w);  
 [segf, hst]=filter(h, 1, seg, hst);  
 outsp2(1+(k-1)\*w:k\*w)=segf;  
end  
soundsc(outsp2);

plot(abs(outsp2));



以上代码我们在filter() 指令中加入了历史状态序列，每次运行滤波器都存储了内部历史状态。在每帧输入时声明历史状态，我们在整个处理过程里成功将这些状态置位成应有的量，因此我们确保帧之间的一个平滑滤波操作。前面的代码产生的输出音频应该避免了明显的咔哒声和不连续性。

注：1：matlab中序列第一位是1不是0，但本书的讨论在工程实际基础上从0开始。

**2.5 分析窗尺寸**

2.3讨论了一些切片操作，但仍不知道怎样确定切片、帧或分析窗的具体大小。一般，大多（完全基于matlab的）音频算法在较大的数据块上运行更高效。很自然有处理更大的分析帧的趋势，诸如电话语音之类的处理中延迟时间等因素将会是很大问题。

另一个限制窗大小的主要原因是声音特质在分析窗内部变化的时候。Infobox Visualisation of signals on page 32 对这个问题描述得更好，简单来说FFT的切片过大时将隐藏一些声音细节。

于是接下来将讨论：信号稳定性和时域-频域分辨率。

**2.5.1 信号稳定性**

大多信号需要时变的分析手段。一个演奏的音符是稳定的，当转音到另一个音符时声音变化的过程很明显（频率、音量、音调、音色等）。

一个分析当前演奏音色的音符的软件，粗切片分析窗为单个音符变化的部分或更小是正确的。每个窗运行一个fft，再关注峰值点。如果窗跨越了两个音符，就很难确定是哪一个了。

Fft默认其分析的东西是频率稳定的,不稳定的信号必然导致分析的不精确。

保证好窗的大小能满足分析需求，实际应用中足够小的才行。

语音分析中，说话者的肌肉运动导致音色的缓慢变化。通常认为20–30 ms 内语音是近似稳定的，分析帧小于20ms这个长度就可以。

稳定性分析也常见于线性预测（5.2.1）等地方，使用时，必须仔细每个匹配已知要处理的音频信号的特征。

**2.5.2 时频分辨率**

回到FFT，输出频率矢量对采样频率Fs赫兹的音频做N采样FFT、包含N/2+1个正值的频率点。每个点从原始信号中的一个小范围的频率中计算能量，槽宽度与采样率和分析采样的数量相关（Fs/N ）。或者说其宽度等于分析窗口所涵盖的时间跨度的倒数。

为了实现更高的频率分辨率，我们需要分析采样率的时间更长。然而，对于快速变化的信号，收集更多的采样意味着我们可能会错过一些时域特征，如2.5.1节所述，点击信息可视化在32页。

百度：频率分辨率可以理解为在使用DFT时，在频率轴上的所能得到的最小频率间隔，其中N为采样点数， IMG_257 为[采样频率](http://baike.baidu.com/subview/82683/82683.htm" \t "http://baike.baidu.com/view/_blank)， IMG_258 为采样间隔。所以 IMG_259 就是采样前模拟信号的时间长度T，所以信号长度越长，频率分辨率越好。

关于fft的工作原理：

以这里有一个基本的原则经营的不确定性：一个FFT可以权衡更高的频率分辨率(样本)或更高的时间分辨率(样本少)，但无法同时兼顾。解决方案随问题的要求而变，但有几个替代FFT的频率估计，这往往可能在长期和短期分析窗口分别执行两个FFTs，

后来6.2.2节中我们将介绍更多的计算密集型试图满足需求的高频率分辨率、高时间分辨率的方法。

**2.6 可视化**

波形绘图：最容易和基础的波形可视化方式，长录音最方便快捷的浏览方式。但无法表达全部的声音信息。

频率光谱图：之前提到过的基础好用的工具。不恰当使用容易观丢失音频特征。整段长音频的可视化工具：

短时傅里叶变换（Short-time Fourier transform (STFT)）：滑动窗窄边傅里叶变换，在很长恶采样间依次重复工作的时-频分解或分析。结果为时序独立光谱，以时间为横轴，不是三位坐标，光谱为数据转换成的大量小方块：spectrogram x(x) = |X(τ,ω)|2.

在Matlab这是很容易执行的specgram()函数，然而是注定不幸Matlab在未来版本中删除。. 更换的spectrogram() 是信号处理工具箱中可用的多做同样的事情的函数：[有许多](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=having%20many%20options%20regarding%20analysis%20window%20size%2C%20overlap%20and%20number%20of%20sample%20bins." \t "https://cn.bing.com/dict/_blank)的[选择](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=having%20many%20options%20regarding%20analysis%20window%20size%2C%20overlap%20and%20number%20of%20sample%20bins." \t "https://cn.bing.com/dict/_blank)——[关于分析窗口大小、重叠和](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=having%20many%20options%20regarding%20analysis%20window%20size%2C%20overlap%20and%20number%20of%20sample%20bins." \t "https://cn.bing.com/dict/_blank)采样[数。](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=having%20many%20options%20regarding%20analysis%20window%20size%2C%20overlap%20and%20number%20of%20sample%20bins." \t "https://cn.bing.com/dict/_blank)

[一个光谱基本上是一组STFT绘制为对时间频率与强度(z轴)作为](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=A%20spectrogram%20is%20essentially%20a%20set%20of%20STFT%20plotted%20as%20frequency%20against%20time%20with%20the%20intensity%20(z-axis)%20given%20as%20a%20greyscale%2C%20or%20colour%20pixel." \t "https://cn.bing.com/dict/_blank)[灰度](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=A%20spectrogram%20is%20essentially%20a%20set%20of%20STFT%20plotted%20as%20frequency%20against%20time%20with%20the%20intensity%20(z-axis)%20given%20as%20a%20greyscale%2C%20or%20colour%20pixel." \t "https://cn.bing.com/dict/_blank)[或](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=A%20spectrogram%20is%20essentially%20a%20set%20of%20STFT%20plotted%20as%20frequency%20against%20time%20with%20the%20intensity%20(z-axis)%20given%20as%20a%20greyscale%2C%20or%20colour%20pixel." \t "https://cn.bing.com/dict/_blank)[颜色](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=A%20spectrogram%20is%20essentially%20a%20set%20of%20STFT%20plotted%20as%20frequency%20against%20time%20with%20the%20intensity%20(z-axis)%20given%20as%20a%20greyscale%2C%20or%20colour%20pixel." \t "https://cn.bing.com/dict/_blank)[像素](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=A%20spectrogram%20is%20essentially%20a%20set%20of%20STFT%20plotted%20as%20frequency%20against%20time%20with%20the%20intensity%20(z-axis)%20given%20as%20a%20greyscale%2C%20or%20colour%20pixel." \t "https://cn.bing.com/dict/_blank)[。](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=A%20spectrogram%20is%20essentially%20a%20set%20of%20STFT%20plotted%20as%20frequency%20against%20time%20with%20the%20intensity%20(z-axis)%20given%20as%20a%20greyscale%2C%20or%20colour%20pixel." \t "https://cn.bing.com/dict/_blank)语音分析，频谱是一个很好的可视化语音结构以及它如何随时间变化的方法。

利用Matlab绘制的一个例子显示了一个声谱图2.8所示，灰度中绘制。一些音频研究人员希望他们的图谱的颜色。这其实只是个人喜好的问题。

**2.6.1 关于坐标的简短说明**

FFT或频谱的横轴，传统上用来表示频率，垂直轴将显示幅度。. 一个光谱，相比之下，情节时间沿水平轴、垂直的频率和振幅对z轴(如颜色或灰度)

[由于音频信号的动态范围大，](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=Due%20to%20the%20large%20dynamic%20range%20of%20audio%20signals%2C" \t "https://cn.bing.com/dict/_blank)[很常见](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=it%20is%20common%20to%20plot%20the%20logarithm%20of%20absolute%20amplitude%2C%20rather%20than%20the%20amplitude%20directly%3B" \t "https://cn.bing.com/dict/_blank)的[情节绝对振幅的对数，而不是直接](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=it%20is%20common%20to%20plot%20the%20logarithm%20of%20absolute%20amplitude%2C%20rather%20than%20the%20amplitude%20directly%3B" \t "https://cn.bing.com/dict/_blank)的[振幅；](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=it%20is%20common%20to%20plot%20the%20logarithm%20of%20absolute%20amplitude%2C%20rather%20than%20the%20amplitude%20directly%3B" \t "https://cn.bing.com/dict/_blank)[因此](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=thus%20the%20spectrum%20will%20often%20be%20plotted%20as%20a%20power%20spectrum%2C%20using%2020%20%C3%97%20log10(spectrum)%2C%20or%20with%20the%20Matlab%3A" \t "https://cn.bing.com/dict/_blank)，[频谱往往会被绘制为功率谱，使用20×log10(光谱)，或用Matlab：](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=thus%20the%20spectrum%20will%20often%20be%20plotted%20as%20a%20power%20spectrum%2C%20using%2020%20%C3%97%20log10(spectrum)%2C%20or%20with%20the%20Matlab%3A" \t "https://cn.bing.com/dict/_blank)

semilogy(spectrum); 半对数坐标系(光谱)；

x轴显示频率一般线性绘制。此标签轴默认的模式在FFT输出光谱在大多数情况下不是特别有用。一个更好的方法是指定自己要么指定坐标单位缩放到0和1赫兹之间（DC到奈奎斯频率），[或弧度](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=or%20in%20radians%20between%200%20and%20%CF%80." \t "https://cn.bing.com/dict/_blank)缩放到[0和Π之间。](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=or%20in%20radians%20between%200%20and%20%CF%80." \t "https://cn.bing.com/dict/_blank)

res=pi/size(spectrum);  
semilogy(res:res:pi, spectrum);

这个弧度单位，通常记作w的独立频率变量，2Π表示采样频率，0作为直流分量，π作为奈奎斯特频率。它被称为角频率或偶尔的固有频率，并认为频率是围绕一圈。

当处理系统或者是过/欠采样时它是很有用的，撇开这点不谈，它在数学上更连续，因为这意味着可以导出方程，来描述不取决于采样率的绝对值的采样系统。

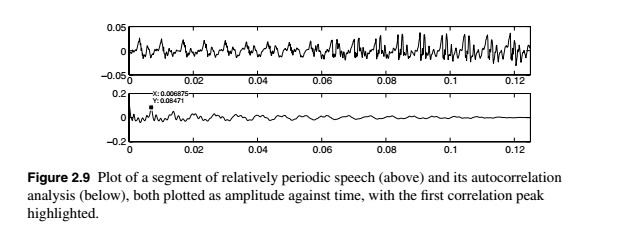
**2.6.2 其它可视化方法**

正如你期望，许多其他更多可视化方法存在，有些已演变为特别适合查看某些量。对语音与音频工作最有用之一是将5.2.1节中描述的线性预测系数谱。. 在这里，另一方面，提出了两种非常普遍的和有用的方法–自相关图和倒频谱。

**2.6.2.1 自相关图**

自相关图是绘制一个信号的自相关的细节。相关性是指比较两个信号的相似性，它们之间可能存在的任何目前或过去的相似性（多为过去）。数学上这是相对简单的。

[我们先](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=We%20will%20start%20with%20an%20equation%2C" \t "https://cn.bing.com/dict/_blank)来看[一](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=We%20will%20start%20with%20an%20equation%2C" \t "https://cn.bing.com/dict/_blank)看[方程，](https://cn.bing.com/dict/clientsearch?mkt=zh-CN&setLang=zh&form=BDVEHC&ClientVer=BDDTV3.5.1.4320&q=We%20will%20start%20with%20an%20equation%2C" \t "https://cn.bing.com/dict/_blank)定义两个向量x和y之间在时间t进行互相关，过去k个瞬间计算时间：

在Matlab中的这种分析是使用xcorr 函数执行函数的以两个向量中长度最短的那个为跨度进行分析：[c,lags]=xcorr(x,y);

“自相关是指信号在1个时刻的瞬时值与另1个时刻的瞬时值之间的依赖关系，是对1个随机信号的时域描述。”

只有一个参数输入xcorr时，它将对这个量做自相关分析。可观察其周期。

如：输入有两个峰值时，自相关输入在一个峰以另一个峰倍乘为结尾时会很大。时移k是两峰值商最大输出的响应。更复杂的系统里，自相关函数是对周期敏感的手段。

假设有一段短音频speech。有一段声音在绘图中看起来很规则，或许是浊音。假设它处于第9000和10000个采样之间，先将它提取出来：segment=speech(9000:10000);

接着做自相关分析，绘制输入输出图像：

[c,lags]=xcorr(segment);  
subplot(2,1,1);  
plot([0:1:1000]/8000,segment);  
axis([0, 0.125. -0.05, 0.05]);  
subplot(2,1,2);  
plot(lags(1001:2001)/8000,c(1001:2001));  
axis([0, 0.125. -0.2, 0.2]);

Axis指定y的范围（-0.05到0.05等）由输入的采样幅值和成分决定，正确使用时图像会铺满所有绘图区域。见图2.9。第一个峰出现在x值为0.006875即6.875ms位置。测量两峰值的距离后与第一个自相关峰值出现的位置，发现了其第一大用途：测量周期。

**2.6.2.2 倒谱**

以倒频率/逆频率作图，它能一定程度上相似但其实不同的信号（as we will see  
in Chapter 3 ）。

倒谱（cepstrum)一种信号的傅里叶变换谱经对数运算后再进行的傅里叶反变换。由于一般傅里叶谱是复数谱，因而又称复倒谱。是的，真的是傅里叶变换的两个步骤，虽然在实践中，第二个是经常作为傅里叶逆变换，

再利用Matlab，一个非常简单的例子就是演讲的情节倒频谱分析与上图。这是相当简单的情节–不很准确地按倒谱的原始意义，但肯定有用：

ps=log(abs(fft(hamming(length(segment)).\*segment)));  
plot(abs(ifft( ps )));

最有可能的，如果语音片段是一样大，用于自相关的例子。由此产生的倒谱将产生巨大的z直流峰值和很多细节模糊。可以使用matlab视图工具放大内容，也可减少被分析的内容。

倒谱和自相关图在相同位置有峰值。在8hz语音的256点倒谱上x轴的56单位处对应4000∗56/256=875hz。6.1.5处将用到这个分析。

以上两种分析都能显示基频，执行类似任务时他们长度不同：倒频谱检测可能会更易于自动实现（假设限制搜索范围），当然，倒可以突出自相关图中错过的特征。相关图另一方面容易计算和缩放远远大于倒倒谱。都是可视化和分析工具提供给研究人员的语音，很多时候同时应该采用，以分析比较它们不同的地方。

最后，注意的是，倒谱常常采用非线性频率波段的映射，由此产生的Mel频率倒谱系数(MFCC)用于说话人识别，语音识别和其他几个语音分析任务。

**2.7 声音生成**