Chapter 6

Fourier Analysis - Practice

In the previous sections we have developed three frequency representations for the three main types of discrete-time signals; the derivation was eminently theoretical and concentrated mostly upon the mathematical properties of the transforms seen as a change of basis in Hilbert space. In the following sections we will see how to put the Fourier machinery to practical use.

We have seen two fundamental ways to look at a signal: its time-domain representation, in which we consider the values of the signal as a function of discrete time, and its frequency-domain representation, in which we consider its energy and phase content as a function of digital frequency. Both of the two representations contain exactly the same information, as guaranteed by the invertibility of the Fourier transform; yet, from the analysis point of view, we can choose to concentrate on one or the other domain according to what we are specifically looking for. Consider for instance a piece of music; such a signal contains two coexisting perceptual features, *rhythm* and *melody*. Rhythm is determined by the sequence of musical notes which are played: its "natural" domain is therefore the time domain; melody, on the other hand, is determined by the pitch of the notes which are played: since pitch is related to the frequency content of the sound, the natural domain of this feature is the frequency domain.

6.1 The Transforms in Practice

We will recall the DTFT is mostly a theoretical analysis tool; the DTFT can be exactly computed only for a small set of sequences (i.e., those in which the sum in (5.1) can be computed in closed form); yet, these sequences are highly representative and they will be

used over and over to illustrate a prototypical behavior. The DFT[1], on the other hand, is fundamentally a *numerical* tool in that it defines a finite set of operations which can be computed in a finite amount of time; in fact, a very efficient algorithmic implementation of the DFT exists under the name of Fast Fourier Transform (FFT) which only requires on the order of $N\log(N)$ operations to compute the DFT of an N-point vector. The DFT, as we know, only applies to finite-length signals but this is actually fine since in practice all measured signals have finite support; in principle, therefore, the DFT suffices for the spectral characterization of real-world sequences. Since the transform of a finite-length signal and its DTFT are related by (5.30) or by (5.31) according to the underlying model for the infinite-length extension, we can always use the DTFT to illustrate the fundamental concepts of spectral analysis for the general case and then particularize the results for finite-length sequences.

6.1.1 Plotting Spectral Data

The first question we ask ourselves is how to represent spectral data. Since the transform values are complex numbers, it is customary to separately plot their magnitude and their phase; more often than not, we will concentrate on the magnitude only, which is related to the energy distribution of the signal in the frequency domain[2]. For infinite sequences whose DTFT can be computed exactly, the graphical representation of the transform is akin to a standard function graph — again, the interest here is mostly theoretical. Consider now a finite-length signal of length N; its DFT can be computed numerically, and it yields a length-N vector of complex spectral values. These values can be displayed as such (and we obtain a plain DFT plot) or they can be used to obtain the DTFT of the periodic or finite-support extension of the original signal.

Consider for example the length-16 triangular signal $x[n]$ in Figure 6.1-(a); note in passing that the signal is symmetric according to our definition in (3.34) so that its DFT is real. The DFT coefficients $|X[k]|$ are plotted in Figure 6.1-(b); according to the fact that $x[n]$ is a real sequence, the set of DFT coefficients is symmetric (again according to (3.34)). The k-th DFT coefficient corresponds to the frequency $(2\pi/N)k$ and therefore the plot's abscissa extends implicitly from 0 to 2π; this is a little different than what we are used to in the case of the DTFT, where we usually consider the $[-\pi,\pi]$ interval, but it is customary. Furthermore, the difference is easily eliminated if we consider the sequence of $X[k]$ as being N-periodic (which it is, as we showed in Section 3.2) and plot the values

[1] And the DFS, of course, which are formally identical. As a general remark, whenever we talk about the DFT of a length-N signal, the same will hold for the DFS of an N-periodic signal; for simplicity, from now on we will just concentrate on the DFT.

[2] A notable exception is the case of transfer function for digital filters, in which phase information is extremely important; we will study this in the next chapter.

from $-k/2$ to $k/2$ for k even, or from $-(k-1)/2$ to $(k-1)/2$ for k odd.

This can be made explicit by considering the N-periodic extension of $x[n]$ and by using the DFS-DTFT relationship (5.10); the standard way to plot this is as in Figure 6.1-(c). Here the pulse trains, $\tilde{\delta}(\omega - (2\pi/N)k)$, are represented as lines (or arrows) scaled by the magnitude of the corresponding DFT coefficients. By plotting the representative $[-\pi, \pi]$ interval, we can appreciate in full the symmetry of the transform's magnitude.

On the other hand, by considering the finite-support extension of $x[n]$, and by plotting the magnitude of its DTFT, we obtain Figure 6.1-(d). The points in the plot can be computed directly from the summation defining the DTFT (which, for finite-support signals only contains a finite number of terms) and by evaluating the sum over a sufficiently fine grid of values for ω in the $[-\pi, \pi]$ interval; alternatively, the whole set of points can be obtained in one shot from an FFT with a sufficient amount of zero-padding (we will see more about this later). Again, the DTFT of a finite-support extension is just a smooth interpolation of the original DFT points and no new information is added.

6.1.2 Computing the Transform: the FFT

The Fast Fourier Transform, or FFT, is *not* another type of transform but simply the name of an efficient algorithm to compute the DFT. The algorithm, in its different flavors, is so ubiquitous and so important that the acronym FFT is often used liberally to indicate the DFT (or the DFS, which would be more appropriate since the underlying model is that of a periodic signal).

We have already seen in (3.6) that the DFT can be expressed in terms of a matrix vector multiplication

$$\mathbf{X} = \mathbf{W}\mathbf{x},$$

as such, the computation of the DFT requires on the order of N^2 operation. The FFT algorithm exploits the highly structured nature of $\mathbf{W}$ to reduce the number of operations to $N\log(N)$. In matrix form this is equivalent to decomposing $\mathbf{W}$ into the product of a series of matrices with mostly zero or unity elements. The algorithmic details of the FFT will be studied later; we can already state, however, that the FFT algorithm is particularly efficient for data lengths which are a power of two and that, in general, the more prime factors the data length can be decomposed into, the more efficient the FFT implementation.

6.1.3 Cosmetics: Zero Padding

FFT algorithms are tailored to the specific length of the input signal. When the input signal's length is a large prime number or when the FFT algorithms is available only for

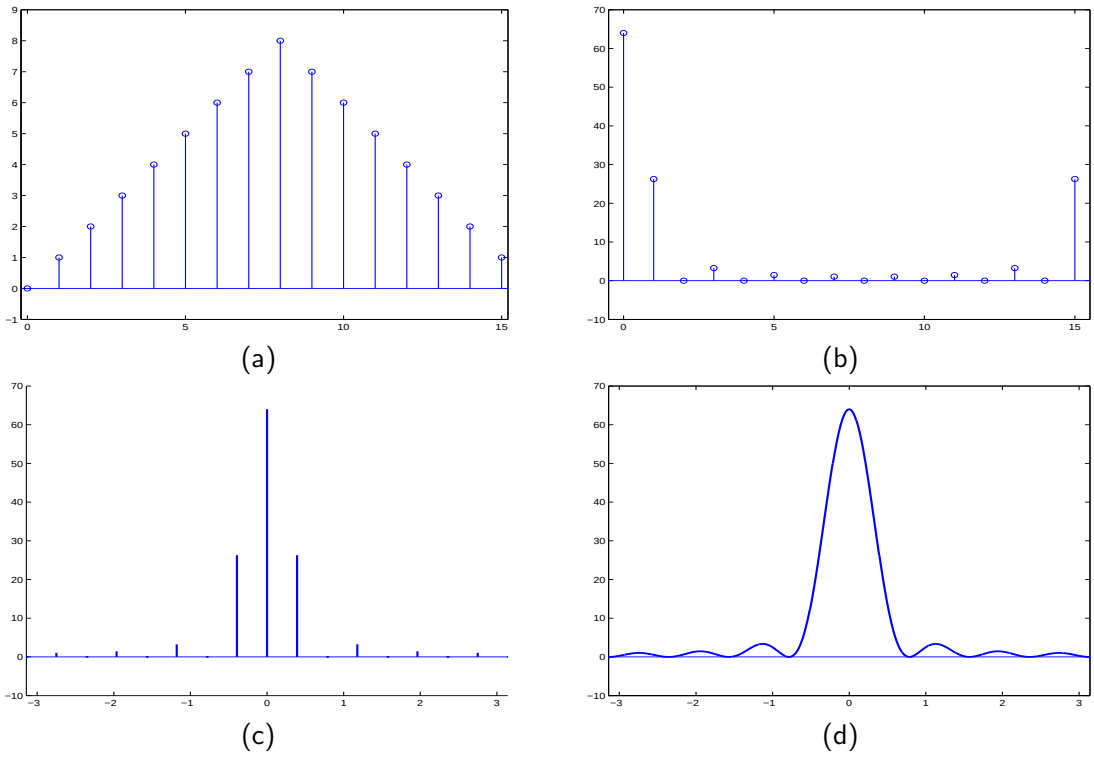

Figure 6.1: Plotting spectral information (all transforms are plotted in magnitude only). (a) Original length-16 signal; (b) Its DFT (or, equivalently, one period of its DFS); (c) The DTFT of its periodic extension; (d) The DTFT of its finite-support extension.

particular lengths (when, for instance, all we have is the radix-2 algorithm, which processes input vector with length a power of two), it is customary to extend the length of the signal to match the algorithmic requirements. This is usually achieved by *zero padding*, i.e., the length-N data vector is extended to a chosen length M by appending $(M - N)$ zeros to it. The maximum resolution of the original N-point DFT, i.e., the separation between frequency components, is $2\pi/N$. By extending the signal to a longer length M, we are indeed reducing the separation between frequency components. One may think that this artificial increase in resolution will allow the DFT to show finer details of the input signal's

spectrum. It is not so.

The M-point DFT $\mathbf{X}^{(M)}$ of an N-point data vector $\mathbf{x}$, obtained via zero-padding, can be obtained directly from the "canonical" N-point DFT of the vector $\mathbf{X}^{(N)}$ via a simple matrix multiplication

$$\mathbf{X}^{(M)} = \mathbf{T}_{M,N}\mathbf{X}^{(N)}. \tag{6.1}$$

Here the $M \times N$ matrix $\mathbf{T}_{M,N}$ is given by

$$\mathbf{T}_{M,N} = \mathbf{W}'_M \mathbf{W}_N^H$$

where $\mathbf{W}_N$ is the standard DFT matrix for length N, and $\mathbf{W}'_M$ is the $M \times N$ matrix obtained by keeping just the first N columns of the standard DFT matrix $\mathbf{W}_M$, the standard DFT matrix for length M. The fundamental meaning of (6.1) is that, by zero padding, we are adding no information to the spectral representation of a finite-length signal. Details of the spectrum which were not apparent in an N-point DFT still won't be apparent in a zero padded version of the same. It can be shown that (6.1) is a form of Lagrangian interpolation of the original DFT samples; therefore the zero-padded DFT will be more attractive in a "cosmetic" fashion since the new points, when plotted, will show a smooth curve between the original DFT points (and this is how plots such as the one in Figure 6.1-(d) are obtained).

6.2 Spectral Analysis

The spectrum is a complete alternative representation of a signal; by analyzing the spectrum one can obtain at a glance the fundamental information to characterize and classify a signal in the frequency domain.

Magnitude The magnitude of a signal's spectrum obtained by the Fourier transform represents the energy distribution of the signal in the frequency domain. It is customary to broadly classify discrete time signals into three classes:

- **Lowpass** (or *baseband*) signals, for which the magnitude spectrum is concentrated around $\omega = 0$ and negligible elsewhere (Figure 6.2-(a)).
- **Highpass** signals, for which the spectrum is concentrated around $\omega = \pi$ and negligible elsewhere, notably around $\omega = 0$ (Figure 6.2-(b)).
- **Passband** signals, for which the spectrum is concentrated around $\omega = \omega_p$ and negligible elsewhere, notably around $\omega = 0$ and $\omega = \pi$ (Figure 6.2-(c)).

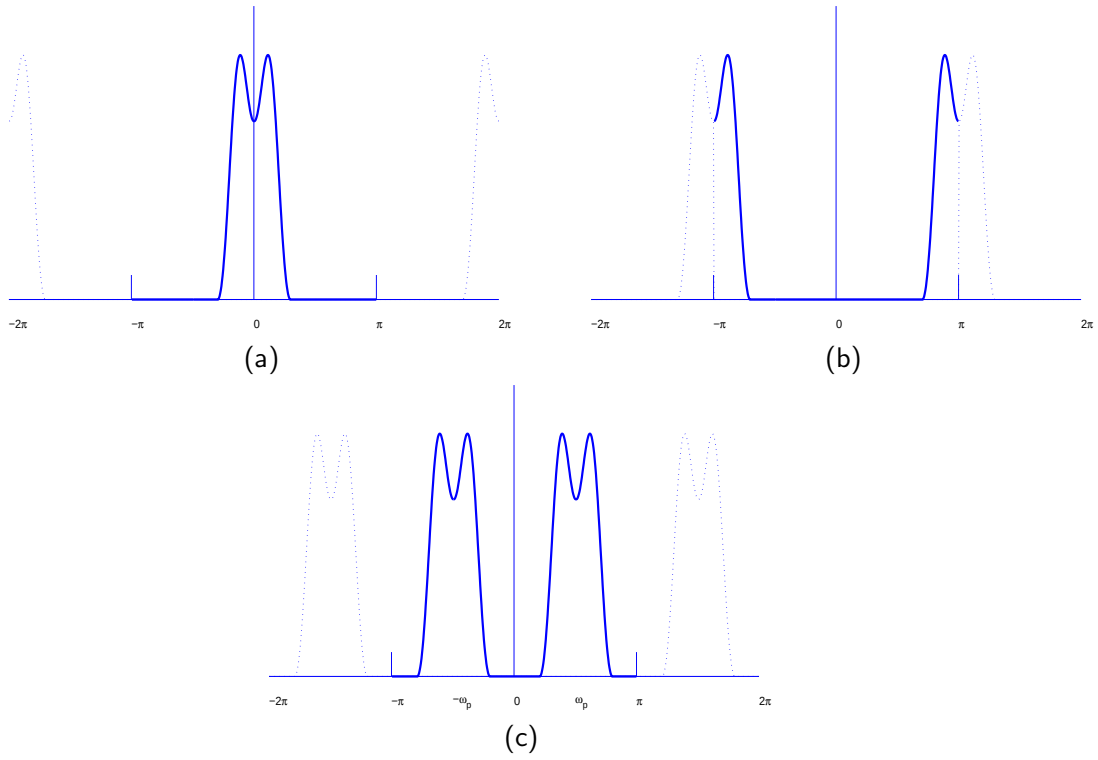

Figure 6.2: Classification of signals based on spectral magnitude. (a) Baseband; (b) Highpass; (c) Passband. Note the 2π-periodicity of the spectrum (the replicas are plotted with a dotted line).

For real-valued signals the magnitude spectrum is a symmetric function and the above classifications take this symmetry into account. Also, all spectra are 2π periodic so that the above definitions are to be interpreted in a 2π-periodic fashion.

Phase As we said before, the Fourier representation allows us to think of any signal as the sum of the outputs of a (potentially infinite) number of sinusoidal generators. While the magnitude of the spectrum defines the inherent power produced by each of the generators, its phase defines the relative alignment of the generated sinusoids. This alignment determines the *shape* of the signal in the discrete-time domain. To illustrate

this with an example, consider the following 64-periodic signal[3]:

$$\begin{aligned}\tilde{x}[n] &= \sum_{k=0}^{3} \frac{1}{2k+1} \sin(\frac{2\pi}{64}(2k+1)n + \phi_k) \\ &= \sin((2\pi/64)n + \phi_0) + (1/3)\sin((2\pi/64)3n + \phi_1) + \\ &\quad (1/5)\sin((2\pi/64)5n + \phi_2) + (1/7)\sin((2\pi/64)7n + \phi_3);\end{aligned}$$

the magnitude of the DFS $\tilde{X}[k]$ is independent of the values of $\phi_0, \ldots, \phi_3$ and it is plotted in Figure 6.3-(a). If we set $\phi_k = 0, k = 0, 1, 2, 3$ we obtain the discrete-time signal which is plotted in Figure 6.3-(b). Now consider modifying the individual phases so that $\phi_k = 2\pi k/3$; the resulting signal is the one depicted in Figure 6.3-(c) for which, as we just noted, the magnitude DFS does not change. This shows that representation of a signal with only the magnitude of its Fourier transform is not complete, and a single magnitude plot can be corresponded to various functions in the time domain.

6.3 Time-Frequency Analysis

Recall our example at the beginning of this chapter, when we considered the time and frequency information contained in a piece of music. We said that the melodic information is related to the frequency content of the signal; obviously this is only partially true, since the melody is determined not only by the pitch values but also by their duration and order. Now, if we take a global Fourier Transform of the entire musical piece we have a *comprehensive* representation of the frequency content of the piece: in the resulting spectrum there is information about the frequency of each played note[4]. The time information, however, that is the information pertaining to the order in which the notes are played, is completely hidden by the spectral representation. This makes us consider whether we can consider a *time-frequency* representation of a signal, in which both time and frequency information are readily apparent.

6.3.1 The Spectrogram

The simplest time-frequency transformation is called the spectrogram. This consists in splitting the signal into small consecutive (and possibly overlapping) length-N pieces and

[3]The signal is the sum of the first four terms of the canonical trigonometric expansion of a square wave of period 64.

[4]Of course, even with the efficiency of the FFT algorithm, the computation of the DFT of an hour-long signal is beyond practical means.

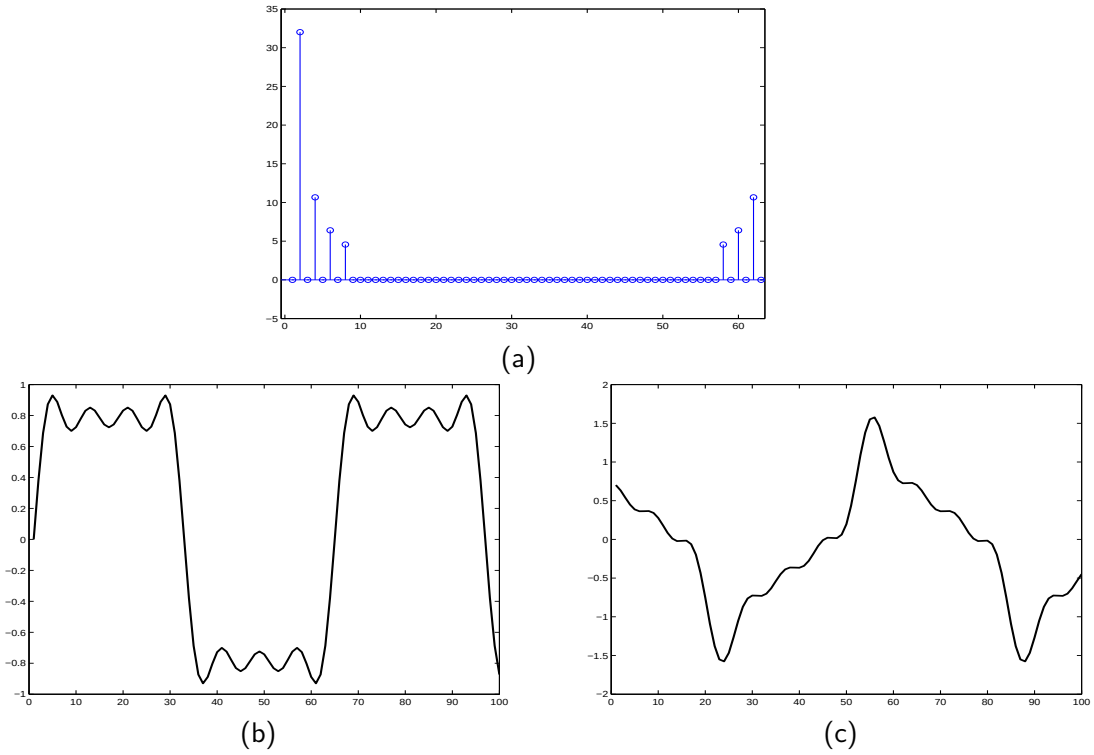

Figure 6.3: Effects of phase shift. (a) The magnitude DFS; (b) The signal with zero phase; (c) The same signal with a linear phase term.

computing the DFT of each. What we obtain is the following function of discrete-time and frequency index:

$$S[k,m] = \sum_{i=0}^{N-1} x[mM+i]W_N^{ik} \tag{6.2}$$

where M, $1 \leq M \leq N$, controls the overlap between segments. In matrix notation we have

$$\mathbf{S} = \mathbf{W}_N \begin{bmatrix} x[0] & x[M] & x[2M] & \cdots \\ x[1] & x[M+1] & x[2M+1] & \cdots \\ \vdots & \vdots & \vdots & \cdots \\ x[N-1] & x[M+N-1] & x[L] & \cdots \end{bmatrix} \tag{6.3}$$

The resulting spectrogram is therefore an $N \times \lfloor L/M \rfloor$ matrix, where L is the total length of the signal $x[n]$. It is usually represented graphically as a plot in which the x-axis is the discrete-time index m, the y-axis is the discrete frequency index k and a color is the magnitude of $S[k, m]$, with darker colors for larger values.

As an example of the insight we can gain from the spectrogram, consider analyzing the well-known "Bolero" by Ravel. Figure 6.4 shows the spectrogram of the initial 37 seconds of the piece. In the first 13 seconds the only instrument playing is the snare drum, and the vertical line in the spectrogram represent at the same time the wide frequency content of a percussive instrument and its rhythmic pattern: if we look at the spacing between lines, we can identify the "trademark" drum pattern of Ravel's Bolero. After 13 seconds, the flute starts playing the theme; this is identifiable in the dark horizontal stripes which denote a high energy content around the frequencies which correspond to the pitches of the melody; with further analysis we could even hope to identify the exact notes. The clarity of this plot is due to the simple nature of the signal; if we now plot the spectrogram of the last 20 seconds of the piece, we obtain Figure 6.5. Here the orchestra is playing full blast, as indicated by the high energy activity across the whole spectrum; we can barely detect the rhythmic shouts that precede the final chord.

6.3.2 The Uncertainty Principle

Each of the columns of $\mathbf{S}$ represents the "local" spectrum of the signal for a time interval of length N. We can therefore say that the *time resolution* of the spectrogram is N samples since the value of the signal at time n_0 will influence the DFT of the N-point window around n_0. Seen from another point of view, the time information is "smeared" over an N-point interval. At the same time, the *frequency resolution* of the spectrogram is $2\pi/N$ (and we cannot increase it by zero-padding, as we just showed). The conflict is therefore apparent: if we want to increase the frequency resolution we need to take longer windows but by doing so we lose the time localization of the spectrogram; likewise, if we want to achieve a fine resolution in time, the corresponding spectral information for each 'time slice" will be very coarse. It is easy to show that the amount of overlap does not change the situation. In practice we will have to choose an optimal tradeoff taking the characteristics of the signal into consideration.

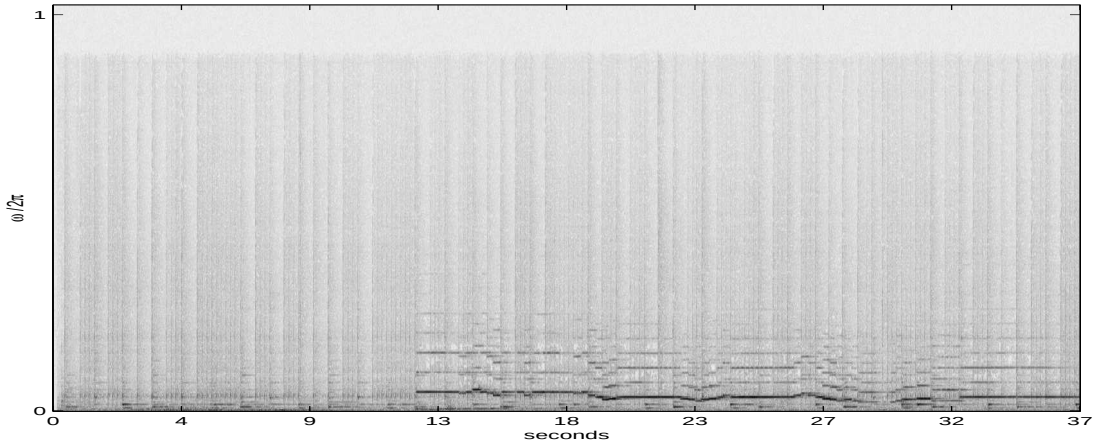

Figure 6.4: Spectrogram representation of the beginning of Ravel's Bolero. DFT size is 1024 samples, overlap is 512 samples.

The above problem, described for the case of the spectrogram, is actually a particular instance of a general uncertainty principle for time-frequency analysis. The principle states that, independently of the analysis tools we put in place, we can never hope to achieve arbitrarily good resolution in both time and frequency since there exists a lower bound greater than zero for the product of the localization measure in time and frequency.

6.4 Digital Frequency vs. Real Frequency

We have seen that, in the representation of discrete-time signals, the notion of a dimensionless discrete "time" makes the whole ensemble of signal processing proofs and tools independent of the underlying physical nature that the signals represent. Similarly, we have just derived a frequency representation for signals which is based on a notion of frequency dimensionless; because of the periodicity of the Fourier basis, all we know is that π is the highest digital frequency we can represent. Again, the power of generality is (or will soon be) apparent: a digital filter which is designed to remove the upper half of a signal's spectrum can be used with any type of input sequence and the results will stay the same. This abstraction, however, is not without its drawbacks from the point of view of intuition; after all, we are very familiar with signals in the real world for which time is expressed in seconds and frequency is expressed in Hertz. We say for instance that speech has a bandwidth up to 4KHz, that the human ear is sensitive to frequencies

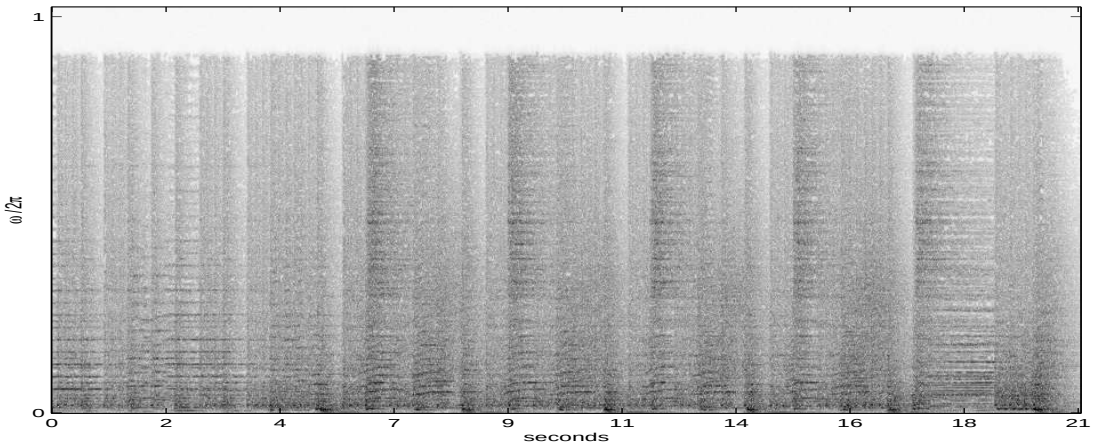

Figure 6.5: Spectrogram representation of the end of Ravel's Bolero.

up to 20KHz, that a cell phone transmits in the GHz band, and so on. What does "π" mean in these cases? The precise, formal link between real-world signal and discrete-time signal processing is given by the sampling theorem, which we will study later. The fundamental idea, however, is that we can remove the abstract nature of a discrete-time signal (and, correspondingly, of a dimensionless frequency) *by associating a time duration to the interval between successive discrete-time indices in the sequence.*

Let us say that the "real-world" time between indices n and $n+1$ in a discrete-time sequence is T seconds; this could correspond to sampling a signal every T seconds or to generating a synthetic sequence with a DSP chip whose clock cycle is T seconds. Recall that the phase increment between successive samples of a generic complex exponential $e^{j\omega n}$ is ω radians. The oscillation will therefore complete a full cycle in $n_f = (2\pi/\omega)$ samples. If T is the real-world time between samples, the full cycle will be completed in $n_f T$ seconds and so its frequency will be $f = (n_f T)^{-1}$. The relationship between the digital frequency ω and the "real" frequency f in Hertz as determined by the "clock" period T is therefore:

$$f \overset{T}{\longleftrightarrow} \frac{1}{2\pi}\frac{\omega}{T}. \tag{6.4}$$

In particular, the highest real frequency which can be represented in the discrete-time system (which corresponds to $\omega = \pi$) is

$$F_{\max} = F_s/2,$$

where we have used $F_s = (1/T)$; F_s is nothing but the operating frequency of the discrete time system (also called the *sampling frequency* or clock frequency). With this notation,

the digital frequency ω_0 corresponding to a real frequency f_0 is

$$\omega_0 = 2\pi \frac{f_0}{F_s}.$$

The compact disk system, for instance, operates at a frequency $F_s = 44.1$KHz; the maximum representable frequency for the system is 22.05KHz (which constitutes the highest-pitched sound which can be encoded on and reproduced by a CD).

6.5 Problems

Problem 6.1 (Fast Fourier Transform) *Let $W_N = e^{j\frac{2\pi}{N}}$. Then, one can write the DFT as*

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}$$

for $0 \leq n \leq N-1$.

1. *To compute $X(0)...X(N-1)$, how many complex multiplications and additions do you have to perform (as a function of N) using the formula above?*

2. *Let $N = LM$. Instead of storing $x(n)$ in a vector, we now store store it in a table such that $x(l,m) = x(l+mL)$ as shown in Table 6.1.*

Table 6.1: Table representation of a sequence

l m	0	1	...	M-1
0	x(0)	x(L)	...	
1	x(1)	x(L+1)		
2	...			
...				
L-1				x(LM-1)

Similarly, let $X(p,q) = X(Mp+q)$. Show that:

$$X(p,q) = \sum_{l=0}^{L-1} \left\{ W_N^{lq} \left[\sum_{m=0}^{M-1} x(l,m) W_M^{mq} \right] \right\} W_L^{lp}$$

3. *One can decompose the above operation as follows:*

- $F(l,q) = \sum_{m=0}^{M-1} x(l,m) W_M^{mq}$
  ($0 \leq q \leq M-1$, for each of the rows $l = 0...L-1$)
- $G(l,q) = W_N^{lq} F(l,q)$
  ($0 \leq l \leq L-1$ and $0 \leq q \leq M-1$)
- $X(p,q) = \sum_{l=0}^{L-1} G(l,q) W_L^{lp}$
  ($0 \leq p \leq L-1$, for each column $q = 0...M-1$)

How many complex multiplications and additions do you now need to compute the Fourier transform (as a function of N,M and L)? Compare the answers in in point 1 and this question when N=1000, L=2 and M=500.

Chapter 7

Linear Systems

7.1 Definition and Properties

In its most general form, a *discrete-time system* can be described as a black box accepting a number of discrete-time sequences as inputs and producing another number of discrete-time sequences at its output. In this course we are interested in studying the class of *linear time-invariant* (LTI) discrete-time systems with a single input and a single output; a system of this type will be referred to as a *filter*. A linear time-invariant systems $\mathcal{H}$ can thus be viewed as an operator which transforms an input sequence into an output sequence:

$$y[n] = \mathcal{H}\{x[n]\}.$$

Linearity is expressed by the equivalence

$$\mathcal{H}\{ax_1[n] + bx_2[n]\} = a\mathcal{H}\{x_1[n]\} + b\mathcal{H}\{x_2[n]\} \tag{7.1}$$

for any two sequences $x_1[n]$ and $x_2[n]$ and any two scalars $a, b \in \mathbb{C}$. Time-invariance is expressed by

$$y[n] = \mathcal{H}\{x[n]\} \Leftrightarrow \mathcal{H}\{x[n - n_0]\} = y[n - n_0] \tag{7.2}$$

For a linear time-invariant system, knowledge of the system response to the input $\delta[n]$ is sufficient to completely characterize the system; $\mathcal{H}\{\delta[n]\}$ is called the *impulse response* of the system. Indeed, we know that for any sequence we can always write the canonical orthonormal expansion (i.e. the famous reproducing formula)

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n - k]$$

and therefore, if we let $\mathcal{H}\{\delta[n]\} = h[n]$, we can apply (7.1) and (7.2) to obtain

$$y[n] = \mathcal{H}\{x[n]\} = \sum_{k=-\infty}^{\infty} x[k]h[n-k]. \tag{7.3}$$

The above sum is called the *convolution* of sequences $x[n]$ and $h[n]$ and will be denoted by the convolution operator "$*$":

$$y[n] = x[n] * h[n].$$

Clearly, for the convolution of two sequences to exist, the sum in (7.3) must be finite and this is always the case if both sequences are absolutely summable. As in the case of the DTFT, absolute summability is just a sufficient condition and the sum (7.3) can be well defined in certain other cases as well. A few notes on the impulse response:

- Since the impulse response is defined as the transformation of the discrete-time delta and since the delta is an infinite-length signal, *the impulse response is always an infinite-length signal*, i.e. a sequence. From now on, except when otherwise indicated, we will assume any impulse response to be at least in $l_2(\mathbb{Z})$; sometimes, we will also need absolute summability.

- When the impulse response is nonzero only for a finite number of sequence indices, i.e. when the impulse response is a finite-support sequence, the resulting filter is called a *Finite Impulse Response* filter (FIR). In all other cases the filter is called *Infinite Impulse Response* (IIR).

- The nonzero values of a filter's impulse response are often called *taps.* An FIR filter always has a finite number of taps.

- The convolution is commutative since, with a change of variable, (7.3) becomes:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] = \sum_{k=-\infty}^{\infty} h[k]x[n-k].$$

- For FIR filters, the convolution sum entails only a finite number of operations; if $h[n] = 0$ for $n < 0$ and $n \geq N$, the above expression becomes simply

$$y[n] = \sum_{k=0}^{N-1} h[k]x[n-k].$$

  Convolution sums involving a finite-support impulse response, therefore, are always well defined. We express this also by saying that FIR filter are *unconditionally stable.*

- To make the notation and the derivations easier, in the following we will assume that filter impulse responses are real-valued sequences.
- Sometimes, to indicate the value of the convolution at a particular time index n_0, we will write $y[n_0] = (x * y)[n_0]$

7.1.1 Properties of the convolution

The basic properties of the convolution operator are:

- Linearity:

$$x[n] * (\alpha \cdot y[n] + \beta \cdot w[n]) = \alpha \cdot x[n] * y[n] + \beta \cdot x[n] * w[n] \tag{7.4}$$

- Time-invariance:

$$w[n] = x[n] * y[n] \;\Leftrightarrow\; x[n] * y[n-k] = w[n-k] \tag{7.5}$$

- Commutativity: (Figure 7.1)

$$x[n] * y[n] = y[n] * x[n] \tag{7.6}$$

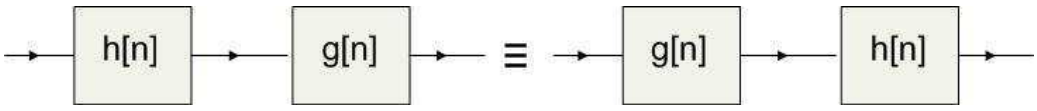

Figure 7.1: Commutativity

- Associativity:

$$x[n] * (y[n] * w[n]) = (x[n] * y[n]) * w[n]. \tag{7.7}$$

This last property describes the effect of connecting two filters in cascade; the resulting impulse response is the convolution of the impulse responses. Note however that the property does not hold for sequences which are not square summable. A classic counterexample is the following: if you take the three sequences

$x[n] = u[n]$	*the unit step*
$y[n] = \delta[n] - \delta[n-1]$	*the first-difference operator*
$w[n] = 1$	*a constant signal*

where clearly $x[n], w[n] \notin l_2(\mathbb{Z})$, it is easy to verify that

$$\begin{aligned} x[n] * (y[n] * w[n]) &= 0 \\ (x[n] * y[n]) * w[n] &= 1. \end{aligned}$$

7.1.2 The meaning of the convolution

It is immediate to see that for two sequences $x[n]$ and $h[n]$ it is

$$x[n] * h[n] = \langle h^*[n-k],\ x[k] \rangle;$$

that is, the value at index n of the convolution of two sequences is the inner product (in $l_2(\mathbb{Z})$) of the first sequence – conjugated[1], time-reversed and re-centered at n – with the input sequence. The above expression describes the output of a filtering operation as a series of "localized" inner products; filtering, therefore, measures the localized similarity (in the inner product sense, i.e. in the sense of the correlation) between the input sequence and a prototype sequence (the time-reversed impulse response).

In general, the convolution operator for a signal is defined with respect to the inner product of its underlying Hilbert space. For the space of N-periodic sequences, for instance, the convolution is defined as

$$\tilde{x}[n] * \tilde{y}[n] = \sum_{k=0}^{N-1} \tilde{x}[k]\tilde{y}[n-k] \tag{7.8}$$

$$= \sum_{k=0}^{N-1} \tilde{x}[n-k]\tilde{y}[k] \tag{7.9}$$

which is consistent with the inner product definition in (4.55).

7.1.3 Convolution of frequency spectrum

We can also consider the convolution of DTFT's. In this case, since we are in the space of 2π-periodic functions of a real variable, the convolution is defined as

$$X(e^{j\omega}) * Y(e^{j\omega}) = \frac{1}{2\pi}\langle X^*(e^{j(\omega-\sigma)}),\ Y(e^{j\sigma}) \rangle \tag{7.10}$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j(\omega-\sigma)})Y(e^{j\sigma})d\sigma \tag{7.11}$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})Y(e^{j(\omega-\sigma)})d\sigma \tag{7.12}$$

which is consistent with the inner product definition for $L_2[-\pi, \pi]$ signals in (4.30).

[1] Since we consider only real impulse responses, the conjugation operator is in this case redundant.

7.2 Circular convolution

We can extend the thought process of convolution as inner products and define it for periodic sequences *over a period* rather than throughout. That is, we define the circular convolution as

$$\tilde{x}[n] \circledast \tilde{y}[n] = \sum_{k=0}^{N-1} \tilde{x}[k]\tilde{y}[n-k], \tag{7.13}$$

for periodic sequences $\tilde{x}[n], \tilde{y}[n]$. But we can extend it to any finite-length sequence by taking its periodic extension, i.e.

$$x[n] \circledast [n] = \sum_{k=0}^{N-1} x[k]y[(n-k)_N], \tag{7.14}$$

where

$$(n-k)_N = \begin{cases} n-k & n-k \geq 0 \\ N+(n-k) & n-k < 0. \end{cases} \tag{7.15}$$

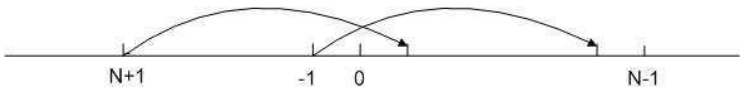

Figure 7.2: Circular convolution

7.3 Stability

A system is bounded-input, bounded-output (BIBO) stable, if its output is bounded for all bounded input sequences. That is, if $x[n]$ is such that there exists a constant $A \in \mathbb{R}^+$ for which

$$|x[n]| < A \qquad \forall\ n,$$

then we want there to exist a constant $B \in \mathbb{R}^+$ such that,

$$|y[n]| = |h[n] * x[n]| = |(h * x)[n]| < B \qquad \forall\ n.$$

A necessary and sufficient condition for BIBO stability is that $h[n]$ (its impulse response) is absolutely summable, i.e.

$$\sum_{k=-\infty}^{+\infty} |h[k]| < C < \infty, \tag{7.16}$$

for some $C \in \mathbb{R}^+$. The sufficiency can be seen by noticing

$$\begin{aligned} |y[n]| &= \left| \sum_{k=-\infty}^{+\infty} h[k]x[n-k] \right| \\ &\overset{(a)}{\leq} \sum_{k=-\infty}^{+\infty} |h[k]|\ |x[n-k]| \\ &\overset{(b)}{<} A \sum_{k=-\infty}^{+\infty} |h[k]| \\ &\overset{(c)}{<} A \cdot C \triangleq B < \infty, \end{aligned}$$

where (a) follows due to the property of complex numbers that $|a+b| \leq |a| + |b|$, and (b) follows from bounded input assumption and (c) from absolute summability of $h[n]$. The necessity is seen by considering

$$x[n] = \begin{cases} \frac{h^*[-n]}{|h[-n]|} & \text{if } h[-n] \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

Since,

$$|x[n]| = \frac{|h^*[-n]|}{|h^*[n]|} = 1 < \infty,$$

then,

$$y[0] = \sum_{k=-\infty}^{+\infty} x[k]h[0-k] = \sum_{k=-\infty}^{+\infty} \frac{h^*[-k]}{|h[-k]|} h[-k] = \sum_{k=-\infty}^{+\infty} |h[-k]| = \sum_{k=-\infty}^{+\infty} |h[k]|\,,$$

which means that if $h[n]$ is not absolutely summable, then $y[0]$ is unbounded and hence the system is BIBO unstable.

7.3.1 Causality

A system is called *causal* if its output does not depend on future values of the input. For an LTI system this implies that the associated impulse response is zero for negative indices.

$$h[n] = 0, \quad \text{for } n < 0, \tag{7.17}$$

since for such a system

$$y[n] = \sum_{k=-\infty}^{+\infty} h[k]x[n-k] = \sum_{k=0}^{\infty} h[k]x[n-k].$$

Therefore $y[n]$ depends only on $x[n], x[n-1], \ldots$

More generally, consider a filter $\mathcal{F}$ for which there exists an $M \in \mathbb{Z}$ such that its impulse response is zero for $n < M$. If we consider the *pure delay* filter $\mathcal{D}$, whose impulse response is

$$d[n] = \delta[n-1],$$

we can easily see that $\mathcal{F}$ can be made strictly causal by cascading M delays in front of it. Clearly, an FIR filter is always causal up to a delay.

Example 7.1 (Linearity and Time Invariance)

For each of the following systems, determine if they are time variant or time invariant, linear or non-linear, causal or not causal.

(a) $y\,[n] = nx\,[n]$

(b) $y\,[n] = x\,[-n]$

(c) $y\,[n] = x\,[n]\cos(\omega_0 n)$

Solution:

Let us express the general relationship between $x\,[n]$ *and* $y\,[n]$ *as* $y\,[n] \equiv \mathcal{T}(x\,[n])$

(a)

- $\mathcal{T}(x\,[n-k]) = nx\,[n-k] \neq y\,[n-k] = [n-k]\,x\,[n-k] \rightarrow$ time variant *(feeding a delayed version of the input to the system does not produce the same output as feeding the original signal to the system and delaying the output).*
- $\mathcal{T}(\alpha x_1\,[n] + \beta x_2\,[n]) = n(\alpha x_1\,[n] + \beta x_2\,[n]) = \alpha n x_1\,[n] + \beta n x_2\,[n] = \alpha\mathcal{T}(x_1\,[n]) + \beta\mathcal{T}(x_2\,[n]) \rightarrow$ linear.
- *The output only depends on present inputs* $\rightarrow$ *it does not depend on future inputs* $\rightarrow$ causal.

(b)
- $\mathcal{T}(x[n-k]) = x[-n-k] \neq y[n-k] = x[-n+k] \rightarrow$ time variant
- $\mathcal{T}(\alpha x_1[n] + \beta x_2[n]) = \alpha x_1[-n] + \beta x_2[-n]) = \alpha\mathcal{T}(x_1[n]) + \beta\mathcal{T}(x_2[n]) \rightarrow$ linear.
- *The output can depend on future inputs (*e.g., $y[-3] = x[3]$*)* $\rightarrow$ not causal.

(c)
- $\mathcal{T}(x[n-k]) = x[n-k]\cos(\omega_0 n) \neq y[n-k] = x[n-k]\cos(\omega_0(n-k)) \rightarrow$ time variant
- $\mathcal{T}(\alpha x_1[n] + \beta x_2[n]) = (\alpha x_1[n] + \beta x_2[n])\cos(\omega_0 n) = \alpha\mathcal{T}(x_1[n]) + \beta\mathcal{T}(x_2[n]) \rightarrow$ linear.
- *The output only depends on present inputs* $\rightarrow$ *it does not depend on future inputs* $\rightarrow$ causal.

Example 7.2 (Stability)

(a) Consider an LTI system with impulse response

$$h[n] = \begin{cases} a^n & \text{for } n \geq 0 \\ b^n & \text{for } n < 0. \end{cases}$$

Give the values of $a, b \in \mathbb{R}$ *for which it is BIBO stable. (Hint: for what values of* a *and* b *is* $\sum_{n=-\infty}^{\infty} |h[n]| < \infty$*)?)*

(b) Consider the system given by

$$y[n] = ay[n-1] + x[n],$$

where a *is a constant. Let* $y[-1] = 0$*. What is the impulse response of this system? Prove that this system is not BIBO stable for any* a *(Hint: compute successive values of* $y[n], n \geq 0$*, then express* $y[n]$ *as a function of* $x[n]$ *and set* $x[n] = \delta[n]$ *to get the impulse response).*

(c) We know that the output of an LTI system is bounded if

$$S_h = \sum_{k=-\infty}^{\infty} |h[k]| < \infty$$

. A direct consequence is that $h[n]$*, the impulse response, goes to zero as* n *approaches infinity (*i.e., $h[n]$ *is identically* 0 *for large enough* n*). Show that the output* $y[n]$ *of such a system goes to zero as* n *approaches infinity, if* $x[n] < M_x$ *for* $n < n_0$ *and* $x[n] = 0$ *for* $n \geq n_0$*. (Hint: bound* $|y[n_0 + N]|$ *and look at the limit when* $N \rightarrow \infty$*).*

Solution:

(a) $\sum_{n=-\infty}^{\infty} |h[n]| = \sum_{n=0}^{\infty} |a^n| + \sum_{n=-\infty}^{-1} |b^n|$. *The first sum converges when* $|a| < 1$. *For the second sum, we have* $\sum_{n=-\infty}^{-1} |b^n| = \sum_{n=1}^{\infty} (\frac{1}{|b|})^n = \frac{1}{|b|}(1 + \frac{1}{|b|} + \frac{1}{|b|^2} + ...)$, *which converges when* $|b| > 1$. *Hence, to guarantee convergence, we must have* $|a| < 1 < |b|$.

(b) Observe that:

$$\begin{aligned} y[0] &= ay[-1] + x[0] = x[0] \\ y[1] &= ax[0] + x[1] \\ y[2] &= a^2 x[0] + ax[1] + x[2] \\ &... \\ y[n] &= \sum_{k=0}^{n} a^k x[n-k]. \end{aligned}$$

Replacing $x[n]$ *by* $\delta[n]$, *we get* $y[n] = h[n] = a^n u[n]$. *As we have seen in the previous question, this sum does not converge for* $a \geq 1$, *and consequently a bounded input (*e.g., *a* δ *function) does not necessarily lead to a bounded output (*$a^n \to \infty$, *as* $n \to \infty$*).*

(c) We have

$$\begin{aligned} |y[n_0 + N]| = & |\overbrace{\sum_{k=-\infty}^{N-1} h[k]\, x[n_0 + N - k]}^{=0, x[n]=0 \ for \ n \geq n_0} + \sum_{k=N}^{\infty} h[k]\, x[n_0 + N - k]| \\ & \leq \sum_{k=N}^{\infty} |h[k]|\, |x[n_0 + N - k]| \\ & \leq M_x \sum_{k=N}^{\infty} |h[k]| \end{aligned}$$

then,

$$\lim_{N \to \infty} |y[n_0 + N]| \leq M_x \lim_{N \to \infty} \sum_{k=N}^{\infty} |h[k]| = 0$$

Example 7.3

Consider the interconnection of systems shown in Fig. 7.3.

(a) Express the overall impulse response h *in terms of* h_1,h_2,h_3 *and* h_4.

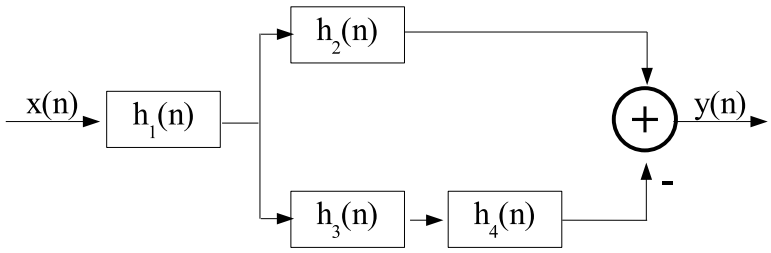

Figure 7.3: System for Problem 7.3

(b) Determine $h[n]$ *when*

$$h_1[n] = \begin{cases} \frac{1}{2} & \text{for } n = 0 \text{ and } n = 2 \\ \frac{1}{4} & \text{for } n = 1 \\ 0 & \text{otherwise,} \end{cases}$$

$h_2[n] = h_3[n] = (n+1)u[n]$ *and* $h_4[n] = \delta[n-2]$.

(c) Determine the response of the system in the previous question when $x[n] = \delta[n+2] + 3\delta[n-1] - 4\delta[n-3]$.

Solution:

(a) One can immediately see that:

$$h[n] = h_1[n] * \overbrace{(h_2[n] - \underbrace{h_3[n] * h_4[n]}_{h_{34}})}^{h_{234}}.$$

(b) Referring to the notation in the previous point, we can calculate:

$$h_{34}[n] = \sum_{k=-\infty}^{\infty} u[k](k+1)\delta[n-2-k] = u[n-2](n-1)$$

$$h_{234}[n] = h_2[n] - h_{34}[n] = (n+1)u[n] - (n-1)u[n-2] = \begin{cases} 0 & \text{when } n < 0 \\ 1 & \text{when } n = 0 \\ 2 & \text{otherwise} \end{cases}$$

$$h[n] = h_1[n] * h_{234}[n] = \sum_{k=0}^{2} h_1[k] h_{234}[n-k] = \left\{ \overbrace{\frac{1}{2}}^{n=0}, \frac{5}{4}, 2, \frac{5}{2}, \frac{5}{2}, \frac{5}{2}, ... \right\}$$

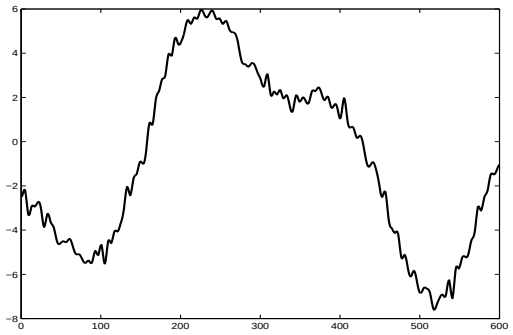

Figure 7.4: Noisy signal.

(c) Finally, we calculate the value of the output when the input x is given:

$$\begin{aligned} y[n] &= \delta[n+2] * h[n] + 3\delta[n-1] * h[n] - 4\delta[n-3] * h[n] \\ &= h[n+2] + 3h[n-1] - 4h[n-3] \\ &= \left\{ ..., 0, 0, \overbrace{\frac{1}{2}}^{n=-2}, \frac{5}{4}, 2, 4, \frac{25}{4}, \frac{13}{2}, 5, 2, 0, 0, 0, ... \right\}. \end{aligned}$$

7.4 Introduction to Filtering

Filtering and filter design are the most fundamental topics in signal processing. We will now introduce the key concepts related to filtering by means of two examples. In both cases we are considering the following problem: we are given a sequence like the one in Figure (7.4) and we want to smooth out the little wiggles in the plot, which are probably due to noise, to improve the readability of the data.

7.4.1 FIR filtering

An intuitive, basic approach to remove noise from data is to replace each point of the sequence $x[n]$ by a local average, taking the point at n and, say, its $N-1$ predecessors into account. The points for the new plot can therefore be computed as:

$$y[n] = \frac{1}{N} \sum_{k=0}^{N-1} x[n-k]$$

This is easily recognized as a convolution sum, and we can obtain the impulse response of the associated filter by letting $x[n] = \delta[n]$; it is easy to see that

$$h[n] = \frac{1}{N}\sum_{k=0}^{N-1}\delta[n-k] = \begin{cases} (1/N) & \text{for } 0 \le n < N \\ 0 & \text{for } n < 0 \text{ and } n \ge N \end{cases}$$

The impulse response, as it turns out, is a finite-support sequence so the filter we just built is an FIR filter; this particular filter goes under the name of *Moving Average* (MA) filter. The "smoothing power" of this filter is dependent on the number of samples we take into account in the average or, in other words, on the length N of its impulse response. The filtered version of the original sequence for increasing values of N is plotted in Figure 7.5. Intuitively we can see that as N grows, more and more wiggles are removed. We will soon see how to handle the "smoothing power" of a filter in a precise, quantitative way. One thing to notice right away, and which is a general characteristic of FIR filters, is that the value of the output does not depend on values of the input which are more than N steps away; FIR filters are therefore called *memoryless* filters. Another remark we can mention right away concerns the *delay* introduced by the filter: each output value is the average of a window of N input values whose representative sample is the one falling in the middle; there is therefore a delay of $N/2$ samples between input and output, and the delay grows with N.

7.4.2 IIR filtering

The moving average filter we built in the previous section has an obvious drawback; the more we want to smooth the signal, the more points we need to consider and, therefore, the more computations we have to perform to obtained the filtered value. Consider now the formula for the output of a length-M moving average filter:

$$y_M[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k].$$

We can easily see that:

$$\begin{aligned} y_M[n] &= \frac{M-1}{M} y_{M-1}[n-1] + \frac{1}{M}x[n] \\ &= \lambda y_{M-1}[n-1] + (1-\lambda)x[n] \end{aligned}$$

where we have defined $\lambda = (M-1)/M$. Now, as M grows large, we can safely assume that if we compute the average over $M-1$ or over M points the result is basically the

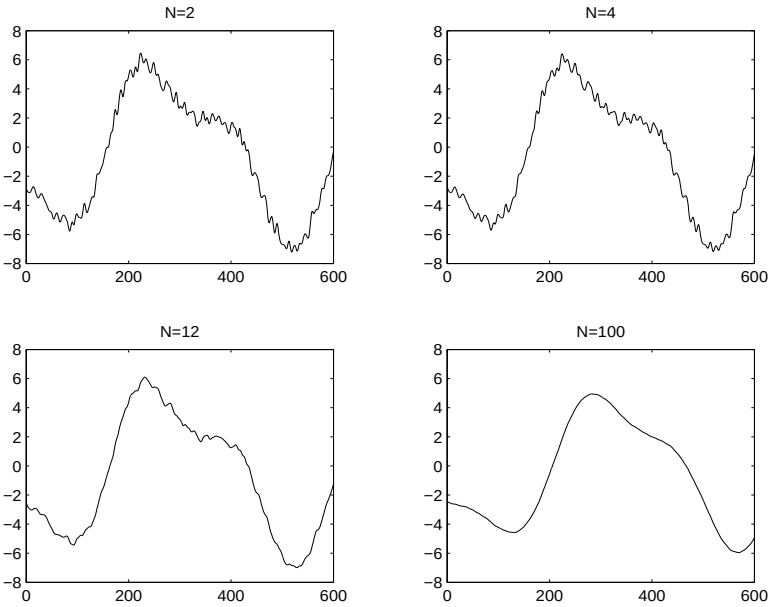

Figure 7.5: Moving averages for different values of N.

same: in other words, for M large, we can say that $y_{M-1}[n] \approx y_M[n]$. This suggests a new way to compute the smoothed version of a sequence in a *recursive* fashion:

$$y[n] = \lambda y[n-1] + (1-\lambda)x[n] \tag{7.18}$$

This does not look anymore like a convolution sum; it is, instead, an instance of a *constant coefficients difference equation*. We might wonder whether the transformation realized by (7.18) is still linear and time-invariant and, in this case, what its impulse response is. The first problem that we face in addressing this question stems from the recursive nature of (7.18): each new output value depends on the previous output value. We need to somehow define a starting value for $y[n]$ or, in system theory parlance, we need to set the *initial conditions*. The choice which guarantees that the system defined by (7.18) is linear and time-invariant corresponds to requiring that the system response to a sequence identically zero be zero for all n; this requirement is also known as *zero initial conditions*, since it corresponds to setting $y[n] = 0$ for $n < N_0$ where N_0 is some time in the past.

Linearity of (7.18) can now be proved this way. Assume that the output sequence for the system defined by (7.18) is $y[n]$ when the input is $x[n]$. It is immediate to see that $y_1[n] = \alpha y[n]$ satisfies (7.18) for an input equal to $\alpha x[n]$. All we need to prove is that this

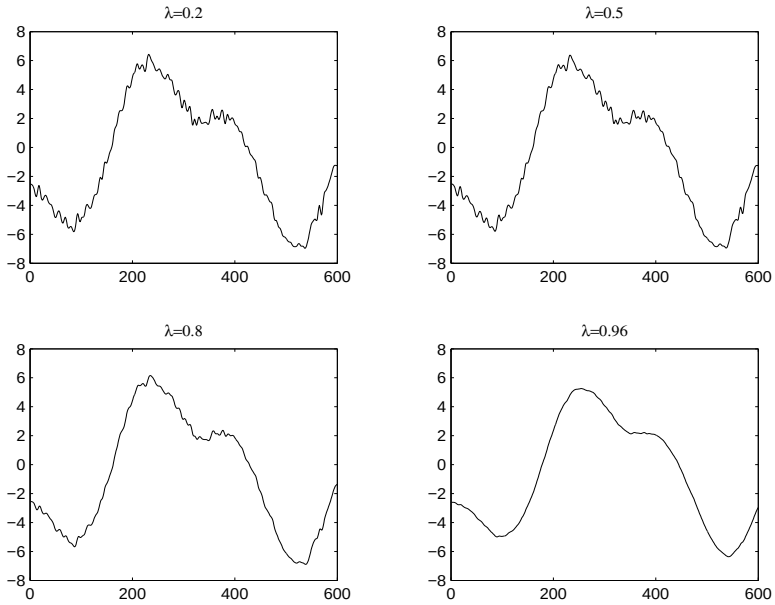

Figure 7.6: Outputs of the leaky integrator for different values of λ.

is the only solution. Assume this is not the case and call $y_2[n]$ the other solution; we have:

$$\begin{aligned} y_1[n] &= \lambda y_1[n-1] + (1-\lambda)(\alpha x[n]) \\ y_2[n] &= \lambda y_2[n-1] + (1-\lambda)(\alpha x[n]) \end{aligned}$$

We can now subtract the second equation from the first. What we have is that the sequence $y_1[n] - y_2[n]$ is the system's response to the zero sequence, and therefore is zero for all n. Linearity with respect to the sum and time invariance can be proven in the exact same way.

Now that we know that (7.18) defines an LTI system, we can try to compute its impulse response. Assuming zero initial conditions and $x[n] = \delta[n]$ we have:

$$\begin{aligned} y[n] &= 0 \quad \text{for } n < 0 \\ y[0] &= 1 - \lambda \\ y[1] &= (1-\lambda)\lambda \\ y[2] &= (1-\lambda)\lambda^2 \\ &\dots \\ y[n] &= (1-\lambda)\lambda^n \end{aligned} \tag{7.19}$$

so that the impulse response is:

$$h[n] = (1-\lambda)\lambda^n u[n]. \tag{7.20}$$

The impulse response clearly defines an IIR filter and therefore the immediate question is whether the filter is stable. Since a sufficient condition for stability is that the impulse response is absolutely summable; we have:

$$\sum_{n=-\infty}^{\infty} |h[n]| = \lim_{n\to\infty} |1-\lambda| \frac{1-|\lambda|^{n+1}}{1-|\lambda|} \tag{7.21}$$

We can see that the above limit is finite for $|\lambda| < 1$ and so the system is BIBO stable for these values. The value of λ (which is, as we will see, the *pole* of the system) determines the smoothing power of the filter. As $\lambda \to 1$, the input is smoothed more and more as can be seen in Figure (7.6), at a constant computational cost. The system implemented by (7.18) is often called a *leaky integrator*, in the sense that it approximates the behavior of an integrator with a leakage (or forgetting) factor λ. The delay introduced by the leaky integrator is more difficult to analyze than for the moving average but, again, it grows with the smoothing power of the filter; we will soon see how to proceed in order to quantify the delay introduced by IIR filters.

As we can infer from this simple analysis, IIR filters are much more delicate entities than FIR filters; in the next chapters we will also discover that their design is also much less straightforward and offers less flexibility. This is why, in the practice, FIR filters are the filters of choice. IIR filters, however, and especially the simplest ones such as the leaky integrator, are extremely attractive when computational power is a scarce resource.

7.5 Filtering in the Frequency Domain

The above examples have introduced the notion of filtering in an operational and intuitive way. In order to make more precise statements on the characteristics of a discrete-time

filter we need to move to the frequency domain. What does a filtering operation translate to in the frequency domain? The fundamental result of this section is the convolution theorem for discrete-time signals: a convolution in the discrete-time domain is equivalent to a multiplication of Fourier transforms in the frequency domain. This result opens up a very fruitful perspective on filtering and filter design, together with alternative approaches to the implementation of filtering devices, as we will see.

7.5.1 Preliminaries

Before we proceed to stating the convolution theorem, let us consider what happens if the input to a linear time-invariant system $\mathcal{H}$ is a complex exponential sequence of frequency ω_0; we have

$$\begin{aligned}
\mathcal{H}\{e^{j\omega_0 n}\} &= \sum_{k=-\infty}^{\infty} e^{j\omega_0 k} h[n-k] \\
&= \sum_{k=-\infty}^{\infty} h[k] e^{j\omega_0 (n-k)} \\
&= e^{j\omega_0 n} \sum_{k=-\infty}^{\infty} h[k] e^{-j\omega_0 k} \\
&= H(e^{j\omega_0}) e^{j\omega_0 n} \qquad (7.22)
\end{aligned}$$

where $H(e^{j\omega_0})$ (i.e. the DTFT of $h[n]$ at $\omega = \omega_0$) is called the *frequency response* of the filter at frequency ω_0. The above result states the fundamental fact that *complex exponentials are eigenfunctions of linear-time invariant systems.* Some remarks:

- If we move to polar form, $H(e^{j\omega_0}) = A_0 e^{j\theta_0}$ and we can write:

  $$\mathcal{H}\{e^{j\omega_0 n}\} = A_0 e^{j(\omega_0 n + \theta_0)}$$

  i.e., the output oscillation is scaled in amplitude by $A_0 = |H(e^{j\omega_0}|$, the magnitude of the DTFT, and it is shifted in phase by $\theta_0 = \measuredangle H(e^{j\omega_0})$, the phase of the DTFT.

- If the input to a linear time-invariant system is a sinusoidal oscillation, the output will always be a sinusoidal oscillation of the same frequency (or zero if $H(e^{j\omega_0}) = 0$). In other words, linear time-invariant systems cannot shift or duplicate frequencies. This strength is also a weakness in some applications and that is why sometimes in practice *nonlinear transformations* are used.

7.5.2 The Convolution and Modulation theorems

We are now ready to state the fundamental result of this section: consider two sequences $x[n]$ and $h[n]$, both absolutely summable. The discrete-time Fourier transform of the convolution $y[n] = x[n] * h[n]$ is:

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega}). \tag{7.23}$$

The proof is as follows: if we take the DTFT of the convolution sum we have

$$Y(e^{j\omega}) = \sum_{n=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} x[k]h[n-k]e^{-j\omega n}$$

by interchanging the order of summation (which can be done because of the absolute summability of both sequences) and by splitting the complex exponential we obtain

$$Y(e^{j\omega}) = \sum_{k=-\infty}^{\infty} x[k]e^{-j\omega k} \sum_{n=-\infty}^{\infty} h[n-k]e^{-j\omega(n-k)}$$

from which the result immediately follows after a change of variable. Before discussing the implications of the theorem, we want to state and prove its dual, called the Modulation Theorem. Consider the discrete-time sequences $x[n]$ and $w[n]$, both absolutely summable, with discrete-time Fourier transforms $X(e^{j\omega})$ and $W(e^{j\omega})$. The discrete-time Fourier transform of the product $y[n] = x[n]w[n]$ is:

$$Y(e^{j\omega}) = X(e^{j\omega}) * W(e^{j\omega}) \tag{7.24}$$

where the DTFT convolution is via the convolution operator for 2π-periodic functions defined in (7.12). This is easily proven as follows: we start from the DTFT inversion formula of the DTFT convolution:

$$\frac{1}{2\pi}\int_{-\pi}^{\pi}(X*Y)(e^{j\omega})e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{1}{2\pi}\int_{-\pi}^{\pi}X(e^{j(\omega-\sigma)})Y(e^{j\sigma})e^{j\omega n}d\sigma d\omega =$$

and we split the last integral to obtain

$$= \left(\frac{1}{2\pi}\int_{-\pi}^{\pi}X(e^{j(\omega-\sigma)})e^{j(\omega-\sigma)n}d\omega\right)\left(\frac{1}{2\pi}\int_{-\pi}^{\pi}Y(e^{j\sigma})e^{j\sigma n}d\sigma\right) = x[n]y[n].$$

These fundamental results are summarized in Table 7.1.

Time Domain	Frequency Domain
$x[n] * y[n]$	$X(e^{j\omega})Y(e^{j\omega})$
$x[n]y[n]$	$X(e^{j\omega}) * Y(e^{j\omega})$

Table 7.1: The Convolution and Modulation Theorems

7.6 The Frequency Response

Just as the impulse response completely characterizes a filter in the discrete-time domain, its Fourier transform, called the filter's *frequency response*, completely characterizes the filter in the frequency domain. The properties of LTI systems are described in terms of their DTFT's magnitude and phase, each of which controls different features of the system's behavior.

7.6.1 Magnitude

The most powerful intuition arising from the convolution theorem is obtained by considering the magnitude of the spectra involved in a filtering operation. Recall that a Fourier spectrum represents the energy distribution of a signal in frequency; by appropriately "shaping" the magnitude spectrum of a filter's impulse response we can easily boost, attenuate and even completely eliminate a given part of the frequency content in the filtered input sequence. According to the way the magnitude spectrum is affected by the filter, we can classify filters into three broad categories (here as before we assume that the impulse response is real, and therefore the associated magnitude spectrum is symmetric; also, the 2π periodicity of the spectrum is implicitly understood):

- **Lowpass filters**, for which the magnitude of the transform is concentrated around $\omega = 0$; these filter preserve the low-frequency energy of the input signals and attenuate or eliminate the high-frequency components.
- **Highpass filters**, for which the magnitude of the transform is concentrated around $\omega = \pm\pi$; these filter preserve the high-frequency energy of the input signals and attenuate or eliminate the low-frequency components.
- **Bandpass filters**, for which the magnitude of the transform is concentrated around $\omega = \pm\omega_p$; these filter preserve the energy of the input signals around the frequency ω_p and attenuate the signals elsewhere, notably around $\omega = 0$ and $\omega = \pm\pi$.

- **Allpass filters**, for which the magnitude of the transform is a *constant* over the entire $[-\pi, \pi]$ interval. These filters do not affect their input's spectral magnitude (except for a constant gain factor) and they are designed entirely in terms of their phase response (typically, to introduce or compensate for a delay).

The frequency interval (or intervals) for which the magnitude of the frequency response is zero (or practically negligible) is called the *stopband*. Conversely, the frequency interval (or intervals) for which the magnitude is non-negligible is called the *passband*.

7.6.2 Phase

The phase response of a filter has an equally important effect on the output signal, even though it is less immediately perceivable.

Phase as a generalized delay. Consider equation (7.22); we can see that a single sinusoidal oscillation undergoes a phase shift equal to the phase of the impulse response's Fourier transform. A phase offset for a sinusoid is equivalent to a delay in the time domain. This is immediate to see for a trigonometric function defined on the real line since we can always write

$$\cos(\omega t + \phi) = \cos(\omega(t - t_0)), \quad t_0 = -\phi/\omega.$$

For discrete-time sinusoids it is not always possible to express the phase offset in terms of an integer number of samples (exactly for the same reasons for which a discrete-time sinusoid is not always periodic in its index n); yet the effect is the same, in that a phase offset corresponds to an implicit delay of the sinusoid. When the phase offset for a complex exponential is not an integer multiple of its frequency, we say we are in the presence of a *fractional delay*. Now, since each sinusoidal component of the input signal may be delayed by an arbitrary amount, the output signal will be composed of sinusoids whose relative alignment may be very different than the original. Phase alignment determines the shape of the signal in the time domain, as we have seen in section 6.2. A filter with unit magnitude across the spectrum, which does not affect the amplitude of the sinusoidal components, but whose phase response is not linear, will completely change the shape of a filtered signal[2].

Linear phase. A very important type of phase response is *linear phase*:

$$\measuredangle H(e^{j\omega}) = e^{-j\omega d} \tag{7.25}$$

[2]In all fairness, the phase response of a system is not very important in most audio applications, since the human ear is largely insensitive to phase. Phase is however extremely important in data transmission applications.

Consider a simple system which just delays its input, i.e. $y[n] = x[n-D]$ with $D \in \mathbb{Z}$; this is obviously an LTI system with impulse response $h[n] = \delta[n-D]$ and frequency response $H(e^{j\omega}) = e^{-j\omega D}$. This means that, if the value d in (7.25) is an integer, (7.25) defines a pure delay system; since the magnitude is constant and equal to one, this is an example of an allpass filter. If d is not an integer, (7.25) still defines an allpass delay system for which the delay is fractional, and we should interpret its effect as explained in the previous section. In particular, if we think of the original signal in terms of its Fourier reconstruction formula, the fractionally delayed output is obtained by stepping forward the initial phase of *all* oscillators by a non-integer multiple of the frequency. In the discrete-time domain we will then have a signal which takes values "between" the original samples but, since the relative phase of any one oscillator with respect to the others has remained the same as in the original signal, the shape of the signal in the time domain is unchanged.

For a general filter with linear phase we can always write

$$H(e^{j\omega}) = |H(e^{j\omega})|e^{-j\omega d}$$

In other words, the net effect of the filter is that of a cascade of two systems: a zero-phase filter which affects only the spectral magnitude of the input and therefore introduces no phase distortion, followed by a (possibly fractional) delay system (which, again, introduces just a delay but no phase distortion).

Group delay. When a filter does not have linear phase, it is important to quantify the amount of phase distortion both in amount and in location. Nonlinear phase is not always a problem; if a filter's phase is nonlinear just in the stopband, for instance, the actual phase distortion is negligible. The concept of group delay is a measure of nonlinearity in the phase; the idea is to express the phase response around any given frequency ω_0 using a first order Taylor approximation. Define $\varphi(\omega) = \measuredangle H(e^{j\omega})$ and approximate $\varphi(\omega)$ around ω_0 as $\varphi(\omega_0 + \tau) = \varphi(\omega_0) + \tau\varphi'(\omega_0)$; we can write

$$\begin{aligned} H(e^{j(\omega_0+\tau)}) &= |H(e^{j(\omega_0+\tau)})|e^{j\varphi(\omega_0+\tau)} \\ &\approx \left(|H(e^{j(\omega_0+\tau)})|e^{j\varphi(\omega_0)}\right)e^{j\varphi'(\omega_0)\tau} \end{aligned} \tag{7.26}$$

so that, approximately, the frequency response of the filter is linear phase for at least a *group* of frequencies around a given ω_0. The delay for this group of frequencies is the negative of the derivative of the phase, from which the definition of group delay:

$$\mathrm{grd}\{H(e^{j\omega})\} = -\varphi'(\omega) = -\frac{d\measuredangle H(e^{j\omega})}{d\omega} \tag{7.27}$$

For truly linear phase systems, the group delay is a constant. Deviations from a constant value quantify the amount of phase distortion introduced by a filter in terms of the (possibly non-integer) number of samples a frequency component is delayed by.

7.7 Examples of Filters

7.7.1 Ideal Filters

Ideal filters are what their name suggests: ideal abstractions which capture the essence of the basic filtering operation. While not realizable in practice, they are the "gold standard" of filter design.

Ideal Lowpass Filter. The ideal lowpass filter is a filter which kills all frequency content above a *cutoff frequency* ω_c and leaves all frequency content below ω_c untouched; it is defined in the frequency domain as

$$H_{lp}(e^{j\omega}) = \begin{cases} 1 & |\omega| \leq \omega_c \\ 0 & \omega_c < |\omega| \leq \pi \end{cases} \tag{7.28}$$

Clearly, the filter has zero phase delay. The ideal lowpass filter can also be defined in terms of its *bandwidth* $\omega_b = 2\omega_c$. The DTFT inversion formula gives the corresponding impulse response:

$$h_{lp}[n] = \frac{\sin(\omega_c n)}{\pi n}. \tag{7.29}$$

The impulse response turns out to be a symmetric infinite sequence and the filter is therefore IIR; unfortunately, however, it can be proved that no realizable system (i.e., no algorithm with a finite number of operations per output sample) can exactly implement the above impulse response. More bad news: the decay of the impulse response is slow, going to zero only as $1/n$, and it is not absolutely summable; this means that any FIR approximation of the ideal lowpass obtained by truncating $h[n]$ will need a lot of samples to achieve some accuracy; and that, in any case, convergence to the ideal frequency response will only be in the mean square sense (see section 5). An immediate consequence of these facts is that, when designing realizable filters, we will take an entirely different approach.

Despite these practical difficulties, the ideal lowpass filter and its associated DTFT pair are so important as a theoretical paradigm that two special function names are used to denote the above expressions. We define

$$\text{rect}(x) = \begin{cases} 1 & |x| \leq 1/2 \\ 0 & |x| > 1/2 \end{cases} \tag{7.30}$$

$$\text{sinc}(x) = \begin{cases} \dfrac{\sin(\pi x)}{\pi x} & x \neq 0 \\ 1 & x = 0 \end{cases} \tag{7.31}$$

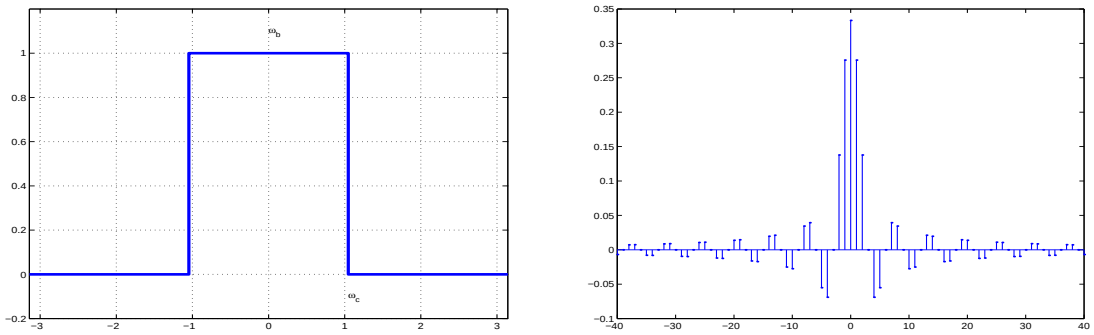

Figure 7.7: Ideal lowpass filter, $\omega_c = \pi/3$. (a) Frequency response; (b) Impulse response (portion).

Note that the sinc function is zero for all integer values of the argument except zero. With this notation, and with respect to the bandwidth of the filter, the ideal lowpass filter's frequency response between $-\pi$ and π becomes:

$$H_{lp}(e^{j\omega}) = \text{rect}\left(\frac{\omega}{\omega_b}\right) \tag{7.32}$$

(obviously 2π-periodized over all $\mathbb{R}$). Its impulse response in terms of bandwidth becomes:

$$h_{lp}[n] = \frac{\omega_b}{2\pi}\,\text{sinc}\left(\frac{\omega_b}{2\pi}n\right) \tag{7.33}$$

or, in terms of cutoff frequency,

$$h_{lp}[n] = \frac{\omega_c}{\pi}\,\text{sinc}\left(\frac{\omega_c}{\pi}n\right). \tag{7.34}$$

The DTFT pair:

$$\frac{\omega_b}{2\pi}\,\text{sinc}\left(\frac{\omega_b}{2\pi}n\right) \overset{\text{DTFT}}{\longleftrightarrow} \text{rect}\left(\frac{\omega}{\omega_b}\right) \tag{7.35}$$

constitutes one of the fundamental relationships of digital signal processing. Note that as $\omega_b \to 2\pi$, we re-obtain the well-known DTFT pair $\delta[n] \longleftrightarrow 1$, while as $\omega_b \to 0$ we can re-normalize by $(2\pi/\omega_b)$ to obtain $1 \longleftrightarrow \tilde{\delta}(\omega)$.

Ideal Highpass Filter. The ideal highpass filter with cutoff frequency ω_c is the complementary filter to the ideal lowpass filter, in the sense that it eliminates all frequency content below the cutoff frequency. Its frequency response is

$$H_{hp}(e^{j\omega}) = \begin{cases} 0 & |\omega| \leq \omega_c \\ 1 & \omega_c < |\omega| \leq \pi \end{cases} \tag{7.36}$$

where the 2π-periodicity is as usual implicitly assumed. From the relation $H_h(e^{j\omega}) = 1 - \text{rect}(\omega/2\omega_c)$ the impulse response is easily obtained as

$$h_{hp}[n] = \delta[n] - \frac{\omega_c}{\pi} \,\text{sinc}\left(\frac{\omega_c}{\pi} n\right)$$

Ideal Bandpass Filter. The ideal bandpass filter with center frequency ω_0 and bandwidth ω_b, $\omega_b/2 < \omega_0$ is defined in the frequency domain between $-\pi$ and π as:

$$H_{bp}(e^{j\omega}) = \begin{cases} 1 & \omega_0 - \omega_b/2 \leq \omega \leq \omega_0 + \omega_b/2 \\ 1 & -\omega_0 - \omega_b/2 \geq \omega \geq -\omega_0 + \omega_b/2 \\ 0 & \text{elsewhere} \end{cases} \tag{7.37}$$

where the 2π-periodicity is as usual implicitly assumed. It is left as an exercise to prove that the impulse response is

$$h_{bp}[n] = 2\cos(\omega_0 n) \,\frac{\omega_b}{2\pi} \,\text{sinc}\left(\frac{\omega_b}{2\pi} n\right). \tag{7.38}$$

Hilbert Filter. The Hilbert filter is defined in the frequency domain as

$$H(e^{j\omega}) = \begin{cases} -j & 0 \leq \omega < \pi \\ +j & -\pi \leq \omega < 0 \end{cases} \tag{7.39}$$

where the 2π-periodicity is as usual implicitly assumed. Its impulse response is easily computed as

$$h[n] = \frac{2\sin^2(\pi n/2)}{\pi n} = \begin{cases} 0 & \text{for } n \text{ even} \\ \frac{2}{n\pi} & \text{for } n \text{ odd} \end{cases} \tag{7.40}$$

It is clearly $|H(e^{j\omega})| = 1$, so this filter is allpass. It introduces a phase shift of $\pi/2$ in the input signal so that, for instance,

$$h[n] * \cos(\omega_0 n) = -\sin(\omega_0 n). \tag{7.41}$$

as one can verify from (5.26) and (5.27). More generally, the Hilbert filter is used in communication systems to build efficient demodulation schemes, as we will see later. The

fundamental concept is the following: consider a *real* signal $x[n]$ and its DTFT $X(e^{j\omega})$; consider also the signal processed by the Hilbert filter $y[n] = h[n] * x[n]$. Define:

$$A(e^{j\omega}) = \begin{cases} X(e^{j\omega}) & \text{for } 0 \leq \omega < \pi \\ 0 & \text{for } -\pi \leq \omega < 0 \end{cases}$$

i.e. $A(e^{j\omega})$ is the positive-frequency part of the spectrum of $x[n]$. Since $x[n]$ is real, its DTFT has symmetry $X(e^{j\omega}) = X^*(e^{-j\omega})$ and therefore we can write

$$X(e^{j\omega}) = A^*(e^{-j\omega}) + A(e^{j\omega}).$$

By separating real and imaginary part we can always write $A(e^{j\omega}) = A_R(e^{j\omega}) + jA_I(e^{j\omega})$ and so:

$$X(e^{j\omega}) = A_R(e^{-j\omega}) - jA_I(e^{-j\omega}) + A_R(e^{j\omega}) + jA_I(e^{j\omega})$$

For the filtered signal we have $Y(e^{j\omega}) = H(e^{j\omega})X(e^{j\omega})$ and therefore

$$Y(e^{j\omega}) = jA_R(e^{-j\omega}) + A_I(e^{-j\omega}) - jA_R(e^{j\omega}) + A_I(e^{j\omega})$$

It is therefore easy to see that

$$x[n] + jy[n] \overset{\text{DTFT}}{\longleftrightarrow} 2A(e^{j\omega}), \tag{7.42}$$

i.e., the spectrum of the signal $a[n] = x[n]+jy[n]$ contains only the positive-frequency components of the original signal $x[n]$. The signal $a[n]$ is called the *analytic signal* associated to $x[n]$.

7.7.2 Examples Revisited

The following is a frequency domain analysis of the two practical filters which we saw earlier. These filters are realizable, in the sense that their operation can be implemented with practical efficient algorithms as we will study in the next chapters. The frequency domain analysis allows us to qualify and quantify precisely the smoothing properties which we described in an intuitive fashion in section 7.4.

Moving Average. The frequency response of the moving average filter in section 7.4.1 can be shown to be

$$H(e^{j\omega}) = \frac{1}{N} \frac{\sin(\omega N/2)}{\sin(\omega/2)} e^{-j\frac{N-1}{2}\omega}. \tag{7.43}$$

In the above expression it is immediate to recognize the magnitude and the phase of the frequency response; they are plotted in Figure 7.8. Note that the phase is "wrapped" onto

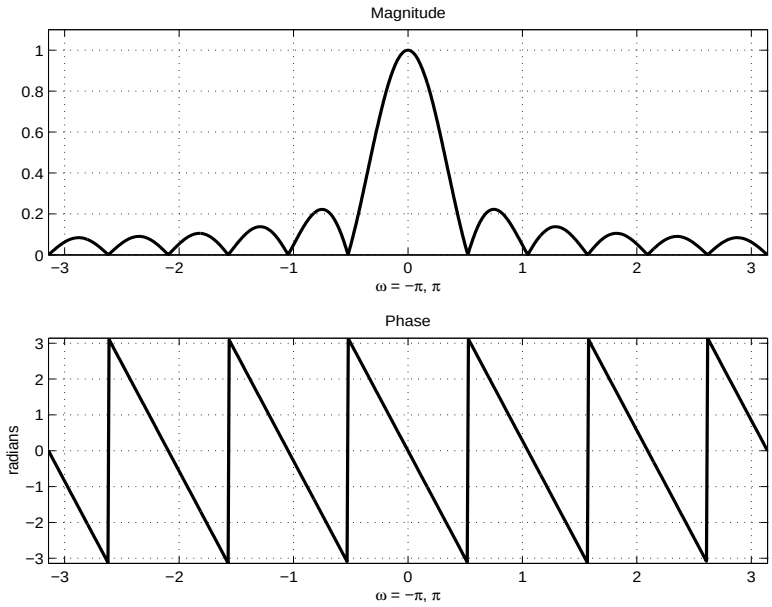

Figure 7.8: Magnitude and phase response of the Moving Average filter for $N = 12$.

the interval $[-\pi, \pi]$, which is customary plotting practice. The group delay for the filter is the constant $(N-1)/2$, which means that the filters delays its output by $(N-1)/2$ samples (i.e. there is a fractional delay for N even).

Leaky Integrator. The frequency response of the leaky integrator in section 7.4.2 is:

$$H(e^{j\omega}) = \frac{1-\lambda}{1-\lambda e^{-j\omega}} \tag{7.44}$$

Magnitude and phase are respectively:

$$|H(e^{j\omega})|^2 = \frac{(1-\lambda)^2}{1+\lambda^2-2\lambda\cos(\omega)} \tag{7.45}$$

$$\measuredangle H(e^{j\omega}) = \arctan\left[-\frac{\lambda\sin(\omega)}{1-\lambda\cos(\omega)}\right] \tag{7.46}$$

and they are plotted in Figure 7.9. The group delay, also plotted in Figure 7.9, is obtained by differentiating the phase response:

$$\mathrm{grd}\{H(e^{j\omega})\} = \frac{\lambda\cos(\omega)-\lambda^2}{1+\lambda^2-2\lambda\cos(\omega)} \tag{7.47}$$

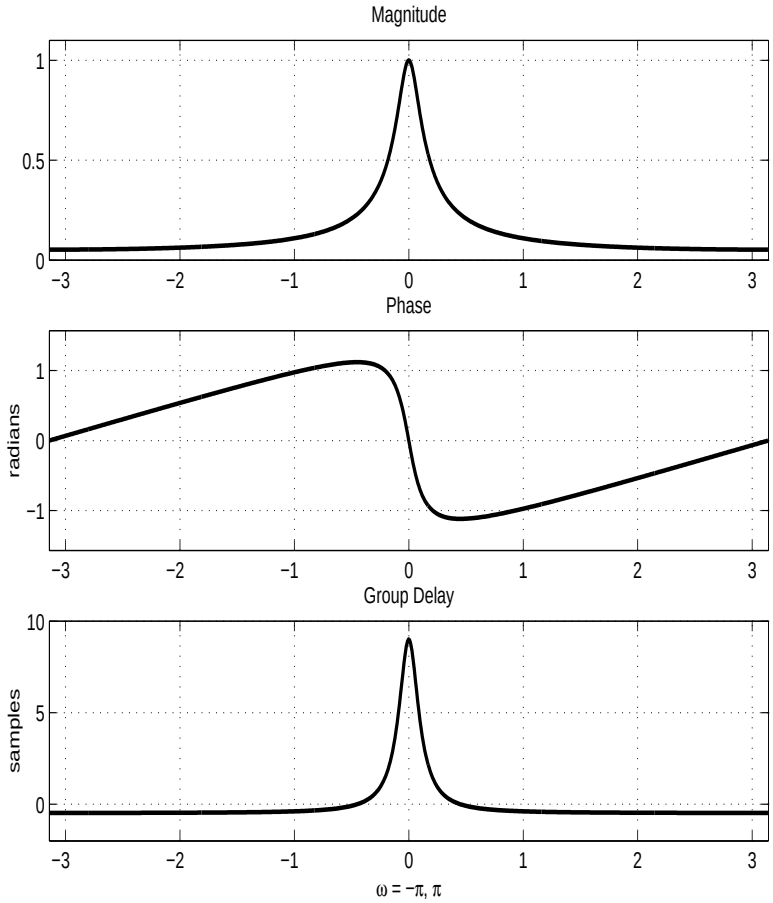

Figure 7.9: Magnitude and phase response of the leaky integrator for $\lambda = 0.9$.

Note that, according to the classification in section 7.6.1, both the moving average and the leaky integrator are lowpass filters.

7.8 Filtering and Signal Classes

We have so far shown the main properties of filters as applied to generic (infinite) sequences. We will now consider the other two main classes of discrete-time signals, namely finite-length signals and periodic sequences.

7.8.1 Filtering of Finite-Length Signals

The convolution sum in (7.3) is defined for infinite sequences. For a finite-length signal of length N we may choose to write simply

$$y[n] = \mathcal{H}\{x[n]\} = \sum_{k=0}^{N-1} x[k]h[n-k] \tag{7.48}$$

i.e. we let the summation index span only the indices for which the signal is defined. It is immediate to see, however, that in so doing we are actually computing $y[n] = \bar{x}[n] * h[n]$, where $\bar{x}[n]$ is the finite support extension of $x[n]$ as in (2.23)); that is, by using (7.48), we are *implicitly* assuming a finite support extension for the input signal.

Even when the input is finite-length, the output of an LTI system is not necessarily a finite-support sequence. When the impulse response is FIR, however, the output has finite support; specifically, if the input sequence has support N and the impulse response has support L, the support of the output will be $N + L - 1$. While the convolution theorem obviously still holds, and the DTFT of the input is as in (5.31), no special insight can be gained from its analytical expression.

7.8.2 Filtering of Periodic Sequences

For periodic sequences, the convolution sum in (7.3) is well defined so there is no special care to be taken. It is easy to see that, for any LTI system, an N-periodic input produces an N-periodic output. A case of particular interest is the following: consider a length-N signal $x[n]$ and its N-periodic extension $\tilde{x}[n]$. Consider then a filter whose impulse response is FIR with a length-N support; if we call $h[n]$ the length-N signal obtained by considering only the values of the impulse response over its finite support, we have that the impulse response of the filter is $\bar{h}[n]$ (see (2.23)). In this case we can write:

$$\tilde{y}[n] = \sum_{k=-\infty}^{\infty} \tilde{x}[k]\bar{h}[n-k] = \sum_{k=0}^{N-1} h[k]\tilde{x}[(n-k) \mod N] \tag{7.49}$$

Note that in the last sum, only the first period of $\tilde{x}[n]$ is used; we can therefore define the sum just in terms of the two N-point signals $x[n]$ and $h[n]$:

$$\tilde{y}[n] = \sum_{k=0}^{N-1} h[k]x[(n-k) \mod N] \tag{7.50}$$

The above summation is called the *circular convolution* of $x[n]$ and $h[n]$ and is sometimes indicated as

$$\tilde{y}[n] = x[n] \circledN h[n]$$

Note that, for periodic sequences, the convolution as defined in (7.8) and the circular convolution coincide. The circular convolution, just like the standard convolution operator, is associative and commutative:

$$x[n] \circledN h[n] = h[n] \circledN x[n]$$

$$(h[n] + f[n]) \circledN x[n] = h[n] \circledN x[n] + f[n] \circledN x[n]$$

as we will easily prove later on.

Consider now the output of the filter, expressed using the commutative property of the circular convolution:

$$\tilde{y}[n] = \sum_{k=0}^{N-1} x[k]h[(n-k) \mod N];$$

since the output sequence $\tilde{y}[n]$ is itself N-periodic we can consider the finite-length signal $y[n] = \tilde{y}[n], n = 0, \ldots, N-1$, i.e. the first period of the output sequence. The circular convolution can now be expressed in matrix form as

$$\mathbf{y} = \mathbf{H}\mathbf{x} \tag{7.51}$$

where $\mathbf{y}, \mathbf{x}$ are the usual vector notation for the finite-length signals $y[n], x[n]$ and where

$$\mathbf{H} = \begin{bmatrix} h[0] & h[N-1] & h[N-2] & \cdots & h[2] & h[1] \\ h[1] & h[0] & h[N-1] & \cdots & h[3] & h[2] \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ h[N-1] & h[N-2] & h[N-3] & \cdots & h[1] & h[0] \end{bmatrix} \tag{7.52}$$

The above matrix is called a circulant matrix, since each row is obtained by a right circular shift of the previous row. A fundamental result, whose proof is left as an exercise, is that the length-N DFT basis vectors $\mathbf{w}^{(k)}$ defined in (3.5) *are left eigenvectors of* $N \times N$ *circulant matrices*:

$$(\mathbf{w}^{(k)})^T \mathbf{H} = H[k]\mathbf{w}^{(k)}$$

where $H[k]$ is the k-th DFT coefficient of the length-N signal $h[n], n = 0, \ldots, N-1$. If we now take the DFT of (7.51) we have

$$\mathbf{Y} = \mathbf{W}\mathbf{H}\mathbf{x} = \mathbf{\Gamma}\mathbf{W}\mathbf{x} = \mathbf{\Gamma}\mathbf{X}$$

with

$$\mathbf{\Gamma} = \mathrm{diag}(H[0], H[1], \ldots, H[N-1])$$

or, in other words

$$Y[k] = H[k]X[k]. \tag{7.53}$$

We have just proven a finite-length version of the convolution theorem. To repeat the main points:

- the convolution of an N-periodic sequence with a N-tap FIR impulse response is equal to the periodic convolution of two finite-length signals of length N, where the first signal is one period of the input and the second signal is the values of the impulse response over the support
- the periodic convolution can be expressed as a matrix-vector product in which the matrix is circulant
- the DFT of the circular convolution is simply the product of the DFT's of the two finite-length signals; in particular, (7.53) can be used to easily prove the commutativity and distributivity of the circular convolution.

The importance of this particular case of filtering stems from the following fact: the matrix-vector product in (7.51) requires $O(N^2)$ operations. The same product can however be written as

$$\mathbf{y} = \frac{1}{N}\mathbf{W}^H\mathbf{\Gamma}\mathbf{W}\mathbf{x} = \mathrm{DFT}^{-1}\{\mathbf{\Gamma}\ \mathrm{DFT}\{\mathbf{x}\}\}$$

which, by using the FFT algorithm, requires approximately $N + 2N\log_2 N$ operations and is therefore much more efficient even for moderate values of N. Practical applications of this idea will be studied in detail later on; suffice it to say for now that, if you want to filter a long signal with an N-tap FIR filter, a computationally attractive way to do it is to break the signal up into consecutive length-N pieces and use the FFT to filter each piece. Efficient methods to glue together the output pieces into the correct final results are called *overlap-save* and *overlap-add* filtering methods.

Finally, we want to show that we could have quickly arrived at the same results only by considering the formal DTFT's of the sequences involved; this is an instance of the power of the DTFT formalism. From (5.30) and (5.31) we have:

$$\begin{aligned} Y(e^{j\omega}) &= \bar{H}(e^{j\omega})\tilde{X}(e^{j\omega}) \\ &= \left(\sum_{k=0}^{N-1} H[k]\Lambda(\omega - \frac{2\pi}{N}k)\right)\left(\frac{1}{N}\sum_{k=0}^{N-1} X[k]\tilde{\delta}(\omega - \frac{2\pi}{N}k)\right) \\ &= \frac{1}{N}\sum_{k=0}^{N-1} H[k]X[k]\tilde{\delta}(\omega - \frac{2\pi}{N}k) \end{aligned} \tag{7.54}$$

where in the last passage we have exploited the sifting property of the Dirac delta (see 5.18) and the fact that $\Lambda(0) = 1$. It is immediate to recognize in the last expresion the DTFT of a periodic sequence whose DFS coefficients are given by $H[k]X[k]$, which is what we wanted to show.

7.9 Summary

This chapter introduced the concept of discrete-time linear time-invariant systems, also known as filters. The main points have been:

- Characterization of LTI systems in terms of their impulse response; IIR and FIR impulse responses;
- The convolution operator and its properties;
- BIBO stability;
- Filtering in the frequency domain; the Convolution and Modulation theorems;
- Magnitude and phase responses; generalized delay and group delay; linear phase;
- Ideal filters: lowpass, highpass, bandpass; Hilbert filter;
- Realizable filters: moving average and leaky integrator;
- Filtering of periodic sequences; circular convolution.

7.10 Problems

Problem 7.1 (Filter Design: Parks-McClellan Algorithm) *In this exercise, our goal is to design an optimal lowpass filter minimizing the maximum error, with passband* $0 \leq \omega \leq \omega_p$ *and stopband* $\omega_s \leq \omega \leq \pi$. *Hence, the desired frequency response* $|H_{dr}(e^{j\omega})|$ *is* 1 *in the passband and* 0 *elsewhere. We would like the response of the designed filter to be within* δ_1 *of* $|H_{dr}(e^{j\omega})|$ *in the passband and within* δ_2 *of* $|H_{dr}(e^{j\omega})|$ *in the stopband. Fig. 7.1 illustrates this idea.*

1. *Show that* $h[n] = h_e[n] + h_o[n]$, *where* $h_e[n] = \frac{1}{2}(h[n] + h[-n])$ *is an even sequence and* $h_o[n] = \frac{1}{2}(h[n] - h[-n])$ *is an odd sequence. Show that it is easy to recover* $h[n]$ *from its even part for* $0 \leq n \leq \infty$ *if* $h[n]$ *is causal. Finally, show that* $h_e[n] \overset{DTFT}{\leftrightarrow} H_R(e^{j\omega})$ *if* $h[n]$ *is real valued and causal (*i.e., $H_e(e^{j\omega})$ *is real).*

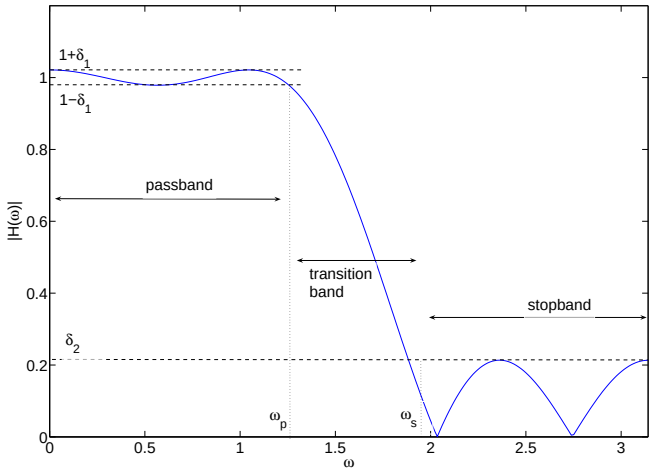

Figure 7.10: Filter Design: We want the impulse response in the passband to be within δ_1 of the desired frequency response 1 and within δ_2 of 0 in the stopband.

2. *Let $h[n]$ be of length M. If $h[n] = h[M-1-n]$ (the unit sample response is symmetric) and M is odd, we have:*

$$H_R(e^{j\omega}) = h\left[\frac{M-1}{2}\right] + 2\sum_{n=0}^{(M-3)/2} h[n]\cos(\omega(\frac{M-1}{2} - n))$$

Further, by making an appropriate change of variable, we have:

$$P(e^{j\omega}) = H_R(e^{j\omega}) = \sum_{k=0}^{(M-1)/2} a[k]\cos(\omega k) \equiv \sum_{k=0}^{L} a[k]\cos(\omega k)$$

Let

$$W(e^{j\omega}) = \begin{cases} \delta_2/\delta_1 & \omega \text{ in the passband} \\ 1 & \omega \text{ in the stopband} \end{cases}$$

and the error function be

$$E(e^{j\omega}) = W(e^{j\omega})(H_{dr}(e^{j\omega}) - P(e^{j\omega}))$$

We seek the solution to the problem:

$$\min_{over\ a[k]} (\max_{\omega \in S} |E(e^{j\omega})|)$$

where S represents the disjoint union of frequency bands over which the optimization is to be performed (in our case S is the union of the passband and the stopband frequencies). The alternation theorem tells us that a necessary and sufficient condition for $P(e^{j\omega})$ to be the best weighted Chebyshev approximation to $H_{dr}(e^{j\omega})$ in S is that the error function $E(e^{j\omega})$ exhibit at least $L+2$ extremal frequencies in S. That is, there must be at least $L+2$ ($L = (M-1)/2$ in our case) frequencies $\{\omega_i\}$ in S such that $\omega_1 < \omega_2 < ... < \omega_{L+2}$, $E(e^{j\omega_i}) = -E(e^{j\omega_{i+1}})$ and $|E(e^{j\omega_i}))| = \max_{\omega \in S} |E(e^{j\omega})|$. At the desired extremal frequencies, we have the set of equations:

$$W(e^{j\omega_n})(H_{dr}(e^{j\omega_n}) - P(e^{j\omega_n})) = (-1)^n \delta_2 \text{ for } n = 0, 1, 2, ..., L+1$$

Show that this set of equations can be written in matrix form as

$$\begin{bmatrix} 1 & \cos(\omega_0) & \cos(2\omega_0) & ... & \cos(L\omega_0) & \frac{1}{W(\omega_0)} \\ 1 & \cos(\omega_1) & \cos(2\omega_1) & ... & \cos(L\omega_1) & \frac{-1}{W(\omega_1)} \\ & & & & & \\ 1 & \cos(\omega_{L+1}) & \cos(2\omega_{L+1}) & ... & \cos(L\omega_{L+1}) & \frac{(-1)^{L+1}}{W(\omega_{L+1})} \end{bmatrix} \begin{bmatrix} a[0] \\ a[1] \\ ... \\ a[L] \\ \delta_2 \end{bmatrix} = \begin{bmatrix} H_{dr}(e^{j\omega_0}) \\ H_{dr}(e^{j\omega_1}) \\ ... \\ H_{dr}(e^{j\omega_{L+1}}) \end{bmatrix}$$

3. *The above set of equations can be solved iteratively by first guessing the extremal frequencies and then solving the system for a and δ_2. Subsequently, given $E(e^{j\omega})$ we find new extremal frequencies and repeat the process. In Matlab, the* `firpm` *command (McClellan Algorithm) solves the problem efficiently. Plot the impulse response and frequency response of a $M = 21$ taps filter, with $\omega_p = 0.45$ and $\omega_s = 0.55$ and $\delta_2/\delta_1 = 5$ and give the error. (Hint: you can use the same set of arguments as for the* `remez` *command presented in the course notes, so play around with this function and see what happens; also take a look at the help). Finally, mark the extremal frequencies on your plot (you can do it by hand).*

Problem 7.2 *An* operator *S is a transformation of a given signal and is indicated by the notation:*

$$y[n] = S\{x[n]\}.$$

For instance, the delay operator D is indicated as

$$D\{x[n]\} = x[n-1],$$

and the differentiation operator is indicated as

$$\Delta\{x[n]\} = x[n] - D\{x[n]\} = x[n] - x[n-1]. \tag{7.55}$$

A linear *operator is one for which the following holds:*

$$\begin{cases} S\{\alpha x[n]\} = \alpha S\{x[n]\} \\ S\{x[n] + y[n]\} = S\{x[n]\} + S\{y[n]\} \end{cases}$$

1. *Show that the delay operator D is linear.*
2. *Show that the differentiation operator Δ is linear.*
3. *Show that the squaring operator $S\{x[n]\} = x^2[n]$ is* not *linear.*

In $\mathbb{C}^N$, any linear operator on a vector $\mathbf{x}$ can be expressed as a matrix-vector multiplication for a suitable matrix $\mathbf{A}$. In $\mathbb{C}^N$, define the delay operator as the left circular shift of a vector:

$$D\{\mathbf{x}\} = [x_{N-1}\ x_0\ x_1\ \ldots\ x_{N-2}]^T.$$

Assume $N = 4$ for convenience; it is easy to see that

$$D\{\mathbf{x}\} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x} = \mathbf{D}\mathbf{x}$$

4. *Using the same definition of the differentiation operator as in (7.55), write out the matrix form of the differentiation operator in* $\mathbb{C}^4$.

5. *Consider the following matrix:*

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

Which operator do you think it corresponds to?

Problem 7.3 *Let* $x[n]$ *be a signal. Consider the following systems with output* $y[n]$. *Determine if such systems are: linear, time invariant, stable (BIBO) and causal or anti-causal. Characterize the systems by their impulse response.*

1. $y[n] = x[-n]$
2. $y[n] = e^{-j\omega n}x[n]$
3. $y[n] = \sum_{k=n-n_0}^{n+n_0} x[k]$
4. $y[n] = ny[n-1] + x[n]$, *such that if* $x[n] = 0$ *for* $n < n_0$, *then* $y[n] = 0$ *for* $n < n_0$. *(*Hint*: Since the system is causal and satisfies initial-rest conditions, we can recursively find the response to any input as, for instance,* $\delta[n]$.*)*

Problem 7.4 *Consider an operator* $\mathcal{R}$ *which turns a sequence into its time-reversed version:*

$$\mathcal{R}\{x[n]\} = x[-n].$$

1. *The operator is clearly linear. Show that it is* not *time-invariant.*

Suppose you have an LTI filter $\mathcal{H}$ *with impulse response* $h[n]$ *and you perform the following sequence of operations in order:*

1) $s[n] = \mathcal{H}\{x[n]\}$

2) $r[n] = \mathcal{R}\{s[n]\}$

3) $w[n] = \mathcal{H}\{r[n]\}$

4) $y[n] = \mathcal{R}\{w[n]\}$

2. *Show that the input-output relation between $x[n]$ and $y[n]$ is an LTI transformation.*

3. *Give the frequency response of the equivalent filter realized by the series of transformations and show that it has zero phase.*