6

Short-Time Fourier Transform

S. Hamid Nawab
Boston University

Thomas F. Quatieri
Lincoln Laboratory

6.0 INTRODUCTION

In this chapter, we examine the short-time Fourier transform (STFT), which has played a significant role in digital signal processing, including speech, music, and sonar applications. It has been found that in such applications it is advantageous to combine traditional time-domain and frequency-domain concepts into a single framework. For example, the physical phenomenon of Doppler shift in signals from moving sources is generally characterized as a change in center frequency over time. If this center frequency is defined in terms of the Fourier transform, we encounter the problem that there is only one Fourier transform for the *entire* signal. It is thus impossible to characterize a change in the center frequency over time. In contrast, the short-time Fourier transform consists of a separate Fourier transform for each instant in time. In particular, we associate with each instant the Fourier transform of the signal in the neighborhood of that instant. To illustrate this, we consider the Fourier transform plots in temporal order in Fig. 6.1. These plots were obtained from successive 2-s intervals of an acoustic recording of a moving helicopter. Each plot contains a gap whenever the Fourier transform magnitude exceeds a threshold. By following such gaps from plot to plot, we can track the frequencies of highest energy. In particular,

S. Hamid Nawab is with the Electrical, Computer, and Systems Engineering Department, Boston University, Boston, MA 02215. Thomas F. Quatieri is with Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA 02173. This work was sponsored by the Department of the Air Force. The views expressed are those of the authors and do not reflect the official policy or position of the U.S. government.

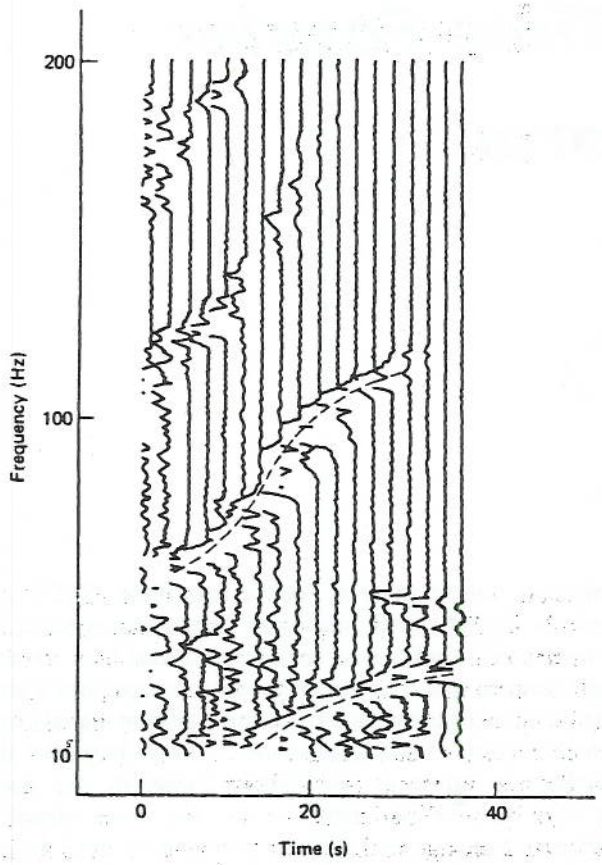

Figure 6.1 Consecutive Fourier transform magnitudes of an acoustic recording of a moving helicopter. Each plot contains a gap whenever the Fourier transform magnitude exceeds a threshold. By following such gaps from plot to plot, we can track the frequencies of highest energy. The frequency tracks of the two highest-energy harmonics are indicated by dashed lines.

we can clearly see the Doppler shift in the various harmonics of the underlying periodic signal. As we will see later in this chapter, the STFT is also useful for characterizing other time-dependent frequency changes in applications such as sonar, music, and speech processing. In fact, it has even been suggested that the human ear performs this type of time-frequency analysis on speech.

The rigorous development of the STFT originally took place in the context of analog signals through the works of Fano [1] and Schroeder and Atal [2]. This work was motivated by previous experimental work for measuring time-dependent spectra with analog devices such as the sound spectrograph [3]. The time-dependent spectrum at the output of such a device is generally displayed in a form known as a spectrogram, an example of which is illustrated in Fig. 6.2 for a segment of speech. In this two-dimensional display, the horizontal axis is time and the vertical axis denotes frequency. The gray level indicates the spectral magnitude, with the darkest regions corresponding to the highest energy. The real-time constraint for such analysis means that the transform at any time should depend only on the past values of the signal. Fano thus defined and developed formal properties of a transform that at any instant weighted the past values of the signal with a decaying exponential and took the squared magnitude of the Fourier transform of the result. Schroeder and Atal extended the concept by using arbitrary weighting functions instead of the exponential. The

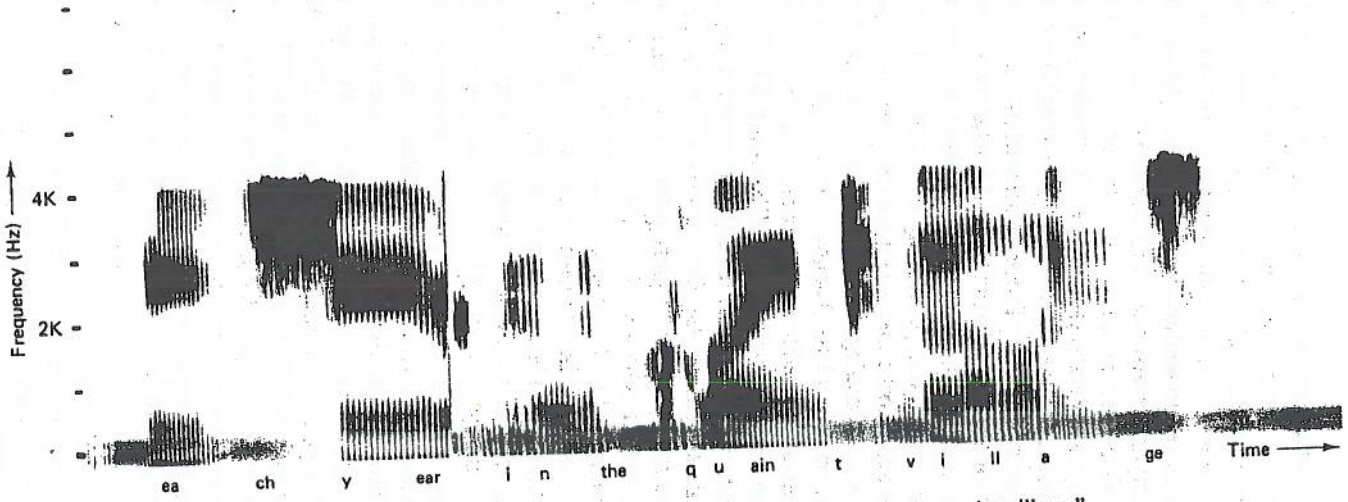

Figure 6.2 The spectrogram for the speech utterance "Each year in the quaint village."

resulting time-frequency function, as we will see, is the magnitude squared of the STFT.

For application in digital signal processing it is necessary to extend the STFT framework to discrete-time signals. An early example of such an extension was the digital spectrogram [4]. This was a digital system for generating speech spectrograms of the type produced by analog devices such as the sound spectrograph. Just as the analog spectrogram can be related to the analog STFT, the digital spectrogram can be related to the concept of a digital or discrete-time STFT. Since in this chapter we are primarily interested in the discrete-time case, we will assume that the STFT corresponds to a discrete-time signal unless stated otherwise. The underlying idea for the discrete-time STFT is once again to take a separate (discrete-time) Fourier transform in the neighborhood of each time sample. These Fourier transforms can be displayed for analysis as in the case of the digital spectrogram. Alternatively, these Fourier transforms can be individually processed and then recombined to form a new processed signal. This enables the signal processing to be adapted to the individual spectral characteristics of each short-time region. Examples of such adaptive processing for the purpose of speech coding and time-scale modification, beamforming in sonar, and image enhancement will be discussed at the end of the chapter.

For practical implementation, each Fourier transform in the STFT has to be replaced by the discrete Fourier transform (DFT). The resulting STFT is discrete in *both* time and frequency and thus is suitable for digital implementation. We call this the *discrete STFT* to distinguish it from the *discrete-time STFT,* which is continuous in frequency. These two transforms, their properties, interrelationships, and applications are the primary focus of this chapter. The discrete-time STFT is particularly useful as a conceptual and analytical tool, while the discrete STFT helps us understand the specific computational details of the algorithms based on the STFT.

This chapter can roughly be divided into two parts, the first four sections dealing with fundamental concepts and issues and the remaining sections dealing with the extension and application of the basic ideas. We begin in Section 6.1 with a formal introduction to the discrete-time and the discrete STFT of a sequence. In particular, we emphasize the similarities and differences in the various properties of the two transforms. In Section 6.2 we illustrate the issues involved in choosing an appropriate framework for computing the STFT in any given situation. We then consider in Section 6.3 the problem of obtaining a sequence back from its STFT. While this is straightforward for the discrete-time STFT, a number of important STFT concepts are introduced for explaining the more complicated area of sequence recovery or synthesis from the discrete STFT. The basic theory part of the chapter is essentially concluded in Section 6.4, which deals with concepts that have been developed for treating the magnitude of the STFT as a transform in its own right. In Section 6.5, we consider the very important practical problem of estimating a signal from a processed STFT that does not satisfy the definitional constraints of the STFT. This area has led to many practical applications of the STFT. However, the STFT is not the only transform to have been considered for time-dependent frequency analysis. In Section 6.6, we consider the relationship of the STFT to some of the other transforms that have been proposed for time-frequency analysis. Finally, in Section 6.7, we illustrate the role the STFT has played in application areas such as speech processing, sensor array processing, and image processing.

6.1 SHORT-TIME FOURIER TRANSFORM OF A SEQUENCE

In this section, we define the STFT representation for a sequence and show how this representation is related to the time- and frequency-domain properties of the original sequence. A major theme used throughout this section is that the representation of the STFT of a sequence is analogous to the Fourier transform representation of a sequence. This analogy is extensively used for deriving STFT properties, including the existence of inverse relations for obtaining a sequence back from its STFT.

6.1.1 Fourier Transform View

The STFT is presented in this section as an extension to the basic Fourier transform definitions for a sequence. In particular, we introduce the *discrete-time STFT* and the *discrete STFT* as counterparts to the discrete-time Fourier transform and the discrete Fourier transform, respectively.

The discrete-time STFT is related to the discrete-time Fourier transform, which is given by

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} \tag{6.1}$$

where ω is a continuous variable denoting frequency. The discrete-time STFT of $x(n)$ is a set of such discrete-time Fourier transforms corresponding to different time sections of $x(n)$. The time section for time n_0 is obtained by multiplying $x(n)$ with a shifted sequence $w(n_0 - n)$. The expression for the discrete-time STFT at time n_0 is therefore given by

$$X(n_0, \omega) = \sum_{n=-\infty}^{\infty} x(n)w(n_0 - n)e^{-j\omega n} \tag{6.2}$$

where $w(n)$ is referred to as the *analysis window* or sometimes as the *analysis filter* for reasons that will become clear later in this chapter. The sequence $f_{n_0}(n) = x(n)w(n_0 - n)$ is generally called a *short-time section* of $x(n)$ at time n_0. This sequence is obtained by time-reversing the analysis window $w(n)$, shifting the result by n_0 points, and multiplying it with $x(n)$. This series of operations is illustrated in Fig. 6.3. Once we have the short-time section for time n_0, we can take its Fourier transform to obtain the frequency function $X(n_0, \omega)$ with n_0 fixed. To obtain $X(n_0 + 1, \omega)$, we slide the time-reversed analysis window one point from its previous position, multiply it with $x(n)$, and take the Fourier transform of the resulting short-time section. Continuing this way, we generate a set of discrete-time Fourier transforms that together constitute the discrete-time STFT. We obtain the mathematical representation for the STFT by replacing the fixed n_0 of Eq. (6.2) by the variable n. To avoid confusion, we rename the variable of summation in Eq. (6.2) as m. We thus obtain the STFT definition:

$$X(n, \omega) = \sum_{m=-\infty}^{\infty} x(m)w(n - m)e^{-j\omega m} \tag{6.3}$$

The analysis window is generally considered to be part of the specification of the

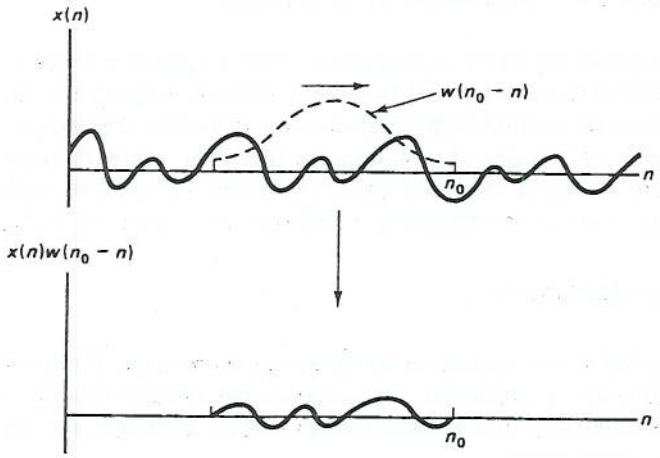

Figure 6.3 The series of operations required to compute a short-time section.

STFT. Since a short-time section of $x(n)$ is the product $x(n)w(n_0 - n)$, it is clear that changing the analysis window will generally change all the short-time sections and therefore the STFT. Typically, the analysis window is selected to have a much shorter duration than the signal $x(n)$ for which the STFT is computed. For example, Fig. 6.4 gives an illustration of an analysis window that is commonly used in speech applications. It is a 256-point window known as a *Hamming window*. In contrast, if $x(n)$ is obtained from a speech sentence lasting 3 s and sampled at 10 kHz, $x(n)$ is a 30,000-point sequence. The shorter duration of the analysis window is what constitutes the *short-time* nature of the STFT.

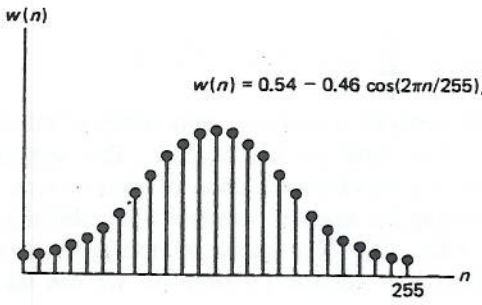

Figure 6.4 A 256-point Hamming window.

For digital processing, we use the discrete STFT, which is related to the discrete-time STFT in the same manner as the DFT is related to the discrete-time Fourier transform. Recall that the DFT $X(k)$ of a finite-duration sequence $x(n)$ is obtained by sampling the discrete-time Fourier transform over one period. That is,

$$X(k) = X(\omega)\big|_{\omega=2\pi k/N} R_N(k) \tag{6.4}$$

where N is the frequency sampling factor and $R_N(k)$ is an N-point rectangular sequence given by

$$R_N(k) = u(k) - u(k - N) \tag{6.5}$$

In analogy, the discrete STFT is obtained from the discrete-time STFT through the following relation:

$$X(n, k) = X(n, \omega)\big|_{\omega=2\pi k/N} R_N(k) \tag{6.6}$$

where we have sampled the discrete-time STFT with a frequency sampling interval of $2\pi/N$ to obtain the discrete STFT. Substituting Eq. (6.3) into Eq. (6.6), we obtain the following relation between the discrete STFT and its corresponding sequence $x(n)$:

$$X(n, k) = \sum_{m=-\infty}^{\infty} x(m)w(n - m)e^{-j2\pi km/N} R_N(k) \tag{6.7}$$

In many applications, the time variation (the n dimension) of $X(n, k)$ is decimated by a temporal decimation factor L to yield the function $X(nL, k)$.

Just as the discrete-time STFT can be viewed as a set of Fourier transforms of the short-time sections $f_n(m)$, the discrete STFT in Eq. (6.7) is easily seen to be a set of DFTs of the short-time sections $f_n(m)$. When the time dimension of the discrete STFT is decimated, the corresponding short-time sections $f_{nL}(m)$ are a subset of $f_n(m)$ obtained by incrementing n by multiples of L. This notion is illustrated in Fig. 6.5.

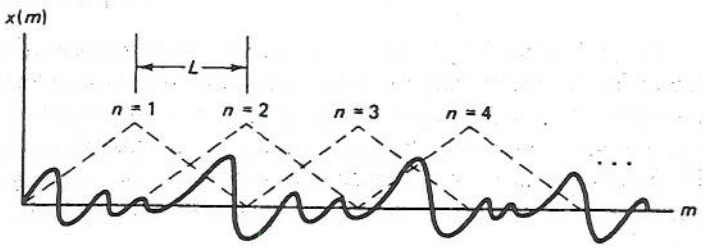

Figure 6.5 The analysis window positions used in computing $X(nL, k)$.

6.1.2 Filtering View

The STFT can also be viewed as the output of a filtering operation where the analysis window $w(n)$ plays the role of the filter impulse response; hence the alternative name *analysis filter* for $w(n)$. For the filtering view of the STFT, we fix the value of ω at ω_0 and rewrite Eq. (6.3) as

$$X(n, \omega_0) = \sum_{m=-\infty}^{\infty} [x(m)e^{-j\omega_0 m}]w(n - m) \tag{6.8}$$

We then recognize from the form of Eq. (6.8) that it represents the convolution of the sequence $x(n)e^{-j\omega_0 n}$ with the sequence $w(n)$. We rewrite Eq. (6.8) as

$$X(n, \omega_0) = [x(n)e^{-j\omega_0 n}] * w(n) \tag{6.9}$$

where $*$ denotes convolution. Furthermore, the product $x(n)e^{-j\omega_0 n}$ can be interpreted as the modulation of $x(n)$ up to frequency ω_0. Thus, $X(n, \omega_0)$ for each ω_0 is a sequence in n that is the output of the process illustrated in Fig. 6.6. The signal $x(n)$ is modulated

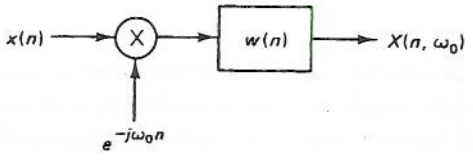

Figure 6.6 Filtering view of STFT analysis at frequency ω_0. Complex exponential modulation is followed by a lowpass filter.

with $e^{-j\omega_0 n}$ and the result passed through a filter whose impulse response is the analysis window, $w(n)$.

A slight variation on the filtering and modulation view of the STFT is obtained by manipulating Eq. (6.9) into the following form:

$$X(n, \omega_0) = e^{-j\omega_0 n}[x(n) * w(n)e^{+j\omega_0 n}] \tag{6.10}$$

In this case, the sequence $x(n)$ is first passed through the same filter as in the previous case except for a linear phase factor. The filter output is then modulated by $e^{-j\omega_0 n}$. This view of the time variation of the STFT for a fixed frequency is illustrated in the block diagram of Fig. 6.7.

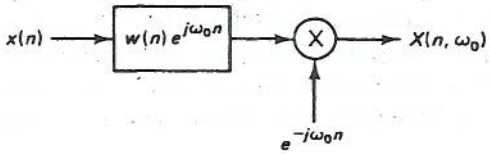

Figure 6.7 Alternative filtering view of STFT analysis at frequency ω_0. A bandpass filter is followed by complex exponential modulation.

The discrete STFT of Eq. (6.7) can also be interpreted from the filtering viewpoint. In particular, having a finite number of frequencies allows us to view the discrete STFT as the output of the filter bank shown in Fig. 6.8. Note that each filter is acting as a bandpass filter centered around its selected frequency. Thus the discrete STFT can be viewed as a collection of sequences, each corresponding to the frequency components of $x(n)$ falling within a particular frequency band.

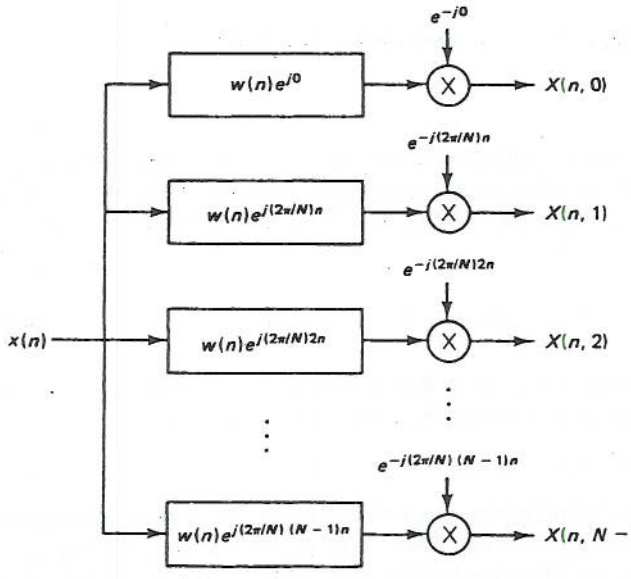

Figure 6.8 The discrete STFT as the output of a filter bank consisting of bandpass filters.

6.1.3 Properties

In this section we develop a number of properties of the discrete and the discrete-time STFT. These properties provide insight into the characteristics of these transforms for various classes of signals and certain commonly used signal manipulations (e.g., time

shifting). The sampling relationship between the discrete-time STFT, $X(n, \omega)$, and the discrete STFT, $X(n, k)$, makes it convenient to first investigate the properties of $X(n, \omega)$ alone and then to see how they are affected by the sampling. We will see that most of the properties of the discrete-time STFT have similar counterparts for the discrete STFT. A number of properties of the discrete-time STFT are easily obtained by using the Fourier transform viewpoint, while a different set of properties is most convenient to derive from the filtering viewpoint.

We begin with a set of properties based on the interpretation of the STFT as a collection of Fourier transforms corresponding to the short-time sections of the sequence. In particular, we view $X(n, \omega)$ as a Fourier transform for each fixed n. Therefore, the frequency function $X(n, \omega)$ for each n has all the general properties of a Fourier transform. Table 6.1 lists a number of these properties.

TABLE 6.1 PROPERTIES OF THE DISCRETE-TIME STFT BASED ON THE FOURIER TRANSFORM INTERPRETATION

Property 1:	$X(n, \omega) = X(n, \omega + 2\pi)$
Property 2:	$x(n)$ real $\longleftrightarrow X(n, \omega) = X^*(n, -\omega)$
Property 3:	$x(n)$ real $\longleftrightarrow \lvert X(n, \omega)\rvert = \lvert X(n, -\omega)\rvert$
Property 4:	$x(n)$ real $\longleftrightarrow \arg[X(n, \omega)] = -\arg[X(n, -\omega)]$
Property 5:	$x(n - n_0) \longleftrightarrow e^{-j\omega n_0} X(n - n_0, \omega)$

The first property follows from the periodic nature of the discrete-time Fourier transform, whereas the next three properties follow from the conjugate-symmetric property of the Fourier transform of real sequences. The fifth property is analogous to the Fourier transform property that a time shift in a sequence leads to a linear phase factor in the frequency domain. A shift by n_0 in the original time sequence also means that we obtain the same short-time sections as before except that each short-time section has also been shifted in time by n_0. For this reason there is also a time shift indicated in the STFT of property 5.

The properties of the discrete-time STFT given in Table 6.1 can be extended to the discrete STFT using the relationship in Eq. (6.6) where $2\pi/N$ is the sampling interval in frequency. The resulting properties of the discrete STFT are shown in Table 6.2.

The first property emphasizes the aperiodic nature of the discrete STFT in the frequency dimension. The next three symmetry properties are natural counterparts to the properties of the discrete-time STFT. The shifting property is also a straightforward extension of the corresponding property of the discrete-time STFT. It should

TABLE 6.2 PROPERTIES OF THE DISCRETE STFT

Property 1:	$X(n, k)$ is zero outside $0 \le k < N$
Property 2:	$x(n)$ real $\longleftrightarrow X(n, k) = X^*(n, N - k)$ for $0 < k < N$
Property 3:	$x(n)$ real $\longleftrightarrow \lvert X(n, k)\rvert = \lvert X(n, N - k)\rvert$ for $0 < k < N$
Property 4:	$x(n)$ real $\longleftrightarrow \arg[X(n, k)] = -\arg[X(n, N - k)]$ for $0 < k < N$
Property 5:	$x(n - n_0) \longleftrightarrow e^{-j(2\pi n_0 k/N)} X(n - n_0, k)$

be noted that if the discrete STFT is decimated in time by a factor L, the first four properties still hold. However, for the fifth property, when the shift is not an integer multiple of L, there is no general relationship between the discrete STFTs of $x(n)$ and $x(n - n_0)$. This happens because the short-time sections corresponding to $X(nL, k)$ cannot be expressed as shifted versions of the short-time sections corresponding to the discrete STFT of $x(n - n_0)$.

As with the Fourier transform interpretation, the filtering view also allows us to easily deduce a number of STFT properties. In particular, we view $X(n, \omega)$ as a filter output for each fixed frequency. Therefore, the time variation of $X(n, \omega)$ for each ω has all the general properties of a filtered sequence. Table 6.3 lists a number of these properties for the discrete-time STFT.

TABLE 6.3 PROPERTIES OF DISCRETE-TIME STFT BASED ON THE FILTERING INTERPRETATION

Property 1:	$X(n, 0) = x(n) * w(n)$
Property 2:	$x(n)$ length N, $w(n)$ length M, $X(n, \omega)$ length $N + M - 1$ along n
Property 3:	Bandwidth of sequence $X(n, \omega_0) \leq$ Bandwidth of $w(n)$
Property 4:	The sequence $X(n, \omega_0)$ has spectrum centered at the origin
Property 5:	$x(n)$ causal, $w(n)$ causal, $X(n, \omega)$ causal in time

The first property is obtained by substituting $\omega_0 = 0$ in Eq. (6.9). In the second property, we make use of a standard result for the length of a sequence obtained through the convolution of any two sequences of lengths N and M. For the third property, we note that $X(n, \omega)$ as a function of n is the output of a filter whose bandwidth is the bandwidth of the analysis window. The fourth property follows from the modulation step used in obtaining $X(n, \omega_0)$. The fifth property is a standard result on the convolution of causal sequences. The STFT properties from the filtering viewpoint remain the same for the discrete STFT since, for a fixed frequency, the time variation of the discrete STFT is the same as the time variation of the discrete-time STFT at that frequency.

We have seen in this section that a number of STFT properties can be derived from either the Fourier transform viewpoint or the filtering viewpoint. There are of course many other properties of the STFT, but in this section we have concentrated on the ones typically encountered in various application areas.

6.1.4 Invertibility

In this section we consider the problem of obtaining a sequence back from its discrete or discrete-time STFT. Whereas the discrete-time STFT is always invertible in this sense, the discrete STFT requires certain constraints on its sampling rate for invertibility.

The invertibility of the discrete-time STFT is best seen in analogy with the discrete-time inverse Fourier transform. In the Fourier transform interpretation, the discrete-time STFT is viewed for each value of n as a function of frequency obtained

by taking the Fourier transform of the short-time section $f_n(m) = x(m)w(n - m)$. It follows that if for each n we take the inverse Fourier transform of the corresponding function of frequency, then we obtain the sequence $f_n(m)$. If we evaluate this short-time section at $m = n$ we obtain the value $x(n)w(0)$. Assuming that $w(0)$ is nonzero, we can divide by $w(0)$ to recover $x(n)$. The process of taking the inverse Fourier transform of $X(n, \omega)$ for a specific n and then dividing by $w(0)$ is represented by the following relation:

$$x(n) = \left[\frac{1}{2\pi w(0)}\right] \int_{2\pi} X(n, \omega) e^{j\omega n}\, d\omega \tag{6.11}$$

This equation represents a *synthesis equation* for the discrete-time STFT. In fact, there are numerous synthesis equations that map $X(n, \omega)$ uniquely back to $x(n)$. We will discuss these in Section 6.4.

In contrast to the discrete-time STFT, the discrete STFT $X(n, k)$ is not always invertible. For example, consider the case when $w(n)$ is bandlimited with bandwidth B. Figure 6.9 shows the filter regions used to obtain $X(n, k)$ for the case when the sampling interval $2\pi/N$ is greater than B. Note that in this case there are frequency components of $x(n)$ that do not pass through any of the filter regions of the discrete STFT. Those frequency components can have arbitrary values and yet the discrete STFT would be the same. Thus in such cases the discrete STFT is not a unique representation of $x(n)$ and therefore cannot be invertible. The invertibility problem is also of interest when the discrete STFT has been decimated in time. For example, consider the case when the analysis window $w(n)$ is nonzero over its finite length N_w. Figure 6.10 shows the case when temporal decimation factor L is greater than N_w. Note that in this case there are samples of $x(n)$ that are not included in any short-time section of the discrete STFT. These samples can have arbitrary values and yet the time-decimated discrete STFT would be the same. Thus in such cases the discrete STFT is not a unique representation of $x(n)$ and therefore cannot be invertible.

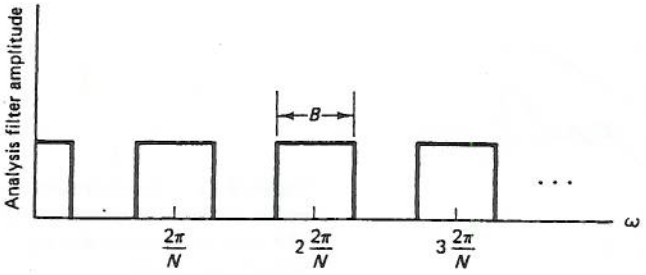

Figure 6.9 Undersampled STFT when the frequency sampling interval $2\pi/N$ is greater than the analysis filter bandwidth B.

By selecting appropriate constraints on the frequency-sampling and time-decimation rates, the discrete STFT becomes invertible. For example, let's consider the case of a finite-length analysis window. We have already seen that in such cases the discrete STFT is not invertible if the temporal decimation factor L is greater than the analysis window length N_w. We will now see that if the temporal decimation factor is less than or equal to the analysis window length, then the discrete STFT is invertible provided we impose constraints on the frequency sampling interval. Suppose that the temporal decimation factor is equal to the analysis window length. The discrete STFT

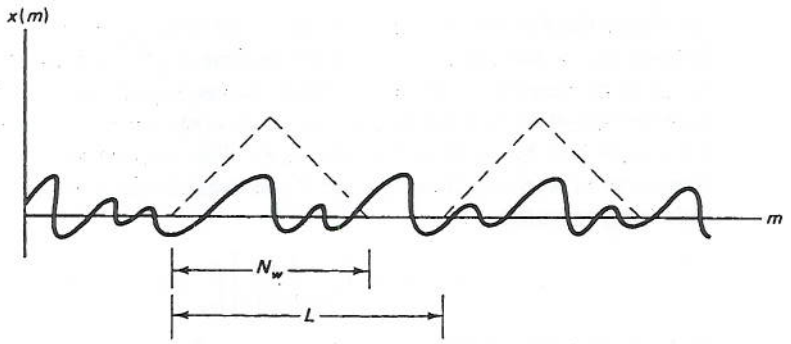

Figure 6.10 Underdecimated STFT when the decimation factor L is larger than the length N_w of the analysis window.

in this case consists of the DFTs of adjacent but nonoverlapping short-time sections, as illustrated in Fig. 6.11. It is clear that the signal $x(n)$ can be obtained only if each of these short-time sections is known for all time. Thus to reconstruct $x(n)$ from its discrete STFT, we must require that each N_w-point short-time section be completely recoverable from its DFT. However, it is well known that the DFT of an N_w-point sequence is invertible provided its frequency sampling interval is less than $2\pi/N_w$. It follows from this discussion that the discrete STFT is invertible for situations where the analysis window is nonzero over its finite-length N_w, the temporal decimation factor L is less than N_w, and the frequency sampling interval $2\pi/N$ is less than $2\pi/N_w$. Even tighter bounds than these can be derived, as we will see in Section 6.4. Furthermore, we will also see that a variety of other assumptions about the analysis window lead to different tradeoffs between time and frequency sampling.

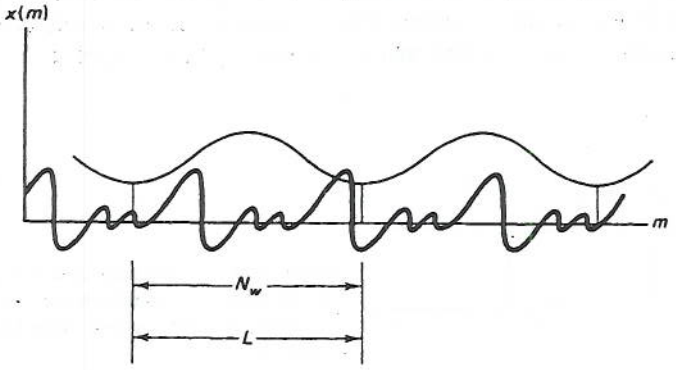

Figure 6.11 The decimation factor L equals the analysis window length N_w. The window positions are adjacent without any overlap or missed regions.

6.2 SHORT-TIME FOURIER ANALYSIS

As we discussed in Section 6.0, the time-varying spectral characteristics of a sequence can be analyzed from its STFT. To carry out such short-time Fourier analysis we must select the analysis window as well as the algorithm for computing the discrete STFT. In Section 6.2.1, we discuss the factors influencing analysis window selection. Some of these factors depend on the algorithm used to compute the discrete STFT. There are

two basic approaches to computing the discrete STFT that correspond to the Fourier transform and filtering interpretations of the STFT discussed in Section 6.1. In Sections 6.2.2 and 6.2.3, we examine each of these approaches and the computational issues associated with them.

6.2.1 Selection of Analysis Window

The properties of the STFT are sensitive to the selection of the analysis window. To analyze the output of STFT analysis we should have an understanding of the effects of the analysis window on the characteristics of the STFT. In this section we discuss the nature of the dependence of the STFT properties on the analysis window.

A basic issue in analysis window selection is the compromise required between a long window for frequency resolution and a short window for not allowing the temporal properties of the signal to vary appreciably. To see this, we first recall that the STFT $X(n, \omega)$ is the Fourier transform of the short-time section $f_n(m) = x(m)w(n - m)$. From Fourier transform theory, we know that the Fourier transform of the product of two sequences is given by the convolution of their respective Fourier transforms. With $X(\omega)$ as the Fourier transform of $x(n)$, and $W(-\omega)e^{-j\omega n}$ as the Fourier transform of $w(n - m)$ with respect to the variable m, we can write the STFT as

$$X(n, \omega) = \left(\frac{1}{2\pi}\right)\int_{-\pi}^{\pi} W(\theta)e^{j\theta n}X(\omega + \theta)\, d\theta \tag{6.12}$$

Thus, any frequency variation of the STFT for any fixed time may be interpreted as a smoothed version of the Fourier transform of the underlying signal, as illustrated in Fig. 6.12. Thus, for faithful reproduction of the properties of $X(\omega)$ in $X(n, \omega)$, the function $W(\omega)$ should appear as an impulse with respect to $X(\omega)$. The closer $W(\omega)$ is to an impulse (i.e., narrow bandwidth), $X(n, \omega)$ is said to have better frequency resolution. However, for a given window, frequency resolution varies inversely with the length of the window. Thus, better frequency resolution requires longer analysis windows, whereas the desire for short-time analysis requires shorter analysis windows. An example of this tradeoff is shown in Fig. 6.13. Here we have the Fourier transform of a section of a chirp signal whose frequency is a linear function of time. The aim is to measure the instantaneous frequency at time t_0. This is done using analysis windows of various lengths. Very short windows give low resolution because

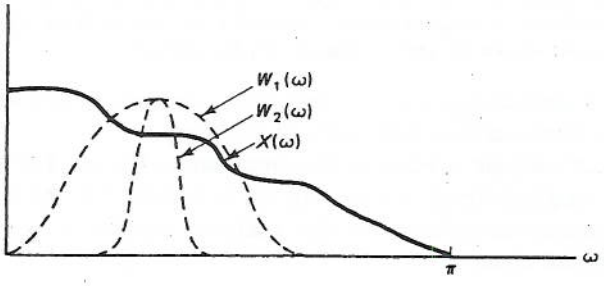

Figure 6.12 The window Fourier transform as a smoothing function for the Fourier transform of the underlying signal. A narrowband and a wideband window are illustrated.

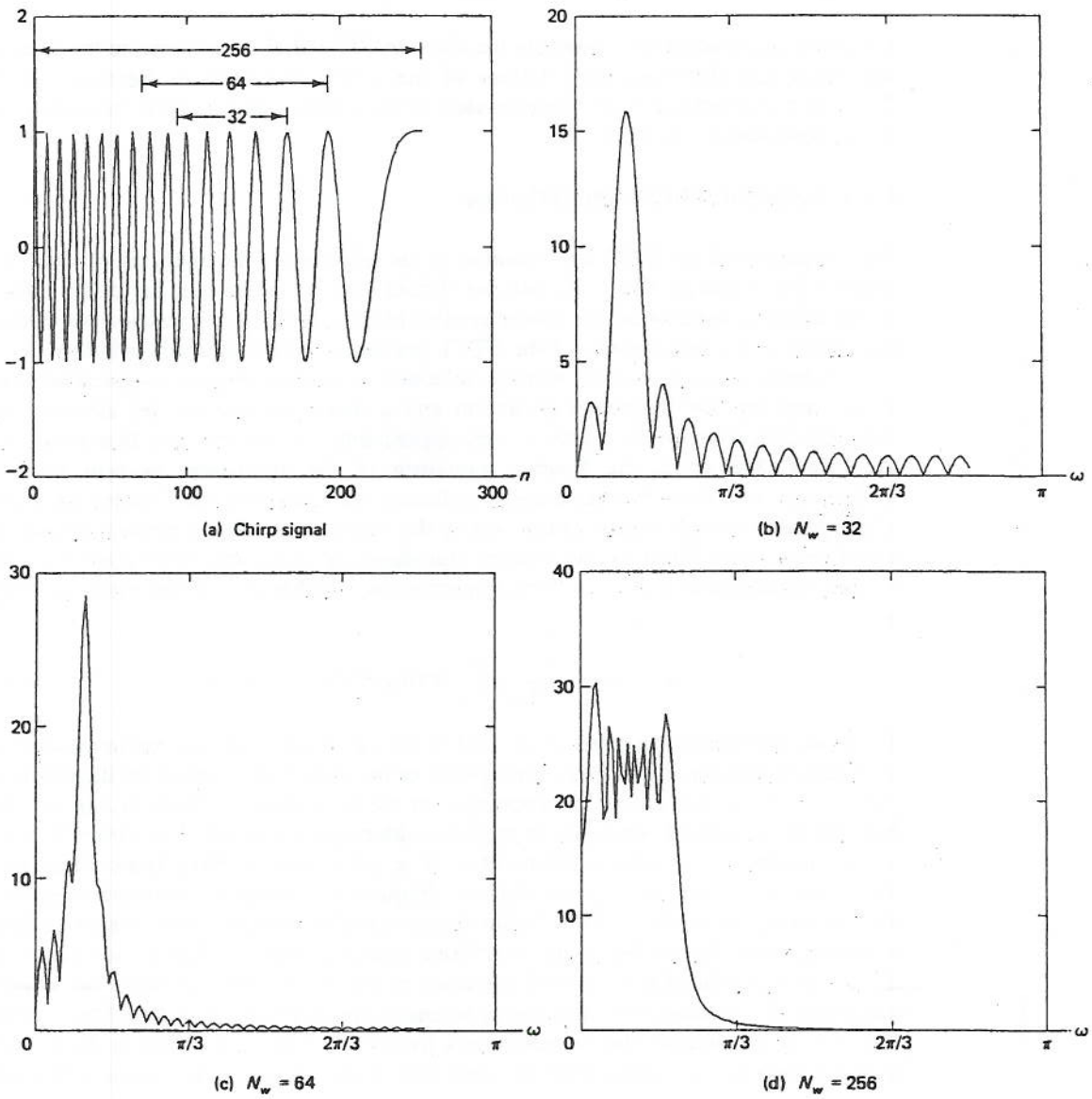

Figure 6.13 Effect of the length of the analysis window on the Fourier transform of a linearly frequency-modulated sinusoid of length 256 samples whose frequency increases from $\pi/4$ to $\pi/8$. The Fourier transform is taken for sections centered around the 128th sample (frequency $3\pi/16$). Transforms are shown for window lengths 32, 64, and 256.

of window spectrum smoothing, whereas very long windows yield a wideband spectrum because of the time-varying frequency of the chirp.

The selection of analysis window is also important when the STFT is to be used for synthesizing the original signal. As we will see in Section 6.3, the STFT synthesis methods impose various constraints on the analysis window to guarantee reconstruction of the original signal.

6.2.2 Fast Fourier Transform Implementation

We have seen that the discrete STFT can be viewed as the DFT of each short-time section corresponding to a different shift in the finite-length analysis window $w(n)$. Thus one approach to discrete STFT analysis is to implement a DFT process that computes equally spaced frequency samples of $X(n, \omega)$ for each possible value of n. The DFT computation is generally on the order of the square of the DFT length. However, if the number of frequency samples N is chosen to be a highly composite number (usually a power of two) then, as we will see in this section, the fast Fourier transform (FFT) algorithm can be used to efficiently carry out the DFT computations in the discrete STFT.

To see how the FFT can be used for short-time Fourier analysis, let us assume that the analysis window $w(n)$ has length N_w. The function we are interested in computing for $k = 0, 1, \ldots, N-1$, is given by

$$X(n,k) = \sum_{m=-\infty}^{\infty} w(n-m)x(m)e^{-j2\pi mk/N} = \sum_{m=-\infty}^{\infty} f_n(m)e^{-j2\pi mk/N} \tag{6.13}$$

where $f_n(m)$ are the short-time sections of $x(n)$. First let us consider the case where the desired number of frequency samples N is greater than or equal to N_w. In this case, each short-time section $f_n(m)$, with n fixed, is a finite-length sequence whose length is less than or equal to N. It follows that $X(n, k)$ for a fixed n can be computed as an N-point FFT applied to $f_n(m)$ padded with the appropriate number of zeros.

Now we consider the case where the number of frequency samples N is less than the analysis window length N_w. Since $X(n, \omega)$ is the Fourier transform of $f_n(m)$, uniform frequency sampling of $X(n, \omega)$ with sampling interval $2\pi/N$ corresponds in the time domain to convolution of $f_n(m)$ with a periodic impulse train with period N. The result of this convolution is given by

$$\tilde{f}_n(m) = \sum_{k=-\infty}^{\infty} f_n(m + kN) \tag{6.14}$$

The sequence $\tilde{f}_n(m)$, known as the time-aliased version of $f_n(m)$, is periodic with period N, and the N-point FFT of one period of $\tilde{f}_n(m)$ yields the desired N frequency samples of the discrete STFT. If $X(nL, k)$ is the desired function, we compute the $\tilde{f}_n(m)$ only for every Lth short-time section. That is,

$$\tilde{f}_{nL}(m) = \sum_{k=-\infty}^{\infty} f_{nL}(m + kN) \tag{6.15}$$

In summary, the FFT can be used to efficiently compute $X(n, k)$ by computing the time-aliased version of each short-time section and then applying the N-point FFT to each of those sections. Note that if N is greater than or equal to the analysis window length, the computation of the time-aliased version is eliminated.

6.2.3 The Filter Bank Approach

Short-time Fourier analysis can also be carried out through computations suggested by the filtering interpretation of the discrete STFT. In this section we develop the formal

details of the filter bank analysis. In particular, we present several alternatives for such implementation. The filter bank analysis is advantageous in applications requiring infinite or long-duration windows or where the STFT is to be computed at a small number of frequencies. These advantages will be discussed in more detail at the end of the section.

To develop the filter bank implementation techniques, we begin with the filtering view of the discrete STFT:

$$X(n,k) = e^{-j\omega_k n}[x(n) * w(n)e^{j\omega_k n}] \tag{6.16}$$

where $\omega_k = 2\pi k/N$ for $k = 0, 1, \ldots, N-1$. The idea is to pass the signal $x(n)$ through a bank of bandpass filters, shown in Fig. 6.8, where the output of each filter is the time variation of the STFT at the frequency ω_k. If the output of each filter is decimated in time by a factor L, we obtain the discrete STFT, $X(nL,k)$. Each branch of the filter bank corresponds to a bandpass filter centered around the frequency $\omega_k = 2\pi k/N$.

An alternative implementation of the filter bank analysis replaces the bandpass filters by lowpass filters as in Fig. 6.14. The equation for each branch of the lowpass filter bank is given by

$$X(n,k) = x(n)e^{-j\omega_k n} * w(n) \tag{6.17}$$

where $\omega_k = 2\pi k/N$ for $k = 0, 1, \ldots, N-1$. As in the bandpass implementation, we can obtain $X(nL,k)$ by decimating each filter output by the decimation factor L.

Both the bandpass and the lowpass techniques of Figs. 6.8 and 6.14 involve complex exponential modulations. To carry out this complex modulation in terms of real operations, a more detailed specification of the filter bank implementations is needed. For the lowpass implementation, we note that

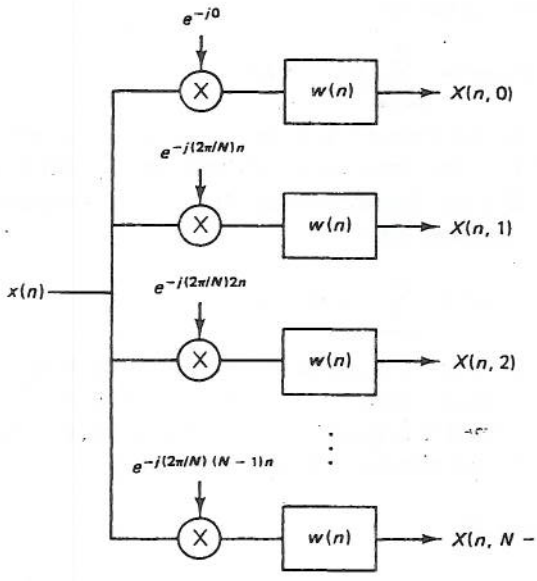

Figure 6.14 The discrete STFT as the output of a filter bank consisting of lowpass filters.

$$\begin{aligned} X(n,k) &= x(n)e^{-j\omega_k n} * w(n) \\ &= [x(n)\cos(\omega_k n) * w(n)] + j[x(n)\sin(\omega_k n) * w(n)] \\ &= X_r(n,k) + jX_i(n,k) \end{aligned} \tag{6.18}$$

where $X_r(n,k)$ and $X_i(n,k)$ can be implemented as shown in Fig. 6.15. Similarly, we can derive a real implementation of the bandpass analysis. The computation for this implementation is represented in the block diagram of Fig. 6.16.

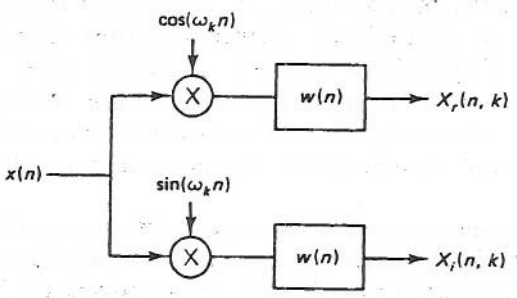

Figure 6.15 Implementation with real operations of each branch of the lowpass filter bank for STFT analysis.

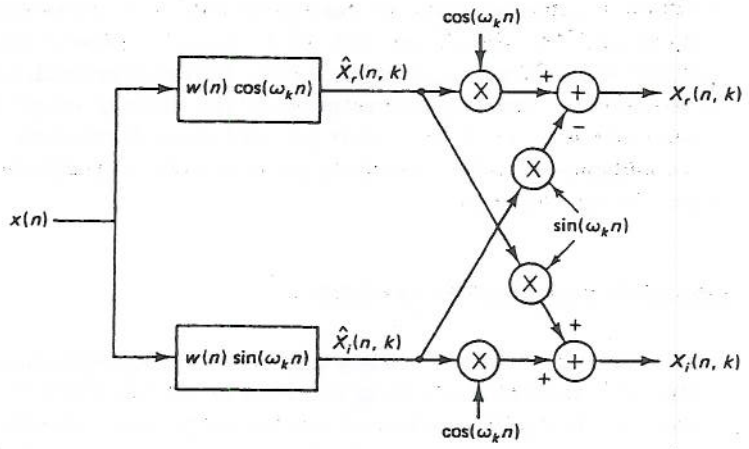

Figure 6.16 Implementation with real operations of each branch of the bandpass filter bank for STFT analysis.

If the bandpass filter bank has the modulation components removed, then the overall structure becomes much simpler and it computes samples of $X(n,\omega)e^{j\omega n}$ rather than the samples of $X(n,\omega)$. Such an output often suffices for STFT analysis. In particular, the linear phase factor does not affect the kinds of spectral characteristics (e.g., bandwidth, LPC parameters) that are generally of interest in short-time Fourier analysis of signals like speech. Another reason, which we will explain further in Section 6.3, is that under certain conditions the outputs of the filter bank in Fig. 6.8 can be simply summed to retrieve the original sequence $x(n)$. In particular, for this purpose it is required that the frequency responses of the filters should add up to give unity across the entire bandwidth of $x(n)$.

The filter bank implementations are particularly useful when the analysis win-

dow is infinitely long (e.g., an exponential) or well approximated by an infinitely long window. In such cases the filters can be implemented by recursive (infinite impulse response) techniques. It is noted that the FFT approach to STFT analysis can handle infinite-duration windows only by approximating them with finite-length windows. As an example, consider a window $w(n) = (0.9)^n u(n)$. A recursive implementation of this analysis filter is given by

$$y(n) = 0.9y(n - 1) + x(n) \tag{6.19}$$

where $x(n)$ and $y(n)$ are the input and output of the filter with impulse response $w(n)$. This requires on the order of two operations per sample of the discrete STFT. On the other hand, using an N-point FFT implementation, where N is the frequency sampling factor, requires Log N operations per sample of the discrete STFT. In our example, the exponential window may be approximated to within 16-bit precision by a 128-point truncation. In this case, the FFT analysis would require Log(128) = 7 operations per sample of the discrete STFT. This compares with 2 operations for the filter bank analysis.

The filter bank approach may also be advantageous when the STFT is to be computed at a relatively small number of frequencies. In some applications, one is interested in the STFT over only a small band of frequencies. For example, with a 256-point analysis window one may desire four consecutive frequencies with resolution $2\pi/256$. In this case the FFT approach would require $N \log N = 256 \log(256) = 2048$ operations per unit time. On the other hand, the filter bank approach would use four finite impulse response (FIR) filters of length 256 points each. This means a total of 1024 operations per unit time. Furthermore, if the filters can be suitably approximated by infinite impulse reponse (IIR) filters, the computation may require fewer operations.

6.3 SHORT-TIME FOURIER SYNTHESIS

As discussed in Section 6.0, many digital processing applications of the STFT employ methods for synthesizing a sequence from its discrete STFT. We have already seen in Section 6.1 that such synthesis is not always possible since the discrete STFT is not invertible under all conditions. On the other hand, we have seen that the *discrete-time* STFT is always invertible. A common approach to developing synthesis methods for the discrete STFT has been to start from one of the many *synthesis equations* that express a sequence in terms of its discrete-time STFT. A discretized version of such an equation is then considered as the basis for a candidate synthesis method. Conditions are derived under which such a method can synthesize a sequence from its discrete STFT. The various synthesis methods differ not only with respect to the conditions under which they are valid but also in terms of their computational properties.

In this section we present a number of synthesis methods as well as their underlying theory. In particular, we begin with two classical methods that have been widely used for short-time synthesis. They are the filter bank summation (FBS) method and the overlap-add (OLA) method. There are other useful synthesis methods that are best understood by introducing the concept of a *synthesis filter* for the STFT. This concept and the associated methods are also discussed in this section.

6.3.1 Filter Bank Summation (FBS) Method

In this section we present a traditional short-time synthesis method that is commonly referred to as the filter bank summation (FBS) method. This method is best described in terms of the filtering interpretation of the discrete STFT. In this interpretation, the discrete STFT is considered to be the set of outputs of a bank of filters. In the FBS method, the output of each filter is modulated with a complex exponential and these modulated filter outputs are summed at each instant of time to obtain the corresponding time sample of the original sequence, as shown in Fig. 6.17. For dealing with temporal decimation, the traditional strategy is to perform a temporal interpolation filtering on the discrete STFT to restore the temporal decimation factor to unity. The FBS method is then performed on the interpolated output.

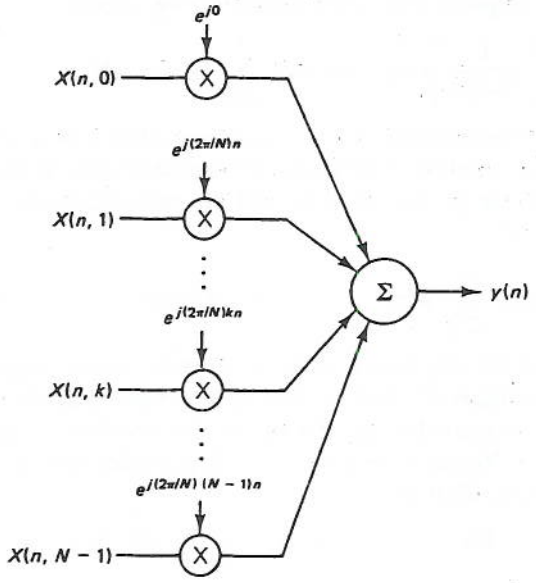

Figure 6.17 Filter bank summation procedure for signal synthesis from the discrete STFT.

The FBS method is motivated by the following relation between a sequence and its discrete-time STFT:

$$x(n) = \left[\frac{1}{2\pi w(0)}\right] \int_{-\pi}^{\pi} X(n, \omega) e^{j\omega n}\, d\omega \tag{6.20}$$

where without loss of generality we assume that $w(0)$ is nonzero. We derived this equation in Section 6.2 in the context of the invertibility of the discrete-time STFT. The FBS method carries out a discretized version of the operations suggested on the right side of Eq. (6.20). That is, given a discrete STFT $X(n, k)$, the FBS method synthesizes a sequence $y(n)$ satisfying the following equation:

$$y(n) = \left[\frac{1}{Nw(0)}\right] \sum_{k=0}^{N-1} X(n, k) e^{j2\pi nk/N} \tag{6.21}$$

We are of course interested in the FBS method for those situations where the sequence $y(n)$ in Eq. (6.21) is the same as the sequence $x(n)$ corresponding to the

discrete STFT $X(n, k)$. Substituting $X(n, k)$ in Eq. (6.21) for the FBS method, we obtain

$$y(n) = \left[\frac{1}{Nw(0)}\right] \sum_{k=0}^{N-1} \sum_{m=-\infty}^{\infty} x(m)w(n-m)e^{-j2\pi km/N} e^{j2\pi nk/N} \tag{6.22}$$

Using the linear filtering interpretation of the STFT, this equation reduces to

$$y(n) = \left[\frac{1}{Nw(0)}\right] x(n) * \sum_{k=0}^{N-1} w(n)e^{j2\pi nk/N} \tag{6.23}$$

Taking $w(n)$ out of the summation and noting that the finite sum over the complex exponentials reduces to an impulse train with period N, we obtain

$$y(n) = \left[\frac{1}{Nw(0)}\right] x(n) * w(n)N \sum_{r=-\infty}^{\infty} \delta(n - rN) \tag{6.24}$$

In Eq. (6.24) we note that $y(n)$ is obtained by convolving $x(n)$ with a sequence that is the product of the analysis window with a periodic impulse train. It follows that if we desire $y(n) = x(n)$, then the product of $w(n)$ and the periodic impulse train must reduce to $Nw(0)\delta(n)$. That is,

$$w(n)N \sum_{r=-\infty}^{\infty} \delta(n - rN) = Nw(0)\delta(n) \tag{6.25}$$

This will clearly be satisfied for any causal analysis window whose length N_w is less than the number of analysis filters N. That is, *any* finite-length analysis window can be used in the FBS method provided the length of the window is less than the frequency sampling factor N. We can even have $N < N_w$ provided $w(n)$ is chosen such that every Nth sample is zero. That is,

$$w(rN) = 0 \qquad \text{for} \qquad r = -1, 1, -2, 2, -3, 3, \ldots \tag{6.26}$$

as illustrated in Fig. 6.18.

Equation (6.25) is known as the FBS constraint because this is the requirement on the analysis window that ensures exact signal synthesis with the FBS method. This constraint is more commonly expressed in the frequency domain. Taking the Fourier transform of both sides of Eq. (6.25), we obtain

$$\sum_{k=0}^{N-1} W(\omega - 2\pi k/N) = Nw(0) \tag{6.27}$$

This constraint essentially states that the frequency responses of the analysis filters should sum to a constant across the entire bandwidth, as shown in Fig. 6.19. We have already seen that any finite-length analysis window whose length is less than the frequency sampling factor N satisfies this constraint. We conclude that a filter bank with N filters, based on an analysis filter of length less than N, is *always* an allpass system.

As noted before, the FBS method is just one of the many different methods available for synthesizing a sequence from its discrete STFT. Each such method imposes its own set of constraints on the analysis window as for the FBS method in

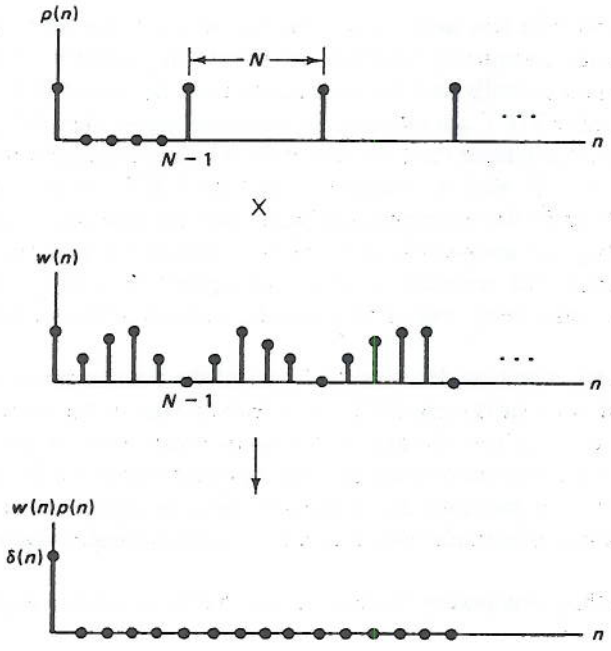

Figure 6.18 Example of an analysis window and how it satisfies the FBS constraint. The analysis window length is longer than the frequency sampling factor.

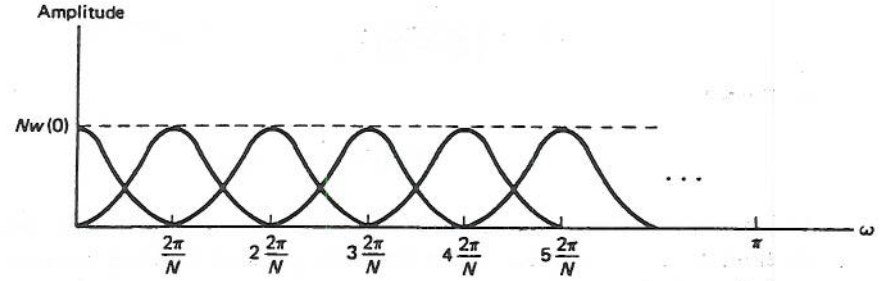

Figure 6.19 The FBS constraint visualized in the frequency domain.

Eq. (6.27). Additionally, each method has its own computational characteristics. The FBS method for decimation factor $L = 1$ and an analysis filter of length N_w requires on the order of N_w operations (complex additions and multiplications) per sample of $x(n)$. On the other hand, if the decimation factor $L \gg 1$, the FBS method must be extended to include interpolation of each analysis filter output. This leads to a computational requirement of order N_w^2 operations per sample of $x(n)$.

6.3.2 Overlap-Add (OLA) Method

We now consider another classical method, the overlap-add (OLA) method for short-time synthesis. Just as the FBS method was motivated from the filtering view of the STFT, the OLA method is motivated from the Fourier transform view of the STFT.

The simplest method obtainable from the Fourier transform view is in fact not the OLA method. It is instead a method known as the inverse discrete Fourier

transform (IDFT) method. In this method, for each fixed time, we take the inverse DFT of the corresponding frequency function and divide the result by the analysis window. This method is generally not favored in practical applications because the slightest perturbation in the STFT can result in a synthesized signal very different from the original. For example, consider the case where the STFT is multiplied by a linear phase factor of the form $e^{j\omega n_0}$ with n_0 unknown. Then the IDFT for each fixed time results in a shifted version of the corresponding short-time section. Since the shift n_0 is unknown, dividing by the analysis window without taking the shift into account introduces a distortion in the resulting synthesized signal. In contrast, the OLA method, which we describe next, results in a shifted version of the original signal without distortion.

The OLA method is also best described in terms of the Fourier transform view. In the OLA method, we take the inverse DFT for each fixed time in the discrete STFT. However, instead of dividing out the analysis window from each of the resulting short-time sections, we perform an overlap-and-add operation between the short-time sections. This method works provided the analysis window is designed such that the overlap-and-add operation effectively eliminates the analysis window from the synthesized sequence.

The OLA method is motivated by the following relation between a sequence and its discrete-time STFT:

$$x(n) = \left[\frac{1}{2\pi W(0)}\right] \int_{-\pi}^{\pi} \sum_{r=-\infty}^{\infty} X(r, \omega) e^{j\omega n} \, d\omega \tag{6.28a}$$

where

$$W(0) = \sum_{n=-\infty}^{\infty} w(n) \tag{6.28b}$$

To derive this synthesis equation, we note that if we take the Fourier transform of both sides of Eq. (6.9) with respect to the variable n and evaluate the result at frequency zero, we obtain

$$\tilde{X}(\phi, \omega)\big|_{\phi=0} = X(\omega) W(0) \tag{6.29}$$

where we denote the Fourier transform in n of $X(n, \omega)$ by $\tilde{X}(\phi, \omega)$. But because a Fourier transform evaluated at zero frequency is equal to the sum of all the samples of the time-domain sequence, we have

$$\tilde{X}(\phi, \omega)\big|_{\phi=0} = \sum_{r=-\infty}^{\infty} X(r, \omega) \tag{6.30}$$

Now, dividing Eq. (6.29) by $W(0)$ and substituting for $\tilde{X}(\phi, \omega)\big|_{\phi=0}$ the expression in Eq. (6.30), we obtain

$$X(\omega) = \left[\frac{1}{W(0)}\right] \sum_{r=-\infty}^{\infty} X(r, \omega) \tag{6.31}$$

Taking the inverse Fourier transform of Eq. (6.31), which maps ω to n, we obtain the desired relation in Eq. (6.28).

The OLA method carries out a discretized version of the operations suggested on the right of Eq. (6.28a). That is, given a discrete STFT $X(n, k)$, the OLA method synthesizes a sequence $y(n)$ satisfying the following equation:

$$y(n) = \left[\frac{1}{W(0)}\right] \sum_{p=-\infty}^{\infty} \left[\frac{1}{N} \sum_{k=0}^{N-1} X(p, k) e^{j2\pi kn/N}\right] \tag{6.32}$$

The term inside the rectangular brackets on the right is an inverse DFT that for each p gives

$$y_p(n) = x(n)w(p - n) \tag{6.33}$$

The expression for $y(n)$ therefore becomes

$$y(n) = \left[\frac{1}{W(0)}\right] \sum_{p=-\infty}^{\infty} x(n)w(p - n) \tag{6.34}$$

which then reduces to

$$y(n) = x(n)\left[\frac{1}{W(0)}\right] \sum_{p=-\infty}^{\infty} w(p - n) \tag{6.35}$$

In Eq. (6.35) we note that $y(n)$ will be equal to $x(n)$ provided

$$\sum_{p=-\infty}^{\infty} w(p - n) = W(0) \tag{6.36}$$

Furthermore, if the discrete STFT has been decimated in time by a factor L, it can be similarly shown that if the analysis window satisfies

$$\sum_{p=-\infty}^{\infty} w(pL - n) = \frac{W(0)}{L} \tag{6.37}$$

then $x(n)$ can be synthesized using the following relation:

$$x(n) = \left[\frac{L}{W(0)}\right] \sum_{p=-\infty}^{\infty} \left[\frac{1}{N} \sum_{k=0}^{N-1} X(pL, k) e^{j2\pi kn/N}\right] \tag{6.38}$$

Equation (6.37) is the general constraint imposed by the OLA method on the analysis window. It requires the sum of all the analysis windows (obtained by sliding $w(n)$ with L-point increments at a time) to add up to a constant, as shown in Fig. 6.20. It is

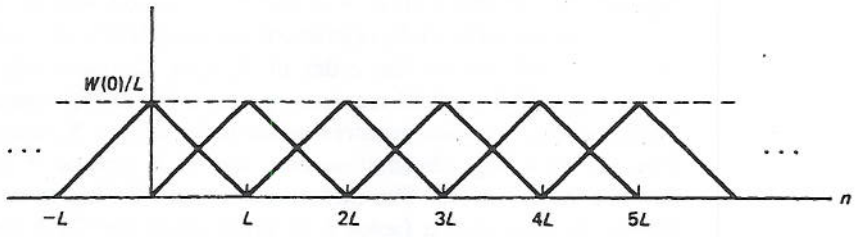

Figure 6.20 The OLA constraint visualized in the time domain.

interesting to note the duality between this constraint and the FBS constraint in Eq. (6.27) where the shifted versions of the Fourier transform of the analysis window were required to add up to a constant. For the FBS method we also saw that all finite-length analysis windows whose length is less than the frequency sampling factor satisfy the FBS constraint. Analogously, we can show that the OLA constraint in Eq. (6.37) is satisfied by all finite-bandwidth analysis windows whose maximum frequency is less than $2\pi/L$, where L is the temporal decimation factor.

To see how finite-bandwidth analysis windows satisfy the OLA constraint, suppose that the analysis window has maximum frequency ω_c, and consequently bandwidth $2\omega_c$, as illustrated in Fig. 6.21. If we let $\hat{w}(p)$ denote the sequence $w(pL - n)$, then the OLA constraint in Eq. (6.37) can be rewritten as

$$\hat{W}(0) = W(0)/L \tag{6.39}$$

where $\hat{W}(\omega)$ denotes the Fourier transform of $\hat{w}(p)$. Noting that $\hat{w}(p)$ is a sampled version of $w(p - n)$, we can easily show that

$$\hat{W}(\omega) = \frac{1}{L} \sum_{k=-\infty}^{\infty} e^{-j(\omega - k2\pi/L)n} W(\omega - k2\pi/L) \tag{6.40}$$

If there is no overlap between $W(\omega)$ and $W(\omega - k2\pi/L)$ at $\omega = 0$, then Eq. (6.40) gives the OLA constraint expressed in Eq. (6.39). To have no overlap at $\omega = 0$ between $W(\omega)$ and $W(\omega - k2\pi/L)$ it is easy to see that we must have $\omega_c < 2\pi/L$, where ω_c is the maximum frequency in $W(\omega)$. We conclude that any finite-bandwidth window whose maximum frequency is less than $2\pi/L$ will satisfy the OLA constraint in Eq. (6.37).

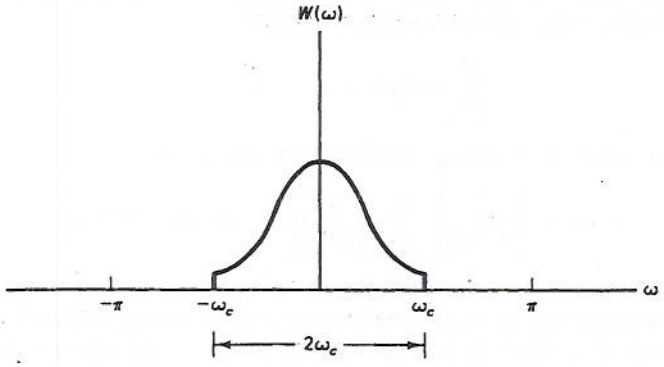

Figure 6.21 Example of a Fourier transform of an analysis window with bandwidth $2\omega_c$.

The OLA method has computational properties that differ from those of the FBS method. Recall that with $L = 1$, the FBS method with an analysis filter of length N_w requires on the order of N_w operations per sample of $x(n)$. In contrast, the OLA method for $L = 1$ requires on the order of N_w Log N_w operations per sample of $x(n)$. For $L \gg 1$, the FBS method requires on the order of N_w^2 operations per sample of $x(n)$, while the OLA method requires on the order of Log N_w operations per sample of $x(n)$. That is, for a large decimation rate, the OLA method is significantly more efficient than the FBS method. The difference is that FBS has to carry out an interpolation to reduce the decimation factor L to unity while the OLA method uses the decimated STFT directly.

6.3.3 Generalized Filter Bank Summation

In this section we present the generalized FBS method, which allows a wider range of analysis windows than the ordinary FBS method discussed in Section 6.3.1. It also includes the OLA method as a special case. The generalized FBS method consists of a two-dimensional smoothing of the discrete STFT followed by the application of the ordinary FBS method.

The generalized FBS method is motivated by the following relation between a sequence and its discrete-time STFT:

$$x(n) = \left[\frac{1}{2\pi w(0)}\right] \int_{-\pi}^{\pi} Y(n, \omega) e^{j\omega n}\, d\omega \tag{6.41a}$$

where

$$Y(n, \omega) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \sum_{r=-\infty}^{\infty} F(n - r, \omega - \phi) X(r, \phi)\, d\phi \tag{6.41b}$$

and $F(n, \omega)$ is an arbitrary smoothing function. It should be noted that Eq. (6.41a) has the same form as the motivating equation (6.20) for the FBS method except that $X(n, \omega)$ has been smoothed to form $Y(n, \omega)$. If we combine Eqs. (6.41a) and (6.41b) and denote by $f(n, m)$ the inverse Fourier transform (mapping ω to m) of $F(n, \omega)$, then we obtain

$$x(n) = \left(\frac{1}{2\pi}\right) \int_{-\pi}^{\pi} \left[\sum_{r=-\infty}^{\infty} f(n, n - r) X(r, \omega)\right] e^{j\omega n}\, d\omega \tag{6.42}$$

where the function $f(n, m)$ is generally referred to as the *synthesis filter*. Note that the motivating equation for ordinary FBS can be obtained by setting the synthesis filter to be a nonsmoothing filter, i.e., $f(n, m) = \delta(m)$ in Eq. (6.41). It can also be easily shown that $f(n, m) = 1/W(0)$ yields the synthesis equation in Eq. (6.28) for the OLA method. In fact, it can be shown [5] that any $f(n, m)$ that satisfies

$$\sum_{m=-\infty}^{\infty} f(n, -m) w(m) = 1 \tag{6.43}$$

will make Eq. (6.41) a valid synthesis equation.

The generalized FBS method carries out a discretized version of the operations suggested by Eq. (6.42). That is, given a discrete STFT that has been decimated in time by a factor L, the generalized FBS method synthesizes a sequence $y(n)$ satisfying the following equation:

$$y(n) = \frac{L}{N} \sum_{i=-\infty}^{\infty} \sum_{k=0}^{N-1} f(n, n - iL) X(iL, k) e^{-j2\pi nk/N} \tag{6.44}$$

Although Eq. (6.44) contains the time-varying synthesis filter $f(n, n - iL)$, in this section we consider only the time-invariant case $f(n, n - iL) = f(n - iL)$ because of its practical importance. For example, we have already seen that the synthesis equations for both the FBS and OLA methods can be described through time-invariant synthesis filters. For a time-invariant synthesis filter, Eq. (6.44) reduces to

$$y(n) = \frac{L}{N} \sum_{i=-\infty}^{\infty} \sum_{k=0}^{N-1} f(n - iL) X(iL, k) e^{-j2\pi nk/N} \tag{6.45}$$

This equation holds only when certain constraints are satisfied by the analysis and synthesis filters as well as the temporal decimation and frequency sampling factors. To find these constraints, we first rewrite Eq. (6.45) as

$$y(n) = L \sum_{i=-\infty}^{\infty} f(n - iL) \frac{1}{N} \sum_{k=0}^{N-1} X(iL, k) e^{-j2\pi nk/N} \tag{6.46}$$

and note that the second summation represents an inverse DFT that evaluates as follows:

$$\frac{1}{N} \sum_{k=0}^{N-1} X(iL, k) e^{-j2\pi nk/N} = \sum_{p=-\infty}^{\infty} x(n - pN) w(iL - n + pN) \tag{6.47}$$

Equation (6.46) therefore becomes

$$y(n) = L \sum_{p=-\infty}^{\infty} \left[\sum_{i=-\infty}^{\infty} f(n - iL) w(iL - n + pN) \right] x(n - pN) \tag{6.48}$$

For the right side of Eq. (6.48) to reduce to $x(n)$, we clearly require the term in rectangular brackets to reduce to $\delta(p)$. This condition can then be stated as

$$L \sum_{i=-\infty}^{\infty} f(n - iL) w(iL - n + pN) = \delta(p) \qquad \text{for all } n \tag{6.49}$$

which is a discretized version of the constraint of Eq. (6.43) for a time-invariant synthesis filter. This constraint essentially states that the product of $f(n)$ and $w(-n)$ should be such that samples that are an integer multiple of L points apart from each other add up to unity. Furthermore, N should be such that whenever $w(-n)$ is shifted by an integer multiple of N, then its product with $f(n)$ should be such that samples that are an integer multiple of L points apart from each other add up to zero. An easy way to satisfy the latter constraint is to set N to be larger than or equal to the analysis window length and then to restrict the synthesis filter length to be the analysis window length.

The constraint in Eq. (6.49) for generalized FBS reduces to the OLA and FBS constraints when the appropriate synthesis window is used for each method. For example, let us consider how the constraint in Eq. (6.49) reduces to the constraint in Eq. (6.27) that we derived for the ordinary FBS method. The synthesis filter for the ordinary FBS method is $f(n, m) = \delta(m) = f(m)$ regardless of the analysis filter. Substituting $f(n) = \delta(n)$ in Eq. (6.48), with $L = 1$, we obtain the synthesis equation

$$y(n) = \sum_{p=-\infty}^{\infty} \left[\sum_{i=-\infty}^{\infty} \delta(n - i) w(i - n + pN) \right] x(n - pN) \tag{6.50}$$

and the corresponding constraint of Eq. (6.49) becomes

$$\sum_{i=-\infty}^{\infty} \delta(n - i) w(i - n + pN) = \delta(p) \tag{6.51}$$

Note that the impulse function on the left side of Eq. (6.51) is nonzero only if $n = i$. It is easy to see that in this case, the left-hand side reduces to $w(pN)$. Therefore, Eq. (6.51) becomes

$$w(pN) = \delta(p) \tag{6.52}$$

This can be rewritten as

$$w(m) \sum_{p=-\infty}^{\infty} \delta(m - pN) = w(0)\delta(m) \tag{6.53}$$

which in the frequency domain is given by

$$\sum_{k=0}^{N-1} W(\omega - 2\pi k/N) = Nw(0) \tag{6.54}$$

This is the same as the condition in Eq. (6.27) for the FBS method. It should also be noted that if $L \neq 1$, and if we let $h(n)$ be the interpolating filter preceding the FBS method, then the synthesis equation (6.45) with the synthesis filter $f(n) = h(n)$ turns out to be equivalent to the entire process of interpolation and filter bank summation.

Let us now reconsider in detail the general FBS method as an implementation of Eq. (6.45). Assuming a finite-length interpolation filter and interchanging the summations, Eq. (6.45) can be rewritten as

$$y(n) = \frac{L}{N} \sum_{k=0}^{N-1} \left[\sum_{i=N_\ell}^{N_u} f(n - iL)X(iL, k) \right] e^{-j2\pi nk/N} \tag{6.55}$$

where N_ℓ and N_u are determined by the region of support of the interpolation filter. The term within the rectangular brackets

$$\text{rec}(n, k) = \sum_{i=N_\ell}^{N_u} f(n - iL)X(iL, k) \tag{6.56}$$

represents the temporal convolution of $f(n)$ and $X(nL, k)$. The computation in Eq. (6.56) has the same general form as the interpolation required before application of the FBS method to the discrete STFT with temporal decimation factor $L > 1$. The remainder of Eq. (6.55) (i.e., outside of $\text{rec}(n, k)$) is identical to the FBS method; it requires modulation followed by summation.

The amount of computation required by the generalized FBS method is linearly proportional to the frequency sampling rate, and it is inversely proportional to the temporal decimation factor. Let N be the frequency sampling factor for the STFT, L the temporal decimation factor for the STFT, and K the length of the interpolation filter. For each sample of $x(n)$ we require N interpolations, followed by a modulation and summation of the resulting N samples. Each interpolation requires on the order of K/L operations. Thus, the N interpolations require a total of NK/L operations. The modulation and summation involves on the order of N operations. Thus the total number of operations required per sample of $x(n)$ is on the order of $(NK/L) + N = N(K + L)/L$ operations. In the next section we will consider another synthesis method that also utilizes the synthesis filter but offers a different computational tradeoff.

6.3.4 Helical Interpolation Method

We will now investigate the helical interpolation synthesis method for implementing Eq. (6.45), which is a discretized version of the STFT synthesis equation (6.42) with a time-invariant synthesis filter. In the previous section we saw how Eq. (6.45) could be implemented as a generalized FBS method. In particular, Eq. (6.45) was interpreted as an interpolation with a synthesis filter followed by a modulation and summation of the interpolated output. In contrast, for the helical interpolation method we interpret Eq. (6.45) as a series of inverse discrete Fourier transforms whose outputs are combined through a sophisticated interpolation procedure generally known as helical interpolation. More specifically, we rewrite Eq. (6.45) as follows:

$$y(n) = L \sum_{i=N_\ell}^{N_u} f(n - iL)\left[\frac{1}{N}\sum_{k=0}^{N-1} X(iL, k)e^{-j2\pi nk/N}\right] \tag{6.57}$$

where we have assumed that the synthesis filter is finite length. The term inside the rectangular brackets in Eq. (6.57) is an inverse DFT that can be efficiently computed using the FFT. Denoting the result of this inverse DFT by $f_{iL}(n)$, we can rewrite Eq. (6.57) as

$$y(n) = L \sum_{i=N_\ell}^{N_u} f(n - iL)\tilde{f}_{iL}(n) \tag{6.58}$$

where $\tilde{f}_{iL}(n)$ represents a periodic extension of the short-time section $f_{iL}(n)$. This equation represents the computation of $y(n)$ for some particular n as the interpolation of the nth sample of each of the $(N_u - N_\ell)/L$ functions $\tilde{f}_{iL}(n)$ centered about n. Since the functions $\tilde{f}_{iL}(n)$ are periodic with period N, it follows that $\tilde{f}_{iL}(n) = \tilde{f}_{iL}(n \text{ modulo } N)$. Thus Eq. (6.58) can be rewritten as

$$y(n) = L \sum_{i=N_\ell}^{N_u} f(n - iL)\tilde{f}_{iL}(n \text{ modulo } N) \tag{6.59}$$

The fact that n modulo N when plotted as a function of n resembles a helical function leads to the name *helical interpolation* for the operation suggested by Eq. (6.59).

The helical interpolation method for short-time synthesis is computationally more efficient than the generalized FBS method for realistic situations. If the temporal decimation factor is L, the frequency sampling factor is N, and the synthesis filter is K points long, then we have already seen that the generalized FBS method requires on the order of $N(K + L)/L$ operations per sample of $x(n)$. On the other hand, with the helical interpolation method we require an FFT with N Log N operations for every L samples of $x(n)$ and K/L operations per sample of $x(n)$ for the helical interpolation. Thus, the helical interpolation method requires a total of $(K + N \text{ Log } N)/L$ operations per sample of $x(n)$. If we hold K and L constant, we see that generalized FBS computation is on the order of N operations per sample while helical interpolation is on the order of N Log N operations per sample. However, for realistic values of K, L, and N, helical interpolation is always preferred. To illustrate this, Fig. 6.22 presents a plot of the number of operations per sample of $x(n)$ as a function of the frequency sampling factor N. For this figure we have selected typical values for L and K as 32

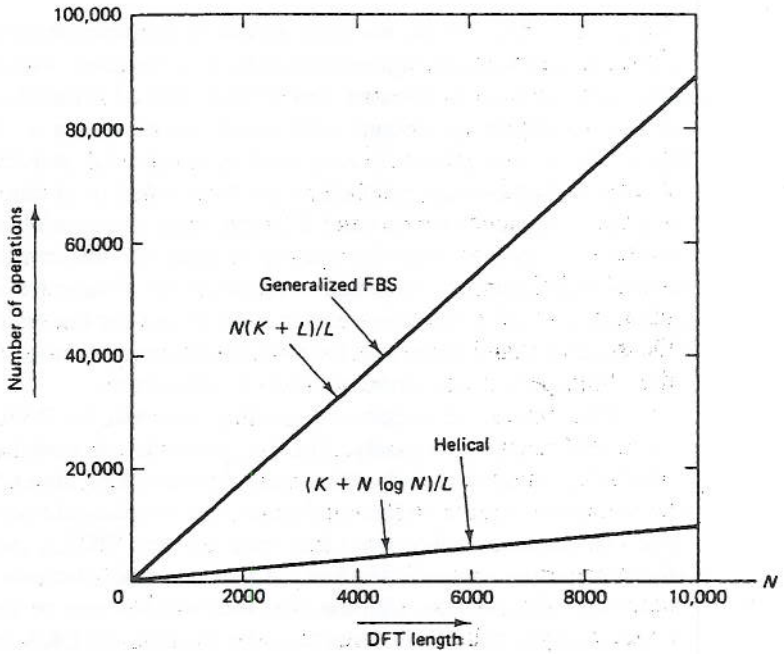

Figure 6.22 Comparison of the number of operations per sample used in the helical interpolation method and the generalized FBS method. It is assumed that the interpolation filter is 256 points long and the decimation factor $L = 32$.

and 256. From the figure we see that helical interpolation is always better than the generalized FBS method for realistic values of N.

6.3.5 Weighted Overlap-Add

We will now investigate a third synthesis method for implementing Eq. (6.45). This method is known as the weighted overlap-add (WOLA) synthesis method and it uses the same number of arithmetic operations as the helical interpolation method. However, the order in which the operations are performed is different for the two methods and thus leads to different space-time tradeoffs.

As with the helical interpolation method, the WOLA method is best viewed in terms of Eq. (6.55) when it is rewritten in the form of Eq. (6.57). The WOLA method, like the helical interpolation method, uses the FFT to compute the inverse DFT represented in Eq. (6.57) by the term in rectangular brackets. If the inverse DFT of $X(iL, k)$ is denoted by $\tilde{f}_{iL}(n)$, then Eq. (6.57) can be written as in Eq. (6.59). The difference between the helical interpolation method and the WOLA method lies in the algorithm used to compute the right side of Eq. (6.59). In the helical interpolation method, the algorithm was based on an interpretation of that equation as a filtering operation with respect to the index i. That is, for each n, $f(n - iL)$ as a sequence in i is multiplied with $\tilde{f}_{iL}(n)$ and the result is summed over all i to obtain $x(n)$ for the particular n. If the interpolation filter is K points long, this means that to compute the

result for a particular n, we need access to approximately K/L short-time sections. This implies a storage requirement of $N_w K/L$ samples, where N_w is the length of each short-time section. In contrast, the WOLA method is based on interpreting Eq. (6.59) as an overlap-add procedure with respect to the index n. In particular, for each i, $f(n - iL)$ as a sequence in n is used to weight the sequence $\tilde{f}_{iL}(n)$. The weighted sequences that overlap each other are then added to obtain $x(n)$ for all n. Since the weighted sequences are at most K points long, the overlap-add procedure requires at most $K - 1$ points from the past to be used for the computation of any particular sample. This implies a storage requirement of K samples in the WOLA method in contrast to $N_w K/L$ samples needed to be stored for the helical interpolation method. On the other hand, since both the methods are implementing the same equation (6.59), they both perform the same number of operations.

It is interesting to note the parallels between the WOLA method and the OLA method discussed previously. The two methods are essentially the same except that the WOLA method includes the extra step of multiplying each short-time section with the synthesis window before performing the overlap-add procedure. The extra step is required because unlike the OLA method, the WOLA method does not place restrictions on the analysis window. In fact, the multiplication in the WOLA method of each short-time section with the synthesis window can be viewed as a transformation on the analysis window in order to make it satisfy the OLA requirement of Eq. (6.37).

6.4 SHORT-TIME FOURIER TRANSFORM MAGNITUDE (STFTM) OF A SEQUENCE

In speech applications, the spectrogram that can be related to the magnitude of the STFT has played a major role. For example, visual cues in the spectrogram have been related to parameters important for speech perception. In fact, it has been suggested [6] that the human ear extracts perceptual information strictly from a spectrogram-like representation of speech. In particular, this representation is a nonnegative time-frequency function. On the other hand, the STFT is generally a complex-valued function and for applications such as time-scale modification of speech, estimation of the phase of this function is computationally difficult [7]. In contrast, a number of techniques have been developed where the processed signal is estimated from only the STFT magnitude (STFTM), thus circumventing the phase estimation problem.

In this section we introduce the magnitude of the STFT as an alternative time-frequency signal representation. We will see in this section that many signals can be represented by the real-valued and nonnegative STFTM. Furthermore, we can develop analysis and synthesis techniques for the STFTM just as we did for the STFT. We will see that while STFTM analysis is similar to STFT analysis, short-time synthesis is very different for the two transforms.

As with the discrete-time STFT, the discrete-time STFTM is symmetric and periodic with period 2π. However, unlike the STFT, the STFTM is a real representation of the signal $x(n)$. The STFTM can be related to another function, the short-time autocorrelation, $r(n, m)$, through the following Fourier relationships:

$$r(n, m) = \frac{1}{2\pi} \int_{-\pi}^{\pi} |X(n, \omega)|^2 e^{j\omega m} \, d\omega \tag{6.60a}$$

$$|X(n, \omega)|^2 = \sum_{m=-\infty}^{\infty} r(n, m) e^{-j\omega m} \tag{6.60b}$$

where $r(n, m)$ is given by the convolution of a short-time section with its time-reversed version:

$$\begin{aligned} r(n, m) &= [x(m)w(m - n)] * [x(-m)w(-m - n)] \\ &= \sum_{p=-\infty}^{\infty} x(p)w(p - n)x(p - m)w(p - m - n) \end{aligned} \tag{6.60c}$$

with $*$ denoting convolution. Generally, the short-time section $x(m)w(m - n)$ cannot be obtained from its short-time autocorrelation function [8]. This relationship between the short-time section, its Fourier transform magnitude, and its autocorrelation is illustrated in Fig. 6.23. However, as we will see shortly, the autocorrelations of short-time sections that have partial overlap in time can be used jointly to solve for the underlying short-time sections. This will enable us under certain conditions to use the STFTM as a unique representation of the underlying signal.

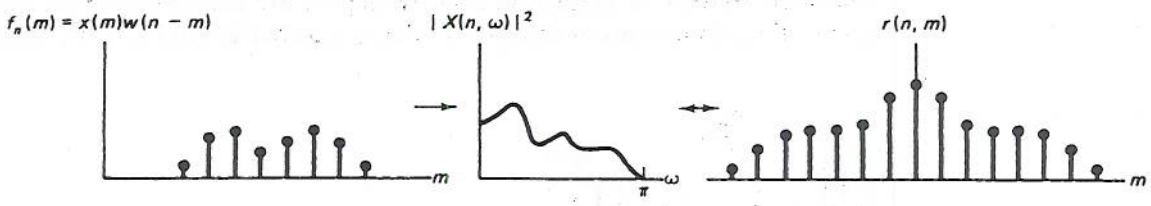

Figure 6.23 Illustration of the noninvertibility of the short-time Fourier transform magnitude for a fixed n. The Fourier transform magnitude is one-to-one with the autocorrelation.

6.4.1 Signal Representation

We will now consider the problem of determining when the discrete-time STFTM can be used to represent a sequence uniquely. That the STFTM is not a unique representation in all cases is easily seen from the simple observation that $x(n)$ and its negative, $-x(n)$, have the same STFTM. We will in fact demonstrate that there are other kinds of situations where the STFTM is not a unique signal representation. We will then proceed to show that by imposing certain mild restrictions on the analysis window and the signal, unique signal representation is indeed possible with the discrete-time STFTM. In particular, if the analysis window is nonzero over its length N_w, then one-sided sequences can be represented uniquely to within a sign factor except for those sequences that have $N_w - 2$ or more consecutive zero samples between two nonzero samples. Furthermore, if we know the sign of just one sample of the sequence to be represented, the STFTM representation becomes completely unique.

To develop insight into the kinds of situations where a sequence cannot be represented uniquely by its discrete-time STFTM, let us consider the case of a

sequence $x(n)$ with a gap of zero samples between two nonzero portions. That is, suppose $x(n)$ is the sum of two signals $x_1(n)$ and $x_2(n)$ occupying different regions of the n-axis as depicted in Fig. 6.24(a). Suppose further that the gap of zeros between $x_1(n)$ and $x_2(n)$ is large enough so that there is no analysis window position for which the corresponding short-time section includes nonzero contribution from $x_1(n)$ as well as $x_2(n)$. Clearly, in such a situation, the STFTM of $x(n)$ is the sum of the STFT magnitudes of $x_1(n)$ and $x_2(n)$. However, we have previously observed that a signal and its negative have the same STFTM. It follows that $x(n)$ has the same STFTM as the signals obtained from the differences $x_1(n) - x_2(n)$ and $x_2(n) - x_1(n)$ shown in Figs. 6.24(b) and (c). We conclude that if there is a large enough gap of zero samples, there will be sign ambiguities on either side of the gap. Consequently, any uniqueness conditions must include a restriction on the length of the zero gaps between nonzero portions of the signal.

Let us now see how a one-sided sequence $x(n)$ can be recovered from its discrete-time STFTM when the analysis window is nonzero over its finite duration and $x(n)$ satisfies the appropriate zero-gap restriction. The key to recovering $x(n)$ is the observation that $|X(n, \omega)|$ has additional information about the short-time sections of $x(n)$ besides their spectral magnitudes. This information is contained in the overlap of the analysis window positions. If the short-time section at time n is known, then the signal corresponding to the spectral magnitude of the adjacent section at time $n + 1$ must be *consistent* in the region of overlap with the known short-time section. In particular, if the analysis window were nonzero and of length N_w, then, as illustrated

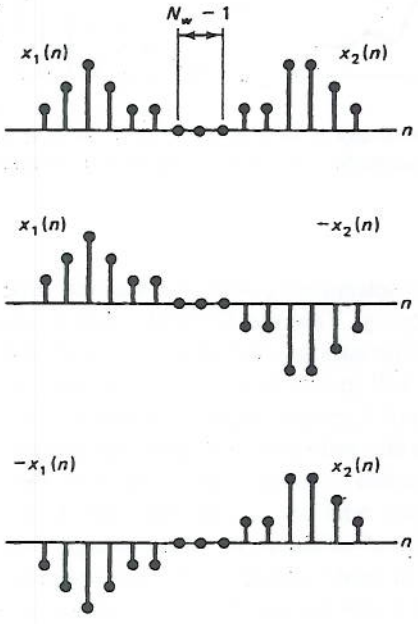

Figure 6.24 Three sequences with the same STFTM.

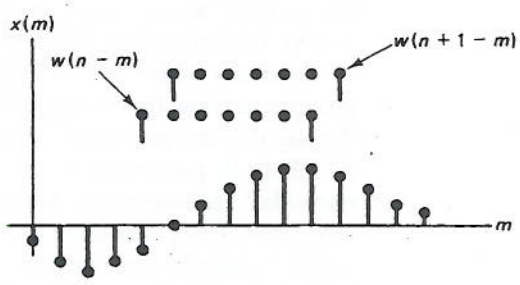

Figure 6.25 Illustration of the consistency required among adjacent short-time sections. Note the samples that are common to the adjacent sections.

in Fig. 6.25, after dividing out the analysis window, the first $N_w - 1$ samples of the segment at time $n + 1$ must equal the last $N_w - 1$ samples of the segment at time n. Therefore, if we could *extrapolate* the last sample of a segment from its first $N_w - 1$ values, we could repeat this process to obtain the entire signal $x(n)$.

To develop the procedure for extrapolating the next sample of a sequence using its STFTM, assume that the sequence $x(n)$ has been obtained up to some time $n - 1$. Thus, as illustrated in Fig. 6.26, the first $N_w - 1$ samples under the analysis window positioned at time n are known. The goal is to compute the sample $x(n)$ from these initial samples and the STFT magnitude $|X(n + 1, \omega)|$ or, equivalently, $r(n, m)$. Note that the last value of the short-time autocorrelation function $r(n, N_w - 1)$ is given by the product of the first and last value of the segment

$$r(n, N_w - 1) = [w(0)x(n)][w(N_w - 1)x(n - (N_w - 1))] \tag{6.61}$$

as illustrated in Fig. 6.27. Therefore $x(n)$ is given by

$$x(n) = \frac{r(n, N_w - 1)}{w(0)w(N_w - 1)x(n - (N_w - 1))} \tag{6.62}$$

If the first value of the short-time section, i.e., $x(n - (N_w - 1))$, happens to equal zero, we must then find the first nonzero value within the section and again use the

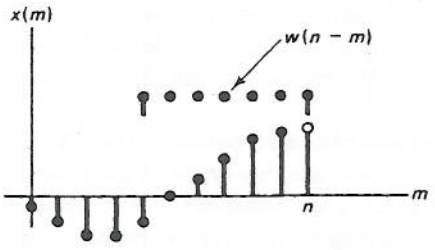

Figure 6.26 Illustration of the samples under $w(n - m)$.

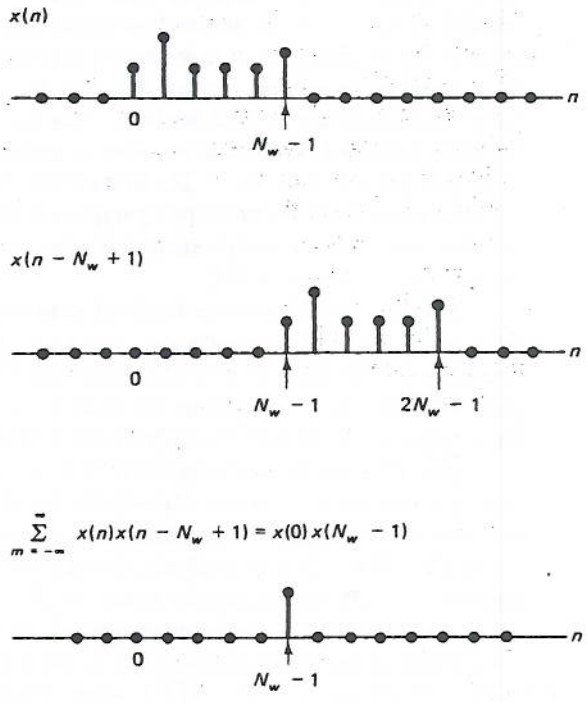

Figure 6.27 Computation of the last nonzero autocorrelation sample (assuming a rectangular analysis window).

product relation given by Eq. (6.62). We can always find such a sample since we have assumed at most $N_w - 2$ consecutive zero samples between any two nonzero samples of $x(n)$.

6.4.2 STFTM Analysis

Like the STFT, the STFTM can be used for analyzing the time-varying spectral characteristics of a sequence. To carry out such STFTM analysis on a digital computer, we need to introduce the *discrete* STFTM. By sampling the frequency dimension of the STFTM, $|X(n, \omega)|$, we obtain the discrete STFTM, which is defined as $|X(n, k)|$, the magnitude of the discrete STFT. In the last section, we saw that under certain conditions, the discrete-time STFTM is a unique signal representation. That theory can be easily extended to the discrete STFTM. In particular, the uniqueness conditions of the previous section relied on using the short-time autocorrelation functions of adjacent short-time sections that are overlapping in time. These autocorrelation functions can be obtained even if the STFTM is sampled in frequency. That is, if the analysis window is N_w points long, then each short-time autocorrelation function is at most $2N_w - 1$ points long and thus can be obtained (without aliasing) from $2N_w - 1$ frequency samples of the STFTM. Therefore, the uniqueness conditions of the discrete-time STFTM extend without change to the discrete STFTM with adequate frequency sampling. To consider the effects of temporal decimation with factor L, we note that adjacent short-time sections now have an overlap of $N_w - L$ instead of $N_w - 1$. The successive extrapolation procedure discussed in the previous section can be extended to this case by requiring the extrapolation of the L last samples of a short-time section, using the first $N_w - L$ samples and the short-time autocorrelation function of that section. This can be accomplished provided the overlap between adjacent short-time sections is greater than $N_w/2$ and there are no zero gaps of length greater than $N_w - 2L$. In addition, to initialize the extrapolation procedure, L initial samples of the underlying sequence have to be known. There are also a variety of other uniqueness conditions that express the tradeoff between time decimation and frequency sampling [9,10].

Having established the kinds of conditions required for the discrete STFTM to be a valid signal representation, we now discuss the various implementation approaches for the discrete STFTM. Since the STFTM and the STFT are closely related, their analysis implementations are similar. In particular, we can use the FFT and filter bank approaches of STFT analysis for STFTM analysis.

The FFT implementation of STFT analysis can be extended to STFTM analysis in a straightforward manner. Recall that the FFT implementation of the discrete STFT involved the computation of a time-aliased version of each short-time section followed by an FFT. For STFTM analysis, we can follow the FFT by the magnitude operation applied to each output sample from the FFT.

The filter bank implementation of STFTM analysis also parallels the STFT implementations. From Section 6.2.3, we know that there are two types of filter bank implementations for the STFT—the lowpass filter and bandpass filter implementations. Each of these implementations has a corresponding implementation for the discrete STFTM. The STFTM lowpass implementation shown in Fig. 6.28 merely

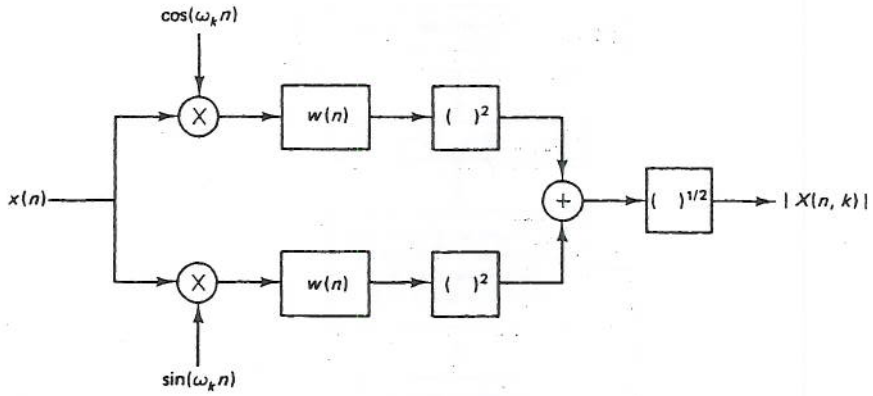

Figure 6.28 Implementation with real operations of each branch of the lowpass filter bank for STFTM analysis.

involves cascading the magnitude operation after the computation of the STFT. However, the bandpass implementation shown in Fig. 6.29 is considerably simplified for the STFTM. This happens because the complex modulation following the bandpass filter is eliminated by the magnitude operation.

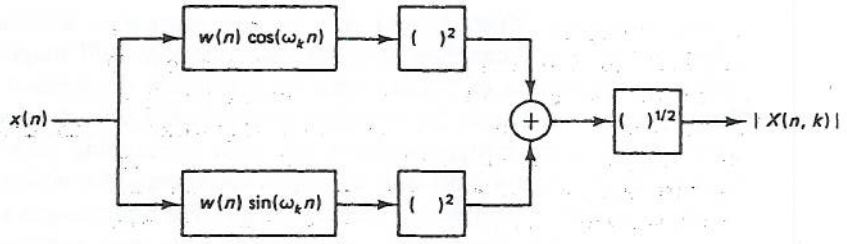

Figure 6.29 Implementation with real operations of each branch of the bandpass filter bank for STFTM analysis.

6.4.3 STFTM Synthesis

A number of methods [9] are available for synthesizing a sequence from its discrete STFTM. These methods are not related in any simple way to the STFT synthesis methods because of the nonlinear mapping from the STFT to the STFTM. In this section, we will discuss only one of these methods as a way of illustrating the basic issues involved.

The synthesis method we will now discuss is based on the *sequential extrapolation approach*, illustrated in Fig. 6.30. For synthesizing a sequence $x(n)$ from $|X(nL, k)|$, we assume that the analysis window $w(n)$ is a known sequence with no zero samples over its finite length. Furthermore, these nonzero samples are in the region $0 \leq n < N_w$. The signal $x(n)$ has no more than $N_w - 2L$ consecutive zeros separating any two nonzero samples. It is also assumed that the first nonzero sample of $x(n)$ falls at $n = 0$. Finally, we assume that the L samples of $x(n)$ for $0 \leq n < L$ are known. The L known samples of $x(n)$ completely determine the short-time section

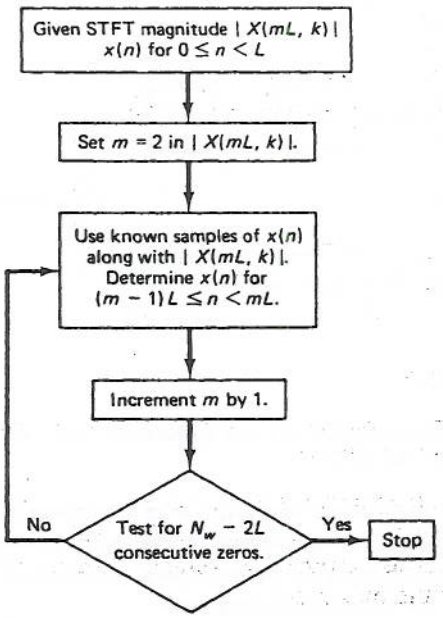

Figure 6.30 The sequential extrapolation approach for STFTM synthesis.

corresponding to $|X(nL, k)|$ for $n = 1$. The short-time section corresponding to $|X(nL, k)|$ for $n = 2$ can then be extrapolated from its DFT magnitude and its known samples in the region of overlap with the previously determined short-time section. This process continues as the complete extrapolation of each new short-time section makes possible the extrapolation of the next overlapping short-time section. The synthesis stops when a short-time section is encountered for which the known samples are not sufficient to complete the extrapolation. For finite-length signals the synthesis stops after all the nonzero short-time sections have been extrapolated.

6.5 SIGNAL ESTIMATION FROM MODIFIED STFT OR STFTM

In many applications it is desired to synthesize a signal from a time-frequency function formed by modifying an STFT or STFTM of a signal we wish to process. Such modifications may arise due to quantization errors in, for example, speech coding or *purposeful* time-varying filtering for signal processing applications such as speech enhancement. An arbitrary function of time and frequency, however, does not necessarily represent the STFT or STFTM of a signal. This is because the definitions of these transforms impose a structure on their time and frequency variations. In particular, because of the overlap between short-time sections, adjacent short-time segments cannot have arbitrary variations. A necessary but not sufficient condition on these variations is that the short-time section corresponding to each time instant must lie within the duration of the corresponding analysis window. For example, the short-time section corresponding to $X(0, \omega)$ is given by $f_0(n) = x(n)w(-n)$ and therefore it must lie within the duration of $w(-n)$. Even if this time-placement constraint is satisfied, a further condition that the STFT or STFTM must satisfy is that adjacent short-time

sections should be consistent in their region of overlap. When the STFT or STFTM of a signal is modified, the resulting time-frequency function does not generally satisfy such constraints. In this section we consider various ways of estimating signals from such arbitrary time-frequency functions.

The synthesis methods we discussed in Sections 6.3 and 6.4 were derived with the assumption that the time-frequency functions to which they are applied satisfy the constraints in the definitions of the STFT or STFTM. Given a function that does not satisfy those constraints, the synthesis methods have no theoretical validity for their application. However, under certain conditions, those methods can be shown to yield *reasonable* results in the presence of modifications. For example, in Section 6.5.1 we will illustrate conditions under which the FBS and OLA methods yield intuitively satisfying results when the STFT has been modified with a multiplicative factor. In Section 6.5.2 we discuss a theoretically based approach to signal synthesis from modified STFT. A similar approach is then discussed in Section 6.5.3 for signal estimation from the modified STFTM.

6.5.1 Heuristic Application of STFT Synthesis Methods

Historically, signal estimation from modified STFT has been performed by applying the FBS and OLA synthesis methods of Section 6.3 on time-frequency functions that are not valid STFT functions. For example, if the valid STFT $X(n, \omega)$ is multiplied by a linear phase factor $e^{j\omega n_0}$ the resulting time-frequency function $Y(n, \omega)$ is not a valid STFT. The reason for this is that the time-placement constraint imposed by the STFT definition is violated by the linear phase modification. In particular, if $w(-n)$ falls between $n = N_1$ and $n = N_2$, then the time-placement constraint requires that the inverse Fourier transform of $Y(0, \omega)$ should fall between $n = N_1$ and N_2. However, because of the linear phase factor the inverse Fourier transform of $Y(0, \omega)$ is shifted by n_0. Even though $Y(n, \omega)$ is not a valid STFT, it is desirable that applying a synthesis method to $Y(n, \omega)$ should yield a reasonable result.

Such heuristic application of the synthesis methods has been practically utilized in many signal processing applications. Since the synthesis methods have no theoretical basis for their application in such situations, it is common to analyze the effects that the methods have on the synthesized signal [11].

In this section, we will contrast the FBS and OLA synthesis methods when they are applied to the STFT that has been modified through multiplication with another time-frequency function. For both methods the resulting synthesized signal can be shown to be a time-varying convolution between $x(n)$ and a function $\hat{p}(n, m)$ as illustrated in Fig. 6.31(a). Let us assume that the STFT $X(n, k)$ has been modified to give the function $Y(n, k)$:

$$Y(n, k) = X(n, k)P(n, k) \tag{6.63}$$

where $P(n, k)$ is the modifying function. Also let $2\pi/N$ be the frequency sampling factor for $X(n, k)$ and let $p(n, m)$ for each n be the N-point inverse DFT of $P(n, k)$. For the FBS method it can be easily shown that $\hat{p}(n, m)$ can be obtained by multiplying $p(n, m)$ for each n by the window function $w(m)$, as shown in Fig. 6.31(b). On the other hand, for the OLA method it can be similarly shown that $\hat{p}(n, m)$ can be obtained

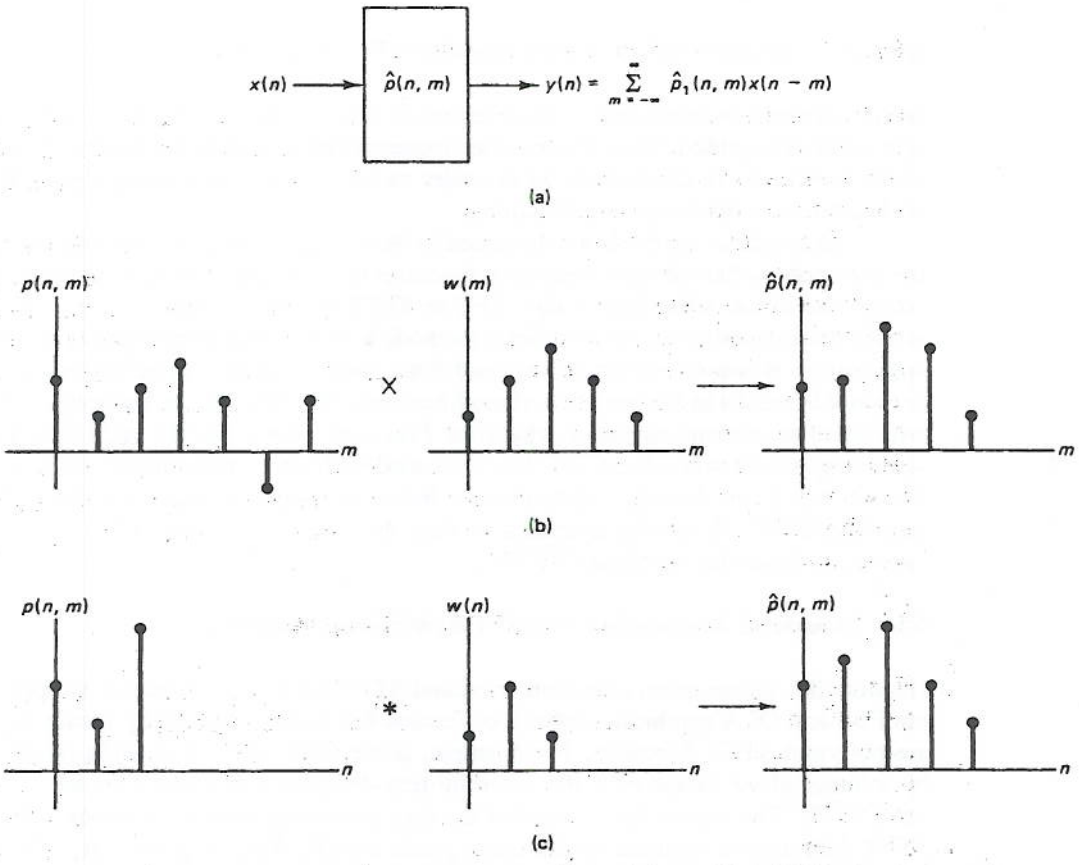

Figure 6.31 Signal synthesis from a multiplicatively modified STFT using FBS and OLA methods. (a) $\hat{p}(m, n)$ relates the original $x(n)$ to the synthesized signal $y(n)$; (b) and (c) illustration of how $\hat{p}(m, n)$ is obtained for FBS and OLA, respectively.

by convolving $p(n, m)$ for each m by the window function $w(n)$, as illustrated in Fig. 6.31(c). It is interesting to note that if $p(n, m)$ is independent of n, i.e., $p(n, m) = p(m)$, then the FBS method results in a synthesized signal that is the convolution of $x(n)$ with $p(n)w(n)$. On the other hand, the time-invariant case for the OLA method results in a synthesized signal that is the convolution of $x(n)$ with $p(n)$.

In this section we have seen how the effects of applying the FBS and OLA methods to the modified STFT may be analyzed for the case of multiplicative modifications. A similar analysis may also be carried out for situations where a time-frequency function has been added to a valid STFT [11].

6.5.2 Least-Squares Signal Estimation from Modified STFT

Rather than applying the FBS and OLA methods in a brute force manner, we will now consider a different approach that is specifically designed for signal estimation from

the modified STFT. In this approach we estimate a signal whose STFT is closest in a least-squares sense to the modified STFT. More specifically, we wish to minimize the mean-square error between the discrete-time STFT $X_e(n, \omega)$ of the signal estimate and the modified discrete-time STFT, which we denote by $Y(n, \omega)$. This optimization results in the following solution for the estimated signal $x_e(n)$:

$$x_e(n) = \frac{\sum_{m=-\infty}^{\infty} w(m - n) f_m(n)}{\sum_{m=-\infty}^{\infty} w^2(m - n)} \tag{6.64}$$

where $f_m(n)$ is the inverse Fourier transform of the frequency variation at time m of the modified STFT $Y(m, \omega)$. Since in practice we have only the discrete function $Y(n, k)$, the short-time sections $f_n(m)$ can be obtained provided the frequency sampling factor N is large enough to avoid aliasing in the short-time sections. The specific distance measure used in the minimization is the squared error between $X_e(n, \omega)$ and $Y(n, \omega)$ integrated over all ω and summed over all n:

$$D[X_e(n,\omega), Y(n, \omega)] = \sum_{m=-\infty}^{\infty} \frac{1}{2\pi} \int_{-\pi}^{\pi} |X_e(m, \omega) - Y(m, \omega)|^2 \, d\omega \tag{6.65}$$

Note that although the distance measure is defined over continuous frequency, the solution for $x_e(n)$ that minimizes the distance measure does not involve continuous frequency. However, it is required that the frequency sampling be large enough so that unaliased versions of the short-time sections are obtained.

The solution in Eq. (6.64) extends in a simple manner to the case involving temporal decimation. Specifically, if L is the temporal decimation factor, then the solution in Eq. (6.64) becomes

$$x_e(n) = \frac{\sum_{m=-\infty}^{\infty} w(mL - n) f_{mL}(n)}{\sum_{m=-\infty}^{\infty} w^2(mL - n)} \tag{6.66}$$

In general, the sum in the denominator of the right side of Eq. (6.66) is a function of n. However, there exist analysis windows $w(n)$ such that the sum in the denominator is independent of n. It should be noted that the sum in the denominator has the same form as the sum in the constraint equation (6.37) for the OLA method except that the analysis window is replaced by its square. That is, any window whose square satisfies the OLA constraint will make the denominator sum in Eq. (6.66) independent of n. If this happens, then Eq. (6.66) can be simply interpreted as an overlap-add operation among the short-time sections corresponding to $Y(n, \omega)$. That is, if the square of the analysis window satisfies the OLA constraint of Eq. (6.37), then the solution for $x_e(n)$ that minimizes Eq. (6.65) is obtained by essentially applying the WOLA synthesis method to the modified STFT. In particular, if the analysis window is rectangular, the WOLA method reduces to the OLA method. We can thus conclude that applying the OLA method with brute force to a modified STFT with a rectangular analysis window will indeed give the solution that minimizes Eq. (6.65).

6.5.3 Least-Squares Signal Estimation from Modified STFTM

The least-squares approach can also be used for signal estimation from the modified STFTM. The resulting method estimates a sequence $x_e(n)$ from a desired time-frequency function $|X_d(n, \omega)|$, which is a modified version of an original STFTM, $|X(n, \omega)|$. The method iteratively reduces the following distance measure between the STFTM $|X_e(n, \omega)|$ of the signal estimate and the modified STFTM $|X_d(n, \omega)|$:

$$D[|X_e(n, \omega)|, |X_d(n, \omega)|] = \sum_{m=-\infty}^{\infty} \frac{1}{2\pi} \int_{-\pi}^{\pi} [|X_e(m, \omega)| - |X_d(m, \omega)|]^2 \, d\omega \qquad (6.67)$$

The solution is found iteratively because as yet no closed-form solution has been discovered for $x_e(n)$ using the distance criterion in Eq. (6.67). The iteration takes place as follows. An arbitrary sequence (usually white noise) is selected as the first estimate $x_e^1(n)$ of $x_e(n)$. We then compute the STFT of $x_e^1(n)$ and modify it by replacing its magnitude by the desired magnitude $|X_d(n, \omega)|$. From the resulting modified STFT, we can obtain a signal estimate using the method based on Eq. (6.64) in the previous section. This process continues iteratively, as shown in Fig. 6.32. In particular, the $(i + 1)$st estimate $x_e^{i+1}(n)$ is first obtained by computing the STFT $X_e^i(n, \omega)$ of $x_e^i(n)$

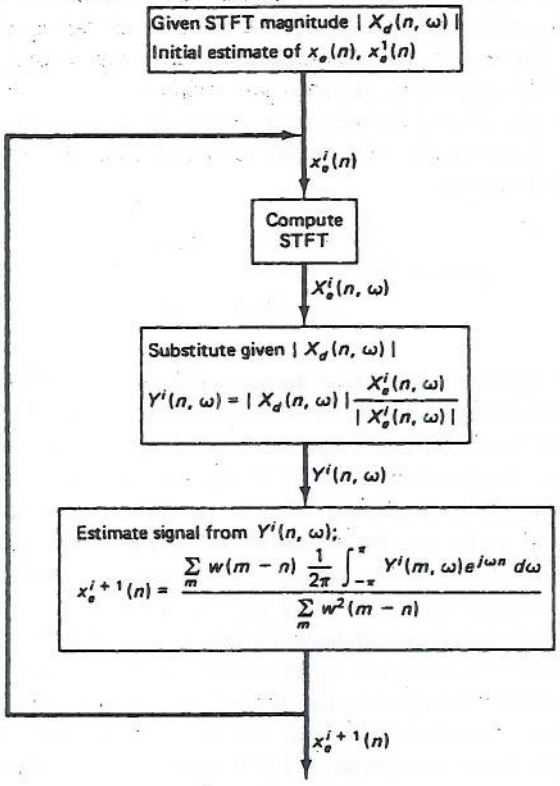

Figure 6.32 Least-squares signal estimation from modified STFTM.

and replacing its magnitude by $|X_d(n, \omega)|$ to obtain $Y^i(n, \omega)$. The signal with the STFT closest to $Y^i(n, \omega)$ is found by using Eq. (6.64). All steps in the iteration can be summarized in the following update equation:

$$x_e^{i+1}(n) = \frac{\sum_{m=-\infty}^{\infty} w(m-n) \frac{1}{2\pi} \int_{-\pi}^{\pi} Y^i(m, \omega) e^{j\omega n} \, d\omega}{\sum_{m=-\infty}^{\infty} w^2(m-n)} \tag{6.68a}$$

where

$$Y^i(m, \omega) = |X_d(m, \omega)| \frac{X_e^i(m, \omega)}{|X_e^i(m, \omega)|} \tag{6.68b}$$

It has been shown [12] that this iterative procedure reduces the distance measure of Eq. (6.67) on every iteration. Furthermore, the process converges to one of the critical points, not necessarily the global minimum of that distance measure. Although we restricted the preceding discussion to the discrete-time STFT, these results are easily extendable to the case where the STFT has been decimated in time. Furthermore, even with discrete frequency the method appears to iteratively reduce the distance measure in Eq. (6.67) provided the frequency sampling factor is sufficiently large to avoid aliasing when determining the short-time sections corresponding to $Y^i(n, \omega)$.

6.6 TIME-FREQUENCY DISTRIBUTIONS

Over the years, various frameworks have been proposed [13,14] for capturing the largely intuitive notions of "instantaneous" or "time-varying" spectra. These frameworks generally have as their central component a signal transform (or "distribution") that maps the original signal into a two-dimensional space with one dimension associated with time and the other with frequency. Clearly, the STFT and STFTM share these characteristics. Furthermore, as we illustrate in this section, the STFT and STFTM are closely related to well-known time-frequency distributions such as the complex energy density (CED) [13], the Wigner distribution [14] and the ambiguity function [15].

The discrete-time CED of a sequence $x(n)$ is defined by

$$E_x(n, \omega) = x(n)X^*(\omega)e^{-j\omega n} \tag{6.69}$$

where * denotes complex conjugation. The CED has a number of properties that are often associated with "instantaneous" spectra. For example, integration along the frequency dimension of the CED gives the instantaneous power $|x(n)|^2$. Conversely, summation along the time dimension of the CED gives the spectral power $|X(\omega)|^2$ at the corresponding frequency in the underlying signal. Furthermore, the CED is invertible to within a sign factor. The basic idea is to let ω be zero in Eq. (6.69). We thus obtain $x(n)X^*(0)$. The magnitude of $X^*(0)$ can be obtained by summing $E_x(n, 0)$ for all n. In summary, $x(n)$ to within a sign factor is represented by

$$y(n) = \frac{E_x(n, 0)}{\sum_{n=-\infty}^{\infty} E_x(n, 0)} \tag{6.70}$$

The CED is related to the STFTM through a convolution process. In particular, it can be shown that the square of the STFTM is equal to the two-dimensional convolution of the CED of the original signal with the CED of the analysis window used in the STFT. For example, if $E_x(n, \omega)$ is the CED of $x(n)$, $E_w(n, \omega)$ is the CED of $w(n)$, and $|X(n, \omega)|$ is the STFTM of $x(n)$, then

$$|X(n, \omega)|^2 = \sum_{k=-\infty}^{\infty} \int_0^{2\pi} E_x(k, v) E_w(n - k, \omega + v)\, dv \tag{6.71}$$

To derive this relationship, we observe that the CED of the convolution of two sequences is equal to the convolution (in time) of the CED of each sequence. That is, if $y(n) = x_1(n) * x_2(n)$, then

$$E_y(n, \omega) = \sum_{k=-\infty}^{\infty} E_{x_1}(k, \omega) E_{x_2}(n - k, \omega) \tag{6.72}$$

Integrating both sides of Eq. (6.72) with respect to frequency, we obtain

$$\int_0^{2\pi} E_y(n, \omega)\, d\omega = \int_0^{2\pi} \sum_{k=-\infty}^{\infty} E_{x_1}(k, \omega) E_{x_2}(n - k, \omega)\, d\omega \tag{6.73}$$

If we let $x_1(n) = x(n)e^{jvn}$ and $x_2(n) = w(n)$, then the left-hand side of Eq. (6.73) becomes the square of the STFTM of $x(n)$, and Eq. (6.73) can be rewritten as

$$|X(n, v)|^2 = \sum_{k=-\infty}^{\infty} \int_0^{2\pi} E_x(k, \omega) E_w(n - k, \omega + v)\, d\omega \tag{6.74}$$

where we have used the fact that the CED of $x(n)e^{j\omega n}$ is $E_x(n, \omega - v)$. Interchanging v and ω, we see that Eq. (6.74) becomes identical to Eq. (6.71). We have thus established the convolutional relationship between the CED and the STFTM. Similar convolutional relationships exist between the STFTM and other time-frequency functions such as the Wigner distribution [14].

It is also interesting to note that the ambiguity function (AF), well known in the radar field, is also closely related to time-frequency functions such as the STFT, the STFTM, and the CED. The ambiguity function (AF) of a sequence $x(n)$ is expressed as

$$A_x(n, \omega) = \sum_{m=-\infty}^{\infty} x(m) x(m - n) e^{j\omega m} \tag{6.75}$$

This function can be viewed as the STFT of $x(n)$ obtained with respect to the analysis window $x(-n)$. Furthermore, the two-dimensional Fourier transform of the AF is the CED of $x(n)$. Since the CED is invertible to within a sign factor, it follows that the same is true for the ambiguity function. We can also show that the two-dimensional Fourier transform of the STFTM of a sequence is equal to the product of the ambiguity functions of the sequence and analysis window, respectively. This follows easily by

taking the two-dimensional Fourier transform of Eq. (6.71) and mapping the CED convolution into a product of ambiguity functions. In fact, this relationship can be used to show that the discrete-time STFTM is invertible to within a sign factor provided the length of the analysis window is longer than that of the signal being represented [16]. In particular, the recovery procedure consists of taking the two-dimensional Fourier transform of the square of the STFTM and dividing the result by the AF of the analysis window. We thus obtain the AF of the original sequence. We can obtain the original sequence (to within a sign ambiguity) from this AF by taking its Fourier transform and using the result in Eq. (6.70). The restriction to long analysis windows results from the division by the analysis window AF in this procedure. For situations involving shorter analysis windows, a different approach such as the one described in Section 6.4 has to be adopted.

6.7 APPLICATIONS

The STFT concepts introduced in this chapter are used in a number of signal processing applications. Most prominent among these applications is speech processing; however, the STFT is also useful in such diverse areas as acoustic beamforming and image restoration. In this section, we present some examples that illustrate the role of the STFT in such applications.

6.7.1 Speech Processing

The STFT has played a major role in speech processing applications such as time-scale modification [7] and bandwidth reduction [17]. In time-scale modification, we are interested in changing the apparent rate of articulation of the original speech while maintaining its perceptual quality. Controls for time-scale modification on a tape recorder, for example, would allow users to rapidly scan large quantities of material or slowly play back difficult to understand speech such as a foreign language. For the blind, this is a particularly encouraging prospect since even normal recorded speech offers a reading rate two to three times that for Braille. In bandwidth reduction, we are also interested in preserving the perceptual quality of the speech in a parsimonious digital representation for limited-bandwidth transmission. To understand why the STFT is suitable for such applications, it is useful to examine the nature of the speech waveform in terms of a simplified model for speech production.

Speech can be modeled as the output of a linear time-varying filter that approximates the transmission characteristics of the vocal tract, as illustrated in Fig. 6.33. The input to the filter is constantly changing. Voiced sounds (e.g., the vowel *e*) are produced by exciting the vocal tract with approximately periodic pulses of airflow caused by vibration of the vocal cords at some time-varying fundamental frequency or "pitch." Unvoiced or what is often referred to as fricative sounds (e.g., *s*) are produced by exciting the vocal tract with a noiselike excitation generated by forcing air through the constricted vocal cavity. Thus the STFT of a voiced sound takes on a harmonic appearance, while the STFT of an unvoiced sound is noiselike, as depicted

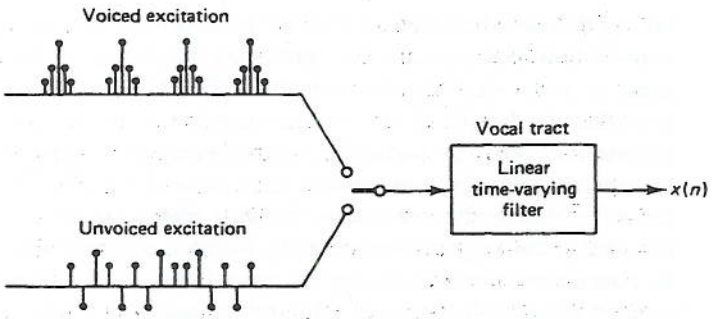

Figure 6.33 Speech production model.

in Fig. 6.34. More generally, the speech production mechanism can generate various combinations of these sounds as, for example, voiced fricatives. Typically, each of these sounds lasts for about 20 ms, during which spectral content of the speech remains stationary except for gradual changes leading into the next sound. Stationarity of spectral content is important for bandwidth reduction, while the rate of change of spectral content is important for time-scale modification. In particular, bandwidth reduction of speech is often achieved by efficiently representing the spectral content of each stationary sound. On the other hand, time-scale modification is obtained by increasing or decreasing the rate at which the spectral content changes from one sound to another.

Various speech researchers have established a relationship between the STFT and time-scale modification of speech. In particular, it is generally agreed that time-scale modification of speech leads to a linear time scaling of the STFTM. The effect of time-scale modification on the STFT phase has also been modeled, but this turns out to be a rather complex and nonlinear effect. However, this effect has been explicitly taken into account in the work of Portnoff [7]. In his technique, the linearly time-scaled STFTM and the nonlinearly processed STFT phase are combined to obtain a modified STFT. This modified STFT is used in the helical interpolation method of Section 6.3.4 to synthesize the desired speech with time-scale modification. An alternative approach to time-scale modification ignores the STFT phase entirely. Instead, the STFTM is linearly time-scaled and used as input to the least-squares technique of Section 6.5.3 for signal estimation from the modified STFTM. The resulting signal has an STFTM that is closest in the least-squares sense to the linearly time-scaled STFTM.

Another common speech application of the STFT is in the area of bandwidth reduction. The digital representation of the speech waveform is often impractical for speech transmission over a limited-bandwidth channel. For example, speech sampled at 8000 samples/s with a 12-bit accuracy contains 96,000 bits/s, and hence a bandwidth requirement too large for most practical channels. The purpose of the vocoder (*voice coder*) is to reduce the bandwidth requirement by reducing the required number of bits transmitted. One approach to bandwidth reduction is to quantize either the discrete STFT or parameters of an STFT model, according to their relative perceptual importance. These quantized values are transmitted and handed to a synthesizer to

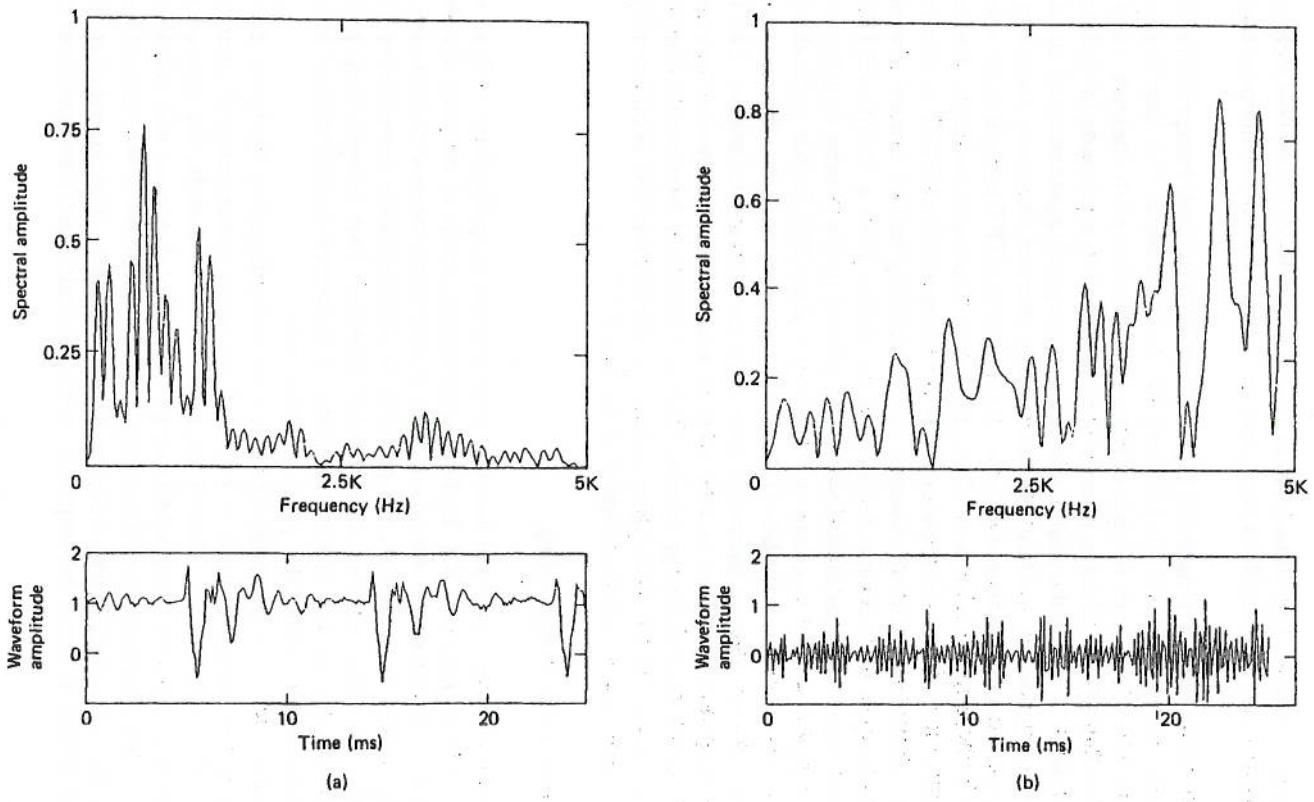

Figure 6.34 Fourier transform magnitudes of (a) typical voiced and (b) typical unvoiced sounds.

obtain a signal estimate. Two illustrative examples of bandwidth reduction techniques are based respectively on the filter bank and the Fourier transform interpretations of the STFT.

A bandwidth reduction method [18] based on the filter bank interpretation of the STFT divides the speech band (0–3.2 kHz) into 4–8 "subbands." A filter bank with a filter at the center frequency of each of these subbands is used to compute a discrete STFT. The output of each of these filters is then quantized (bits per sample) differently for each subband. In particular, subbands of less perceptual importance are quantized more coarsely. Generally, the temporal decimation rate and frequency sampling rate are chosen such that the number of samples in the discrete STFT is equal to the number of samples in the original sequence. Thus, the key to bandwidth reduction lies in the coarse quantization of various subbands. The resulting modified STFT is then transmitted over a limited-bandwidth channel. At the receiving end we can use any of the synthesis techniques in Section 6.5 for signal estimation from a modified STFT.

Another approach to bandwidth reduction is based on the Fourier transform view of the STFT and is known as *adaptive transform coding* [19]. In this method a DFT computation results in typically 64–512 uniform frequency samples, thus requiring a much larger time decimation interval L than does the subband technique. The large number of frequency samples in this method make evident important perceptual information about the underlying speech. This information is then used to adaptively quantize the discrete STFT. The signal estimate at the receiver end is obtained by the weighted overlap-add synthesis method of Section 6.3.5.

6.7.2 Sensor Array Processing

The short-time Fourier transform has also been utilized in applications such as sonar and geophysical exploration [20]. In these applications, arrays of spatially distributed sensors such as microphones are often used to determine various characteristics of propagating waves. In particular, the problem of isolating wave components from a particular direction has received considerable attention. This *beamforming* problem can be addressed in a number of ways, including a frequency-domain method in which the short-time Fourier transform plays a central role.

To isolate the wave components from a particular direction, the basic idea is to delay each sensor signal in such a way that the wave components of interest are time-aligned over all sensors. Thus if we sum the sensor signals, the desired wave components will add coherently while energy from other directions will add incoherently. The signal resulting from this delay-and-sum operation is thus considered an estimate of the wave components from the desired direction. If the signal from the ith sensor is digitized to form the sequence $x_i(n)$, the delay-and-sum operation is expressed as

$$x(n) = \frac{1}{N} \sum_{i=0}^{N-1} x_i(n - n_i) \tag{6.76}$$

where n_i is the delay applied to the ith sensor signal and there are a total of N sensors. For the frequency-domain method of beamforming, we express Eq. (6.76) in the

frequency domain. That is, taking the Fourier transform of both sides of Eq. (6.76), we obtain

$$X(\bar{\omega}) = \frac{1}{N}\sum_{i=0}^{N-1} X_i(\omega)e^{-j\omega n_i} \tag{6.77}$$

In practice, the frequency content of the sensor signals slowly changes as a function of time. For example, the propagating waves may be originating from sources whose directions are slowly changing with respect to the array. To capture this time dependence of the frequencies coming from a particular direction, the frequency-domain beamforming method employs the STFT for each sensor signal instead of the Fourier transform in Eq. (6.77). We thus obtain

$$X_n(\omega) = \frac{1}{N}\sum_{i=0}^{N-1} X_i(n, \omega)e^{-j\omega n_i} \tag{6.78}$$

where $X_i(n, \omega)$ is the STFT of $x_i(n)$ and $X_n(\omega)$ is the final result of the frequency-domain beamforming method. The time-frequency function $X_n(\omega)$ represents the time-varying frequency content of the wave components coming from a particular direction. For digital implementations, the frequency variable is discretized, resulting in the use of the discrete STFT.

6.7.3 Image Processing

Image processing techniques in the frequency domain such as Wiener filtering for noise reduction generally are based on the assumption that the frequency content of an image is the same over all subsections of the image. However, this assumption is typically not valid for many practical images since they usually consist of many different regions with disparate characteristics. To overcome this problem, one approach is to divide the image into smaller sections and process each of them separately; the size of the sections is chosen to be small enough so that most sections consist of a uniform pattern. A convenient mechanism for implementing this idea is a two-dimensional extension of the short-time Fourier transform. This extension of the STFT to a two-dimensional signal $x(n_1, n_2)$ is given by

$$X(n_1, n_2, \omega_1, \omega_2) = \sum_{m_1}\sum_{m_2} x(m_1, m_2)w(n_1 - m_1, n_2 - m_2)e^{-j(\omega_1 m_1 + \omega_2 m_2)} \tag{6.79}$$

where $w(n_1, n_2)$ is the two-dimensional analysis window. As in the case of the one-dimensional STFT, the two-dimensional STFT can also be interpreted as a set of Fourier transforms obtained by sliding the analysis window (in two dimensions now) over the original signal and taking the Fourier transform of the product at each new position. Thus the analysis window helps divide the original image into separate but overlapping sections, each the same size as the analysis window. The convenience of the STFT derives from the fact that after we have processed the separate sections of the image, it is necessary to combine the sections to form the entire processed image. The overlap-add synthesis technique for the STFT is ideal for accomplishing this

purpose. In particular, for the OLA method to succeed we require that the overlapping analysis window positions should sum to a constant over the entire image. A particularly convenient way of generating two-dimensional analysis windows with such a property makes use of one-dimensional windows that satisfy the OLA constraint. In particular, if $w_1(n)$ and $w_2(n)$ are any two analysis windows satisfying the OLA constraint, then it can be shown [21] that the window $w(n_1, n_2) = w_1(n_1)w_2(n_2)$ satisfies the two-dimensional OLA constraint.

REFERENCES

1. R. M. Fano, "Short-Time Autocorrelation Functions and Power Spectra," *J. Acoustical Soc. Amer.*, Vol. 22, pp. 546–550, Sept. 1950.
2. M. R. Schroeder and B. S. Atal, "Generalized Short-Time Power Spectra and Autocorrelation Functions," *J. Acoustical Soc. Amer.*, Vol. 34, pp. 1679–1683, Nov. 1962.
3. W. Koening, H. K. Dunn, L. Y. Lacey, "The Sound Spectrograph," *J. Acoustical Soc. Amer.*, Vol. 18, pp. 19–49, 1946.
4. A. V. Oppenheim, "Speech Spectrograms Using the Fast Fourier Transform," *IEEE Spectrum*, Vol. 7, pp. 57–62, August 1970.
5. M. R. Portnoff, "Representation of Digital Signals and Systems Based on the Short-Time Fourier Transform," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 28, pp. 55–69, Feb. 1980.
6. J. C. Anderson, "Speech Analysis/Synthesis Based on Perception," Ph.D. Thesis, MIT, Cambridge, MA, Sept. 1984.
7. M. R. Portnoff, "Time-Scale Modification of Speech Based on Short-Time Fourier Analysis, *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 30, pp. 374–390, June 1981.
8. A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1983.
9. S. H. Nawab, T. F. Quatieri, and J. S. Lim, "Signal Reconstruction from Short-Time Fourier Transform Magnitude," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 31, pp. 986–998, Aug. 1983.
10. D. Israelevitz, "Some Results on the Time-Frequency Sampling of the Short-Time Fourier Transform Magnitude," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 33, pp. 1611–1613, Dec. 1985.
11. J. B. Allen and L. R. Rabiner, "A Unified Theory of Short-Time Spectrum Analysis and Synthesis," *Proc. IEEE*, Vol. 65, pp. 1558–1564, Nov. 1977.
12. D. Griffin and J. S. Lim, "Signal Estimation from Modified Short-Time Fourier Transform," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 32, pp. 236–243, April 1984.
13. A. W. Rihaczek, "Signal Energy Distribution in the Time and Frequency Domain," *IEEE Trans. Information Theory*, Vol. 10, May 1968.
14. E. Wigner, *Physical Review*, No. 40, p. 749, 1932.
15. A. V. Oppenheim, *Applications of Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1978.

16. R. A. Altes, "Detection, Estimation, and Classification with Spectrograms," *J. Acoustical Soc. Amer.*, Vol. 67, pp. 1232–1246, April 1980.
17. L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, NJ, 1978.
18. R. E. Crochiere, "On the Design of Sub-Band Coders for Low-Bit Rate Speech Communication," *Bell Syst. Tech. J.*, Vol. 65, pp. 747–770, May–June 1977.
19. J. S. Tribolet and R. E. Crochiere, "Frequency Domain Coding of Speech," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 27, pp. 512–530, Oct. 1979.
20. D. E. Dudgeon and R. W. Mersereau, *Multidimensional Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1984.
21. J. S. Lim, "Image Restoration by Short-Space Spectral Subtraction," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. 29, pp. 191–197, Aug. 1980.

7

Two-Dimensional Signal Processing

Jae S. Lim
Massachusetts Institute of Technology

7.0 INTRODUCTION

At a conceptual level, there is a great deal of similarity between two-dimensional (2-D) signal processing and one-dimensional (1-D) signal processing. In 1-D signal processing, the concepts discussed are filtering, Fourier transforms, discrete Fourier transforms, fast Fourier transforms, etc. In 2-D signal processing, we again are concerned with concepts such as filtering, Fourier transforms, discrete Fourier transforms, and fast Fourier transforms. As a consequence, the general concepts that we develop in 2-D signal processing can be viewed, in many cases, as straightforward extensions of the results in 1-D signal processing.

At a more detailed level, however, considerable differences exist between 1-D and 2-D signal processing. One major difference is the amount of data involved in typical applications. In speech processing, an important 1-D signal processing application, speech is typically sampled at a 10-kHz rate and we have 10,000 data points to process in a second. However, in video processing, where processing an image frame is an important 2-D signal processing application, we may have 30 frames/s, with each frame consisting of 500 × 500 pixels (picture elements). In this case, we would have 7.5 million data points to process per second, which is orders of magnitude greater than the case of speech processing. Due to this difference in data rate requirements, the computational efficiency of a signal processing algorithm plays a much more important role in 2-D signal processing, and advances in hardware tech-

Jae S. Lim is with the Research Laboratory of Electronics and the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. This work has been supported in part by the National Science Foundation under Grant ECS-8407285 and in part by the Advanced Research Projects Agency monitored by ONR under contract N00014-81-K-0742.