Signal Processing for Communications
Winter Semester 2007 - 2008

Instructor: Prof. Suhas Diggavi
LICOS, EPFL

based on lecture notes by Suhas Diggavi (LICOS), Paolo Prandoni and Martin Vetterli (LCAV), EPFL

Contents

Chapter 0

Mathematical Prerequisites

0.1 Complex Numbers

A complex number is a number of the form $a + bi$, where a and b are real numbers, and i is the imaginary unit, with the property $i^2 = -1$. The real number a is called the real part of the complex number, and the real number b is the imaginary part of the complex number. When the imaginary part b is 0, the complex number is just the real number a.

For example, $3 + 2i$ is a complex number, with real part 3 and imaginary part 2. If $z = a+bi$, the real part (a) is denoted $Re(z)$, and the imaginary part (b) is denoted $Im(z)$.

Complex numbers can be added, subtracted, multiplied, and divided like real numbers, but they have additional elegant properties. For example, real numbers alone do not provide a solution for every polynomial algebraic equation with real coefficients, while complex numbers do (the fundamental theorem of algebra).

In some fields (in particular, electrical engineering and electronics, where i is a symbol for current), complex numbers are written as $a + bj$.

0.1.1 Operations on Complex Numbers

The set of all complex numbers is usually denoted as $\mathbb{C}$. The additions, subtractions, and multiplications of complex numbers follow the associative, commutative, and distributive laws of algebra. Combining the latter properties with the equation $i^2 = -1$, it is easy to see that:

$$(a + bi) + (c + di) = (a + c) + (b + d)i \tag{0.1}$$

$$(a + bi) - (c + di) = (a - c) + (b - d)i \tag{0.2}$$

$$(a + bi)(c + di) = ac + bci + adi + bdi^2 = (ac - bd) + (bc + ad)i \tag{0.3}$$

0.1.2 The Complex Number Field

Formally, the complex numbers can be defined as ordered pairs of real numbers (a, b) together with the operations:

$$(a, b) + (c, d) = (a + c, b + d) \tag{0.4}$$
$$(a, b) \cdot (c, d) = (ac - bd, bc + ad) \tag{0.5}$$

So defined, the complex numbers form a field, the complex number field, denoted by $\mathbb{C}$.

Since a complex number $a + bi$ is uniquely specified by an ordered pair (a, b) of real numbers, the complex numbers are in one-to-one correspondence with points on a plane, called the *complex plane*.

We identify the real number a with the complex number $(a, 0)$, and in this way the field of real numbers $\mathbb{R}$ becomes a subfield of $\mathbb{C}$. The imaginary unit i is the complex number $(0, 1)$.

In $\mathbb{C}$, we have:

- additive identity ("zero"): $(0, 0)$
- multiplicative identity ("one"): $(1, 0)$
- additive inverse of $(a, b) : (-a, -b)$
- multiplicative inverse (reciprocal) of non-zero (a, b): $\left(\frac{a}{a^2+b^2}, \frac{-b}{a^2+b^2}\right)$

0.1.3 The Complex Plane

A complex number $z = a + bi$ can be viewed as a point or a position vector on a two-dimensional Cartesian coordinate system called the complex plane. The Cartesian coordinates of the complex number are the real part a and the imaginary part b, while the polar coordinates are $r = |z|$, called the absolute value or modulus, and $\phi = arg(z)$, called the complex argument of z (mod-arg form). Together with Euler's formula we have (see figure 0.1)

$$z = a + bi = r(\cos\varphi + i\sin\varphi) = re^{i\varphi} \tag{0.6}$$

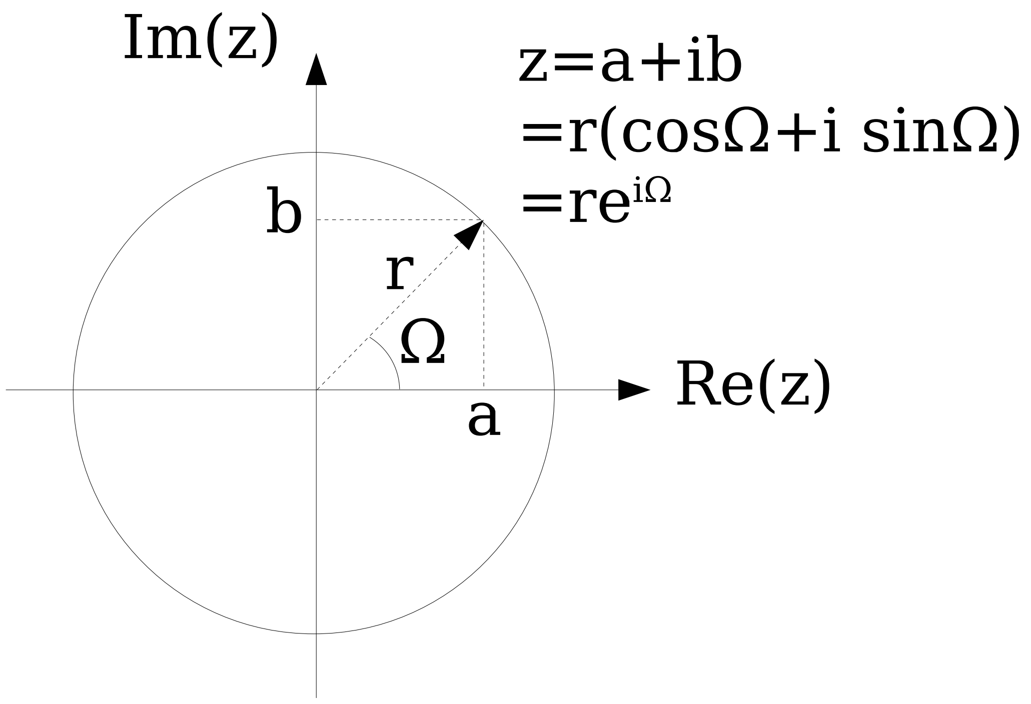


Figure 0.1: Complex plane representation

It is easy to see that:

$$r = |z| = \sqrt{a^2 + b^2} \tag{0.7}$$

$$\tan\varphi = \frac{b}{a} \tag{0.8}$$

$$\cos\varphi = \frac{e^{i\varphi} + e^{-i\varphi}}{2} \tag{0.9}$$

$$\sin\varphi = \frac{e^{i\varphi} - e^{-i\varphi}}{2i} \tag{0.10}$$

By simple trigonometric identities, we see that

$$r_1 e^{i\varphi_1} \cdot r_2 e^{i\varphi_2} = r_1 r_2 e^{i(\varphi_1 + \varphi_2)} \tag{0.11}$$

and that

$$\frac{r_1 e^{i\varphi_1}}{r_2 e^{i\varphi_2}} = \frac{r_1}{r_2} e^{i(\varphi_1 - \varphi_2)}. \tag{0.12}$$

This gives you an easy way to calculate the powers and the roots of complex numbers. Pay attention to the fact that $re^{i\varphi} = re^{i(\varphi+2k\pi)}$, $k \in \mathbb{Z}$, so $(re^{i\varphi})^{1/N} = r^{1/N}e^{i(\varphi+2k\pi)/N}$, $k = 0, 1, ..., N-1$.

Now the addition of two complex numbers is just the vector addition of two vectors, and the multiplication with a fixed complex number can be seen as a simultaneous rotation and stretching.

Multiplication with i corresponds to a counterclockwise rotation by 90 degrees ($\pi/2$ radians). The geometric content of the equation $i^2 = -1$ is that a sequence of two 90 degree rotations results in a 180 degree (π radians) rotation. Even the fact $(-1) \cdot (-1) = +1$ from arithmetic can be understood geometrically as the combination of two 180 degree rotations.

0.1.4 Absolute Value, Conjugation and Distance

One can check readily that the absolute value has three important properties:

$$|z| = 0, \qquad \text{if and only if } z = 0 \tag{0.13}$$

$$|z+w| \leq |z| + |w|, \qquad \text{(triangle inequality)} \tag{0.14}$$

$$|zw| = |z| \cdot |w| \tag{0.15}$$

for all complex numbers z and w. It then follows, for example, that $|1| = 1$ and $|z/w| = |z|/|w|$. By defining the distance function $d(z, w) = |z - w|$ we turn the complex numbers into a metric space and we can therefore talk about limits and continuity. The addition, subtraction, multiplication and division of complex numbers are then continuous operations. Unless anything else is said, this is always the metric used on the complex numbers.

The complex conjugate of the complex number $z = a+ib$ is defined to be $a-ib$, written as $\bar{z}$ or z^*. $\bar{z}$ is the "reflection" of z by the real axis. The following can be checked:

$$\overline{z+w} = \bar{z} + \bar{w} \tag{0.16}$$

$$\overline{zw} = \bar{z}\bar{w} \tag{0.17}$$

$$\overline{(z/w)} = \bar{z}/\bar{w} \tag{0.18}$$

$$\bar{\bar{z}} = z \tag{0.19}$$

$$\bar{z} = z \quad \text{if and only if } z \text{ is real} \tag{0.20}$$

$$|z| = |\bar{z}| \tag{0.21}$$

$$|z|^2 = z\bar{z} \tag{0.22}$$

$$z^{-1} = \bar{z}|z|^{-2} \quad \text{if } z \text{ is non-zero.} \tag{0.23}$$

The latter formula is the method of choice to compute the inverse of a complex number if it is given in rectangular coordinates.

That conjugation commutes with all the algebraic operations (and many functions, e.g., $\sin \bar{z} = \overline{\sin z}$) is rooted in the ambiguity in the choice of i (-1 has two square roots).

0.2 Summations

Let f be a function whose domain includes the integers from n through m. We define

$$\sum_{i=n}^{m} f(i) = f(n) + f(n+1) + ... + f(m) \tag{0.24}$$

We call i the index of summation, n is the lower limit of summation, and m is the upper limit of summation. One can show that:

$$\sum_{k=1}^{n} c = c + c + ... + c = cn \tag{0.25}$$

$$\sum_{k=1}^{n} k = 1 + 2 + ... + n = \frac{n(n+1)}{2} \tag{0.26}$$

$$\sum_{k=1}^{n} k^2 = 1 + 4 + ... + n^2 = \frac{n(n+1)(2n+1)}{6} \tag{0.27}$$

$$\tag{0.28}$$

Another well-known result is the following:

$$S_n = \sum_{k=0}^{n} r^k = 1 + r + r^2 + ... + r^n = \begin{cases} \frac{1-r^{n+1}}{1-r} & \text{if } r \neq 1 \\ n+1 & \text{else} \end{cases} \tag{0.29}$$

Note that when $r < 1$:

$$\lim_{n \to +\infty} S_n = \frac{1}{1-r} \tag{0.30}$$

Additionally, be very cautious when taking squares of summations:

$$\left[\sum_{i=n}^{m} f(i)\right]^2 = \left[\sum_{l=n}^{m} f(l)\right]\left[\sum_{k=n}^{m} f(k)\right] = \sum_{l=n}^{m}\sum_{k=n}^{m} f(l)f(k) \tag{0.31}$$

Finally, let $S_N = \sum_{n=1}^{N} a_n$ and $S = \lim_{N\to+\infty} S_N$. If the sequence of partial sums is divergent (i.e., either the limit does not exist or is infinite) then we call the series divergent.

If $|S| = c < \infty$, we call the series convergent and we call S the sum or value of the series. The Cauchy convergence criterion states that a series $\sum_{n=1}^{\infty} a_n$ converges if and only if the sequence of partial sums is a Cauchy sequence. This means that for every $\varepsilon > 0$, there is a positive integer N such that for all $n \geq m \geq N$ we have:

$$\left| \sum_{k=m}^{n} a_k \right| < \varepsilon \tag{0.32}$$

which is equivalent to

$$\lim_{\substack{n\to\infty \\ m\to\infty}} \sum_{k=n}^{n+m} a_k = 0 \tag{0.33}$$

0.3 Integration

Besides being comfortable with the basic properties of integrals and methods for integration (e.g. substitution, integration by parts) it is important to know the definition and basic properties of the convolution integral.

The convolution between two functions f and g, both with domain $\mathbb{R}$, is itself a function, let's call it h, and is defined by

$$h(x) := f(x) * g(x) = \int_{\mathbb{R}} f(y) g(x-y) dy.$$

The following properties are easy to show:

- $f(x) * g(x) = g(x) * f(x)$.
- $f(x) * (g(x) * h(x)) = (f(x) * g(x)) * h(x)$.
- $f(x) * (\alpha \cdot g(x) + \beta \cdot h(x)) = \alpha \cdot f(x) * g(x) + \beta \cdot f(x) * h(x)$.

0.4 Linear Algebra

0.4.1 Matrices

Let A be a matrix with n rows and m colums of complex entries. That is, we have

$$A := \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,m} \\ A_{2,1} & A_{2,2} & \dots & A_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,m} \end{bmatrix},$$

$A_{i,j} \in \mathbb{C}$.

One of the basic operations on A is taking the *transpose*, denoted by A^T and defined as $\left(A^T\right)_{i,j} = A_{j,i}$, $i = 1, \ldots, n$, $j = 1, \ldots, m$. More explicitely we have

$$A^T = \begin{bmatrix} A_{1,1} & A_{2,1} & \ldots & A_{n,1} \\ A_{1,2} & A_{2,2} & \ldots & A_{n,2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1,m} & A_{2,m} & \ldots & A_{m,n} \end{bmatrix}.$$

The *conjugate transpose* of A, denoted by A^*, is defined as $(A^*)_{i,j} = A^*_{j,i}$. Besides taking the transpose of A we take the complex conjugate of each element. Note that A^* is also known as the Hermitian of A.

Based on the above operations we define *symmetric* and *Hermitian* matrices. A real matrix A is symmetric if $A^T = A$, a (complex) matrix is Hermitian if $A^* = A$.

The matrix A can be right multiplied with a m by p matrix, say B, resulting in a n by p matrix. Remember that matrix multiplication is defined by $(AB)_{i,j} = \sum_{k=1}^m A_{i,k}B_{k,j}$. Note that **matrix multiplication is not commutative**, i.e. $AB \neq BA$ (assuming $n = m$).

0.4.2 Vectors

Let c and d be length n resp. m vectors, i.e., $c = [c_1, c_2, \ldots, c_n]$ and $d = [d_1, d_2, \ldots, d_m]$. We will usually assume that vectors are column vectors. This allows us to right multiply our matrix A with vector d. The result Ad is a length n vector. Similarly $b^T A$ gives a length m row vector.

The *inner product* between two n length vectors a and b is defined by

$$\langle a, b \rangle \ := \sum_{i=1}^n a_i b_i = a^T b.$$

Note that matrix multiplication is nothing more than taking inner products between rows and columns of the two matrices. The most common way to define the *norm* of a vector is through the inner product. This gives that the norm of x, $\|x\|_2$, is defined as

$$\|x\|_2 = \langle x, x \rangle^{1/2}.$$

A very useful relation is the *Cauchy-Schwartz inequality*, which states that

$$|\langle x, y \rangle| \leq \|x\|_2 \|y\|_2.$$

0.4.3 Determinants

One of the most used properties of a matrix is its determinant. The determinant of a 2 by 2 matrix

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

is given by

$$\det(A) = ad - bc.$$

In general for a square n by n matrix A we have, for any row $i = 1, \ldots, n$

$$\det(A) = \sum_{j=1}^{n} A_{i,j}(-1)^{i+j} \det\left(A^{\setminus(i,j)}\right),$$

where $A^{\setminus(i,j)}$ is the matrix resulting after removing row i and column j from matrix A. We can also expand along any column $j = 1, \ldots, m$, which gives us

$$\det(A) = \sum_{i=1}^{n} A_{i,j}(-1)^{i+j} \det\left(A^{\setminus(i,j)}\right).$$

An important result to keep in mind is that

a matrix is invertible if and only if its determinant is not equal to zero.

Finally, we note the following basic relations:

- $\det(AB) = \det(A)\det(B)$.
- $\det\left(A^{-1}\right) = \det\left(A\right)^{-1}$.
- $\det(A^*) = \det(A)^*$.

0.4.4 Eigenvalues and Eigenvectors

The *eigenvalues* of a matrix A are the solutions for λ in the equation

$$\det(A - \lambda I) = 0,$$

which is called the *characteristic equation.* Given that $\tilde{\lambda}$ is an eigenvalue of A, we call the vector x for which

$$Ax = \tilde{\lambda} x$$

the *eigenvector* corresponding to $\tilde{\lambda}$.

One can verify that the eigenvalues of A and A^* are the same. On the other hand, the eigenvectors of A and A^* are different.

0.5 Problems

Problem 0.1 *1. Let $s[n] := \frac{1}{2^n} + j\frac{1}{3^n}$. Compute $\sum_{n=1}^{\infty} s[n]$.*

2. Same question with $s[n] := (\frac{j}{3})^n$.

3. Characterize the set of complex numbers satisfying $z^* = z^{-1}$.

4. Find 3 complex numbers $\{z_0, z_1, z_2\}$ which satisfy $z_i^3 = 1$, $i = 1, 2, 3$.

5. What is the following infinite product $\prod_{n=1}^{\infty} e^{j\pi/2^n}$?

Problem 0.2 (Geometric Series) *Consider the sequence $x[n] = a \cdot r^n$ for some real r. Let $S[n] = \sum_{k=0}^{n} x[k]$.*

(a) The goal is to find a closed form expression for $S[n]$.

- *Compare the two sequences $S[n]$ and $S[n+1]$.*
- *Multiply each term in $S[n]$ by r and obtain $\tilde{S}[n] = rS[n]$. Compare $\tilde{S}[n]$ with $S[n+1]$.*
- *We have obtained a system of two equations in the unknowns $S[n]$ and $S[n+1]$. Solve this system and express $S[n]$ in terms of a, r and n.*

(b) Let $|r| < 1$. Find $S[n]$ when n goes to infinity, i.e., *$S = \sum_{k=0}^{\infty} x[k]$.*

(c) Find an expression for the summation $\sum_{k=n+1}^{m} x[k]$.

(d) Apply the obtained formula and compute $\sum_{k=0}^{\infty} t[k]$, where $t[k] = \frac{1}{3^k} + (\frac{1}{2j})^k$.

(e) How can we use this formula to compute $\Pi_{n=1}^{\infty} e^{j\pi/2^n}$?

Problem 0.3 (Complex Numbers) *(a) Find all the roots of $x^3 + 2x^2 + 2x + 1 = 0$. What is the summation of the roots?*
Hint: *Try small integers to find one of the roots and then solve the remaining degree 2 polynomial.*

(b) Compute j^j, where $j = \sqrt{-1}$.

(c) Consider the polar representation of the complex numbers and find all which satisfy $\arg(z) = |z|$. *(Note that $0 \le \arg(z) \le 2\pi$.)*

(d) Characterize the set of complex numbers satisfying $z^* = z^{-1}$.

Problem 0.4 (Linear Algebra) *(a) Compute the determinant of the following matrix.*

$$A = \begin{bmatrix} 2 & 0 & -1 & 0 \\ 1 & 0 & 2 & 1 \\ 0 & 0 & 2 & 1 \\ -1 & -3 & 2 & 0 \end{bmatrix}.$$

(b) Consider the matrices

$$B = \begin{bmatrix} j & -1 & 4 \\ 0 & 2-3j & 1 \\ -1 & 2j & 0 \\ 3 & 0 & 4-j \end{bmatrix} \qquad \text{and} \qquad C = \begin{bmatrix} 0 & 0 & j & 1 \\ 1-5j & 1 & 4j & 2+2j \\ 1 & 3-j & 0 & -7 \end{bmatrix}.$$

Which of the following operations are well-defined? (Note that you do NOT have to compute.) $C+B$, $C \cdot A^{-1}$, $B \cdot C$, $A - C$, $B + B^T$, $A + A^T$, $C^{-1} \cdot B^{-1}$, $C^* + B$,

(c) Let $x = [1,\ 2j,\ 1+j,\ 0]$. *Compute* Ax^T *and* xB.

(d) Compute the determinant of $D = xx^T$ *and* $E = x^T x$.

Chapter 1

What Is Signal Processing ?

1.1 Introduction

As implied by the name, signal processing deals with signals on the one hand, and operations on signals on the other hand. That is, the "black box" view of signal processing is as shown in this scheme:

$$\begin{tikzcd}
\text{Input Signal} \arrow[r] & \boxed{\text{Processing}} \arrow[r] & \text{Output Signal}
\end{tikzcd}$$

The purpose of these notes is to help you understand what a signal is, how it relates to the world around us, and how we can manipulate it. That is, we want to understand what to put inside the "black box" of the previous figure in order to be able to treat relevant signals from the real world, and produce output signals as required by applications, in particular in the context of communication systems.

As stated above, the realm of signal processing might seem too vague, or too all-encompassing. Any physical quantity evolving over time or space qualifies as a signal, and any possible computation performed on such a signal is a signal processing operation. That is, in many areas of applied sciences and engineering, people run what are essentially "signal processing" algorithms, not always knowingly. Or, to paraphrase Molière's Monsieur Jourdain, many people "font du traitement du signal sans le savoir". Our aim will not be to claim as large a field as possible, but to clearly specify that there are many other possible applications beyond those which we will study in detail; in our case, the focus will be mostly on telecommunications and related fields in electrical engineering and computer sciences. Whenever relevant, we will also try to illustrate the overlapping between signal processing and other scientific disciplines such as acoustics, statistics, geophysics, applied and computational harmonic analysis, and so on.

The outline of this introductory chapter is as follows. First, we will showcase a “gallery of signals”, from the floods of the Nile to the stock market, in order to point out the common traits and the differences between various signals. Next we will describe a series of prototypical “black box” systems, ranging from the very simple to the very complicated, all of which perform some meaningful signal processing task. We will then discuss the idea of an “underlying model” for a signal, which can be put to advantage in the design of signal processing algorithms. We will then sketch a very brief history of modern signal processing, highlighting the main achievements and their impact. Finally we will conclude the chapter with an overview of the structure of these notes. But before moving on, we need to introduce some initial, elementary technical concepts, which will then be refined along the way.

1.2 Elementary Concepts

Let $\mathbb{Z}, \mathbb{R}, \mathbb{C}$ denote the set of integer, real, and complex numbers respectively. Most real world signals can be modeled as real or complex functions of one or more real arguments. Signals of this type are called *continuous-time signals* and will be indicated by the familiar lowercase notation used for real functions, e.g. $f(t)$, $x(t)$, and so on. Signals which are functions of a single real argument are called one-dimensional signals. In this case the real argument usually represents time, while the signal itself is the evolution of a given quantity (often an electrical signal) over time signals which are functions of two arguments are called two-dimensional signals, and the two arguments usually represent a coordinate pair in the plane; this is for instance the case of a signal representing an image. For most continuous-time signals considered as functions on the real line, the Fourier transform (as defined in the standard calculus course) is a well defined operator; we will indicate the Fourier transform of a signal $f(t)$ as $F(j\Omega)$.

A discrete-time signal is a sequence of numbers (real or complex) indexed by one or more integer arguments. Again, a lower case letter will represent a given sequence but, in order to make explicit the discrete nature of the argument, the latter is enclosed in square brackets: $f[n]$, $x[n]$, and so on. This somewhat unusual notation is actually quite standard in the signal processing literature. Now, discrete-time signals are very often obtained from “sampling” a continuous-time signal. By sampling, we mean taking the value of a signal at regular intervals, or integer multiples of a sampling period T. This process will be studied in detail in Chapter 10, but we shall note right away that sampling a continuous-time signal $f(t)$ leads to a discrete-time signal $f[n]$ (please note the round parentheses and the square brackets):

$$f[n] = f(nT) \quad n \in \mathbb{Z} \tag{1.1}$$

where T is a real number greater than 0. A key question is of course whether we can recover $f(t)$ from its samples $f[n]$, a question we will settle in the aforementioned Chapter 10. A Fourier representation indeed exists for most discrete-time signals as well, and its study will be the subject of Chapter 3. In general, the Fourier transform of a discrete-time sequence $f[n]$ will be indicated by $F(e^{j\omega})$.

Sometimes, one considers a *finite-length* signal, with indexes $0 \ldots N-1$, where N denotes the length of the signal. It is then natural to consider such a signal as a vector in $\mathbb{R}^N$ or $\mathbb{C}^N$, the N-dimensional real and complex Euclidean spaces. In such a case, we will use linear algebra notation, that is, vectors will be denoted by lower case bold letters, e.g. $\mathbf{f}, \mathbf{x}$, etc. Linear transforms are given by matrices, denoted by bold upper case letters, e.g. $\mathbf{M}$. For example, a finite-dimensional version of the Fourier transform, called the discrete Fourier transform (DFT) and discussed in Section 3.3, will map an N-dimensional "signal" $\mathbf{x}$ into its N-dimensional transform $\mathbf{y}$ using a special matrix $\mathbf{W}$:

$$\mathbf{X} = \mathbf{W} \cdot \mathbf{x} \tag{1.2}$$

Note that the Fourier transform is also denoted by bold upper case letters, and the difference will be clear from the context. This concludes our elementary notations, where we have introduced three types of objects central to signal processing. These types are summarized in Table 1.1, which will be our rudimentary road map for the rest of the chapter.

Signal Type	**Time-domain Notation**	**Frequency-domain Notation**
Continuous-time	$x(t), \quad t \in \mathbb{R}$	$X(j\Omega)$
Discrete-time	$x[n], \quad n \in \mathbb{N}$	$X(e^{j\omega})$
Finite-length	$\mathbf{x}, \quad \mathbf{x} \in \mathbb{C}^N$	$\mathbf{X} = \mathbf{W}\mathbf{x}$

Table 1.1: Fundamental signal types.

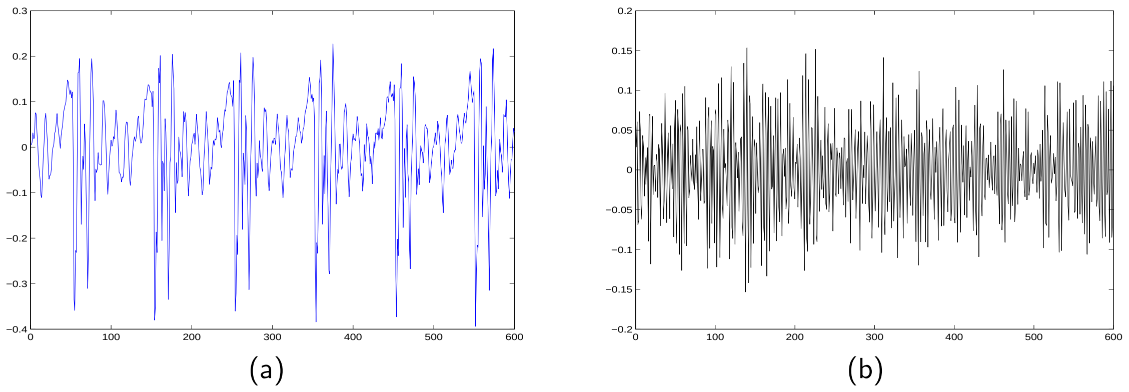


Figure 1.1: Speech signals. (a) Voiced sound, corresponding to the sound "a", (b) Unvoiced sound corresponding to "shh...".

1.3 Examples of Signals

We will now present a series of signals drawn from our everyday experience. Most of these are signals which are "processed" by our senses (hearing, vision). Others are more abstract and represent our analysis of real-world phenomena.

1.3.1 Speech

For a human being, the most natural signal is depicted in Figure 1.1. It is a speech signal in two of its typical forms, i.e. *voiced* speech in part (a) and *unvoiced* speech in part (b). It is fairly obvious from the picture that voiced sounds look almost periodic, while unvoiced speech possesses a noise-like character. One fact is immediately obvious from Figure 1.1: most of us would not recognize the vowel "a" or the sound "sh" from the plot of the signal, yet all of us will instantly recognize the sounds when played to our ears. Welcome to signal processing! A task which is trivial for the human ear and its attached processor (the brain) becomes a very hard problem for a computer. In particular, computer recognition of continuous speech (as opposed to isolated words like digits for example) from an unknown speaker (that is, without training tuned for a particular speaker) is still an open problem, even though a child can master it easily. Speech has fascinated signal processing researchers for decades, and due to its obvious economical importance, vast amounts of work have gone into "understanding" the speech signal; the two main lines of research involve studying speech production, i.e. how humans produce speech signals, and speech recognition, i.e. how humans perceive and analyze speech.

Speech can clearly be modeled as a continuous-time signal, corresponding to sound waves produced by the vocal cords and filtered by the trachea and mouth. Yet, most of speech processing (except in "old" analog telephony) is done in discrete-time, typically at a sampling rate of 8kHz, or a sampling interval of $T = 0.125$ ms. As we will see, such sampling does not impair the signal very much, and is thus standard in "digital" telephony.

1.3.2 Music

Next to speech, musical sounds are the most ancient signals produced by humans. Figure 1.2 shows several examples of signals produced by musical instruments, ranging from the "simplest" to the most "complex". As quite obvious from looking at the picture (and even more so by listening to the sound!), there is a wide variety of musical sounds, from the simple, sinusoidal flute to the polyphonic complexity of a full orchestra. Music through the ages has been about synthesizing interesting sounds, and signal processing has added many new "instruments" to the toolbox in the last decades. From vocoders (to help aspiring singers stay in tune...) to full blown synthesizers able to imitate almost any instrument to near perfection, examples abound where sophisticated signal processing techniques create "new sounds". Most are based on prior fine analysis of actual instruments, in order to best imitate them, while others are completely "artificial". Depending on one's musical taste, some of the achievements have been a mixed blessing for music. A few examples are given in Figure 1.3.

Besides producing music, signal processing is also involved in the more mundane task of recording, storage, transmission, etc. Again, while the signal is a sound wave, a discrete-time version with sampling at 44 KHz is most often used, with little degradation.

1.3.3 Other One-Dimensional Signals

So far, we have looked at the two most natural signals, namely speech and music. Many other examples are possible, and we will pick a few outstanding examples. Figure 1.4 shows two key signals for humans, namely the electro-cardiogram, and the electro-encephalogram. These indicate heart and brain activity, respectively. Such signals have been studied by doctors for decades in order to monitor illness, or predict future problems. Continuous monitoring by means of automatic analysis is becoming a reality as processing has become sophisticated and reliability has increased (e.g. automatic defibrillation devices for heart patients).

Mother nature is a great producer of interesting signals, from seismic activity to solar spots. A famous example is what is considered to be on many accounts the *oldest discrete-*

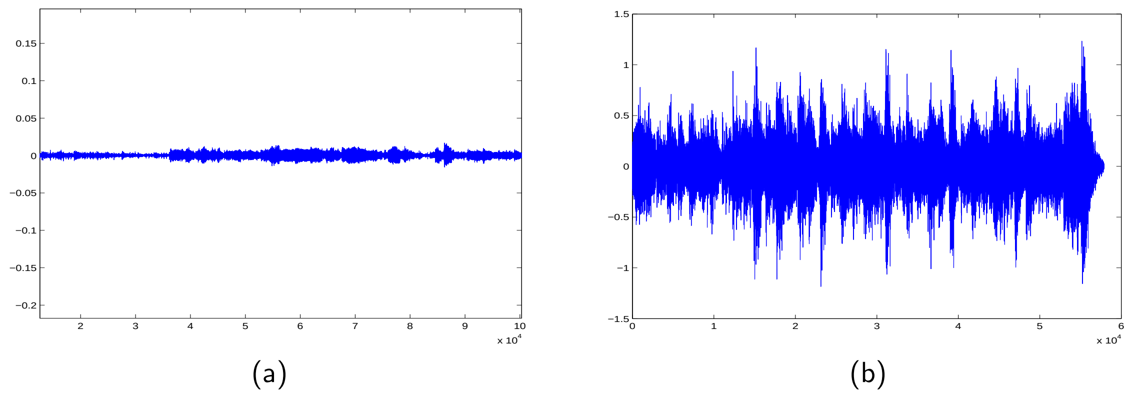
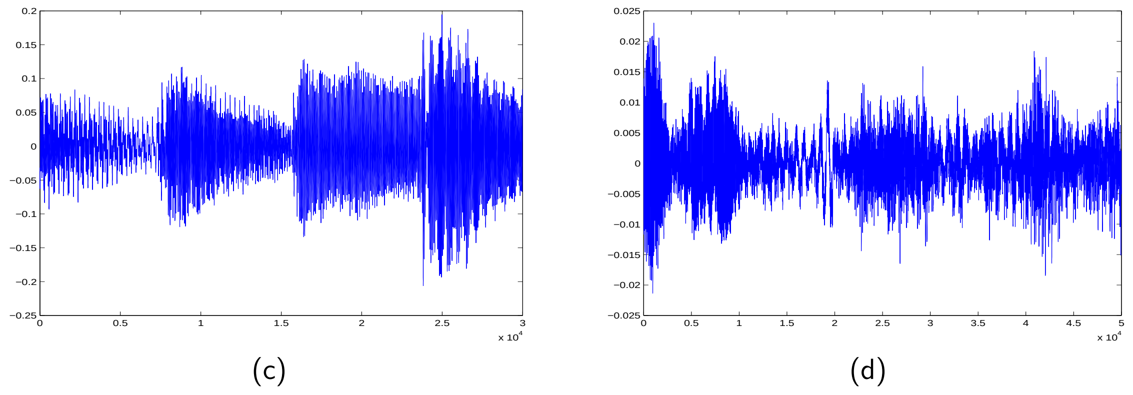



Figure 1.2: Examples of musical sounds. (a) Flute (beginning of Ravel's Bolero), (b) Full orchestra (ending of Ravel's Bolero), (c) Piano (from Bach's Goldberg Variations), (d) String quartet (from Schubert's Op.29).

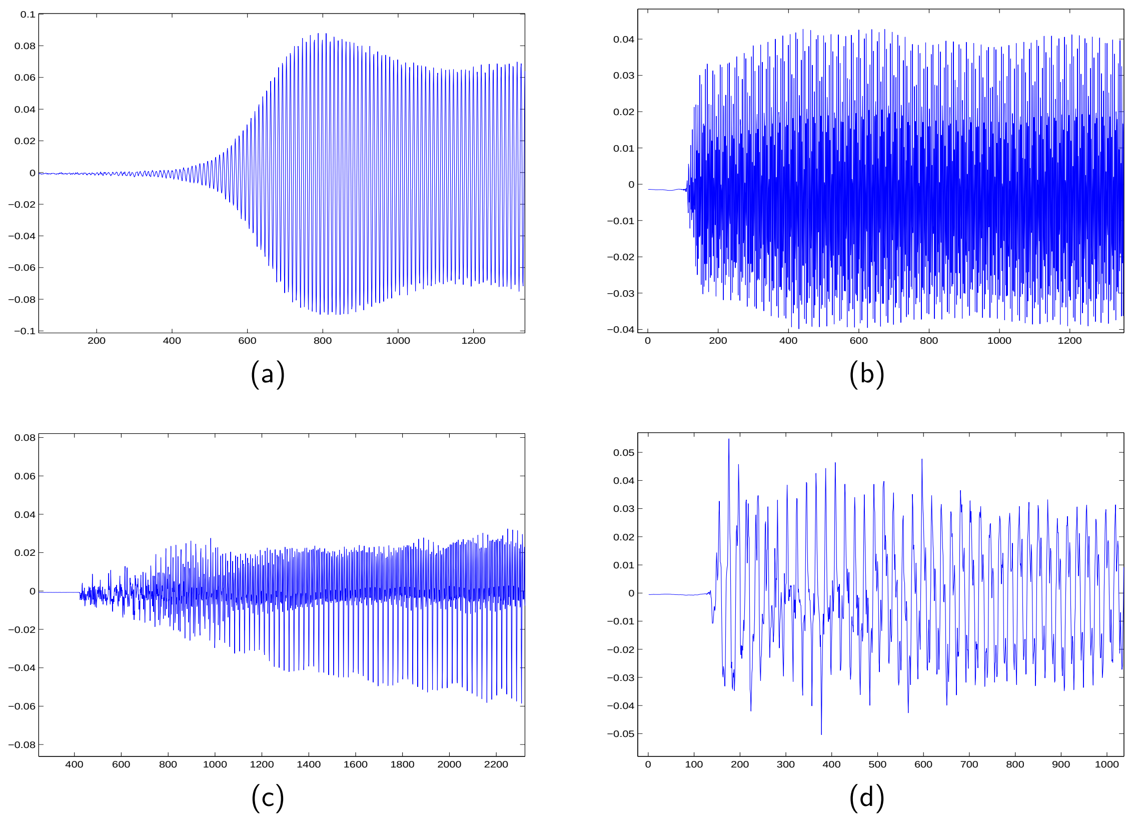


Figure 1.3: Examples of musical sounds from synthetic instruments. (a) Flute (with additive synthesis), (b) Trumpet (with FM modulation), (c) Violin (with physical modeling), (d) Piano (with digital waveguide).

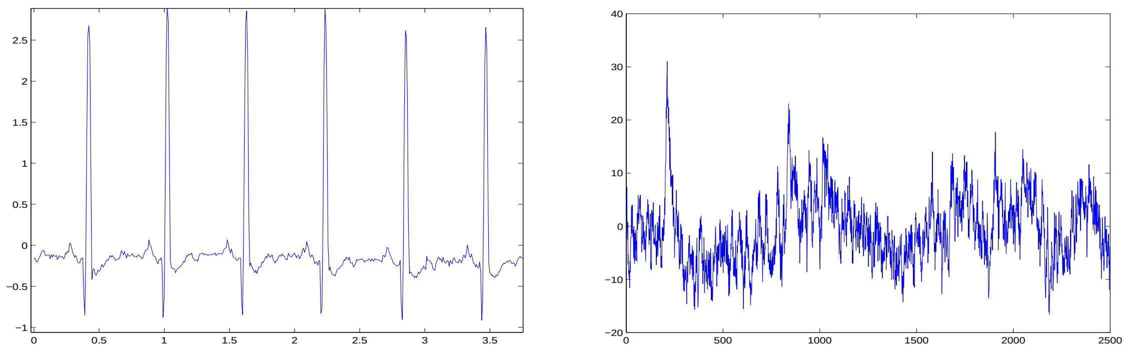


Figure 1.4: Medical signals of vital importance.
(a) Electro-cardiogram (ECG), (b) Electro-encephalogram (EEG).

time signal on record. It is actually a time-series, with a sampling interval of one year which, amongst other things, indicates the height of the floods of the Nile in ancient Egypt from 2925 BC to 2325 BC. The floods of the Nile were studied in modern times by Hurst in order to spot any regularity. Instead, he found the “Hurst parameter”, a fractal measure of long-range dependence that is used today in the analysis of internet traffic; see Figure 1.5 for examples of both.

A famous time series that has obsessed many people is the stock market index, which is known for its trends but also its unpredictability due to abrupt changes (e.g. 1929...). The million dollar question is: What is the market value tomorrow? Yet another view of human activity is at the center of an intense scientific and political controversy. The signal at the heart of the debate is fairly simple, since it is the measure of the temperature on planet earth over the last hundred years or so. But there is much more at stake than in the stock market question: is there a global warming phenomenon, which could wipe out civilization as we know it? You can try your guess on Figure 1.6!

1.3.4 Images

So far, all signals we considered were functions of a single variable, typically time. If we look at signals of two variables, say $f(x,y)$ or $f[n,m]$, we get in the realm of image processing. Of course, long before the advent of photography and image processing, humans had “projected” images of the real world onto a two-dimensional surface, from Lascaux

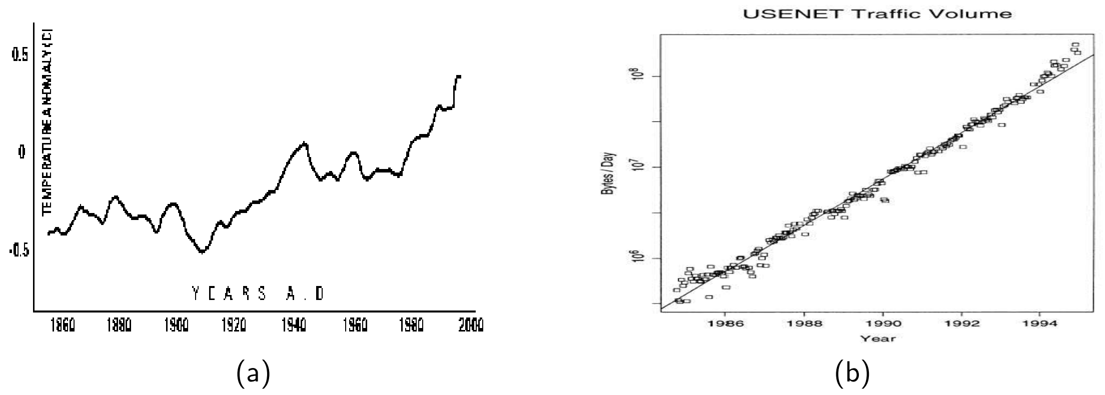


Figure 1.5: Long range dependent time-series. (a) The floods of the Nile in modern times, (b) Internet traffic.

to Giza and more. Yet the modern age started with the invention of photography in the 19th century, which soon produced "scientific" images of the world around us.

A few particularities should be noted about images. Typically, images are of finite extent, whereas many one-dimensional signals had infinite length (ignoring the big-bang for a moment, and hoping the best for the future...). Images are either black and white ($f(x, y)$ is real and positive) or color ($f(x, y)$ is a vector function depending on a color space). Finally, until recently, images were continuous in the space dimension, whereas now, many images are "digital" right from the start. An image in digital form is typically an array of 512 by 512 or 1024 by 1024 picture elements (pixels). It is to be noted that a high quality photograph on chemical film is of much higher quality, even though the gap is constantly narrowing.

1.3.5 Other Types of Images

Besides "classic" photographs, many other types of images are possible. For example, in the medical field, X-ray pictures and ultrasound images are very common, examples of which are shown in Figure 1.7. Clearly, such images have very different characteristics from "natural" images, a fact that obviously will influence the signal processing techniques used on such data.

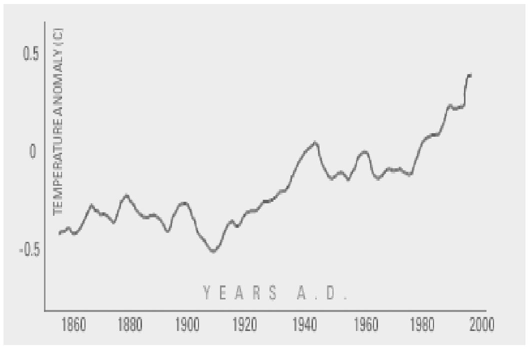


Figure 1.6: A vital signal about the health of planet earth: temperature evolution from 1830 to 2000.

1.3.6 Higher-Dimensional Signals

Stepping up from two to three dimensions, the dominant signal is certain video and film, that is, "moving pictures". An important point to notice right away is that in such three-dimensional data, the time dimension is always discrete, e.g. 24 frames/sec in movies, 50 or 60 fields/sec in television. This is shown schematically in Figure 1.8.

Moving pictures have fascinated viewers ever since their invention late in the 19th century. Because of the sampling in the time dimension, they are also one of the first examples of a sampled data system. This sampling can also cause some artifacts that are well known to fans of "western" movies (among others). It is the famous backward

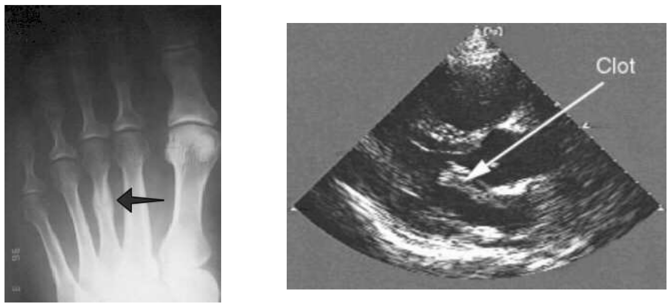


Figure 1.7: Medical images. (a) X-ray image (details), (b) Ultrasound image (details).

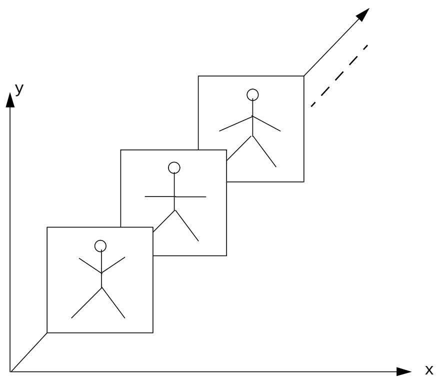


Figure 1.8: Three dimensional signal as in film or video. While the spatial dimension x and y can be continuous (film) or partly discrete (video where there is a line structure) the time dimension is always discrete (given by the number of images per second that are captured).

turning wagon-wheel effect, shown schematically in Figure 1.9. This is an example of aliasing, a phenomenon typical of sampling. The reason for the “visual illusion” of the wheel turning backward is that there are several possible continuous time events that map to the same sampled sequence, and the human eye will pick out the most likely. In the example above, let’s say that the wheel turns clockwise by $3\pi/4$ between each sampling instant. The sampled version is as in Figure 1.9. However, because of symmetry, the same wheel turning backward (or counter clockwise) by $-\pi/4$ leads to the same sampled sequence. This motion being smaller, it will be the ”most likely” explanation that the viewer will see. The issue of aliasing in sampling will be studied in detail in Chapter 10.

While film and video is the most common and visible three-dimensional data set, there are many other examples of such signals, e.g. geophysical data (representation of the earth interior for oil exploration purposes), tomographic data (interior of the human body for medical analysis), etc. Finally, let us mention an example of a four-dimensional data set, used in medical signal processing. While its importance is clear, its acquisition is very difficult: it is the “image” of the beating heart. That is, using tomographic techniques, one reconstructs a three-dimensional image of the heart, and this over time.

In conclusion for this section on signals, let us simply remark that signals are everywhere. Wherever you look, listen or sense, signals are to be found. Humans are very good at processing signals for which they are equipped. But there are also many key signals

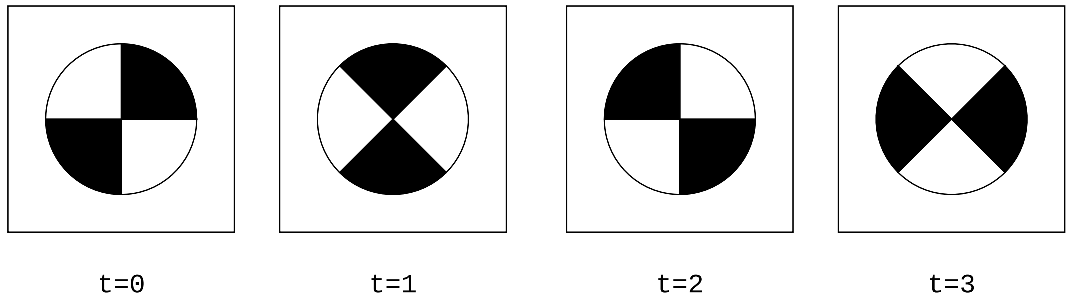


Figure 1.9: The wagonwheel "illusion". Is this wheel turning forward or backward?

beyond the “natural” ones, beyond the reach of humans. And all of these signals are of interest to signal processing systems.

1.4 Systems

A system, in our view of the world, is a box that takes a signal in, and produces an output, typically another signal. Such a general view, applied to the many signals we have seen, produces a wealth of possible signal processing systems. Instead of an exhaustive list, we will pick a few examples that are emblematic signal processing tasks. A “naive” picture of such systems is shown in Figure 1.10, where a given signal is transformed into a desired signal.

1.4.1 Speech Recognition

Given a speech signal, a speech recognition system tries to “understand” the words as a human would do. This seemingly elementary task (as seen from a human perspective) is actually dauntingly difficult for a computer in its full generality. While recognizing individual elementary sounds spoken by a known speaker is easy, understanding a continuous stream of speech by an unknown speaker is still not solved satisfactorily as of today. Signal processing plays a key role in the first stages of a speech recognition system, that is in pre-processing (e.g. creating a compact representation of the speech waveform, for example through a local analysis of the spectrum) and in the modelization of speech (e.g. linear predictive models, hidden Markov models). After such signal processing based

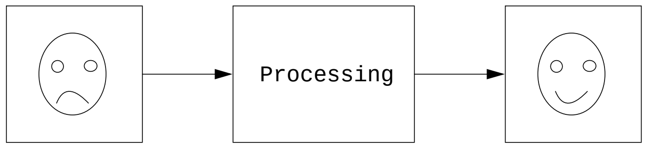


Figure 1.10: The "naive" view of a signal processing task: transforming a given signal (here a picture) into a desired signal.

pre-processing, higher level methods are used (e.g. grammatical models for the structure of sentences). Speech recognition is even more complex in real environments, e.g. when noise is involved, as in a car for example.

1.4.2 Denoising

Very often, instead of getting the signal we want, we get a signal corrupted by noise. An obvious signal processing task is therefore to clean out the noise as well as possible without "damaging" the signal. To solve this problem, we need a model for the signal and the noise, so as to best estimate the signal given the observed signal. One such simple model is the additive noise model, where the noise is assumed to be independent of the signal, as shown in Figure 1.11. Many methods exist for attacking this problem, from filtering methods to non-linear denoising algorithms.

1.4.3 Inverse Problems

Numerous signal processing problems belong to the class of *inverse problems*. A generic example, which is also quite intuitive, is the following: assume you take a picture with your camera, but unfortunately, you take it out of focus. The result is a blurred image, and you would like to undo the blur and recover a sharp picture[1]. If and how to solve this is a typical inverse problem. In our example, the blurring operator is typically singular (which means certain components of the original image are forever lost) and thus its inverse is badly behaved (or ill-conditioned). Fig. 1.12 shows schematically the situation,

[1]You may think that only total amateurs would run into such problems, but actually professionals can run into similar problems: NASA set up a rather expensive space telescope called Hubble, which was unfortunately "out of focus".

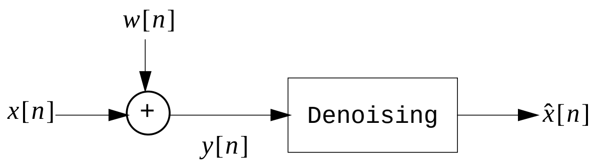


Figure 1.11: A signal $x[n]$ is corrupted by independent additive noise $w[n]$, followed by a denoising algorithm that produces an estimate $\hat{x}[n]$ given the observed signal $y[n] = x[n] + w[n]$.

where the inverse of the blur operator needs to be "regularized" so it is well-conditioned. There are many other instances of such inverse problems, like for example tomographic reconstruction in medical imaging or equalization for communication channels.

An additional reason why inverse problems are difficult is that very often, noise is present. That is, in our scheme shown in Figure 1.12, instead of y, we get $y + w$ where w is some noise signal. In that case, the inverse of the blur function can *amplify* the noise, and so, while the result might be sharper, it may also be very noisy. Then, combined "deblurring" and "denoising" is needed, a much more complex task.

1.4.4 Decision Systems

In many cases, a system takes a signal as its input, but produces just a binary output. For example, in the electrocardiogram case shown in Figure 1.7, a monitoring system simply needs to decide if the patient is healthy or not, but obviously, such a decision can be a matter of life or death. Similarly, a system monitoring the stock market index needs only to decide on buying or selling a particular stock. The characteristic of such systems is that huge amounts of data are available, and all of it might be relevant to take the right decision. Such systems typically analyze time-series, and are thus under study in that particular field of statistics. Yet, similar problems exist in communication systems, where for example particular waveforms have to be detected, but are typically buried in noise.

1.4.5 Compression Systems

For storage and communication purposes, signals need to be represented by binary digits. That is, a discrete-time signal with real values $x[n] \in \mathbb{R}$ needs to be represented by a finite

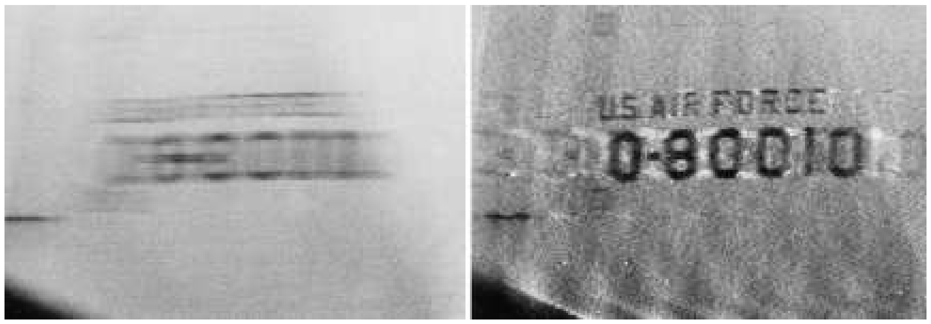


Figure 1.12: Example of an inverse problem. The original image (on the left) is blurred by the acquisition procedure; the image on the right is obtained digitally by "inverting" the blur operator (© Los Alamos National Laboratory)

precision approximation so as to be representable by a binary number. For example, the voice samples are typically approximated using an 8 bit number, i.e 256 different values. This seems coarse, but is normally sufficient. But, beyond such simple sample by sample approximation, compression systems try to remove as much redundancy as possible from a given sampled and quantized representation. For speech, an original stream of 64 kbits/sec (corresponding to 8000 samples per sec.), each with an 8 bit representation) can be "compressed" to 8 kbits/sec, or even down to 2.4 kbits/sec, using sophisticated representation methods. A block diagram of such a compression system is shown in Figure 1.13.

Other well known compression systems are used for digital audio and video and such compression methods are key in all digital applications, from multi-media on CD-ROM's to video over the internet. The ubiquitous MP3 audio format, for example, is a sophisticated compression scheme which exploits a *perceptual model* of human hearing together with highly optimized quantization techniques. The perceptual model analyzes the audio input and determines which portions of the signal cannot be heard anyway due to masking effects (masking occurs when a strong spectral component "saturates" the ear around its frequency location, thereby making nearby components inaudible). Furthermore, the number of bits allotted to quantization is a time-varying quantity, determined so as to push the quantization noise below the masking threshold for the signal under analysis.

Image compression, on the other hand, exploits the high spatial redundancy of digital pictures and the fact that the eye is more sensitive to sharp edges than to color gradients. In the JPEG compression standard, the image is divided into a grid of square blocks and each block is processed individually. In the MPEG video compression standard (and in its derivative, DiVX), the former approach is complemented by a sophisticated prediction

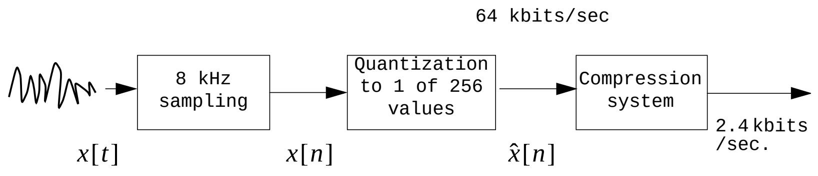


Figure 1.13: The original, continuous-time speech signal is first sampled at 8kHz, leading to a discrete-time signal $x[n]$. Each sample value is then approximated by 1 out of 256 values, and thus represented by an 8 bit number. This 64 kbits/sec digital stream is used in a complex compression system, that creates an approximate representation using only 2.4 kbits/sec.

mechanism called *motion compensation*, so that the correlation between neighboring blocks in successive video frames is exploited to reduce the number of bits used to encode each image in the sequence of frames.

1.4.6 A Communication Systems Example

As an example of the ubiquity of signal processing in communication systems, consider Figure 1.14. Depicted is an interconnected system of networks of different kinds, with many different services that utilize signal processing in one way or another. As can be seen, signal processing is an enabling technology for communication systems, since it sits at the heart of the communication links (e.g. equalization, modulation), as well as at the heart of many applications, from voice to image and video communication, but also medical applications, multimedia databases, etc.

1.5 World Models

In many cases, we have prior knowledge about the signal we are processing or else, we can acquire knowledge about the signal as we process it. In both cases, there is a notion of a model behind the signal, and having good models for given signals is at the heart of signal processing. This leads to model based processing as shown in Figure 1.15. To be more specific consider a speech processing application. Speech (see Figure 1.1) is a very partic-

.

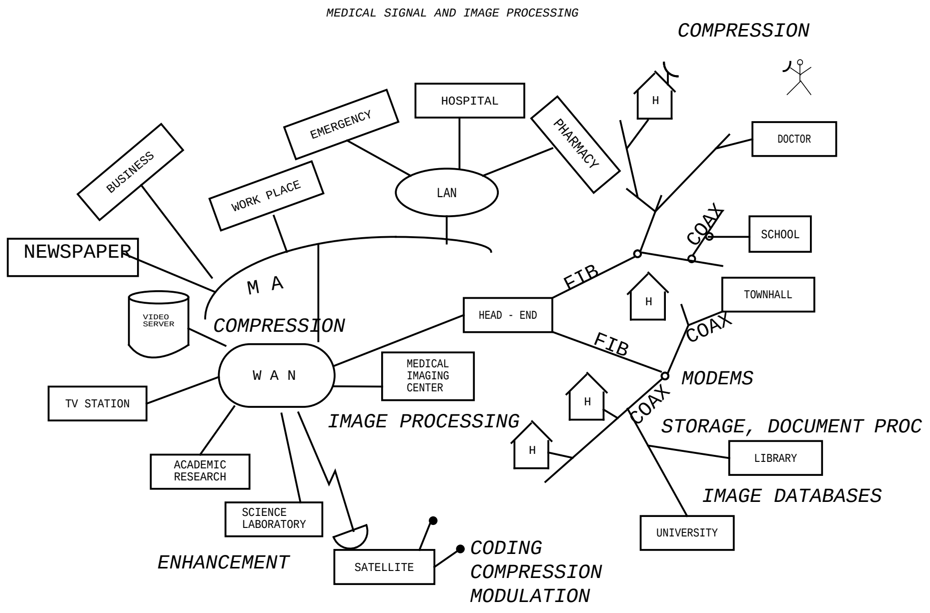


Figure 1.14: Signal processing in communication systems. In an interconnected world, with many applications using signals, signal processing is ubiquitous.

ular signal. It is usually produced by humans (we ignore for the moment talking parrots) and the speech production system is very well understood. Roughly speaking, speech is either voiced (in which case it has a harmonic or almost periodic structure) or unvoiced (that is noise-like). On top of this basic structure, the trachea, mouth, lip and nose filter the signal, producing a spectral shaping. Thus, an elementary speech production model is as shown in Figure 1.16. Now, any speech processing task can be helped by referring to this model. In speech recognition, determining the voiced/unvoiced nature is a key task, as is the recovery of the fundamental frequency of the voiced part and the spectral shaping parameters. All these parameters feed into a "pattern matching" algorithm that performs the actual recognition. In speech compression, recovering the fundamental parameters like voiced/unvoiced nature, pitch period, and spectral parameters leads to a very efficient representation, much more so than trying to approximate the original waveform. Finally,

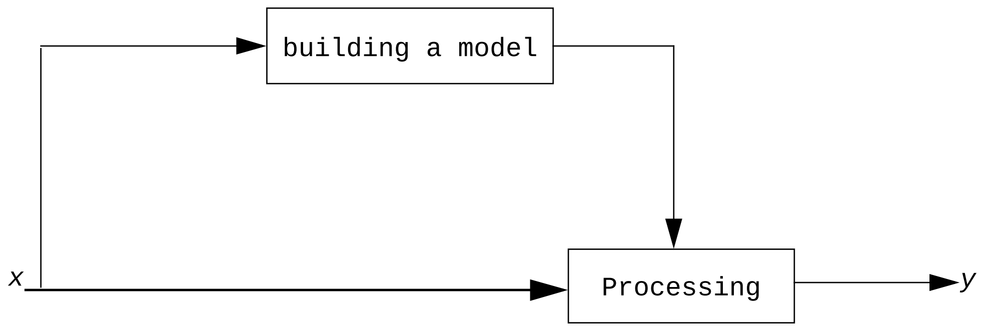


Figure 1.15: Model building and model based signal processing.

in speech synthesis (e.g. in text-to-speech synthesis systems), the model of Figure 1.16 is used to generate speech that sounds fairly natural.

Yet, there are potential problems with models. The first is complexity: models could become arbitrarily complex, thus difficult to estimate. The second is model mismatch: in our speech example, if the sound was actually from a parrot (unlikely but possible), one would have a hard time to find the parameters that are specific to humans!

1.6 Analog and Digital Worlds

Signal processing is at the intersection of the analog and digital worlds. These worlds are very different, and are linked by sampling, quantization, and interpolation. After reviewing briefly these two worlds, we discuss advantages of each.

1.6.1 The Analog World (Continuous Time, Continuous Amplitude)

The world of analog signals is the world of functions on the real line, where the function is typically real valued. Thus, the time axis is continuous, and so is the amplitude. Most signals from the physical world are of this type: sound waves, electrical signals, physical measurements, etc. But also many man-made signals are of this type, like the output of a loudspeaker or the image on a video screen. Many systems from the physical world process such analog signals to produce other analog signals. For example, a physical communication channel takes an analog input (the signaling waveform) and produces an analog output (the received signal), even if the goal is to transmit a purely discrete

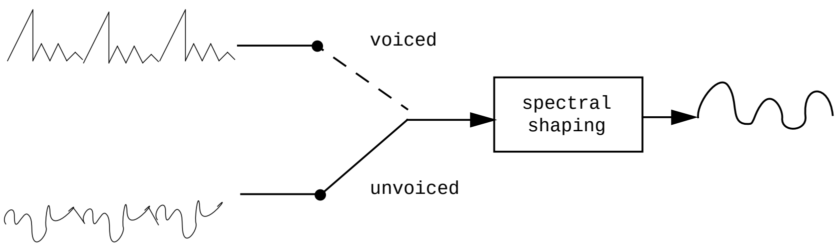


Figure 1.16: Basic speech production model, where two modes are considered (voiced/unvoiced) and time-varying spectral shaping is applied.

information (like a bit from a file transfer). Analog signal processing, which has a long history, is typically performed with analog circuits that perform operations like filtering, amplifying, clipping etc. Analog filter design is a very well studied topic, with many "classic" designs. Nevertheless, it remains true that "good" analog filters are typically expensive, since they require high quality analog components. The main advantage is that analog processing is "instantaneous" (at least in principle), that is, there is no time lag between input and output (other than phase factors or group delays).

1.6.2 Discrete-Time, Analog Worlds (Discrete Time, Continuous Amplitude)

In certain applications, continuous-time signals are sampled, but the real-valued samples are not further digitized (see next section). In that case, discrete-time analog circuits are used for the processing. An example is found in charge-coupled devices (CCD's), used for example in video cameras. While this type of processing corresponds mathematically to the "sampled system" case, it is rather the exception than the rule.

1.6.3 Digital Worlds (Discrete Time, Discrete Amplitude)

After sampling as in the previous section, the analog values are discretized to a countable set (typically a finite set). That is, both the time dimension and the amplitude dimension are now discrete. So we have the cascade of two operations:

$$x(t) \longrightarrow x[n] \longrightarrow \hat{x}[n]$$

the first being sampling, the second being quantization. The second operation cannot be undone, since quantization is a many-to-one mapping. Figure 1.17 shows the 3 types of signals we have just seen. Now, why would one give up the real, continuous amplitude world for this discretized and approximate representation? The reason lies of course in the fact that such discretized values can be represented in computer memory, and that if the discretization is fine enough, the representation is adequate. Because of the dominant position of digital computers in the technological world, the digital representation of signals is by now the most common. Note that both fixed point and floating point arithmetic[2] are possible for such digital signal representation. However both methods are discrete and finite amplitude representations.

1.6.4 Analog versus Digital

In many signal processing tasks, a key question is often: how much processing should be "analog", how much should be "digital". Take the design of a mobile phone system: clearly, input (voice) and output (loudspeaker) are analog, but inside, the system will probably go several times between analog and digital representations. Processing inside the mobile phone is digital, the digital communication over the wireless connection is analog, the base station converts back to digital, which goes over wireline backbone networks. From there, to reach another user, the reverse process is done, until an analog, acoustic signal is generated to reach the recipient's ear. Schematically, this is shown in Figure 1.18.

Another example is at the same time the best explanation of the amazing advance of digital communication systems and a paradigm for the pervasiveness of "digital" processing: analog versus digital telephony as seen over transatlantic links; this example is very simple and intuitive, but is at the heart of the digital "revolution". Given a transatlantic cable, should you use analog or digital transmission? In the analog case, you need repeaters, but these will boost signal and noise almost equally (See Figure 1.19-(a)). In the digital case, there is an inital "noise" due to quantization, but then, as long as the noise added by the channel can be corrected by digital techniques (error correction codes as used in CD's), the noise gets annihilated, maintaining a reliable end-to-end quality (see Figure 1.19-(b)). The same phenomenon can be seen in copying analog signals (e.g. audio cassettes) versus digital signals (CD's). While a few cascaded copies from cassette to cassette will be too noisy to use, an arbitrary number of copies of CD's is no problem

[2]Fixed point arithmetic uses a fixed and finite set of values to represent amplitude, e.g. $[-N_1, -N_1 + 1, \ldots - 1, 0, 1, \ldots N_2 - 1, N_2]$ where $N_i \in \mathbb{Z}$. The major problem is under and overflow during arithmetic operations. Floating point arithmetic uses a mantissa (similar to the fixed point numbers we just saw) and a scale factor given by an exponent with respect to the basis used (e.g. 2^k). While this allows a much better approximation of the real numbers $\mathbb{R}$, it is still a finite representation, and underflows and round off errors are still problematic.

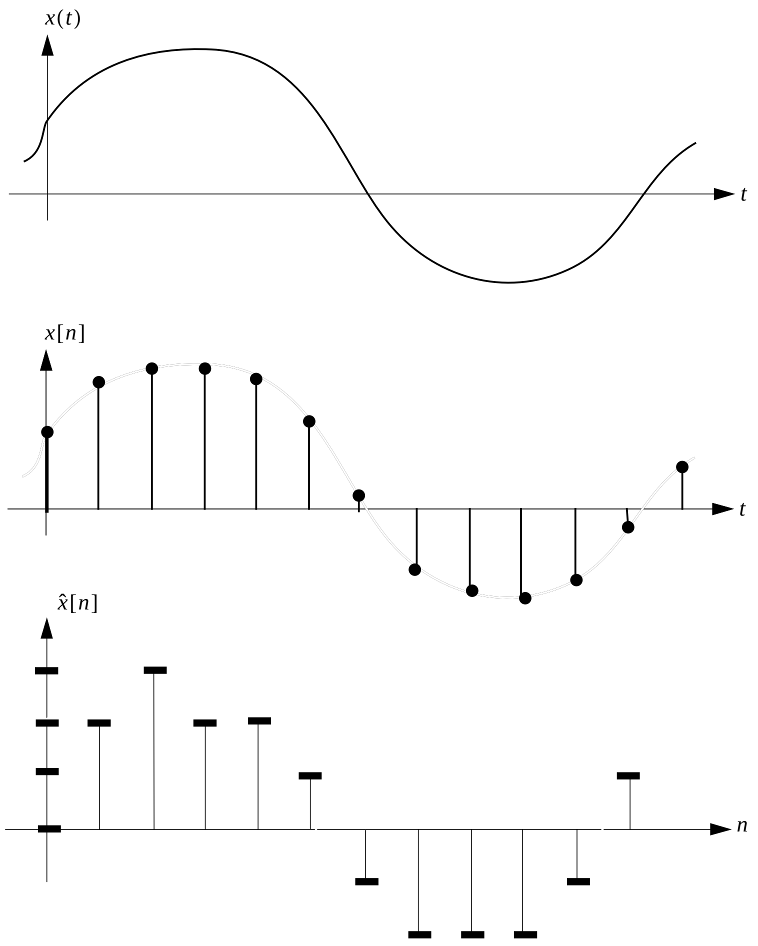


Figure 1.17: Various forms of signals between the analog and digital worlds.
(a) Analog, continuous-time signal. Both axes are continuous.
(b) Discrete-time, continuous-amplitude signal. The time axis is discrete.
(c) Discrete-time and quantized signal. Both axes are now discrete.

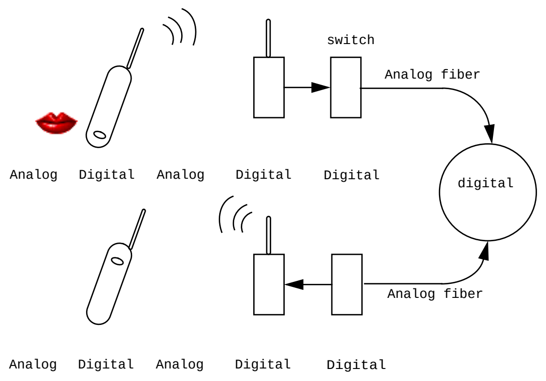


Figure 1.18: Multiple conversions between analog and digital representations in an end-to-end wireless phone communication.

at all. (Except of course for the copyright owner!)

Finally, Table 1.2 summarizes some of the positive and negative points of analog versus digital representations and processing. From this table we can see that the comparison offers a mixed picture. Yet, in reality, the techniques of digital processing have advanced more and more and, at the same time, the devices used to implement digital signal processing algorithms have become more and more powerful and inexpensive. Today's standard desktop computers can easily perform in real time extremely complex tasks such as decoding DVD data, compress voice for internet telephony and modulate data for dial-up connections, and often in parallel. What's more, the time required for the industry to *develop and test* such algorithms is immensely inferior to what would be necessary to design their analog counterparts, admitting that that were at all possible. The global picture is that of an increasingly digital world with analog processing confined to the extreme boundaries, i.e., to the places where an interface to the physical mediums is necessary. This is why the stress of this course is on discrete-time processing techniques.

1.7 Overview of the Course

In this chapter we presented a general overview of signal processing and we tried to show the broad range of signals and systems where signal processing methods can be used. Interestingly, a number of methods are common to this vast array of applications, and these are at the center of our study. The rest of the course will try to lay a solid foundation

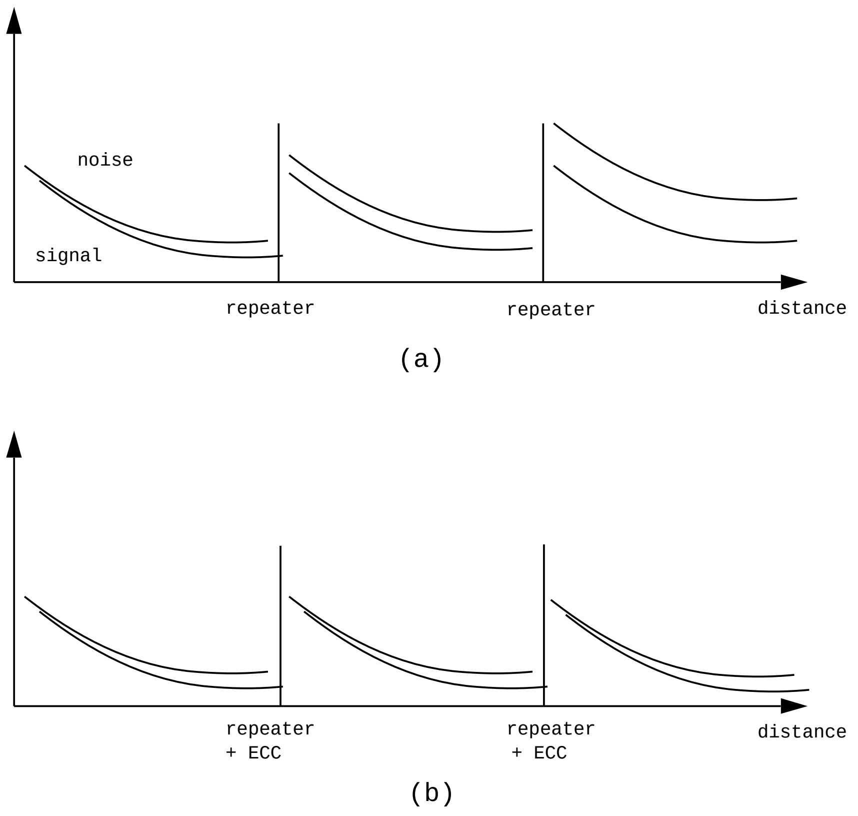


Figure 1.19: Comparison between analog and digital transmission over a transatlantic cable. (a) In the analog case, both the signal and the noise are amplified at the repeaters; (b) In the digital case, if the noise is not too much at the first repeater, a perfect reconstruction can be achieved, and the same holds for subsequent repeaters: the noise does not grow.

	Analog	Digital
+	World mostly analog Precision in principle ∞ Speed is arbitrary Arbitrary signals are possible	Digital computers dominate Calculations exact, reproducible Storage easy No noise
-	Computing mostly digital Noise difficult to control Storage difficult	World is analog, so need interface Initial imprecision due to quantization Speed limit Restricted signals

Table 1.2: Analog versus digital.

for the mathematical basis of signal processing and proceed from there to illustrate the design techniques and applications in more details. Here is a short description of each of the next chapters:

Chapter 2 will introduce more formally the classes of discrete-time signals which we will use in the course. It will also give an informal description of the sampling theorem that describes how to obtain discrete-time signals from continuous-time signal without information loss.

Chapter 3 will explain the different representations of periodic and finite block-lenght discrete-time sequences. Representations in terms of the discrete-time Fourier series (DFS) and the discrete Fourier transform (DFT) are introduced.

Chapter 4 will review background material from applied mathematics and linear algebra, with an emphasis on geometric intuition via the concept of Hilbert spaces. In particular, this background is useful in extending the representations in Chapter 3 to infinite length discrete-time sequences.

Chapter will be devoted to the discrete-time Fourier transform (DTFT) which is a representation of discrete-time sequences. We will also study the properties of DTFT and relate it to the DFS and DFT studied in Chapter 3.

Chapter 6 will develop the ideas of how the Fourier representations can be used in practice. In particular, applications to spectral analysis, time-frequency analysis etc., are introduced.

Chapter 7 will introduce the notion of system, with an emphasis on linear time invariant systems. We will introduce the concept of convolution sum and its application to filtering, both in the time and the frequency domain. The concepts of stability and causality of a system are also introduced.

Chapter 8 will introduce the z-transform and its properties. The z-transform is the generalization of the Fourier transform to the complex plane in the discrete-time domain, just as the Laplace transform is the generalization of the Fourier transform in the continuous-time domain.

Chapter 9 will study the problem of digital filter design and filter implementation, with particular emphasis on FIR filters.

Chapter 10 will deal with the fundamental operation of sampling, by which a continuous-time signal is converted into a discrete-time sequence, and interpolation, by which a discrete-time signal is converted to a continuous-time signal.

Chapter 11 will develop a generalization of standard, single sampling rate signal processing to deal with multiple sampling rates, which could be needed for sampling rate conversions. We develop the basic concepts of this rich topic by introducing filterbanks, sub-band decomposition and basic ideas of wavelets.

Chapter 12 will tackle quantization, or approximate representations. In particular, the problem of analog-to-digital conversion is studied in detail, including oversampling. We also study analog-to-digital conversion and its counterpart, digital-to-analog conversion.

Chapter 13 contains some application ideas such as denoising and a small project on multicarrier communications.

Appendix 1.A Historical Notes

The roots of signal processing are to be found in mathematics and applied mathematics, but the driving force behind its development lies in technological advances.

The idea of signal processing is probably as old as science itself (e.g. prediction of an eclipse based on past observations). Closer to us, the founding father of harmonic analysis, Joseph Fourier (1768-1830), is usually considered an important historical figure, given that Fourier series are central in signal processing. The importance of Fourier analysis is due to several facts, including the eigenfunction property of complex exponentials in linear time-invariant systems and the orthogonal expansion given by Fourier bases. Convergence questions (e.g. the Gibbs phenomenon) are also still important today, as they were when leading mathematicians questioned Fourier's claim that any periodic function could be written as a linear combination of harmonic sine and cosine waves.

A cornerstone result for discrete-time signal processing is the sampling theorem, often attributed to Shannon. While the theorem is indeed shown in Shannon's 1948 landmark paper on communication theory, it was due earlier to Whittaker and Kotelnikov. That bandlimited functions can be represented uniquely by their samples taken at twice the maximum frequency is one example of an interpolation formula, in this case using the *sinc* function. Another view of this result is that *sinc* functions and their translates form an orthonormal basis for the space of bandlimited functions. This view has been generalized recently with the theory of wavelets.

The sampling theorem of the mid 1940's led to sampled data systems, which allowed to use the first computers for applications like ballistics. Such military applications also motivated Wiener to study the problem of prediction and noise removal, leading to Wiener filtering and its variants, some of which are still in use today. The key is that sampled systems allow the use of computers to implement sophisticated algorithms.

The 1950's and 1960's saw a lot of theoretical and practical research in fields like speech analysis and synthesis (e.g. the Vocoder) and very early image processing.

But the real "digital signal processing" revolution started in 1965, with the publication by Cooley and Tukey of an efficient algorithm for the computation of Fourier series, the fast Fourier transform (FFT). Together with the availability of better computers and dedicated hardware, signal processing became a reality outside of the laboratory setting where it was confined until then. In particular, digital signal processing was a key element in making communication systems moved into the digital age: speech moves from analog to digital, digital communications use equalization, teleconferencing is demonstrated, etc.

The late 1960's and the following two decades were periods of intense developments. In particular, image and later video processing came of age, for example with the definition of compression standards that allowed digital communication of images and moving pictures. But also medical imaging, with tomography and echography moving from research to actual usage, was revolutionized by digital processing. The most audible digitalization was of course the introduction of compact discs in 1984, which in a short time replaced analog records by a digital version.

In parallel, the microprocessor revolution led to the introduction of specialized machines called signal processors, that is, microprocessors optimized for signal, image or even video processing tasks. Specialized chips using VLSI technology also became common place for particular, high end signal processing tasks.

The end of the 20th century saw signal processing everywhere, from personal computers with multimedia capabilities to complete, end-to-end digital communication for all forms of signals. While many of the techniques studied in the last decades are now part of everyday objects (e.g. portable phones), new questions constantly arise. To make an exhaustive list is difficult as well as a moving target, but a few topics that come to mind are the following:

- fundamental limits in compression
- joint source-channel coding, e.g. for channels like internet and mobile
- security issues, like watermarking of signals for copyright protection
- recognition, especially for very large data sets, e.g. image databases.

While standard signal processing tools are well understood, new tools are needed for some of the challenges listed above.

Appendix 1.B Literature

Because signal processing is both a popular and a useful topic, it has spawned many publications. An exhaustive list of books would take pages, and therefore we will only mention a subset that is either directly related to our notes, or which we feel is part of the general culture around signal processing.

On basic signal processing, there are several good books, including the recommended textbook for the class, which is *Discrete-Time Signal Processing*, by A. V. Oppenheim and R. W. Schafer (Prentice-Hall, 1989). For advanced signal processing, we can mention *A Wavelet Tour of Signal Processing*, by S. Mallat (Academic Press, 1999), and *Wavelets and Subband Coding*, by M. Vetterli and J. Kovacevic (Prentice Hall, 1995).

For background in signals and systems, we suggest *Signals and Systems* by Oppenheim, Wilsky and Nawab (Prentice Hall, 1997). For Fourier analysis, a nice engineering book is *The Fourier Transform and Its Applications*, by R. Bracewell (McGraw-Hill, 1999), while there are many mathematics textbooks on the topic such as *Fourier Analysis*, by T. W. Korner (Cambridge University Press, 1989).

The research literature on signal processing is published by the Institute of Electronics and Electrical Engineers (IEEE) in several transactions, but mainly in the one on Signal Processing, on Image Processing, and on Speech and Audio Processing. Several for profit publishers also run signal processing journals (e.g. Elsevier's *Signal Processing* magazine).

Chapter 2

Discrete-Time Signals

In this chapter we will introduce more formally the concept of discrete-time signal and we will establish an associated basic taxonomy which we will use in the remainder of the course. Historically, discrete-time signals have often been introduced as the discretized version of continuous-time signals, i.e. as the *sampled* values of analog quantities such as the voltage at the output of an analog circuit; accordingly, many of the derivations proceeded within the framework of an underlying continuous-time reality. In truth, the discretization of analog signals is only part of the story, and a rather minor one nowadays. Digital signal processing, especially in the context of communication systems, is much more concerned with the *synthesis* of discrete-time signals rather than with sampling. That is why we will choose to introduce discrete-time signals from a abstract, self-contained point of view.

2.1 Continuous and Discrete-time Signals

Almost all the signals we described were defined over a continuous space. For example we described a speech signal as a pressure-intensity function over time. Therefore a speech signal $s(t)$ is defined over $t \in \mathbb{R}$, the real line. This gives the speech signal a continuous-time representation. However, most computers deal with discrete-time signals and therefore a fundamental question is how we can represent a continuous system in discrete-time.

$$s[k] = s(kT_s), \qquad k \in \mathbb{Z} \tag{2.1}$$

By sampling a continuous-time signal one can produce a discrete-time signal. However a basic question that arises is how much fidelity such a representation has to the original signal. This turns out to be a very fundamental question in signal representation. For signals with some properties it is possible it to have a completely faithful representation using a discrete-time signal. In some other cases, the representation can be as faithful as one wants by appropriately choosing the sampling period T_s. This is a question we re-visit later.

2.2 Informal description of sampling theorem

Given a signal

$$x(t), \qquad t \in \mathbb{R}$$

a discrete-time signal can be obtained by sampling it at regular intervals of T_s seconds, i.e.,

$$x[n] = x(nT_s), \qquad n \in \mathbb{Z}, \qquad T_s \in \mathbb{R}$$

If T_s is sufficiently small, then it is possible to reconstruct the original signal $x(t)$ from its samples $\{x(kT_s)\}_{k=-\infty}^{k=\infty}$ through an interpolation function.
Take a sinusoidal continuous-time signal

$$x(t) = A\cos(2\pi f_0 t + \theta_0). \tag{2.2}$$

If we sample this at times nT_s, we produce

$$x[n] = x(nT_s) = A\cos(2\pi f_0 nT_s + \theta_0). \tag{2.3}$$

Now suppose $f_0' = f_0 + \frac{1}{T_s}$, and we have a signal

$$x'(t) = A\cos(2\pi f_0' t + \theta_0) \tag{2.4}$$

Sampling $x'(t)$ at the same times nT_s gives

$$\begin{aligned} x'[n] &= A\cos(2\pi f_0' nT_s + \theta_0) && (2.5)\\ &= A\cos(2\pi(f_0 + \frac{1}{T_s})nT_s + \theta_0) && (2.6)\\ &= A\cos(2\pi f_0 nT_s + \theta_0) && (2.7)\\ &= x[n] && (2.8)\end{aligned}$$

Therefore, if we sample at a frequency $\frac{1}{T_s}$, we can only distinguish between frequencies of 0 and $\frac{1}{2T_s}$, all others produce the same samples as a sinusoid of lower frequency(or negative frequency)

Example 2.1 *Sampling of sinusoidal signals:*
Let

$$x_1(t) = \cos(2\pi 10t),$$

$$x_2(t) = \cos(2\pi 50t).$$

and let $T_s = \frac{1}{40}$ *, i.e.,* $F_s = 40Hz$ *is the sampling frequency. Then*

$$x_1[n] = \cos(2\pi \frac{10}{40} n) = \cos(\frac{\pi}{2} n)$$

$$x_2[n] = \cos(2\pi \frac{50}{40} n) = \cos(\frac{5\pi}{2} n) = \cos(\frac{\pi}{2} n + 2\pi n) = \cos(\frac{\pi}{2} n)$$

Hence the sampled version of $x_1(t)$ *and* $x_2(t)$ *at a sampling rate of* $40Hz$ *are indistinguishable.*

One can see that if we know the largest frequency that occurs, we can determine the sampling frequency needed to faithfully and uniquely represent such a signal.
In general, if

$$x_a(t) = \sin(2\pi f_0 t) \tag{2.9}$$

and T_s= sampling period

$$x[n] = \sin(2\pi f_0 n T_s) = \sin(2\pi [f_0 T_s + r] n) \tag{2.10}$$

Hence $f_0' = f_0 + \frac{r}{T_s} \qquad r \in \mathbb{Z}$ are indistinguishable.
For real signals, for every frequency f_0, there is a mirror image at $-f_0$.

Example 2.2 *Negative and positive frequency:*

$$\sin(-2\pi f_0 t) = -\sin(2\pi f_0 t)$$

$$\cos(-2\pi f_0 t) = \cos(2\pi f_0 t)$$

We need to accommodate *both* the positive and negative mirrors for distinguishablity.

Hence, we can say that for a sampling period T_s, only frequencies

$$-\frac{1}{2T_s} \leq f \leq \frac{1}{2T_s} \tag{2.11}$$

are distinguishable. Therefore, if the maximum frequency is F_{max}, we need $F_s = 2F_{max}$ for any hope of reconstruction.

Example 2.3 *Sampling*
Let signals $x(t)$ and $x'(t)$ be given by

$$x(t) = \sin(2\pi\epsilon t) + \sin(2\pi 75t) + \sin(2\pi 150t),$$

$$x'(t) = \sin(2\pi 75t).$$

If we sample these signals with a sampling frequency of $F_s = (150+\epsilon)$ Hz we get

$$\begin{aligned}x[n] &= \sin(2\pi\epsilon\frac{n}{150+\epsilon}) + \sin(2\pi\frac{75}{150+\epsilon}n) + \sin(2\pi\frac{150}{150+\epsilon}n)\\ &= \sin(2\pi\epsilon\frac{n}{150+\epsilon}) + \sin(2\pi\frac{75}{150+\epsilon}n) + \sin(2\pi\frac{-\epsilon}{150+\epsilon}n)\\ &= \sin(2\pi\frac{75}{150+\epsilon}n)\end{aligned}$$

and

$$x'[n] = \sin(2\pi\frac{75}{150+\epsilon}n).$$

The sampled signals are indistinguishable!
However, if $F_s = (300+\epsilon)Hz$ then

$$x[n] = \sin(2\pi\epsilon\frac{\epsilon}{300+\epsilon}n) + \sin(2\pi\frac{75}{300+\epsilon}n) + \sin(2\pi\frac{150}{300+\epsilon}n)$$

and

$$x'[n] = \sin(2\pi\frac{75}{300+\epsilon}n)$$

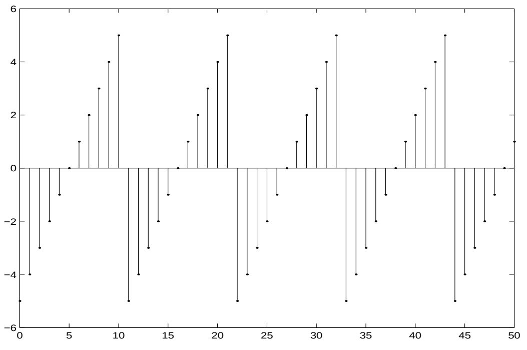
are completely distinguishable.

Remark: Distinguishablity is a necessary condition but not a sufficient condition for being able to reconstruct.

The following is a deep result in signal representation and has impact in a lot of areas.

Theorem 2.1 *If the signal $x(t)$ satisfies the regularity condition that it is band-limited and f_{max} is the largest frequency (i.e. bandwidth), then samples of $x(t)$, $\{x(kT_s)\}$ for $\frac{1}{T_s} > 2f_{max}$ will be sufficient to reconstruct the signal $x(t)$.*

This is a result we will return to and prove formally, but for now it is important to understand that the discrete-time sequences that we will work with for the next few weeks can be connected to physical, continuous-time signals under some mild conditions.


(a)

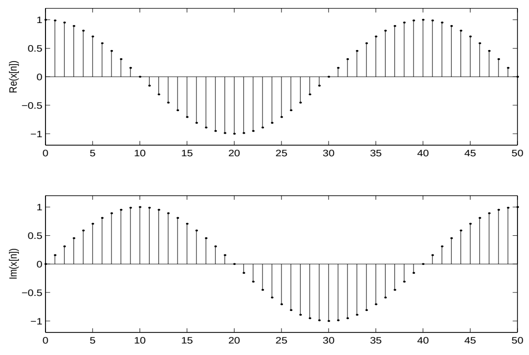

(b)

Figure 2.1: Examples of signals. (a) triangular wave; (b) complex exponential .

2.3 Discrete-time sequences

A sequence is a set of numbers denoted as

$$x[n], \qquad n \in \mathbb{Z}$$

i.e. defined over the set of integers

Discrete-time sequences can arise from sampling a continuous-time sequence, a discrete-time signal can also arise in its own. For example

$$x[n] = (n \mod 11) - 5 \tag{2.12}$$

which is the "triangular" waveform plotted in Figure 2.1-(a), or

$$x[n] = e^{j\frac{\pi}{20}n} \tag{2.13}$$

which is a complex exponential of period 40 samples and which is plotted in Figure 2.1-(b). Two example of a sequence drawn from the real world are

$$x[n] = \text{The average stock market index in year } n,$$

and

$$x[n] = \text{Number of hits to a web-page in the } n^{th}\text{hour},$$

which are inherently discrete-time and therefore need not be represented as a physical modeling of a continuous-time signal. Therefore, we will deal with discrete-time signals in their own merit and connect them to continuous-time entities much later in the class.

2.3.1 Basic Signals

The following sequences are fundamental building blocks in the theory of signal processing:

- The discrete-time impulse (Figure 2.2-(a))

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

- The discrete-time unit step (Figure 2.2-(b))

$$u[n] = \begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}$$

  which can be represented as

$$u[n] = \sum_{k=-\infty}^{n} \delta[k] = \sum_{k=0}^{\infty} \delta[n-k].$$

- The discrete-time exponential decay (Figure 2.2-(c))

$$x[n] = a^n u[n], \quad a \in \mathbb{C}, |a| < 1$$

- The discrete-time sinusoidal oscillations (Figure 2.2-(d))

$$x[n] = \sin(\omega_0 n + \phi)$$

$$x[n] = \cos(\omega_0 n + \phi)$$

- The discrete-time complex exponential (Figure 2.1-(b))

$$x[n] = e^{j(\omega_0 n + \phi)}$$

Example 2.4 *Combining basic sequences*

$$y[n] = \begin{cases} A\alpha^n & n \geq 0 \\ 0 & n < 0 \end{cases}$$

takes $y[n] = x[n]u[n]$, where $x[n] = A\alpha^n$ and $u[n]$ is discrete-time unit step. (Figure 2.2-(d))

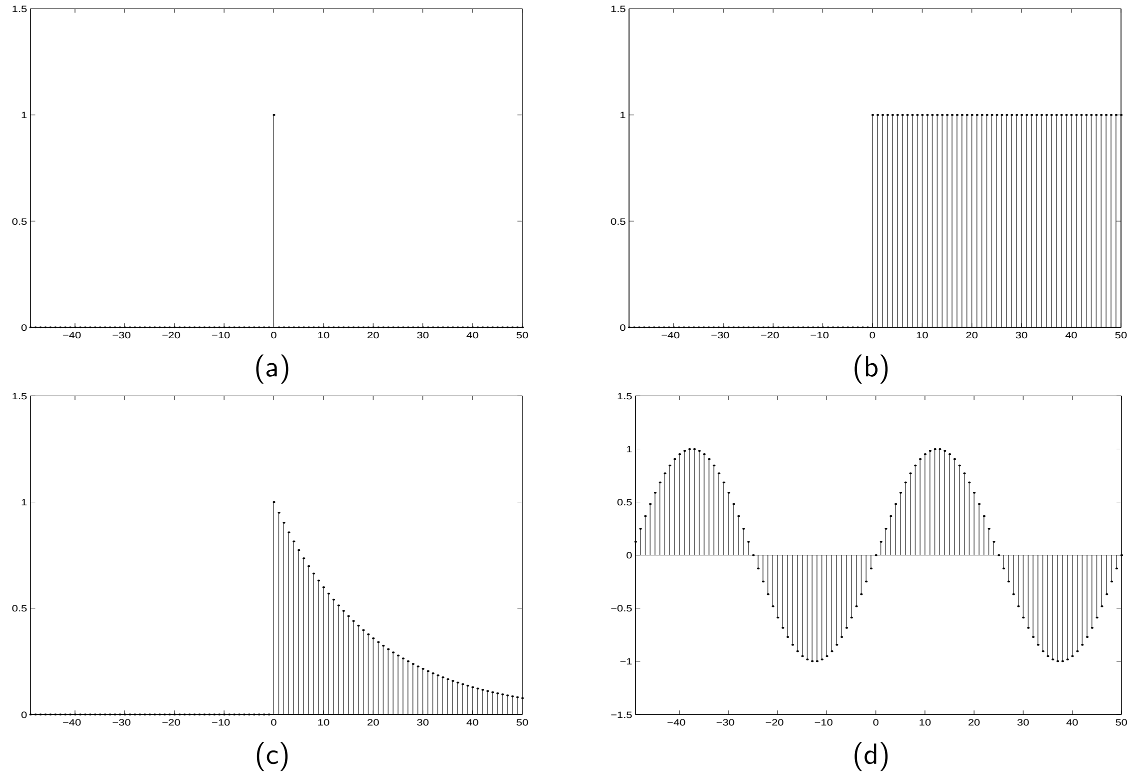


Figure 2.2: Basic signals.

Definition 2.1 *A sequence $\{x[n]\}$ is said to have period N if*

$$x[n] = x[n+N], \qquad \textit{for all } n \in \mathbb{Z}$$

For a complex exponential $x[n] = e^{j\omega_0 n}$ if it is to have a period of N, we need

$$e^{j\omega_0 n} = e^{j\omega_0 (n+N)}, \quad \forall n$$

i.e.,

$$\omega_0 N = 2\pi r, r \in \mathbb{Z}$$

or $\omega_0 = \frac{2\pi r}{N}$ which means that since $N \in \mathbb{Z}$, we need ω_0 to be rational for a complex exponential to be periodic in a discrete sense.

Example 2.5 *Let $x_1[n] = \cos(\frac{\pi n}{4})$. Since for all n, $x_1[n] = x_1[n+8]$, we have a period of $N = 8$ for the sequence.*
However, $x_2[n] = \cos(\frac{3\pi n}{8})$ gives $x_2[n+8] = \cos[\frac{3\pi n}{8} + 3\pi] = -x_2[n]$ and hence does not *have a period of 8. In fact it has a period of $N = 16$. Therefore, even though we think of a higher "'frequency"' for $x_2[n]$ in comparison to $x_1[n]$, it has a larger period. This is due to the limitations imposed by integer time index n for discrete-time signals.*

2.3.2 Digital Frequency

With respect to the last two examples a note on the concept of "frequency" is in order. In the analog world the usual unit of measure for frequency is the Hertz, which has a physical dimension of s^{-1}. In the discrete-time world, where the index n represents dimensionless time, "digital" frequency is expressed in radians which is itself an dimensionless quantity[1]. The best way to appreciate this is to consider an algorithm to generate successive samples of a discrete time sinusoid at a digital frequency ω_0:

```
$\omega \leftarrow 0$;                          initialization
$\phi \leftarrow$ initial phase value;
repeat
    $x \leftarrow \sin(\omega + \phi)$;         compute next value
    $\omega \leftarrow \omega + \omega_0$;      update phase
until done
```

At each iteration[2] , the argument of the trigonometric function is incremented by ω_0 and a new output sample is produced. With this in mind, it is easy to see that the highest frequency manageable by a discrete-time system is 2π; for any frequency larger than this, the inner 2π-periodicity of the trigonometric functions "maps back" the output values to a frequency between 0 and 2π. In formulas:

$$\sin(n(\omega + 2k\pi) + \phi) = \sin(n\omega + \phi) \tag{2.14}$$

[1]An angle measure in radians is dimensionless since it is defined in terms of the ratio of two lengths, the radius and the arc subtended by the measured angle on an arbitrary circle.

[2]Here is the same algorithm written as a C function, if it helps:

```
extern double omega0;
extern double phi;
static double omega = 0;
double GetNextValue()
{
    omega += omega0;
    return sin(omega + phi);
}
```

for all values of $k \in \mathbb{Z}$. This 2π-equivalence of digital frequencies is a pervasive concept in digital signal processing and it has many important consequences which we will study in detail throughout the course.

2.3.3 Elementary Operators

Elementary operations on sequences are defined as follows:

- **Shift.** The shifted version of the sequence $x[n]$ by an integer k is

  $$y[n] = x[n-k]$$

  If k is positive, the signal has been *delayed*; if k is negative, it has been *advanced.*

- **Scaling.** The scaled version of the sequence $x[n]$ by a factor $\alpha \in \mathbb{C}$ is

  $$y[n] = \alpha x[n]$$

- **Sum.** The sum of two sequences $x[n]$ and $w[n]$ is their term-by-term sum,

  $$y[n] = x[n] + w[n]$$

- **Product.** The product of two sequences $x[n]$ and $w[n]$ is their term-by-term product,

  $$y[n] = x[n]w[n]$$

- **Moving average**

  $$y[n] = \frac{1}{M_1 + M_2 + 1} \sum_{k=-M_1}^{M_2} X[n-k].$$

- **Integration.** The discrete-time equivalent of integration is the running sum:

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

- **Differentiation.** A discrete-time approximation to differentiation is the first-order difference[3]:

$$y[n] = x[n] - x[n-1]$$

With respect to section 2.3.1, note how the unit step can be obtained by applying the integration operator to the discrete-time impulse; conversely, the impulse can be obtained by applying the differentiation operator to the unit step.

Definition 2.2 *Let $\mathcal{L}(\cdot)$ be an operation. The operation is linear if*

$$\mathcal{L}(\alpha x_1[n] + \beta x_2[n]) = \alpha\mathcal{L}(x_1[n]) + \beta\mathcal{L}(x_1[n]), \tag{2.15}$$

for any sequences $x_1[n]$, $x_2[n]$ and scalars α and β.

All the operations defined above are linear operations.

Example 2.6 *The shift operation is linear. Let $\mathcal{L}(x[n]) = x[n-k]$. Now, if $y_1[n] = x_1[n-k]$ and $y_2[n] = x_2[n-k]$, then $y[n] = \mathcal{L}(\alpha x_1[n] + \beta x_2[n]) = \alpha x_1[n-k] + \beta x_2[n-k])$ which can be expressed as $y[n] = \alpha y_1[n] + \beta y_2[n]$.*

Example 2.7 *Suppose $y[n] = x^2[n]$. If $y_1[n] = x_1[n]^2$, and $y_2[n] = x_2[n]^2$, then $y[n] = \mathcal{L}(x_1[n]+x_2[n]) = (x_1[n]+x_2[n])^2 = x_1[n]^2+x_2[n]^2+2x_1[n]x_2[n]$ which can not be expressed as $y_1[n] + y_2[n]$. So this operation is not linear.*

2.3.4 The Reproducing Formula

The signal reproducing formula is a simple application of the basic signal and signal properties we have just seen and it states that:

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k] \tag{2.16}$$

In words, any signal can be expressed as a linear combination of suitably weighed shifted impulses. In this case, the weights are simply the signal values. While apparently self-evident, this formula will reappear in a multitude of reincarnations in the rest of the course. You are encouraged to spend a few minutes thinking about how it actually works.

[3]We will see later, when we study filters, that the "correct" approximation to differentiation is given by a filter $H(e^{j\omega}) = j\omega$. For most application, however, the first-order difference will suffice.

2.3.5 Energy and power

We define the *energy* of a discrete-time signal as

$$E_x = ||x||_2^2 = \sum_{n=-\infty}^{\infty} |x[n]|^2 \tag{2.17}$$

(where the squared-norm notation will be clearer after the next chapter.) This definition is consistent with the idea that, if the values of the sequence represent a time-varying voltage, the above sum would express the total energy (in joules) dissipated over a 1Ω-resistor. Obviously, the energy is finite only if the above sum converges, i.e., if the sequence $x[n]$ is *square-summable*. A signal with this property is sometimes referred to as a *finite-energy signal*. For a simple example of the converse, note that a periodic signal which is not identically zero is *not* square-summable.

We define the *power* of a signal as the usual ratio of energy over time, taking the limit over the number of samples considered:

$$P_x = \lim_{N\to\infty} \frac{1}{2N} \sum_{-N}^{N-1} |x[n]|^2; \tag{2.18}$$

Clearly, signals whose energy is finite have zero total power (i.e. their energy dilutes to zero over an infinite time duration). Note however that many signals whose energy is infinite do have finite power and, in particular, so do periodic signals (such as sinusoids and combinations thereof). Due to their periodic nature, however, the above limit is undetermined; we therefore *define* their power to be simply the *average energy over a period*. Assuming that the period is N samples, we have:

$$P_x = \frac{1}{N} \sum_{n=0}^{N-1} |x[n]|^2. \tag{2.19}$$

2.4 Classes of Discrete-Time Signals

The examples of discrete-time signals in (2.12) and (2.13) are two-sided, infinite sequences. Of course, in the practice of signal processing, it is impossible to deal with infinite sequences: for a processing algorithm to compute in a finite amount of time and use a finite amount of storage, the input data must be of finite length; even for algorithms that operate on the fly, i.e. algorithms that produce an output sample for each new input sample, an implicit finiteness is imposed by the necessarily limited life span of the processing device or, in the extreme limit, of the supervising engineer. This limitation was eminently clear in our attempt to plot the sequences in Figures 2.1-(a), (b): we were content with showing a representative portion of the sequences, and we relied on their analytical description to describe their behavior outside of the observation window we chose for the plot. When the discrete-time signal admits no closed-form representation, as is basically always the case when dealing with real-world signals, its finite time support arises naturally because of the finite time we spend measuring said signal: every piece of music has a beginning and an end, and so does any phone conversation. In the case of the sequence representing the Dow Jones index, for instance, we sort of cheated since the index does not even exist for years before 1884, and its value tomorrow is certainly not known – so that's not really a sequence. But, more importantly and more often, the finiteness of a discrete-time signal is arbitrarily imposed since we are interested in concentrating our processing efforts on a small portion of an otherwise much longer signal; in a speech recognition system, for instance, the practice is to cut up a speech signal into small segments and try to identify the phonemes associated to each one of them[4]. A special case is that of periodic signals; even though these are bona-fide infinite sequences, it is clear that all information about them is contained in just one period. By describing graphically or otherwise this period, we are in fact providing a complete description of the sequence. In order to capture these particular cases, we will divide signals into three main families.

2.4.1 Finite-Length Signals

As we just mentioned, finite-length discrete-time signals of length N are just a collection of N complex values. To introduce a point that will reappear throughout these notes, a finite-length signal of length N is entirely equivalent to a vector in $\mathbb{C}^N$. This equivalence is of immense import since all the tools of linear algebra become readily available for describing and manipulating finite-length signals. We can represent an N-point finite-length signal using the standard vector notation

$$\mathbf{x} = [x_0 \quad x_1 \quad \ldots \quad x_{N-1}]^T;$$

note the transpose operator, which declares $\mathbf{x}$ as a *column* vector; this is the customary practice in the case of complex-valued vectors. Alternatively, we can (and often will) use a notation that mimics that which we use for proper sequences:

$$x[n], \quad n = 0, \ldots, N-1;$$

here we *must* remember that, although we use the notation $x[n]$, $x[n]$ is *not defined* for values outside its support, i.e. for $n < 0$ or for $n \geq N$. Note that we can always obtain a finite-length signal from an infinite sequence by simply dropping the sequence values

[4]Note that, in the end, phonemes are pasted together into words and words into sentences; therefore, for a complete speech recognition system, long-range dependencies become important again.

outside the indices of interest. Vector and sequence notations are equivalent and will be used interchangeably according to convenience; in general, the vector notation is useful when we want to stress the algorithmic or geometric nature of certain signal processing operations. The sequence notation is useful in stressing the algebraic structure of signal processing.

Finite-length signals are extremely convenient entities: their energy is always finite as long as the elements in the signals are finite; as a consequence, no stability issues arise in processing. From the computational point of view, they are not only a necessity but often the cornerstone of very efficient algorithmic design (as we will see for instance in the case of the FFT); one could say that all "practical" signal processing lives in $\mathbb{C}^N$. It would be extremely awkward, however, to develop the whole theory of signal processing only in terms of finite-length signals; the asymptotic behavior of algorithms and transformations for infinite sequences is extremely valuable as well since a stability result proven for a general sequence will hold for all finite-length signals too. Furthermore the notational flexibility which infinite sequences derive from their function-like definition is extremely practical from the point of view of the notation. We can immediately recognize and understand the expression $x[n-k]$ as a k-point shift of a sequence $x[n]$; but, in the case of finite-support signals, how are we to define such a shift? We would have to explicitly take into account the finiteness of the signal and the associated "border effects", i.e. the behavior of operations at the edges of the signal. This is why, in most derivations which involve finite-length signal, these signals will be *embedded* into a proper sequences, as we will see momentarily.

2.4.2 Infinite, Aperiodic Signals

The most general type of discrete-time signal is represented by a generic infinite complex sequence. Although, as we said, they lie beyond our processing and storage capabilities, they are invaluably useful as a generalization in the limit. As such, they must be handled with some care when it comes to their properties. We will see shortly that two of the most important properties of infinite sequences concern their summability: this can take the form of either *absolute summability* (stronger condition) or *square summability* (weaker condition corresponding to finite energy).

2.4.3 Periodic Signals and Periodic Extensions

A periodic sequence with period N is one for which

$$\tilde{x}[n] = \tilde{x}[n + kN], \quad k \in \mathbb{Z}. \tag{2.20}$$

The tilde notation $\tilde{x}[n]$ will be used whenever we need to explicitly stress a periodic behavior. Clearly a N-periodic sequence is completely defined by its N values over a

$$\begin{array}{rcl} \tilde{x}[n] &=& \ldots x_{N-2}, x_{N-1}, \ \overbrace{x_0, x_1, x_2, \ldots, x_{N-2}, x_{N-1},}^{\mathbf{x}} \ x_0, x_1, \ldots \\ && \qquad\qquad\qquad \updownarrow\!\rightarrow n=0 \\ \tilde{x}[n-1] &=& \ldots x_{N-3}, x_{N-2}, \ \underbrace{x_{N-1}, x_0, x_1, x_2, \ldots, x_{N-2},}_{\mathbf{x}'} \ x_{N-1}, x_0, x_1, \ldots \end{array}$$

Figure 2.3: Equivalence between a right shift by one of a periodized signal and the circular shift of the original signal. $\mathbf{x}$ and $\mathbf{x}'$ are the length-N original signal and its right circular shift by one, respectively.

period; that is, a periodic sequence "carries no more information" that a finite-length signal of length N. In this sense, periodic sequences are a bridge between finite-length signals and infinite sequences. We are therefore ready to discover the first way to embed a finite-length signal $x[n],\ n = 0, \ldots, N-1$ into a sequence which is by taking its periodized version:

$$\tilde{x}[n] = x[n \mod N], \quad n \in \mathbb{Z}; \tag{2.21}$$

this is called the *periodic extension* of the finite length signal $x[n]$. This type of extension is the "natural" one in many contexts, for reasons which will be apparent later when we study the frequency-domain representation of discrete-time signals. Note that now an arbitrary shift of the periodic sequence correspond to the periodization of a *circular shift* of the original finite-length signal. A circular shift by $k \in \mathbb{Z}$ is easily visualized by imagining a shift register; if we are shifting towards the right ($k > 0$), the values which pop out of the rightmost end of the shift register are pushed back in at the other end[5]. The relationship between circular shift of a finite-length signal and the linear shift of its periodic extension is depicted in Figure 2.3. Finally, the energy of a periodic extension becomes infinite, while its power is simply the energy of the finite-length original signal scaled by $1/N$.

Example 2.8 *What is the period of the following sequence?*

$$\tilde{x}[n] = 2 + \sin(\frac{2\pi}{3}n) + \cos(\frac{4\pi}{5}n)$$

[5]For example, if $\mathbf{x} = [1\,2\,3\,4\,5]$, a right circular shift by 2 yields $\mathbf{x} = [4\,5\,1\,2\,3]$.

To answer this, we need to look for N such that for all n, $\tilde{x}[n+N] = \tilde{x}[n]$. This means that we need to find N for which

$$2 + \sin\left(\frac{2\pi}{3}(n+N)\right) + \cos\left(\frac{4\pi}{5}(n+N)\right) = 2 + \sin\left(\frac{2\pi}{3}n\right) + \cos\left(\frac{4\pi}{5}n\right). \tag{2.22}$$

We have that $\sin\left(\frac{2\pi}{3}(n+N_1)\right) = \sin\left(\frac{2\pi}{3}n\right)$ *for* $N_1 = 3$ *and* $\sin\left(\frac{4\pi}{5}(n+N_2)\right) = \sin\left(\frac{4\pi}{5}n\right)$ *for* $N_2 = 5$. *If we take N equal to the least common multiple of N_1 and N_2 we satisfy (2.22). Hence $N = 15$.*

2.4.4 Finite-Support Signals

An infinite discrete-time sequence $\bar{x}[n]$ is said to have *finite support* if its values are zero for all indices outside of an interval; that is, there exist N and $M \in \mathbb{Z}$ such that

$$\bar{x}[n] = 0 \text{ for } n < M \text{ and } n > M + N - 1.$$

Note that, although $\bar{x}[n]$ is an infinite sequence, the knowledge of M and of the N nonzero values of the sequence completely specify the entire signal. This suggest another approach to embedding a finite-length signal $x[n]$, $n = 0, \ldots, N-1$ into a sequence, i.e.

$$\bar{x}[n] = \begin{cases} x[n] & \text{if } 0 \le n < N-1 \\ 0 & \text{otherwise} \end{cases} \qquad n \in \mathbb{Z} \tag{2.23}$$

where we have chosen $M = 0$ (but any other choice of M would do as well). Note that here, in contrast to the the periodic extension of $x[n]$, we are actually adding arbitrary information in the form of the the zero values outside of the support interval. This is not without consequences, as we will see in the following chapters. In general we will use the bar notation $\bar{x}[n]$ for sequences defined as the finite support extension of a finite-length signal. Note that now the shift of the finite-support extension gives rise to a zero-padded shift of the signal locations between M and $M + N - 1$; the dynamics of the shift are shown in Figure 2.4.

2.5 Summary

The main points introduced by this chapter have been:

- The formal definition for the concept of discrete-time signal.
- A gallery of prototypical signals and fundamental signal operators.
- A discussion of digital frequency and its 2π-periodic nature.

$$\begin{array}{rcll} \bar{x}[n] & = & \ldots,0,0, & \overbrace{x_0,x_1,x_2,\ldots,x_{N-2},x_{N-1},}^{\mathbf{x}} \quad 0,0,0,0,\ldots \\ & & & \updownarrow\rightarrow n=0 \\ \bar{x}[n-1] & = & \ldots 0,0, & \underbrace{0,x_0,x_1,x_2,\ldots,x_{N-3},x_{N-2},}_{\mathbf{x}'} \quad x_{N-1},0,0,\ldots \end{array}$$

Figure 2.4: Relationship between the right shift by one of a finite-support extension and the zero padded shift of the original signal. $\mathbf{x}$ and $\mathbf{x}'$ are the length-N original signal and its zero-padded shift by one, respectively.

- The definitions of energy and power.
- A classification of signals into finite-length, infinite-length, periodic and finite-support signals, with their respective properties.

The fundamental discrete-time signal types, along with their properties, are summarized in Table 2.1.

Signal Type	**Notation**	**Energy**	**Power**
Finite-Length	$x[n], \quad n = 0, 1, \ldots, N-1$ $\mathbf{x}, \quad \mathbf{x} \in \mathbb{C}^N$	$\sum_{n=0}^{N-1} \lvert x[n]\rvert^2$	undef.
Infinite-Length	$x[n], \quad n \in \mathbb{Z}$	eq. (2.17)	eq. (2.18)
N-Periodic	$\tilde{x}[n], \quad n \in \mathbb{Z},$ $\tilde{x}[n] = \tilde{x}[n + kN]$	∞	eq. (2.19)
Finite-Support	$\bar{x}[n], \quad n \in \mathbb{Z}$ $\bar{x}[n] \neq 0$ for $M \leq n \leq M + N - 1$	$\sum_{n=M}^{M+N-1} \lvert x[n]\rvert^2$	0

Table 2.1: Basic discrete-time signal types.

Chapter 3

Representation of Discrete-Time Sequences (DFS, DFT)

Fourier theory has a long history, from J. Fourier's early work on the transmission of heat to recent results on non-harmonic Fourier series and related topics. Fourier theory is a branch of harmonic analysis, and in that sense, a topic in pure and applied mathematics. At the same time, because of its usefulness in practical applications, Fourier analysis is a key tool in several engineering branches, and in signal processing in particular.

Why is Fourier analysis so important? To understand this, it is useful to take a little philosophical detour. Interesting signals are time-varying quantities: you can imagine for instance the voltage level at the output of a microphone or the measured level of the tide at a particular location; in all cases, the variation of a signal over time implies that a transfer of energy is happening someplace, and this is what ultimately we want to study. Now, a time-varying value which only *increases* over time is not only a physical impossibility but a recipe for disaster for whatever system is supposed to deal with it: fuses will blow, wires will melt and so on. Oscillations, on the other hand, are nature's and man's way to keep things in motion without trespassing all physical bounds; from Maxwell's wave equation to the mechanics of the vocal cords, from the motion of an engine to the ebb and flow of tide, oscillatory behavior is the recurring theme. Sinusoidal oscillations, as it stands, are the purest form of such a constrained motion and, in a nutshell, Fourier's immense contribution was to show that (at least mathematically) one could express any given phenomenon as the combined output of a number of sinusoidal "generators".

Sinusoids have another remarkable property which justifies their ubiquitous presence. Indeed, *any linear transformation of a sinusoid is a sinusoid at the same frequency*: we express this by saying that sinusoidal oscillations are eigenfunctions of linear systems. This is a formidable tool for the analysis and design of signal processing structures, as we will see in much detail in the context of linear systems.

The purpose of the present chapter is to review key results on Fourier series and Fourier transforms in the context of discrete-time signal processing. As it turns out, and as we hinted at in the previous chapter, the Fourier transform of a signal is a change of basis in its appropriate Hilbert space. While this notion constitutes an extremely useful unifying framework, we will also point out the peculiarities of its specialization within the different classes of signals. In particular, for finite-length signals we will highlight the eminently algebraic nature of the transform, which will lead to efficient computational procedures; for infinite sequences, we will analyze some of its interesting mathematical subtleties.

A periodic sequence has the property that

$$\tilde{x}[n+N] = \tilde{x}[n] \quad \forall n,$$

and N is a period. Therefore a periodic sequence is completely specified by N values. Without loss of generality, we can take these N values from $x[0], \ldots, x[N-1]$.
One representation of this sequence is in the "time domain", but one can imagine completely equivalent representations in other forms (bases). We will consider other representations of such signals in this class such as

- Fourier Transform (through complex exponentials)
- Z-Transform
- Time-frequency representation (like wavelets)

A way to think of these different representations is that each of them has some canonical properties suited to particular scenarios. The most important representation, historically as well as in applications, is the so-called Fourier representation.

3.1 Preliminaries

3.1.1 Terminology

The Fourier transform of a signal is an alternative representation of the data in the signal. While a signal lives in the *time domain*[1], its Fourier representation lives in the *frequency domain*. We can move back and forth at will from one domain to the other using the direct and inverse Fourier operators, since these operators are invertible.

In this chapter we will study three types of Fourier transforms which apply to the three main classes of signals we have seen so far:

[1] *Discrete*-time, of course.

- the Discrete Fourier Transform (DFT), which maps length-N signals into a set of N discrete frequency components
- the Discrete Fourier Series (DFS), which maps N-periodic sequences into a set of N discrete frequency components
- the Discrete-Time Fourier Transform (DTFT), which maps infinite sequences into the space of 2π-periodic functions of a real-valued argument.

The frequency representation of a signal (given by a set of coefficients in the case of the DFT and DFS and by a frequency distribution in the case of the DTFT) is called the *spectrum*.

3.1.2 Complex Oscillations? Negative Frequencies?

In the introduction, we hinted at the fact that Fourier analysis allows us to decompose a physical phenomenon into oscillatory components. It may seem odd, however, that we chose to use complex oscillation for the analysis of real-world signals. It may seem even more odd that these oscillations can have a negative frequency and that, as we will soon see in the context of the DTFT, the spectrum extends over to the negative axis.

The starting point in answering these legitimate questions is to recall that the use of complex exponentials is essentially a matter of convenience. One could develop a complete theory of frequency analysis for real signals using only the basic trigonometric functions. You may actually have seen this in the context of Fourier series; yet the notational overhead is undoubtedly heavy since it involves two separate sets of coefficients for the sine and cosine basis functions, plus a distinct term for the zero-order coefficient. The use of complex exponentials elegantly unifies these separate series into a single complex-valued sequence. Yet, one may ask again, what does it mean for the spectrum of a musical sound to be complex? Simply put, the complex nature of the spectrum is a compact way of representing two concurrent pieces of information which uniquely define each spectral component: its *frequency* and its *phase*. This couple of values is a two-element vector in $\mathbb{R}^2$ but, since $\mathbb{R}^2$ is isomorphic to $\mathbb{C}$, we use complex numbers for their mathematical convenience.

What about negative frequencies, then? Again, first of all consider a basic complex exponential sequence such as $x[n] = e^{j\omega n}$. We can visualize its evolution over discrete-time as a series of points on the unit circle in the complex plane. At each step, the angle increases by ω, defining a counterclockwise circular motion. It is easy to see that a complex exponential sequence of frequency $-\omega$ is just the same series of points which moves *clockwise* instead; this is illustrated in detail in Figure 3.1. We will show that if we decompose a *real* signal into complex exponentials, for any given frequency value,

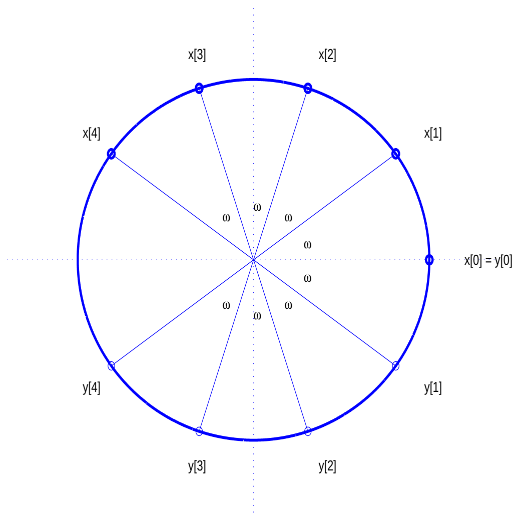


Figure 3.1: Complex exponentials as a series of points on the unit circle; $x[n] = e^{j\omega n}$ and $y[n] = e^{-j\omega n}$ for $\omega = \pi/5$.

the phases of the positive and negative components are always opposite in sign; as the two oscillations move in opposite directions along the unit circle, their complex part will always cancel out exactly, thus returning a purely real signal[2].

The final step in developing a comfortable feeling for complex oscillations comes from the realization that, in the *synthesis* of discrete-time signals (and especially in the case of communication systems) it is actually more convenient to work with complex-valued signals themselves. While in the end the transmitted signal of a device like an ADSL box is a real signal, the internal representation of the underlying sequences is complex, and therefore complex oscillations become a necessity.

[2]To anticipate a question which may appear later, the fact that modulation "makes negative frequencies appear in the positive spectrum" is really a consequence of the very mundane formula:

$$\cos\alpha\cos\beta = \frac{1}{2}[\cos(\alpha+\beta) + \cos(\alpha-\beta)].$$

3.1.3 Complex Exponentials

The basic ingredient of all Fourier representations (transforms) is the complex exponential which we have seen before

$$x[n] = Ae^{j\omega n} = A\cos(\omega n) + jA\sin(\omega n) \tag{3.1}$$

A natural question to ask is why we would use a complex oscillating signal, when most signals we encounter are real. The simplest answer is in terms of notational convenience. It is possible to develop representation using only real sinusoids, but in order to account for the phase as well as frequency, it becomes more cumbersome.
A real sinusoid can always be represented using complex sinusoids as follows:

$$\sin(\omega n) = \frac{e^{j\omega n} - e^{-j\omega n}}{2j}$$

$$\cos(\omega n) = \frac{e^{j\omega n} + e^{-j\omega n}}{2}$$

Moreover, a representation using complex sinusoids is inherently more general.

3.2 Representation of Periodic Sequences: The Discrete-Time Fourier Series (DFS)

Since we want to represent periodic signals of period N using complex exponentials, we need to find a set of complex exponentials which contain a whole number of periods over N. Let us examine

$$w_k[n] = e^{j\omega_k n}.$$

Since we want $w_k[n]$ to contain a whole number of periods over N samples, we need to have ω_k such that

$$w_k[0] = w_k[N],$$

i.e.

$$w_k[N] = 1 = e^{j\omega_k N}.$$

Clearly this equation has N possible solutions,

$$\omega_k = \frac{2\pi}{N}k, \qquad k = 0, \cdots, N-1.$$

Therefore, if we define

$$W_N = e^{-j\frac{2\pi}{N}}$$

then the family of sequences with the property of having complete periods over N samples, is

$$w_k[n] = W_N^{-nk}, \qquad n = 0, \ldots, N-1, \qquad k = 0, \ldots, N-1.$$

That is we have defined a family of N sequences which have a complete period over N samples.
Now suppose that we want to represent the periodic signals using the family of sequences $\{w_k[n]\}_{k=0}^{N-1}$. Given a periodic sequence $\tilde{x}[n]$, we want to solve for $\tilde{X}_k, k = 0, \cdots, N-1$, in

$$\tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}_k w_k[n] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}_k e^{j\frac{2\pi}{N}nk}.$$

The factor $1/N$ has been added for notational convenience further along in the analysis.

We can write this as

$$\tilde{x}[0] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}_k \tag{3.2}$$

$$\tilde{x}[1] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}_k e^{j\frac{2\pi}{N}k} \tag{3.3}$$

$$\vdots$$

$$\tilde{x}[N-1] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}_k e^{j\frac{2\pi(N-1)}{N}k} \tag{3.4}$$

We want to express the above equations in matrix form. Therefore, we introduce

$$\mathbf{w}_k^{(k)} = \left[\ 1\ W_N^{-k}\ W_N^{-2k}\ \ldots\ W_N^{-(N-1)k}\ \right]^T \tag{3.5}$$

and the matrix Λ defined as

$$\Lambda^H = \begin{bmatrix} \left(\mathbf{w}^{(0)}\right)^T \\ \left(\mathbf{w}^{(1)}\right)^T \\ \vdots \\ \left(\mathbf{w}^{(N-1)}\right)^T \end{bmatrix}.$$

In matrix form, we have

$$\begin{bmatrix} \tilde{x}[0] \\ \tilde{x}[1] \\ \vdots \\ \tilde{x}[N-1] \end{bmatrix} = \frac{1}{N} \underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{j\frac{2\pi}{N}1} & \cdots & e^{j\frac{2\pi}{N}(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\frac{2\pi(N-1)}{N}} & \cdots & e^{j\frac{2\pi(N-1)^2}{N}} \end{bmatrix}}_{\Lambda^H} \begin{bmatrix} \tilde{X}[0] \\ \tilde{X}[1] \\ \vdots \\ \tilde{X}[N-1] \end{bmatrix}. \tag{3.6}$$

We have N unknowns and N equations, and hence if the equations are linearly independent we can expect a solution.
Now, note that

$$\begin{aligned} \left(\mathbf{w}^{(k)}\right)^H \cdot \mathbf{w}^{(m)} &= \begin{bmatrix} 1 \; e^{-j\frac{2\pi}{N}k} \; \ldots \; e^{-j\frac{2\pi}{N}k(N-1)} \end{bmatrix} \begin{bmatrix} 1 \\ e^{j\frac{2\pi}{N}m} \\ \vdots \\ e^{j\frac{2\pi}{N}m(N-1)} \end{bmatrix} \\ &= \sum_{i=0}^{N-1} e^{j\frac{2\pi}{N}(m-k)i} \\ &= \begin{cases} N & k = m \\ 0 & k \neq m \end{cases}. \end{aligned} \tag{3.7}$$

This follows from the fact that for $k \neq m$

$$\sum_{i=0}^{N-1} e^{j\frac{2\pi}{N}(m-k)i} = \frac{1 - e^{j\frac{2\pi}{N}(m-k)N}}{1 - e^{j\frac{2\pi}{N}(m-k)}} = 0.$$

Therefore, the rows of the matrix Λ in (3.6) are orthogonal (Not orthonormal, but that can be fixed by normalizing by $\sqrt{N}$.)

Hence (3.7) shows that

$$\langle \mathbf{w}^{(k)}, \mathbf{w}^{(m)} \rangle = \begin{cases} N & k = m \\ 0 & k \neq m \end{cases}$$

making $\{\mathbf{w}^{(0)}, \cdots, \mathbf{w}^{(N-1)}\}$ an orthognal set of vectors in $\mathbb{C}^N$.

Theorem 3.1 $\Lambda \; \Lambda^H = NI$

Proof

$$\begin{bmatrix} \mathbf{w}^{(0)} \\ \vdots \\ \mathbf{w}^{(N-1)} \end{bmatrix} \begin{bmatrix} \left(\mathbf{w}^{(0)}\right)^H & \cdots & \left(\mathbf{w}^{(N-1)}\right)^H \end{bmatrix} = [a_{p,q}]_{N\times N} = NI,$$

where

$$a_{p,q} = \langle \mathbf{w}^{(p)}, \mathbf{w}^{(q)} \rangle.$$

□

Hence, looking at equation (3.6), we see that

$$\begin{bmatrix} \tilde{X}[0] \\ \tilde{X}[1] \\ \vdots \\ \tilde{X}[N-1] \end{bmatrix} = \Lambda \begin{bmatrix} \tilde{x}[0] \\ \tilde{x}[1] \\ \vdots \\ \tilde{x}[N-1] \end{bmatrix} \tag{3.8}$$

$$= \begin{bmatrix} 1 & \cdots & 1 \\ e^{-j\frac{2\pi}{N}0} & \cdots & e^{-j\frac{2\pi}{N}(N-1)} \\ \vdots & \ddots & \vdots \\ e^{-j\frac{2\pi(N-1)}{N}0} & \cdots & e^{-j\frac{2\pi(N-1)^2}{N}} \end{bmatrix} \begin{bmatrix} \tilde{x}[0] \\ \tilde{x}[1] \\ \vdots \\ \tilde{x}[N-1] \end{bmatrix}. \tag{3.9}$$

Therefore,

$$\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{N}kn}. \tag{3.10}$$

Hence, we have the following representation of the periodic sequence $\{\tilde{x}[n]\}$ through $\{\tilde{X}[k]\}_{k=0}^{N-1}$ and the class of periodic exponentials $\{w_k[n]\}_{k=0}^{N-1}$ as

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}[k] w_k[n], \tag{3.11}$$

where

$$\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] w_k[n]^H. \tag{3.12}$$

Thus $\{\tilde{X}[k]\}_{k=0}^{N-1}$ can be thought of as the weights on the complex exponentials to represent a periodic sequence.

Notes

1. *Any* periodic sequence can be therefore be represented as a weighted sum of complex exponentials. This is a akin to decomposing a periodic sequence into elementary periodic functions.

2. Since $\langle \mathbf{w}^{(k)}, \mathbf{w}^{(m)} \rangle = \delta_{k-m} N$, the vectors $\mathbf{w}^{(k)}, \quad k = 0, \cdots N-1$ are orthogonal and form a *basis* of $\mathbb{C}^N$. The representation for periodic sequences is a consequence of this property. We implicitly used this in in inverting Λ in 3.6 to find the weights $\{\tilde{X}[k]\}_{k=0}^{N-1}$.

3. Extending this thought process, one can envisage expanding $\{\tilde{x}[n]\}$ in *any* basis of $\mathbb{C}^n$, which is also periodically extended.

In summary, we have the following Discrete-time Fourier Series (DFS) representation of discrete-time periodic sequences. The *synthesis formula*,

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}[k] e^{j\frac{2\pi}{N}kn}. \tag{3.13}$$

The *analysis formula*:

$$\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{N}kn}. \tag{3.14}$$

This set of equations describe how to synthesize $\{\tilde{x}[n]\}$ given the Discrete-time Fourier Series (DFS) coefficients $\{\tilde{X}[k]\}$ and how to analyze $\{\tilde{x}[n]\}$ to produce DFS coefficients $\{\tilde{X}[k]\}$.

Example 3.1 *Suppose* $x[n] = \sin(\frac{2\pi}{N}n)$
This has a period of N. *Since*

$$\begin{aligned} \sin(\frac{2\pi}{N}n) &= \frac{1}{2j} e^{j\frac{2\pi}{N}n} - \frac{1}{2j} e^{-j\frac{2\pi}{N}n} \\ &= \frac{1}{2j} e^{j\frac{2\pi}{N}n} - \frac{1}{2j} e^{j\frac{2\pi}{N}(N-1)n} \end{aligned}$$

Hence we see that

$$\tilde{X}[0] = \frac{1}{2j}, \quad \tilde{X}[N-1] = -\frac{1}{2j}, \quad \tilde{X}[k] = 0, k = 1, \cdots N-2$$

In general the synthesis and analysis equations can be written in matrix form as:

$$\tilde{\mathbf{x}} = \frac{1}{N} \Lambda^H \tilde{\mathbf{X}} \tag{3.15}$$

$$\tilde{\mathbf{X}} = \Lambda \tilde{\mathbf{x}} \tag{3.16}$$

where

$$\tilde{\mathbf{x}} = [\tilde{x}[0], \cdots, \tilde{x}[N-1]]^T$$

$$\tilde{\mathbf{X}} = [\tilde{X}[0], \cdots, \tilde{X}[N-1]]^T$$

Note that we can find the energy in one period of the sequence in terms of its Fourier series coefficients as

$$||\tilde{\mathbf{x}}||_2^2 = \sum_{n=0}^{N-1} |\tilde{x}[n]|^2 = \frac{1}{N^2}\tilde{\mathbf{X}}^H \Lambda \cdot \Lambda^H \tilde{\mathbf{X}} = \frac{1}{N}\sum_{k=0}^{N-1} |\tilde{X}[n]|^2 = \frac{1}{N}||\tilde{\mathbf{X}}||_2^2$$

Therefore, the energy in one period of the periodic signal is N-times the energy in the Fourier series coefficients.

$$N||\tilde{\mathbf{x}}||_2^2 = ||\tilde{\mathbf{X}}||_2^2$$

This is called *Parseval's relationship.*

3.2.1 Interpretation of the Fourier series

We have expressed the periodic sequence $\tilde{x}[n]$ as a weighted sum of N sinusoids,

$$\tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}[k] e^{j\frac{2\pi}{N}kn} = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}[k] w_k[n].$$

The magnitude and the phase weighting of each sinusoid $w_k[n] = e^{j\frac{2\pi}{N}kn}$ is given by $\tilde{X}[k]$, which is the Fourier series coefficient. Therefore $\tilde{X}[k]$ shows "how much" of an oscillatory behavior at freqency $\frac{2\pi}{N}k$ is contained in the periodic signal $\tilde{x}[n]$. The coefficients $\{\tilde{X}[k]\}$ can therefore be interpreted as the *spectrum* of the signal. Parseval's relationship shows that up to a scaling factor of N, the energy contained in the spectrum of the signal is the same as the energy in the signal itself, i.e. , $N\sum_{n=0}^{N-1} |\tilde{x}[n]|^2 = \sum_{n=0}^{N-1} |\tilde{X}[n]|^2$.
One can view $\{\tilde{X}[k]\}$ as just a *different* representation of $\{\tilde{x}[n]\}$.

Example 3.2 (Discrete Fourier Series) *Consider the periodic discrete signal* $\tilde{x}[n]$ *of period* 10, *defined on* $0 \le n \le 9$ *as*

$$\tilde{x}[n] = \begin{cases} 1 & 0 \le n \le 4 \\ -1 & 5 \le n \le 9 \end{cases}$$

(a) Find $\tilde{X}[k]$, the discrete Fourier series of $\tilde{x}[n]$.

(b) Compare $\tilde{X}[3]$ and $\tilde{X}[-33]$. What is the period of $\tilde{X}[n]$?

(c) Define $\tilde{y}[n] = \tilde{x}[n-5]$. What is the discrete Fourier series of $\tilde{y}[n]$?

(d) Let $\tilde{z}[n] = \tilde{X}[n]$. Find the discrete Fourier series of $\tilde{z}[n]$ and compare it to $\tilde{x}[n]$.

The answers to these questions are:

(a)

$$\begin{aligned}
\tilde{X}[k] &= \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{10}nk} \\
&= \sum_{n=0}^{4} e^{-j\frac{2\pi}{10}nk} - \sum_{n=5}^{9} e^{-j\frac{2\pi}{10}nk} \\
&= \sum_{n=0}^{4} e^{-j\frac{2\pi}{10}nk} - e^{-j\pi k} \sum_{n'=0}^{4} e^{-j\frac{2\pi}{10}n'k} \\
&= \left(1 - e^{-j\pi k}\right) \frac{1 - e^{-j\frac{2\pi}{10}5k}}{1 - e^{-j\frac{2\pi}{10}k}} \\
&= \begin{cases} 0, & \textit{for } k \textit{ even} \\ \frac{4}{1-e^{-j\frac{2\pi}{10}k}}, & \textit{for } k \textit{ odd.} \end{cases}
\end{aligned}$$

(b) We can immediately say that the period of $\tilde{X}[k]$ is $N = 10$. For any sequence of period N, it's DFS is of period N. This is a basic property of the DFS that should be known. We can easily derive this property as follows: Let $\tilde{y}[n]$ be a (any) sequence of period M. Then

$$\begin{aligned}
\tilde{Y}[k+M] &= \sum_{n=0}^{M-1} \tilde{y}[n] e^{-j\frac{2\pi}{M}n(k+M)} \\
&= \sum_{n=0}^{M-1} \tilde{y}[n] e^{-j\frac{2\pi}{M}nk} e^{-j2\pi n} \\
&= \sum_{n=0}^{M-1} \tilde{y}[n] e^{-j\frac{2\pi}{M}nk} \\
&= \tilde{Y}[k],
\end{aligned}$$

so $\tilde{Y}[k]$, the DFS of $\tilde{y}[n]$ has period M.

Since $\tilde{X}[k]$ has period 10, $\tilde{X}[-33] = \tilde{X}[7] = \tilde{X}[N-3]$.

We know that for any DFS of a real periodic sequence

$$\tilde{X}[N-k] = \tilde{X}[k]^*, \qquad k = 0, \ldots, N-1. \tag{3.17}$$

This gives $\tilde{X}[-33] = \tilde{X}[3]^*$.

Equation (3.17) can easily be derived:

$$\begin{aligned}\tilde{X}[N-k]^* &= \left(\sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{N}n(N-k)}\right)^* \\ &= \sum_{n=0}^{N-1} \tilde{x}[n]^* e^{-j\frac{2\pi}{N}nk} \\ &= \tilde{X}[k],\end{aligned}$$

where the last equality follows from the assumption that the input sequence is real.

(c) For $\tilde{y}[n] = \tilde{x}[n-5]$ we use another basic property of the DFS:

$$\tilde{x}[n-n_0] \overset{DFS}{\leftrightarrow} e^{-j\frac{2\pi}{N}n_0 k}\tilde{X}[k]. \tag{3.18}$$

Again, we can easily derive this ourselves:

$$\begin{aligned}\tilde{Y}[k] &= \sum_{n=0}^{N-1} \tilde{x}[n-n_0] e^{-j\frac{2\pi}{N}nk} \\ &= \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{N}(n+n_0)k} \\ &= e^{-j\frac{2\pi}{N}n_0 k}\tilde{X}[k].\end{aligned}$$

Using (3.18) gives us

$$\tilde{Y}[k] = e^{-j\frac{2\pi}{10}5k}\tilde{X}[k] = -\tilde{X}[k].$$

(d) We are asked to compute

$$\tilde{Z}[n] = \sum_{k=0}^{N-1} \tilde{X}[k] e^{-j\frac{2\pi}{N}nk}.$$

Remember that all sequences are periodic. If we consider $\tilde{Z}[-n]/N$, we get

$$\tilde{Z}[-n]/N = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}[k] e^{j\frac{2\pi}{N}nk},$$

which is exactly the reconstruction formula for $\tilde{x}[n]$, so $\tilde{Z}[n] = \tilde{x}[-n]/N$.

3.3 The Discrete Fourier Transform (DFT)

We will now develop a similar Fourier representation for a finite-length signal. The basic idea is to consider a signal of length N, such that

$$x[n] = 0, \qquad n \geq N, n < 0$$

and represent it as a sum of N *finite-length* complex exponentials. To do this we borrow the idea from the discrete Fourier series representation. We can think of the a-periodic finite length sequence $\{x[n]\}_{n=0}^{n=N-1}$ as a *single-period* of a periodic sequence $\tilde{x}[n]$ by constructing $\tilde{x}[n]$ as ,

$$\tilde{x}[n] = x[n], \qquad 0 \leq n \leq N-1$$

$$\tilde{x}[n] = \tilde{x}[n+N], \qquad \forall n$$

Therefore using the Fourier-series representation we have

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}kn} \tag{3.19}$$

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn} \tag{3.20}$$

In matrix form, the *Discrete Fourier Transform (DFT)* can be written as

$$\mathbf{x} = \frac{1}{N}\Lambda^H \cdot \mathbf{X} \tag{3.21}$$

$$\mathbf{X} = \Lambda \cdot \mathbf{x} \tag{3.22}$$

where $\mathbf{x} = \big[x[0]x[1]\ldots x[N-1]\big]^T$ and $\mathbf{x} = \big[X[0]X[1]\ldots X[N-1]\big]^T$.

Therefore, another interpretation of the DFT is in terms of basis representation. Given any vector $\mathbf{x} \in \mathbb{C}$, we can think of

$$\mathbf{x} = \sum_{n=0}^{N-1} x[n]\mathbf{e}^n$$

where $\mathbf{e}^n \in \mathbb{C}^N$ is the "unit vector"

$$\mathbf{e}^n = \begin{bmatrix} e_0^{(n)} \\ e_1^{(n)} \\ \vdots \\ e_{N-1}^{(n)} \end{bmatrix} \in \mathbb{C}^N$$

with components

$$e_i^{(n)} = \begin{cases} 1 & i = n \\ 0 & i \neq n \end{cases}$$

By looking at the DFT, we see that

$$\mathbf{x} = \frac{1}{N}\Lambda^H \cdot \mathbf{X}$$

where

$$\Lambda^H = \begin{bmatrix} \left(\mathbf{w}^{(0)}\right)^T \\ \left(\mathbf{w}^{(1)}\right)^T \\ \vdots \\ \left(\mathbf{w}^{(N-1)}\right)^T \end{bmatrix} = \begin{bmatrix} 1 & \cdots & 1 \\ e^{j\frac{2\pi}{N}0} & \cdots & e^{j\frac{2\pi}{N}(N-1)} \\ \vdots & \ddots & \vdots \\ e^{j\frac{2\pi(N-1)}{N}0} & \cdots & e^{j\frac{2\pi(N-1)^2}{N}} \end{bmatrix}$$

and

$$\mathbf{w}^{(k)} = \begin{bmatrix} 1 \\ e^{j\frac{2\pi}{N}k} \\ \vdots \\ e^{j\frac{2\pi(N-1)}{N}k} \end{bmatrix} \in \mathbb{C}^{N\times 1}$$

is an expansion of $\mathbf{x}$ in the orthogonal *basis*,

$$\left\{ \left(\mathbf{w}^{(0)}\right)^T, \cdots, \left(\mathbf{w}^{(N-1)}\right)^T \right\}$$

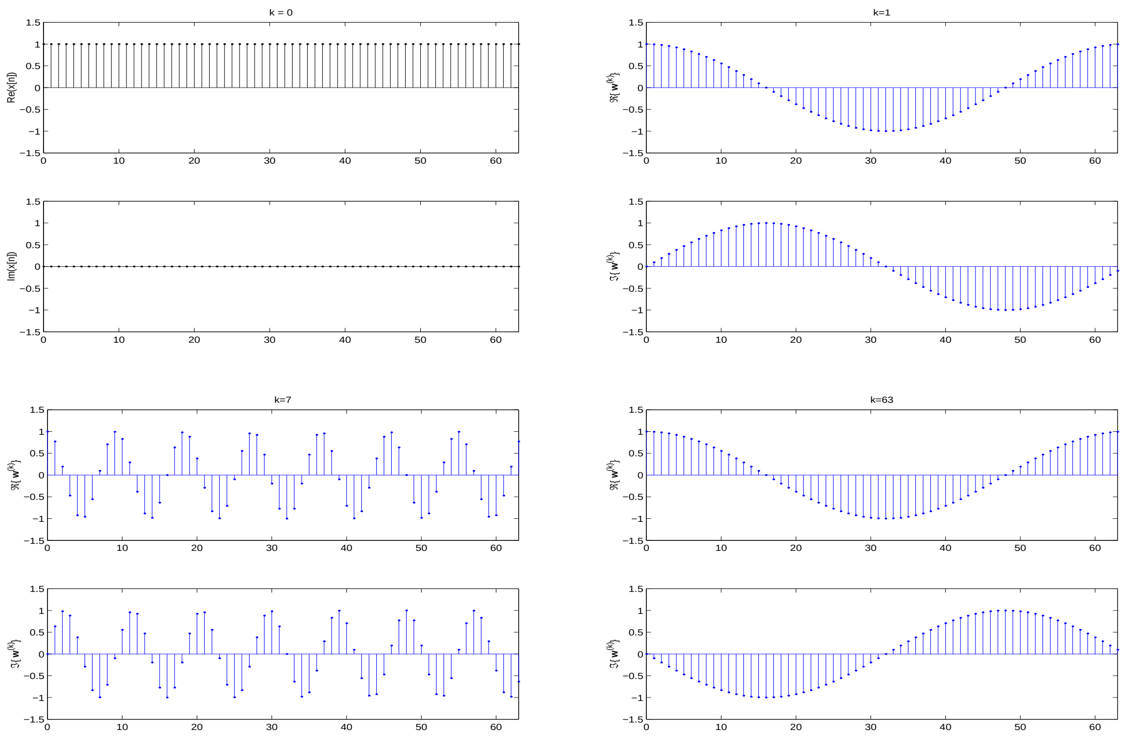


Figure 3.2: Some DFT basis vectors $\mathbf{w}^{(k)}$ for $N = 64$; $k = 0, 1, 7$ and 63.

with coefficients $\frac{1}{N}\tilde{X}[k]$. This gives another interpretation of the DFT as an expansion of a sequence in another "basis" set, and therefore giving it an alternate representation. This viewpoint is actually quite useful and general and to be able to utilize it we make a detour to understand vector spaces.

Example 3.3 (Discrete Fourier Transform) *Derive the DFT for a general sinusoidal sequence,* $\tilde{x}[n] = \sin(\frac{2\pi L}{N}n + \theta)$, $n = 0, \ldots, N$.

For L a positive integer we have

$$\begin{aligned}
X[k] &= \sum_{n=0}^{N-1} \sin(\frac{2\pi L}{N} n + \theta) e^{-j\frac{2\pi}{N}nk} \\
&= \sum_{n=0}^{N-1} \frac{e^{j(\frac{2\pi L}{N}n+\theta)} - e^{-j(\frac{2\pi L}{N}n+\theta)}}{2j} e^{-j\frac{2\pi}{N}nk} \\
&= \frac{e^{j(-\pi/2+\theta)}}{2} \sum_{n=0}^{N-1} e^{-j(\frac{2\pi}{N}n(k-L)} + \frac{e^{j(\pi/2-\theta)}}{2} \sum_{n=0}^{N-1} e^{-j(\frac{2\pi}{N}n(k+L)} \\
&= \begin{cases} \frac{N}{2} e^{j(-\pi/2+\theta)} & k = L \\ \frac{N}{2} e^{j(\pi/2-\theta)} & k = N - L. \end{cases}
\end{aligned}$$

3.4 Properties of the DFS

Symmetries & Structure. The DFS of a ***real*** sequence $\tilde{x}[n] \in \mathbb{R}$ possesses the following symmetries:

$$\tilde{X}[k] = \tilde{X}^*[-k] \qquad \textit{the transform is conjugate-symmetric} \tag{3.23}$$

$$|\tilde{X}[k]| = |\tilde{X}[-k]| \qquad \textit{the magnitude is symmetric} \tag{3.24}$$

$$\measuredangle \tilde{X}[k] = -\measuredangle \tilde{X}[-k] \qquad \textit{the phase is antisymmetric} \tag{3.25}$$

$$\mathrm{Re}\{\tilde{X}[k]\} = \mathrm{Re}\{\tilde{X}[-k]\} \qquad \textit{the real part is symmetric} \tag{3.26}$$

$$\mathrm{Im}\{\tilde{X}[k]\} = -\mathrm{Im}\{\tilde{X}[-k]\} \qquad \textit{the imaginary is antisymmetric} \tag{3.27}$$

Finally, if $x[n]$ is real and symmetric (using the symmetry definition in (3.34)), then the DFS is real:

$$\tilde{x}[k] = \tilde{x}[-k] \Longleftrightarrow \tilde{X}[k] \in \mathbb{R} \tag{3.28}$$

while, for real antisymmetric signals we have that the DFS is purely imaginary.

Linearity & Shifts. The DFS is a linear operator, since it is a matrix-vector product. A shift in the discrete-time domain leads to multiplication by a phase term in the frequency domain:

$$\tilde{x}[n - n_0] \overset{\text{DFS}}{\longleftrightarrow} W_N^{kn_0} \tilde{X}[k] \tag{3.29}$$

while multiplication of the signal by a complex exponential of frequency a multiple of $2\pi/N$ leads to a shift in frequency:

$$W_N^{-nL} \tilde{x}[n] \overset{\text{DFS}}{\longleftrightarrow} \tilde{X}[k - L]. \tag{3.30}$$

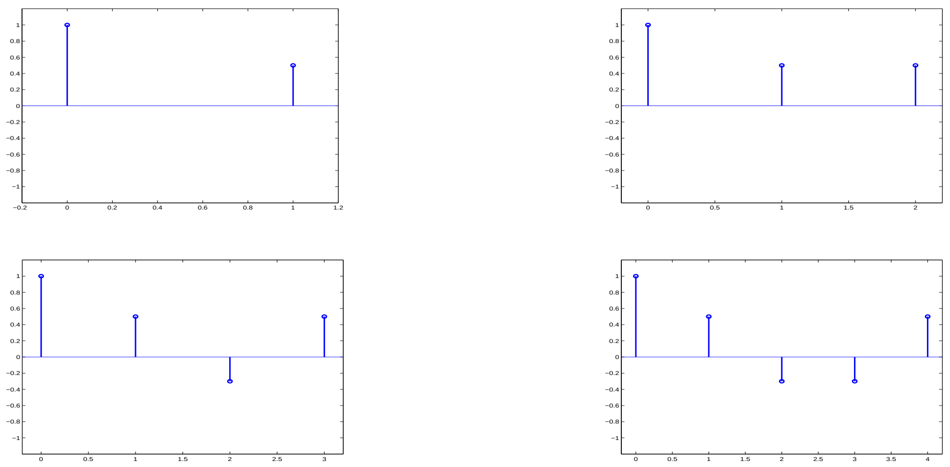


Figure 3.3: Examples of finite-length symmetric signals for $N = 2, 3, 4, 5$.

Energy Conservation. We have already seen the conservation of energy property in the context of basis expansion. Here, we will simply recall Parseval's Theorem, which states:

$$\sum_{n=0}^{N-1} |\tilde{x}[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |\tilde{X}[k]|^2. \tag{3.31}$$

3.5 Properties of the DFT

The properties of the DFT are obviously the same as those for the DFS, given the formal equivalence of the transforms. The only detail is how to interpret shifts, index reversal and symmetries for finite, length-N vectors; this is easily solved by considering the fact that the DFT subsumes an N-periodic structure and therefore the underlying model for the signal is that of periodic extension. We can therefore consider the periodized version of the signal, operate the shift and then take the values from 0 to $N-1$. Explicitly, shifts and index reversal of a length-N vector are carried out modulo N; the reversal of a signal $x[n] = \mathbf{x} = [x[0]\ x[1]\ \ldots\ x[N-1]]$ is:

$$x[-n \mod N] = [x[0]\ x[N-1]\ x[N-2]\ \ldots\ x[2]\ x[1]] \tag{3.32}$$

whereas its shift by k is the circular shift:

$$x[(n-k) \mod N] = [x[k]\ x[k-1]\ \ldots\ x[0]\ x[N-1]\ x[N-2]\ \ldots\ x[k+1]]. \tag{3.33}$$

This implies that, when we say a length-N signal $x[k]$ is *symmetric*, we have in fact:

$$x[k] = x[N-k], \quad k = 1, 2, \ldots, \lfloor (N-1)/2 \rfloor; \tag{3.34}$$

note that the index k starts off at one in the above definition and ends at the *floor* of $(N-1)/2$; this means that X[0] is always unconstrained and so is $x[N/2]$ for even-length signals. Figure 3.3 shows some examples of symmetric length-N signals for different values of N. Of course the same definition can be used for antisymmetric signals with just a change of sign.

Symmetries & Structure. The DFT of a ***real*** sequence $x[n] \in \mathbb{R}$ possesses the following symmetries:

$$X[k] = X^*[-k \mod N] \qquad \textit{the transform is conjugate-symmetric} \tag{3.35}$$

$$|X[k]| = |X[-k \mod N]| \qquad \textit{the magnitude is symmetric} \tag{3.36}$$

$$\measuredangle X[k] = -\measuredangle X[-k \mod N] \qquad \textit{the phase is antisymmetric} \tag{3.37}$$

$$\mathrm{Re}\{X[k]\} = \mathrm{Re}\{X[-k \mod N]\} \qquad \textit{the real part is symmetric} \tag{3.38}$$

$$\mathrm{Im}\{X[k]\} = -\mathrm{Im}\{X[-k \mod N]\} \qquad \textit{the imaginary is antisymmetric} \tag{3.39}$$

Finally, if $x[n]$ is real and symmetric (using the symmetry definition in (3.34), then the DFT is real:

$$x[k] = x[N-k], \quad k = 1, 2, \ldots, \lfloor (N-1)/2 \rfloor \Longleftrightarrow X[k] \in \mathbb{R} \tag{3.40}$$

while, for real antisymmetric signals we have that the DFT is purely imaginary.

Linearity & Shifts. The DFT is obviously a linear operator. A circular shift in the discrete-time domain leads to multiplication by a phase term in the frequency domain:

$$x[(n-n_0) \mod N] \overset{\mathrm{DFT}}{\longleftrightarrow} W_N^{kn_0} X[k] \tag{3.41}$$

while the finite-length equivalent of the Modulation theorem states:

$$W_N^{-nL} x[n] \overset{\mathrm{DFT}}{\longleftrightarrow} X[(k-L) \mod N]. \tag{3.42}$$

Energy Conservation. See (3.31).

3.6 Summary

This chapter introduced the concept of Fourier Transform for digital signals. The main points have been:

- A review of complex exponentials, finding a set of orthogonal complex exponentials;
- The DFS for periodic sequences;

- The DFT as a change of basis in $\mathbb{C}^N$, both in matrix and explicit form;
- Symmetries and structures of the two transforms.

Here is a tables of common DFT transforms:

Some DFT pairs for length-N signals:	$(n, k = 0, 1, \ldots, N-1)$
$x[n] = \delta[n-k]$	$X[k] = e^{-j\frac{2\pi}{N}k}$
$x[n] = 1$	$X[k] = N\delta[k]$
$x[n] = e^{j\frac{2\pi}{N}L}$	$X[k] = N\delta[k-L]$
$x[n] = \cos(\frac{2\pi}{N}Ln + \phi)$	$X[k] = (N/2)[e^{j\phi}\delta[k-L] + e^{-j\phi}\delta[k-N+L)]]$
$x[n] = \sin(\frac{2\pi}{N}Ln + \phi)$	$X[k] = (-jN/2)[e^{j\phi}\delta[k-L] - e^{-j\phi}\delta[k-N+L]]$
$x[n] = \begin{cases} 1 & \text{for } n \leq M-1 \\ 0 & \text{for } M \leq n \leq N-1 \end{cases}$	$X[k] = \dfrac{\sin((\pi/N)Mk)}{\sin((\pi/N)k)}\, e^{-j\frac{\pi}{N}(M-1)k}$

3.7 Problems

Problem 3.1 *Derive the formula for the DFT of the length-N signal*

$$x[n] = \cos((2\pi/N)Ln + \phi).$$

Problem 3.2 *Consider a length-64 signal $x[n]$ which is the sum of the three sinusoidal signals plotted in Figure 3.4. Compute the DFT coefficients $X[k], k = 0, 1, \ldots, 63$ using the results from Problem 3.1.*

Problem 3.3 *The DFT and inverse DFT (IDFT) formulas are similar, but not identical. Consider a length-N signal $x[n], N = 0, \ldots, N-1$; what is the length-N signal $y[n]$ obtained as*

$$y[n] = DFT\{DFT\{x[n]\}\}$$

(i.e. by applying the DFT algorithm twice in a row)?

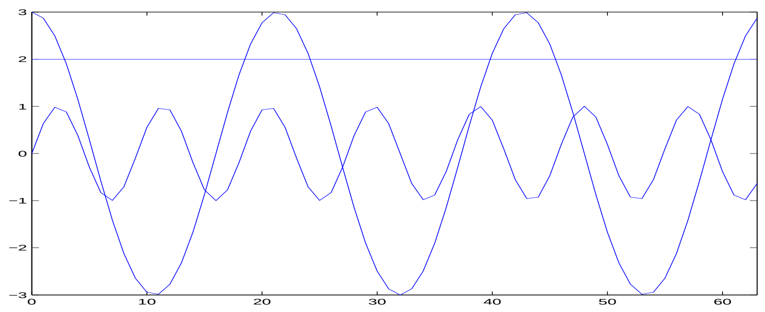


Figure 3.4: Three sinusoidal signals.

Problem 3.4 (Implementing DFT in MATLAB) *In this exercise we want to provide a simple m-file in MATLAB to compute the discrete Fourier transform of a given sequence. The inputs of the function are the input sequence x as a row-vector and the length of the transform N. It checks the length of x to be satisfied with N. Then a transformation matrix W will be formed and the DFT vector X will be produced by a matrix-vector multiplication. The magnitude of the DFT should be plotted at the end.*

Download the m-file `myDFT` from the course website and put it into your `work` directory. Fill the blanks and run the function to compute and plot the DFT of $x[n]$, $n = 0, \ldots, 45$, given in Exercise 1. Read MATLAB help for the standard function "`fft`". Compare the output of your function to output of `fft`.

Problem 3.5 (DFT with Different Lengths) *Consider the finite length sequences*

$$x[n] = y[n] = \begin{cases} 1 & 0 \leq n \leq 5 \\ 0 & \textit{otherwise} \end{cases}.$$

(a) Use your `myDFT` m-file to find the DFT of length 6 for $x[n]$.

(b) Repeat part (a) to compute length 12 DFT of $x[n]$.

Let $a[n]$ and $b[n]$ be two length N sequences. The circular convolution *of the two sequences $a[n]$ and $b[n]$ is defined as*

$$a[n] \otimes b[n] = \sum_{m=0}^{N-1} a[m] b[(n-m) \bmod N].$$

Note that $b[(n-m) \bmod N]$ is a circular shifted version of $b[n]$, i.e.

$$b[(n-m) \bmod N] = \Big[b[N-m]\ b[N-m+1]\ \ldots\ b[N-1]\ b[0]\ \ldots\ b[N-m-1]\Big].$$

In the remaining parts of this exercise we are going to implement the circular convolution in MATLAB.

(c) Download the `rcshift` m-file from the website and fill the blanks. At the end compute the circular shift of length 3 to the right of `t(1:10)=sin([1:10])`.

(d) Download the `cir_conv` and complete it according to its comments.

(e) Use your `cir_conv` m-file to compute the 6-point circular convolution of $x[n]$ and $y[n]$.

(f) Compute the 12-point circular convolution of $x[n]$ and $y[n]$ and call it in $z[n]$.

(g) Compare the results of the two circular convolutions.

(h) compare the DFT of $z[t]$ to the multiplication of DFT's of $x[n]$ and $y[n]$.

Problem 3.6 *Compute the DFS of $x[n] = \cos(\pi \frac{n}{3})$ and $y[n] = 1 + \cos(\pi \frac{n}{3})$*

Chapter 4

Signals and Hilbert Space

In the 17th century, algebra and geometry started to interact in a fruitful synergy which continues to the present day. Descartes's original idea of translating geometric constructs into algebraic form spurred a new line of attack in mathematics; soon, a series of astonishing results was produced for a number of problems which had long defied geometrical solutions (such as, famously, the trisection of the angle). It also spearheaded the notion of vector space, in which a geometrical point could be represented as an n-tuple of coordinates; this, in turn, readily evolved into the theory of linear algebra. Later, the concept proved useful in the opposite direction: many algebraic problems could benefit from our innate geometrical intuition once they were cast in vector form; from the easy three-dimensional visualization of concepts such as distance and orthogonality, more complex algebraic constructs could be brought within the realm of intuition. The final leap of imagination came with the realization that the concept of vector space could be applied to much more abstract entities such as infinite-dimensional objects and functions. In so doing, however, spatial intuition could be of limited help and so the notion of vector space had to be formalized in much more rigorous terms; we will see that the definition of Hilbert space is one such formalization.

Most of the signal processing theory which we will study in the course can be usefully cast in terms of vector notation and the advantages of this approach are exactly what we just delineated before. First of all, all the standard machinery of linear algebra becomes immediately available and applicable; this greatly simplifies the formalism used in the mathematical proofs which will follow and, at the same time, it fosters a good intuition with respect to the underlying principles which are being put in place. Furthermore, the vector notation creates a frame of thought which seamlessly links the more abstract results involving infinite sequences to the algorithmic reality involving finite-length signals. Finally, on the practical side, vector notation is the standard paradigm for numerical analysis packages such as Matlab; signal processing algorithm expressed in vector notation translate to working code with very little effort.

In the previous chapter we established the basic notation for the different classes of discrete-time signals which we will encounter time and again in the rest of the course and we hinted at the fact that a tight correspondence can be established between the concept of signal and that of vector space. In this chapter we will pursue this link further, firstly by reviewing the familiar Euclidean spaces in finite dimensions and then by extending the concept of basic vector spaces to infinite-dimensional Hilbert spaces.

4.1 A Quick Review of Euclidean Geometry

Euclidean geometry is a straightforward formalization of our spatial sensory experience; hence its cornerstone role in developing a basic intuition for vector spaces. Everybody is (or should be) familiar with Euclidean geometry and the natural "physical" spaces like $\mathbb{R}^2$ (the plane) and $\mathbb{R}^3$ (the three-dimensional space). The notion of *distance* is clear, *orthogonality* is intuitive and maps to the idea of a "right angle". Even a more abstract concept such as that of *basis* is rather easy to think of (the standard coordinate concepts of latitude, longitude and height, which correspond to the three orthogonal axes in $\mathbb{R}^3$). Unfortunately, immediate spatial intuition fails us for higher dimensions (i.e. for $\mathbb{R}^N$ with $N > 3$), yet the basic concepts introduced for $\mathbb{R}^3$ generalize easily to $\mathbb{R}^N$ so that it is easier to state such concepts for the higher-dimensional case and specialize them with examples for $N = 2$ or $N = 3$. These notions ultimately will be generalized even further to more abstract types of vector spaces. For the moment, let us review the properties of $\mathbb{R}^N$, the N-dimensional Euclidean space.

Vectors and Notation. A point in $\mathbb{R}^N$ is specified by an N-tuple of coordinates[1]:

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} = [x_0\ x_1\ \dots\ x_{N-1}]^T$$

where $x_i \in \mathbb{R},\ i = 0, 1, \dots, N-1$. We call this set of coordinates a *vector* and the N-tuple will be denoted synthetically by the symbol $\mathbf{x}$; coordinates are usually expressed with

[1] N-dimensional vectors are by default *column* vectors.

respect to a "standard" orthonormal basis[2]. The vector $\mathbf{0} = [0\ 0\ \ldots\ 0]^T$, i.e. the null vector, is considered the origin of the coordinate system.

The generic n-th element in vector $\mathbf{x}$ is indicated by the subscript $\mathbf{x}_n$. In the following we will often consider a *set* of M arbitrarily chosen vectors in $\mathbb{R}^N$ and this set will be indicated by the notation $\{\mathbf{x}^{(k)}\}_{k=0\ldots M-1}$. Each vector in the set is indexed by the superscript $\cdot^{(k)}$. The n-th element of the k-th vector in the set is indicated by the notation $\mathbf{x}_n^{(k)}$

Inner Product. The inner product between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$ is defined as:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{n=0}^{N-1} x_n y_n \tag{4.1}$$

We say that $\mathbf{x}$ and $\mathbf{y}$ are orthogonal, or $\mathbf{x} \perp \mathbf{y}$, when the inner product is zero:

$$\mathbf{x} \perp \mathbf{y} \iff \langle \mathbf{x}, \mathbf{y} \rangle = 0 \tag{4.2}$$

Norm. The norm of a vector is defined in terms of the inner product as

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{n=0}^{N-1} x_n^2} = \langle \mathbf{x}, \mathbf{x} \rangle^{1/2} \tag{4.3}$$

It is easy to visualize geometrically that the norm of a vector corresponds to its length, i.e. to the distance between the origin and the point identified by the vector's coordinates. A remarkable property linking the inner product and the norm is the Cauchy-Schwarz inequality (whose proof is a little tricky); given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$ we *always* have:

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

Distance. The concept of norm is used to introduce the notion of Euclidean *distance* between two vectors $\mathbf{x}$ and $\mathbf{y}$:

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{n=0}^{N-1} (x_n - y_n)^2}. \tag{4.4}$$

From this, we can easily derive the Pythagorean theorem for N dimensions: if two vectors are orthogonal, $\mathbf{x} \perp \mathbf{y}$, and we consider the sum vector $\boldsymbol{z} = \mathbf{x} + \mathbf{y}$, we have:

$$\|\boldsymbol{z}\|_2^2 = \|\mathbf{x}\|_2^2 + \|\mathbf{y}\|_2^2 \tag{4.5}$$

[2]The concept of basis will be defined more precisely later on; for the time being, consider a standard set of orthogonal axes.

The above properties are graphically shown in Figure 4.1 for $\mathbb{R}^2$.

Bases. Consider a set of M arbitrarily chosen vectors in $\mathbb{R}^N$: $\{\mathbf{x}^{(k)}\}_{k=0\ldots M-1}$. Given such a set, a key question is that of completeness: can *any* vector in $\mathbb{R}^N$ be written as a linear combination of vectors from the set? In other words, we ask ourselves whether for any $\mathbf{z} \in \mathbb{R}^N$ we can find a set of M coefficients $\alpha_k \in \mathbb{R}$ such that $\mathbf{z}$ can be expressed as:

$$\mathbf{z} = \sum_{k=0}^{M-1} \alpha_k \mathbf{x}^{(k)}. \tag{4.6}$$

Clearly, M needs to be greater or equal to N, but what conditions does a set of vectors $\{\mathbf{x}^{(k)}\}_{k=0\ldots M-1}$ need to satisfy so that (4.6) holds for any $\mathbf{z} \in \mathbb{R}^N$? There needs to be a set of M vectors that *span* $\mathbb{R}^N$, and it can be shown that this is equivalent to saying that the set must contain at least N *linearly independent* vectors. In turn, N vectors $\{\mathbf{y}^{(k)}\}_{k=0\ldots N-1}$ are linearly independent if the equation

$$\sum_{k=0}^{N-1} \beta_k \mathbf{y}^{(k)} = 0 \tag{4.7}$$

is satisfied only when all the β_k's are zero. A set of N linearly independent vectors for $\mathbb{R}^N$ is called a *basis* and, amongst bases, the ones with mutually orthogonal vectors of norm equal to one are called *orthonormal bases*. For an orthonormal basis $\{\mathbf{y}^{(k)}\}$ we therefore have:

$$\langle \mathbf{y}^{(k)}, \mathbf{y}^{(h)} \rangle = \begin{cases} 1 & \text{if } k = h \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

Figure 4.2 reviews the above concepts in low dimensions.

The standard orthonormal basis for $\mathbb{R}^N$ is the *canonical basis* $\{\boldsymbol{\delta}^{(k)}\}_{k=0\ldots N-1}$ with

$$\boldsymbol{\delta}_n^{(k)} = \delta[n-k] = \begin{cases} 1 & \text{if } n = k \\ 0 & \text{otherwise} \end{cases}$$

The orthonormality of such a set is immediately apparent. Another important orthonormal basis for $\mathbb{R}^N$ is the normalized *Fourier basis* $\{\mathbf{w}^{(k)}\}_{k=0\ldots N-1}$ for which:

$$\mathbf{w}_n^{(k)} = \frac{1}{\sqrt{N}} e^{-j\frac{2\pi}{N}nk};$$

the orthonormality proof for the basis is left as an exercise.

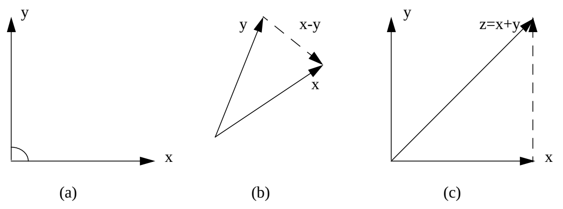


Figure 4.1: Elementary properties of vectors in $\mathbb{R}^2$. (a) Orthogonality of two vectors $\mathbf{x}$ and $\mathbf{y}$. (b) Difference vector $\mathbf{x} - \mathbf{y}$ and distance between $\mathbf{x}$ and $\mathbf{y}$. (c) Sum of two orthogonal vectors $\boldsymbol{z} = \mathbf{x} + \mathbf{y}$, and Pythagorean theorem.

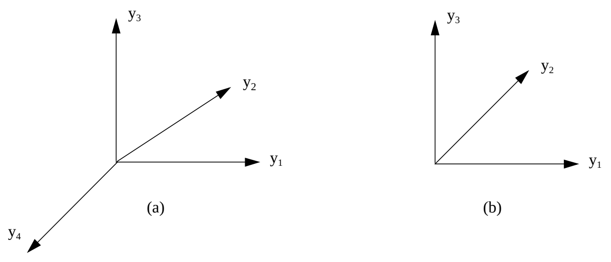


Figure 4.2: Linear independence and bases. (a) $(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \mathbf{y}^{(3)})$ are coplanar in $\mathbb{R}^3$, and so do not form a basis. $\mathbf{y}^{(4)}$ and any two of $\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \mathbf{y}^{(3)}\}$ are linearly independent. (b) Any two of $\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \mathbf{y}^{(3)}\}$ form a basis, and $\{\mathbf{y}^{(1)}, \mathbf{y}^{(3)}\}$ form an orthogonal basis.

4.2 From Vector Spaces to Hilbert Spaces

The purpose of the previous review was to briefly review the elementary notions and spatial intuitions of Euclidean geometry. A thorough study of vectors in $\mathbb{R}^N$ and $\mathbb{C}^N$ is the subject of linear algebra; yet, the idea of vectors, orthogonality and bases is much more general, the basic ingredients being an inner product and the use of a square norm as in (4.3).

While the analogy between vectors in $\mathbb{C}^N$ and length-N signal is readily apparent, the question now hinges on how we are to proceed in order to generalize the above concepts to the class of infinite sequences. Intuitively, for instance, we can let N grow to infinity and obtain $\mathbb{C}^\infty$ as the Euclidean space for infinite sequences; in this case, however, much care must be exercised with expressions such as (4.1) and (4.3) which can diverge for sequences as simple as $x[n] = 1$ for all n. In fact, the proper generalization of $\mathbb{C}^N$ to an infinite number of dimensions is in the form of a particular vector space called *Hilbert space*; the structure of this kind of vector space imposes a set of constraints on its elements so that divergence problems such as the one we just mentioned no longer bother us. When we embed infinite sequences into a Hilbert space, these constraints translate to the condition that the corresponding signals have finite energy — which is a mild and reasonable requirement.

Finally, it is important to remember the notion of Hilbert space is applicable to much more general vector spaces than $\mathbb{C}^N$; for instance, we can easily consider spaces of functions over an interval or over the real line. This generality is actually the cornerstone of a branch of mathematics called *functional analysis*. While we will not tread very far into these kind of generalizations, we will certainly point out a few of them along the way. The space of square integrable functions, for instance, will turn out to be a marvelous tool a few chapters from now when, finally, the link between continuous- and discrete-time signals will be explored in detail.

4.2.1 The Recipe for Hilbert Space

A word of caution: we are now starting to operate in a world of complete abstraction. Here a vector is an entity *per se* and, while analogies and examples in terms of Euclidean geometry can be useful visually, they are by no means exhaustive. In other words: vectors are no longer just N-tuples of numbers; they can be anything. This said, a Hilbert space can be defined in incremental steps starting from an general notion of vector space and by supplementing this space with two additional features: the existence of an inner product and the property of completeness.

Vector Space. Consider a set of vectors V and a set of scalars S (which can be either $\mathbb{R}$

or $\mathbb{C}$ for our purposes). A vector space $H(V, S)$ is completely defined by the existence of a vector addition operation and a scalar multiplication operation which satisfy the following properties for any $\mathbf{x}, \mathbf{y}, \mathbf{z}, \in V$ and any $\alpha, \beta \in S$:

- Addition is commutative:

$$\mathbf{x} + \mathbf{y} = \mathbf{y} + \mathbf{x} \tag{4.9}$$

- Addition is associative:

$$(\mathbf{x} + \mathbf{y}) + \mathbf{z} = \mathbf{x} + (\mathbf{y} + \mathbf{z}) \tag{4.10}$$

- Scalar multiplication is distributive:

$$\alpha(\mathbf{x} + \mathbf{y}) = \alpha\mathbf{x} + \alpha\mathbf{y} \tag{4.11}$$

$$(\alpha + \beta)\mathbf{x} = \alpha\mathbf{x} + \beta\mathbf{x} \tag{4.12}$$

$$\alpha(\beta\mathbf{x}) = (\alpha\beta)\mathbf{x} \tag{4.13}$$

- There exists a null vector $\mathbf{0}$ in V which is the additive identity so that $\forall\, \mathbf{x} \in V$:

$$\mathbf{x} + \mathbf{0} = \mathbf{0} + \mathbf{x} = \mathbf{x} \tag{4.14}$$

- $\forall\, \mathbf{x} \in V$ there exists in V an additive inverse $-\mathbf{x}$ such that:

$$\mathbf{x} + (-\mathbf{x}) = (-\mathbf{x}) + \mathbf{x} = \mathbf{0} \tag{4.15}$$

- There exists an identity element "1" for *scalar* multiplication so that $\forall\, \mathbf{x} \in V$:

$$1 \cdot \mathbf{x} = \mathbf{x} \cdot 1 = \mathbf{x}. \tag{4.16}$$

Inner Product Space. What we have so far is the simplest type of vector space; the next ingredient which we will consider is the *inner product* which is essential to build a notion of *distance* between elements in a vector space. A vector space with an inner product is called an inner product space. An inner product for $H(V, S)$ is a function from $V \times V$ to S which satisfies the following properties for any $\mathbf{x}, \mathbf{y}, \mathbf{z}, \in V$:

- It is *distributive* with respect to vector addition:

$$\langle \mathbf{x}+\mathbf{y}, \mathbf{z}\rangle = \langle \mathbf{x}, \mathbf{z}\rangle + \langle \mathbf{y}, \mathbf{z}\rangle \tag{4.17}$$

- It possesses the *scaling property* with respect to scalar multiplication[3]:

$$\langle \mathbf{x}, \alpha\mathbf{y}\rangle = \alpha\langle \mathbf{x}, \mathbf{y}\rangle \tag{4.18}$$

$$\langle \alpha\mathbf{x}, \mathbf{y}\rangle = \alpha^*\langle \mathbf{x}, \mathbf{y}\rangle \tag{4.19}$$

- It is commutative within complex conjugation:

$$\langle \mathbf{x}, \mathbf{y}\rangle = \langle \mathbf{y}, \mathbf{x}\rangle^* \tag{4.20}$$

- The self-product is real and positive:

$$\langle \mathbf{x}, \mathbf{x}\rangle \geq 0 \tag{4.21}$$

$$\langle \mathbf{x}, \mathbf{x}\rangle \in \mathbb{R}$$

$$\langle \mathbf{x}, \mathbf{x}\rangle = 0 \Longleftrightarrow \mathbf{x} = \mathbf{0}. \tag{4.22}$$

From this definition of the inner product a series of additional definitions and properties can be derived: first of all, orthogonality between two vectors is defined with respect to the inner product, and we will say that non-zero vectors $\mathbf{x}$ and $\mathbf{y}$ are orthogonal, or $\mathbf{x} \perp \mathbf{y}$, if and only if

$$\langle \mathbf{x}, \mathbf{y}\rangle = 0 \tag{4.23}$$

From the definition of an inner product we can define the *norm* of a vector as:

$$||\mathbf{x}|| = \langle \mathbf{x}, \mathbf{x}\rangle^{1/2} \tag{4.24}$$

In turn, the norm satisfies the *Cauchy-Schwartz inequality*:

$$|\langle \mathbf{x}, \mathbf{y}\rangle| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\| \tag{4.25}$$

with strict equality if and only if $\mathbf{x} = \alpha\mathbf{y}$.

[3]Note that in our notation, the left operand is conjugated.

Proof: If $\mathbf{y} = \mathbf{0}$, then (4.25) holds since $\langle \mathbf{x}, \mathbf{0} \rangle = 0$.
If $\mathbf{y} \neq \mathbf{0}$, then for every scalar α we have

$$0 \leq \|\mathbf{x} - \alpha\mathbf{y}\|^2 = \langle \mathbf{x} - \alpha\mathbf{y}, \mathbf{x} - \alpha\mathbf{y} \rangle = \langle \mathbf{x}, \mathbf{x} \rangle - \alpha\langle \mathbf{x}, \mathbf{y} \rangle - \alpha^* \left[\langle \mathbf{y}, \mathbf{x} \rangle - \alpha\langle \mathbf{y}, \mathbf{y} \rangle\right].$$

If we choose $\alpha = \frac{\langle \mathbf{y},\mathbf{x} \rangle}{\langle \mathbf{y},\mathbf{y} \rangle}$ then we have

$$0 \leq \langle \mathbf{x}, \mathbf{x} \rangle - \frac{\langle \mathbf{y}, \mathbf{x} \rangle}{\langle \mathbf{y}, \mathbf{y} \rangle}\langle \mathbf{x}, \mathbf{y} \rangle = \frac{\langle \mathbf{x}, \mathbf{x} \rangle\langle \mathbf{y}, \mathbf{y} \rangle - |\langle \mathbf{x}, \mathbf{y} \rangle|^2}{\langle \mathbf{y}, \mathbf{y} \rangle},$$

or

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\|,$$

with equality iff $\mathbf{x} = \alpha\mathbf{y}$. ■

The norm also satisfies the *triangle inequality*:

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\| \tag{4.26}$$

with strict equality if and only if $\mathbf{x} = \alpha\mathbf{y}$ and $\alpha \in \mathbb{R}^+$.

Proof:

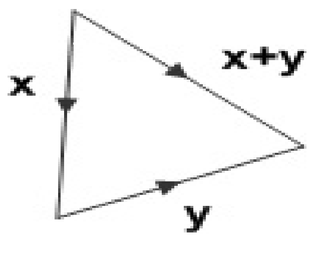
$$\|\mathbf{x} + \mathbf{y}\|^2 = \langle \mathbf{x} + \mathbf{y}, \mathbf{x} + \mathbf{y} \rangle = \|\mathbf{x}\|^2 + \langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{y}, \mathbf{x} \rangle + \|\mathbf{y}\|^2.$$

Since for any complex number u, $\mathrm{Re}(u) \leq |u|$ and $\mathrm{Im}(u) \leq |u|$, we have $\langle \mathbf{x}, \mathbf{y} \rangle \leq |\langle \mathbf{x}, \mathbf{y} \rangle|$, and $\langle \mathbf{y}, \mathbf{x} \rangle \leq |\langle \mathbf{x}, \mathbf{y} \rangle|$.
Hence

$$\|\mathbf{x} + \mathbf{y}\|^2 \leq \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 + 2|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 + 2\|\mathbf{x}\|\|\mathbf{y}\| = \left(\|\mathbf{x}\| + \|\mathbf{y}\|\right)^2.$$

Taking square roots we get

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|. \quad ■$$



For orthogonal vectors, the triangle inequality becomes the famous Pythagorean Theorem:

$$\|\mathbf{x}+\mathbf{y}\|^2 = \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 \quad \text{for } \mathbf{x} \perp \mathbf{y} \tag{4.27}$$

Hilbert Space. A vector space $H(V, S)$ equipped with an inner product is called an inner product space. To obtain a Hilbert space, we need completeness. This is a slightly more technical notion, which essentially implies that convergent sequences of vectors in V have a limit that is also in V. To gain intuition, think of the set of rational numbers $\mathbb{Q}$ versus the set of real numbers $\mathbb{R}$. The set of rational numbers is incomplete, because there are convergent sequences in $\mathbb{Q}$ which converge to irrational numbers. The set of real numbers contains these irrational numbers, and is in that sense the completion of $\mathbb{Q}$. Completeness is usually hard to prove in the case of infinite-dimensional spaces; in the following it will be tacitly assumed and the interested reader can easily find the relevant proofs in advanced analysis textbooks. As a last technicality, we will also only consider *separate* Hilbert spaces, which are the ones that admit orthonormal bases.

4.2.2 Examples of Hilbert Spaces

Finite Euclidean Spaces. The vector space $\mathbb{C}^N$, with the "natural" definition for the sum of two vectors $\mathbf{z} = \mathbf{x} + \mathbf{y}$ as:

$$z_n = x_n + y_n \tag{4.28}$$

and the definition of the inner product as:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{n=0}^{N-1} x_n^* y_n \tag{4.29}$$

is a Hilbert space.

Polynomial Functions. An example of "functional" Hilbert space is the vector space $\mathbb{P}_N([0,1])$ of polynomial functions on the interval $[0,1]$ with maximum degree N. It is a good exercise to show that $\mathbb{P}_\infty([0,1])$ is not complete: consider for instance the sequence of polynomials

$$p_n(x) = \sum_{k=0}^{n} \frac{x^k}{k!}$$

this series converges as $p_n(x) \to e^x \notin \mathbb{P}_\infty([0,1])$.

Square Summable Functions. Another interesting example of functional Hilbert space is the space of *square integrable functions* over a finite interval. For instance, $L_2([-\pi, \pi])$ is the space of real or complex functions on the interval $[-\pi, \pi]$ which have finite norm. The inner product over $L_2([-\pi, \pi])$ is defined as:

$$\langle f, g \rangle = \int_{-\pi}^{\pi} f^*(t) g(t) dt, \tag{4.30}$$

so that the norm of $f(t)$ is:

$$\|f\| = \sqrt{\int_{-\pi}^{\pi} |f(t)|^2 dt}. \tag{4.31}$$

For $f(t)$ to belong to $L_2([-\pi, \pi])$ it must be $\|f\| < \infty$.

4.2.3 Inner Products and Distances

The inner product is a fundamental tool in a vector space since it allows us to introduce a notion of *distance* between vectors. The key intuition about this is a typical instance in which a geometric construct helps us generalize a basic idea to much more abstract scenarios. Indeed, take the simple Euclidean space $\mathbb{R}^N$ and a given vector $\mathbf{x}$; for any vector $\mathbf{y} \in \mathbb{R}^N$ the inner product $\langle \mathbf{x}, \mathbf{y} \rangle$ is the measure of the *orthogonal projection* of $\mathbf{y}$ over $\mathbf{x}$. We know that the orthogonal projection defines the point on $\mathbf{x}$ which is closest to $\mathbf{y}$ and therefore it indicates "how well" we can approximate $\mathbf{y}$ by a simple scaling of $\mathbf{x}$. To see this, just note that

$$\langle \mathbf{x}, \mathbf{y} \rangle = \|\mathbf{x}\| \|\mathbf{y}\| \cos \theta$$

where θ is the angle between the two vectors (you can work out the expression in $\mathbb{R}^2$ to easily convince you of this; the result generalizes to any other dimension). Clearly, if the vectors are orthogonal, the cosine is zero and no approximation will be possible. Since the inner product is dependent on the angular separation between the vectors, it represents a first rough measure of similarity between $\mathbf{x}$ and $\mathbf{y}$; in broad terms, it provides a measure of the difference in *shape* between vectors.

In the context of signal processing, this is particularly relevant since the difference in "shape" between signals is what we are interested in most of the time. As we stated several times before, discrete-time signals *are* vectors; the computation of their inner product will assume different names according to the processing context we find ourselves in: it will be called *filtering*, when we are trying to approximate or modify a signal; or it will be called *correlation* when we are trying to detect one particular signal amongst many. Yet, in all cases, it will still be an inner product, i.e. a *qualitative* measure of similarity between vectors. In particular, the concept of orthogonality between signal implies that the signals are perfectly distinguishable or, in other words, that their shape is completely different.

The need for a *quantitative* measure of similarity in some applications calls for the introduction of the Euclidean distance, which is derived from the inner product as:

$$d(\mathbf{x},\mathbf{y}) = \langle \mathbf{x}-\mathbf{y}, \mathbf{x}-\mathbf{y}\rangle^{1/2} = \|\mathbf{x}-\mathbf{y}\|. \tag{4.32}$$

In particular, for $\mathbb{C}^N$ the Euclidean distance is defined as:

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{n=0}^{N-1} |x_n - y_n|^2}; \tag{4.33}$$

whereas for $L_2([-\pi,\pi])$ we have:

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\int_{-\pi}^{\pi} |x(t)-y(t)|^2 dt}. \tag{4.34}$$

In the practice of signal processing, the Euclidean distance is referred to as the *root mean square error*[4]; this is a global, quantitative goodness-of-fit measure when trying to approximate signal $\mathbf{y}$ with $\mathbf{x}$.

Incidentally, there are other types of distance measures which do not rely on a notion of inner product; for example in $\mathbb{C}^N$ we could define

$$d(\mathbf{x},\mathbf{y}) = \max_{0\le n<N} |\mathbf{x}_n - \mathbf{y}_n|. \tag{4.35}$$

This distance is based on the supremum norm and is usually indicated by $\|\mathbf{x}-\mathbf{y}\|_\infty$; however, it can be shown that there is no inner product from which this norm can be derived and therefore no Hilbert space can be constructed where $\|\cdot\|_\infty$ is the natural norm. Nonetheless, this norm will reappear later, in the context of optimal filter design.

4.3 Subspaces, Bases, and Projections

Now that we have defined the properties of Hilbert space, it is only natural to start looking at the consequent inner *structure* of such a space. The best way to do so is by introducing the concept of *basis*. You can think of a basis as the "skeleton" of a vector space, a structure which holds everything together; yet, this skeleton is flexible and we can twist it, stretch it and rotate it in order to highlight some particular structure of the space and

[4]Almost always, the square distance is considered instead; its name is then the *mean square error*, or MSE

bring out the particular information we are interested in. All this is accomplished by a linear transformation called a *change of basis*; for instance, the Fourier transform is an instance of basis change.

Sometimes we will be interested in exploring more in detail a specific subset of a given vector space; this is accomplished via the concept of *subspace*. A subspace is, as the name implies, a restricted region of the global space with the additional properties that it is *closed* under the usual vector operations. This implies that, once in a subspace, we can operate freely without ever leaving its confines; just like a full-fledged space, a subspace has its own skeleton (i.e. the basis) and, again, we can exploit the properties of this basis to highlight the features we are interested in.

4.3.1 Definitions

Assume $H(V, S)$ is a Hilbert space, with V a vector space and S a set of scalars (i.e. $\mathbb{C}$).

Subspace. A subspace of V is defined as a subset $P \subseteq V$ that satisfies the following properties:

- Closure under addition, i.e.

$$\forall\, \mathbf{x}, \mathbf{y} \in P \Rightarrow \mathbf{x} + \mathbf{y} \in P \tag{4.36}$$

- Closure under scalar multiplication, i.e.

$$\forall\, \mathbf{x} \in P, \forall\, \alpha \in S \Rightarrow \alpha\mathbf{x} \in P. \tag{4.37}$$

Clearly, V is a subspace of itself.

Span. Given an arbitrary set of M vectors $W = \{\mathbf{x}^{(m)}\}_{m=0,1,\ldots,M-1}$, the *span* of these vector is defined as:

$$\text{span}(W) = \left\{ \sum_{m=0}^{M-1} \alpha_m \mathbf{x}^{(m)} \right\}, \quad \alpha_m \in S \tag{4.38}$$

i.e. the span of W is the set of all possible linear combinations of the vectors in W. The set of vectors W is called *linearly independent* if the following holds:

$$\sum_{m=0}^{M-1} \alpha_m \mathbf{x}^{(m)} = 0 \quad \Longleftrightarrow \quad \alpha_m = 0 \text{ for } m = 0, 1, \ldots, M-1. \tag{4.39}$$

Basis. A set of K vectors $W = \{\mathbf{x}^{(k)}\}_{k=0,1,\ldots,K-1}$ from a subspace P is a *basis* for that subspace if:

- The set W is linearly independent.
- Its span covers P, i.e. $\text{span}(W) = P$.

The last statement affirms that any $\mathbf{y} \in P$ can be written as a linear combination of $\{\mathbf{x}^{(k)}\}_{k=0,1,\ldots,K-1}$ or that, for all $\mathbf{y} \in P$, there exist K coefficients α_k such that

$$\mathbf{y} = \sum_{k=0}^{K-1} \alpha_k \mathbf{x}^{(k)}; \tag{4.40}$$

this is equivalently expressed by saying that the set W is *complete* in P.

Orthogonal/Orthonormal Basis. An orthonormal basis for a subspace P is a set of K basis vectors $W = \{\mathbf{x}^{(k)}\}_{k=0,1,\ldots,K-1}$ for which:

$$\langle \mathbf{x}^{(i)}, \mathbf{x}^{(j)} \rangle = \delta[i-j] \quad 0 \leq i, j < K \tag{4.41}$$

which means orthogonality across vectors and unit norm. Sometimes, we can have that the set of vectors is orthogonal but not normal (i.e. the norm of the vectors is not unitary). This is hardly a problem provided that we remember to include the appropriate normalization factors in the analysis and/or synthesis formulas. Alternatively, an orthogonal set of vectors can be normalized via the Gram-Schmidt procedure, which you can find in any linear algebra textbook.

Among all bases, *orthonormal bases* are the most "beautiful" in some sense because of their structure and their properties. One of the most important properties for finite-dimensional spaces is the following:

- A set of N orthogonal vectors in an N-dimensional subspace is a basis for the subspace.

In other words, in finite dimensions, once we find a full set of orthogonal vectors we are sure that the set spans the space.

4.3.2 Properties of Orthonormal Bases

Let $W = \{\mathbf{x}^{(k)}\}_{k=0,1,\ldots,K-1}$ be an orthonormal basis for a (sub)space P. Then the following properties hold (all of which are easily verified):

Analysis Formula. The coefficients in the linear combination (4.40) are obtained simply as:

$$\alpha_k = \langle \mathbf{x}^{(k)}, \mathbf{y} \rangle \tag{4.42}$$

The coefficients $\{\alpha_k\}$ are called the *Fourier coefficients*[5] of the orthonormal expansion of $\mathbf{y}$ with respect to the basis W and (4.42) is called the Fourier *analysis formula*; conversely, Equation (4.40) is called the *synthesis formula*.

Parseval's Identity For an orthonormal basis, there is a norm conservation property given by *Parseval's identity*:

$$\|\mathbf{y}\|^2 = \sum_{k=0}^{K-1} |\langle \mathbf{x}^{(k)}, \mathbf{y}\rangle|^2. \tag{4.43}$$

For physical quantities, the norm is dimensionally equivalent to a measure of energy; accordingly, Parseval's identity is also known as the *energy conservation formula*.

Basis Completion Let $G = \left\{\mathbf{z}^{(\ell)}\right\}_{\ell=0}^{L-1}$ be a set of orthonormal vectors in a subspace P of dimension $K > L$. Clearly, G is not a basis for P. If $\left\{\mathbf{x}^{(k)}\right\}_{k=0}^{K-1}$ form an orthonormal basis, then we can find vectors $\left\{\tilde{\mathbf{z}}^{(\ell)}\right\}_{\ell=L}^{K-1}$ such that $\left\{\left\{\mathbf{z}^{(\ell)}\right\}_{\ell=0}^{L-1}, \left\{\tilde{\mathbf{z}}^{(\ell)}\right\}_{\ell=L}^{K-1}\right\}$ form an orthonormal basis for P. This can be done in the following manner, since $\left\{\mathbf{x}^{(k)}\right\}_{k=0}^{K-1}$ is an orthonormal basis for P, we want to find $\tilde{\mathbf{z}}^{(L)} \in P$ s.t.

$$\tilde{\mathbf{z}}^{(L)} = \sum_{k=0}^{K-1} \alpha_k \mathbf{x}^{(k)} \quad , \quad \langle \tilde{\mathbf{z}}^{(L)}, \mathbf{z}^{(\ell)}\rangle = 0, \quad \ell = 0, \cdots, L-1$$

$$\Longrightarrow \sum_{k=0}^{K-1} \alpha_k \langle \mathbf{x}^{(k)}, \mathbf{z}^{(\ell)}\rangle = 0, \quad \ell = 0, \cdots, L-1$$

$$\Longrightarrow \begin{bmatrix} \langle \mathbf{x}^{(0)}, \mathbf{z}^{(0)}\rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \mathbf{z}^{(0)}\rangle \\ \vdots & & \vdots \\ \langle \mathbf{x}^{(0)}, \mathbf{z}^{(L-1)}\rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \mathbf{z}^{(L-1)}\rangle \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{K-1} \end{bmatrix} = \mathbf{0}.$$

[5]Fourier coefficients often refer to the particular case of Fourier series. However, the term generally refers to coefficients in any orthonormal basis.

Since $\begin{bmatrix} \langle \mathbf{x}^{(0)}, \mathbf{z}^{(0)} \rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \mathbf{z}^{(0)} \rangle \\ \vdots & & \vdots \\ \langle \mathbf{x}^{(0)}, \mathbf{z}^{(L-1)} \rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \mathbf{z}^{(L-1)} \rangle \end{bmatrix} \in \mathbb{C}^{L\times K}$,

we can find a vector in its null space (which is of dimension $K-L$) to obtain $\tilde{\mathbf{z}}^{(L)}$ (properly normalized). In a similar manner we can find $\tilde{\mathbf{z}}^{(\ell)}, \quad \ell = L, \cdots, K-1$ by

$$\langle \tilde{\mathbf{z}}^{(\ell)}, \mathbf{z}^{(p)} \rangle = 0, \quad \text{for } p = 0, \cdots, L-1$$

$$\langle \tilde{\mathbf{z}}^{(\ell)}, \tilde{\mathbf{z}}^{(q)} \rangle = 0, \quad \text{for } q = L, \cdots, \ell-1.$$

This amounts to finding a vector in the null space of

$$\begin{bmatrix} \langle \mathbf{x}^{(0)}, \mathbf{z}^{(0)} \rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \mathbf{z}^{(0)} \rangle \\ \vdots & & \vdots \\ \langle \mathbf{x}^{(0)}, \mathbf{z}^{(L-1)} \rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \mathbf{z}^{(L-1)} \rangle \\ \langle \mathbf{x}^{(0)}, \tilde{\mathbf{z}}^{(L)} \rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \tilde{\mathbf{z}}^{(L)} \rangle \\ \vdots & & \vdots \\ \langle \mathbf{x}^{(0)}, \tilde{\mathbf{z}}^{(\ell-1)} \rangle & \cdots & \langle \mathbf{x}^{(K-1)}, \tilde{\mathbf{z}}^{(\ell-1)} \rangle \end{bmatrix}$$

Therefore by *defining* $\mathbf{z}^{(\ell)} = \tilde{\mathbf{z}}^{(\ell)}, \quad \ell = L, \cdots, K-1$, we get an orthonormal basis $\{\mathbf{z}^{(\ell)}\}_{\ell=0}^{K-1}$ for P, which extends the orthonormal set $\{\mathbf{z}^{(\ell)}\}_{\ell=0}^{L-1}$ to a basis spanning P. Therefore for any $\mathbf{y} \in P$,

$$\|\mathbf{y}\|^2 = \sum_{k=0}^{K-1} |\langle \mathbf{z}^{(k)}, \mathbf{y} \rangle|^2 \geq \sum_{\ell=0}^{L-1} |\langle \mathbf{z}^{(\ell)}, \mathbf{y} \rangle|^2.$$

Best Approximations. Assume P is a subspace of V; if we try to approximate a vector $\mathbf{y} \in V$ by a linear combination of basis vectors from the subspace P, then we are led to the concepts of least squares approximations and orthogonal projections. First of all, we define the *best* linear approximation $\hat{\mathbf{y}} \in P$ of a general vector $\mathbf{y} \in V$ to be the approximation which minimizes the norm $\|\mathbf{y}-\hat{\mathbf{y}}\|$. Such approximation is easily obtained by projecting $\mathbf{y}$ onto an orthonormal basis for P, as shown in Figure 4.3. With W our usual orthonormal basis for P, the projection is given by:

$$\hat{\mathbf{y}} = \sum_{k=0}^{K-1} \langle \mathbf{x}^{(k)}, \mathbf{y} \rangle \mathbf{x}^{(k)} \tag{4.44}$$

Define the approximation error as the vector $\mathbf{d} = \mathbf{y} - \hat{\mathbf{y}}$. The best approximation $\hat{\mathbf{y}} \in P$ is such that the error $\mathbf{d} = \mathbf{y} - \hat{\mathbf{y}}$ is orthogonal to all vectors $\mathbf{z} \in P$, i.e., $\mathbf{y} - \hat{\mathbf{y}} \perp \mathbf{z} \quad \forall \mathbf{z} \in P$.

Proof: Let $\hat{\mathbf{y}}$ be the unique vector such that $\mathbf{d} = \mathbf{y} - \hat{\mathbf{y}}$ is orthogonal to all vectors $\mathbf{z} \in P$. Consider any vector $\mathbf{g} \in P$ which is a candidate for the best approximation to $\mathbf{y}$ in P.

$$\|\mathbf{y} - \mathbf{g}\|^2 = \|(\mathbf{y} - \hat{\mathbf{y}}) + (\hat{\mathbf{y}} - \mathbf{g})\|^2 = \|\mathbf{y} - \hat{\mathbf{y}}\|^2 + \|\hat{\mathbf{y}} - \mathbf{g}\|^2 + \langle \mathbf{y} - \hat{\mathbf{y}}, \hat{\mathbf{y}} - \mathbf{g} \rangle + \langle \hat{\mathbf{y}} - \mathbf{g}, \mathbf{y} - \hat{\mathbf{y}} \rangle.$$

However, since $\hat{\mathbf{y}}, \mathbf{g} \in P, \quad \hat{\mathbf{y}} - \mathbf{g} \in P$ and hence

$$\langle \hat{\mathbf{y}} - \mathbf{g}, \mathbf{y} - \hat{\mathbf{y}} \rangle = 0 = \langle \mathbf{y} - \hat{\mathbf{y}}, \hat{\mathbf{y}} - \mathbf{g} \rangle.$$

Hence we have

$$\|\mathbf{y} - \mathbf{g}\|^2 = \|\mathbf{y} - \hat{\mathbf{y}}\|^2 + \|\hat{\mathbf{y}} - \mathbf{g}\|^2 \geq \|\mathbf{y} - \hat{\mathbf{y}}\|^2$$

with equality iff $\mathbf{g} = \hat{\mathbf{y}}$. ■

It can be easily shown that the approximation minimizes the error square norm, i.e.

$$\arg\min_{\hat{\mathbf{y}} \in P} \|\mathbf{y} - \hat{\mathbf{y}}\|^2 = \mathbf{d}. \tag{4.45}$$

This approximation with an orthonormal basis has a key property: it can be *successively refined*. Assume you have the approximation with the first m terms of the orthonormal basis:

$$\hat{\mathbf{y}}_m = \sum_{k=0}^{m-1} \langle \mathbf{x}^{(k)}, \mathbf{y} \rangle \mathbf{x}^{(k)} \tag{4.46}$$

and now you want to compute the $(m+1)$-term approximation. This is simply given by

$$\hat{\mathbf{y}}_{m+1} = \sum_{k=0}^{m} \langle \mathbf{x}^{(k)}, \mathbf{y} \rangle \mathbf{x}^{(k)} = \hat{\mathbf{y}}_m + \langle \mathbf{x}^{(m)}, \mathbf{y} \rangle \mathbf{x}^{(m)} \tag{4.47}$$

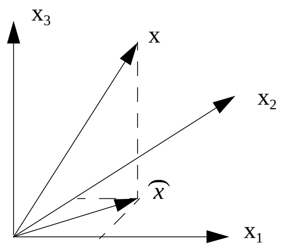


Figure 4.3: Orthogonal projection of the vector $\mathbf{x}$ onto the subspace W spanned by $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$, leading to the approximation $\hat{\mathbf{x}}$. This approximation minimizes the square norm $\|\mathbf{x}-\hat{\mathbf{x}}\|_2$ among all approximations belonging to W.

While this seems obvious, it is actually a small miracle, since it does not hold for more general, non-orthonormal bases.

Bessel's Inequality. The generalization of Parseval's equality is *Bessel's inequality*. Suppose $M = \left\{\mathbf{e}^{(k)}\right\}$ is an orthonormal set, but not a basis for the entire space V. Define a subspace:

$$P = \mathbf{Span}(M) = \mathbf{Span}\left\{\left\{\mathbf{e}^{(k)}\right\}\right\}.$$

Let us consider the projection of an arbitrary vector $\mathbf{x} \in V$, onto P. We have seen that this is given by

$$\hat{\mathbf{x}} = \sum_k \langle \mathbf{e}^{(k)}, \mathbf{x}\rangle \mathbf{e}^{(k)}$$

and that $(\mathbf{x} - \hat{\mathbf{x}}) \perp \mathbf{g}$, $\forall \mathbf{g} \in P$. In particular, since $\hat{\mathbf{x}} \in P$, we see that

$$\mathbf{z} \triangleq \mathbf{x} - \hat{\mathbf{x}} \in P \qquad , \qquad \mathbf{z} \perp \hat{\mathbf{x}} \qquad \text{and}$$

$$\mathbf{x} = \underbrace{\mathbf{x} - \hat{\mathbf{x}}}_{\mathbf{z}} + \hat{\mathbf{x}} = \mathbf{z} + \hat{\mathbf{x}}.$$

Hence by Pythagoras theorem:

$$\|\mathbf{x}\|^2 = \|\mathbf{z}\|^2 + \|\hat{\mathbf{x}}\|^2$$

or: $\quad 0 \leq \|\mathbf{z}\|^2 = \|\mathbf{x}\|^2 - \|\hat{\mathbf{x}}\|^2$

or: $\quad \|\hat{\mathbf{x}}\|^2 \overset{(a)}{=} \sum_k |\langle \mathbf{e}^{(k)}, \mathbf{x}\rangle|^2 \leq \|\mathbf{x}\|^2$

where (a) follows from Parseval's relationship.

Hence for any orthonormal set $\{\mathbf{e}^{(k)}\}$,

$$\sum_k |\langle \mathbf{e}^{(k)}, \mathbf{x}\rangle|^2 \leq \|\mathbf{x}\|^2 \qquad \longrightarrow \qquad \text{Bessel's inequality.} \tag{4.48}$$

4.3.3 Examples of Bases

Considering the examples of 4.2.2, we have the following:

Finite Euclidean Spaces. For the simplest case of Hilbert spaces, namely $\mathbb{C}^N$, orthonormal bases are also the most intuitive since they contain exactly N mutually orthogonal vectors of unit norm. The classical example is the canonical basis $\{\boldsymbol{\delta}^{(k)}\}_{k=0\ldots N-1}$ with

$$\boldsymbol{\delta}_n^{(k)} = \delta[n-k] \tag{4.49}$$

but we will soon study more interesting bases such as the Fourier basis $\{\mathbf{w}^{(k)}\}$, for which

$$\mathbf{w}_n^{(k)} = e^{j\frac{2\pi}{N}nk}.$$

In $\mathbb{C}^N$, the analysis and synthesis formulas (4.42) and (4.40) take a particularly neat form. For any set $\{\mathbf{x}^{(k)}\}$ of N orthonormal vectors one can indeed arrange the conjugates of the basis vectors[6] as the successive rows of an $N \times N$ square matrix $\mathbf{M}$ so that each matrix element is

$$\mathbf{M}_{mn} = (\mathbf{x}_n^{(m)})^*;$$

$\mathbf{M}$ is called a *change of basis* matrix. Given a vector $\mathbf{y}$, the set of expansion coefficient $\{\alpha_k\}_{k=0\ldots N-1}$ can now be written *itself* as a vector[7] $\boldsymbol{\alpha} \in \mathbb{C}^N$. Therefore, we can rewrite the analysis formula (4.42) in matrix-vector form and we have:

$$\boldsymbol{\alpha} = \mathbf{M}\mathbf{y}. \tag{4.50}$$

[6]Other definitions may build $\mathbf{M}$ by stacking the *non*-conjugated basis vectors instead; the procedure is however entirely equivalent. Here we choose this definition in order to be consistent with the usual derivation of the Discrete Fourier Transform, which we will see in the next chapter.

[7]This isomorphism is rather special and at the foundation of Linear Algebra. If the original vector space V is not $\mathbb{C}^N$, the analysis formula will always provide us with a vector of complex values, but this vector will *not* be in V.

The reconstruction formula (4.40) for $\mathbf{y}$ from the expansion coefficients becomes in turn:

$$\mathbf{y} = \mathbf{M}^H \boldsymbol{\alpha} \tag{4.51}$$

where the superscript denotes the Hermitian transpose (transposition and conjugation of the matrix). The previous equation shows that $\mathbf{y}$ is a linear combination of the columns of $\mathbf{M}^H$, which, in turn, are of course the vectors $\{\mathbf{x}^{(k)}\}$. The orthogonality relation (4.49) takes the following forms

$$\mathbf{M}^H\mathbf{M} = \mathbf{I} \tag{4.52}$$

$$\mathbf{M}\mathbf{M}^H = \mathbf{I} \tag{4.53}$$

since left inverse equals right inverse for square matrices; this implies that $\mathbf{M}$ has orthonormal rows as well as orthonormal columns.

Polynomial Functions. A basis for $\mathbb{P}_N([0,1])$ is $\{x^k\}_{0\leq k<N}$. This basis, however, is not an orthonormal basis. It can be transformed to an orthonormal basis by a standard Gram-Schmidt procedure; the basis vectors thus obtained are called *Legendre polynomials*.

Square Summable Functions. An orthonormal basis set for $L_2([-\pi,\pi])$ is the set $\{(1/\sqrt{2\pi})e^{jnt}\}_{n\in\mathbb{Z}}$. This is actually the classic Fourier basis for functions on an interval. Please note that here, as opposed to the previous examples, the number of basis vectors is actually infinite. The orthogonality of these basis vectors is easily verified; their completeness, however, is extremely hard to prove and this, unfortunately, is pretty much the rule for all non-trivial infinite-dimensional basis sets.

4.4 Signal Spaces Revisited

We are now in the position to formalize our intuitions so far with respect to the equivalence between discrete-time signals and vector spaces, with a particularization for the three main classes of signals we have introduced in the previous chapter. Note that in the following, we will liberally interchange the notations $\mathbf{x}$ and $x[n]$ to denote a sequence as a vector embedded in its appropriate Hilbert space.

4.4.1 Finite-Length Signals

The correspondence between the class of finite-length, length-N signals and $\mathbb{C}^N$ should be so immediate at this point that it does not need further comment. As a reminder, the canonical basis is the canonical basis for $\mathbb{C}^N$. The k-th canonical basis vector will often be expressed in signal form as:

$$\delta[n-k] \quad n = 0,\ldots,N-1,\ k = 0,\ldots,N-1$$

4.4.2 Periodic Signals

As we have seen, N-periodic signals are equivalent to length-N signals. The space of N-periodic sequences is therefore isomorphic to $\mathbb{C}^N$. In particular, the sum between two sequences considered as vectors is the standard pointwise sum for the elements:

$$z[n] = x[n] + y[n] \quad n \in \mathbb{Z} \tag{4.54}$$

while, for the inner product, we extend the summation over a period only:

$$\langle x[n], y[n] \rangle = \sum_{n=0}^{N-1} x^*[n]y[n]. \tag{4.55}$$

The canonical basis for the space of N-periodic sequences is the canonical basis for $\mathbb{C}^N$, because of the isomorphism; in general, any basis for $\mathbb{C}^N$ is also a basis for the space of N-periodic sequences. Sometimes, however, we will also consider an explicitly *periodized* version of the basis. For the canonical basis, in particular, the periodized basis is composed of N vectors of infinite length $\{\tilde{\boldsymbol{\delta}}^{(k)}\}_{k=0\ldots N-1}$ with:

$$\tilde{\boldsymbol{\delta}}^{(k)} = \sum_{i=-\infty}^{\infty} \delta[n - k - iN]$$

Such a sequence is called a *pulse train*. Note that here we are abandoning mathematical rigor, since the norm of each of these basis vectors is infinite; yet the pulse train, if handled with care, can be a useful tool in formal derivations.

4.4.3 Inifinite Sequences

Finally, this is where we have been heading to all along. For infinite sequences, whose "natural" Euclidean space would appear to be $\mathbb{C}^\infty$, the situation is rather delicate. While the sum of two sequences can be defined in the usual way, by extending the sum for $\mathbb{C}^N$ to $\mathbb{C}^\infty$, care must be taken when evaluating the inner product. We already pointed out that

$$\langle x[n], y[n] \rangle = \sum_{n=-\infty}^{\infty} x^*[n]y[n] \tag{4.56}$$

can diverge even for simple constant sequences such as $x[n] = y[n] = 1$. A way out of this impasse is to restrict ourselves to $l_2(\mathbb{Z})$, the space of *square summable sequences*, for which

$$\|x\|^2 = \sum_{n\in\mathbb{Z}} |x[n]|^2 < \infty \tag{4.57}$$

This is the space of choice for all the theoretical derivations involving infinite sequences. Note that these sequences are often called “of finite energy”, since the square norm corresponds to the definition of energy as given in (2.17).

The canonical basis for $l_2(\mathbb{Z})$ is simply the set $\{\boldsymbol{\delta}^{(k)}\}_{k\in\mathbb{Z}}$; in signal form:

$$\boldsymbol{\delta}^{(k)} = \delta[n-k], \quad n,k \in \mathbb{Z} \tag{4.58}$$

This is an infinite set, and actually an infinite set of linearly independent vectors, since

$$\delta[n-k] = \sum_{l\in\mathbb{Z}/k} \alpha_l \delta[n-l] \tag{4.59}$$

has no solution for any k. Note that, for an arbitrary signal $x[n]$ the analysis formula gives

$$\alpha_k = \langle \boldsymbol{\delta}^{(k)}, \mathbf{x}\rangle = \langle \delta[n-k], x[n]\rangle = x[k]$$

so that the reconstruction formula becomes

$$x[n] = \sum_{k=-\infty}^{\infty} \alpha_k \boldsymbol{\delta}^{(k)} = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k]$$

which is the reproducing formula (2.16). The Fourier basis for $l_2(\mathbb{Z})$ will be introduced and discussed at length in the next chapter.

As a last remark, note that the space of *all* finite-support signals, which is clearly a subset of $l_2(\mathbb{Z})$, does *not* form a Hilbert space. Clearly, the space is closed under addition and scalar multiplication, and the canonical inner product is well behaved since all sequences have only a finite number of nonzero values. The space however is not complete; to see this, consider the following family of signals

$$y_k[n] = \begin{cases} 1/n & |n| < k \\ 0 & \text{otherwise} \end{cases}$$

For k growing to infinity the sequence of signals converges as $y_k[n] \rightarrow y[n] = 1/n$ for all n; while $y[n]$ is indeed in $l_2(\mathbb{Z})$, since

$$\sum_{n=0}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6},$$

$y[n]$ is clearly *not* a finite-support signal.

4.5 Summary

The purpose of this chapter was to lay a solid geometrical and algebraic foundation to the theory of discrete-time signal processing. This was achieved by establishing a correspondence between signals and vectors in a Hilbert space. The main points we have covered are:

- A review of Euclidean geometry in the context of simple vector spaces such as $\mathbb{R}^N$.
- The step-by-step definition of Hilbert space as a complete, inner product vector space.
- The properties of inner product and their extension to the concept of norm and distance.
- The concepts of basis and orthonormal basis, with a special emphasis on the latter.
- Examples of Hilbert space: $\mathbb{C}^N$, $P_N([0,1])$, $L_2([-\pi,\pi])$ with their respective inner products and bases.
- A correspondence between the three main classes of discrete-time signals and three suitable types of Hilbert space. In particular, $\mathbb{C}^N$ for finite-length and periodic sequences and $l_2(\mathbb{Z})$ for infinite sequences.

4.6 Problems

Problem 4.1 *Consider the Fourier basis* $\{\boldsymbol{w}^{(k)}\}_{k=0,\ldots,N-1}$*, defined as:*

$$\boldsymbol{w}_n^{(k)} = e^{-j\frac{2\pi}{N}nk}.$$

1. *Prove that it is an orthogonal basis in* $\mathbb{C}^N$*.*
2. *Normalize the vectors in order to get an orthonormal basis.*

Chapter 5

The DTFT (Discrete-Time Fourier Transform)

We will now consider a Fourier representation for infinite non-periodic sequences. Let us start out abruptly: the Discrete-Time Fourier Transform of a sequence $x[n]$ is defined as:

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-jwn}. \tag{5.1}$$

The DTFT is therefore a complex-valued function of the *real* argument ω, and, as can be easily verified, it is periodic in ω with period 2π. The somewhat odd notation $X(e^{j\omega})$ is quite standard in the signal processing literature and offers several advantages:

- it stresses the basic periodic nature of the transform since, obviously, $e^{j(\omega+2\pi)} = e^{j\omega}$;
- regardless of context, it immediately identifies a function as the Fourier transform of a discrete-time sequence: something like $U(e^{j\lambda})$ is just as readily recognizable;
- it provides a nice notational framework which unifies the Fourier transform and the z-transform (which we will see later)

The DTFT, when it exists, can be inverted via the integral

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega \tag{5.2}$$

as can be easily verified by substituting (5.1) into 5.2) and using

$$\int_{-\pi}^{\pi} e^{-j\omega(n-k)} = 2\pi\delta[n-k].$$

In fact, due to the 2π periodicity of the DTFT, the integral in (5.2) can be computed over *any* 2π-wide interval on the real line (i.e. between 0 and 2π, for instance). The relation between a sequence $x[n]$ and its DTFT $X(e^{j\omega})$ will be indicated in the general case by:

$$x[n] \overset{\text{DTFT}}{\longleftrightarrow} X(e^{j\omega})$$

While the DFT and DFS were signal transformation which involved only a finite number of quantities, both the infinite summation and the real-valued argument appearing in the DTFT can create an uneasiness which overshadows the conceptual similarities between the transforms. In the following, we will start by defining the mathematical properties of the DTFT and we will try to build an intuitive feeling for this Fourier representation both with respect to its physical interpretation and to its conformity to the "change of basis" framework we used for the DFT and DFS.

Mathematically, the DTFT is a transform operator which maps discrete-time sequences onto the space of 2π-periodic functions. Clearly, for the DTFT to exist, the sum in (5.1) must converge, i.e. the limit for $M \to \infty$ of the partial sum

$$X_M(e^{j\omega}) = \sum_{n=-M}^{M} x[n]e^{-jwn} \tag{5.3}$$

must exist and be finite. Convergence of the partial sum in (5.3) is very easy to prove for *absolutely summable* sequences, that is for sequences satisfying

$$\lim_{M\to\infty} = \sum_{n=-M}^{M} |x[n]| < \infty \tag{5.4}$$

since

$$|X_M(e^{j\omega})| \leq \sum_{n=-M}^{M} |x[n]e^{-jwn}| = \sum_{n=-M}^{M} |x[n]| \tag{5.5}$$

For this class of sequences it can be also proved that the convergence of $X_M(e^{j\omega})$ to $X(e^{j\omega})$ is uniform and that $X(e^{j\omega})$ is continuous. While absolute summability is a sufficient condition, it can be shown that the sum in (5.3) is convergent also for all *square-summable* sequences, i.e. for sequences whose energy is finite; this is very important to us with respect to the discussion in Section 4.4.3 where we defined the Hilbert space $l_2(\mathbb{Z})$. In the case of square summability only, however, the convergence of (5.3) is no longer uniform but takes place only in the mean-square sense, i.e.

$$\lim_{M\to\infty} \int_{-\pi}^{\pi} |X_M(e^{j\omega}) - X(e^{j\omega})|^2 d\omega = 0. \tag{5.6}$$

Convergence in the mean square sense implies that, while the total energy of the error signal becomes zero, the pointwise values of the partial sum may never approach the values of the limit. One manifestation of this odd behavior is called the *Gibbs phenomenon*, which will have important consequences in our approach to filter design, as we will see later. Furthermore, in the case of square-summable sequences, $X(e^{j\omega})$ is no longer guaranteed to be continuous.

5.1 The DTFT as the Limit of a DFS

A way to gain some intuition about the structure of the DTFT formulas is to consider the DFS of periodic sequences with longer and longer period. Intuitively, as we look at the structure of the Fourier basis for the DFS, we can see that the number of basis vectors in (3.19) grows with the length N of the period and, consequently, the frequencies of the underlying complex exponentials become "denser" between 0 and 2π. We want to show that, in the limit, we end up with the reconstruction formula of the DTFT.

To do so, let us restrict ourselves to the domain of absolute summable sequences; for these sequences we know that the sum in (5.1) exists. Now, given an absolutely summable sequence $x[n]$, we can always build an N-periodic sequence $\tilde{x}[n]$ as

$$\tilde{x}[n] = \sum_{i=-\infty}^{\infty} x[n + iN] \tag{5.7}$$

for any value of N; this is guaranteed by the fact that the above sum converges for all $n \in \mathbb{Z}$ (because of the absolute summability of $x[n]$) so that all values of $\tilde{x}[n]$ are finite. Clearly, there is overlap between successive copies of $x[n]$; the intuition, however, is the following: since in the end we will consider very large values for N and since $x[n]$, because of absolute summability, decays rather fast with n, the resulting overlap of "tails" will be negligible. In other words, we have:

$$\lim_{N\to\infty} \tilde{x}[n] = x[n].$$

Consider now the DFS of $\tilde{x}[n]$:

$$\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{N}nk} = \sum_{i=-\infty}^{\infty} \Big(\sum_{n=0}^{N-1} x[n + iN] e^{-j\frac{2\pi}{N}(n+iN)k}\Big) \tag{5.8}$$

where in the last term we have used (5.7), interchanged the order of the summation and exploited the fact that $e^{-j(2\pi/N)(n+iN)k} = e^{-j(2\pi/N)nk}$. We can see that, for every value of

i in the outer sum, the argument of the inner sum varies between iN and $iN+N-1$, i.e. non overlapping intervals, so that the double summation can be simplified as

$$\tilde{X}[k] = \sum_{m=-\infty}^{\infty} x[m] e^{-j\frac{2\pi}{N}mk} \tag{5.9}$$

and therefore

$$\tilde{X}[k] = X(e^{j\omega})\big|_{\omega=\frac{2\pi}{N}k}. \tag{5.10}$$

This already gives us a noteworthy piece of intuition: the DFS coefficients for the periodized signal are a discrete set of values of its DTFT (here considered solely as a formal operator) computed at multiples of $2\pi/N$. As N grows, the spacing between these frequency intervals narrows more and more so that, in the limit, the DFS converges to the DTFT.

To see that this assertion is consistent, we can now write the DFS reconstruction formula using the DFS values given to us by (5.10) in (3.20):

$$\tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} X(e^{j\frac{2\pi}{N}k}) e^{j\frac{2\pi}{N}nk}. \tag{5.11}$$

By defining $\Delta = (2\pi/N)$, we can rewrite the above expression as

$$\tilde{x}[n] = \frac{1}{2\pi}\sum_{k=0}^{N-1} X(e^{j(k\Delta)}) e^{j(k\Delta)n}\Delta \tag{5.12}$$

and the summation is easily recognized as the Riemann sum with step Δ approximating the integral of $f(\omega) = X(e^{j\omega})e^{j\omega n}$ between 0 and 2π. As N goes to infinity (and therefore $\tilde{x}[n] \to x[n]$), we can therefore write

$$\tilde{x}[n] \to \frac{1}{2\pi}\int_0^{2\pi} X(e^{j\omega}) e^{j\omega n} d\omega \tag{5.13}$$

which is indeed the DTFT reconstruction formula (5.2)[1].

Example 5.1 *Consider the signal $x[n]$ shown in Fig. 5.1. We can build a periodic signal with period $N = 3$ based on $x[n]$ which is shown in Fig. 5.2.*

[1]Clearly (5.13) is equivalent to (5.2) in spite of the different integration limits since all the quantities under the integral sign are 2π-periodic and we are integrating over a period.

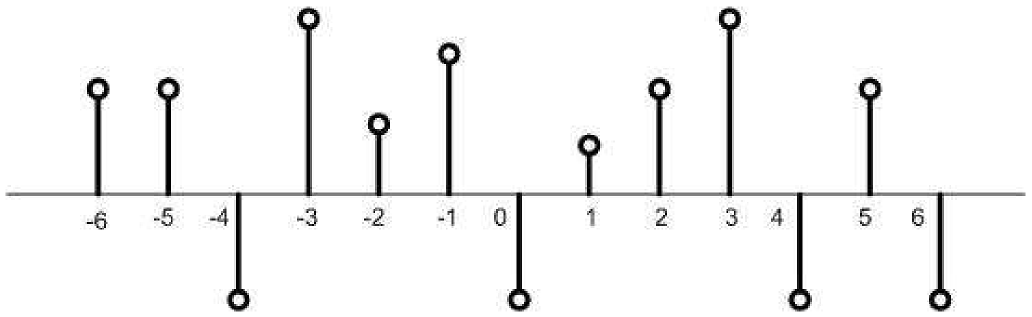


Figure 5.1: $x[n]$

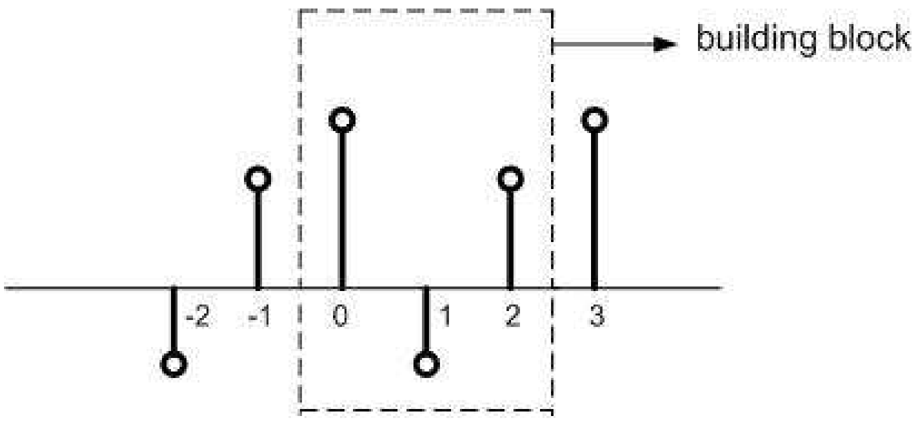


Figure 5.2: $\tilde{x}[n], N = 3$

5.2 The DTFT as a Formal Change of Basis

We will now show that, if we are willing to sacrifice mathematical rigor, the DTFT can be cast in the same conceptual framework we used for the DFT and DFS, namely as a basis change in a vector space. The following formulas are to be taken as nothing more than a set of purely symbolic derivations, since the mathematical hypotheses under which the results are well defined are far from obvious and are completely hidden by the formalism. It is only fair to say, however, that the following expressions represent a very handy and intuitive toolbox to grasp the essence of the duality between the discrete-time and the frequency domains and that they can be put to use very effectively to derive quick results when manipulating sequences.

One way of interpreting equation (5.1) is to see that, for any given value ω_0, the corresponding value of the DTFT is the inner product in $l_2(\mathbb{Z})$ of the sequence $x[n]$ with the sequence $e^{j\omega_0 n}$; at least formally, we are still performing a projection in a vector space akin to $\mathbb{C}^\infty$:

$$X(e^{j\omega}) = \langle e^{j\omega n}, x[n] \rangle$$

here, however, the set of "basis vectors" $\{e^{j\omega n}\}_{\omega\in\mathbb{R}}$ is indexed by the real variable ω and is therefore uncountable. This uncountability is mirrored in the inversion formula (5.2), in which the usual summation is replaced by an integral; in fact, the DTFT operator maps $l_2(\mathbb{Z})$ onto $L_2([-\pi,\pi])$ which is a space of 2π-periodic, square integrable functions. This interpretation preserves the physical meaning given to the inner products in (5.1) as a way to measure the frequency content of the signal at a given frequency; in this case the number of oscillators is infinite and their frequency separation becomes infinitesimally small.

To complete the picture of the DTFT as a change of basis, we want to show that, at least formally, the set $\{e^{j\omega n}\}_{\omega\in\mathbb{R}}$ constitutes an orthogonal "basis" for $l_2(\mathbb{Z})$[2]. In order to do so we need to introduce a quirky mathematical entity called the Dirac delta functional; this is defined in an implicit way by the following formula:

$$\int_{-\infty}^{\infty} \delta(t-\tau) f(t)dt = f(\tau) \tag{5.14}$$

where $f(t)$ is an arbitrary integrable function on the real line; in particular:

$$\int_{-\infty}^{\infty} \delta(t) f(t)dt = f(0). \tag{5.15}$$

While no ordinary function satisfies the above equation, $\delta(t)$ can be interpreted as a shorthand for a limiting operation. Consider for instance the family of parametric functions[3]

$$r_k(t) = k\,\mathrm{rect}(kt) \tag{5.16}$$

which are plotted in Figure 5.3. For any continuous function $f(t)$ we can write

$$\int_{-\infty}^{\infty} r_k(t) f(t)dt = k\int_{-1/2k}^{1/2k} f(t)dt = f(\gamma)|_{\gamma\in[-1/2k,1/2k]} \tag{5.17}$$

[2]You can see here already why this line of thought is shaky: indeed, $e^{j\omega n} \notin l_2(\mathbb{Z})$!

[3]The rect function is introduced in its full glory in Section 7.7.1; it is defined as

$$\mathrm{rect}(x) = \begin{cases} 1 & \text{for } |x| \leq 1/2 \\ 0 & \text{otherwise} \end{cases}$$

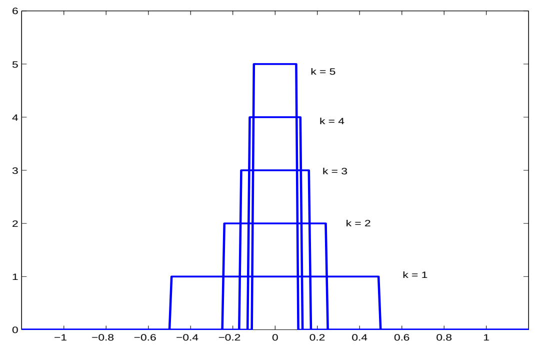


Figure 5.3: The Dirac delta as the limit of a family of rect functions.

where we have used the mean value theorem. Now, as k goes to infinity we have that the integral converges to $f(0)$ and so we can say that the limit of the series of functions $r_k(t)$ converges then to the Dirac delta. The delta, as we said, cannot be considered as a proper function so the expression $\delta(t)$ outside of an integral sign has no mathematical meaning; it is customary however to associate an "idea" of function to the delta and we can think of it as being undefined for $t \neq 0$ and to have a value of ∞ at $t = 0$. This interpretation, together with (5.14), defines the so-called *sifting property* of the Dirac delta; this property allows us to write (outside of the integral sign):

$$\delta(t-\tau)f(t) = f(\tau)\delta(t-\tau) \tag{5.18}$$

The physical interpretation of the Dirac delta is related to quantities expressed as continuous *distributions* for which the most familiar example is probably that of a probability distribution (pdf). These functions represents a value which makes physical sense only over an interval of nonzero measure; the punctual value of a distribution is only an abstraction. The Dirac delta is the operator that extracts this punctual value from a distribution, in a sense capturing the essence of considering smaller and smaller observation intervals.

To see how the Dirac delta applies to our basis expansion, note that equation (5.14) is formally identical to an inner product over the space of functions on the real line; by using the definition of such an inner product we can therefore write:

$$f(t) = \int_{-\infty}^{\infty} \langle \delta(s-\tau), f(s) \rangle \, \delta(t-\tau) d\tau \tag{5.19}$$

which is in turn formally identical to the reconstruction formula of Section 4.4.3. In reality we are interested in the space of 2π-periodic functions, since that is where DTFT's live;

this is easily accomplished by building a 2π-periodic version of the delta as:

$$\tilde{\delta}(\omega) = 2\pi \sum_{k=-\infty}^{\infty} \delta(\omega - 2\pi k). \tag{5.20}$$

where the leading 2π factor is for later convenience. The resulting object is called a *pulse train*, similarly to what we built for the case of periodic sequences in $\tilde{\mathbb{C}}^N$. Using the pulse train and given any 2π-periodic function $f(\omega)$, the reconstruction formula (5.19) becomes:

$$f(\omega) = \frac{1}{2\pi} \int_{\sigma}^{\sigma+2\pi} \langle \tilde{\delta}(\theta - \phi), f(\theta) \rangle \, \tilde{\delta}(\omega - \phi) d\phi \tag{5.21}$$

for any $\sigma \in \mathbb{R}$.

Now that we have the delta notation in place, we are ready to start. First of all, we will show the formal orthogonality of the basis functions $\{e^{j\omega n}\}_{\omega\in\mathbb{R}}$. We can write

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} \tilde{\delta}(\omega - \omega_0) e^{j\omega n} d\omega = e^{j\omega_0 n}; \tag{5.22}$$

the left-hand side of this equation has the exact form of the DTFT reconstruction formula (5.2) and so we have found the the fundamental relationship

$$e^{j\omega_0 n} \overset{\text{DTFT}}{\longleftrightarrow} \tilde{\delta}(\omega - \omega_0). \tag{5.23}$$

Now, the DTFT of a complex exponential $e^{j\sigma n}$ is, in our change of basis interpretation, simply the inner product $\langle e^{j\omega n}, e^{j\sigma n} \rangle$; because of (5.23) we can therefore write:

$$\langle e^{j\omega n}, e^{j\sigma n} \rangle = \tilde{\delta}(\omega - \sigma). \tag{5.24}$$

We will now recall for the last time that the delta notation subsumes a limiting operation: the DTFT pair (5.23) should be interpreted as a shorthand for the limit of the partial sums

$$s_k(\omega) = \sum_{n=-k}^{k} e^{-j\omega n}$$

(where we have chosen $\omega_0 = 0$ for the sake of example). Figure 5.4 plots $|s_k(\omega)|$ for increasing values of k (we show only the $[-\pi, \pi]$ interval, although of course the functions are 2π-periodic). The family of functions $s_k(\omega)$ is exactly equivalent to the family of $r_k(t)$'s we saw in (5.16); they too become narrower and narrower while keeping a constant area (which turns out to be 2π). That is why we can say simply that $s_k(\omega) \to \tilde{\delta}(\omega)$.

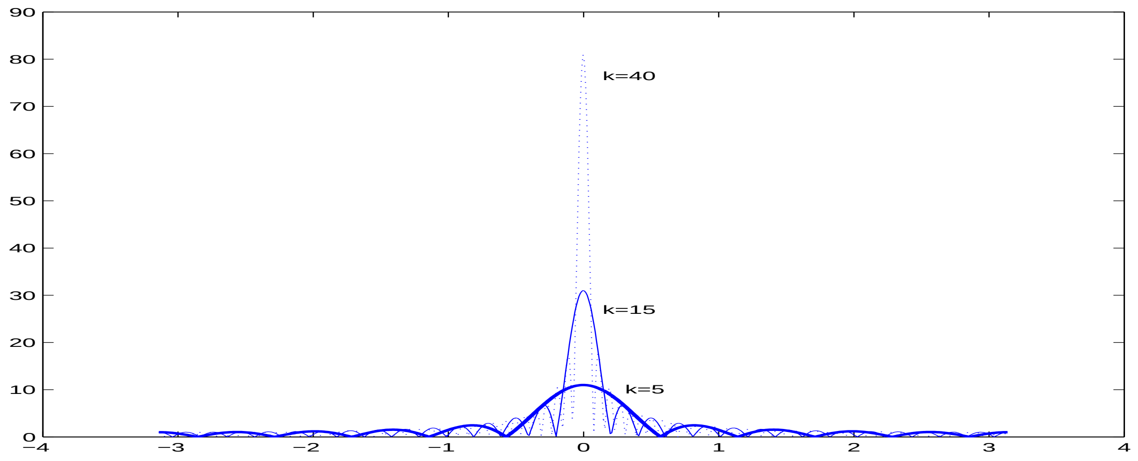


Figure 5.4: Plot of the function $|\sum_{n=-k}^{k} e^{-j\omega n}|$ for different values of k.

From (5.23) we can easily obtain other interesting results: by setting $\omega_0 = 0$ and by exploiting the linearity of the DTFT operator we can derive the DTFT of a constant sequence:

$$\alpha \overset{\text{DTFT}}{\longleftrightarrow} \alpha\tilde{\delta}(\omega); \tag{5.25}$$

or, using Euler's formulas, the DTFT's of sinusoidal functions:

$$\cos(\omega_0 n + \phi) \quad \overset{\text{DTFT}}{\longleftrightarrow} \quad (1/2)[e^{j\phi}\tilde{\delta}(\omega - \omega_0) + e^{-j\phi}\tilde{\delta}(\omega + \omega_0)] \tag{5.26}$$

$$\sin(\omega_0 n + \phi) \quad \overset{\text{DTFT}}{\longleftrightarrow} \quad (-j/2)[e^{j\phi}\tilde{\delta}(\omega - \omega_0) - e^{-j\phi}\tilde{\delta}(\omega + \omega_0)] \tag{5.27}$$

As we can see from the above examples, we are defining the DTFT for sequences which are not even square summable; again, these transforms are purely a notational formalism used to capture a behavior in the limit as we showed before.

5.3 Relationships Between Transforms

We will conclude this section on the DTFT by showing that, thanks to the delta formalism, the DTFT is the most general type of Fourier transform for discrete-time signals. Consider a length-N signal $x[n]$ and its N DFT coefficients $X[k]$; consider also the sequences obtained from $x[n]$ either by periodization or by building a finite-support sequence. The computation of the DTFT's of these sequences will highlight the relationships linking the three types of discrete-time transforms we have seen so far.

Periodic Sequences. Given a length-N signal $x[n]$, $n = 0, \ldots, N-1$, consider the associated N-periodic sequence $\tilde{x}[n] = x[n \mod N]$ and its N DFS coefficients $X[k]$. If we try to write the analysis DTFT formula for $\tilde{x}[n]$ we have:

$$\begin{aligned}
\tilde{X}(e^{j\omega}) &= \sum_{n=-\infty}^{\infty} \tilde{x}[n] e^{-j\omega n} \\
&= \sum_{n=-\infty}^{\infty} \left(\frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}nk} \right) e^{-j\omega n} && (5.28) \\
&= \frac{1}{N} \sum_{k=0}^{N-1} X[k] \left(\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi}{N}nk} e^{-j\omega n} \right) && (5.29)
\end{aligned}$$

where in (5.28) we have used the DFS reconstruction formula. Now it is immediate to recognize in the last term of (5.29) as the DTFT of a complex exponential of frequency $(2\pi/N)k$; we can therefore write

$$\tilde{X}(e^{j\omega}) = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \tilde{\delta}(\omega - \frac{2\pi}{N}k) \tag{5.30}$$

which is the relationship between the DTFT and the DFS. If we restrict ourselves to the $[-\pi, \pi]$ interval, we can see that the DTFT of a periodic sequence is a series of regularly spaced deltas placed at the N roots of unity and whose amplitude is proportional to the DFS coefficients of the sequence. In other words, *the DTFT is uniquely determined by the DFS and vice versa.*

Finite-Support Sequences. Given a length-N signal $x[n], n = 0, \ldots, N-1$ and its N DFT coefficients $X[k]$, consider the associated finite-support sequence:

$$\bar{x}[n] = \begin{cases} x[n] & 0 \leq n < N \\ 0 & \text{otherwise} \end{cases} ;$$

we can easily derive the DTFT of $\bar{x}$ as

$$\bar{X}(e^{j\omega}) = \sum_{k=0}^{N-1} X[k] \Lambda(\omega - \frac{2\pi}{N}k) \tag{5.31}$$

with

$$\Lambda(\omega) = \frac{1}{N} \sum_{m=0}^{N-1} e^{-j\omega m}$$

What the above expression means is that the DTFT of the finite support sequence $\bar{x}[n]$ is again *uniquely defined by the N DFT coefficients of the finite-length signal $x[n]$* and it can be obtained by a simple Lagrangian interpolation. As in the previous case, the values of DTFT at the roots of unity are equal to the DFT coefficients; note, however, that the transform of a finite support sequence is very different from the DTFT of a periodized sequence. The latter, in accordance with the definition of the Dirac delta, is defined only in the limit and for a finite set of frequencies; the former is just a (smooth) interpolation of the DFT.

5.4 Properties of the DTFT

The DTFT possesses the following properties:

Symmetries & Structure. The DTFT of a time-reversed sequence is:

$$x[-n] \overset{\text{DTFT}}{\longleftrightarrow} X(e^{-j\omega}) \tag{5.32}$$

while, for the complex conjugate of a sequence we have

$$x^*[-n] \overset{\text{DTFT}}{\longleftrightarrow} X^*(e^{-j\omega}). \tag{5.33}$$

For the very important case of a *real* sequence $x[n] \in \mathbb{R}$ we have that the DTFT is conjugate-symmetric:

$$X(e^{j\omega}) = X^*(e^{-j\omega}) \tag{5.34}$$

which leads to the following symmetries (again: for **real** signals only):

$$|X(e^{j\omega})| = |X(e^{-j\omega})| \qquad \textit{the magnitude is symmetric} \tag{5.35}$$

$$\measuredangle X(e^{j\omega}) = -\measuredangle X(e^{-j\omega}) \qquad \textit{the phase is antisymmetric} \tag{5.36}$$

$$\text{Re}\{X(e^{j\omega})\} = \text{Re}\{X(e^{-j\omega})\} \qquad \textit{the real part is symmetric} \tag{5.37}$$

$$\text{Im}\{X(e^{j\omega})\} = -\text{Im}\{X(e^{-j\omega})\} \qquad \textit{the imaginary is antisymmetric} \tag{5.38}$$

Finally, if $x[n]$ is real and symmetric, then the DTFT is real:

$$x[n] \in \mathbb{R},\ x[-n] = x[n] \Longleftrightarrow X(e^{j\omega}) \in \mathbb{R} \tag{5.39}$$

while, for real antisymmetric signals we have that the DTFT is purely imaginary:

$$x[n] \in \mathbb{R},\ x[-n] = -x[n] \Longleftrightarrow \text{Re}\{X(e^{j\omega})\} = 0. \tag{5.40}$$

Linearity & Shifts. The DTFT is a linear operator:

$$\alpha x[n] + \beta y[n] \overset{\text{DTFT}}{\longleftrightarrow} \alpha X(e^{j\omega}) + \beta Y(e^{j\omega}). \tag{5.41}$$

A shift in the discrete-time domain leads to multiplication by a phase term in the frequency domain:

$$x[n-n_0] \overset{\text{DTFT}}{\longleftrightarrow} e^{-j\omega n_0} X(e^{j\omega}) \tag{5.42}$$

while multiplication of the signal by a complex exponential (i.e. signal *modulation* by a complex "carrier" at frequency ω_0) leads to:

$$e^{j\omega_0 n} x[n] \overset{\text{DTFT}}{\longleftrightarrow} X(e^{j(\omega-\omega_0)}) \tag{5.43}$$

which means that the spectrum is shifted by ω_0. This last result is known as the *Modulation Theorem.*

Energy Conservation. The DTFT satisfyes the *Plancherel-Parseval* equality:

$$\langle x[n], y[n] \rangle = \frac{1}{2\pi} \langle X(e^{j\omega}), Y(e^{j\omega}) \rangle \tag{5.44}$$

or, using the respective definitions of inner product for $l_2(\mathbb{Z})$ and $L_2([-\pi, \pi])$:

$$\sum_{n=-\infty}^{\infty} x^*[n] y[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X^*(e^{j\omega}) Y(e^{j\omega}) d\omega \tag{5.45}$$

(note the explicit normalization factor $1/2\pi$). The above equality specializes into *Parseval's Theorem* as:

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} |X(e^{j\omega})|^2 d\omega \tag{5.46}$$

which establishes the conservation of energy property between the time and the frequency domains.

Modulation Property

$$x[n]y[n] \overset{DTFT}{\longleftrightarrow} \frac{1}{2\pi} \int_{-\pi}^{\pi} X\left(e^{j\theta}\right) Y\left(e^{j(\omega-\theta)}\right) \mathrm{d}\theta$$

<u>Proof</u>:

$$\sum_n y[n] x[n] e^{-j\omega n} = \sum_n y[n] \frac{1}{2\pi} \int_{-\pi}^{\pi} X\left(e^{j\theta}\right) e^{j\theta n} \mathrm{d}\theta \, e^{-j\omega n}$$

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

Planchard-Parseval equality

$$\sum_n x^*[n]y[n] \quad \overset{DTFT}{\longleftrightarrow} \quad \frac{1}{2\pi}\int_{-\pi}^{\pi} X^*\left(e^{j\omega}\right)Y\left(e^{j\omega}\right)\mathrm{d}\omega$$

$$\text{or} \qquad \langle \mathbf{x}, \mathbf{y}\rangle \quad \overset{DTFT}{\longleftrightarrow} \quad \frac{1}{2\pi}\langle \mathbf{X}, \mathbf{Y}\rangle$$

<u>Proof:</u> From the modulation property, we have

$$z[n] = x^*[n]y[n] \overset{DTFT}{\longleftrightarrow} Z\left(e^{j\omega}\right) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X^*\left(e^{-j\theta}\right)Y\left(e^{j(\omega-\theta)}\right)\mathrm{d}\theta$$

$$\Longrightarrow \sum_n x^*[n]y[n] = \sum_n z[n] = Z\left(e^{j\omega}\right)\Big|_{\omega=0} = \frac{1}{2\pi}\int_{-\pi}^{\pi} X^*\left(e^{-j\theta}\right)Y\left(e^{-j\theta}\right)\mathrm{d}\theta$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} X^*\left(e^{j\theta}\right)Y\left(e^{j\theta}\right)\mathrm{d}\theta$$

Example 5.2

a) Determine the DTFT of the sequence

$$x[n] = \begin{cases} A & \text{for } 0 \le n \le L-1 \\ 0 & \text{otherwise} \end{cases}$$

b) Give an approximate plot of the magnitude and phase of the spectrum (feel free to use matlab: create a sequence $x[n]$ of length $N = 1000$ for example, use `fft` or `myDFT` to compute the N-point DFT and set $A = 1$ and $L = 5$, check out the commands `abs` and `angle`).

c) Given $Y(e^{j\omega}) = e^{-j\omega\beta}X(e^{j\omega})$, calculate $Z(e^{j\omega}) = X(e^{j\omega}) * Y(e^{j\omega})$ for $\beta \in \mathbb{Z}, \beta < 0$.

Solution:

a) First, we calculate the DTFT of $x[n]$ as follows:

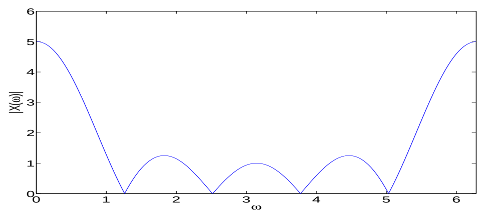
$$X(e^{j\omega}) = \sum_{n=0}^{L-1} Ae^{-j\omega n} = A\frac{1-e^{-j\omega L}}{1-e^{-j\omega}} = Ae^{-j(\frac{\omega}{2})(L-1)}\frac{\sin(\omega L/2)}{\sin(\omega/2)}$$

b) The amplitude and phase of $X(\omega)$ are given by:

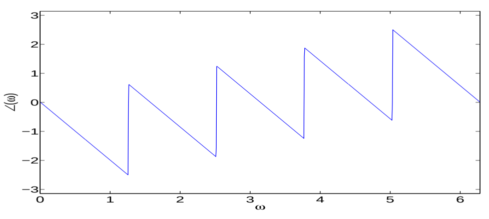
$$|X(e^{j\omega})| = \begin{cases} |A|L & \textit{when } \omega = 0 \\ |A||\frac{\sin(\omega L/2)}{\sin(\omega/2)}| & \textit{otherwise} \end{cases}$$

$$\angle X(e^{j\omega}) = \angle A - (\omega/2)(L-1) + \angle\frac{\sin(\omega L/2)}{\sin(\omega/2)}$$

where we should remember that the phase of a real quantity is zero if the quantity is positive and π if it is negative. The amplitude and phase are plotted in Figure 5.5.



(a) AMPLITUDE



(b) PHASE

Figure 5.5: Amplitude and phase of $X(e^{j\omega})$

c) We know that convolution in the frequency domain is equivalent to multiplication in the time domain. Hence, $z[n] = x[n]\,y[n] \overset{DTFT}{\leftrightarrow} Z(e^{j\omega}) = X(e^{j\omega}) * Y(e^{j\omega})$. Further, $Y(e^{j\omega}) = e^{-j\omega\beta}X(e^{j\omega}) \overset{DTFT}{\leftrightarrow} y[n] = x[n-\beta]$. Given that $\beta < 0$, x is a delayed version of y (see Figure 3). Straightforwardly, $Z(e^{jw})$ is the DTFT of $z[n]$, which has the same form as $x[n]$. Setting $A \to A^2$ and $L-1 \to L-1+\beta$ in the solution to point (a), we get:

$$Z(e^{j\omega}) = A^2 e^{-j(\frac{\omega}{2})(L+\beta-1)}\frac{\sin(\omega(L+\beta)/2)}{\sin(\omega/2)}$$

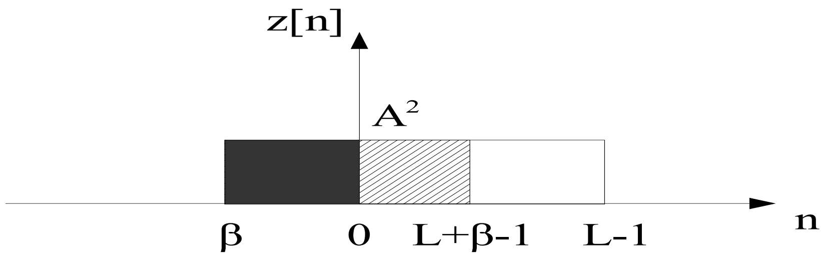


Figure 5.6: The dark rectangle represents $y[n]$, while the blank rectangle represents $x[n]$. The hatched zone is where the two signal overlap and their product is non-zero.

5.5 Summary

This chapter introduced the concept of the discrete-time Fourier Transform. Here is a table of common transforms:

Some DTFT pairs:	
$x[n] = \delta[n-k]$	$X(e^{j\omega}) = e^{-j\omega k}$
$x[n] = 1$	$X(e^{j\omega}) = \tilde{\delta}(\omega)$
$x[n] = u[n]$	$X(e^{j\omega}) = \frac{1}{1-e^{-j\omega}} + \frac{1}{2}\tilde{\delta}(\omega)$
$x[n] = a^n u[n] \quad \|a\| < 1$	$X(e^{j\omega}) = \frac{1}{1-ae^{-j\omega}}$
$x[n] = e^{j\omega_0 n}$	$X(e^{j\omega}) = \tilde{\delta}(\omega - \omega_0)$
$x[n] = \cos(\omega_0 n + \phi)$	$X(e^{j\omega}) = (1/2)[e^{j\phi}\tilde{\delta}(\omega - \omega_0) + e^{-j\phi}\tilde{\delta}(\omega + \omega_0)]$
$x[n] = \sin(\omega_0 n + \phi)$	$X(e^{j\omega}) = (-j/2)[e^{j\phi}\tilde{\delta}(\omega - \omega_0) - e^{-j\phi}\tilde{\delta}(\omega + \omega_0)]$
$x[n] = \begin{cases} 1 & \text{for } 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases}$	$X(e^{j\omega}) = \frac{\sin((N/2)\omega)}{\sin(\omega/2)}\, e^{-j\frac{N-1}{2}\omega}$

5.6 Problems

Problem 5.1 *Let $x[n]$ and $y[n]$ be two complex valued sequences and $X(e^{jw})$ and $Y(e^{jw})$ their corresponding DTFTs.*

(a) Show that

$$\langle x[n], y[n] \rangle = \frac{1}{2\pi} \langle X(e^{jw}), Y(e^{jw}) \rangle,$$

where we use the inner products for $l_2(\mathbb{Z})$ and $L_2([-\pi, \pi])$ respectively.

(b) What is the physical meaning of the above formula when $x[n] = y[n]$?

Problem 5.2 (DFT and DTFT) *Consider the infinite non-periodic sequence*

$$x[n] = \begin{cases} 0 & n < 3 \\ 1 & 3 \le n < 10 \\ -1 & 10 \le n < 15 \\ 0 & n \ge 15. \end{cases}$$

(a) Derive the DTFT $X(e^{jw})$ of $x[n]$. (Don't use MATLAB.)

(b) Use MATLAB to plot the magnitude of $X(e^{jw})$ for 1000 points in the interval $[0, 2\pi]$.

(c) In MATLAB, use your `myDFT` function with $N = 20$ to compute the DFT of $x[n]$. Compare this plot to one obtained in (b).

(d) Repeat part (c) for $N = 50, 100, 1000$. What can you conclude?

(e) Can you prove your your conclusion analytically?

Problem 5.3 *Let $x[n]$ be a discrete-time sequence defined as*

$$x[n] = \begin{cases} M - n & 0 \le n \le M, \\ M + n & -M \le n \le 0, \\ 0 & \textit{otherwise.} \end{cases}$$

for some odd integer M.

(a) Show that $x[n]$ can be express as the convolution of two discrete-time sequences $x_1[n]$ and $x_2[n]$. Check your result with Matlab for $M = 11$.

(b) Using the results found in (a), compute the DTFT of $x[n]$.

Problem 5.4 *Consider the system $\mathcal{H}$ implementing the input-output relation $y[n] = \mathcal{H}\{x[n]\} = x^2[n]$.*

(a) Prove by example that the system is nonlinear.

(b) Prove that the system is time-invariant

Now consider the following cascade system:

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{\mathcal{H}} \arrow[r, "{y[n]}"] & \boxed{\mathcal{G}} \arrow[r] & v[n] \end{tikzcd}$$

where $\mathcal{G}$ is the following ideal highpass filter:

$$G(e^{j\omega}) = \begin{cases} 0 & \textit{for } |\omega| < \pi/2 \\ 2 & \textit{otherwise} \end{cases}$$

(as per usual, $G(e^{j\omega})$ is 2π-periodic (i.e. prolonged by periodicity outside of $[-\pi, \pi]$)). The output of the cascade is therefore $v[n] = \mathcal{G}\{\mathcal{H}\{x[n]\}\}$.

(c) Compute $v[n]$ when $x[n] = \cos(\omega_0 n)$ for $\omega_0 = 3\pi/8$. How would you describe the transformation operated by the cascade on the input?

(d) Compute $v[n]$ as before, with now $\omega_0 = 7\pi/8$

Chapter 6

Fourier Analysis - Practice

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

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

6.1 The Transforms in Practice

We will recall the DTFT is mostly a theoretical analysis tool; the DTFT's which can be computed exactly (i.e. those in which the sum in (5.1 can be solved in closed form) represent only a small set of sequences; yet, these sequences are highly representative and they will be used over and over to illustrate a prototypical behavior. The DFT[1], on the other hand, is fundamentally a *numerical* tool in that it defines a finite set of operations which can be computed in a finite amount of time; in fact, a very efficient algorithmic implementation of the DFT exists under the name of Fast Fourier Transform (FFT) which only requires on the order of $N\log(N)$ operations for an N-point vector. The DFT, as we know, only applies to finite-length signals but this is actually fine since in practice all measured signals have finite support; in principle, therefore, the DFT suffices for the spectral characterization of real-world sequences. Since the transform of a finite-length signal and its DTFT are related by (5.30) or by (5.31) according to the underlying model for the infinite-length extension, we can always use the DTFT to illustrate the fundamental concepts of spectral analysis for the general case and then particularize the results for finite-length sequences.

6.1.1 Plotting Spectral Data

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

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

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

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

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

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

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

6.1.2 Computing the Transform: the FFT

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

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

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

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

6.1.3 Cosmetics: Zero Padding

FFT algorithms are tailored to the specific length of the input signal. When the input signal's length is a large prime number or when only a subset of FFT algorithms is available (when, for instance, all we have is the radix-2 algorithm, which processes input vector

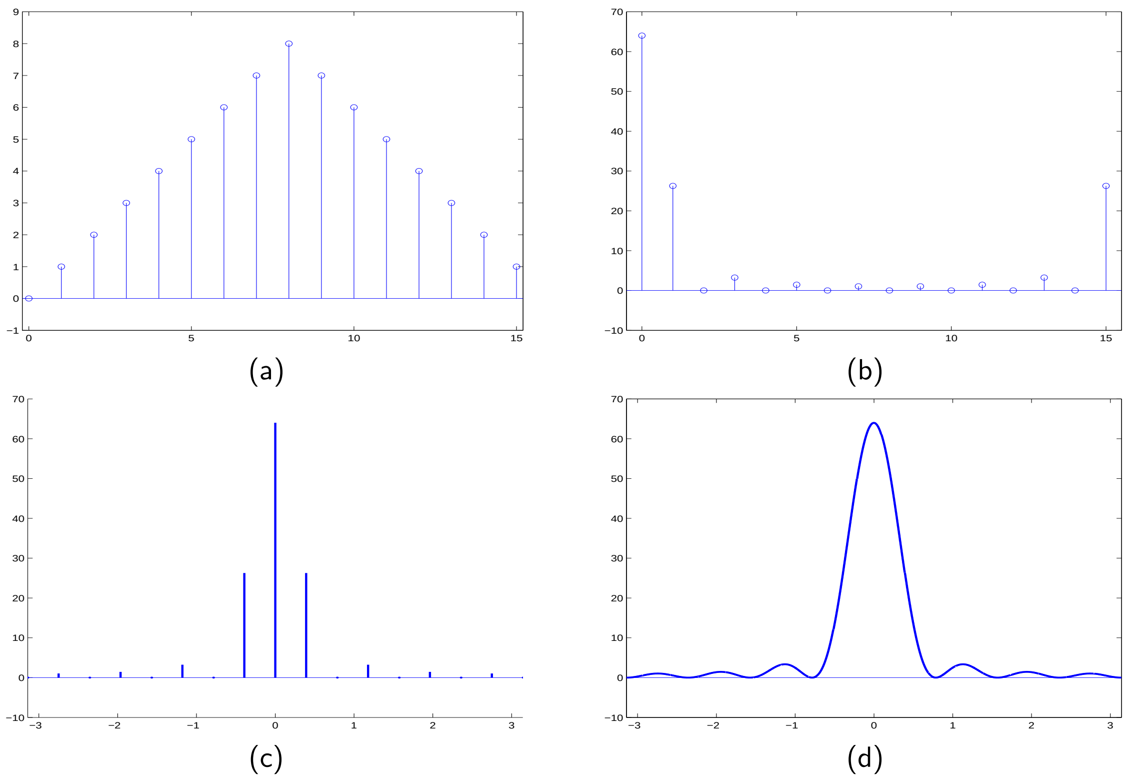


Figure 6.1: Plotting spectral information (all transforms are plotted in magnitude only). (a) Original length-16 signal; (b) Its DFT (or, equivalently, one period of its DFS); (c) The DTFT of its periodic extension; (d) The DTFT of its finite-support extension.

with length a power of two) it is customary to extend the length of the signal to match the algorithmic requirements. This is usually achieved by *zero padding*, i.e. the length-N data vector is extended to a chosen length M by appending $(M - N)$ zeros to it. Now, the maximum resolution of an N-point DFT, i.e. the separation between frequency components, is $2\pi/N$. By extending the signal to a longer length M, we are indeed reducing the separation between frequency components. One may think that this artificial increase in resolution will allow the DFT to show finer details of the input signal's spectrum. It is not so.

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

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

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

$$\mathbf{M}_{M,N} = \mathbf{W}'_M \mathbf{W}^H_N$$

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

6.2 Spectral Analysis

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

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

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

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

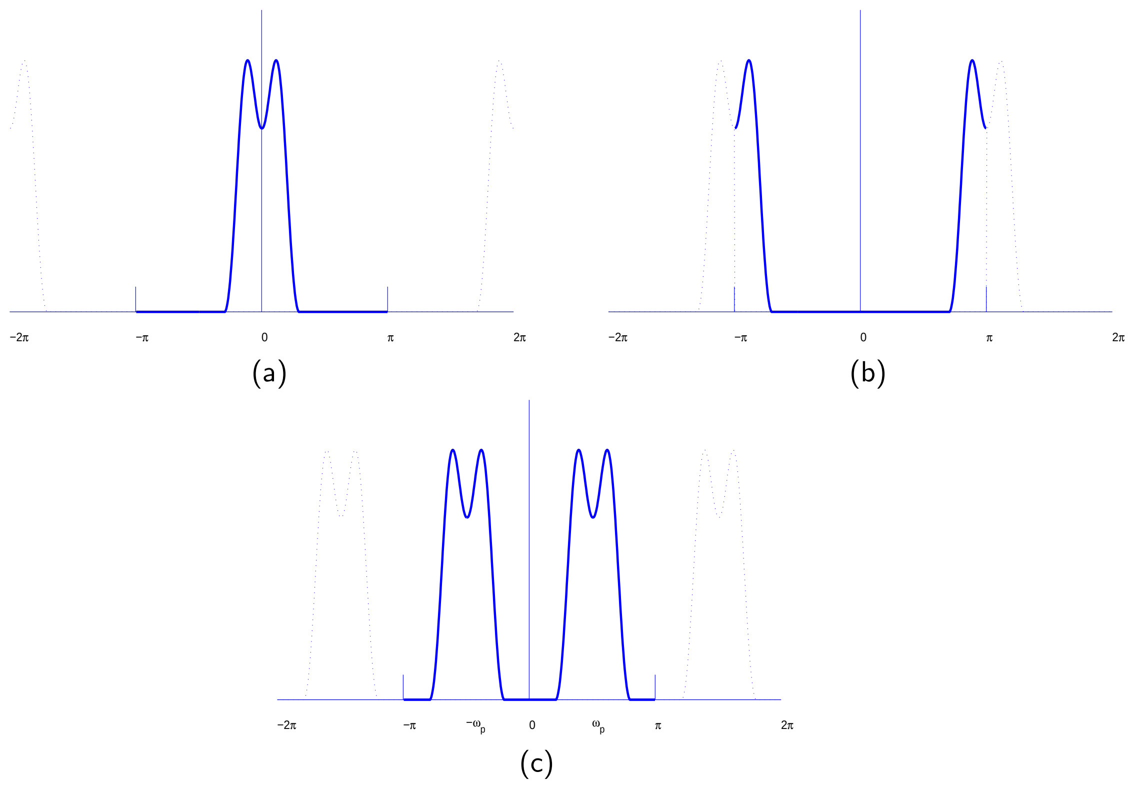


Figure 6.2: Classification of signal based on spectral magnitude.
(a) Baseband; (b) Passband; (c) Highpass. Note the 2π-periodicity of the spectrum (the replicas are plotted with a dotted line).

Phase As we said before, the Fourier representation allows us to think of any signal as the sum of the outputs of a (potentially infinite) number of sinusoidal generators. While the magnitude of the spectrum defines the inherent power produced by each of the generators, its phase defines the relative alignment of the generated sinusoids. This alignment determines the *shape* of the signal in the discrete-time domain. To illustrate this with an example, consider the following 64-periodic signal[3]:

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

the magnitude of the DFS $\tilde{X}[k]$ is independent of the values of $\phi_0, \ldots, \phi_3$ and it is plotted in Figure 6.3-(a). If we set $\phi_k = 0, k = 0, 1, 2, 3$ we obtain the discrete-time signal which is plotted in Figure 6.3-(b). Now consider modifying the individual phases so that $\phi_k = 2\pi k/3$; the resulting signal is the one depicted in Figure 6.3-(c) for which, as we just noted, the magnitude DFS does not change.

6.3 Time-Frequency Analysis

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

6.3.1 The Spectrogram

The simplest time-frequency transformation is called the spectrogram. This consists in splitting the signal into small consecutive (and possibly overlapping) length-N pieces and computing the DFT of each. What we obtain is the following function of discrete-time and frequency index:

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

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

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

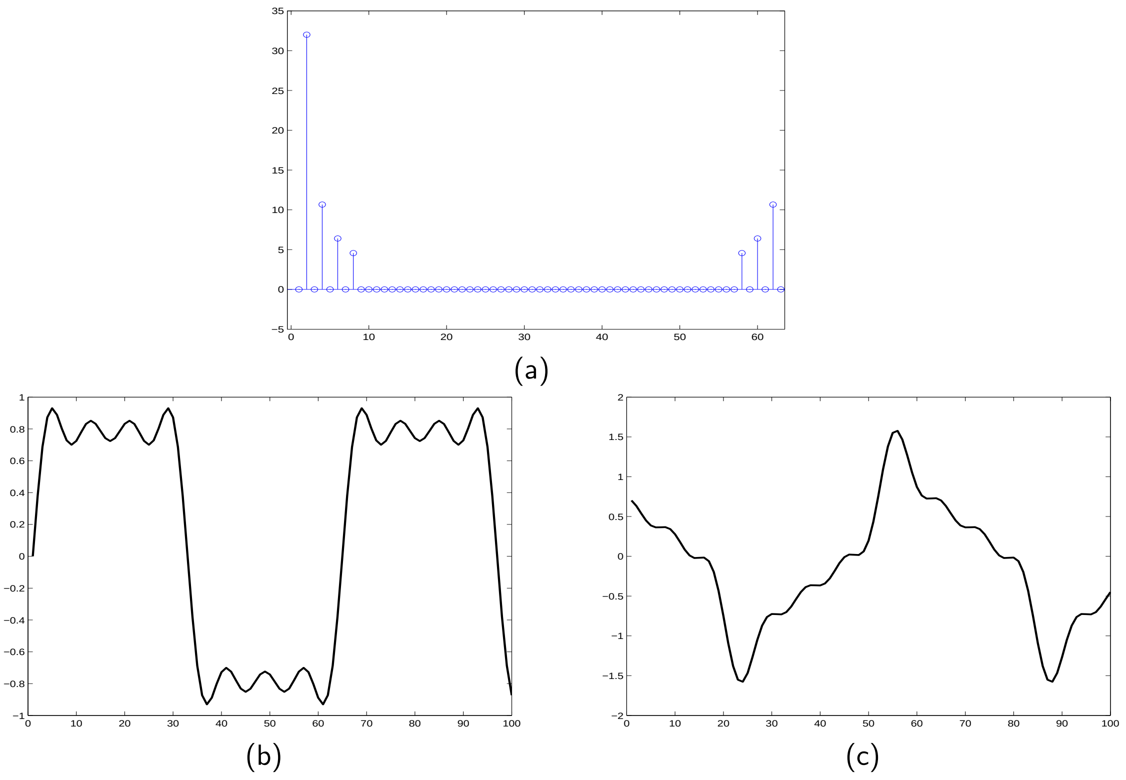


Figure 6.3: Effects of phase shift. (a) The magnitude DFS; (b) The signal with zero phase; (c) The same signal with a linear phase term.

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

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

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

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

6.3.2 The Uncertainty Principle

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

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

6.4 Digital Frequency vs. Real Frequency

We have seen that, in the representation of discrete-time signals, the notion of a dimensionless discrete “time” makes the whole ensemble of signal processing proofs and tools

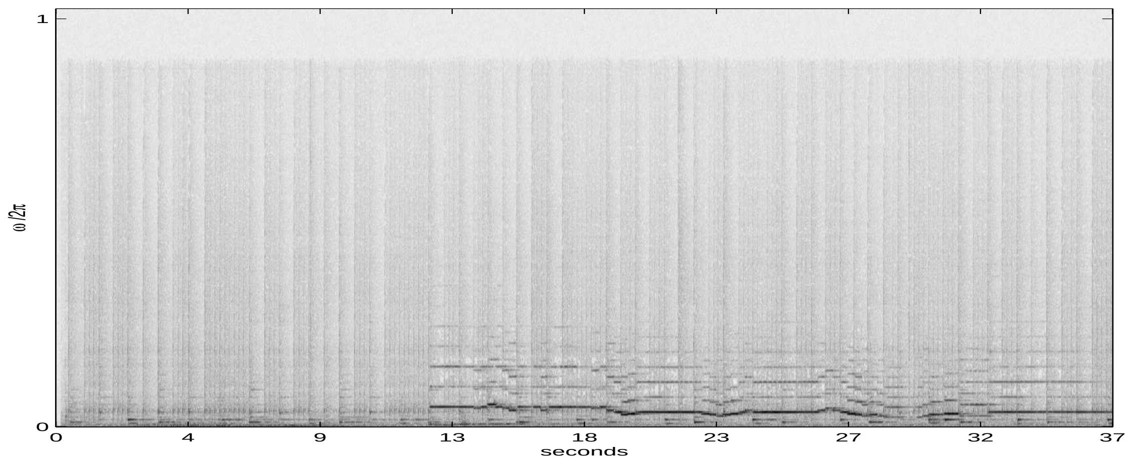


Figure 6.4: Spectrogram representation of the beginning of Ravel's Bolero. DFT size is 1024 samples, overlap is 512 samples.

independent of the underlying physical nature that the signals represent. Similarly, we have just derived a frequency representation for signals which is based on a notion of dimensionless frequency; because of the periodicity of the Fourier basis, all we know is that π is the highest digital frequency we can represent. Again, the power of generality is (or will soon be) apparent: a digital filter which is designed to remove the upper half of a signal's spectrum can be used with any type of input sequence and the results will stay the same. This abstraction, however, is not without its drawbacks from the point of view of intuition; after all, we are very familiar with signals in the real world for which time is expressed in seconds and frequency is expressed in Hertz. We say for instance that speech has a bandwidth up to 4KHz, that the human ear is sensitive to frequencies up to 20KHz, that a cell phone transmits in the GHz band, and so on. What does "π" mean in these cases? The precise, formal link between real-world signal and discrete-time signal processing is given by the sampling theorem, which we will study later. The fundamental idea, however, is that we can remove the abstract nature of a discrete-time signal (and, correspondingly, of a dimensionless frequency) *by associating a time duration to the interval between successive discrete-time indices in the sequence.*

Let us say that the "real-world" time between indices n and $n + 1$ in a discrete-time sequence is T seconds; this could correspond to sampling a signal every T seconds or to generating a synthetic sequence with a DSP chip whose clock cycle is T seconds. Recall that the phase increment between successive samples of a generic complex exponential $e^{j\omega n}$

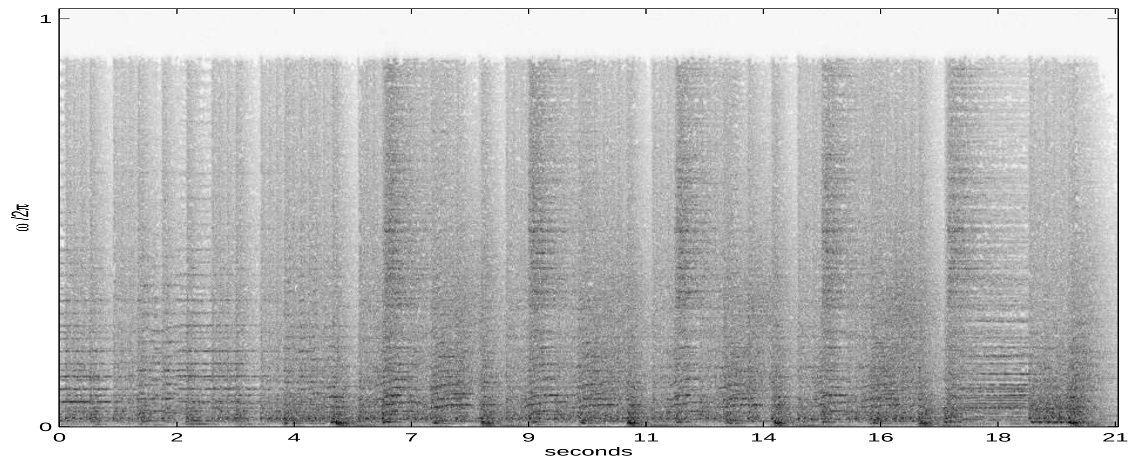


Figure 6.5: Spectrogram representation of the end of Ravel's Bolero.

is ω radians. The oscillation will therefore complete a full cycle in $n_f = (2\pi/\omega)$ samples. If T is the real-world time between samples, the full cycle will be completed in $n_f T$ seconds and so its frequency will be $f = (n_f T)^{-1}$. The relationship between the digital frequency ω and the "real" frequency f in Hertz as determined by the "clock" period T is therefore:

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

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

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

where we have used $F_s = (1/T)$; F_s is nothing but the operating frequency of the discrete time system (also called the *sampling frequency* or clock frequency). With this notation, the digital frequency ω_0 corresponding to a real frequency f_0 is:

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

The compact disk system, for instance, operates at a frequency $F_s = 44.1$KHz; the maximum representable frequency for the system is 22.5KHz (which constitutes the highest-pitched sound which can be encoded on and reproduced by a CD).

6.5 Problems

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

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

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

1. *To compute $X(0)...X(N-1)$, how many complex multiplications and additions do you have to perform (as a function of N) using the formula above?*
2. *Let $N = LM$. Instead of storing $x(n)$ in a vector, we now store store it in a table such that $x(l,m) = x(l+mL)$ as shown in Table 6.1.*

Table 6.1: Table representaion of a sequence

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

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

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

3. *One can decompose the above operation as follows:*
   - $F(l,q) = \sum_{m=0}^{M-1} x(l,m) W_M^{mq}$
     ($0 \leq q \leq M-1$, for each of the rows $l = 0...L-1$)
   - $G(l,q) = W_N^{lq} F(l,q)$
     ($0 \leq l \leq L-1$ and $0 \leq q \leq M-1$)
   - $X(p,q) = \sum_{l=0}^{L-1} G(l,q) W_L^{lp}$
     ($0 \leq p \leq L-1$, for each column $q = 0...M-1$)

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

Chapter 7

Linear Systems

7.1 Definition and Properties

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

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

Linearity is expressed by the equivalence

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.1.1 Properties of the convolution

The basic properties of the convolution operator are:

- Linearity:

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

- Time-invariance:

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

- Commutativity: (Figure 7.1)

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

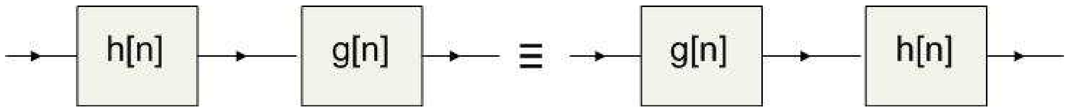


Figure 7.1: Commutativity

- Associativity:

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

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

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

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

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

7.1.2 The meaning of the convolution

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

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

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

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

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

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

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

7.1.3 Convolution of frequency spectrum

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

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

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

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

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

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

7.2 Circular convolution

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

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

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

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

where

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

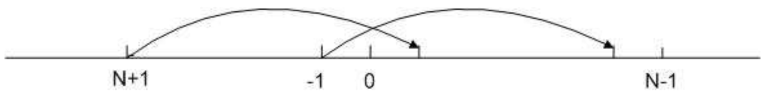


Figure 7.2: Circular convolution

7.3 Stability

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

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

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

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

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

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

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

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

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

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

Since,

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

then,

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

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

7.3.1 Causality

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

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

since for such a system

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

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

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

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

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

Example 7.1 (Linearity and Time Invariance)

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

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

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

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

Solution:

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

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

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

(c)
- $\mathcal{T}(x[n-k]) = x[n-k]\cos(\omega_0 n) \neq y[n-k] = x[n-k]\cos(\omega_0[n-k]) \rightarrow$ time variant
- $\mathcal{T}(\alpha x_1[n]+\beta x_2[n]) = (\alpha x_1[n]+\beta x_2[n])\cos(\omega_0[n]) = \alpha\mathcal{T}(x_1[n])+\beta\mathcal{T}(x_2[n]) \rightarrow$ linear.
- *The output only depends on present (and past) inputs* $\rightarrow$ causal.

Example 7.2 (Stability)

(a) Given an LTI system with impulse response

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

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

(b) Consider the system given by

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

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

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

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

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

Solution:

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

(b) *Observe that:*

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

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

(c) *We have*

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

then,

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

Example 7.3 *Consider the interconnection of systems shown in Fig. 7.3.*

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

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

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

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

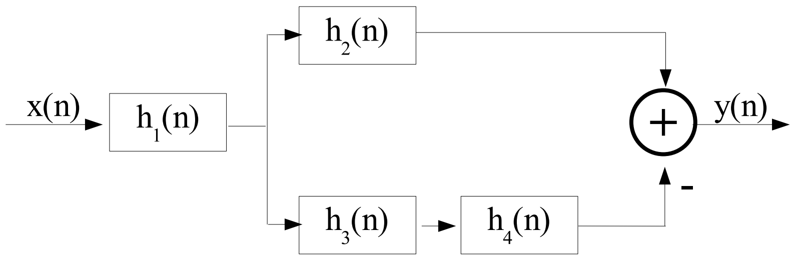


Figure 7.3: System for Problem 7.3

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

Solution:

(a) One can immediately see that:

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

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

$$\begin{aligned} h_{34}[n] = &\quad \textstyle\sum_{k=-\infty}^{\infty} u[k](k+1)\delta[n-2-k] = u[n-2](n-1) \\ h_{234}[n] = &\quad h_2[n] - h_{34}[n] = (n+1)u[n] - (n-1)u[n-2] = \begin{cases} 0 & \textit{when } n<0 \\ 1 & \textit{when } n=0 \\ 2 & \textit{otherwise} \end{cases} \\ h[n] = &\quad h_1[n] * h_{234}[n] = \textstyle\sum_{k=0}^{2} h_1[k]\, h_{234}[n-k] = \left\{ \overbrace{\frac{1}{2}}^{n=0}, \frac{5}{4}, 2, \frac{5}{2}, \frac{5}{2}, \frac{5}{2}, ... \right\} \end{aligned}$$

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

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

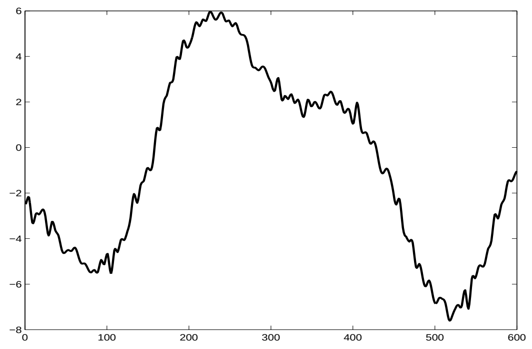


Figure 7.4: Noisy signal.

7.4 Introduction to Filtering

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

7.4.1 FIR filtering

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

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

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

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

The impulse response, as it turns out, is a finite-support sequence so the filter we just built is an FIR filter; this particular filter goes under the name of *Moving Average* (MA) filter. The "smoothing power" of this filter is dependent on the number of samples we take into account in the average or, in other words, on the length N of its impulse response. The

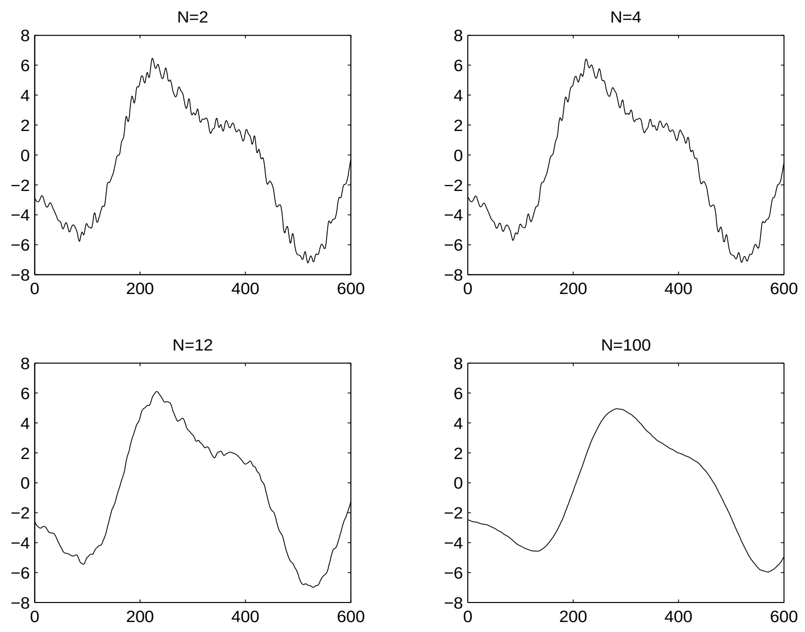


Figure 7.5: Moving averages for different values of N.

filtered version of the original sequence for increasing values of N is plotted in Figure 7.5. Intuitively we can see that as N grows, more and more wiggles are removed. We will soon see how to handle the "smoothing power" of a filter in a precise, quantitative way. One thing to notice right away, and which is a general characteristic of FIR filters, is that the value of the output does not depend on values of the input which are more than N steps away; FIR filters are therefore called *memoryless* filters. Another remark we can mention right away concerns the *delay* introduced by the filter: each output value is the average of a window of N input values whose representative sample is the one falling in the middle; there is therefore a delay of $N/2$ samples between input and output, and the delay grows with N.

7.4.2 IIR filtering

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

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

We can easily see that:

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

where we have defined $\lambda = (M-1)/M$. Now, as M grows large, we can safely assume that if we compute the average over $M-1$ or over M points the result is basically the same: in other words, for M large, we can say that $y_{M-1}[n] \approx y_M[n]$. This suggests a new way to compute the smoothed version of a sequence in a *recursive* fashion:

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

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

Linearity of (7.18) can now be proved this way. Assume that the output sequence for the system defined by (7.18) is $y[n]$ when the input is $x[n]$. It is immediate to see that $y_1[n] = \alpha y[n]$ satisfies (7.18) for an input equal to $\alpha x[n]$. All we need to prove is that this is the only solution. Assume this is not the case and call $y_2[n]$ the other solution; we have:

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

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

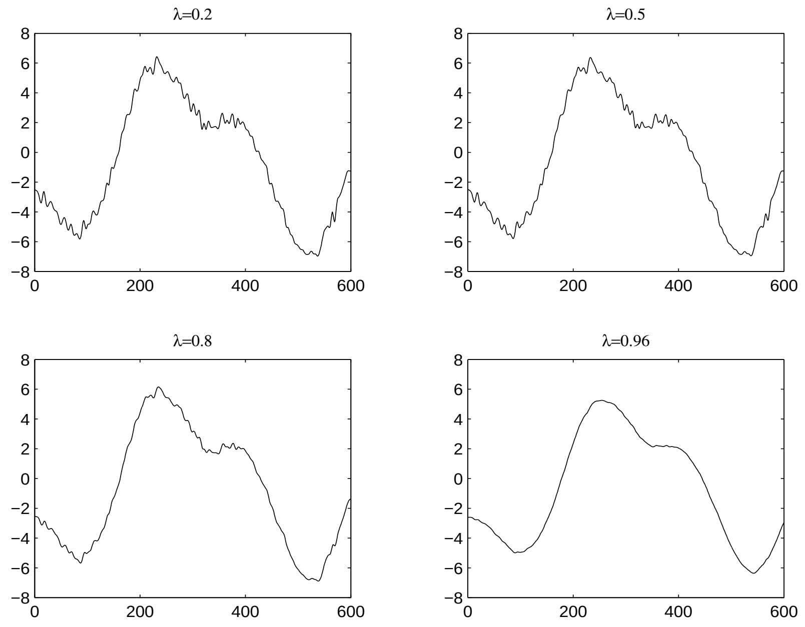


Figure 7.6: Outputs of the leaky integrator for different values of λ.

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

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

so that the impulse response is:

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

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

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

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

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

7.5 Filtering in the Frequency Domain

The above examples have introduced the notion of filtering in an operational and intuitive way. In order to make more precise statements on the characteristics of a discrete-time filter we need to move to the frequency domain. What does a filtering operation translate to in the frequency domain? The fundamental result of this section is the convolution theorem for discrete-time signals: a convolution in the discrete-time domain is equivalent to a multiplication of Fourier transforms in the frequency domain. This result opens up a very fruitful perspective on filtering and filter design, together with alternative approaches to the implementation of filtering devices, as we will see.

7.5.1 Preliminaries

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

ω_0; we have

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

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

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

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

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

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

7.5.2 The Convolution and Modulation theorems

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

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

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

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

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

Table 7.1: The Convolution and Modulation Theorems

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

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

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

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

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

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

and we split the last integral to obtain

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

These fundamental results are summarized in Table 7.1.

7.6 The Frequency Response

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

7.6.1 Magnitude

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

- **Lowpass filters**, for which the magnitude of the transform is concentrated around $\omega = 0$; these filter preserve the low-frequency energy of the input signals and attenuate or eliminate the high-frequency components.
- **Highpass filters**, for which the magnitude of the transform is concentrated around $\omega = \pm\pi$; these filter preserve the high-frequency energy of the input signals and attenuate or eliminate the low-frequency components.
- **Bandpass filters**, for which the magnitude of the transform is concentrated around $\omega = \pm\omega_p$; these filter preserve the energy of the input signals around the frequency ω_p and attenuate the signals elsewhere, notably around $\omega = 0$ and $\omega = \pm\pi$.
- **Allpass filters**, for which the magnitude of the transform is a *constant* over the entire $[-\pi, \pi]$ interval. These filters do not affect their input's spectral magnitude (except for a constant gain factor) and they are designed entirely in terms of their phase response (typically, to introduce or compensate for a delay).

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

7.6.2 Phase

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

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

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

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

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

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

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

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

For a general filter with linear phase we can always write

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

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

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

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

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

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

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

7.7 Examples of Filters

7.7.1 Ideal Filters

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

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

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

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

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

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

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

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

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

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

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

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

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

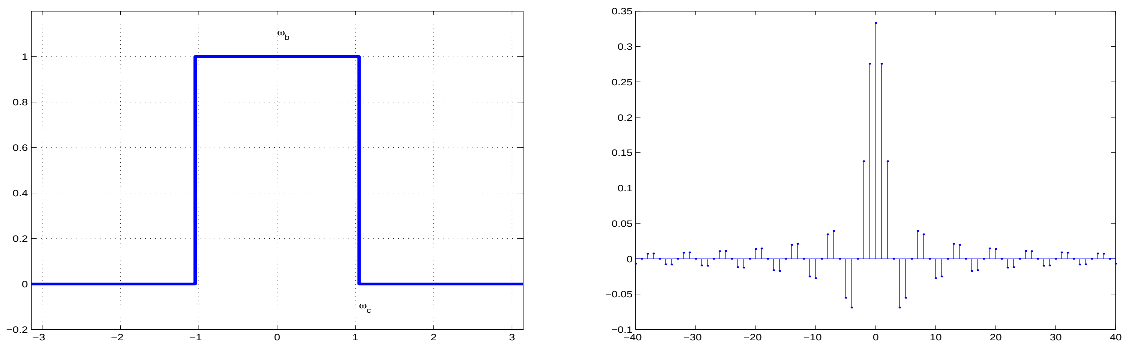


Figure 7.7: Ideal lowpass filter, $\omega_c = \pi/3$. (a) Frequency response; (b) Impulse response (portion).

or, in terms of cutoff frequency,

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

The DTFT pair:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

as one can verify from (5.26) and (5.27). More generally, the Hilbert filter is used in communication systems to build efficient demodulation schemes, as we will see later. The fundamental concept is the following: consider a *real* signal $x[n]$ and its DTFT $X(e^{j\omega})$; consider also the signal processed by the Hilbert filter $y[n] = h[n] * x[n]$. Define:

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

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

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

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

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

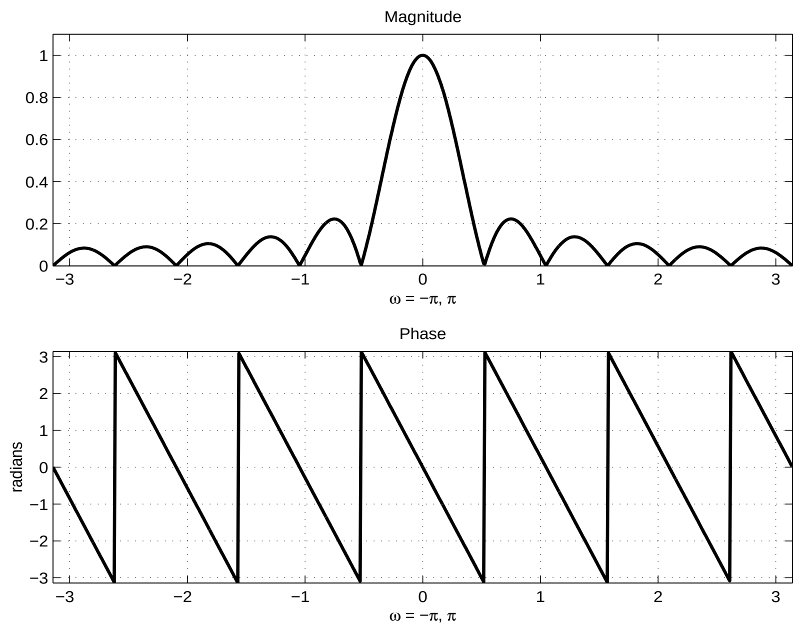


Figure 7.8: Magnitude and phase response of the Moving Average filter for $N = 12$.

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

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

It is therefore easy to see that

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

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

7.7.2 Examples Revisited

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

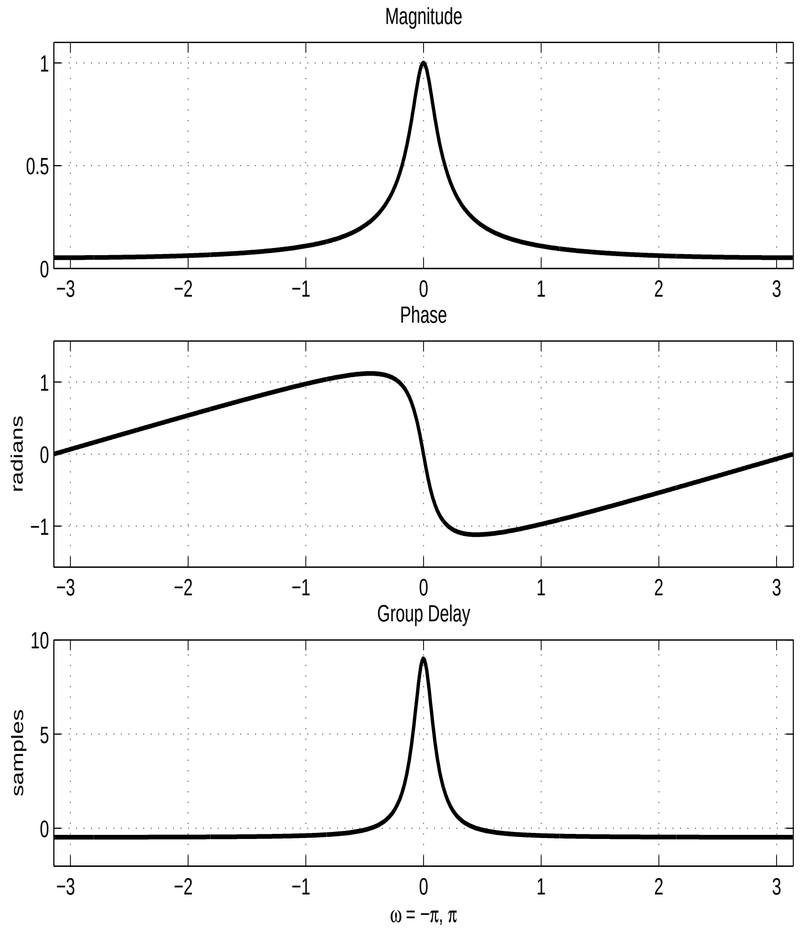


Figure 7.9: Magnitude and phase response of the leaky integrator for $\lambda = 0.9$.

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

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

In the above expression it is immediate to recognize the magnitude and the phase of the frequency response; they are plotted in Figure 7.8. Note that the phase is "wrapped" onto the interval $[-\pi, \pi]$, which is customary plotting practice. The group delay for the filter is the constant $(N-1)/2$, which means that the filters delays its output by $(N-1)/2$ samples (i.e. there is a fractional delay for N even).

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

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

Magnitude and phase are respectively:

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

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

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

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

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

7.8 Filtering and Signal Classes

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

7.8.1 Filtering of Finite-Length Signals

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

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

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

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

7.8.2 Filtering of Periodic Sequences

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

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

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

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

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

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

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

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

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

as we will easily prove later on.

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

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

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

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

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

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

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

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

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

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

with

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

or, in other words

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

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

- the convolution of an N-periodic sequence with a N-tap FIR impulse response is equal to the periodic convolution of two finite-length signals of length N, where the first signal is one period of the input and the second signal is the values of the impulse response over the support
- the periodic convolution can be expressed as a matrix-vector product in which the matrix is circulant

- the DFT of the circular convolution is simply the product of the DFT's of the two finite-length signals; in particular, (7.53) can be used to easily prove the commutativity and distributivity of the circular convolution.

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

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

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

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

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

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

7.9 Summary

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

- Characterization of LTI systems in terms of their impulse response; IIR and FIR impulse responses;

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

7.10 Problems

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

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

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

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

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

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

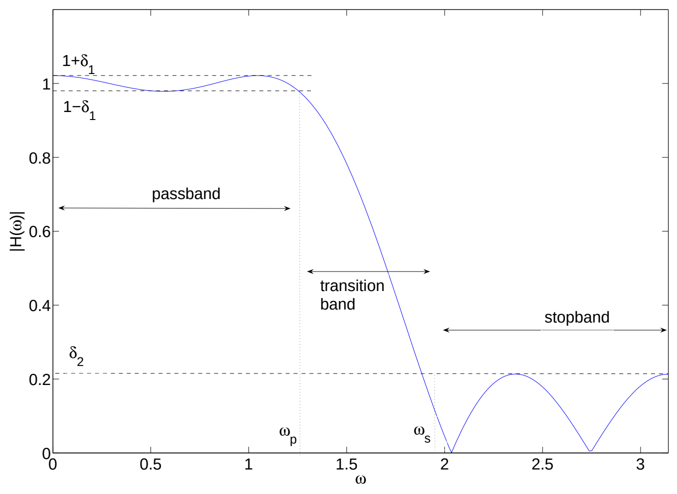


Figure 7.10: Filter Design: We want the impulse response in the passband to be within δ_1 of the desired frequency response 1 and within δ_2 of 0 in the stopband

Let

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

and the error function be

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

We seek the solution to the problem:

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

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

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

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

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

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

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

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

For instance, the delay operator D is indicated as

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

and the differentiation operator is indicated as

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

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

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

1. *Show that the delay operator D is linear.*

2. *Show that the differentiation operator Δ is linear.*

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

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

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

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

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

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

5. *Consider the following matrix:*

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

Which operator do you think it corresponds to?

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

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

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

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

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

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

1) $s[n] = \mathcal{H}\{x[n]\}$
2) $r[n] = \mathcal{R}\{s[n]\}$
3) $w[n] = \mathcal{H}\{r[n]\}$
4) $y[n] = \mathcal{R}\{w[n]\}$

2. *Show that the input-output relation between $x[n]$ and $y[n]$ is an LTI transformation.*
3. *Give the frequency response of the equivalent filter realized by the series of transformations and show that it has zero phase.*

Problem 7.5 *Consider the system shown in Fig. 7.11,*
where $h[n]$ is the impulse response of the LTI system and $H(z)$ exists for

$$0 < r_{\min} < |z| < r_{\max} < \infty.$$

(a) *Can the LTI system with impulse response $h[n]$ be BIBO stable? If so, determine constraints on $r_{\min}$ and $r_{\max}$, otherwise explain why.*

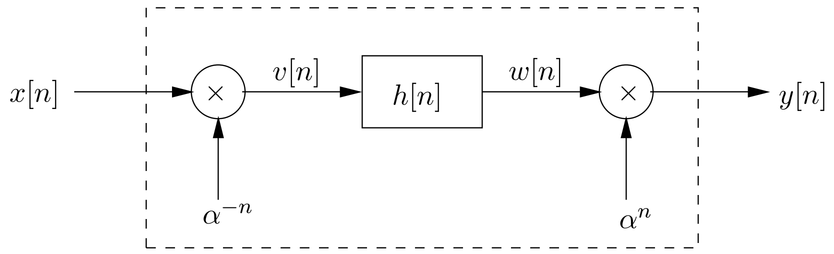


Figure 7.11: System with multiplier.

(b) *Is the system with input $x[n]$ and output $v[n]$ linear? Is it time-invariant?*

(c) *Is the overall system (with input $x[n]$ and $y[n]$) LTI? If so, find the impulse response of the system, otherwise give an example to show your claim.*

(d) *Can the overall system be BIBO stable? If so determine the constraints on α, $r_{\min}$, and $r_{\max}$, otherwise explain why.*

Problem 7.6 *A simple model of multipath communication channel is hown in Fig. 7.12. Assume that $s_c(t)$ is bandlimited such that $S_c(j\Omega) = 0$ for $|\Omega| \geq \pi/T$ and that $x_c(t)$ is sampled with a sampling period T to obtain the sequence $x\,[n] = x_c(nT)$.*

1. *Determine the Fourier transform of $x_c(t)$ and the Fourier transform of $x\,[n]$ in terms of $S_c(j\Omega)$*
2. *We want to simulate the multipath system with a discrete-time system by choosing $H(e^{j\omega})$ in Fig. 7.12, so that the output $r\,[n] = x_c(nT)$ when the input is $s\,[n] = s_c(nT)$. Determine $H(e^{j\omega})$ in terms of T and τ_d.*
3. *Determine the impulse response $h\,[n]$ when (i) $\tau_d = T$ and (ii) $\tau_d = T/2$*

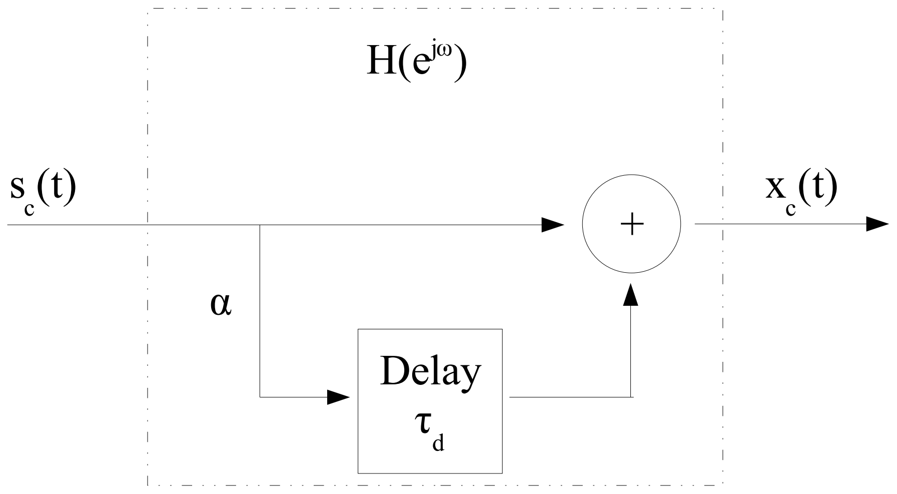
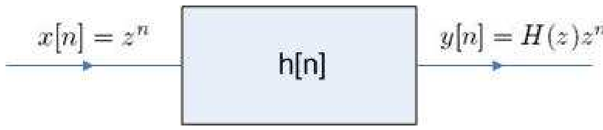


Figure 7.12: A simple multipath communication channel

Chapter 8

The Z-Transform

Consider an input $x[n] = z^n, n \in \mathbb{Z}$ to an LTI system with impulse response $h[n]$. Then



$$y[n] = \sum_{k=-\infty}^{+\infty} h[k]z^{n-k} = z^n \underbrace{\sum_{k=-\infty}^{+\infty} h[k]z^{-k}}_{H(z)} = H(z)z^n. \tag{8.1}$$

Therefore, z^n is an eigenfunction of an LTI system with eigenvalue $H(z)$. As before, a natural question to ask is whether the summation exists. In the case of the summation defined as the Z-transform

$$X(z) = \sum_{n=-\infty}^{+\infty} x[n]z^{-n}, \tag{8.2}$$

the answer can be found by representing $z = re^{j\omega}$, with $r = |z|$. Hence

$$X(z) = X\left(re^{j\omega}\right) = \sum_{n=-\infty}^{+\infty} \left(x[n]r^{-n}\right)e^{-j\omega n} = \sum_{n=-\infty}^{+\infty} \breve{x}[n]e^{-j\omega n},$$

where $\breve{x}[n] = x[n]r^{-1}$. Therefore, we can see that the Z-transform is nothing but the Fourier transform of a scaled sequence, i.e.

$$X(z) = X\left(re^{j\omega}\right) = \mathcal{F}\left\{x[n]r^{-n}\right\}. \tag{8.3}$$

Hence

$$X(z)\Big|_{z=e^{j\omega}} = X\left(e^{j\omega}\right) = \mathcal{F}\left\{x[n]\right\}. \tag{8.4}$$

Now it is clear that for a Z-transform to exist, we would need

$$\sum_{n=-\infty}^{+\infty} \left|x[n]r^{-n}\right| < \infty, \tag{8.5}$$

which for appropriately chosen r would be possible. This already tells us that by choosing r appropriately we can always properly define the Z-transform. The range of values of z for which the Z-transform exists is called the *region-of-convergence*.

Example 8.1 *Let $x[n] = a^n u[n]$. Now*

$$X(z) = \sum_{n=-\infty}^{+\infty} a^n u[n] z^{-n} = \sum_{n=0}^{\infty} a^n z^{-n} = \sum_{n=0}^{\infty} \left(az^{-1}\right)^n.$$

Therefore this needs $\left|az^{-1}\right| < 1$ or $|z| > |a|$, hence

$$X(z) = \frac{1}{1 - az^{-1}} \qquad \textit{for} \quad |z| > |a|. \tag{8.6}$$

Thus the Z-transform is well defined for any $a \in \mathbb{C}$.

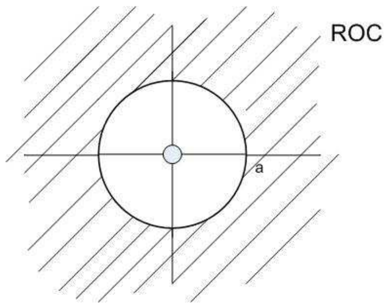


Figure 8.1: Pole-zero plot and region of convergence for Example 8.1.

Example 8.2 *Let* $x[n] = -a^n u[-n-1]$. *Now*

$$X(z) = -\sum_{n=-\infty}^{+\infty} a^n u[-n-1] z^{-n} = \sum_{n=-\infty}^{-1} -a^n z^{-n} = -\sum_{n=1}^{\infty} \left(a^{-1}z\right)^n$$

$$= 1 - \frac{1}{1-a^{-1}z} = \frac{1}{1-az^{-1}} \qquad \textit{for} \quad \left|a^{-1}z\right| < 1 \ \textit{or} \ |z| < |a|. \tag{8.7}$$

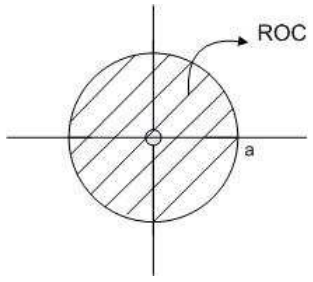


Figure 8.2: Pole-zero plot and region of convergence for Example 8.2.

Example 8.3 *Let* $x[n] = \left(\frac{1}{3}\right)^n u[n] - 2^n u[-n-1]$. *Now*

$$\begin{aligned}
X(z) &= \sum_{n=0}^{\infty} \left(\frac{1}{3} z^{-1}\right)^n - \sum_{n=1}^{\infty} \left(2^{-1}z\right)^n \\
&= \frac{1}{1-\frac{1}{3}z^{-1}} - \left\{1 - \frac{1}{1-2^{-1}z}\right\} \qquad \textit{for} \quad \left|\frac{1}{3}z^{-1}\right| < 1 \ \textit{and} \ \left|2^{-1}z\right| < 1 \\
&= \frac{1}{1-\frac{1}{3}z^{-1}} - \left\{\frac{-2^{-1}z}{1-2^{-1}z}\right\} \qquad \textit{for} \quad |z| > \frac{1}{3} \ \textit{and} \ |z| < 2 \\
&= \frac{1}{1-\frac{1}{3}z^{-1}} - \frac{1}{1-2z^{-1}} \qquad \frac{1}{3} < |z| < 2
\end{aligned}$$

8.1 Region of convergence for the Z-transform

The region of convergence of a Z-transform consists of a ring in the Z-plane centered about the origin. This follows from the fact that the ROC consists of those values of $z = re^{j\omega}$ for which $x[n]r^{-n}$ has a Fourier transform. Since the convergence is *only* dependent on $r = |z|$, it is clear that if a specific value of z is in the ROC, then *all* values of z with the

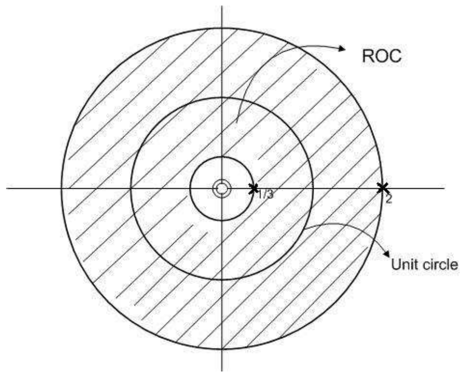


Figure 8.3: Pole-zero plot and region of convergence for Example 8.3.

same magnitude are also in the ROC. Hence this guarantees that the ROC must be in concentric rings. Clearly we have the following properties for the ROC.

Property 1 The ROC is a ring or disk in the Z-plane centered at the origin, i.e. $0 \leq r_R \leq |z| < r_L < \infty$.

Property 2 The DTFT exists if and only if the ROC includes the unit circle.

Property 3 The ROC cannot contain any poles.

Property 4 If $x[n]$ is a *finite-length* sequence, then the ROC is the entire Z-plane with the *possible* exception of $z = 0$ or $z = \infty$.

Property 5 If $x[n]$ is a right-sided sequence, i.e. $x[n] = 0$ for $n < N_1 < \infty$, for some $N_1 \in \mathbb{Z}$, then the ROC extends outwards from the *outermost* finite pole in $X(z)$.

Property 6 If $x[n]$ is a left-sided sequence, i.e. $x[n] = 0$ for $n > N_2 > -\infty$, for some $N_2 \in \mathbb{Z}$, then the ROC extends inwards from the *innermost* finite pole in $X(z)$.

Property 7 A two-sided sequence is an infinite-duration sequence that is neither right-sided nor left-sided. If $x[n]$ is a two-sided sequence, then the ROC will consist of a ring in the Z-plane bounded on the interior and exterior by a pole and does not contain any poles.

Property 8 The ROC must be a connected region.

8.2 The inverse Z-transform

By considering

$$X\left(re^{j\omega}\right) = \mathcal{F}\left\{x[n]r^{-n}\right\},$$

one can obtain

$$x[n]r^{-n} = \mathcal{F}^{-1}\left\{X\left(re^{j\omega}\right)\right\},$$

or

$$x[n]r^{-n} = \frac{1}{2\pi}\int_{-\pi}^{\pi} X\left(re^{j\omega}\right)e^{j\omega n}\,\mathrm{d}\omega,$$

or

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X\left(re^{j\omega}\right)\left(re^{j\omega}\right)^{n}\,\mathrm{d}\omega. \tag{8.8}$$

That is, we can recover $x[n]$ from integrating its Z-transform along a contour $z = re^{j\omega}$ in its ROC, by fixing r and varying ω from $-\pi$ to π. With $z = re^{j\omega}$, for fixed r we see that $\mathrm{d}z = jre^{j\omega}\,\mathrm{d}\omega = jz\,\mathrm{d}\omega$ or $\mathrm{d}\omega = \frac{\mathrm{d}z}{jz}$.
Since the integral in (8.8) is over a 2π interval of ω, it corresponds to a traversal around the circle $|z| = r$. Consequently

$$x[n] = \frac{1}{2\pi j}\oint X(z)z^{n-1}\,\mathrm{d}z, \tag{8.9}$$

where the symbol $\circlearrowleft$ denotes integration around a counter-clockwise closed circular contour centered at the origin and with radius of r. The value of r can be chosen such that $|z| = r$ is in the ROC.

8.3 Partial fraction expansion

$$X(z) = \frac{\sum_{k=0}^{M-1} b_k z^{-k}}{\sum_{k=0}^{N-1} a_k z^{-k}} = \frac{b_0\prod_{k=1}^{M-1}\left(1 - c_k z^{-1}\right)}{a_0\prod_{k=1}^{N-1}\left(1 - d_k z^{-1}\right)}. \tag{8.10}$$

If $M < N$ and all poles are simple, i.e. unique d_k's, we can do a partial fraction expansion to get

$$X(z) = \sum_{k=0}^{N-1}\frac{A_k}{1 - d_k z^{-1}}, \tag{8.11}$$

where

$$A_k = \left(1 - d_k z^{-1}\right) X(z)\Big|_{z=d_k}. \tag{8.12}$$

If $M > N$, we can obtain the right form by first doing a long division of the numerator by the denumerator. This gives gives a general form

$$X(z) = \sum_{r=0}^{M-N} B_r z^{-r} + \sum_{k=0}^{N-1} \frac{A_k}{1 - d_k z^{-1}}, \tag{8.13}$$

where the B_r's are obtained from the long division and the A_k's are as before.

If $X(Z)$ has a pole of order s at d_i and all other poles are simple we get

$$X(z) = \sum_{r=0}^{M-N} B_r z^{-r} + \sum_{k=0, k\neq i}^{N-1} \frac{A_k}{1 - d_k z^{-1}} + \sum_{m=1}^{s} \frac{C_m}{(1 - d_i z^{-1})^m}, \tag{8.14}$$

where

$$C_m = \frac{1}{(s-m)!(-d_i)^{s-m}} \left\{ \frac{\mathrm{d}^{s-m}}{\mathrm{d}w^{s-m}} \left[(1 - d_i w)^s X\left(w^{-1}\right) \right] \right\}_{w=d_i^{-1}} \tag{8.15}$$

and the A_k's and B_r's are obtained as before.

Using (8.14) and the ROC, we can invert the Z-transform using the inspection method, i.e.

$$a^n u[n] \overset{Z}{\longleftrightarrow} \frac{1}{1 - az^{-1}}, \qquad |z| > |a|. \tag{8.16}$$

$$-a^n u[-n-1] \overset{Z}{\longleftrightarrow} \frac{1}{1 - az^{-1}}, \qquad |z| < |a|. \tag{8.17}$$

Example 8.4 (Z-Transform) **(a)** *Let* $W(z) = \frac{1-\frac{1}{2}z^{-1}}{1-z^{-1}+z^{-2}}$ *which is well-defined in its region of convergence* $R_W : |z| > 1$. *Find the inverse z-transform* $w[n]$ *for this function.*

(b) *Find the inverse z-transform for* $X(z) = \ln(1 + z^{-2})$ *with* $|z| > 1$.
Hint: *Differentiate* $X(z)$ *with respect to* z.

(c) *Find the z-transform of* $y[n] = 2^n n u[-n-2]$. *Determine the region of convergence.*
Solution:

(a) In order to find the inverse z-transform of $W(z)$, we have to find its partial fraction expansion. First, we need to find the roots of the polynomial in the denominator.

$$1 - z^{-1} + z^{-2} = 0 \quad or \quad z^2 - z + 1 = 0 \Longrightarrow z = \frac{1 \pm \sqrt{1-4}}{2} = \frac{1 \pm \sqrt{3}i}{2} = e^{\pm\frac{\pi}{3}i}$$

So we have

$$W(z) = \frac{1 - \frac{1}{2}z^{-1}}{(1 - e^{\frac{\pi}{3}i}z^{-1})(1 - e^{-\frac{\pi}{3}i}z^{-1})} = \frac{\alpha}{1 - e^{\frac{\pi}{3}i}z^{-1}} + \frac{\beta}{1 - e^{-\frac{\pi}{3}i}z^{-1}}$$

where

$$\begin{aligned}\alpha &= W(z)(1 - e^{\frac{\pi}{3}i}z^{-1})\Big|_{z=e^{\frac{\pi}{3}i}} = \tfrac{1-\frac{1}{2}e^{-\frac{\pi}{3}i}}{1-e^{-\frac{2\pi}{3}i}} = \tfrac{1}{2}\\ \beta &= W(z)(1 - e^{-\frac{\pi}{3}i}z^{-1})\Big|_{z=e^{-\frac{\pi}{3}i}} = \tfrac{1-\frac{1}{2}e^{\frac{\pi}{3}i}}{1-e^{\frac{2\pi}{3}i}} = \tfrac{1}{2}\end{aligned}$$

Now for the ROC $: |z| > 1$, *we have*

$$\begin{aligned}w[n] &= \alpha(e^{\frac{\pi}{3}i})^n u[n] + \beta(e^{-\frac{\pi}{3}i})^n u[n]\\ &= \frac{e^{\frac{\pi}{3}i} + e^{\frac{-\pi}{3}i}}{2}u[n] = \cos(\frac{\pi}{3}n)u[n]\end{aligned}$$

Remark: *In general, if we have a transfer function* $H(z) = \frac{1-az^{-1}}{1-bz^{-1}+cz^{-2}}$, *and* $a, b, c \in \mathbb{R}$ *where the discriminant of the denominator is negative,* i.e., $\Delta = b^2 - 4c < 0$ *and* $|z| > \sqrt{c}$, *we can use the same scheme to find the inverse z-transform. Let* $p = re^{i\theta}$ *and* p^* *are the roots of the denominator polynomial. It is easy show that* $r^2 = c$ *and* $\theta = \cos^{-1}\frac{b}{2\sqrt{c}}$. *We have*

$$H(z) = \frac{\alpha}{1 - pz^{-1}} + \frac{\beta}{1 - p^*z^{-1}}$$

where

$$\begin{aligned}\alpha &= \frac{a-p}{p^*-p} = \frac{1}{2} + \frac{(a - r\cos\theta)}{2r\sin\theta}i\\ \beta &= \frac{a-p^*}{p-p^*} = \frac{1}{2} + \frac{-(a - r\cos\theta)}{2r\sin\theta}i\end{aligned}$$

and

$$\begin{aligned}
h[n] &= \alpha p^n u[n] + \beta p^{*n} u[n] \\
&= \left[\frac{(re^{i\theta})^n + (re^{-i\theta})^n}{2} + \frac{(r\cos\theta - a)}{r\sin\theta}\frac{(re^{i\theta})^n - (re^{-i\theta})^n}{2i}\right] u[n] \\
&= \left[\cos(\theta n) + \frac{(r\cos\theta - a)}{r\sin\theta}\sin(\theta n)\right] u[n].
\end{aligned}$$

(b) Let $x[n]$ be the inverse z-transform of $X(z)$. Define $y[n] = nx[n]$. According to the properties of the z-transform we have

$$Y(z) = \mathcal{Z}\{y[n]\} = -z\frac{d}{dz}X(z) = -z\frac{-2z^{-3}}{1+z^{-2}} = \frac{2z^{-2}}{1+z^{-2}} = 2 - \frac{2}{1+z^{-2}} = 2 - \frac{1}{1+iz^{-1}} - \frac{1}{1-iz^{-1}}$$

Therefore,

$$\begin{aligned}
y[n] &= \mathcal{Z}^{-1}\{Y(z)\} = 2\delta[n] - i^n u[n] - (-i)^n u[n] \\
&= 2\delta[n] - \begin{cases} -2(-1)^k & \text{if } n = 2k,\ k \in \mathbb{Z}_{\geq 0} \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} -2(-1)^{\frac{n}{2}} u[n-2] & \text{if } n = \text{even} \\ 0 & \text{if } n = \text{odd} \end{cases}
\end{aligned}$$

(c) $y[n] = 2^n n u[-n-2]$, we can define $x[n] = 2^n u[-n-2]$. Then,

$$\begin{aligned}
X(z) &= \sum_{n=-\infty}^{\infty} x[n] z^{-n} = \sum_{n=-\infty}^{\infty} 2^n u[-n-2] z^{-n} \\
&= \sum_{n=-\infty}^{-2} 2^n z^{-n} = \sum_{n=-\infty}^{-2} (\frac{z}{2})^{-n} \\
&= \sum_{m=2}^{\infty} (\frac{z}{2})^m = \frac{(\frac{z}{2})^2}{1 - \frac{z}{2}}
\end{aligned}$$

where the summation is well-defined for $|z| < 2$. Now, we have

$$\begin{aligned}
Y(z) &= -z\frac{d}{dz}X(z) = -z\frac{\frac{1}{2}z - \frac{1}{8}z^2}{1 - z + \frac{1}{4}z^2} \\
&= \frac{1}{2}\frac{1 - 4z^{-1}}{z^{-1} - 4z^{-2} + 4z^{-3}}
\end{aligned}$$

where the ROC is the set of all $z \in \mathbb{C}$ in which $X(z)$ is well-defined, which is $|z| < 2$.

hw4 pb4 2006

Example 8.5 (Region of Convergence) *Consider*

$$H(z) = \frac{1 - 4z^{-1} + 5z^{-2}}{(z+2)(1 + 6z^{-1} + 13z^{-2})}$$

as the transfer function of an LTI system.

(a) *What are the poles and zeros of $H(z)$?*

(b) *How many different regions of convergence can be assigned to $H(z)$? Determine them.*

(c) *For each ROC you have found in part (b), take the inverse z-transform and find the impulse response of the system in time domain.*

(d) *For each impulse response check the causality and stability of the system.*

(e) *In each case determine whether the impulse response is finite-length, right-sided, left-sided, or two-sided?*

Solution:

(a) The poles of the transfer function are the roots of $(z^{-1} + 2)(1 + 6z^{-1} + 13z^{-2}) = 0$, which can be computed as

- $z^{-1} + 2 = 0 \quad \Longrightarrow \quad p_1 = -\frac{1}{2}$
- $1 + 6z^{-1} + 13z^{-2} = 0 \quad$ *or* $\quad z^2 + 6z + 13 = 0 \quad \Longrightarrow \quad p_{2,3} = -3 \pm \sqrt{9 - 13} = -3 \pm 2i$

We can also find the zeros of the transfer function by finding the roots of the nominator polynomial.

$$1 - 4z^{-1} + 5z^{-2} = 0 \quad \textit{or} \quad z^2 - 4z + 5 = 0 \quad \Longrightarrow \quad z_{1,2} = 2 \pm \sqrt{4-5} = 2 \pm i$$

(b) Region of convergence has to be a continuous subset of the point of the $\mathbb{C}$-plane which does not contain any pole. It is also known that $a = re^{i\omega} \in ROC$, then all the points with the same magnitude are also in the ROC. Therefore as it can be seen in Fig. 8.4, there are three possible region of convergence for the transfer function.

$$A = \{z : |z| < \frac{1}{2}\}, \quad B = \{z : \frac{1}{2} < |z| < \sqrt{13}\}, \quad C = \{z : |z| > \sqrt{13}\}. \quad (8.18)$$

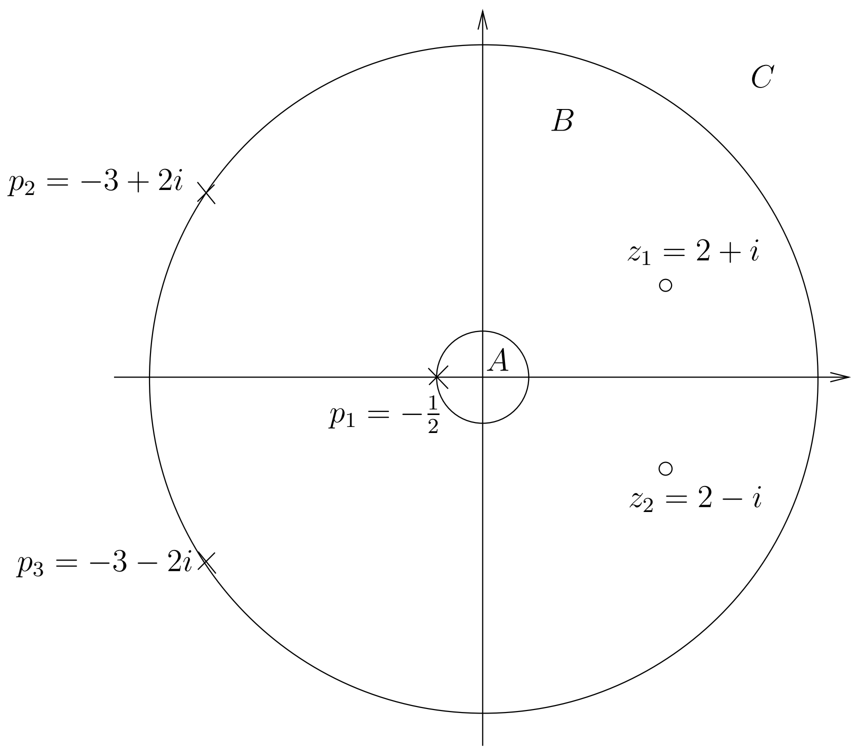


Figure 8.4: The zero-pole plot and three different regions of convergence.

(c) In order to find the inverse z-transform of $H(z)$ we need to find the partial fraction expansion for this.

$$H(z) = \frac{1-4z^{-1}+5z^{-2}}{(z^{-1}+2)(1+6z^{-1}+13z^{-2})} = \frac{\alpha}{z^{-1}+2} + \frac{\beta}{1-(-3+2i)z^{-1}} + \frac{\gamma}{1-(-3-2i)z^{-1}}$$

There two ways for finding α, β and γ and we will consider both of the in the following.

- *First way: solving system of linear equations*

$$\frac{1-4z^{-1}+5z^{-2}}{(z^{-1}+2)(1+6z^{-1}+13z^{-2})} = \frac{\alpha}{z^{-1}+2} + \frac{\beta}{1-(-3+2i)z^{-1}} + \frac{\gamma}{1-(-3-2i)z^{-1}}$$
$$= \frac{(\alpha+2\beta+2\gamma)+(6\alpha+(7+4i)\beta+(7-4i)\gamma)z^{-1}+(13\alpha+(3+2i)\beta+(3-2i)\gamma)z^{-2}}{(z^{-1}+2)(1+6z^{-1}+13z^{-2})}$$

Thus,

$$\begin{cases} \alpha+2\beta+2\gamma = 1 \\ 6\alpha+(7+4i)\beta+(7-4i)\gamma = -4 \\ 13\alpha+(3+2i)\beta+(3-2i)\gamma = 5 \end{cases} \implies \begin{cases} \alpha = 58/82 \\ \beta = (6+95i)/82 \\ \gamma = (6-95i)/82. \end{cases}$$

- *Second way: evaluating the function at its poles*

$$\alpha = H(z)(z^{-1}+2)\Big|_{z=-\frac{1}{2}} = \frac{1-4z^{-1}+5z^{-2}}{1+6z^{-1}+13z^{-2}}\Big|_{z=-\frac{1}{2}} = \frac{29}{41}$$

and

$$\beta = H(z)(1-(-3+2i)z^{-1})\Big|_{z=-3+2i} = \frac{1-4z^{-1}+5z^{-2}}{(z^{-1}+2)(1-(-3-2i)z^{-1})}\Big|_{z=-3+2i} = \frac{6+95i}{82}$$
$$\gamma = H(z)(1-(-3-2i)z^{-1})\Big|_{z=-3-2i} = \frac{1-4z^{-1}+5z^{-2}}{(z^{-1}+2)(1-(-3+2i)z^{-1})}\Big|_{z=-3-2i} = \frac{6-95i}{82}$$

Therefore we have

$$H(z) = \frac{29}{82}\frac{1}{1+\frac{1}{2}z^{-1}} + \frac{6+95i}{82}\frac{1}{1-(-3+2i)z^{-1}} + \frac{6-95i}{82}\frac{1}{1-(-3-2i)z^{-1}}$$

Now we are ready to compute the inverse z-transform for each ROC. In the following computation we use $\phi = \tan^{-1}\frac{-2}{3}$

- *A:*

$$\begin{aligned}
h_A[n] &= -\frac{29}{82}(\frac{1}{2})^n u[-n-1] - \frac{6+95i}{82}(-3+2i)^n u[-n-1] - \frac{6-95i}{82}(-3-2i)^n u[-n-1] \\
&= -\left[\frac{29}{82}\frac{1}{2^n} + \frac{6+95i}{82}(\sqrt{13}e^{i\phi})^n + \frac{6-95i}{82}(\sqrt{13}e^{-i\phi})^n\right] u[-n-1] \\
&= -\left[\frac{29}{82}\frac{1}{2^n} + \frac{6}{41}\sqrt{13}^n \frac{e^{i\phi n}+e^{-i\phi n}}{2} - \frac{95}{41}\sqrt{13}^n \frac{e^{i\phi n}-e^{-i\phi n}}{2i}\right] u[-n-1] \\
&= -\left[\frac{29}{82}\frac{1}{2^n} + \frac{6}{41}\sqrt{13}^n \cos(\phi n) - \frac{95}{41}\sqrt{13}^n \sin(\phi n)\right] u[-n-1]
\end{aligned}$$

- *B:*

$$\begin{aligned}
h_B[n] &= \frac{29}{82}(\frac{1}{2})^n u[n] - \frac{6+95i}{82}(-3+2i)^n u[-n-1] - \frac{6-95i}{82}(-3-2i)^n u[-n-1] \\
&= \frac{29}{82}\frac{1}{2^n}u[n] - \left[\frac{6+95i}{82}(\sqrt{13}e^{i\phi})^n + \frac{6-95i}{82}(\sqrt{13}e^{-i\phi})^n\right] u[-n-1] \\
&= \frac{29}{82}\frac{1}{2^n}u[n] - \left[\frac{6}{41}\sqrt{13}^{-n} \frac{e^{i\phi n}+e^{-i\phi n}}{2} - \frac{95}{41}\sqrt{13}^{-n} \frac{e^{i\phi n}-e^{-i\phi n}}{2i}\right] u[-n-1] \\
&= \frac{29}{82}\frac{1}{2^n}u[n] - \left[\frac{6}{41}\sqrt{13}^n \cos(\phi n) - \frac{95}{41}\sqrt{13}^n \sin(\phi n)\right] u[-n-1]
\end{aligned}$$

- *C:*

$$\begin{aligned}
h_C[n] &= \frac{29}{82}(\frac{1}{2})^n u[n] + \frac{6+95i}{82}(-3+2i)^n u[n] + \frac{6-95i}{82}(-3-2i)^n u[n] \\
&= \left[\frac{29}{82}\frac{1}{2^n} + \frac{6+95i}{82}(\sqrt{13}e^{i\phi})^n + \frac{6-95i}{82}(\sqrt{13}e^{-i\phi})^n\right] u[n] \\
&= \left[\frac{29}{82}\frac{1}{2^n} + \frac{6}{41}\sqrt{13}^n \frac{e^{i\phi n}+e^{-i\phi n}}{2} - \frac{95}{41}\sqrt{13}^n \frac{e^{i\phi n}-e^{-i\phi n}}{2i}\right] u[n] \\
&= \left[\frac{29}{82}\frac{1}{2^n} + \frac{6}{41}\sqrt{13}^n \cos(\phi n) - \frac{95}{41}\sqrt{13}^n \sin(\phi n)\right] u[n]
\end{aligned}$$

(d) *Causality: Since $h_A[n]$ and $h_B[n]$ have some terms in form of $u[-n-1]$, clearly they are not causal. All the terms in $h_C[n]$ are in the form $u[n]$ and it just depend on the future. So, $h_C[n]$ is a causal system.*

Stability: We know that a sequence is stable if and only if the ROC of its z-transform contains the unit circle. According to the region of convergences found in part (b), *the only stable system is $h_B[n]$. We can also check it in time domain:*

$$\begin{aligned}
\sum_{n=-\infty}^{\infty} |h_B[n]| &= \sum_{n=-\infty}^{-1} \left| \frac{6}{41}\sqrt{13}^n \cos(\phi n) - \frac{95}{41}\sqrt{13}^n \sin(\phi n) \right| + \sum_{n=0}^{\infty} \left| \frac{29}{82}\frac{1}{2^n} \right| \\
&\leq \sum_{n=-\infty}^{-1} \left| \frac{6}{41}\sqrt{13}^n \cos(\phi n) \right| + \sum_{n=-\infty}^{-1} \left| \frac{95}{41}\sqrt{13}^n \sin(\phi n) \right| + \sum_{n=0}^{\infty} \left| \frac{29}{82}\frac{1}{2^n} \right| \\
&\leq \frac{6}{41} \sum_{n=-\infty}^{-1} \left| \sqrt{13}^n \right| + \frac{95}{41} \sum_{n=-\infty}^{-1} \left| \sqrt{13}^n \right| + \frac{29}{82} \sum_{n=0}^{\infty} \left| \frac{1}{2^n} \right| \\
&= \frac{101}{41}\frac{1}{\sqrt{13}-1} + \frac{29}{41} < \infty
\end{aligned}$$

We can also check that $\sum_{n=-\infty}^{\infty} |h_A[n]| = \infty$ and $\sum_{n=-\infty}^{\infty} |h_C[n]| = \infty$, but it is a bit long and more complicated.

(e) *We can determine this property either by looking at the impulse responce or by considering the ROC.*

- *Impulse response: All the terms in h_A are in form $u[-n-1]$ and so it is left-sided. $h_B[n]$ has both the terms of form $u[-n-1]$ and $u[n]$ and it is two-sided sequence. Finally, only terms of the form $u[n]$ contribute in $h_C[n]$ and so it is right-sided sequence.*
- *ROC: Region A is inside a circle, so the corresponding sequence should be left-sided. B is a ring and therefore should correspond to a two-sided sequence. Region C is outside of a circle and the inverse transform on this region would be a right-sided sequence.*

8.4 Z-transform properties

1. **Linearity**:

$$a_1x_1[n] + a_2x_2[n] \overset{Z}{\longleftrightarrow} a_1X_1(z) + a_2X_2(z), \qquad \text{ROC contains } R_{x_1} \cap R_{x_2}$$

2. **Time-shifting**:

$$x[n-n_0] \overset{Z}{\longleftrightarrow} z^{-n_0}X(z), \qquad \text{ROC} = R_x$$

(except for the possible addition or deletion of $z = 0$ or $z = \infty$)

3. **Multiplication by exponential sequence**:

$$z_0^n x[n] \xleftrightarrow{Z} X\left(z/z_0\right), \qquad \text{ROC} = |z_0| R_x$$

4. **Differentiation of $X(z)$**:

$$nx[n] \xleftrightarrow{Z} -z\frac{\mathrm{d}X(z)}{\mathrm{d}z}, \qquad \text{ROC} = R_x$$

Example: Inverse of a non-rational Z-transform

$$X(z) = \log\left(1 + az^{-1}\right), \qquad |z| > |a|.$$

$$\frac{\mathrm{d}X(z)}{\mathrm{d}z} = \frac{1}{1+az^{-1}}.\left(-az^{-2}\right)\cdot$$

Hence

$$-z\frac{\mathrm{d}X(z)}{\mathrm{d}z} = \frac{az^{-1}}{1+az^{-1}} \xleftrightarrow{Z^{-1}} a(-a)^{n-1}u[n-1] \qquad |z| > |a|.$$

Hence $nx[n] = a(-a)^{n-1}u[n-1]$, or $x[n] = \frac{a}{n}(-a)^{n-1}u[n-1]$. Hence

$$(-1)^{n-1}\frac{a^n}{n}u[n-1] \xleftrightarrow{Z} \log\left(1 + az^{-1}\right)$$

5. **Conjugation of a complex sequence**:

$$x^*[n] \xleftrightarrow{Z} X^*(z^*) \qquad \text{ROC} = R_x$$

6. **Time-reversal**:

$$x^*[-n] \xleftrightarrow{Z} X^*\left(\frac{1}{z^*}\right), \qquad \text{ROC} = \frac{1}{R_x}$$

7. **Convolution of sequences**:

$$x_1[n] * x_2[n] \xleftrightarrow{Z} X_1(z)X_2(z), \qquad \text{ROC contains } R_{x_1} \cap R_{x_2}$$

8. **Initial value theorem**: If $x[n]$ is zero for $n < 0$, (i.e. if $x[n]$ is causal), then

$$x[0] = \lim_{z\to\infty} X(z)$$

Example 8.6 (Interleaving)

Let $x[n]$ and $y[n]$, $n \in \mathbb{Z}$, be sequences with respective Z-transforms $X(z)$ and $Y(z)$. Now consider a third sequence $u[n]$ that is constructed by interleaving $x[n]$ and $y[n]$. This means that $u[2l] = x[l]$ and $u[2l+1] = y[l]$, $l \in \mathbb{Z}$.

(a) Express the Z-transform of $u[n]$ in terms of $X(z)$ and $Y(z)$.

The ROC of $X(z)$ is $0.64 \le |z| \le 4$ and the ROC of $Y(z)$ is $0.25 \le |z| \le 9$.

(b) What is the ROC of $U(z)$?

Solution:

(a)

$$\begin{aligned} U(z) &= \sum_{n=-\infty}^{\infty} u[n]z^{-n} \\ &= \sum_{l=-\infty}^{\infty} u[2l]z^{-2l} + \sum_{l=-\infty}^{\infty} u[2l+1]z^{-2l-1} \\ &= \sum_{l=-\infty}^{\infty} x[l](z^2)^{-l} + z^{-1} \sum_{l=-\infty}^{\infty} y[l](z^2)^{-l} \\ &= X(z^2) + z^{-1}Y(z^2). \end{aligned}$$

(b) If z_0 is a pole of $X(z)$, then $\pm\sqrt{z_0}$ will be poles of $X(z^2)$. This means we have

$$\begin{aligned} \mathrm{ROC}\{X(z^2)\}: &\quad 0.8 \le |z| \le 2, \\ \mathrm{ROC}\{Y(z^2)\}: &\quad 0.5 \le |z| \le 3. \end{aligned}$$

Multiplying $Y(z^2)$ with z^{-1} might add a pole at 0, which is outside the region of convergence of $Y(z^2)$, so $\mathrm{ROC}\{z^{-1}Y(z^2)\} = \mathrm{ROC}\{Y(z^2)\}$.

For $U(z)$ to exist we need both terms in the sum to exist, so we take the intersection of the ROC of the individual terms. This gives

$$\mathrm{ROC}\{U(z)\}: \quad 0.8 < |z| < 2.$$

8.5 Analysis and characterization of LTI systems using Z-transform

The Z-transform plays a particularly important role in the analysis and representation of discrete-time LTI systems. From the convolution property, for an LTI system with impulse response $h[n] \overset{Z}{\longleftrightarrow} H(z)$ and an input $x[n] \overset{Z}{\longleftrightarrow} X(z)$, we have,

$$y[n] = (h * x)[n] = h[n] * x[n] \overset{Z}{\longleftrightarrow} H(z)X(z), \tag{8.19}$$

where X(z), Y(z) and H(z) are the Z-transforms and the ROC of $Y(z)$ is

$$R_y = R_x \cap R_h. \tag{8.20}$$

$H(z)$ is referred to as the system function or transfer function of the system.

8.5.1 Causality

A causal LTI system has an impulse response that is $h[n] = 0$ for $n < 0$ and therefore is right-sided. Hence the ROC of $H(z)$ is the exterior of a circle on the Z-plane.

Since for a causal LTI system

$$H(z) = \sum_{n=0}^{\infty} h[n]z^{-n} \tag{8.21}$$

does *not* include any positive powers of z, the ROC includes infinity. Hence a discrete LTI system is causal if and only if the ROC of its system function is exterior of a circle and includes infinity.

8.5.2 Stability

We know that an LTI system is BIBO stable if and only if it is absolutely summable. This implies that its DTFT exists, which in turn means that the unit circle is in the ROC of the system function. Therefore, we have the following:

An LTI system is stable if and only if the ROC of its system function $H(z)$ includes the unit circle.

Combining causality and stability, we see that a causal LTI system with rational system function $H(z)$ is stable if and only if all its poles of $H(z)$ lie inside the unit circle.

8.5.3 LTI systems and linear constant-coefficient difference equations

If

$$y[n] = \sum_{k=1}^{N-1} a_k y[n-k] + \sum_{k=0}^{M-1} b_k x[n-k],$$

then

$$Y(z) = \sum_{k=1}^{N-1} a_k z^{-k} Y(z) + \sum_{k=0}^{M-1} b_k z^{-k} X(z).$$

Hence

$$Y(z) \left[1 - \sum_{k=1}^{N-1} a_k z^{-k} \right] = X(z) \sum_{k=0}^{M-1} b_k z^{-k}$$

or

$$\frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M-1} b_k z^{-k}}{1 - \sum_{k=1}^{N-1} a_k z^{-k}} = H(z).$$

Therefore the system function for such systems are easy to write. This form of $H(z) = \frac{Y(z)}{X(z)}$ is also called the transfer function of a linear constant-coefficient difference equation system.

8.6 Problems

Problem 8.1 (DFT and Z-transform) *Let us consider a sequence $x(n)$ having z-transform $X(z)$. If the sequence has finite duration of length N or less, it can be recovered from its N-point DFT. Hence its z-transform is uniquely determined by its N-point DFT. Show that:*

$$X(z) = \sum_{n=0}^{N-1} x(n) z^{-n} = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{j2\pi k/N} z^{-1}}$$

Problem 8.2 *We consider a causal system with transfer function*

$$H(z) = \frac{1 - cz^{-1}}{1 - dz^{-1}}, \qquad |z| > \frac{1}{2}, \tag{8.22}$$

where $c = \frac{1}{2} e^{j(\phi+\pi)}$ and $d = \frac{1}{2} e^{j\phi}$. The variable ϕ is a parameter to the system. In this exercise we will analyze the behavior of this system as a function of ϕ.

We start with an analysis in the Z-domain.

(a) Give the poles and zeros of $H(z)$ as a function of ϕ. Give a pole-zero plot of $H(z)$ for $\phi = 0$ and $\phi = \pi$.

In the Fourier domain we will consider the magnitude response only. In the analysis of LTI systems it is often convenient to consider the log-magnitude response $20 \log_{10} \left|H(e^{j\omega})\right|$.

(b) Let $y[n]$ be the output of the system at input $x[n]$. Express $20 \log_{10} \left|Y(e^{j\omega})\right|$ in terms of $20 \log_{10} \left|X(e^{j\omega})\right|$ and $20 \log_{10} \left|H(e^{j\omega})\right|$.

(c) Show that for a generic r and θ the following holds:

$$20 \log_{10} \left|1 - re^{j\theta}\right| = 10 \log_{10} \left(1 + r^2 - 2r\cos(\theta)\right). \tag{8.23}$$

(d) Derive the general expression for $20 \log_{10} \left|H(e^{j\omega})\right|$ for $H(z)$ given in (8.22). How does the system behave at $\omega = \phi$ and $\omega = \phi + \pi$?

(e) Let $\phi = 0$. In MATLAB, create a plot of $20 \log_{10} \left|H(e^{j\omega}\right|$. Is this system all-pass, low-pass, high-pass or band-pass?

(f) Let $\phi = \pi$. In MATLAB, create a plot of $20 \log_{10} \left|H(e^{j\omega}\right|$. Is this system all-pass, low-pass, high-pass or band-pass?

Finally an analysis in the time-domain.

(g) For arbitrary ϕ, determine the impulse repsonse of the system.

(h) Let $\phi = 0$. Based on the impulse response only, what can you say about the behaviour of the system at a unit step input? (You can think of the unit step function as a low frequency signal)

(i) Let $\phi = \pi$. Based on the impulse response only, what can you say about the behaviour of the system at a unit step input.

Problem 8.3 (Minimum Phase System) *A system is called* minimum phase *if the system and its inverse are causal and stable. A rational transfer function will be minimum phase if and only if all its zeros and poles are inside the unit circle. For example $H_1(z) = \frac{1-az^{-1}}{1-bz^{-1}}$, where $a = \frac{1}{3}e^{\frac{3\pi}{4}i}$ and $b = \frac{1}{2}e^{\frac{\pi}{3}i}$, is a minimum phase system.*

However, there exist other transfer functions such as

$$H_2(z) = \frac{z^{-1}-a^*}{1-bz^{-1}}, \quad H_2(z) = \frac{1-az^{-1}}{z^{-1}-b^*}, \quad \text{and } H_3(z) = \frac{z^{-1}-a^*}{z^{-1}-b^*}$$

which have the same magnitude as $H_1(z)$ on the unit circle. In this problem we investigate two of important properties of minimum phase systems which identify .the minimum phase system among all systems with the same magnitude.

(a) [MINIMUM GROUP-DELAY] *Let $H(z) = \frac{(1-cz^{-1})}{(1-dz^{-1})}$ where $c = |c|e^{i\theta}$, $d = |d|e^{i\phi}$, $|c| > 1$ and $|d| < 1$. Rewrite $H(z) = H_{\min}(z)H_{ap}(z)$ where*

- *All zeros and poles of $H_{\min}(z)$ are inside the unit circle,*
- *$H_{ap}(z)$ be a causal all-pass filter.*

(i) *Replace z in $H_{ap}(z)$ by $e^{i\omega}$ and find the group-delay expression for $H_{ap}(z)$ in terms of $|c|, |d|, \theta, \phi, \omega$ and show that it is positive for any ω.*

(ii) *Write the group-delay expression for $H(z)$ in terms of the group-delay of $H_{\min}(z)$ and $H_{ap}(z)$. Compare the group-delay of $H(z)$ and $H_{\min}(z)$.*

(b) [MINIMUM ENERGY DELAY] *Let $H_{\min}(z)$ be a minimum phase system which has a zero at α. We can write $H_{\min}(z) = Q(z)(1 - \alpha z^{-1})$, where $Q(z)$ is also minimum phase. Now consider another system with transfer function $H(z)$ such that $|H(z)| = |H_{\min}(z)|$ and $H(z)$ has a zero at $1/\alpha^*$ instead of α.*

(i) *Compare $\sum_{n=0}^{\infty} |h_{\min}[n]|^2$ to $\sum_{n=0}^{\infty} |h[n]|^2$.*

(ii) *Express $H(z)$ in terms of $Q(z)$.*

(iii) *Express $h_{\min}[n]$ and $h[n]$ in terms of $q[n]$ and α.*

(iv) *Write the expression for*

$$\sum_{n=0}^{m} |h_{\min}[n]|^2 - \sum_{n=0}^{m} |h[n]|^2$$

and simplify it to find an expression in terms of $q[m]$ and α.

(v) *Compare $\sum_{n=0}^{m} |h_{\min}[n]|^2$ to $\sum_{n=0}^{m} |h[n]|^2$ and conclude that the minimum phase system has the minimum energy-delay among all the systems with the same magnitude response.*

Problem 8.4 *For this exercise, you first need to download the file* `santa_corrupt.wav` *from the course website. Load the file into matlab $[data, fs] = wavread('santa\_corrupt.wav')$. Listen to the file using $soundsc(data, fs)$.*

(a) Using matlab, provide a plot of the amplitude of the DFT of the sound sequence. After having listened to the sound sequence and looked at the frequency response, you should realize that the sequence was corrupted by high frequency noise!

(b) It is now up to you to design a 101 tap filter to de-noise the sequence. You are free to use either the windowing method or the Parks-McLellan algorithm. At the end, you should provide us a plot of the frequency response of the de-noised sequence, as well as a plot of the frequency and time response of the de-noising filter you designed. Further, provide a ***short*** *explanation of the procedure you followed. (Hint: do not try to recover the original sequence exactly, rather try to get rid of high frequencies).*

Problem 8.5 *Let $x[n]$ be a discrete-time sequence and $X(z)$ its corresponding z-transform with appropriate ROC.*

1. *Prove that the following relation holds:*

$$nx[n] \overset{Z}{\longleftrightarrow} -z\frac{d}{dz}X(z).$$

2. *Using (a), show that*

$$(n+1)\alpha^n u[n] \overset{Z}{\longleftrightarrow} \frac{1}{(1-\alpha z^{-1})^2}, \qquad |z| > |\alpha|.$$

3. *Suppose that the above expression corresponds to the impulse response of an LTI system. What can you say about the causality of such a system? About its stability?*

4. *Let $\alpha = 0.8$, what is the spectral behavior of the corresponding filter? What if $\alpha = -0.8$?*

Problem 8.6 *Consider a causal discrete system represented by the following difference equation:*

$$y[n] - 3.25y[n-1] + 0.75y[n-2] = x[n-1] + 3x[n-2].$$

1. *Compute the transfer function and check the stability of this system both analytically and graphically.*

2. *If the input signal is $x[n] = \delta[n] - 3\delta[n-1]$, compute the z-transform of the output signal and discuss the stability.*

3. *Take an arbitrary input signal that does not cancel the unstable pole of the transfer function and repeat b).*

Problem 8.7 *Consider two two-sided sequences $h[n]$ and $g[n]$ and consider a third sequence $x[n]$ which is built by interleaving the values of $h[n]$ and $g[n]$:*

$$x[n] = \ldots, h[-3], g[-3], h[-2], g[-2], h[-1], g[-1], h[0], g[0], h[1], g[1], h[2], g[2], h[3], g[3], \ldots$$

with $x[0] = h[0]$.

(a) *Express the z-transform of $x[n]$ in terms of the z-transforms of $h[n]$ and $g[n]$.*

(b) *Assume that the ROC of $H(z)$ is $0.64 < |z| < 4$ and that the ROC of $G(z)$ is $0.25 < |z| < 9$. What is the ROC of $X(z)$?*

Chapter 9

Filters and Filter Design

We have already shown that, from a mathematical point of view, a linear time-invariant system is completely characterized by its impulse response. While any absolutely summable impulse response defines a stable LTI system, the computation of an output sample for the system might require an infinite number of operations; this is for instance the case of the ideal filters in section 7.7.1. In practice, of course, we are interest in *realizable* systems, i.e. systems which can be implemented with a finite number of operations. It is immediate to see that any FIR filter belongs to this category, but we have also seen in 7.4.2 that there exist IIR systems (whose impulse response is an infinite sequence) which can still be implemented with a finite amount of computation and storage. It turns out that the most general class of such realizable discrete-time systems is described by *constant-coefficient difference equations*. The general concept of filter design usually starts with a given set of specifications, which in all but a handful of cases are expressed in terms of a desired frequency response; the design problem is solved by finding the appropriate coefficients for a suitable difference equation which implements the filter. We will show that realizable filters possess a transfer function which is a ratio of polynomials in the complex variable z^{-1}; as a consequence, filter design can be cast in terms of a polynomial optimization procedure for a given error measure. Finally, the structure of difference equation defines an explicit operational procedure for computing the filter's output values; by arranging the terms of the equation in different ways, we can arrive at different algorithmic structures for the implementation of digital filters.

9.1 Realizable Filters: General Properties

We will start our discussion with a general analysis of constant-coefficient difference equations and associated transfer functions, from which the essential properties of linear filters (including stability) are easily derived.

9.1.1 Difference Equations & Initial Conditions

In its most general form, a constant-coefficient difference equation defines a relationship between an input signal $x[n]$ and an output signal $y[n]$ as

$$\sum_{k=0}^{N-1} a_k y[n-k] = \sum_{k=0}^{M-1} b_k x[n-k]; \tag{9.1}$$

in the rest of these notes we will restrict ourselves to the case in which all the coefficients a_k and b_k are real. Usually, it is $a_0 = 1$, so that the above equation can easily be rearranged as:

$$y[n] = \sum_{k=0}^{M-1} b_k x[n-k] - \sum_{k=1}^{N-1} a_k y[n-k]; \tag{9.2}$$

Clearly, the above relation defines each output sample $y[n]$ as a linear combination of past and present input values and past output values. However, it is easy to see that if $a_{N-1} \neq 0$ we can for instance rearrange (9.1) as

$$y[n-N+1] = \sum_{k=0}^{M-1} b'_k x[n-k] - \sum_{k=0}^{N-2} a'_k y[n-k];$$

where $a'_k = a_k/a_{N-1}$ and $b'_k = b_k/a_{N-1}$. With the change of variable $m = n-N+1$, this becomes

$$y[m] = \sum_{k=N-M}^{N-1} b'_k x[m+k] - \sum_{k=1}^{N-1} a'_k y[m+k]; \tag{9.3}$$

which shows that the difference equation can be computed in the other way as well, namely by expressing $y[m]$ as a linear combination of *future* values of input and output. It is rather intuitive that the first approach defines a causal behavior, while the second approach is anticausal. The main point is that, contrary to the differential equations used in the characterization of continuous-time systems, difference equation can be used directly to translate the transformation operated by the system into an *explicit algorithmic form.* To see this, and to gain a lot of insight on the properties of difference equations, it may be useful to consider a possible implementation of the system in (9.2), here written in C:

```
extern double a[N];     // The a’s coefficients
extern double b[M];     // The b’s coefficients

static double x[M];     // Delay line for x
static double y[N];     // Delay line for y

double GetOutput(double input)
{
  int k;
  for (k = N-1; k > 0; k--)     // Shift delay line for x
    x[k] = x[k-1];
  x[0] = input;                 // new input value x[n]

  for (k = M-1; k > 0; k--)     // Shift delay line for x
    y[k] = y[k-1];

  double y = 0;
  for (k = 0; k < M; k++)
    y += b[k] * x[k];
  for (k = 1; k < M; k++)
    y -= a[k] * y[k];           // New value for y[n];
  y[0] = y;                     // Store in delay line

  return y;
}
```

It is immediate to verify that

1. the above routine realizes the difference equation in (9.2)
2. the storage required is $(N + M)$
3. each output sample is obtained via $(N + M - 1)$ multiplications and additions
4. the transformation is causal

If we try to compile and run the above routine, however, we immediately run into an *initialization* problem: the first time (actually, the first $\max(N, M - 1)$ times) we call the function, the delay lines which hold past values of $x[n]$ and $y[n]$ will contain undefined values. Most likely, the compiler will notice this condition and will print a warning message signaling that the static arrays have not been properly initialized. We are back to the problem of setting the initial conditions of the system. *The choice which guarantees linearity and time invariance is called the* zero initial conditions *and corresponds to setting the delay lines to zero before starting the algorithm.* This choice implies that the system response to the zero sequence is the zero sequence and, in this way, linearity and time invariance can be proven as in section 7.4.2.

9.1.2 Transfer Functions

The best way to analyze the properties of the system implemented by (9.1) is to apply the z-transform to both sides of the equation; we obtain:

$$Y(z) = X(z) \sum_{n=0}^{M-1} b_n z^{-n} - Y(z) \sum_{n=1}^{N-1} a_n z^{-n} \tag{9.4}$$

The above equation can be rearranged as

$$Y(z) = H(z)X(z) \tag{9.5}$$

where $H(z)$ is the *transfer function* of the system and is given by

$$H(z) = \frac{b_0 + b_1 z^{-1} + \ldots + b_{M-1} z^{-(M-1)}}{1 + a_1 z^{-1} + \ldots + a_{N-1} z^{-(N-1)}} \tag{9.6}$$

Such a transfer function is called a *rational transfer function* and is the ratio of two polynomials in z^{-1}; note that the degree of the polynomial at the numerator is $M-1$ and that of the denumerator is $N-1$. As such, it can be written in factored form as

$$H(z) = b_0 \frac{\prod_{n=1}^{M-1} (1 - z_n z^{-1})}{\prod_{n=1}^{N-1} (1 - p_n z^{-1})} \tag{9.7}$$

where the z_n are called the *zeros* of the filter and p_n are called the *poles*; the zeros are the roots of the numerator of the transfer function while the poles are the roots of the denominator. Clearly, if $z_i = p_k$ for some i and k (i.e. if a pole and a zero coincide) the corresponding first-order factors will cancel each other out and the degrees of numerator and denominator are both decreased by one. In the following, we will assume that such factors have been already removed and that the numerator and denominator polynomials of a given rational transfer function are coprime.

Recall that the roots of a polynomial with real-valued coefficients are either real or they occur in complex-conjugate pairs; a pair of complex-conjugate roots translates to a second-order term with real coefficients:

$$(1 - az^{-1})(1 - a^* z^{-1}) = 1 - 2\mathrm{Re}\{a\} z^{-1} + |a|^2 z^{-2} \tag{9.8}$$

As a consequence, the transfer function can be factored in the product of first- and second-order terms in which the coefficients are all strictly real; namely:

$$H(z) = b_0 \frac{\prod_{n=1}^{M_r}(1 - z_n z^{-1}) \prod_{n=1}^{M_c}(1 - 2\mathrm{Re}\{z_n\}z^{-1} + |z_n|^2 z^{-2})}{\prod_{n=1}^{N_r}(1 - p_n z^{-1}) \prod_{n=1}^{N_c}(1 - 2\mathrm{Re}\{p_n\}z^{-1} + |p_n|^2 z^{-2})} \tag{9.9}$$

where M_r is the number of real zeros, M_c is the number of complex-conjugate zeros and $M_r + 2M_c = M$ (with the same holding for the poles representation, i.e. $N_r + 2N_c = N$).

9.1.3 Stability Analysis

If we consider equation (9.5), $Y(z) = H(z)X(z)$, it is clear that $H(z)$ is the z-transform of the filter's impulse response. Therefore, the BIBO stability of a digital filter as described by (9.1) is easily inferred from the properties of the z-transform: *for a filter to be stable the ROC of $H(z)$ must contain the unit circle* because this guarantees the absolute summability of $h[n]$. Since the ROC is determined by the location of the poles of the transform, the above condition translates to the following:

- **For causal filters** the ROC of $H(z)$ is a region on the complex plane extending *outwards*, and therefore a necessary and sufficient condition for stability is that *all the poles of $H(z)$ are inside the unit circle.*

- **For anticausal filters** the ROC of $H(z)$ is a region on the complex plane extending *inwards*, and therefore a necessary and sufficient condition for stability is that *all the poles of $H(z)$ are outside the unit circle.*

9.2 Filter Design - Introduction

As we have seen, a realizable filter is completely described by its rational transfer function; designing a filter corresponds to determining the coefficients of the transfer function with respect to the desired filter characteristics. For an FIR filter of length M, there are M coefficients that have to be determined, and they correspond directly to the filter's impulse response. In a similar way, an IIR filter with a numerator of degree $M-1$ and a denominator of degree $N-1$, has $M+N-1$ coefficients to determine (since we always assume $a_0 = 1$). The main questions are the following:

- How do we choose the filter's coefficients in order to obtain the desired filtering characteristics?

- What are the criteria to measure the quality of the obtained filter?
- What is the best algorithmic structure (software or hardware) to implement a given digital filter?

The first two questions are optimization problems in a parameter space of dimension $M + N - 1$ with a given optimality criterion (for instance, minimum square error or minimax). The last question is an algorithmic design problem subject to constraints of computational speed, storage and precision, which we will analyze in detail at the end of the chapter.

9.2.1 FIR versus IIR

Filter design has a long and noble history in the analog domain: a linear electronic network can be described in terms of a differential equation linking, for instance, the voltage as a function of time at the input of the network to the voltage at the output. The arrangement of the capacitors, inductances and resistors in the network determine the form of the differential equation, while their values determine its coefficients. A fundamental difference between an analog filter and a digital filter is that the transformation from input to output is almost always considered *instantaneous* (i.e. the propagation effects along the circuit are neglected). In digital filters, on the other hand, the delay is always explicit and is actually the fundamental building block in a processing system. Because of the physical properties of capacitors, which are ubiquitous in analog filters, the transfer function of an analog filter (expressed in terms of its Laplace transform) is "similar" to the transfer function of an IIR filter, in the sense that it contains both poles and zeros. In a sense, IIR filters can be considered the discrete-time counterpart of classic analog filters. FIR filters, on the other hand, are the flagship of digital signal processing; while one could conceive of an analog equivalent to an FIR, its realization would require the use of analog delay lines, which are costly and impractical components to manufacture. In a digital signal processing scenario, on the other hand, the designer can freely choose between two lines of attack with respect to a filtering problem, IIR or FIR, and therefore it is important to highlight advantages and disadvantages of each.

FIR Filters. The main advantages of FIR filters can be summarized as follows:

✓ Unconditional stability;

✓ Precise control of the phase response and, in particular, exact linear phase;

✓ Optimal algorithmic design procedures;

✓ Robustness with respect to finite numerical precision hardware while their disadvantages are mainly:

× Longer input-output delay

× Higher computational cost with respect to IIR solutions

IIR Filters. IIR filters are often an afterthought in the context of digital signal processing in the sense that they are designed by mimicking established design procedures in the analog domain; their appeal lies mostly in their compact formulation: for a given computational cost, i.e for a given number of operations per input sample, they can offer a much better magnitude response than an equivalent FIR filter. Furthermore, there are a few fundamental processing tasks (such as DC removal, as we will see later) which are the natural domain of IIR filters. The drawbacks of IIR filter, however, mirror in the negative the advantages of FIR's. The main advantages of FIR filters can be summarized as follows:

✓ Lower computational cost with respect to an FIR with similar behavior

✓ Shorter input-output delay

✓ Compact representation

while their disadvantages are mainly:

× Stability is not guaranteed;

× Phase response is difficult to control;

× Design is complex in the general case;

× Sensitive to numerical precision.

For these reasons, in these notes and in the course we will concentrate mostly on the FIR design problem and we will tackle the design of IIR filters mostly for some specific processing tasks which are often encountered in the practice.

9.2.2 Filter Specifications & Tradeoffs

A set of filter specifications represents the guidelines for an application-oriented filter design. Real-world filters are designed with a variety of practical requirements in mind, most of which are conflicting. One such requirement, for instance, is to obtain a low "computational price" for the filtering operation; this cost is obviously proportional to the number of coefficients in the filters, as we have seen in the introduction, but it also depends heavily on

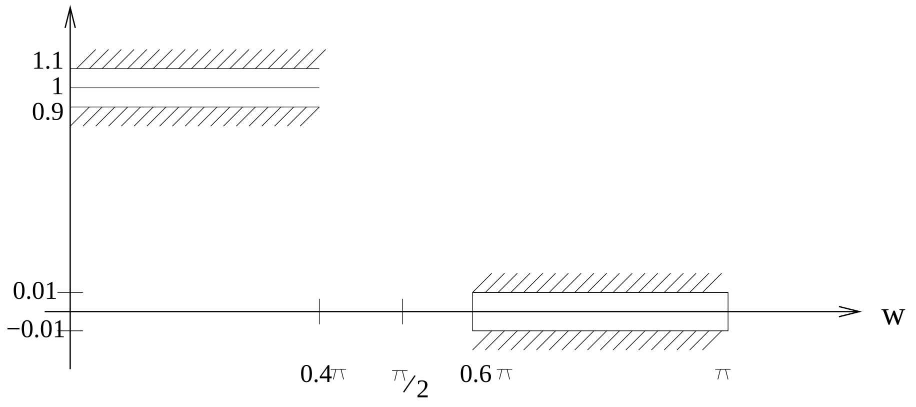


Figure 9.1: Filter specifications

the underlying hardware architecture. The tradeoffs between disparate requirements such as cost, precision or numerical stability are very subtle and not altogether obvious; the art of the digital filter designer, although probably less dazzling than the art of the *analog* filter designer, is to determine the best design strategy for a given practical problem.

In this chapter we won't certainly consider all the variables which enter a filter design scenario; the starting point, however, is almost always a set of *filter specifications* in the frequency domain. These are best illustrated by example: suppose our goal is to design a half-band lowpass filter, i.e. a lowpass filter with cutoff frequency $\pi/2$. The practical contraints to consider are the following:

- **Filter Type.** Whether we design an FIR or an IIR, is dependent on a variety of factors which are specific to the practical application. This decision, however, is the first to be made since the specifications are dependent on it. A closely related design choice determines the maximum filter order which we can afford.

- **Transition band.** We should know by now (and we shall see again shortly) that we cannot obtain an arbitrarily sharp transition band in a realizable filter. Therefore, we must be willing to allow for the existence of a *transition band* from passband to stopband; suppose we estimate that its width can be up to 20% of the total bandwidth: since the cutoff is supposed to be at 0.5π, the transition band will thus extend from 0.4π to 0.6π.

- **Tolerances.** Similarly, we cannot simply impose a strict value of 1 for the passband and a value of 0 for the stopband but we must allow for *tolerances*; after examining the problem we are designing the filter for, suppose we decide we can afford a 10% error in the passband and a 1% error in the stopband. Note that, in filter design parlance, the attenuation in the stopband is frequently expressed on a decibel logarithmic scale:

$$A_{\mathrm{dB}} = 20\log_{10}(\delta_s),$$

where δ_s is the maximum tolerated error in the stopband. In the previous example, we are thus requiring an attenuation of 40 dB.

These specifications can be represented graphically as in Figure 9.1; the filter design problem consists now in finding the minimum size FIR or IIR filter which fulfills the required specifications.

9.3 FIR Filter Design

In this section we will explore two fundamental strategies for FIR filter design, the window method and the minimax (or Parks-McClellan) method. Both methods seek to minimize the error between a desired (and often ideal) filter transfer function and the transfer function of the designed filter; they differ in the error measure which is used in the minimization. The window method is completely straightforward and it is often used for quick designs. The minimax method, on the other hand, is the procedure of choice for accurate, optimal filters. Both methods will be illustrated with respect to the design of a lowpass filter.

9.3.1 FIR Filter Design by Windowing

Consider the problem of designing a lowpass filter with cutoff frequency w_c: with no further specifications (i.e. with no declared tolerance or approximations) the only solution is simply the inverse Fourier transform of the desired transfer function. The resulting impulse response $h[n]$ is of course the usual sinc function which we saw in section 7.7.1:

$$\begin{aligned} h[n] &= \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})e^{j\omega n}d\omega \\ &= \frac{1}{2\pi}\int_{-w_c}^{w_c} e^{j\omega n}d\omega \\ &= \frac{1}{2\pi jn}\left[e^{j\omega_c n} - e^{-j\omega_c n}\right] \\ &= \frac{\sin(\omega_c n)}{\pi n} \\ &= \frac{\omega_c}{\pi}\,\mathrm{sinc}(\frac{\omega_c}{\pi}n) \end{aligned}$$

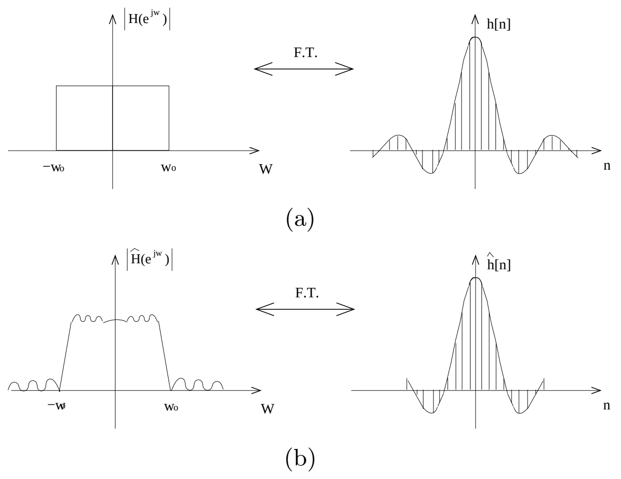


Figure 9.2: a) Ideal filter. b) Approximated filter.

The resulting filter, however, is an ideal filter and it cannot be represented by a rational transfer function with a finite number of coefficients. Our only hope is that, by somehow relaxing the design constraints, we can arrive at a realizable approximation $\hat{H}(e^{j\omega})$ of the ideal filter $H(e^{j\omega})$. Let us start by considering an approximated FIR filter obtained by simply truncating the original impulse response:

$$\hat{h}[n] = \begin{cases} h[n] & -N \leq n \leq N \\ 0 & \text{otherwise} \end{cases} \tag{9.10}$$

This is a $(2N+1)$-tap FIR; Figure 9.2(a) shows the ideal filter and Figure 9.2(b) shows the approximated filter, with their corresponding Fourier transforms (the reasons for their shapes will be clear later). The approximation we just created was obtained in a sort of "intuitive" way; we will now show, however, that it actually satisfies a very precise approximation criterion, namely the minimization of the mean square error (MSE) between the original and approximated filters. Let's denote by E_2 such error, that is:

$$E_2 = \int_{-\pi}^{\pi} |H(e^{j\omega}) - \hat{H}(e^{j\omega})|^2 dw.$$

Therefore one optimization problem for a FIR filter of length M (where $M = 2N+1$) could be

$$\begin{array}{ll} \text{minimize} & \left\| H\left(e^{j\omega}\right) - \hat{H}\left(e^{j\omega}\right) \right\|_2^2 \\ \quad\text{s.t.} & \\ & \hat{H}\left(e^{j\omega}\right) = \sum_{n=-N}^{N} \hat{h}[n] e^{-j\omega n}. \end{array} \tag{9.11}$$

We can apply Parseval's theorem (see (5.46)) to obtain the equivalent expression in the discrete-time domain:

$$E_2 = 2\pi \sum_{n\in\mathcal{Z}} |h[n] - \hat{h}[n]|^2$$

If we now recall that $\hat{h}[n] = 0$ for $|n| > N$, we have

$$E_2 = 2\pi \left[\sum_{n=-N}^{N} |h[n] - \hat{h}[n]|^2 + \sum_{n=N+1}^{\infty} |h[n]|^2 + \sum_{n=-\infty}^{-N-1} |h[n]|^2 \right].$$

Obviously the last two terms are nonnegative and independent of $\hat{h}[n]$. Therefore, the minimization of E_2 with respect to $\hat{h}[n]$ is equivalent to the minimization of the first term only, and this is easily obtained by letting

$$\hat{h}[n] = h[n] \quad \text{for } n = -N, \ldots, N$$

If we look at what we are doing, it is apparent that we are trying to approximate a Fourier sum with a finite number of terms (we are actually working backwards, since the Fourier coefficients are the discrete-time filter values and the approximated function is a frequency response). Since the approximated function is discontinuous in ω_0 we will incur the Gibbs phenomenon, which explains the non-negligible ripples around the transition point. Also, because of the finite number of terms, the transition from passband to stopband will be less sharp; this is clearly apparent in Figure 9.2.

The Window Concept. Another way to look at the resulting approximation is to express $\hat{h}[n]$ as:

$$h[n] = h[n]w[n], \tag{9.12}$$

where $w[n]$ is a rectangular window of length $(2N + 1)$ taps centered at index zero:

$$w[n] = \text{rect}(n/N) = \begin{cases} 1 & -N \leq n \leq N \\ 0 & \text{otherwise} \end{cases}. \tag{9.13}$$

We know from the modulation theorem in (7.24) that the Fourier transform of (9.12) is

$$\hat{H}(e^{j\omega}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(e^{j\omega}) W(e^{j(\omega-\theta})d\theta$$

where $W(e^{j(\omega)}$ is the Fourier transform of the window $w[n]$:

$$W(e^{j\omega}) = \sum_{n=-N}^{N} e^{-j\omega n} = \frac{\sin\left(\omega\left(N + \frac{1}{2}\right)\right)}{\sin(\omega/2)} \tag{9.14}$$

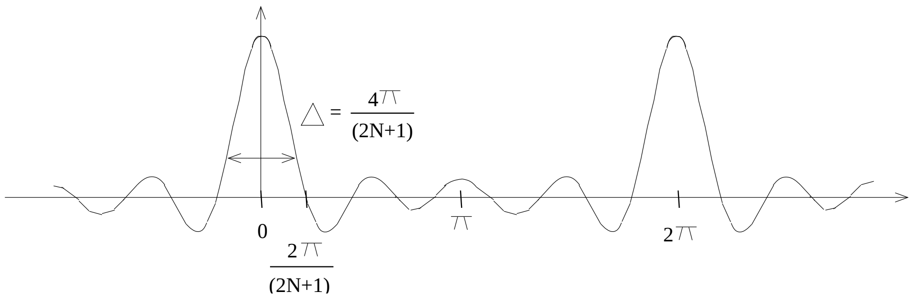


Figure 9.3: Approximated filter Fourier transform

An example of $W(e^{j\omega})$ for $N = 6$ is shown in Figure 9.3. By visual inspection, we can determine the following facts:

- The first zero crossing of $W(e^{j\omega})$ occurs at $\omega = 2\pi/(2N + 1)$
- The width of the main lobe of the magnitude response is $\Delta = 4\pi(2N + 1)$
- The magnitude response shows the presence of *sidelobes*, an oscillatory effect around the main lobe.

Therefore, the windowing operation on the ideal impulse response, i.e. the circular convolution of the ideal frequency response with $W(e^{jw})$ produces two main effects:

1. The sharp transition from passband to stopband is smoothed by the convolution with the main lobe of width Δ.
2. Ripples appear both in the stopband and the passband due to the convolution with the sidelobes.

These effects are shown in Figure 9.4. It appears that the sharpness of the transition band and the size of the ripples are dependent on the shape of the window's Fourier transform; indeed, by carefully designing the shape of the windowing sequence we can

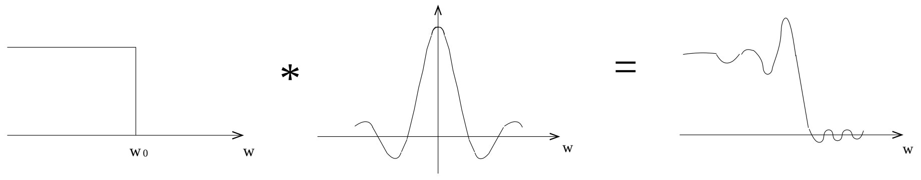


Figure 9.4: Convolution of the ideal filter with the window.

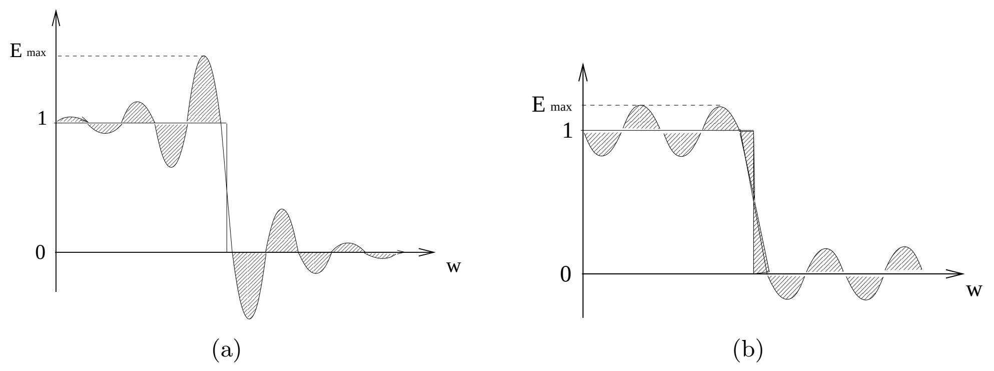


Figure 9.5: a) Minimum square error solution. b) Minimax solution.

trade mainlobe width for sidelobe amplitude and obtain a more controlled behavior in the frequency response of the approximation filter.

Although the MSE minimization procedure can lead to perfectly usable filters, its drawback is the lack of control of the maximum error in the frequency response with respect to the ideal filter. A more suitable approximation criterion may therefore be the *minimax* criterion, where we aim to explicitly minimize the *maximum* approximation error over the entire frequency support; this will be explained in detail in the next section. We can already say, however, that while the minimum square error is an integral criterion, the minimax is a pointwise criterion; or, more mathematically, that the MSE and the minimax are respectively $L_2([-\pi,\pi])$- and $L_\infty([-\pi,\pi])$-norm minimizations for the error function $E(\omega) = \hat{H}(e^{j\omega}) - H(e^{j\omega})$. Figure 9.5 illustrates the typical result of applying both criteria to the ideal lowpass problem. As it can be seen, the minimum square and minimax solutions are very different.

More General Windows. We have seen in the previous discussion the fundamental principles of FIR filter design by windowing. The starting point is the desired frequency response $H(e^{j\omega})$, from which the ideal impulse response $h[n]$ is obtained by the usual DTFT inversion formula

$$h[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})e^{j\omega n}d\omega.$$

While the analytical evaluation of the above integral may be difficult or impossible in the general case, for frequency responses $H(e^{j\omega})$ which are *piecewise linear*, the computation of $h[n]$ can be carried out in an exact (albeit not trivial) way; the result will be a linear combination of modulated sinc and sinc-squared sequences[1]. The FIR approximation is

[1]For more details one can analyze the Matlab `fir1` function.

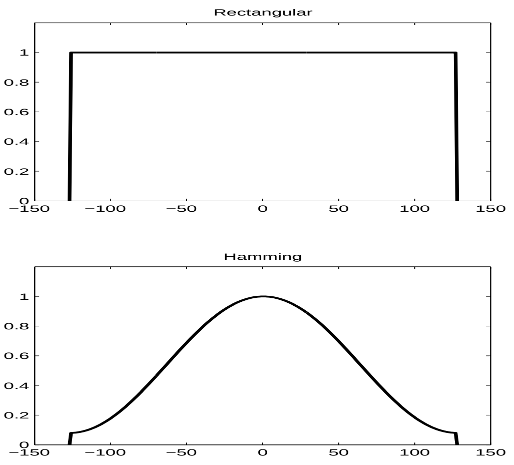


Figure 9.6: Some common windows in the time domain for $N = 127$ (total support 256 samples).

then obtained by applying a finite-length window $w[n]$ to the ideal impulse response:

$$\hat{h}[n] = w[n]h[n]$$

Since the frequency response of the FIR approximation is the circular convolution (in the frequency domain) of the desired response with the Fourier transform of the window, the window itself should be designed with the following goals in mind:

1. the window should be short, as to minimize the length of the FIR and therefore its computational cost

2. the spectrum of the window should be concentrated in frequency around zero as to minimize the "smearing" of the original frequency response; in other words, the window's main lobe should be as narrow as possible (it is clear that for $W(e^{j\omega}) = \delta(\omega)$ the resulting frequency response is identical to the original)

3. the unavoidable sidelobes of the window's spectrum should be small as to minimize the rippling effect in the resulting frequency response

It is clear that the first two requirements are openly in conflict; indeed, the width of the main lobe Δ is inversely proportional to the length of the window (we have seen, for instance, that for the rectangular window $\Delta = 4\pi/M$, with M the length of the filter). The second and third requirements are also in conflict, although the relationship between

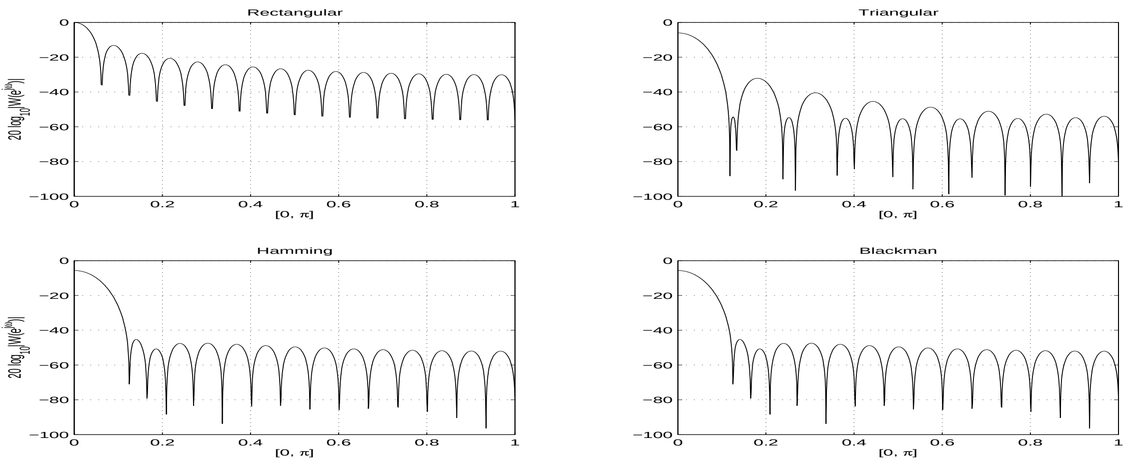


Figure 9.7: Some common windows in the frequency domain (magnitude in dBs).

mainlobe width and sidelobe amplitude is not straightforward and can be considered a design parameter. In the frequency response, reduction of the sidelobe amplitude implies that the Gibbs phenomenon is decreased, but at the price of an enlargement of the filter's transition band. For example, using a triangular window instead of a rectangular window strongly attenuates the ripples but, as a consequence, the transition band almost doubles.

There is a large body of literature concerning the design of windows; the most commonly used windows are those which admit a simple representation in closed form; amongst those we can name the triangular window, the Hamming window, and the Blackman window, which are plotted in Figure 9.6. For example, the zero-centered $(2N + 1)$-point Hamming window is defined as:

$$w(n) = 0.54 - 0.46\cos(2\pi(n+N)/2N), \quad |n| \leq N-1$$

while the Blackman window is defined as:

$$w(n) = 0.42 - 0.5\cos(2\pi(n+N)/2N) + 0.08\cos(4\pi(n+N)/2N), \quad |n| \leq N-1$$

Finally, it is worth mentioning the existence of the Kaiser window, in which a user definable parameter β is used to "tune" the mainlobe-sidelobe tradeoff independently of the window length.

9.3.2 Minimax FIR Filter Design

As we saw in the opening example, FIR filter design by windowing minimizes the overall mean square error between the desired frequency response and the actual response of the filter. Since this might lead to a very large error at frequencies near the transition band, we will now consider a different approach, namely the design by minimax optimization. This techniques minimizes the maximum allowable error in the filter's magnitude response, both in the passband and in the stopband. Optimality in the minimax sense requires therefore the explicit stating of a set of *tolerances* in the prototypical frequency response, in the form of design specifications as seen in section 9.2.2. Before tackling the design procedure itself, we will need a series of intermediate results.

Generalized Linear Phase. In section 7.6.2 we introduced the concept of linear phase; a filter with linear phase response is particularly desirable since the phase response translates to just a time delay (possibly fractional) and we can concentrate on the magnitude response only. We also introduced the notion of group delay and showed that linear phase corresponds to constant group delay. Clearly the converse is not true: a frequency response of the type

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

has constant group delay but differs from a linear phase system by a constant phase factor $e^{j\alpha}$. We will call this type of phase response *generalized linear phase*. Important cases are those for which $\alpha = 0$ (strictly linear phase) and $\alpha = \pi/2$ (generalized linear phase used in differentiators).

FIR Filter Types. Consider a causal, M-tap FIR filter with impulse response $h[n]$, $n = 0, 1, \ldots, M-1$; in the following we will be interested in filters whose impulse response is *symmetric or antisymmetric around its "midpoint"*. If the number of taps is odd, the midpoint of the impulse response coincides with the center tap $h[(M-1)/2]$; if the number of taps is even, on the other hand, the midpoint is still at $(M-1)/2$ but this value does not coincide with a tap since it is located "right in between" taps $h[M/2-1]$ and $h[M/2]$. Symmetric and antisymmetric FIR filters are important since their frequency response has generalized linear phase. The delay introduced by these filters is equal to $(M-1)/2$ samples; clearly this is an integer delay if M is odd, and it is fractional (half a sample more) if M is even. There are clearly four different possibilities for linear phase FIR impulse responses, which are listed here with their corresponding generalized linear phase parameters :

The generalized linear phase of (anti)symmetric FIR's is easily shown. Consider for instance a Type I filter, and define $C = (M-1)/2$, the location of the center tap; we can

Filter Type	**Number of Taps**	**Symmetry**	**Delay**	**Phase Factor**
Type I	odd	symmetric	integer	$\alpha = 0$
Type II	even	symmetric	fractional	$\alpha = 0$
Type III	odd	antisymmetric	integer	$\alpha = \pi/2$
Type IV	even	antisymmetric	fractional	$\alpha = \pi/2$

compute the transfer function of the shifted impulse response $h_d[n] = h[n + C]$, which is now symmetric around zero. i.e. $h_d[-n] = h_d[n]$:

$$\begin{aligned} H_d(z) &= \sum_{n=-C}^{C} h_d[n]z^{-n} \\ &= h_d[0] + \sum_{n=-C}^{-1} h_d[n]z^{-n} + \sum_{n=1}^{C} h_d[n]z^{-n} \\ &= h_d[0] + \sum_{n=1}^{C} h_d[n](z^n + z^{-n}) \end{aligned} \tag{9.15}$$

By undoing the time shift we obtain the original Type I transfer function:

$$H(z) = z^{-\frac{M-1}{2}} H_d(z). \tag{9.16}$$

On the unit circle we have

$$\begin{aligned} H_d(e^{j\omega}) &= h_d[0] + \sum_{n=1}^{C} h_d[n](e^{j\omega n} + e^{-j\omega n}) \\ &= h_d[0] + 2\sum_{n=1}^{C} h_d[n]\cos n\omega \end{aligned} \tag{9.17}$$

which is a *real* function; the original Type I frequency response is obtained from (9.16)

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

which is clearly linear phase with delay $d = (M-1)/2$ and $\alpha = 0$. The generalized linear phase of the other three FIR types can be shown in exactly the same way.

Zero Locations. The symmetric structures of the four types of FIR filters impose some constraints on the locations of the zeros of the transfer function. Consider again a Type I filter; from (9.15) it is easy to see that $H_d(z^{-1}) = H_d(z)$; by using (9.16) we therefore have

$$\begin{cases} H(z) = z^{-\frac{M-1}{2}} H_d(z) \\ H(z^{-1}) = z^{\frac{M-1}{2}} H_d(z) \end{cases}$$

which leads to:

$$H(z^{-1}) = z^{M-1} H(z). \tag{9.18}$$

It is easy to show that the above relation is also valid for Type II filters, while for Type III and Type IV (antisymmetric filters) we have:

$$H(z^{-1}) = -z^{M-1} H(z). \tag{9.19}$$

These relations mean that if z_0 is a zero of a linear phase FIR, then so is z_0^{-1}. This result, coupled with the usual fact that all complex zeros come in conjugate pairs, implies that if z_0 is a zero of $H(z)$ then:

- If $z_0 = \rho \in \mathbb{R}$ then $(\rho, 1/\rho)$ are zeros.
- If $z_0 = \rho e^{j\theta}$ then $(\rho e^{j\theta}, (1/\rho)e^{j\theta}, \rho e^{-j\theta}, (1/\rho)e^{-j\theta})$ are zeros.

Consider now equation (9.18) again; if we set $z = -1$ we have

$$H(-1) = (-1)^{M-1} H(-1); \tag{9.20}$$

for Type II filters, $M - 1$ is an odd number, which leads to the conclusion that $H(-1) = 0$; in other words, Type II filters *must* have a zero at $\omega = \pi$. Similar results can be demonstrated for the other filter types, and they are summarized as such:

Filter Type	**Relation**	**Constraint on Zeros**
Type I	$H(z^{-1}) = z^{M-1}H(z)$	No constraints
Type II	$H(z^{-1}) = z^{M-1}H(z)$	Zero at $z = -1$ (i.e. $\omega = \pi$)
Type III	$H(z^{-1}) = -z^{M-1}H(z)$	Zeros at $z = \pm 1$ (i.e. at $\omega = \pi$, $\omega = 0$)
Type IV	$H(z^{-1}) = -z^{M-1}H(z)$	Zero at $z = 1$ (i.e. $\omega = 0$)

These constraints are important in the choice of the filter type for a given set of specifications. Type II and Type III filters are not suitable in the design of highpass filters, for instance; similarly, Type III and Type IV filters are not suitable in in the design of lowpass filters.

Chebyshev Polynomials. Chebyshev polynomials are a family of orthogonal polynomials $\{T_k(x)\}_{k\in\mathbb{N}}$ which have, amongst others, the following interesting property:

$$\cos n\omega = T_n(\cos\omega); \tag{9.21}$$

in other words, the cosine of an integer multiple of an angle ω can be expressed as a polynomial in the variable $\cos\omega$. The first few Chebyshev polynomials are:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_2(x) &= 2x^2 - 1 \\ T_3(x) &= 4x^3 - 3x \\ T_4(x) &= 8x^4 - 8x^2 + 1 \end{aligned}$$

and, in general, they can be derived from the recursion formula

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x). \tag{9.22}$$

From the above table it is easy to see that we can write, for instance

$$\cos(3\omega) = 4\cos^3\omega - 3\cos\omega$$

The interest in Chebyshev polynomials comes from the fact that the zero-centered frequency response of a linear phase FIR can be expressed as a linear combination of cosine functions, as we have seen in detail for Type I filters in (9.17). By using Chebyshev polynomials we can rewrite such a response as just one big polynomial in the variable $\cos\omega$. Let us consider an explicit example for a length-7, Type I filter with nonzero coefficients $h[n] = [d\ c\ b\ a\ b\ c\ d]$; we have:

$$H_d(e^{j\omega}) = a + 2b\cos\omega + 2c\cos 2\omega + 2d\cos 3\omega$$

and by using the first four Chebyshev polynomials we can write:

$$\begin{aligned} H_d(e^{j\omega}) &= a + 2b\cos\omega + 2c(2\cos^2\omega - 1) + 2d(4\cos^3\omega - 3\cos\omega) \\ &= (a - 2c) + (2b - 6d)\cos\omega + 4c\cos^2\omega + 8d\cos^3\omega. \end{aligned} \tag{9.23}$$

In this case, $H_d(e^{j\omega})$ turns out to be a third degree polynomial in the variable $\cos\omega$. This is the case for any Type I filter, for which we can always write

$$H_d(e^{j\omega}) = \sum_{k=0}^{(M-1)/2} c_k \cos^k\omega \tag{9.24}$$

$$= P(x)\big|_{x=\cos\omega}. \tag{9.25}$$

where $P(x)$ is a polynomial of degree $(M-1)/2$ whose coefficients c_k are derived as linear combinations of the original filter coefficients a_k; we showed an example of this in (9.23). For the other types of linear phase FIR, a similar representation can be obtained after a few trigonometric manipulations. The general expression is:

$$\begin{aligned} H_d(e^{j\omega}) &= f(\omega)\sum_{k=0}^{L} c_k \cos^k \omega \\ &= f(\omega)P(x)|_{x=\cos\omega}; \end{aligned}$$

where the c_k's are still linear combinations of the original filter coefficients and where $f(\omega)$ is a weighting trigonometric function. Both $f(\omega)$ and the polynomial degree K vary as a function of the filter type[2]. In the following sections, however, we will concentrate only on the design of Type I filters, so these details will be overlooked; in practice, since the design is always carried out using numerical packages, the appropriate formulation for the filter expression is taken care of automatically.

Polynomial Optimization. Back to the filter design problem, we know that the FIR filters are (generalized) linear phase, so we can concentrate on the real frequency response of the zero-centered filter, which is represented by the trigonometric polynomial (9.25). Also, since the impulse response is real and symmetric, the aforementioned real frequency response is also symmetric around $\omega = 0$. Therefore the filter design procedure can be carried out only for values of ω over the interval $[0, \pi]$, with the other half of the spectrum obtained by symmetry. For these values of ω, the variable $x = \cos\omega$ is mapped onto the interval $[1, -1]$ and the mapping is invertible. Therefore, *the filter design problem becomes a problem of polynomial approximation over an interval.*

To illustrate the procedure by example, consider once more the set of filter specifications in Figure 9.1 and suppose we decide to use a Type I filter. Recall that we required the prototype filter to be lowpass, with a transition band from $\omega_p = 0.4\pi$ to $\omega_s = 0.6\pi$; we further stated that the tolerances for the realized filter's magnitude must not exceed 10% in the passband and 1% in the stopband. This implies that the maximum magnitude error between the prototype filter and the FIR filter response $H(e^{j\omega})$ must not exceed $\delta_p = 0.1$ in the interval $[0, \omega_p]$ and must not exceed $\delta_s = 0.01$ in the interval $[\omega_s, \pi]$. We

[2]For the sake of completeness, here is a summary of the details:

Filter Type	L	$f(\omega)$
Type I	$(M-1)/2$	1
Type II	$(M-2)/2$	$\cos(\omega/2)$
Type III	$(M-3)/2$	$\sin(\omega)$
Type IV	$(M-2)/2$	$\sin(\omega/2)$

can formulate this as follows: the frequency response of the desired filter is:

$$H_D(e^{j\omega}) = \begin{cases} 1 & \omega \in [0, \omega_p] \\ 0 & \omega \in [\omega_s, \pi] \end{cases} \tag{9.26}$$

(note that $H_D(e^{j\omega})$ is not specified in the transition band). Since the tolerances on passband and stopband are different, they can be expressed in terms of a weighting function $H_W(\omega)$ such that the tolerance on the error is constant over the two bands:

$$H_W(\omega) = \begin{cases} 1 & \omega \in [0, \omega_p] \\ \delta_p/\delta_s & \omega \in [\omega_s, \pi] \end{cases} \tag{9.27}$$

The design problem can now be reformulated as follows by defining an error function

$$E(\omega) = H_W(\omega) \left[H_d\left(e^{j\omega}\right) - H_D\left(e^{j\omega}\right) \right]. \tag{9.28}$$

Then the optimization problem becomes

$$\min_{\hat{\mathbf{h}}} \left\{ \max_{\omega \in F} |E(\omega)| \right\}, \tag{9.29}$$

where F is the closed subset of $0 \leq \omega \leq \pi$ such that

$$F = \{[0, \omega_p] \cup [\omega_s, \pi]\} = I_p \cup I_s \quad with \quad I_p = [0, \omega_p], I_s = [\omega_s, \pi], \tag{9.30}$$

and the question now is to find the coefficients for $h[n]$ (their number M and their values) which minimize the above error. If we have a feasible solution to (9.29) such that

$$\max_{\omega \in F} |E(\omega)| \leq \delta_p, \tag{9.31}$$

then we know that for $\omega \in I_p = [0, \omega_p]$,

$$\left| H_{des}\left(e^{j\omega}\right) - \hat{H}\left(e^{j\omega}\right) \right| \leq \delta_p, \tag{9.32}$$

and for $\omega \in I_s = [\omega_s, \pi]$, we have

$$\left| H_{des}\left(e^{j\omega}\right) - \hat{H}\left(e^{j\omega}\right) \right| \leq \delta_s. \tag{9.33}$$

Hence a feasible solution satisfying (9.31) is what is needed for a design. This optimization framework allows us to check if we can meet the desired specification. Note that we leave the transition band unconstrained (i.e. it doesn't affect the minimization of the error). Thus we can define

$$\tilde{W}(\omega) = \begin{cases} 1 & \omega \in [0, \omega_p] \\ \delta_p/\delta_s & \omega \in [\omega_s, \pi] \\ 0 & \omega \in (\omega_p, \omega_s), \end{cases} \tag{9.34}$$

and write

$$\tilde{E}(\omega) = \tilde{W}(\omega)\left[H_d\left(e^{j\omega}\right) - H_D\left(e^{j\omega}\right)\right], \tag{9.35}$$

with the feasibility check of

$$\|\tilde{E}(\omega)\|_\infty \triangleq \max_{\omega} |\tilde{E}(\omega)| \le \delta_p. \tag{9.36}$$

Since

$$\|\tilde{E}(\omega)\|_\infty = \max_{\omega} |\tilde{E}(\omega)| = \max_{\omega \in I_p \cup I_s} |\tilde{E}(\omega)|,$$

this gives the same conditions as (9.32) and (9.33).

Therefore, the optimization problem we have is a $\|\cdot\|_\infty$ norm minimization instead of $\|\cdot\|_2$ norm minimization we had from before in (9.11). This philosophy of viewing the different filter design criteria as just optimization problems is a useful viewpoint which can be generalized to other design criteria. That is (9.29) is just

$$\min_{\hat{H}(e^{j\omega})} \|\tilde{E}(\omega)\|_\infty.$$

The next step is to use (9.25) to reformulate the above expression as a polynomial optimization problem. To do so we replace the frequency response $H_d(e^{j\omega})$ with its polynomial equivalent and set $x = \cos\omega$; the passband interval $[0, \omega_p]$ and the stopband interval $[\omega_s, \pi]$ are mapped into the intervals for x:

$$\begin{aligned} I_p &= [\cos\omega_p, 1] \\ I_s &= [-1, \cos\omega_s]; \end{aligned}$$

respectively; similarly, the desired response becomes:

$$D(x) = \begin{cases} 1 & \omega \in I_p \\ 0 & \omega \in I_s \end{cases} \tag{9.37}$$

and the weighting function becomes:

$$W(x) = \begin{cases} 1 & \omega \in I_p \\ \delta_p/\delta_s & \omega \in I_s \end{cases} \tag{9.38}$$

The new set of specifications are shown in Figure 9.8. Within this polynomial formulation, the optimization problem becomes:

$$\max_{x \in I_p \cup I_s} \{W(x)|P(x) - D(x)|\} = \max\{|E(x)|\} \le \delta_p \tag{9.39}$$

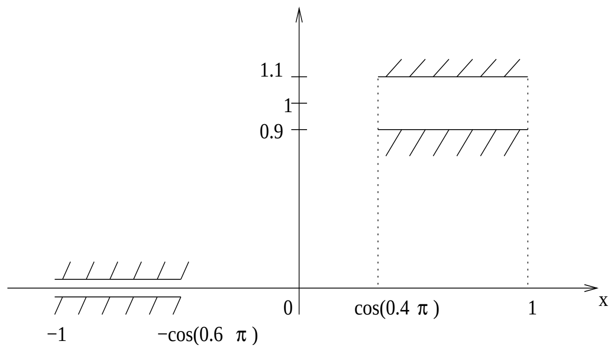


Figure 9.8: The filter specifications as in Figure 9.1 formulated here in terms of polynomial approximation, i.e. $x = \cos\omega$ for $\omega \in [0, \pi]$.

where $P(x)$ is the polynomial representation of the FIR frequency response as in (9.25).

Alternation Theorem:

The optimization problem stated by (9.39) can be solved by using the following theorem:

Theorem 9.1 *Consider a set* $\{I_k\}$ *of* closed*, disjoint intervals on the real axis and their union* $I = \cup_k I_k$*. Consider further:*

- *a polynomial* $P(x)$ *of degree* L, $P(x) = \sum_{n=0}^{L} a_n x^n$
- *a desired function* $D(x)$*, continuous over* I
- *a positive weighting function* $W(x)$

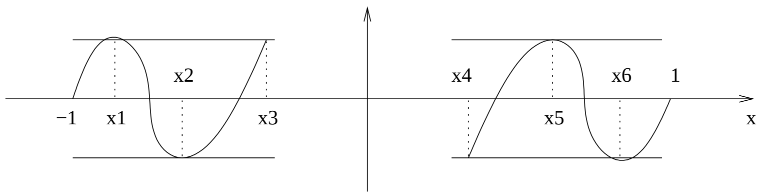


Figure 9.9: Equiripple error in passband and stopband

Consider now the approximation error function

$$E(x) = W(x)[D(x) - P(x)]$$

and the associated maximum approximation error over the set of closed intervals

$$E_{\max} = \max_{x \in I}\{|E(x)|\}$$

Then $P(x)$ is the unique *order-L polynomial which minimizes $E_{\max}$ if and only if there exist* at least $L + 2$ *successive values x_i in I such that $|E(x_i)| = E_{\max}$ and*

$$E(x_i) = -E(x_{i+1}).$$

In other words, the error function must have at least $L + 2$ alternations between its maximum and minimum values. Such a function is called equiripple.

Back to our lowpass filter example, assume we are trying to design a 9-tap optimal filter. This theorem tells us that if we found a polynomial $P(x)$ of degree 4 such that the error function (9.39) over I_p and I_s looks as in Figure 9.9 (6 alternations), then the polynomial would be the *optimal* and *unique* solution. Note that the extremal points (i.e. the values of the error function at the edges of the optimization intervals) *do* count in the number of alternations since the intervals I_k are closed.

The above theorem may seem a bit far-fetched since it does not tell us how to find the coefficients but it only gives us a test to verify their optimality. This test, however, is at the core of an *iterative* algorithm which refines the polynomial from an initial guess until the optimality condition is met. Before considering the optimization procedure more in detail, we will state without formal proof three consequences of the alternation theorem as it applies to the design of Type I lowpass filters:

- The minimum number of alternations for an optimal M-tap lowpass filter is $L + 2$, with $L = (M - 1)/2$; this is the result of the alternation theorem. The *maximum* number of alternation, however, is $L + 3$; filters with $L + 3$ alternation are called extraripple filters.
- Alternations always take place at $x = \cos\omega_p$ and $x = \cos\omega_s$ (i.e. at $\omega = \omega_p$ and $\omega = \omega_s$.
- If the error function has a local maximum or minimum, its absolute value at the extremum must be equal to $E_{\max}$ except possibly in $x = 0$ or $x = 1$. In other words, all local maxima and minima of the frequency response must be alternation except in $\omega = 0$ or $\omega = \pi$.
- If the filter is extraripple, the extra alternation occurs at either $\omega = 0$ or $\omega = \pi$.

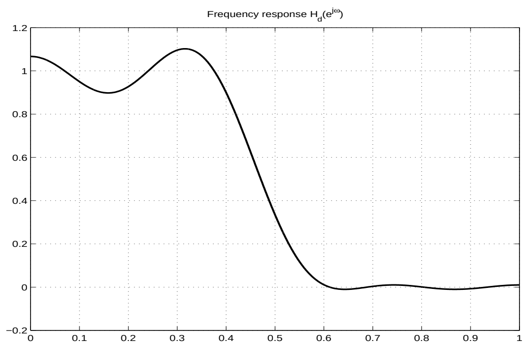


Figure 9.10: An optimal 13-tap Type I filter which does not meet the error specifications.

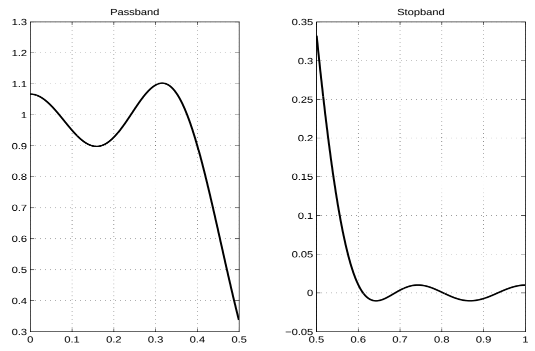


Figure 9.11: Details of passband and stopband of the frequency response in Figure 9.10.

Optimization Procedure. Finally, by putting all the elements together, we are ready to state an algorithmic optimization procedure for the design of optimal minimax FIR filters; this procedure is usually called the Parks-McClellan algorithm. Remember, we are trying to determine a polynomial $P(x)$ such that the approximation error in (9.39) is equiripple; for this, we need to determine both the degree of the polynomial and its coefficients. For a given degree L, for which the resulting filter will have $2L+1$ taps, the L coefficients are found by an iterative procedure which successively refines an initial guess for the $L+2$ alternation points x_i until the error is equiripple[3]. After the iteration has converged, we need to check that the corresponding $E_{\max}$ satisfies the upper bound

[3]Details about this crucial optimization step can be found in the bibliographic references. While a thorough discussion of the algorithm is beyond the scope of these notes, suffice it to say that at each iteration the new set of candidate extremal points is obtained by exchanging the old set with the ordinates of the current local maxima. This trick is also known as the Remez exchange algorithm and that is why, in Matlab, the Parks-McClellan algorithm is named `remez`.

imposed by the specifications; when this is not the case, the degree of the polynomial (and therefore the length of the filter) must be increased and the procedure must be restarted. Once the conditions on the error are satisfied, the filter coefficients can be obtained by inverting the Chebyshev expansion.

As a final note, an initial guess for the number of taps can be obtained using the empirical formula by Kaiser; for an M-tap FIR $h[n]$, $n = 0, \ldots, M-1$:

$$M \simeq \frac{-10\log_{10}(\delta_p\delta_s) - 13}{2.324\Omega} + 1$$

where δ_p is the passband tolerance, δ_s is the stopband tolerance and $\Omega = \omega_s - \omega_p$ is the width of the transition band.

The final design. We will now summarize the design steps for the specifications in Figure 9.1. We will use a Type I FIR. We start by using Kaiser's formula to obtain an estimate of the number of taps: since $\delta_p\delta_s = 10^{-3}$ and $\Omega = 0.2\pi$, we obtain $M = 12.6$ which we will round up to 13 taps. At this point we can use any numerical package for filter design to run the Parks-McClellan algorithm. In Matlab this would be

```
[h, err] = remez(12, [0 0.4 0.6 1], [1 1 0 0], [1 10]);
```

The resulting frequency response is plotted in Figure 9.10; please note that we are plotting the frequency responses of the zero-centered filter $h_d[n]$, which is a real function of ω. We can verify that the filter has indeed $(M-1)/2 = 6$ alternation by looking at a blowup picture of the passband and the stopband, as in Figure 9.11. The maximum error as returned by Matlab is however 0.102 which is larger than what our specifications called for, i.e. 0.1. We are thus forced to increase the number of taps; since we are using a Type I filter, the next choice is $M = 15$. Again, the error turns out to be larger than 0.1, since in this case we have $E_{\max} = 0.1006$. The next choice, $M = 17$, finally yields an error $E_{\max} = 0.05$, which exceeds the specifications by a factor of 2. It's the designer's choice to decide whether the computational gains of a shorter filter ($M = 15$) outweigh the small excess error. The impulse response and the frequency response of the 17-tap filter are plotted in Figure 9.12.

Other Types of Filters. The Parks-McClellan optimal FIR design procedure can be made to work for arbitrary filter types as well, such as highpass and passband of course but also for more sophisticated frequency responses. The constraints imposed by the zero locations as we saw on page 216 determine the type of filter to use; once the desired response $H_D(e^{j\omega})$ is expressed as a trigonometric function, the optimization algorithm can take its course. For arbitrary frequency responses, however, the fact that the transition bands are left unconstrained may lead to unacceptable peaks which render the filter useless. In these

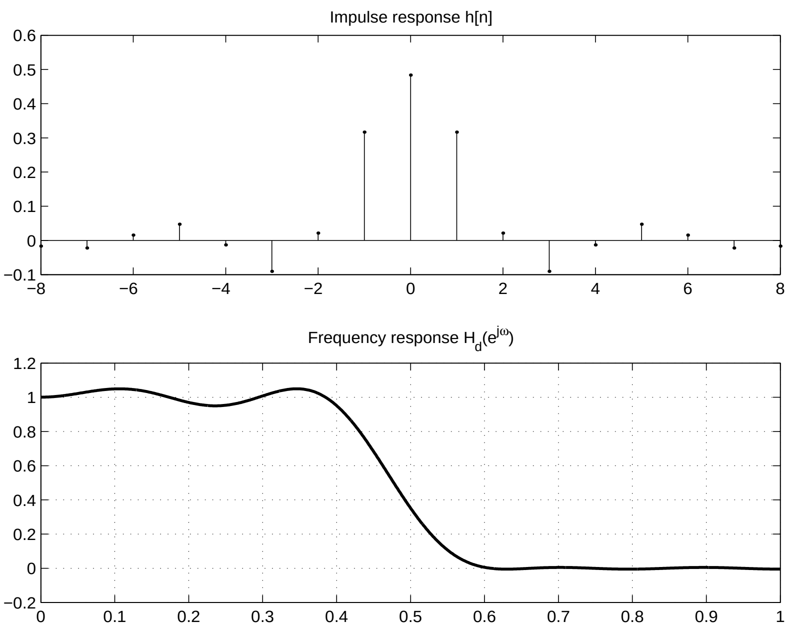


Figure 9.12: The 17-tap filter meeting the specifications.

cases, visual inspection of the obtained response is mandatory and experimentation with different filter lengths and tolerance may improve the final result.

Example 9.1

In the window method for filter design, one multiplies the desired impulse response $h[n]$ *(potentially of infinite length) with a window* $w[n]$ *of finite length* M *to obtain an approximation of the desired response* i.e., $h_{approx}[n] = h\left[n - \frac{M-1}{2}\right] w[n]$. *The Hamming window is defined as*

$$w[n] = \begin{cases} 0.54 - 0.46\cos\frac{2\pi n}{M-1} & \textit{for } 0 \leq n \leq M-1 \\ 0 & \textit{otherwise} \end{cases}$$

where M *is the number of taps (see Fig. 9.13).*

(a) We will use the window method with a Hamming window to design a $M = 21$*-tap differentiator. The frequency response of a differentiator is* $H(e^{j\omega}) = j\omega$ *for* $-\pi \leq$

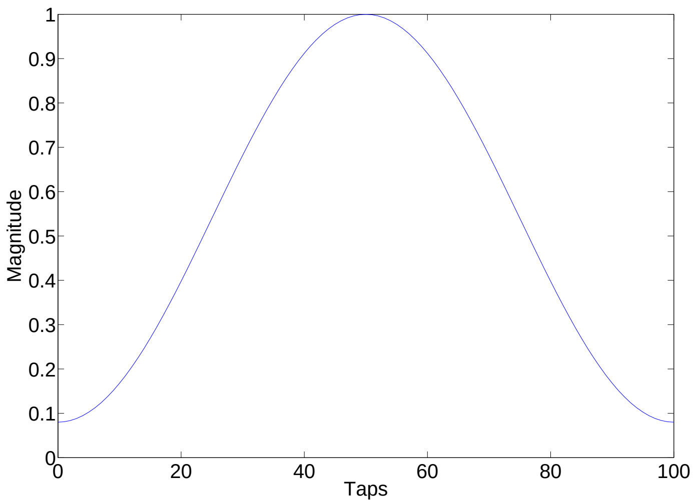


Figure 9.13: A Hamming window of length $M = 101$

$\omega \leq \pi$. *Give an expression for the impulse response* $h_{approx}[n]$. *Provide a single plot with* $h_{approx}[n]$, $w[n]$ *and* $h[n]$ *(Hint: you can superimpose plots using the* `hold on` *command in matlab* e.g. `plot([1 2 3],[3 2 1],'r:');hold on;plot([1 2 3],[3 3 3],'g-');plot([1 2 3],[4 5 29],'b*')`*)*

(b) *Provide matlab plots of the phase and amplitude of the frequency response* $H_{approx}(e^{j\omega})$, *as well as plots of the phase and amplitude of the desired frequency response* $H(e^{j\omega})$, *if possible combine the amplitude plots on one figure and the phase plots on one figure (Use matlab to compute the DFT* $H_{approx}(e^{j\omega})$, *plot* $H(e^{j\omega})$ *for a large number of values of* ω*). Comment the result.*

Solution:

(a) *We first need to compute* $h[n]$ *for* $n \neq 0$*:*

$$h[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi} j\omega e^{j\omega n}d\omega = \frac{\cos \pi n}{n}$$

For $n = 0$,

$$h[0] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})e^{0}d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi} j\omega d\omega = 0$$

Hence,

$$\begin{aligned} h_{approx}[n] &= h[n-\tfrac{M-1}{2}]w[n] \\ &= (0.54 - 0.46\cos\tfrac{2\pi n}{M-1})(\tfrac{\cos(\pi(n-\frac{M-1}{2}))}{n-\frac{M-1}{2}}) \\ &= 0.54\tfrac{\cos(\pi(n-\frac{M-1}{2}))}{n-\frac{M-1}{2}} - 0.46\tfrac{\cos\frac{2\pi n}{M-1}\cos(\pi(n-\frac{M-1}{2}))}{n-\frac{M-1}{2}} \end{aligned}$$

In Fig. 9.14 we show $h[n-\frac{M-1}{2}]$, $w[n]$ and $h_{approx}[n]$.

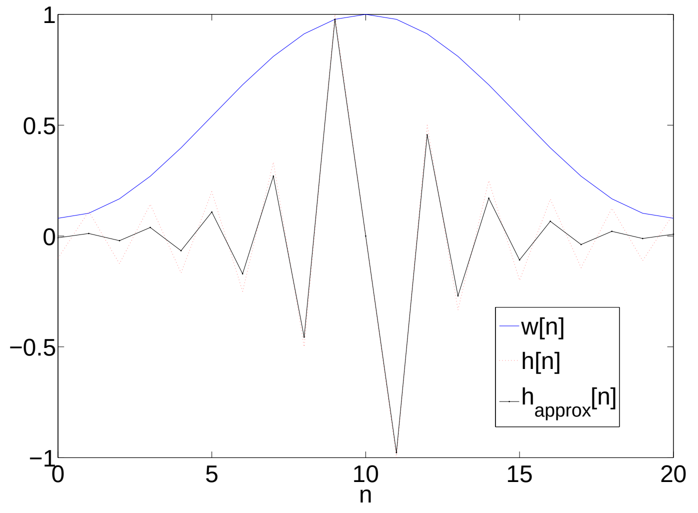


Figure 9.14: $h[n-\frac{M-1}{2}]$, $w[n]$ and $h_{approx}[n]$

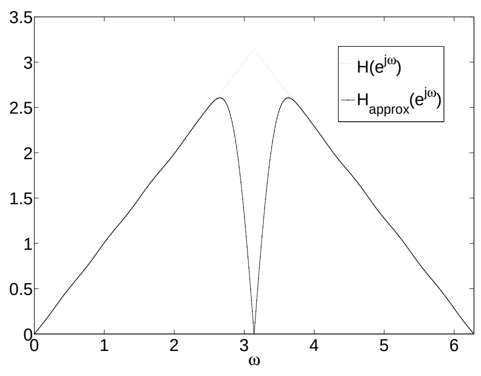
(b) In Fig. 9.15, we show the amplitude and frequency responses of $H_{approx}(e^{j\omega})$ and $H(e^{j\omega})$.

Example 9.2 (Gibbs Phenomenon) *In this exercise we will demonstrate the Gibbs phenomenon through rectangular windowing.*

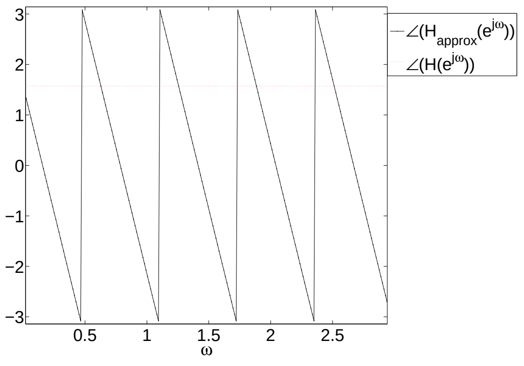
Suppose we want to design a lowpass filter with a cut-off frequency of $\pi/2$, ie.e we have a desired frequency response

$$H_{\text{des}}\left(e^{j\omega}\right) = \begin{cases} 1 & -\frac{1}{2} \leq \omega \leq \frac{1}{2} \\ 0 & \textit{elsewhere.} \end{cases}$$

Let $h_{\text{des}}[n]$ be the corresponding impulse response.



(a) Amplitude response



(b) Phase response

Figure 9.15: Amplitude and phase responses of $H_{approx}(e^{j\omega})$ and $H(e^{j\omega})$

We want to create a $2N+1$ taps filter that represents the desired response as close as possible. We do this by applying a rectangular window $w[n]$ that is defined by

$$w[n] = \begin{cases} 1 & -N \leq n \leq N \\ 0 & \textit{elsewhere.} \end{cases}$$

The resulting filter is given by

$$\hat{h}[n] = h_{\text{des}}[n]w[n]. \tag{9.40}$$

The goal of this exercise is to see the difference between $\left|\hat{H}\left(e^{j\omega}\right)\right|$ and $\left|H_{\text{des}}\left(e^{j\omega}\right)\right|$.

(a) Give the desired impulse response $h_{\text{des}}[n]$.

(b) Let $N = 10$. Use the ***fft*** *function in MATLAB to plot 1000 points of $\left|\hat{H}_{\text{des}}\left(e^{j\omega}\right)\right|$ in the interval $0.4\pi \leq \omega \leq 0.6\pi$. Make sure that the axes of your plot are labeled and have the right values. You can plot the points as a continuous line. Hand in your code (printout or handwritten) and a printout of the plot.* Hint: Have a look at problem 5 of Homework 2.

(c) Repeat Part (b) for $N = 100$ and $N = 200$. Give a printout of the plot only.

(d) How does the maximum of $\left|\hat{H}\left(e^{j\omega}\right)\right|$ depend on N?

(e) *We have seen in class that applying a rectangular window corresponds to the optimal solution according to some optimization criteria/constraints. Formulate the complete optimization problem that has Equation (9.40) as a solution.*

Solution:

(a) *As seen in class*

$$h_{\text{des}}[n] = \frac{1}{2}\text{sinc}\left(\frac{n}{2}\right).$$

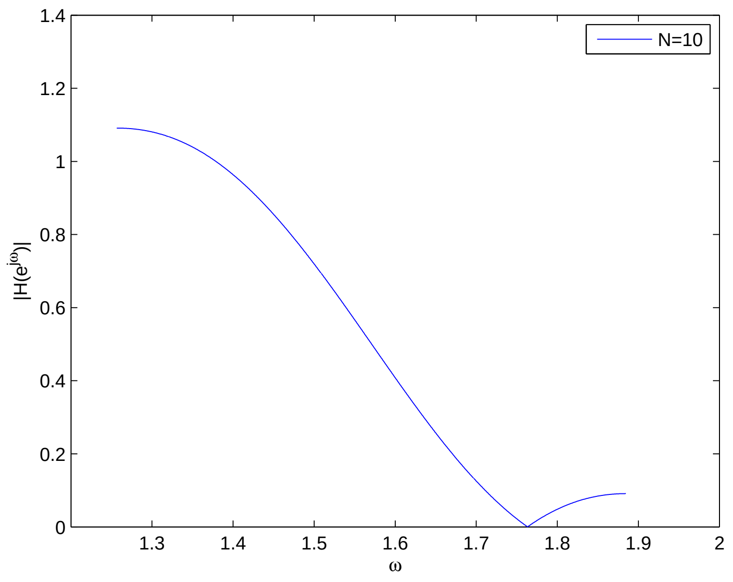
(b) *We want* 10^3 *points in the interval* $0.4\pi \leq \omega \leq 0.6\pi$*. This interval has length* 0.2π*. This means that we need* 10^4 *points in the entire interval. We take a DFT with* 10^4 *points.*

```
>> N = 10;
>> h_des = .5*sinc([-N:N]/2);
>> H_des = fft(h_des,1e4);
```

The interval $0.4\pi \leq \omega \leq 0.6\pi$ *corresponds to elements* $2001 \ldots 3000$ *of the DFT.*

```
>> w = linspace(0.4*pi,0.6*pi,1e3);
>> plot(w,abs(H_des(2001:3000)));
>> xlabel('\omega');
>> ylabel('|H(e^{j\omega})|');
>> legend('N=10')
```

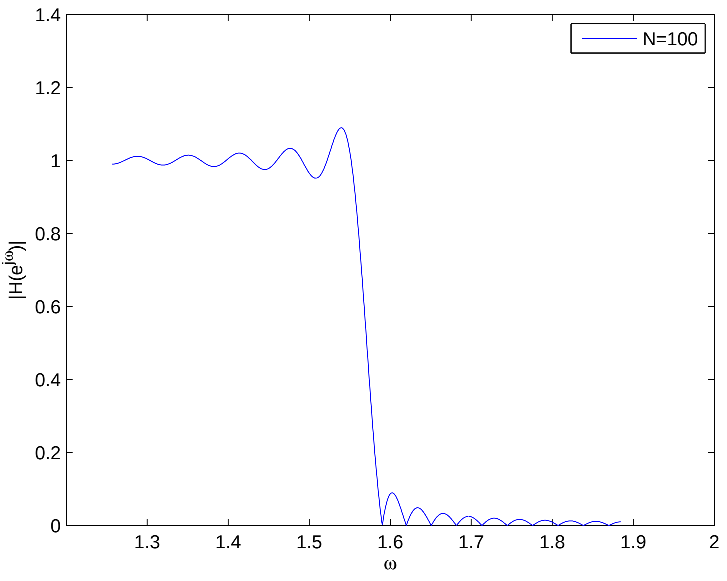
This gives us



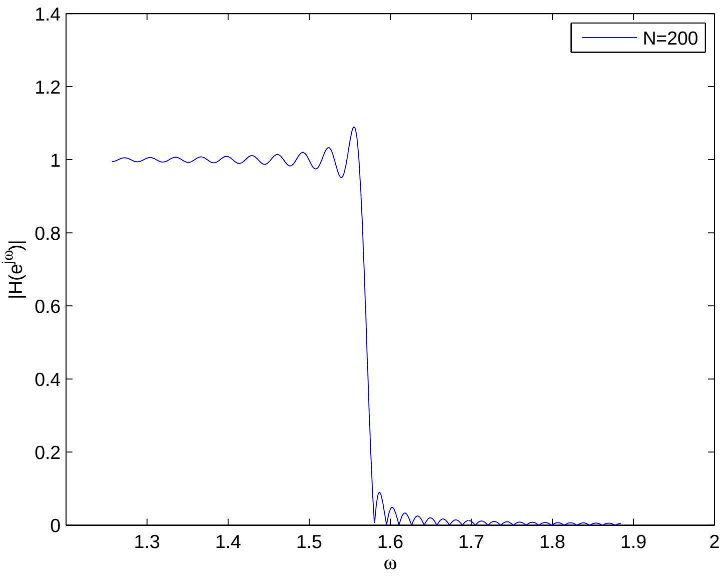
(c) For $N = 100$ we have

```
>> N = 100;
>> h_des = .5*sinc([-N:N]/2);
>> H_des = fft(h_des,1e4);
>> w = linspace(0.4*pi,0.6*pi,1e3);
>> plot(w,abs(H_des(2001:3000)));
>> xlabel('\omega');
>> ylabel('|H(e^{j\omega})|');
>> legend('N=100')
```

which gives



For $N = 200$ we give the plot only:



(d) The maximum does not depend on N, it is always around 1.09.

(e)

$$\begin{aligned} \textit{minimize} \qquad & \left\| \hat{H}\left(e^{j\omega}\right) - H\left(e^{j\omega}\right) \right\|_2^2, \\ \textit{subject to} \qquad & \hat{H}\left(e^{j\omega}\right) = \sum_{n=-N}^{N} \hat{h}[n] e^{-j\omega n}, \end{aligned}$$

where $\|\cdot\|_2^2$ *is the* $L_2[-\pi, \pi]$ *norm as defined in class.*

We see from part (d) that for increasing N *the approximation gets better in the mean-square sense, but that the maximum error remains about* 9%*.*

9.4 IIR Filter Design

As we mentioned in the introductory remarks, no optimal procedure exists for the design of IIR filters. The fundamental reason is that the optimization of the coefficients of a rational transfer function is a highly nonlinear problem and no satisfactory algorithm has yet been developed for the task. This, coupled with the impossibility of obtaining a linear phase response with an IIR[4] makes the design of IIR filter a much less formal art. Here we will concentrate on some basic IIR filters which are very simple and which are commonly used in the practice and we will briefly illustrate the basic principles behind more general IIR design techniques.

9.4.1 All-Time Classics

There are a few tried-and-true applications in which simple IIR structures are the design of choice. These filters are so simple and so well behaved that they are a fundamental tool in the arsenal of any signal processing engineer.

DC Removal and Mean Estimation. The DC component of a signal is its mean value; a signal with zero mean is also called an AC signal. This nomenclature comes from electrical circuit parlance: DC is shorthand for *direct current*, while AC stands for *alternating current*; you might be familiar with these terms in relation to the current provided by a battery (constant and hence DC) and the current available from a mains socket (alternating at 50 or 60 Hz and therefore AC).

For a given sequence $x[n]$, one can always write

$$x[n] = x_{\text{AC}}[n] + x_{\text{DC}}$$

where x_{DC} is the mean of the sequence values. Please note that:

- The DC value of a finite-support signal is the value of its Fourier transform at $\omega = 0$ times the length of the signal's support
- The DC value of an infinite-support signal must be zero for the signal to be absolutely summable.

In most signal processing applications, where the input signal comes from an acquisition device (such as a sampler, a soundcard and so on), it is important to remove the DC component; this is because the DC offset is often a random offset caused by ground mismatches between the acquisition device and the associated hardware.

[4]There actually is a theorem which states that an infinite impulse response with linear phase is not realizable.

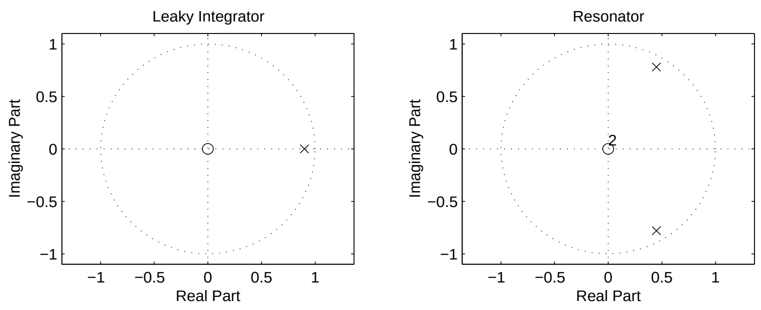


Figure 9.16: Pole-zero plots for the leaky integrator and the simple resonator.

For finite-length signals, computation of the mean is straightforward since it involves a finite number of operations. In most cases, however, we do not want to wait until the end of the signal before we try to remove its mean; what we need is some way to perform DC removal *on line*. The approach is therefore to obtain at each instant an *estimate* of the DC component from the past signal values, with the assumption that the estimate converges to the real mean of the signal. In order to obtain such an estimate, i.e. in order to obtain the average value of the past input samples, both approaches detailed in section 7.4 are of course valid (i.e. the Moving Average and the Leaky Integrator filters) . We have seen, however, that the leaky integrator provides a superior cost/benefit tradeoff and therefore the output of a leaky integrator with λ very close to one (usually 10^{-3}) is the estimate of choice for the DC component of a signal. The closer λ is to one, the more accurate the estimation; the speed of convergence of the estimate however becomes slower and slower as $\lambda \to 1$. This can be easily seen from the group delay at $\omega = 0$, which is

$$\operatorname{grd}\{H(1)\} = \frac{\lambda}{1-\lambda}$$

Resonator and Notch Filter Let's look again at how the leaky integrator works. Consider its Z-transform

$$H(z) = \frac{1-\lambda}{1-\lambda z^{-1}}$$

and notice that what we really want the filter to do is to extract the zero-frequency component (i.e. the frequency component that does not oscillate, i.e. the DC component). To do so, we placed a pole near $z = 1$, which of course corresponds to $z = e^{j\omega}$ for $\omega = 0$. Since the magnitude response of the filter will exhibit a peak near a pole, and since the peak will be higher the closer the pole is to the unit circle, we are in fact amplifying the zero-frequency component; this is apparent from the plot of the filter's frequency response in Figure 7.9. The numerator, $1 - \lambda$, is chosen such that the magnitude of the filter at $\omega = 0$ is one; the net result is that the zero-frequency component will pass unmodified while all the other frequencies will be attenuated. The value of a filter's magnitude at a given frequency is often called the *gain*.

The very same approach can now be used to extract a signal component at *any* frequency. We will use a pole whose magnitude is still close to one (i.e. a pole near the unit circle) but whose phase is that of the frequency we want to extract. We will then choose a numerator so that the magnitude is unity at the frequency of interest. The one extra detail is that, since we want a real-valued filter, we will have to place a complex conjugate pole as well. The resulting filter is called a resonator and a typical pole-zero plot is shown in Figure 9.16. The Z-transform of a resonator at frequency ω_0 is therefore determined by the pole $p = \lambda e^{j\omega_0}$ and by its conjugate:

$$H(z) = \frac{G_0}{(1 - pz^{-1})(1 - p^* z^{-1})} = \frac{G_0}{1 - (2\lambda \cos \omega_0) z^{-1} + \lambda^2 z^{-2}} \tag{9.41}$$

The numerator value G_0 is computed so that the filter's gain at $\pm\omega_0$ is one; since in this case $|H(e^{j\omega_0})| = |H(e^{-j\omega_0})|$, we have

$$G_0 = (1 - \lambda)\sqrt{1 + \lambda^2 - 2\lambda \cos 2\omega_0}$$

The magnitude and phase of a resonator with $\lambda = 0.9$ and $\omega_0 = \pi/3$ are shown in Figure 9.17.

A simple variant on the basic resonator can be obtained by considering the fact that the resonator is just a bandpass filter with a very narrow passband. As for all passband filters, we can therefore place a zero at $z = \pm 1$ and sharpen its midband frequency response. The corresponding Z-transform is now

$$H(z) = G_1 \frac{1 - z^{-2}}{1 - (2\lambda \cos \omega_0) z^{-1} + \lambda^2 z^{-2}}$$

with

$$G_1 = \frac{G_0}{\sqrt{2(1 - \cos 2\omega_0)}}$$

The corresponding magnitude response is shown in Figure 9.18.

9.4.2 IIR Design by Bilinear Transformation

As we mentioned in the introduction, analog filter design techniques give rise to analog filters whose transfer function (i.e. the Laplace transform of the continuous-time impulse response) is formally similar to the rational Z-transforms obtained from a constant-coefficient

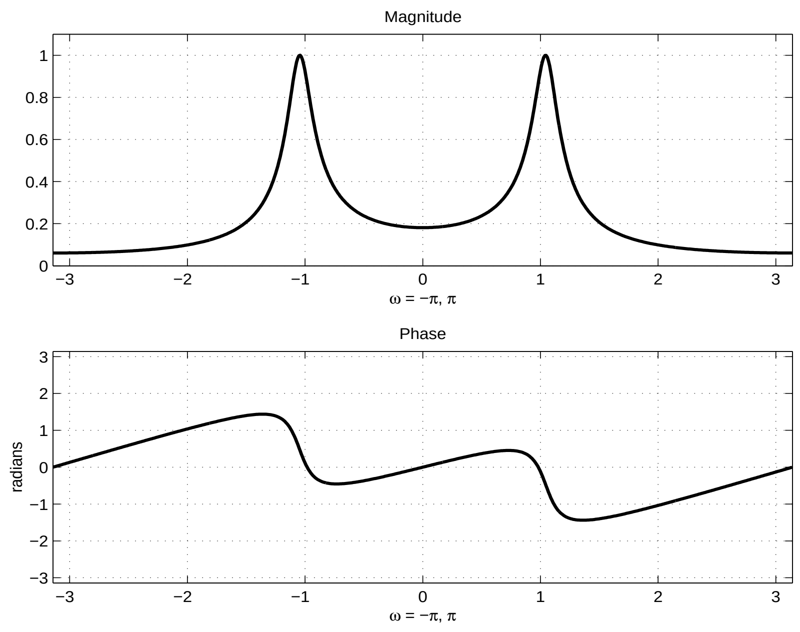


Figure 9.17: Frequency response of the simple resonator.

difference equation. This suggests associating a discrete-time IIR filter to the filter prototypes obtained in the continuous-time. While the details of analog filter design are outside the scope of these notes, it is important to mention that the techniques involved have an established and proven tradition; tabulated coefficients are readily available to determine the exact characteristics of an analog filter and the values of the electronic components which need to be employed. This is in stark contrast with the lack of an optimized IIR design procedure in the discrete-time domain.

The Analog Prototype. We will illustrate the design procedure by example. There are three fundamental families of (passive) analog lowpass filters: Butterworth, Chebyshev and Elliptic filters with the simplest type being Butterworth filters. Since we are going to illustrate the IIR design procedure by example, we will concentrate on this filter family.

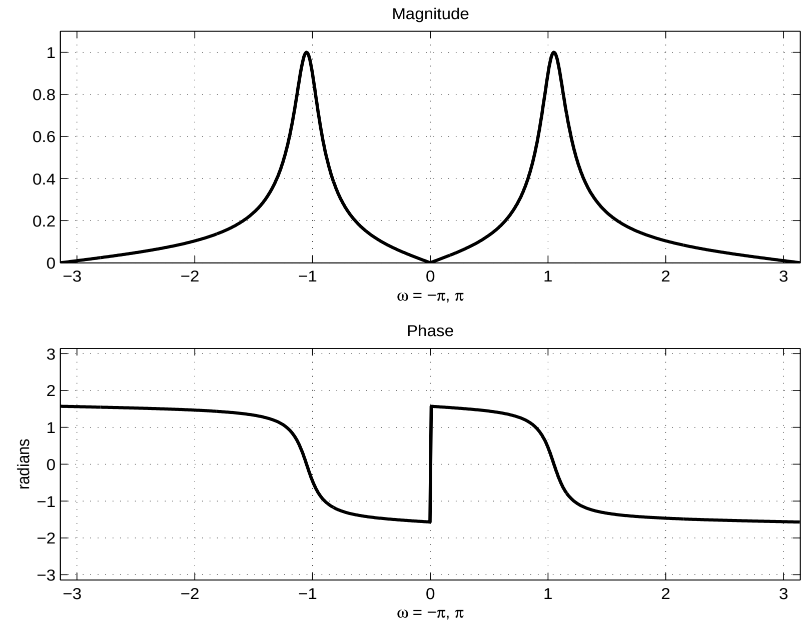


Figure 9.18: Frequency response of the modified resonator.

A *normalized* Butterworth filter of order N has an all-pole transfer function of the form

$$H(s) = \frac{1}{(s-s_0)(s-s_1)\dots(s-s_{N-1})}$$

$$= \left[\prod_{k=0}^{N-1}(s-s_k)\right]^{-1} \tag{9.42}$$

$$= \left[1+\sum_{k=1}^{N}a_k s^k\right]^{-1} \tag{9.43}$$

The values of the poles are derived by imposing that the square magnitude of the frequency response (i.e the magnitude of the continuous-time Fourier transform) must be of the form

$$|H(j\Omega)|^2 = \frac{1}{1+\Omega^{2N}} \tag{9.44}$$

which is plotted in Figure 9.19 for different values of N. The most important feature of this magnitude function is that it is monotonic. It is immediate to see that the squared magnitude response is equal to $1/2$ at $\Omega = 1$, which is the cutoff frequency of the *normalized*

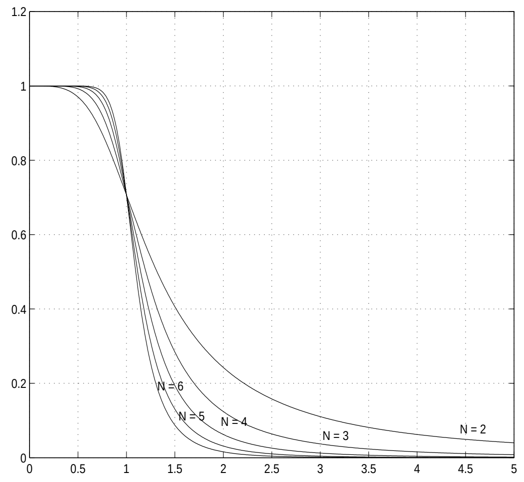


Figure 9.19: Magnitude of $H(j\Omega)$ for Butterworth filters of increasing order.

Butterworth. Since the impulse response must be real (we are designing an analog filter after all) we have that $H(-s) = H^*(s)$ and therefore (9.44) for $s = j\Omega$ translates to

$$|H(s)|^2 = H(s)H(-s) = \frac{1}{1 + (s/j)^{2N}}$$

which, in turn, gives the implicit locations of the poles as $s^{2N} = -j^{2N} = (-1)^{N-1}$. By solving for s, we have finally

$$s_k = e^{j\frac{\pi}{N}(k+\frac{N-1}{2})}, \quad k = 0, 1, \ldots, 2N-1$$

The poles, as it appears, are regularly distributed around the unit circle in the s-plane and their angular spacing is π/N; to obtain a stable and causal filter of order N, we just need to select the N poles in the left half of the s-plane (this is of course a standard result of System Theory). At this point we could plug these values back in (9.42) and obtain the transfer function coefficients in (9.43); in reality this is not how it's done, since tables exist which directly provide not only the a_k's for all practical values of N, but also the values of the electronic components necessary to build the filter. Clearly, Butterworth design is extremely straightforward from a practical point of view. As a last comment, note that if a different cutoff frequency Ω_c is desired, one needs just replace s_k by $\Omega_c s_k$ which, in turn, corresponds to scaling each a_k by Ω_c^k.

Bilinear Transformation. The analog design part of a Butterworth filter is just a simple table lookup and we now want to transfer the properties of the filter to the discrete-time domain. The idea is to transform the transfer function of a continuous time filter into a transfer function for a discrete-time filter; this can be achieved by replacing the variable s in $H(s)$ by a suitable function of the variable z. The function in question has to satisfy certain properties and, above all, it has to preserve the stability of the resulting filter. A common mapping function is the *bilinear transformation*:

$$s = 2\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right)$$

which is invertible via:

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

It can be verified that:

- The "frequency" axis $j\Omega$ in the s-plane is mapped onto the unit circle in the z-plane. This preserves the overall characteristic of the frequency response
- The left half of the s-plane is mapped *inside* the unit circle in the z-plane. This preserves the filter's stability.

The mapping linking the continuous frequency axis $j\Omega$ in the s plane to the periodic frequency axis $e^{j\omega}$ in the z-plane is given by:

$$\omega = 2\arctan(\Omega/2) \tag{9.45}$$

and conversely:

$$\Omega = 2\tan(\omega/2) \tag{9.46}$$

This represents a non-linear compression of the frequency axis, and therefore care must be taken in designing the filter specifications. An example of bilinear transformation is represented graphically in Figure 9.20.

A Design Example. Given a set of discrete-time specifications such as those in Figure 9.1, the design of a Butterworth digital filter involves the following steps: the starting point is the square-magnitude expression for the non-normalized filter

$$|H_c(j\Omega)|^2 = \frac{1}{1 + (\Omega/\Omega_c)^{2N}}$$

from which

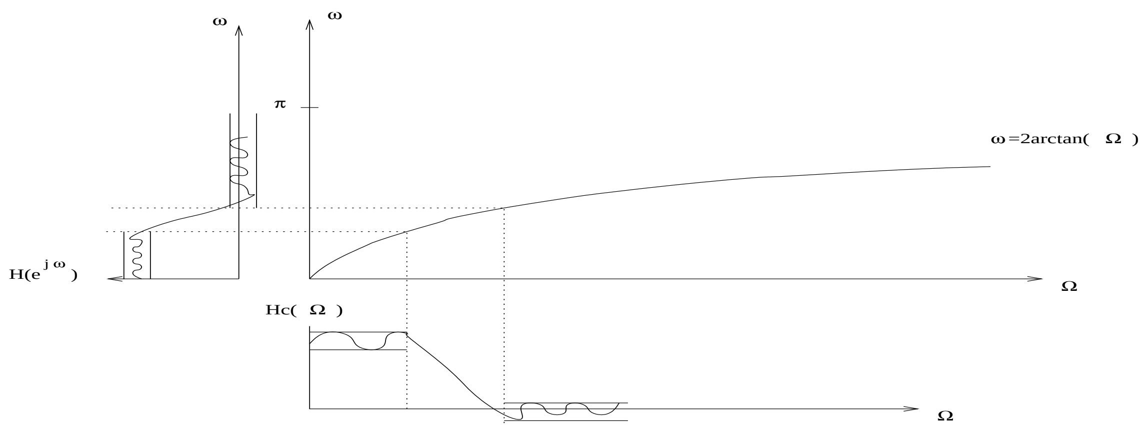


Figure 9.20: The mapping between Ω and ω operated by the bilinear transformation.

- Translate of the specifications in ω to specifications in Ω via (9.46):

$$\begin{aligned}\Omega_p &= 2\tan\omega_p\\ \Omega_s &= 2\tan\omega_s\end{aligned}$$

- Set the value of the square magnitude equal to the tolerance values (the monotonicity of the magnitude guarantees that the tolerance are satisfied if the conditions are imposed at the band edges)

$$\begin{aligned}|H_c(j\Omega_p)|^2 &= \delta_p^2\\ |H_c(j\Omega_s)|^2 &= \delta_s^2\end{aligned}$$

- Solve the above system of equations for N (the filter order) and Ω_c
- Find the normalized filter coefficients for the order N from a table
- Scale the coefficients by Ω_c^k
- Build the transfer function $H(s)$
- Apply the bilinear transformation to obtain $H(z)$

Numerical examples can be found in the bibliography. As a last remark, note that the Matlab command `butter` can be used to design digital Butterworth filters.

9.5 Filter Structures

We have seen in Section 9.1.1 a practical implementation of a rational transfer function. That was just one particular way of translating equation (9.1) into a working structure; it served well as an illustration but the design choices one can make are many, and we will now approach the design problem from a more general point of view.

It is easy to see, from inspection, that the basic building block which enter the recipe for a real-world filter are:

1. An addition operator for sequence values, implementing $y[n] = x_1[n] + x_2[n]$ (Fig. 9.21(a)).
2. A scalar multiplication operator, implementing $y[n] = ax[n]$ (Fig. 9.21(b)).
3. A unit delay operator, implementing $y[n] = x[n-1]$ (Fig. 9.21(c)).

By properly combining these elements and by exploiting the different possible factorization of a filter's rational transfer function, we can arrive at a variety of different working implementations of a filter.

9.5.1 FIR Filter Structures

In the Z-transform representation of an FIR transfer function as in (9.5), all the denominator coefficients a_n are zero; we have therefore

$$H(z) = b_0 + b_1 z^{-1} + \ldots + b_{M-1} z^{-(M-1)}$$

where, of course, the coefficients correspond to the nonzero values of the impulse response $h[n]$, i.e. $b_n = h[n]$. Using the constitutive elements outlined above, we can immediately draw a block diagram of an FIR filter as in Figure 9.22. In practice, however, additions will be distributed as shown in Figure 9.23; this kind of implementation is called a *transversal filter*. If the filter taps are all real, we can also consider the factored form of $H(z)$ as

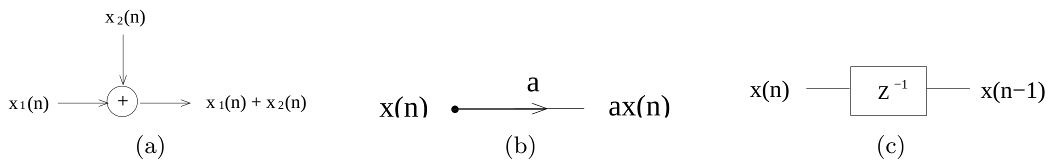


Figure 9.21: Constitutive elements for a filter: (a) Addition, (b) Multiplication, (c) Delay.

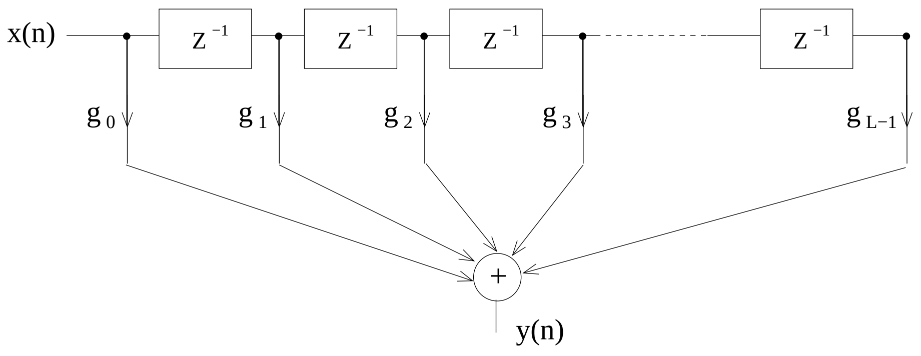


Figure 9.22: Direct FIR implementation.

in (9.9), i.e.

$$H(z) = b_0 \prod_{n=1}^{M_r} (1 - z_n z^{-1}) \prod_{n=1}^{M_c} (1 - \mathrm{Re}\{z_n\} z^{-1} + |z_n|^2 z^{-2})$$

where $M_r + 2M_c = M$. From this representation of the transfer function we can obtain an alternative structure for the FIR called *cascade*, which is shown in Figure 9.24. This cascade form is very important for IIR filters as well, as we will see later. Special optimizations of the FIR structures can be obtained in the the case of symmetric and antisymmetric filters; these are considered in the exercises.

9.5.2 IIR filters structures

For IIR filter, both the a_n's and the b_n's in (9.5) are nonzero. One possible implementation based on the direct form of the transfer function is given in Figure 9.25. This implementation is called *Direct Form I* and it is immediate to see that the C-code implementation at the beginning of the chapter realizes a Direct Form I algorithm.

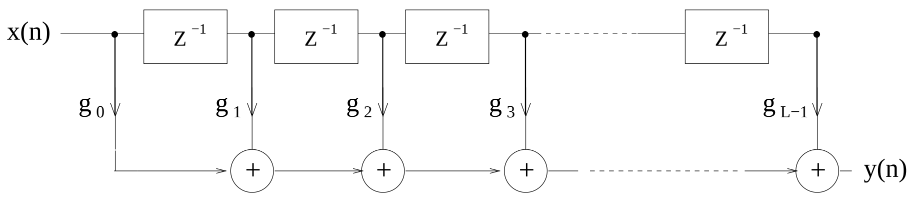


Figure 9.23: Transversal FIR implementation

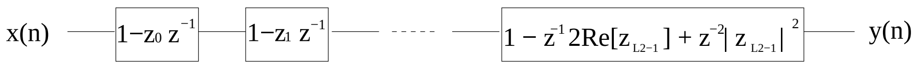


Figure 9.24: Cascade form of a filter.

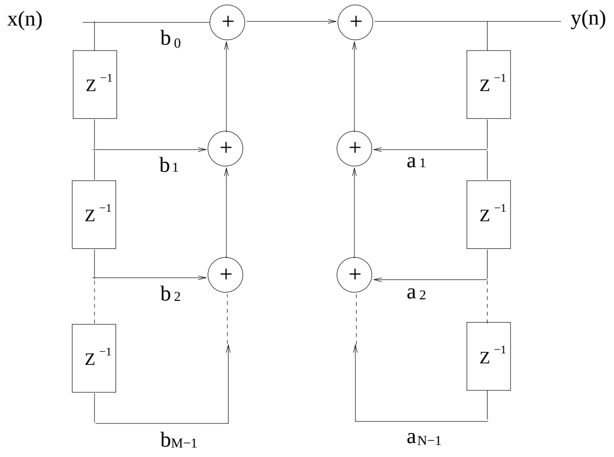


Figure 9.25: Direct Form implementation of an IIR filter.

By the commutative properties of the Z-transform, we can invert the order of computation to turn the Direct Form I structure into the structure shown in Figure 9.26(a); we can then combine the parallel delays together to obtain the structure in Figure 9.26(b). Here, for simplicity, we have assumed $N = M$ but obviously we can set some a_n's or b_n's to zero if this is not the case. This implementation is called *Direct Form II*; its obvious advantage is the reduced number of the required delay elements (hence of memory storage). A particular case, important for what follows, is the second order filter:

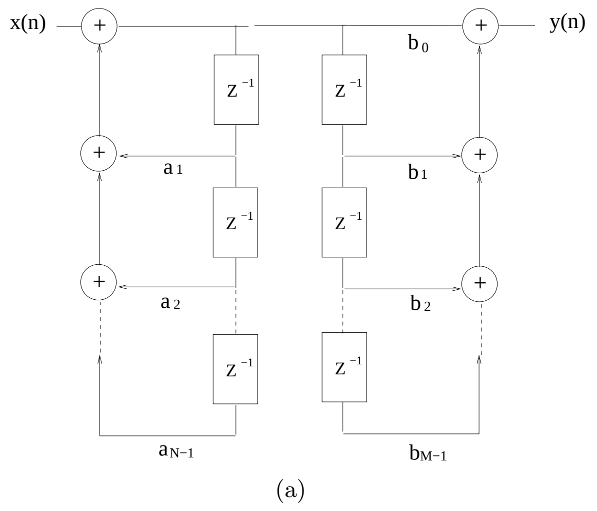
$$H(z) = \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2}}$$

which gives rise to the second order section displayed in Figure 9.27.

Again, for a real valued filter, we can consider the factored form of $H(z)$ as in (9.9). If we combine the complex conjugate poles and zeros, and group the real poles and zeros in twos, we can create a modular structure composed of second order sections. For instance, Figure 9.28 represents a 6^{th} order system.

There are still more possible implementations. For example, if we consider the partial fraction expansion of $H(z)$, we can rewrite the transfer function as the sum

$$H(z) = \sum_n D_n z^{-n} + \sum_n \frac{A_n}{1 - p_n z^{-1}} + \sum_n \frac{B_n + C_n z^{-1}}{(1 - p_n z^{-1})(1 - p_n^* z^{-1})}. \tag{9.47}$$



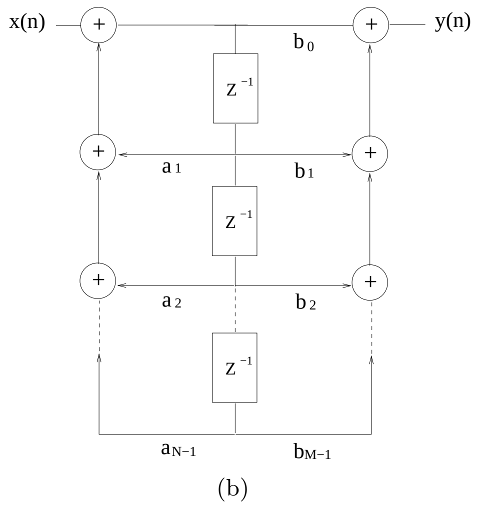


Figure 9.26: IIR filter structures: (a) Direct form I with inverted order. (b) Direct form II.

This generates a parallel structure of filters, whose outputs are summed together. The first branch corresponds to the first sum and it is an FIR filter; a further set of branches are associated to each term in the second sum, each one of them a first order IIR; the last set of branches is a collection of second order sections, one for each term of the third sum.

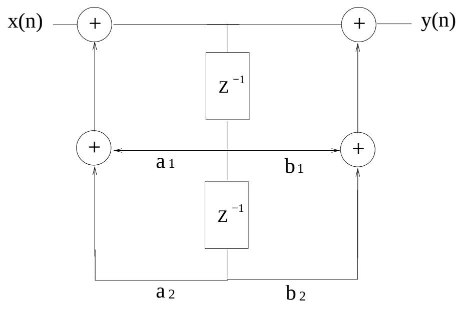


Figure 9.27: Direct Form II implementation for a 2^{nd} order filter with $b_0 = 1$

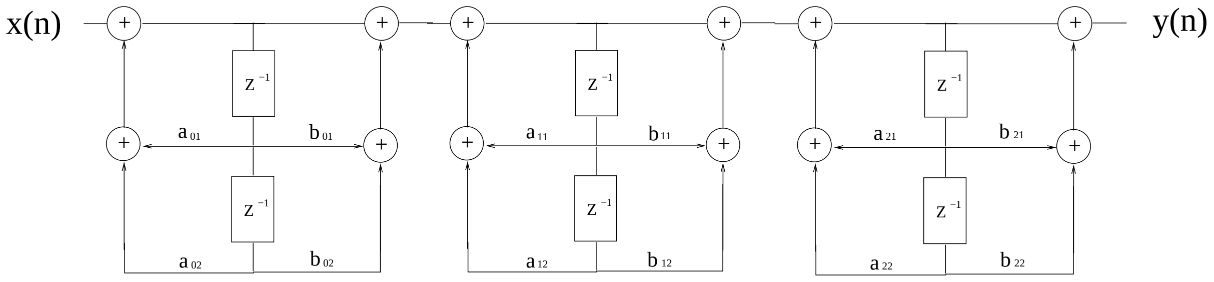


Figure 9.28: 6^{th} order filter implementation.

9.5.3 Some Remarks on Numerical Stability

A very important issue with digital filters is their numerical behavior for a given implementations. Two key questions are:

- Assume the computations are made with (basically) infinite precision but that the filter coefficients are represented internally with finite precision. How good is the resulting filter? Is it still stable?
- If computations are also made with finite precision arithmetic (which implies rounding and truncation error), what is the resulting numerical behavior of the system?

One important difference is that, in the first case, the system is at least guaranteed to be linear; in the second case, however, we can have non-linear effects such as overflows and limit cycles.

Precision and computational issues are very hard to analyze. Here, we will just note that the direct from implementation is more sensible to precision errors than the cascade form, which is why the cascade form is usually preferred in practice. Also, alternative filter structures such as the *lattice* are designed to provide robustness for systems with low numerical precision, albeit at a higher computational cost.

9.6 Problems

Problem 9.1 *Consider the following set of complex numbers*

$$z_k = e^{j\pi(1-2^{-k})} \qquad k = 1, 2, \ldots, M$$

For $M = 4$*,*

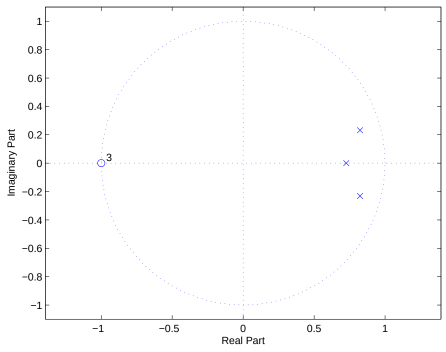
1. *Plot* z_k*,* $k = 1, 2, 3, 4$*, on the complex plane.*
2. *Consider an FIR whose transfer function* $H(z)$ *has the following zeros:*

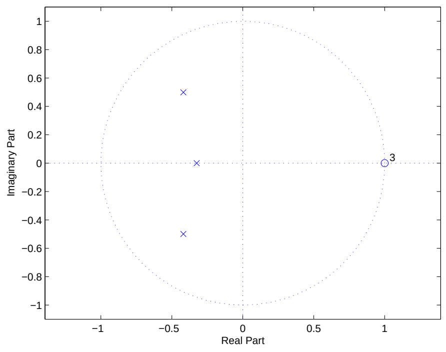
   $$\{z_1, z_2, z_1^*, z_2^*, -1\}$$

   and write out explicitly the expression for $H(z)$*.*
3. *How many nonzero taps will the impulse response* $h[n]$ *have at most?*
4. *Sketch the magnitude of* $H(e^{j\omega})$*.*
5. *What can you say about this filter:*
   (a) *What FIR type is it? (I, II, etc.)*
   (b) *Is it lowpass, bandpass, highpass?*
   (c) *Is it equiripple?*
   (d) *Is this a "good" filter? (By "good" we mean a filter which is close to 1 in the passband, close to zero in the stopband and which has a narrow transition band).*

Problem 9.2 *Transfer functions, zeros and poles Figure 9.29 shows the zeros and poles of three different filters with the unit circle for reference. Each zero is represented with a 'o' and each pole with a 'x' on the plot. Multiple zeros and poles are indicated by the multiplicity number shown to the upper right of the zero or pole.*

Sketch the magnitude of each frequency response and determine the type of filter.


(a) Diagram 1


(b) Diagram 2

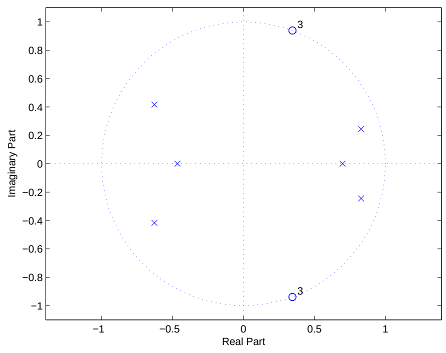

(c) Diagram 3

Figure 9.29: Zeros and Poles Diagrams

Chapter 10

Interpolation and Sampling

In the introduction to these notes we remarked that discrete-time signals are the mathematical model of choice in two signal processing situations: the first, which encompasses the long-established tradition of observing physical phenomena, captures the process of repeatedly measuring the value of a physical quantity at successive instants in time for *analysis* purposes. The second, which is much more recent and dates back to the first digital processors, is the ability to *synthesize* discrete-time signal by means of iterative numerical algorithms.

The repeated measurement of a "natural" signal is called *sampling*; at the base of the notion is a view of the world in which physical phenomena have a potentially infinitely small granularity, in the sense that measurements can be achieved with arbitrary denseness. For this reason, it is customary to model real-world phenomena as *functions of a real variable* (the variable being time or space); defining a quantity over the real line allows for infinitely small subdivisions of the function's domain and therefore infinitely precise localization of its values. We will refer to this model of the world as to the *continuous-time* paradigm. Whether philosophically valid[1] or physically valid[2], the continuous-time paradigm is a model of immense usefulness in the description of analog signal processing systems. So useful, in fact, that even in the completely discrete-time synthesis scenario, we will often find ourselves in the need of converting a sequence to a well defined function of a continuous variable in order to interface our digital world to the analog world outside. The process, which can be seen as the dual of sampling, is called *interpolation*.

[1]Remember Zeno's paradoxes...

[2]The shortest unit of time at which the usual laws of gravitational physics still hold is called *Planck time* and is estimated at 10^{-43} seconds. Apparently, therefore, the universe works in discrete-time...

10.1 Preliminaries and Notation

Interpolation. Interpolation comes into play when discrete-time signals need to be converted to continuous-time signals. The need arises at the interface between the digital world and the analog world; as an example, consider a discrete-time waveform synthesizer which is used to drive an analog amplifier and loudspeaker. In this case, it is useful to express the input to the amplifier as a function of a real variable, defined over the entire real line; this is because the behavior of analog circuitry is best modeled by continuous-time functions. We will see that at the core of the interpolation process is the association of a physical time duration T_s to the intervals between samples of the discrete-time sequence. The fundamental questions concerning interpolation involve the spectral properties of the interpolated function with respect to those of the original sequence.

Sampling.

A typical method to obtain a discrete-time representation of a continuous-time signal is through periodic sampling (*uniform sampling*) where a sequence of samples $x[n]$ are obtained from a continuous-time signal $x_c(t)$ as,

$$x[n] = x_c(nT_s), \qquad -\infty < n < \infty \tag{10.1}$$

where T_s is the sampling period and $F_s = \frac{1}{T_s}$ is the sampling frequency.

A natural question we asked is whether such a sampling process extends a loss of information, i.e. given $\{x[n]\}$, can we reconstruct $x_c(t)$ for any t?

This would mean that we can *interpolate* between values of $\{x_c(nT_s)\}$ to reconstruct $x_c(t)$. If the answer is in the negative (at least for a given class of signals), this means that all the processing tools developed in the discrete-time domain can be applied to continuous-time signals as well, after sampling. The fundamental question is whether this is possible, and if so what are the interpolating functions.

Notation. In the rest of this chapter we will encounter a series of variables which are all interrelated and whose different forms will be used interchangeably according to convenience. They are summarized here for a quick reference:

Name	Description	Units	Relations
T_s	Sampling period	seconds	$T_s = 1/F_s$
F_s	Sampling frequency	Hertz	$F_s = 1/T_s$
Ω_s	Sampling frequency (angular)	rad/sec	$\Omega_s = 2\pi F_s = 2\pi/T_s$
Ω_N	Nyquist frequency (angular)	rad/sec	$\Omega_N = \Omega_s/2 = \pi/T_s$

10.2 Continuous-Time signals

Interpolation and sampling constitute the bridges between the discrete- and continuous-time worlds. Before we proceed to the core of the matter, it is useful to take a quick tour of the main properties of continuous-time signals, which we will simply state without formal proofs.

Continuous-time signals are modeled by complex functions of a real variable t which usually represents time (in seconds) but which can represent other physical coordinates of interest. For maximum generality, no special requirement is imposed on functions modeling signals; just as in the discrete-time case, the functions can be periodic or aperiodic, or they can have a finite support (in the sense that they are nonzero over a finite interval only). A common condition on an aperiodic signal is that its modeling function be square integrable; this corresponds to the reasonable requirement that the signal have finite energy.

Inner product and convolution. We have already encountered some examples of continuous-time signals in conjunction with Hilbert spaces; in section 4.2.2, for instance, we introduced the space of square integrable functions over an interval and, in a short while, we will introduce the space of bandlimited signals. For inner product spaces whose elements are functions on the real line, we will use the following inner product definition:

$$\langle f(t), g(t)\rangle = \int_{-\infty}^{\infty} f^*(t)g(t)dt \tag{10.2}$$

The *convolution* of two real continuous-time signals is defined as usual from the definition of the inner product; in particular

$$\begin{aligned}(f * g)(t) &= \langle f(t-\tau), g(\tau)\rangle &(10.3)\\ &= \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau &(10.4)\end{aligned}$$

The convolution operator in continuous time is linear and time invariant, as can be easily verified. Note that, just like in discrete-time, convolution represents the operation of filtering a signal with a continuous-time LTI filter, whose impulse response is of course a continuous-time function.

Frequency-Domain Representation of Continuous-Time Signals. The Fourier transform of a continuous-time signal $x(t)$ and its inversion formula are defined as[3]:

$$\begin{aligned} X(j\Omega) &= \int_{-\infty}^{\infty} x(t)e^{-j\Omega t}dt && (10.5) \\ x(t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\Omega)e^{j\Omega t}d\Omega && (10.6) \end{aligned}$$

the convergence of the above integrals is assured for functions which satisfy the so-called Dirichlet conditions. In particular, the FT is always well defined for square integrable (finite energy) continuous-time signals. The Fourier transform in continuous time is a linear operator; for a list of its properties, which mirror those we saw for the DTFT, we refer to the bibliography. Suffice here to recall the conservation of energy, also known as Parseval's theorem:

$$\int_{-\infty}^{\infty} |x(t)|^2 dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |X(j\Omega)|^2 d\Omega.$$

The FT representation can be formally extended to signals which are not square summable by means of the Dirac delta notation as we saw in Section 5.2. In particular we have

$$\text{CTFT}\{e^{j\Omega_0 t}\} = \delta(\Omega - \Omega_0) \tag{10.7}$$

from which the Fourier transforms of sine, cosine, and constant functions can be easily derived. Please note that, in continuous-time, the FT of a complex sinusoid is *not* a train of impulses but just a single impulse.

The Convolution Theorem. The convolution theorem for continuous-time signal exactly mirrors the theorem in section 7.5.2; it states that if $h(t) = (f * g)(t)$ then the Fourier transforms of the three signals are related by $H(j\Omega) = F(j\Omega)G(j\Omega)$. In particular we can use the convolution theorem to compute

$$(f * g)(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(j\Omega)G(j\Omega)e^{j\Omega t}d\Omega \tag{10.8}$$

10.3 Bandlimited Signals

A signal whose Fourier transform is nonzero only over a finite frequency interval is called *bandlimited*. In other words, the signal $x(t)$ is bandlimited if there exists a frequency Ω_N

[3]The notation $X(j\Omega)$ mirrors the specialized notation we used for the DTFT; in this case, by writing $X(j\Omega)$ we indicate that the Fourier transform is just the (two-sided) Laplace transform $X(s) = \int x(t)e^{-st}dt$ computed on the imaginary axis.

such that[4]

$$X(j\Omega) = 0 \quad \text{for } |\Omega| \geq \Omega_N.$$

Such a signal will be called Ω_N-bandlimited and Ω_N is often called the *Nyquist frequency.* It may be useful to mention that, symmetrically, a continuous-time signal which is nonzero over a finite time interval only is called a *time-limited* signal (or finite-support signal). A fundamental theorem states that a bandlimited signal cannot be time-limited, and vice versa. While this can be proved formally without too much effort, here we simply give the intuition behind the statement. The time-scaling property of the Fourier transform states that

$$\text{CTFT}\{f(at)\} = \frac{1}{a}\,F(j\frac{\Omega}{a})$$

so that the more "compact" in time a signal is, the wider it frequency support becomes.

The Sinc Function. Let us now consider a prototypical Ω_N-bandlimited signal $\varphi(t)$ whose Fourier transform is *constant* over the interval $[-\Omega_N, \Omega_N]$ and zero everywhere else. If we define the rect function as (see also section 7.7.1):

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

we can express the Fourier transform of the prototypical Ω_N-bandlimited signal as

$$\Phi(j\Omega) = \frac{\pi}{\Omega_s}\text{rect}\left(\frac{\Omega}{2\Omega_N}\right) \tag{10.9}$$

where the leading factor is just a normalization term. The time-domain expression for the signal is easily obtained from the inverse Fourier transform as

$$\varphi(t) = \frac{\sin\Omega_N t}{\Omega_N t} = \text{sinc}\left(\frac{t}{T_s}\right) \tag{10.10}$$

where we have used $T_s = \pi/\Omega_N$ and defined the sinc function as

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

The sinc function is plotted in Figure 10.1.

Note the following:

[4]The use of $\geq$ instead of $>$ is a technicality which will be useful in conjunction with the sampling theorem below.

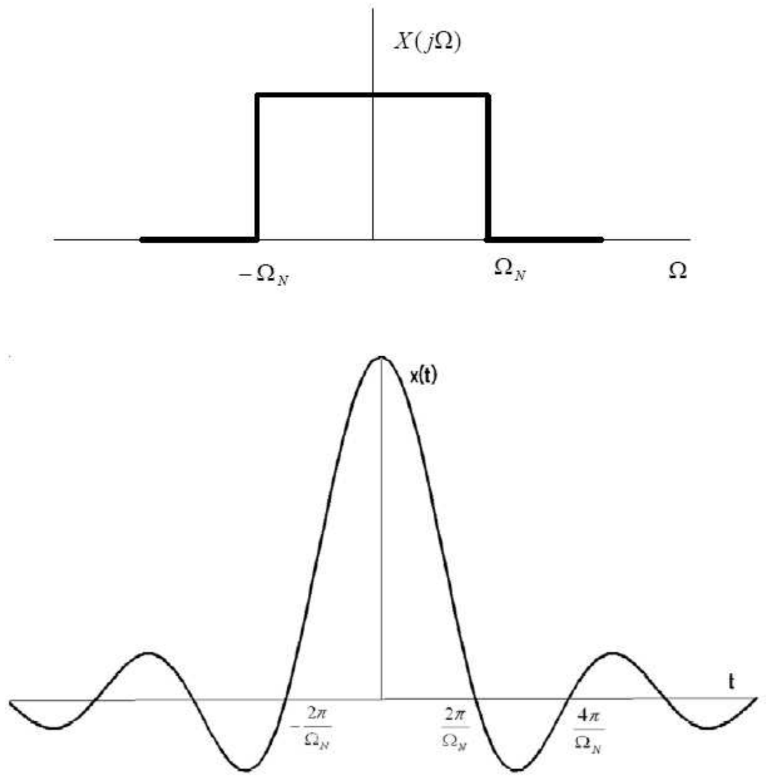


Figure 10.1: The sinc function in Frequency ($X(j\Omega)$) and Time ($x(t)$) domains.

- The function is symmetric, $\text{sinc}(x) = \text{sinc}(-x)$
- The sinc function is zero for all integer values of its argument, except in zero. This feature is called the *interpolation property* of the sinc, as we will see more in detail later.
- The sinc function is square integrable (it has finite energy) but it is not absolutely integrable (hence the discontinuity of its Fourier transform).
- The decay is slow, asymptotic to $1/x$.
- The scaled sinc function represents the impulse response of an ideal, continuous-time lowpass filter with cutoff frequency Ω_N.

10.4 The Sampling Theorem

We have seen in the previous section that the "natural" polynomial interpolation scheme leads to the so-called sinc interpolation for infinite discrete time sequences. Another way to look at the previous result is that any square summable discrete-time signal can be interpolated into a continuous-time signal which is smooth in time and strictly bandlimited in frequency. This suggests that the class of bandlimited functions must play a special role in bridging the gap between discrete and continuous time and this deserves further investigation. In particular, since any discrete-time signal can be interpolated exactly into a bandlimited function, we now ask ourselves whether the converse is true: can any bandlimited signal be transformed into a discrete-time signal with no loss of information?

10.4.1 Frequency-domain representation of sampling

Given a continuous-time signal $x_c(t)$, we do periodic sampling by producing

$$x[n] = x_c(nT_s) = x_c(t)\big|_{t=nT_s}. \tag{10.11}$$

Let us define a new continuous-time signal which places Dirac delta impulses at the sampling locations, i.e.

$$x_s(t) = \sum_n x[n]\delta(t-nT_s) = \sum_n x_c(nT_s)\delta(t-nT_s), \tag{10.12}$$

which is a fictitious signal serving as an intermediate step between the continuous and discrete-time worlds.

We can also write

$$x_s(t) = \sum_n x_c(nT_s)\delta(t-nT_s) = x_c(t)\sum_n \underbrace{\delta(t-nT_s)}_{s(t)}, \tag{10.13}$$

i.e.

$$x_s(t) = x_c(t)s(t). \tag{10.14}$$

Hence we see that from the modulation property of continuous-time Fourier transforms,

$$X_s(j\Omega) = \underbrace{\frac{1}{2\pi}\int_{-\infty}^{+\infty} X_c(j\theta)S\left(j(\Omega-\theta)\right)\,\mathrm{d}\theta}_{X_c(j\Omega)*S(j\Omega)}. \tag{10.15}$$

Now

$$\sum_n \delta(t-nT_s) \overset{\text{CTFT}}{\Longleftrightarrow} \frac{2\pi}{T_s}\sum_{k=-\infty}^{+\infty}\delta(\Omega-k\Omega_s), \tag{10.16}$$

where $\Omega_s = \frac{2\pi}{T_s}$. Using this in (10.15) we see that

$$X_s(j\Omega) = \frac{1}{2\pi} X_c(j\Omega) * S(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c\left(j(\Omega - k\Omega_s)\right). \tag{10.17}$$

Therefore, we see that the sampled sequence has a Fourier transform which consists of periodically repeated copies of the original CTFT of $x_c(t)$, shifted by integer multiples and superimposed.

To observe its effects, see Figure 10.2 representing a bandlimited Fourier transform with bandwidth Ω_N, Figure 10.3 is the periodic impulse train $S(j\Omega)$ and finally Figure 10.4 is $X_s(j\Omega)$ along with $X\left(e^{j\omega}\right)$ in Figure 10.5. From Figure 10.4 it is clear that to retain information through sampling we need

$$\Omega_s - \Omega_N > \Omega_N \qquad \text{or} \qquad \Omega_s > 2\Omega_N, \tag{10.18}$$

so that the replicas of $X_c(j\Omega)$ do not overlap when they are added together in (10.17). If this condition is satisfied, it is clear that one can recover $x_c(t)$ from $x[n]$ (or $X_c(j\Omega)$ from $X\left(e^{j\omega}\right)$) by taking the inverse CTFT of one of the replicas, i.e by taking

$$X_r(j\Omega) = H_r(j\Omega) X_s(j\Omega), \tag{10.19}$$

where

$$H_r(j\Omega) = \begin{cases} T_s & |\Omega| \leq \Omega_c \\ 0 & \text{else}, \end{cases} \tag{10.20}$$

$$\Omega_N \leq \Omega_c \leq \Omega_s - \Omega_N.$$

This leads us to the sampling theorem:

If $x_c(t)$ is a bandlimited signal with $X_c(j\Omega) = 0, \quad |\Omega| > \Omega_N$, then $x_c(t)$ is uniquely determined by its samples $x[n] = x_c(nT_s)$, if $\Omega_s = \frac{2\pi}{T_s} \geq 2\Omega_N$.

This gives us an idea on how to reconstruct the original signal from the samples using (10.19). We use (10.19) to see that

$$\begin{aligned} x_r(t) &= h_r(t) * x_s(t) = \int_\tau h_r(t-\tau) x_s(\tau)\, \mathrm{d}\tau \\ &= \int_\tau \sum_n \delta(\tau - nT_s) x_c(nT_s) h_r(t-\tau)\, \mathrm{d}\tau \\ &= \sum_n x_c(nT_s) \int_\tau h_r(t-\tau) \delta(\tau - nT_s)\, \mathrm{d}\tau \end{aligned}$$

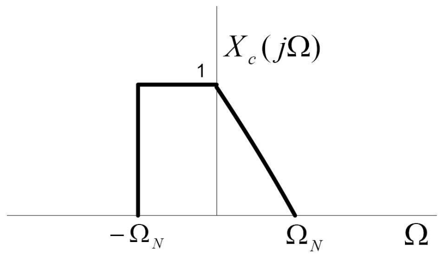


Figure 10.2: $X_c(j\Omega)$

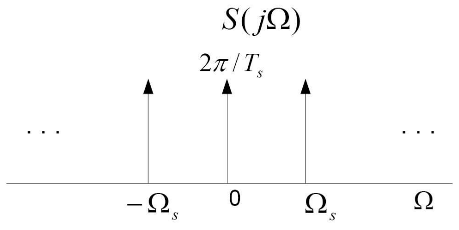


Figure 10.3: $S(j\Omega)$

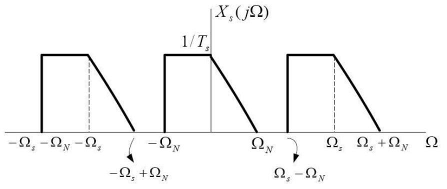


Figure 10.4: $X_s(j\Omega)$

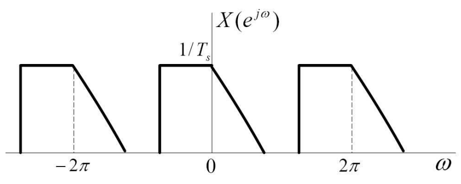


Figure 10.5: $X(e^{j\omega})$

$$\longrightarrow x_r(t) = \sum_n x_c(nT_s) h_r(t - nT_s), \tag{10.21}$$

where $h_r(t)$ is the inverse CTFT of $H_r(j\Omega)$. The form of (10.21) shows the underlying operation as one of *interpolating* between the sampled values. This point-of-view will be developed next in an alternate proof of the sampling theorem in terms of Hilbert spaces and bases functions. Finally note that since we choose $\Omega_s \geq 2\Omega_N$, we have perfect reconstruction, i.e.

$$x_r(t) = x_c(t).$$

10.5 The Space of Bandlimited Signals.

The class of Ω_N-bandlimited functions of finite energy forms a Hilbert space, with the inner product defined in (10.2). An orthogonal basis for the space of Ω_N-bandlimited functions can be obtained easily from the prototypical bandlimited function, the sinc; indeed, consider the family

$$\varphi^{(n)}(t) = \text{sinc}\left(\frac{t - nT_s}{T_s}\right), \qquad n \in \mathbb{Z} \tag{10.22}$$

where, once again, $T_s = \pi/\Omega_N$. Note that we have $\varphi^{(n)}(t) = \varphi^{(0)}(t - nT_s)$ so that each basis function is simply a translated version of the prototype basis function $\varphi^{(0)}$. Orthogonality can be easily proved as follows: first of all, because of the symmetry of the sinc function and the time-invariance of the convolution, we can write

$$\begin{aligned} \langle \varphi^{(n)}(t), \varphi^{(m)}(t) \rangle &= \langle \varphi^{(0)}(t - nT_s), \varphi^{(0)}(t - mT_s) \rangle \\ &= \langle \varphi^{(0)}(nT_s - t), \varphi^{(0)}(mT_s - t) \rangle \\ &= (\varphi^{(0)} * \varphi^{(0)})((n-m)T_s) \end{aligned}$$

We can now apply the convolution theorem and (10.9) to obtain

$$\begin{aligned} \langle \varphi^{(n)}(t), \varphi^{(m)}(t) \rangle &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \left(\frac{\pi}{\Omega_N} \text{rect}\left(\frac{\Omega}{\Omega_N}\right) \right)^2 e^{j\Omega(n-m)T_s} d\Omega \\ &= \frac{\pi}{2\Omega_N^2} \int_{-\Omega_N}^{\Omega_N} e^{j\Omega(n-m)T_s} d\Omega \\ &= \begin{cases} \dfrac{\pi}{\Omega_N} = T_s & \text{if } n = m \\ 0 & \text{if } n \neq m \end{cases} \end{aligned}$$

so that $\{\varphi^{(n)}(t)\}_{n\in\mathbb{Z}}$ is orthogonal with normalization factor Ω_N/π (or, equivalently, $1/T_s$).

In order to show that the space of Ω_N-bandlimited functions is indeed a Hilbert space, we should also prove that the space is complete. This is a more delicate notion to show[5] and here it will simply be assumed.

10.5.1 Sampling as a Basis Expansion.

Now that we have an orthogonal basis, we can compute coefficients in the basis expansion of an arbitrary Ω_N-bandlimited function $x(t)$. We have

$$\begin{aligned}
\langle\varphi^{(n)}(t), x(t)\rangle &= \langle\varphi^{(0)}(t-nT_s), x(t)\rangle && (10.23)\\
&= (\varphi^{(0)} * x)(nT_s) && (10.24)\\
&= \frac{1}{2\pi}\int_{-\infty}^{\infty}\frac{\pi}{\Omega_N}\mathrm{rect}\left(\frac{\Omega}{\Omega_N}\right)X(j\Omega)e^{j\Omega nT_s}d\Omega && (10.25)\\
&= \frac{\pi}{\Omega_N}\ \frac{1}{2\pi}\int_{-\Omega_N}^{\Omega_N}X(j\Omega)e^{j\Omega nT_s}d\Omega && (10.26)\\
&= T_s\ x(nT_s) && (10.27)
\end{aligned}$$

in the derivation we have first rewritten the inner product as a convolution operation, then we have applied the convolution theorem, and recognized the penultimate line as simply the inverse CTFT of $X(j\Omega)$ calculated in $t = nT_s$. We therefore have the remarkable result that the n-th basis expansion coefficient is *proportional to the sampled value of $x(t)$* at $t = nT_s$. For this reason, the sinc basis expansion is also called *sinc sampling.*

Reconstruction of $x(t)$ from its projections can now be achieved via the orthonormal basis reconstruction formula (4.40); since the sinc basis is just orthogonal rather than orthonormal, (4.40) needs to take into account the normalization factor and we have:

$$\begin{aligned}
x(t) &= \frac{1}{T_s}\sum_{n=-\infty}^{\infty}\langle\varphi^{(n)}(t), x(t)\rangle\,\varphi^{(n)}(t)\\
&= \sum_{n=-\infty}^{\infty}x(nT_s)\mathrm{sinc}\left(\frac{t-nT_s}{T_s}\right) && (10.28)
\end{aligned}$$

which corresponds to the interpolation formula (10.37).

The Sampling Theorem. *If $x(t)$ is a Ω_N-bandlimited continuous-time signal, a* sufficient *representation of $x(t)$ is given by the discrete-time signal $x[n] = x(nT_s)$, with $T_s = \pi/\Omega_N$.*

[5]Completeness of the sinc basis can be proven as a consequence of the completeness of the Fourier basis in the continuous-time domain.

The continuous time signal $x(t)$ can be exactly reconstructed from the discrete-time signal $x[n]$ as:

$$x(t) = \sum_{n=-\infty}^{\infty} x[n]\text{sinc}\left(\frac{t - nT_s}{T_s}\right).$$

A few notes:

- The proof of the theorem is inherent to the properties of the Hilbert space of bandlimited functions, and it is trivial after having proved the existence of an orthogonal basis.
- Clearly, if a signal is Ω_N-bandlimited, then it is also Ω-bandlimited for all $\Omega \geq \Omega_N$. Therefore, an Ω_N-bandlimited signal $x(t)$ is uniquely represented by all sequences $x[n] = x(nT)$ for which $T \leq T_s = \pi/\Omega_N$; T_s is the largest sampling period which guarantees perfect reconstruction (i.e. we cannot take fewer than $1/T_s$ samples per second).
- Another way to state the above point is to say that the *minimum* sampling frequency Ω_s for perfect reconstruction is exactly twice the Nyquist frequency, where the Nyquist frequency is the highest frequency of the bandlimited signal; the sampling frequency Ω must therefore satisfy the relationship:

  $$\Omega \geq \Omega_s = 2\Omega_N$$

  or, in Hertz,

  $$F \geq F_s = 2F_N$$

10.5.2 Examples for the sampling theorem

We have seen that if a signal has a maximum frequency of $f_{\max}$, then sampling at a rate $f_s \geq 2f_{\max}$ is sufficient to retain all the information in the samples. Moreover, we can recover the original continuous-time signal from its samples using sinc interpolation.

Example 10.1 *Let $x_c(t) = \cos(4000\pi t) = \cos[2\pi(2000)]\,t$, which is shown in Fig. 10.6.*

$$X_c(j\Omega) = \pi\delta(\Omega - 4000\pi) + \pi\delta(\Omega + 4000\pi).$$

Thus $f_{\max} = 2000$ for this case and we need $f_s \geq 4000$ as the sampling rate. Let $f_s = \frac{1}{T_s} = 6000$, $\Omega_s = 2\pi f_s$.

$$x[n] = x_c(nT_s) = \cos(2\pi 2000 nT_s) = \cos\left(2\pi\frac{2000}{6000}n\right) = \cos\left(\frac{2\pi}{3}n\right).$$

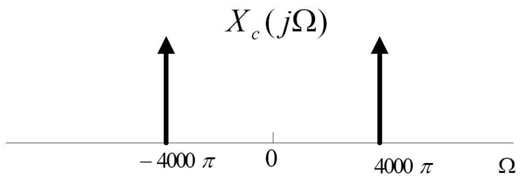
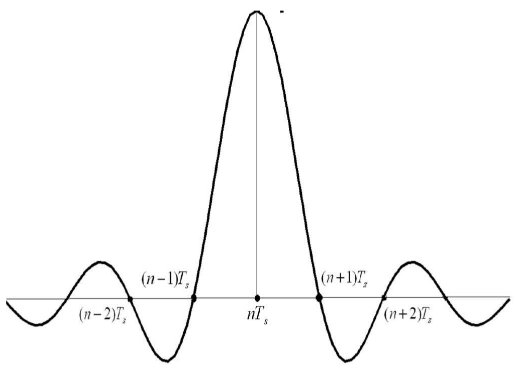


Figure 10.6: $X_c(j\Omega)$

Now for reconstruction we get

$$\hat{x}_c(t) = \sum_{n=-\infty}^{+\infty} x[n]\mathrm{sinc}\left(\frac{t-nT_s}{T_s}\right) = \sum_{n=-\infty}^{+\infty} \cos\left(\frac{2\pi}{3}n\right)\frac{\sin\pi(6000t-n)}{\pi(6000t-n)}.$$



Let us look at this pictorially (Fig. 10.7).

$$\hat{X}_c(j\Omega) = H_r(j\Omega)X_s(j\Omega) = X_c(j\Omega).$$

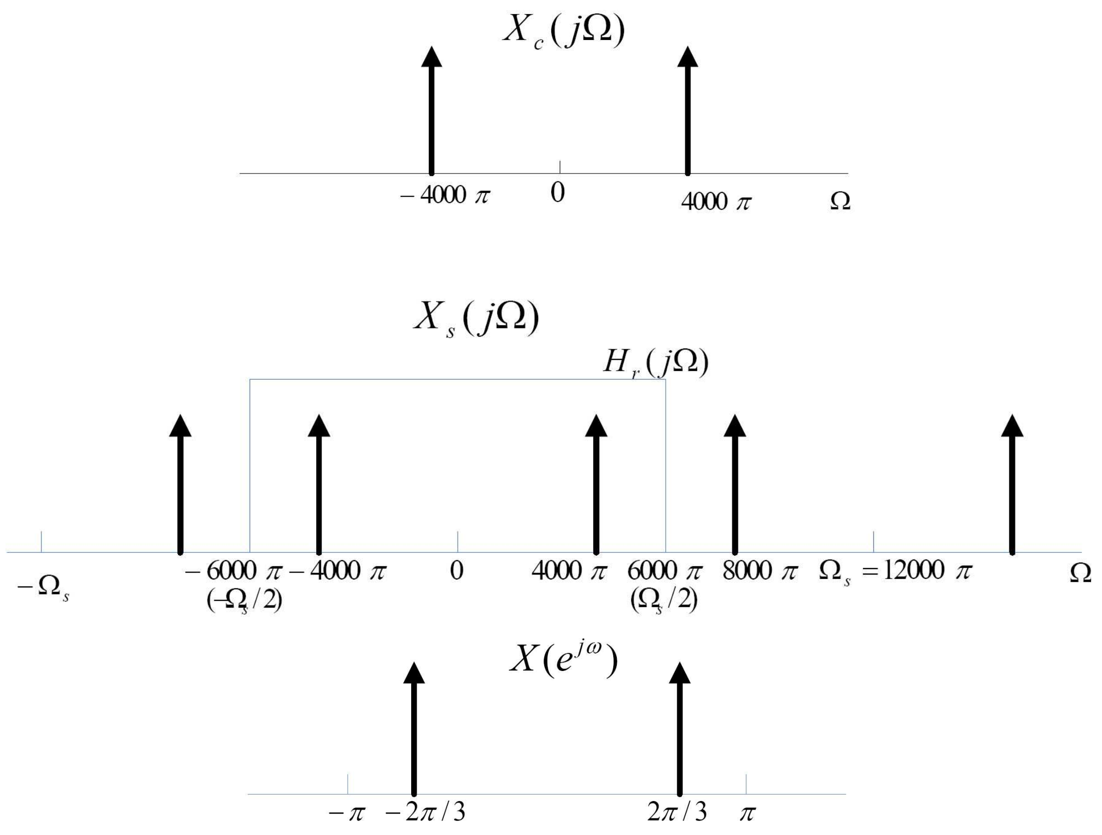


Figure 10.7: Pictorial representation of sampling of $x_c(t) = \cos(4000\pi t)$

Now, if $f_s = 1500 < 4000$*, then* $\Omega_s = 2\pi f_s = 3000\pi$*.*
But $\hat{x}_c(t) = \cos(1000\pi t) \neq \cos(4000\pi t) = x_c(t)$*.* $x[n] = \cos\frac{2\pi}{3}n$*, same as before! Fig. 10.8 shows the sampling and reconstruction in this case.*

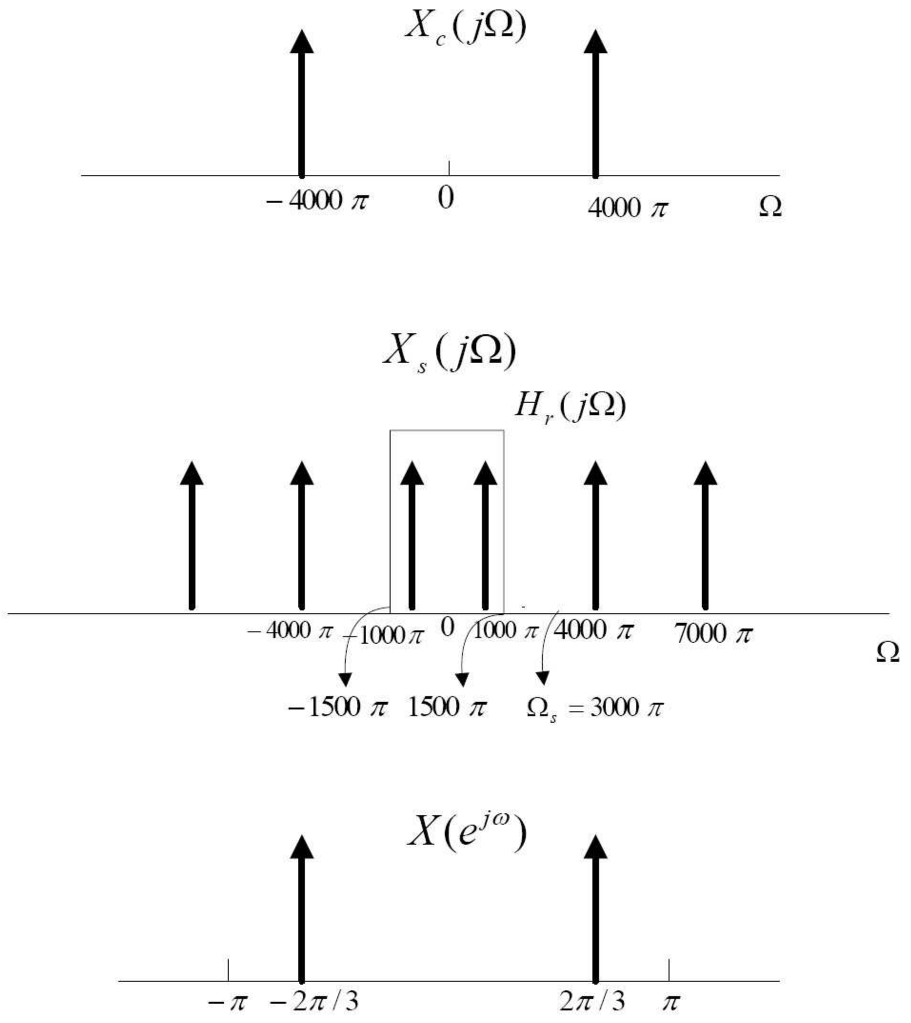


Figure 10.8: $X_c(j\Omega)$, $X_s(j\Omega)$, and $X\left(e^{j\omega}\right)$.

Example 10.2 *Fig. 10.9 shows signal $X_c\left(j\Omega\right)$ and its sampled version.*

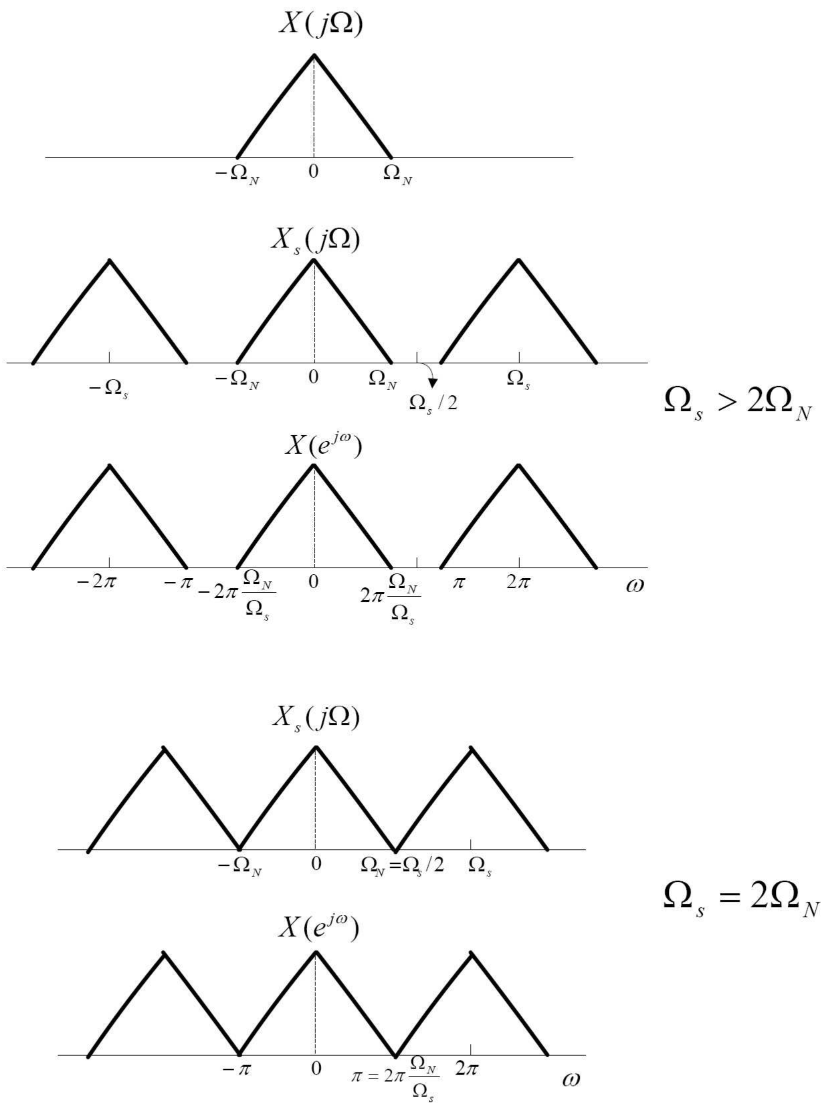


Figure 10.9: Example. 10.2

10.6 Interpolation

Interpolation is a procedure whereby we convert a discrete-time sequence $x[n]$ to a continuous-time function $x(t)$. Since this can be done in an arbitrary number of ways, we have to start by formulating some requirements on the resulting signal. At the heart of the interpolating procedure, as we have mentioned, is the association of a physical time duration T_s to the interval between the samples in the discrete-time sequence. An intuitive requirement on the interpolated function is that its values at multiples of T_s be equal to the corresponding points of the discrete-time sequence, i.e.

$$x(t)|_{t=nT_s} = x[n];$$

the interpolation problem now reduces to "filling the gaps" between these instants.

10.6.1 Local Interpolation

The simplest interpolation schemes create a continuous-time function $x(t)$ from a discrete-time sequence $x[n]$ by setting $x(t)$ to be equal to $x[n]$ for $t = nT_s$ and by setting $x(t)$ to be some linear combination of neighboring sequence values when t lies in between interpolation instants. In general, the local interpolation schemes can be expressed by the following formula:

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] I(\frac{t - nT_s}{T_s}) \tag{10.29}$$

where $I(t)$ is called the interpolation function (for linear functions the notation $I_N(t)$ is used and the subscript N indicates how many discrete-time samples besides the current one enter in the computation of the interpolated values for $x(t)$). The interpolation function must satisfy the fundamental *interpolation properties*:

$$\begin{cases} I(0) = 1 \\ I(k) = 0 \text{ for } k \in \mathbb{Z} \setminus \{0\} \end{cases} \tag{10.30}$$

where the second requirement implies that, no matter what the support of $I(t)$ is, its values should not affect other interpolation instants. By changing the function $I(t)$, we can change the type of interpolation and the properties of the interpolated signal $x(t)$.

Note that (10.29) can be interpreted either simply as a linear combination of shifted interpolation functions or, more interestingly, as a "mixed domain" convolution product, where we are convolving a discrete-time signal $x[n]$ with a continuous-time "impulse response" $I(t)$ scaled in time by the interpolation period T_s.

Zero-Order Hold. The simplest approach for the interpolating function is the piecewise-constant interpolation; here the continuous-time signal is kept constant between discrete sample values, yielding:

$$x(t) = x[n] \quad \text{for } (n - \frac{1}{2})T_s \le t < (n + \frac{1}{2})T_s$$

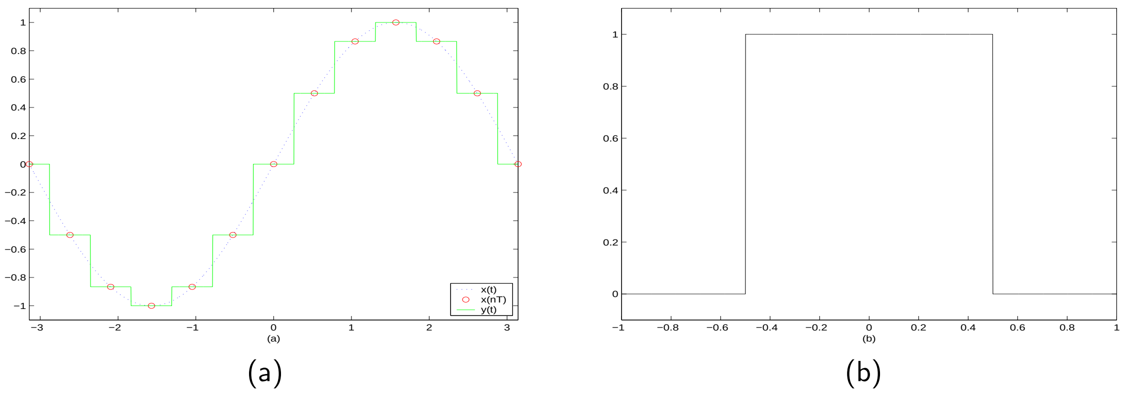


Figure 10.10: Interpolation with zero-order hold. (a) Interpolation of the samples of a sinusoid. Note the discontinuities introduced by this simple scheme. (b) The rect function can be used to describe mathematically the zero-order hold.

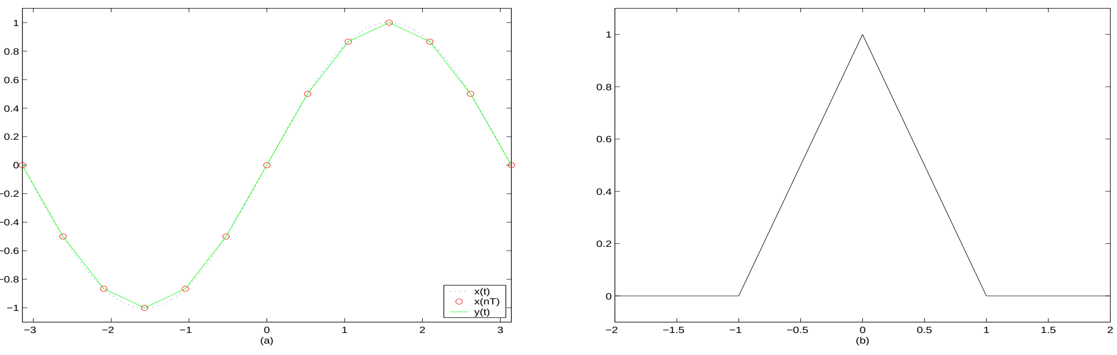
and an example is shown in Figure 10.10(a); it is apparent that the resulting function is far from smooth since the interpolated function is discontinuous. The interpolation function is simply:

$$I_0(t) = \mathrm{rect}(t)$$

and the values of $x(t)$ depend only on the current discrete-time sample value.

First-Order Hold. A linear interpolator (sometimes called a first-order hold) simply connects the points corresponding to the samples with straight lines. An example is shown in Figure 10.11(a); note that now $x(t)$ depends on two consecutive discrete-time samples, across which a connecting straight line is drawn. From the point of view of smoothness, this interpolator already represents a good improvement over the zero-order hold: indeed the interpolated function is now continuous, although its first derivative is not. The first-order hold can be expressed in the same notation as in (10.29) by defining the following triangular function

$$I_1(t) = \begin{cases} 1 - |t| & \text{if } |t| < 1 \\ 0 & \text{otherwise} \end{cases}$$



(a) (b)

Figure 10.11: Linear interpolation (also called first-order hold). (a) Interpolation of the samples of a sinusoid using linear interpolation. (b) The triangular function is the interpolating function corresponding to the linear interpolation.

which is shown in Figure 10.11(b)[6]. It is immediate to verify that $I_1(t)$ satisfies the interpolation properties (10.30).

Higher-Order Interpolators. The zero- and first-order interpolators are widely used in practical circuits due to their extreme simplicity. These schemes can be extended to higher order interpolation functions and, in general, $I_N(t)$ will be a N-th order polynomial in t. The advantage of the local interpolation schemes is that, for small N, they can be easily implemented in practice as *causal* interpolation schemes (locality is akin to FIR filtering); their disadvantage is that, because of the locality, their N-th derivative will be discontinuous. This discontinuity represents a lack of smoothness in the interpolated function; from a spectral point of view this corresponds to a high frequency energy content, which is usually undesirable.

10.6.2 Polynomial Interpolation

The lack of smoothness of local interpolations is easily eliminated when we need to interpolate just a *finite* number of discrete-time samples. In fact, in this case the task becomes a classic polynomial interpolation problem for which the optimal solution has been known

[6]Note that $I_1(t) = (I_0 * I_0)(t)$.

for a long time under the name of *Lagrange interpolation.* Note that a polynomial interpolating a finite set of samples is a maximally smooth function in the sense that it is continuous together with all its derivatives.

Consider a length $(2N+1)$ discrete-time signal $x[n]$, with n= $-N,\ldots,N$. Associate to each sample an abscissa $t_n = nT_s$; we know from basic algebra that there is one and only one polynomial $P(t)$ of degree $2N$ which passes through all the $2N+1$ pairs $(t_n, x[n])$ and this polynomial is the Lagrange interpolator. The coefficients of the polynomial could be found by solving the set of $2N+1$ equations:

$$\{P(t_n) = x[n]\}_{n=-N,\ldots,N} \tag{10.31}$$

but a simpler way to determine the expression for $P(t)$ is to use the set of $2N+1$ *Lagrange polynomials* of degree $2N$:

$$\begin{aligned} L_n^{(N)}(t) &= \prod_{\substack{k=-N \\ k\neq n}}^{N} \frac{(t-t_k)}{(t_n-t_k)} \\ &= \prod_{\substack{k=-N \\ k\neq n}}^{N} \frac{t/T_s - k}{n-k} \qquad n=-N,\ldots,N \end{aligned} \tag{10.32}$$

The polynomials $L_n^{(N)}(t)$ for $T_s = 1$ and $N = 2$ (i.e. interpolation of 5 points) are plotted in Figure 10.12-(a). By using this notation, the *global* Lagrange interpolator for a given set of abscissa/ordinate pairs can now be written as a simple linear combination of Lagrange polynomials:

$$P(t) = \sum_{n=-N}^{N} x[n] L_n^{(N)}(t) \tag{10.33}$$

and it is easy to verify that this is the unique interpolating polynomial of degree $2N$ in the sense of (10.31). Note that *each* of the $L_n^{(N)}(t)$ satisfies the interpolation properties (10.30) or, concisely (for $T_s = 1$):

$$L_n^{(N)}(m) = \delta[n-m];$$

the interpolation formula, however, cannot be written in the form of (10.29) since the Lagrange polynomials are not simply shifts of a single prototype function. The continuous time signal $x(t) = P(t)$ is now a *global* interpolating function for the finite-length discrete-time signal $x[n]$, in the sense that it depends on *all* samples in the signal; as a consequence, $x(t)$ is maximally smooth ($x(t) \in C^\infty$). An example of Lagrange interpolation for $N = 2$ is plotted in Figure 10.12-(b).

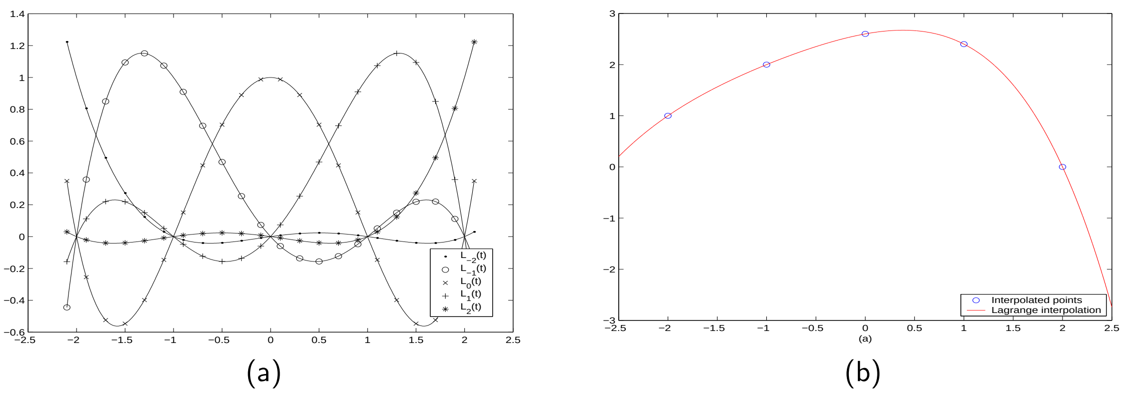


Figure 10.12: Lagrange interpolation. (a) The polynomials $L_n^{(N)}(t)$ used to compute the interpolation for $N = 2$ and $T = 1$. Note that $L_n^{(N)}(m)$ is zero except for $m = n$, where it is 1. (b) Interpolation using 5 points.

10.6.3 Sinc Interpolation

The beauty of local interpolation schemes lies in the fact that the interpolated function is simply a linear combination of shifted versions of the *same* prototype interpolation function $I(t)$; this unfortunately has the disadvantage of creating a continuous-time function which lacks smoothness. Polynomial interpolation, on the other hand, is perfectly smooth but it only works in the finite-length case and it requires different interpolation functions with different signal lengths. Yet, both approaches can come together in a nice mathematical way and we are now ready to introduce the maximally smooth interpolation scheme for infinite discrete-time signals.

Let us take the expression for the Lagrange polynomial of degree N in (10.32) and consider its limit for N going to infinity. We have:

$$\begin{aligned}
\lim_{N\to\infty} L_n^{(N)}(t) &= \prod_{\substack{k=-\infty\\k\neq n}}^{\infty} \frac{t/T_s - k}{n-k} \\
&= \prod_{\substack{m=-\infty\\m\neq 0}}^{\infty} \frac{t/T_s - n + m}{m} \\
&= \prod_{\substack{m=-\infty\\m\neq 0}}^{\infty} \left(1 + \frac{t/T_s - n}{m}\right) \\
&= \prod_{m=1}^{\infty} \left(1 - \left(\frac{t/T_s - n}{m}\right)^2\right) \qquad (10.34)
\end{aligned}$$

$$(10.35)$$

where we have used the change of variable $m = n - k$. We can now invoke Euler's infinite product expansion for the sine function

$$\sin(\pi\tau) = (\pi\tau)\prod_{k=1}^{\infty}\left(1 - \frac{\tau^2}{k^2}\right)$$

(whose derivation is in the appendix) to finally obtain

$$\lim_{N\to\infty} L_n^{(N)}(t) = \operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right) \qquad (10.36)$$

The convergence of the Lagrange polynomial $L_0^{(N)}(t)$ to the sinc function is illustrated in Figure 10.13. Note that now, as the number of points becomes infinite, the Lagrange polynomials converge to shifts of the *same* prototype function, i.e. the sinc; therefore the interpolation formula can be expressed as in (10.29) with $I(t) = \operatorname{sinc}(t)$; indeed, if we consider an infinite sequence $x[n]$ and apply the Lagrange interpolation formula (10.33) we obtain:

$$x(t) = \sum_{n=-\infty}^{\infty} x[n]\operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right) \qquad (10.37)$$

Spectral Properties of the Sinc Interpolation. The sinc interpolation of a discrete-time sequence gives rise to a strictly bandlimited continuous-time function. If the DTFT $X(e^{j\omega})$

of the discrete-time sequence exists, the spectrum of the interpolated function $X(j\Omega)$ can be obtained as follows:

$$\begin{aligned} X(j\Omega) &= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[n]\mathrm{sinc}\left(\frac{t-nT_s}{T_s}\right) e^{-j\Omega t} dt \\ &= \sum_{n=-\infty}^{\infty} x[n] \int_{-\infty}^{\infty} \mathrm{sinc}\left(\frac{t-nT_s}{T_s}\right) e^{-j\Omega t} dt \end{aligned}$$

now we use (10.9) to get the Fourier Transform of the scaled and shifted sinc

$$= \sum_{n=-\infty}^{\infty} x[n] \left(\frac{\pi}{\Omega_N}\right) \mathrm{rect}\left(\frac{\Omega}{2\Omega_N}\right) e^{-jnT_s\Omega}$$

and use the fact that, as usual, $T_s = \pi/\Omega_N$

$$\begin{aligned} &= \left(\frac{\pi}{\Omega_N}\right) \mathrm{rect}\left(\frac{\Omega}{2\Omega_N}\right) \sum_{n=-\infty}^{\infty} x[n] e^{-j\pi(\Omega/\Omega_N)n} \\ &= \begin{cases} (\pi/\Omega_N) X(e^{j\pi\Omega/\Omega_N}) & \text{for } |\Omega| \le \Omega_N \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

In other words, the continuous-time spectrum is just a scaled and stretched version of the DTFT of the discrete-time sequence between $-\pi$ and π. The duration of the interpolation interval T_s is inversely proportional to the resulting bandwidth of the interpolated signal. Intuitively, a slow interpolation (T_s large) will result in a spectrum concentrated around the low frequencies; conversely, a fast interpolation (T_s small) will result in a spread-out spectrum (more high frequencies are present)[7].

10.7 Aliasing

The "naive" notion of sampling, as we have seen, is associated to the very practical idea of measuring the instantaneous value of a continuous-time signal at uniformly spaced instants in time. For bandlimited signals, we have seen that this is actually equivalent to an orthogonal decomposition in the space of bandlimited functions, which guarantees that the set of samples $x(nT_s)$ uniquely determines the signal and allows its perfect reconstruction. We now want to address the following question: what happens if we simply sample an *arbitrary* time signal in the "naive" sense (i.e. in the sense of simply taking $x[n] = x(nT_s)$) and what can we reconstruct from the set of samples thus obtained?

[7]To find a simple everyday analogy, think of a 45rpm vinyl record played at either 33rpm (slow interpolation) or at 78rmp (fast interpolation) and remember the acoustic effect on the sounds.

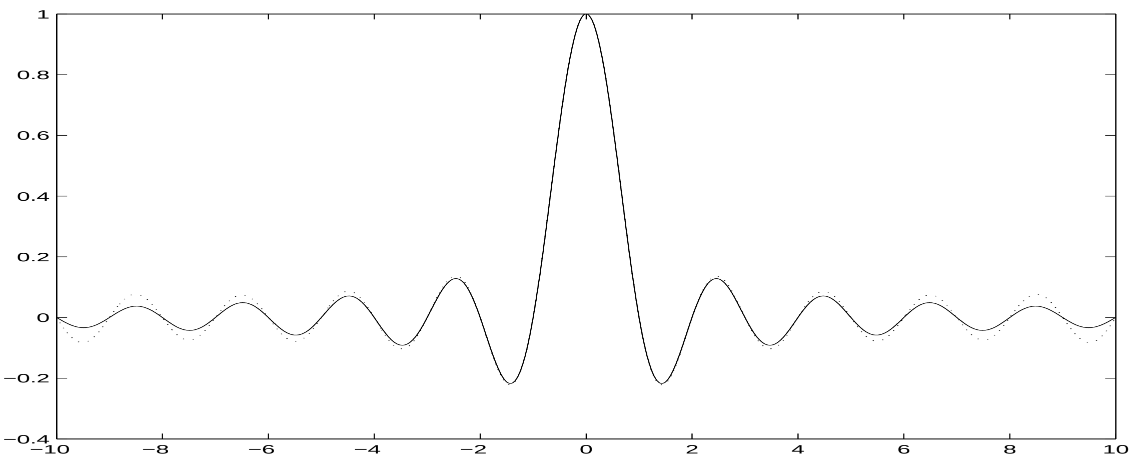


Figure 10.13: The sinc function (solid) and its Lagrange approximation (dashed) as in (10.34) for 100 factors in the product.

10.7.1 Non-Bandlimited Signals

Given a sampling period of T_s seconds, the sampling theorem ensures that there is no loss of information by sampling the class of Ω_N-bandlimited signals, where as usual $\Omega_N = \pi/T_s$. If a signal $x(t)$ is not Ω_N-bandlimited (i.e. its spectrum is nonzero at least somewhere outside of $[-\Omega_N, \Omega_N]$) then the approximation properties of orthogonal bases state that its *best* approximation in terms of uniform samples T_s seconds apart is given by the samples *of its projection over the space of Ω_N-bandlimited signals.* This is easily seen in (10.26), where the projection is easily recognizable as an ideal lowpass filtering operation on $x(t)$ (with gain T_s) which truncates its spectrum outside of the $[-\Omega_N, \Omega_N]$ interval.

Sampling as the result of a sinc basis expansion automatically includes this lowpass filtering operation; for a Ω_N-bandlimited signal, obviously, the filtering is just a scaling by T_s. For an arbitrary signal, however, we can now decompose the sinc sampling as in Figure 10.14, where the first block is a continuous-time lowpass filter with cutoff frequency Ω_N and gain $T_s = \pi/\Omega_N$. The discrete time sequence $x[n]$ thus obtained is the best discrete-time approximation of the original signal when the sampling is uniform.

10.7.2 Aliasing: Intuition

Now let's go back to the naive sampling scheme in which simply $x[n] = x(nT_s)$, with $F_s = 1/T_s$ the sampling frequency of the system; what is the error we incur if $x(t)$ is

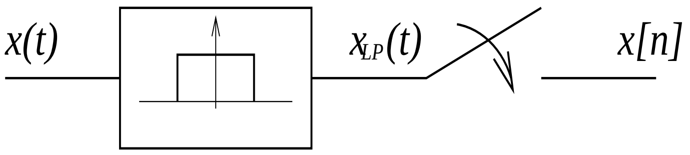


Figure 10.14: Bandlimited sampling (sinc basis expansion) as a combination of lowpass filtering (in the continuous-time domain) and sampling; $x_{LP}(t)$ is the projection of $x(t)$ over the space of Ω_N-bandlimited functions.

not bandlimited or, equivalently, if the sampling frequency is less than twice the Nyquist frequency? We will develop the intuition by starting with the simple case of a single sinusoid and we will move on to a formal demonstration of the aliasing phenomenon. In the following examples we will work with frequencies in Hertz, both out of practicality and to give an example of a different form of notation.

Sampling of Sinusoids. Consider a simple continuous-time signal such as $x(t) = e^{j2\pi f_0 t}$ and its sampled version $x[n] = e^{j2\pi(f_0/F_s)n} = e^{j\omega_0 n}$ with

$$\omega_0 = 2\pi\frac{f_0}{F_s}. \tag{10.38}$$

Clearly, since $x(t)$ contains only one frequency, it is Ω-bandlimited for all $\Omega > 2\pi|f_0|$. If the frequency of the sinusoid satisfies $|f_0| < F_s/2 = F_N$, then $\omega_0 \in (-\pi, \pi)$ and the frequency of the original sinusoid can be univocally determined from the sampled signal. Now assume that $f_0 = F_N = F_s/2$; we have

$$x[n] = e^{j\pi n} = e^{-j\pi n}$$

In other words, we encounter a first ambiguity with respect to the direction of rotation of the complex exponential: from the sampled signal we cannot determine whether the original frequency was $f_0 = F_N$ or $f_0 = -F_N$. If we increase the frequency further, say $f_0 = (1+\alpha)F_N$, we have

$$x[n] = e^{j(1+\alpha)\pi n} = e^{-j\alpha\pi n}$$

Now the ambiguity is both on the direction and on the frequency value: if we try to infer the original frequency from the sampled sinusoid from (10.38) we cannot discriminate between $f_0 = (1+\alpha)F_N$ or $f_0 = -\alpha F_N$. Matters get even worse if $f_0 > F_s$. Suppose we can write $f_0 = F_s + f_b$ with $f_b < F_s/2$; we have

$$x[n] = e^{j(2\pi F_s T_s + 2\pi f_b T_s)n} = e^{j(2\pi+\omega_b)n} = e^{j\omega_b n}$$

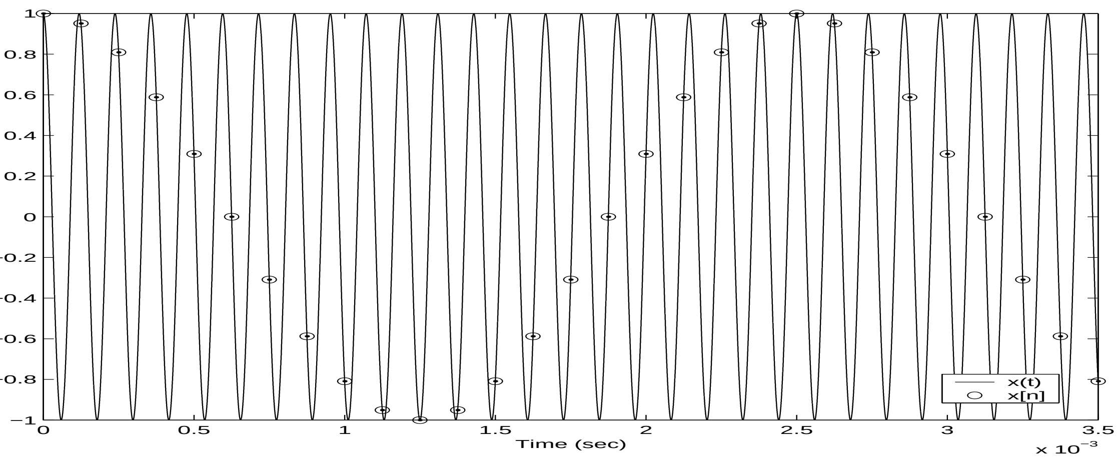


Figure 10.15: Example of aliasing: complex sinusoid at 8400 Hz, $x(t) = e^{j(2\pi \cdot 8400)t}$; sampling frequency $F_s = 8000$ Hz. The sampled signal is indistinguishable from a sinusoid at 400 Hz sampled at F_s (in the plot, only the real part is shown).

so that the sinusoid is completely undistinguishable from a sinusoid of frequency f_b sampled at F_s; the fact that two continuous-time frequencies are mapped to the same discrete-time frequency is called *aliasing*. An example of aliasing is depicted in Figure 10.15.

In general, because of the 2π-periodicity of the discrete-time complex exponential, we can always write

$$\omega_b = (2\pi f_0 T_s) + 2k\pi$$

and choose $k \in \mathbb{Z}$ so that ω_b falls in the $[-\pi, \pi]$ interval. Seen the other way, all continuous-time frequencies of the form

$$f = f_b + kF_s$$

with $f_b < F_N$ are aliased to the same discrete-time frequency ω_b.

Consider now the signal $y(t) = Ae^{j2\pi f_b t} + Be^{j2\pi(f_b+F_s)t}$, with $f_b < F_N$. If we sample this signal with sampling frequency F_s we obtain

$$\begin{aligned} x[n] &= Ae^{j2\pi(f_b/F_s)n} + Be^{j2\pi(f_b/F_s+1)nT_s} \\ &= Ae^{j\omega_b n} + Be^{j\omega_b n}e^{j2\pi n} \\ &= (A+B)e^{j\omega_b n} \end{aligned}$$

In other words, two continuous-time exponential which are F_s Hz apart will give rise to a single discrete-time complex exponential, whose amplitude is equal to the sum of the amplitudes of both the original sinusoids.

Energy Folding of the Fourier Transform. To understands what happens to a general signal, consider the interpretation of the Fourier transform as a bank of (infinitely many) complex oscillators initialized with phase and amplitude, each contributing to the energy content of the signal at their respective frequency. Since in the sampled version any two frequencies F_s apart are undistinguishable, their contributions to the discrete-time Fourier transform of the sampled signal will add up. This aliasing can be represented as a spectral *superposition*: the continuous-time spectrum above F_N is cut, shifted back to $-F_N$, summed over $[-F_N, F_N]$, and the process is repeated again and again; the same for the spectrum below $-F_N$. This process is nothing but the familiar periodization of a signal:

$$\sum_{k=-\infty}^{\infty} X(j2\pi f + j2k\pi F_s)$$

as we will prove formally in the next section.

10.7.3 Aliasing: Proof

In the following we will consider the relationship between the DTFT of a sampled signal $x[n]$ and the CTFT of the originating continuous-time signal $x_c(t)$. For clarity, we will add the subscript $(\cdot)_c$ to all continuous-time quantities so that, for instance, we will write $x[n] = x_c(nT_s)$.

Consider $X(e^{j\omega})$, the DTFT of the sampled sequence (with, as usual, $T_s = (1/F_s) = (\pi/\Omega_N)$). The inversion formula states:

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega \tag{10.39}$$

We can arrive at an expression for $x[n]$ also from $X_c(j\Omega)$, the Fourier transform of the continuous-time function $x_c(t)$; indeed:

$$x[n] = x_c(nT_s) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_c(j\Omega)e^{j\Omega\, nT_s}d\Omega \tag{10.40}$$

The idea is to split the integration interval in the above expression as the sum of non overlapping intervals whose width is equal to the sampling bandwidth $\Omega_s = 2\Omega_N$; this stems from the realization that, in the inversion process, all frequencies Ω_s apart will give undistinguishable contribution to the discrete-time spectrum. We have:

$$\begin{aligned} x[n] &= \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\int_{(2k-1)\Omega_N}^{(2k+1)\Omega_N} X_c(j\Omega)e^{j\Omega nT_s}d\Omega \\ &= \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\int_{-\Omega_N}^{\Omega_N} X_c(j\Omega - jk\Omega_s)e^{j\Omega nT_s}d\Omega \qquad (10.41)\\ &= \frac{1}{2\pi}\int_{-\Omega_N}^{\Omega_N}\left\{\sum_{k=-\infty}^{\infty} X_c(j\Omega - jk\Omega_s)\right\}e^{j\Omega nT_s}d\Omega \qquad (10.42)\\ &= \frac{1}{2\pi}\int_{-\Omega_N}^{\Omega_N}\tilde{X}_c(j\Omega)e^{j\Omega nT_s}d\Omega \qquad (10.43)\\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{1}{T_s}\tilde{X}_c(j\frac{\theta}{T_s})e^{j\theta n}d\theta \qquad (10.44)\end{aligned}$$

A few notes on the above derivation:

(a) In (10.41) we have exploited the Ω_s-periodicity of $e^{j\Omega nT_s}$ (i.e. $e^{j(\Omega+k\Omega_s)nT_s} = e^{j\Omega nT_s}$).

(b) In (10.42) we have interchanged the order of integration and summation. This can be done under fairly broad conditions on $x_c(t)$, for which we refer to the bibliography.

(c) In (10.43) we have defined

$$\tilde{X}_c(j\Omega) = \sum_{k=-\infty}^{\infty} X_c(j\Omega - jk\Omega_s)$$

which is just the periodized version of $X_c(j\Omega)$.

(d) In (10.44) we have operated the change of variable $\theta = \Omega T_s$. It is immediate to verify that $\tilde{X}_c(j(\theta/T_s))$ is now 2π-periodic in θ.

If we now compare (10.44) to (10.39) we can easily see that (10.44) is nothing but the DTFT inversion formula for the 2π-periodic function $(1/T_s)\tilde{X}(j\theta/T_s)$; since the inversion formulas (10.44) and (10.39) yield the same result (namely, $x[n]$) we can conclude that

$$X(e^{j\omega}) = \frac{1}{T_s}\sum_{k=-\infty}^{\infty} X_c(j\frac{\omega}{T_s} - j\frac{2\pi k}{T_s}) \qquad (10.45)$$

which is the relationship between the Fourier transform of a continuous-time function and the DTFT of its sampled version, with T_s being the sampling period. The above result is a particular version of a more general result in Fourier theory called the *Poisson sum formula*. In particular, when $x_c(t)$ is Ω_N-bandlimited, the copies in the periodized spectrum do not overlap and the (periodic) discrete-time spectrum between $-\pi$ and π is simply

$$X(e^{j\omega}) = \frac{1}{T_s} X_c(j\frac{\omega}{T_s})$$

10.7.4 Examples

Figures 10.16 to 10.19 illustrate several examples of the relationship between the continuous-time spectrum and the discrete-time spectrum. For all figures, the top panel shows the continuous-time spectrum, with labels indicating the Nyquist frequency (where applicable) and the sampling frequency. In particular:

- Figure 10.16 shows the result of sampling a bandlimited signal with a sampling frequency in excess of the minimum (twice the Nyquist frequency); in this case we say that the signal has been *oversampled*. The result is that in the periodized spectrum the copies do not overlap and the discrete-time spectrum is just a scaled version of the original spectrum (with even a narrower support than the full $[-\pi, \pi]$ range because of the oversampling).

- Figure 10.17 shows the result of sampling a bandlimited signal with a sampling frequency exactly equal to twice the Nyquist frequency; in this case we say that the signal has been *critically sampled*. In the periodized spectrum the copies again do not overlap and the discrete-time spectrum is a scaled version of the original spectrum.

- Figure 10.18 shows the result of sampling a bandlimited signal with a sampling frequency less than the minimum sampling frequency. Now in the periodized spectrum the copies do overlap and the resulting discrete-time spectrum is an aliased version of the original; the original spectrum cannot be reconstructed from the sampled signal.

- Finally, Figure 10.19 shows the result of sampling a non-bandlimited signal with a sampling frequency which is chosen as a tradeoff between alias and number of samples per second. The idea is to disregard the low-energy "tails" of the original spectrum so that their alias does not corrupt too much the discrete-time spectrum. In the periodized spectrum the copies do overlap and the resulting discrete-time spectrum is an aliased version of the original, which is similar to the original; the original spectrum, however, cannot be reconstructed from the sampled signal. In a practical sampling scenario, the correct design choice would have been to lowpass filter (in the continuous-time domain) the original signal so as to eliminate the spectral tails beyond $\pm\Omega_s/2$.

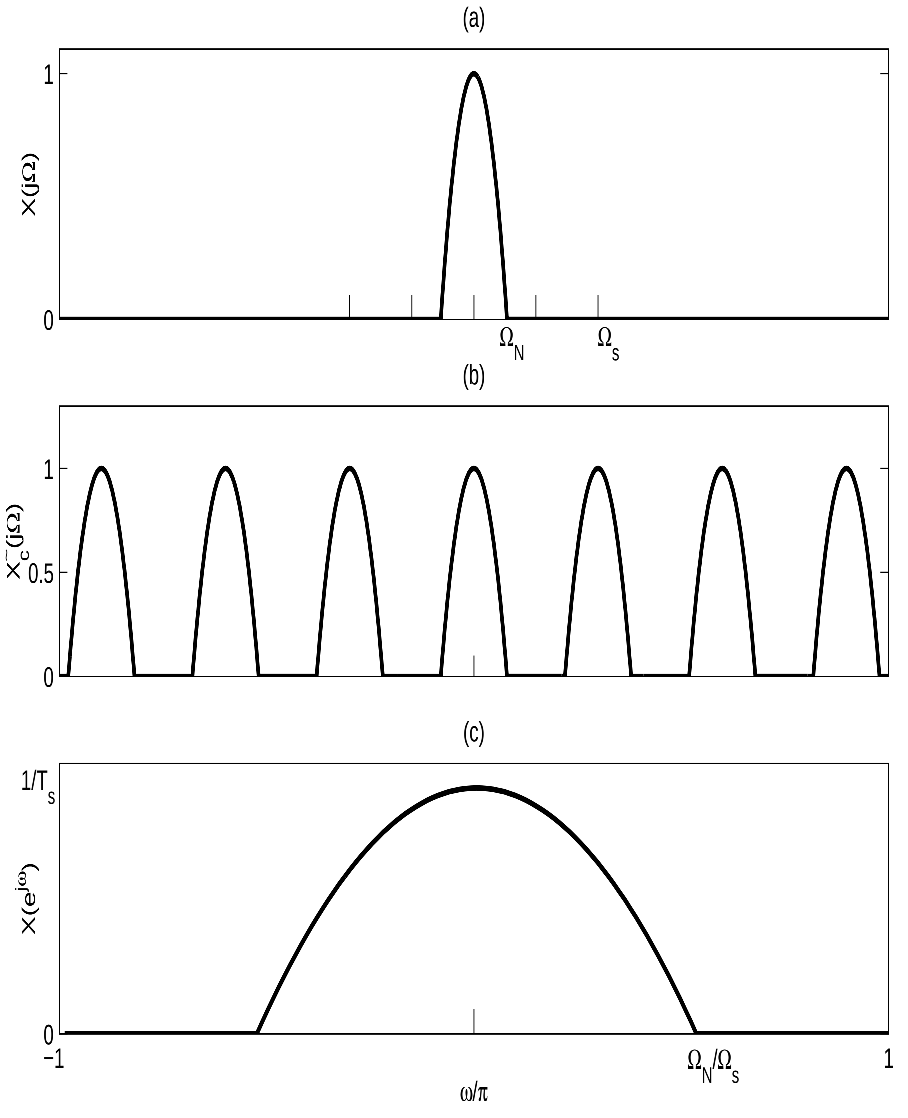


Figure 10.16: Sampling of a bandlimited signal – Case 1: $\Omega_s > 2\Omega_N$.
(a) Original continuous-time spectrum $X_c(j\Omega)$; (b) Periodized spectrum (thick line) $\tilde{X}_c(j\Omega)$; (c) Discrete-time spectrum $X(e^{j\omega})$ in the interval $[-\pi, \pi]$.

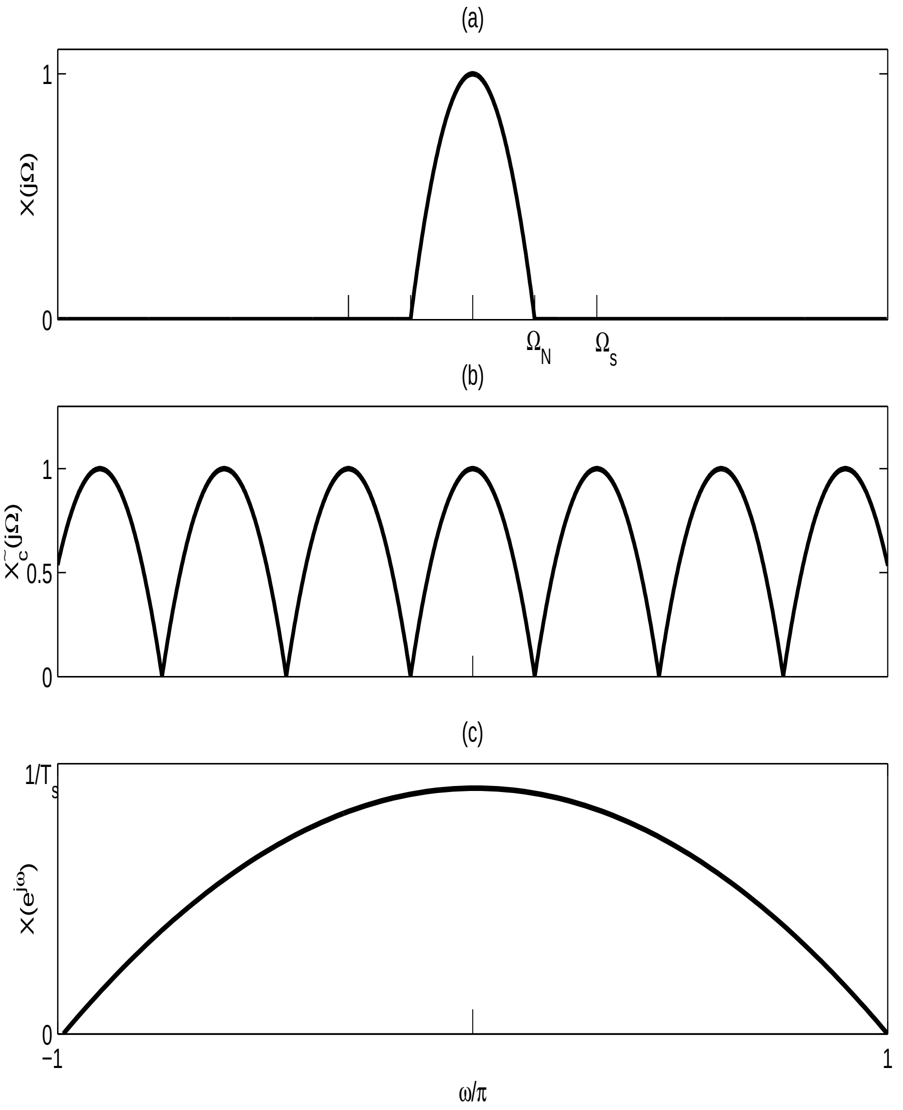


Figure 10.17: Sampling of a bandlimited signal – Case 2: $\Omega_s = 2\Omega_N$.

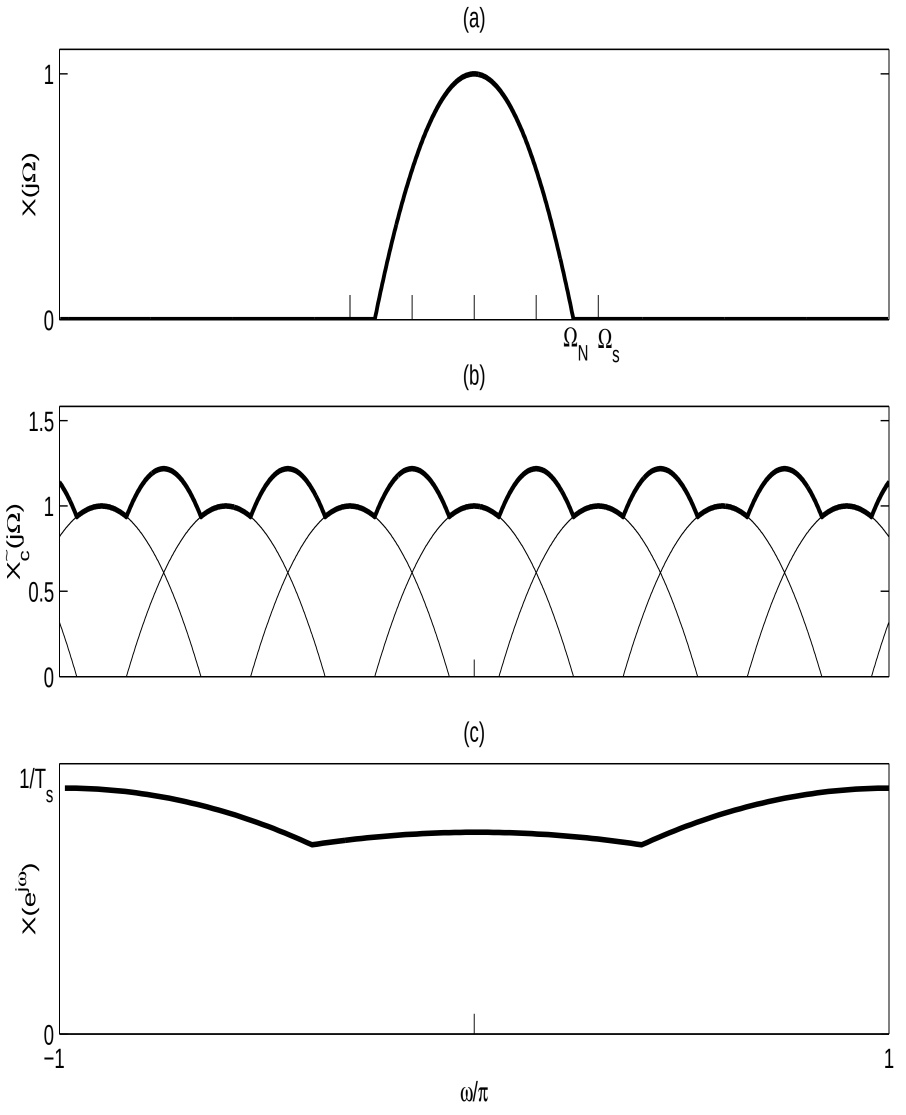


Figure 10.18: Sampling of a bandlimited signal – Case 3: $\Omega_s < 2\Omega_N$ (aliasing).

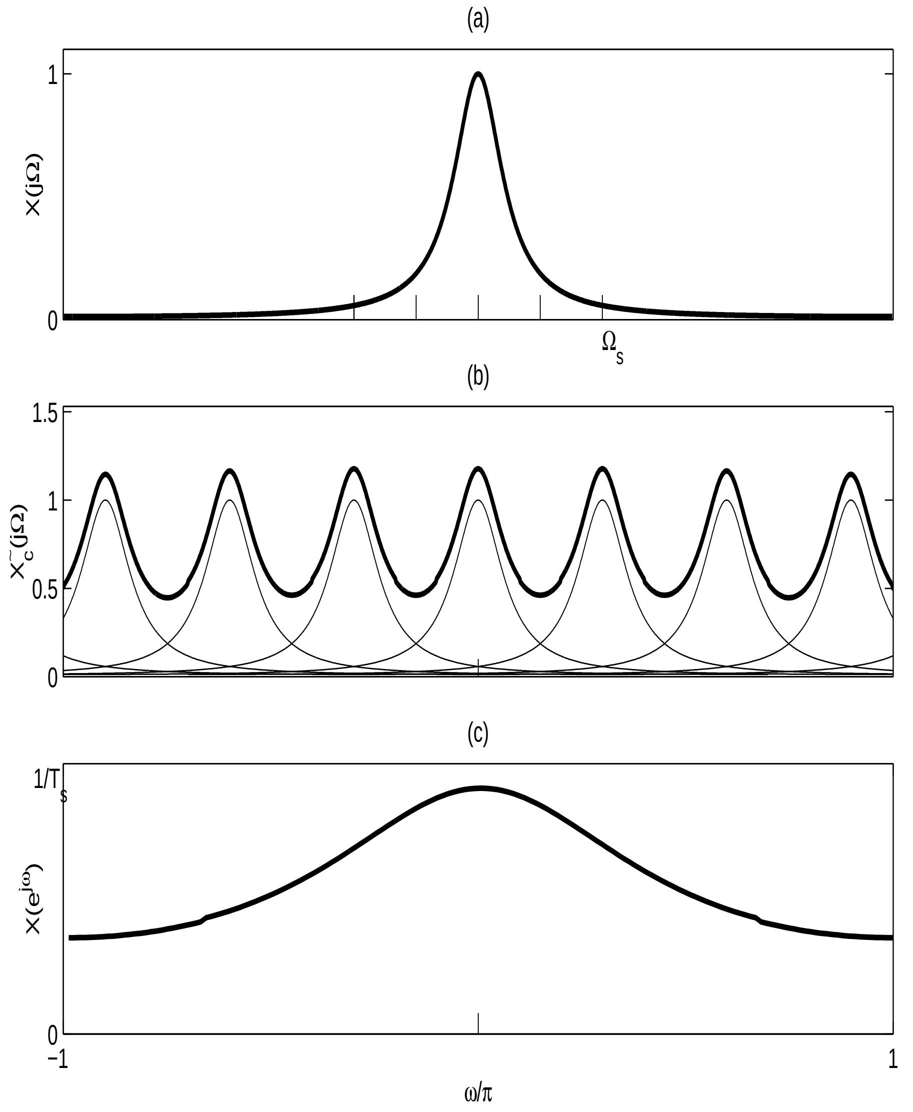


Figure 10.19: Sampling of a non-bandlimited signal.

Example 10.3 *Consider*

$$x_c(t) = e^{j2\pi f_0 t}$$

with corresponding sampled version

$$x[n] = x_c(nT_s) = e^{j2\pi f_0 \frac{n}{f_s}} = e^{j2\pi\left(\frac{f_0}{f_s}\right)n} = e^{j\omega_0 n},$$

where

$$\omega_0 = 2\pi\left(\frac{f_0}{f_s}\right).$$

Let $f_0 = \frac{f_s}{2}$*, then* $\omega_0 = \pi$ *and*

$$x[n] = e^{j\pi n} = (-1)^n = e^{-j\pi n}.$$

Hence we have an ambiguity and from sampled signal we cannot tell if the original signal was $e^{j2\pi f_0 t}$ *or* $e^{-j2\pi f_0 t}$ *(Fig. 10.20).*

Example 10.4 *Consider the continuous-time signal*

$$x_a(t) = \cos(2\pi F_0 t)$$

(a) *Compute analytically the spectrum* $X_a(F)$ *of* $x_a(t)$*. (Hint:*$e^{jat} \overset{\mathcal{F}}{\leftrightarrow} \delta(f - \frac{a}{2\pi})$*))*

(b) *Compute analytically the spectrum of the signal* $x[n] = x_a[nT]$, $T = \frac{1}{F_s}$.

(c) *Plot the magnitude spectrum* $|X_a(F)|$ *for* $F_0 = 10$ *Hz.*

(d) *Plot the magnitude spectrum* $|X(F)|$ *for* $F_s = 10, 20, 40$ *and* 100 *Hz.*

(e) *Explain the results obtained in the previous part in terms of aliasing effects.*

Solution:

(a) *It can easily be seen that* $X_a(F) = \frac{1}{2}(\delta(F - F_0) + \delta(F + F_0))$*. Indeed,*

$$\begin{aligned} X_a(F) &= \int \cos(2\pi F_0 t)e^{-j2\pi F t}dt \\ &= \tfrac{1}{2}\left[\int e^{j2\pi F_0 t}e^{-j2\pi F t}dt + \int e^{-j2\pi F_0 t}e^{-j2\pi F t}dt\right] \\ &= \tfrac{1}{2}(\delta(F - F_0) + \delta(F + F_0)) \end{aligned}$$

using the hint.

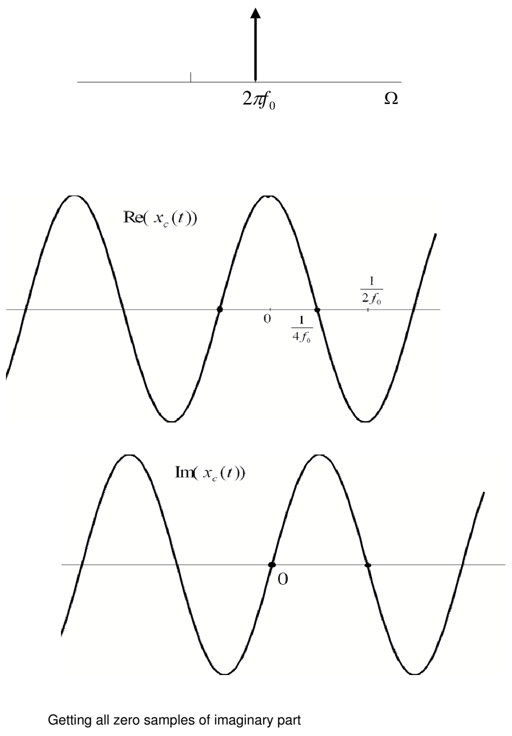


Figure 10.20: Example. 10.3

(b) *We know that* $X(e^{j\omega}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_a(e^{j\frac{\omega}{T}-j\frac{2\pi k}{T}})$. *Further, we know that* $\omega = 2\pi\frac{F}{F_s}$. *Hence,* $X(F) = F_s\sum_{k=-\infty}^{\infty} X_a(F-kF_s) = \frac{F_s}{2}\sum_{k=-\infty}^{\infty}[\delta(F-F_0-kF_s)+\delta(F+F_0-kF_s)]$. *This means that the spectrum of the continuous time signal is repeated every* F_s *when we sample it!*

(c) *See Fig. 10.21.*

(d) *See Fig. 10.22.*

(e) *When the sampling frequency is below or equal to 2 times* F_0 i.e., *when* $F_s \leq 2F_0$,

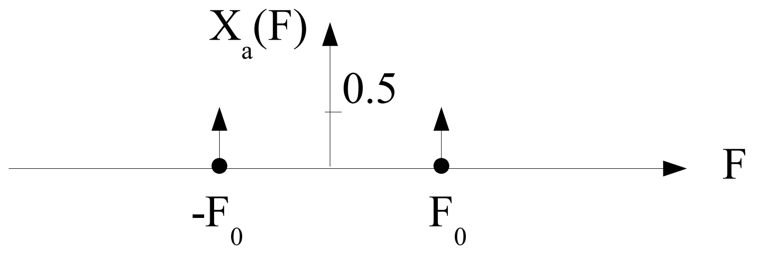


Figure 10.21: Spectrum of $X_a(F)$

we can see that the signal gets aliased.

10.8 Problems

Problem 10.1 *Consider a real function $f(t)$ for which the Fourier transform is well defined:*

$$F(j\Omega) = \int_{-\infty}^{\infty} f(t)e^{-j\Omega t}dt\,. \tag{10.46}$$

Suppose that we only possess a discrete-time version of $f(t)$, that is, we only know the value of $f(t)$ at times $t = n\Delta, n \in \mathbb{Z}$ for a fixed interval Δ. We want to approximate $F(j\Omega)$ with the following expression:

$$\hat{F}(j\Omega) = \sum_{n=-\infty}^{\infty} \Delta \cdot f(n\Delta)e^{-j\Delta n\Omega}\,. \tag{10.47}$$

Remark that $F(j\Omega)$ in (10.46) is computed using the values of $f(t)$ for all t, while the approximation in (10.47) uses only the values of $f(t)$ for a countable number of t: $\ldots, -2\Delta, -\Delta, 0, \Delta, 2\Delta, \ldots$.

Consider now the periodic repetition *of $F(j\Omega)$:*

$$\tilde{F}(j\Omega) = \sum_{n=-\infty}^{\infty} F(j(\Omega + \frac{2\pi}{\Delta}n))\,. \tag{10.48}$$

That is, $F(j\Omega)$ is repeated (with possible overlapping) with period $2\pi/\Delta$ (same Δ as in the approximation (10.47)).

(a) Show that the approximation $\hat{F}(j\Omega)$ is equal to the periodic repetition of $F(j\Omega)$, i.e.

$$\hat{F}(j\Omega) = \tilde{F}(j\Omega)$$

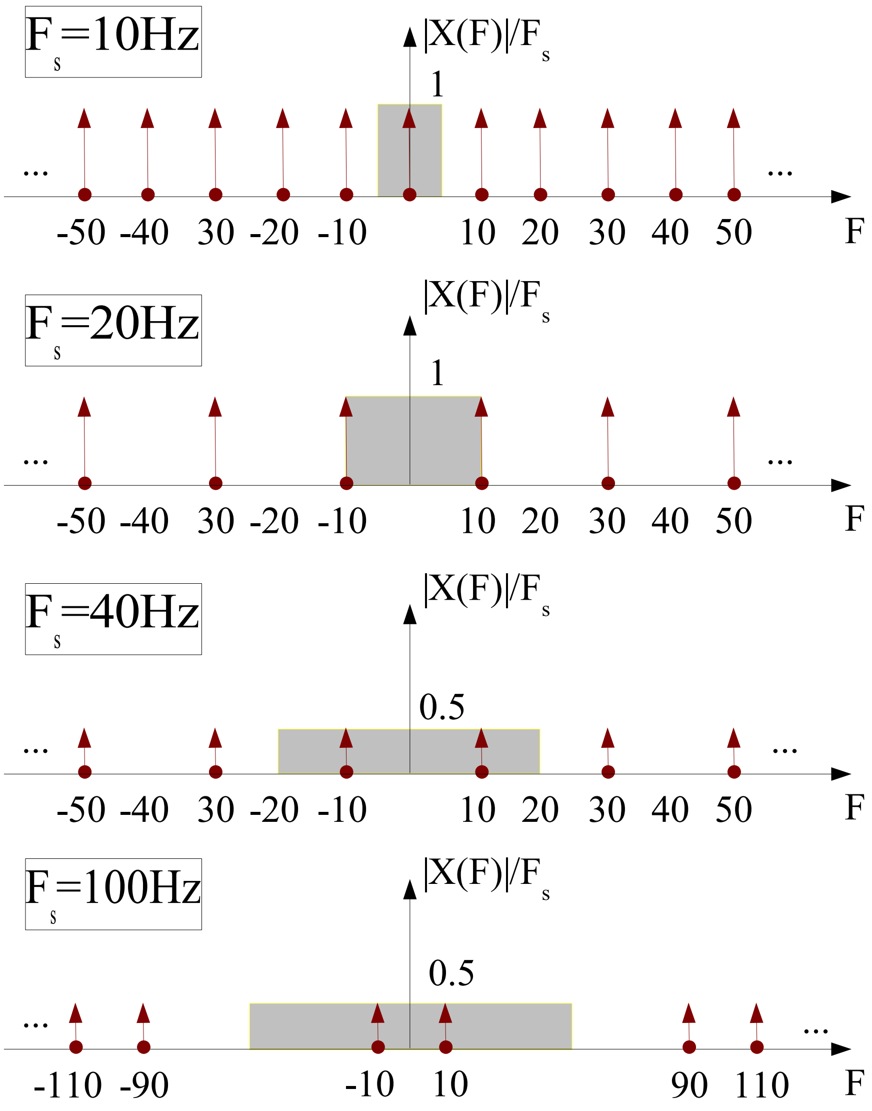


Figure 10.22: Spectrum of $X(F)$ for sampling frequencies $F_s = 10, 20, 40, 100$ Hz

for any value of Δ. *(Hint: consider the periodic nature of $\tilde{F}(j\Omega)$ and remember that a periodic function has a* Fourier series *expansion).*

(b) Give a qualitative description of the result.

(c) For $F(j\Omega)$ as in Figure 10.23, sketch the resulting approximation $\hat{F}(j\Omega)$ for $\Delta = 2\pi/\Omega_0$,$\Delta = \pi/\Omega_0$ and $\Delta = \pi/(100/\Omega_0)$.

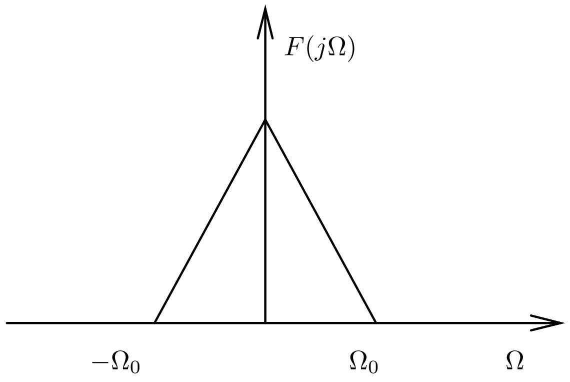


Figure 10.23: Fourier transform $F(j\Omega)$ in Problem 10.1.

Problem 10.2 *One of the standard ways of describing the sampling operation relies on the concept of "modulation by a pulse train". Choose a sampling interval T_s and define a continuous-time pulse train $p(t)$ as:*

$$p(t) = \sum_{k=-\infty}^{\infty} \delta(t - kT_s).$$

The Fourier Transform of the pulse train is

$$P(j\Omega) = (2\pi/T_s) \sum_{k=-\infty}^{\infty} \delta(\Omega - k(2\pi/T_s))$$

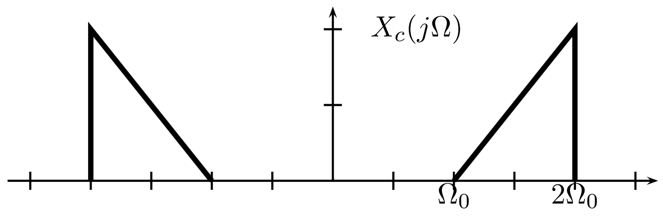
This is tricky to show, so just take the result as is. The "sampled" signal is simply the modulation of an arbitrary-continuous time signal $x(t)$ by the pulse train:

$$x_s(t) = p(t)\,x(t)$$

Note that now, this sampled signal is still continuous time but, by the properties of the delta function, is non-zero only at multiples of T_s; in a sense, $x_s(t)$ is a discrete-time signal brutally embedded in the continuous time world.

Here's the question: derive the Fourier transform of $x_s(t)$ and show that if $x(t)$ is bandlimited to π/T_s then we can reconstruct $x(t)$ from $x_s(t)$.

Problem 10.3 *Consider a real, continuous-time signal $x_c(t)$ with the following spectrum:*



(a) What is the bandwidth of the signal? What is the minimum sampling period in order to satisfy the sampling theorem?

(b) Take a sampling period $T_s = \pi/\Omega_0$; clearly, with this sampling period, there will be aliasing. Plot the DTFT of the discrete-time signal $x_a[n] = x_c(nT_s)$.

(c) Suggest a block diagram to reconstruct $x_c(t)$ from $x_a[n]$.

(d) With such a scheme available, we can therefore exploit aliasing to reduce the sampling frequency necessary to sample a bandpass signal. In general, what is the minimum sampling frequency to be able to reconstruct with the above strategy a real signal whose frequency support on the positive axis is $[\Omega_0, \Omega_1]$ (with the usual symmetry around zero, of course)?

Appendix 10.A The Sinc Product Expansion Formula

The goal is to prove the product expansion

$$\frac{\sin(\pi t)}{\pi t} = \prod_{n=1}^{\infty} \left(1 - \frac{t^2}{n^2}\right). \tag{10.49}$$

We will present two proofs; the first was proposed by Euler in 1748 and, while it certainly lacks rigor by modern standards, it has the irresistible charm of elegance and simplicity in that it relies only on basic algebra. The second proof is more rigorous, and is based on the theory of Fourier series for periodic functions; relying on Fourier theory, however, hides most the convergence issues.

Euler's Proof. Consider the N roots of unity for N odd. They will be $z = 1$ plus $N-1$ complex conjugate roots of the form $z = e^{\pm j\omega_N k}$ for $k = 1, \ldots, (N-1)/2$ and $\omega_N = 2\pi/N$. If we group the complex conjugate roots pairwise we can factor the polynomial $z^N - 1$ as

$$z^N - 1 = (z-1) \prod_{k=1}^{(N-1)/2} (z^2 - 2z\cos(\omega_N k) + 1).$$

The above expression can be immediately generalized to

$$z^N - a^N = (z-a) \prod_{k=1}^{(N-1)/2} (z^2 - 2az\cos(\omega_N k) + a^2).$$

Now replace z and a in the above formula by $z = (1 + x/N)$ and $a = (1 - x/N)$; we obtain:

$$\begin{aligned}
\left(1+\frac{x}{N}\right)^N - \left(1-\frac{x}{N}\right)^N &= \frac{4x}{N} \prod_{k=1}^{(N-1)/2} \left((1-\cos(\omega_N k)) + \frac{x^2}{N^2}(1+\cos(\omega_N k))\right) \\
&= \frac{4x}{N} \prod_{k=1}^{(N-1)/2} (1-\cos(\omega_N k)) \left(1 + \frac{x^2}{N^2} \cdot \frac{1+\cos(\omega_N k)}{1-\cos(\omega_N k)}\right) \\
&= Ax \prod_{k=1}^{(N-1)/2} \left(1 + \frac{x^2\,(1+\cos(\omega_N k))}{N^2\,(1-\cos(\omega_N k))}\right)
\end{aligned}$$

where A is just the finite product $(4/N)\prod_{k=1}^{(N-1)/2}(1-\cos(\omega_N k))$. The value A is also the coefficient for the degree-one term x in the right-hand side and it can be easily seen from the expansion of the left hand-side that $A = 2$ for all N; actually, this is an application of Pascal's triangle and it was proven by Pascal in the general case in 1654. As N grows large we have that

$$\left(1 \pm \frac{x}{N}\right)^N \approx e^{\pm x};$$

at the same time, if N is large, then $\omega_N = 2\pi/N$ is small and, for small values of the angle, the cosine can be approximated as

$$\cos(\omega) \approx 1 - \omega^2/2$$

so that the denominator in the general product term can in turn be approximated as:

$$N^2(1 - \cos((2\pi/N)k) \approx N^2 \cdot \frac{4k^2\pi^2}{2N^2} = 2k^2\pi^2.$$

By the same token, for large N, the numerator can be approximated as $1+\cos((2\pi/n)k) \approx 2$ and therefore the above expansion becomes (by bringing $A = 2$ over to the left-hand side):

$$\frac{e^x - e^{-x}}{2} = x\left(1+\frac{x^2}{\pi^2}\right)\left(1+\frac{x^2}{4\pi^2}\right)\left(1+\frac{x^2}{9\pi^2}\right)\ldots$$

Finally, we replace x by $j\pi t$ to obtain

$$\frac{\sin(\pi t)}{\pi t} = \prod_{n=1}^{\infty}\left(1-\frac{t^2}{n^2}\right).$$

Rigorous Proof. Consider the Fourier series expansion of the *even* function $f(x) = \cos(\tau x)$ periodized over the interval $[-\pi, \pi]$. We have

$$f(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(nx)$$

with

$$\begin{aligned} a_n &= \frac{1}{\pi}\int_{-\pi}^{\pi} \cos(\tau x)\cos(nx)dx \\ &= \frac{2}{\pi}\int_0^{\pi} \frac{1}{2}[\cos((\tau+n)x) + \cos((\tau-n)x)]dx \\ &= \frac{1}{\pi}\left[\frac{\sin((\tau+n)\pi)}{\tau+n} + \frac{\sin((\tau-n)\pi)}{\tau-n}\right] \\ &= \frac{2\sin(\tau\pi)}{\pi}\,\frac{(-1)^n\tau}{\tau^2-n^2} \end{aligned}$$

so that

$$\cos(\tau x) = \frac{2\tau\sin(\tau\pi)}{\pi}\left(\frac{1}{2\tau^2} - \frac{\cos(x)}{\tau^2-1} + \frac{\cos(2x)}{\tau^2-2^2} - \frac{\cos(3x)}{\tau^2-3^2} + \ldots\right)$$

In particular, for $x = \pi$ we have

$$\cot(\pi\tau) = \frac{2\tau}{\pi}\left(\frac{1}{2\tau^2} + \frac{1}{\tau^2-1} + \frac{1}{\tau^2-2^2} + \frac{1}{\tau^2-3^2} + \ldots\right)$$

which we can rewrite as

$$\pi\left(\cot(\pi\tau) - \frac{1}{\pi\tau}\right) = \sum_{n=1}^{\infty}\frac{-2\tau}{n^2-\tau^2}$$

If we now integrate between 0 and t both sides of the equation we have:

$$\int_0^t \left(\cot(\pi\tau) - \frac{1}{\pi\tau}\right) d\pi\tau = \ln \frac{\sin(\pi\tau)}{\pi\tau}\bigg|_0^t = \ln\left[\frac{\sin(\pi t)}{\pi t}\right]$$

and

$$\int_0^t \sum_{n=1}^{\infty} \frac{-2\tau}{n^2 - \tau^2}\, d\tau = \sum_{n=1}^{\infty} \ln\left(1 - \frac{t^2}{n^2}\right) = \ln\left[\prod_{n=1}^{\infty}\left(1 - \frac{t^2}{n^2}\right)\right]$$

from which, finally,

$$\frac{\sin(\pi t)}{\pi t} = \prod_{n=1}^{\infty}\left(1 - \frac{t^2}{n^2}\right).$$

Chapter 11

Multirate Signal Processing

Multirate signal processing refers to systems which allow sequences which arise from *different* sampling rates to be processed together.

There are two early applications that motivate its use in digital audio. Suppose we have an audio signal $x_a(t)$ which has a significant energy only up to $f_M = 22,000$Hz. One way is to implement a bandlimiting filter of width 22kHz with a sharp transition from pass-band to stop-band as illustrated in the previous chapter. This requires the design

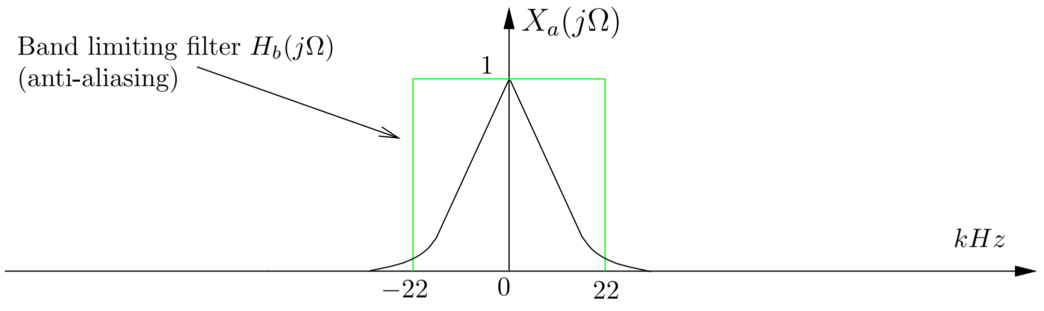


Figure 11.1: Spectrum of the continuous signal.

of a very good analog filter.

Another method is to have a much less stringent front-end anti-aliasing filter and then *oversampling* it, *i.e.,* sample at a rate larger than required and then change sampling rate in discrete domain. This is illustrated in Figures 11.1 and 11.2.

Then we pass the over-sampled discrete-time signal through a digital filter and then *down-sample* by a factor of 2, *i.e.,* drop every second discrete sample to get back to original sampling rate.

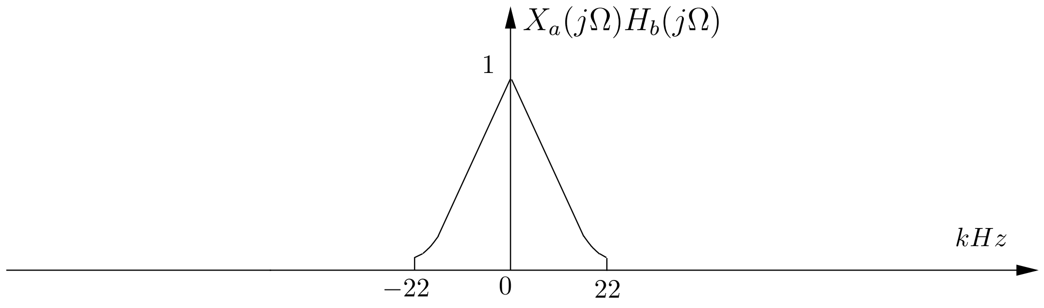


Figure 11.2: Filtering with an ideal analog low-pas filter.

The second application is in altering sampling rate of a system. For example CD is sampled at 44kHz and DAT (for digital audio players) is sampled at 48kHz. Instead of going back to continuous-time, we can do this alteration purely in discrete domain.

We have seen that periodic sampling of a continuous-time signal $x_c(t)$ at a sampling rate of $\frac{1}{T_s}$ is given by

$$x[n] = x_c(nT_s).$$

As seen in the digital audio example, it is sometimes necessary to change the sampling rate of a discrete-time signal to obtain a new discrete-time representation of the underlying continuous-time signal $x_c(t)$ as,

$$y[n] = x_c(nT_s')$$

for a different sampling period $T_s' \neq T_s$. A trivial approach to obtain such a sequence $y[n]$ from $x[n]$ would be to reconstruct $x_c(t)$ from $x[n]$ using the optimal interpolator, and then resample the reconstructed signal with period T_s'. Often this is not desirable since we would have non-ideality in the reconstruction filter (interpolation). Therefore it is of interest to consider methods that change the sampling rate by only discrete-time operations. For example if $T_s' = MT_s$, then we see that since

$$x[n] = x_c(nT_s), \qquad y[n] = x_c(nT_s') = x_c(nMT_s),$$

we can write directly the new discrete-time sequence in terms of $x[n]$ as

$$y[n] = x[Mn],$$

and therefore obtain it completely by discrete-time operations.

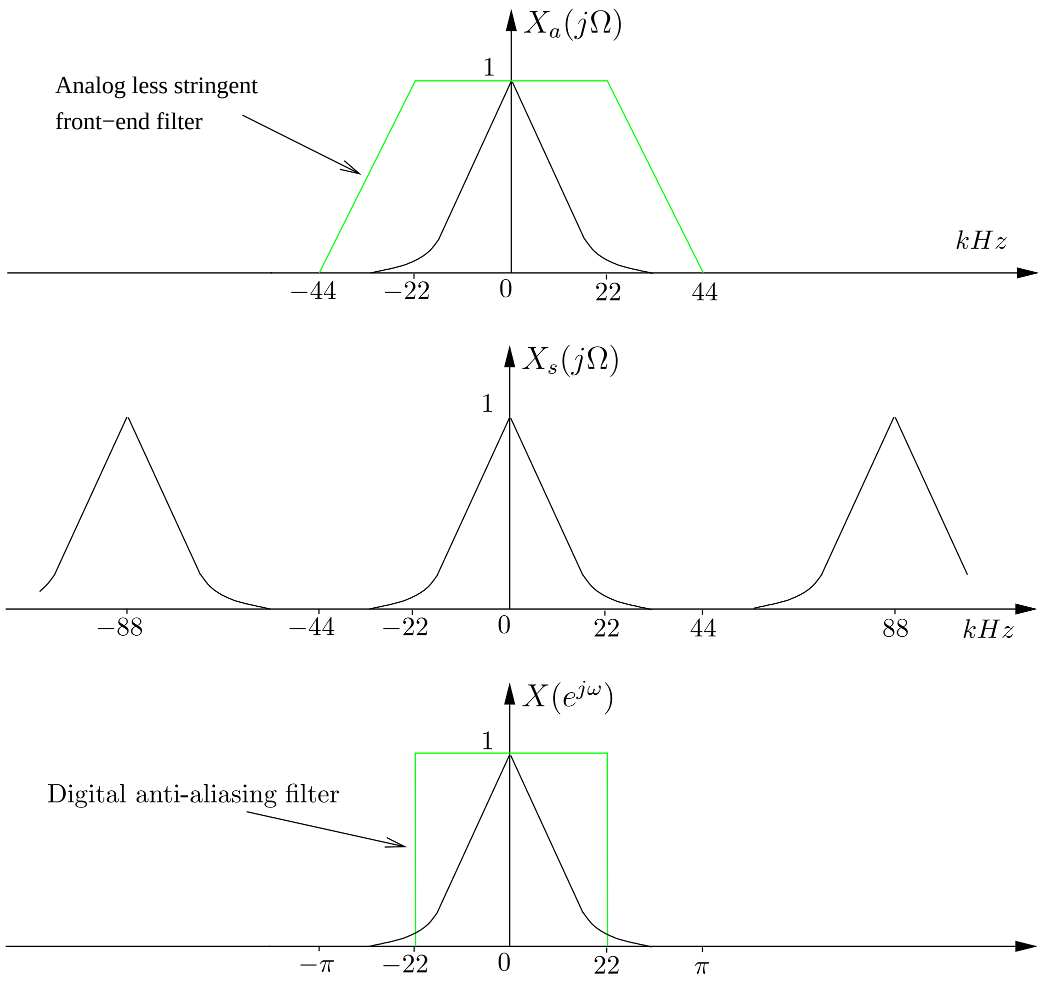


Figure 11.3: Filtering with a much less stringent front-end anti-aliasing filter.

11.1 Downsampling: Sampling rate reduction by an integer factor

This need for change in sampling rate was seen in the motivating digital audio example where one might *oversample* a signal (*i.e.*, sample at a rate higher than necessary) and then after discrete-time processing reduce the sampling rate by *sub-sampling* or *decimation* or *down-sampling*. This means keeping only every M-th sample of the discrete-time process. This is usually represented as shown in Fig. 11.4.

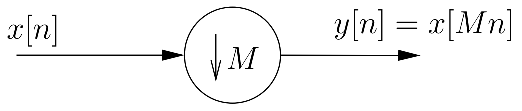


Figure 11.4: Down-sampling by a factor M.

Example 11.1 *The MP3 audio supports a range of sampling frequencies, 8kHz, 11.025 kHz, 12 kHz, 16kHz, 22.05kHz, 24kHz, 32kHz, 44.1kHz, and 48kHz. For example for high end audio music since the human ear can at most listen up to 20kHz, if we band-limit audio at 20kHz, a sampling rate of 44.1kHz (standard for CDs) or 48kHz (standard for DAT) is suitable. However, if one wants to either save storage space or the content is a speech then one can have a smaller sampling rate. Suppose we have an MP3 file already at sampling rate 44.1kHz, and one wants to convert it into a 22.05kHz format, what is the most efficient way to do it?*

An obvious or naive method is to play-back the audio and resample the continuous-time signal at the lower rate. However, we notice that since the information needed for lower sampling rate is already contained in the 44.1kHz sampled audio, therefore by retaining only every other samples from the 44.1kHz sampled audio one can produce an MP3 file in the 22.05 kHz sampling rate format.

For example the down-sampling is shown pictorially for a discrete-time sequence in Fig. 11.6.

Example 11.2 $x[n] = \cos\left(\frac{2\pi}{3}n\right)$

$$y[n] = x[3n] = \cos(2\pi n) = 1$$

This sampling process is shown in Fig. 11.7.

Note that the origin of time is very important, since if we down-sample $x'[n] = x[n+1]$ instead of $x[n]$, the output is

$$y'[n] = x'[2n] = x[2n+1] \neq x[2n] = y[n].$$

Example 11.3 *If we down-sample from*

$$x[n] = \cos\left(\frac{2\pi}{3}n\right)$$

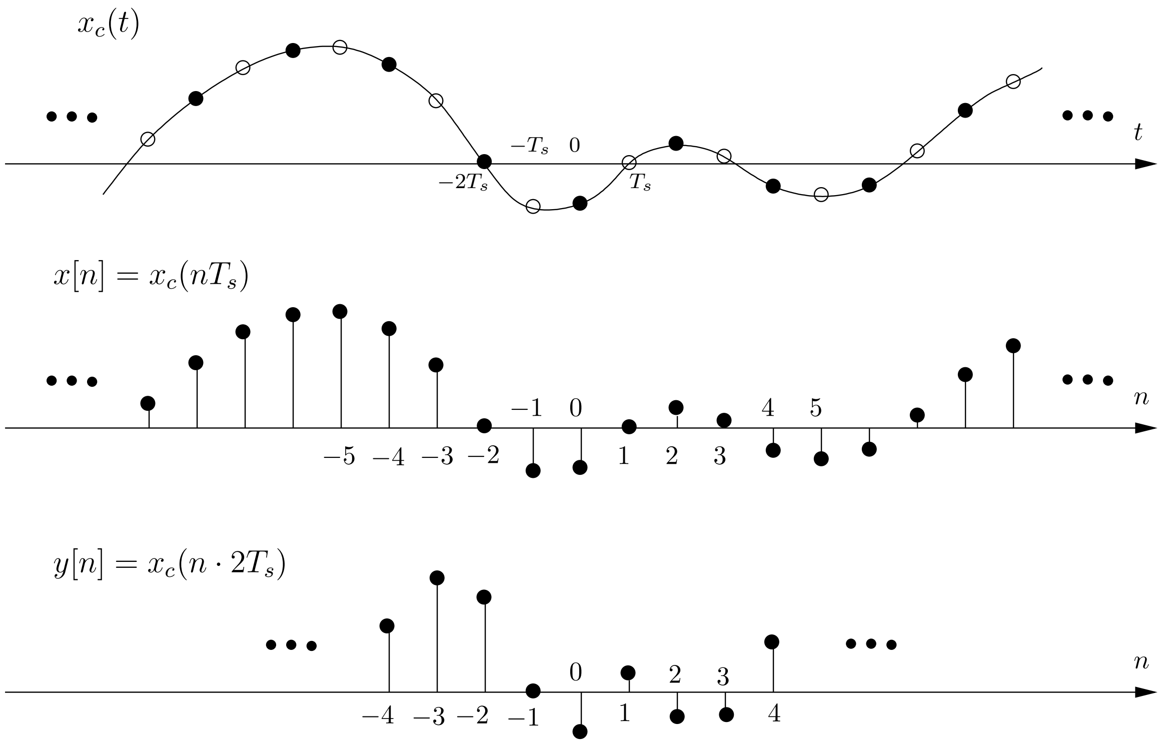


Figure 11.5: Sampling from a continuous-time signal with different periods.

and

$$x'[n] = x[n+1] = \cos\left(\frac{2\pi}{3}(n+1)\right) = \cos\left(\frac{2\pi}{3}n + \frac{2\pi}{3}\right)$$

with down-sampling factor 3*, we have*

$$y'[n] = x'[3n] = \cos\left(\frac{2\pi}{3} \cdot 3n + \frac{2\pi}{3}\right) = \cos\frac{2\pi}{3} = -\frac{1}{2},$$

and clearly $y'[n] \neq y[n] = x[3n] = 1$.

Therefore, down-sampling is *not* a time-invariant operation.

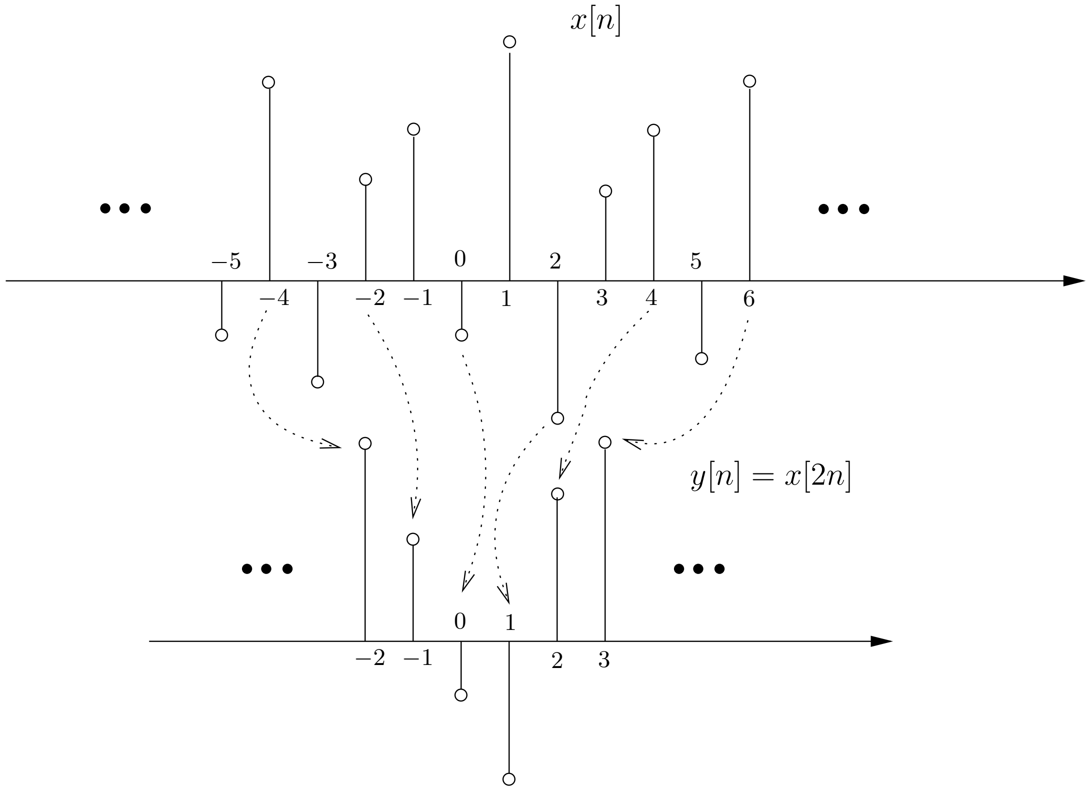


Figure 11.6: Down-sampling by a factor 2.

Example 11.4 *Let*

$$
\begin{aligned}
x[n] &= \sin\left(\frac{2\pi}{3}n\right), \\
x_1[n] &= x[n-1] = \sin\left(\frac{2\pi}{3}n - \frac{2\pi}{3}\right), \\
x_2[n] &= x[n-2] = \sin\left(\frac{2\pi}{3}n - \frac{4\pi}{3}\right), \\
x_3[n] &= x[n-3] = \sin\left(\frac{2\pi}{3}n - 2\pi)\right) = \sin\left(\frac{2\pi}{3}n\right).
\end{aligned}
$$

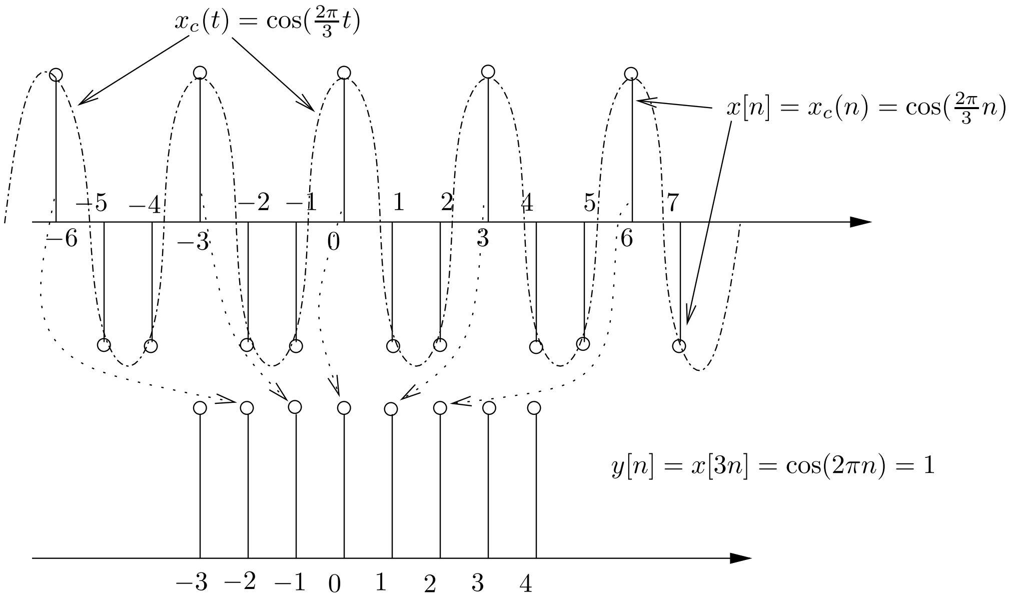

$x_c(t) = \cos(\frac{2\pi}{3}t)$
$x[n] = x_c(n) = \cos(\frac{2\pi}{3}n)$
$y[n] = x[3n] = \cos(2\pi n) = 1$
−6 −5 −4 −3 −2 −1 0 1 2 3 4 5 6 7
−3 −2 −1 0 1 2 3 4

Figure 11.7: Down-sampling from $x_c(t) = \cos(\frac{2\pi}{3}t)$ with down-sampling factor 3.

Hence

$$\begin{aligned}
y[n] &= x[3n] = \sin(2\pi n) = 0, \\
y_1[n] &= x_1[3n] = x[3n-1] = \sin\left(-\frac{2\pi}{3}n\right) = -\sin\left(\frac{2\pi}{3}n\right) = -\frac{\sqrt{3}}{2}, \\
y_2[n] &= x_2[3n] = x[3n-2] = \sin\left(-\frac{4\pi}{3}n\right) = \frac{\sqrt{3}}{2}, \\
y_3[n] &= x_3[3n] = x[3n-3] = \sin(2\pi n) = 0.
\end{aligned}$$

Here we see a periodicity in the shifting property.

Example 11.5 *Let we sample from the sequences*

$$\begin{aligned}
x[n] &= \left(\frac{1}{2}\right)^n u[n], \\
x_1[n] &= x[n-1] = \left(\frac{1}{2}\right)^{n-1} u[n-1], \\
x_2[n] &= x[n-2] = \left(\frac{1}{2}\right)^{n-2} u[n-2]
\end{aligned}$$

with down-sampling factor 2. Then we have

$$\begin{aligned}
y[n] &= x[2n] = \left(\frac{1}{2}\right)^{2n} u[2n],\\
y_1[n] &= x_1[2n] = x[2n-1] = \left(\frac{1}{2}\right)^{2n-1} u[2n-1],\\
y_2[n] &= x_2[2n] = x[2n-2] = \left(\frac{1}{2}\right)^{2n-2} u[2n-2] = \left(\frac{1}{2}\right)^{2(n-1)} u[2(n-1)] = y[n-1].
\end{aligned}$$

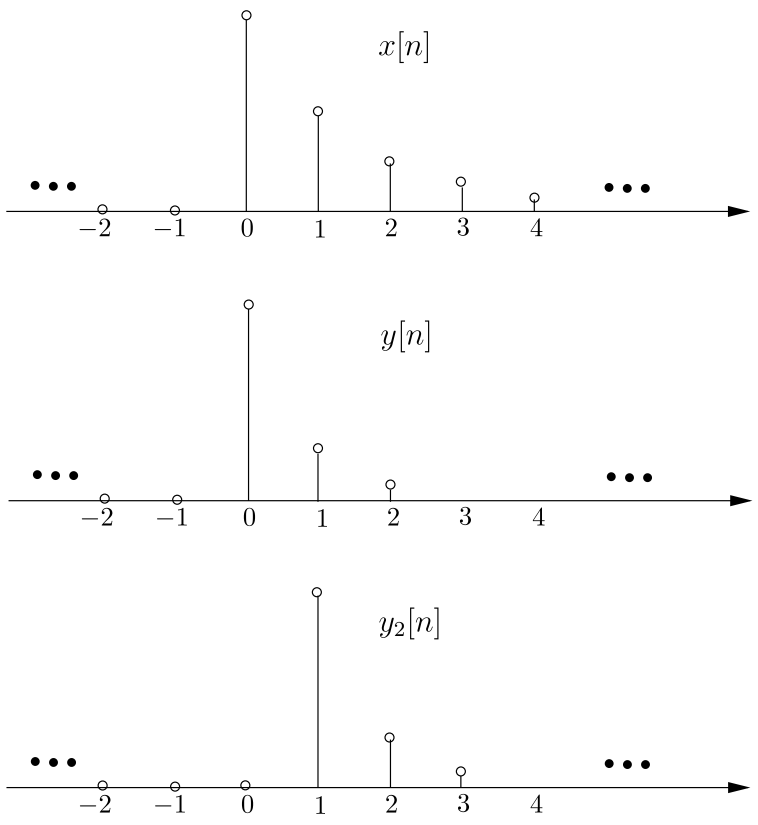


Figure 11.8: Sampling from $x[n] = \left(\frac{1}{2}\right)^n u[n]$.

If we denote by $D_M(\cdot)$, the operator that down-samples a signal by M, *i.e.,*

$D_M\left(x[n]\right) = x[Mn]$,

then we can state the following property.

Property: Down-sampling by M or $D_M(\cdot)$ is a linear, periodically time-varying operator with period M.

Proof: Since

$$D_M\left(\alpha x_1[n] = \beta x_2[n]\right) = \alpha x_1[Mn] + \beta x_2[Mn] = \alpha D_M\left(x_1[n]\right) + \beta D_M\left(x_2[n]\right)$$

clearly $D_M(\cdot)$ is a linear operator.

It has a *periodically time-varying* property because if a sequence is shifted by M, its down-sampled version is shifted by 1. More precisely,

$$\begin{aligned} D_M\left(\delta[n-kM]\right) &= \delta[n-k], \qquad k \in \mathbb{Z} \\ D_M\left(\delta[n-kM-\ell]\right) &= 0, \qquad \ell = 1, 2, \ldots, M-1. \end{aligned}$$

Hence, if $y[n] = D_M\left(x[n]\right)$, then $D_M\left(x[n-kM]\right) = y[n-k]$. ■

Therefore, for the down-sampling system the time-varying property implies that complex sinusoids are *no longer eigen-functions*. This can be seen from the following argument. Let

$$x[n] = e^{j\pi n} = (-1)^n.$$

If we down-sample by a factor 2,

$$y[n] = x[2n] = e^{j\pi 2n} = 1 \neq c \cdot x[n]$$

for any constant c. Hence the complex sinusoid is not an eigen-function for the down-sampling operator.

Recall that for a linear time-invariant system, the complex exponential was an eigen-function, *i.e.,* if

$$\mathcal{L}\left\{x[n]\right\} = \sum_m h[m]x[n-m],$$

then for $x[n] = e^{j\omega_0 n}$,

$$y[n] = \mathcal{L}\left\{e^{j\omega_0 n}\right\} = H\left(e^{j\omega_0}\right)x[n]$$

was a scaled version of $x[n]$ with scaling factor $H\left(e^{j\omega_0}\right)$ which is a constant independent of n.

We have now seen that the time-invariant property is crucial for this to be true. This is because $D_M(\cdot)$ is linear but *not* time-invariant and does not have the complex exponential as an eigen-sequence or eigen-function.

An important consideration is to understand what happens to the z-transform (or discrete-time Fourier transform) when one down-samples a signal. To be specific, we first consider down-sampling by a factor of 2 of a discrete-time signal with z-transform $X(z)$.

First let us define a new discrete-time sequence $x'[n]$ with z-transform $X'(z)$ as

$$\begin{aligned}
X'(z) &= \frac{1}{2}\left[X(z)+X(-z)\right] = \frac{1}{2}\sum_n x[n]z^{-n} + \frac{1}{2}\sum_n x[n](-1)^{-n}z^{-n} \\
&= \frac{1}{2}\sum_n \left\{x[n]+(-1)^{-n}x[n]\right\} z^{-n} \\
&= \sum_{n:\text{even}} x[n]z^{-n} + \underbrace{\sum_{n:\text{odd}} x[n](1-1)z^{-n}}_{0} \\
&= \sum_n x[2n]z^{-2n}.
\end{aligned}$$

Now, if

$$y[n] = x[2n],$$

then

$$Y(z) = \sum_n y[n]z^{-n} = \sum_n x[2n]z^{-n} = X'\left(z^{\frac{1}{2}}\right) = \frac{1}{2}\left[X\left(z^{\frac{1}{2}}\right) + X\left(-z^{\frac{1}{2}}\right)\right].$$

Evaluating it on the unit circle, if the ROC includes it, gives

$$Y\left(e^{j\omega}\right) = \frac{1}{2}\left[X\left(e^{j\frac{\omega}{2}}\right) + X\left(e^{-j\frac{\omega}{2}}\right)\right] = \frac{1}{2}\left[X\left(e^{j\frac{\omega}{2}}\right) + X\left(e^{j(\frac{\omega}{2}-\pi)}\right)\right].$$

Example 11.6 *Let*

$$x[n] = a^n u[n], \qquad 0 < a < 1.$$

Then

$$X(z) = \frac{1}{1-az^{-1}}, \qquad \textit{ROC: } |z| > a.$$

Now define

$$y[n] = x[2n] = a^{2n}u[2n] = \begin{cases} (a^2)^n, & n \geq 0 \\ 0 & \text{else.} \end{cases}$$

Hence

$$Y(z) = \frac{1}{1-z^2 z^{-1}}, \qquad \textit{ROC: } |z| > a^2.$$

But we see that

$$\begin{aligned}\frac{1}{2}\left[X\left(z^{\frac{1}{2}}\right)+X\left(-z^{\frac{1}{2}}\right)\right] &= \frac{1}{2}\left[\frac{1}{1-az^{-\frac{1}{2}}}+\frac{1}{1+az^{-\frac{1}{2}}}\right] \\ &= \frac{1}{2}\left[\frac{2}{(1-az^{-\frac{1}{2}})(1+az^{-\frac{1}{2}})}\right]=\frac{1}{1-a^2z^{-1}} \\ &= Y(z)\end{aligned}$$

with ROC: $|z^{\frac{1}{2}}| > a$ *or* $|z| > a^2$.

This illustrates the following important points:

- The output is a function of not only $X(z)$ but also $X(-z)$ (which corresponds to the z-transform of a sequence $(-1)^n x[n]$, *i.e.,* a modulated version of $x[n]$).
- Since we have a time-scaling by a factor of 2 in going from $x'[n]$ to $y[n]$, it corresponds to $z \longrightarrow z^{\frac{1}{2}}$ or $e^{j\omega} \longrightarrow e^{j\omega/2}$ (contraction in time implies expansion in frequency).

Example 11.7 *Let* $x[n]$ *be a sequence with the spectrum* $X\left(e^{j\omega}\right)$ *given in Fig. 11.9. Let*

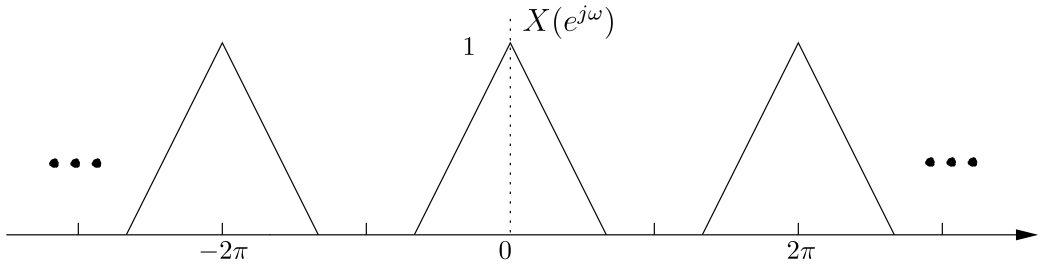


Figure 11.9: $X\left(e^{j\omega}\right)$.

$y[n] = x[2n]$ *be a down-sampled version of* $x[n]$ *with factor* 2. *Note that* $Y\left(e^{j\omega}\right) = X'\left(e^{j\frac{\omega}{2}}\right)$, *where*

$$X'\left(e^{j\omega}\right)=\frac{1}{2}\left[X\left(e^{j\omega}\right)+X\left(e^{j(\omega-\pi)}\right)\right].$$

Now consider the plot shown in Fig. 11.10, wherein $X\left(e^{j\omega}\right)$ *is shown in solid line and* $X\left(e^{j(\omega-\pi)}\right)$ *is in dashed line. Therefore* $X'\left(e^{j\omega}\right)$ *can be constructed as given in Fig. 11.11.*

Finally, we construct $Y\left(e^{j\omega}\right)$ *where* $Y\left(e^{j\omega}\right) = X'\left(e^{j\frac{\omega}{2}}\right)$ *means that it just expands the* ω *axis in going from* $X'\left(e^{j\omega}\right)$ *to* $Y\left(e^{j\omega}\right)$.

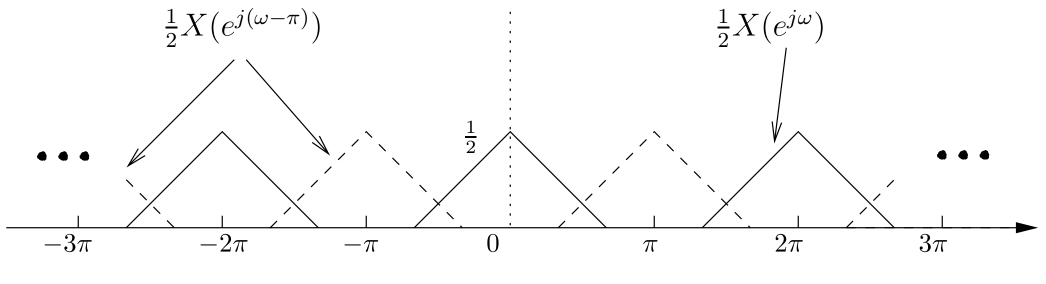


Figure 11.10: $\frac{1}{2}X\left(e^{j\omega}\right)$ in solid line and $\frac{1}{2}X\left(e^{(j\omega-\pi)}\right)$in dashed line.

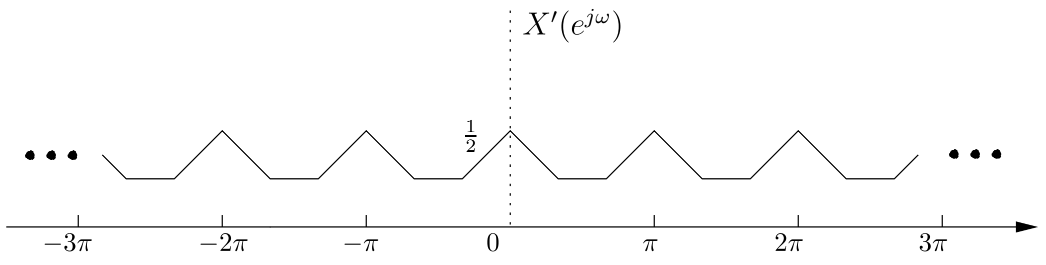


Figure 11.11: $X'\left(e^{j\omega}\right)$.

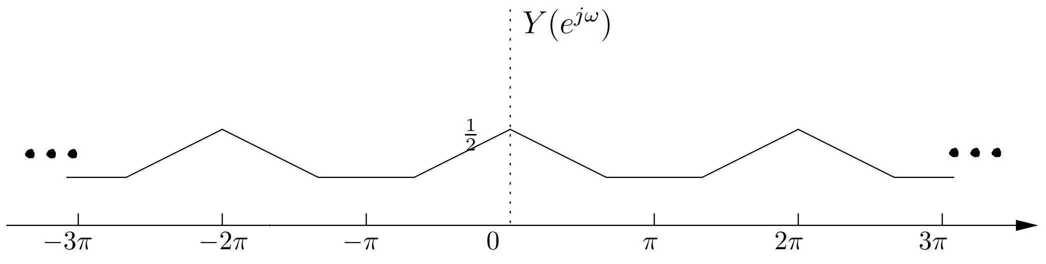


Figure 11.12: $Y\left(e^{j\omega}\right)$.

The result for down-sampling by 2 generalizes to arbitrary integer M as follows.

Proposition 11.1 *Given a signal $x[n]$ with z-transform $X(z)$, its down-sampled by factor M as*

$$y[n] = D_M\left(x[n]\right) = x[Mn]$$

has the following z-transform,

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(e^{-j\frac{2\pi}{M}k} z^{\frac{1}{M}}\right)$$

and if the unit circle is in the ROC then,

$$Y\left(e^{j\omega}\right) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(e^{j\left(\frac{\omega}{M}-\frac{2\pi}{M}k\right)}\right).$$

Proof. Using the same idea as done for the $M = 2$ case, let

$$\begin{aligned}
X'(z) &= \frac{1}{M}\sum_{k=0}^{M-1} X\left(e^{-j\frac{2\pi}{M}k} z\right) \\
&= \frac{1}{M}\sum_{k=0}^{M-1}\sum_{n} x[n]\left(e^{-j\frac{2\pi}{M}k} z\right)^{-n} \\
&= \sum_{n} x[n] \underbrace{\left\{\frac{1}{M}\sum_{k=0}^{M-1} e^{-j\frac{2\pi}{M}kn}\right\}}_{\delta[n-\ell M],\text{ for }\ell\in\mathbb{Z}} z^{-n} \\
&= \sum_{\ell} x[\ell M] z^{-\ell M}.
\end{aligned}$$

Then we see that

$$X'\left(z^{\frac{1}{M}}\right) = \sum_{\ell} x[\ell M] z^{-\ell} = Y(z).$$

11.2 Filtering and Down-sampling

As seen in Example. 11.7, we notice that the down-sampled version has aliasing. Because of this, it is usually better to low-pass filter a signal before sub-sampling. Let us do Example. 11.7 again along with this filtering.

Example 11.8 *Let $x[n]$ be a sequence for which the DTFT is given in Fig. 11.13. We pass it through a low-pass filter with cut-off frequency at $\frac{\pi}{2}$. Thus, we obtain $X_{LP}\left(e^{j\omega}\right)$ as shown in Fig. 11.14. Thus $X'\left(e^{j\omega}\right)$ and $Y\left(e^{j\omega}\right)$ will be obtained as in Fig. 11.15 and Fig. 11.16.*

Therefore, we see that

$$X_{LP}(z) = H(z)X(z)$$

where $H(z)$ is a low-pass filter operation. From here it is clear that for $y[n] = x_{LP}[2n]$,

$$\begin{aligned} Y(z) &= \frac{1}{2}\left[X_{LP}\left(z^{\frac{1}{2}}\right) + X_{LP}\left(-z^{\frac{1}{2}}\right)\right] \\ &= \frac{1}{2}\left[H\left(z^{\frac{1}{2}}\right)X\left(z^{\frac{1}{2}}\right) + H\left(-z^{\frac{1}{2}}\right)X\left(-z^{\frac{1}{2}}\right)\right]. \end{aligned}$$

This operator is presented in Fig. 11.17.

If the unit circle is in the ROC, then

$$Y\left(e^{j\omega}\right) = \frac{1}{2}\left[H\left(e^{j\frac{\omega}{2}}\right)X\left(e^{j\frac{\omega}{2}}\right) + H\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)X\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right].$$

Similar to Proposition. 11.1, we see that for a general filtering and down-sampling by M (see fig. 11.17), it follows that if $y[n] = x[Mn]$,then

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} H\left(e^{-j\frac{2\pi}{M}k}z^{\frac{1}{M}}\right)X\left(e^{-j\frac{2\pi}{M}k}z^{\frac{1}{M}}\right).$$

11.3 Upsampling: increasing the sampling rate by an integer factor

We have seen that the reduction of the sampling rate of a discrete-time signal by an integer factor involves sampling the sequence in a manner analogous to sampling the continuous-time signal. A natural question to ask is whether we can *increase* the sampling rate. Clearly we cannot get more information by such an operation, but this is useful when we want to change the sampling rate by a rational factor as we will see later.

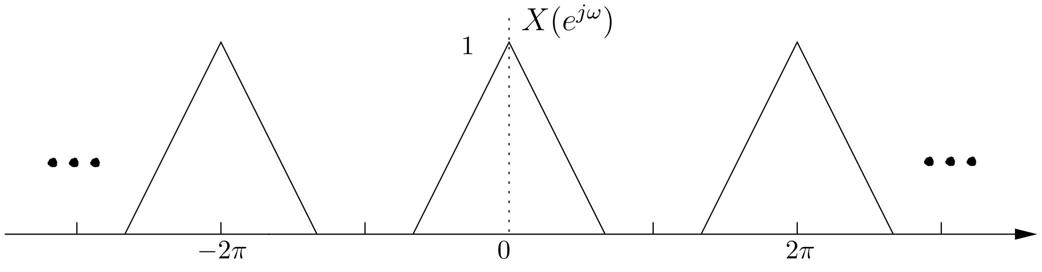


Figure 11.13: $X\left(e^{j\omega}\right)$.

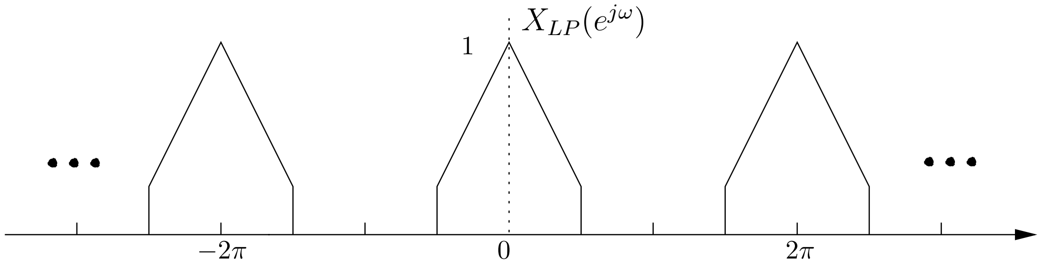


Figure 11.14: $X_{LP}\left(e^{j\omega}\right)$.

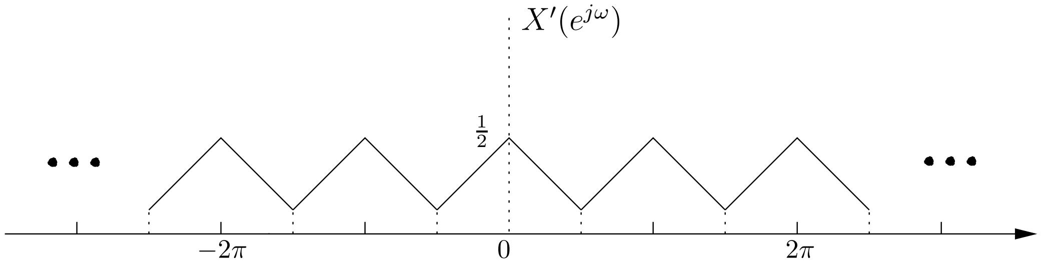


Figure 11.15: $X'\left(e^{j\omega}\right)$.

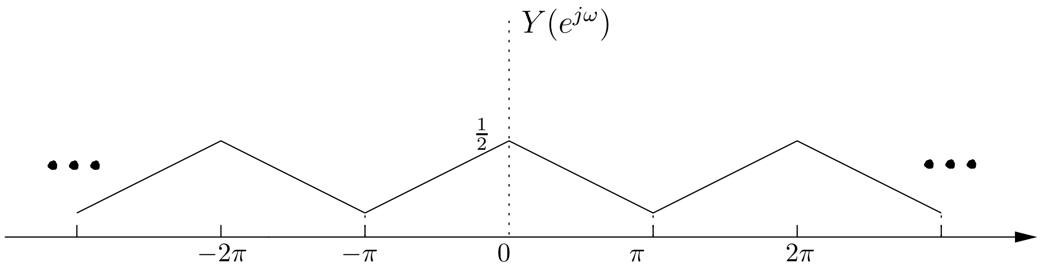


Figure 11.16: $Y\left(e^{j\omega}\right)$.

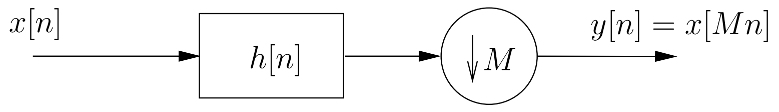


Figure 11.17: Filtering before subsampling. Typically filter is a low-pass filter with cut-off frequency $\frac{\pi}{M}$.

Suppose we have a sequence $x[n]$ whose sampling rate we wish to increase by a factor L. If we consider the underlying continuous-time signal $x_c(t)$, the objective is to obtain samples

$$x'[n] = x_c(n{T_s}'), \tag{11.1}$$

where ${T_s}' = \frac{T_s}{L}$, from the sequence of samples

$$x[n] = x_c(nT_s). \tag{11.2}$$

We will refer to the operation of increasing the sampling rate as *up-sampling.* From (11.1) and (11.2) it follows that

$$x'[n] = x\left[n/L\right] = x_c\left(nT_s/L\right) \text{ for } n = kL,\, k \in \mathbb{Z}.$$

For the samples $n \neq kL$, $k \in \mathbb{Z}$, we simply replace it with zero, *i.e.,* $y[n]$ is the up-sampled version of $x[n]$ if

$$y[n] = \begin{cases} x\left[n/L\right] & n = kL, k \in \mathbb{Z} \\ 0 & \text{otherwise.} \end{cases} \tag{11.3}$$

Equivalently,

$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]\delta[n-kL]. \tag{11.4}$$

Example 11.9 *See Fig. 11.18.*

Let us now compute teh z-transform of the upsampled version of a sequence. We denote the up-sampling operator by $\mathrm{U}_L(\cdot)$. Then we have,

$$\begin{aligned} Y(z) &= \sum_n y[n]z^{-n} = \sum_n \left(\sum_{k=-\infty}^{+\infty} x[k]\delta[n-kL] \right) z^{-n} \\ &= \sum_{k=-\infty}^{+\infty} x[k]z^{-kL} = X\left(z^L\right). \end{aligned} \tag{11.5}$$

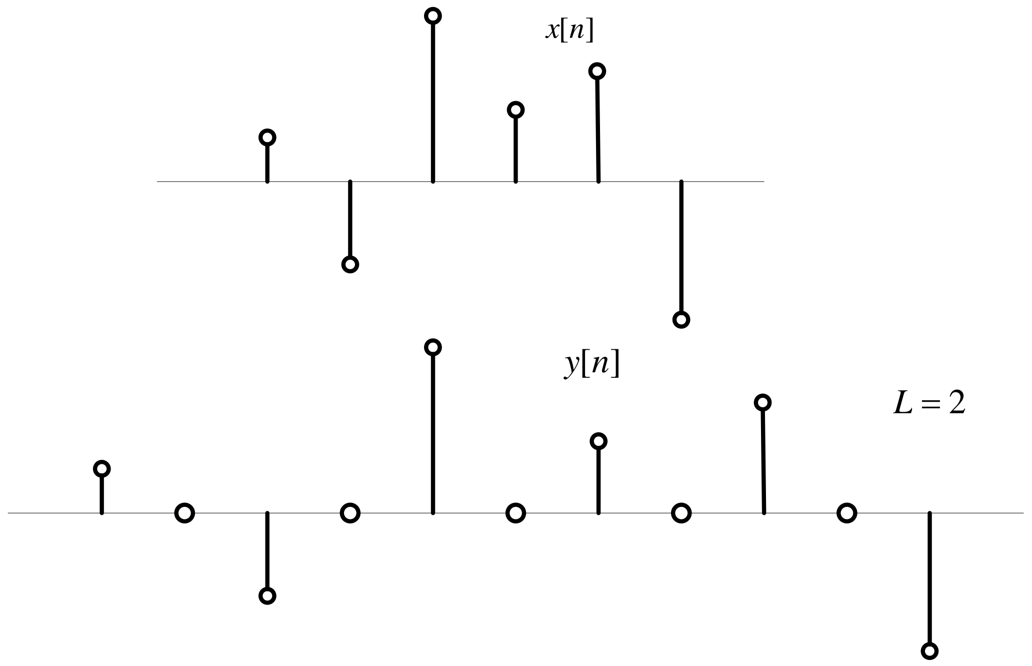


Figure 11.18: Example. 11.9

If the unit circle is in the ROC, then

$$Y\left(e^{j\omega}\right) = x\left(e^{j\omega L}\right).$$

Note that unlike downsampling, where there is a loss of information (and hence is not an invertible operation), upsampling can be easily inverted since,

$$\mathrm{D}_L\left(\mathrm{U}_L\left(x[n]\right)\right) = x[n]. \tag{11.6}$$

Example 11.10 *The up-sampling and its inverse are illustrated in Fig. 11.19.*

The up-sampling property is summarized in the following proposition.

Proposition 11.2 *If*

$$\begin{aligned} y[n] &= U_L\left(x[n]\right) = \begin{cases} x\left[n/L\right] & n = kL, k \in \mathbb{Z} \\ 0 & \textit{else.} \end{cases} \\ &= \sum_{k=-\infty}^{+\infty} x[k]\delta[n-kL], \end{aligned} \tag{11.7}$$

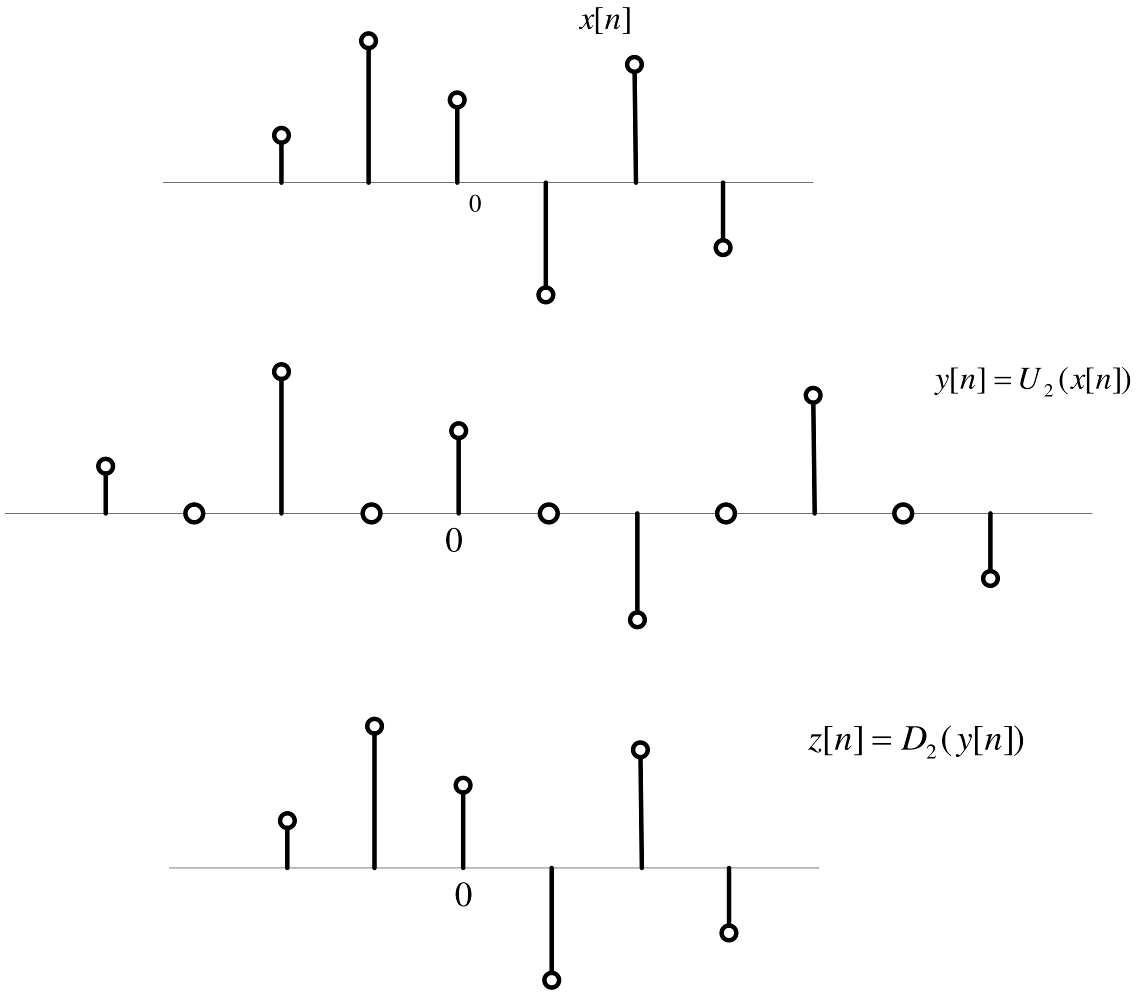


Figure 11.19: Up-sampling and down-sampling.

then

$$Y(z) = X\left(z^L\right), \tag{11.8}$$

and if $|z| = 1 \in \mathrm{ROC}_x$,

$$Y(e^{j\omega}) = X\left(e^{j\omega L}\right). \tag{11.9}$$

From (11.9) it is clear that up-sampling determines a contraction of the frequency axis.

Example 11.11 *Let $x[n] = x_c(nT_s)$ corresponds to the spectrum shown in Fig. 11.20, where $T_s = 2\Omega_N$ and $y[n] = U_2(x[n])$.*

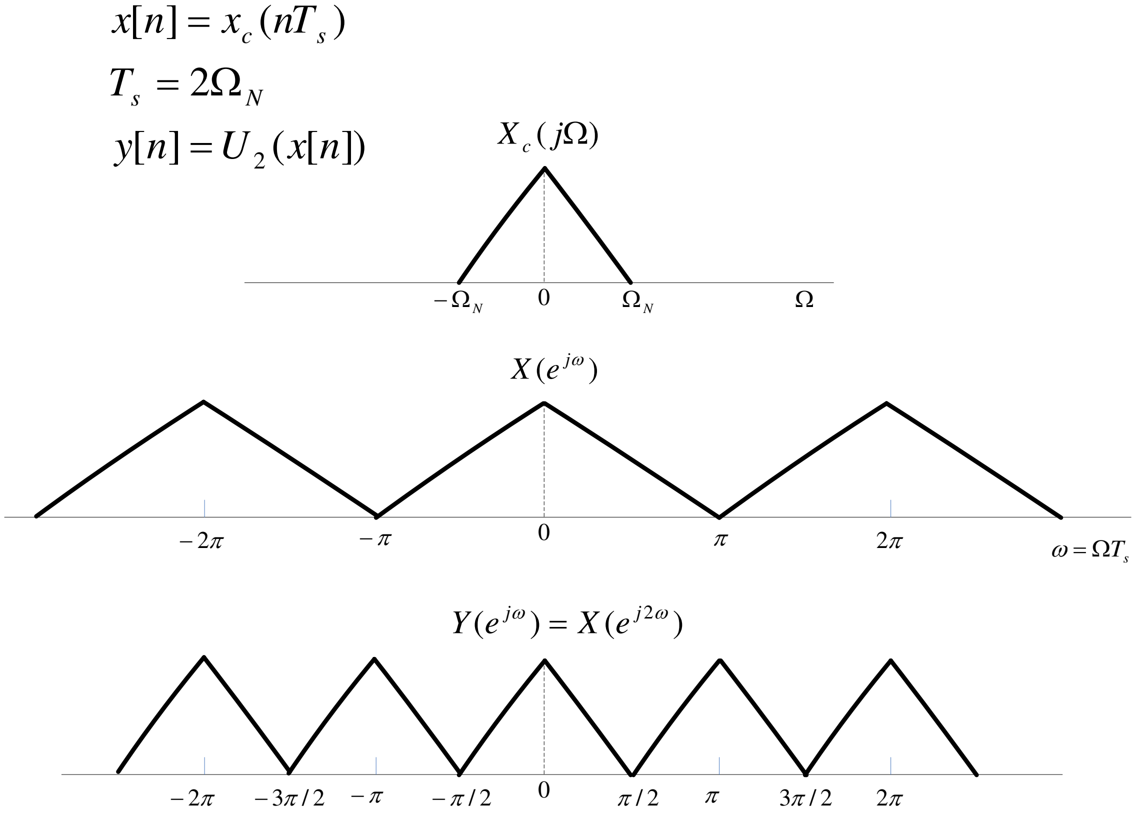


Figure 11.20: Up-sampling contracts the frequency axis.

Recall that the goal of up-sampling is to obtain samples at a higher rate than available. Clearly such a goal is impossible if we had not sampled the underlying continuous-time signal at the required sampling rate.x

For example, if we had sample $x[k]$, we could find a reconstruction through optimal interpolation as

$$x_r(t) = \sum_k x[k]\text{sinc}\left(\frac{t - kT_s}{T_s}\right).$$

Then we could resample this at sampling period ${T_s}'$ to produce

$$z[n] = x_r(n{T_s}') = \sum_k x[k]\text{sinc}\left(\frac{n{T_s}' - kT_s}{T_s}\right).$$

If $\frac{{T_s}'}{T_s} = \frac{1}{L}$, then we have

$$z[n] = \sum_k x[k]\text{sinc}\left(\frac{n}{L} - k\right) = \sum_k x[k]\text{sinc}\left(\frac{n-kL}{L}\right).$$

Now

$$\text{sinc}\left(\frac{n-kL}{L}\right) = 0 \quad \text{for} \quad n = kL \pm L, kL \pm 2L, \ldots$$

For $n = kL$,

$$\text{sinc}\left(\frac{n-kL}{L}\right) = 1.$$

Therefore we see that for $n = mL, m \in \mathbb{Z}$,

$$z[mL] = \sum_k x[k]\text{sinc}\left(\frac{mL-kL}{L}\right) = x[m].$$

But it also obtains all the samples in the middle.

Example 11.12 *Look at Fig. 11.21, wherein the sampling period has been changed from* T_s *to* $T'_s = \frac{T_s}{L}$.

Example 11.13 $\quad x_c(t) = \cos(4000\pi t)$

$$T_s = \frac{1}{6000} \quad , \quad x[n] = x_c(nT_s) \quad , \quad \Omega_s = 2\pi \times 6000 = 12000\pi$$

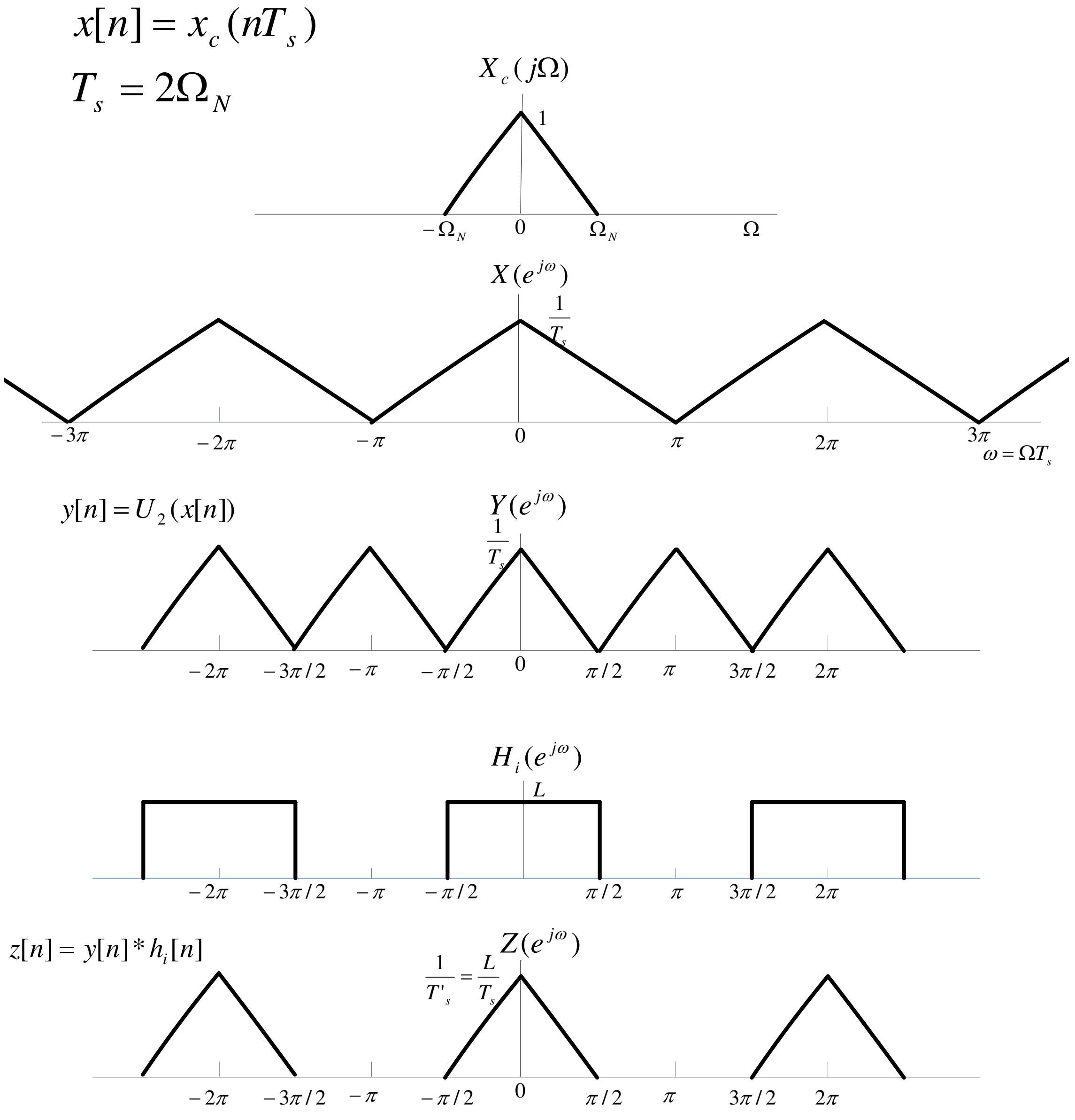
$$x[n] = \cos\left(4000\pi\frac{n}{6000}\right) = \cos\left(\frac{2\pi}{3}n\right)$$

$$y[n] = \text{U}_2\left(x[n]\right) = \begin{cases} x\left[n/2\right] & n = 0, \pm 2, \pm 4, \ldots \\ 0 & \textit{else.} \end{cases}$$

$${T_s}' = \frac{1}{12000} \quad , \quad z[n] = x_c(n{T_s}')$$

$$z[n] = \cos\left(4000\pi\frac{n}{12000}\right) = \cos\left(\frac{\pi n}{3}\right).$$

The relationship between the spectrum of all the signals in this example is illustrated in Fig. 11.22.



$x[n] = x_c(nT_s)$
$T_s = 2\Omega_N$
$X_c(j\Omega)$
1
$-\Omega_N$ 0 Ω_N Ω
$X(e^{j\omega})$
$\frac{1}{T_s}$
-3π -2π $-\pi$ 0 π 2π 3π
$\omega = \Omega T_s$
$y[n] = U_2(x[n])$
$Y(e^{j\omega})$
$\frac{1}{T_s}$
-2π $-3\pi/2$ $-\pi$ $-\pi/2$ 0 $\pi/2$ π $3\pi/2$ 2π
$H_i(e^{j\omega})$
L
-2π $-3\pi/2$ $-\pi$ $-\pi/2$ $\pi/2$ π $3\pi/2$ 2π
$z[n] = y[n] * h_i[n]$
$Z(e^{j\omega})$
$\frac{1}{T'_s} = \frac{L}{T_s}$
-2π $-3\pi/2$ $-\pi$ $-\pi/2$ 0 $\pi/2$ π $3\pi/2$ 2π

Figure 11.21: changing the up-sampling rate for $\frac{T'_s}{T_s} = \frac{1}{L}$.

11.4 Changing sampling rate by a rational non-integer factor

Till now, we have shown how to increase or decrease the sampling rate of a sequence by an integer factor. By combining downsampling and upsampling, it is possible to change

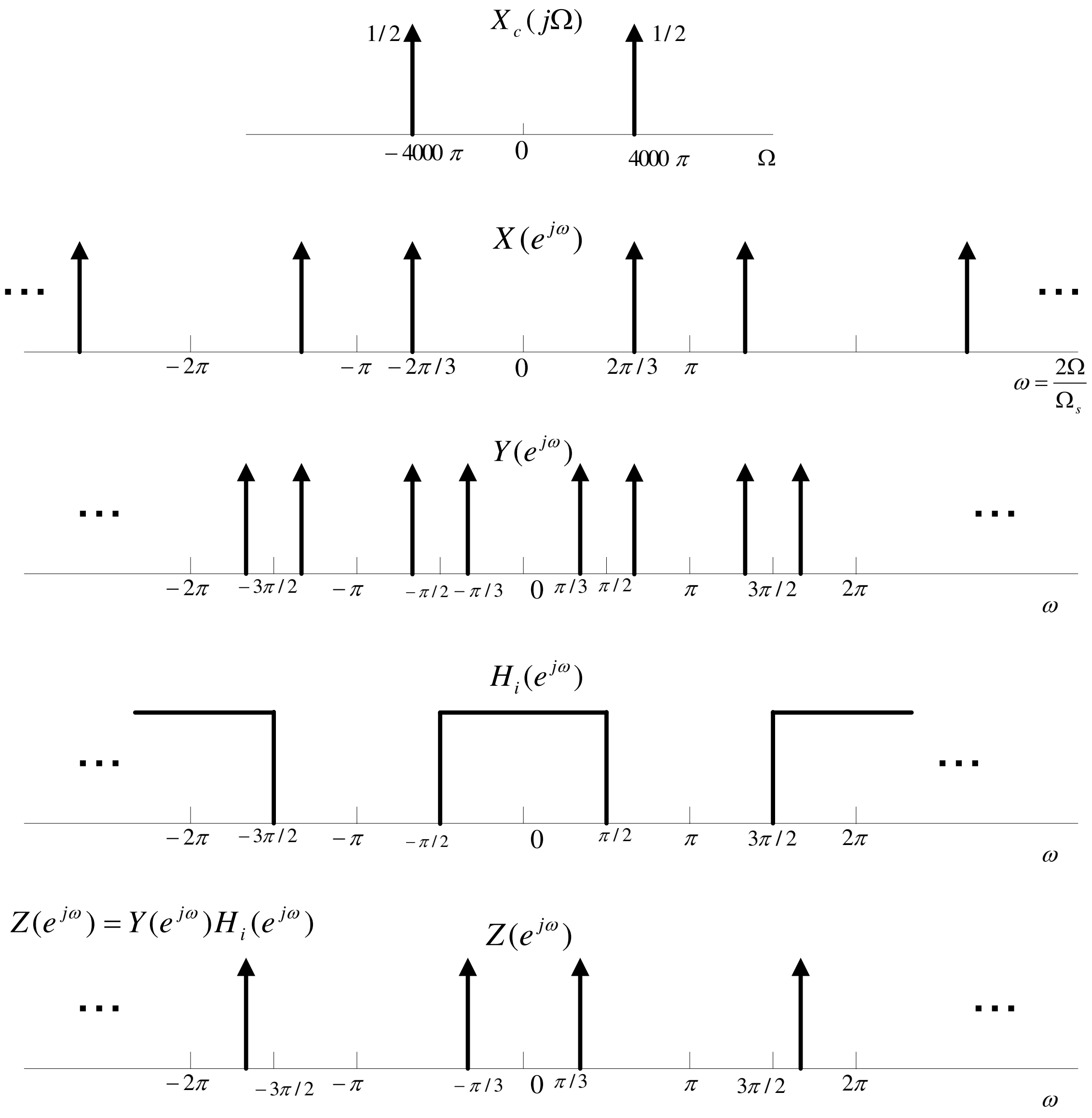


Figure 11.22: Example. 11.13

the sampling rate by a non-integer (rational) factor.

We have seen till now two operations illustrated in Figures 11.23 and 11.24.

Specifically we can cascade an upsampling and downsampling operation to produce a

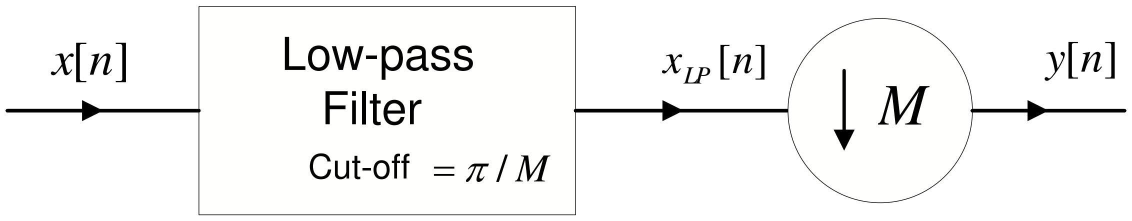


Figure 11.23: Filtering followed by downsampling

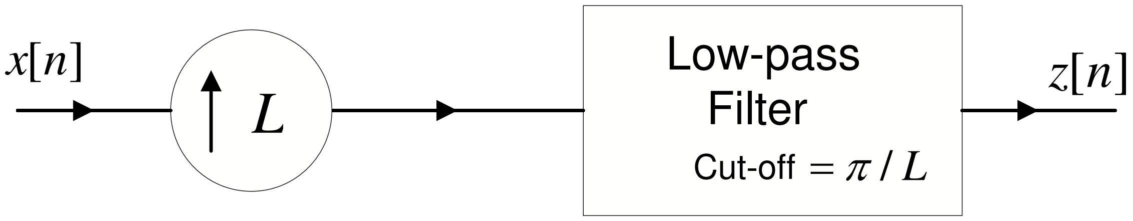


Figure 11.24: Upsampling followed by interpolation

rational sampling frequency change as illustrated in Figure 11.25. This can be simplified as in Figure 11.26.

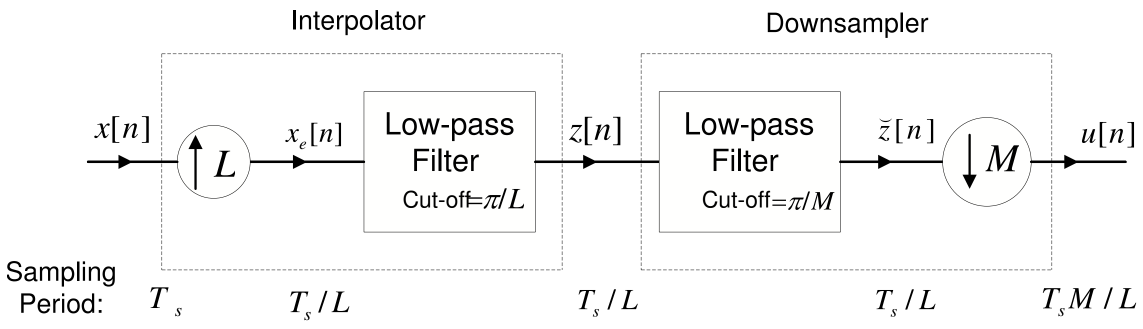


Figure 11.25: System for changing sampling rate by factor $\frac{M}{L}$

Example 11.14 *If* $x_c(t) = \cos(4000\pi t)$, $T_s = \frac{1}{6000}$.

$$x[n] = x_c\left(n\frac{1}{6000}\right) = \cos\left(\frac{2\pi}{3}n\right).$$

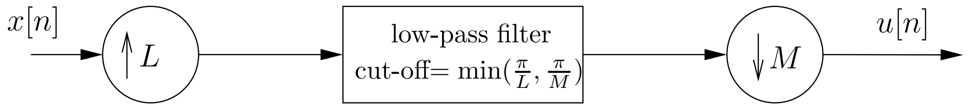


Figure 11.26: Same system as Figure 11.25 which is simplified by combining the two low-pass filters

We want to sample at $T_s{}' = \frac{1}{8000}$*, i.e.* $\frac{T_s}{T_s{}'} = \frac{8000}{6000} = \frac{4}{3}$.

That is we finally want,

$$u[n] = x_c(nT_s{}') = \cos\left(\frac{4000\pi n}{8000}\right) = \cos\left(\frac{\pi}{2}n\right).$$

Can we do this through discrete-time operations on $x[n]$*?*

Since $T_s{}' = \frac{3}{4}T_s$*, we have* $M = 3, L = 4$ *for down-sampling and up-sampling factors. This has been shown in Fig. 11.27.*

Clearly

$$\underbrace{\cos\left(\frac{\pi}{2}n\right)}_{u[n]} \overset{DTFT}{\Longleftrightarrow} \underbrace{\frac{1}{2}\tilde{\delta}\left(\omega - \frac{\pi}{2}\right) + \frac{1}{2}\tilde{\delta}\left(\omega + \frac{\pi}{2}\right)}_{U(e^{j\omega})}$$

Example 11.15 [*Sampling rate conversion for digital audio.*]

If we want to go from 44.1kHz sampling rate to 48.0kHz sampling rate we need:

$$T_s = \frac{10^{-3}}{44.1} \qquad , \qquad T_s{}' = \frac{10^{-3}}{48}$$

$$\frac{T_s{}'}{T_s} = \frac{44.1}{48} = \frac{14.7}{16} \quad or \quad T_s{}' = T_s\left(\frac{147}{160}\right).$$

Therefore if $M = 147, L = 160$*, we can go from 44.1kHz to 48kHz sampling rate.*

Example 11.16 *Suppose* $T_s{}' = \left(\frac{3}{2}\right)T_s$*, we have* $L = 2, M = 3$*. Look at Fig. 11.28*

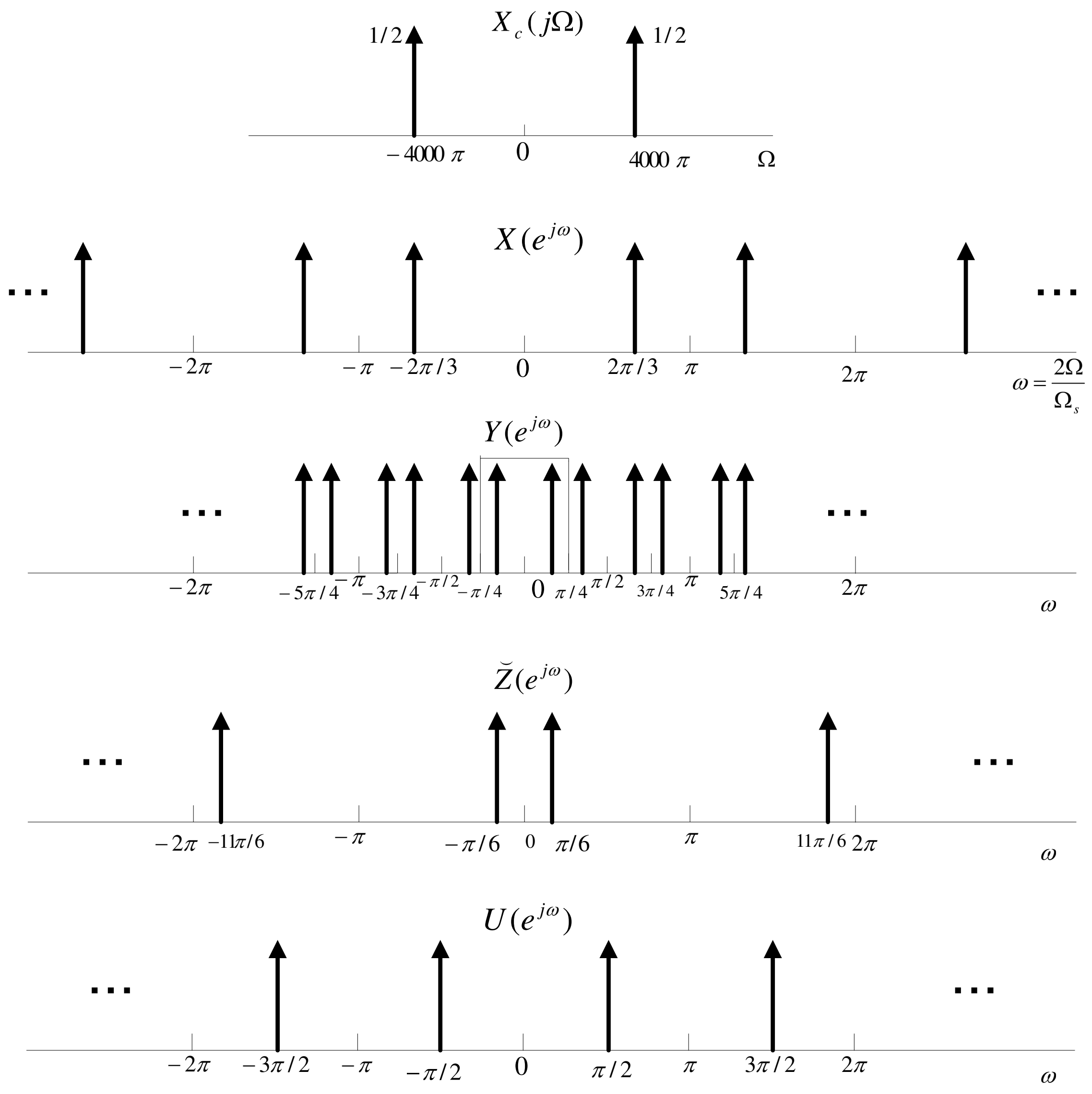


Figure 11.27: Changing the sampling rate by a rational non-integer factor.

11.5 Interchange of filtering and down-sampling/up-sampling

In this section we examine an important property of down-sampling and upsampling process. Consider the two systems shown in Fig. 11.29.

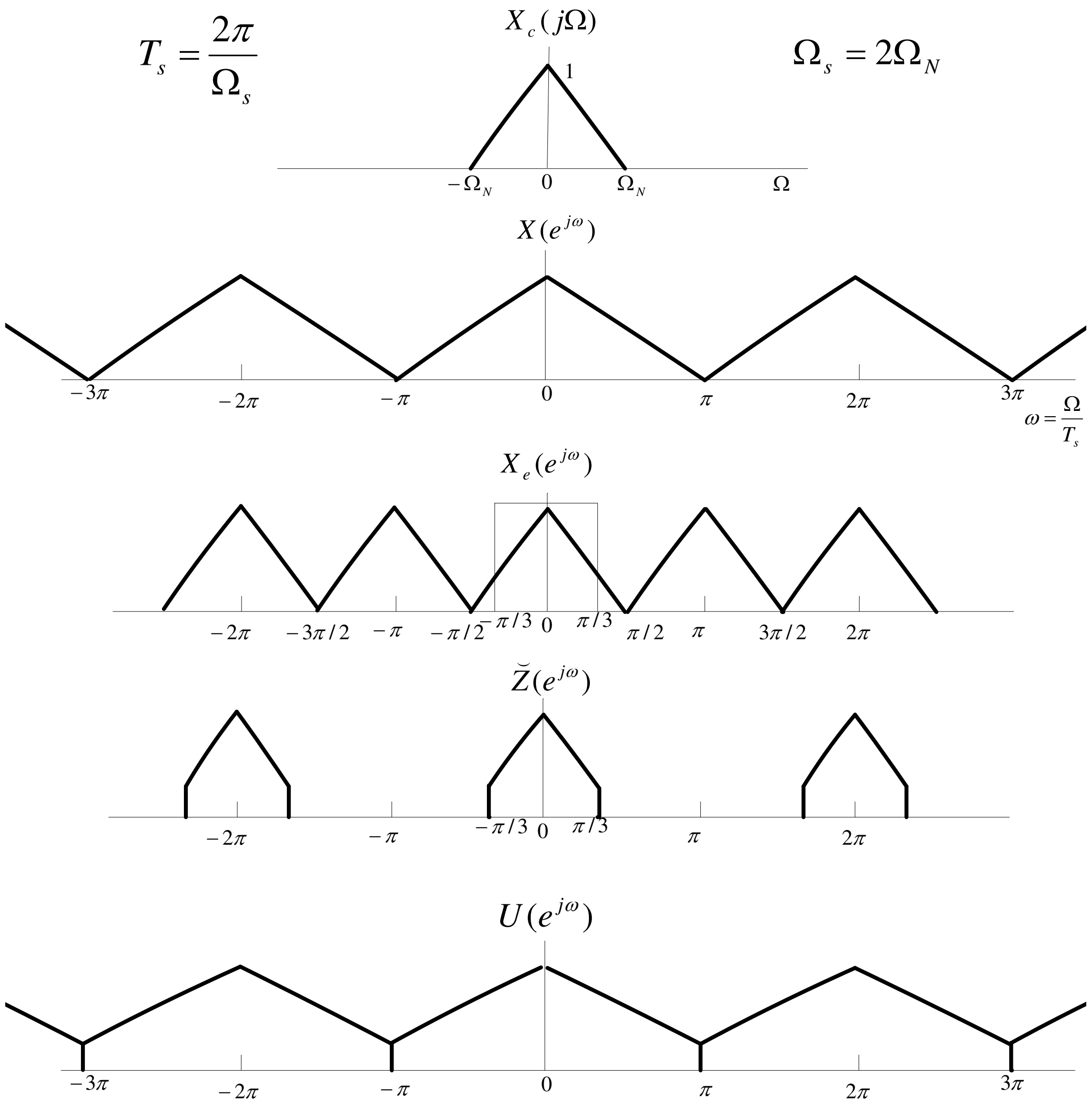


Figure 11.28: Example. 11.16: up-sampling and down-sampling with $L = 2$ and $M = 3$.

We show that these two systems are equivalent first through an example and then more formally.

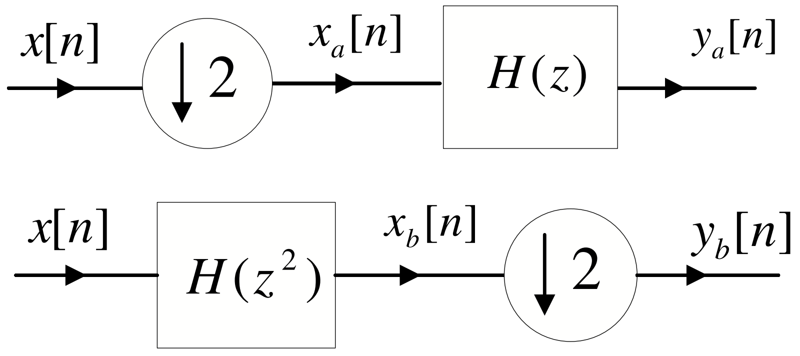


Figure 11.29: Two equivalent systems based on downsampling identities.

Example 11.17

$$\begin{aligned}
X_a\left(e^{j\omega}\right) &= \frac{1}{2}\left\{X\left(e^{j\omega/2}\right)+X\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right\}\\
Y_a\left(e^{j\omega}\right) &= H\left(e^{j\omega}\right)X_a\left(e^{j\omega}\right)=\frac{1}{2}H\left(e^{j\omega}\right)\left\{X\left(e^{j\omega/2}\right)+X\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right\}\\
X_b\left(e^{j\omega}\right) &= H\left(e^{j2\omega}\right)X\left(e^{j\omega}\right)
\end{aligned}$$

$$\begin{aligned}
Y_b\left(e^{j\omega}\right) &= \frac{1}{2}\left\{X_b\left(e^{j\omega/2}\right)+X_b\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right\}\\
&= \frac{1}{2}\left\{H\left(e^{j\omega}\right)X\left(e^{j\omega/2}\right)+H\left(e^{j\left(\frac{\omega}{2}-\pi\right)2}\right)X\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right\}\\
&= \frac{1}{2}H\left(e^{j\omega}\right)\left\{X\left(e^{j\omega/2}\right)+X\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right\}.
\end{aligned}$$

Hence we see that $Y_a\left(e^{j\omega}\right)=Y_b\left(e^{j\omega}\right)$.

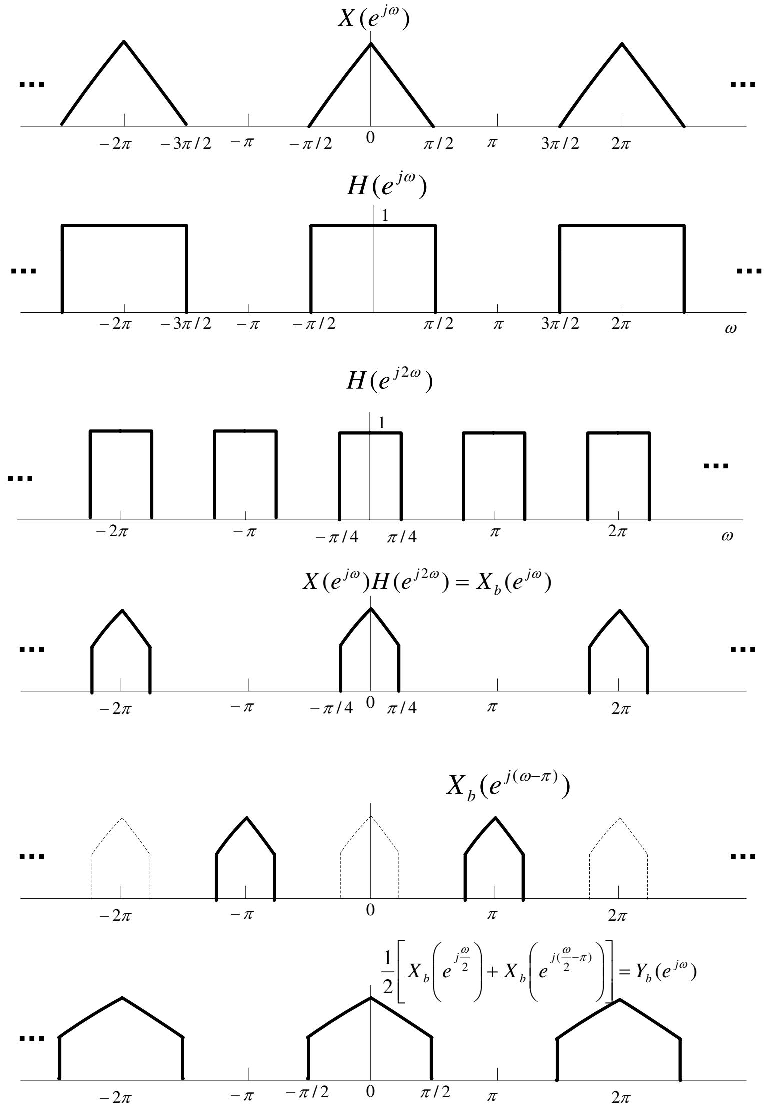


Figure 11.30: Example 11.17: Comparison of the spectrun of two equivalence systems.

The example suggests the following simple formal proof for the equivalence of the Fig. 11.29.

We know that if

$$y[n] = \mathrm{D}_2\left(x_f[n]\right),$$

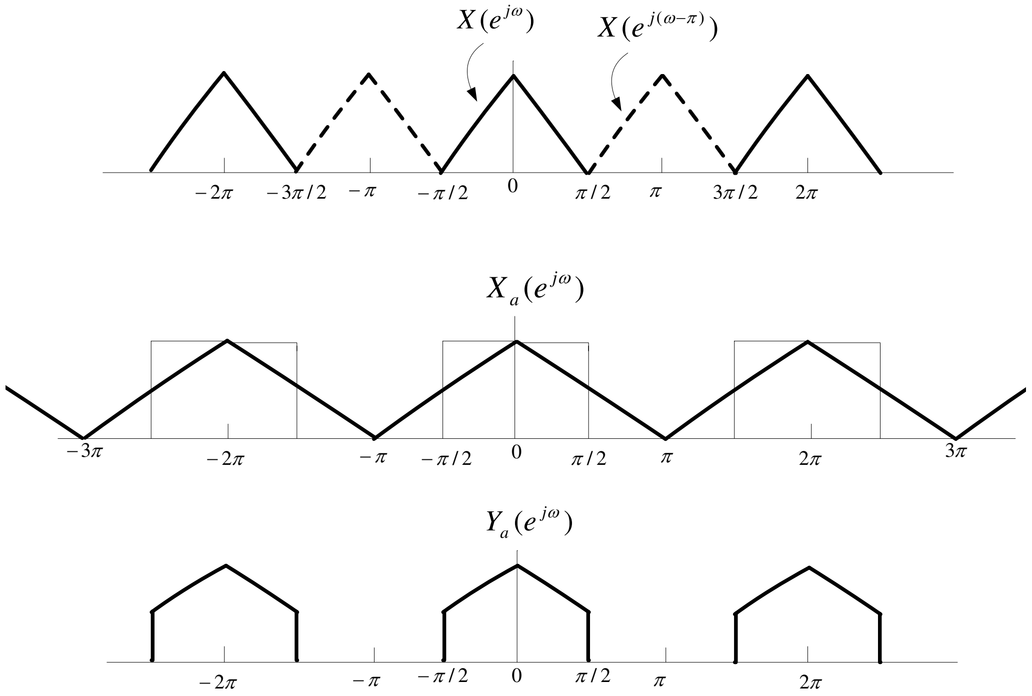


Figure 11.31: Continuation of Fig. 11.30 for Example 11.17.

then

$$Y(z) = \frac{1}{2}\left[X_f\left(z^{1/2}\right) + X_f\left(-z^{1/2}\right)\right],$$

or if $|z| = 1$ is in the ROC,

$$Y\left(e^{j\omega}\right) = \frac{1}{2}\left[X_f\left(e^{j\omega/2}\right) + X_f\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right].$$

Now since $x_a[n] = \mathrm{D}_2\left(x[n]\right)$,

$$X_a(z) = \frac{1}{2}\left\{X\left(z^{1/2}\right) + X\left(-z^{1/2}\right)\right\},$$

and

$$Y_a(z) = H(z)X_a(z) = \frac{1}{2}H(z)\left\{X\left(z^{1/2}\right) + X\left(-z^{1/2}\right)\right\}. \tag{11.10}$$

Next,

$$X_b(z) = H\left(z^2\right)X(z),$$

therefore,

$$X_b\left(z^{1/2}\right) = H\left(\left(z^{1/2}\right)^2\right) X\left(z^{1/2}\right) = H(z) X\left(z^{1/2}\right),$$

and

$$X_b\left(-z^{1/2}\right) = H\left(\left(-z^{1/2}\right)^2\right) X\left(-z^{1/2}\right) = H(z) X\left(-z^{1/2}\right).$$

Hence,

$$\begin{aligned} Y_b(z) &= \frac{1}{2}\left\{X_b\left(z^{1/2}\right) + X_b\left(-z^{1/2}\right)\right\} \\ &= \frac{1}{2} H(z)\left\{X\left(z^{1/2}\right) + X\left(-z^{1/2}\right)\right\} \\ &= Y_a(z). \quad \blacksquare \end{aligned}$$

A similar identity applies to up-sampling as shown in Fig. 11.32.

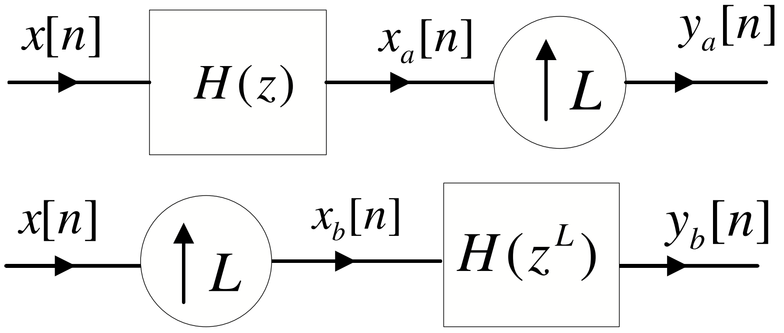


Figure 11.32: Two equivalent systems based on upsampling identities.

This can be seen from the following.

$$Y_a(z) = X_a\left(z^L\right) = H\left(z^L\right) X\left(z^L\right),$$

and

$$Y_b(z) = H\left(z^L\right) X_b(z) = H\left(z^L\right) X\left(z^L\right).$$

Hence

$$Y_a(z) = Y_b(z),$$

and since the ROC's are the same,

$$y_a[n] = y_b[n].$$

Example 11.18 (Polyphase Implementation of Downsampling) *Consider the downsampling system given in Figure 11.33, where $H(z)$ is an arbitrary filter with impulse*

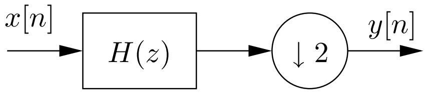


Figure 11.33: Downsampling system for Exercise 11.18.

response $h[n]$.

We define

$$e_0[n] = h[2n], \qquad \textit{and} \qquad e_1[n] = h[2n+1].$$

Prove that the system of Figure 11.34 is equivalent to the one given in Figure 11.33.

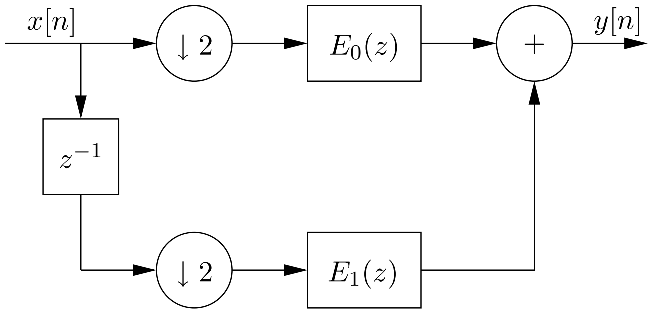


Figure 11.34: Equivalent system.

Solution:

Using one of the noble identities on Figure 11.34 results in Figure 11.35. Downsampling with a factor two is linear, so can perform it after the addition.

Now define $g_0[n]$ to be the impulse response corresponding to $E_0(z^2)$. We have

$$g_0[n] = \begin{cases} h[n], & n \text{ even,} \\ 0, & n \text{ odd.} \end{cases}$$

Similarly, defining $g_1[n]$ to be the impulse response corresponding to $E_1(z^2)$, we have

$$g_1[n] = \begin{cases} h[n+1], & n \text{ even,} \\ 0, & n \text{ odd.} \end{cases}$$

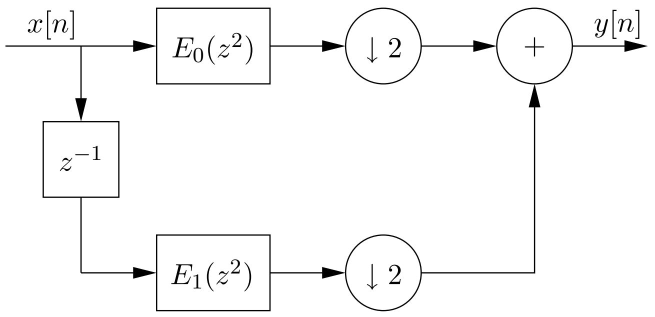


Figure 11.35: Equivalent system.

Now if we consider the filtering operation without downsampling, we have

$$\begin{aligned} y[n] &= \sum_{l=-\infty}^{\infty} h[l]x[n-l] \\ &= \sum_{k=-\infty}^{\infty} h[2k]x[n-2k] + \sum_{k=-\infty}^{\infty} h[2k+1]x[n-2k-1], \end{aligned}$$

which we see is exactly the operation performed by the system in Figure 11.35.

Example 11.19 (Haar Decomposition) *In this example we consider a filter bank where the analysis and synthesis filters are not perfect low and high-pass filters.*

We start with the 2 component analysis filter bank given in Figure 11.36. The impulse

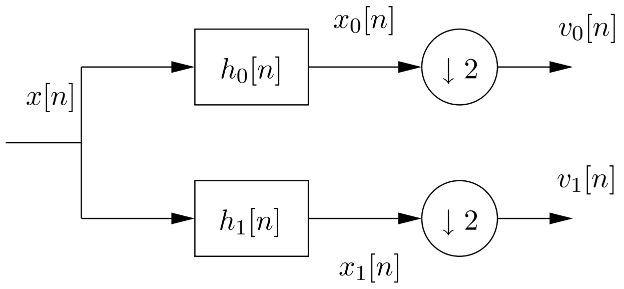


Figure 11.36: Analysis filter bank for Exercise 11.19.

responses of the filters $h_0[n]$ and $h_1[n]$, in this case are given by

$$h_0[n] = \begin{cases} 1, & \textit{if } n = 0, 1 \\ 0, & \textit{otherwise} \end{cases}, \qquad h_1[n] = \begin{cases} -1, & \textit{if } n = 0 \\ 1, & \textit{if } n = 1 \\ 0, & \textit{otherwise.} \end{cases}$$

(a) *Consider*

$$\begin{bmatrix} v_0[n] \\ v_1[n] \end{bmatrix} = A \begin{bmatrix} x[2n-1] \\ x[2n] \end{bmatrix}.$$

Find A. Show that it is orthonormal.

(b) *Show that $x[2n-1]$ and $x[2n]$ can be recovered from $v_0[n]$ and $v_1[n]$.*

In the second part of this exercise we will find the synthesis filter bank that reconstructs $x[n]$ from $v_0[n]$ and $v_1[n]$. The structure of the synthesis filter bank is given in Figure 11.37. Let

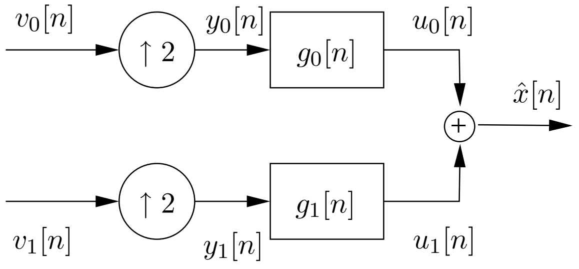


Figure 11.37: Synthesis filter bank for Example 11.19.

$$g_0[n] = \frac{1}{2} h_0[-n+1], \qquad \textit{and } g_1[n] = \frac{1}{2} h_1[-n+1].$$

(c) *Find $u_0[n]$ and $u_1[n]$ and prove that $\hat{x}[n] = x[n-1]$.*

Solution:

(a)

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

The rows of A are orthogonal, since

$$\Big\langle \begin{bmatrix} 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & -1 \end{bmatrix} \Big\rangle = 0.$$

(b) *Since the rows of A are orthogonal it is full rank and hence invertible. We have*

$$\begin{bmatrix} x[2n-1] \\ x[2n] \end{bmatrix} = A^{-1} \begin{bmatrix} v_0[n] \\ v_1[n] \end{bmatrix},$$

where

$$A^{-1} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Let

$$g_0[n] = \frac{1}{2} h_0[-n+1], \qquad \text{and } g_1[n] = \frac{1}{2} h_1[-n+1].$$

(c) *We have*

$$\begin{aligned} v_0[n] &= x[2n] + x[2n-1] \\ v_1[n] &= -x[2n] + x[2n-1]. \end{aligned}$$

$$g_0[n] = \frac{1}{2} h_0[-n+1] = \begin{cases} \frac{1}{2}, & n = 0 \\ \frac{1}{2}, & n = 1 \\ 0, & \text{otherwise}, \end{cases}$$

$$g_1[n] = \frac{1}{2} h_1[-n+1] = \begin{cases} \frac{1}{2}, & n = 0 \\ -\frac{1}{2}, & n = 1 \\ 0, & \text{otherwise}. \end{cases}$$

$$y_0[n] = \begin{cases} x[n] + x[n-1], & n \text{ even} \\ 0, & n \text{ odd}, \end{cases}$$

$$y_1[n] = \begin{cases} -x[n] + x[n-1], & n \text{ even} \\ 0, & n \text{ odd}. \end{cases}$$

$$u_0[n] = \frac{1}{2}\left(y_0[n] + y_0[n-1]\right) = \begin{cases} \frac{1}{2}\left(x[n] + x[n-1]\right), & n \text{ even} \\ \frac{1}{2}\left(x[n-1] + x[n-2]\right), & n \text{ odd}, \end{cases}$$

$$u_1[n] = \frac{1}{2}\left(y_1[n] - y_1[n-1]\right) = \begin{cases} \frac{1}{2}\left(-x[n] + x[n-1]\right), & n \text{ even} \\ \frac{1}{2}\left(x[n-1] - x[n-2]\right), & n \text{ odd}, \end{cases}$$

$$\hat{x}[n] = \begin{cases} x[n-1], & n \text{ even} \\ x[n-1], & n \text{ odd}. \end{cases}$$

11.6 Sub-band decompositions

As we have seen, it is possible to change the sampling rate of a discrete-time signal by a combination of upsampling (with interpolation) and downsampling. Multirate techniques refer in general to utilizing upsampling, downsampling and filtering in a variety of ways to analyze and process a discrete-time signal. Filterbanks take a discrete-time signal and pass then through a parallel set *bank* of filtering and multirate operations. These ideas have become the corner-stone of modern signal processing techniques.

In digital audio (for example MP3), the representation of audio signal is based on human auditory perception. The goal is to encode the audio signal in a manner that is *transparent* perceptually, i.e. to avoid annoying auditory effects. This is where the model of auditory system as a *filterbank* is critically used. This also motivates the study of filterbanks which are central to multirate signal processing.

11.6.1 Perceptual models

As a first approximation, the human auditory systems analyzes the sounds by passing it through a bank of filters as shown in Fig. 11.38.

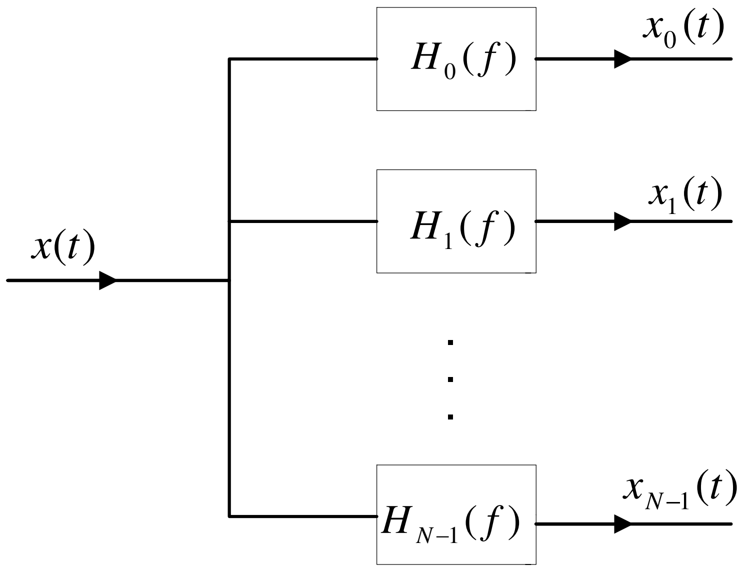


Figure 11.38: Filterbank model of human auditory system

The auditory filters have been empirically (experimentally) characterized and are illustrated (approximately) in Fig. 11.39.

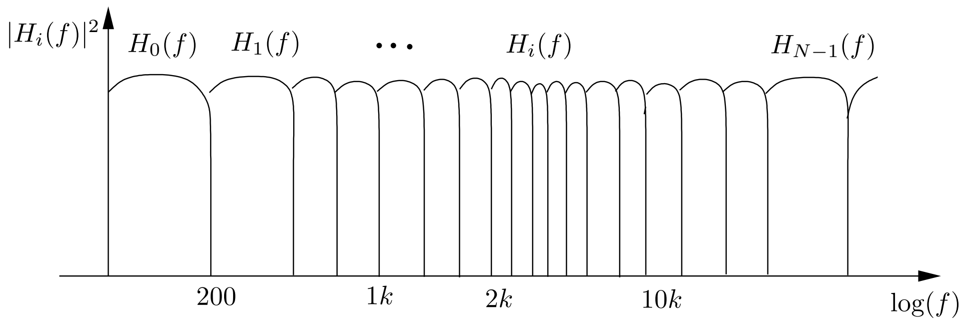


Figure 11.39: Approximate frequency responses of auditory filters

The human auditory system is able to distinguish signals across these filters but not within. That is, the output of each filter is a single entity with the dominant (highest energy) signal within that band masking all other signals in the same band. This masking process is quite complicated and is still not completely understood, but the basic principle of filterbanks have been very successfully used in digital audio coding. Therefore we will study the basic principles of filterbanks and in this sampling rate conversions are crucial. These ideas also form the basis for other signal representation and transform techniques such as *wavelets* which is a topic for an advanced class on signal processing.

11.7 Sub-band or filterbank decomposition of signals

As seen in the auditory perceptual model, there is value to analyzing and representing signals using a bank of filters. If the filter have (almost) non-overlapping responses (such as a low-pass and high-pass), then we are splitting the signals into frequency sub-bands and hence the terminology sub-band decomposition.

A basic structure for a sub-band or filterbank decomposition is illustrated in Fig. 11.40.

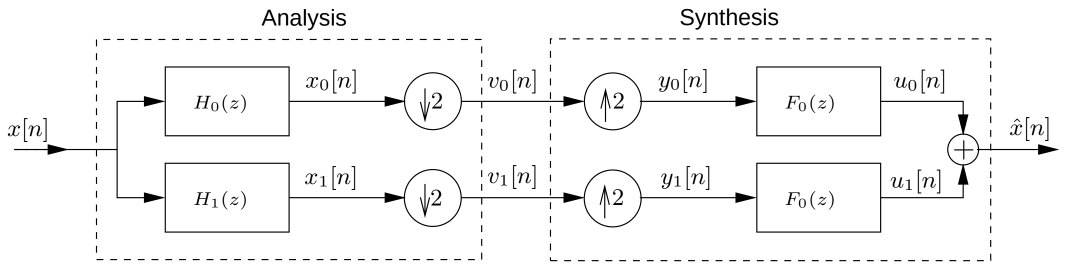


Figure 11.40: Analysis and synthesis of sub-band decomposition. This is also called the two-band quadrature mirror filter (QMF) bank.

Example 11.20 *Look at Figures 11.41, 11.42, 11.43, and 11.45.*

Hence, we see that this filterbank reconstructs $x[n]$ afther the synthesis. Such a filterbank which synthesizes the original signal $x[n]$ (except perhaps sealing and delaying) is called a *perfect reconstruction filterbank.* In this example we achieved this by having *non-overlapping* frequency responses for the *analysis* filters $(H_0(z), H_1(z))$ and the synthesis filters $(F_0(z), F_1(z))$. However, such a perfect reconstruction also holds (surprisingly) for carefully designed filters $H_0(z), H_1(z), F_0(z), F_1(z)$ which might have *overlapping* frequency responses.

To understand the implications of this example more thoroughly, notice that $v_0[n]$ and $v_1[n]$ were sufficient to reconstruct $x[n]$. Therefore, inherently we are *representing* $x[n]$ through $v_0[n]$ and $v_1[n]$. This is because $v_0[n]$ and $v_1[n]$ are downsampled versions of $x_0[n]$ and $x_1[n]$ and hence together have the same number of samples as $x[n]$.

Since $H_0\left(e^{j\omega}\right)$ and $H_1\left(e^{j\omega}\right)$ are orthogonal and span the whole space (i.e. cover all the frequencies), we can easily understand that $x_0[n]$ and $x_1[n]$ allow us to reconstruct $x[n]$. However, the non-trivial observation is that downsampled versions of $x_0[n]$ and $x_1[n]$ i.e. $v_0[n]$, $v_1[n]$ are also sufficient for reconstructing $x[n]$.

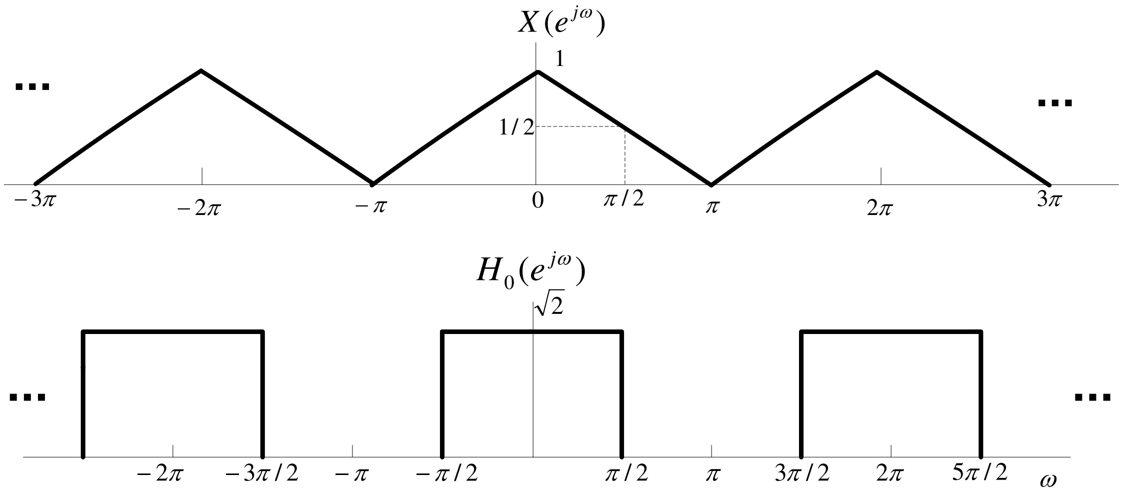


Figure 11.41: Example 11.20.

Since

$$\hat{X}\left(e^{j\omega}\right) = F_0\left(e^{j\omega}\right)Y_0\left(e^{j\omega}\right) + F_1\left(e^{j\omega}\right)Y_1\left(e^{j\omega}\right),$$

and

$$\hat{X}\left(e^{j\omega}\right) = X\left(e^{j\omega}\right),$$

we see that the decomposition is

$$X\left(e^{j\omega}\right) = F_0\left(e^{j\omega}\right)V_0\left(e^{j2\omega}\right) + F_1\left(e^{j\omega}\right)V_1\left(e^{j2\omega}\right).$$

and

$$\begin{aligned}\hat{X}\left(e^{j\omega}\right) &= F_0\left(e^{j\omega}\right)Y_0\left(e^{j\omega}\right) + F_1\left(e^{j\omega}\right)Y_1\left(e^{j\omega}\right)\\ &= F_0\left(e^{j\omega}\right)V_0\left(e^{j2\omega}\right) + F_1\left(e^{j\omega}\right)V_1\left(e^{j2\omega}\right).\end{aligned}$$

Now,

$$\begin{aligned}V_0\left(e^{j\omega}\right) &= \frac{1}{2}\left\{X_0\left(e^{j\omega/2}\right) + X_0\left(e^{j\left(\frac{\omega}{2}-\pi\right)}\right)\right\}\\ &= \frac{1}{2}\left\{X_0\left(e^{j\omega/2}\right) + X_0\left(-e^{j\omega/2}\right)\right\}\end{aligned}$$

Hence,

$$\begin{aligned} V_0\left(e^{j2\omega}\right) &= \frac{1}{2}\left\{X_0\left(e^{j\omega}\right)+X_0\left(-e^{j\omega}\right)\right\} \\ &= \frac{1}{2}\left\{H_0\left(e^{j\omega}\right)X\left(e^{j\omega}\right)+H_0\left(-e^{j\omega}\right)X\left(-e^{j\omega}\right)\right\} \end{aligned}$$

and similarly,

$$\begin{aligned} V_1\left(e^{j2\omega}\right) &= \frac{1}{2}\left\{X_1\left(e^{j\omega}\right)+X_1\left(-e^{j\omega}\right)\right\} \\ &= \frac{1}{2}\left\{H_1\left(e^{j\omega}\right)X\left(e^{j\omega}\right)+H_1\left(-e^{j\omega}\right)X\left(-e^{j\omega}\right)\right\} \end{aligned}$$

Therefore,

$$\begin{aligned} \hat{X}\left(e^{j\omega}\right) &= \frac{1}{2}F_0\left(e^{j\omega}\right)H_0\left(e^{j\omega}\right)X\left(e^{j\omega}\right)+\frac{1}{2}F_0\left(e^{j\omega}\right)H_0\left(e^{j(\omega-\pi)}\right)X\left(e^{j(\omega-\pi)}\right) \\ &+ \frac{1}{2}F_1\left(e^{j\omega}\right)H_1\left(e^{j\omega}\right)X\left(e^{j\omega}\right)+\frac{1}{2}F_1\left(e^{j\omega}\right)H_1\left(e^{j(\omega-\pi)}\right)X\left(e^{j(\omega-\pi)}\right). \end{aligned}$$

But $F_0\left(e^{j\omega}\right)H_0\left(e^{j(\omega-\pi)}\right)=0=F_1\left(e^{j\omega}\right)H_1\left(e^{j(\omega-\pi)}\right)$ since

$$F_0\left(e^{j\omega}\right)=H_0\left(e^{j\omega}\right)=\begin{cases}\sqrt{2} & |\omega|\le\pi/2\\ 0 & \text{else}\end{cases}$$

and

$$F_1\left(e^{j\omega}\right)=H_1\left(e^{j\omega}\right)=\begin{cases}\sqrt{2} & |\omega|>\pi/2\\ 0 & \text{else.}\end{cases}\quad .$$

Therefore,

$$\begin{aligned} \hat{X}\left(e^{j\omega}\right) &= \frac{1}{2}\left|H_0\left(e^{j\omega}\right)\right|^2X\left(e^{j\omega}\right)+\frac{1}{2}\left|H_1\left(e^{j\omega}\right)\right|^2X\left(e^{j\omega}\right) \\ &= \frac{1}{2}\left\{\left|H_0\left(e^{j\omega}\right)\right|^2+\left|H_1\left(e^{j\omega}\right)\right|^2\right\}X\left(e^{j\omega}\right)=X\left(e^{j\omega}\right). \end{aligned}$$

Note that in the example shown above, the analysis and synthesis filters were *ideal*, i.e. perfect low-pass and high-pass filters. This means that we need infinite length filters to implement such a system. A natural question to ask is whether we can obtain such a property by using *finite-length* filters. At first, this seems impossible since finite-length filters will definitely have overlapping spectra. However, this is possible and we illustrate this idea using a signal decomposition method developed by Haar in 1917, which was largely forgotten till a couple of decades ago.

We will look at the Haar signal decomposition in two ways. First through a filter bank and the second through a basis expansion.

Consider the filter $h_0[n]$ with impulse response

$$h_0[n] = \begin{cases} \frac{1}{\sqrt{2}} & n = -1, 0 \\ 0 & \text{else.} \end{cases} \tag{11.11}$$

Note that such a filter is non-causal, but has a finite length and therefore can be implemented using a delay. Consider, as in Figure 11.45,

$$\begin{aligned} x_0[n] &= h_0[n] \star x[n] = h_0[0]x[n] + h_0[1]x[n+1] \\ &= \frac{1}{\sqrt{2}} \{x[n] + x[n+1]\} \end{aligned} \tag{11.12}$$

Now consider the filter $h_1[n]$ with impulse response,

$$h_1[n] = \begin{cases} \frac{1}{\sqrt{2}} & n = 0 \\ -\frac{1}{\sqrt{2}} & n = -1 \\ 0 & \text{else.} \end{cases} \tag{11.13}$$

Again, such a filter is non-causal but FIR and hence implementable. We therefore obtain,

$$x_1[n] = h_1[n] \star x[n] = \frac{1}{\sqrt{2}} \{x[n] - x[n-1]\} \tag{11.14}$$

Observing that $v_0[n]$ and $v_1[n]$ are downsampled versions of $x_0[n]$ and $x_1[n]$ respectively (see Figure 11.45), we get

$$\begin{aligned} v_0[n] &= x_0[2n] = \frac{1}{\sqrt{2}} \{x[2n] + x[2n+1]\} \\ v_1[n] &= x_1[2n] = \frac{1}{\sqrt{2}} \{x[2n] - x[2n+1]\}. \end{aligned} \tag{11.15}$$

Now it is easy to see from (11.15) that $\{v_0[n]\}$ and $\{v_1[n]\}$ sequences are enough to retain *all* the information about $x[n]$ since we see

$$\begin{aligned} v_0[n] + v_1[n] &= \sqrt{2}x[2n] \\ v_0[n] - v_1[n] &= \sqrt{2}x[2n-1]. \end{aligned} \tag{11.16}$$

This observation implies that even by filtering through filters $H_0\left(e^{j\omega}\right)$ and $H_1\left(e^{j\omega}\right)$ with *overlapping* spectra and downsampling, we still retain all the information about the input sequence.

This picture can be completed formally by constructing the corresponding synthesis filter $g_0[n]$ and $g_1[n]$.

$$g_0[n] = h_0[-n] = \begin{cases} \frac{1}{\sqrt{2}} & n = 0, 1 \\ 0 & \text{else.} \end{cases} \tag{11.17}$$

$$g_1[n] = h_1[-n] = \begin{cases} \frac{1}{\sqrt{2}} & n = 0 \\ -\frac{1}{\sqrt{2}} & n = 1 \\ 0 & \text{else,} \end{cases} \tag{11.18}$$

which are now *causal* FIR filters.

Now we have (see Figure 11.45),

$$y_0[n] = \begin{cases} v_0\,[n/2]\,, & n \text{ even} \\ 0, & \text{else} \end{cases} = \begin{cases} \frac{1}{\sqrt{2}}\,(x[n] + x[n+1])\,, & n \text{ even} \\ 0, & \text{else.} \end{cases}$$

$$y_1[n] = \begin{cases} v_1\,[n/2]\,, & n \text{ even} \\ 0, & \text{else} \end{cases} = \begin{cases} \frac{1}{\sqrt{2}}\,(x[n] - x[n+1])\,, & n \text{ even} \\ 0, & \text{else.} \end{cases} \tag{11.19}$$

Therefore, we get using (11.17) and (11.18) in (11.19),

$$\begin{aligned} u_0[n] &= \frac{1}{\sqrt{2}}\,(y_0[n] + y_0[n-1]) = \begin{cases} \frac{1}{\sqrt{2}} y_0[n], & n \text{ even} \\ \frac{1}{\sqrt{2}} y_0[n-1], & n \text{ odd} \end{cases} \\ u_1[n] &= \frac{1}{\sqrt{2}}\,(y_1[n] + y_1[n-1]) = \begin{cases} \frac{1}{\sqrt{2}} y_1[n], & n \text{ even} \\ -\frac{1}{\sqrt{2}} y_1[n-1], & n \text{ odd} \end{cases} \end{aligned} \tag{11.20}$$

yielding

$$\begin{aligned} u_0[n] &= \begin{cases} \frac{1}{2}\,(x[n] + x[n+1])\,, & n \text{ even} \\ \frac{1}{2}\,(x[n-1] + x[n])\,, & n \text{ odd} \end{cases} \\ u_1[n] &= \begin{cases} \frac{1}{2}\,(x[n] - x[n+1])\,, & n \text{ even} \\ \frac{1}{2}\,(x[n] - x[n-1])\,, & n \text{ odd.} \end{cases} \end{aligned} \tag{11.21}$$

Therefore, we get for $\hat{x}[n]$ (see Figure 11.45),

$$\hat{x}[n] = u_0[n] + u_1[n] = \begin{cases} x[n] & n \text{ even} \\ x[n] & n \text{ odd} \end{cases} = x[n]. \tag{11.22}$$

Therefore we get perfect reconstruction.

This illustrates the fact that we can get perfect reconstruction even with analysis (and synthesis) filters with finite length. This immediately means that the spectra of the filters $H_0\left(e^{j\omega}\right)$ and $H_1\left(e^{j\omega}\right)$ are overlapping as shown below in Figure 11.48.

$$\begin{aligned} H_0\left(e^{j\omega}\right) &= \frac{1}{\sqrt{2}}\left(1+e^{-j\omega}\right) \\ H_1\left(e^{j\omega}\right) &= \frac{1}{\sqrt{2}}\left(1-e^{-j\omega}\right) = \frac{1}{\sqrt{2}}\left(1+e^{-j(\omega-\pi)}\right) \\ \left|H_0\left(e^{j\omega}\right)\right| &= \left|H_0\left(e^{j(\omega-\pi)}\right)\right| = [1+\cos\omega]^{1/2} \end{aligned}$$

An alternate interpretation of this property arises by viewing this through basis functions. This viewpoint is quite general and powerful since it leads to the idea of wavelets of which the Haar decomposition is a simple example.

Consider Haar *basis* functions as,

$$\varphi_{2k}[n] = \begin{cases} \frac{1}{\sqrt{2}} & n = 2k, 2k+1 \\ 0 & \text{else.} \end{cases} \tag{11.23}$$

$$\varphi_{2k+1}[n] = \begin{cases} \frac{1}{\sqrt{2}} & n = 2k \\ -\frac{1}{\sqrt{2}} & n = 2k+1 \\ 0 & \text{else,} \end{cases} \tag{11.24}$$

We notice that

$$\varphi_{2k}[n] = \varphi_0[n-2k] \qquad , \qquad \varphi_{2k+1}[n] = \varphi_1[n-2k].$$

Moreover,

$$\langle \varphi_{2k}, \varphi_{2\ell} \rangle = \sum_n \varphi_{2k}^*[n]\varphi_{2\ell}[n] = \delta[k-\ell].$$

$$\langle \varphi_{2k}, \varphi_{2k+1} \rangle = \sum_n \varphi_{2k}^*[n]\varphi_{2k+1}[n] = \frac{1}{\sqrt{2}}\{1-1\} = 0,$$

and for $\ell \neq k$

$$\langle \varphi_{2k}, \varphi_{2\ell+1} \rangle = \sum_n \varphi_{2k}^*[n]\varphi_{2\ell+1}[n] = 0.$$

Hence we see that $\{\varphi_\ell\}_\ell$ form an orthonormal set, i.e.

$$\langle \varphi_p, \varphi_q \rangle = \delta[p-q]. \tag{11.25}$$

Now the Haar analysis function (see Figure 11.45) can be interpreted as a projection by noticing that,

$$v_0[k] = x_0[2k] = \langle \varphi_{2k}, x \rangle = \frac{1}{\sqrt{2}} \{x[2k] + x[2k+1]\}. \quad (11.26)$$

$$v_1[k] = x_1[2k] = \langle \varphi_{2k+1}, x \rangle = \frac{1}{\sqrt{2}} \{x[2k] - x[2k+1]\}. \quad (11.27)$$

Therefore from this point-of-view, it is not surprising that $\{v_0[k]\}$,$\{v_1[k]\}$ sequences are *equivalent* to the original sequence $x[n]$, since they are just representation of $x[n]$ in the basis $\{\varphi_\ell\}_\ell$. Therefore, the reconstruction is also quite simple,

$$\begin{aligned} x[n] &= \sum_{k\in\mathbb{Z}} x_0[2k]\varphi_{2k}[n] + \sum_{k\in\mathbb{Z}} x_1[2k]\varphi_{2k+1}[n] \\ &= \sum_{k\in\mathbb{Z}} v_0[k]\varphi_{2k}[n] + \sum_{k\in\mathbb{Z}} v_1[k]\varphi_{2k+1}[n], \end{aligned} \quad (11.28)$$

as is usual for any orthonormal basis.

We see that (11.28) is clearly equivalent to (11.22), just expressed in the form of basis functions instead of filters. More concretely we see that

$$\begin{aligned} \sum_{k\in\mathbb{Z}} v_0[k]\varphi_{2k}[n] &= \sum_{k} \frac{1}{\sqrt{2}} \{v_0[k]\delta[n-2k] + v_0[k]\delta[n-2k-1]\} \\ &= \frac{1}{\sqrt{2}} \underbrace{\sum_{k\in\mathbb{Z}} v_0[k]\delta[n-2k]}_{y_0[n]=U_2(v_0[n])} + \frac{1}{\sqrt{2}} \underbrace{\sum_{k\in\mathbb{Z}} v_0[k]\delta[n-2k-1]}_{y_0[n-1]} \\ &= \frac{1}{\sqrt{2}} \{y_0[n] + y_0[n-1]\} = u_0[n]. \end{aligned}$$

$$\begin{aligned} \sum_{k\in\mathbb{Z}} v_1[k]\varphi_{2k+1}[n] &= \frac{1}{\sqrt{2}} \underbrace{\sum_{k\in\mathbb{Z}} v_1[k]\delta[n-2k]}_{y_1[n]=U_2(v_1[n])} - \frac{1}{\sqrt{2}} \underbrace{\sum_{k\in\mathbb{Z}} v_1[k]\delta[n-2k-1]}_{y_1[n-1]} \\ &= \frac{1}{\sqrt{2}} \{y_1[n] - y_1[n-1]\} = u_1[n]. \end{aligned}$$

Therefore (11.28) just expresses the same relationship as in (11.22) as

$$\begin{aligned} x[n] &= u_0[n] + u_1[n] \\ &= \frac{1}{\sqrt{2}} \{y_0[n] + y_0[n-1]\} + \frac{1}{\sqrt{2}} \{y_1[n] - y_1[n-1]\}, \end{aligned} \quad (11.29)$$

which was done through a filterbank interpretation.

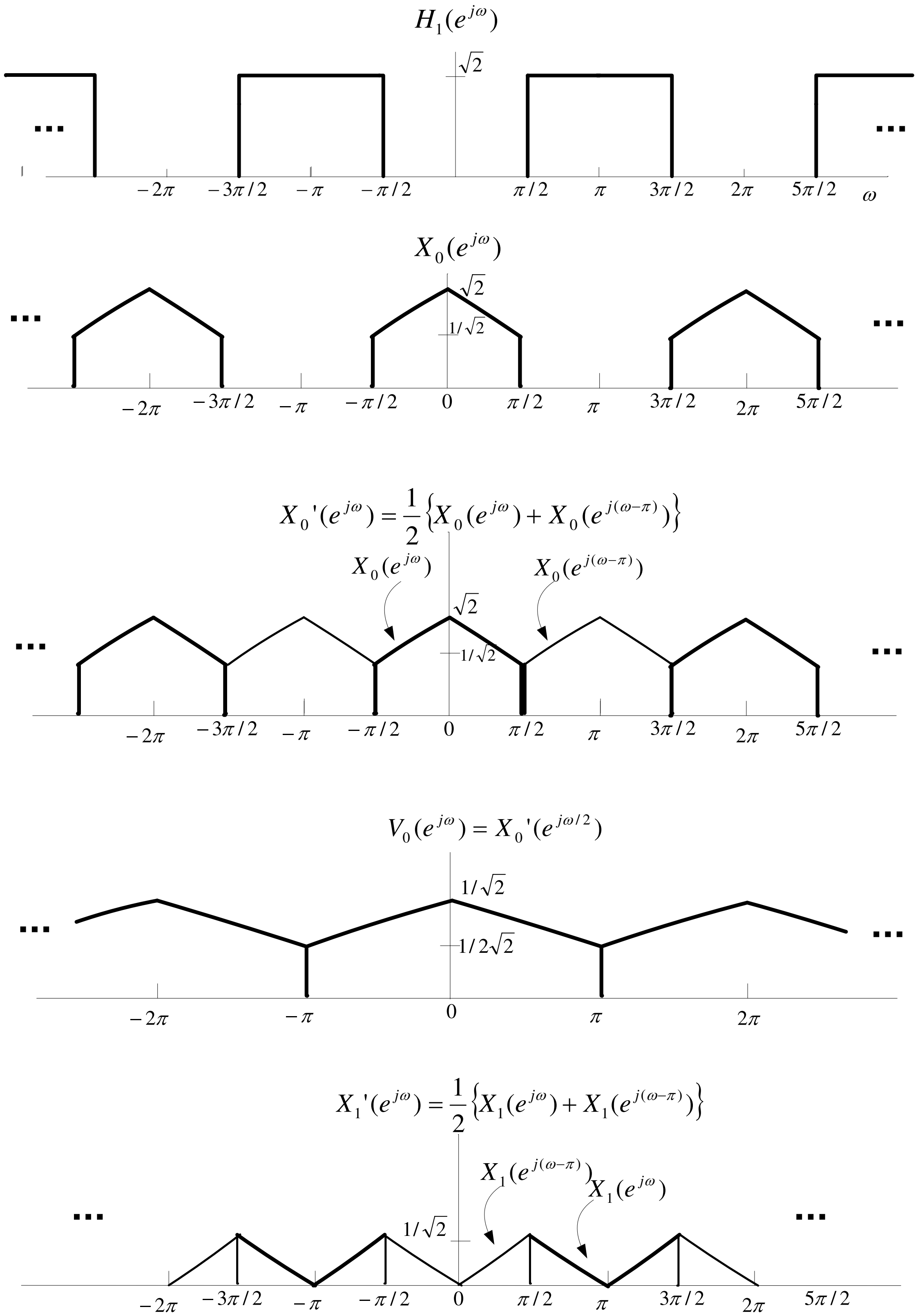


Figure 11.42: Continuation of Fig.11.41: Example 11.20.

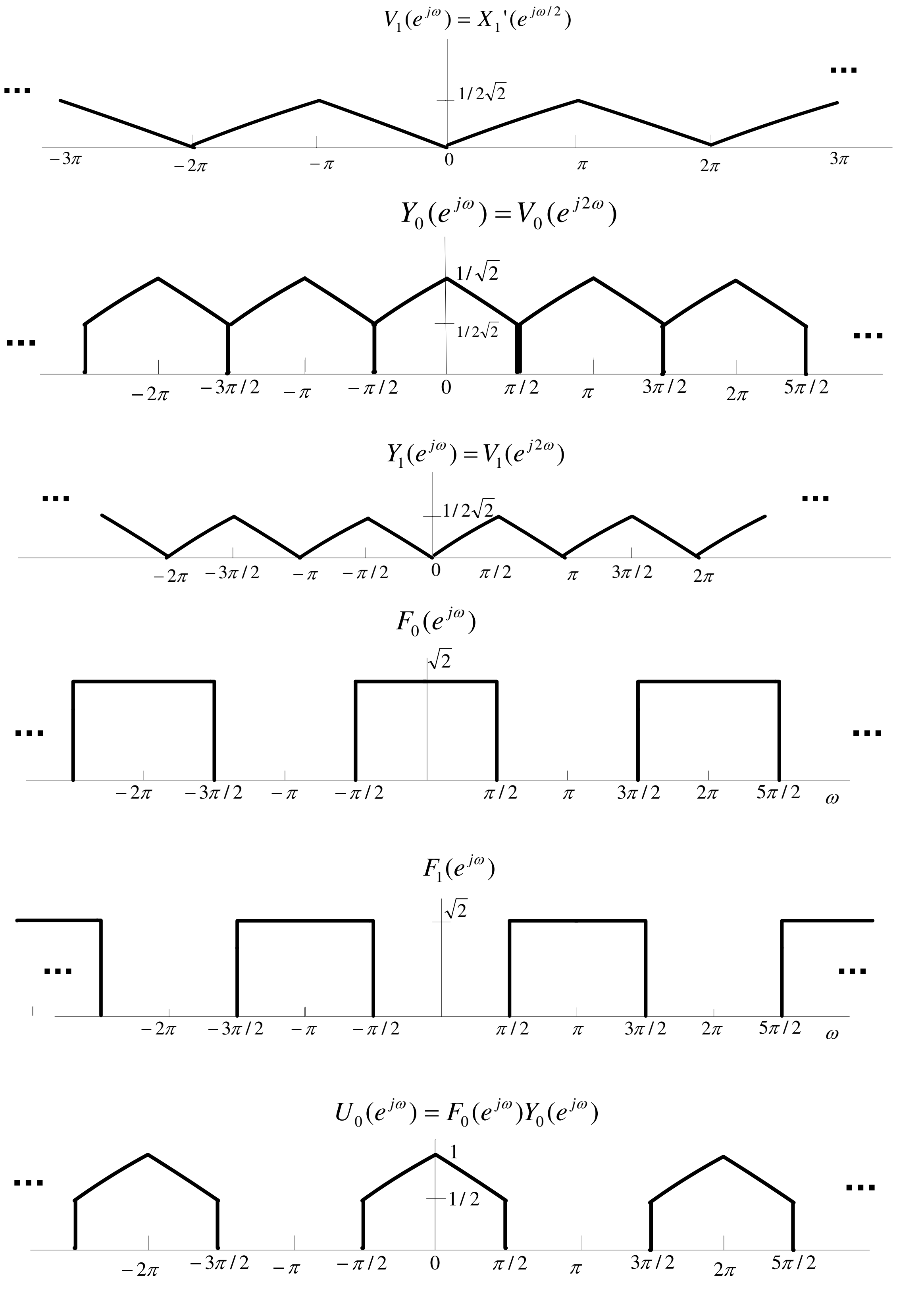


Figure 11.43: Continuation of Fig.11.42: Example 11.20.

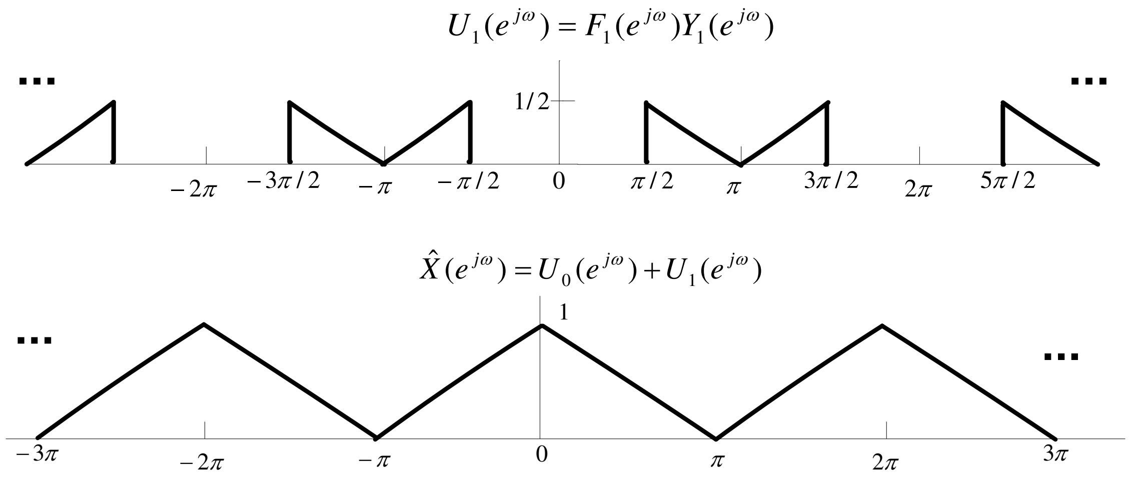


Figure 11.44: Continuation of Fig.11.43: Example 11.20.

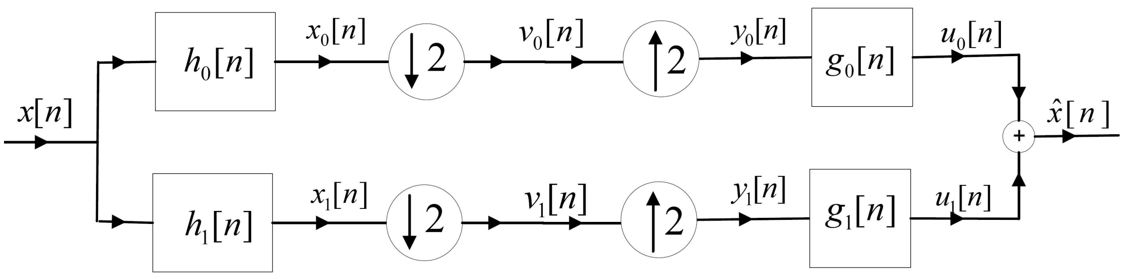


Figure 11.45: Two-channel filter bank representation of the Haar decomposition

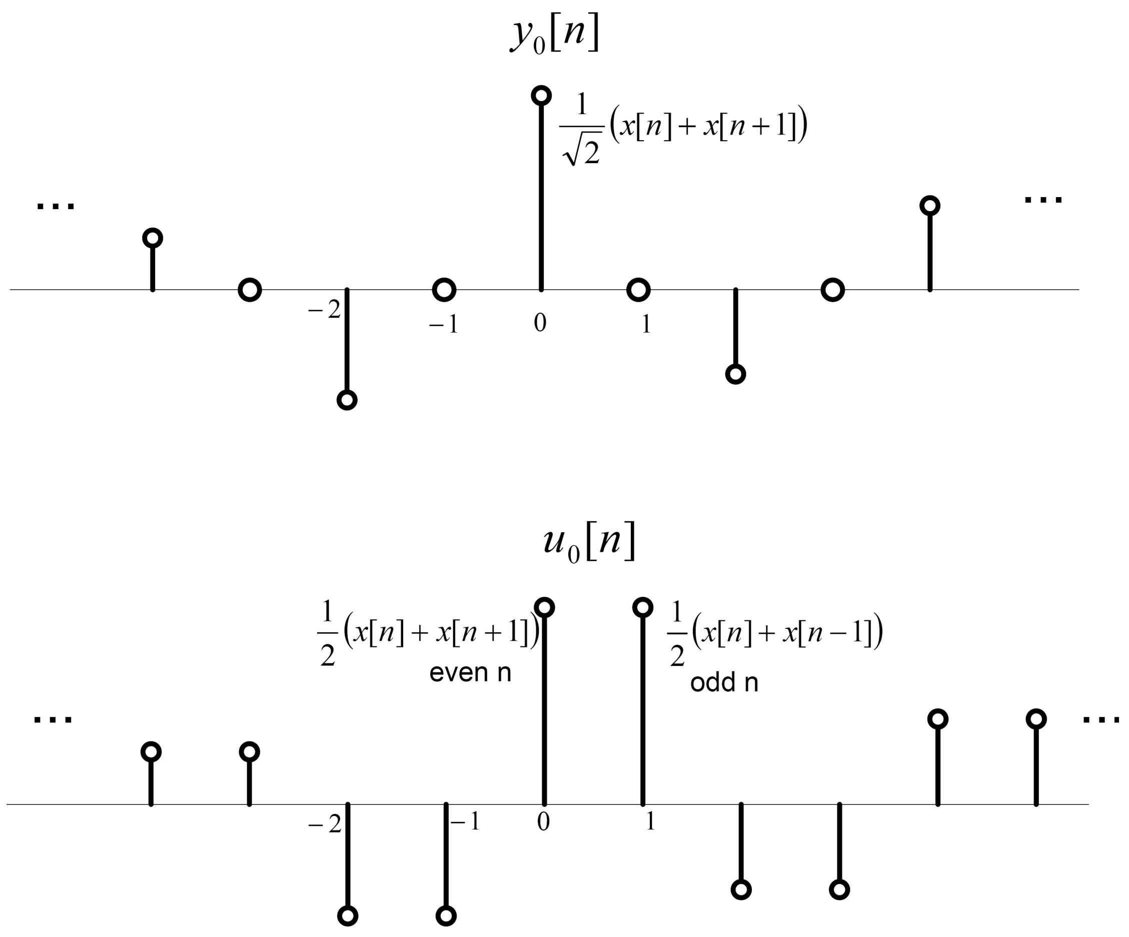

$y_0[n]$
$\frac{1}{\sqrt{2}}(x[n]+x[n+1])$
...
-2
-1
0
1
$u_0[n]$
$\frac{1}{2}(x[n]+x[n+1])$
even n
$\frac{1}{2}(x[n]+x[n-1])$
odd n
-2
-1
0
1

Figure 11.46: Illustration of synthesis filter operations.

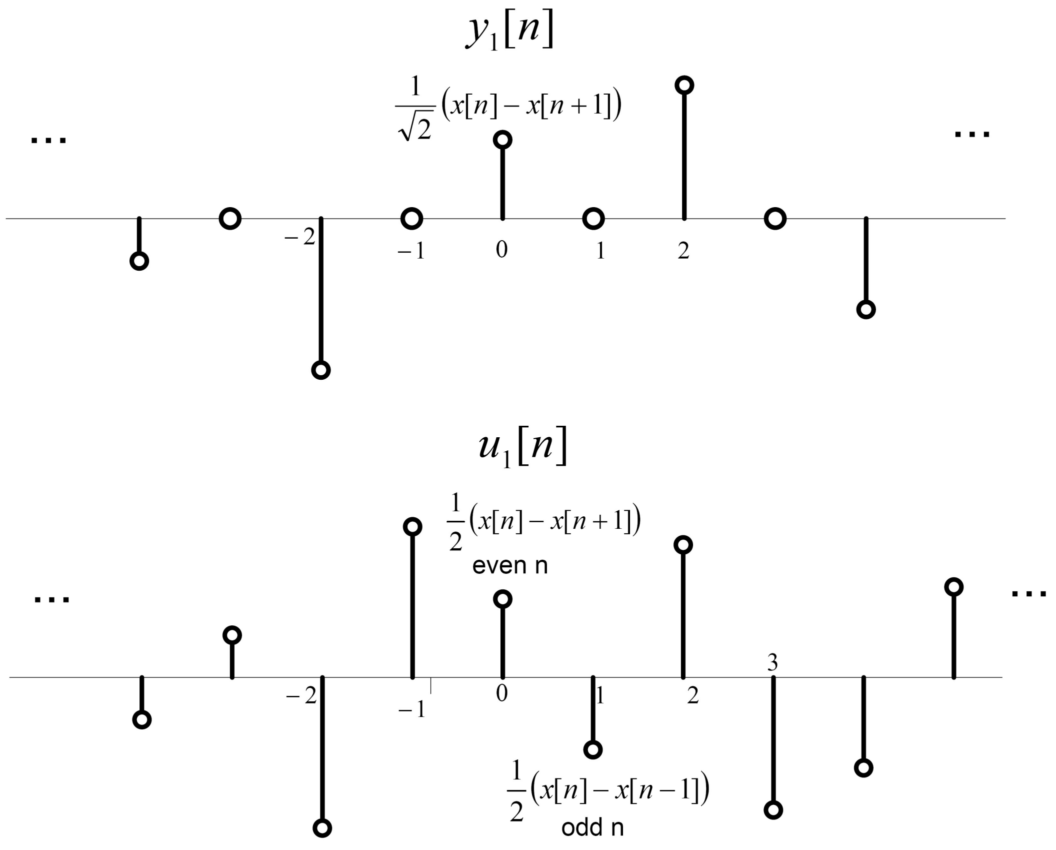

$y_1[n]$
$\frac{1}{\sqrt{2}}(x[n]-x[n+1])$
...
−2
−1
0
1
2
$u_1[n]$
$\frac{1}{2}(x[n]-x[n+1])$
even n
3
$\frac{1}{2}(x[n]-x[n-1])$
odd n

Figure 11.47: Continue to Fig. 11.46: Illustration of synthesis filter operations.

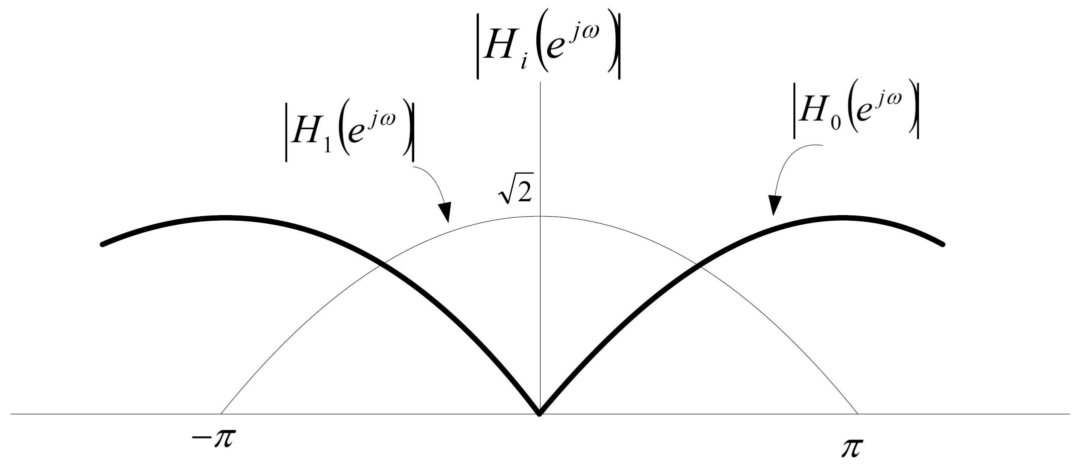


Figure 11.48: Overlapping spectra of analysis filters.

11.8 Problems

Problem 11.1 *Prove the equivalence of the two down-sampling and up-sampling with interpolator configurations shown in Fig. 11.49. These equivalent relations are called the "noble identities".*

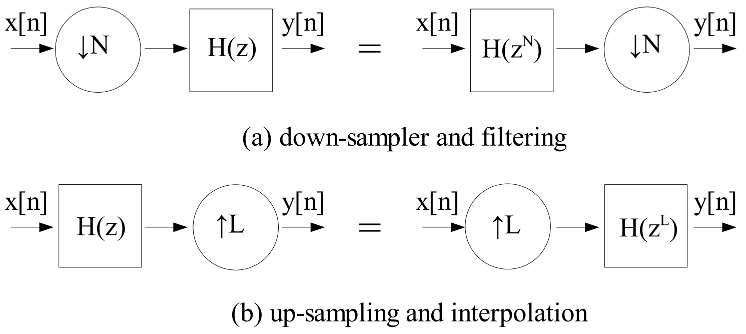

x[n]
↓N
H(z)
y[n]
=
x[n]
H(z^N)
↓N
y[n]
(a) down-sampler and filtering
x[n]
H(z)
↑L
y[n]
=
x[n]
↑L
H(z^L)
y[n]
(b) up-sampling and interpolation

Figure 11.49: Show these two equivalences

Problem 11.2 *The system shown in Fig 11.50 approximately interpolates the sequence $x[n]$ by a factor of L. Suppose that the linear filter has impulse response $h[n]$ such that $h[n] = h[-n]$ and $h[n] = 0$ for $|n| > (RL-1)$, where R and L are integers,* i.e., *the impulse response is symmetric and of length $(2RL-1)$.*

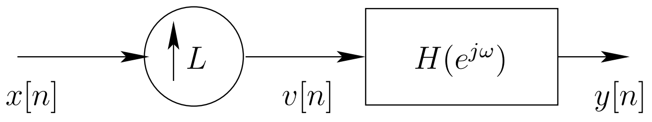


Figure 11.50: Interpolating system

(a) What condition must be satisfied by $h[n]$ in order that $y[n] = x[n/L]$ for $n = 0, \pm L, \pm 2L, \ldots$? Note that $h[n] = \delta[n]$ is a trivial example for this condition. Can you give any other example?
Hint: *Note that there is no condition on $y[n]$ where n is not divisible by L.*

(b) Let $z[n]$ be the downsampled version of $y[n]$ by sampling factor of $2L$. Find the spectrum of $z[n]$ and compare it to spectrum of $x[n]$ which is shown in Fig. 11.51. What is teh relationship between $x[n]$ and $z[n]$?

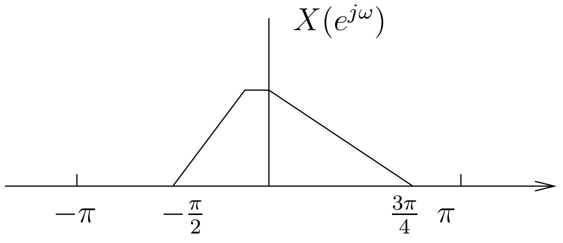


Figure 11.51: Spectrum of $x[n]$

Problem 11.3 (Downsampling in MATLAB) *In this problem we will consider the effect of aliasing in an audio file. This will be an actual experiment on using prefiltering for sampling. Download the file* hw6.wav *from the course website and load it in Matlab in the variable* `a`

```
>>[a,fs]=wavread('hw6.wav');
```

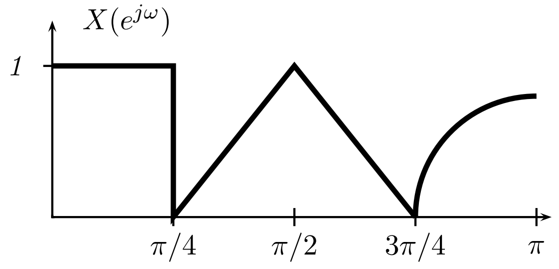
(a) *Listen to the file:*
```
>>soundsc(a);
```

(b) *Compute the DFT of* `a` *and save it in* `A`*:*
```
>>A=fft(a);
```
Read the Matlab help of `wavread`*:*
```
>>help wavread;
```
and find the sampling frequency of the file. Estimate the the bandwidth of the original continuous time file, i.e., $\omega = \frac{2\Omega}{\Omega_s}$*, where* ω *and* Ω *respectively denote the frequency in transform of the discrete-time and continuous-time signals,.*
Hint: *Recall the relationship between the DFT and DTFT and note that after the sampling process, the original spectrum will be repeated at the multiples of the sampling frequency,* $k\Omega_s$*, and then it is filtered by the appropriate low-pass filter.*

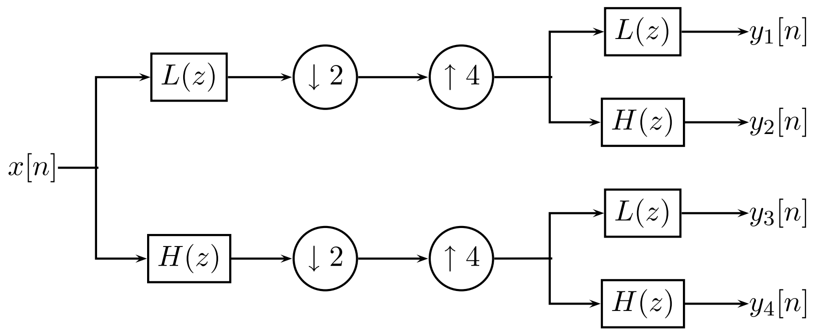
(c) *Read the Matlab help for* `downsample`*:*
```
>>help downsample;
```
Produce the sequence `b` *which is the down-sampled version of* `a` *with down-sampling factor* 3, i.e., $b[n] = a[3n]$. *Listen to* `b`*:*
```
>>soundsc(b);
```
Does it sound like the original audio file?

(d) *Plot the spectrum of* `b`*:*
```
>>B=fft(b);
>>plot(abs(B));
```

Problem 11.4 *Consider a real discrete-time signal $x[n]$ with the following spectrum:*



Note that the spectrum of a real signal is symmetric, i.e., $X(e^{j\omega}) = X(e^{-j\omega})$, *and hence we just plotted it for $\omega \geq 0$. Now consider the following multirate processing scheme in which $L(z)$ is an ideal* lowpass *filter with cutoff frequency $\pi/2$ and $H(z)$ is an ideal* highpass *filter with cutoff frequency $\pi/2$:*



Plot the four spectra $Y_1(e^{j\omega}), Y_2(e^{j\omega}), Y_3(e^{j\omega}), Y_4(e^{j\omega})$ for $\omega \geq 0$.

Problem 11.5 *Consider the system given in Figure 11.52.*

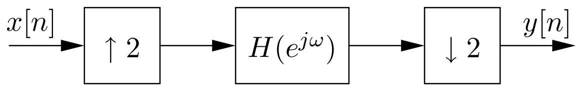


Figure 11.52: System for Problem 1.

(a) *Let $H(e^{j\omega})$ and $X(e^{j\omega})$ be as given in Figures 11.53 and 11.54 respectively. We assume that $\angle H(e^{j\omega}) = 0$. Draw $Y(e^{j\omega})$.*

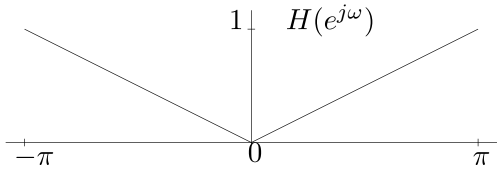


Figure 11.53: Problem 1. $H(e^{j\omega})$ to be used for Parts (a) and (c).

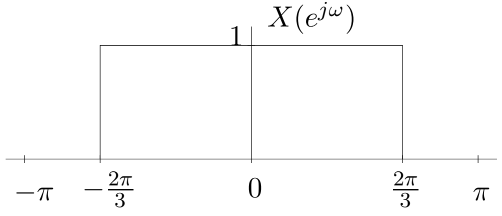


Figure 11.54: Problem 1. $X(e^{j\omega})$ to be used for Parts (a) and (c).

(b) *Consider the system given in Figure 11.55. For arbitrary $h[n]$ find an expression for*

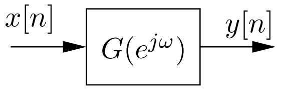


Figure 11.55: Equivalence to be shown for Problem 1.

$g[n]$ in terms of $h[n]$, such that the systems of Figures 11.52 and 11.55 are equivalent.

(c) *Using the results from Part (b), compute $Y(e^{j\omega})$ for $H(e^{j\omega})$ and $X(e^{j\omega})$ as given in Part (a).*

Problem 11.6 *Consider the system in Fig. 11.56 with $H_0(z)$, $H_1(z)$, and $H_2(z)$ as the transfer functions of LTI systems. Assume that $x[n]$ is an arbitrary stable complex signal without any symmetry properties.*

(a) *Let $H_0(z) = 1$, $H_1(z) = z^{-1}$, and $H_2(z) = z^{-2}$. Can you reconstruct $x[n]$ from $y_0[n]$, $y_1[n]$, and $y_2[n]$? If so, how? If not, justify your answer.*

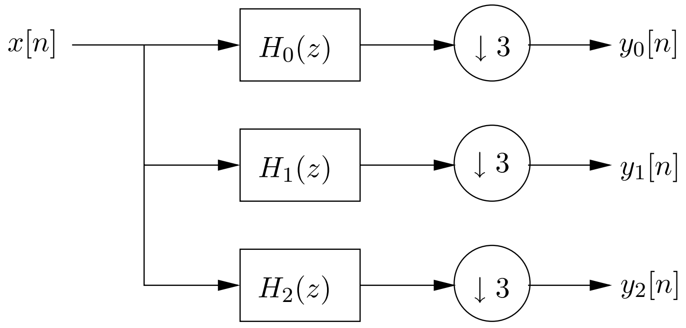


Figure 11.56: Sampling system 1.

(b) *Assume that* $H_0\left(e^{j\omega}\right)$, $H_1\left(e^{j\omega}\right)$, *and* $H_2\left(e^{j\omega}\right)$ *are as follows:*

$$H_0\left(e^{j\omega}\right)=\begin{cases}1, & |\omega|\le\frac{\pi}{3},\\ 0, & \textit{otherwise},\end{cases}$$
$$H_1\left(e^{j\omega}\right)=\begin{cases}1, & \frac{\pi}{3}|<\omega|\le\frac{2\pi}{3},\\ 0, & \textit{otherwise},\end{cases}$$
$$H_2\left(e^{j\omega}\right)=\begin{cases}1, & \frac{2\pi}{3}<|\omega|\le\pi,\\ 0, & \textit{otherwise}.\end{cases}$$

Can you reconstruct $x[n]$ *from* $y_0[n]$, $y_1[n]$, *and* $y_2[n]$*? If so, how? If not, justify your answer.*

(c) *Now consider the system in Fig. 11.57.*

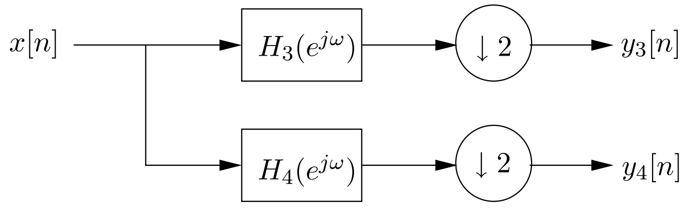


Figure 11.57: Sampling system 2.

Let $H_3\left(e^{j\omega}\right)=1$ *and*

$$H_4\left(e^{j\omega}\right)=\begin{cases}1, & 0\le\omega<\pi,\\ -1, & -\pi\le\omega<0.\end{cases}$$

Can you reconstruct $x[n]$ from $y_3[n]$ and $y_4[n]$? If so, how? If not, justify your answer.

Problem 11.7 *In your grandmother's attic you just found a treasure: a collection of super-rare 78rpm vinyl jazz records. The first thing you want to do is to transfer the recordings to compact discs, so you can listen to them without wearing out the originals. Your idea is obviously to play the record on a turntable and use an A/D converter to convert the line-out signal into a discrete-time sequence, which you can then burn onto a CD. The problem is, you only have a "modern" turntable, which plays records at 33rpm. Since you're a DSP wizard, you know you can just go ahead, play the 78rpm record at 33rpm and sample the output of the turntable at 44.1 KHz. You can then manipulate the signal in the discrete-time domain so that, when the signal is recorded on a CD and played back, it will sound right.*

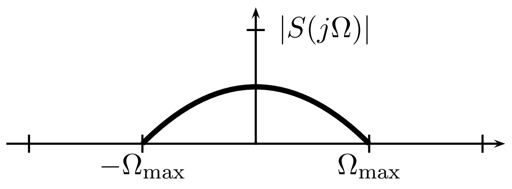
Design a system which performs the above conversion. If you need to get on the right track, consider the following:

- *Call $s(t)$ the continuous-time signal encoded on the 78rpm vinyl (the jazz music)*
- *Call $x(t)$ the continuous-time signal you obtain when you play the record at 33rpm on the modern turntable*
- *Let $x[n] = x(nT_s)$, with $T_s = 1/44100$.*

and answer the following questions:

(a) *Express $x(t)$ in terms of $s(t)$.*

(b) *Sketch the Fourier transform $X(j\Omega)$ when $S(j\Omega)$ is as in the following figure. The highest nonzero frequency of $S(j\Omega)$ is $\Omega_{\max} = (2\pi)\cdot 16000$ Hz (old records have a smaller bandwidth than modern ones).*



(c) *Design a system to convert $x[n]$ into a sequence $y[n]$ so that, when you interpolate $y[n]$ to a continuous-time signal $y(t)$ with interpolation period T_s, you obtain $Y(j\Omega) = S(j\Omega)$.*

Chapter 12

Quantization and AD/DA Conversions

In the digital world, all variables are represented with a finite precision. That is, usual, "analog" variables which take values from the real set need to be approximated, or quantized. This topic is central to signal processing, not only because analog-to-digital conversion is a key component in many signal processing and communication systems, but also because quantization is at the heart of signal compression. In this chapter we study several facets of quantization, from basic schemes and their performance to more advanced methods like the ones used in oversampled analog-to-digital conversion. We also describe uniform scalar quantization as used in simple compression systems.

12.1 Introduction

The sampling theorem described in Chapter 10 allows one to represent a bandlimited signal by means of its samples at instants nT under the condition that $T \leq T_s = \pi/\Omega_N$, Ω_N being the maximum frequency present in the signal. In order to process this infinite sequence of numbers $x[n] = x(nT), n \in \mathbb{Z}$, we need to transform the real values $x[n]$ into numbers that can be represented in a computer, that is integers or floating point numbers (which are integers with a scale factor). Thus, in both cases, we need to map the real line into a countable set, or more often, an interval of the real line onto a finite set of values. This mapping is obviously irreversible, or many-to-one, leading to approximation errors.

A simple example is the following: consider an input signal $x(t)$, bandlimited to $\Omega_N = 2\pi \cdot 1$ KHz, is known to have amplitude between -1 and 1. After sampling with sampling period $T_s = \pi/\omega_N = 0.5$ milliseconds, we need to store each sample as a 1 byte integer, i.e one of 256 possible values. The obvious solution is to divide the interval $[-1, 1]$ into

256 subintervals of size 1/128, or

$$I_k = [-1 + \frac{k}{128}, -1 + \frac{k+1}{128}], \quad k = 0 \dots 255 \tag{12.1}$$

Then, define the quantization function $Q(\cdot)$ as

$$Q(x[n]) = \{k | x[n] \in I_k\} \tag{12.2}$$

Now, if $Q(x[n]) = k$, what is the best approximation to $x[n]$, called $\hat{x}[n]$? In absence of any other information on $x[n]$ (e.g. probability density function), it is reasonable to choose the middle of the interval I_k as the approximation. Then

$$\hat{x}[n] = \{-1 + \frac{k+1/2}{128} | x[n] \in I_k\} \tag{12.3}$$

The quantization function $Q(\cdot)$ and the reconstruction points are shown in Figure 12.1 for the case of 3-bit representations, that is, 8 quantization intervals.

Now, the reconstruction of $x(t)$ from $\hat{x}[n]$ will be approximate as well. In the interpolation formula, the sample values are replaced by the approximations leading to:

$$\hat{x}(t) = \sum_{k=-\infty}^{\infty} \hat{x}[k] \operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right) \tag{12.4}$$

While simple, the example above contains all the basic question related to quantization, namely:

i) What is the set of numbers that need to be represented, e.g. all of $\mathbb{R}$ or a specific interval $[a, b]$.

ii) Into how many intervals N can we split the original interval, or stated in terms of binary representations, how many bits B can be used, where $N = 2^B$.

iii) How shall we split the original interval into the N subintervals I_k. In our example we chose a uniform splitting as in (12.1) - (12.2).

iv) Given that $x[n]$ falls into I_k, how should $\hat{x}[n]$ be chosen. In our example, we picked the midpoint as in (12.3).

v) Given the set of approximate values $\{\hat{x}[k]\}k \in \mathbb{Z}$, how can the continuous-time approximation $\hat{x}(t)$ be reconstructed, like for example (12.4).

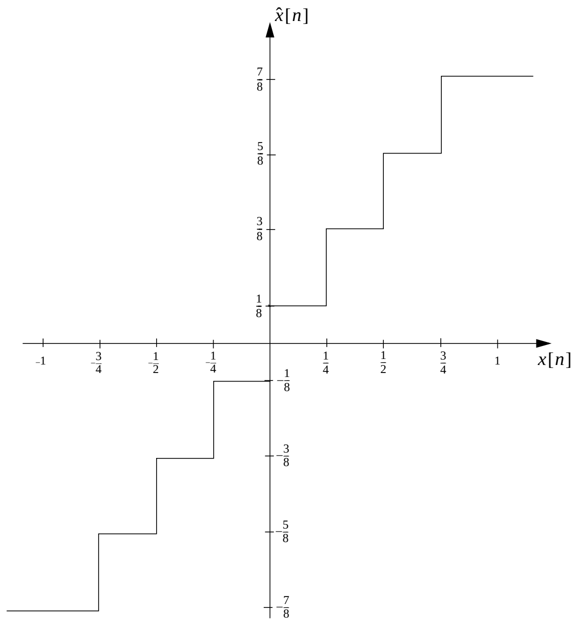


Figure 12.1: Quantization of $x[n]$ in $[-1, 1]$ to a 3-bit number, or 8 intervals. Here, uniform intervals between -1 and 1 are chosen, and the approximation $\hat{x}[n]$ is chosen as the mid-point of the interval.

It is to be noted that sometimes, the input is already a discrete-time sequence, in which case the last point is irrelevant. Also, sometimes $x[n]$ already belongs to a countable set, yet it needs to be represented more coarsely. For example, $x[n]$ may be an 8-bit integer from the set $[-128, -127, \ldots, -1, 0, 1 \ldots 127]$ which however needs to be represented by a 4-bit number. This is typical to the case of *data compression*, and will be discussed in detail in a later chapter. Thus, we will be mostly concerned with the "classical" question as outlined in the example and summarized in the five questions above.

Before moving on, it is worthwhile to point out that quantization is a non-linear operation, and therefore quite difficult to analyze in general. Therefore, simplified models are often used for the sake of making the problem tractable.

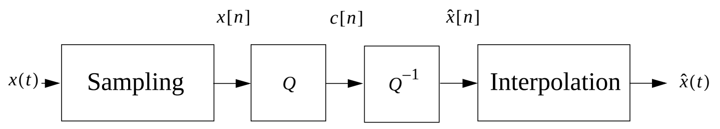


Figure 12.2: Quantization of a bandlimited signal $x(t)$. First, the signal is sampled into a sequence $x[n]$, and then the values of $x[n]$ are quantized to a countable set $c[n]$, from which an approximation sequence $\hat{x}[n]$ can be reconstructed. Interpolating this sequence leads to the continuous-time approximation $\hat{x}(t)$.

12.2 Quantization in analog-to-digital conversion

When a deterministic function or a stochastic process is bandlimited, then the sampling theorem and the associated reconstruction formula allow a perfect reconstruction in the mean-squared squared error sense. Thus, we can restrict our attention to sequences $x[n]$, either deterministic or random. In particular, any approximation $\hat{x}[n]$ of the sequence $x[n]$ which induces a quadratic error $\| x - \hat{x} \|_2$ induces *the same quadratic error* (within a scale factor) on the continuous-time reconstruction. To see this, remember that the set of functions

$$\frac{1}{\sqrt{T_s}} \operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right), \qquad n \in \mathbb{Z}$$

is an orthonormal basis for the space of Ω_N-bandlimited functions for $\Omega_N = \pi/T_s$. Thus, the statement above is just an application of Parseval's formula. Therefore, in the bandlimited case, best approximations in discrete- and continuous-time are equivalent .

We thus concentrate on quantization of discrete-time sequences. Such a system is depicted in Figure 12.2, where the sequence $c[n]$ represents a sequence of *codes* labelling the countable set of quantized values of $\hat{x}[n]$. To make matters simple, we assume an i.i.d sequence, where samples have a probability density function (pdf) $f_X(x)$. Therefore, the problem of quantizing such a signal is reduced to quantizing a random variable X with pdf $f_X(x)$, since in analog-to-digital conversion, each sample is quantized individually. This is called *scalar quantization*, as opposed to the joint quantization of several samples or *vector quantization* that will be discussed in conjunction with data compression applications.

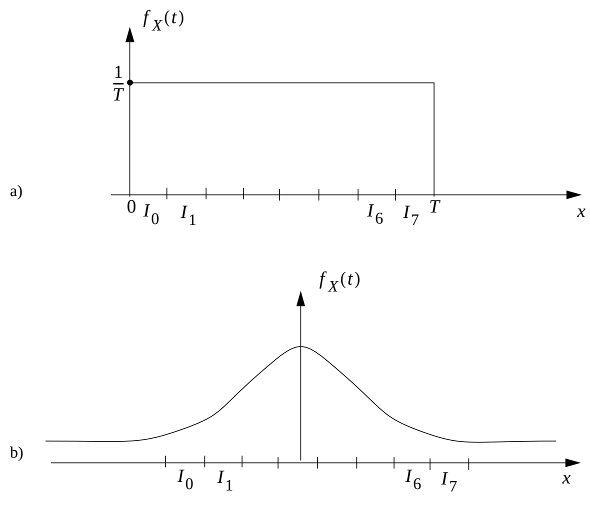


Figure 12.3: Uniform quantization into 8 cells. (a) Uniform distribution. (b) Gaussian distribution. In this case, the choice of the left most and right most boundary is critical, and here one possible but arbitrary choice is given.

12.2.1 Scalar Quantization

We now concentrate on the simplest form of scalar quantization, which is uniform. That is, a step size $\triangle$ is chosen, and the real line is subdivided into intervals I_k of width $\triangle$:

$$I_k = [k\triangle, k + \triangle) \qquad k \in \mathbb{Z} \tag{12.5}$$

In practice, only a finite number of intervals is considered, that is, a finite interval of the real line is quantized into cells of size $\triangle$. More precisely, given an interval $[a, b]$ of the real line to be quantized into N uniform cells, we use $\triangle = (b - a)/N$ and

$$I_k = [a + k\triangle, a + (k + 1)\triangle) \qquad k = 0 \dots N - 1 \tag{12.6}$$

If $f_X(x)$ is zero outside of the interval $[a, b]$, the restriction to a finite interval is of no consequence. If not, then the regions $[-\infty, a]$ and $[b, \infty]$ are mapped to the interval 0 and $N-1$ respectively, for lack of better choice. The above two cases are shown in Figure 12.3, where the quantization of uniform and a Gaussian random variable are shown in parts (a) and (b) respectively.

Uniform distribution Let us analyze the case of the uniform distribution first. If the random variable X falls into the interval I_k, the best approximation in the mean square sense is the expected value of X conditioned on the fact that $X \in I_k$. This is clearly the midpoint of the interval.

$$E(X|X \in I_k) = (k + 1/2)\triangle \tag{12.7}$$

What is the distortion, or squared norm with such a reconstruction? Take I_0 for simplicity, and calculate the variance of $X - \hat{X}$, assuming $X \in I_0$. The conditional pdf of $X/X \in I_0$ is the indicator function of the interval $[0, \triangle]$ of height $1/\triangle$, and given $\hat{X} = \triangle/2$ when $X \in I_0$ thus

$$\begin{aligned} \mathrm{E}[(X - \triangle/2)^2|X \in I_0] &= \frac{1}{\triangle}\int_0^{\triangle}(X - 1/2)^2 dx \\ &= \frac{1}{\triangle}\int_{-\triangle/2}^{\triangle/2} y^2 dy \\ &= \frac{1}{\triangle}\frac{y^3}{3}\Big/_{-\triangle/2}^{\triangle/2} \\ &= \frac{\triangle^2}{12} \end{aligned} \tag{12.8}$$

We assume a uniform pdf between 0 and $T = N \cdot \triangle$, divided into N intervals of size $\triangle$. The total expected squared distortion is

$$\begin{aligned} E_2 &= \sum_k p(x \in I_k) \cdot \frac{\triangle^2}{12} \\ &= \frac{\triangle^2}{12} \end{aligned} \tag{12.9}$$

since, given N quantization intervals of size $\triangle$ between 0 and $N \cdot \triangle, p(x \in I_k) = 1/N$. It is useful to compare this quadratic error with respect to the variance of the original distribution, which is

$$\sigma^2 = T^2/12. \tag{12.10}$$

Then,

$$E_2 = \sigma^2/N^2 \tag{12.11}$$

The result above, while simple, is fundamental. It says that the variance of the quantization error is quadratic in the quantization bin size. Now, if we quantize a uniformly distributed random variable into an R-bit representation, that is, into $N = 2^R$ quantization bins, then the quadratic error E_2 is, from (12.8):

$$E_2 = \triangle^2/12 = \frac{2^{-2R}}{12} \tag{12.12}$$

or, with respect to the variance $\sigma^2 X$, using (12.11):

$$E_2 = \sigma^2 \cdot 2^{-2R} \tag{12.13}$$

This exponential decay of the error as a function of the number of bits is central not only in analog-to-digital conversion but also, more widely, in signal compression.

Gaussian distribution Consider now the quantization of a Gaussian random variable into uniform bins of size $\triangle$.

Since the support of the probability density function is infinite, we either need an infinite number of bins, or we need to "clip" the largest values (in magnitude) to the right and left most quantization bins (see Figure 12.3). In the case of an infinite number of quantization bins, it can be proven that the squared error for a Gaussian random variable of variance σ^2 quantized into bins of size $\triangle$ is:

$$E_2 = \frac{\sqrt{3}\pi}{2} \cdot \sigma^2 \cdot \triangle^2 \tag{12.14}$$

that is, we have a similar behavior to what we have seen for uniform random variables.

In a practical system, when a finite number of bins is used, then there is an additional error due to the tails of the distribution to the left and right of the intervals I_0 and I_{N-1}, respectively. Typically, one chooses a number of intervals that covers most of the probability density. A rule of thumb is the so-called 4σ rule, that is, the quantization bins cover an interval from -2σ to 2σ for a random variable with zero mean and variance σ^2. For the Gaussian case, this rule will catch over 99.5% of the probability mass. The squared error will increase because of the clipping of values beyond the interval $[-2\sigma, 2\sigma]$, but not by much. Essentially, the behavior will still be given by an expression like (12.14), or translated into a number of bits R:

$$E_2 = C \cdot \sigma^2 \cdot 2^{-2R} \tag{12.15}$$

where C is a constant larger but of the same order as $\sqrt{3}\pi/2 = 2.72$ in (12.14).

12.2.2 Sampling and Quantization

A practical analog-to-digital converter looks like the one shown in Figure 12.4. The input signal (which is assumed to be zero mean), is first filtered in the continuous-time domain

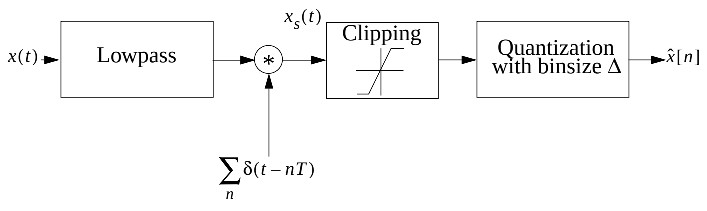


Figure 12.4: Analog-to-digital conversion. After lowpass filtering and sampling, the values are clipped to the interval $[-I/2, I/2]$, before being quantized into bins of size $\triangle = I/N$.

to ensure a bandlimited characteristic and is then sampled; the samples are then clipped to an interval $[-I/2, I/2]$. That interval is then quantized into N bins of size $\triangle = I/N$ each.

The practical circuit implementing the analog-to-digital conversion does not use sinc sampling however; instead it uses a "sample-and-hold" circuit. That is, a piecewise constant signal $x_{SH}(t)$ is generated, with the following relation to the original signal $x(t)$:

$$x_{SH}(t) = x(nT_s) \qquad nT_s \leq t < (n+1)T_s \tag{12.16}$$

During each period T_s, the value of $x_{SH}(t)$, which is now constant, can be converted into a binary string representing the best R bit approximation to $x(nT_s)$. This is shown in Figure 12.4.

Symbolically, $\hat{x}_{SH}(t)$ represents the quantized version of $x_{SH}(t)$, even so we now really have a sequence of quantized samples,

$$\hat{x}[n] = \hat{x}_{SH}(nT_s). \tag{12.17}$$

Example 12.1 *Let $x[n]$ is a bounded signal,* i.e., $|x[n]| < K, \quad \forall n$ *for some* $K \in \mathbb{R}^+$. *We pass $x[n]$ through an LTI system to obtain $y[n]$ at the output.*

(a) *Show that if $h[n]$ be an absolutely summable impulse response, then output of the system is also bounded.*

(b) *Now before feeding $x[n]$ to the system, at first we quantize it by a 4 bits quantizer to obtain $\hat{x}[n]$ and then pass it through the system.*

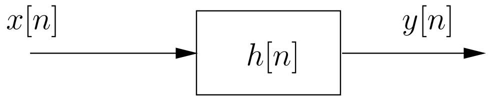


Figure 12.5: System for Example 12.1

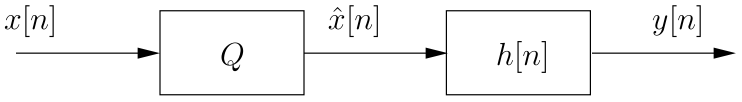


Figure 12.6: System with quantized input in Example 12.1

What is the maximum error of quantization, i.e., *the maximum difference between $x[n]$ and $\hat{x}[n]$? Compute the maximum error at output caused by quantization of the input.*

Solution:

(a) $h[n]$ is absolutely summable, i.e., *there exists some $M \in \mathbb{R}^+$ such that $\sum_n |h[n]| < M$. Therefore $y[n]$ can be bounded as*

$$\begin{aligned}
|y[n]| &= |h[n] * x[n]| = \Big|\sum_m h[m]x[n-m]\Big| \\
&\leq \sum_m |h[m]||x[n-m]| \\
&< \sum_m K \sum_m |h[n]| \\
&< KM.
\end{aligned}$$

(b) We use a 4-bit quantizer, which has $2^4 = 16$ levels. Since $x[n]$ is in the range $(-K, +k)$ of length $2K$, each level has the length $2K/16 = K/8$, and the difference between the actual value of $x[n]$ and its quantized version $\hat{x}[n]$ cannot exceed $\frac{K/8}{2}$, i.e., $|\hat{x}[n] - x[n]| < \frac{K}{16}$.

Let $\hat{y}[n]$ be the output of the filter for the input $\hat{x}[n]$.

$$\begin{aligned}
|\hat{y}[n] - y[n]| &= \Big|\sum_m h[m]\hat{x}[n-m] - \sum_m h[m]x[n-m]\Big| \\
&= \Big|\sum_m h[m](\hat{x}[n-m] - x[n-m])\Big| \\
&\leq \sum_m |h[n]||\hat{x}[n-m] - x[n-m])| \\
&\leq \frac{K}{16}\sum_m |h[n]| < \frac{KM}{16}.
\end{aligned}$$

12.3 Oversampled analog-to-digital conversion

In our previous analysis, the quantization noise was assumed to be independent of the signal [1] and white. That is, the noise spectrum was flat, and of energy $\triangle^2/12$. Pictorially, this is shown in Figure 12.8 (a).

Is there a way to improve the quantization of a signal after decreasing the quantization bin size? One method is oversampled analog-to-digital conversion. Instead of sampling at the critical Nyquist rate, one oversamples the signal by some factor K. In the digital domain, a digital lowpass filter keeps the original signal while rejecting the out-of-band quantization noise. The noise and signal spectra are shown in Figure 12.8 (b), and the overall system is depicted in Figure 12.9.

It turns out than an analysis of this system is difficult. It is intuitively clear than some of the quantization noise will be rejected by this procedure, however, assumptions like independence of the quantization noise do not hold in reality.

One result that is often quoted says that only $1/K$-th of the quantization noise energy is keep through the digital lowpass filter, so the resulting noise variance is reduced by a factor $Klat$. While the argument is intuitive looking at Figure 12.8 (b), it is only partially correct. Nonetheless, as a rule of thumb, the signal-to-noise ratio is improved by about $3dB$ per octave of oversampling. That is, each factor of 2 of oversampling contributes to reducing the noise variance by a factor of 2, or $20\log_{10}(2) \simeq 3dB$.

[1] which is clearly an approximation

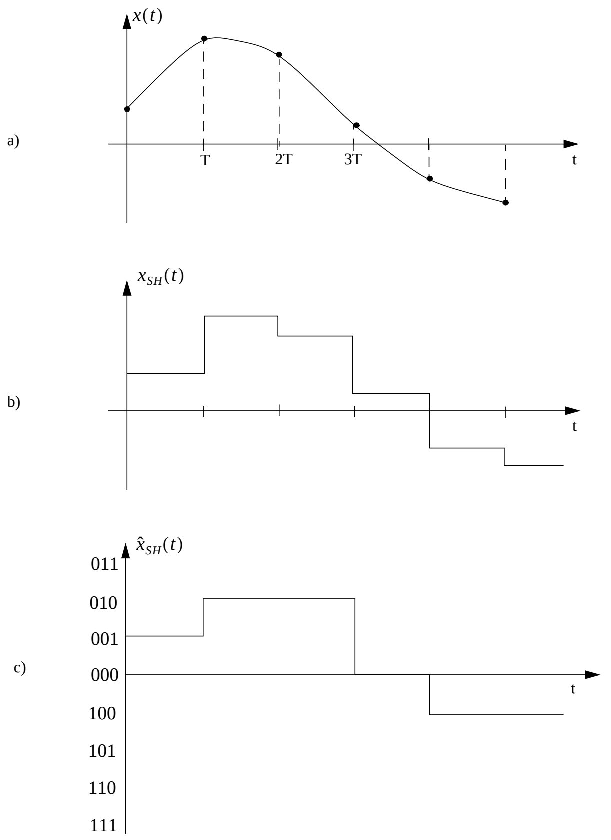


Figure 12.7: Sample-and-hold operation. (a) Original signal, sampled at integer multiples of the period T. (b) Sample-and-hold version, where the value of $x(nT)$ is held for the interval $[nT, (n+1)T]$. (c) Quantized version to 3 bits, or 8 possible values. The code given on the left is somewhat arbitrary, we used a sign + magnitude code.

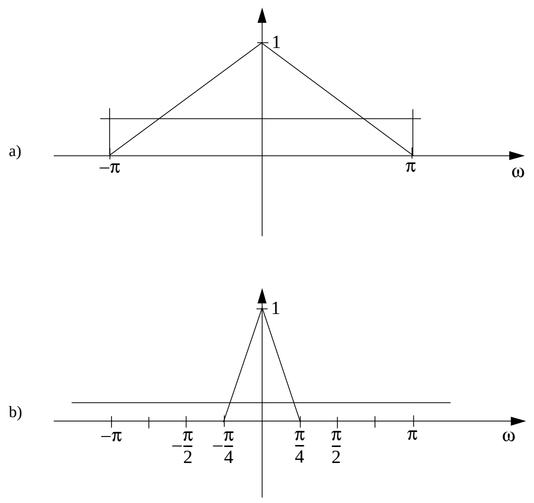


Figure 12.8: Signal spectrum and quantization noise spectrum. Because of the assumptions on the quantization noise, its spectrum is flat, and of energy $\triangle^2/12 = E_2$. (a) Critically sampled case. (b) Oversampled by a factor of 4.

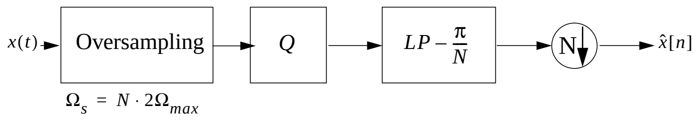


Figure 12.9: Oversampled analog-to-digital comversion. The oversampled signal is quantized first, and then lowpass filtered and subsampled by N.

12.4 Problems

Problem 12.1 *You are given the continuous time signal $s(t)$ shown in Fig. 12.10, which is 0 when $t \notin [0, 2T]$. We start quantizing at time $t = T_0$, where $0 < T_0 \leq 1$. The signal is sampled with a sampling period $T_s \geq 1$ and by definition $x[n] = s(nT_s)$. You can make the assumption that T is an integer for simplicity, and that $\frac{T-T_0}{T_s} \in \mathbb{N}$).*

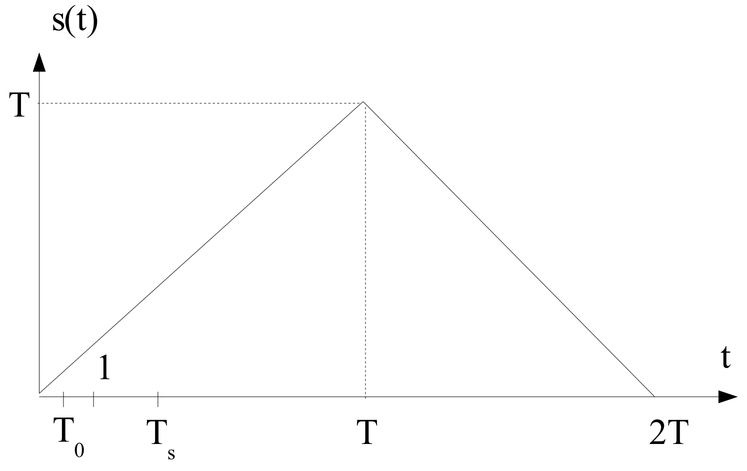


Figure 12.10: Signal $s(t)$

i) *What is the range of $s(t)$* i.e, *what is the difference between* $\max_t s(t)$ *and* $\min_t s(t)$*?*

ii) *How many quantization levels do you need, as a function of the parameters above, if the maximum error you allow is $e_q[n] = 0.5$?*

iii) *How many bits do you need to store the quantized signal between T_0 and $2T$?*

iv) *A 1-bit differential quantizer (also called delta modulator) is shown in Fig. 12.11. How many bits does the CODER produce between time $t = T_0$ and time $t = 2T$, when we feed the signal $s(t)$ to it (you can assume that $y[1] = 0$)? What is the maximum error when we decode the signal* i.e., *what is $e_{max} = max_n |x[n] - x_q[n]|$ at the output of the DECODER? Make a plot (by hand) of the output of the DECODER.*

v) *What should the sampling frequency T_s and T_0 be if the maximum tolerable error e_{max} is 0.5? For this sampling frequency T_s and T_0, how does the number of bits you*

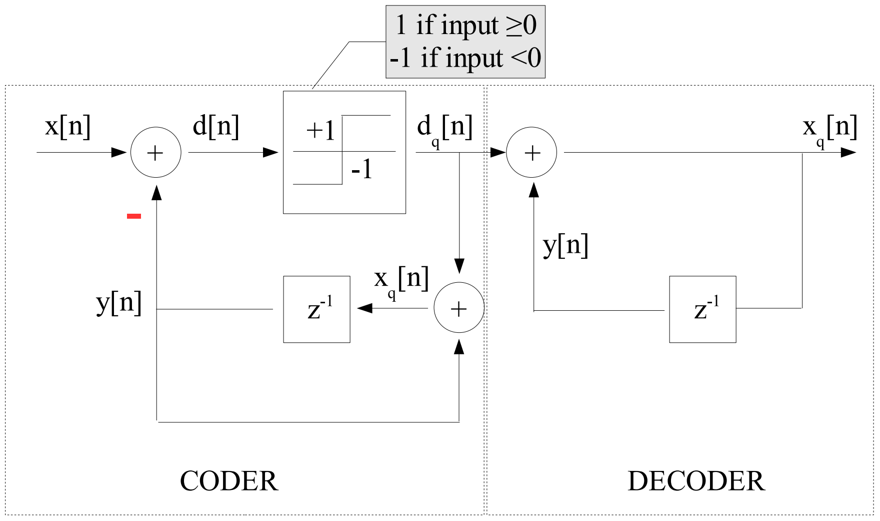


Figure 12.11: Delta-modulator

need with the delta modulator compare to the number of bits you need for classical quantization (point (c))?

Chapter 13

Additional Material

13.1 Example of Frequency Domain Manipulations: Denoising

This short example shows how one can easily identify noise components in the frequency domain while a similar task is very difficult in the time domain. This is not a homework task, so the only purpose is to help you understand what is happening. Proceed as follows (**if you copy-paste, you might have to change the quote signs ' in matlab**):

i) Download handeln.wav from the course website (under additional material).

ii) Open Matlab, move to the adequate directory and read in the handeln.wav file. Subsequently, play the sound:

```
[datan,fs]=wavread('handeln.wav');
soundsc(datan,fs);
```

iii) Plot the 500 first sample of `datan`

```
plot(datan(1:500))
```

It seems very difficult to distinguish the music and the annoying noise!

iv) Fortunately, all hope is not lost. Let's have a look at what happens in the frequency domain by calculating the DFT (fft is an algorithm allowing you to compute the DFT in a very efficient way, you will learn more on that in upcoming lectures...) of `datan`:

```
XN=fft(datan);
plot((1:length(XN))/length(XN),abs(XN(1:end)))
xlabel('normalized f')
ylabel('|XN(f)|')
```

The plot you just created is a function of $f = \frac{F}{F_s} = \frac{\omega}{2\pi}$. The lowest possible sampling frequency F_s to avoid aliasing is $2max(F)$. Hence, high frequencies are represented close to 0.5. Also note that the spectrum is 1-periodic, so values $0.5 + \epsilon$ correspond to values $-0.5 + \epsilon$ (so also high frequencies)

v) We can see that there are three high frequency components which look different from the rest of the signal, mainly concentrated in the low frequencies. Let us simply set values close to 0.5 to zero (remove high frequencies) and see what happens in the frequency domain first.

```
xm=(1+73113)/2;
inter=[xm-7000:xm+7000];
```

```
X=XN;
X(inter)=zeros(1,length(inter));
plot((1:length(X))/length(X),abs(X(1:end)))
```

Obviously we have removed the suspicious high frequency components, but unfortunately also a part of the signal, even though there was little energy in the high frequencies.

vi) Let's listen to what we obtained in the time domain

```
data=ifft(X);
soundsc(abs(data),fs)
```

13.2 Small Project

Uncompressed digital CD-quality audio signals consume a large amount of data and are therefore not suited for storage and transmission. The need to reduce this amount without any noticeable quality loss was stated in the late 80ies by the International Organization for Standardization (ISO). A working group within the ISO referred to as the Moving Pictures Experts Group (MPEG), developed a standard that contained several techniques for both audio and video compression. The audio part of the standard included three modes with increasing complexity and performance. The third mode, called Layer III, manages to compress CD music from 1.4 Mbit/s to 128 kbit/s with almost no audible degradation. This technique, also known as MP3, has become very popular and is widely used in applications today (see [2] for an overview of the technology). Since the MPEG-1 Layer III is a complex audio compression method it may be quite complicated to get hold of all different components and to get a full overview of the technique.

In Fig. 13.1 we show a block diagram of an MP3 encoder. While the complete encoding process is very complex and far beyond the scope of this course, we will try in this homework to understand some specific parts. In particular, we will investigate the subband decomposition (A on figure), the cosine transform (B on figure) and try to get an idea of what a Psycho-accoustic model is (C onfigure).

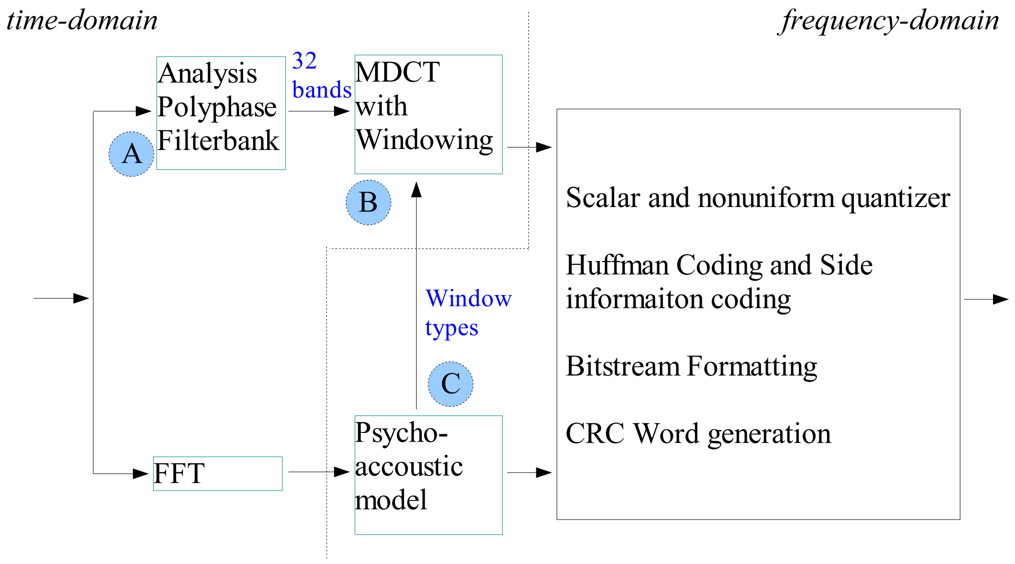


Figure 13.1: Block diagram of an MPEG-Layer III (MP3) encoder

Problem 13.1 ***Filterbank*** *In the polyphase filterbank, a sequence of 1152 PCM samples (Pulse Code Modulation, simply think of it as a sequence of 1152 discrete time symbols) are filtered into 32 equally spaced frequency subbands depending on the Nyquist frequency of the PCM signal. Assume that the sample frequency of the PCM signal is 44.1 kHz.*

i) *What was the Nyquist frequency F_N of the PCM signal, assuming that we sample the signal at the lowest possible frequency?*

ii) *How wide will every subband be (in Hz)?*

iii) *Every sample might contain signal components from 0 to the Nyquist frequency that will be filtered in an appropriate subband. How many samples do you need to store directly after the subband decomposition?*

iv) *In practice, the next step is to downsample the output of all 32 passband filters by a factor 32. Assuming that the subband decomposition can be done using perfect passband filters, we will show that the downsampling operation can be performed without introducing any aliasing.*

In particular, we will show that you can downsample by a factor M a real bandpass signal $S_c(e^{j\omega})$, where $S_c(j\Omega) = 0$ for $0 \leq |\Omega| \leq \Omega_c$ and for $|\Omega| \geq \Omega_c + \Delta\Omega$, even if the sampling frequency is below $2M \times (\Omega_c + \Delta\Omega)$. The trick is of course that the signal is only non-zero for a very small portion of the spectrum. Make the following assumptions: (i) The signal was sampled at the Nyquist rate i.e $\frac{2\pi}{T} = 2(\Omega_c + \Delta\Omega)$ *and (ii) M is the largest Integer less than or equal to $2\pi/\Delta\omega$. Show that the spectrum of $S_d(e^{j\omega})$ in Fig. 13.2 is not aliased. A Hilbert transformer has the following frequency response:*

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

(Hint: draw the spectrum of an arbitrary bandpass signal and try to understand what happens)

v) *How many samples do you need to store after the downsampling operation?*

vi) *In practice, will you get aliasing? Which operation can introduce aliasing?*

By applying a modified discrete cosine transform to each time frame of subband samples the 32 subbands will be split into 18 finer subbands creating a granule with a total of 576 frequency lines. But prior to the MDCT each subband signal has to be windowed.

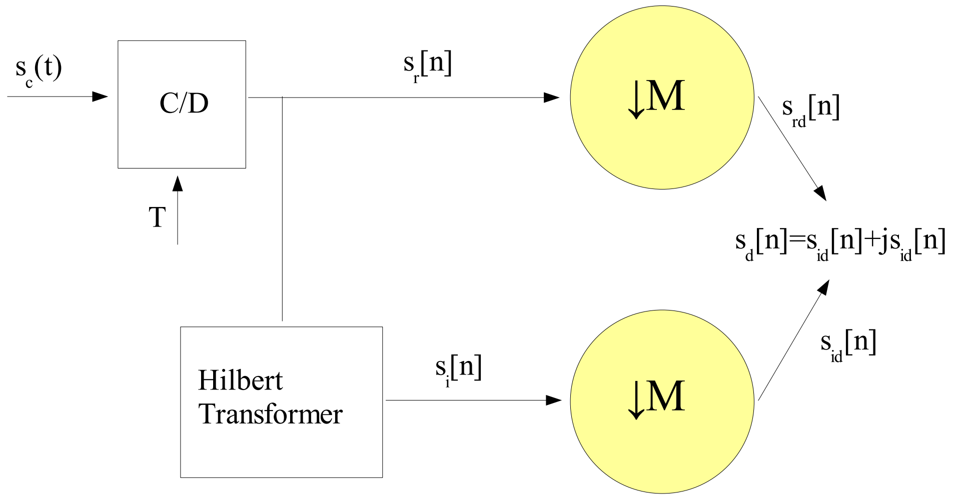


Figure 13.2: Circuit for downsampling of bandpass signals

Windowing is done to reduce artefacts caused by the edges of the time-limited signal segment. There are four different window types defined in the MPEG standard. Depending on the degree of stationarity the psychoacoustic model determines which window type to apply and forwards the information to this block. If the psychoacoustic model decides that the subband signal at the present time frame shows little difference from the previous time frame, then the long window type is applied, which will enhance the spectral resolution given by the MDCT. Alternatively, if the subband signal shows considerable difference from the previous time frame, then the short windows is applied. This type of window consists of three short overlapped windows and will improve the time resolution given by the MDCT. A higher time resolution is necessary in order to control time artifacts, for instance pre-echoes. In order to obtain a better adaptation when windows transitions are required, two windows referred to as start windows and stop windows, are defined. A long window becomes a start window if it is immediately followed by a short window. Similarly, a long window becomes a stop window if it is immediately preceded by a short window. The start and stop windows are skewed to account for the steeper sides of the adjacent short window.

The DFT is perhaps the most common example of finite length transform representation of the form

$$A[k] = \sum_{n=0}^{N-1} x[n]\phi_k^*[n] \tag{13.1}$$

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} A[k]\phi_k[n] \tag{13.2}$$

In the case of the DFT, the sequence $A[k]$ is in general complex when the sequence $x[n]$ is real. It is natural to ask whether there exists transforms which yield a real sequence $A[k]$ when $x[n]$ is real. This has led to the definition of a series of other orthogonal transform representations such as the Haar transform and the Discrete Cosine Transform (DCT), commonly used in speech processing, and in particular in MP3 encoders

Problem 13.2 ***Cosine Transform*** *There are different types of cosine transforms. In this homework, we will study DCT of type II defined as follows:*

$$X^c[k] = 2\sum_{n=0}^{N-1} x[n]\cos(\frac{\pi k(2n+1)}{2N}) \quad ,0 \leq k \leq N-1 \tag{13.3}$$

$$x[n] = \frac{1}{N}\sum_{n=0}^{N-1} \beta[k]X^c[k]\cos(\frac{\pi k(2n+1)}{2N}) \quad ,0 \leq n \leq N-1 \tag{13.4}$$

$$\beta[k] = \begin{cases} \frac{1}{2} & ,\textit{when } k=0 \\ 1 & ,\textit{when } 1 \leq k \leq N-1 \end{cases} \tag{13.5}$$

Note that some definitions also include a normalization factor which we omit for simplicity reasons. We will now explore some of the properties of the DCT and try and understand why it is commonly used instead of the DFT

i) Show that you can write the DCT in the form X^c=Wx by specifying the $N \times N$ matrix W.

ii) Show that the columns of W^T form an orthogonal basis.

iii) Define:

$$X_2[k] = X[k] + X^*[k]e^{j2\pi k/(2N)}$$

*where $X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi nk/(2N)}$ (*i.e., *$X[k]$ is the 2N-point DFT of the N point sequence $x[n]$). Show that*

$$X^c[k] = e^{-j\pi k/(2N)}X_2[k]$$

iv) The DCT is used in many data compression applications in preference to the DFT because of a property known as "energy compaction". More precisely, this means that the DCT has its coefficients more concentrated at low frequencies than the DFT does. Consequently, by only storing the first components of the DCT, we lose less information than if we stored the same number of components of the DFT. Write a small program that computes the 32-points DCT of the sequence $x[n] = a^n\cos(0.1\pi n)$

and compare it to the 16 point DFT (remember that for the DFT we must in practice store the 16 complex and the 16 real coefficients). Is the energy more concentrated in the first coefficients of the DCT?

Psychoacoustics is the research field where you aim to understand how the ear and brain interact as various sounds enter the ear. Humans are constantly exposed to an extreme quantity of radiation. These waves are within a frequency spectrum consisting of zillions of different frequencies, but only a small fraction of all waves are perceptible by our sense organs; the light we see and the sound we hear. Infrared and ultraviolet light are examples of light waves we cannot percept. Regarding our hearing, most humans can not sense frequencies below 20 Hz nor above 20 kHz. Additionally, this bandwidth tends to narrow as we get older. A middle aged man, for instance, will usually not hear much above 16 kHz. Frequencies ranging from 2 kHz to 4 kHz are easiest to perceive, they are detectable at a relatively low volume. As the frequencies changes towards the ends of the audible bandwidth, the volume must also be increased for us to detect them (see Fig. 13.3). That is why we usually set the equalizer on our stereo in a certain symmetric way. As we are more sensitive to midrange frequencies these are reduced whereas the high and low frequencies are increased. This makes the music more comfortable to listen to since we become equal sensitive to all frequencies.As our brain cannot process all the data available to our five senses at a given time, it can be considered as a mental filter of the data reaching us. A perceptual audio codec is a codec that takes advantage of this human characteristic. While playing a CD it is impossible to percept all data reaching your ears, so there is no point in storing the part of the music that will be inaudible. The process that makes certain samples inaudible is called masking. There are two masking effects that the perceptual codec need to be aware of; simultaneous masking and temporal masking. Experiments have shown that the human ear has 24 frequency bands. Frequencies in these so called critical bands are harder to distinguish by the human ear. Suppose there is a dominant tonal component present in an audio signal. The dominant noise will introduce a masking threshold that will mask out frequencies in the same critical band (see Fig. 13.3). This frequency-domain phenomenon is known as simultaneous masking, which has been observed within critical bands.

MP3 encoding takes advantage of this masking phenomenon by not encoding masked tones. In practice, this is done by allocating more bits to the subbands which are not masked and fewer or no bits to the masked subbands. We refer you to [1] for a deeper explanation of psychoacoustic models. Note that this is a lossy coding procedure in the sense that we cannot reconstruct the original signal exactly after we have allocated bits this way.

Problem 13.3 *i) Based on what you have read in this document, the problems you solved and (maybe) the documents referenced hereunder,* ***briefly*** *explain the overall*

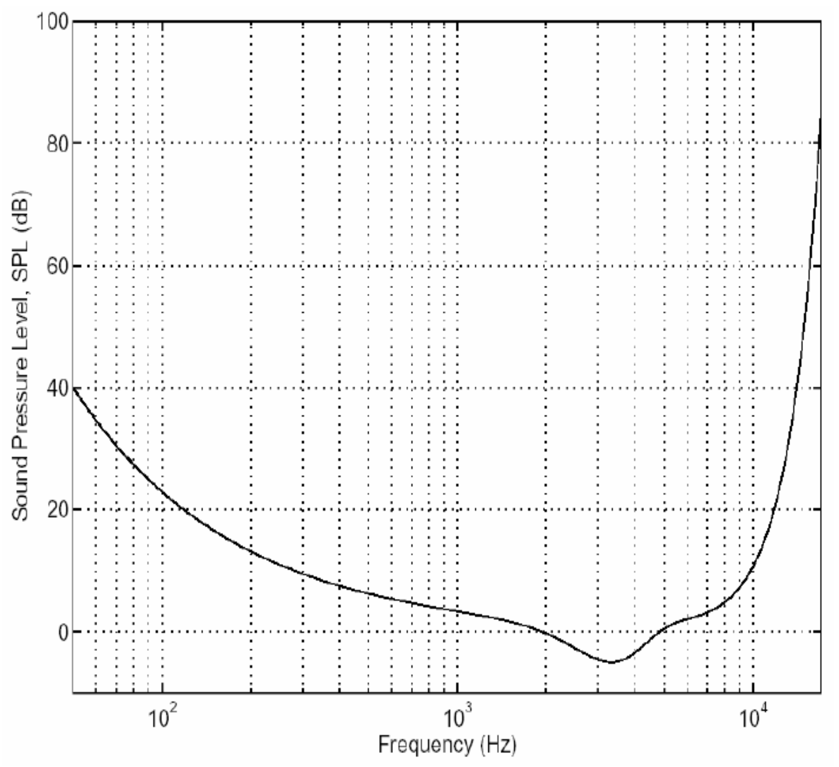


Figure 13.3: The absolute threshold of hearing (from [1])

MP3 encoding process (the part we studied). In particular explain conceptually what phenomenon MP3 exploits to achieve a strong coding gain and how. Explain how it is possible that even though the coding procedure is lossy, the quality of the sound we hear hardly degrades.

[1] Ted Painter, Andreas Spanias, “A Review of Algorithms for Perceptual Coding of Digital Audio Signals”, 1997

[2] Rassol Raissi, “The Theory Behind Mp3”, 2002

13.3 Design Example: OFDM

13.3.1 Introduction

In this lecture we will have a look at some of the principles used in modem design for Digital Subscriber Line (DSL) communication. Asymetric Digital Subscriber Line (ADSL), which is a specific instance of DSL technology, is currently one of the main techniques used to provide broadband internet access.

DSL operates over twisted pair copper telephone lines. Telephone lines have originally been used for transmission of voice only, using the spectrum up to 3.4kHz. DSL shares the telephone line with regular voice transmission, but operates at frequencies roughly from 10kHz up until 1.1MHz.

Transmitting over a twisted pair copper line requires digital-to-analog conversion at the transmitter and analog-to-digital conversion an the receiver. We will consider only the discrete-time domain and treat A/D and D/A conversion as a part of our communication channel.

If one transmits over a twisted pair copper telephone line at high frequencies, the signal will get corrupted. Since we transmit over a fixed connection, we can assume that the system is time-invariant. It turns out that we can also use the assumption the the system is linear. Our time-discrete channel is therefore a linear time-invariant (LTI) system. We will interchangeably use the terms system and channel.

One of the techniques that are used to transmit over LTI channels is Orthogonal Frequency Division Multiplexing (OFDM). This technique is also known as Discrete Multi-Tone (DMT). In this lecture we will look at the principles behind OFDM. We will also give a complete MATLAB implementation of these ideas.

13.3.2 Problem

We will design a DSL communication system. As seen in the introduction we can model the communication channel as a Linear Time-Invariant system, see Figure 13.4.

Let $h[n]$ be the impulse response of the LTI channel. We assume that it is of finite-

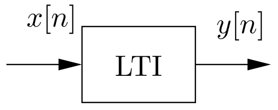


Figure 13.4: DSL communication channel modelled as a linear time-invariant system.

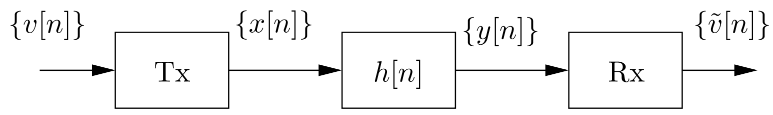


Figure 13.5: Tranmission of a block of symbols $\{v[n]\}$. The transmitter (Tx) produces N symbols $\{x[n]\}$ that are sent through the channel. At the output of the channel, the receiver (Rx) takes $\{y[n]\}$ and produces $\{\tilde{v}[n]\}$.

length, i.e. there is a ν such that

$$h[n] = 0, \quad \text{for } n < 0, \text{and } n > \nu - 1.$$

We also assume that the receiver knows $h[n]$. In the example and MATLAB implementation we consider in this handout, we have a channel for which $\nu = 128$.

The specific problem that we will consider is that of transmission of a sound signal using a DSL system. We denote the sequence that is to be transmitted by $\ldots, v[-2], v[-1], v[0], v[1], v[2], \ldots$.

We will split up this sequence in blocks of N symbols. For each block of symbols, say the symbols $v[0], \ldots, v[N-1]$, a transmitter produces N symbols $x[0], \ldots, x[N-1]$ that are sent through the channel. At the output of the channel, a receiver takes $y[0], \ldots, y[N-1]$ and produces $\tilde{v}[0], \ldots, \tilde{v}[N-1]$, see Figure 13.5. We use the notation $\{v[n]\}$ to denote the block $v[0], \ldots, v[N-1]$.

We restrict ourselves to transmitter and receiver structures that produce one input symbol to the channel for each symbol of the sound sequence. One can think of this as streaming of audio over the LTI channel. Implicitely we consider the audio samples arriving at the tranmitter at a certain rate (the number of symbols per second). If we want to have any hope of successfully streaming the sequence over the channel, we can not send more than one channel input for each audio symbol.

A trivial transmitter and receiver would directly sent each incoming symbol over the channel and directly use it, i.e. $\{x[n]\} = \{v[n]\}$ and $\{\tilde{v}[n]\} = \{y[n]\}$. The MATLAB implementations for these trivial transmitter and receiver are given in Figure 13.8. The code to test this system is given in Figure 13.6.

If we run this code and listen to the received sound sequence $\tilde{v}[n]$, we notice that it is corrupted. This should not be surprising, since we know that if $y[n] = h[n] * x[n]$, then

$$Y\left(e^{j\omega}\right) = H\left(e^{j\omega}\right) X\left(e^{j\omega}\right).$$

This means that the different frequency components in the signal $x[n]$, get attenuated differently.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Loading audio
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Load the sound sequence that we will
% use for testing.
[v,fs] = wavread('handel.wav');

% Let's listen to it...
soundsc(v,fs);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Divide the sequence in blocks
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% We choose to use a blocklength
% of N = 1024.
N = 1024;

% Add zeros to the sound sequence such that the
% total length is a multiple of N.
v = [v; zeros(1024-mod(length(v),N),1)];

% Divide the sequence in blocks.
% Each column of v_block is a block of N symbols
v_block = reshape(v,N,length(v)/N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% The channel
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
impulse_response = [ <<removed for handout>> ];
```

Figure 13.6: MATLAB implementation: Initialization. Loading audio and dividing it into blocks. The channel impulse response is known at the receiver, so we give it here.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Transmit the signal over the channel. We use a
% trivial transmitter and receiver.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Initialization
x = zeros(size(v_block,1),1);
y = zeros(size(v_block,1),1);
v_tilde_block = zeros(size(v_block));
% Now do all operations per block
for i=1:size(v_block,2)
x = trivial_transmitter(v_block(:,i));
y = lti_channel(x);
v_tilde_block(:,i) = trivial_receiver(y);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Check the result.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Put all blocks back together in one sequence.
v_tilde = reshape(v_tilde_block,size(v));

% ... and listen to what we received.
soundsc(v_tilde,fs);
```

Figure 13.7: MATLAB implementation: Transmission using the trivial transmitter and receiver from Figure 13.8.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Trivial transmitter.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function x = trivial_transmitter(v)
x = v;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Trivial receiver.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function v_tilde = trivial_receiver(y)
v_tilde = y;
```

Figure 13.8: MATLAB implementation: A trivial transmitter and receiver.

In the next sections we will develop a different transmitter/receiver structure.

13.3.3 Basic Idea

The basic idea of our new transmitter and receiver is to use the fact that complex exponentials are eigenfunctions of LTI systems. This follows immediately from the following DTFT relation

$$x[n] * h[n] \Leftrightarrow X\left(e^{j\omega}\right) H\left(e^{j\omega}\right). \tag{13.6}$$

Suppose we want to transmit a single value v. If we let $x[n] = \mathrm{DTFT}^{-1}\left\{v\delta(n-\omega_0)\right\}$, with ω_0 arbitrarily chosen, we have

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

and

$$Y\left(e^{j\omega}\right) = H\left(e^{j\omega_0}\right) v. \tag{13.7}$$

From (13.7) we see that we can recover v as

$$\tilde{v} = H\left(e^{j\omega_0}\right)^{-1} \mathrm{DTFT}\left\{y[n]\right\}.$$

Building on this idea, if we want to transmit $v[0], \ldots, v[N-1]$ we can select some frequencies $\omega_0, \ldots, \omega_{N-1}$ and let

$$x[n] = \mathrm{DTFT}^{-1}\left\{\sum_{l=0}^{N-1} v[l]\delta(n-\omega_l)\right\}. \tag{13.8}$$

After receiving $y[n]$ we evaluate the $Y(e^{j\omega})$ at the points $\omega_0, \ldots, \omega_{N-1}$ to get

$$Y\left(e^{j\omega_l}\right) = H\left(e^{j\omega_l}\right) v[l], \qquad l = 0, \ldots, N-1.$$

We see that we can find $\tilde{v}[0], \ldots, \tilde{v}[N-1]$ as

$$\tilde{v}[l] = H\left(e^{j\omega_l}\right)^{-1} \mathrm{DTFT}\left\{y[n]\right\}, \qquad l = 0, \ldots, N-1.$$

There is a fundamental problem with the idea presented above. The sequence $x[n]$ in (13.8) is of infinite length and can never be transmitted over the LTI channel. We see that the above is a nice mathematical construction, but it can not be implemented. In the next section, however, we will see that we can obtain similar results if we start from the fact that $x[n]$ is a finite-length sequence.

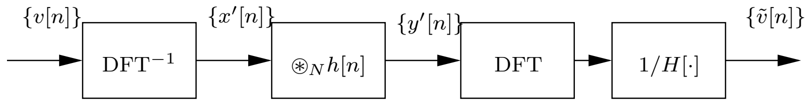


Figure 13.9: Basic idea of the communication system, based on DFT relation (13.9).

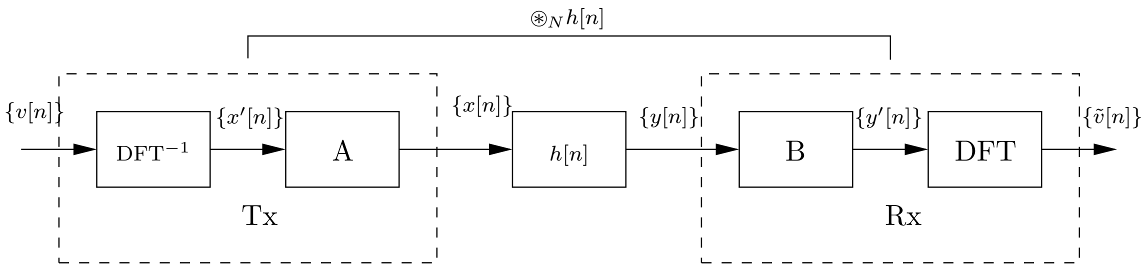


Figure 13.10: Expansion of Figure 13.9. The transmitter and receiver perform additional processing A and B in order to get an overall behavior of circular convolution.

13.3.4 Finite Length Inputs

The DTFT equivalent for finite-length sequences is the DFT. The DFT equivalent to relation (13.6) for N-length sequences is

$$x'[n] \circledast_N h[n] \Leftrightarrow X'[k]H[k], \tag{13.9}$$

where $\circledast_N$ is the circular convolution. The idea is to exploit relation (13.9) in a way as depicted in Figure 13.9. We see that if the LTI channel would perform a circular convolution, we would be done. The channel, however, performs a linear convolution, i.e. $y[n] = x[h] * h[n]$. Therefore, we need to perform a trick to get an overall behavior of a circular convolution. See Figure 13.10, where the circular convolution comes from the LTI channel together with some additional processing in the transmitter (A) and in the receiver (B). We will now analyze how to implement the systems A and B in order to get an overall circular convolution.

This is where we need the assumption that $h[n]$ has finite length. Remember from Section 13.3.2 that

$$h[n] = 0, \quad \text{for } n < 0, \text{and } n > \nu.$$

We have

$$y'[l] = x'[l] \circledast_N h'[l] = \sum_{k=0}^{\nu} h[k]x'[(l-k)_N], \tag{13.10}$$

$l = 0, \ldots, N-1$, where

$$(l-k)_N = \begin{cases} l-k, & \text{if } l-k \geq 0 \\ N+l-k, & \text{if } l-k < 0. \end{cases}$$

Now, the trick is to add a *cyclic prefix*. If $x'[0] \ldots x'[N-1]$ is the output of the inverse DFT in Figure 13.10, we send ν additional symbols over the channel. To simplify notation we start indexing $\{x[n]\}$ at $-\nu$, we have $\{x[n]\} = x[-\nu] \ldots x[N-1]$. If we now let

$$x[l] = \begin{cases} x'[l], & \text{if } 0 \leq l \leq N-1 \\ x'[N+l], & \text{if } -\nu \leq l < 0, \end{cases}$$

we see that at the output of the channel we have, for $l = 0, \ldots, N-1$.

$$\begin{aligned} y[l] &= h[l] * x[l] \\ &= \sum_{k=0}^{\nu} h[k]x[(l-k)] \\ &= \sum_{k=0}^{\nu} h[k]x'[[(l-k)_N] \\ &= x'[l] \circledast_N h'[l] \end{aligned}$$

The overall behavior is that of a circular convolution. Note that the receiver will only have to keep $y[0], \ldots, y[N-1]$ and process these symbols to retrieve $\{v[n]\}$.

Figure 13.11 shows how the cyclic prefix is added.

There is one problem left to solve. In Section 13.3.2 we specified that the transmitter needs to create blocks of length N. In our current solution it creates an output of length $N + \nu$.

13.3.5 Downsampling

To make sure that our transmitter outputs a block of N samples we will downsample the sequence $\{v[n]\}$ before applying the inverse DFT.

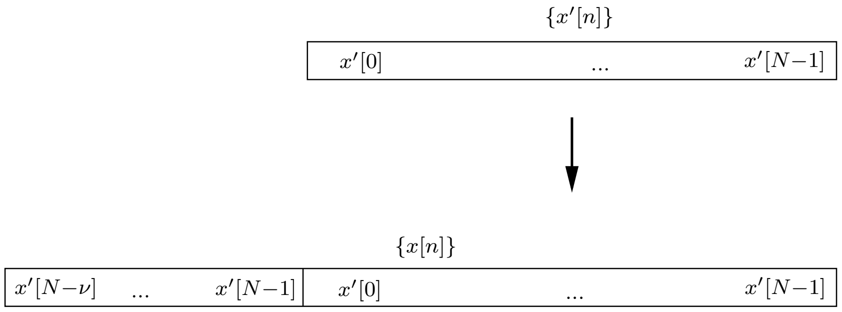


Figure 13.11: The cyclic prefix.

We need to determine the downsampling factor. Let N' be the length of the downsampled block. After applying the cyclic prefix of length ν we need a length N, i.e. we need $N' + \nu = N$. Therefore we need to resample with a factor

$$\left(1 - \frac{\nu}{N}\right).$$

The receiver, after removing the cyclic prefix, performing the DFT and scaling according to $1/H[\cdot]$, resamples with a factor $1/(1 - \nu/N)$.

In the example implementation we consider here, we have $\nu = 128$ and $N = 1024$. The transmitter downsamples with a factor 8/7 and the receiver upsamples with that same factor.

Figures 13.12 and 13.13 give the complete MATLAB implementation of the transmitter and the receiver. The code from Figure 13.6 can be easily adjusted to use the new transmitter and receiver. If we know listen to the received sound sequence we notice that there is no audible difference with the original.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% OFDM transmitter.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function x = ofdm_transmitter(v,nu)

% The receiver needs nu as an additional argument
% to know how long the cyclic prefix needs to be

% Resampling
v_down = resample(v,7,8);
% (We could compute the factors 7/8 from the function arguments
%  but we cheat a bit here...)

% Inverse DFT
x_prime = ifft(v_down);

% Apply the cyclic prefix
x = [x_prime(end-nu+1:end); x_prime];
```

Figure 13.12: MATLAB implementation: OFDM transmitter.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% OFDM receiver.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function v_tilde = ofdm_receiver(y,nu,impulse_response)

% The OFDM receiver has 2 additional arguments, the length
% of the cyclic prefix and the impulse response of the
% channel.

% Remove the cyclic prefix
y_cut = y(nu+1:end);

% DFT
Y = fft(y_cut);

% Scale
H = fft(impulse_response,length(Y));
v_tilde_down = Y./H.';
% ...we are still working in the "downsampled regime"

% We know that v_tilde_down is real, but MATLAB keeps it as a
% complex variable.
v_tilde_down = real(v_tilde_down);

v_tilde = resample(v_tilde_down,8,7);
v_tilde = v_tilde(1:1024);
% (Again, we could have computed the upsampling factor and the
% block length from the function arguments, but we cheat a bit
% to keep the code easy.)
```

Figure 13.13: MATLAB implementation: OFDM receiver.