CHAPTER

14

Introduction to Digital Filters

Digital filters are used for two general purposes: (1) separation of signals that have been combined, and (2) restoration of signals that have been distorted in some way. Analog (electronic) filters can be used for these same tasks; however, digital filters can achieve far superior results. The most popular digital filters are described and compared in the next seven chapters. This introductory chapter describes the parameters you want to look for when learning about each of these filters.

Filter Basics

Digital filters are a very important part of DSP. In fact, their extraordinary performance is one of the key reasons that DSP has become so popular. As mentioned in the introduction, filters have two uses: signal *separation* and signal *restoration*. Signal separation is needed when a signal has been contaminated with interference, noise, or other signals. For example, imagine a device for measuring the electrical activity of a baby's heart (EKG) while still in the womb. The raw signal will likely be corrupted by the breathing and heartbeat of the mother. A filter might be used to separate these signals so that they can be individually analyzed.

Signal restoration is used when a signal has been distorted in some way. For example, an audio recording made with poor equipment may be filtered to better represent the sound as it actually occurred. Another example is the deblurring of an image acquired with an improperly focused lens, or a shaky camera.

These problems can be attacked with either analog or digital filters. Which is better? Analog filters are cheap, fast, and have a large dynamic range in both amplitude and frequency. Digital filters, in comparison, are vastly superior in the level of performance that can be achieved. For example, a low-pass digital filter presented in Chapter 16 has a gain of 1 +/- 0.0002 from DC to 1000 hertz, and a gain of less than 0.0002 for frequencies above

章

14 数字滤波器简介

数字滤波器主要承担两大核心功能：(1) 解析混合信号，(2) 恢复失真信号。虽然模拟（电子）滤波器也能完成这些任务，但数字滤波器在性能上有着显著优势。接下来的七章将详细解析并对比各类主流数字滤波器。作为入门章节，本章将为您介绍学习各类滤波器时需要重点关注的参数指标。

过滤基础

数字滤波器是数字信号处理（DSP）的核心组件。其卓越性能正是DSP技术风靡全球的关键所在。正如前文所述，滤波器主要承担两大功能：信号*分离*与信号*恢复*。当信号受到干扰、噪声或其他信号污染时，就需要进行信号分离。举个例子，假设有一种设备能在胎儿腹中监测心电图（EKG），原始信号很可能被母体的呼吸声和心跳声所干扰。此时就需要借助滤波器将这些干扰信号分离出来，以便进行单独分析。

信号恢复技术用于处理因设备故障导致的信号失真。例如，可对使用劣质设备录制的音频进行滤波处理，使其更真实还原原始声音；同样地，该技术也可用于消除因镜头对焦不当或摄像机抖动导致的图像模糊。

这些问题可通过模拟滤波器或数字滤波器解决。哪种方式更优？模拟滤波器具有成本低廉、响应迅速、在幅频动态范围宽广等优势。相比之下，数字滤波器在性能表现上具有显著优势。例如，第16章介绍的低通数字滤波器在直流至1000赫兹范围内增益为1±0.0002，而高于该频率时增益则小于0.0002。

1001 hertz. The entire transition occurs within only 1 hertz. Don't expect this from an op amp circuit! Digital filters can achieve *thousands* of times better performance than analog filters. This makes a dramatic difference in how filtering problems are approached. With analog filters, the emphasis is on handling limitations of the electronics, such as the accuracy and stability of the resistors and capacitors. In comparison, digital filters are so good that the performance of the filter is frequently ignored. The emphasis shifts to the limitations of the *signals*, and the *theoretical* issues regarding their processing.

It is common in DSP to say that a filter's input and output signals are in the *time domain*. This is because signals are usually created by sampling at regular intervals of *time*. But this is not the only way sampling can take place. The second most common way of sampling is at equal intervals in *space*. For example, imagine taking simultaneous readings from an array of strain sensors mounted at one centimeter increments along the length of an aircraft wing. Many other domains are possible; however, time and space are by far the most common. When you see the term *time domain* in DSP, remember that it may actually refer to samples taken over time, or it may be a general reference to any domain that the samples are taken in.

As shown in Fig. 14-1, every linear filter has an **impulse response**, a **step response** and a **frequency response.** Each of these responses contains complete information about the filter, but in a different form. If one of the three is specified, the other two are fixed and can be directly calculated. All three of these representations are important, because they describe how the filter will react under different circumstances.

The most straightforward way to implement a digital filter is by *convolving* the input signal with the digital filter's *impulse response*. All possible linear filters can be made in this manner. (This should be obvious. If it isn't, you probably don't have the background to understand this section on filter design. Try reviewing the previous section on DSP fundamentals). When the *impulse response* is used in this way, filter designers give it a special name: the **filter kernel**.

There is also another way to make digital filters, called **recursion**. When a filter is implemented by convolution, each sample in the output is calculated by *weighting* the samples in the input, and adding them together. Recursive filters are an extension of this, using previously calculated values from the *output*, besides points from the *input*. Instead of using a filter kernel, recursive filters are defined by a set of **recursion coefficients**. This method will be discussed in detail in Chapter 19. For now, the important point is that all linear filters have an impulse response, even if you don't use it to implement the filter. To find the impulse response of a recursive filter, simply feed in an impulse, and see what comes out. The impulse responses of recursive filters are composed of sinusoids that exponentially decay in amplitude. In principle, this makes their impulse responses *infinitely long*. However, the amplitude eventually drops below the round-off noise of the system, and the remaining samples can be ignored. Because

1001赫兹。整个转换过程仅需1赫兹即可完成。这可不是运算放大器电路能实现的！数字滤波器的性能可比模拟滤波器好上*数千倍*，这彻底改变了滤波问题的处理方式。模拟滤波器主要关注电子元件的局限性，比如电阻和电容的精度与稳定性。相比之下，数字滤波器性能优异到常常被忽略，重点转向了*信号本身的*局限性，以及处理信号时涉及的*理论*问题。

在数字信号处理领域，人们常说滤波器的输入和输出信号都处于*时域*。这是因为信号通常通过在*时间*的固定间隔进行采样生成。但采样方式并非只有时域，另一种常见方式是*空间*的等间隔采样。比如想象一下，沿着飞机机翼长度方向每隔一厘米安装一组应变传感器，同时采集它们的读数。虽然还有其他可能的采样域，但时域和空域无疑是最常见的。当看到数字信号处理中的*时域*这个术语时，要记住它既可以指随时间采集的样本，也可以泛指采样所处的任何时域。

如图14-1所示，每个线性滤波器都具有**冲激响应**、**阶跃响应**和**频率响应。**这三种响应形式各自包含着关于滤波器的完整信息，只是呈现方式不同。只要确定其中一种响应，另外两种就能直接计算得出。这三种表示方式都至关重要，因为它们描述了滤波器在不同工况下的响应特性。

实现数字滤波器最直接的方法是将输入信号与滤波器的*卷积响应*进行*卷积运算*。所有线性滤波器都可以通过这种方式构建（这应该是显而易见的。如果还不明白，说明你可能没有足够的背景知识来理解本节的滤波器设计内容。建议先复习下前文的DSP基础章节）。当*卷积响应*以这种方式使用时，滤波器设计者会为其赋予一个特殊名称：**滤波器核**。

还有一种实现数字滤波器的方法称为**递归法**。当通过卷积实现滤波器时，输出中的每个样本都是通过对输入样本进行*加权处理*并相加得到的。递归滤波器是该方法的扩展，除了使用来自*输入*的点值外，还会利用先前计算出的*输出*值。递归滤波器不使用滤波核，而是通过一组**递归系数**来定义。这种方法将在第19章详细讨论。目前需要强调的是，所有线性滤波器都具有冲激响应，即使你没有用它来实现滤波器。要找到递归滤波器的冲激响应，只需输入一个冲激信号，观察输出结果即可。递归滤波器的冲激响应由振幅呈指数衰减的正弦波组成。从理论上讲，这使得它们的冲激响应具有*无限长*的特性。然而，振幅最终会降至系统舍入噪声以下，剩余的样本可以忽略不计。因为

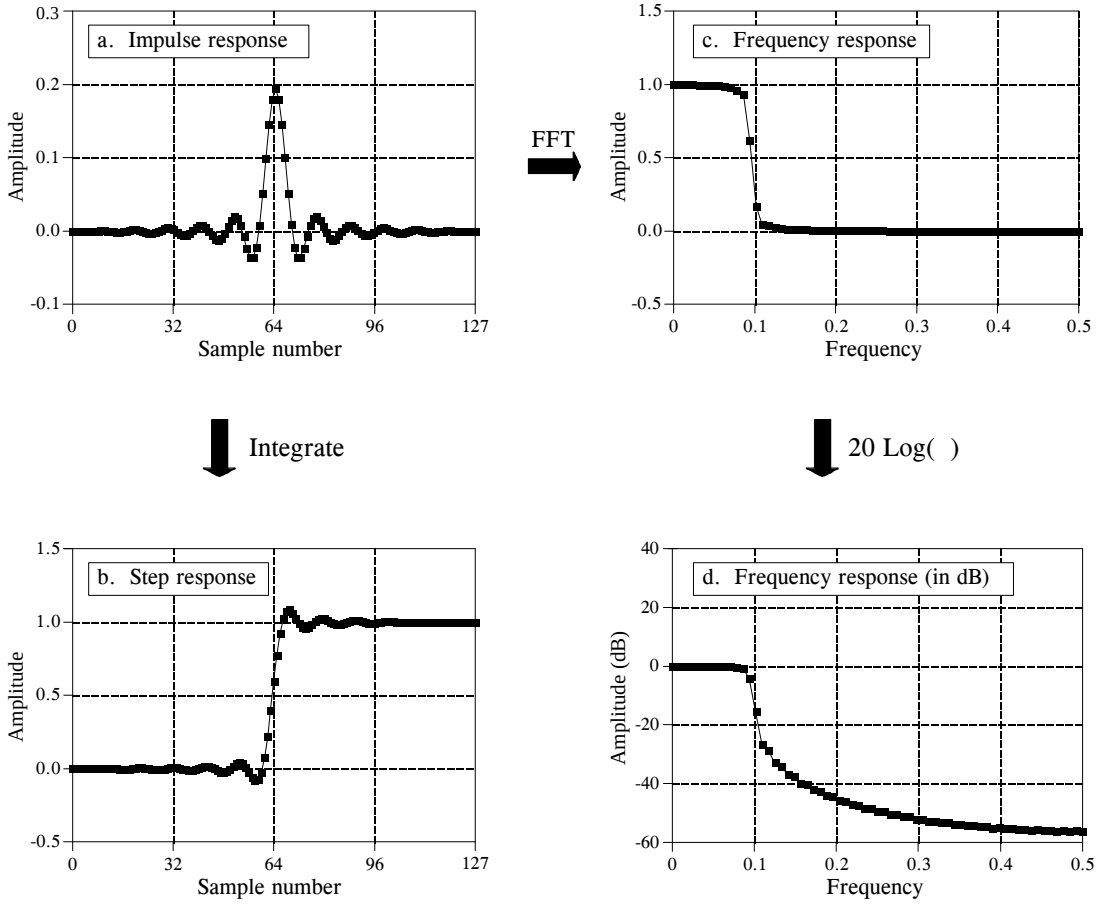

FIGURE 14-1
Filter parameters. Every linear filter has an impulse response, a step response, and a frequency response. The step response, (b), can be found by discrete integration of the impulse response, (a). The frequency response can be found from the impulse response by using the Fast Fourier Transform (FFT), and can be displayed either on a linear scale, (c), or in decibels, (d).

of this characteristic, recursive filters are also called **Infinite Impulse Response or IIR** filters. In comparison, filters carried out by convolution are called **Finite Impulse Response or FIR** filters.

As you know, the *impulse response* is the output of a system when the input is an *impulse*. In this same manner, the *step response* is the output when the input is a *step* (also called an *edge*, and an *edge response*). Since the step is the integral of the impulse, the step response is the integral of the impulse response. This provides two ways to find the step response: (1) feed a step waveform into the filter and see what comes out, or (2) integrate the impulse response. (To be mathematically correct: *integration* is used with continuous signals, while *discrete integration*, i.e., a running sum, is used with discrete signals). The frequency response can be found by taking the DFT (using the FFT algorithm) of the impulse response. This will be reviewed later in this

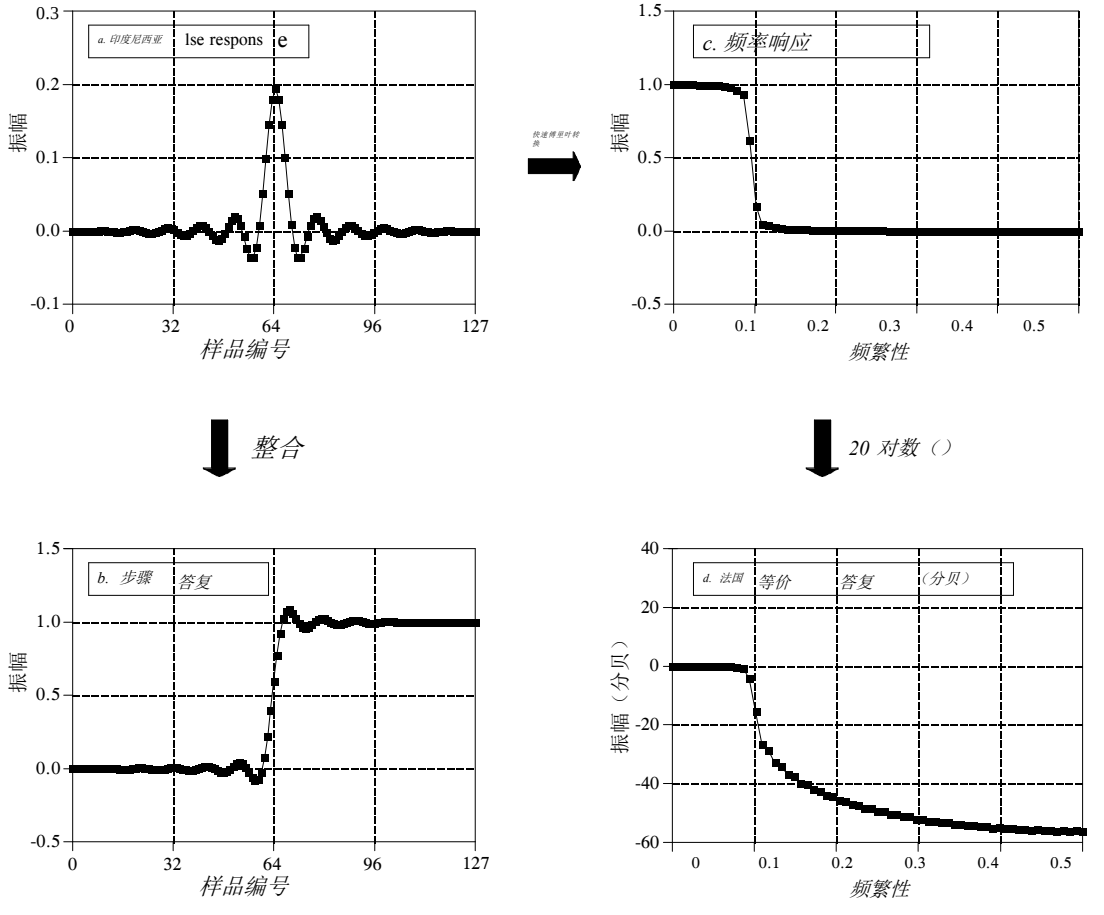

图14-1
滤波器参数。每个线性滤波器都具有冲激响应、阶跃响应和频率响应。阶跃响应(b)可通过冲激响应(a)的离散积分获得。频率响应可通过快速傅里叶变换（FFT）从冲激响应中提取，并可在线性刻度(c)或分贝(d)上显示。

由于这一特性，递归滤波器也被称为**无限脉冲响应或 IIR**滤波器。相比之下，通过卷积实现的滤波器被称为**有限脉冲响应或FIR**滤波器。

正如你所知，*冲激响应*是系统在输入为*冲激*时的输出。同理，*阶跃响应*是输入为*阶跃*（也称为*边沿*或*边沿响应*）时的输出。由于阶跃是冲激的积分，阶跃响应即为冲激响应的积分。这提供了两种求取阶跃响应的方法：(1) 将阶跃波形输入滤波器并观察输出，或 (2) 对冲激响应进行积分。（数学上严格来说：*积分*适用于连续信号，而*离散积分*（即逐次累加）适用于离散信号）。频率响应可通过对冲激响应进行DFT（使用 FFT 算法）获得。这将在本节后续内容中回顾

chapter. The frequency response can be plotted on a linear vertical axis, such as in (c), or on a logarithmic scale (decibels), as shown in (d). The linear scale is best at showing the passband ripple and roll-off, while the decibel scale is needed to show the stopband attenuation.

Don't remember decibels? Here is a quick review. A **bel** (in honor of Alexander Graham Bell) means that the power is changed by a *factor of ten*. For example, an electronic circuit that has 3 bels of amplification produces an output signal with $10 \times 10 \times 10 = 1000$ times the power of the input. A **decibel (dB)** is one-tenth of a bel. Therefore, the decibel values of: -20dB, -10dB, 0dB, 10dB & 20dB, mean the power ratios: 0.01, 0.1, 1, 10, & 100, respectively. In other words, every *ten* decibels mean that the power has changed by a factor of ten.

Here's the catch: you usually want to work with a signal's *amplitude*, not its *power*. For example, imagine an amplifier with 20dB of gain. By definition, this means that the power in the signal has increased by a factor of 100. Since amplitude is proportional to the square-root of power, the amplitude of the output is 10 times the amplitude of the input. While 20dB means a factor of 100 in power, it only means a factor of 10 in amplitude. Every *twenty* decibels mean that the amplitude has changed by a factor of ten. In equation form:

EQUATION 14-1
Definition of decibels. Decibels are a way of expressing a *ratio* between two signals. Ratios of power (P_1 & P_2) use a different equation from ratios of amplitude (A_1 & A_2).

$$\text{dB} = 10 \log_{10} \frac{P_2}{P_1}$$

$$\text{dB} = 20 \log_{10} \frac{A_2}{A_1}$$

The above equations use the base 10 logarithm; however, many computer languages only provide a function for the base *e* logarithm (the natural log, written $\log_e x$ or $\ln x$). The natural log can be use by modifying the above equations: $\text{dB} = 4.342945 \log_e(P_2/P_1)$ and $\text{dB} = 8.685890 \log_e(A_2/A_1)$.

Since decibels are a way of expressing the ratio between two signals, they are ideal for describing the gain of a system, i.e., the ratio between the output and the input signal. However, engineers also use decibels to specify the amplitude (or power) of a *single* signal, by referencing it to some standard. For example, the term: **dBV** means that the signal is being referenced to a 1 volt rms signal. Likewise, **dBm** indicates a reference signal producing 1 mW into a 600 ohms load (about 0.78 volts rms).

If you understand nothing else about decibels, remember two things: First, -3dB means that the amplitude is reduced to 0.707 (and the power is

【补充】：见笔记：分贝

章节。频率响应可采用线性纵轴（如图(c)所示）或对数刻度（分贝）（如图(d)所示）进行绘制。线性刻度最能清晰呈现通带波纹与滚降特性，而分贝刻度则适用于展示阻带衰减情况。

不记得分贝了？这里快速回顾一下。一个**贝尔**（为纪念亚历山大·格拉汉姆·贝尔）表示功率变化了*十倍*。例如，一个放大3贝尔的电子电路产生的输出信号功率是输入信号的10×10×10=1000倍。一个**分贝（dB）**是贝尔的十分之一。因此，-20dB、-10dB、0dB、10dB和20dB的分贝值分别表示功率比为：0.01、0.1、1、10和100。换句话说，每增加*十*分贝，功率就会变化十倍。

这里有个关键点：我们通常关注的是信号的*幅度*而非*功率*。举个例子，假设某个放大器的增益为20dB。根据定义，这意味着信号功率提升了100倍。由于幅度与功率的平方根成正比，输出信号的幅度是输入信号的10倍。虽然20dB表示功率提升了100倍，但幅度仅提升了10倍。每*二十*分贝就意味着幅度变化了十倍。用公式表示：

方程14-1
分贝的定义。分贝是一种表示两个信号之间*比率*的方法。功率比率（P_1和P2）使用一个
比值方程
振幅（A_1& A2）。

$$\mathrm{dB} = 10\log_{10}\frac{P_2}{P_1}$$

$$\mathrm{dB} = 20\log_{10}\frac{A_2}{A_1}$$

上述方程使用以10为底的对数；然而，许多计算机语言仅提供以*e*为底的对数函数（自然对数，记作$\log_e x$或$\ln x$）。通过修改上述方程可使用自然对数：$\mathrm{dB}=4.342945\log_e(P_2/P1)$以及$\mathrm{dB}=8.685890\log_e(A_2/A1)$。

由于分贝是表示两个信号之间比值的单位，因此非常适合用来描述系统的增益，即输出信号与输入信号的比值。不过工程师们也会用分贝来定义*单个*信号的幅度（或功率），通过将其与某个标准进行对比。例如，术语**dBV**表示该信号是相对于1伏特有效值的信号而言的。同理，**dBm**则表示参考信号在600欧姆负载下会产生1毫瓦的功率（约0.78伏特有效值）。

若你对分贝（decibel）一无所知，请牢记两点：首先，-3dB表示振幅降至0.707（功率为

therefore reduced to 0.5). Second, memorize the following conversions between decibels and *amplitude* ratios:

60dB	=	1000
40dB	=	100
20dB	=	10
0dB	=	1
-20dB	=	0.1
-40dB	=	0.01
-60dB	=	0.001

How Information is Represented in Signals

The most important part of any DSP task is understanding how *information* is contained in the signals you are working with. There are many ways that information can be contained in a signal. This is especially true if the signal is manmade. For instance, consider all of the modulation schemes that have been devised: AM, FM, single-sideband, pulse-code modulation, pulse-width modulation, etc. The list goes on and on. Fortunately, there are only two ways that are common for information to be represented in naturally occurring signals. We will call these: **information represented in the time domain**, and **information represented in the frequency domain**.

Information represented in the time domain describes when something occurs and what the amplitude of the occurrence is. For example, imagine an experiment to study the light output from the sun. The light output is measured and recorded once each second. Each sample in the signal indicates what is happening at that instant, and the level of the event. If a solar flare occurs, the signal directly provides information on the time it occurred, the duration, the development over time, etc. Each sample contains information that is interpretable without reference to any other sample. Even if you have only one sample from this signal, you still know something about what you are measuring. This is the simplest way for information to be contained in a signal.

In contrast, information represented in the frequency domain is more indirect. Many things in our universe show periodic motion. For example, a wine glass struck with a fingernail will vibrate, producing a ringing sound; the pendulum of a grandfather clock swings back and forth; stars and planets rotate on their axis and revolve around each other, and so forth. By measuring the frequency, phase, and amplitude of this periodic motion, information can often be obtained about the system producing the motion. Suppose we sample the sound produced by the ringing wine glass. The fundamental frequency and harmonics of the periodic vibration relate to the mass and elasticity of the material. A single sample, in itself, contains no information about the periodic motion, and therefore no information about the wine glass. The information is contained in the *relationship* between many points in the signal.

因此降至0.5）。其次，牢记以下换算公式
分贝与*振幅*比：

60分贝 = 1000
40分贝 = 100
20分贝 = 10
0dB = 1
-20分贝 = 0.1
-40分贝 = 0.01
-60分贝 = 0.001

信号中信息的表示方式

在数字信号处理（DSP）任务中，最关键的部分在于理解信号中蕴含的*信息*内容。信号承载信息的方式多种多样，对于人工信号而言更是如此。以调制方式为例，AM调幅、FM调频、单边带、脉冲编码调制、脉冲宽度调制等技术方案层出不穷。值得庆幸的是，在自然信号中，信息通常通过两种常见方式呈现：**时间域信息**和**频域信息**。

时域信息描述了事件发生的时间及其强度。例如，假设我们进行一项研究太阳光输出的实验，每秒测量并记录一次光输出。信号中的每个样本都反映了该时刻发生的情况及事件强度。若发生太阳耀斑，信号会直接提供其发生时间、持续时长、随时间演变等信息。每个样本都包含独立可解读的信息，无需参照其他样本即可理解。即使仅有一个样本，我们仍能了解所测量的内容。这是信号中信息最简单的呈现方式。

相比之下，频域信息的呈现方式更为间接。宇宙中许多事物都表现出周期性运动特征：比如用指甲敲击酒杯会发出清脆的叮当声；老式座钟的摆锤持续摆动；恒星行星自转公转，诸如此类。通过测量这些周期性运动的频率、相位和振幅，我们往往能获取关于运动系统的相关信息。假设我们采集了酒杯叮当声的样本，其基频和谐波与材料的密度和弹性密切相关。单个样本本身并不包含周期性运动的信息，因此也无法反映酒杯的特性。真正的信息蕴含在信号中多个点的*关系*之中。

This brings us to the importance of the step and frequency responses. The *step response* describes how information represented in the *time domain* is being modified by the system. In contrast, the *frequency response* shows how information represented in the *frequency domain* is being changed. This distinction is absolutely critical in filter design because it is not possible to optimize a filter for both applications. Good performance in the time domain results in poor performance in the frequency domain, and vice versa. If you are designing a filter to remove noise from an EKG signal (information represented in the time domain), the step response is the important parameter, and the frequency response is of little concern. If your task is to design a digital filter for a hearing aid (with the information in the frequency domain), the frequency response is all important, while the step response doesn't matter. Now let's look at what makes a filter optimal for time domain or frequency domain applications.

Time Domain Parameters

It may not be obvious why the step response is of such concern in time domain filters. You may be wondering why the impulse response isn't the important parameter. The answer lies in the way that the human mind understands and processes information. Remember that the step, impulse and frequency responses all contain identical information, just in different arrangements. The step response is useful in time domain analysis because it matches the way humans view the information contained in the signals.

For example, suppose you are given a signal of some unknown origin and asked to analyze it. The first thing you will do is divide the signal into regions of similar characteristics. You can't stop from doing this; your mind will do it automatically. Some of the regions may be smooth; others may have large amplitude peaks; others may be noisy. This segmentation is accomplished by identifying the points that separate the regions. This is where the step function comes in. The step function is the purest way of representing a division between two dissimilar regions. It can mark when an event starts, or when an event ends. It tells you that whatever is on the *left* is somehow different from whatever is on the *right*. This is how the human mind views time domain information: a group of step functions dividing the information into regions of similar characteristics. The step response, in turn, is important because it describes how the dividing lines are being modified by the filter.

The step response parameters that are important in filter design are shown in Fig. 14-2. To distinguish events in a signal, the duration of the step response must be shorter than the spacing of the events. This dictates that the step response should be as *fast* (the DSP jargon) as possible. This is shown in Figs. (a) & (b). The most common way to specify the **risetime** (more jargon) is to quote the number of samples between the 10% and 90% amplitude levels. Why isn't a very fast risetime always possible? There are many reasons, noise reduction, inherent limitations of the data acquisition system, avoiding aliasing, etc.

这就引出了阶跃响应和频率响应的重要性。*阶跃响应*描述了系统如何改变*时间域*中的信息，而*频率响应*则展示了*频率域*中信息的变化规律。这种区分在滤波器设计中至关重要，因为无法同时优化两种应用场景。时间域表现优异的滤波器在频率域可能表现欠佳，反之亦然。例如，若要设计用于去除心电图信号噪声（时间域信息）的滤波器，阶跃响应才是关键参数，频率响应则无关紧要；而若要为助听器设计数字滤波器（涉及频率域信息），频率响应至关重要，此时阶跃响应则无足轻重。接下来，让我们探讨为何滤波器在时间域或频率域应用中会呈现不同特性。

时域参数

阶跃响应为何在时域滤波器中备受关注，这或许并不明显。你可能会疑惑，为什么冲激响应不是更重要的参数？答案在于人类大脑理解和处理信息的方式。请记住，阶跃响应、冲激响应和频率响应都包含相同的信息，只是排列方式不同。阶跃响应在时域分析中具有实用价值，因为它与人类感知信号信息的方式相吻合。

举个例子，假设你接收到一个来源不明的信号并需要进行分析。首先你会将信号划分为特征相似的区域。这种划分是无法停止的，大脑会自动完成这个过程。有些区域可能平滑无波，有些可能呈现大幅波动的峰值，还有些区域可能充满噪声。这种分割是通过识别分隔不同区域的分界点实现的，这正是阶跃函数的用武之地。阶跃函数是表征两个不同区域分界最纯粹的方式，它能标记事件的起始或终止时刻，表明*左侧*与*右侧*的信息存在本质差异。这正是人类大脑处理时域信息的方式：通过一组阶跃函数将信息划分为特征相似的区域。而阶跃响应之所以重要，是因为它描述了滤波器如何对这些分界线进行修正。

图14-2展示了滤波器设计中重要的阶跃响应参数。要区分信号中的不同事件，阶跃响应的持续时间必须短于事件间隔。这就要求阶跃响应应尽可能*快速*（这是数字信号处理领域的专业术语），如图(a)和(b)所示。指定**上升时间**（更专业的术语）最常见的方式是引用10%与90%幅度水平之间的采样点数。为何无法实现极快的上升时间？原因有很多：噪声抑制、数据采集系统的固有局限、避免混叠效应等。

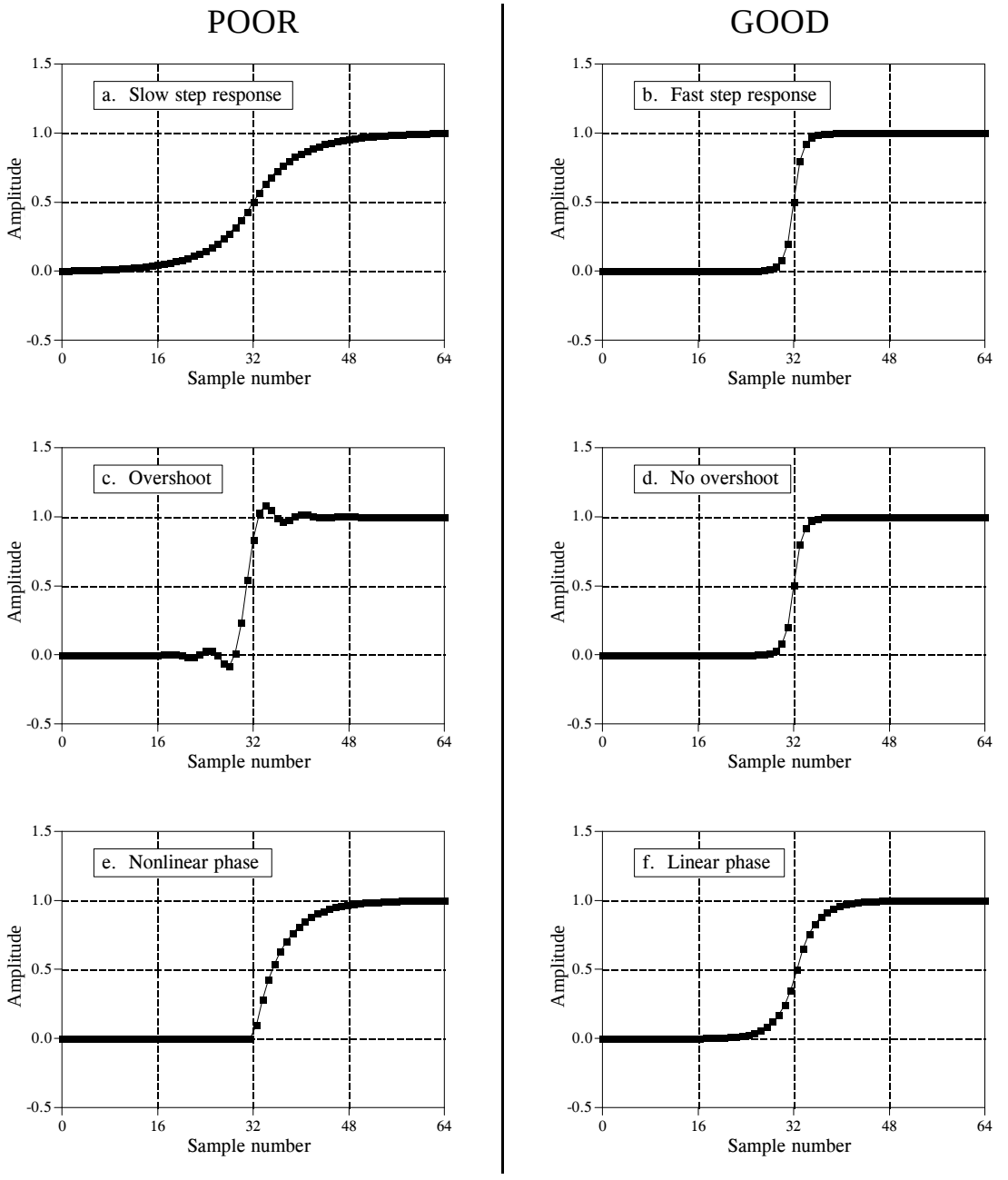

FIGURE 14-2
Parameters for evaluating *time domain* performance. The step response is used to measure how well a filter performs in the time domain. Three parameters are important: (1) transition speed (risetime), shown in (a) and (b), (2) overshoot, shown in (c) and (d), and (3) phase linearity (symmetry between the top and bottom halves of the step), shown in (e) and (f).

Figures (c) and (d) shows the next parameter that is important: **overshoot** in the step response. Overshoot must generally be eliminated because it changes the amplitude of samples in the signal; this is a basic distortion of the information contained in the time domain. This can be summed up in

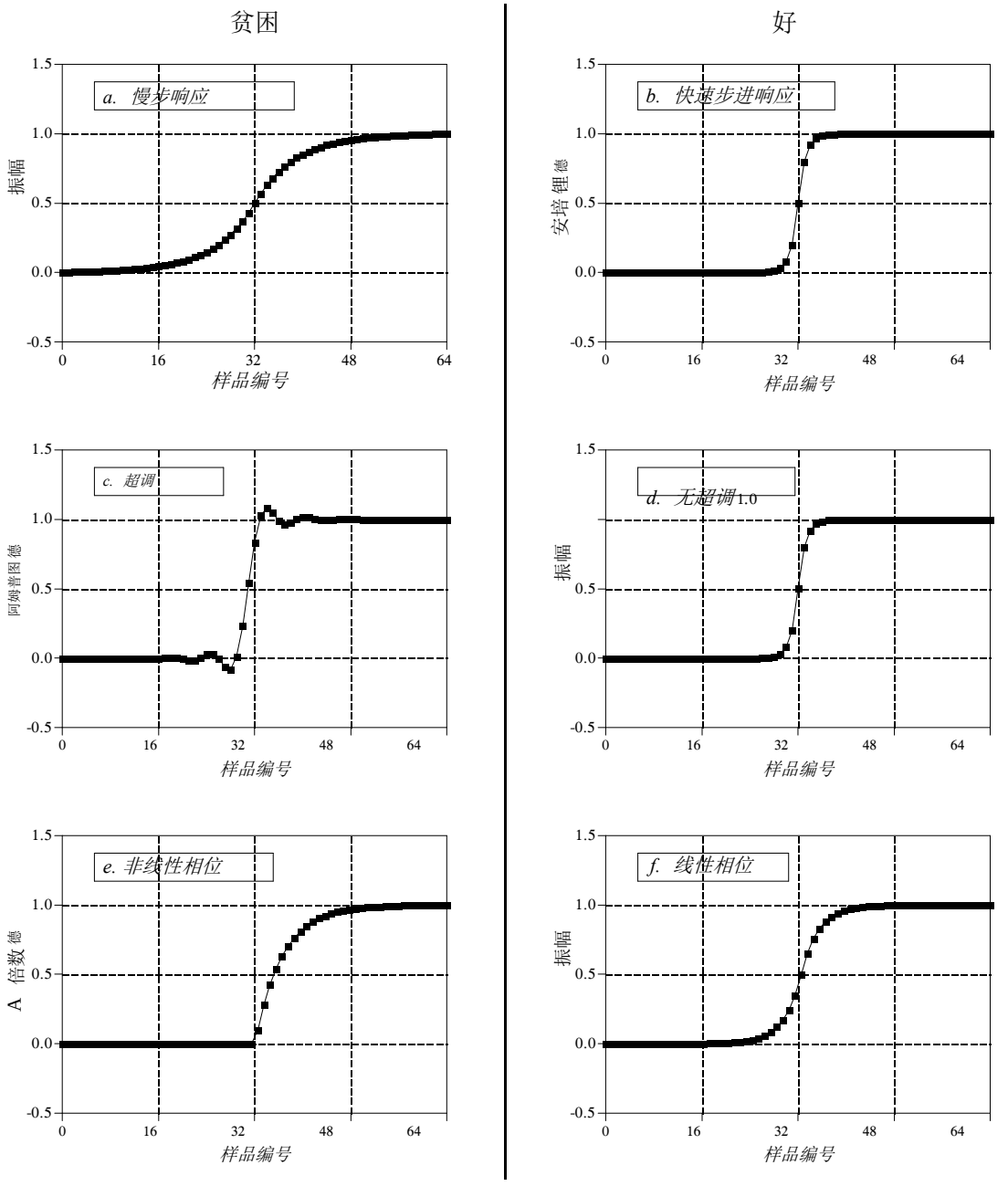

图14-2
评估*时域*性能的参数。阶跃响应用于衡量滤波器在时域中的性能表现。三个参数至关重要：(1)过渡速度（上升时间），如(a)和(b)所示；(2)超调量，如(c)和(d)所示；(3)相位线性度（阶跃信号上下半部分的对称性），如(e)和(f)所示。

图(c)和(d)展示了另一个关键参数：阶跃响应中的**超调量**。通常需要消除超调量，因为它会改变信号中样本的幅度，这是时域信息的基本失真。这可以概括为

one question: Is the overshoot you observe in a signal coming from the thing you are trying to measure, or from the filter you have used?

Finally, it is often desired that the upper half of the step response be symmetrical with the lower half, as illustrated in (e) and (f). This symmetry is needed to make the *rising edges* look the same as the *falling edges*. This symmetry is called **linear phase**, because the frequency response has a phase that is a straight line (discussed in Chapter 19). Make sure you understand these three parameters; they are the key to evaluating time domain filters.

Frequency Domain Parameters

Figure 14-3 shows the four basic frequency responses. The purpose of these filters is to allow some frequencies to pass unaltered, while completely blocking other frequencies. The **passband** refers to those frequencies that are passed, while the **stopband** contains those frequencies that are blocked. The **transition band** is between. A **fast roll-off** means that the transition band is very narrow. The division between the passband and transition band is called the **cutoff frequency.** In analog filter design, the cutoff frequency is usually defined to be where the amplitude is reduced to 0.707 (i.e., -3dB). Digital filters are less standardized, and it is common to see 99%, 90%, 70.7%, and 50% amplitude levels defined to be the cutoff frequency.

Figure 14-4 shows three parameters that measure how well a filter performs in the frequency domain. To separate closely spaced frequencies, the filter must have a **fast roll-off**, as illustrated in (a) and (b). For the passband frequencies to move through the filter unaltered, there must be no **passband ripple**, as shown in (c) and (d). Lastly, to adequately block the stopband frequencies, it is necessary to have good **stopband attenuation**, displayed in (e) and (f).

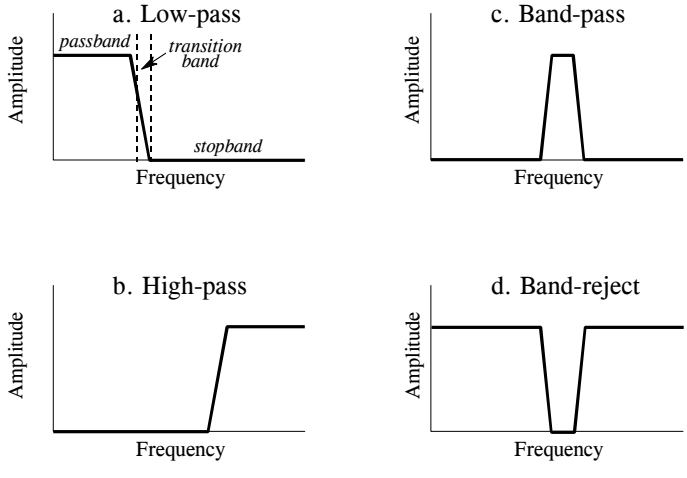

FIGURE 14-3
The four common frequency responses. Frequency domain filters are generally used to pass certain frequencies (the *passband*), while blocking others (the *stopband*). Four responses are the most common: low-pass, high-pass, band-pass, and band-reject.

问题：您观察到的信号过冲现象，是源自待测对象本身，还是源于所使用的滤波器？

最后，人们通常希望阶跃响应的上半部分与下半部分保持对称，如(e)和(f)所示。这种对称性是为了让*上升沿*与*下降沿*看起来一致。这种对称性被称为**线性相位**，因为频率响应的相位呈现直线特征（详见第19章）。请务必理解这三个参数，它们是评估时域滤波器的关键要素。

频域参数

图14-3展示了四种基本频率响应。这类滤波器的作用是让部分频率保持原样通过，同时完全阻断其他频率。其中，**通带**指被允许通过的频率范围，**阻带**则包含被完全阻断的频率，而**过渡带**介于两者之间。当过渡带非常狭窄时，我们称之为**快速滚降**。通带与过渡带的分界点称为**截止频率**。在模拟滤波器设计中，截止频率通常定义为振幅降至0.707（即-3dB）的频率。数字滤波器则相对不那么标准化，常见的截止频率定义包括99%、90%、70.7%和50%的振幅水平。

图14-4展示了衡量滤波器在频域性能的三个关键参数。要有效分离相邻频率，滤波器必须具备**快速滚降特性**，如图(a)和(b)所示。为确保通带频率在通过滤波器时保持原样，必须消除**通带纹波**，如图(c)和(d)所示。最后，要充分阻断阻带频率，就需要具备良好的**阻带衰减**，如图(e)和(f)所示。

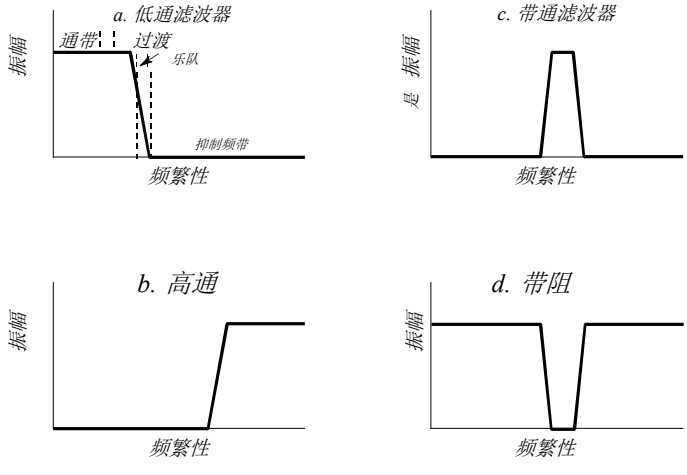

图14-3
四种常见的频率响应。频域滤波器通常用于通过特定频率（*通带*），同时阻断其他频率（*阻带*）。最常见的四种响应是：低通、高通、带通和带阻。

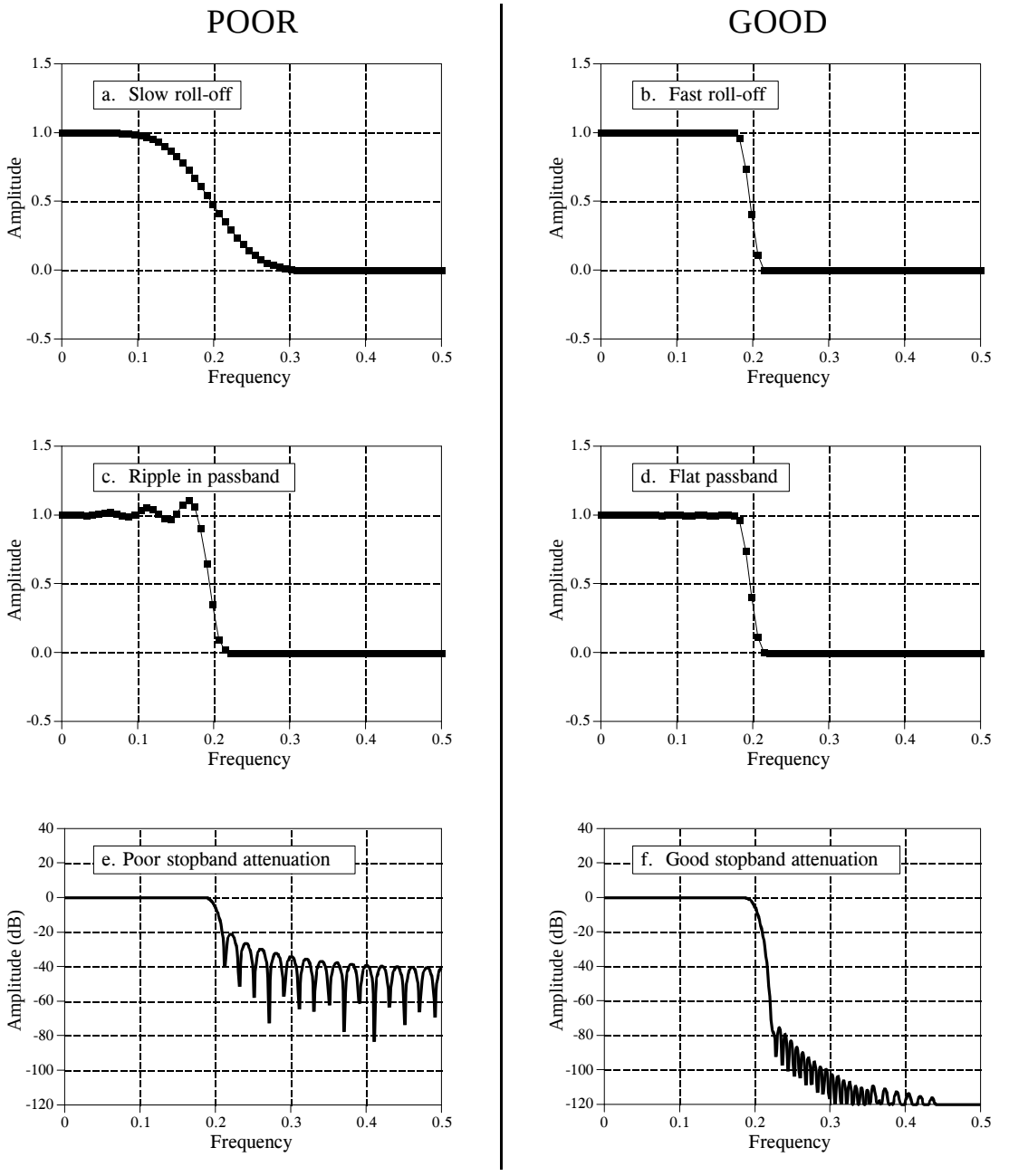

FIGURE 14-4
Parameters for evaluating *frequency domain* performance. The frequency responses shown are for low-pass filters. Three parameters are important: (1) roll-off sharpness, shown in (a) and (b), (2) passband ripple, shown in (c) and (d), and (3) stopband attenuation, shown in (e) and (f).

Why is there nothing about the *phase* in these parameters? First, the phase isn't important in most frequency domain applications. For example, the phase of an audio signal is almost completely random, and contains little useful information. Second, if the phase is important, it is very easy to make digital

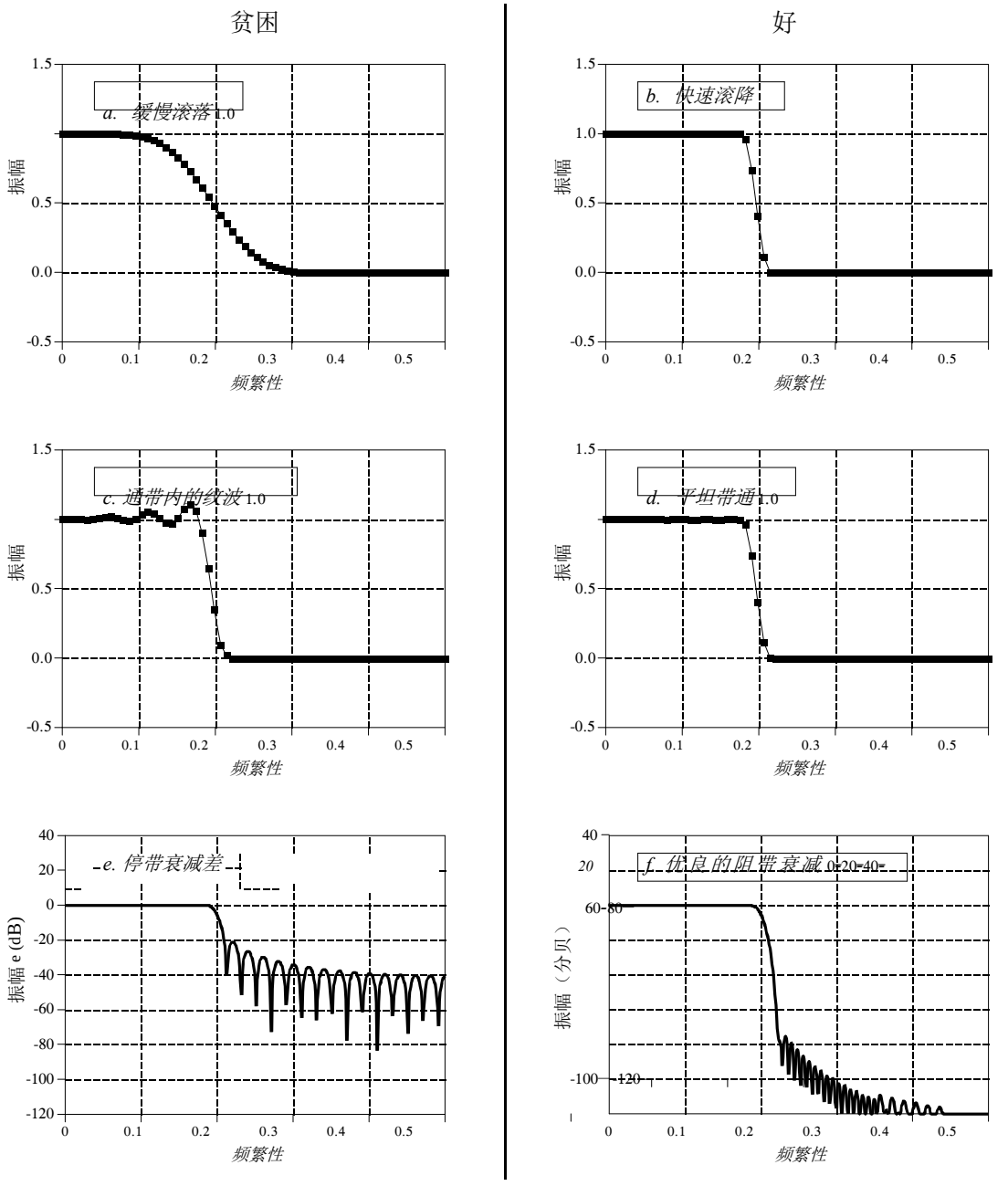

图14-4

评估频率*域*性能的参数。所示频率响应为低通滤波器。三个参数很重要：(1)滚降陡度，如(a)和(b)所示，(2)通带纹波，如(c)和(d)所示，以及(3)阻带衰减，如(e)和(f)所示。

为什么这些参数中没有关于*相位*的内容？首先，相位在大多数频域应用中并不重要。例如，音频信号的相位几乎完全是随机的，且包含的有用信息很少。其次，如果相位很重要，那么很容易通过数字方式实现。

filters with a *perfect* phase response, i.e., all frequencies pass through the filter with a zero phase shift (also discussed in Chapter 19). In comparison, analog filters are ghastly in this respect.

Previous chapters have described how the DFT converts a system's impulse response into its frequency response. Here is a brief review. The quickest way to calculate the DFT is by means of the FFT algorithm presented in Chapter 12. Starting with a filter kernel N samples long, the FFT calculates the frequency spectrum consisting of an N point *real part* and an N point *imaginary part*. Only samples 0 to $N/2$ of the FFT's real and imaginary parts contain useful information; the remaining points are duplicates (negative frequencies) and can be ignored. Since the real and imaginary parts are difficult for humans to understand, they are usually converted into polar notation as described in Chapter 8. This provides the magnitude and phase signals, each running from sample 0 to sample $N/2$ (i.e., $N/2+1$ samples in each signal). For example, an impulse response of 256 points will result in a frequency response running from point 0 to 128. Sample 0 represents DC, i.e., zero frequency. Sample 128 represents one-half of the sampling rate. Remember, no frequencies higher than one-half of the sampling rate can appear in sampled data.

The number of samples used to represent the impulse response can be arbitrarily large. For instance, suppose you want to find the frequency response of a filter kernel that consists of 80 points. Since the FFT only works with signals that are a power of two, you need to add 48 zeros to the signal to bring it to a length of 128 samples. This *padding with zeros* does not change the impulse response. To understand why this is so, think about what happens to these added zeros when the input signal is convolved with the system's impulse response. The added zeros simply *vanish* in the convolution, and do not affect the outcome.

Taking this a step further, you could add *many* zeros to the impulse response to make it, say, 256, 512, or 1024 points long. The important idea is that longer impulse responses result in a closer spacing of the data points in the frequency response. That is, there are more samples spread between DC and one-half of the sampling rate. Taking this to the extreme, if the impulse response is padded with an *infinite* number of zeros, the data points in the frequency response are infinitesimally close together, i.e., a continuous line. In other words, the frequency response of a filter is really a *continuous* signal between DC and one-half of the sampling rate. The output of the DFT is a *sampling* of this continuous line. What length of impulse response should you use when calculating a filter's frequency response? As a first thought, try $N=1024$, but don't be afraid to change it if needed (such as insufficient resolution or excessive computation time).

Keep in mind that the "good" and "bad" parameters discussed in this chapter are only generalizations. Many signals don't fall neatly into categories. For example, consider an EKG signal contaminated with 60 hertz interference. The information is encoded in the *time domain*, but the interference is best dealt with in the *frequency domain*. The best design for this application is

具有*完美*相位响应的滤波器，即所有频率通过时相位偏移为零（详见第19章）。相比之下，模拟滤波器在这方面表现极差。

前几章已经描述了DFT如何将系统的脉冲响应转换为其频率响应。这里做一个简要回顾。计算DFT最快的方法是通过第12章介绍的 FFT 算法。从一个长度为N个样本的滤波器核开始， FFT 计算由N个点*实部*和N个点*虚部*组成的频率谱。只有 FFT 实部和虚部的第0到$N/2$个样本包含有用信息；其余点是重复（负频率）可以忽略。由于实部和虚部对人类来说难以理解，它们通常被转换为极坐标表示法，如第8章所述。这提供了幅度和相位信号，每个信号从样本0到样本$N/2$（即每个信号中$N/2+1$个样本）。例如，一个256点的脉冲响应将产生从点0到128的频率响应。样本0代表直流，即零频率。样本128代表采样率的一半。需注意，采样数据中不得出现高于采样率一半的频率。

用于表征脉冲响应的样本数量可以任意增大。举个例子，假设你想计算一个由80个点组成的滤波器核的频率响应。由于该 FFT 仅支持二的幂次方信号，你需要在信号中添加48个零点使其达到128个样本长度。这种*用零点填充*的操作并不会改变脉冲响应。要理解其中原理，可以想象当输入信号与系统脉冲响应进行卷积运算时，这些新增的零点会发生什么变化。这些零点在卷积运算中会直接*消失*，不会对输出结果产生任何影响。

更进一步说，你可以向冲激响应中添加*大量*零值，使其长度达到256、512或1024个点。关键在于，更长的冲激响应会使频率响应中的数据点间距更紧密。也就是说，在直流电与采样率一半之间分布的采样点会更多。若将这种趋势推向极端，当冲激响应被*无限*个零值填充时，频率响应中的数据点会无限接近，形成一条连续的曲线。换言之，滤波器的频率响应本质上是从直流电到采样率一半之间的*连续*信号。而离散傅里叶变换的输出正是这条连续曲线的*采样*结果。在计算滤波器频率响应时，应该采用多长的冲激响应？初步建议可尝试N=1024，但若遇到分辨率不足或计算耗时过长等情况，也无需拘泥于此。

需要特别注意的是，本章讨论的“优”与“劣”参数仅是概括性概念。许多信号无法简单归类。例如，考虑一个受到60赫兹干扰的ECG信号：其信息编码在*时域*，但干扰处理最适宜采用*频域*方法。针对此类应用场景，最佳设计方案是

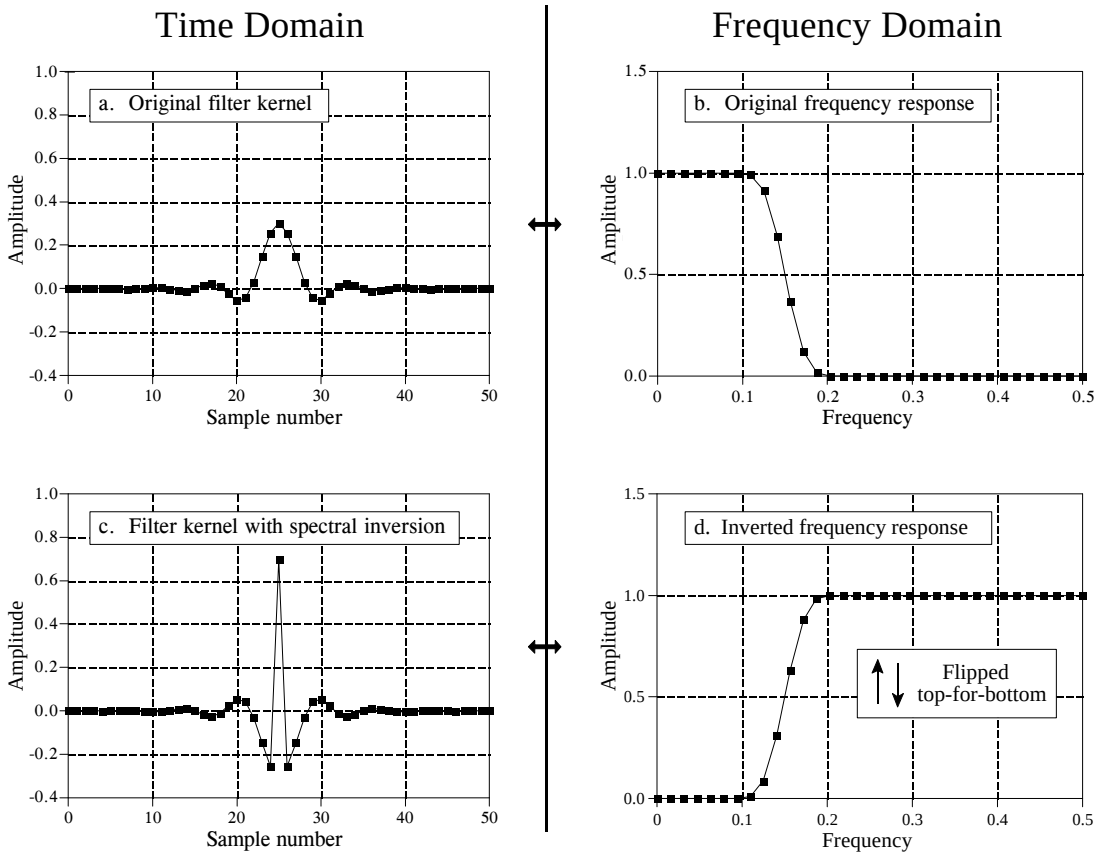

FIGURE 14-5
Example of spectral inversion. The low-pass filter kernel in (a) has the frequency response shown in (b). A high-pass filter kernel, (c), is formed by changing the sign of each sample in (a), and adding one to the sample at the center of symmetry. This action in the time domain *inverts* the frequency spectrum (i.e., flips it top-for-bottom), as shown by the high-pass frequency response in (d).

bound to have trade-offs, and might go against the conventional wisdom of this chapter. Remember the number one rule of education: *A paragraph in a book doesn't give you a license to stop thinking*.

High-Pass, Band-Pass and Band-Reject Filters

High-pass, band-pass and band-reject filters are designed by starting with a low-pass filter, and then converting it into the desired response. For this reason, most discussions on filter design only give examples of low-pass filters. There are two methods for the low-pass to high-pass conversion: **spectral inversion** and **spectral reversal**. Both are equally useful.

An example of *spectral inversion* is shown in 14-5. Figure (a) shows a low-pass filter kernel called a windowed-sinc (the topic of Chapter 16). This filter kernel is 51 points in length, although many of samples have a value so small that they appear to be zero in this graph. The corresponding

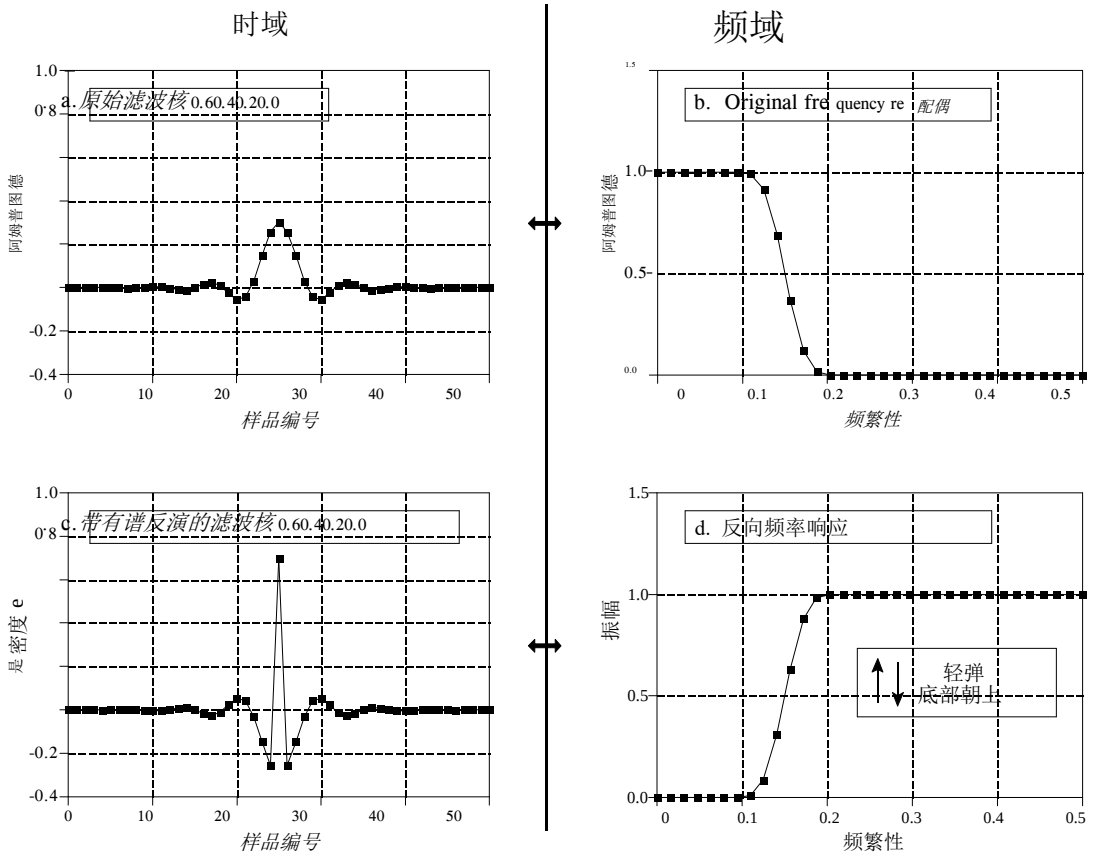

图14-5
频谱反演示例。图(a)中的低通滤波核具有图(b)所示的频率响应。高通滤波核(c)是通过改变图(a)中每个样本的符号，并在对称中心处添加一个样本形成的。时域中的这一操作*将频谱倒置*（即上下翻转），如图(d)中的高通频率响应所示。

必然会有权衡，而且可能与本章的常识相悖。记住教育的第一条规则：*书中的一个段落不会给你一个停止思考的许可证*。

高通、带通和带阻滤波器

高通、带通和带阻滤波器的设计都是从低通滤波器开始，然后将其转换为所需的响应。因此，大多数关于滤波器设计的讨论只给出低通滤波器的例子。从低通到高通的转换有两种方法：**频谱反转**和**频谱反转**。这两种方法同样有用。

*频谱反演*的一个实例展示在14-5中。图(a)展示了一个名为窗函数Sinc的低通滤波器核（详见第16章）。该滤波器核长度为51个采样点，尽管许多采样点的数值极小，在此图中显示为零。对应的

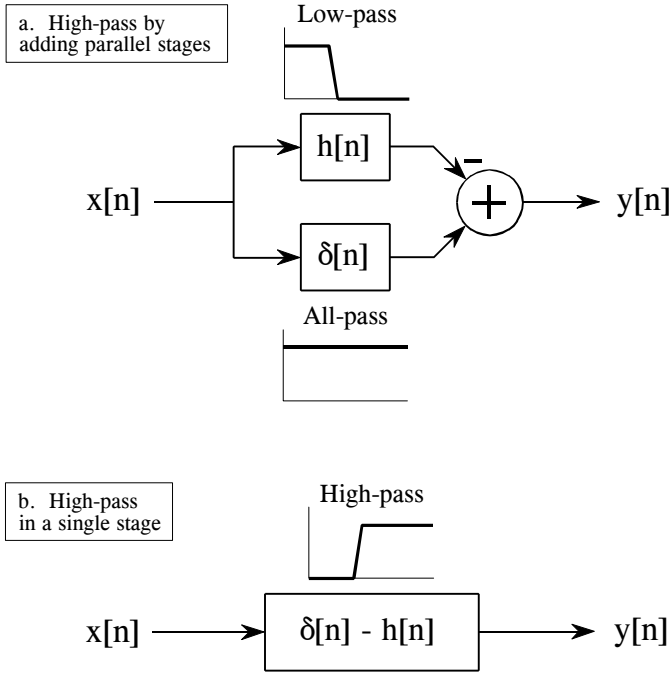

FIGURE 14-6
Block diagram of spectral inversion. In (a), the input signal, $x[n]$, is applied to two systems in parallel, having impulse responses of $h[n]$ and $\delta[n]$. As shown in (b), the combined system has an impulse response of $\delta[n] - h[n]$. This means that the frequency response of the combined system is the *inversion* of the frequency response of $h[n]$.

frequency response is shown in (b), found by adding 13 zeros to the filter kernel and taking a 64 point FFT. Two things must be done to change the low-pass filter kernel into a high-pass filter kernel. First, change the sign of each sample in the filter kernel. Second, add *one* to the sample at the center of symmetry. This results in the high-pass filter kernel shown in (c), with the frequency response shown in (d). Spectral inversion *flips* the frequency response *top-for-bottom*, changing the passbands into stopbands, and the stopbands into passbands. In other words, it changes a filter from low-pass to high-pass, high-pass to low-pass, band-pass to band-reject, or band-reject to band-pass.

Figure 14-6 shows why this two step modification to the time domain results in an inverted frequency spectrum. In (a), the input signal, $x[n]$, is applied to two systems in parallel. One of these systems is a low-pass filter, with an impulse response given by $h[n]$. The other system does *nothing* to the signal, and therefore has an impulse response that is a delta function, $\delta[n]$. The overall output, $y[n]$, is equal to the output of the all-pass system *minus* the output of the low-pass system. Since the low frequency components are subtracted from the original signal, only the high frequency components appear in the output. Thus, a high-pass filter is formed.

This could be performed as a two step operation in a computer program: run the signal through a low-pass filter, and then subtract the filtered signal from the original. However, the entire operation can be performed in a signal stage by combining the two filter kernels. As described in Chapter

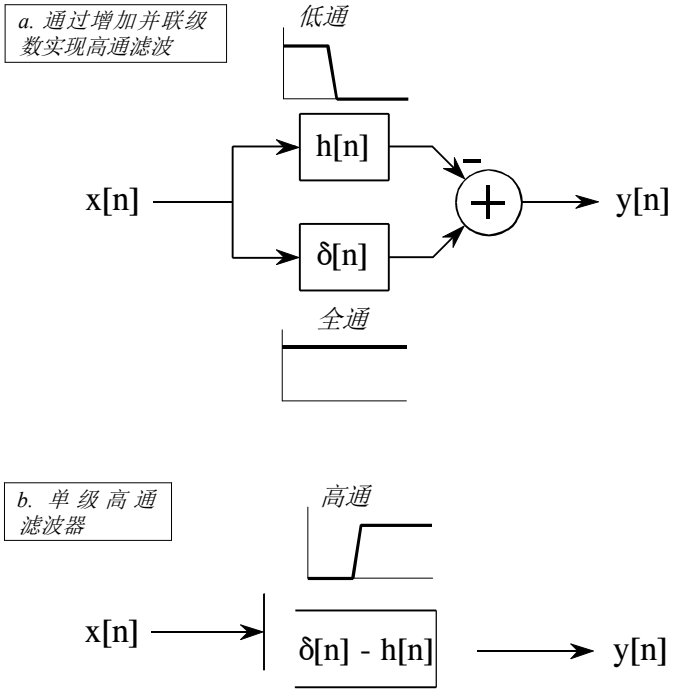

图14-6
频谱反演的框图。在(a)中，输入信号*x*[*n*]被并行施加到两个系统，其脉冲响应分别为*h*[*n*]和δ[*n*]。如(b)所示，组合系统的脉冲响应为δ[*n*] -*h*[*n*]。这意味着组合系统的频率响应是*h*[*n*]频率响应的*反演*。

频率响应如(b)所示，该结果是通过在滤波器核中添加13个零点并进行64点FFT 获得的。要将低通滤波器核转换为高通滤波器核，需完成两项操作：首先改变滤波器核中每个样本的符号，其次在对称中心处的样本值上加*1*。经过这些操作后，得到的高通滤波器核如(c)所示，其频率响应如(d)所示。频谱反转*将*频率响应*上下颠倒*，使通带变为阻带，阻带变为通带。换言之，该操作可将滤波器从低通转换为高通，从高通转换为低通，从带通转换为带阻，或从带阻转换为带通。

图14-6展示了时域的两步修改为何会导致频谱倒置。在(a)中，输入信号*x*[*n*]被并行施加到两个系统上。其中一个系统是低通滤波器，其冲激响应由*h*[*n*]给出。另一个系统对信号*不做任何处理*，因此其冲激响应为δ函数δ[*n*]。总输出*y*[*n*]等于全通系统的输出*减去*低通系统的输出。由于低频分量从原始信号中被扣除，输出中仅保留高频分量。因此，形成了一个高通滤波器。

该操作可通过计算机程序分两步完成：先将信号通过低通滤波器处理，再从原始信号中减去滤波后的信号。不过，整个操作也可通过将两个滤波核合并来在信号处理阶段完成。如第X章所述

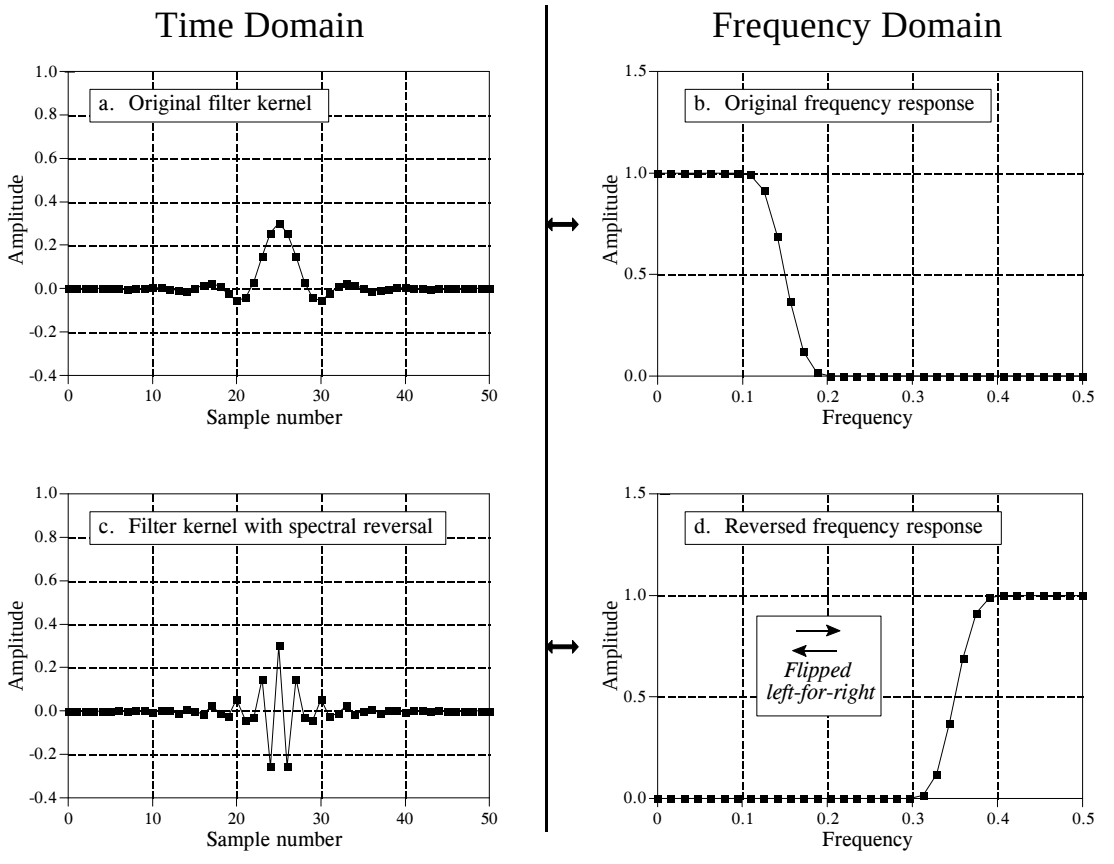

FIGURE 14-7
Example of spectral reversal. The low-pass filter kernel in (a) has the frequency response shown in (b). A high-pass filter kernel, (c), is formed by changing the sign of every other sample in (a). This action in the time domain results in the frequency domain being flipped *left-for-right*, resulting in the high-pass frequency response shown in (d).

7, parallel systems with added outputs can be combined into a single stage by adding their impulse responses. As shown in (b), the filter kernel for the high-pass filter is given by: $\delta[n] - h[n]$. That is, change the sign of all the samples, and then add one to the sample at the center of symmetry.

For this technique to work, the low-frequency components exiting the low-pass filter must have the same phase as the low-frequency components exiting the all-pass system. Otherwise a complete subtraction cannot take place. This places two restrictions on the method: (1) the original filter kernel must have left-right symmetry (i.e., a zero or linear phase), and (2) the impulse must be added at the center of symmetry.

The second method for low-pass to high-pass conversion, *spectral reversal*, is illustrated in Fig. 14-7. Just as before, the low-pass filter kernel in (a) corresponds to the frequency response in (b). The high-pass filter kernel, (c), is formed by *changing the sign of every other sample* in (a). As shown in (d), this flips the frequency domain *left-for-right*: 0 becomes 0.5 and 0.5

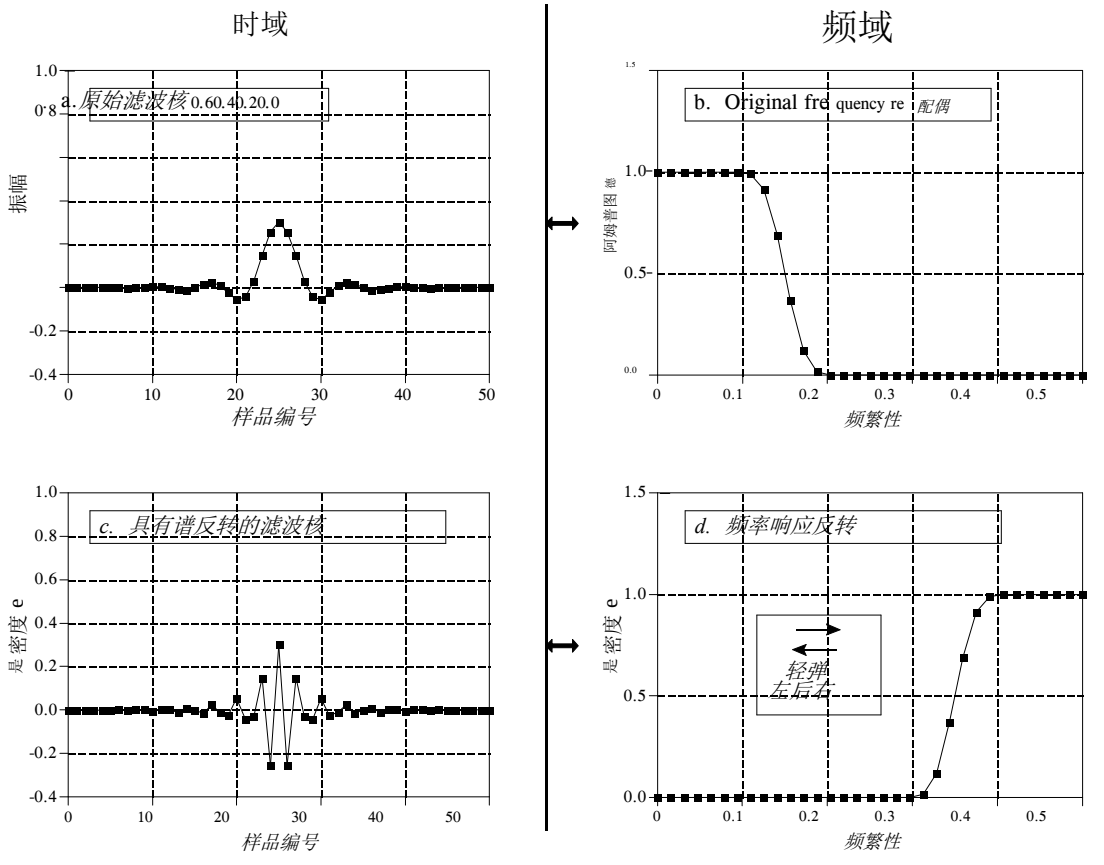

图14-7
频谱反转示例。图(a)中的低通滤波核的频率响应如图(b)所示。高通滤波核(c)是通过改变图(a)中每隔一个样本的符号形成的。时域中的这一操作导致频域被*从左到右*翻转，从而得到图(d)所示的高通频率响应。

7、具有附加输出的并行系统可以通过叠加其脉冲响应合并为单级系统。如(b)所示，高通滤波器的滤波核由$\delta[n] - h[n]$给出。即改变所有样本的符号，然后在对称中心的样本上加一。

为使该技术有效，低通滤波器输出的低频分量必须与全通系统输出的低频分量具有相同相位，否则无法实现完全抵消。该方法存在两项限制条件：(1) 原始滤波核必须具有左右对称性（即零相位或线性相位），(2) 脉冲必须施加于对称中心。

低通到高通转换的第二种方法，*频谱反转*，如图14-7所示。与之前一样，(a)中的低通滤波器核对应于(b)中的频率响应。高通滤波器核(c)是通过*改变(a)中每隔一个样本的符号*形成的。如(d)所示，这将频域*从左到右*反转：0变为0.5,0.5变为0。

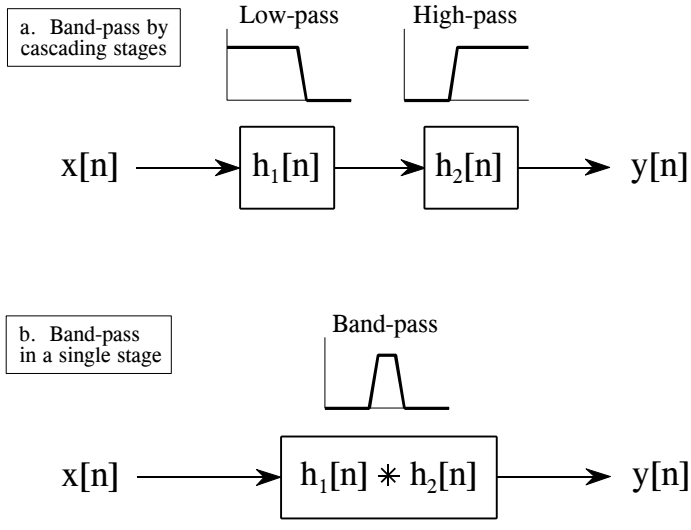

FIGURE 14-8
Designing a band-pass filter. As shown in (a), a band-pass filter can be formed by cascading a low-pass filter and a high-pass filter. This can be reduced to a single stage, shown in (b). The filter kernel of the single stage is equal to the *convolution* of the low-pass and high-pass filter kernels.

becomes 0. The cutoff frequency of the example low-pass filter is 0.15, resulting in the cutoff frequency of the high-pass filter being 0.35.

Changing the sign of every other sample is equivalent to multiplying the filter kernel by a sinusoid with a frequency of 0.5. As discussed in Chapter 10, this has the effect of *shifting* the frequency domain by 0.5. Look at (b) and imagine the negative frequencies between -0.5 and 0 that are of mirror image of the frequencies between 0 and 0.5. The frequencies that appear in (d) are the negative frequencies from (b) shifted by 0.5.

Lastly, Figs. 14-8 and 14-9 show how low-pass and high-pass filter kernels can be combined to form band-pass and band-reject filters. In short, *adding* the filter kernels produces a *band-reject* filter, while *convolving* the filter kernels produces a *band-pass* filter. These are based on the way cascaded and parallel systems are be combined, as discussed in Chapter 7. Multiple combination of these techniques can also be used. For instance, a band-pass filter can be designed by adding the two filter kernels to form a stop-pass filter, and then use *spectral inversion* or *spectral reversal* as previously described. All these techniques work very well with few surprises.

Filter Classification

Table 14-1 summarizes how digital filters are classified by their *use* and by their *implementation*. The use of a digital filter can be broken into three categories: *time domain*, *frequency domain* and *custom*. As previously described, time domain filters are used when the information is encoded in the shape of the signal's waveform. Time domain filtering is used for such actions as: smoothing, DC removal, waveform shaping, etc. In contrast, frequency domain filters are used when the information is contained in the

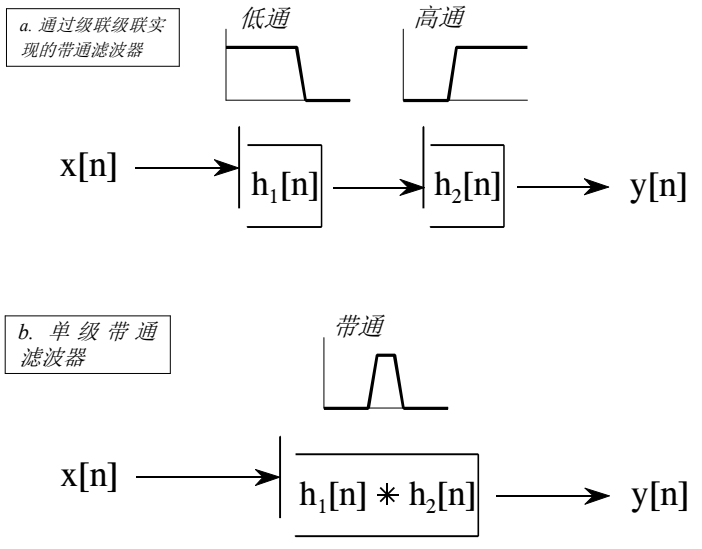

图14-8
设计一个带通滤波器。如(a)所示，带通滤波器可以通过级联低通滤波器和高通滤波器来形成。这可以简化为一个单级，如(b)所示。单级滤波器的核等于低通和高通滤波器核的*卷积*。

示例低通滤波器的截止频率为0.15，由此得出高通滤波器的截止频率为0.35。

改变每隔一个样本的符号相当于将滤波核乘以频率为0.5的正弦波。如第10章所述，这会产生将频域*移动*0.5的效果。观察(b)并想象-0.5到0之间的负频率是0到0.5之间频率的镜像。出现在(d)中的频率是(b)中负频率经0.5位移后的结果。

最后，图14-8和14-9展示了如何将低通滤波器和高通滤波器核组合形成带通和带阻滤波器。简而言之，*叠加*滤波器核会产生*带阻*滤波器，而*卷积*滤波器核则生成*带通*滤波器。这些原理基于第七章讨论的级联与并行系统组合方式。多种技术组合也可灵活运用：例如，通过叠加两个滤波器核形成带阻滤波器后，再采用前文所述的*频谱反转*或*频谱反演*技术即可设计出带通滤波器。所有这些技术都能以简单方式实现高效运作，且效果稳定可靠。

过滤分类

表14-1总结了数字滤波器如何根据其*用途*和*实现方式*进行分类。数字滤波器的用途可分为三类：*时域*、*频域*和*定制*。如前所述，当信息以信号波形形式编码时，会使用时域滤波器。时域滤波常用于平滑处理、直流消除、波形整形等操作。相比之下，当信息以

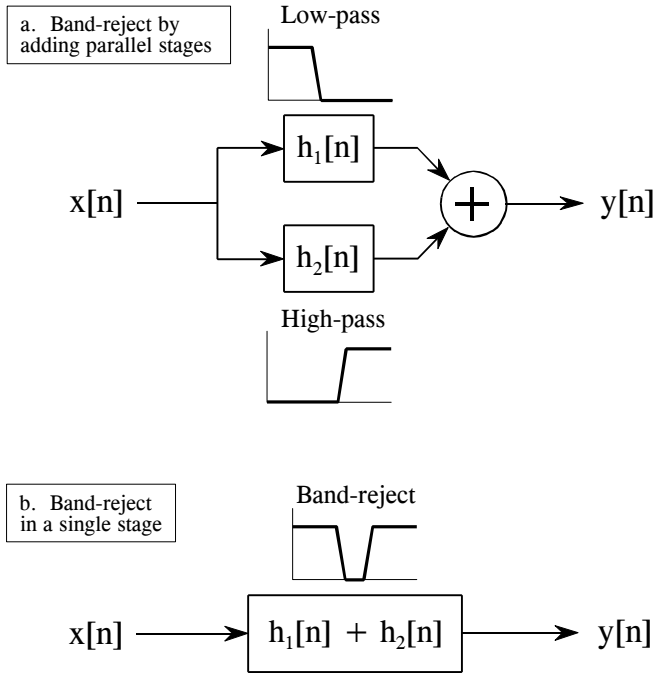

FIGURE 14-9
Designing a band-reject filter. As shown in (a), a band-reject filter is formed by the parallel combination of a low-pass filter and a high-pass filter with their outputs added. Figure (b) shows this reduced to a single stage, with the filter kernel found by *adding* the low-pass and high-pass filter kernels.

amplitude, frequency, and phase of the component sinusoids. The goal of these filters is to separate one band of frequencies from another. Custom filters are used when a special action is required by the filter, something more elaborate than the four basic responses (high-pass, low-pass, band-pass and band-reject). For instance, Chapter 17 describes how custom filters can be used for *deconvolution*, a way of counteracting an unwanted convolution.

		FILTER IMPLEMENTED BY:	
		Convolution *Finite Impulse Response (FIR)*	Recursion *Infinite Impulse Response (IIR)*
FILTER USED FOR:	Time Domain *(smoothing, DC removal)*	Moving average (Ch. 15)	Single pole (Ch. 19)
	Frequency Domain *(separating frequencies)*	Windowed-sinc (Ch. 16)	Chebyshev (Ch. 20)
	Custom *(Deconvolution)*	FIR custom (Ch. 17)	Iterative design (Ch. 26)

TABLE 14-1
Filter classification. Filters can be divided by their *use*, and how they are *implemented*.

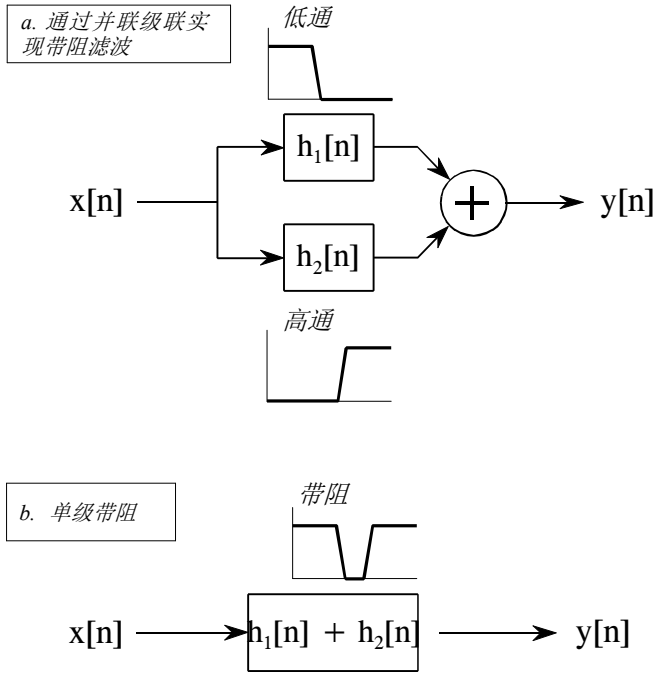

图14-9
设计一个带阻滤波器。如(a)所示，带阻滤波器是由低通滤波器和高通滤波器并联组合而成，其输出相加。图(b)显示了这一过程简化为单级，滤波器核是通过*将*低通和高通滤波器核*相加*得到的。

这些滤波器的作用是分离不同频段的信号，通过调节正弦波的振幅、频率和相位来实现。当滤波器需要执行比高通、低通、带通和带阻这四种基本功能更复杂的特殊操作时，就会使用定制滤波器。例如第17章详细说明了如何运用定制滤波器进行*反卷积*——这种技术能有效消除信号中产生的非预期卷积现象。

所用过滤器：	过滤器实施者：卷积 有限脉冲响应	递归 无限脉冲响应（IIR）
时域（平滑、去直流）	*移动平均线（第15章）*	*单极（第19章）*
频域（分离频率）	*窗形同步（第16章）*	Chebyshev (Ch. 20)
自定义（反卷积）	*FIR定制（第17章）*	*迭代设计（第26章）*

表14-1
过滤器分类。过滤器可以根据*使用*和*实现*方式进行分类。

Digital filters can be implemented in two ways, by *convolution* (also called *finite impulse response* or *FIR*) and by *recursion* (also called *infinite impulse response* or *IIR*). Filters carried out by convolution can have far better performance than filters using recursion, but execute much more slowly.

The next six chapters describe digital filters according to the classifications in Table 14-1. First, we will look at filters carried out by convolution. The *moving average* (Chapter 15) is used in the time domain, the *windowed-sinc* (Chapter 16) is used in the frequency domain, and *FIR custom* (Chapter 17) is used when something special is needed. To finish the discussion of FIR filters, Chapter 18 presents a technique called FFT convolution. This is an algorithm for increasing the speed of convolution, allowing FIR filters to execute faster.

Next, we look at recursive filters. The *single pole* recursive filter (Chapter 19) is used in the time domain, while the *Chebyshev* (Chapter 20) is used in the frequency domain. Recursive filters having a custom response are designed by *iterative techniques*. For this reason, we will delay their discussion until Chapter 26, where they will be presented with another type of iterative procedure: the neural network.

As shown in Table 14-1, *convolution* and *recursion* are rival techniques; you must use one or the other for a particular application. How do you choose? Chapter 21 presents a head-to-head comparison of the two, in both the time and frequency domains.

数字滤波器可以通过两种方式实现，即通过*卷积*（也称为*有限脉冲响应*或*FIR*）和通过*递归*（也称为*无限脉冲响应*或*IIR*）。通过卷积实现的滤波器可能比使用递归的滤波器性能更好，但执行速度要慢得多。

接下来的六章将按照表14-1的分类标准，系统阐述各类数字滤波器。首先我们将重点解析基于卷积运算的滤波器：时域应用的*移动平均*滤波器（第15章）、频域使用的*窗口Sinc*滤波器（第16章），以及需要特殊处理时采用的*自定义FIR*滤波器（第17章）。为深入探讨FIR滤波器，第18章将介绍FFT 卷积技术——这项算法能显著提升卷积运算速度，使FIR滤波器实现更快速的运算。

接下来我们探讨递归滤波器。在时域中使用的是*单极点*递归滤波器（第19章），而在频域中则采用*切比雪夫*滤波器（第20章）。具有自定义响应的递归滤波器需通过*迭代技术*进行设计。因此我们将推迟到第26章再详细讨论这类滤波器，届时将介绍另一种迭代方法——神经网络。

如表14-1所示，*卷积*和*递归*是相互竞争的技术，对于特定的应用，你必须使用其中一种。如何选择？第21章在时域和频域上对两者进行了头对头的比较。