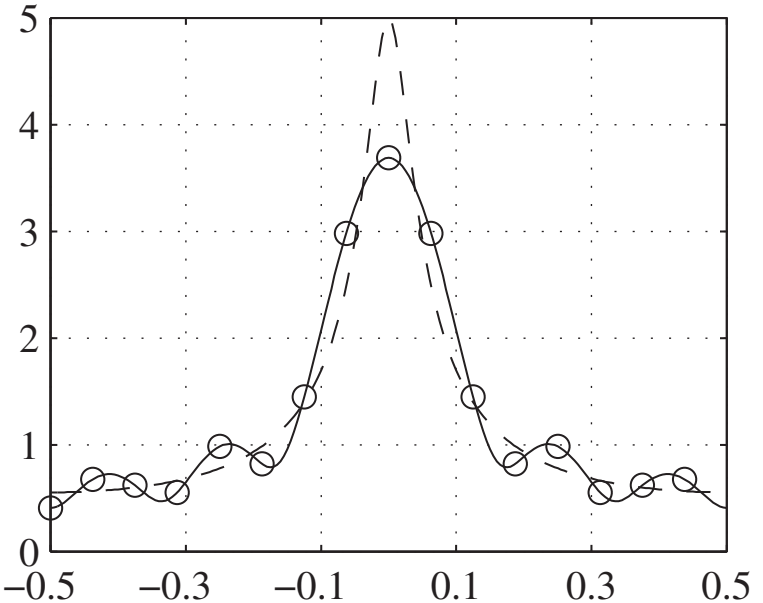
An Introduction to Discrete-Time Signal Processing

John A. Gubner

January 3, 2011



Contents

CHAPTER 1
From Continuous Time to Discrete Time and Back

1.1. Does Sampling *Always* Lose Information?

Given a continuous-time waveform $x(t)$, suppose we extract samples $x(t_n)$ at distinct points in time t_n. Is it possible to reconstruct $x(t)$ for all t using only the distinct values $x(t_n)$? Your initial reaction is probably, "No way!" However, if we assume that $x(t)$ belongs to a restricted class of waveforms, your initial answer may change.

***Example* 1.1.1.** If we know that $x(t)$ is a straight line, then

$$x(t) = x(t_1) + \frac{t - t_1}{t_2 - t_1}\big[x(t_2) - x(t_1)\big].$$

For this class of waveforms, just two samples, taken at distinct times, allow reconstruction of the complete waveform for all time.

The foregoing example easily generalizes to the class of polynomials of degree at most n. In this case, just $n+1$ samples, taken at distinct times, determine the polynomial.

As we shall see, given any waveform $x(t)$ in the class of waveforms bandlimited to f_c, its Fourier transform satisfies

$$X(f) = \frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s) e^{-j2\pi f m/f_s}, \quad |f| \le f_s/2, \tag{1.1}$$

provided $f_s > 2f_c$. Hence, by taking the inverse Fourier transform, we see that $x(t)$ itself can be reconstructed from its samples $\{x(n/f_s)\}_{n=-\infty}^{\infty}$. Here f_c is called the **cutoff frequency**, f_s is called the **sampling frequency**, and $2f_c$ is called the **Nyquist rate**. The statement that a bandlimited waveform can be recovered from its samples if the sampling frequency is greater than the Nyquist rate is called the **Sampling Theorem**.

Now consider a linear time-invariant system with impulse response $h(t)$. The response of this system to an input $x(t)$ is given by the convolution integral

$$y(t) = \int_{-\infty}^{\infty} h(t-\tau)x(\tau)\,d\tau.$$

If the impulse response is bandlimited to f_c, then the output $y(t)$ is also bandlimited to f_c. Hence, $y(t)$ can be recovered from its samples $y(n/f_s)$ if $f_s > 2f_c$. In fact, we will show that if the input is bandlimited too, then the required samples can be obtained from the *discrete-time* convolution,

$$y(n/f_s) = \frac{1}{f_s} \sum_{m=-\infty}^{\infty} h([n-m]/f_s)x(m/f_s). \tag{1.2}$$

Loosely speaking, this explains how computers and compact disc (CD) players handle audio information. Since humans can only hear sounds up to about $f_c = 20$ kHz, it suffices to sample music and speech at about $f_s = 2f_c = 40$ kHz. As a practical matter, to allow for non-ideal electronics, CDs contain samples taken at $f_s = 44.1$ kHz.

Of course, there are a few details that we have glossed over. Computers cannot evaluate the infinite sums in (1.1) and (1.2). Therefore these infinite sums have to be approximated by finite sums. In the case of (1.1), approximation by a finite sum results in a "blurred" or "smeared" version of $X(f)$. The digital signal processing community has devoted a considerable literature to the development of "windows" to compensate for the distortion introduced in the approximation of Fourier transforms. In the case of (1.1) in which a finite impulse response (FIR) filter is used to approximate a filter with ideal cutoff, the Gibbs phenomenon necessarily results. This distortion has been addressed by developing suitable "windows" as well as other techniques.

1.2. Review of Fourier Analysis

In Section 1.2.1, we recall Fourier series for continuous-time, periodic signals. In Section 1.2.2, we recall the discrete-time Fourier transform for discrete-time, aperiodic signals. The duality between these two situations is then readily apparent.

In Section 1.2.3, we motivate the continuous-time Fourier transform by examining the limiting form of the Fourier-series representation of truncations of the time signal.

1.2.1. Continuous-Time Periodic Signals

Suppose $x(t)$ is a periodic signal with period T and **Fourier series** expansion

$$x(t) = \sum_{n=-\infty}^{\infty} x_n e^{j2\pi nt/T}. \tag{1.3}$$

Then it is easy to show that the **Fourier coefficients** are given by

$$x_n = \frac{1}{T}\int_{-T/2}^{T/2} x(t)e^{-j2\pi nt/T}\,dt.$$

This can be demonstrated by considering the integral

$$\begin{aligned}\int_{-T/2}^{T/2} x(t)e^{-j2\pi mt/T}\,dt &= \int_{-T/2}^{T/2}\left[\sum_{n=-\infty}^{\infty} x_n e^{j2\pi nt/T}\right]e^{-j2\pi mt/T}\,dt \\ &= \sum_{n=-\infty}^{\infty} x_n \underbrace{\int_{-T/2}^{T/2} e^{j2\pi t(n-m)/T}\,dt}_{=T\delta_{nm}},\end{aligned}$$

where δ_{nm} is the **Kronecker delta**, which is one for $n=m$ and zero otherwise. Hence, this last sum reduces to $x_m \cdot T$.

If $x(t)$ and $y(t)$ both have period T, then we have **Parseval's equation for Fourier series**,

$$\frac{1}{T}\int_{-T/2}^{T/2} x(t)\overline{y(t)}\,dt = \sum_{n=-\infty}^{\infty} x_n\overline{y_n}, \tag{1.4}$$

where the overbar denotes the complex conjugate. This is easy to derive by substituting (1.3) on the left-hand side of (1.4).

Arbitrary Signals Restricted to Finite Intervals

It is important to observe that nonperiodic signals can be represented on a finite interval $[a,b]$ by using Fourier series if we restrict attention to $[a,b]$. The trick is to consider the periodic repetition of the piece of the waveform on $[a,b]$ ***[PICTURE]***. It follows that we can write

$$x(t) = \sum_{n=-\infty}^{\infty} x_n e^{j2\pi nt/(b-a)}, \quad t\in[a,b], \tag{1.5}$$

with

$$x_n = \frac{1}{b-a}\int_a^b x(t)e^{-j2\pi nt/(b-a)}\,dt.$$

We emphasize that in general, the left-hand side of (1.5) is *not* periodic, while the right-hand side *is* periodic; hence, equality in (1.5) usually holds *only* for $t\in[a,b]$.

1.2.2. Discrete-Time Aperiodic Signals

Given a sequence y_n, its **discrete-time Fourier transform** (DTFT) is

$$Y(f) := \sum_{n=-\infty}^{\infty} y_n e^{-j2\pi fn},$$

which is a periodic function of f with period one. Since

$$Y(-f) = \sum_{n=-\infty}^{\infty} y_n e^{j2\pi fn},$$

it is immediate from the preceding subsection that

$$y_n = \int_{-1/2}^{1/2} Y(f) e^{j2\pi fn}\, df.$$

***Example* 1.2.1.** Fix a frequency $f_0 \in [-1/2, 1/2]$, and suppose[1] $Y(f) = \delta(f - f_0)$ for $|f| \le 1/2$. For other f, $Y(f)$ is defined by the periodic repetition of what we have on $[-1/2, 1/2]$. Then

$$y_n = \int_{-1/2}^{1/2} \delta(f - f_0) e^{j2\pi fn}\, df = e^{j2\pi f_0 n}.$$

In other words, the DTFT of discrete-time complex exponential $e^{j2\pi f_0 n}$ is the periodic repetition of $\delta(f - f_0)$.

***Example* 1.2.2.** Let $0 < a < 1$ and put $y_n := a^n$ for $n \ge 0$ and $y_n = 0$ for $n < 0$. Then

$$Y(f) = \sum_{n=0}^{\infty} a^n e^{-j2\pi fn} = \sum_{n=0}^{\infty} (ae^{-j2\pi f})^n = \frac{1}{1 - ae^{-j2\pi f}},$$

where we have used the **geometric series** formula (Problem 1.6).

If x_n and y_n are sequences with corresponding DTFTs $X(f)$ and $Y(f)$, then we have the corresponding version of **Parseval's equation for the DTFT**,

$$\sum_{n=-\infty}^{\infty} x_n \overline{y_n} = \int_{-1/2}^{1/2} X(f) \overline{Y(f)}\, df.$$

[1] Here the symbol δ denotes the **Dirac delta function** of a continuous variable; i.e., $\delta(f) = 0$ for $f \neq 0$ and $\int_{-\infty}^{\infty} \delta(f)\, df = 1$.

If we approximate the DTFT $X(f)$ by

$$X(f) \approx \sum_{n=n_1}^{n_2} x_n e^{-j2\pi fn}, \tag{1.6}$$

then the right-hand side can be plotted in MATLAB as follows. Assuming $\mathtt{x} = [x_{n_1}, \ldots, x_{n_2}]$,

```
f = linspace(-1/2,1/2,201);
nvec = [n1:n2];
X = dtft(f,x,nvec);
plot(f,X)
```

where

```
function y = dtft(f,x,nvec)
fvec = reshape(f,1,prod(size(f))); % convert f to row vector
y = x*exp(-j*2*pi*nvec.'*fvec);
y = reshape(y,size(f)); % make output have shape of f.
```

will plot an approximation of $X(f)$. Of course, if x_n is a finite-duration signal on $n_1 \le n \le n_2$, then `dtft` computes $X(f)$ exactly. ***Warning:*** **If the lengths of `f` and `nvec` are both large, then `dtft` will be quite slow. In this case, the fast Fourier transform can be used, as shown in the function `dtftfft` given later in Section 3.1.3.**

Application to Continuous-Time Fourier Series

On the right-hand side of (1.6), if we replace f with $-t/T$, we get

$$\sum_{n=n_1}^{n_2} x_n e^{j2\pi nt/T}.$$

Thus, if a continuous-time periodic signal $x(t)$ has Fourier series coefficients x_n, then we can approximate $x(t)$ with the MATLAB command `dtft(-t/T,x,nvec)`.

1.2.3. Continuous-Time Aperiodic Signals

If $x(t)$ is an aperiodic signal, consider the **truncated signal** ***[PICTURE]***

$$x^T(t) := \begin{cases} x(t), & |t| \le T/2, \\ 0, & |t| > T/2. \end{cases}$$

Then, for $|t| \le T/2$, we can use the formulas in Section 1.2.1 to write

$$x^T(t) = \sum_{n=-\infty}^{\infty} x_n^T e^{j2\pi nt/T}, \quad |t| \le T/2, \tag{1.7}$$

where

$$\begin{aligned} x_n^T &= \frac{1}{T}\int_{-T/2}^{T/2} x^T(t) e^{-j2\pi nt/T}\, dt \\ &= \frac{1}{T}\int_{-T/2}^{T/2} x(t) e^{-j2\pi nt/T}\, dt. \end{aligned}$$

Observe that if we put $\Delta f := 1/T$, and change t to τ, then

$$x_n^T = \Delta f \int_{-T/2}^{T/2} x(\tau) e^{-j2\pi(n\Delta f)\tau}\, d\tau.$$

Making these substitutions into (1.7), and noting that for $|t| \le T/2$, $x^T(t) = x(t)$, we can write

$$\begin{aligned} x(t) &= \sum_{n=-\infty}^{\infty} \Delta f \left[\int_{-T/2}^{T/2} x(\tau) e^{-j2\pi(n\Delta f)\tau}\, d\tau\right] e^{j2\pi(n\Delta f)t} \\ &\approx \int_{-\infty}^{\infty} \left[\int_{-T/2}^{T/2} x(\tau) e^{-j2\pi f\tau}\, d\tau\right] e^{j2\pi ft}\, df. \end{aligned}$$

Letting $T \to \infty$, we have

$$x(t) = \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} x(\tau) e^{-j2\pi f\tau}\, d\tau\right] e^{j2\pi ft}\, df. \tag{1.8}$$

This inner integral is called the **Fourier transform**. If we put

$$X(f) := \int_{-\infty}^{\infty} x(\tau) e^{-j2\pi f\tau}\, d\tau,$$

then (1.8) says that

$$x(t) = \int_{-\infty}^{\infty} X(f) e^{j2\pi ft}\, df. \tag{1.9}$$

Equation (1.9) is called the **Fourier inversion formula** or **inverse Fourier transform**.

***Example* 1.2.3.** Although we introduced the Fourier transform as applying to aperiodic signals, we can apply it to periodic signals as follows. If $x(t)$ is periodic, then we can substitute its Fourier series expansion

$$x(t) = \sum_{n=-\infty}^{\infty} x_n e^{j2\pi nt/T}$$

Taking the Fourier transform term by term, and recalling that $e^{j2\pi f_0 t}$ and $\delta(f - f_0)$ are Fourier transform pairs, we see that

$$X(f) = \sum_{n=-\infty}^{\infty} x_n \delta(f - n/T).$$

In other words, the nth Fourier series coefficient x_n is associated with the frequency $f = n/T$.

Some Important Facts about Fourier Transforms

If $x(t)$ and $y(t)$ are continuous-time aperiodic signals with respective Fourier transforms $X(f)$ and $Y(f)$, then we have **Parseval's equation for Fourier transforms**,

$$\int_{-\infty}^{\infty} x(t)\overline{y(t)}\,dt = \int_{-\infty}^{\infty} X(f)\overline{Y(f)}\,df. \tag{1.10}$$

The **convolution** of x and y is

$$(x * y)(t) := \int_{-\infty}^{\infty} x(t-\tau)y(\tau)\,d\tau = \int_{-\infty}^{\infty} x(\theta)y(t-\theta)\,d\theta.$$

[Do graphical example for two rectangular pulses of different widths. "Smearing effect." Specialize to same width. Review unit impulse.]

The Fourier transform of $(x * y)(t)$ is $X(f)Y(f)$. ***[EXAMPLE: Apply to centered pulse with itself and get*** sinc^2***.]***

The convolution of X and Y is

$$(X * Y)(f) = \int_{-\infty}^{\infty} X(f-\nu)Y(\nu)\,d\nu = \int_{-\infty}^{\infty} X(\theta)Y(f-\theta)\,d\theta.$$

The inverse transform of $(X * Y)(f)$ is $x(t)y(t)$. ***[Example: Rectangular windowing in time becomes convolution with*** sinc ***in frequency.]***

Transforms of Real Signals

If $x(t)$ is real valued, then its transform $X(f)$ has the property $\overline{X(f)} = X(-f)$ (see Problem 1.19). It then follows that

$$|X(f)|^2 = X(f)\overline{X(f)} = X(f)X(-f),$$

which implies that $|X(f)|^2$ is an even function of f. Because $|X(f)|^2$ (and $|X(f)|$ as well) are even, we often plot them only for positive values of f. For this reason, we define the **bandwidth** of a signal to be the length of the frequency band of positive frequencies over which the transform is nonzero. ***[Draw typical two-sided spectra of LPF, BPF.]***

1.3. Relating the Continuous-Time and the Discrete-Time Fourier Transforms

Consider a continuous-time waveform $x(t)$ with Fourier transform $X(f)$. Suppose we sample $x(t)$ at the **sampling rate** f_s; i.e., we sample $x(t)$ at multiples of the **sampling interval** $T_s := 1/f_s$. Denote the DTFT of the samples $x(nT_s) = x(n/f_s)$ by

$$X_{\text{DTFT}}(f) = \sum_{n=-\infty}^{\infty} x(n/f_s)e^{-j2\pi fn}.$$

We begin our analysis by applying the the inverse DTFT to write

$$\begin{aligned} x(n/f_s) &= \int_{-1/2}^{1/2} X_{\text{DTFT}}(f)e^{j2\pi fn}\,df \\ &= \frac{1}{f_s}\int_{-f_s/2}^{f_s/2} X_{\text{DTFT}}(\nu/f_s)e^{j2\pi\nu n/f_s}\,d\nu, \end{aligned} \tag{1.11}$$

where for convenience we have made the change of variable $\nu = f_s \cdot f$, $d\nu = f_s\,df$. On the other hand, by the continuous-time inverse Fourier transform,

$$x(t) = \int_{-\infty}^{\infty} X(f)e^{j2\pi ft}\,df.$$

Specializing to the case $t = n/f_s$, we obtain

$$x(n/f_s) = \int_{-\infty}^{\infty} X(f)e^{j2\pi f(n/f_s)}\,df. \tag{1.12}$$

1.3.1. The Bandlimited Case

We say that $x(t)$ is **bandlimited** if for some positive, finite frequency f_c, called the **cutoff frequency**, $X(f) = 0$ for $|f| > f_c$. In this case, (1.12) can be rewritten as

$$x(n/f_s) = \int_{-f_s/2}^{f_s/2} X(f) e^{j2\pi fn/f_s}\, df, \quad \text{if } f_s/2 \geq f_c. \tag{1.13}$$

Since this integral and the one in (1.11) are both equal to $x(n/f_s)$ for all n, we must have[2]

$$X(f) = \frac{1}{f_s} X_{\text{DTFT}}(f/f_s), \quad |f| \leq f_s/2. \tag{1.14}$$

We write this more explicitly as

$$X(f) = \frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s) e^{-j2\pi fm/f_s}, \quad |f| \leq f_s/2. \tag{1.15}$$

This shows that for a bandlimited signal, its Fourier transform can be computed without numerical integration. We only need to compute the DTFT of equally spaced samples!

1.3.2. The General Case

When the waveform is bandlimited and $f_s/2 \geq f_c$, we passed directly from (1.12) to (1.13). If the waveform is bandlimited but $f_s/2 < f_c$ or if the waveform is not bandlimited, we proceed as follows. We break up the range of integration in (1.12) into intervals of length f_s with one of the intervals centered at $f = 0$. This allows us to rewrite (1.12) as

$$\begin{aligned} x(n/f_s) &= \sum_{k=-\infty}^{\infty} \int_{-(2k-1)f_s/2}^{(2k+1)f_s/2} X(f) e^{j2\pi fn/f_s}\, df \\ &= \sum_{k=-\infty}^{\infty} \int_{-f_s/2}^{f_s/2} X(\nu + kf_s) e^{j2\pi(\nu+kf_s)n/f_s}\, d\nu, \quad \nu = f - kf_s, \\ &= \sum_{k=-\infty}^{\infty} \int_{-f_s/2}^{f_s/2} X(\nu + kf_s) e^{j2\pi\nu n/f_s}\, d\nu, \quad \text{since } e^{j2\pi kn} = 1, \end{aligned}$$

[2] The integrals in (1.13) and (1.11) are Fourier-coefficient integrals. If these integrals are equal for all n, then the functions $X(f)$ and $(1/f_s)X_{\text{DTFT}}(f/f_s)$ are the same (assuming these are ordinary functions not containing any impulses at $f = \pm f_s/2$). To see that we must exclude impulses at the end points, observe that

$$\int_{-1/2}^{1/2} \delta(f - 1/2) e^{j2\pi fn}\, df \quad \text{and} \quad \int_{-1/2}^{1/2} \delta(f + 1/2) e^{j2\pi fn}\, df$$

are both equal to $(-1)^n$.

$$= \int_{-f_s/2}^{f_s/2} \left[\sum_{k=-\infty}^{\infty} X(\nu + k f_s) \right] e^{j2\pi\nu n/f_s} \, d\nu.$$

Comparing this with (1.11), we conclude that

$$\frac{1}{f_s} X_{\mathrm{DTFT}}(\nu/f_s) = \sum_{k=-\infty}^{\infty} X(\nu + k f_s).$$

Letting $\nu = f$ and making the change of variable $n = -k$ in the sum, we obtain the **Poisson summation formula**,

$$\frac{1}{f_s} X_{\mathrm{DTFT}}(f/f_s) = \sum_{n=-\infty}^{\infty} X(f - n f_s), \tag{1.16}$$

which we write more explicitly as

$$\widetilde{X}(f) := \sum_{n=-\infty}^{\infty} X(f - n f_s) = \frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s) e^{-j2\pi f m/f_s}. \tag{1.17}$$

An example of shifted transforms and their sum is shown in Figure 1.1.

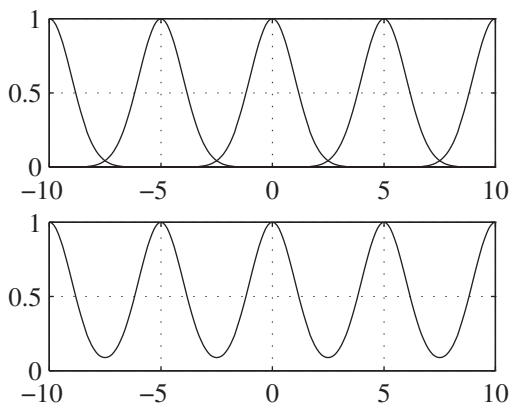


Figure 1.1. Shifts $X(f - n f_s)$ (top), and $\widetilde{X}(f)$ (bottom) for $X(f) = e^{-f^2/2}$ and $f_s = 5$.

1.3.3. Approximating the Continuous-Time Fourier Transform in MATLAB

For a bandlimited waveform $x(t)$, it is easy to use (1.14) and our MATLAB function `dtft` to approximate the Fourier transform $X(f)$. The following MATLAB script

does this for $x(t) = \cos(2\pi f_0 t)$ with $f_0 = 10$. In this case, the cutoff frequency is 10 and so we must choose $f_s > 20$. Since we plan to plot the transform for $-15 \le f \le 15$, must take $f_s/2 \ge 15$. For this reason, we use $f_s = 30$.

```
subplot(2,1,1)              % Plot waveform x(t)
t = linspace(0,1,200);
plot(t,cos(2*pi*10*t))
title('\itx\rm(\itt\rm)=cos(2*pi*10*\itt\rm)')
grid on

fs = 30;                % Select sampling rate.
t1 = 0;                 % Sample x(t) for t1<=t<=t2 ...
t2 = 1;

n1 = ceil(fs*t1);       % ... using time values of the form n/fs
n2 = floor(fs*t2);
nvec = [n1:n2];

x = cos(2*pi*10*nvec/fs); % Compute function values

f = linspace(-15,15,401); % Approx. Fourier transform on any
y = dtft(f/fs,x,nvec)/fs; % subinterval of [-fs/2,fs/2].

subplot(2,1,2)
plot(f,abs(y))
title('Approximate Fourier Transform')
grid on
```

If we apply this script to a waveform that is not bandlimited, then aliasing will occur as implied by (1.16). However, if f_s is taken large enough good results can be obtained. You can try this by modifying the script to use $x(t) = e^{-|t|}$. To check the result, you can also plot its exact transform, which is $X(f) = 2/(1+(2\pi f)^2)$.

Warning: **As mentioned in Section 1.2.2, if the lengths of `f` and `nvec` are both large, then `dtft` may be quite slow. In this case, the fast Fourier transform can be used as shown in Section 3.1.4.**

1.4. The Sampling Theorem

For bandlimited waveforms, (1.15) shows that $X(f)$, where it is nonzero, is completely determined by the waveform samples. However, if we know $X(f)$, we know $x(t)$ *for all* t by the inverse Fourier transform! We thus have the **Sampling Theorem**: For a bandlimited waveform, its Fourier transform, and therefore the waveform itself, can be recovered from the waveform samples if the sampling rate f_s is greater than or equal to $2f_c$ (called the **Nyquist rate**).

1.4.1. The Sinc Reconstruction Formula

When $x(t)$ is bandlimited to f_c and $f_s \geq 2f_c$, we showed that (1.15) holds. Keeping this in mind, write

$$\begin{aligned}
x(t) &= \int_{-\infty}^{\infty} X(f) e^{j2\pi ft}\, df \\
&= \int_{-f_s/2}^{f_s/2} X(f) e^{j2\pi ft}\, df, \quad \text{since } X(f) = 0 \text{ for } |f| > f_s/2, \\
&= \int_{-f_s/2}^{f_s/2} \left[\frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s) e^{-j2\pi fm/f_s} \right] e^{j2\pi ft}\, df, \quad \text{by (1.15)}, \qquad (1.18) \\
&= \frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s) \int_{-f_s/2}^{f_s/2} e^{j2\pi f(t-m/f_s)}\, df \\
&= \frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s) \cdot f_s \operatorname{sinc}(f_s[t-m/f_s]), \quad \text{by Problem 1.25}, \\
&= \sum_{m=-\infty}^{\infty} x(m/f_s) \operatorname{sinc}(f_s[t-m/f_s]), \qquad (1.19)
\end{aligned}$$

which is known as the **sinc reconstruction formula** or the **sinc interpolation formula**.

A Signal Cannot Be Both Time Limited and Bandlimited

If $x(t)$ is bandlimited, then (1.19) holds. But if $x(t)$ is also time limited, then the sum in (1.19) has only a finite number of nonzero terms. Intuitively, there is no way this finite sum of sinc functions can achieve complete cancellation for all $|t|$ bigger than some time limit.

As a practical matter, all signals are time limited. They started as some time in the finite past, and they can be measured only up to the present time. Similarly, all signals are bandlimited. For example, electrical signals at light frequencies cannot propagate in a copper wire. An interesting discussion relating these practicalities to the mathematics of Fourier transform theory can be found in Slepian's paper [9].

1.4.2. Aliasing

When the conditions of the sampling theorem are violated, the waveform that results from applying the sinc reconstruction formula is not equal to $x(t)$. In this case the sinc reconstruction result is a distorted version of $x(t)$. This distortion is called **aliasing**.

To gain further insight into what the sinc reconstruction formula does, first note that the steps starting with (1.18) and ending with (1.19) still hold. Combining this with the Poisson summation formula (1.17), where $\widetilde{X}(f)$ is defined, we see that

$$\sum_{m=-\infty}^{\infty} x(m/f_s)\,\mathrm{sinc}(f_s[t-m/f_s]) = \int_{-f_s/2}^{f_s/2} \widetilde{X}(f)e^{j2\pi ft}\,df. \tag{1.20}$$

If we let $\tilde{x}(t)$ denote the inverse Fourier transform of $\widetilde{X}(f)$, then (1.20) says that the sinc reconstruction operation is equivalent to passing $\tilde{x}(t)$ through an ideal lowpass filter of bandwidth $f_s/2$. It is important to point out here that if $x(t)$ is bandlimited to f_c and $f_s > 2f_c$, then the shifted copies of $X(f)$ that make up $\widetilde{X}(f)$ do not overlap ***[Draw pictures]***, implying that $\widetilde{X}(f) = X(f)$ for $|f| \le f_s/2$. This allows the right-hand side of (1.20) to be rewritten as

$$\begin{aligned}\int_{-f_s/2}^{f_s/2} X(f)e^{j2\pi ft}\,df &= \int_{-\infty}^{\infty} X(f)e^{j2\pi ft}\,df, \quad \text{since } x(t) \text{ is bandlimited,}\\ &= x(t).\end{aligned}$$

We thus recover the fact that under the conditions of the sampling theorem, the sinc formula recovers $x(t)$. We also see that the sinc formula gives a distorted version of $x(t)$ precisely when the shifts of $X(f)$ that make up $\widetilde{X}(f)$ overlap; i.e., aliasing is simply the overlap of the shifts of $X(f)$. ***[Illustrate with ramp*** $X(f) = (1+f)/2$ ***on*** $[-1,1]$***. Let*** f_s ***go from a little above*** $2f_c$ ***to a little below*** $2f_c$***.]***

1.4.3. The Zero-Order Hold

As above, we continue to let $\tilde{x}(t)$ denote the inverse Fourier transform of $\widetilde{X}(f)$. We now derive a simple formula for $\tilde{x}(t)$. Using the Fourier transform pair

$$\delta(t-t_0) \leftrightarrow e^{-j2\pi ft_0}$$

and the right-hand side of the Poisson summation formula (1.17), we have

$$\tilde{x}(t) = \frac{1}{f_s}\sum_{m=-\infty}^{\infty} x(m/f_s)\delta(t-m/f_s).$$

The preceding formula is called **impulse sampling** of $x(t)$. The point here is that one way to reconstruct the waveform $x(t)$ is to apply $\tilde{x}(t)$ to an ideal lowpass filter. One problem is that impulse sampling is an idealization that is not exactly realizable. However, what happens if we replace $\delta(t)$ with a realizable pulse $p(t)$? Let

$$x_p(t) := \sum_{m=-\infty}^{\infty} x(m/f_s)p(t-m/f_s). \tag{1.21}$$

Taking Fourier transforms yields

$$\begin{aligned} X_p(f) &= \sum_{m=-\infty}^{\infty} x(m/f_s)P(f)e^{-j2\pi fm/f_s} \\ &= \left(\sum_{m=-\infty}^{\infty} x(m/f_s)e^{-j2\pi fm/f_s}\right)P(f) \\ &= f_s\widetilde{X}(f)P(f), \end{aligned}$$

where $P(f)$ denotes the Fourier transform of the pulse $p(t)$. Hence, if we apply $x_p(t)$ to a filter $H(f)$ with

$$H(f) = \begin{cases} 1/[f_sP(f)], & |f| \le f_c \\ 0, & |f| > f_c, \end{cases} \tag{1.22}$$

then the output will be

$$\begin{aligned} \int_{-\infty}^{\infty} H(f)X_p(f)e^{j2\pi ft}\,df &= \int_{-f_c}^{f_c} H(f)X_p(f)e^{j2\pi ft}\,df \\ &= \int_{-f_c}^{f_c} \frac{X_p(f)}{f_sP(f)}e^{j2\pi ft}\,df \\ &= \int_{-f_c}^{f_c} \frac{f_sP(f)\widetilde{X}(f)}{f_sP(f)}e^{j2\pi ft}\,df \\ &= \int_{-f_c}^{f_c} \widetilde{X}(f)e^{j2\pi ft}\,df \\ &= \int_{-f_{s/2}}^{f_{s/2}} \widetilde{X}(f)e^{j2\pi ft}\,df, \end{aligned}$$

if $x(t)$ is bandlimited to f_c and $f_s \ge 2f_c$. In this case, the last integral is equal to $x(t)$. A common choice for the pulse $p(t)$ is $p(t) = 1$ for $0 \le t \le T$ and $p(t) = 0$ otherwise. This can be realized with a **zero-order hold** circuit. In the case of the zero-order hold, $P(f) = T\,\mathrm{sinc}(Tf)e^{-j\pi Tf}$ is nonzero for $T|f| < 1$. Typically, we would take $T = 1/f_s < 1/(2f_c)$, in which case, $P(f)$ is nonzero for $|f| < 2f_c$. ***[For ZOH, show $|P(f)|$ is nearly flat for $|f| < f_c$ if $f_s \gg f_c$. This means that $H(f)$ can be any LPF that is flat over the passband. Draw approx. system with D/A followed by LPF. Run Matlab demo zohscript.]***

1.5. The Continuous-Time Domain and the Discrete-Time Domain

In an introductory course on signals and systems, one of the most important skills to learn is how to think about a signal in the time domain and in the frequency domain.

For example, convolution in one domain corresponds to multiplication in the other domain; a finite-duration pulse in one domain corresponds to a sinc function in the other domain.

In the study of the discrete-time processing of continuous-time signals, it is important to be able to think about signals in the continuous-time domain and in the discrete-time domain. More precisely, we need to keep in mind the relationship between $X(f)$ and $X_{\text{DTFT}}(f)$. In the general case, recall that we defined $\widetilde{X}(f) := \sum_n X(f - nf_s)$ and that from (1.16) and (1.17), we have

$$\widetilde{X}(f) = \frac{1}{f_s} X_{\text{DTFT}}(f/f_s). \tag{1.23}$$

By multiplying this equation by f_s, and replacing f with $f_s \cdot f$, we obtain

$$X_{\text{DTFT}}(f) = f_s \widetilde{X}(f_s f). \tag{1.24}$$

In the special case that the continuous-time signal is bandlimited and we sample fast enough; i.e., $f_s > 2f_c$, we find that

$$X(f) = \frac{1}{f_s} X_{\text{DTFT}}(f/f_s), \quad |f| \le f_s/2, \tag{1.25}$$

and

$$X_{\text{DTFT}}(f) = f_s X(f_s f), \quad |f| \le 1/2. \tag{1.26}$$

The relationship (1.26) is illustrated graphically in Figure 1.2. To transform the graph at the left into the one on the right, first mark off the points $\pm f_s/2$ so as to contain all the nonzero parts of $X(f)$. Then multiply all the heights by f_s and divide all the frequencies by f_s. The reverse transformation is given by (1.25). In this case, we start with a graph of $X_{\text{DTFT}}(f)$ for $|f| \le 1/2$. We divide all the heights by f_s and we multiply all the frequencies by f_s.

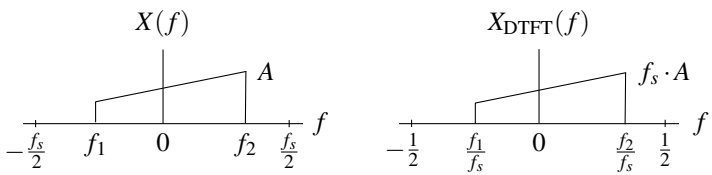


Figure 1.2. For a bandlimited signal that is sampled at faster than the Nyquist rate, it is easy to pass back and forth between the continuous-time Fourier transform of the signal and the DTFT of its samples.

As a simple application of the foregoing observations, consider the continuous-time system described by $Y(f) = H(f)X(f)$. Now assume that the input signal and

the system are bandlimited to f_c. Then the output is also bandlimited to f_c. By the sampling theorem, if we sample $y(t)$ at $f_s > 2f_c$, then sinc reconstruction of the samples $y(n/f_s)$ will recover $y(t)$. Now observe that

$$Y_{\mathrm{DTFT}}(f) = f_s Y(f_s f) = f_s H(f_s f) X(f_s f) = H(f_s f) X_{\mathrm{DTFT}}(f), \quad |f| \le 1/2.$$

This says that if we pass the discrete-time signal $x(m/f_s)$ through the discrete-time filter with transfer function (DTFT) equal to $H(f_s f)$ for $|f| \le 1/2$, then the discrete-time signal we get has DTFT $Y_{\mathrm{DTFT}}(f)$. In other words, a bandlimited, continuous-time system operating on bandlimited signals can be implemented with discrete-time signal processing as shown in Figure 1.3.

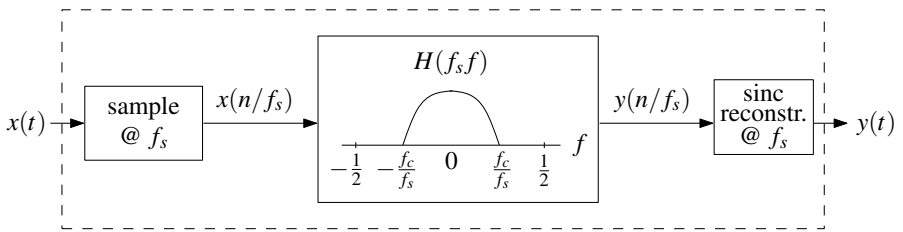


Figure 1.3. Implementation of a continuous-time system with discrete-time signal processing.

1.6. Bandlimited Waveforms and Systems

Suppose that $x(t)$ and $y(t)$ are both bandlimited to f_c. Then in addition to (1.15), there is an analogous formula for $Y(f)$. In other words, both $X(f)$ and $Y(f)$ have Fourier series expansions with Fourier coefficients being the waveform samples divided by f_s. Hence, by Parseval's equation for Fourier series,

$$\frac{1}{f_s}\int_{-f_s/2}^{f_s/2} X(f)\overline{Y(f)}\,df = \sum_{m=-\infty}^{\infty} [x(m/f_s)/f_s]\cdot\overline{[y(m/f_s)/f_s]}.$$

Keeping this in mind, and starting from Parseval's equation for continuous-time Fourier transforms (1.10), we can write

$$\begin{aligned}\int_{-\infty}^{\infty} x(\tau)\overline{y(\tau)}\,d\tau &= \int_{-\infty}^{\infty} X(f)\overline{Y(f)}\,df \\ &= \int_{-f_s/2}^{f_s/2} X(f)\overline{Y(f)}\,df \\ &= \frac{1}{f_s}\sum_{m=-\infty}^{\infty} x(m/f_s)\overline{y(m/f_s)}. \end{aligned} \tag{1.27}$$

In particular, if $h(t)$ is the impulse response of a linear time-invariant system that is bandlimited to f_c, then taking $y(\tau) = \overline{h(t-\tau)}$, we have

$$\int_{-\infty}^{\infty} h(t-\tau)x(\tau)\,d\tau = \frac{1}{f_s}\sum_{m=-\infty}^{\infty} h(t-m/f_s)x(m/f_s). \tag{1.28}$$

Since the output of a bandlimited system is bandlimited, there is no loss of information if we restrict attention to the output at the sample times $t = n/f_s$. In other words, if we compute the samples

$$\int_{-\infty}^{\infty} h(n/f_s-\tau)x(\tau)\,d\tau = \frac{1}{f_s}\sum_{m=-\infty}^{\infty} h([n-m]/f_s)x(m/f_s), \tag{1.29}$$

which is a discrete-time convolution, then we can recover the continuous-time convolution by applying the sinc reconstruction formula to the discrete-time convolution outputs. In summary, the convolution (1.29) is exactly what the discrete-time system in the middle of Figure 1.3 is doing; i.e., $h(n/f_s)/f_s$ is the discrete-time impulse response corresponding to the DTFT transfer function $H(f_s f)$ for $|f| \le 1/2$ (see Problem 1.36).

1.7. The Gap Between Theory and Practice

Theoretically, discrete-time signal processing says that for bandlimited signals, the continuous-time Fourier transform is completely determined by the DTFT of the signal samples via (1.14). Furthermore, for a bandlimited input to a bandlimited system, the output is completely determined by the discrete-time convolution (1.29).

Unfortunately, there are some practical constraints that tarnish the theory: (i) Computers cannot do the infinite sums in (1.15) and (1.29). (ii) If we truncate these sums to N terms and compute them for N values of f or n respectively, then we must do on the order of N^2 operations. For large N, this is not practical. Note that the sums in (1.15) and (1.29) are never finite — recall that bandlimited waveforms never have finite duration (except the zero waveform). (iii) Computers do not work with infinite-precision numbers.

To remove some of the tarnish due to truncation, we will study windowing techniques. We will apply them to spectral analysis and to filter design. To remove some of the tarnish due to the order N^2 computational complexity, we will invoke the **fast Fourier transform** both for spectral analysis and for convolution. To remove some of the tarnish due to finite-precision computation, we will briefly consider alternative filter realizations.

Problems

1.1. Given samples of a waveform $x(t)$ at distinct times $t_1, \ldots, t_{n+1}$, show that there is a unique polynomial $p(t)$ of degree at most n such that $p(t_k) = x(t_k)$ for $k = 1, \ldots, n+1$.

1.2. Let $x(t)$ be a continuous-time, periodic signal with period one and Fourier series coefficients x_n. Let $y(t) := x(t/T)$. Find the period of $y(t)$. How are the Fourier series coefficients of $y(t)$ related to the x_n?

1.3. Carry out the details to verify the formula

$$\int_{-T/2}^{T/2} e^{j2\pi t(n-m)/T}\, dt = T\,\delta_{nm}.$$

1.4. If $x(t)$ has period T and is real valued, show that its Fourier series coefficients x_n satisfy $x_{-n} = \overline{x_n}$. Then show that

$$x(t) = x_0 + 2\,\mathrm{Re}\left\{ \sum_{n=1}^{\infty} x_n e^{j2\pi nt/T} \right\}.$$

1.5. The foregoing problem suggests using MATLAB to compute the approximation, for real waveforms,

$$x_N(t) = x_0 + 2\,\mathrm{Re}\left\{ \sum_{n=1}^{N} x_n e^{j2\pi nt/T} \right\}.$$

If `x` $= [x_1, \ldots, x_N]$, then the above series can be computed with the command `dtfft(-t/T,x,[1:N])`, as noted in Section 1.2.2. Compute the $N = 5$ approximation of the square wave of period one whose values on $[-1/2, 1/2]$ are given by

$$x(t) = \begin{cases} 1, & |t| \le 1/4, \\ 0, & 1/4 < |t| \le 1/2. \end{cases}$$

Show your work to compute the coefficients x_n. Plot the square wave for $|t| \le 1$.

1.6. (a) Derive the geometric series formula

$$\sum_{n=0}^{N-1} z^n = \frac{1 - z^N}{1 - z}, \quad z \neq 1.$$

Hint: Put $S_N := 1 + z + \cdots + z^{N-1}$ and consider the difference $S_N - zS_N$.

(b) Show that

$$\sum_{n=0}^{\infty} z^n = \frac{1}{1 - z}, \quad |z| < 1.$$

1.7. ***Modulation Property.*** Let x_n have DTFT $X(f)$. Express the DTFTs of $y_n := x_n e^{j2\pi f_0 n}$ and $z_n := x_n \cos(2\pi f_0 n)$ in terms of $X(f)$.

1.8. ***Time Translation.*** Let x_n have DTFT $X(f)$, and let $y_n := x_{n-M}$. Show that $Y(f) = e^{-j2\pi fM} X(f)$. Note in particular that this implies $|Y(f)| = |X(f)|$.

1.9. Let $0 < a < 1$ and $|f_0| \le 1/2$.

(a) Show that the DTFT $Y(f) = 1/(1 - ae^{j2\pi f})$, which appears in Example 1.2.2, satisfies

$$|Y(f)|^2 = \frac{1}{1 - 2a\cos(2\pi f) + a^2}.$$

Observe that the denominator satisfies

$$(1-a)^2 \le 1 - 2a\cos(2\pi f) + a^2 \le (1+a)^2$$

and that $(1-a)^2 < 1$ and $(1+a)^2 > 1$. The denominator has a maximum at $f = \pm 1/2$ and a minimum at $f = 0$. For a typical value of a, sketch the denominator and then sketch $|Y(f)|^2$. What happens as $a \to 1$?

(b) Show that the DTFT of $y_n = a^{|n|}$ is

$$Y(f) = \frac{1 - a^2}{1 - 2a\cos(2\pi f) + a^2}$$

Sketch $Y(f)$.

(c) Find the DTFT of $y_n = a^n e^{j2\pi f_0 n}$ for $n \ge 0$ and $y_n = 0$ for $n < 0$.

(d) Find the DTFT of $y_n = a^{|n|} e^{j2\pi f_0 n}$.

1.10. In this problem, you explore how Example 1.2.1 changes if $f_0 \notin [-1/2, 1/2]$. To begin, we use the fact that every f_0 can be expressed in the form $f_0 = \widehat{f_0} + k_0$ for some some $\widehat{f_0} \in [-1/2, 1/2]$ and some integer k_0.

(a) Show that $e^{j2\pi \widehat{f_0} n} = e^{j2\pi f_0 n}$.

(b) Recall that in Example 1.2.1, $Y(f)$ is defined by periodic repetition; i.e., when $f_0 \in [-1/2, 1/2]$,

$$Y(f) = \sum_{k=-\infty}^{\infty} \delta([f-k] - f_0).$$

Show that

$$\sum_{k=-\infty}^{\infty} \delta([f-k] - \widehat{f_0}) = \sum_{k=-\infty}^{\infty} \delta([f-k] - f_0).$$

In other words, the formula for $Y(f)$ holds even when $f_0 \notin [-1/2, 1/2]$.

1.11. Sketch the DTFT of $x_n := e^{j2\pi(4/3)n}$ for $-1/2 \le f \le 1/2$.

1.12. Let x_n be a finite-duration signal of length N with $x_n = 0$ for $n < 0$ and $n \ge N$. Denote its DTFT by $X(f)$. Let M be a positive integer, and define the new signal

$$y_n := \sum_{i=-\infty}^{\infty} x_{n-iM}.$$

(a) Show that y_n has period M.

(b) Let

$$Y_k := \sum_{n=0}^{M-1} y_n e^{-j2\pi kn/M}$$

denote the DFT of $y_0, \ldots, y_{M-1}$. Express Y_k in terms of $X(f)$.

1.13. Derive Parseval's formula (1.4). *Hint:* Start with the left-hand side of (1.4) and replace $x(t)$ with its Fourier-series expansion (do not substitute for $y(t)$). Then rearrange and simplify.

1.14. Show that $Y(f)$ defined in Section 1.2.2 has period one.

1.15. Carry out the details to verify the integral formula for y_n in Section 1.2.2.

1.16. **MATLAB.** Let $a = 0.75$. On one graph, plot `abs(Y(f))` from Example 1.2.2 and the absolute value of its approximation using the function `dtft` given in Section 1.2.2. In order to use `dtft`, it is convenient to put `nvec=[0:N-1]` and `x=[a.^nvec]`. Make one graph with $N = 10$ and make another graph with $N = 20$. Repeat for `a=0.75*exp(j*2*pi*f0)` where `f0=0.25`.

1.17. Derive Parseval's formula (1.10). *Hint:* Start with the left-hand side of (1.10) and replace $x(t)$ with its representation as the inverse Fourier transform of $X(f)$ (do not substitute for $y(t)$). Then rearrange and simplify.

1.18. Let t be fixed. If $y(\tau) = \overline{h(t-\tau)}$, show that $Y(f) = \overline{H(f)} e^{-j2\pi ft}$.

1.19. Show that a waveform $x(t)$ is real value if and only if its transform $X(f)$ satisfies $\overline{X(f)} = X(-f)$.

1.20. Let $x(t)$ be a periodic waveform of period T and Fourier series coefficients x_n. Suppose that $x(t)$ is applied to a linear time-invariant system with impulse response $h(t)$. Let $y(t)$ denote the system output.

(a) Show that $y(t)$ is periodic.

(b) Let y_n denote the nth Fourier series coefficient of $y(t)$. Show that $y_n = H(n/T)x_n$, where $H(f)$ is the Fourier transform of $h(t)$.

1.21. **MATLAB.** Modify the script in Section 1.3.3 to plot the Fourier transform of $x(t) = \mathrm{sinc}^2 t$ for $|f| \le 3$. Use values of $\mathrm{sinc}^2 t$ for $|t| \le 3$. Make one plot with $f_s = 6$ and another with $f_s = 3$.

1.22. **MATLAB.** Modify the script in Section 1.3.3 to plot the Fourier transform of $x(t) = 1/(1+t^2)$ for $|f| \le 1$. Use values of t for $|t| \le 5$. Although this signal

is not bandlimited, its spectrum, $X(f) = \pi e^{-2\pi|f|}$, decays rapidly. Try using $f_s = 10$. Also plot the exact transform on top of the approximation.

1.23. Show that $\widetilde{X}(f)$ defined in (1.17) has period f_s.

1.24. Let $x(t)$ have continuous-time Fourier transform

$$X(f) = \begin{cases} 1, & |f| \le 10, \\ 1/10, & 10 < |f| \le 11, \\ 0, & \text{otherwise.} \end{cases}$$

Sketch, and carefully label, the DTFT of the samples $x(n/f_s)$ if $f_s = 20$.

1.25. Show that

$$\int_{-f_c}^{f_c} e^{j2\pi f[t-m/f_s]}\,df = 2f_c\,\mathrm{sinc}(2f_c[t-m/f_s]),$$

where

$$\mathrm{sinc}(t) := \begin{cases} \dfrac{\sin(\pi t)}{\pi t}, & t \neq 0, \\ 1, & t = 0. \end{cases}$$

1.26. The preceding problem implies that the Fourier transform of $2f_c\,\mathrm{sinc}(2f_c[t - m/f_s])$ is $e^{-j2\pi fm/f_s} I_{[-f_c,f_c]}(f)$, where I denotes the **indicator function**: If $B \subset \mathbb{R}$,

$$I_B(f) := \begin{cases} 1, & f \in B, \\ 0, & f \notin B. \end{cases}$$

Use this fact to show that

$$\int_{-\infty}^{\infty} 2f_c\,\mathrm{sinc}(2f_c[t-m/f_s])\,dt = 1.$$

1.27. Consider a periodic, continuous-time signal $x(t)$ with period T_0. Given any sampling period T_s, we could obtain samples $x(mT_s)$. Now consider a longer sampling period $T_s' = T_s + \Delta$, where $\Delta > 0$. Find Δ such that $x(mT_s') = x(mT_s)$ for all m.

1.28. Let $x(t)$ be bandlimited to f_c. If $y(t) = x(t)\cos(2\pi f_0 t)$, determine the Nyquist rate of $y(t)$.

1.29. Let $x(t)$ have Fourier transform $X(f) = (1+f)/2$ for $|f| \le 1$.

(a) By hand, sketch $X(f)$ for $|f| \le 4$.

(b) What is the Nyquist rate?

(c) By hand, on one graph, sketch $X(f+f_s)$, $X(f)$, and $X(f-f_s)$ when f_s is equal to the Nyquist rate.

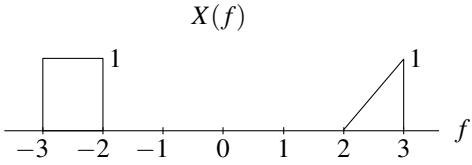
(d) Repeat part (c) if f_s is just a little bit less than the Nyquist rate.

1.30. Let $x(t) = e^{j2\pi t}$.

(a) By hand, sketch $X(f)$.

(b) By hand, sketch $\widetilde{X}(f)$ for $|f| \le f_s/2$ if $f_s = 1.5$.
(c) If the sinc reconstruction formula is applied to the samples $x(m/f_s)$ with $f_s = 1.5$, what signal will be reconstructed?

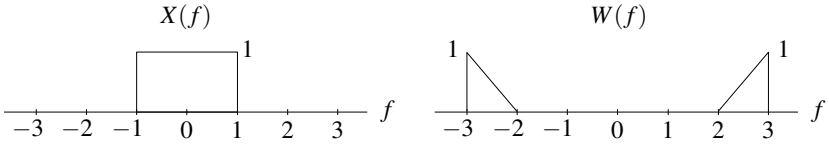
1.31. A signal $x(t)$ with continuous-time Fourier transform



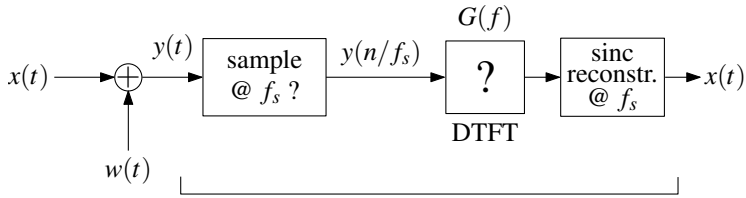
is sampled at $f_s = 2$. Sketch $\widetilde{X}(f)$ for $|f| \le f_s/2$ and $X_{\text{DTFT}}(f)$ for $|f| \le 1/2$.

1.32. The continuous-time signal $x(t) = e^{j2\pi(5)t} + e^{j2\pi(10)t}$ is to be sampled and applied to the discrete-time ideal lowpass filter with DTFT $H(f) = I_{[-1/4,1/4]}(f)$ so as to remove the high-frequency term $e^{j2\pi(10)t}$. Determine the highest possible sampling rate f_s for which this is possible.

1.33. Suppose that $y(t) = x(t) + w(t)$, where $x(t)$ is a lowpass signal and $w(t)$ is bandpass noise with Fourier transforms

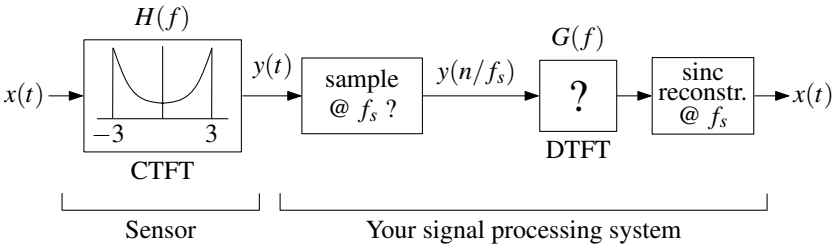


(a) Sketch $Y(f)$.
(b) Instead of passing $y(t)$ through a continuous-time lowpass filter to remove $w(t)$ and leave only $x(t)$, you are to design a system



that samples $y(t)$ and passes the samples $y(n/f_s)$ through a discrete-time ideal lowpass filter of impulse response g_n and corresponding DTFT $G(f)$ (of period one) such that the output of the discrete-time filter is $y(n/f_s)$, which can then be passed through a D/A to supply $x(t)$. (i) What sampling rate f_s will you use? (ii) Sketch your choice of $G(f)$.

1.34. A signal $x(t)$, bandlimited to $f_c = 3$, is measured using a sensor with transfer function $H(f) = 1 + f^2$ for $|f| \le 3$ and $H(f) = 0$ for $|f| > 3$.



Specify a sampling rate f_s for $y(t)$ and a discrete-time transfer function $G(f)$ (give a formula) such that the output of the D/A is $x(t)$. In other words, your discrete-time filter $G(f)$ should "undo" the distortion $H(f)$ of the sensor.

1.35. Use the following hints to give an alternative derivation of the **Poisson summation formula** (1.17). *Hints:* (*i*) Observe that $x(t_0)\delta(t-t_0) = x(t)\delta(t-t_0)$. (Consider the two cases $t = t_0$ and $t \neq t_0$.)
(*ii*) Apply the previous hint to the impulse sampling

$$\sum_{m=-\infty}^{\infty} x(m/f_s)\delta(t-m/f_s). \qquad (*)$$

(*iii*) Recall that the Fourier transform of a product is the convolution of the transforms; i.e., the Fourier transform of $x(t)y(t)$ is

$$\int_{-\infty}^{\infty} X(f-\nu)Y(\nu)\,d\nu.$$

(*iv*) Recall that if $y(t)$ is periodic with period T and its Fourier series expansion is $\sum_m y_m e^{j2\pi tm/T}$, then its Fourier transform is $\sum_m y_m \delta(f-m/T)$. Use this to find the Fourier transform of the periodic impulse train $y(t) = \sum_m \delta(t-m/f_s)$.
(*v*) You should now be able to show that the Fourier transform of $(*)$ is equal to $f_s \sum_m X(f-mf_s)$.

1.36. Given a continuous-time transfer function $H(f)$ with corresponding impulse response $h(t)$, consider the discrete-time system whose transfer function (or DTFT) is equal to $H(f_s f)$ for $|f| \leq 1/2$, where $f_s > 2f_c$. Find the discrete-time impulse response by writing down the inverse DTFT, doing a change of variable, and using the fact that $H(f) = 0$ for $|f| > f_c$.

1.37. Use the Poisson summation formula (1.17) to derive the Fourier transform pair

$$\sum_{m=-\infty}^{\infty} \delta(t-m/f_s) \leftrightarrow f_s \sum_{n=-\infty}^{\infty} \delta(f-nf_s);$$

i.e., the Fourier transform of an **impulse train** is an impulse train. *Hints:* In (1.17), let $X(f) = \delta(f)$ so that $x(t) \equiv 1$. Then use the fact that the inverse Fourier transform of $e^{-j2\pi f t_0}$ is $\delta(t-t_0)$.

1.38. Formula (1.27) provides a numerical integration procedure for products of bandlimited time functions. What about the integral of a single such function, say $\int_{-\infty}^{\infty} x(\tau)\,d\tau$? Recall that the Fourier transform of $y(\tau) \equiv 1$ is the **unit impulse**, or **Dirac delta function**, $Y(f) = \delta(f)$, which is certainly bandlimited. In this case, (1.27) reduces to

$$\int_{-\infty}^{\infty} x(\tau)\,d\tau = \frac{1}{f_s} \sum_{m=-\infty}^{\infty} x(m/f_s).$$

Now derive this formula in a different way by integrating (1.19) with respect to t and using the result of Problem 1.26.

1.39. **MATLAB.** Use the result of the preceding problem with $f_s = 0.32$ and MATLAB to approximate $\int_{-\infty}^{\infty} \sin(\tau)/\tau\,d\tau$. Explain why it is permissible to use $f_s = 0.32$.

1.40. If $x(t)$ is bandlimited, show that the $x(t)^2$ is also bandlimited, but has twice the cutoff frequency of $x(t)$. *Hint:* Think about setting up the convolution of $X(f)$ with itself graphically.

1.41. **MATLAB.** Suppose we approximate the right-hand side of (1.28) by replacing the infinite sum with a finite sum going from $m = -M$ to $m = M$. Then we can implement the approximation in MATLAB as follows. Assume that $h(t)$ and $x(t)$ are computed by MATLAB functions stored in the M-files `h.m` and `x.m`. The command `y=blconv(t,'h','x',fs,M)` computes the desired approximation, assuming that `t`, `fs`, and `M` have been defined and that the function `blconv`, which stands for "bandlimited convolution," is stored in `blconv.m` and defined as follows

```
function y = blconv(t,hfun,xfun,fs,M)
tauvec = [-M:M]/fs;
[tmat,taumat] = meshgrid(t,tauvec);
y = feval(xfun,tauvec)*feval(hfun,tmat-taumat)/fs;
```

Plot the convolution of $h(t) = 2f_c\,\mathrm{sinc}(2f_c t)$ and $\mathrm{sinc}^2(t)$ for $f_c = 2$ and for $f_c = 1/2$. To plot your results for t in the interval $[-10, 10]$, the command `t=linspace(-10,10,200)` may be helpful. What sampling frequency f_s did you use? What is the smallest value of M that you were able to use?

1.42. **MATLAB.** In this problem you will apply an ideal lowpass filter to the $N = 5$ square wave approximation of Problem 1.5. Use the variable `lpfcf` to denote the lowpass filter cutoff frequency. Plot the input signal when `N=5` and the output signal for `lpfcf` equal to 6, 4, and 2. Plot the output signal values for $t \in [-1, 1]$. What can you plot to check that your output is correct in each case? Do it. The function `blconv` of the preceding problem may be helpful.

CHAPTER 2
Discrete-Time Convolution

At the end of Chapter 1, we showed that signals and linear time-invariant systems that are bandlimited are equivalent to discrete-time signals and systems. In particular, (1.29) shows that continuous-time convolution corresponds to the discrete-time convolution of the sampled impulse response and signal, divided by f_s.

In this chapter, we make a few basic observations about discrete-time convolution that will be very helpful in the study of digital signal processing.

The **discrete-time convolution** of two sequences x and y is defined by

$$(x*y)_n := \sum_{m=-\infty}^{\infty} x_m y_{n-m}.$$

There are three situations that we need to consider.

1. Both x and y are finite-duration signals.
2. The signal x is finite duration, but y is not.
   (a) We only care about a finite, predetermined range of n that is not too large. We show how to reduce this to a type-1 problem.
   (b) The range of n is very large or is indeterminate. The latter would be the case in real-time signal processing. We show how to reduce this to a collection of type-1 problems whose results can be added together in what is known as the **overlap-add** method.
3. Neither x nor y is finite duration, but we use the approximation

$$(x*y)_n \approx \sum_{m=M_1}^{M_2} x_m y_{n-m},$$

   which falls into the type 2(a) or 2(b) category.

2.1. Convolution of Two Finite-Duration Signals

Suppose that x_m is a finite-duration signal on $M_1^x \le m \le M_2^x$ and y_m is a finite-duration signal on $M_1^y \le m \le M_2^y$. It is easy to see that the convolution $(x*y)_n$ can be nonzero only for ***[set this up graphically]***

$$M_1^x + M_1^y \le n \le M_2^x + M_2^y,$$

in other words, for

$$\text{sum of lower limits} \le n \le \text{sum of upper limits.}$$

Notice that the duration of x_m is $M_2^x - M_1^x + 1$ (difference of limits plus one), the duration of y_m is $M_2^y - M_1^y + 1$, and the duration of $(x*y)_n$ is at most

$$\begin{aligned}(M_2^x + M_2^y) - (M_1^x + M_1^y) + 1 &= (M_2^x - M_1^x + 1) + (M_2^y - M_1^y + 1) - 1 \\ &= \text{duration}(x) + \text{duration}(y) - 1.\end{aligned}$$

In other words, the duration of the convolution is the sum of durations minus one.

If `x` and `y` are vectors in MATLAB, their convolution is easily found with the command `z=conv(x,y)`. ***Warning:*** **Later we will see that if `x` and `y` are large, then `conv` is not very efficient, and other methods are faster (see Section** 3.3**).** Notice that `x` and `y` have no time index information. However, if we know M_1^x and M_1^y, then the appropriate vector of time indexes corresponding to the convolution are easy to find, as shown in the following script.

```
x = ones(1,5); M1x = -2;
y = ones(1,9); M1y = -4;
z = conv(x,y);
M2x = M1x+length(x)-1;
M2y = M1y+length(y)-1;
n = [M1x+M1y:M2x+M2y];
stem(n,z,'filled')
```

2.2. Convolution of a Finite-Duration Signal and an Infinite-Duration Signal

If x_m is a finite-duration signal on $M_1^x \le m \le M_2^x$, then

$$(x*y)_n = \sum_{m=M_1^x}^{M_2^x} x_m y_{n-m}.$$

2.2.1. Limited Observation Window

If n is restricted to $N_1 \le n \le N_2$, then the subscript on y varies from a minimum of $N_1 - M_2^x$ to a maximum of $N_2 - M_1^x$. This suggests that if we define the finite-duration signal

$$\hat{y}_m := \begin{cases} y_m, & N_1 - M_2^x \le m \le N_2 - M_1^x, \\ 0, & \text{otherwise,} \end{cases}$$

then $(x*\hat{y})_n = (x*y)_n$ for $N_1 \le n \le N_2$. A ***[graphical argument]*** with y_{n-m}, $\hat{y}_{n-m}$, and x_m shows that this is indeed the case. Some caution is needed, however. Observe that $(x*\hat{y})_n$ is nonzero for

$$M_1^x + (N_1 - M_2^x) \le n \le M_2^x + (N_2 - M_1^x)$$

or equivalently,

$$N_1 - (M_2^x - M_1^x) \le n \le N_2 + (M_2^x - M_1^x),$$

which is a larger range than the $N_1 \le n \le N_2$ that we are interested in. The point here is that if we use `z=conv(x,yhat)`, then `z` contains many entries that we do not need. Specifically, there are $M_2^x - M_1^x$ entries at the beginning and end of `z` that we do not need. Our desired data is.

```
z((M2x-M1x)+1:end-(M2x-M1x)).
```

2.2.2. Unlimited Observation Window (The Overlap-Add Method)

In this case, we break up the infinite sequence y into infinitely many nonoverlapping pieces, each of *finite* length M. More specifically, if we put

$$y_m^{(k)} := \begin{cases} y_m, & kM \le m < (k+1)M, \\ 0, & \text{otherwise}, \end{cases}$$

then

$$y_m = \sum_{k=-\infty}^{\infty} y_m^{(k)}.$$

Since convolution is linear, we can write

$$x*y = x*\left(\sum_{k=-\infty}^{\infty} y^{(k)}\right) = \sum_{k=-\infty}^{\infty} x*y^{(k)}.$$

Notice that term $x*y^{(k)}$ is computable because it is the convolution of two finite-length sequences. Also, $(x*y^{(k)})_n$ is nonzero only for

$$M_1^x + kM \le n \le M_2^x + (k+1)M - 1.$$

Since we can always incorporate a delay into x, there is no loss of generality if we assume x is causal, which implies $M_1^x = 0$. Then $x*y^{(k)}$ is nonzero only for

$$kM \le n \le [(k+1)M - 1] + M_2^x.$$

Thus, even though $y_m^{(k)}$ is nonzero only for $kM \le n \le (k+1)M-1$, the convolution $x*y^{(k)}$ extends M_2^x samples beyond; i.e., it overlaps with the next term $x*y^{(k+1)}$. To prevent $x*y^{(k)}$ from overlapping with $x*y^{(k+2)}$, we require

$$[(k+1)M-1]+M_2^x < (k+2)M,$$

which simplifies to $M > M_2^x - 1$ or $M \ge M_2^x$. Since the length of x is M_2^x+1, this says that the length of the blocks into which we partition y should be at least the length of x minus one. We always assume this to be the case.

Up to this point we have assumed that y is doubly infinite. However, in practice, we do not start measuring y until some finite time, which we take here to be zero. So the first step is to collect the data $y_0,\ldots,y_{M-1}$. This allows us to compute $(x*y^{(0)})_n$ for $n=0,\ldots,M+M_2^x-1$. However, we only output $(x*y^{(0)})_n$ for $n=0,\ldots,M-1$. We then collect $y_M,\ldots,y_{2M-1}$ and compute $(x*y^{(1)})_n$ for $n=M,\ldots,2M+M_2^x-1$. It is convenient to display the results in more detail as

$$\begin{array}{ll} (x*y^{(0)})_n & \text{for } n=0,\ldots,M-1,M,M+1,\ldots,M+M_2^x-1 \\ (x*y^{(1)})_n & \text{for } n= \qquad\qquad M,M+1,\ldots,M+M_2^x-1,\ldots,2M,\ldots,2M+M_2^x-1. \end{array}$$

Since $(x*y^{(2)})_n$ does not start until time $2M$, we can provide the next M outputs

$$(x*y)_n = (x*y^{(0)})_n + (x*y^{(1)})_n, \quad n=M,\ldots,2M-1.$$

In general, after computing $(x*y^{(k)})_n$ for $n=kM,\ldots,(k+1)M-1+M_2^x$, we output

$$(x*y)_n = (x*y^{(k-1)})_n + (x*y^{(k)})_n, \quad n=kM,\ldots,(k+1)M-1.$$

To express this in MATLAB, suppose that `oldz` contains $x*y^{(k-1)}$ and `newz` contains $x*y^{(k)}$, both of length $M+M_2^x$. Then we output the vector of length M given by

```
[oldz(M+1:end)+newz(1:M2x)   newz(M2x+1:M)];
```

Problems

2.1. If x, y, and z are sequences with lengths N_x, N_y, and N_z, find the length of $x*y*z$.

CHAPTER 3
The DFT and the FFT

In Chapter 1, we saw that signal processing for continuous-time, bandlimited waveforms and systems can be accomplished by discrete-time signal processing. However, computers can only evaluate *finite* sums.

Recall that for an infinite sequence x_n, its DTFT is

$$X(f) := \sum_{m=-\infty}^{\infty} x_m e^{-j2\pi fm} \approx \sum_{m=M_1}^{M_2} x_m e^{-j2\pi fm}$$

for large M_2 and large (negative) M_1. The sum on the right contains $M_2 - M_1 + 1$ terms. In order to write the finite sum as a sum starting from zero, we can make the change of variable $n = m - M_1$ to get

$$X(f) \approx \sum_{n=0}^{M_2-M_1} x_{n+M_1} e^{-j2\pi f(n+M_1)}.$$

Although the foregoing approximation involves only a finite sum, a computer cannot evaluate it for all values of f in one period, say $|f| \le 1/2$. Instead, the computer can only evaluate the sum for finitely many values of f. Let $N := M_2 - M_1 + 1$, and let $f = k/N$ to get

$$\begin{aligned} X(k/N) &\approx \sum_{n=0}^{N-1} x_{n+M_1} e^{-j2\pi k(n+M_1)/N} \\ &= e^{-j2\pi kM_1/N} \sum_{n=0}^{N-1} x_{n+M_1} e^{-j2\pi kn/N}. \end{aligned} \tag{3.1}$$

Regarding this last sum as a function of k, observe that it has period N; i.e., replacing k with $k+N$ does not change the value of the sum. So we only need to evaluate the sum for $k = 0, \ldots, N-1$.

3.1. The Discrete Fourier Transform (DFT)

Given a finite sequence $y_0, \ldots, y_{N-1}$, its **discrete Fourier transform** (DFT) is

$$Y_k := \sum_{n=0}^{N-1} y_n e^{-j2\pi kn/N}. \tag{3.2}$$

It is easy to show that Y_k is a periodic function of k with period N. We show below in Section 3.1.8 that the sequence y_n can be recovered from the DFT sequence $Y_0,\ldots,Y_{N-1}$ by the **inverse DFT** (IDFT)

$$y_n = \frac{1}{N}\sum_{k=0}^{N-1} Y_k e^{j2\pi kn/N}. \tag{3.3}$$

Of course the right-hand side is a periodic function of n with period N. Hence, although y_n is only defined for $n = 0,\ldots,N-1$, we often think of it as being an infinite-duration periodic signal with period N.

3.1.1. Zero Padding

Given a sequence $y_0,\ldots,y_{M-1}$, its DTFT is

$$Y(f) = \sum_{n=0}^{M-1} y_n e^{-j2\pi fn}.$$

Now consider the zero-padded vector

$$z = [y_0,\ldots,y_{M-1},0,\ldots,0].$$

If the length of z is N, then the DFT of z is

$$\begin{aligned}\sum_{n=0}^{N-1} z_n e^{-j2\pi kn/N} &= \sum_{n=0}^{M-1} y_n e^{-j2\pi kn/N} \\ &= Y(k/N).\end{aligned}$$

In other words, given M data samples $y_0,\ldots,y_{M-1}$, computing a zero-padded DFT gives you samples of $Y(f)$ that are more closely spaced in frequency.

Now suppose that x_n is a **causal sequence** (i.e., $x_n = 0$ for $n < 0$) of infinite duration with DTFT

$$X(f) = \sum_{n=0}^{\infty} x_n e^{-j2\pi fn}.$$

Put

$$X_M(f) := \sum_{n=0}^{M-1} x_n e^{-j2\pi fn}.$$

Then as $M \to \infty$, $X_M(f) \to X(f)$. For fixed M, the DFT of $[x_0,\ldots,x_{M-1},0,\ldots,0]$ zero padded to length N is

$$X_M(k/N) = \sum_{n=0}^{M-1} x_n e^{-j2\pi kn/N}, \quad k = 0,\ldots,N-1.$$

In other words, for fixed M, if we compute the DFT of $[x_0,\ldots,x_{M-1},0,\ldots,0]$ with more and more zeros padded, we do *not* get closer to $X(f)$, we get more closely spaced frequency samples of $X_M(f)$. This is illustrated in Figure 3.1.

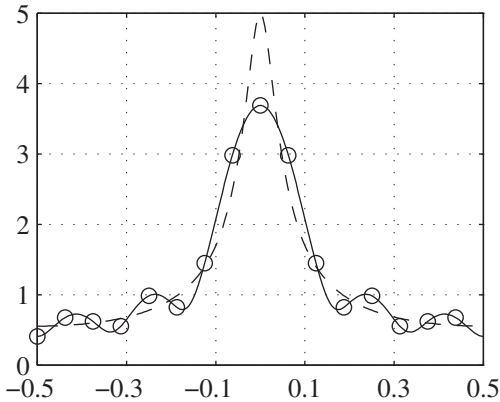


Figure 3.1. The DTFT of an infinite sequence (dashed line), the DTFT of the first M elements (solid line), and the DFT of M elements with zero padding (circles).

3.1.2. The Fast Fourier Transform (FFT)

If $\mathtt{y} = [y_0,\ldots,y_{N-1}]$, then the DFT of $\mathtt{y}$, $\mathtt{Y} = [Y_0,\ldots,Y_{N-1}]$, can be computed in MATLAB with the command `Y = fft(y)`. Here **FFT** stands for **fast Fourier transform**. The FFT is a special algorithm that computes the DFT very quickly.

Since the DFT is periodic,[1]

$$\begin{aligned} Y_{-1} &= Y_{-1+N} &&= Y_{N-1} \\ Y_{-2} &= Y_{-2+N} &&= Y_{N-2} \\ &\vdots \\ Y_{-N/2} &= Y_{-N/2+N} &&= Y_{N/2}. \end{aligned}$$

So, to plot Y_k for $k=-N/2$ to $k=N/2-1$, we need to take

$$\mathtt{Y} = [Y_0,\ldots,Y_{N/2-1},Y_{N/2},\ldots,Y_{N-1}]$$

and convert it to

$$[Y_{N/2},\ldots,Y_{N-1},Y_0,Y_1,\ldots,Y_{N/2-1}].$$

[1] If N is not even, then $N/2$ should be replaced the greatest integer that is less than or equal to $N/2$, which is denoted by $\lfloor N/2 \rfloor$ and is given by `floor` in MATLAB.

This is done with the MATLAB command `fftshift`. The corresponding vector of k values can be given by `k=[0:N-1]-N/2`, and we could then use the command `plot(k, fftshift(Y))`.

3.1.3. Using the FFT to Approximate the DTFT

If we are approximating the sum in (3.1), then we would use `k/N` to have the horizontal axis run from $-1/2$ to $1/2$. The MATLAB function `dtftfft` given below computes the right-hand side of (3.1) using `fft` and also takes care of the bookkeeping to return to you the corresponding frequencies f in $[-1/2, 1/2]$. The command for this is `[y,f]=dtftfft(x,M1)`, where the elements of `x` are $[x_{M_1}, \ldots, x_{M_2}]$. If `M1` is zero, it can be omitted and you can write `[y,f]=dtftfft(x)` instead. There is also an optional third argument if you want to force `fft` to zero-pad `x`. Remember, the spacing between frequencies in the DFT and the FFT is `1/length(x)`. The required command is `[y,f]=dtftfft(x,M1,N)` to force `dtftfft` to add enough zeros to `x` to make its length N. If the length of `x` is greater than N, `x` will be truncated to N elements. Using $N = 0$ causes `dtftfft` to zero-pad `x` so its length is a power of 2. Using a negative value of N causes `dtftfft` to zero pad `x` so that its length is a power of 2 that is not only greater than or equal to the length of `x` but also greater than or equal to $-N$. Note that `fft` operates most efficiently on vectors whose length is a power of 2. Here is the function.

```
function [y,f] = dtftfft(x,varargin)
N = length(x);
if nargin==3
   N = varargin{2};
   if N==0 % If N=0, zero pad x to a power of 2.
      N = 2^ceil(log2(length(x)));
   elseif N<0 % If N<0, zero pad x to a power of 2 >= -N
      N = 2^ceil(log2(max(length(x),-N))); % and >= length(x)
   end
end
y = fft(x,N);
k = [0:N-1];
if nargin>=2
   M1 = varargin{1};
   if M1~=0
      y = exp(-j*2*pi*M1/N*k).*y;
   end
end
y = fftshift(y);
% For N=2m even, change 0,...,m,...,2m-1
% to -m,...,0,...,m-1.  For N=2m+1 odd, change
% 0,...,m-1,m,m+1,...,2m to -m,...,-1,0,1,...,m.
```

```
% Then divide by N to get frequencies in [-1/2,1/2].
f = (k-floor(N/2))/N;
```

3.1.4. Using the FFT to Approximate the Continuous-Time Fourier Transform

It is very easy to modify the MATLAB script in Section 1.3.3 to use `dtftfft` instead of `dtft`. In the script, replace the two lines `f=...` and `y=...` with the three lines

```
[y,f] = dtftfft(x,n1,0);  % Compute DTFT on [-1/2,1/2]
f = f*fs;                 % Convert freq. to [-fs/2,fs/2]
y = y/fs;                 % Scale DTFT
```

If your curve does not have enough points on it, you can change the third argument of `dtftfft` to increase the number of points in `f` as discussed above in Section 3.1.3.

3.1.5. Summing a Periodic Sequence over a Period

Let z_n have period N. Then for any m,

$$\sum_{n=m}^{m+(N-1)} z_n = \sum_{n=0}^{N-1} z_n.$$

This is most easily seen pictorially. For example, if z_n has period $N = 5$, we see from the diagram

$$\begin{array}{r|ccc|ccccc|cc} n: & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ z_n: & a_0 & a_1 & a_2 & a_3 & a_4 & a_0 & a_1 & a_2 & a_3 & a_4 \end{array}$$

that $z_0 + \cdots + z_4$ and $z_3 + \cdots + z_7$ are both equal to $a_0 + \cdots + a_4$.

A simple application of the foregoing is to the IDFT formula (3.3) when $N = 2M + 1$. Then

$$y_n = \frac{1}{N} \sum_{k=-M}^{M} Y_k e^{j2\pi kn/N},$$

where we have used the fact that since Y_k and $e^{j2\pi kn/N}$ have period N, so does their product.

3.1.6. Evaluation of Fourier-Series Coefficients

We say that a continuous-time, periodic function is bandlimited if its Fourier-series expansion has only finitely many terms, say

$$x(t) = \sum_{n=-M}^{M} x_n e^{j2\pi nt/T}.$$

Let $N = 2M+1$, and define $\{\tilde{x}_n\}_{n=-\infty}^{\infty}$ to be the N-periodic extension of $\{x_n\}_{n=-M}^{M}$. Then we can write

$$\begin{aligned} x(kT/N) &= \sum_{n=-M}^{M} x_n e^{j2\pi kn/N} \\ &= \sum_{n=-M}^{M} \tilde{x}_n e^{j2\pi kn/N} \\ &= \underbrace{\sum_{n=0}^{N-1} \tilde{x}_n e^{j2\pi kn/N}}_{N \text{ IDFT of } \tilde{x}_n}, \end{aligned}$$

since the terms, as a function of n, have period N. It follows that $\tilde{x}_n$ is $1/N$ times the DFT of $x(kT/N)$; i.e.,

$$\tilde{x}_n = \frac{1}{N}\sum_{k=0}^{N-1} x(kT/N)e^{-j2\pi kn/N}, \quad \text{for all } n, \tag{3.4}$$

$$= x_n, \qquad |n| \le M. \tag{3.5}$$

We can now write

$$\begin{aligned} x(t) &= \sum_{n=-M}^{M} x_n e^{j2\pi nt/T} \\ &= \sum_{n=-M}^{M} \tilde{x}_n e^{j2\pi nt/T} \\ &= \sum_{n=-M}^{M} \left[\frac{1}{N}\sum_{k=0}^{N-1} x(kT/N)e^{-j2\pi kn/N}\right] e^{j2\pi nt/T}, \quad \text{by (3.4)}, \\ &= \frac{1}{N}\sum_{k=0}^{N-1} x(kT/N)\cdot N\frac{\operatorname{sinc}(N[t/T-k/N])}{\operatorname{sinc}(t/T-k/N)}, \quad \text{by Problem 3.3}, \\ &= \sum_{k=0}^{N-1} x(kT/N)\frac{\operatorname{sinc}(N[t/T-k/N])}{\operatorname{sinc}(t/T-k/N)}. \end{aligned}$$

We thus have a **sampling theorem** and **sinc reconstruction formula** for periodic signals that are bandlimited. Notice that here we get exact reconstruction with a finite number of terms! Furthermore, from (3.4)–(3.5) we can recover the x_n *exactly* without computing the integral that is normally required for finding Fourier-series coefficients!

3.1.7. The Geometric Series

In order to derive the inverse DFT, we need the **geometric series** formula (Problem 1.6 in Chapter 1),

$$\sum_{k=0}^{N-1} z^k = \begin{cases} N, & z = 1, \\ \dfrac{1-z^N}{1-z}, & z \neq 1. \end{cases}$$

We will need this identity when z has the form $z = e^{j2\pi(m-n)/N}$. When z has this form, $z = 1$ if and only if $(m-n)/N$ is an integer. Now, if $0 \leq m, n \leq N-1$, then $(m-n)/N$ is an integer if and only if $m = n$. Furthermore,

$$z^N = \left[e^{j2\pi(m-n)/N}\right]^N = e^{j2\pi(m-n)} = 1,$$

and then $(1-z^N)/(1-z) = 0$.

3.1.8. Derivation of the IDFT

We can now establish the IDFT formula (3.3). Fix $0 \leq m \leq N-1$. Then

$$\begin{aligned} \frac{1}{N}\sum_{k=0}^{N-1} Y_k e^{j2\pi km/N} &= \frac{1}{N}\sum_{k=0}^{N-1}\left(\sum_{n=0}^{N-1} y_n e^{-j2\pi kn/N}\right) e^{j2\pi km/N} \\ &= \frac{1}{N}\sum_{n=0}^{N-1} y_n \sum_{k=0}^{N-1} e^{j2\pi k(m-n)/N} \\ &= \frac{1}{N}\sum_{n=0}^{N-1} y_n \underbrace{\sum_{k=0}^{N-1}\left[e^{j2\pi(m-n)/N}\right]^k}_{N\delta_{mn}} \\ &= \frac{1}{N}\sum_{n=0}^{N-1} y_n \cdot N\delta_{mn} = y_m. \end{aligned}$$

3.2. Circular Convolution

If x_n and y_n are periodic with period N, their **circular convolution** is defined by

$$(x \circledast y)_n := \sum_{m=0}^{N-1} x_m y_{n-m}.$$

We first show that this formula is equal to

$$\sum_{k=0}^{N-1} x_{n-k} y_k.$$

In the defining sum, make the change of variable $k = n - m$. Then

$$\begin{aligned}(x \circledast y)_n &= \sum_{k=n}^{n-(N-1)} x_{n-k} y_k. \\ &= \sum_{k=n-(N-1)}^{n} x_{n-k} y_k \\ &= \sum_{k=0}^{N-1} x_{n-k} y_k.\end{aligned}$$

Now, the second line follows because the order in which we add up the N terms does not matter. To understand the last line, note that as a function of k, the product $x_{n-k}y_k$ is periodic with period N. If we add up N consecutive products, it does not matter where we start the sum since we will sum over one period.

We next show that the DFT of the circular convolution is the product of the individual DFTs. Write

$$\begin{aligned}\sum_{n=0}^{N-1} (x \circledast y)_n e^{-j2\pi kn/N} &= \sum_{n=0}^{N-1} \left[\sum_{m=0}^{N-1} x_m y_{n-m} \right] e^{-j2\pi kn/N} \\ &= \sum_{m=0}^{N-1} x_m \left[\sum_{n=0}^{N-1} y_{n-m} e^{-j2\pi kn/N} \right] \\ &= \sum_{m=0}^{N-1} x_m \left[\sum_{l=-m}^{-m+N-1} y_l e^{-j2\pi k(l+m)/N} \right] \\ &= \sum_{m=0}^{N-1} x_m e^{-j2\pi km/N} \left[\sum_{l=-m}^{-m+N-1} y_l e^{-j2\pi kl/N} \right] \\ &= \sum_{m=0}^{N-1} x_m e^{-j2\pi km/N} \left[\sum_{l=0}^{N-1} y_l e^{-j2\pi kl/N} \right] \\ &= X_k Y_k.\end{aligned}$$

3.2.1. The Operation Count

To evaluate the formula for $(x \circledast y)_n$ for one value of n requires N multiplications along with $N-1$ additions. To do this for N times; i.e., for $n = 0, \ldots, N-1$, requires $N(2N-1)$ operations. In other words, it requires $O(N^2)$ operations.

As an alternative to this approach, suppose we use the FFT to compute the DFTs X_k and Y_k, compute the N products $X_k Y_k$, and then use the **inverse FFT** (IFFT) to compute the IDFT to obtain $(x \circledast y)_n$ for $n = 0, \ldots, N-1$. Since each FFT requires $O(N \log N)$ operations, the total operation count for this alternative is still $O(N \log N)$.

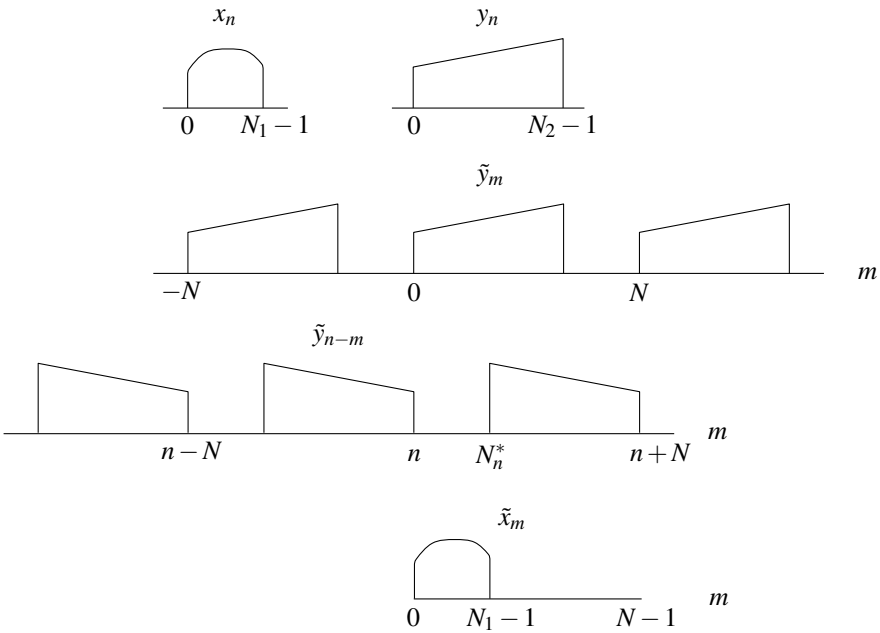


Figure 3.2. Computation of linear convolution via zero padding and circular convolution. Note that $N_n^* = n+N-(N_2-1) = n+N_1+N_2-1-N_2+1 = n+N_1 \geq N_1$ for $n \geq 0$.

3.3. Fast (Ordinary) Convolution

Recall that the ordinary convolution of two sequences x_n and y_n is defined by

$$(x*y)_n := \sum_{m=-\infty}^{\infty} x_m y_{n-m}.$$

If both sequences are causal, then

$$(x*y)_n = \sum_{m=0}^{n} x_m y_{n-m}, \quad n \geq 0, \tag{3.6}$$

and $(x*y)_n = 0$ for $n < 0$. We now further assume that x_n and y_n have finite durations N_1 and N_2, respectively. More specifically, assume x_n and y_n are causal and satisfy $x_n = 0$ for $n \geq N_1$ and $y_n = 0$ for $n \geq N_2$ as shown at the top in Figure 3.2. Sketching x_m and y_{n-m} as functions of m, it is easy to see that $(x*y)_n = 0$ for $n-(N_2-1) \geq N_1$, or equivalently, for $n \geq N_1+N_2-1$. Thus, $x*y$ has duration $N := N_1+N_2-1$.

Let $\tilde{x}_n$ and $\tilde{y}_n$ denote the N-periodic extensions of $x_0,\dots,x_{N-1}$ and $y_0,\dots,y_{N-1}$, respectively. For example, $\tilde{y}_m$ is shown in Figure 3.2. Then, for $n = 0,\dots,N-1$,

$$\begin{aligned}(x \circledast y)_n &= \sum_{m=0}^{N-1} \tilde{x}_m \tilde{y}_{n-m} \\ &= \sum_{m=0}^{n} \tilde{x}_m \tilde{y}_{n-m} + \sum_{m=n+1}^{N-1} \tilde{x}_m \tilde{y}_{n-m} \\ &= \sum_{m=0}^{n} x_m y_{n-m} + \sum_{m=n+1}^{N-1} x_m \tilde{y}_{n-m}.\end{aligned}$$

Observe that this last sum is zero if $n+1 \geq N_1$. On the other hand, if $n+1 \leq N_1 - 1$, the last sum is

$$\sum_{m=n+1}^{N_1-1} x_m \tilde{y}_{n-m+N}.$$

Now, in this sum, $m \leq N_1 - 1$ or $-m \geq -N_1 + 1$. Then

$$\begin{aligned}n - m + N \geq n - N_1 + 1 + N &= n - N_1 + 1 + N_1 + N_2 - 1 \\ &= n + N_2 \\ &\geq N_2.\end{aligned}$$

Also, $m > n$ implies $n - m < 0$, which implies $n - m + N < N$. So,

$$\tilde{y}_{n-m+N} = y_{n-m+N} = 0.$$

Hence, $(\tilde{x} \circledast \tilde{y})_n = (x * y)_n$ for $0 \leq n \leq N-1$.

3.3.1. The Operation Count

For causal signals of finite-duration, to compute $(x * y)_n$ using (3.6) requires n multiplications and $n-1$ additions, and we must do this for $n = 0,\dots,N-1$. Using the formula

$$\sum_{n=1}^{N} n = \frac{N(N+1)}{2},$$

it is easy to see that evaluating (3.6) for $n = 0,\dots,N-1$ requires $N(N+1)/2$ multiplications along with $(N-1)N/2$ additions. Hence, the number of operations is $O(N^2)$. However, using the FFT method on the periodic extensions $\tilde{x}_n$ and $\tilde{y}_n$ requires only $O(N \log N)$ operations.

3.3.2. How Does Circular Convolution with FFTs compare with `conv`?

From our theoretical discussion, there is no question that the circular convolution of zero-padded sequences is faster than the direct convolution in Section 2.1. To see the difference in action, run the following MATLAB script, which computes the linear convolution of two causal, finite-duration sequences by both methods and reports the time taken by each.

```
x=ones(1,2000);
y=ones(1,5000);
tic
N = length(x)+length(y)-1;
N = 2^ceil(log2(N));          % round up to a power of 2
v = ifft(fft(x,N).*fft(y,N));
ffttime = toc;
tic
w = conv(x,y);
convtime = toc;
plot([0:length(v)-1],real(v)); grid on
fprintf('conv takes %g times longer.\n',convtime/ffttime)
```

3.4. Conclusion

In Section 3.1.3 we corralled the beast of order N^2 operations for computing the DTFT, and in Section 3.3 we did the same for computing the convolution of two finite-length sequences.

Problems

3.1. Show that the DFT Y_k in (3.2) is a periodic function of k with period N.

3.2. Show that the DTFT of

$$w_n := \begin{cases} e^{j2\pi n f_0}, & 0 \le n < N, \\ 0, & \text{otherwise}, \end{cases}$$

is $e^{-j\pi(f-f_0)(N-1)} \sin\big(\pi N(f-f_0)\big) / \sin\big(\pi(f-f_0)\big)$.

3.3. Show that

$$\sum_{n=-M}^{M} e^{j\pi\theta n} = N \frac{\operatorname{sinc}(N\theta/2)}{\operatorname{sinc}(\theta/2)}, \quad \text{where } N := 2M+1.$$

3.4. **MATLAB.** Write a MATLAB function `convfft(x,y)` that computes the linear convolution by computing the inverse FFT of the product of the FFTs of

zero-padded versions x and y. Using your function plot the convolution of

$$\mathtt{x} = \mathtt{ones}(1,100) \quad \text{and} \quad \mathtt{y} = \mathtt{ones}(1,50)$$

CHAPTER 4
Window Techniques

4.1. The Basics of Windows

Consider a discrete-time signal x_n. Even if it is of finite duration, the duration may be so long that it is not feasible to compute its DTFT. Also, its frequency content may change over time. For example, if x_n is obtained by sampling a piece of music, the pitch and tempo may vary. For this reason, we are interested in the DTFT of the windowed signal

$$y_n = \begin{cases} x_n, & n = 0, \ldots, N-1, \\ 0, & \text{otherwise.} \end{cases}$$

More generally, we consider $y_n = w_n x_n$, where $w_n = 0$ for $n < 0$ and $n \geq N$. The key observation here is that since multiplication in the time domain corresponds to **periodic convolution** in the frequency domain, when we compute

$$\sum_{n=0}^{N-1} w_n x_n e^{-j2\pi fn}, \tag{4.1}$$

we do *not* get $X(f)$, we get

$$\int_{-1/2}^{1/2} W(\nu) X(f-\nu)\, d\nu. \tag{4.2}$$

If we put $f = k/N$, then (4.1) reduces to the DFT, which can be evaluated efficiently with any FFT algorithm, e.g., the MATLAB command `fft`. Of course we would like (4.2) to equal $X(f)$, but this happens if and only if $W(\nu) = \delta(\nu)$ on $[-1/2, 1/2]$, and this requires $w_n = 1$ for all n, which violates the window requirement $w_n = 0$ for $n < 0$ and $n \geq N$. So we want to use a window sequence such that $W(\nu)$ looks as much like $\delta(\nu)$ as possible; i.e., a narrow pulse. Let's consider the following well-known windows.

4.1.1. The Rectangular Window

The **rectangular window** uses $w_n = 1$ for $0 \leq n < N$. As shown in the problems, the DTFT of the rectangular window is

$$W_{\text{rect}}(f) = \frac{\sin(\pi N f)}{\sin \pi f} e^{-j\pi f(N-1)}. \tag{4.3}$$

[Sketch $\sin(\pi Nf)$ ***and*** $\sin \pi f$***.]*** It is easily verified that the maximum magnitude of the DTFT occurs at $f = 0$ and the zeros occur at integer multiples of $1/N$. The **main lobe** is the part of the magnitude curve between the zeros on each side of the origin. Hence, the main-lobe **width** is $2/N$. The parts of the curve between other adjacent zeros are called **side lobes**. The maxima of the side lobes occur *approximately* halfway between the zeros; i.e., at odd multiples of $1/(2N)$. ***[Draw picture; mark zeros, extreme points.]*** A key characteristic of a window is the **side-lobe level**, which is the ratio of the maximum side lobe to the maximum of the main lobe. This is usually expressed in terms of **decibels** (dB). For the rectangular window it is

$$20\log_{10}\frac{|W_{\text{rect}}(3/(2N))|}{|W_{\text{rect}}(0)|} = 20\log_{10}|W_{\text{rect}}(3/(2N))| - 20\log_{10}|W_{\text{rect}}(0)|,$$

which is easily found by subtracting levels when the frequency response is plotted on a dB scale. To evaluate $W_{\text{rect}}(0)$, observe that

$$\frac{\sin(\pi Nf)}{\sin \pi f} = \frac{\text{sinc}(Nf)}{\text{sinc}(f)} \cdot \frac{\pi Nf}{\pi f}.$$

Thus, $W_{\text{rect}}(0) = N$. Next, it is easy to see that $|W_{\text{rect}}(3/(2N))| = 1/|\sin(3\pi/(2N))|$. For large N the argument of the sine is small, and we can use the approximation $\sin x \approx x$. This leads to $|W_{\text{rect}}(3/(2N))| \approx 2N/(3\pi)$, and

$$20\log_{10}\frac{2N/(3\pi)}{N} = 20\log_{10}\frac{2}{3\pi} \approx -13.46 \text{ dB}.$$

4.1.2. The Bartlett Window

For odd N, the **Bartlett window** is defined to be the convolution of the rectangular window of length $(N+1)/2$ with itself and then multiplied by $2/(N+1)$. A simple graphical convolution argument shows that the resulting triangular window ***[sketch]*** is given by

$$w_n = 1 - \frac{|2n-N+1|}{N+1}, \quad 0 \le n < N \text{ (odd)}.$$

Since convolution in the time domain corresponds to multiplication in the frequency domain, the frequency response of the Bartlett window is

$$W_{\text{Bartlett}}(f) = \frac{2}{N+1}e^{-j\pi f(N-1)}\left(\frac{\sin(\pi[N+1]f/2)}{\sin \pi f}\right)^2.$$

[Sketch numerator. Mark zero crossings, approx extreme pts.] The zero crossings occur at multiples of $2/(N+1)$, and the side-lobe extrema occur *approximately* halfway between the adjacent zeros; i.e., at odd multiples of $1/(N+1)$.

Hence, the main-lobe width is $4/(N+1)$, and for large N the side-lobe level is approximately

$$20\log_{10}\frac{|W_{\text{Bartlett}}(3/(N+1))|}{|W_{\text{Bartlett}}(0)|} = 20\log_{10}\frac{\frac{2}{N+1}\left(\frac{1}{\sin(3\pi/(N+1))}\right)^2}{\frac{2}{N+1}\left(\frac{N+1}{2}\right)^2}$$
$$\approx 20\log_{10}(2/3\pi)^2 \approx -26.93 \text{ dB}.$$

We see that in moving from the rectangular window to the Bartlett window, the main-lobe width has nearly doubled from $2/N$ to $4/(N+1)$, and the side-lobe level has also doubled (by squaring on a log scale).

4.1.3. The Hann (Hanning) Window

The **Hann window** (sometimes called the **Hanning window**)[1] is the raised cosine window ***[sketch]*** given by

$$w_n = 0.5 - 0.5\cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \le n < N.$$

A simple calculation shows that

$$W_{\text{Hann}}(f) = 0.5\left[W_{\text{rect}}(f) - 0.5W_{\text{rect}}\left(f - \frac{1}{N-1}\right) - 0.5W_{\text{rect}}\left(f + \frac{1}{N-1}\right)\right].$$

The Hann window is designed in the frequency domain with the idea being to partially cancel the side lobes by subtracting suitable terms.

The Modified Hann Window

Since the Hann window has the property that $w_0 = w_{N-1} = 0$, the first and last values of $w_n x_n$ are multiplied by zero and deleted. For this reason, the modified Hann window

$$w_n = 0.5 - 0.5\cos\left(\frac{2\pi(n+1)}{N+1}\right), \quad 0 \le n < N.$$

is sometimes used instead. In this formula, note that $w_{-1} = W_N = 0$.

[1] The Hann window is named after J. von Hann. According to [5, p. 468], the term "Hanning" or "hanning" was introduced by Blackman and Tukey [1].

4.1.4. The Hamming Window

The **Hamming window** is given by[2]

$$w_n = 0.54 - 0.46\cos\Big(\frac{2\pi n}{N-1}\Big), \quad 0 \le n < N$$

and is similar in appearance to the Hann window, except the first and last values are not zero. The DTFT of the Hamming window is

$$W_{\text{Hamming}}(f) = 0.54W_{\text{rect}}(f) - 0.23\Big[W_{\text{rect}}\Big(f - \frac{1}{N-1}\Big) + W_{\text{rect}}\Big(f + \frac{1}{N-1}\Big)\Big].$$

4.1.5. The Blackman Window

The **Blackman window** is given by[3][4]

$$w_n = 0.42 - 0.5\cos\Big(\frac{2\pi n}{N-1}\Big) + 0.08\cos\Big(\frac{4\pi n}{N-1}\Big), \quad 0 \le n < N.$$

Its DTFT is

$$\begin{aligned} W_{\text{Blackman}}(f) = {} & 0.42W_{\text{rect}}(f) - 0.25\Big[W_{\text{rect}}\Big(f - \frac{1}{N-1}\Big) + W_{\text{rect}}\Big(f + \frac{1}{N-1}\Big)\Big] \\ & + 0.04\Big[W_{\text{rect}}\Big(f - \frac{2}{N-1}\Big) + W_{\text{rect}}\Big(f + \frac{2}{N-1}\Big)\Big]. \end{aligned}$$

4.2. More Advanced Analysis of Windows

The DTFTs of the five preceding windows are shown in Figure 4.1. To analyze the last three, when N is large, we use the approximation $1/(N-1) \approx 1/N$ in the DTFT formulas. This makes the zero crossings of the different terms fall on multiples of $1/N$. ***[Draw pics of shifted*** W_{rect}***.]***

For example, in the Hann and Hamming windows, all three terms will be (approximately) zero for $f = k/N$ except for $k = 0$ and $k = \pm 1$. In the Blackman window, we also exclude $k = \pm 2$. It follows immediately that the main-lobe widths of the Hann and Hamming windows are both $4/N$. The main-lobe width of the Blackman window is $6/N$.

[2] Blackman and Tukey [1, pp. 98–99] point out that 0.54 is a good approximation to $25/46$, and 0.46 is good approximation to $42/92$.

[3] Blackman and Tukey [1, pp. 98–99] use the approximations $3969/9304 \approx 0.42$ (even though 0.43 is closer), $2310/4652 \approx 0.5$, and $1430/18608 \approx 0.08$.

[4] The Blackman window, like the Hann window, satisfies $w_0 = w_{N-1} = 0$ and can be similarly modified.

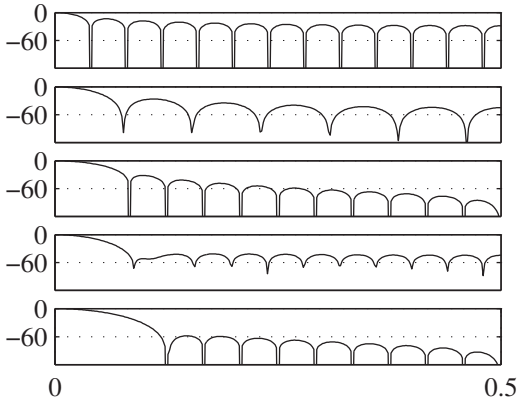


Figure 4.1. DTFTs (in dB) of windows with $N = 25$ (top to bottom): rectangular, Bartlett, Hann, Hamming, and Blackman.

To evaluate the side-lobe level of the Hann window, we begin by evaluating the DTFT halfway between the first two zeros. We have

$$\begin{aligned}
&W_{\text{Hann}}(2.5/N) \\
&\quad = 0.5\left[W_{\text{rect}}\left(\frac{2.5}{N}\right) - 0.5W_{\text{rect}}\left(\frac{2.5}{N} - \frac{1}{N-1}\right) - 0.5W_{\text{rect}}\left(\frac{2.5}{N} + \frac{1}{N-1}\right)\right] \\
&\quad \approx 0.5\left[W_{\text{rect}}\left(\frac{2.5}{N}\right) - 0.5W_{\text{rect}}\left(\frac{2.5}{N} - \frac{1}{N}\right) - 0.5W_{\text{rect}}\left(\frac{2.5}{N} + \frac{1}{N}\right)\right] \\
&\quad = 0.5\left[W_{\text{rect}}\left(\frac{2.5}{N}\right) - 0.5W_{\text{rect}}\left(\frac{1.5}{N}\right) - 0.5W_{\text{rect}}\left(\frac{3.5}{N}\right)\right].
\end{aligned}$$

Now in (4.3), we also make the approximation $N-1 \approx N$ in the exponential factor and use $\sin x \approx x$ as before. We find that

$$W_{\text{Hann}}(2.5/N) \approx -(jN/\pi)[1/5 - 0.5/3 - 0.5/7] = (jN/\pi)(4/105).$$

Similarly,

$$W_{\text{Hann}}(0) \approx 0.5[W_{\text{rect}}(0) - 0.5W_{\text{rect}}(-1/N) - 0.5W_{\text{rect}}(1/N)] = 0.5N.$$

It follows that the Hann side-lobe level is $20\log_{10}(8/(105\pi)) = -32.3$ dB.

In the case of the Hamming window, the side lobe between $3/N$ and $4/N$ is the biggest, and so we evaluate W_{Hamming} at $3.5/N$. Write

$$W_{\text{Hamming}}(3.5/N) \approx -(j2N/\pi)[0.54/7 - 0.23/5 - 0.23/9].$$

Since $W_{\text{Hamming}}(0) \approx 0.54N$, we find that the Hamming side-lobe level is -43.6 dB.

In the case of the Blackman window, we find empirically maximum of the first sidelobe occurs at about $3.21/N$. We have

$$\begin{aligned} W_{\text{Blackman}}(3.5/N) &\approx 0.42W_{\text{rect}}(3.5/N) - 0.25\Big[W_{\text{rect}}(2.5/N) + W_{\text{rect}}(4.5/N)\Big] \\ &\quad + 0.04\Big[W_{\text{rect}}(1.5/N) + W_{\text{rect}}(5.5/N)\Big] \\ &\approx -j(2N/\pi)[0.42/7 - 0.25/5 - 0.25/9 + 0.04/3 + 0.04/11] \end{aligned}$$

and

$$\begin{aligned} W_{\text{Blackman}}(0) &\approx 0.42W_{\text{rect}}(0) - 0.25\Big[W_{\text{rect}}(-1/N) + W_{\text{rect}}(1/N)\Big] \\ &\quad + 0.04\Big[W_{\text{rect}}(-2/N) + W_{\text{rect}}(2/N)\Big] \\ &= 0.42N. \end{aligned}$$

We find that the Blackman side-lobe level is -58.2 dB.

Name	Main-Lobe Width	Side-Lobe Level (dB)
Rectangular	$2/N$	-13.46
Bartlett (N odd)	$4/(N+1)$	-26.93
Hann	$4/N$	-32.3
Hamming	$4/N$	-43.6
Blackman	$6/N$	-58.2

Table 4.1. Some common window widths and side-lobe levels. The parameter N is the window length.

4.3. The Kaiser Window

We conclude our discussion of windows by introducing the **Kaiser window**. The Kaiser window is defined by

$$w_n = \frac{I_0\left(\beta\sqrt{1-\left(\frac{|2n-N+1|}{N-1}\right)^2}\right)}{I_0(\beta)}, \quad 0 \le n < N,$$

where β is a shape parameter and I_0 denotes the modified **Bessel function** of the first kind order zero,

$$I_0(t) := \sum_{n=0}^{\infty}\left(\frac{(t/2)^n}{n!}\right)^2.$$

Since the terms are squared, we see that $I_0(t)$ is an even function. To compute $I_0(t)$ in MATLAB, use the command `besseli(0,t)`. The following script computes the DTFT of the Kaiser window $W_{\text{Kaiser}}(f)$.

```
function W = WKaiser(f,N,beta)
% Compute Kaiser window seq. wn.
n = [0:N-1];
arg = abs(2*n-N+1)/(N-1);
argi = beta*sqrt(1-arg.^2);
wn = besseli(0,argi)/besseli(0,beta);
W = wn*exp(-j*2*pi*n'*f); % Compute DTFT of wn
```

[Run wXscript to illustrate windowing and spectral analysis.]

Problems

4.1. Write down the formula for the length $(N+1)/2$ **rectangular window**. Use this result to derive the formula given for DTFT of the length N **Bartlett window**, $W_{\text{Bartlett}}(f)$.

4.2. Use Euler's formula $\cos\theta = [e^{j\theta} + e^{-j\theta}]/2$ to compute the DTFT of the **Hann window** to verify the formula given for $W_{\text{Hann}}(f)$.

4.3. Use the method of the preceding problem to compute the DTFT of the **Blackman window** to verify the formula given for $W_{\text{Blackman}}(f)$.

4.4. Let v be the rectangular window of length M, and put $w_n := (v * v * v)_n$. Let N denote the length of w.
   (a) Express M as a function of N.
   (b) Find the main-lobe width.
   (c) Find the (approximate) side-lobe level in dB.

4.5. Let u_n be the Hann window of length M, and let v_n be the Hann window of length $M+1$. Let w_n denote the convolution of u_n and v_n. Assuming M is large, find the approximate side-lobe level of the window w_n.

CHAPTER 5
The *z* Transform

Because linear time-invariant systems are characterized by convolution, and because of the convolution property of the z transform, the z transform is a powerful tool for the analysis and design of such systems.

5.1. Basic Definitions

Given a sequence $\{x_n\}_{n=-\infty}^{\infty}$, its **bilateral** or **two-sided z transform** is defined as

$$X(z) := \sum_{n=-\infty}^{\infty} x_n z^{-n},$$

for all complex z such that the series converges absolutely; i.e., all

$$z \in \mathrm{ROC}(x) := \left\{ z : \sum_{n=-\infty}^{\infty} \left| x_n z^{-n} \right| < \infty \right\}.$$

Here ROC stands for **region of convergence**.

***Example* 5.1.1.** If

$$x_n = \begin{cases} a^n, & n \geq 0, \\ 0, & n < 0, \end{cases}$$

then

$$X(z) = \sum_{n=0}^{\infty} a^n z^{-n} = \sum_{n=0}^{\infty} (az^{-1})^n = \frac{1}{1-az^{-1}},$$

for $|az^{-1}| < 1$ or $|z| > |a|$. ***[DRAW ROC]***.

***Example* 5.1.2.** If

$$x_n = \begin{cases} 0, & n > 0, \\ a^n, & n \leq 0, \end{cases}$$

then

$$X(z) = \sum_{n=-\infty}^{0} a^n z^{-n} = \sum_{m=0}^{\infty} a^{-m} z^m = \sum_{m=0}^{\infty} (a^{-1}z)^m = \frac{1}{1-a^{-1}z},$$

for $|a^{-1}z| < 1$ or $|z| < |a|$. ***[DRAW ROC]***.

***Example* 5.1.3.** For $|a| < 1$, if $x_n = a^{|n|}$, then

$$X(z) = \sum_{n=-\infty}^{\infty} a^{|n|} z^{-n} = \sum_{n=0}^{\infty} (az^{-1})^n + \sum_{n=-\infty}^{-1} a^{-n} z^{-n}.$$

If $|z| > |a|$, then the first sum on the right converges to $1/(1-az^{-1})$. The second term on the right is equal to

$$\sum_{m=-1}^{\infty} (a^{-1}z^{-1})^m = \sum_{m=1}^{\infty} (az)^n = -1 + \sum_{m=0}^{\infty} (az)^n,$$

which converges to

$$\frac{1}{1-az} - 1 = \frac{1-(1-az)}{1-az} = \frac{az}{1-az},$$

if $|az| < 1$ or $|z| < 1/|a|$. Hence,

$$X(z) = \frac{1}{1-az^{-1}} + \frac{az}{1-az}, \quad |a| < |z| < 1/|a|.$$

[DRAW ROC].

This last example illustrates the general case. If

$$R_1 := \inf\left\{ r \geq 0 : \sum_{n=-\infty}^{\infty} |x_n| r^{-n} < \infty \right\} \quad \text{and} \quad R_2 := \sup\left\{ r \geq 0 : \sum_{n=-\infty}^{\infty} |x_n| r^{-n} < \infty \right\},$$

then

$$\{z : R_1 < |z| < R_2\} \subseteq \mathrm{ROC}(x).$$

An important observation is that the ROC contains all circles of the form $\{z : |z| = r\}$ for $R_1 < r < R_2$. The ROC may or may not contain some z with $|z| = R_1$ or $|z| = R_2$. In Example 5.1.2, $R_1 = 0$, $R_2 = |a|$, and $z = 0$ is in the ROC. In Example 5.1.1, $R_1 = |a|$, $R_2 = \infty$, and $\lim_{z\to\infty} X(z) = x_0$; i.e., $z = \infty$ is in the ROC.

5.1.1. Importance of the ROC

Let $x_n^+ := a^n$ for $n \geq 0$ and $x_n^+ = 0$ for $n < 0$. Let $x_n^- := -a^n$ for $n < 0$ and $x_n^- = 0$ for $n \geq 0$. Then by Example 5.1.1, the z transform of x_n^+ is $1/(1-az^{-1})$. By the calculations in Example 5.1.3 (or Problem 5.1), the z transform of x_n^- is also $1/(1-az^{-1})$. What's going on?

The "catch" is that we have ignored the region of convergence. In the case of x_n^+, the ROC is $|z| > |a|$, while in the case of x_n^-, the ROC is $|z| < |a|$. The ROC is part of the z transform of a signal. It is not enough just to give the formula; you must also say for what values of z the formula holds — you must also give the ROC.

To say that two z transforms are the same means that they have the same ROC and that their formulas are equal to each other for all z in the ROC. So, two transforms are different if they have different ROCs, even if their formulas are the same. Two transforms are different if their ROCs are the same but their formulas are not equal for some z in their common ROC.

5.1.2. The Inverse *z* Transform

As mentioned above, the ROC always contains circles of the form $\{z : |z| = r\}$ for $R_1 < r < R_2$. For such r, consider evaluating the z transform at points z of the form $z = re^{j2\pi f}$, where $|f| \le 1/2$. We get

$$X(z)\big|_{z=re^{j2\pi f}} = \sum_{n=-\infty}^{\infty} x_n \left(re^{j2\pi f}\right)^{-n} = \sum_{n=-\infty}^{\infty} x_n r^{-n} e^{-j2\pi fn}.$$

We recognize this as the DTFT of the sequence $x_n r^{-n}$. It follows by the inverse DTFT that

$$x_n r^{-n} = \int_{-1/2}^{1/2} X\left(re^{j2\pi f}\right) e^{j2\pi fn}\, df,$$

or

$$x_n = r^n \int_{-1/2}^{1/2} X\left(re^{j2\pi f}\right) e^{j2\pi fn}\, df. \tag{5.1}$$

This implies that if two sequences have z transforms that are equal on a common circle of radius $0 < r < \infty$, then the two sequences must be the same. For example, if u_n and v_n have transforms $U(z)$ and $V(z)$, put $x_n := u_n - v_n$. It suffices to show that $x_n = 0$. Now, since $x_n = u_n - v_n$, by linearity of the z transform, $X(z) = U(z) - V(z)$. If $U(z) = V(z)$ on a circle of radius r, then $X(z) = 0$ on that circle, and then (5.1) implies $x_n = 0$.

5.2. Properties

Linearity. The z transform is linear. This means that if $y_n = ax_n + bw_n$, where a and b are real or complex numbers, then $Y(z) = aX(z) + bW(z)$. ***[Derive.]***

Delay/Advance. For fixed m, if $y_n = x_{n-m}$, then $Y(z) = z^{-m}X(z)$. ***[Derive.]***

Convolution. If $y_n = (w * x)_n$, then $Y(z) = W(z)X(z)$. ***[Derive.]***

5.3. DTFTs from *z* Transforms

If x_n is a sequence whose z transform $X(z)$ has a ROC that contains the unit circle, then we can evaluate $X(z)$ for $z = e^{j2\pi f}$ to get

$$X(e^{j2\pi f}) = \sum_{n=-\infty}^{\infty} x_n \left(e^{j2\pi f}\right)^{-n} = \sum_{n=-\infty}^{\infty} x_n e^{-j2\pi fn},$$

which is the DTFT of x_n.

***Example* 5.3.1.** Plot the absolute value of the DTFT of $x_n = (1/2)^n$ for $n \geq 0$.

Solution. From Example 5.1.1, we know that the z transform of this signal is $X(z) = 1/(1-(1/2)z^{-1}) = 2/(2-z^{-1})$. The MATLAB code

```
f = linspace(-1/2,1/2,200);
zinv = exp(-j*2*pi*f);
plot(f,abs(2./(2-zinv)))
```

generates the desired plot.

5.4. Transform Inversion by Partial Fractions

Suppose that

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_q z^{-q}}{1 + a_1 z^{-1} + \cdots + a_p z^{-p}}, \quad q < p.$$

We first point out that the denominator has exactly p roots. Consider the polynomial

$$\Xi(z) := 1 + a_1 z + \cdots + a_p z^p.$$

Since $\Xi(1/z) = A(z)$, we see that $A(z) = 0$ if and only if $\Xi(1/z) = 0$. Since Ξ is a polynomial of degree p, it has exactly p roots, say $\xi_1, \ldots, \xi_p$. Furthermore, since $\Xi(0) = 1$, none of the ξ_k is zero. Hence, the p roots of $A(z)$ are $\alpha_1 := 1/\xi_1, \ldots, \alpha_p := 1/\xi_p$.

We now show that if the denominator roots α_k are distinct, then we can write[1]

$$H(z) = \sum_{k=1}^{p} \frac{C_k}{1 - \alpha_k z^{-1}} \tag{5.2}$$

[1] In order for (5.2) to hold, it is necessary that $q < p$. To see why this is so, observe that if we put the right-hand side of (5.2) over a common denominator, the resulting numerator polynomial in z^{-1} has degree at most $p-1$.

for some constants C_k. Suppose that such a formula for $H(z)$ exists. Write it as

$$\frac{B(z)}{\prod_{k=1}^{p}(1-\alpha_k z^{-1})} = \sum_{k=1}^{p} \frac{C_k}{1-\alpha_k z^{-1}}.$$

Multiply both sides by $1-\alpha_i z^{-1}$ to get

$$\frac{B(z)}{\prod_{k\neq i}^{p}(1-\alpha_k z^{-1})} = C_i + \sum_{k\neq i} \frac{C_k(1-\alpha_i z^{-1})}{1-\alpha_k z^{-1}}.$$

In this expression, now set $z=\alpha_i$ to get

$$\frac{B(\alpha_i)}{\prod_{k\neq i}^{p}(1-\alpha_k/\alpha_i)} = C_i. \tag{5.3}$$

***Example* 5.4.1.** Find the partial fraction expansion of

$$H(z) = \frac{1}{1-7z^{-1}+12z^{-2}}.$$

Solution. Writing

$$1-7z^{-1}+12z^{-2} = z^{-2}(z^2-7z+12) = z^{-2}(z-3)(z-4),$$

we see that the **poles** (roots of the denominator) are $z=3$ and $z=4$. Write

$$\frac{1}{(1-3z^{-1})(1-4z^{-1})} = \frac{?}{1-3z^{-1}} + \frac{?}{1-4z^{-1}}.$$

The first missing numerator is

$$\left.\frac{1}{1-4z^{-1}}\right|_{z=3} = \frac{1}{1-4/3} = -3$$

and the second missing numerator is

$$\left.\frac{1}{1-3z^{-1}}\right|_{z=4} = \frac{1}{1-3/4} = 4.$$

Hence,

$$\frac{1}{(1-3z^{-1})(1-4z^{-1})} = \frac{-3}{1-3z^{-1}} + \frac{4}{1-4z^{-1}}.$$

In many cases, the degrees of $A(z)$ and $B(z)$ are the same. In this case, observe that

$$\begin{aligned}\frac{B(z)}{A(z)} &= \frac{\left[B(z)-\frac{b_p}{a_p}A(z)\right]+\frac{b_p}{a_p}A(z)}{A(z)} \\ &= \frac{\left[\{b_pz^{-p}+\cdots\}-\frac{b_p}{a_p}\{a_pz^{-p}+\cdots\}\right]+\frac{b_p}{a_p}A(z)}{A(z)} \\ &= \frac{\widetilde{B}(z)}{A(z)}+\frac{b_p}{a_p}, \end{aligned} \tag{5.4}$$

where $\widetilde{B}(z)=B(z)-\frac{b_p}{a_p}A(z)$ is a polynomial of degree $q<p$. One can then find the partial fraction expansion of $\widetilde{B}(z)/A(z)$. It is a useful savings of work to note that in applying (5.3) to $\widetilde{B}(z)/A(z)$, we do not need $\widetilde{B}(z)$ for all z, but only for $z=\alpha_i$. Observe that from (5.4),

$$\begin{aligned}\widetilde{B}(\alpha_i) &= B(\alpha_i)-\frac{b_p}{a_p}A(\alpha_i) \\ &= B(\alpha_i),\end{aligned}$$

since $A(\alpha_i)=0$.

***Example* 5.4.2.** Find the partial fraction expansion of

$$\frac{2z^{-1}-z^{-2}}{1-5z^{-1}+6z^{-2}}.$$

***Solution*.** We first note that $b_2/a_2=-1/6$. Second, writing $1-5z^{-1}+6z^{-2}=z^{-2}(z^2-5z+6)$, we see that the poles are $\alpha_1=2$ and $\alpha_2=3$. Thus,

$$\begin{aligned}\frac{2z^{-1}-z^{-2}}{1-5z^{-1}+6z^{-2}} &= \frac{2z^{-1}-z^{-2}}{(1-2z^{-1})(1-3z^{-1})} \\ &= \frac{?}{1-2z^{-1}}+\frac{?}{1-3z^{-1}}-\frac{1}{6} \\ &= \frac{-3/2}{1-2z^{-1}}+\frac{5/3}{1-3z^{-1}}-\frac{1}{6}.\end{aligned}$$

Problems

5.1. If $x_n=-a^n$ for $n<0$ and $x_n=0$ for $n\geq 0$, show that its z transform is

$$X(z)=\frac{1}{1-az^{-1}}, \quad |z|<|a|.$$

5.2. Find the partial fraction expansion of

$$\frac{6z^{-1}}{1-6z^{-1}+11z^{-2}-6z^{-3}}.$$

Answer (numerators): $3, -12, 9$.

5.3. Find the partial fraction expansion of

$$\frac{2+z^{-1}}{1-12z^{-1}+35z^{-2}}.$$

Answer (numerators): $15/2, -11/2$.

5.4. Find the partial fraction expansion of

$$\frac{(17/6)z^{-1}-(1/6)}{1-5z^{-1}+6z^{-2}}.$$

Answer (numerators): $-5/2, 7/3$.

5.5. Find the partial fraction expansion of

$$\frac{4-23z^{-1}+70z^{-2}}{1-12z^{-1}+35z^{-2}}.$$

Answer (numerators): $-11/2, 15/2$.

5.6. Find the partial fraction expansion of

$$\frac{z^{-2}(1+z^{-1})}{1-6z^{-1}+11z^{-2}-6z^{-3}}.$$

Answer (numerators): $1, -3/2, 2/3$.

5.7. **MATLAB.** Write a MATLAB function that takes as input $\mathtt{a} = [1, a_1, \ldots, a_p]$ and $\mathtt{b} = [b_0, \ldots, b_q]$ for $q \le p$ and returns the coefficients C_i (and the corresponding poles α_i) in (5.3). Write a script to test your function on each of the preceding partial-fraction problems. For each problem, print out `a`, `b`, and the corresponding coefficients C_i and poles α_i. *Hints:* To find the poles, use the MATLAB function `roots`. To evaluate polynomials from knowledge of their coefficients, use the MATLAB function `polyval`. To this end, the MATLAB function `fliplr` will also be helpful. To evaluate products, the MATLAB function `prod` can be used. We also mention that the MATLAB functions `rat` and `rats` or the command `format rat` may be useful to express answers as fractions.

CHAPTER 6
Discrete-Time Systems

A discrete-time system can be viewed as an operator A that associates an input sequence $x = \{x_n\}_{n=-\infty}^{\infty}$ with an output sequence $Ax = \{(Ax)_n\}_{n=-\infty}^{\infty}$. In this chapter, we show that every system A that is both linear and time-invariant has the representation $Ax = h * x$ for some **impulse response** sequence h. On account of the convolution property of the z transform, it is not surprising that the z transform is a powerful tool in the design and analysis of linear, time-invariant discrete-time systems.

6.1. Linearity

The system A is said to be **linear** if for any two input sequences x and w and any two constants a and b, the output sequence $A(ax+bw)$ satisfies $A(ax+bw) = a(Ax)+b(Aw)$. In more detail, A satisfies

$$[A(ax+bw)]_n = a(Ax)_n + b(Aw)_n.$$

***Example* 6.1.1.** Consider the system that associates to an input sequence x the output sequence defined by the convolution of x with another sequence h; i.e., $Ax := h * x$. Since convolution is linear; i.e., $h*(ax+bw) = a(h*x)+b(h*w)$, we have immediately that A is linear.

***Example* 6.1.2.** Show that

$$(Ax)_n := \sum_{k=n-3}^{n} x_k$$

defines a linear system.

***Solution*.** Write

$$\begin{aligned}
[A(ax+bw)]_n &= \sum_{k=n-3}^{n} (ax+bw)_k \\
&= \sum_{k=n-3}^{n} ax_k + bw_k \\
&= a\sum_{k=n-3}^{n} x_k + b\sum_{k=n-3}^{n} w_k \\
&= a(Ax)_n + b(Aw)_n.
\end{aligned}$$

To say that this holds for all n is to say that $A(ax+bw) = a(Ax)+b(Aw)$.

Another way to do the preceding example is to observe that

$$\sum_{k=n-3}^{n} x_k = \sum_{m=0}^{3} x_{n-m} \quad (\text{let } m = n-k)$$

is the convolution of x with the sequence $h_m = 1$ for $m = 0,1,2,3$ and $h_m = 0$ otherwise. Since $Ax = h * x$, we already know A is linear.

6.2. Time Invariance

For the definition of **time invariance**, we need the following definition. For any sequence x and delay m, we define the delayed sequence $x^{(m)} := \{x_{n-m}\}_{n=-\infty}^{\infty}$. A negative delay can be called an advance. A system A is said to be **time invariant** if for every delay m, $(Ax^{(m)}) = (Ax)^{(m)}$, or in more detail $(Ax^{(m)})_n = (Ax)_{n-m}$ for all n. Time invariance means that the response to the delayed input is equal to the delay of the response to the original input.

***Example* 6.2.1.** Show that the system of Example 6.1.2 is time invariant.

Solution. We first compute

$$(Ax^{(m)})_n = \sum_{k=n-3}^{n} (x^{(m)})_k = \sum_{k=n-3}^{n} x_{k-m}.$$

Now make the change of variable $r = k-m$ to get

$$(Ax^{(m)})_n = \sum_{r=(n-3)-m}^{n-m} x_r = \sum_{r=(n-m)-3}^{n-m} x_r = (Ax)_{n-m}.$$

Thus, A is time invariant.

This example is a particular case of the general result that a system of the form $Ax = h * x$ is time invariant. To see this, first write

$$(h * x^{(m)})_n = \sum_{k=-\infty}^{\infty} h_{n-k}(x^{(m)})_k = \sum_{k=-\infty}^{\infty} h_{n-k}x_{k-m}.$$

Then make the change of variable $r = k-m$ to get

$$(h * x^{(m)})_n = \sum_{r=-\infty}^{\infty} h_{n-(r+m)}x_r = \sum_{r=-\infty}^{\infty} h_{(n-m)-r}x_r = (h*x)_{n-m}.$$

6.3. Characterization of Linear Time-Invariant Systems

We now show that every linear time-invariant system A is of the form $Ax = h * x$ for some **impulse response** sequence h.

To begin, define the discrete-time unit impulse sequence by $\delta_n = 1$ for $n = 0$ and $\delta_n = 0$ for $n \neq 0$. Define the system **impulse response** $h_n := (A\delta)_n$. Observe that

$$x_n = \sum_{k=-\infty}^{\infty} x_k \delta_{n-k} = \sum_{k=-\infty}^{\infty} x_k (\delta^{(k)})_n.$$

Since this holds for all n, we can write this more compactly as

$$x = \sum_{k=-\infty}^{\infty} x_k \delta^{(k)}.$$

By linearity of A,

$$Ax = \sum_{k=-\infty}^{\infty} A(x_k \delta^{(k)}) = \sum_{k=-\infty}^{\infty} x_k A(\delta^{(k)}).$$

By the time-invariance of A,

$$Ax = \sum_{k=-\infty}^{\infty} x_k (A\delta)^{(k)}.$$

This implies that for all n,

$$(Ax)_n = \sum_{k=-\infty}^{\infty} x_k [(A\delta)^{(k)}]_n = \sum_{k=-\infty}^{\infty} x_k (A\delta)_{n-k} = \sum_{k=-\infty}^{\infty} x_k h_{n-k} = (h * x)_n.$$

6.4. Stability

To define stability, we first need the notion of a **bounded sequence**. A sequence x is said to be **bounded** if there is a nonnegative, finite constant B such that $|x_n| \leq B$ for all n.

A system A is said to be **bounded-input bounded-output stable (BIBO stable)** if for every bounded input sequence x, the corresponding output sequence Ax is also bounded. If an unstable system is implemented on a computer, overflow could occur and the system would behave in an unpredictable manner. For this reason, we only want to design stable systems.

Our main result here is that a linear time-invariant system is stable if and only if its impulse response h satisfies

$$\sum_{k=-\infty}^{\infty} |h_k| < \infty.$$

To establish this result, we have to prove two things. First, we assume the sum is finite, and we show that this implies the system must be stable. Second, we assume the sum is not finite, and we show that the system is not stable; i.e., we construct a bounded input for which the output is not bounded.

Suppose $C := \sum_{k=-\infty}^{\infty} |h_k| < \infty$, and suppose that x is a bounded input with $|x_k| \leq B$. Then

$$|(Ax)_n| = \left| \sum_{k=-\infty}^{\infty} h_{n-k} x_k \right| \leq \sum_{k=-\infty}^{\infty} |h_{n-k} x_k| = \sum_{k=-\infty}^{\infty} |h_{n-k}|\, |x_k| \leq B \sum_{k=-\infty}^{\infty} |h_{n-k}|.$$

In this last sum, make the change of variable $r = n-k$ to get

$$|(Ax)_n| \leq B \sum_{r=-\infty}^{\infty} |h_r| = BC < \infty.$$

Now suppose that $\sum_{k=-\infty}^{\infty} |h_k| = \infty$. Consider the bounded input

$$x_k := \begin{cases} \dfrac{\overline{h_{-k}}}{|h_{-k}|}, & \text{if } h_{-k} \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Then

$$(Ax)_0 = \sum_{k=-\infty}^{\infty} h_{-k} x_k = \sum_{k:h_{-k}\neq 0} \frac{|h_{-k}|^2}{|h_{-k}|} = \sum_{k:h_{-k}\neq 0} |h_{-k}| = \sum_{k=-\infty}^{\infty} |h_{-k}| = \sum_{r=-\infty}^{\infty} |h_r| = \infty.$$

Thus, a bounded input yields infinite output at time $n = 0$.

6.5. Causality

A system A is said to be a **causal *system*** if for all input sequences x, and for all times n, $(Ax)_n$ depends on x only through $\{x_k\}_{k=-\infty}^{n}$. In other words, the system cannot see into the future. All real-time systems must be causal. We now show that a linear time-invariant system is causal if and only if its impulse response satisfies $h_n = 0$ for $n < 0$. To see that this is indeed the case, write

$$(Ax)_n = \sum_{k=-\infty}^{\infty} h_{n-k} x_k = \sum_{k=-\infty}^{n} h_{n-k} x_k + \sum_{k=n+1}^{\infty} h_{n-k} x_k. \tag{6.1}$$

If $h_m = 0$ for $m < 0$, then the last sum is zero, and we see that $(Ax)_n$ depends on x only through x_k for $k \leq n$. Conversely, suppose A is causal. Then in (6.1), no matter

how we change the values of x_k for $k > n$, the value of $(Ax)_n$ cannot change. For example, if $x_k = 0$ for $k > n$, then (6.1) tells us that

$$(Ax)_n = \sum_{k=-\infty}^{n} h_{n-k} x_k.$$

On the other hand if $x_k = \overline{h_{n-k}}$ for $k > n$, then (6.1) tells us that

$$(Ax)_n = \sum_{k=-\infty}^{n} h_{n-k} x_k + \sum_{k=n+1}^{\infty} |h_{n-k}|^2.$$

Comparing these two expressions for $(Ax)_n$ shows that

$$\sum_{k=n+1}^{\infty} |h_{n-k}|^2 = 0.$$

Making the change of variable $m = k - n$ shows that

$$\sum_{m=1}^{\infty} |h_{-m}|^2 = 0,$$

which implies $h_n = 0$ for $n < 0$.

For a linear, time-invariant system that is causal, we can write

$$(Ax)_n = \sum_{k=-\infty}^{n} h_{n-k} x_k.$$

In particular, if $n < 0$, this sum involves x_k only for $k < 0$. Hence, if $x_k = 0$ for $k < 0$, then the output $(Ax)_n$ must be zero for $n < 0$.

A sequence x with $x_k = 0$ for $k < 0$ is called a **causal *sequence***. What we have shown above is that the impulse response of a linear, time-invariant, causal *system* is a causal *sequence*.

6.6. Transfer Functions

The z transform of the impulse response of a linear time-invariant system is called the **transfer function**.

6.6.1. Stability

Consider a linear, time-invariant system with impulse response h_n and corresponding transfer function (z transform) $H(z)$. A point z belongs to the region of

convergence if and only if

$$\sum_{n=-\infty}^{\infty} |h_n z^{-n}| < \infty.$$

For points z on the unit circle; i.e., for $z = e^{j2\pi f}$,

$$\sum_{n=-\infty}^{\infty} |h_n z^{-n}| = \sum_{n=-\infty}^{\infty} |h_n e^{-j2\pi f}| = \sum_{n=-\infty}^{\infty} |h_n|.$$

Hence, points on the unit circle belong to the ROC if and only if the system is BIBO stable.

***Example* 6.6.1.** Consider a linear, time-invariant system with impulse response $h_n = 1/n$ for $n \geq 1$ and $h_n = 0$ for $n < 0$. Since $\sum_{n=1}^{\infty} 1/n = \infty$ (use an integral comparison test), this system is not stable.

In contrast, if $h_n = 1/n^2$ for $n \geq 1$, then since $\sum_{n=1}^{\infty} 1/n^2 < \infty$ (use an integral comparison test), the system would have been stable.

6.6.2. Causality

If a linear time-invariant system is causal, its transfer function must be of the form

$$H(z) = \sum_{n=0}^{\infty} h_n z^{-n},$$

and so its ROC cannot contain the origin $z = 0$ (unless $H(z) = h_0$). Now suppose that for some $z_0 \neq 0$,

$$\sum_{n=0}^{\infty} |h_n z_0^{-n}| < \infty.$$

Then for any z with $|z| > |z_0|$

$$\begin{aligned}
\sum_{n=0}^{\infty} |h_n z^{-n}| &= \sum_{n=0}^{\infty} \left| h_n \frac{z^{-n}}{z_0^{-n}} z_0^{-n} \right| \\
&= \sum_{n=0}^{\infty} |h_n z_0^{-n}| \left(\frac{|z_0|}{|z|} \right)^n \\
&\leq \sum_{n=0}^{\infty} |h_n z_0^{-n}| < \infty.
\end{aligned}$$

Thus, for a causal system, if $z_0 \in \mathrm{ROC}(h)$, then $\{z : |z| > |z_0|\} \subset \mathrm{ROC}(h)$. This means that the ROC must contain $\{z : |z| > R_1\}$ for some $R_1 \geq 0$. It may happen that $R_1 = \infty$; i.e., there is no z for which the series is summable.

***Example* 6.6.2.** Consider the sequence $h_n = e^{n^2}$ for $n \geq 0$. For real $r > 0$,

$$H(r) = \sum_{n=0}^{\infty} e^{n^2} r^{-n} = \sum_{n=0}^{\infty} e^{n^2 - n \ln r}.$$

The terms of this sum do not go to zero, and so the sum does not converge.

We conclude that if a system is causal and stable, then its ROC must contain $\{z : |z| \geq 1\}$.

6.7. Difference Equations

Consider a pair of sequences x and y that satisfy

$$y_n = -\sum_{k=1}^{p} a_k y_{n-k} + \sum_{k=0}^{q} b_k x_{n-k}. \tag{6.2}$$

If all the a_k are zero, we have the special case

$$y_n = \sum_{k=0}^{q} b_k x_{n-k},$$

which is the convolution of the finite-duration sequence b with the input x. Such a system is called a **finite impulse response filter** or **FIR filter** for short. An FIR filter is sometimes called a **moving average filter**. This terminology is suggested by the special case $b_0 = \cdots = b_q = 1/(q+1)$, which makes y_n the numerical average of the $q+1$ numbers $x_n, \ldots, x_{n-q}$. For an FIR filter, the largest value of k with $b_k \neq 0$ is called the **order of the filter**.

If one of the a_k is nonzero, then the system is said to be an **infinite impulse response filter**. For an IIR filter, the largest value of k with $a_k \neq 0$ is called the **order of the filter**.

6.7.1. Nonuniqueness

In the IIR case, it is important to note that for a given x, (6.2) does *not* uniquely determine y unless we make further assumptions. This is most easily seen in the special case

$$y_n = \alpha y_{n-1} + \delta_n. \tag{6.3}$$

First suppose that $y_n = 0$ for $n < 0$. Then

$$\begin{aligned} y_0 &= \alpha y_{-1} + \delta_0 = \alpha \cdot 0 + 1 = 1, \\ y_1 &= \alpha y_0 + 0 = \alpha, \\ y_2 &= \alpha y_1 + 0 = \alpha^2, \end{aligned}$$

and we see that $y_n = \alpha^n u(n)$ when $x_n = \delta_n$. However, suppose that instead of assuming $y_n = 0$ for $n < 0$, we assume that $y_n = 0$ for $n \geq 0$. Rewriting (6.3) as

$$y_{n-1} = (y_n - \delta_n)/\alpha,$$

we find that

$$\begin{aligned} y_{-1} &= (y_0 - 1)/\alpha = -1/\alpha, \\ y_{-2} &= (y_{-1} - 0)/\alpha = 1/\alpha^2, \end{aligned}$$

and we see that $y_n = -\alpha^n u(-n-1)$, even though we have again assumed that $x_n = \delta_n$.

6.7.2. The Causal Case

From now on, when we talk about a system whose input and output satisfy a difference equation of the form (6.2), we also assume that the system is causal. In particular, this means that if the input is a causal signal, then the output $y_n = 0$ for $n < 0$ and[1]

$$\begin{aligned} y_0 &= b_0 x_0, \\ y_1 &= -a_1 y_0 + b_0 x_1 + b_1 x_0, \\ y_2 &= -a_1 y_1 - a_2 y_0 + b_0 x_2 + b_1 x_1 + b_2 x_0, \\ &\vdots \end{aligned} \tag{6.4}$$

We thus see that when a causal system satisfying a difference equation of the form (6.2) is restricted to causal inputs, the difference equation can be directly used to compute the corresponding output. This is true even of the larger class of **right-sided inputs**; i.e., inputs x such that $x_k = 0$ for all $k < k_0$ for some finite k_0. It should also be clear that because the a_k and b_k do not depend on n, the system must be time invariant on the class of right-sided inputs. Finally, it can be proved using the difference equation and induction that the system must be linear on the class of right-sided inputs.

The simplest causal signal, and the most important one for characterizing a linear, time-invariant system, is the unit impulse $x_n = \delta_n$. Since the corresponding output is

[1] In other words, to apply a causal signal to a causal system satisfying (6.2) means to solve (6.2) with zero initial conditions.

denoted by h_n, we have $h_n = 0$ for $n < 0$ and

$$\begin{aligned}
h_0 &= b_0\delta_0 = b_0,\\
h_1 &= -a_1h_0 + b_0\delta_1 + b_1\delta_0 = -a_1h_0 + b_1,\\
h_2 &= -a_1h_1 - a_2h_0 + b_0\delta_2 + b_1\delta_1 + b_2\delta_0 = -a_1h_1 - a_2h_0 + b_2,\\
&\vdots
\end{aligned}$$

We now show that if h_n is computed as we have just done, then for any input x, causal or not,

$$\hat{y}_n := \sum_{m=-\infty}^{n} h_{n-m}x_m$$

satisfies the difference equation (6.2). To see this, first write h_n more compactly as

$$h_n := \begin{cases} 0, & n < 0,\\ -\sum_{k=1}^{p} a_k h_{n-k} + \sum_{k=0}^{q} b_k\delta_{n-k}, & n \geq 0. \end{cases}$$

Then

$$\begin{aligned}
\hat{y}_n &:= \sum_{m=-\infty}^{n} h_{n-m}x_m\\
&= \sum_{m=-\infty}^{n}\left[-\sum_{k=1}^{p} a_k h_{n-m-k} + \sum_{k=0}^{q} b_k\delta_{n-m-k}\right]x_m\\
&= -\sum_{k=1}^{p} a_k\left[\sum_{m=-\infty}^{n} h_{n-m-k}x_m\right] + \sum_{k=0}^{q} b_k\left[\sum_{m=-\infty}^{n}\delta_{n-m-k}x_m\right]\\
&= -\sum_{k=1}^{p} a_k\left[\sum_{m=-\infty}^{n} h_{(n-k)-m}x_m\right] + \sum_{k=0}^{q} b_k\left[\sum_{m=-\infty}^{n}\delta_{(n-k)-m}x_m\right]\\
&= -\sum_{k=1}^{p} a_k\hat{y}_{n-k} + \sum_{k=0}^{q} b_k x_{n-k}.
\end{aligned}$$

6.7.3. Solving Difference Equations with MATLAB

Consider a causal system described by the difference equation (6.2). In the case of a causal input, to compute the response y_k for $k = 0,\ldots,n$, use the MATLAB function `y=filter(b,a,x)`, where `a` $= [1, a_1, \ldots, a_p]$, `b` $= [b_0, b_1, \ldots, b_q]$, and `x` $= [x_0, x_1, \ldots, x_n]$.

6.7.4. *z* Transforms of Difference Equations

Let us rewrite the difference equation (6.2) as

$$\sum_{k=0}^{p} a_k y_{n-k} = \sum_{k=0}^{q} b_k x_{n-k}, \tag{6.5}$$

where $a_0 := 1$. We recognize the left-hand side as the convolution of the output y with the finite-duration sequence a, where it is understood that $a_k := 0$ for $k < 0$ and $k > p$. Similarly, the left-hand side is the convolution of the input x with the finite-duration signal b, where it is understood that $b_k := 0$ for $k < 0$ and $k > q$. Using the convolution property of the z transform, it follows that

$$A(z)Y(z) = B(z)X(z),$$

where

$$A(z) := 1 + a_1 z^{-1} + \cdots + a_p z^{-p} \quad \text{and} \quad B(z) := b_0 + b_1 z^{-1} + \cdots + b_q z^{-q}. \tag{6.6}$$

What we would like to do next is divide $A(z)Y(z) = B(z)X(z)$ by $A(z)$ to obtain $Y(z) = H(z)X(z)$, where $H(z) := B(z)/A(z)$, and declare the inverse z transform of $H(z)$ to be the impulse response of the system. However, there are some potential stumbling blocks, which we now consider.

The Regions of Convergence

As usual, we assume that our system is causal. We also assume that the input x is a causal sequence. Hence, so is the output y. Thus, all four transforms $A(z)$, $B(z)$, $X(z)$, and $Y(z)$ are defined for $|z| > R_1$ for some sufficiently large R_1.

We Cannot Divide by Zero

To study the zeros of $A(z)$, it is convenient to introduce the polynomial

$$\Xi(z) := 1 + a_1 z + \cdots + a_p z^p.$$

Since $A(z) = \Xi(1/z)$, $A(z) = 0$ if and only if $1/z$ is a root of Ξ. Since Ξ is a polynomial of degree p, it has exactly p complex roots, say $\xi_1, \ldots, \xi_p$. Furthermore, since $\Xi(0) = 1$, $\xi_k \neq 0$. Hence, $A(z)$ has exactly k zeros, $1/\xi_1, \ldots, 1/\xi_p$.

Let $R := \min\{|\xi_1|, \ldots, |\xi_p|\}$. Since $\Xi(z)$ is nonzero for $|z| < R$, $A(z)$ is nonzero for $|z| > 1/R$. Thus,

$$Y(z) = \frac{B(z)}{A(z)} X(z), \quad |z| > \max\{1/R, R_1\}.$$

We showed in Section 6.7.2 that if a causal system satisfies (6.2) and a causal input is applied, then the output is uniquely determined by the difference equation. In particular, we showed that the impulse response can be determined in this way. We then showed that $y = h * x$ also satisfies the difference equation. Since we have just shown that $Y(z) = [B(z)/A(z)]X(z)$, it follows that $H(z) = B(z)/A(z)$.

Frequency Response

When we study the design of discrete-time filters, it often necessary to plot the frequency response; i.e., the DTFT of the system impulse response sequence. When the system is described by a difference equation and we know the transfer function $H(z)$, we simply plot $H(e^{j2\pi f})$. This assumes that all points $z = e^{j2\pi f}$, which lie on the unit circle, belong to the ROC of $H(z)$; i.e., we assume the system is stable. If $H(z) = B(z)/A(z)$, where $A(z)$ and $B(z)$ are polynomials in z^{-1} as in (6.6), we can use the MATLAB command `polyval`[2] for evaluating polynomials. Suppose that $\mathtt{a} = [1, a_1, \ldots, a_p]$ and $\mathtt{b} = [b_0, \ldots, b_q]$. Then

```
f = linspace(-1/2,1/2,200);
zinv = exp(-j*2*pi*f);
y = polyval(fliplr(b),zinv)./polyval(fliplr(a),zinv);
```

computes $y = H(e^{j2\pi f})$ for $|f| \le 1/2$.

When we do filter design, $A(z)$ and $B(z)$ are sometimes specified in terms of their roots, say

$$H(z) = \frac{b_0 \prod_{k=1}^{q} (1 - \beta_k z^{-1})}{\prod_{k=1}^{p} (1 - \alpha_k z^{-1})} = \frac{b_0 + b_1 z^{-1} + \cdots + b_q z^{-q}}{1 + a_1 z^{-1} + \cdots + a_p z^{-p}}.$$

The vector $\mathtt{a} = [1, a_1, \ldots, a_p]$ is returned by the MATLAB command `a=poly(alpha)` if $\mathtt{alpha} = [\alpha_1, \ldots, \alpha_p]$. Similarly, `b=b0*poly(beta)` returns the vector $\mathtt{b} = [b_0, \ldots, b_q]$. Using the coefficient vectors `a` and `b`, we can evaluate the DTFT $H(e^{j2\pi f})$ as above using `polyval` and `fliplr`.

6.7.5. Stable Inverses and Minimum Phase

Recall that a system $H(z) = B(z)/A(z)$ is stable if and only if all the poles of $H(z)$ (roots of $A(z)$) lie strictly inside the unit circle. The inverse system, whose transfer function is $A(z)/B(z)$, is stable if and only if all the zeros of $H(z)$ (roots of $B(z)$) lie strictly inside the unit circle.

[2] To evaluate $c_0 x^n + c_1 x^{n-1} + \cdots + c_{n-1} x + c_n$, call `polyval(c,x)` with $\mathtt{c} = [c_0, c_1, \ldots, c_n]$. To evaluate $c_0 + c_1 x + \cdots + c_n x^n$, use `polyval(fliplr(c),x)`, where the `fliplr` reverses the order of the elements from left to right; i.e., applying `fliplr` to $[c_0, \ldots, c_n]$ returns $[c_n, \ldots, c_0]$.

A system $H(z) = B(z)/A(z)$ is called **minimum phase** if all the zeros of $H(z)$ are strictly inside the unit circle.

A system $H(z)$ is stable and minimum phase if and only if its inverse is also stable and minimum phase.

We now give some insight into the terminology "minimum phase."

***Example* 6.7.1.** If $H(z) = 1 - az^{-1}$ for $0 < a < 1$, plot $H(e^{j2\pi f})$. Repeat for $a > 1$.

Solution. Since $H(e^{j2\pi f}) = 1 - ae^{-j2\pi f}$, we begin by plotting the path of the complex number $ae^{-j2\pi f}$ in the complex plane as f goes from 0 to $1/2$ and from 0 to $-1/2$. See the drawing at the left in Figure 6.1. The number $ae^{-j2\pi f}$ lies on the

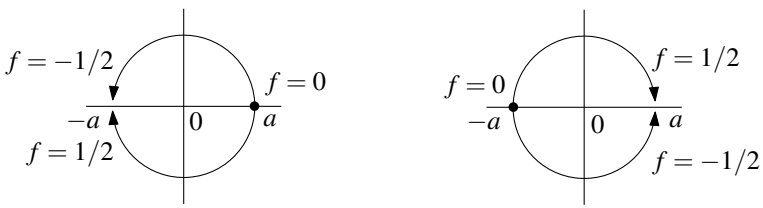


Figure 6.1. Paths of $ae^{-j2\pi f}$ (left) and $-ae^{-j2\pi f}$ (right).

circle of radius a centered at the origin. When $f = 0$, the point is at a on the real axis. As f becomes positive, the point drops below the real axis and continues on the lower half circle, ending at $ae^{-j2\pi(1/2)} = ae^{-j\pi} = -a$ on the real axis. As f starts at zero and becomes negative, the point leaves a and travels along the top half circle, again ending at $ae^{-j2\pi(-1/2)} = ae^{j\pi} = -a$. A little thought shows that the path of $-ae^{-j2\pi f}$ is given by the drawing at the right in Figure 6.1. For $0 < a < 1$, we see that path of $1 - ae^{-j2\pi f}$ is given by the drawing at the left in Figure 6.2. The drawing for $a > 1$ is at the right in Figure 6.2.

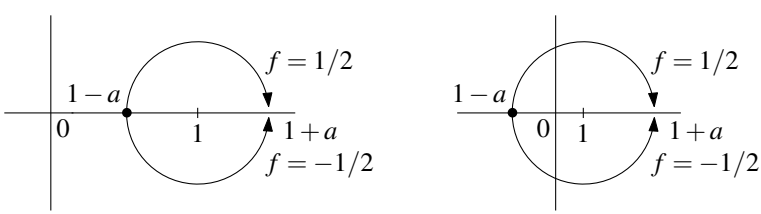


Figure 6.2. Path of $1 - ae^{-j2\pi f}$ for $0 < a < 1$ (left) and $a > 1$ (right).

Since the points on the circle at the left in Figure 6.2 are all in the right half plane, the points on the circle all have phase strictly between $\pm\pi/2$. A plot of the phase of $1-ae^{-j2\pi f}$ for $|f| \le 1/2$ is shown at the top in Figure 6.3. In contrast, for the circle

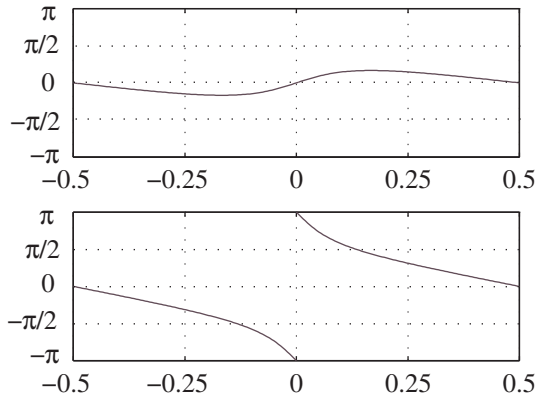


Figure 6.3. The phase of $1-ae^{-j2\pi f}$ for $0 < a < 1$ (top) and $a > 1$ (bottom)

.

at the right in Figure 6.2, some points are in the left half plane and some are in the right. There are two points on the imaginary axis, and their phases are $\pm\pi/2$. In fact, there are points of all phases between $\pm\pi$ as well as a jump discontinuity of 2π at $f = 0$.

We now generalize the example to complex a, say $a = \alpha e^{j2\pi f_0}$, where $\alpha > 0$. Then $H(e^{j2\pi f}) = 1-\alpha e^{j2\pi f_0}e^{-j2\pi f} = 1-\alpha e^{-j2\pi(f-f_0)}$. Hence, the curves in Figure 6.3, which are periodic with period one, shift by f_0. The point of all this is that the phase variation of $1-az^{-1}$ for $z = e^{j2\pi f}$ has minimum variation when $|a| < 1$.

6.7.6. All-Pass Systems

An **all-pass system** is one for which $|H(e^{j2\pi f})|$ is a positive constant for all f. The simplest example is

$$H(z) = C\frac{1-(1/\overline{\lambda})z^{-1}}{1-\lambda z^{-1}},$$

where $\lambda \neq 0$ and C is an arbitrary constant. This transfer function has one pole, λ, and one zero, $1/\overline{\lambda}$, which is the conjugate reciprocal of the pole. To see that $H(z)$ is

indeed all pass, first rewrite it as

$$H(z) = C\frac{-z^{-1}}{\overline{\lambda}} \cdot \frac{1-\overline{\lambda}z}{1-\lambda z^{-1}}.$$

In this form, it is easy to see that

$$H(e^{j2\pi f}) = C\frac{-e^{-j2\pi f}}{\overline{\lambda}} \cdot \frac{1-\overline{\lambda}e^{j2\pi f}}{1-\lambda e^{-j2\pi f}}.$$

Since the factor on the right is the quotient of complex conjugates, it has magnitude one. Hence, $|H(e^{j2\pi f})| = C/|\lambda|$ for all f.

We can now show that every stable transfer function can be written as the product of a stable all-pass system and a minimum-phase stable system as follows. If $H(z)$ is not minimum phase, it has a zero, say μ, with $|\mu| > 1$. Then we can write $H(z)$ in the form $H(z) = (1-\mu z^{-1})H_0(z)$ for some other stable transfer function with one less zero. Now write

$$\begin{aligned} H(z) &= (1-\mu z^{-1})H_0(z) \\ &= \underbrace{\frac{1-\mu z^{-1}}{1-(1/\overline{\mu})z^{-1}}}_{\text{stable all-pass}} \cdot \underbrace{(1-(1/\overline{\mu})z^{-1})H_0(z)}_{\text{one less zero outside unit circle}}, \end{aligned}$$

where we have used the fact that $|\mu| > 1$ implies $|1/\overline{\mu}| < 1$. Continuing in this way, we can write $H(z) = H_{\text{ap}}(z)H_{\text{mp}}(z)$, where $H_{\text{mp}}(z)$ is stable and minimum phase and H_{ap} is stable and all pass and of order equal to the number of zeros of $H(z)$ outside the unit circle.

6.8. Summary

We have shown that every linear, time-invariant system can be represented as a convolution. Furthermore, the system is stable if and only if the impulse response is absolutely summable, and the system is causal if and only if the impulse response is zero for negative time.

In the z transform domain, stability is equivalent to having the ROC of the transfer function contain the unit circle. Causality corresponds to a transfer-function ROC that is the exterior of a circle.

For causal systems that are described by difference equations of the form (6.2), stability corresponds to $A(z)$ having all its roots strictly inside the unit circle.

As we have mentioned several times, the general discrete-time linear time-invariant system corresponds to a convolution involving an infinite sum of the form

$$\sum_{m=-\infty}^{\infty} h_k x_{n-k}.$$

In the case of a causal input and a causal system, this reduces to

$$\sum_{m=0}^{n} h_k x_{n-k}.$$

Although this involves only a finite amount of computation for each $n \geq 0$, the amount of computation grows with n and will eventually become intractable. However, in the case of an FIR filter, e.g., $h_k = b_k$ for $k = 0, \ldots, q$ and $h_k = 0$ otherwise, the upper limit on the sum can be replaced by q. In the case of an IIR filter described by the difference equation 6.2, the output can again be computed with a finite, *nongrowing* amount of computation by recursively evaluating the difference equation.

Problems

6.1. Find the impulse response of the system defined by

$$(Ax)_n := a^{-n} \sum_{k=-\infty}^{n} a^k x_k.$$

6.2. Consider the system with transfer function $H(z) = z + 3 + 7z^{-1} + 5z^{-2}$. Is the system causal?

6.3. Can an FIR filter be unstable? Why or why not?

6.4. Consider the nonlinear system defined by

$$(Ax)_n := \sum_{k=n-3}^{n} x_k^2.$$

(a) Is the system causal? (Yes/No)
(b) Determine whether or not the system is time invariant.
(c) Determine whether or not the system is stable.

6.5. A linear, time-invariant system has impulse response $h_n = 1/(1+n^2)$. Determine whether or not the system is stable.

6.6. A linear, time-invariant system has impulse response $h_n = 1/(1+|n|)$. Determine whether or not the system is stable.

6.7. A causal system satisfies the difference equation

$$y_n = \tfrac{7}{12} y_{n-1} - \tfrac{1}{12} y_{n-2} + x_n.$$

Determine whether or not the system is stable.

6.8. **MATLAB.** For the difference equation of the preceding problem, plot y_n for $n = 0, \ldots, 10$ when $x_n = \delta_n$.

6.9. A causal system satisfies the difference equation

$$y_n = 6y_{n-1} - 9y_{n-2} + x_n.$$

Determine whether or not the system is stable.

6.10. **MATLAB.** For the difference equation of the preceding problem, plot y_n for $n = 0, \ldots, 10$ when $x_n = \delta_n$.

6.11. Consider the causal discrete-time system with transfer function

$$H(z) = \frac{7 - z^{-2}}{1 - 5z^{-1} + 6z^{-2}}.$$

Write out the difference equation for this system, and determine whether or not the system is stable.

6.12. Consider the FIR filter with transfer function

$$H(z) = 1 - e^{j2\pi f_0} z^{-1}.$$

(a) Write out the difference equation for this system.
(b) Use the difference equation to compute the output y_n corresponding to $x_n = e^{j2\pi f_0 n}$.
(c) Write out formulas for the DTFTs $X(f)$, $H(e^{j2\pi f})$, and their product, $Y(f) := H(e^{j2\pi f})X(f)$. *Be sure to simplify your formula for $Y(f)$ as much as possible.*
(d) Now use the difference equation to compute y_n corresponding to the causal input $x_n = e^{j2\pi f_0 n}u(n)$, where u is the unit-step function.

6.13. Consider the IIR filter with transfer function

$$H(z) = \frac{1 - e^{j2\pi f_0} z^{-1}}{1 - (1/9)z^{-2}}.$$

(a) Is the system stable?
(b) If $x_n = e^{j2\pi f_0 n}$ is the input to this system, determine the output y_n.
(c) If $x_n = e^{j2\pi f_0 n}u(n)$ is input to this system, determine the output y_n.

6.14. The signal $x_n = e^{jn\pi/2}$ is applied to the filter

$$H(z) = \frac{1 - jz^{-1}}{1 - (1/4)z^{-2}}.$$

Determine the output signal.

CHAPTER 7
IIR Filter Design

7.1. The Bilinear Transformation

Our goal is to design discrete-time systems described by a difference equations, since these systems are readily implemented on a computer. One approach is to first design a Laplace transform $\mathscr{H}(s)$ that is a quotient of polynomials is s and then convert it to a z transform that is a quotient of polynomials in z^{-1}, and therefore corresponds to a difference equation. The conversion method we use here is called the **bilinear transformation**. Suppose $\mathscr{H}(s)$ is a quotient of polynomials and that the polynomials have been factored so that

$$\mathscr{H}(s) = \mathscr{H}_0 \frac{\prod_{k=1}^{m}(s-\mu_k)}{\prod_{k=1}^{n}(s-\lambda_k)}. \tag{7.1}$$

The degree of the denominator polynomial is called the **order** of the filter. We convert $\mathscr{H}(s)$ into $H(z)$ by means of the bilinear transformation[1]

$$s = \frac{2(z-1)}{T(z+1)}. \tag{7.2}$$

***Example* 7.1.1.** If $\mathscr{H}(s) = (s-1)/(s^2+7s+12)$, use the bilinear transformation with $T = 2$ to find $H(z)$.

Solution. First note that the poles of $\mathscr{H}(s)$ are $s = -3$ and $s = -4$. Thus, $\mathscr{H}(s) = (s-1)/[(s+3)(s+4)]$, and we have

$$H(z) = \frac{(z-1)/(z+1)-1}{((z-1)/(z+1)+3)((z-1)/(z+1)+4)}$$

[1] A simple calculation shows that the bilinear transformation is invertible with

$$z = \frac{1+sT/2}{1-sT/2}.$$

In particular, if s is pure imaginary, then the numerator and denominator are complex conjugates, which implies $|z| = 1$; i.e., the inverse transformation maps the imaginary axis of the s-plane to the unit circle in the z-plane.

$$
\begin{aligned}
&= \frac{-z-1}{10z^2+11z+3} \\
&= \frac{-z-1}{10(z+3/5)(z+1/2)} \\
&= \frac{-z^{-1}(1+z^{-1})}{10(1+(3/5)z^{-1})(1+(1/2)z^{-1})}.
\end{aligned}
$$

To see what happens in general, write

$$
\begin{aligned}
H(z) &:= \mathscr{H}\left(\frac{2(z-1)}{T(z+1)}\right) \\
&= \mathscr{H}_0 \frac{(T/2)^{n-m}\prod_{k=1}^{m}(1-\mu_k T/2)}{\prod_{k=1}^{n}(1-\lambda_k T/2)} \cdot \frac{(1+z^{-1})^{n-m}\prod_{k=1}^{m}(1-\widehat{\mu}_k z^{-1})}{\prod_{k=1}^{n}(1-\widehat{\lambda}_k z^{-1})},
\end{aligned} \tag{7.3}
$$

where

$$
\widehat{\mu}_k := \frac{1+\mu_k T/2}{1-\mu_k T/2} \quad \text{and} \quad \widehat{\lambda}_k := \frac{1+\lambda_k T/2}{1-\lambda_k T/2}. \tag{7.4}
$$

Notice that the numerator and denominator always have the same number of roots. The key to deriving (7.3) is to write

$$
\begin{aligned}
s-\mu_k &= \frac{2(z-1)}{T(z+1)} - \mu_k = \frac{2(1-z^{-1})}{T(1+z^{-1})} - \mu_k \\
&= \frac{2}{T(1+z^{-1})}[(1-z^{-1}) - (1+z^{-1})\mu_k T/2] \\
&= \frac{2[(1-\mu_k T/2) - (1+\mu_k T/2)z^{-1}]}{T(1+z^{-1})} = \frac{2(1-\mu_k T/2)}{T(1+z^{-1})}(1-\widehat{\mu} z^{-1})
\end{aligned}
$$

and a similar expression for $s-\lambda_k$. There are several important relationships between (7.1) and (7.3).

1. The formulas for $\widehat{\mu}_k$ and $\widehat{\lambda}_k$ show how to convert the poles and zeros of (7.1) into those of (7.3).

2. Since

$$
\left|\widehat{\lambda}_k\right|^2 = \frac{(1+\lambda_k^{\text{real}} T/2)^2 + (\lambda_k^{\text{imag}})^2}{(1-\lambda_k^{\text{real}} T/2)^2 + (\lambda_k^{\text{imag}})^2},
$$

we see that $\widehat{\lambda}_k$ is strictly inside the unit circle if and only if λ_k is in the strict left half plane; i.e., the continuous-time system is stable if and only if the discrete-time system is stable.

3. For $n \geq m$, which is the typical case, the number of poles is preserved, and the number of zeros is increased by $n-m$.

4. If we put $z = e^{j2\pi f}$ in the bilinear transformation (7.2), we find that

$$\begin{aligned} s = \frac{2(e^{j2\pi f}-1)}{T(e^{j2\pi f}+1)} = \frac{2}{T}\cdot\frac{(e^{j2\pi f}-1)(e^{-j2\pi f}+1)}{(e^{j2\pi f}+1)(e^{-j2\pi f}+1)} &= j\frac{2}{T}\cdot\frac{2\sin(2\pi f)}{2[1+\cos(2\pi f)]} \\ &= j\frac{2}{T}\cdot\frac{2\sin(\pi f)\cos(\pi f)}{2\cos^2(\pi f)} \\ &= j\frac{2}{T}\tan(\pi f). \end{aligned}$$

Hence,

$$H(z)\big|_{z=e^{j2\pi f}} = \mathscr{H}(j\omega),$$

where

$$\omega = \frac{2}{T}\tan(\pi f). \tag{7.5}$$

In other words, DTFT frequencies in $[-1/2, 1/2]$ correspond to radian frequencies $\omega = (2/T)\tan(\pi f)$.

7.2. Analog Transfer Functions

Given a circuit with input voltage or current $x(t)$ and output voltage or current $y(t)$, the output is related to the input via the convolution integral $y(t) = \int_{-\infty}^{\infty} h(t-\tau)x(\tau)\,d\tau$. Furthermore, since a circuit is a causal system, the impulse response $h(t)$ must be causal; i.e., $h(t) = 0$ for $t < 0$. Recall that the **transfer function** is the one-sided **Laplace transform**,

$$\mathscr{H}(s) := \int_0^{\infty} h(t)e^{-st}\,dt,$$

where s is a complex variable, and the **frequency response** is

$$\mathscr{H}(j\omega) = \mathscr{H}(s)\big|_{s=j\omega} = \int_0^{\infty} h(t)e^{-j\omega t}\,dt,$$

where ω is a real variable. Since voltages and currents in physical systems are real, the impulse response must be real. Hence, $\mathscr{H}(-j\omega) = \overline{\mathscr{H}(j\omega)}$, and we have

$$|\mathscr{H}(j\omega)|^2 = \mathscr{H}(j\omega)\overline{\mathscr{H}(j\omega)} = \mathscr{H}(j\omega)\mathscr{H}(-j\omega). \tag{7.6}$$

When the circuit is composed of resistors, capacitors, inductors, and operational amplifiers, the transfer function $\mathscr{H}(s)$ will have the form $\mathscr{H}(s) = p(s)/q(s)$, where $p(s)$ and $q(s)$ are polynomials in s. Hence,

$$\mathscr{H}(s)\mathscr{H}(-s) = \frac{p(s)}{q(s)} \cdot \frac{p(-s)}{q(-s)}.$$

Let $e(s)$ denote the terms of $p(s)$ with even powers of s and let $o(s)$ denote the terms of $p(s)$ with odd powers of s. Then $e(s)$ is even and $o(s)$ is odd; i.e., $e(-s) = e(s)$ and $o(-s) = -o(s)$. It follows that

$$p(s)p(-s) = [e(s)+o(s)][e(s)-o(s)] = e(s)^2 - o(s)^2,$$

which involves only even powers of s. Similarly, $q(s)q(-s)$ involves only even powers of s. Hence, $\mathscr{H}(s)\mathscr{H}(-s)$ also involves only even powers of s. In particular, $|\mathscr{H}(j\omega)|^2$ involves only even powers of $j\omega$, which means that $|\mathscr{H}(j\omega)|^2$ is a real-valued function of ω^2.

7.3. Butterworth Filters

Continuous-time **Butterworth filters** are designed to have a frequency response satisfying

$$|\mathscr{H}(j\omega)|^2 = \frac{H_0^2}{1+(\omega/\omega_c)^{2n}}. \tag{7.7}$$

The maximum possible value of $|\mathscr{H}(j\omega)|^2$ is H_0^2 (and occurs at $\omega = 0$). For $|\omega| < \omega_c$, $|\mathscr{H}(j\omega)|^2 > H_0^2/2$ and for $|\omega| > \omega_c$, $|\mathscr{H}(j\omega)|^2 < H_0^2/2$. Thus, ω_c is the frequency at which the ratio of $|\mathscr{H}(j\omega)|^2$ to its maximum possible value is equal to one half. On the dB scale,

$$10\log_{10}\tfrac{1}{2} \approx -3.01 \text{ dB}.$$

Hence, ω_c is called the **3-dB cutoff frequency**. The parameter n is the order of the filter. As $\omega \to \infty$, $|\mathscr{H}(j\omega)|^2 \to 0$. In Figure 7.1, it is easily seen that the maximum attenuation in the **passband** occurs at $\omega = \omega_p$ and is equal to

$$10\log_{10}|\mathscr{H}(j0)|^2 - 10\log_{10}|\mathscr{H}(j\omega_p)|^2 = 10\log_{10}(1+(\omega_p/\omega_c)^{2n}).$$

The minimum attenuation in the **stopband** occurs at $\omega = \omega_s$ and is equal to

$$10\log_{10}|\mathscr{H}(j0)|^2 - 10\log_{10}|\mathscr{H}(j\omega_s)|^2 = 10\log_{10}(1+(\omega_s/\omega_c)^{2n}).$$

Now suppose that the passband attenuation is required to be at most A_p (in dB) and the stopband attenuation is required to be at least A_s (in dB). Mathematically, we want to have

$$10\log_{10}(1+(\omega_p/\omega_c)^{2n}) \le A_p \quad \text{and} \quad 10\log_{10}(1+(\omega_s/\omega_c)^{2n}) \ge A_s,$$

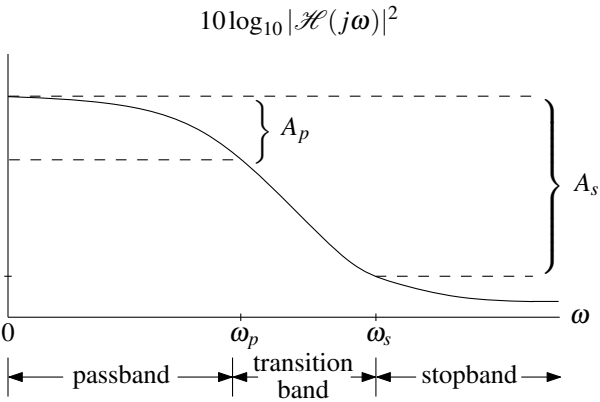


Figure 7.1. Frequency response (in dB) of a Butterworth filter with passband $0 \le \omega \le \omega_p$ and stopband $\omega > \omega_s$.

which we rearrange as

$$(\omega_p/\omega_c)^n \le \sqrt{10^{A_p/10}-1} \quad \text{and} \quad (\omega_s/\omega_c)^n \ge \sqrt{10^{A_s/10}-1}. \tag{7.8}$$

It follows that

$$\ln \frac{(\omega_s/\omega_c)^n}{(\omega_p/\omega_c)^n} \ge \ln \sqrt{\left(10^{A_s/10}-1\right)/\left(10^{A_p/10}-1\right)},$$

or

$$n \ge \frac{\ln \sqrt{\left(10^{A_s/10}-1\right)/\left(10^{A_p/10}-1\right)}}{\ln(\omega_s/\omega_p)}. \tag{7.9}$$

Hence, given the passband and stopband frequencies and attenuations, the above formula tells us how to choose the order of the Butterworth filter. It still remains to choose ω_c. Now that n has been fixed *as an integer* satisfying the above inequality, we can use (7.8) to bound the selection of ω_c. We find that ω_c can be any frequency satisfying

$$\omega_p\left[10^{A_p/10}-1\right]^{-1/(2n)} \le \omega_c \le \omega_s\left[10^{A_s/10}-1\right]^{-1/(2n)}. \tag{7.10}$$

To design a specific circuit with the frequency response in (7.7), we need to know more than $|\mathscr{H}(j\omega)|^2$, we need to know $\mathscr{H}(s)$. Comparing (7.6) and (7.7), make the substitution $s = j\omega$; i.e., $\omega = s/j = -js$. Then we need to find $\mathscr{H}(s)$ such that

$$\mathscr{H}(s)\mathscr{H}(-s) = \frac{H_0^2}{1+(-js/\omega_c)^{2n}}.$$

The problem is to factor the right-hand side so that we can identify $\mathscr{H}(s)$. The poles are the solutions of $(-js/\omega_c)^{2n} = -1$. Thus, $-js/\omega_c$ is the $2n$th root of -1, which implies that

$$-js_k/\omega_c = e^{j\pi(2k-1)/(2n)}, \quad k = 1,\ldots,2n,$$

or

$$s_k = \omega_c e^{j\pi(n+2k-1)/(2n)}, \quad k = 1,\ldots,2n. \tag{7.11}$$

First note that we cannot have $s_k = \pm j\omega_c = \omega_c e^{\pm j\pi/2}$ because we cannot have $(n+2k-1)/n$ equal to an odd integer. Second, the only way to have $s_k = \pm\omega_c = \omega_c e^{j\pi m}$ is to have $(n+2k-1)/(2n)$ equal to an integer, which requires n to be odd. Third, since $\mathrm{Re}\, s_k < 0$ if and only if $k = 1,\ldots,n$, we take

$$\mathscr{H}(s) = \frac{H_0}{\mathscr{B}_n(s-s_1)\cdots(s-s_n)},$$

where $\mathscr{B}_n$ is chosen so that $\mathscr{H}(0) = H_0$; i.e., $\mathscr{B}_n = 1\big/\prod_{k=1}^{n}(-s_k)$. Note that $\mathscr{B}_n$ is real because all roots (except $s_k = \pm\omega_c$) occur in conjugate pairs (Problem 7.4). Another consequence of fact that the roots occur in conjugate pairs is that the coefficients of

$$(s-s_1)\cdots(s-s_n) = s^n + \beta_1 s^{n-1} + \cdots + \beta_{n-1}s + \beta_n$$

must be real (Problem 7.6). These coefficients can be found from the roots using the MATLAB function `poly`. If `svec` $= [s_1,\ldots,s_n,s_{n+1},\ldots,s_{2n}]$, and `beta` $= [1,\beta_1,\ldots,\beta_n]$, then `beta=poly(svec(1:n))`. Although theory says that the β_k are real, due to roundoff error `poly` may return some elements of `beta` with nonzero imaginary parts. For this reason, it is better to use the longer expression `beta=real(poly(svec(1:n)))`.

If our goal is a discrete-time lowpass filter, we really do not need to know the β_k. Here is what we need to do.

1. Define parameters of the desired discrete-time lowpass filter, f_p, A_p, f_s, and A_s.
2. Use (7.5) to **prewarp** f_p and f_s to get ω_p and ω_s.
3. Compute the filter order n to be an *integer* satisfying (7.9).
4. Choose a value of ω_c satisfying (7.10).
5. Compute the poles s_k of $H(s)$ using (7.11) for $k = 1,\ldots,n$.
6. Use (7.4) to convert $\lambda_k = s_k$ into poles of the discrete-time filter $\widehat{\lambda}_k$.
7. Use the MATLAB command `poly` to convert the $\widehat{\lambda}_k$ into the difference-equation coefficients a_k.

8. Since a Butterworth filter has no zeros, the corresponding discrete-time filter has n repeated zeros at at $z = -1$ according to (7.3). Use `poly` to convert the n zeros at $z = -1$ into the difference-equation coefficients b_k.

The foregoing recipe is carried out by the following script, which also plots the Butterworth poles s_k and the frequency response $10\log_{10}|H(e^{j2\pi f}|^2$ of the designed discrete-time filter.

```
fp = .1;  Ap = .25; % User's parameters for
fs = .15; As = 40;  % discrete-time lowpass filter

wp = 2*tan(pi*fp);  % Prewarp DTFT frequencies
ws = 2*tan(pi*fs);

dp = 10^(Ap/10)-1;  % Compute order n
ds = 10^(As/10)-1;
n = ceil(log(ds/dp)/log(ws/wp)/2);

twon = 2*n; xx = 1/twon;      % For omega_c, use the average
wc = (wp/dp^xx+ws/ds^xx)/2;   % of its upper and lower bounds

k = [1:twon];
svec = wc*exp(j*pi*(n+2*k-1)/twon); % poles of H(s)H(-s)
lambda = svec(1:n);                 % poles of H(s)

t = linspace(0,2*pi,200);     % plot circle
x = wc*cos(t); y = wc*sin(t);
figure(1); plot(x,y,'m'); grid on;
hold on
plot(svec,'x')                % plot all 2n roots
plot(lambda,'o');             % circle LHP roots
hold off

lambdahat = (1+lambda/2)./(1-lambda/2); % poles of z transform
a = real(poly(lambdahat));  % difference eq. coefficients
b = poly(-ones(1,n));
b = (sum(a)/sum(b))*b;      % normalize to make H(z)=1 for z=1

f = linspace(0,1/2,2000);   % Plot H(exp(j*2*pi*f)) on dB scale
zinv = exp(-j*2*pi*f);
Hf = polyval(fliplr(b),zinv)./polyval(fliplr(a),zinv);
ff = [fp fs]; Hff = -[Ap As]; % Put o's on graph at design pnts
figure(2)
plot(f,10*log10(Hf.*conj(Hf)),ff,Hff,'ro'); grid on
```

7.4. Chebyshev Filters of the First Kind

Continuous-time **Chebyshev filters of the first kind**, or **Chebyshev-I filters** for short, are designed to have a frequency response satisfying

$$|\mathscr{H}(j\omega)|^2 = \frac{H_0^2}{1+\varepsilon^2 T_n(\omega/\omega_p)^2}, \tag{7.12}$$

where $T_n(\omega)$ is the nth-degree Chebyshev polynomial, and $0 < \varepsilon < 1$. The order of the filter is n.

7.4.1. The Chebyshev Polynomials

The Chebyshev polynomials are defined by

$$\begin{aligned} T_0(x) &:= 1, \\ T_1(x) &:= x, \\ T_{n+1}(x) &:= 2xT_n(x) - T_{n-1}(x), \quad n \geq 1. \end{aligned}$$

It is easy to check that

$$\begin{aligned} T_2(x) &= 2x^2 - 1, \\ T_3(x) &= 4x^3 - 3x, \\ T_4(x) &= 8x^4 - 8x^2 + 1, \end{aligned}$$

which are sketched in Figure 7.2. For $n = 0,\ldots,4$, we see that $T_n(x)$ has n real roots in $(-1,1)$ and that for $-1 \leq x \leq 1$, we have $|T_n(x)| \leq 1$. We also note that as $x \to \infty$, $T_n(x) \to \infty$. To show that these results hold for arbitrary positive integers n, we need the fact, derived later, that

$$T_n(x) = \cos(n\cos^{-1}x), \quad \text{for } -1 \leq x \leq 1. \tag{7.13}$$

(In MATLAB use `acos` for $\cos^{-1}$.) This formula implies that if $|x| \leq 1$, then $T_n(x) = 0$ if and only if $n\cos^{-1}x$ is an odd multiple of $\pi/2$. Solving for x, we find that for $n \geq 1$, the n distinct roots of $T_n(x)$ are

$$x_k = \cos\left([2k-1]\frac{\pi}{2n}\right), \quad k = 1,2,\ldots,n. \tag{7.14}$$

Hence, $T_n(x)$ has n real roots satisfying

$$-1 < x_n < \cdots < x_1 < 1.$$

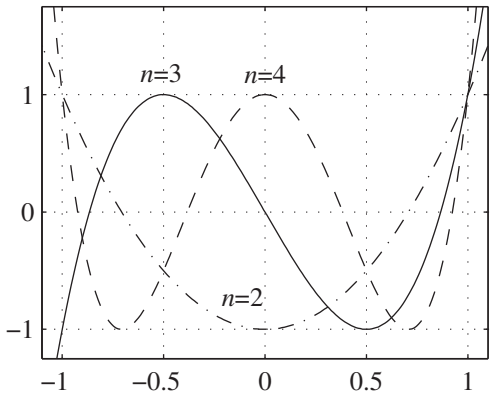


Figure 7.2. Chebyshev polynomials $T_n(x)$ for $n = 2,3,4$.

Now write

$$T_n(x) = c_n \prod_{k=1}^{n} (x - x_k).$$

Since all the factors of the product are positive when $x \geq 1$, and since $T_n(1) = \cos(n \cos^{-1} 1) = \cos(0) = 1$, $c_n > 0$. It follows that for $n \geq 1$, $T_n(x)$ is strictly increasing for $x \geq 1$ and that $T_n(x) \geq c_n(x - x_1)^n \to \infty$ as $x \to \infty$.

Derivation of (7.13)

For $|x| \leq 1$, put $\widetilde{T}_n(x) := \cos(n \cos^{-1} x)$. Then $\widetilde{T}_0(x) = 1 = T_0(x)$ and $\widetilde{T}_1(x) = x = T_1(x)$. Next we use the **trigonometric identity**

$$\cos(A \pm B) = \cos A \cos B \mp \sin A \sin B$$

to write, for $n \geq 1$ and $\theta := \cos^{-1} x$,

$$\begin{aligned} \widetilde{T}_{n\pm1}(x) &= \cos([n \pm 1]\theta) \\ &= \cos(n\theta)\cos(\theta) \mp \sin(n\theta)\sin(\theta) \\ &= \widetilde{T}_n(x)x \mp \sin(n\theta)\sin(\theta). \end{aligned}$$

It follows that

$$\widetilde{T}_{n+1}(x) + \widetilde{T}_{n-1}(x) = 2x\widetilde{T}_n(x),$$

or

$$\widetilde{T}_{n+1}(x) = 2x\widetilde{T}_n(x) - \widetilde{T}_{n-1}(x).$$

Since $\widetilde{T}_0(x) = T_0(x)$ and $\widetilde{T}_1(x) = T_1(x)$, and since $\widetilde{T}_n$ and T_n obey the same recursion, we must have $\widetilde{T}_n(x) = T_n(x)$ for $|x| \le 1$.

7.4.2. Chebyshev-I Filters

Several Chebyshev-I frequency responses are shown in Figure 7.3. Using the

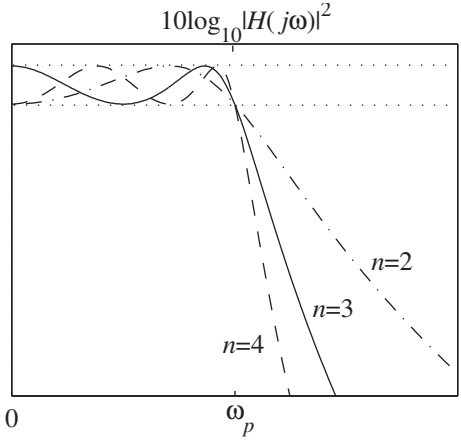


Figure 7.3. Frequency response (in dB) of Chebyshev-I filters for $n = 2,3,4$. The distance between the two dotted horizontal lines is the ripple (in dB). The level of the top dotted line is $10\log_{10} H_0^2$, and the level of the bottom dotted line is $10\log_{10}[H_0^2/(1+\varepsilon^2)]$.

foregoing analysis of the Chebyshev polynomials T_n, we can infer some simple properties of $|\mathscr{H}(j\omega)|^2$. The maximum attenuation in the passband $0 \le \omega \le \omega_p$ occurs several times and is equal to

$$10\log_{10} H_0^2 - 10\log_{10} \frac{H_0^2}{1+\varepsilon^2} = 10\log_{10}(1+\varepsilon^2).$$

To make this at most A_p requires

$$\varepsilon \le \sqrt{10^{A_p/10} - 1}.$$

The stopband will begin at some $\omega_s > \omega_p$, and the minimum attenuation occurs at ω_s and is equal to

$$10\log_{10} H_0^2 - 10\log_{10} \frac{H_0^2}{1+\varepsilon^2 T_n(\omega_s/\omega_p)^2} = 10\log_{10}(1+\varepsilon^2 T_n(\omega_s/\omega_p)^2).$$

To make this greater than or equal to A_s, we need

$$T_n(\omega_s/\omega_p) \geq \frac{\sqrt{10^{A_s/10}-1}}{\varepsilon} \geq \sqrt{\frac{10^{A_s/10}-1}{10^{A_p/10}-1}}.$$

Since $\omega_s > \omega_p$, we can use the fact that $T_n(\omega_s/\omega_p) = \cosh(n\cosh^{-1}(\omega_s/\omega_p))$ (see Problem 7.8) to write

$$n \geq \frac{\cosh^{-1}\left(\sqrt{\left(10^{A_s/10}-1\right)/\left(10^{A_p/10}-1\right)}\right)}{\cosh^{-1}(\omega_s/\omega_p)}. \tag{7.15}$$

(In MATLAB use `acosh` for $\cosh^{-1}$.)

To find $\mathscr{H}(s)$, we proceed as we did for Butterworth filters by writing

$$|\mathscr{H}(j\omega)|^2 = \mathscr{H}(j\omega)\overline{\mathscr{H}(-j\omega)} = \frac{H_0^2}{1+\varepsilon^2 T_n(\omega/\omega_p)^2}$$

and substituting $\omega = s/j$. We must then find the roots of the resulting denominator polynomial on the right. Now $1+\varepsilon^2 T_n(-js/\omega_p)^2 = 0$ if and only if

$$T_n(-js/\omega_p) = \pm j/\varepsilon,$$

or

$$\cos(n\cos^{-1}(-js/\omega_p)) = \pm j/\varepsilon.$$

Following [2, p. 45], let u and v denote the real and imaginary parts of $\cos^{-1}(-js/\omega_p)$. Then using the identity in Problem 7.9,

$$\begin{aligned}\pm j/\varepsilon &= \cos(n[u+jv]) \\ &= \cos nu\cosh nv - j\sin nu\sinh nv.\end{aligned}$$

Since the left-hand side is pure imaginary, and since the hyperbolic cosine of a real number is positive, we must have $\cos nu = 0$; i.e., nu must be an odd multiple of $\pi/2$. We therefore have

$$u = \frac{(2k-1)\pi/2}{n} = \frac{2k-1}{4n}2\pi, \quad k = 1,\ldots,2n. \tag{7.16}$$

For such u, $\sin nu = \pm 1$, and we have

$$\pm j/\varepsilon = -j(\pm 1)\sinh nv.$$

We therefore take $v = \sinh^{-1}(1/\varepsilon)/n$. (In MATLAB use `asinh` for $\sinh^{-1}$.) Recalling that $u + jv = \cos^{-1}(-js/\omega_p)$, we have $\cos(u+jv) = -js/\omega_p$, or

$$\begin{aligned} s &= j\omega_p \cos(u+jv) \\ &= j\omega_p[\cos u \cosh v - j \sin u \sinh v] \\ &= \omega_p[\sin u \sinh v + j \cos u \cosh v]. \end{aligned}$$

The roots with negative real part occur for $u < 0$; i.e., when $k = n+1, \ldots, 2n$ in (7.16). We therefore take

$$\mathscr{H}(s) = \frac{H_0}{c_n(s - s_{n+1}) \cdots (s - s_{2n})}, \tag{7.17}$$

where c_n is chosen so that $\mathscr{H}(0) = H_0$ when n is odd and $\mathscr{H}(0) = H_0/\sqrt{1+\varepsilon^2}$ when n is even.

7.5. Chebyshev Filters of the Second Kind

Continuous-time **Chebyshev filters of the second kind**, or **Chebyshev-II filters** for short, are designed to have a frequency response satisfying

$$\begin{aligned} |\mathscr{H}(j\omega)|^2 &= H_0^2 - \frac{H_0^2}{1 + \varepsilon^2 T_n(\omega_s/\omega)^2} \\ &= \frac{H_0^2 \varepsilon^2 T_n(\omega_s/\omega)^2}{1 + \varepsilon^2 T_n(\omega_s/\omega)^2}. \end{aligned} \tag{7.18}$$

By multiplying the numerator and denominator by ω^{2n}, we can see that this frequency response is the quotient of polynomials of degree $2n$ in ω. Hence, the order of the filter is again n. Several Chebyshev-II frequency responses are shown in Figure 7.4. Notice that in the passband, the response decreases monotonically, while there is ripple in the stopband. In addition, for even n, the response does *not* decay to zero but levels off at a positive value. For odd n, the response does decay to zero. Finally, notice that the responses all have zeros in the stopband.

To make the attenuation in the stopband at least A_s, we need

$$\varepsilon \le \frac{1}{\sqrt{10^{A_s/10} - 1}}. \tag{7.19}$$

To make the attenuation at $\omega_p < \omega_s$ at most A_p, we need

$$n \ge \frac{\cosh^{-1}\left(\sqrt{(10^{A_s/10} - 1)/(10^{A_p/10} - 1)}\right)}{\cosh^{-1}(\omega_s/\omega_p)}, \tag{7.20}$$

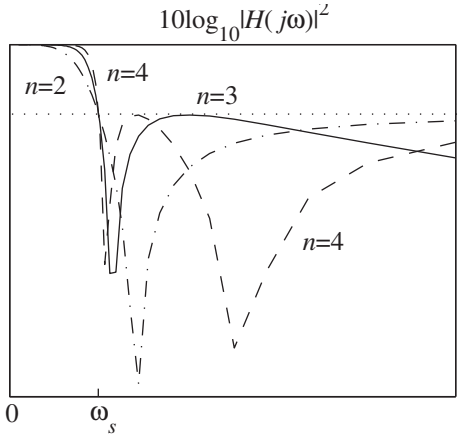


Figure 7.4. Frequency response (in dB) of Chebyshev-II filters for $n = 2,3,4$. The level of the dotted line is $10\log_{10}[H_0^2\varepsilon^2/(1+\varepsilon^2)]$.

which is the same condition as for the Chebyshev-I filters.

We now put $\omega = s/j$ in (7.18) and find the zeros and poles. Recall that T_n has n distinct roots given by (7.14). When n is even none of these roots is zero. If n is odd, the root with $k = (n+1)/2$ is zero. For a nonzero root, there is always a unique value of s such that $\omega_s/(-js)$ is equal to the root. Otherwise, there is no such s. Hence, the distinct zeros are

$$z_k = \frac{j\omega_s}{\cos\left([2k-1]\dfrac{\pi}{2n}\right)}, \quad k = 1,\ldots,n \text{ and } k \neq \frac{n+1}{2}. \tag{7.21}$$

Finding the poles is easy if we use our results for Chebyshev-I filters. Letting s_k^{I} denote the poles found in the analysis of the Chebyshev-I filters, we find that for the Chebyshev-II filters, the poles of $\mathscr{H}(s)\mathscr{H}(-s)$ are given by

$$s_k^{\mathrm{II}} = \frac{-\omega_p\omega_s}{s_k^{\mathrm{I}}}, \quad k = 1,\ldots,2n.$$

Observe that the minus sign interchanges the left and right half plane poles. Since $s_{n+1}^{\mathrm{I}},\ldots,s_{2n}^{\mathrm{I}}$ were the right half plane poles of the Chebyshev-I filter, the Chebyshev-II transfer function is

$$\mathscr{H}(s) = \frac{H_0 \displaystyle\prod_{\substack{k=1 \\ k\neq(n+1)/2}}^{n} (s - z_k)}{c_n^{\mathrm{II}}(s - s_1^{\mathrm{II}})\cdots(s - s_n^{\mathrm{II}})}, \tag{7.22}$$

where c_n^{II} is chosen so that $\mathscr{H}(0) = H_0$. For n even, the number of zeros and the number of poles is the same, which implies that the bilinear transformation does not add any zeros at $z = -1$. For n odd, the number of zeros is one less than the number of poles, and so the bilinear transformation adds a single zero at $z = -1$.

Problems

7.1. If $h(t)$ is a real-valued waveform, show that $\overline{\mathscr{H}(j\omega)} = \mathscr{H}(-j\omega)$.

7.2. Let $\mathscr{H}(s)$ be a continuous-time transfer function. Show that if $\mu_k = j\omega_k$ is an imaginary zero of $\mathscr{H}(s)$, then after the bilinear transformation, $\widehat{\mu}_k$ in (7.4) lies on the unit circle.

7.3. In the study of analog filters, the parameter

$$d := \sqrt{\left(10^{A_p/10} - 1\right) \Big/ \left(10^{A_s/10} - 1\right)}$$

is called the **discrimination factor**. For an ideal filter $A_s = \infty$, and so3.2em $d = 0$. For practical filters, $A_s < \infty$, and so $d > 0$. The parameter $k := \omega_p/\omega_s$ is called the **selectivity factor**. For an ideal filter, the transition band would have zero width, meaning $\omega_p = \omega_s$, which corresponds to a selectivity factor of one. For practical filters, $k < 1$. Show that (7.9) is equivalent to

$$n \geq \frac{\ln(1/d)}{\ln(1/k)} = \frac{\ln d}{\ln k}.$$

7.4. Show that the Butterworth filter poles s_k in (7.11) occur in conjugate pairs; specifically, show that

$$s_{n-k+1} = \overline{s_k}, \quad \text{for } k = 1, \ldots, n.$$

Hint: It may be helpful to use the fact that $3n = 4n - n$.

7.5. For the Butterworth poles s_k in (7.11), show that $\mathscr{B}_n := \prod_{k=1}^{n}(-s_k) = (-\omega_c)^n$.

7.6. Consider the polynomial

$$(s - r)(s - \overline{r}) = s^2 + \beta_1 s + \beta_2.$$

Show that β_1 and β_2 are real numbers.

7.7. With d and k being the discrimination and selectivity factors defined in Problem 7.3, show that for Chebyshev filters, (7.15) and (7.20) are equivalent to

$$n \geq \frac{\cosh^{-1}(1/d)}{\cosh^{-1}(1/k)}.$$

7.8. Use the following procedure to show that for $x \geq 1$, $\widehat{T}_n(x) := \cosh(n \cosh^{-1} x) = T_n(x)$.

(a) Recall that $\cosh x := (e^x + e^{-x})/2$ and $\sinh x := (e^x - e^{-x})/2$. Verify that

$$\cosh x \cosh y \pm \sinh x \sinh y = \cosh(x \pm y).$$

(b) It is obvious that $\widehat{T}_0(x) = 1$, and $\widehat{T}_1(x) = x$. Show that

$$\widehat{T}_{n+1}(x) = 2x\widehat{T}_n(x) - \widehat{T}_{n-1}(x).$$

7.9. Recall that for any complex number z, $\cos z := (e^{jz} + e^{-jz})/2$ and $\sin z := (e^{jz} - e^{-jz})/(2j)$. Verify that for real numbers α and β,

$$\cos\alpha \cosh\beta \mp j \sin\alpha \sinh\beta = \cos(\alpha \pm j\beta).$$

7.10. Show that for complex z, $\sin^{-1} z = -j \log(\sqrt{1-z^2} + jz)$. *Hints:* If $\theta = \sin^{-1} z$, then θ solves the equation $\sin\theta = z$. Into this equation, substitute $\sin\theta = (e^{j\theta} - e^{-j\theta})/(2j)$. Multiply the result by $e^{j\theta}$ to get a quadratic in $e^{j\theta}$. Apply the quadratic formula to solve for $e^{j\theta}$. Then take logarithms to solve for θ.

7.11. Use the approach of the preceding problem to derive the formula $\cos^{-1} z = -j \log(j\sqrt{1-z^2} + z)$. Then use this formula to show that if z is a real number with $|z| \geq 1$, then

$$\cos(n \cos^{-1} z) = \left[\left(\sqrt{z^2-1} + z\right)^n + \left(\sqrt{z^2-1} + z\right)^{-n}\right]/2.$$

7.12. Use the identity $\cos(\pi/2 - \theta) = \sin\theta$ to derive the formula $\cos^{-1} z = \pi/2 - \sin^{-1} z$. Then use the result of Problem 7.10 to show that $\cos^{-1} z = \pi/2 + j \log(\sqrt{1-z^2} + jz)$.

7.13. Show that the roots of $1 + \varepsilon^2 T_n(-js/\omega_p)^2$ lie on an ellipse with foci at $\pm j$. *Hints:* For $0 \leq u \leq 2\pi$, let $z := \omega_p(\sin u \sinh v + j \cos u \cosh v)$ and show that $|z - j| + |z - (-j)| =$ constant. Identify the constant. Use the following facts: $\cosh^2 v - \sinh^2 v = 1$; $v > 0$; and $\cosh v \geq 1$.

7.14. Show that the poles of the Chebyshev-I filter in (7.17) occur in conjugate pairs.

7.15. Show that the zeros of the Chebyshev-II filter in (7.21) satisfy $z_{n+1-k} = -z_k$.

7.16. **MATLAB.** Adapt the MATLAB script from Section 7.3 for Chebyshev-I filters.

7.17. Derive conditions (7.19) and (7.20) for Chebyshev-II filters.

7.18. Derive formulas (7.21) and (7.22) for Chebyshev-II filters.

7.19. **MATLAB.** Adapt the MATLAB script from Section 7.3 for Chebyshev-II filters.

7.20. Consider the problem of designing a discrete-time filter with passband edge $f_p = 0.2$ and stopband edge $f_s = 0.3$. The maximum passband attenuation is $A_p = 2$ dB and the minimum stopband attenuation is $A_s = 40$ dB. If you design your filter by applying the bilinear transformation to a Butterworth or Chebyshev filter, what is the minimum order n possible in each case?

7.21. You are to implement a continuous-time lowpass filter using discrete-time signal processing. The design constraints on the continuous-time filter are as follows. The passband attenuation should be no more than 1.5 dB and the passband edge should be 3 kHz. The stopband attenuation should be at least 40 dB and the stopband edge should be 4 kHz. You may assume that all incoming signals are bandlimited to 6 kHz.

(a) If the passband cannot have any ripple, which kind of IIR filter(s) (Butterworth, Chebyshev-I, Chebyshev-II) could you use?

(b) If the filter order cannot exceed 8, which kind of IIR filter(s) (Butterworth, Chebyshev-I, Chebyshev-II) could you use?

CHAPTER 8
FIR Filters

8.1. Motivation

Consider the signal

$$x_n := c_1 e^{j2\pi f_1 n} + c_2 e^{j2\pi f_2 n} + c_3 e^{j2\pi f_3 n},$$

where $0 < f_1 < f_2 < f_3 < 1/2$. Its DTFT is ***[DRAW PICTURE]***

$$X(f) = c_1\delta(f-f_1) + c_2\delta(f-f_2) + c_3\delta(f-f_3), \quad |f| \le 1/2.$$

Suppose we want to extract the third term of x_n, $c_3 e^{j2\pi f_3 n}$, by using a lowpass filter with cutoff frequency f_c somewhere between f_2 and f_3, say $H(f) = I_{[-f_c,f_c]}(f)$ ***[DRAW PICTURE]***. If x_n is applied to this filter, then the output in the frequency domain is

$$X^{1,2}(f) := H(f)X(f) = c_1\delta(f-f_1) + c_2\delta(f-f_2),$$

which corresponds to the time-domain signal

$$x_n^{1,2} = c_1 e^{j2\pi f_1 n} + c_2 e^{j2\pi f_2 n}.$$

In this case, the difference $x_n - x_n^{1,2} = c_3 e^{j2\pi f_3 n}$ is the desired signal.

Suppose instead we use a different filter as follows. As before, let f_c lie between f_2 and f_3, but now let $f_{c'}$ lie between f_1 and f_2. Let us apply x_n to the filter ***[DRAW PICTURE]***

$$G(f) = I_{[-f_{c'},f_{c'}]}(f) + I_{(f_{c'},f_c]}(|f|)e^{-j2\pi f n_0}.$$

Like the first filter, this one also has the property $|G(f)| = I_{[-f_c,f_c]}(f)$. However, now the filter output in the frequency domain is

$$Y(f) := G(f)X(f) = c_1\delta(f-f_1) + c_2\delta(f-f_2)e^{-j2\pi f_2 n_0},$$

which corresponds to the time-domain signal

$$\begin{aligned} y_n &= c_1 e^{j2\pi f_1 n} + c_2 e^{j2\pi f_2 n} e^{-j2\pi f_2 n_0} \\ &= c_1 e^{j2\pi f_1 n} + c_2 e^{j2\pi f_2 (n-n_0)}. \end{aligned}$$

Hence, the difference

$$\begin{aligned} x_n - y_n &= c_2 e^{j2\pi f_2 n} + c_3 e^{j2\pi f_3 n} - c_2 e^{j2\pi f_2 (n-n_0)} \\ &= c_2 e^{j2\pi f_2 n}(1 - e^{-j2\pi f_2 n_0}) + c_3 e^{j2\pi f_3 n}. \end{aligned}$$

We see that the second term is not canceled by this procedure.

To conclude the motivation, consider using the filter

$$\begin{aligned} L(f) &= I_{[-f_{c'},f_{c'}]}(f)e^{-j2\pi f n_0} + I_{(f_{c'},f_c]}(|f|)e^{-j2\pi f n_0} \\ &= \left\{I_{[-f_{c'},f_{c'}]}(f) + I_{(f_{c'},f_c]}(|f|)\right\}e^{-j2\pi f n_0}, \end{aligned}$$

which again has the property $|L(f)| = I_{[-f_c,f_c]}(f)$. If we apply x_n to this filter, the output in the frequency domain is

$$V(f) := L(f)X(f) = c_1\delta(f-f_1)e^{-j2\pi f_1 n_0} + c_2\delta(f-f_2)e^{-j2\pi f_2 n_0},$$

and the corresponding time-domain signal is

$$v_n = c_1 e^{j2\pi f_1(n-n_0)} + c_2 e^{j2\pi f_2(n-n_0)}.$$

We see that the difference $x_{n-n_0} - v_n = c_3 e^{j2\pi f_3(n-n_0)}$, which is a delayed version of the desired third term of x_n.

The filter $L(f)$ is an example of a filter property called **linear phase**. Linear-phase filters delay all frequency components by the same amount of time. As we shall see, all linear-phase filters that are causal must be FIR.

8.2. Linear-Phase Filters

Recall that every nonzero complex number z can be written in the form $z = re^{j\theta}$, where r is the magnitude of z and θ is the **angle** or **phase** of z. In MATLAB, the angle of a complex number `z` is returned by the command `angle(z)`. This command always returns the **principal angle**, which is defined to lie in the range $-\pi < \theta \leq \pi$.

A filter $H(f)$ of the form

$$H(f) = |H(f)|e^{-j2\pi\tau f}$$

for some real constant τ is said to have **linear phase** because its phase is a linear function of f.[1] The constant τ is called the **group delay**. For example, the DTFT of the rectangular window, $W_{\text{rect}}(f)$ and the DTFT of the Bartlett window $W_{\text{Bartlett}}(f)$ defined in Chapter 4 are obviously linear phase with group delay $\tau = (N-1)/2$. The DTFTs of the other windows are also linear phase, but this is harder to see.

A filter is said to have to have ***generalized*** **linear phase** (GLP) if it has the form

$$H(f) = A(f)e^{-j(2\pi\tau f - \varphi_0)}, \tag{8.1}$$

[1] If we use the MATLAB function `angle` to plot the phase of $H(f)$ as a function of f, then the plot will be linear only for small f. As $|f|$ increases, the quantity $2\pi\tau f$ will eventually exceed $\pm\pi$ and the plot will exhibit jump discontinuities.

where A is a real-valued function (not necessarily nonnegative), τ is a real constant, and φ_0 is a real constant in the range $(-\pi, \pi)$. (Allowing $\varphi_0 = \pm\pi$ would be the same as replacing $A(f)$ with $-A(f)$ and φ_0 with zero.) If we know τ and φ_0, it is easy to plot $A(f) = H(f)e^{j(2\pi\tau f - \varphi_0)}$. In MATLAB, the formula on the right may not be pure real due to round off, and so it is best to take the real part. When $A(f)$ is positive, the phase of $H(f)$ and the phase of $e^{-j(2\pi\tau f-\varphi_0)}$ are the same; otherwise, they differ by $\pm\pi$. In particular, we shall see cases in which $A(f)$ is odd, which means that $A(f)$ will undergo a sign change as f passes through zero. In such cases there will be a jump discontinuity in the phase at $f = 0$.

8.2.1. GLP Implies 2τ Must Be an Integer

Recall that DTFTs are periodic with period one. If a filter is GLP, then $H(f+1) = H(f)$ says that

$$A(f+1)e^{-j(2\pi\tau[f+1]-\varphi_0)} = A(f)e^{-j(2\pi\tau f-\varphi_0)}.$$

Canceling common factors, we see that

$$A(f) = A(f+1)e^{-j2\pi\tau}. \tag{8.2}$$

Since A is real valued, 2τ must be an integer. If 2τ is an even integer, then (8.2) says that $A(f) = A(f+1)$; i.e., A has period one. If 2τ is an odd integer, then (8.2) says that $A(f) = -A(f+1)$. Since this is true for all f, replacing f with $f+1$ yields $A(f+1) = -A(f+2)$, and we see that $A(f) = -[-A(f+2)] = A(f+2)$; i.e., A has period two.

8.2.2. GLP Is Equivalent to Generalized Symmetry

For a GLP filter, the impulse response is

$$\begin{aligned} h_n &= \int_{-1/2}^{1/2} A(f)e^{-j(2\pi\tau f-\varphi_0)}e^{j2\pi fn}\,df \\ &= \int_{-1/2}^{1/2} e^{j\varphi_0}A(f)e^{j2\pi f(n-\tau)}\,df. \end{aligned}$$

Hence,

$$\begin{aligned} h_{2\tau-n} &= \int_{-1/2}^{1/2} e^{j\varphi_0}A(f)e^{j2\pi f([2\tau-n]-\tau)}\,df \\ &= \int_{-1/2}^{1/2} e^{j\varphi_0}A(f)e^{j2\pi f(\tau-n)}\,df. \end{aligned}$$

Since $A(f)$ is real,

$$\begin{aligned} h_{2\tau-n} &= e^{j2\varphi_0} \overline{\int_{-1/2}^{1/2} e^{j\varphi_0} A(f) e^{j2\pi f(n-\tau)}\, df} \\ &= e^{j2\varphi_0} \overline{h_n}. \end{aligned}$$

We can rewrite this as the **generalized symmetry condition**

$$h_n = e^{j2\varphi_0} \overline{h_{2\tau-n}}. \tag{8.3}$$

If τ is an integer, we can replace n with $\tau+n$ to get

$$h_{\tau+n} = e^{j2\varphi_0} \overline{h_{\tau-n}},$$

or

$$h_{\tau+n} e^{-j\varphi_0} = \overline{h_{\tau-n} e^{-j\varphi_0}}. \tag{8.4}$$

In other words $g_n := h_{\tau+n} e^{-j\varphi_0}$ is conjugate symmetric about $n=0$ (satisfies $g_n = \overline{g_{-n}}$). If τ is a half integer, we can replace n with $\tau+1/2+n$ in (8.3) to get

$$h_{\tau+1/2+n} = e^{j2\varphi_0} \overline{h_{\tau-1/2-n}},$$

or

$$h_{\tau+1/2+n} e^{-j\varphi_0} = \overline{h_{\tau-1/2-n} e^{-j\varphi_0}}, \tag{8.5}$$

which says that $g_n := h_{\tau+1/2+n} e^{-j\varphi_0}$ satisfies $g_n = \overline{g_{-n-1}}$.

We have shown that GLP (8.1) implies the generalized symmetry condition (8.3). The converse is also true. If h_n satisfies (8.4) or (8.5), then the DTFT $H(f)$ is GLP (satisfies (8.1)). This is left to the problems. The importance of this equivalence is that it gives us two ways of checking for linear phase. If we know the DTFT, we can see if it has the form (8.1). But if we do not have a formula for the DTFT, we can check if the impulse response satisfies the generalized symmetry condition. For example, the DTFT of the Kaiser window has no closed-form expression, but it is easy to see that the window function w_n satisfies the generalized symmetry condition.

8.2.3. GLP and Causality Imply FIR

If h_n is a causal sequence and $n > 2\tau$, then $h_{2\tau-n} = 0$. If $H(f)$ is GLP, it follows immediately from the generalized symmetry condition (8.3) that h_n is FIR with $h_n = 0$ for $n < 0$ and $n > 2\tau$. Hence, causal GLP filters have order of at most 2τ.[2] In this

[2] Recall that the order of an FIR filter is one less than its length.

case, it is convenient to list the times n at which h_n may be nonzero as follows. When 2τ is even,

$$n = \underbrace{\overbrace{0,1,\ldots,\tau-1}^{\tau \text{ values}},\tau,\overbrace{\tau+1,\ldots,2\tau}^{\tau \text{ values}}}_{2\tau+1 \text{ values}}$$

and when 2τ is odd,

$$n = \underbrace{\overbrace{0,1,\ldots,\tau-1/2}^{(\tau+1/2) \text{ values}},\overbrace{\tau+1/2,\ldots,2\tau}^{(\tau+1/2) \text{ values}}}_{2\tau+1 \text{ values}}$$

Since $\lfloor\tau\rfloor$ is equal to τ when 2τ is even and $\tau-1/2$ when 2τ is odd, once arbitrary values of h_n have been specified for $n = 0,\ldots,\lfloor\tau\rfloor$, then for $n = \lfloor\tau\rfloor+1,\ldots,2\tau$, the values of h_n must be $h_n = e^{j2\varphi_0}\overline{h_{2\tau-n}}$. The only "catch" is that when 2τ is even $h_\tau e^{-j\varphi_0}$ must be real. If this is not already true, h_τ can be replaced by $e^{j\varphi_0}\,\mathrm{Re}(h_\tau e^{-j\varphi_0})$.

[Run filtertypes.m demo]

8.2.4. GLP and Real Impulse Response Imply φ_0 Is 0 or $\pi/2$

If h_n is real, then $H(-f) = \overline{H(f)}$. For a GLP filter, this implies, since A is real,

$$A(-f)e^{-j(2\pi\tau(-f)-\varphi_0)} = A(f)e^{j(2\pi\tau f-\varphi_0)},$$

from which we see that

$$e^{j2\varphi_0} = \frac{A(f)}{A(-f)} = \text{real}. \tag{8.6}$$

Hence, $2\varphi_0$ must be a multiple of π. Since $\varphi_0 \in (-\pi,\pi)$, φ_0 must be either zero or $\pi/2$. We further point out that when $\varphi_0 = 0$ in (8.6), it follows that $A(f) = A(-f)$; i.e., A is an even function. When $\varphi_0 = \pi/2$, it follows that $A(f) = -A(-f)$; i.e., A is an odd function.

8.2.5. Symmetry Conditions for GLP and Real Impulse Response

If h_n is real, and $H(f)$ is GLP, then (8.3) implies the following **symmetry conditions**. First, since h_n is real, we can drop the complex conjugate. Second, also because h_n is real, we can restrict $2\varphi_0$ to be zero or π. Hence, (8.3) becomes

$$h_n = \pm h_{2\tau-n}, \tag{8.7}$$

where the plus sign (symmetry) corresponds to $\varphi_0 = 0$ and the minus sign (antisymmetry) corresponds to $\varphi_0 = \pi/2$. Recalling that $A(f)$ is even (symmetric about $f = 0$)

for $\varphi_0 = 0$ and odd (antisymmetric about $f = 0$) for $\varphi_0 = \pi/2$, we see that h_n and $A(f)$ are both symmetric or both antisymmetric.

Because 2τ can be an even or an odd integer and φ_0 can be zero or $\pi/2$, there are four possible kinds of GLP filters with real impulse response.

	$\varphi_0 = 0$ symmetric	$\varphi_0 = \pi/2$ antisymmetric
2τ even, $A(f+1) = A(f)$	I	III
2τ odd, $A(f+1) = -A(f)$	II	IV

Table 8.1. Types of GLP filters with real impulse response.

Symmetry Implications

First consider an antisymmetric filter, either type-III or type-IV. Then since $A(f)$ is odd, $A(0) = 0$, which implies $H(0) = 0$. Hence, antisymmetric filters cannot be lowpass filters.

Next consider a type-II filter. Since $A(f+1) = -A(f)$, taking $f = 1/2$ yields $A(1/2) = -A(-1/2)$. However, since $A(f)$ is also even, $A(-1/2) = A(1/2)$. Putting these two equations together implies $A(1/2) = -A(1/2)$, which implies $A(1/2) = 0$. Again using the fact that A is even, we have $A(-1/2) = 0$ as well. Thus, $H(\pm 1/2) = 0$. This means that a type-II filter cannot be highpass.

A similar argument shows that a type-III filter cannot be highpass.

***Example* 8.2.1** (Differentiator of Type IV). Recall that the transfer function of an analog differentiator is $H(f) = j2\pi f$. Suppose, however, that we agree to apply it only to waveforms bandlimited to f_c. Then we can use the system $H(f) = j2\pi f I_{[-f_c,f_c]}(f)$ instead. To implement this system using discrete-time signal processing with sampling rate $f_s = 2f_c$, we require a discrete-time filter with DTFT equal to $H(f_s f)$ for $|f| \le 1/2$. In other words, $H(f_s f) = j2\pi f_s f$ for $|f| \le 1/2$. Since $j = e^{j\varphi_0}$ with $\varphi_0 = \pi/2$, our filter is GLP of type III or IV. If we also impose a group delay of τ, our discrete-time frequency response is $j2\pi f_s f e^{-j2\pi\tau f}$ for $|f| \le 1/2$. Noting that the absolute value of this function is proportional to $|f|$, we see that a differentiator is a kind of HPF. Hence, a type-III filter is not appropriate. We therefore consider only a type-IV filter with 2τ being odd. A careful calculation shows that the impulse response sequence is

$$h_n = f_s(-1)^{n-\tau+1/2}/[\pi(n-\tau)^2].$$

Using only h_n for $n = 0, \ldots, 2\tau$, we obtain an GLP, causal, FIR filter.

[Run diffscript.m]

8.3. Windowing of Impulse Responses of GLP Filters

Suppose we have a GLP filter with group delay τ and phase φ_0 whose impulse response h_n is not FIR. Then as we have seen, it cannot be causal. However, suppose we multiply it by one of our windows from Chapter 4 of length $2\tau+1$. Using the formulas for w_n (which are real), it is easy to check that they satisfy the symmetry property (8.7) and are therefore GLP with zero phase and group delay τ. Since these windows are causal and FIR, the product $w_n h_n$ is causal and FIR. Furthermore, by Problem 8.6 the GLP property is preserved with the same group delay and the same phase as $H(f)$.

***Example* 8.3.1** (LPFs of Types I and II). Let $0 < f_c < 1/2$. The filter defined by $H(f) = e^{-j2\pi\tau f} I_{[-f_c,f_c]}(f)$ for $|f| \le 1/2$ is an ideal lowpass filter that is linear phase if 2τ is an integer. The impulse response $h_n = 2f_c \operatorname{sinc}(2f_c[n-\tau])$. If we multiply h_n by a window of length $2\tau+1$, we get an FIR filter of order 2τ.

***Example* 8.3.2** (BPFs of Types I and II). Let $0 < f_1 < f_2 \le 1/2$. Then the filter $H(f) = e^{-j2\pi\tau f} I_{[f_1,f_2]}(|f|)$ for $|f| \le 1/2$ is an ideal bandpass filter (BPF) with linear phase if 2τ is an integer. The impulse response is

$$h_n = 2f_2 \operatorname{sinc}(2f_2[n-\tau]) - 2f_1 \operatorname{sinc}(2f_1[n-\tau]).$$

If we multiply h_n by a window of length $2\tau+1$, we get an FIR filter of order 2τ. Since $f_2 = 1/2$ is a highpass filter (HPF), we cannot use a type-II filter (2τ odd); i.e., we cannot use a window of even length in this case.

[Run modscriptAM.m demo]

The Hilbert Transform

The **Hilbert transform** of a waveform $x(t)$ is denoted by $\hat{x}(t)$. We show below that the signals

$$x(t)\cos(2\pi f_0 t) \pm \hat{x}(t)\sin(2\pi f_0 t)$$

are single sideband. Such signals are very important in communication systems. Our interest here is in approximating $\hat{x}(t)$ using a discrete-time signal processing system with an FIR filter.

The signal $\hat{x}(t)$ is defined to be the output of the linear time-invariant system with transfer function $H(f) := -j\,\mathrm{sgn}(f)$, where $\mathrm{sgn}(f)$ is the **signum function** or **sign function** defined by

$$\mathrm{sgn}(f) := \begin{cases} 1, & f > 0, \\ 0, & f = 0, \\ -1, & f < 0. \end{cases}$$

If we let $\hat{X}(f)$ denote the Fourier transform of $\hat{x}(t)$, then we have $\hat{X}(f) = H(f)X(f) = -j\,\mathrm{sgn}(f)X(f)$.

To see what the Hilbert transform does, it is convenient to write $X(f)$ in the form $L(f)+R(f)$, where $L(f)$ is the left half of $X(f)$ and $R(f)$ is the right half; more precisely, $L(f) := X(f)$ for $f < 0$ and $R(f) := X(f)$ for $f \geq 0$, and $L(f)$ and $R(f)$ are zero otherwise. See Figure 8.1.

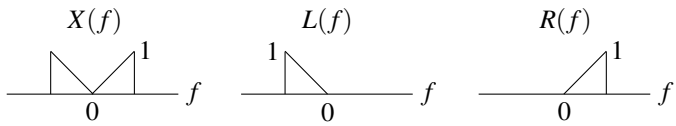


Figure 8.1. A continuous-time Fourier transform $X(f)$ and its left and right halves $L(f)$ and $R(f)$.

Since

$$X(f) = L(f)+R(f) = \begin{cases} R(f), & f \geq 0, \\ L(f), & f < 0, \end{cases}$$

we see that the Hilbert transform of $x(t)$ is, in the frequency domain,

$$\hat{X}(f) := -j\,\mathrm{sgn}(f)X(f) = \begin{cases} -jR(f), & f \geq 0, \\ jL(f), & f < 0, \end{cases}$$

which is shown in Figure 8.2. Note that since $L(f) = 0$ for $f \geq 0$ and $R(f) = 0$ for $f < 0$, we can also write $\hat{X}(f) = -j[R(f) - L(f)]$.

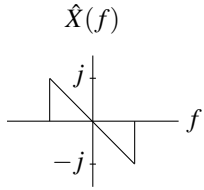


Figure 8.2. The Fourier transform of $\hat{x}(t)$.

Now recall that for any signal $y(t)$, the Fourier transform of $y(t)\sin(2\pi f_0 t)$ is

$$\tfrac{1}{2j}[Y(f-f_0) - Y(f+f_0)].$$

Replacing $y(t)$ with $\hat{x}(t)$ and using the fact that $\hat{X}(f) = -j[R(f) - L(f)]$, we find that the Fourier transform of $\hat{x}(t)\sin(2\pi f_0 t)$ is

$$\tfrac{1}{2j}\left[\{-j[R(f-f_0) - L(f-f_0)]\} - \{-j[R(f+f_0) - L(f+f_0)]\}\right],$$

which simplifies to

$$\tfrac{1}{2}[L(f-f_0)-R(f-f_0)+R(f+f_0)-L(f+f_0)] \tag{8.8}$$

and is shown in Figure 8.3.

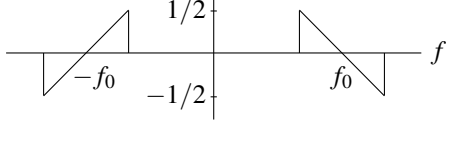
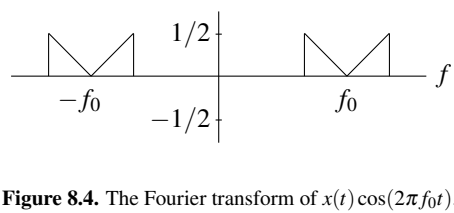


Figure 8.3. The Fourier transform of $\hat{x}(t)\sin(2\pi f_0 t)$.

It is interesting to compare this with the Fourier transform of $x(t)\cos(2\pi f_0 t)$, which is $\frac{1}{2}[X(f-f_0)+X(f+f_0)]$, or in more detail,

$$\tfrac{1}{2}\big[\{L(f-f_0)+R(f-f_0)\}+\{L(f+f_0)+R(f+f_0)\}\big], \tag{8.9}$$

and is shown in Figure 8.4. We see from (8.8) and (8.9) that the Fourier transform of



$\frac{1}{2}[X(f-f_0)+X(f+f_0)]$

Figure 8.4. The Fourier transform of $x(t)\cos(2\pi f_0 t)$.

$$x(t)\cos(2\pi f_0 t)-\hat{x}(t)\sin(2\pi f_0 t)$$

is $L(f+f_0)+R(f-f_0)$, which is shown in Figure 8.5. Similarly, the Hilbert transform of

$$x(t)\cos(2\pi f_0 t)+\hat{x}(t)\sin(2\pi f_0 t)$$

is $R(f+f_0)+L(f-f_0)$.

***Example* 8.3.3** (Hilbert Transform of Type IV). The analog Hilbert transform has transfer function $-j\,\mathrm{sgn}(f)$. However, suppose that we agree to apply it only to waveforms bandlimited to f_c. Then we can use the system $H(f)=-j\,\mathrm{sgn}(f)I_{[-f_c,f_c]}(f)$ instead. To implement this system using discrete-time signal processing with sampling rate $f_s=2f_c$, we require a discrete-time filter with DTFT equal to $H(f_s f)$ for

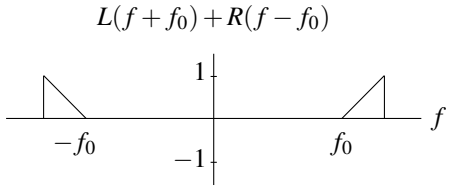


Figure 8.5. The Fourier transform of $x(t)\cos(2\pi f_0 t) - \hat{x}(t)\sin(2\pi f_0 t)$.

$|f| \le 1/2$. In other words, $H(f_s f) = -j\,\mathrm{sgn}(f)$ for $|f| \le 1/2$. Since $j = e^{j\varphi_0}$ for $\varphi_0 = \pi/2$, our filter is GLP of type III or IV. If we also impose a group delay of τ, our discrete-time frequency response is $-j\,\mathrm{sgn}(f)e^{-j2\pi\tau f}$ for $|f| \le 1/2$. Noting that the absolute value of this function is 1 for $f \neq 0$, we see that a Hilbert transform is a kind of HPF. Hence, a type-III filter is not appropriate. We therefore consider only a type-IV filter with 2τ being odd. A careful calculation shows that the impulse response sequence is

$$h_n = \frac{1-\cos(\pi[n-\tau])}{\pi(n-\tau)}.$$

Since our DTFT has a jump discontinuity at $f = 0$, we will have the Gibbs phenomenon. We therefore multiply h_n by a window of length $2\tau+1$ such as a Kaiser window.

[Run hilbertscript.m demo]

8.3.1. Filter Specifications

Consider a lowpass DTFT with $|H(f)|$ like the one shown in Figure 8.6. The parameters in Figure 8.6 are related to the parameters in Figure 7.1 via

$$A_p = 20\log_{10}\frac{1+\delta^+}{1-\delta^-} \quad \text{and} \quad A_s = 20\log_{10}\frac{1+\delta^+}{\delta_s}.$$

When specifying IIR filters, we typically take $\delta^+ = 0$. When specifying FIR filters, we typically take $\delta^+ = \delta^-$ and denote this common value by δ_p.

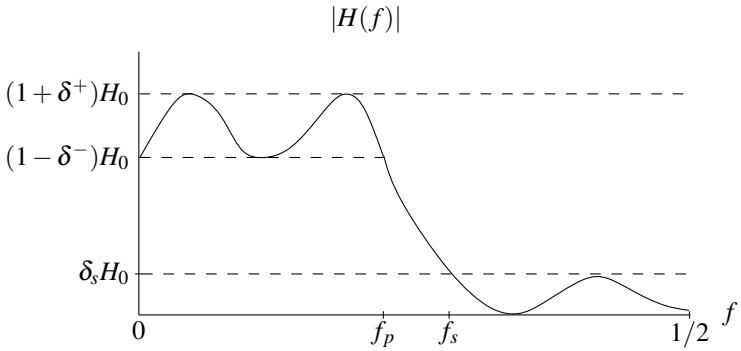


Figure 8.6. Parameters for specifying discrete-time filters.

8.3.2. The Kaiser Window

The preferred window to use is the Kaiser window. In terms of the notation of this chapter, the Kaiser window is given by

$$w_n = \frac{I_0\left(\beta\sqrt{1-\left(\frac{n-\tau}{\tau}\right)^2}\right)}{I_0(\beta)}, \quad n = 0,\ldots,2\tau,$$

where the shape parameter β and the order 2τ are initially selected as follows. Let

$$A := -20\log_{10}\big(\min\{\delta_p,\delta_s\}\big).$$

Then we try setting the integer order

$$2\tau \geq \frac{A-7.95}{2.285(2\pi|f_s-f_p|)},$$

and the shape parameter as

$$\beta = \begin{cases} 0.1102(A-8.7), & A > 50, \\ 0.5842(A-21)^{0.4}+0.07886(A-21), & 21 < A \leq 50, \\ 0, & A \leq 21. \end{cases}$$

Note that $\beta = 0$ reduces the the rectangular window. Kaiser developed these formulas empirically, and they serve as a starting point. The DTFT of the windowed ideal impulse response must be plotted to check if it meets the specifications. If not, it may be necessary to increase N or β or both.

The formula for the filter order depends on the width of the transition band, $|f_s - f_p|$ of the DTFT. This works generally for multiband filters; just use the smallest transition band.

Once you have chosen 2τ and β, the window $\mathtt{w} = [w_0, \ldots, w_{2\tau}]'$ can be obtained with the MATLAB Signal Processing Toolbox command `w=kaiser(twotau+1, beta)`.

[Run modscriptAM.m]

8.4. Equiripple Filters and the Parks–McClellan Algorithm

For ease of exposition, consider a type-I filter with $H(f) = A(f)e^{-j2\pi\tau f}$, where 2τ is even. For a causal filter,

$$H(f) = \sum_{n=0}^{2\tau} h_n e^{-j2\pi fn},$$

or

$$\begin{aligned} A(f) &= \sum_{n=0}^{2\tau} h_n e^{-j2\pi f(n-\tau)} \\ &= \sum_{m=-\tau}^{\tau} \underbrace{h_{m+\tau}}_{=:a_m} e^{-j2\pi fm}. \end{aligned}$$

Since $A(f)$ is real and even, the a_m are real and even, and we can write

$$A(f) = \sum_{m=0}^{\tau} g_m \cos(2\pi fm), \tag{8.10}$$

where $g_0 = a_0$ and $g_m = 2a_m$ for $m = 1, \ldots, \tau$.

Suppose we want to design a lowpass filter with passband edge f_p and stopband edge f_s. Then we want to find $g_0, \ldots, g_m$ that achieve

$$\min_{g_0,\cdots,g_\tau} \max_{0 \le f \le f_p,\, f_s \le f \le 1/2} |A(f) - A_d(f)|,$$

where $A_d(f)$ is the ideal amplitude response, $A_d(f) := 1$ for $|f| \le f_p$ and $A_d(f) := 0$ for $f_s \le |f| \le 1/2$. Note that $A_d(f)$ is not specified outside of the passband and stopband. Also, because $A(f)$ and $A_d(f)$ are even, it suffices to maximize over non-negative f.

The first step in solving this problem is to make the substitution $2\pi f = \cos^{-1}(x)$ in (8.10). This is equivalent to $x = \cos(2\pi f)$. Thus, our optimization problem becomes

$$\min_{g_0,\ldots,g_\tau} \max_{x\in X} \left| \sum_{m=0}^{\tau} g_m \underbrace{\cos(m\cos^{-1}(x))}_{=T_m(x)} - \widetilde{A}_d(x) \right|,$$

where $\widetilde{A}_d(x) := A_d(\cos^{-1}(x)/(2\pi))$ and $X = \cos(2\pi\{[0, f_p] \cup [f_s, 1/2]\})$, and T_m is the mth Chebyshev polynomial. Since $\sum_{m=0}^{\tau} g_m T_m(x)$ is just a polynomial of degree τ, we see that the optimal filter-design problem is really a polynomial approximation problem in disguise! The classical method for the polynomial approximation problem is due to Remez (or Remes) and is called the **Remez Exchange Algorithm**. It was adapted to FIR filter design by Parks and McClellan [6] and is known as the **Parks–McClellan algorithm**. A retrospective discussion of the development of the Parks–McClellan algorithm recently appeared in [4].

With more work it can be shown that optimization of GLP filters of types II–IV can be put into the same framework [7]. The key MATLAB functions are `firpmord` for computing 2τ and `firpm` for generating the filter coefficients h_n.

8.4.1. Alternation and Exchange

We sketch some of the basic ideas behind polynomial approximation.

Alternation Theorem. *A polynomial of degree n is the best approximation (in the sense of uniform error) of a function on a bounded subset of the real line if and only if there exist $n+2$ points in the subset such that the worst-case error on the subset occurs at each of these points and the error at adjacent points alternates in sign.* ***[Draw a picture of the error*** $f(x) - p(x)$***.]***

The proof of necessity is beyond our scope here, but can be found, for example, in [8, pp. 26–27]. However, it is not too hard to prove sufficiency, which we do a bit later below.

Application to a Set of n+2 Points

Suppose we have any function f defined on the finite set X containing *exactly* $n+2$ points, say $x_1 < \cdots < x_{n+2}$, and we want to approximate f on this set by a polynomial p of degree at most n. We do this by taking p to be the difference of two polynomials of degree $n+1$ such that their powers of x^{n+1} cancel and so that the error $f(x) - p(x)$ satisfies the conditions of the Alternation Theorem. We will then know that $\max_{x\in X} |f(x) - p(x)|$ is the minimum possible among all polynomials of degree at most n.

Let g denote the unique polynomial of degree at most $n+1$ such that $g(x_k) = f(x_k)$ for $k = 1, \ldots, n+2$. ***[Draw a generic picture.]*** Similarly, let s denote the

unique polynomial of degree at most n such that $s(x_k) = (-1)^k$ for $k = 1, \ldots, n+2$. ***[Draw a picture.]*** Since s changes sign $n+1$ times, it has $n+1$ roots, say $r_1 < \cdots < r_{n+1}$, and

$$s(x) = c(x - r_1) \cdots (x - r_{n+1})$$

for some nonzero constant c. Hence, the degree of s is exactly $n+1$. If $g(x) = g_{n+1}x^{n+1} + \cdots$, we put

$$\begin{aligned} p(x) &:= g(x) - \frac{g_{n+1}}{c} s(x) \\ &= \{g_{n+1}x^{n+1} + \cdots\} - \frac{g_{n+1}}{c}\{cx^{n+1} + \cdots\}, \end{aligned}$$

which is a polynomial of degree at most n. Then for $k = 1, \ldots, n+2$,

$$\begin{aligned} f(x_k) - p(x_k) &= f(x_k) - \left[g(x_k) - \frac{g_{n+1}}{c} s(x_k)\right] \\ &= f(x_k) - f(x_k) + (-1)^k \frac{g_{n+1}}{c} \\ &= (-1)^k \frac{g_{n+1}}{c}. \end{aligned}$$

This shows that the error has the same magnitude, $|g_{n+1}/c|$, at every point in X and alternates in sign. By the Alternation Theorem, p is optimal.

Application to a Finite Set of More Than n+2 Points

Now suppose that X is a finite set, but with more than $n+2$ points. Let Y be a subset of X with exactly $n+2$ points, say $y_1 < \cdots < y_{n+2}$. Let p_Y denote the best approximation on Y of degree at most n as constructed above. If p_Y is the best approximation on all of X, we are finished. Otherwise, we proceed as follows. Since p_Y was constructed to have the alternation property, we can write

$$f(y_{k+1}) - p_Y(y_{k+1}) = -[f(y_k) - p_Y(y_k)], \quad k = 1, \ldots, n+1. \tag{8.11}$$

Since p_Y is not optimal on all of X there must be some $x_0 \in X$ but $x_0 \notin Y$ with

$$|f(x_0) - p_Y(x_0)| > |f(y_k) - p_Y(y_k)|, \quad k = 1, \ldots, n+2.$$

We now replace a $y_0 \in Y$ with x_0, where y_0 is chosen as follows. Suppose $y_k < x_0 < y_{k+1}$. Recall that according to (8.11), ***[Draw a picture of the error*** $f(x) - p(x)$ ***noting the points*** $x = y_k$***.]***

$$f(y_{k+1}) - p_Y(y_{k+1}) \quad \text{and} \quad f(y_k) - p_Y(y_k)$$

have opposite signs. We take y_0 to be y_k or y_{k+1} so that $f(y_0) - p_Y(y_0)$ and $f(x_0) - p_Y(x_0)$ have the same sign. Now suppose $x_0 < y_1$. If the sign of $f(x_0) - p_Y(x_0)$ is the same as the sign of $f(y_1) - p_Y(y_1)$, we take $y_0 = y_1$; otherwise we take $y_0 = y_{n+2}$. Finally, suppose $x_0 > y_{n+2}$. If the sign of $f(x_0) - p_Y(x_0)$ is the same as the sign of $f(y_{n+2}) - p_Y(y_{n+2})$, we take $y_0 = y_{n+2}$; otherwise we take $y_0 = y_1$. As a result of this exchange, the errors on our modified Y alternate in sign and the error at x_0 has a greater magnitude than all the others (whose errors all have the same magnitude).

On the modified Y, we can construct a new best approximation of degree at most n. It can be shown that the worst-case error of this new approximation of the modified Y is strictly greater that before. Continuing in this way, we obtain a subset of X having $n+2$ points and a polynomial of degree at most n that satisfies the sufficiency conditions of the Alternation Theorem.

Proof of Sufficiency of the Alternation Theorem

Let f be a function defined on a bounded subset of the real line, and let p be a polynomial of degree at most n. Suppose that the worst-case error between $f(x)$ and $p(x)$ is achieved at $n+2$ points $x_1 < \cdots < x_{n+2}$ and that these errors alternate in sign; i.e.,

$$f(x_{k+1}) - p(x_{k+1}) = -[f(x_k) - p(x_k)], \quad k = 1, \ldots, n+1. \tag{8.12}$$

We must show that p is a best approximation of f on X. Our proof is by contradiction. Suppose there is a polynomial q that is better than p. In other words, the worst-case error between f and q is strictly less than the worst-case error between f and p. In particular, this implies that for $k = 1, \ldots, n+2$,

$$|f(x_k) - q(x_k)| < |f(x_k) - p(x_k)| = \text{constant} = \text{worst-case error between } f \text{ and } p.$$

Now this inequality implies that the sign of the difference

$$[f(x_k) - p(x_k)] - [f(x_k) - q(x_k)] = q(x_k) - p(x_k)$$

is equal to the sign of the leftmost term in brackets (separately consider the two possibilities for the sign of the second term in brackets). Hence, on account of (8.12), the polynomial $q(x) - p(x)$ changes sign $n+1$ times. By the intermediate-value theorem, this polynomial has $n+1$ roots. But since the degree of $q - p$ is at most n, we conclude that $q - p$ is the zero polynomial.

Problems

8.1. Determine whether or not the discrete-time filter

$$H(z) = \frac{1 - z^{-1}}{1 + (1/2)z^{-1} - (1/2)z^{-2}}$$

is a lowpass filter. Also determine whether or not the filter is stable.

8.2. Consider a GLP filter with negative group delay τ. Suppose further that the impulse response is **anticausal**; i.e., $h_n = 0$ for $n > 0$. Identify *all* n such that h_n must be zero.

8.3. Let $H(f)$ be the frequency response of a discrete-time system that is GLP and whose impulse response is real. Consider the cascade system $G(f)$ in Figure 8.7. Determine whether or not $G(f)$ is GLP, and if it is GLP, determine its

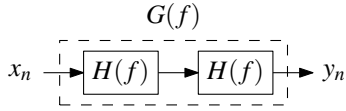


Figure 8.7. Cascade system for Problem 8.3.

type (I, II, III, or IV).

8.4. If $H(f)$ is GLP, show that for any integer k, $H(f + k/(2\tau))$ is also GLP with the same group delay τ and the same phase φ_0. It is now easy to see that the Hann, Hamming, and Blackman windows are linear phase with group delay $\tau = (N-1)/2$ and zero phase.

8.5. Show that the Bartlett, Hann, and Kaiser window functions w_n defined in Chapter 4 satisfy the symmetry condition (8.7) with group delay $\tau = (N-1)/2$.

8.6. Let $H(f)$ be a GLP filter with group delay τ and phase φ_0. Let h_n denote the corresponding impulse response. Let w_n be an arbitrary window function that is also GLP with the same phase delay τ but with a possibly different phase θ. Show that the new impulse response $g_n := w_n h_n$ is GLP and identify its group delay and phase. *Hint:* It suffices to show that g_n satisfies the generalized symmetry condition (8.3).

8.7. Let g_n be conjugate symmetric about $n = 0$; i.e., $g_n = \overline{g_{-n}}$. Show that its DTFT $G(f)$ is real valued. More specifically, show that $G(f) = g_0 + 2\,\mathrm{Re}\,G_1(f)$ where $G_1(f) := \sum_{n=1}^{\infty} g_n e^{-j2\pi fn}$. In particular, you must show that g_0 is real.

8.8. Let $H(f)$ denote the DTFT of h_n. Assuming h_n satisfies (8.4) for some integer group delay τ and some phase $\varphi_0 \in (-\pi, \pi)$, show that $H(f)$ is GLP. *Hints:* Write out the definition of $H(f)$ and make the change of variable $m = n - \tau$. Use the fact that g_n defined below (8.4) is conjugate symmetric about $n = 0$.

8.9. Let $g_n = \overline{g_{-n-1}}$. Show that its DTFT $G(f)$ has the form

$$G(f) = G_0(f) + \overline{G_0(f)} e^{j2\pi f},$$

where $G_0(f) := \sum_{n=0}^{\infty} g_n e^{-j2\pi fn}$.

8.10. Let $H(f)$ denote the DTFT of h_n. Assuming h_n satisfies (8.5) for some half integer group delay τ and some phase $\varphi_0 \in (-\pi, \pi)$, show that $H(f)$ is GLP.

Hints: Write out the definition of $H(f)$ and make the change of variable $n = \tau + 1/2 + m$. Use the fact that g_n defined below (8.5) satisfies $g_n = \overline{g_{-n-1}}$.

8.11. Show that a type-III filter cannot be highpass.

8.12. Derive the impulse-response sequence for the LPF in Example 8.3.1.

8.13. Derive the impulse-response sequence for the BPF in Example 8.3.2. In the case of a type-I HPF ($f_2 = 1/2$), show that

$$h_n = \delta_{n-\tau} - 2f_1 \operatorname{sinc}(2f_1[n-\tau]).$$

8.14. Consider an ideal bandstop filter with stopband $f_1 \le |f| \le f_2$. For an ideal type-I filter, show that the impulse-response sequence is

$$h_n = \delta_{n-\tau} - 2f_2 \operatorname{sinc}(2f_2[n-\tau]) + 2f_1 \operatorname{sinc}(2f_1[n-\tau]).$$

8.15. Derive the impulse-response sequence for the differentiator in Example 8.2.1.

8.16. Consider the DTFT $H(f) = j2\pi f e^{-j2\pi\tau f}$ for $|f| \le 1/2$. If τ is an integer, say $\tau = k$, show that $H(-1/2) = -H(1/2)$. If τ is a half integer, say $\tau = k + 1/2$, show that $H(-1/2) = H(1/2)$.

8.17. Let $\hat{x}(t)$ denote the Hilbert transform of $x(t)$. Show that the Hilbert transform of $\hat{x}(t)$ is $-x(t)$. *Hint:* This is easy in the frequency domain.

8.18. If $x(t)$ is real valued, show that $\hat{x}(t)$ is real valued. *Hint:* The result of Problem 1.19 in Chapter 1 may be helpful.

8.19. Let $x(t)$ have Fourier transform $X(f) = L(f) + R(f)$ as in the text. Let $\hat{x}(t)$ denote the Hilbert transform of $x(t)$. The waveform $x_+(t) := x(t) + j\hat{x}(t)$ is called the **pre-envelope** or **analytic signal** of $x(t)$. Show that the Fourier transform of $x_+(t)$ is $2R(f)$. The point here is that the Fourier transform of $x_+(t)$ is nonzero only for $f \ge 0$. *Hint:* Use the fact that $\hat{X}(f) = -j[R(f) - L(f)]$.

8.20. ***Introduction.*** If we put $\psi(t) := x_+(t)e^{-j2\pi f_0 t}$, then $\psi(t)e^{j2\pi f_0 t} = x_+(t) = x(t) + j\hat{x}(t)$. If $x(t)$ is real, then we know from Problem 8.18 that $\hat{x}(t)$ is also real. This means that $x(t)$ and $\hat{x}(t)$ are the real and imaginary parts of $x_+(t)$, respectively. It then follows that $\operatorname{Re}\psi(t)e^{j2\pi f_0 t} = x(t)$. The point here is that if $x(t)$ is a *real* bandpass signal centered at f_0, then $\psi(t)$, whose transform is $X_+(f + f_0)$, is a *complex* lowpass signal. In this case, we call $\psi(t)$ the **equivalent lowpass signal** of $x(t)$. In general, $\psi(t)$ is called the **complex envelope** of $x(t)$.

Problem. If we let $x_I(t)$ and $x_Q(t)$ denote the real and imaginary parts of $\psi(t)$, then

$$\operatorname{Re}\psi(t)e^{j2\pi f_0 t} = x_I(t)\cos(2\pi f_0 t) - x_Q(t)\sin(2\pi f_0 t).$$

We call $x_I(t)$ and $x_Q(t)$ the **in-phase** and **quadrature** components of $x(t)$. Assuming $x(t)$ is real, show that

$$x_I(t) = x(t)\cos(2\pi f_0 t) + \hat{x}(t)\sin(2\pi f_0 t)$$

and

$$x_Q(t) = \hat{x}(t)\cos(2\pi f_0 t) - x(t)\sin(2\pi f_0 t).$$

Also show that if $x(t)$ is bandpass, then $x_I(t)$ and $x_Q(t)$ are lowpass. *Hint:* Formulas such as (8.8) and (8.9) may be helpful.

8.21. Show that the Fourier transform of $y(t)\sin(2\pi f_0 t)$ is

$$\tfrac{1}{2j}[Y(f-f_0) - Y(f+f_0)].$$

8.22. Show that the inverse DTFT of $-j\,\mathrm{sgn}(f)e^{-j2\pi\tau f}$ for $|f| \le 1/2$ is equal to

$$h_n = \frac{1-\cos(\pi[n-\tau])}{\pi(n-\tau)}, \quad n \neq \tau.$$

8.23. Show that the inverse Fourier transform of $-j\,\mathrm{sgn}(f)$ is $1/(\pi t)$. *Hint:* Sketch

$$S_n(f) := \begin{cases} e^{-f/n}, & f > 0, \\ -e^{f/n}, & f < 0, \\ 0, & f = 0, \end{cases}$$

and observe that as $n \to \infty$, $S_n(f) \to \mathrm{sgn}(f)$. This suggests that if you first compute the inverse Fourier transform of $-jS_n(f)$ for finite n and then let $n \to \infty$, you will obtain the inverse Fourier transform of $-j\,\mathrm{sgn}(f)$.

8.24. Show that the Fourier transform of the unit-step function $u(t)$ is $(1/2)\delta(f) + 1/(j2\pi f)$. *Hint:* Use the result of the preceding problem and a duality argument to show that the Fourier transform of $\mathrm{sgn}(t)$ is $1/(j\pi f)$. Then use the fact that $u(t) = (1/2)[1 + \mathrm{sgn}(t)]$.

8.25. Let g_n be symmetric or antisymmetric about $n = 0$; i.e., $g_n = g_{-n}$ for all n or $g_n = -g_{-n}$ for all n. Show that if z is a zero of the z transform of g_n, then so are $\bar{z}$, $1/z$, and $1/\bar{z}$. *Hint:* Show that $G(\bar{z}) = \overline{G(z)}$ and that $G(1/z) = G(z)$.

8.26. If $g_n = g_{-n-1}$ for all n or if $g_n = -g_{-n-1}$ for all n, show that the z transform of g_n has the property that if z is a zero, then so are $\bar{z}$, $1/z$, and $1/\bar{z}$.

CHAPTER 9
Filter Implementation

9.1. Quantization of Difference-Equation Coefficients

Consider a second-order difference equation of the form

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} + x_n.$$

In any realization of this system, either in MATLAB or in special-purpose DSP hardware, the coefficients a_1 and a_2 will be represented using only a finite number of bits. Our goal here is to understand how this quantization affects the poles of the resulting system.

The transfer function of the above system is $1/(1+a_1z^{-1}+a_2z^{-2})$. Writing the denominator as

$$\begin{aligned} 1+a_1z^{-1}+a_2z^{-2} = z^{-2}(z^2+a_1z+a_2) &= z^{-2}(z-r_1)(z-r_2) \\ &= z^{-2}[z^2-(r_1+r_2)z+r_1r_2], \end{aligned}$$

we see that if a_1 is real, then the imaginary parts of r_1 and r_2 must be equal and opposite. If a_2 is also real, and if the imaginary parts of r_1 and r_2 are not zero, then $\mathrm{Re}\, r_1 = \mathrm{Re}\, r_2$. Hence, if r_1 and r_2 are not pure real, then r_1 and r_2 are complex conjugates and can be written in the form $r_1 = re^{j\theta}$ and $r_2 = re^{-j\theta}$. Thus, if a_1 and a_2 are real and the poles are not pure real, $a_2 = r^2 > 0$. If $0 < \theta < \pi$, then r_1 is in the upper half plane and r_2 is in the lower half plane.

In the case of poles of the form $re^{\pm j\theta}$, we have

$$a_1 = -2r\cos\theta \quad \text{and} \quad a_2 = r^2.$$

Note that for a stable system ($r < 1$), $|a_1| < 2$ and $|a_2| < 1$. The formulas for a_1 and a_2 can be solved for r and θ whenever $a_2 > 0$ and $|a_1| \le 2\sqrt{a_2}$. We have[1]

$$r = \sqrt{a_2} \quad \text{and} \quad \theta = \cos^{-1}\left(\frac{-a_1}{2\sqrt{a_2}}\right).$$

Now suppose that the coefficients a_1 and a_2 restricted to be represented by m-bit words. Then a_1 and a_2 can take only finitely many values, and so the same is true of

[1] The case $|a_1| = 2\sqrt{a_2}$ results in $\theta = 0$ or $\theta = \pi$. In this case, there is a repeated pole at r ($\theta = 0$) or a repeated pole at $-r$ ($\theta = \pi$).

r and θ via the foregoing formulas. To be more precise about this, let a_1 and a_2 be of the form $a_1 = 2k/2^{m-1}$ and $a_2 = l/2^{m-1}$, where k and l are integers in the range $-2^{m-1}, \ldots, -1, 0, 1, \ldots, 2^{m-1} - 1$, with $l > 0$. (The forms of a_1 and a_2 imply $|a_1| < 2$ and $|a_2| < 1$, which is required for stability.) Then the possible pole locations (first quadrant only) are shown in Figure 9.1. ***[Run rootscript.m]***

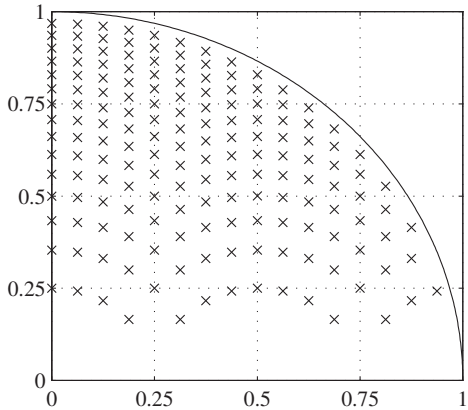


Figure 9.1. Possible locations (first quadrant only) of stable poles of a second-order difference equation if the coefficients are quantized to $m = 5$ bits.

Before developing a way to overcome this problem, we need to learn a little bit about block diagrams.

9.2. Block Diagrams

Consider the system shown in Figure 9.2 In this diagram, the block labeled z^{-1}

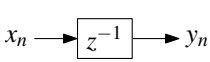


Figure 9.2. A simple block diagram of the unit-delay system.

is the **unit-delay system**. That is, its transfer function is $H(z) = z^{-1}$ and its impulse response is $h_n = \delta_{n-1}$. Hence, the output is

$$y_n = \sum_{k=-\infty}^{\infty} h_k x_{n-k} = \sum_{k=-\infty}^{\infty} \delta_{k-1} x_{n-k} = x_{n-1}.$$

Of course, saying $y_n = x_{n-1}$ is the same as saying $x_n = y_{n+1}$. So the diagram could just as well be drawn as shown in Figure 9.3

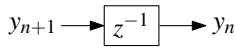


Figure 9.3. An equivalent labeling of the unit-delay system.

Now consider the more complicated situation in Figure 9.4. Since the signal

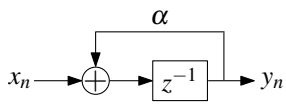


Figure 9.4. A more complicated system with feedback.

entering the unit delay is y_{n+1}, the block diagram says that

$$y_{n+1} = \alpha y_n + x_n.$$

Next consider a pair of such systems interconnected as shown in Figure 9.5. In

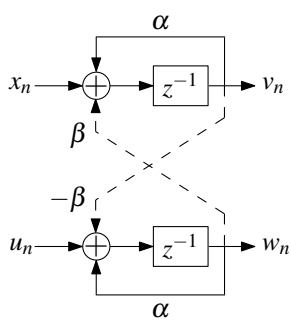


Figure 9.5. An interconnection of systems with feedback.

the time domain, this diagram says that

$$v_{n+1} = \alpha v_n + \beta w_n + x_n$$
$$w_{n+1} = \alpha w_n - \beta v_n + u_n.$$

To work in the transform domain, we could take the z transform of these coupled equations. However, we can write the z transform equations directly from the diagram by imagining each time function as being replaced by its z transform. This substitution, which we normally just do in our heads, is shown explicitly here in Figure 9.6. We can now write

$$V(z) = z^{-1}[\alpha V(z) + \beta W(z) + X(z)]$$
$$W(z) = z^{-1}[\alpha W(z) - \beta V(z) + U(z)].$$

To see what is going on, we write this in matrix-vector form as

$$\begin{bmatrix} V(z) \\ W(z) \end{bmatrix} = \begin{bmatrix} \alpha z^{-1} & \beta z^{-1} \\ -\beta z^{-1} & \alpha z^{-1} \end{bmatrix} \begin{bmatrix} V(z) \\ W(z) \end{bmatrix} + z^{-1} \begin{bmatrix} X(z) \\ U(z) \end{bmatrix},$$

or

$$z \begin{bmatrix} 1-\alpha z^{-1} & -\beta z^{-1} \\ \beta z^{-1} & 1-\alpha z^{-1} \end{bmatrix} \begin{bmatrix} V(z) \\ W(z) \end{bmatrix} = \begin{bmatrix} X(z) \\ U(z) \end{bmatrix}.$$

It follows that

$$\begin{bmatrix} V(z) \\ W(z) \end{bmatrix} = \frac{z^{-1}}{1-2\alpha z^{-1} + (\alpha^2+\beta^2) z^{-2}} \begin{bmatrix} 1-\alpha z^{-1} & \beta z^{-1} \\ -\beta z^{-1} & 1-\alpha z^{-1} \end{bmatrix} \begin{bmatrix} X(z) \\ U(z) \end{bmatrix}. \quad (9.1)$$

9.3. An Alternative Realization

To address the pole-location problem introduced at the beginning of the chapter, consider the modifications of Figure 9.6 shown in Figure 9.7. We see that

$$y_n = Cv_n + Kw_n + x_n,$$

or in the transform domain,

$$Y(z) = CV(z) + KW(z) + X(z).$$

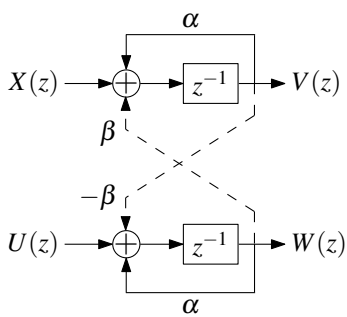


Figure 9.6. The z transform of Figure 9.5.

We can now apply (9.1), noting that from Figure 9.7, $U(z) = 0$. Thus,

$$\begin{aligned} Y(z) &= \left[\frac{Cz^{-1}(1-\alpha z^{-1}) - K\beta z^{-2}}{1 - 2\alpha z^{-1} + (\alpha^2+\beta^2)z^{-2}} + 1 \right] X(z) \\ &= \frac{1 + (C-2\alpha)z^{-1} + (\alpha^2+\beta^2 - C\alpha - K\beta)z^{-2}}{1 - 2\alpha z^{-1} + (\alpha^2+\beta^2)z^{-2}} X(z). \end{aligned}$$

Given any constants real b_1 and b_2, if we take

$$C := b_1 + 2\alpha, \quad K := \frac{\alpha^2+\beta^2 - C\alpha - b_2}{\beta}, \quad \text{and} \quad r := \alpha + j\beta,$$

then

$$Y(z) = \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{(1 - rz^{-1})(1 - \overline{r}z^{-1})} X(z). \tag{9.2}$$

In other words, we can realize any second-order filter directly in terms of its poles by using the structure in Figure 9.7, assuming that the poles are complex conjugates with nonzero imaginary parts. If we now plot such poles when α and β have the form $k/2^{m-1}$ for $k = -(2^{m-1}-1), \ldots, 2^{m-1}-1$ and lie inside the unit circle, we get the plot shown in Figure 9.8 (first quadrant only). ***[Run roots2script.m Then Run butterworthscript.m and rootscript.m and roots2script.m]***

9.4. Realization of IIR Filters

Recall that IIR filters designed by applying the bilinear transformation to analog Butterworth or Chebyshev filters result in a z-domain transfer function in which the numerator and denominator have the same degree, which we denote here by N and write

$$H(z) = \frac{b_0 \prod_{k=1}^{N} (1 - \beta_k z^{-1})}{\prod_{k=1}^{N} (1 - \alpha_k z^{-1})}.$$

Furthermore, when N is odd, there is one real pole and the rest occur in complex-conjugate pairs. When N is even, all the poles occur in complex-conjugate pairs, and there are no real poles. Hence, for even N, we can write

$$H(z) = b_0 \prod_{k=1}^{N/2} \frac{1 + b_1^{(k)} z^{-1} + b_2^{(k)} z^{-2}}{1 + a_1^{(k)} z^{-1} + a_2^{(k)} z^{-2}},$$

where the coefficients are real. This transfer function can be realized by the **cascade** shown in Figure 9.9, where $H_k(z) = (1+b_1^{(k)}z^{-1}+b_2^{(k)}z^{-2})/(1+a_1^{(k)}z^{-1}+a_2^{(k)}z^{-2})$. For odd N we can write

$$H(z) = \left[b_0 \prod_{k=1}^{(N-1)/2} \frac{1+b_1^{(k)}z^{-1}+b_2^{(k)}z^{-2}}{1+a_1^{(k)}z^{-1}+a_2^{(k)}z^{-2}} \right] \frac{1-b^{(0)}z^{-1}}{1-a^{(0)}z^{-1}},$$

which can be realized by the cascade shown in Figure 9.10. Here $H_0(z) = (1-b^{(0)}z^{-1})/(1-a^{(0)}z^{-1})$, and for $k \geq 1$, the $H_k(z)$ are as before.

An alternative to the cascade realization is the **parallel realization**. To design a parallel realization, we first expand $H(z)$ using partial fractions to write

$$H(z) = C_0 + \sum_{k=1}^{N} \frac{C_k}{1-\alpha_k z^{-1}}.$$

When N is even, we can group terms containing complex-conjugate pairs and write

$$H(z) = C_0 + \sum_{k=1}^{N/2} \frac{\gamma_0^{(k)}+\gamma_1^{(k)}z^{-1}}{1+a_1^{(k)}z^{-1}+a_2^{(k)}z^{-2}},$$

which can be realized by the parallel form shown in Figure 9.11, where $\widetilde{H}_k(z) = (\gamma_0^{(k)}+\gamma_1^{(k)}z^{-1})/(1+a_1^{(k)}z^{-1}+a_2^{(k)}z^{-2})$.
Similarly, for odd N,

$$H(z) = C_0 + \frac{\gamma^{(0)}}{1-\alpha z^{-1}} + \sum_{k=1}^{(N-1)/2} \frac{\gamma_0^{(k)}+\gamma_1^{(k)}z^{-1}}{1+a_1^{(k)}z^{-1}+a_2^{(k)}z^{-2}},$$

which can be realized by the parallel form shown in Figure 9.12, where $\widetilde{H}_0(z) = \gamma^{(0)}/(1-\alpha z^{-1})$, and for $k \geq 1$, the $\widetilde{H}_k(z)$ are as before.

Although there are many ways to realize the second-order subsystems $H_k(z)$ for $k \geq 1$, when the hardware has short word length, the structure of Figure 9.7 should be considered.

9.5. Direct-Form Realizations

The difference equation

$$y_n = -\sum_{k=1}^{N} a_k y_{n-k} + x_n \tag{9.3}$$

is easily seen to be realized by the architecture shown in Figure 9.13. This is equivalent to Figure 9.14.

With an eye toward realizing the more general difference equation

$$y_n = -\sum_{k=1}^{N} a_k y_{n-k} + \sum_{k=0}^{N} b_k x_{n-k}, \tag{9.4}$$

observe that $\sum_{k=0}^{N} b_k x_{n-k}$ is easily realized by the architecture in Figure 9.15. If we take the output of Figure 9.15 and use it as the input to Figure 9.14, we can realize the general difference equation (9.4). This is shown in Figure 9.16. The architecture in Figure 9.4 is called **direct form I**.

Although the direct form I architecture is easy to understand, it is wasteful of hardware. The direct form I architecture requires $2N$ delays. We next introduce the **direct form II** architecture, which uses only N delays. The idea is to look at the problem in the z transform domain and write $B(z)/A(z)$ as the cascade $B(z)\cdot(1/A(z))$. Put $V(z) := (1/A(z))X(z)$ so that $Y(z) = B(z)V(z) = (B(z)/A(z))X(z)$. The system $V(z) = (1/A(z))X(z)$ can be realized by changing y_n in Figure 9.13 to v_n as shown in Figure 9.17, where we have also indicated that the signals v_{n-k} can be directly tapped off the delays. Now observe that the system $Y(z) = B(z)V(z)$, which corresponds to $\sum_{k=0}^{N} b_k v_{n-k}$, can be obtained by attaching some adders to Figure 9.17 as shown in Figure 9.18.

9.6. Transposed Direct Forms

Suppose we take the delay elements of Figure 9.14 and move them into the branches above as shown in Figure 9.19. This modified architecture still realizes the difference equation (9.3). This can be seen be observing that although the input to the left-most delay element is $-a_3 y_n$, by the time the signal reaches the right-hand end, it is now $-a_3 y_{n-3}$.

We can apply the same changes to Figure 9.15 if we also reverse the order of the coefficients as shown in Figure 9.20.

To obtain the **transposed direct form II**, write $B(z)/A(z)$ as the cascade form $(1/A(z))B(z)$. In more detail, put $W(z) := B(z)X(z)$ so that $Y(z) = (1/A(z))W(z) = (1/A(z))B(z)X(z) = (B(z)/A(z))X(x)$. If we use Figure 9.20 to compute $B(z)X(z)$, we then see that the adders and delays of Figure 9.20 can do double duty to realize $(1/A(z))W(z)$ as shown in Figure 9.21. The architecture in Figure 9.21 is known as the transposed direct form II.

Observe that Figure 9.21 can be obtained from Figure 9.18 by the following steps: (i) reverse the directions of all arrows; (ii) replacing all adders by connections and replacing all connections by adders; and (iii) interchanging the input and the output.

This algorithm is called **transposition**, and it leaves the overall transfer function unchanged [7, p. 395]. Applying the algorithm to Figure 9.21 returns us to Figure 9.18.

9.7. Direct-Form Realizations of Real GLP FIR Filters

Recall that if the impulse response h_n of a GLP filter is real, it must satisfy $h_n = \pm h_{N-n}$, where $N/2$ is the group delay (which must be an integer or a half integer). If the filter is causal, it must be FIR, and its length is $N+1$; i.e., its order is N. Hence, for an even-order filter,

$$\begin{aligned}
\sum_{k=0}^{N} h_k x_{n-k} &= \sum_{k=0}^{N/2-1} h_k x_{n-k} + h_{N/2} x_{n-N/2} + \sum_{k=N/2+1}^{N} h_k x_{n-k} \\
&= \sum_{k=0}^{N/2-1} h_k x_{n-k} + h_{N/2} x_{n-N/2} \pm \sum_{k=N/2+1}^{N} h_{N-k} x_{n-k} \\
&= \sum_{k=0}^{N/2-1} h_k x_{n-k} + h_{N/2} x_{n-N/2} \pm \sum_{m=0}^{N/2-1} h_m x_{n-(N-m)} \\
&= h_{N/2} x_{n-N/2} + \sum_{k=0}^{N/2-1} h_k (x_{n-k} \pm x_{n-N+k}),
\end{aligned}$$

and for an odd-order filter,

$$\sum_{k=0}^{N} h_k x_{n-k} = \sum_{k=0}^{(N-1)/2} h_k (x_{n-k} \pm x_{n-N+k}),$$

By writing the convolutions in this way, we can reduce the number of coefficients we have to store. A direct form II realization of even order 6 is shown in Figure 9.22.

Problems

9.1. Find the transfer function of system in Figure 9.23. Then show that if $K = 0$, the system is unstable.

9.2. Draw a direct form II realization of a real, 5th-order GLP FIR filter.

9.3. Draw the transpose of the implementation in Figure 9.22.

9.4. A causal filter has transfer function $H(z) = 10 + 5z^{-1} + 5z^{-3} + 10z^{-4}$.

(a) What is the minimum number of delays with which the system can be realized?

(b) What is the order of the filter?

(c) Determine whether or not the filter has generalized linear phase (GLP).

9.5. Draw a parallel realization of the system

$$H(z) = \frac{1}{(1 - \frac{1}{2}z^{-1})(1 - \frac{1}{3}z^{-1})}.$$

using two first-order sections. Draw details of the first-order sections.

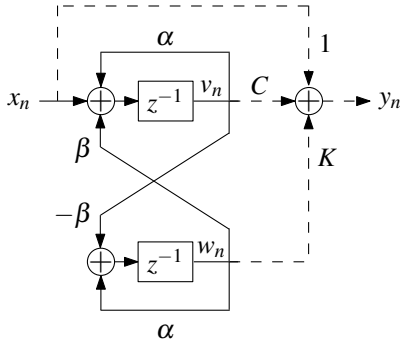


Figure 9.7. Modification of Figure 9.6 that realizes the transfer function in (9.2).

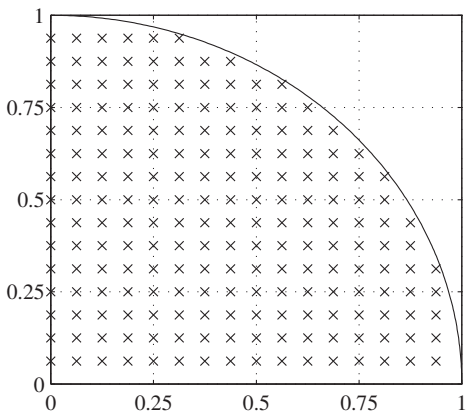


Figure 9.8. Possible locations (first quadrant only) of stable poles $r = \alpha \pm j\beta$ of transfer function in (9.2) when α and β are quantized to $m = 5$ bits.

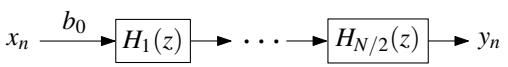


Figure 9.9. Cascade realization of a transfer function composed of an even number of factors.

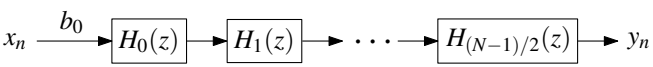


Figure 9.10. Cascade realization of a transfer function composed of an odd number of factors.

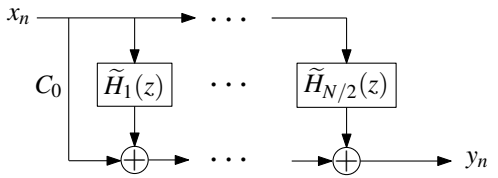


Figure 9.11. Parallel realization of a transfer function of even order.

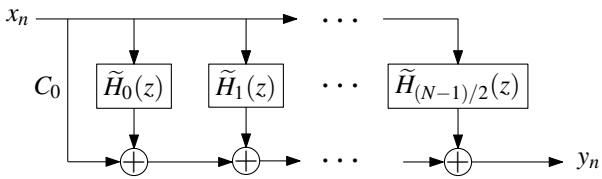


Figure 9.12. Parallel realization of a transfer function of odd order.

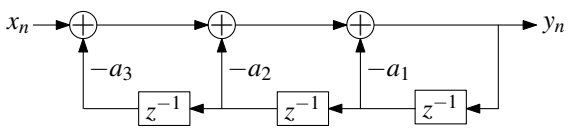


Figure 9.13. An architecture for realizing the difference equation (9.3) with $N = 3$.

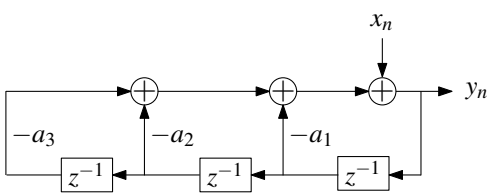


Figure 9.14. An equivalent architecture for realizing the difference equation (9.3) with $N = 3$.

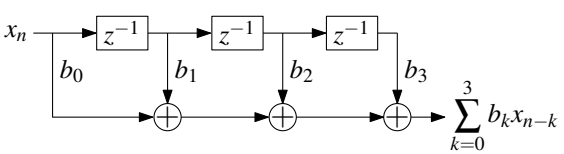


Figure 9.15. An architecture for realizing $\sum_{k=0}^{N} b_k x_{n-k}$ with $N = 3$.

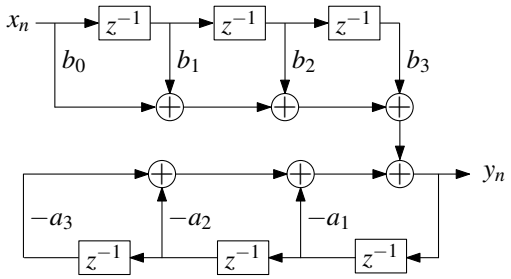


Figure 9.16. The direct form I architecture for realizing (9.4) with $N = 3$.

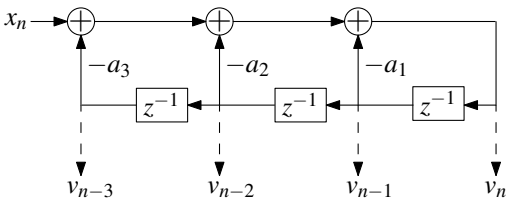


Figure 9.17. Architecture for realizing $V(z) = (1/A(z))X(z)$ with $N = 3$.

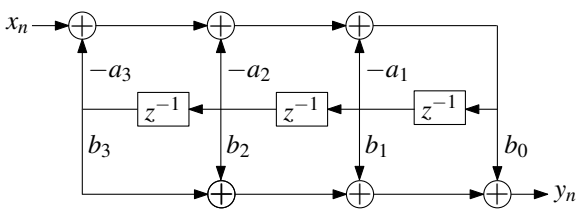


Figure 9.18. The direct form II architecture for realizing the general difference equation (9.4) with $N = 3$.

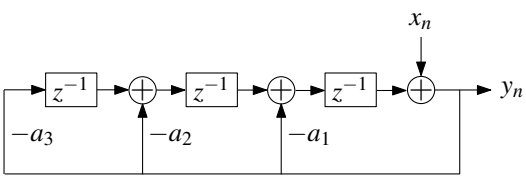


Figure 9.19. Modification of Figure 9.14.

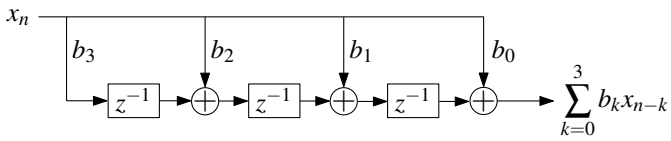


Figure 9.20. Modification of Figure 9.15.

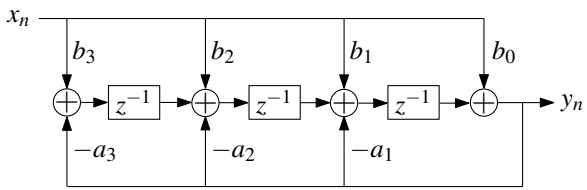


Figure 9.21. Transposed direct form II architecture.

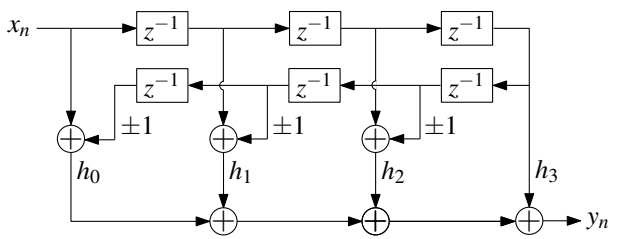


Figure 9.22. Direct form II realization of a real, 6th-order GLP FIR filter.

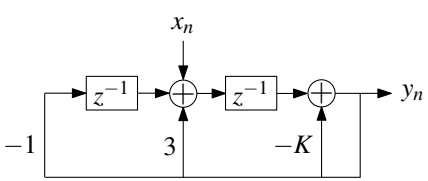


Figure 9.23. Block diagram for Problem 9.1.

CHAPTER 10
Sampling Rate Conversion

We saw in Chapter 1 that a continuous-time waveform $x(t)$ can be recovered from its samples if the waveform is bandlimited and if the samples are close enough together. It follows that if we take such samples, say $x(n/f_s)$, we can always compute new samples taken at a different rate, say $x(n/f_s')$. More explicitly, we can use the sinc reconstruction formula to write

$$x(n/f_s') = \sum_{m=-\infty}^{\infty} x(m/f_s)\,\mathrm{sinc}(f_s[n/f_s' - m/f_s]).$$

What we are doing is going from discrete time to continuous time and back to discrete time. Can we do this completely in discrete time? Under certain conditions, the answer is "yes."

10.1. Upsampling and Interpolation

Let x_n be a discrete-time signal with DTFT $X(f)$. The sequence x_n need not be obtained by sampling a continuous-time waveform. The process of taking a sequence x_n and creating the sequence

$$y_n := \begin{cases} x_{n/I}, & \text{if } n = 0, \pm I, \pm 2I, \ldots, \\ 0, & \text{otherwise,} \end{cases}$$

where I is a positive integer, is called **upsampling**. For example, upsampling

$$\ldots \; x_{-1} \;\; x_0 \;\; x_1 \;\; x_2 \;\; \ldots$$

by $I = 3$ yields

$$\begin{array}{cccccccccccccc} \ldots & x_{-1} & 0 & 0 & x_0 & 0 & 0 & x_1 & 0 & 0 & x_2 & 0 & 0 & \ldots \\ \ldots & y_{-3} & y_{-2} & y_{-1} & y_0 & y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & \ldots \end{array}$$

In other words, to upsample by I is to insert $I-1$ zeros between elements of x_n. MATLAB does this to finite sequences with the command `upsample(x,I)`.

We can generalize the concept of upsampling by inserting nonzero values between the elements of x_n. But what values should we use? Even though x_n may not have come to us by sampling a bandlimited waveform, we can always construct a

virtual waveform $v(t)$ so that x_n is equal to its samples. Without loss of generality, let $v(t)$ be obtained by applying sinc reconstruction to x_n with $f_s = 1$; i.e.,

$$v(t) := \sum_{n=-\infty}^{\infty} x_n \operatorname{sinc}(t-n). \tag{10.1}$$

Since the Fourier transform of $\operatorname{sinc}(t-n)$ is $I_{[-1/2,1/2]}(f)e^{-j2\pi fn}$, the continuous-time Fourier transform of $v(t)$ is

$$V(f) = I_{[-1/2,1/2]}(f) \sum_{n=-\infty}^{\infty} x_n e^{-j2\pi fn} = I_{[-1/2,1/2]}(f)X(f).$$

Thus, the CTFT of $v(t)$ is equal to the DTFT of x_n for $|f| \le 1/2$; i.e., $V(f) = X(f)$ for $|f| \le 1/2$. However, keep in mind that the CTFT $V(f)$ is zero for $|f| > 1/2$. ***[Sketch*** $V(f)$ ***and*** $X(f)$***.]*** Hence, the DTFT of the samples $v(n/I)$ is equal to $IX(If)$ for $|f| \le 1/(2I)$ and is zero for $1/(2I) < |f| \le 1/2$. This is illustrated in Figure 10.1.

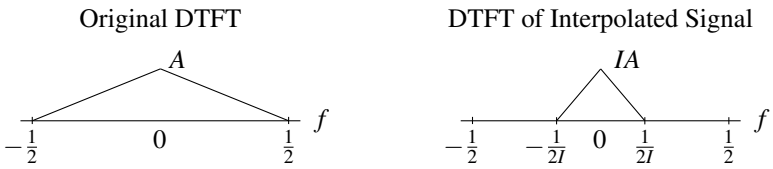


Figure 10.1. The DTFT of x_n and $v(n/I)$.

We now show that because I is an integer, is not necessary to construct the virtual waveform $v(t)$ to obtain the sequence $v(n/I)$. To see why, observe that the DTFT of y_n is

$$Y(f) = \sum_{n=-\infty}^{\infty} y_n e^{-j2\pi fn} = \sum_{m=-\infty}^{\infty} y_{mI} e^{-j2\pi fnmI} = \sum_{m=-\infty}^{\infty} x_m e^{-j2\pi (If)m} = X(If).$$

[Run interpolationplot.m DEMO.] If we apply y_n to a discrete-time ideal lowpass filter of gain I and bandwidth $1/(2I)$, then the resulting filter output sequence will be

$$\begin{aligned}
&\int_{-1/(2I)}^{1/(2I)} IX(If)e^{j2\pi fn}\,df \\
&\qquad = \int_{-1/2}^{1/2} X(\theta)e^{j2\pi(\theta/I)n}\,d\theta \\
&\qquad = \int_{-1/2}^{1/2} V(\theta)e^{j2\pi(\theta/I)n}\,d\theta, \quad \text{since } X(f) = V(f) \text{ for } |f| \le 1/2,
\end{aligned}$$

$$= \int_{-\infty}^{\infty} V(\theta) e^{j2\pi\theta(n/I)}\, d\theta, \quad \text{since } V(f) = 0 \text{ for } |f| > 1/2,$$
$$= v(n/I),$$

Of course, when $n = kI$, $v(n/I) = v(k) = x_k$. This combination of inserting zeros followed by lowpass filtering is called **interpolation**. Such a system is shown in Figure 10.2.

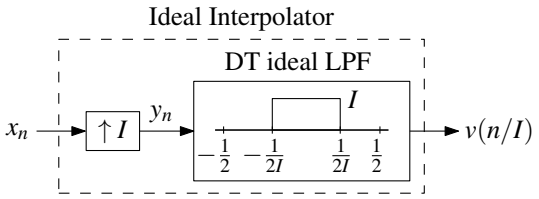


Figure 10.2. Block diagram of an ideal interpolator, consisting of an upsampler followed by an ideal lowpass filter.

MATLAB implements upsampling and lowpass filtering of a finite sequence with the command `interp(x,I)`; of course MATLAB cannot use an ideal lowpass filter.

Terminology. We warn the reader that many texts use both the terms upsampling and interpolation to mean insertion of zeros followed by lowpass filtering.

10.2. Downsampling and Decimation

The process of taking a sequence x_n and creating the sequence

$$y_n := x_{nD},$$

is called **downsampling**. For example, downsampling

$$\ldots \; x_{-3} \; x_{-2} \; x_{-1} \; x_0 \; x_1 \; x_2 \; x_3 \; \ldots$$

by $D = 3$ yields

$$\ldots \; x_{-6} \; x_{-3} \; x_0 \; x_3 \; x_6 \; \ldots$$
$$\ldots \; y_{-2} \; y_{-1} \; y_0 \; y_1 \; y_2 \; \ldots$$

In other words, the values of x_k when k is not a multiple of D are discarded. The downsampling of a finite sequence is implemented in MATLAB with the command `downsample(x,D)`.

As we shall see, it is often a good idea to apply a discrete-time lowpass filter to x_n before subsampling. The process of lowpass filtering followed by subsampling

is called **decimation**. To see the advantage of prefiltering, we need to compute the DTFT of y_n. However, it is convenient to first introduce the sequence

$$\hat{x}_n := \begin{cases} x_n, & \text{if } n = 0, \pm D, \pm 2D, \ldots, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $\hat{x}_n$ is obtained from x_n by setting $x_k = 0$ if k is not a multiple of D. We can also write $\hat{x}_n$ as the product of x_n with a discrete-time periodic impulse train. ***[Draw picture of impulse train.]*** If we put

$$\delta_n^D := \sum_{m=-\infty}^{\infty} \delta_{n-mD},$$

then $\hat{x}_n = x_n \delta_n^D$. With this notation, we can write $\hat{x}_{nD} = x_{nD}\delta_{nD}^D = x_{nD}$. We can now compute the DTFT of y_n. Write

$$\begin{aligned} Y(f) &= \sum_{n=-\infty}^{\infty} y_n e^{-j2\pi fn} \\ &= \sum_{n=-\infty}^{\infty} x_{nD} e^{-j2\pi fn} \\ &= \sum_{n=-\infty}^{\infty} \hat{x}_{nD} e^{-j2\pi fn} \\ &= \sum_{m=-\infty}^{\infty} \hat{x}_m e^{-j2\pi fm/D}, \end{aligned}$$

where we have used the fact that $\hat{x}_m = 0$ when m is not a multiple of D. We next use the geometric series to write

$$\delta_m^D = \frac{1}{D} \sum_{k=0}^{D-1} e^{j2\pi mk/D}.$$

Then

$$\begin{aligned} Y(f) &= \sum_{m=-\infty}^{\infty} x_m \delta_m^D e^{-j2\pi fm/D} \\ &= \sum_{m=-\infty}^{\infty} x_m \left[\frac{1}{D} \sum_{k=0}^{D-1} e^{j2\pi mk/D} \right] e^{-j2\pi fm/D} \\ &= \frac{1}{D} \sum_{k=0}^{D-1} \sum_{m=-\infty}^{\infty} x_m e^{-j2\pi m(f-k)/D} \\ &= \frac{1}{D} \sum_{k=0}^{D-1} X([f-k]/D), \end{aligned}$$

where $X(f)$ is the DTFT of x_n. ***[Run decimationplot.m DEMO.]*** Thus, downsampling causes **aliasing** of the original discrete-time signal, unless it is bandlimited to $1/(2D)$. For this reason, the ideal decimator has the structure shown in Figure 10.3. Simlarly, the MATLAB function `decimate(x,D)` includes a lowpass

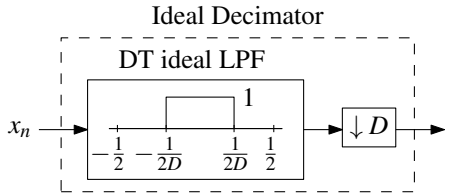


Figure 10.3. Block diagram of an ideal decimator, consisting of an ideal lowpass filter followed by a downsampler.

filter; of course MATLAB cannot use an ideal lowpass filter.

The general situation is shown in Figure 10.4.

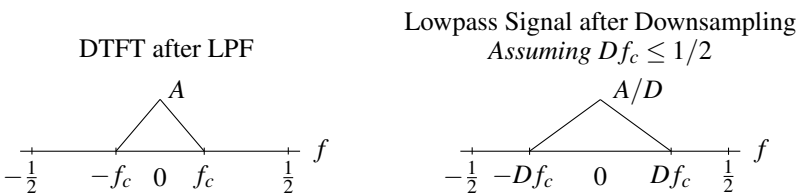


Figure 10.4. If a discrete-time signal is bandlimited to $f_c < 1/2$ and if we downsample by D such that $Df_c \le 1/2$, we avoid aliasing. If f_c is given, we must have $D \le 1/(2f_c)$. If D is given, we must make sure that the signal is bandlimited to $1/(2D)$.

Terminology. We warn the reader that many texts use both the terms downsampling and decimation to mean prefiltering followed by subsampling.

10.3. Sampling Rate Conversion

Suppose we have samples x_n arriving at f_s samples per second. If we upsample by I, we then have If_s samples per second. If we then downsample by D, we end up with $(If_s/D) = (I/D)f_s$ samples per second. In practice, we would usually include a lowpass filter after upsampling (i.e., interpolation), and then another lowpass filter before downsampling (i.e., decimation). Typically the two lowpass filters are combined into a single filter as shown in Figure 10.5, where the cutoff frequency $f_c = \min\{1/(2I), 1/(2D)\}$. For finite sequences, MATLAB does all this (using a

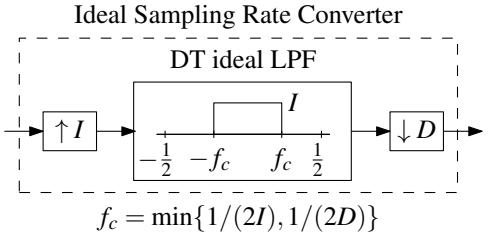


Figure 10.5. An ideal sampling rate converter. There is no loss of information if $D \leq I$.

nonideal filter) with the command `resample(x,I,D)`. The `resample` function makes the `decimate` and `interp` functions obsolete.

10.4. Application to Sampling Continuous-Time Waveforms

In the preceding sections, our starting point has been a discrete-time signal x_n with DTFT $X(f)$. In this section, we start with a continuous-time signal $x(t)$ with CTFT $X(f)$ that is bandlimited to f_c and initially sampled at rate $f_s \geq 2f_c$ to produce the discrete-time signal $x(n/f_s)$. Thus, the DTFT of the samples, $X_{\text{DTFT},f_s}(f)$ is bandlimited to $f_c/f_s \leq 1/2$. If we upsample by I, and then apply an ideal discrete-time lowpass filter of gain I and bandwidth $1/(2I)$, the resulting output sequence will have DTFT $X_{\text{DTFT},If_s}(f)$ and will be bandlimited to $(f_c/f_s)/I$. Such a system is shown in Figure 10.6.

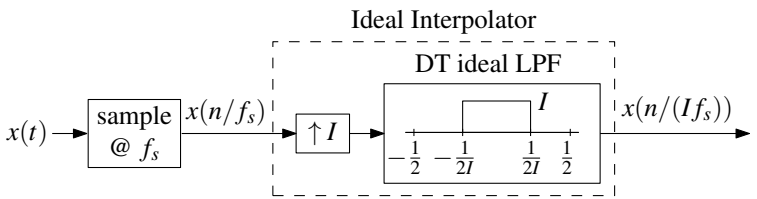


Figure 10.6. A discrete-time interpolation system applied to samples of a bandlimited, continuous-time waveform.

Since $X_{\text{DTFT},If_s}(f)$ is bandlimited to $(f_c/f_s)/I$, we can downsample by D without aliasing for any D such that $D(f_c/f_s)/I \leq 1/2$, or $D \leq If_s/(2f_c)$. This is illustrated in Figure 10.7.

***Example* 10.4.1** (Fractional Delay). Given samples $x(n/f_s)$, it is easy to produce the fractionally delayed samples $x([n-k/I]/f_s)$ by inserting k delays between interpolation by I and decimation by I as shown in Figure 10.8.

$$x\left(\frac{n}{If_s}\right) \longrightarrow \boxed{\downarrow D} \longrightarrow x\left(\frac{nD}{If_s}\right) = x\left(\frac{n}{(I/D)f_s}\right)$$

Figure 10.7. A discrete-time decimation system applied to samples of a bandlimited, continuous-time waveform.

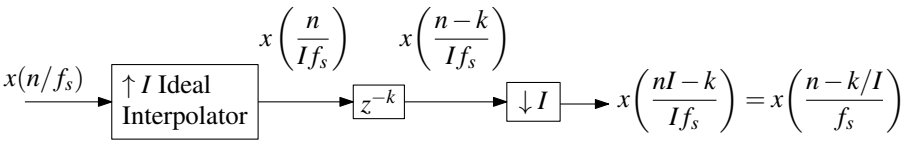


Figure 10.8. A discrete-time system for realizing a fractional delay.

Problems

10.1. The system shown in Figure 10.2 transforms the input sequence x_n into the output sequence $v(n/I)$, where $v(t)$ is defined in (10.1).

(a) Determine whether or not this system is linear.
(b) Determine whether or not this system is time invariant.
(c) Find the response to the unit impulse $x_n = \delta_n$.
(d) Determine whether or not this system is causal.

10.2. Consider the discrete-time system that downsamples by D; i.e., $(Ax)_n := x_{nD}$, where $D \geq 2$ is an integer.

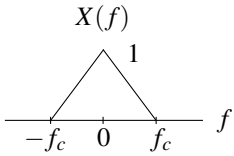
(a) Determine whether or not this system is linear.
(b) Determine whether or not this system is time invariant.
(c) Determine whether or not this system is causal.
(d) Determine whether or not this system is stable.

10.3. Let x_n have DTFT $X(f)$. Show that

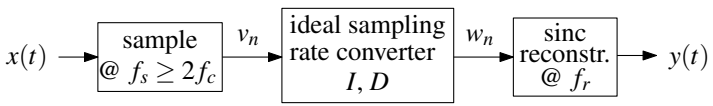
$$\sum_{k=0}^{D-1} X([f-k]/D)$$

as a function of f has period 1. *Hint:* Recall Section 3.1.5.

10.4. Consider the signal $x(t)$ with CTFT $X(f)$.

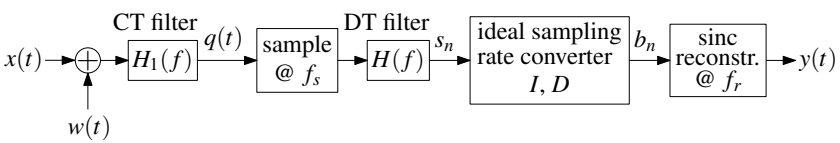


Suppose $x(t)$ is applied to the system

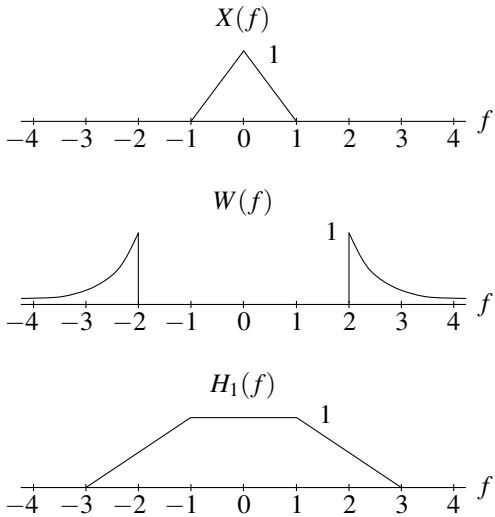


(a) Sketch the DTFT $V(f)$.
(b) Sketch the DTFT $W(f)$, assuming $D \leq If_s/(2f_c)$.
(c) Show that $Y(f) = X(f)$ if $D/I = f_s/f_r$.
(d) If the inequality in (b) holds, and if the equality in (c) holds, show that $f_r \geq 2f_c$.
(e) If $f_c = 7$, $f_s = 20$, and $f_r = 15$, find suitable values of I and D so that $y(t) = x(t)$. Do your choices satisfy $D \leq If_s/(2f_c)$?

10.5. Consider the system



where



Your goal is to make $y(t) = x(t)$. If $f_r = 4$ and $f_s = 6$, specify an ideal DT filter $H(f)$ (gains, passbands, and stopbands), I, and D so that $y(t) = x(t)$.

10.6. Repeat Problem 10.5 for $f_s = 8$, $f_s = 5$, and $f_s = 4$.

Bibliography

[1] R. B. Blackman and J. W. Tukey, *The Measurement of Power Spectra*. New York: Dover, 1958.

[2] L. P. Huelsman, *Active and Passive Analog Filter Design*. New York: McGraw-Hill, 1993.

[3] J. F. Kaiser, "Nonrecursive digital filter design using the I_0-Sinh window function," *Proc. 1974 IEEE Int. Symp. Circuits and Systems*, San Francisco, CA, pp. 20–23, April 1974.

[4] J. H. McClellan and T. W. Parks, "A personal history of the Parks–McClellan algorithm," *IEEE Signal Processing Mag.*, vol. 22, no. 2, pp. 82–86, Mar. 2005.

[5] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1999.

[6] T. W. Parks and J. H. McClellan, "Chebyshev approximation for nonrecursive digital filters with linear phase," *IEEE Trans. Circuit Theory*, vol. CT-19, no. 2, pp. 189–194, Mar. 1972.

[7] B. Porat, *A Course in Digital Signal Processing*. New York: Wiley, 1997.

[8] T. J. Rivlin, *An Introduction to the Approximation of Functions*. New York: Dover, 1981.

[9] D. Slepian, "On bandwidth," *Proc. IEEE*, vol. 64, no. 3, pp. 292–300, Mar. 1976.

Index

***Continuous-Time* Fourier Transform (CTFT)**

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft}\, dt$$

Inversion Formula

$$x(t) = \int_{-\infty}^{\infty} X(f) e^{j2\pi ft}\, df$$

$x(t)$	$X(f)$
$I_{[-T,T]}(t)$	$2T\,\mathrm{sinc}(2Tf)$
$2f_c\,\mathrm{sinc}(2f_c t)$	$I_{[-f_c,f_c]}(f)$
$(1-\lvert t\rvert/T)I_{[-T,T]}(t)$	$T\,\mathrm{sinc}^2(Tf)$
$f_c\,\mathrm{sinc}^2(f_c t)$	$(1-\lvert f\rvert/f_c)I_{[-f_c,f_c]}(f)$
$e^{-\lambda t}u(t)$	$\dfrac{1}{\lambda + j2\pi f}$
$e^{-\lambda\lvert t\rvert}$	$\dfrac{2\lambda}{\lambda^2+(2\pi f)^2}$
$\dfrac{\lambda}{\lambda^2+t^2}$	$\pi e^{-2\pi\lambda\lvert f\rvert}$
$e^{-(t/\sigma)^2/2}$	$\sqrt{2\pi}\,\sigma\, e^{-\sigma^2(2\pi f)^2/2}$
$1/(\pi t)$	$-j\,\mathrm{sgn}(f)$
$u(t)$	$(1/2)\delta(f)+1/(j2\pi f)$

Note. The indicator function $I_{[a,b]}(t) := 1$ for $a \le t \le b$ and $I_{[a,b]}(t) := 0$ otherwise. In particular, $u(t) := I_{[0,\infty)}(t)$ is the unit step function. Also, $\mathrm{sinc}(t) := [\sin(\pi t)]/(\pi t)$ for $t \neq 0$ and $\mathrm{sinc}(0) := 1$.

Discrete-Time Fourier Transform (DTFT)

$$X(f) = \sum_{n=-\infty}^{\infty} x_n e^{-j2\pi fn}$$

Inversion Formula

$$x_n = \int_{-1/2}^{1/2} X(f) e^{j2\pi fn}\, df$$

x_n	$X(f)$	
$e^{j2\pi f_0 n} I_{[0,N-1]}(n)$	$N \dfrac{\operatorname{sinc}\big(N(f-f_0)\big)}{\operatorname{sinc}(f-f_0)} e^{-j\pi(f-f_0)(N-1)}$	
$2f_c \operatorname{sinc}(2f_c n)$	$I_{[-f_c,f_c]}(f)$,	$\underline{\underline{\lvert f\rvert \le 1/2,\ 0 < f_c \le 1/2}}$
$2f_2 \operatorname{sinc}(2f_2 n) - 2f_1 \operatorname{sinc}(2f_1 n)$	$I_{[f_1,f_2]}(\lvert f\rvert)$,	$\underline{\underline{\lvert f\rvert \le 1/2,\ 0 \le f_1 < f_2 \le 1/2}}$
$e^{j2\pi f_0 n}$	$\displaystyle\sum_{k=-\infty}^{\infty} \delta(f-f_0-k)$,	(Dirac delta)
$a^n I_{[0,N-1]}(n)$	$\dfrac{1-(ae^{-j2\pi f})^N}{1-ae^{-j2\pi f}}$,	$\lvert a\rvert \neq 1$
$a^n u(n)$	$\dfrac{1}{1-ae^{-j2\pi f}}$,	$\lvert a\rvert < 1$
$a^{\lvert n\rvert}$	$\dfrac{1-a^2}{1-2a\cos(2\pi f)+a^2}$,	$\lvert a\rvert < 1$

Series Formulas

$$\sum_{k=0}^{N-1} z^k = \frac{1-z^N}{1-z}, \quad z \neq 1$$

$$\sum_{k=0}^{\infty} z^k = \frac{1}{1-z}, \quad |z| < 1$$

$$e^z := \sum_{k=0}^{\infty} \frac{z^k}{k!}$$

$$\lim_{n\to\infty} \left(1+\frac{z}{n}\right)^n = e^z$$

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$$

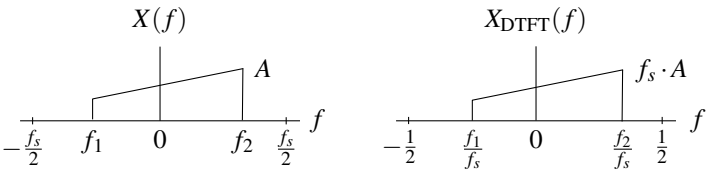
Relationship Between the CTFT and the DTFT

$$\frac{1}{f_s}X_{\mathrm{DTFT}}(f/f_s) = \sum_{n=-\infty}^{\infty} X(f-nf_s) =: \widetilde{X}(f) \qquad X_{\mathrm{DTFT}}(f) = f_s\widetilde{X}(f_s f).$$

If $x(t)$ is bandlimited to $f_c < f_s/2$, then $\widetilde{X}(f) = X(f)$ for $|f| \le f_s/2$,

$$x(t) = \sum_{m=-\infty}^{\infty} x(m/f_s)\,\mathrm{sinc}(f_s[t-m/f_s]),$$

and we have the graphical relationship



The z Transform

$$X(z) = \sum_{n=-\infty}^{\infty} x_n z^{-n}, \qquad \text{for } z \text{ such that } \sum_{n=-\infty}^{\infty} \left|x_n z^{-n}\right| < \infty.$$

x_n	$X(z)$	ROC
$a^n I_{[0,N-1]}(n)$	$\dfrac{1-(az^{-1})^N}{1-az^{-1}}$,	$z \neq a$
$a^n u(n)$	$\dfrac{1}{1-az^{-1}}$,	$\lvert z\rvert > \lvert a\rvert$
$a^n u(-n)$	$\dfrac{1}{1-a^{-1}z}$,	$\lvert z\rvert < \lvert a\rvert$
$a^n u(-n-1)$	$\dfrac{-1}{1-az^{-1}}$,	$\lvert z\rvert < \lvert a\rvert$
$a^{\lvert n\rvert}$	$\dfrac{1}{1-az^{-1}} + \dfrac{az}{1-az}$,	$\lvert a\rvert < \lvert z\rvert < 1/\lvert a\rvert$