Intro to Digital Signal Processing

Collection Editor:

Robert Nowak

Intro to Digital Signal Processing

Collection Editor:

Robert Nowak

Authors:

Behnaam Aazhang
Swaroop Appadwedula
Richard Baraniuk
Matthew Berry
Dan Calderon
Hyeokho Choi
Benjamin Fite
Roy Ha
Michael Haag
Mark Haun
Don Johnson
Douglas L. Jones
Nick Kingsbury
Dima Moussa
Robert Nowak
Justin Romberg
Daniel Sachs
Phil Schniter
Ivan Selesnick
Melissa Selik
Bill Wilson

Online:
< http://cnx.org/content/col10203/1.4/ >

C O N N E X I O N S

Rice University, Houston, Texas

Table of Contents

Chapter 1

DSP Systems I

1.1 Image Restoration Basics[1]

1.1.1 Image Restoration

In many applications (*e.g.*, satellite imaging, medical imaging, astronomical imaging, poor-quality family portraits) the imaging system introduces a slight distortion. Often images are slightly blurred and image restoration aims at **deblurring** the image.

The blurring can usually be modeled as an LSI system with a given PSF $h[m,n]$.

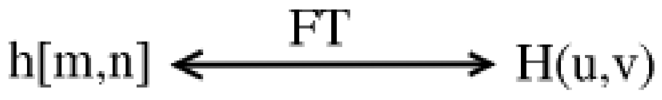


Figure 1.1: Fourier Transform (FT) relationship between the two functions.

The observed image is

$$g[m,n] = h[m,n] * f[m,n] \tag{1.1}$$

$$G(u,v) = H(u,v)\,F(u,v) \tag{1.2}$$

$$F(u,v) = \frac{G(u,v)}{H(u,v)} \tag{1.3}$$

Example 1.1: Image Blurring
Above we showed the equations for representing the common model for blurring an image. In Figure 1.2 we have an original image and a PSF function that we wish to apply to the image in order to model a basic blurred image.

[1]This content is available online at <http://cnx.org/content/m10972/2.2/>.

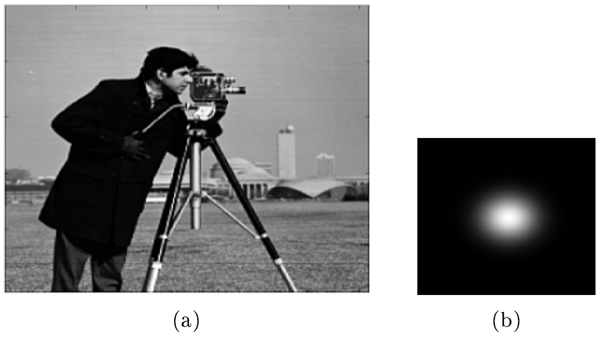


Figure 1.2

Once we apply the PSF to the original image, we receive our blurred image that is shown in Figure 1.3:

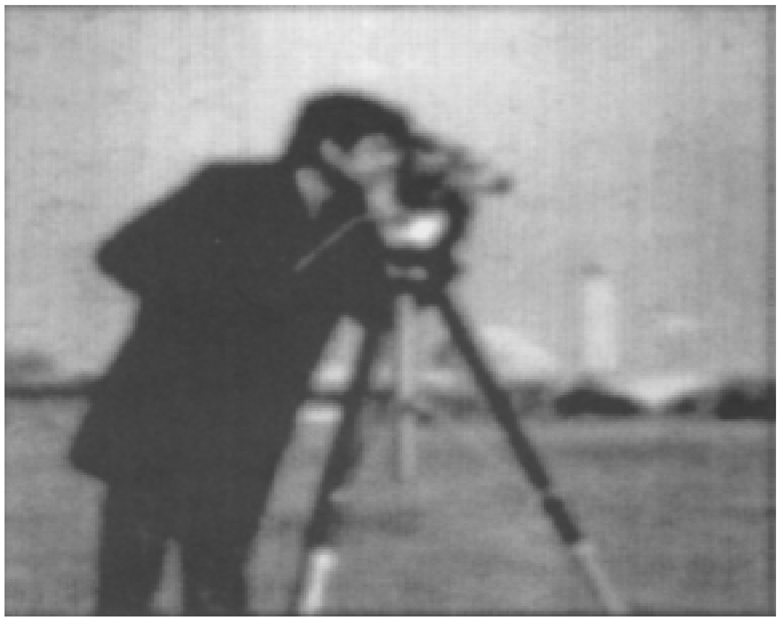


Figure 1.3

1.1.1.1 Frequency Domain Analysis

Example 1.1 (Image Blurring) looks at the original images in its typical form; however, it is often useful to look at our images and PSF in the frequency domain. In Figure 1.4, we take another look at the image blurring example above and look at how the images and results would appear in the frequency domain if we applied the fourier transforms.

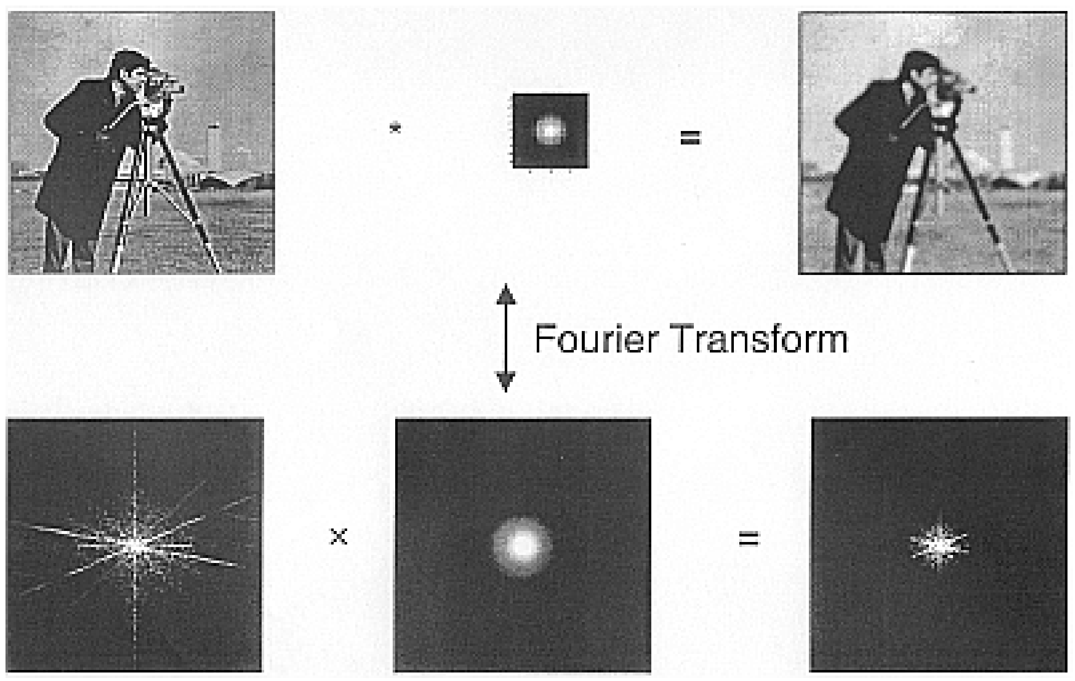


Figure 1.4

1.2 Digital Image Processing Basics[2]

1.2.1 Digital Image Processing

A sampled image gives us our usual 2D array of pixels $f[m,n]$ (Figure 1.5):

[2]This content is available online at <http://cnx.org/content/m10973/2.2/>.

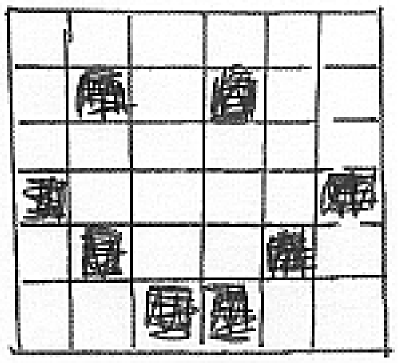


Figure 1.5: We illustrate a "pixelized" smiley face.

We can filter $f[m,n]$ by applying a 2D discrete-space convolution[3] as shown below (where $h[m,n]$ is our PSF):

$$\begin{aligned} g[m,n] &= h[m,n] * f[m,n] \\ &= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} h[m-k,n-l]\, f[k,l] \end{aligned} \tag{1.4}$$

Example 1.2: Sampled Image

[3] "Discrete Time Convolution" <http://cnx.org/content/m10087/latest/>

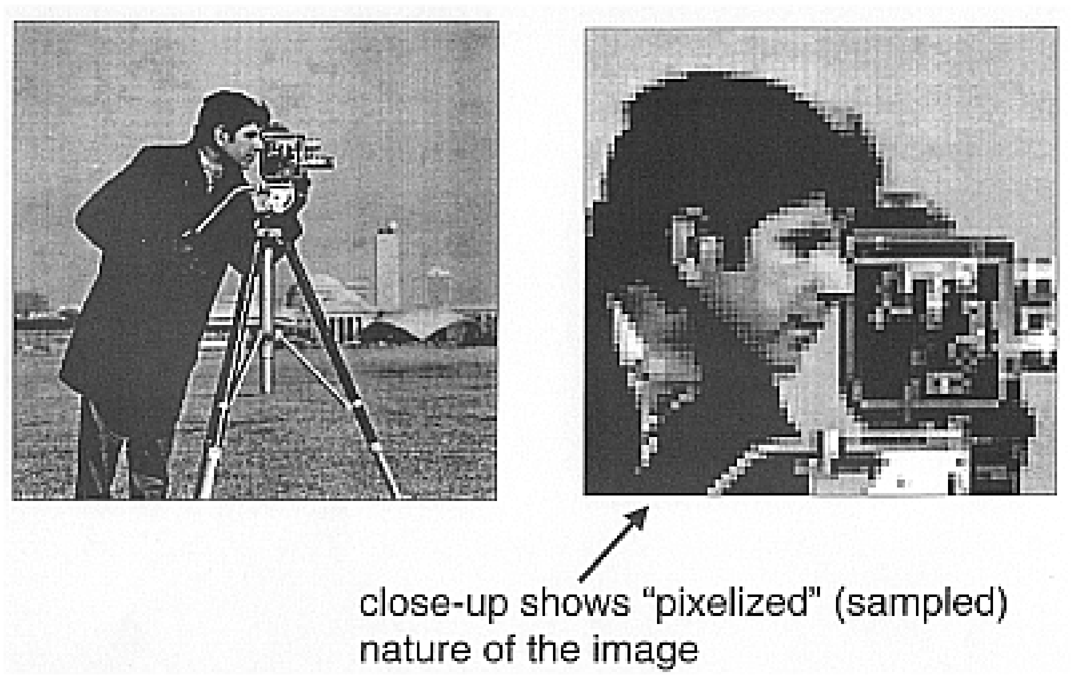


Figure 1.6: Illustrate the "pixelized" nature of all digital images.

We also have discrete-space FTS:

$$F[u,v] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n]\, e^{-(jum)} e^{-(jvm)} \tag{1.5}$$

where $F[u,v]$ is analogous to DTFT[4] in 1D.

NOTE: "Convolution in Time" is the **same** as "Multiplication in Frequency"

$$g[m,n] = h[m,n] * f[m,n] \tag{1.6}$$

which, as stated above, is the same as:

$$G[u,v] = H[u,v]\, F[u,v] \tag{1.7}$$

Example 1.3: Magnitude of FT of Cameraman Image

[4]"Discrete-Time Fourier Transform (DTFT)" <http://cnx.org/content/m10247/latest/>

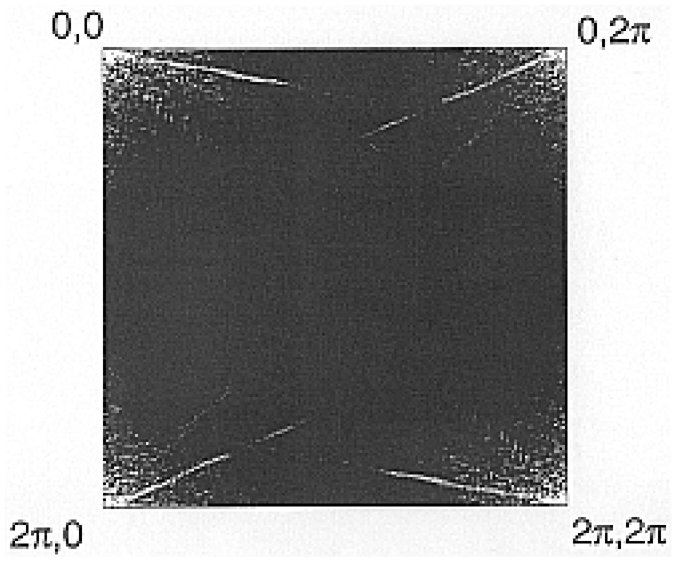


Figure 1.7

To get a better image, we can use the `fftshift` command in Matlab to center the Fourier Transform. The resulting image is shown in Figure 1.8:

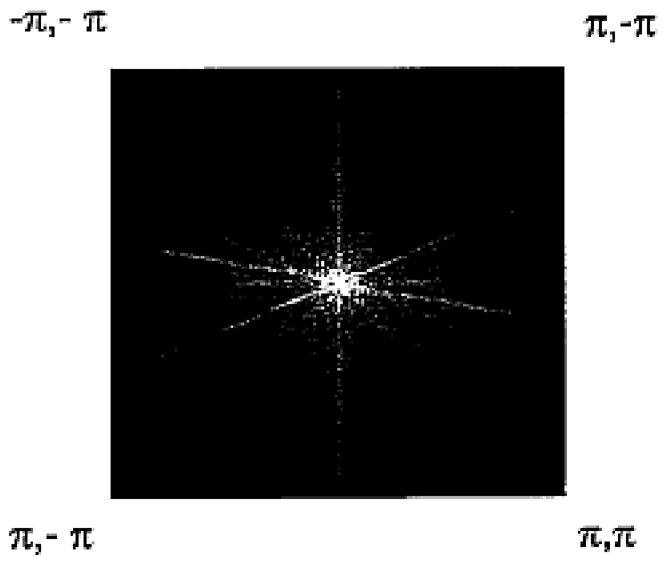


Figure 1.8

1.3 2D DFT[5]

1.3.1 2D DFT

To perform image restoration (and many other useful image processing algorithms) in a computer, we need a Fourier Transform (FT) that is discrete and two-dimensional.

$$F[k,l] = F(u,v)\,|_{u=\frac{2\pi k}{N},v=\frac{2\pi l}{N}} \tag{1.8}$$

for $k = \{0, \dots, N-1\}$ and $l = \{0, \dots, N-1\}$.

$$F(u,v) = \sum_m \sum_n f[m,n]\, e^{-(jum)} e^{-(jvm)} \tag{1.9}$$

$$F[k,l] = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f[m,n]\, e^{(-j)\frac{2\pi km}{N}} e^{(-j)\frac{2\pi ln}{N}} \tag{1.10}$$

where the above equation ((1.10)) has finite support for an NxN image.

1.3.1.1 Inverse 2D DFT

As with our regular fourier transforms, the 2D DFT also has an inverse transform that allows us to reconstruct an image as a weighted combination of complex sinusoidal basis functions.

$$f[m,n] = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} F[k,l]\, e^{\frac{j2\pi km}{N}} e^{\frac{j2\pi ln}{N}} \tag{1.11}$$

Example 1.4: Periodic Extensions

[5]This content is available online at <http://cnx.org/content/m10987/2.4/>.

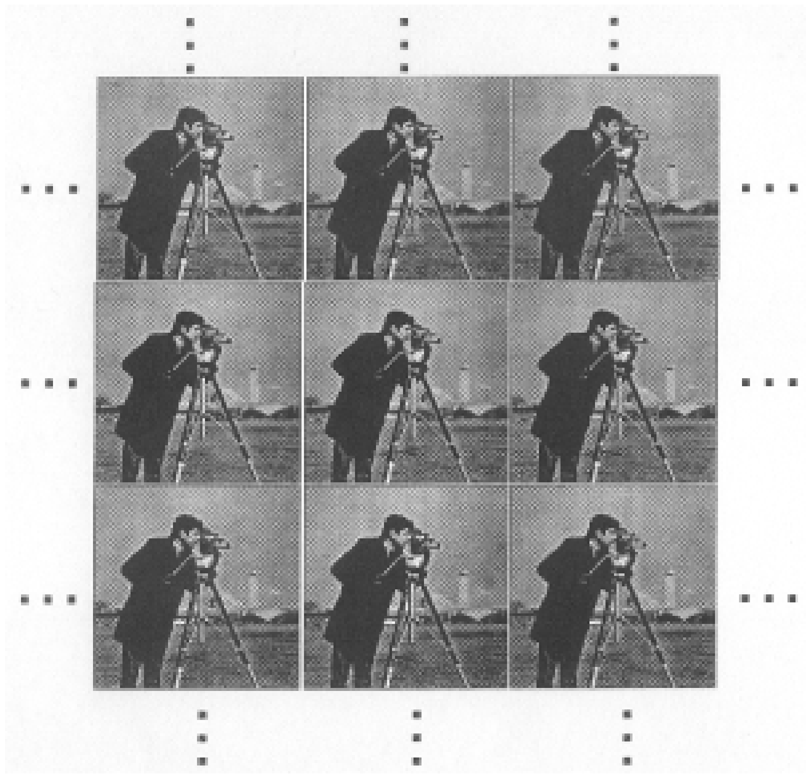


Figure 1.9: Illustrate the periodic extension of images.

1.3.2 2D DFT and Convolution

The regular 2D convolution equation is

$$g[m,n] = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} f[k,l]\, h[m-k, n-l] \tag{1.12}$$

Example 1.5

Below we go through the steps of convolving two two-dimensional arrays. You can think of f as representing an image and h represents a PSF, where $h[m,n] = 0$ for $m \ \ and \ \ n > 1$ and $m \ \ and \ \ n < 0$.

$$h = \begin{pmatrix} h[0,0] & h[0,1] \\ h[1,0] & h[1,1] \end{pmatrix}$$

$$f = \begin{pmatrix} f[0,0] & \dots & f[0,N-1] \\ \vdots & \ddots & \vdots \\ f[N-1,0] & \dots & f[N-1,N-1] \end{pmatrix}$$

Step 1 (Flip h):

$$h[-m,-n] = \begin{pmatrix} h[1,1] & h[1,0] & 0 \\ h[0,1] & h[0,0] & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{1.13}$$

Step 2 (Convolve):

$$g[0,0] = h[0,0]\, f[0,0] \tag{1.14}$$

We use the standard 2D convolution equation ((1.12)) to find the first element of our convolved image. In order to better understand what is happening, we can think of this visually. The basic idea is to take $h[-m,-n]$ and place it "on top" of $f[k,l]$, so that just the bottom-right element, $h[0,0]$ of $h[-m,-n]$ overlaps with the top-left element, $f[0,0]$, of $f[k,l]$. Then, to get the next element of our convolved image, we slide the flipped matrix, $h[-m,-n]$, over one element to the right and get the following result:

$$g[0,1] = h[0,0]\, f[0,1] + h[0,1]\, f[0,0]$$

We continue in this fashion to find all of the elements of our convolved image, $g[m,n]$. Using the above method we define the general formula to find a particular element of $g[m,n]$ as:

$$g[m,n] = h[0,0]\, f[m,n] + h[0,1]\, f[m,n-1] + h[1,0]\, f[m-1,n] + h[1,1]\, f[m-1,n-1] \tag{1.15}$$

1.3.2.1 Circular Convolution

Exercise 1.3.1 ***(Solution on p. 59.)***

What does $H[k,l]\, F[k,l]$ produce?

Due to periodic extension by DFT (Figure 1.10):

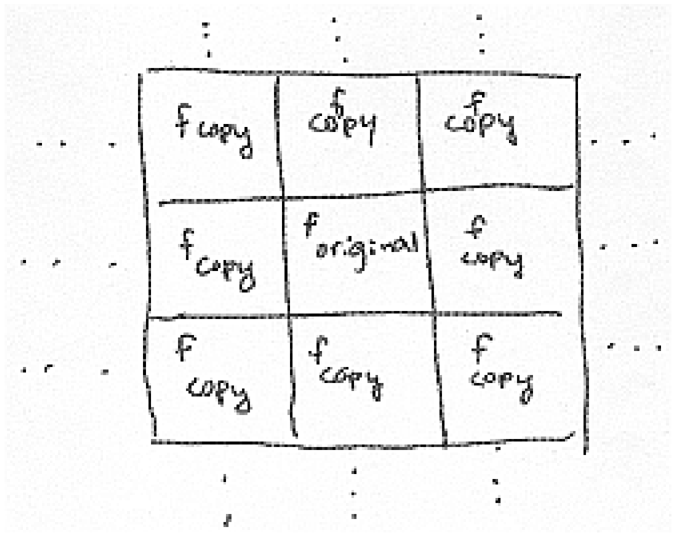


Figure 1.10

Based on the above solution, we will let

$$\tilde{g}\left[m,n\right]=\text{IDFT}\left(H\left[k,l\right]F\left[k,l\right]\right) \quad (1.16)$$

Using this equation, we can calculate the value for each position on our final image, $\tilde{g}\left[m,n\right]$. For example, due to the periodic extension of the images, when circular convolution is applied we will observe a **wrap-around** effect.

$$\tilde{g}\left[0,0\right]=h\left[0,0\right]f\left[0,0\right]+h\left[1,0\right]f\left[N-1,0\right]+h\left[0,1\right]f\left[0,N-1\right]+h\left[1,1\right]f\left[N-1,N-1\right] \quad (1.17)$$

Where the last three terms in (1.17) are a result of the wrap-around effect caused by the presence of the images copies located all around it.

1.3.2.2 Zero Padding

If the support of h is MxM and f is NxN, then we zero pad f and h to $M+N-1$ x $M+N-1$ (see Figure 1.11).

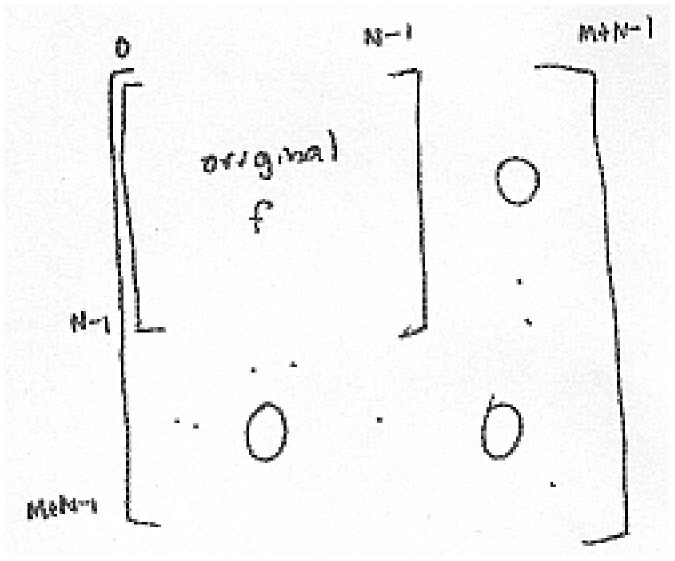


Figure 1.11

NOTE: Circular Convolution = Regular Convolution

1.3.2.3 Computing the 2D DFT

$$F\left[k,l\right]=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}f\left[m,n\right]e^{(-j)\frac{2\pi km}{N}}e^{(-j)\frac{2\pi ln}{N}} \quad (1.18)$$

where in the above equation, $\sum_{n=0}^{N-1} f[m,n]\, e^{(-j)\frac{2\pi ln}{N}}$ is simply a 1D DFT over n. This means that we will take the 1D FFT of each row; if we have N rows, then it will require $N\log N$ operations per row. We can rewrite this as

$$F[k,l] = \sum_{m=0}^{N-1} f'[m,l]\, e^{(-j)\frac{2\pi km}{N}} \tag{1.19}$$

where now we take the 1D FFT of each column, which means that if we have N columns, then it requires $N\log N$ operations per column.

NOTE: Therefore the overall complexity of a 2D FFT is

$$O\left(N^2 \log N\right)$$

where N^2 equals the number of pixels in the image.

1.4 Images: 2D signals[6]

1.4.1 Image Processing

Image not finished

Figure 1.12: Images are 2D functions $f(x,y)$

1.4.2 Linear Shift Invariant Systems

Image not finished

Figure 1.13

H is LSI if:

1. $$H(\alpha_1 f_1(x,y) + \alpha_2 f_2(x,y)) = H(f_1(x,y)) + H(f_2(x,y))$$
   for all images f_1, f_2 and scalar.
2. $$H(f(x-x_0, y-y_0)) = g(x-x_0, y-y_0)$$

LSI systems are expressed mathematically as 2D convolutions:

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x-\alpha, y-\beta)\, f(\alpha,\beta)\, d\alpha d\beta$$

where $h(x,y)$ is the 2D impulse response (also called the **point spread function**).

[6]This content is available online at <http://cnx.org/content/m10961/2.7/>.

1.4.3 2D Fourier Analysis

$$\mathcal{F}(u,v)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,e^{-(jux)}e^{-(jvy)}dxdy$$

where $\mathcal{F}$ is the 2D FT and u and v are frequency variables in $x(u)$ and $y(v)$.

2D complex exponentials are eigenfunctions[7] for 2D LSI systems:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x-\alpha,y-\beta)\,e^{ju_0\alpha}e^{jv_0\beta}d\alpha d\beta=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\alpha',\beta')\,e^{ju_0(x-\alpha')}e^{jv_0(y-\beta')}d\alpha' d\beta'= e^{ju_0x}e^{jv_0y}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\alpha',\beta')\,e^{-(ju_0\alpha')}e^{-(jv_0\beta')}d\alpha' d\beta' \quad (1.20)$$

where

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\alpha',\beta')\,e^{-\left(ju_0\alpha'\right)}e^{-\left(jv_0\beta'\right)}d\alpha' d\beta'\equiv H(u_0,v_0)$$

$H(u_0,v_0)$ is the 2D Fourier transform of $h(x,y)$ evaluated at frequencies u_0 and v_0.

Image not finished

Figure 1.14

$$\begin{aligned} g(x,y) &= h(x,y)*f(x,y) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x-\alpha,y-\beta)\,f(\alpha,\beta)\,d\alpha d\beta \end{aligned} \quad (1.21)$$

$$G(u,v)=H(u,v)\,\mathcal{F}(u,v)$$

Inverse 2D FT

$$g(x,y)=\frac{1}{(2\pi)^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(u,v)\,e^{jux}e^{jvy}dudv \quad (1.22)$$

1.4.4 2D Sampling Theory

Image not finished

Figure 1.15: Think of the image as a 2D surface.

We can **sample** the height of the surface using a 2D impulse array.

[7]"Eigenfunctions of LTI Systems" <http://cnx.org/content/m10500/latest/>

Image not finished

Figure 1.16: Impulses spaced $\Delta(x)$ apart in the horizontal direction and $\Delta(y)$ in the vertical

$$f_s(x,y) = S(x,y)\,f(x,y)$$

where $f_s(x,y)$ is sampled image in frequency
2D FT of $s(x,y)$ is a 2D impulse array in frequency $S(u,v)$

Image not finished

Figure 1.17

$$\text{multiplication in time} \Leftrightarrow \text{convolution in frequency}$$

$$F_s(u,v) = S(u,v) * \mathcal{F}(u,v)$$

Image not finished

Figure 1.18: $\mathcal{F}(u,v)$ is bandlimited in both the horizontal and vertical directions.

Image not finished

Figure 1.19: periodically replicated in (u,v) frequency plane

1.4.5 Nyquist Theorem

Assume that $f(x,y)$ is bandlimited to $\pm(B_x)$, $\pm(B_y)$:

Figure 1.20

If we sample $f(x,y)$ at spacings of $\Delta(x) < \frac{\pi}{B_x}$ and $\Delta(y) < \frac{\pi}{B_y}$, then $f(x,y)$ can be perfectly recovered from the samples by lowpass filtering:

Figure 1.21: ideal lowpass filter, 1 inside rectangle, 0 outside

Aliasing in 2D

(a)

(b)

Figure 1.22

1.5 DFT as a Matrix Operation[8]

[8]This content is available online at <http://cnx.org/content/m10962/2.5/>.

1.5.1 Matrix Review

Recall:

- Vectors in $\mathbb{R}^N$:
$$x = \begin{pmatrix} x_0 \\ x_1 \\ \dots \\ x_{N-1} \end{pmatrix} \quad , \quad x_i \in \mathbb{R}$$
- Vectors in $\mathbb{C}^N$:
$$x = \begin{pmatrix} x_0 \\ x_1 \\ \dots \\ x_{N-1} \end{pmatrix} \quad , \quad x_i \in \mathbb{C}$$
- Transposition:
  a. transpose:
  $$x^T = \begin{pmatrix} x_0 & x_1 & \dots & x_{N-1} \end{pmatrix}$$
  b. conjugate:
  $$x^H = \begin{pmatrix} {x_0}^* & {x_1}^* & \dots & {x_{N-1}}^* \end{pmatrix}$$
- Inner product[9]:
  a. real:
  $$x^T y = \sum_{i=0}^{N-1} x_i y_i$$
  b. complex:
  $$x^H y = \sum_{i=0}^{N-1} {x_n}^* y_n$$
- Matrix Multiplication:
$$Ax = \begin{pmatrix} a_{00} & a_{01} & \dots & a_{0,N-1} \\ a_{10} & a_{11} & \dots & a_{1,N-1} \\ \vdots & \vdots & \dots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \dots & a_{N-1,N-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \dots \\ x_{N-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \dots \\ y_{N-1} \end{pmatrix}$$
$$y_k = \sum_{n=0}^{N-1} a_{kn} x_n$$
- Matrix Transposition:
$$A^T = \begin{pmatrix} a_{00} & a_{10} & \dots & a_{N-1,0} \\ a_{01} & a_{11} & \dots & a_{N-1,1} \\ \vdots & \vdots & \dots & \vdots \\ a_{0,N-1} & a_{1,N-1} & \dots & a_{N-1,N-1} \end{pmatrix}$$

[9]"Inner Products" <http://cnx.org/content/m10755/latest/>

Matrix transposition involved simply swapping the rows with columns.

$$A^H = A^{T*}$$

The above equation is Hermitian transpose.

$$A^T{}_{k,n} = A_{n,k}$$

$$A^H{}_{k,n} = A^*{}_{n,k}$$

1.5.2 Representing DFT as Matrix Operation

Now let's represent the DFT[10] in vector-matrix notation.

$$x = \begin{pmatrix} x[0] \\ x[1] \\ \dots \\ x[N-1] \end{pmatrix}$$

$$X = \begin{pmatrix} X[0] \\ X[1] \\ \dots \\ X[N-1] \end{pmatrix} \in \mathbb{C}^N$$

Here x is the vector of time samples and X is the vector of DFT coefficients. How are x and X related:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\left(j\frac{2\pi}{N}kn\right)}$$

where

$$a_{kn} = \left(e^{-\left(j\frac{2\pi}{N}\right)}\right)^{kn} = W_N{}^{kn}$$

so

$$X = Wx$$

where X is the DFT vector, W is the matrix and x the time domain vector.

$$W_{k,n} = \left(e^{-\left(j\frac{2\pi}{N}\right)}\right)^{kn}$$

$$X = W \begin{pmatrix} x[0] \\ x[1] \\ \dots \\ x[N-1] \end{pmatrix}$$

IDFT:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \left(e^{j\frac{2\pi}{N}}\right)^{nk}$$

[10]"Discrete Fourier Transform (DFT)" <http://cnx.org/content/m10249/latest/>

where

$$\left(e^{j\frac{2\pi}{N}}\right)^{nk} = {W_N}^{nk*}$$

${W_N}^{nk*}$ is the matrix Hermitian transpose. So,

$$x = \frac{1}{N} W^H X$$

where x is the time vector, $\frac{1}{N} W^H$ is the inverse DFT matrix, and X is the DFT vector.

1.6 Fast Convolution Using the FFT[11]

1.6.1 Important Application of the FFT

Exercise 1.6.1 ***(Solution on p. 59.)***

How many complex multiplies and adds are required to convolve two N-pt sequences?

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

1. Zero-pad these two sequences to length $2N-1$, take DFTs using the FFT algorithm

$$x[n] \to X[k]$$

$$h[n] \to H[k]$$

   The cost is

$$O((2N-1)\log(2N-1)) = O(N\log N)$$

2. Multiply DFTs

$$X[k]\, H[k]$$

   The cost is

$$O(2N-1) = O(N)$$

3. Inverse DFT using FFT

$$X[k]\, H[k] \to y[n]$$

   The cost is

$$O((2N-1)\log(2N-1)) = O(N\log N)$$

So the total cost for direct convolution of two N-point sequences is $O(N^2)$. Total cost for convolution using FFT algorithm is $O(N\log N)$. That is a **huge savings** (Figure 1.23).

Figure 1.23

[11] This content is available online at <http://cnx.org/content/m10963/2.6/>.

1.6.2 Summary of DFT

- $x[n]$ is an N-point signal (Figure 1.24).

Figure 1.24

- $$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\left(j\frac{2\pi}{N}kn\right)} = \sum_{n=0}^{N-1} x[n] {W_N}^{kn}$$

  where $W_N = e^{-\left(j\frac{2\pi}{N}\right)}$ is a "twiddle factor" and the first part is the basic DFT.

1.6.2.1 What is the DFT

$$X[k] = X\left[F = \frac{k}{N}\right] = \sum_{n=0}^{N-1} x[n] e^{-(j2\pi Fn)}$$

where $X\left[F = \frac{k}{N}\right]$ is the DTFT of $x[n]$ and $\sum_{n=0}^{N-1} x[n] e^{-(j2\pi Fn)}$ is the DTFT of $x[n]$ at digital frequency F. This is a sample of the DTFT. We can do frequency domain analysis on a computer!

1.6.2.2 Inverse DFT (IDFT)

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

- Build $x[n]$ using **Simple** complex sinusoidal **building block** signals
- Amplitude of each complex sinusoidal building block in $x[n]$ is $\frac{1}{N}X[k]$

1.6.2.3 Circular Convolution

DFT

$$x[n] \oplus h[n] \leftrightarrow X[k] H[k] \tag{1.23}$$

1.6.2.4 Regular Convolution from Circular Convolution

- Zero pad $x[n]$ and $h[n]$ to length $= \text{length}(x) + \text{length}(h) - 1$
- Zero padding increases frequency resolution in DFT domain (Figure 1.25)

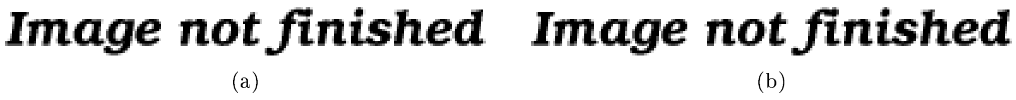


Figure 1.25: (a) 8-pt DFT of 8-pt signal (b) 16-pt DFT of same signal padded with 8 additional zeros

1.6.2.5 The Fast Fourier Transform (FFT)

- Efficient computational algorithm for calculating the DFT
- "Divide and conquer"
- Break signal into even and odd samples keep taking shorter and shorter DFTs, then build N-pt DFT by cleverly combining shorter DFTs
- N-pt DFT: $O\left(N^2\right) \rightarrow O\left(N \log_2 N\right)$

1.6.2.6 Fast Convolution

- Use FFT's to compute circular convolution of zero-padded signals
- Much faster than regular convolution if signal lengths are long
- $O\left(N^2\right) \rightarrow O\left(N \log_2 N\right)$

See Figure 1.26.

Figure 1.26

1.7 The FFT Algorithm[12]

Definition 1.1: FFT
(Fast Fourier Transform) An efficient **computational algorithm** for computing the DFT[13].

1.7.1 The Fast Fourier Transform FFT

DFT can be **expensive** to compute **directly**

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\left(j 2\pi \frac{k}{N} n\right)} \quad , \quad 0 \leq k \leq N-1$$

For each k, we must execute:

[12]This content is available online at <http://cnx.org/content/m10964/2.6/>.
[13]"Discrete Fourier Transform (DFT)" <http://cnx.org/content/m10249/latest/>

- N complex multiplies
- $N-1$ complex adds

The total cost of direct computation of an N-point DFT is

- N^2 complex multiplies
- $N\left(N-1\right)$ complex adds

How many adds and mults of **real** numbers are required?

This " $O\left(N^2\right)$" computation rapidly gets out of hand, as N gets large:

N	1	10	100	1000	10^6
N^2	1	100	10,000	10^6	10^{12}

Table 1.1

Image not finished

Figure 1.27

The FFT provides us with a much more **efficient** way of computing the DFT. The FFT requires only " $O\left(N\text{log}N\right)$" computations to compute the N-point DFT.

N	10	100	1000	10^6
N^2	100	10,000	10^6	10^{12}
$N\text{log}N$	10	200	3000	6×10^6

Table 1.2

How long is $10^{12}\mu$sec? More than 10 days! How long is $6\times10^6\mu$sec?

Image not finished

Figure 1.28

The FFT and digital computers revolutionized DSP (1960 - 1980).

1.7.2 How does the FFT work?

- The FFT exploits the **symmetries** of the complex exponentials ${W_N}^{kn}=e^{-\left(j\frac{2\pi}{N}kn\right)}$
- ${W_N}^{kn}$ are called "**twiddle factors**"

Rule 1.1: Complex Conjugate Symmetry

$$W_N{}^{k(N-n)} = W_N{}^{-(kn)} = W_N{}^{kn\,*}$$

$$e^{-\left(j2\pi\frac{k}{N}(N-n)\right)} = e^{j2\pi\frac{k}{N}n} = e^{-\left(j2\pi\frac{k}{N}n\right)^{*}}$$

Rule 1.2: Periodicity in n and k

$$W_N{}^{kn} = W_N{}^{k(N+n)} = W_N{}^{(k+N)n}$$

$$e^{-\left(j\frac{2\pi}{N}kn\right)} = e^{-\left(j\frac{2\pi}{N}k(N+n)\right)} = e^{-\left(j\frac{2\pi}{N}(k+N)n\right)}$$

$$W_N = e^{-\left(j\frac{2\pi}{N}\right)}$$

1.7.3 Decimation in Time FFT

- Just one of **many** different FFT algorithms
- The **idea** is to build a DFT out of smaller and smaller DFTs by decomposing $x[n]$ into smaller and smaller subsequences.
- Assume $N = 2^m$ (a power of 2)

1.7.3.1 Derivation

N is **even**, so we can complete $X[k]$ by separating $x[n]$ into **two** subsequences each of length $\frac{N}{2}$.

$$x[n] \to \begin{cases} \frac{N}{2} & \text{if} \quad n = \text{even} \\ \frac{N}{2} & \text{if} \quad n = \text{odd} \end{cases}$$

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N{}^{kn} \quad , \quad 0 \le k \le N-1$$

$$X[k] = \sum_{n=2r} x[n] W_N{}^{kn} + \sum_{n=2r+1} x[n] W_N{}^{kn}$$

where $0 \le r \le \frac{N}{2} - 1$. So

$$\begin{aligned} X[k] &= \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_N{}^{2kr} + \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_N{}^{(2r+1)k} \\ &= \sum_{r=0}^{\frac{N}{2}-1} x[2r] \left(W_N{}^2\right)^{kr} + W_N{}^k \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] \left(W_N{}^2\right)^{kr} \end{aligned} \tag{1.24}$$

where $W_N{}^2 = e^{-\left(j\frac{2\pi}{N}2\right)} = e^{-\left(j\frac{2\pi}{\frac{N}{2}}\right)} = W_{\frac{N}{2}}$. So

$$X[k] = \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_{\frac{N}{2}}{}^{kr} + W_N{}^k \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_{\frac{N}{2}}{}^{kr}$$

where $\sum_{r=0}^{\frac{N}{2}-1} x[2r] W_{\frac{N}{2}}{}^{kr}$ is $\frac{N}{2}$-point DFT of even samples ($G[k]$) and $\sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_{\frac{N}{2}}{}^{kr}$ is $\frac{N}{2}$-point DFT of odd samples ($H[k]$).

$$X[k] = G[k] + W_N{}^k H[k] \quad , \quad 0 \le k \le N-1$$

Decomposition of an N-point DFT as a sum of 2 $\frac{N}{2}$-point DFTs.

Why would we want to do this? **Because it is more efficient!**

NOTE: Cost to compute an N-point DFT is approximately N^2 complex mults and adds.

But decomposition into 2 $\frac{N}{2}$-point DFTs + combination requires only

$$\left(\frac{N}{2}\right)^2 + \left(\frac{N}{2}\right)^2 + N = \frac{N^2}{2} + N$$

where the first part is the number of complex mults and adds for $\frac{N}{2}$-point DFT, $G[k]$. The second part is the number of complex mults and adds for $\frac{N}{2}$-point DFT, $H[k]$. The third part is the number of complex mults and adds for combination. And the total is $\frac{N^2}{2} + N$ complex mults and adds.

Example 1.6: Savings

For $N = 1000$,

$$N^2 = 10^6$$

$$\frac{N^2}{2} + N = \frac{10^6}{2} + 1000$$

Because 1000 is small compared to 500,000,

$$\frac{N^2}{2} + N \simeq \frac{10^6}{2}$$

So why stop here?! Keep decomposing. Break each of the $\frac{N}{2}$-point DFTs into two $\frac{N}{4}$-point DFTs, *etc.*, ...

We can keep decomposing:

$$\frac{N}{2^1} = \left\{\frac{N}{2}, \frac{N}{4}, \frac{N}{8}, \dots, \frac{N}{2^{m-1}}, \frac{N}{2^m}\right\} = 1$$

where

$$m = \log_2 N = \text{ times}$$

Computational cost: N-pt DFT $\longrightarrow$ two $\frac{N}{2}$-pt DFTs. The cost is $N^2 \rightarrow 2\left(\frac{N}{2}\right)^2 + N$. So replacing each $\frac{N}{2}$-pt DFT with two $\frac{N}{4}$-pt DFTs will reduce cost to

$$2\left(2\left(\frac{N}{4}\right)^2 + \frac{N}{2}\right) + N = 4\left(\frac{N}{4}\right)^2 + 2N = \frac{N^2}{2^2} + 2N = \frac{N^2}{2^p} + pN$$

As we keep going $p = \{3, 4, \dots, m\}$, where $m = \log_2 N$. We get the cost

$$\frac{N^2}{2^{\log_2 N}} + N\log_2 N = \frac{N^2}{N} + N\log_2 N = N + N\log_2 N$$

$N + N\log_2 N$ is the total number of complex adds and mults.

For large N, cost $\simeq N\log_2 N$ or " $O(N\log_2 N)$", since $N\log_2 N \gg N$ for large N.

Image not finished

Figure 1.29: $N = 8$ point FFT. Summing nodes ${W_n}^k$ twiddle multiplication factors.

NOTE: Weird order of time samples

Image not finished

Figure 1.30: This is called "butterflies."

1.8 The DFT: Frequency Domain with a Computer Analysis[14]

1.8.1 Introduction

We just covered ideal (and non-ideal) (time) sampling of CT signals (Section 1.10). This enabled DT signal processing solutions for CT applications (Figure 1.31):

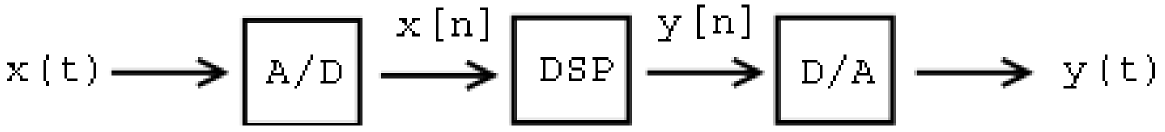


Figure 1.31

Much of the theoretical analysis of such systems relied on frequency domain representations. How do we carry out these frequency domain analysis on the computer? Recall the following relationships:

$$x[n] \overset{\text{DTFT}}{\leftrightarrow} X(\omega)$$

$$x(t) \overset{\text{CTFT}}{\leftrightarrow} X(\Omega)$$

where ω and Ω are continuous frequency variables.

1.8.1.1 Sampling DTFT

Consider the DTFT of a discrete-time (DT) signal $x[n]$. Assume $x[n]$ is of finite duration N (*i.e.*, an N-point signal).

$$X(\omega) = \sum_{n=0}^{N-1} x[n] e^{(-j)\omega n} \tag{1.25}$$

where $X(\omega)$ is the continuous function that is indexed by the real-valued parameter $-\pi \leq \omega \leq \pi$. The other function, $x[n]$, is a discrete function that is indexed by integers.

[14]This content is available online at <http://cnx.org/content/m10992/2.3/>.

We want to work with $X(\omega)$ on a computer. Why not just **sample** $X(\omega)$?

$$\begin{aligned} X[k] &= X\left(\frac{2\pi}{N}k\right) \\ &= \sum_{n=0}^{N-1} x[n]\, e^{(-j)2\pi\frac{k}{N}n} \end{aligned} \tag{1.26}$$

In (1.26) we sampled at $\omega = \frac{2\pi}{N}k$ where $k = \{0, 1, \dots, N-1\}$ and $X[k]$ for $k = \{0, \dots, N-1\}$ is called the **Discrete Fourier Transform (DFT)** of $x[n]$.

Example 1.7

Finite Duration DT Signal

Figure 1.32

The DTFT of the image in Figure 1.32 (Finite Duration DT Signal) is written as follows:

$$X(\omega) = \sum_{n=0}^{N-1} x[n]\, e^{(-j)\omega n} \tag{1.27}$$

where ω is any 2π-interval, for example $-\pi \leq \omega \leq \pi$.

Sample X(ω)

Figure 1.33

where again we sampled at $\omega = \frac{2\pi}{N}k$ where $k = \{0, 1, \dots, M-1\}$. For example, we take

$$M = 10$$

. In the following section (Section 1.8.1.1.1: Choosing M) we will discuss in more detail how we should choose M, the number of samples in the 2π interval.

(This is precisely how we would plot $X(\omega)$ in Matlab.)

1.8.1.1.1 Choosing M

1.8.1.1.1.1 Case 1

Given N (length of $x[n]$), choose $M \gg N$ to obtain a dense sampling of the DTFT (Figure 1.34):

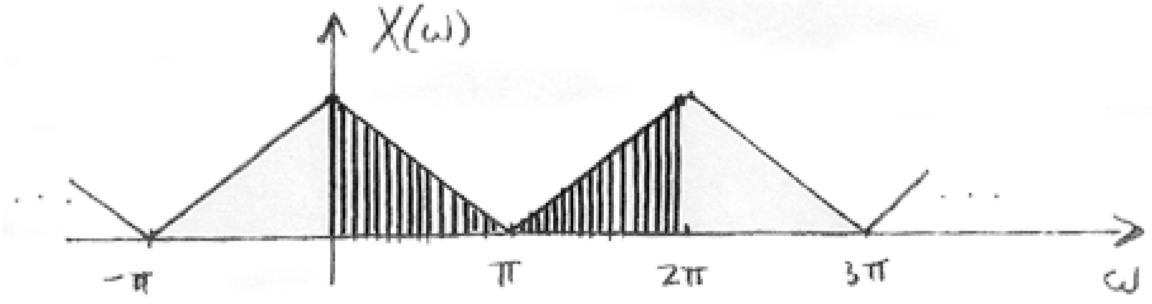


Figure 1.34

1.8.1.1.1.2 Case 2

Choose M as small as possible (to minimize the amount of computation).

In general, we require $M \geq N$ in order to represent all information in

$$x[n] \quad , \quad n = \{0, \dots, N-1\}$$

Let's concentrate on $M = N$:

$$x[n] \overset{\text{DFT}}{\leftrightarrow} X[k]$$

for $n = \{0, \dots, N-1\}$ and $k = \{0, \dots, N-1\}$

$$\text{numbers} \leftrightarrow \text{N} \quad \text{numbers}$$

1.8.2 Discrete Fourier Transform (DFT)

Define

$$X[k] \equiv X\left(\frac{2\pi k}{N}\right) \tag{1.28}$$

where $N = \text{length}(x[n])$ and $k = \{0, \dots, N-1\}$. In this case, $M = N$.

DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{(-j)2\pi \frac{k}{N} n} \tag{1.29}$$

Inverse DFT (IDFT)

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{k}{N} n} \tag{1.30}$$

1.8.2.1 Interpretation

Represent $x[n]$ in terms of a sum of N complex sinusoids[15] of amplitudes $X[k]$ and frequencies

$$\omega_k = \frac{2\pi k}{N} \quad , \quad k \in \{0, \dots, N-1\}$$

NOTE: Fourier Series with fundamental frequency $\frac{2\pi}{N}$

1.8.2.1.1 Remark 1

IDFT treats $x[n]$ as though it were N-periodic.

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{k}{N} n} \tag{1.31}$$

where $n \in \{0, \dots, N-1\}$

Exercise 1.8.1 ***(Solution on p. 59.)***

What about other values of n?

1.8.2.1.2 Remark 2

Proof that the IDFT inverts the DFT for $n \in \{0, \dots, N-1\}$

$$\begin{aligned} \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{k}{N} n} &= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{m=0}^{N-1} x[m] e^{(-j)2\pi \frac{k}{N} m} e^{j2\pi \frac{k}{N} n} \\ &= ??? \end{aligned} \tag{1.32}$$

Example 1.8: Computing DFT

Given the following discrete-time signal (Figure 1.35) with $N = 4$, we will compute the DFT using two different methods (the DFT Formula and Sample DTFT):

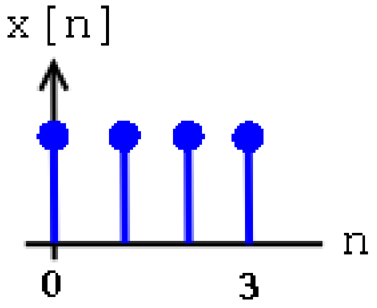


Figure 1.35

[15] "Continuous Time Complex Exponential" <http://cnx.org/content/m10060/latest/>

1. DFT Formula

$$\begin{aligned} X[k] &= \sum_{n=0}^{N-1} x[n]\, e^{(-j)2\pi \frac{k}{N} n} \\ &= 1 + e^{(-j)2\pi \frac{k}{4}} + e^{(-j)2\pi \frac{k}{4}2} + e^{(-j)2\pi \frac{k}{4}3} \\ &= 1 + e^{(-j)\frac{\pi}{2}k} + e^{(-j)\pi k} + e^{(-j)\frac{3}{2}\pi k} \end{aligned} \quad (1.33)$$

Using the above equation, we can solve and get the following results:

$$x[0] = 4$$

$$x[1] = 0$$

$$x[2] = 0$$

$$x[3] = 0$$

2. Sample DTFT. Using the same figure, Figure 1.35, we will take the DTFT of the signal and get the following equations:

$$\begin{aligned} X(\omega) &= \sum_{n=0}^{3} e^{(-j)\omega n} \\ &= \frac{1-e^{(-j)4\omega}}{1-e^{(-j)\omega}} \\ &= ??? \end{aligned} \quad (1.34)$$

Our sample points will be:

$$\omega_k = \frac{2\pi k}{4} = \frac{\pi}{2}k$$

where $k = \{0, 1, 2, 3\}$ (Figure 1.36).

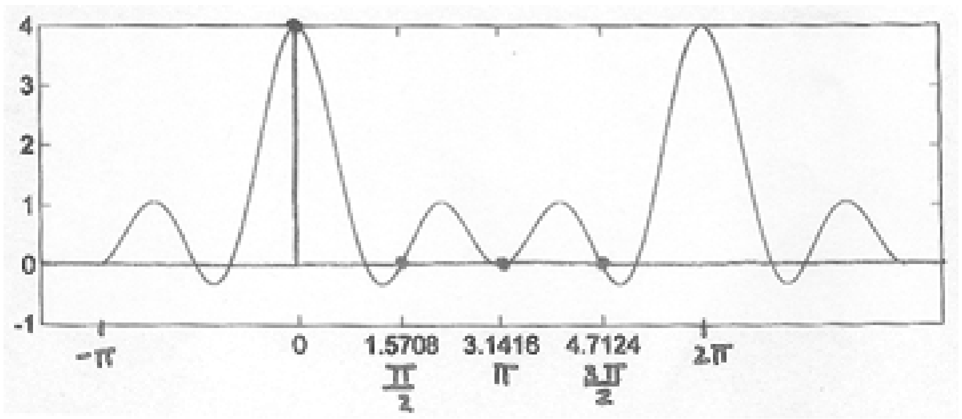


Figure 1.36

1.8.3 Periodicity of the DFT

DFT $X[k]$ consists of **samples** of DTFT, so $X(\omega)$, a 2π-periodic DTFT signal, can be converted to $X[k]$, an N-periodic DFT.

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{(-j)2\pi \frac{k}{N} n} \tag{1.35}$$

where $e^{(-j)2\pi \frac{k}{N} n}$ is an N-periodic basis function (See Figure 1.37).

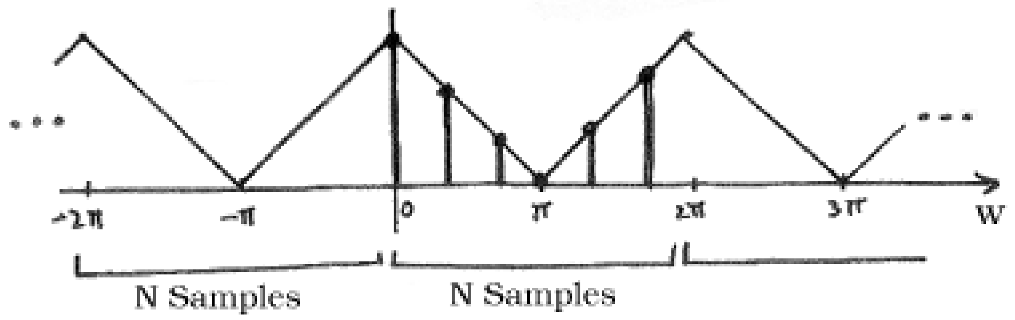


Figure 1.37

Also, recall,

$$\begin{aligned} x[n] &= \frac{1}{N}\sum_{n=0}^{N-1} X[k] e^{j2\pi \frac{k}{N} n} \\ &= \frac{1}{N}\sum_{n=0}^{N-1} X[k] e^{j2\pi \frac{k}{N}(n+mN)} \\ &= ??? \end{aligned} \tag{1.36}$$

Example 1.9: Illustration

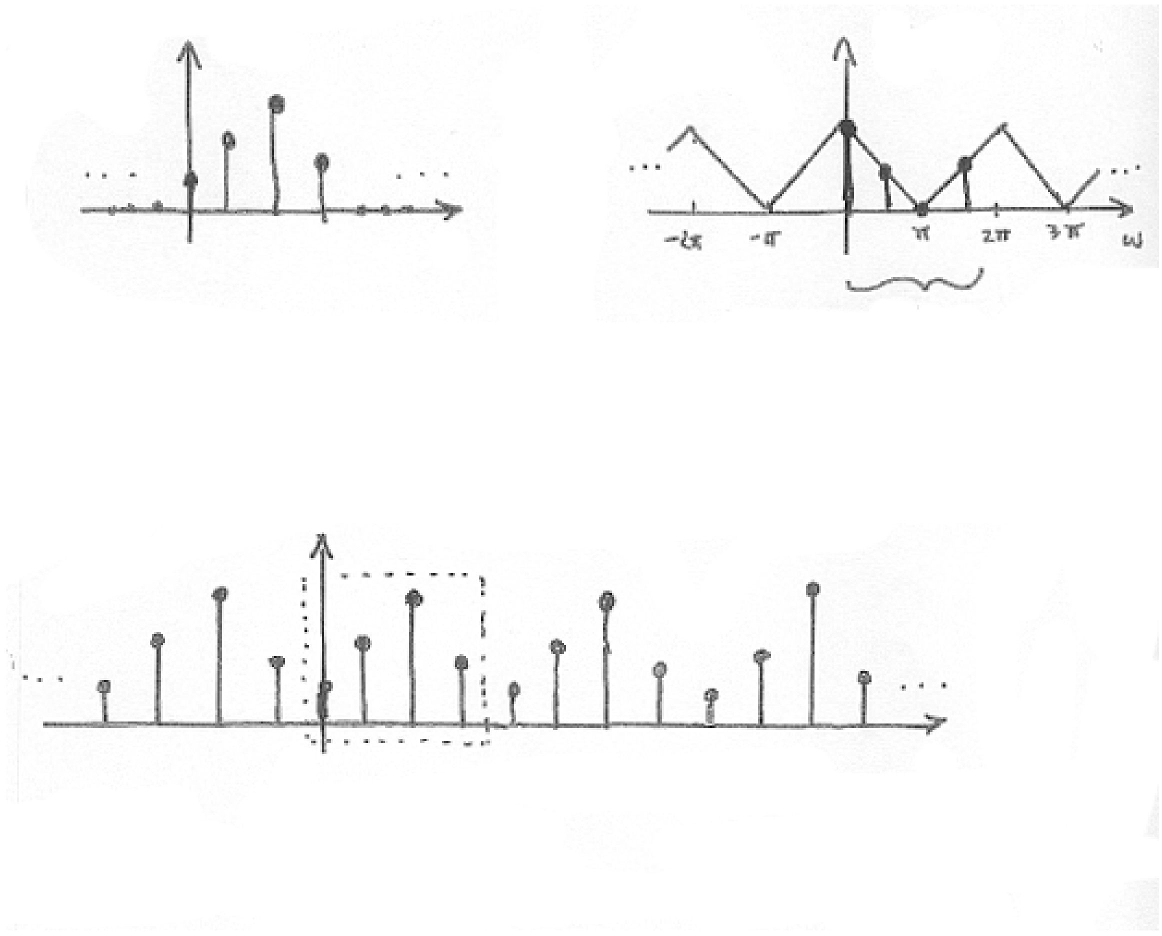


Figure 1.38

NOTE: When we deal with the DFT, we need to remember that, in effect, this treats the signal as an N-periodic sequence.

1.8.4 A Sampling Perspective

Think of sampling the continuous function $X(\omega)$, as depicted in Figure 1.39. $S(\omega)$ will represent the sampling function applied to $X(\omega)$ and is illustrated in Figure 1.39 as well. This will result in our discrete-time sequence, $X[k]$.

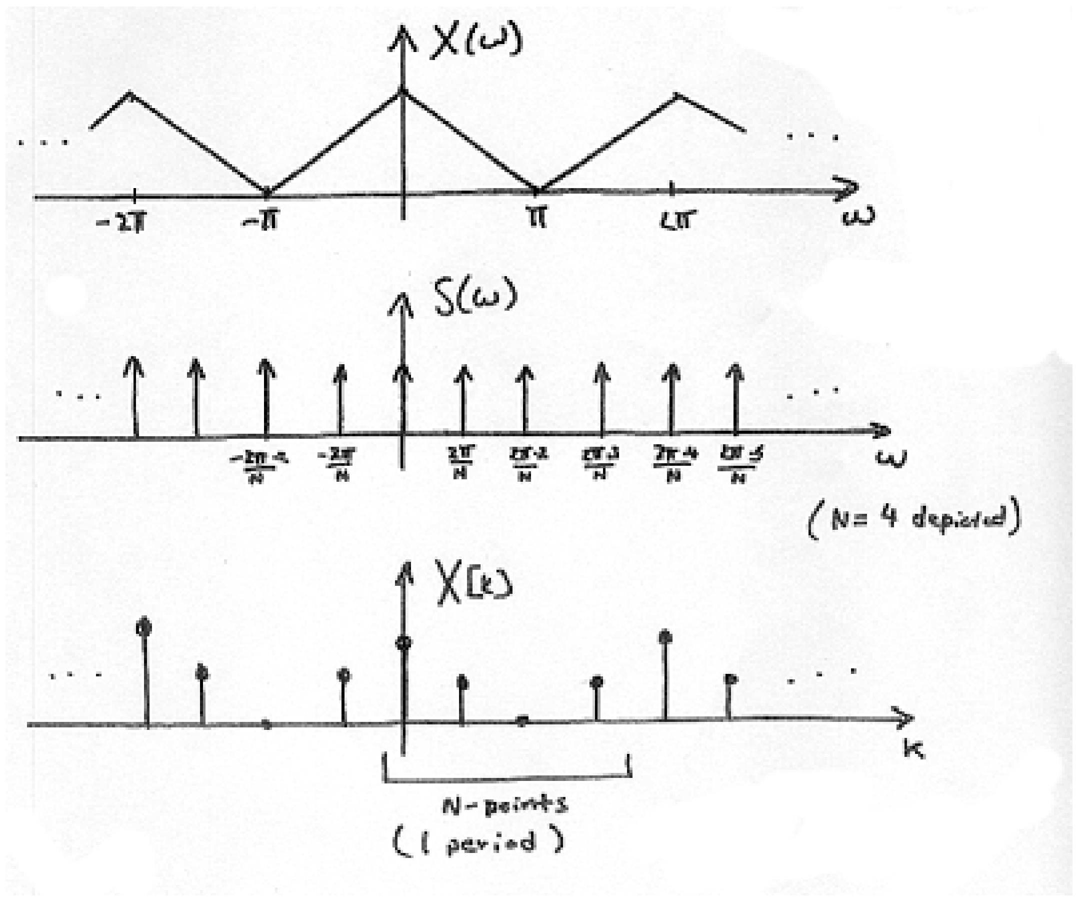


Figure 1.39

NOTE: Remember the multiplication in the frequency domain is equal to convolution in the time domain!

1.8.4.1 Inverse DTFT of S(ω)

$$\sum_{k=-\infty}^{\infty} \delta\left(\omega - \frac{2\pi k}{N}\right) \tag{1.37}$$

Given the above equation, we can take the DTFT and get the following equation:

$$N \sum_{m=-\infty}^{\infty} \delta[n - mN] \equiv S[n] \tag{1.38}$$

Exercise 1.8.2 ***(Solution on p. 59.)***

Why does (1.38) equal $S[n]$?

So, in the time-domain we have (Figure 1.40):

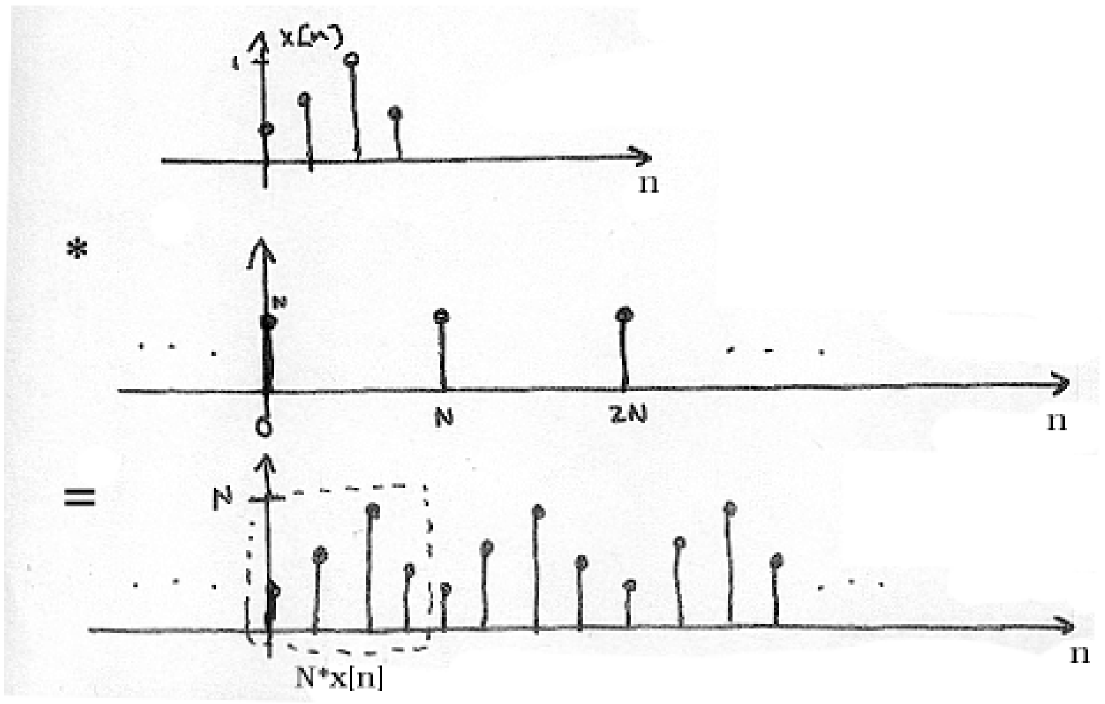


Figure 1.40

1.8.5 Connections

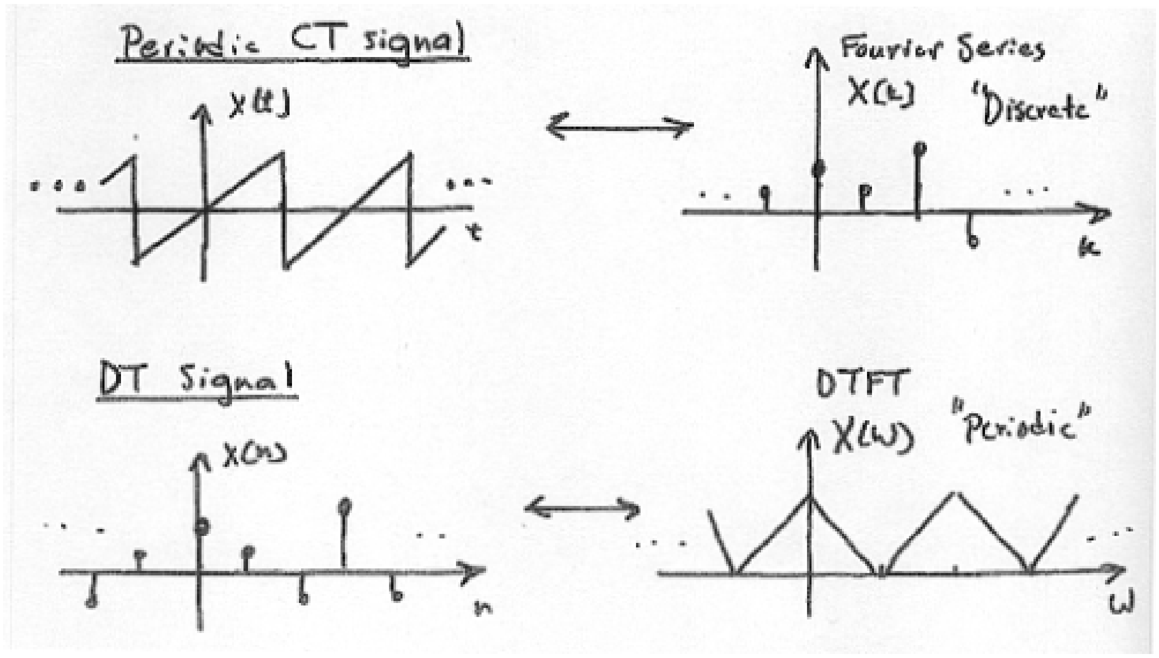


Figure 1.41

Combine signals in Figure 1.41 to get signals in Figure 1.42.

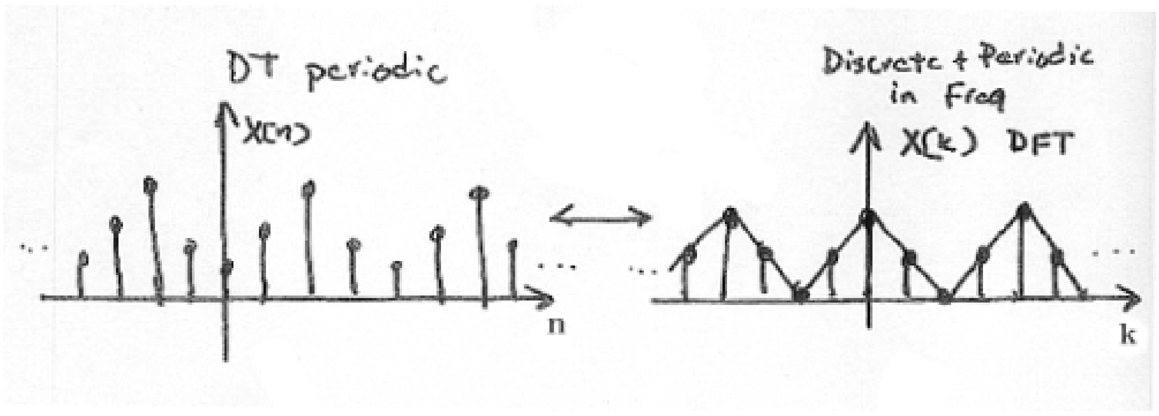


Figure 1.42

1.9 Discrete-Time Processing of CT Signals[16]

1.9.1 DT Processing of CT Signals

DSP System

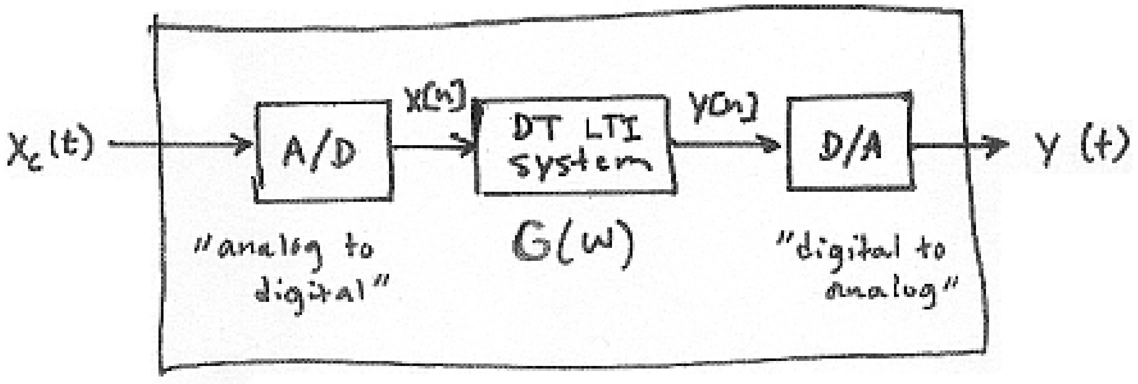


Figure 1.43

1.9.1.1 Analysis

$$Y_c(\Omega) = H_{\text{LP}}(\Omega)\, Y(\Omega T) \tag{1.39}$$

where we know that $Y(\omega) = X(\omega)\, G(\omega)$ and $G(\omega)$ is the frequency response of the DT LTI system. Also, remember that

$$\omega \equiv \Omega T$$

So,

$$Y_c(\Omega) = H_{\text{LP}}(\Omega)\, G(\Omega T)\, X(\Omega T) \tag{1.40}$$

where $Y_c(\Omega)$ and $H_{\text{LP}}(\Omega)$ are CTFTs and $G(\Omega T)$ and $X(\Omega T)$ are DTFTs.

NOTE:

$$X(\omega) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\omega - 2\pi k}{T}\right)$$

OR

$$X(\Omega T) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} X_c(\Omega - k\Omega_s)$$

Therefore our final output signal, $Y_c(\Omega)$, will be:

$$Y_c(\Omega) = H_{\text{LP}}(\Omega)\, G(\Omega T)\left(\frac{2\pi}{T} \sum_{k=-\infty}^{\infty} X_c(\Omega - k\Omega_s)\right) \tag{1.41}$$

[16]This content is available online at <http://cnx.org/content/m10993/2.2/>.

Now, if $X_c(\Omega)$ is bandlimited to $\left[-\frac{\Omega_s}{2}, \frac{\Omega_s}{2}\right]$ and we use the usual lowpass reconstruction filter in the D/A, Figure 1.44:

Image not finished

Figure 1.44

Then,

$$Y_c(\Omega) = \begin{cases} G(\Omega T) X_c(\Omega) & \text{if } |\Omega| < \frac{\Omega_s}{2} \\ 0 & \text{otherwise} \end{cases} \quad (1.42)$$

1.9.1.2 Summary

For bandlimited signals sampled at or above the Nyquist rate, we can relate the input and output of the DSP system by:

$$Y_c(\Omega) = G_{\text{eff}}(\Omega) X_c(\Omega) \tag{1.43}$$

where

$$G_{\text{eff}}(\Omega) = \begin{cases} G(\Omega T) & \text{if } |\Omega| < \frac{\Omega_s}{2} \\ 0 & \text{otherwise} \end{cases}$$

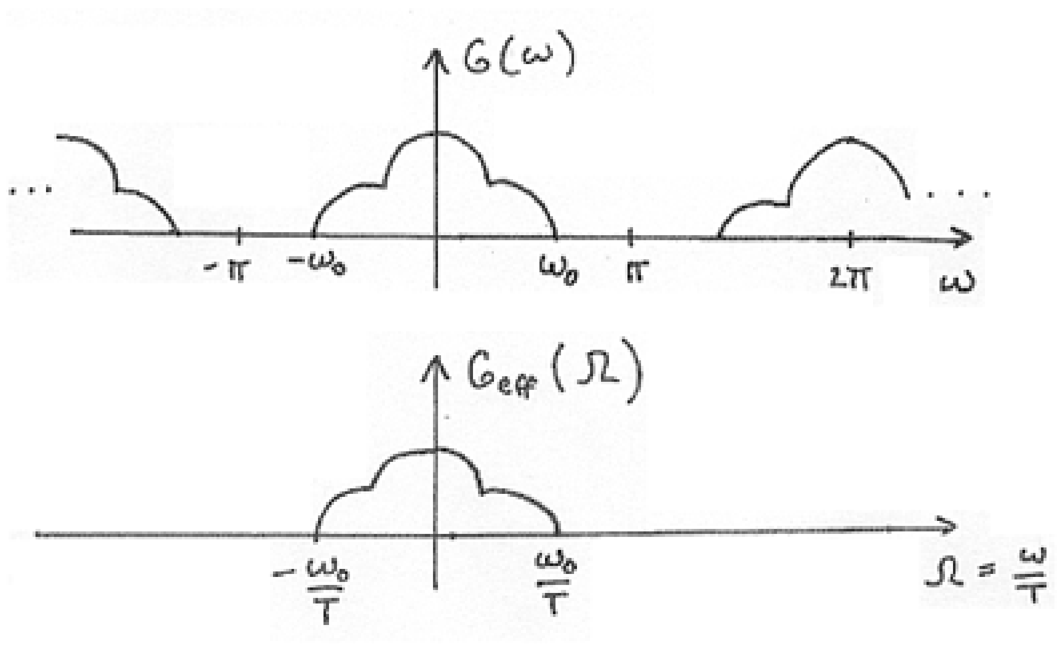


Figure 1.45

1.9.1.2.1 Note

$G_{\text{eff}}(\Omega)$ is LTI if and only if the following two conditions are satisfied:

1. $G(\omega)$ is LTI (in DT).
2. $X_c(T)$ is bandlimited and sampling rate equal to or greater than Nyquist. For example, if we had a simple pulse described by

$$X_c(t) = u(t - T_0) - u(t - T_1)$$

where $T_1 > T_0$. If the sampling period $T > T_1 - T_0$, then some samples might "miss" the pulse while others might not be "missed." This is what we term **time-varying behavior**.

Example 1.10

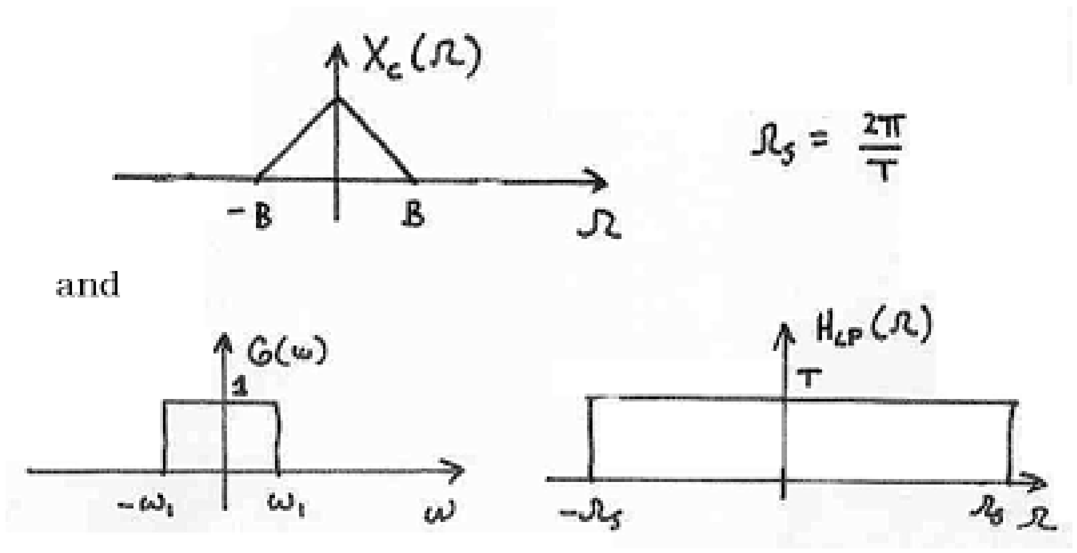


Figure 1.46

If $\frac{2\pi}{T} > 2B$ and $\omega_1 < BT$, determine and sketch $Y_c(\Omega)$ using Figure 1.46.

1.9.2 Application: 60Hz Noise Removal

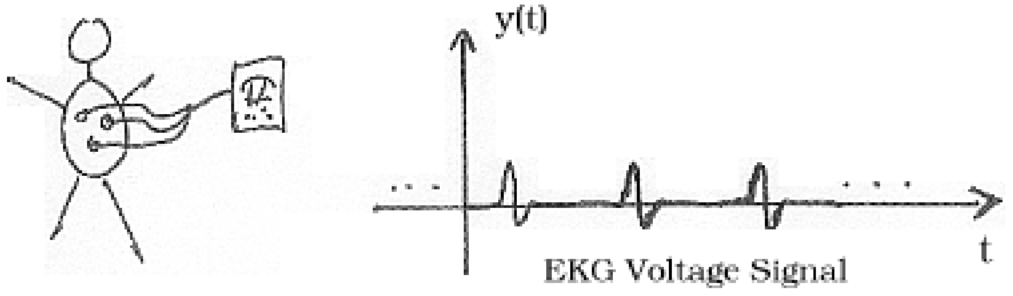


Figure 1.47

Unfortunately, in real-world situations electrodes also pick up ambient 60 Hz signals from lights, computers, *etc.*. In fact, usually this "60 Hz noise" is much greater in amplitude than the EKG signal shown in Figure 1.47. Figure 1.48 shows the EKG signal; it is barely noticeable as it has become overwhelmed by noise.

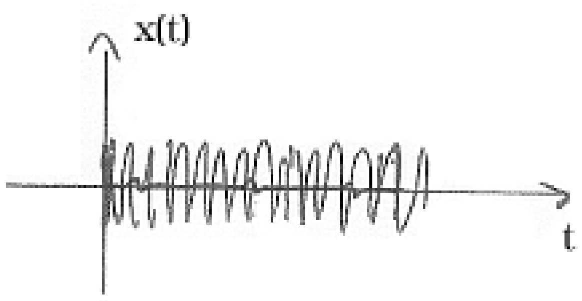


Figure 1.48: Our EKG signal, $y(t)$, is overwhelmed by noise.

1.9.2.1 DSP Solution

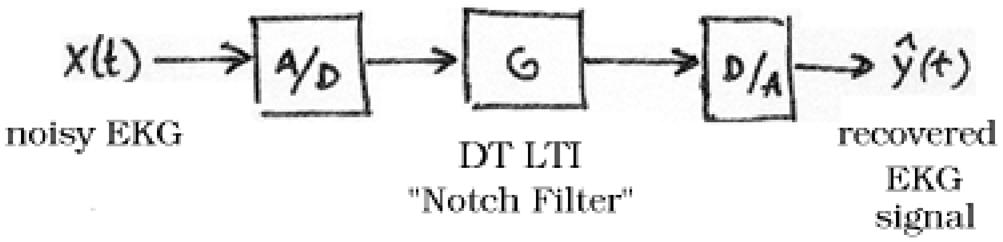


Figure 1.49

Image not finished

Figure 1.50

1.9.2.2 Sampling Period/Rate

First we must note that $|Y(\Omega)|$ is **bandlimited** to ± 60 Hz. Therefore, the minimum rate should be 120 Hz. In order to get the best results we should set

$$f_s = 240\text{Hz}$$

.

$$\Omega_s = 2\pi \times \left(240 \frac{\text{rad}}{s}\right)$$

Figure 1.51

1.9.2.3 Digital Filter

Therefore, we want to design a digital filter that will remove the 60Hz component and preserve the rest.

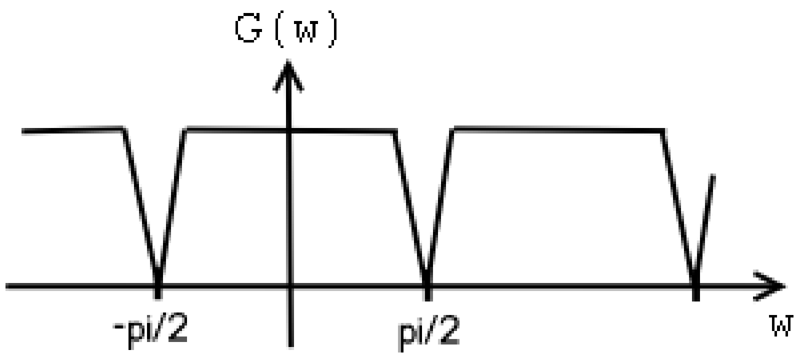


Figure 1.52

1.10 Sampling CT Signals: A Frequency Domain Perspective[17]

1.10.1 Understanding Sampling in the Frequency Domain

We want to relate $x_c(t)$ directly to $x[n]$. Compute the CTFT of

$$x_s(t) = \sum_{n=-\infty}^{\infty} x_c(nT)\,\delta(t - nT)$$

[17]This content is available online at <http://cnx.org/content/m10994/2.2/>.

$$\begin{aligned} X_s(\Omega) &= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x_c(nT)\,\delta(t-nT)\,e^{(-j)\Omega t}dt \\ &= \sum_{n=-\infty}^{\infty} x_c(nT) \int_{-\infty}^{\infty} \delta(t-nT)\,e^{(-j)\Omega t}dt \\ &= \sum_{n=-\infty}^{\infty} x[n]\,e^{(-j)\Omega nT} \\ &= \sum_{n=-\infty}^{\infty} x[n]\,e^{(-j)\omega n} \\ &= X(\omega) \end{aligned} \quad (1.44)$$

where $\omega \equiv \Omega T$ and $X(\omega)$ is the DTFT of $x[n]$.

NOTE:

$$X_s(\Omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c(\Omega - k\Omega_s)$$

$$\begin{aligned} X(\omega) &= \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c(\Omega - k\Omega_s) \\ &= \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\omega - 2\pi k}{T}\right) \end{aligned} \quad (1.45)$$

where this last part is 2π-periodic.

1.10.1.1 Sampling

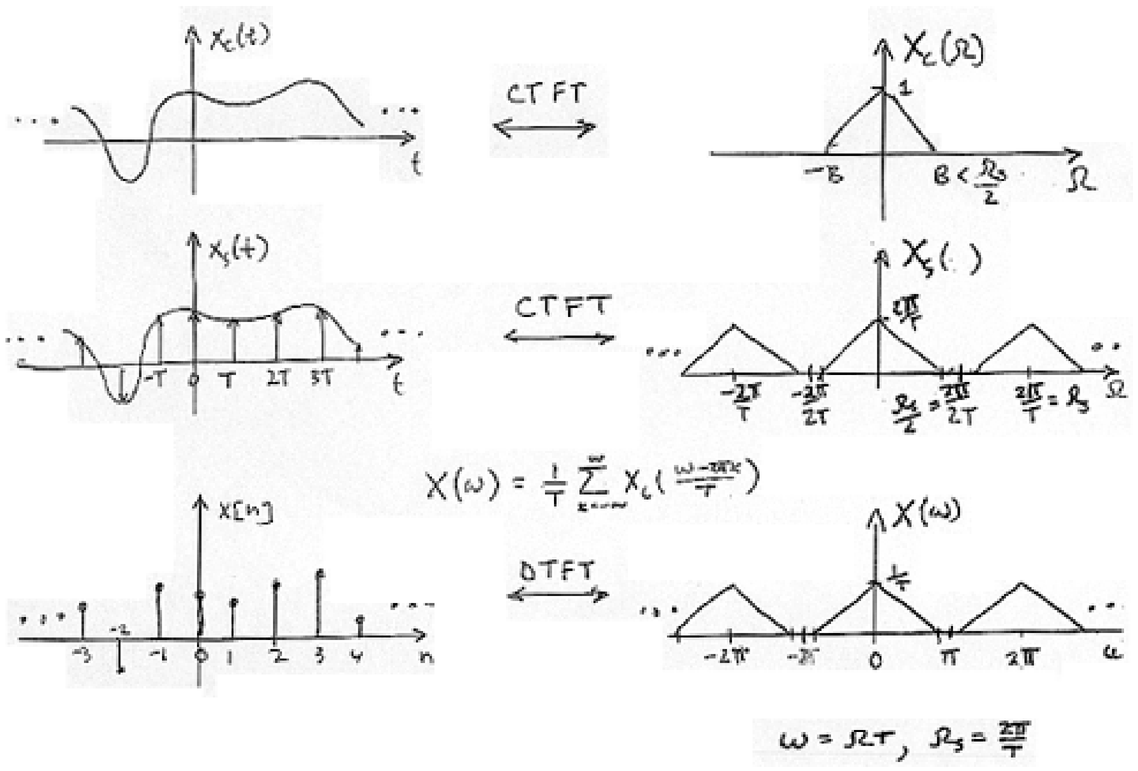

$x_c(t)$
CT FT
$X_c(\Omega)$
1
$-B$
$B < \frac{\Omega_s}{2}$
$x_s(t)$
$-T$ 0 T $2T$ $3T$ t
CTFT
$X_s(\;)$
$\frac{1}{T}$
$-\frac{2\pi}{T}$ $-\frac{2\pi}{2T}$ $\frac{\Omega_s}{2} = \frac{2\pi}{2T}$ $\frac{2\pi}{T} = \Omega_s$
$X(\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c\left(\frac{\omega - 2\pi k}{T}\right)$
$x[n]$
-3 -2 -1 0 1 2 3 4 n
DTFT
$X(\omega)$
$\frac{1}{T}$
-2π $-\pi$ 0 π 2π ω
$\omega = \Omega T, \; \Omega_s = \frac{2\pi}{T}$

Figure 1.53

Example 1.11: Speech
Speech is intelligible if bandlimited by a CT lowpass filter to the band ±4 kHz. We can sample speech as slowly as _____?

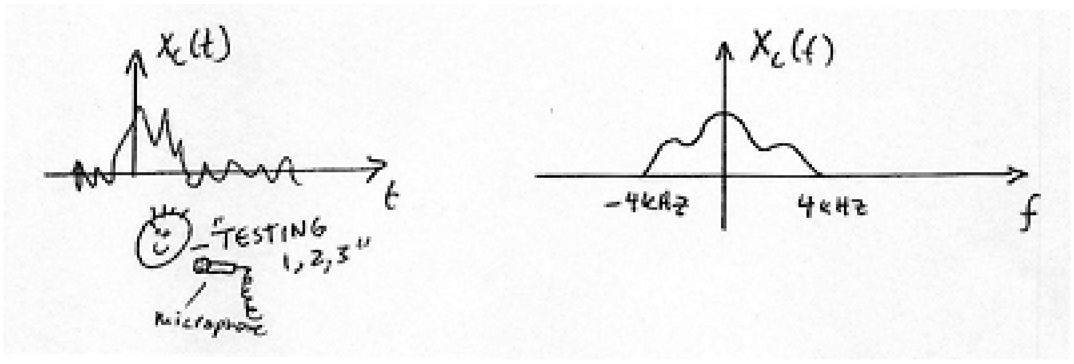
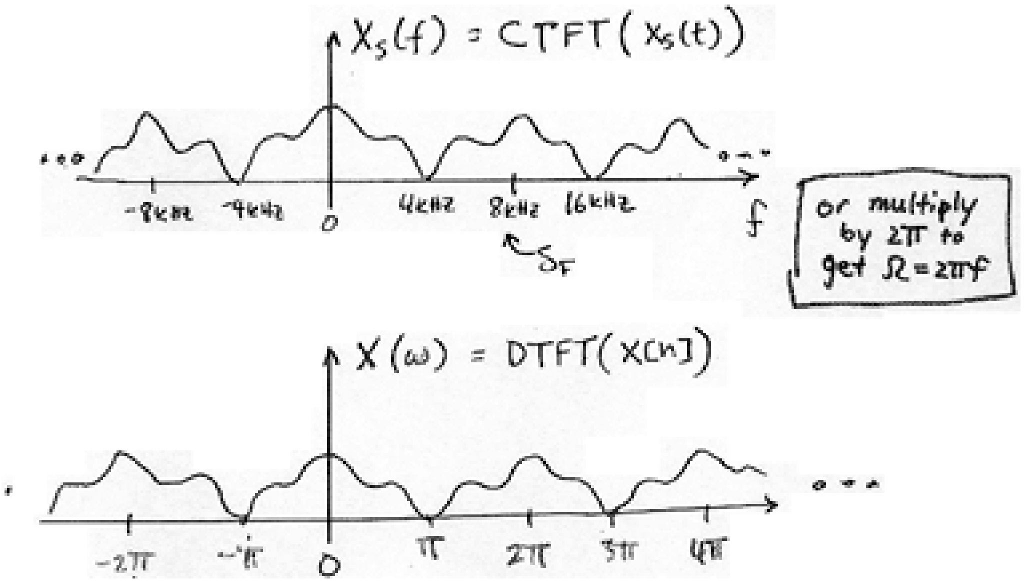


Figure 1.54



$X_s(f)$ = CTFT($x_s(t)$)
-8kHz
-4kHz
0
4kHz
8kHz
16kHz
f
S_F
or multiply by 2π to get $\Omega = 2\pi f$
$X(\omega)$ = DTFT($x[n]$)
-2π
$-\pi$
0
π
2π
3π
4π

Figure 1.55: Note that there is no mention of T or Ω_s!

1.10.2 Relating x[n] to sampled x(t)

Recall the following equality:

$$x_s(t) = \sum_{nn} x(nT)\,\delta(t - nT)$$

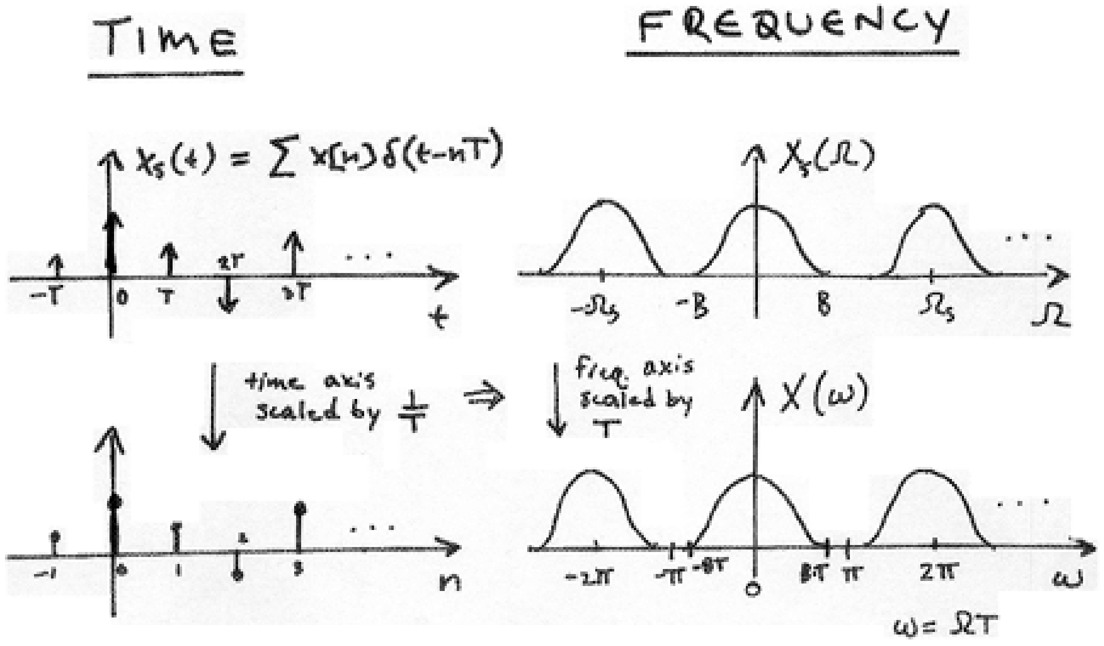


Figure 1.56

Recall the CTFT relation:

$$x(\alpha t) \leftrightarrow \frac{1}{\alpha} X\left(\frac{\Omega}{\alpha}\right) \tag{1.46}$$

where α is a scaling of time and $\frac{1}{\alpha}$ is a scaling in frequency.

$$X_s(\Omega) \equiv X(\Omega T) \tag{1.47}$$

1.11 Filtering with the DFT[18]

1.11.1 Introduction

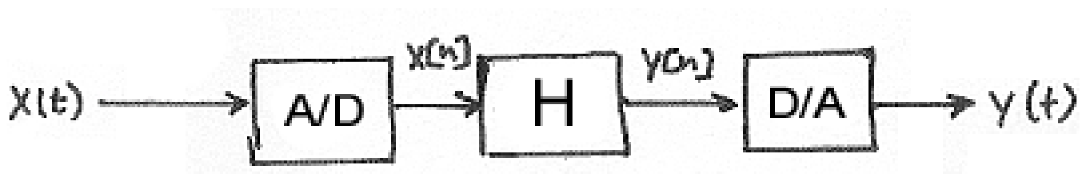


Figure 1.57

$$\begin{aligned} y[n] &= x[n] * h[n] \\ &= \sum_{k=-\infty}^{\infty} x[k]\, h[n-k] \end{aligned} \quad (1.48)$$

$$Y(\omega) = X(\omega)\, H(\omega) \quad (1.49)$$

Assume that $H(\omega)$ is specified.

Exercise 1.11.1 ***(Solution on p. 59.)***

How can we implement $X(\omega)\, H(\omega)$ in a computer?

Recall that the DFT treats N-point sequences as if they are periodically extended (Figure 1.58):

[18]This content is available online at <http://cnx.org/content/m11022/2.3/>.

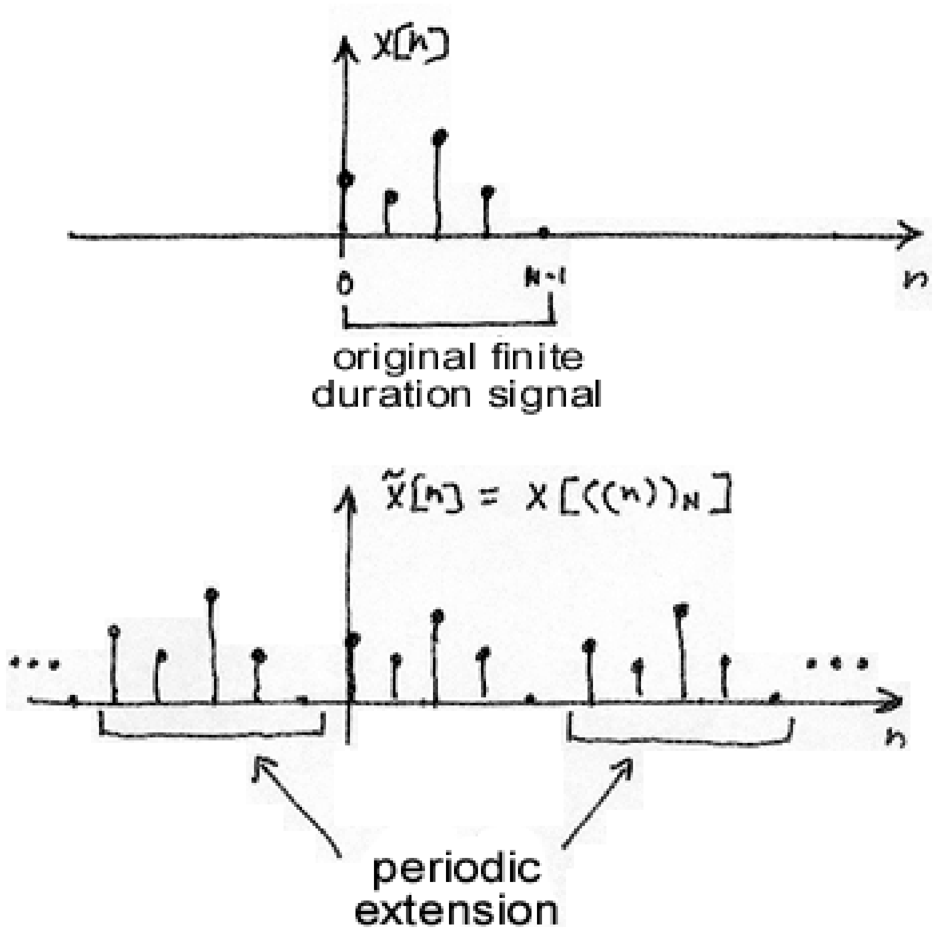


Figure 1.58

1.11.2 Compute IDFT of Y[k]

$$\begin{aligned}\tilde{y}[n] &= \frac{1}{N}\sum_{k=0}^{N-1} Y[k]\, e^{j2\pi\frac{k}{N}n} \\ &= \frac{1}{N}\sum_{k=0}^{N-1} X[k]\, H[k]\, e^{j2\pi\frac{k}{N}n} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\sum_{m=0}^{N-1} x[m]\, e^{-\left(j2\pi\frac{k}{N}m\right)} H[k]\, e^{j2\pi\frac{k}{N}n} \\ &= \sum_{m=0}^{N-1} x[m]\left(\frac{1}{N}\sum_{k=0}^{N-1} H[k]\, e^{j2\pi\frac{k}{N}(n-m)}\right) \\ &= \sum_{m=0}^{N-1} x[m]\, h[((n-m))_N]\end{aligned} \quad (1.50)$$

And the IDFT periodically extends $h[n]$:

$$\tilde{h}[n-m] = h[((n-m))_N]$$

This computes as shown in Figure 1.59:

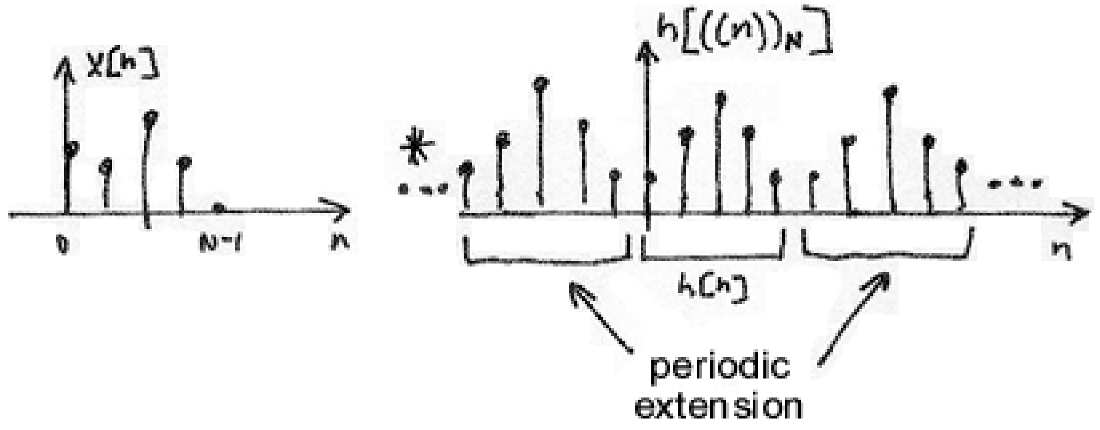


Figure 1.59

$$\tilde{y}[n] = \sum_{m=0}^{N-1} x[m]\, h[((n-m))_N] \tag{1.51}$$

is called **circular convolution** and is denoted by Figure 1.60.

$$\tilde{y}[n] = x[n] \; Ⓝ \; h[n]$$

Figure 1.60: The above symbol for the circular convolution is for an N-periodic extension.

1.11.2.1 DFT Pair

$$x[n] \; Ⓝ \; h[n] \xleftrightarrow{\text{DFT}} (X[k])(H[k])$$

Figure 1.61

Note that in general:

$$x[n] \; \textcircled{N} \; h[n] \neq x[n] * h[n]$$

Figure 1.62

Example 1.12: Regular vs. Circular Convolution
To begin with, we are given the following two length-3 signals:

$$x[n] = \{1, 2, 3\}$$

$$h[n] = \{1, 0, 2\}$$

We can zero-pad these signals so that we have the following discrete sequences:

$$x[n] = \{\dots, 0, 1, 2, 3, 0, \dots\}$$

$$h[n] = \{\dots, 0, 1, 0, 2, 0, \dots\}$$

where $x[0] = 1$ and $h[0] = 1$.

- Regular Convolution:

$$y[n] = \sum_{m=0}^{2} x[m]\, h[n-m] \tag{1.52}$$

  Using the above convolution formula (refer to the link if you need a review of convolution[19]), we can calculate the resulting value for $y[0]$ to $y[4]$. Recall that because we have two length-3 signals, our convolved signal will be length-5.

  · $n = 0$

$$\{\dots, 0, 0, 0, 1, 2, 3, 0, \dots\}$$

$$\{\dots, 0, 2, 0, 1, 0, 0, 0, \dots\}$$

$$\begin{aligned} y[0] &= 1 \times 1 + 2 \times 0 + 3 \times 0 \\ &= 1 \end{aligned} \tag{1.53}$$

  · $n = 1$

$$\{\dots, 0, 0, 1, 2, 3, 0, \dots\}$$

$$\{\dots, 0, 2, 0, 1, 0, 0, \dots\}$$

$$\begin{aligned} y[1] &= 1 \times 0 + 2 \times 1 + 3 \times 0 \\ &= 2 \end{aligned} \tag{1.54}$$

[19]"Discrete Time Convolution" <http://cnx.org/content/m10087/latest/>

- $n = 2$

$$\{\dots, 0, 1, 2, 3, 0, \dots\}$$

$$\{\dots, 0, 2, 0, 1, 0, \dots\}$$

$$\begin{aligned} y[2] &= 1 \times 2 + 2 \times 0 + 3 \times 1 \\ &= 5 \end{aligned} \quad (1.55)$$

- $n = 3$

$$y[3] = 4 \quad (1.56)$$

- $n = 4$

$$y[4] = 6 \quad (1.57)$$

Regular Convolution Result

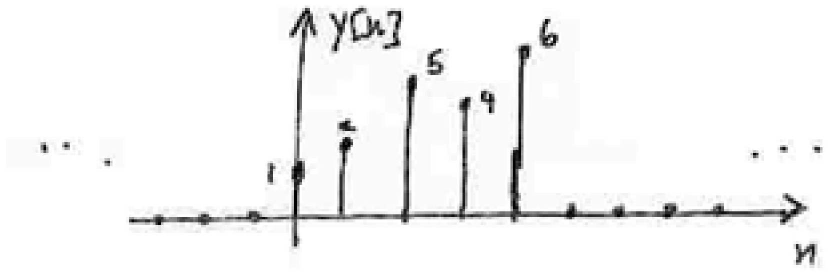


Figure 1.63: Result is finite duration, not periodic!

- Circular Convolution:

$$\tilde{y}[n] = \sum_{m=0}^{2} x[m]\, h[((n-m))_N] \quad (1.58)$$

And now with circular convolution our $h[n]$ changes and becomes a periodically extended signal:

$$h[((n))_N] = \{\dots, 1, 0, 2, 1, 0, 2, 1, 0, 2, \dots\} \quad (1.59)$$

- $n = 0$

$$\{\dots, 0, 0, 0, 1, 2, 3, 0, \dots\}$$

$$\{\dots, 1, 2, 0, 1, 2, 0, 1, \dots\}$$

$$\begin{aligned} \tilde{y}[0] &= 1 \times 1 + 2 \times 2 + 3 \times 0 \\ &= 5 \end{aligned} \quad (1.60)$$

- $n = 1$

$$\{\dots, 0, 0, 0, 1, 2, 3, 0, \dots\}$$

$$\{\dots, 0, 1, 2, 0, 1, 2, 0, \dots\}$$

$$\begin{aligned} \widetilde{y}[1] &= 1 \times 1 + 2 \times 1 + 3 \times 2 \\ &= 8 \end{aligned} \qquad (1.61)$$

- $n = 2$

$$\widetilde{y}[2] = 5 \qquad (1.62)$$

- $n = 3$

$$\widetilde{y}[3] = 5 \qquad (1.63)$$

- $n = 4$

$$\widetilde{y}[4] = 8 \qquad (1.64)$$

Circular Convolution Result

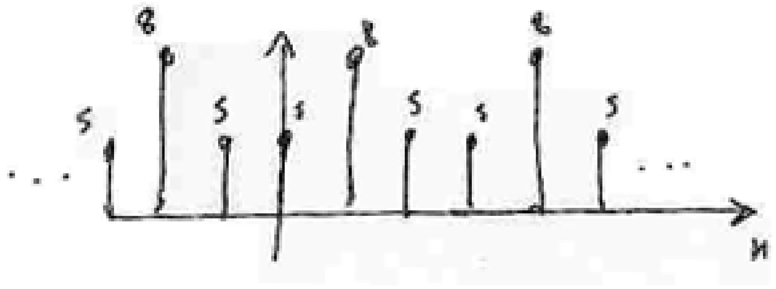


Figure 1.64: Result is 3-periodic.

Figure 1.65 (Circular Convolution from Regular) illustrates the relationship between circular convolution and regular convolution using the previous two figures:

Circular Convolution from Regular

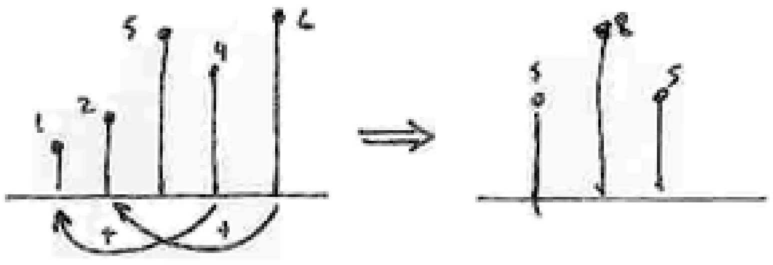


Figure 1.65: The left plot (the circular convolution results) has a "wrap-around" effect due to periodic extension.

1.11.2.2 Regular Convolution from Periodic Convolution

1. "Zero-pad" $x[n]$ and $h[n]$ to avoid the overlap (wrap-around) effect. We will zero-pad the two signals to a length-5 signal (5 being the duration of the regular convolution result):

$$x[n] = \{1, 2, 3, 0, 0\}$$

$$h[n] = \{1, 0, 2, 0, 0\}$$

2. Now take the DFTs of the zero-padded signals:

$$\begin{array}{rcl} \tilde{y}[n] & = & \frac{1}{N}\sum_{k=0}^{4} X[k]\,H[k]\,e^{j2\pi\frac{k}{5}n} \\ & = & \sum_{m=0}^{4} x[m]\,h[((n-m))_5] \end{array} \tag{1.65}$$

Now we can plot this result (Figure 1.66):

Figure 1.66: The sequence from 0 to 4 (the underlined part of the sequence) is the regular convolution result. From this illustration we can see that it is 5-periodic!

NOTE: We can compute the regular convolution result of a convolution of an M-point signal $x[n]$ with an N-point signal $h[n]$ by padding each signal with zeros to obtain two $M+N-1$ length sequences and computing the circular convolution (or equivalently computing the IDFT of $H[k]\,X[k]$, the product of the DFTs of the zero-padded signals) (Figure 1.67).

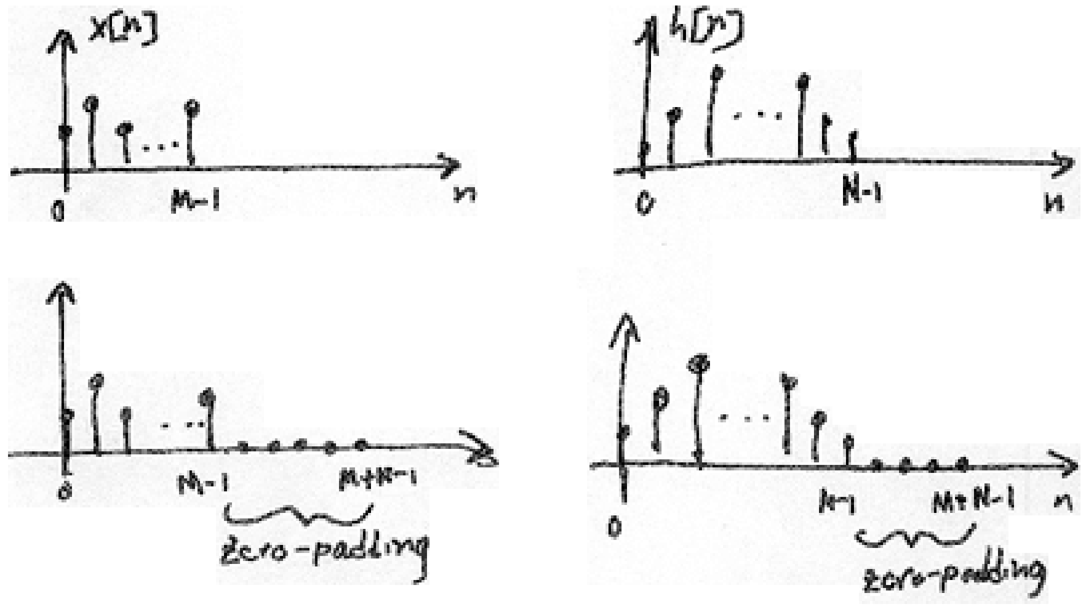


Figure 1.67: Note that the lower two images are simply the top images that have been zero-padded.

1.11.3 DSP System

No Image.

Figure 1.68: The system has a length N impulse response, $h[n]$

1. Sample finite duration continuous-time input $x(t)$ to get $x[n]$ where $n = \{0, \dots, M-1\}$.
2. Zero-pad $x[n]$ and $h[n]$ to length $M + N - 1$.
3. Compute DFTs $X[k]$ and $H[k]$
4. Compute IDFTs of $X[k]H[k]$
$$y[n] = \tilde{y}[n]$$
where $n = \{0, \dots, M + N - 1\}$.
5. Reconstruct $y(t)$

1.12 Ideal Reconstruction of Sampled Signals[20]

1.12.1 Reconstruction of Sampled Signals

How do we go from $x[n]$ to CT (Figure 1.69)?

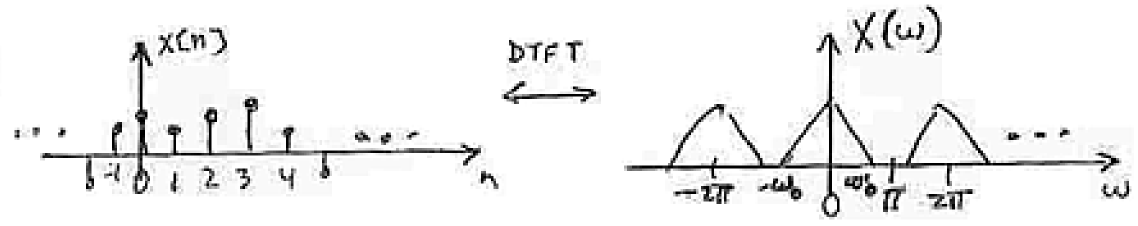


Figure 1.69

1.12.1.1 Step 1

Place $x[n]$ into CT on an impulse train $s(t)$ (Figure 1.70).

$$x_s(t) = \sum_{n=-\infty}^{\infty} x[n]\,\delta(t - nT) \tag{1.66}$$

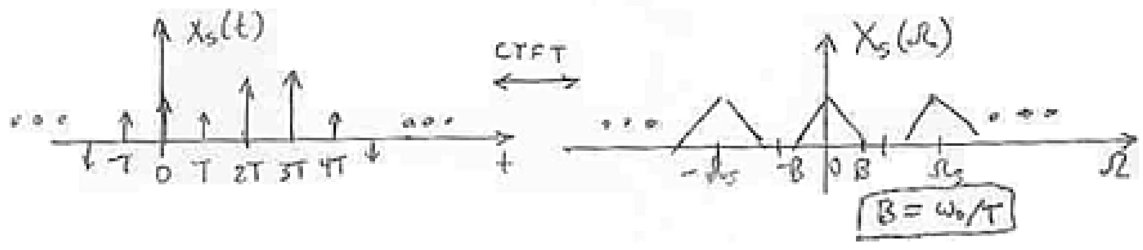


Figure 1.70

1.12.1.2 Step 2

Pass $x_s(t)$ through an idea lowpass filter $H_{\mathrm{LP}}(\Omega)$ (Figure 1.71).

[20]This content is available online at <http://cnx.org/content/m11044/2.3/>.

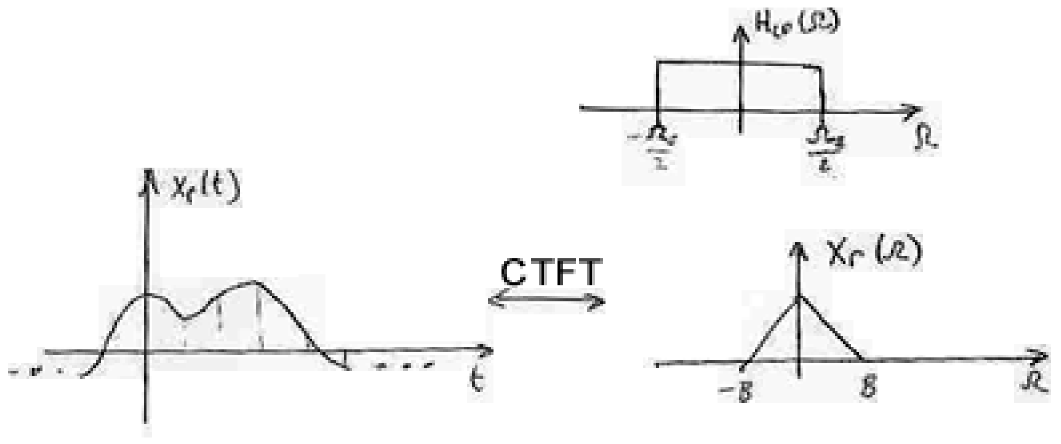


Figure 1.71

If we had no aliasing then $x_r[t] = x_c(t)$, where $x[n] = x_c(nT)$.

1.12.2 Ideal Reconstruction System

Figure 1.72

In Frequency Domain:

1.
$$X_s(\Omega) = X(\Omega T)$$
where $X(\Omega T)$ is the DTFT of $x[n]$ at digital frequency $\omega = \Omega T$.

2.
$$X_r(\Omega) = H_{\text{LP}}(\Omega) X_s(\Omega)$$
NOTE:
$$X_r(\Omega) = H_{\text{LP}}(\Omega) X(\Omega T)$$

In Time Domain:

1.
$$x_s(t) = \sum_{n=-\infty}^{\infty} x[n]\,\delta(t - nT)$$

2.

$$x_r(t) = \sum_{n=-\infty}^{\infty} x[n]\,\delta(t - nT) * h_{\text{LP}}(t)$$

$$h_{\text{LP}}(t) = \text{sinc}\left(\frac{\pi}{T}t\right)$$

NOTE:

$$x_r(t) = \sum_{n=-\infty}^{\infty} x[n]\,\text{sinc}\left(\frac{\pi}{T}(t - nT)\right)$$

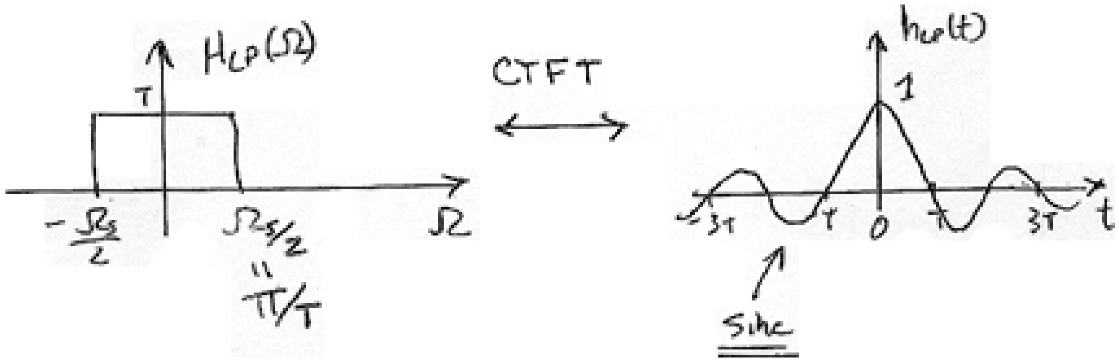


Figure 1.73

$$\begin{aligned} h_{\text{LP}}(t) &= \text{sinc}\left(\frac{\pi}{T}t\right) \\ &= \frac{\sin\left(\frac{\pi}{T}t\right)}{\left(\frac{\pi}{T}t\right)} \end{aligned} \quad (1.67)$$

$h_{\text{LP}}(t)$ "interpolates" the values of $x[n]$ to generate $x_r(t)$ (Figure 1.74).

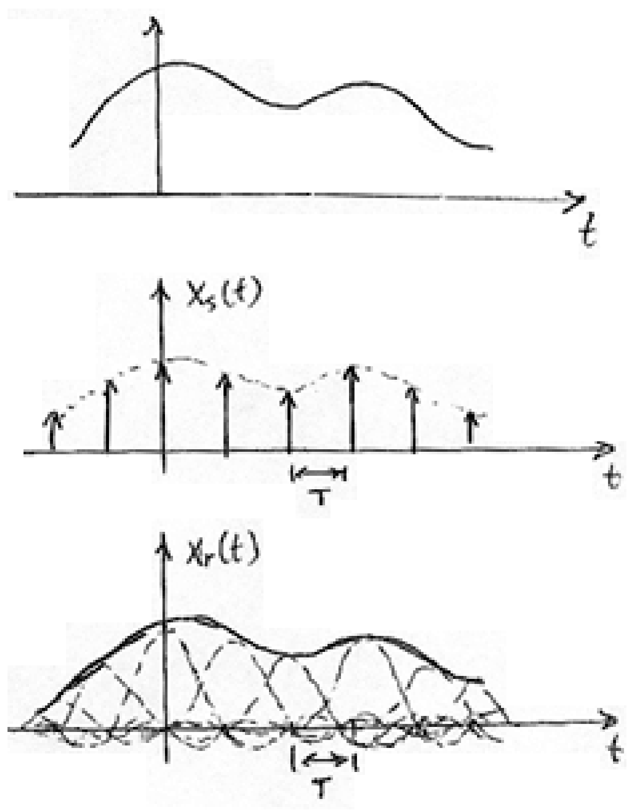


Figure 1.74

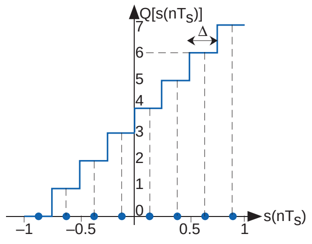
Sinc Interpolator

$$x_r(t) = \sum_{n=-\infty}^{\infty} x[n] \frac{\sin\left(\frac{\pi}{T}(t - nT)\right)}{\frac{\pi}{T}(t - nT)} \tag{1.68}$$

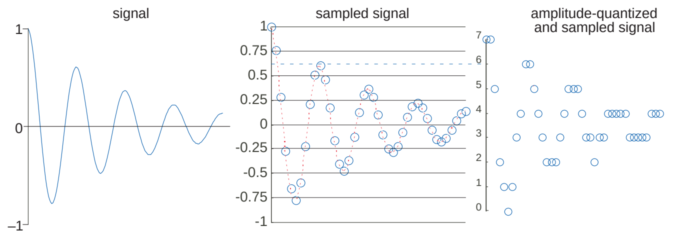
1.13 Amplitude Quantization[21]

The Sampling Theorem says that if we sample a bandlimited signal $s(t)$ fast enough, it can be recovered without error from its samples $s(nT_s)$, $n \in \{\dots, -1, 0, 1, \dots\}$. Sampling is only the first phase of acquiring data into a computer: Computational processing further requires that the samples be **quantized**: analog values are converted into digital[22] form. In short, we will have performed **analog-to-digital (A/D) conversion**.

[21]This content is available online at <http://cnx.org/content/m0051/2.23/>.
[22]"Signals Represent Information": Section Digital Signals <http://cnx.org/content/m0001/latest/#digital>



(a)



(b)

Figure 1.75: A three-bit A/D converter assigns voltage in the range $[-1, 1]$ to one of eight integers between 0 and 7. For example, all inputs having values lying between 0.5 and 0.75 are assigned the integer value six and, upon conversion back to an analog value, they all become 0.625. The width of a single quantization interval Δ equals $\frac{2}{2^B}$. The bottom panel shows a signal going through the analog-to-digital, where B is the number of bits used in the A/D conversion process (3 in the case depicted here). First it is sampled, then amplitude-quantized to three bits. Note how the sampled signal waveform becomes distorted after amplitude quantization. For example the two signal values between 0.5 and 0.75 become 0.625. This distortion is irreversible; it can be reduced (but not eliminated) by using more bits in the A/D converter.

A phenomenon reminiscent of the errors incurred in representing numbers on a computer prevents signal amplitudes from being converted with no error into a binary number representation. In analog-to-digital conversion, the signal is assumed to lie within a predefined range. Assuming we can scale the signal without affecting the information it expresses, we'll define this range to be $[-1, 1]$. Furthermore, the A/D converter assigns amplitude values in this range to a set of integers. A B-bit converter produces one of the integers $\{0, 1, \dots, 2^B - 1\}$ for each sampled input. Figure 1.75 shows how a three-bit A/D converter assigns input values to the integers. We define a **quantization interval** to be the range of values assigned to the same integer. Thus, for our example three-bit A/D converter, the quantization interval Δ is 0.25; in general, it is $\frac{2}{2^B}$.

Exercise 1.13.1 ***(Solution on p. 59.)***

Recalling the plot of average daily highs in this frequency domain problem[23], why is this plot so jagged? Interpret this effect in terms of analog-to-digital conversion.

Because values lying anywhere within a quantization interval are assigned the same value for computer processing, **the original amplitude value cannot be recovered without error**. Typically, the D/A converter, the device that converts integers to amplitudes, assigns an amplitude equal to the value lying halfway in the quantization interval. The integer 6 would be assigned to the amplitude 0.625 in this scheme.

[23]"Frequency Domain Problems", Problem 5 <http://cnx.org/content/m10350/latest/#i4>

The error introduced by converting a signal from analog to digital form by sampling and amplitude quantization then back again would be half the quantization interval for each amplitude value. Thus, the so-called **A/D** error equals half the width of a quantization interval: $\frac{1}{2^B}$. As we have fixed the input-amplitude range, the more bits available in the A/D converter, the smaller the quantization error.

To analyze the amplitude quantization error more deeply, we need to compute the **signal-to-noise** ratio, which equals the ratio of the signal power and the quantization error power. Assuming the signal is a sinusoid, the signal power is the square of the rms amplitude: $\text{power}(s) = \left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2}$. The illustration (Figure 1.76) details a single quantization interval.

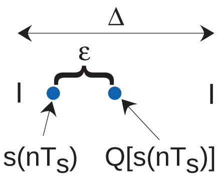


Figure 1.76: A single quantization interval is shown, along with a typical signal's value before amplitude quantization $s(nT_s)$ and after $Q(s(nT_s))$. ϵ denotes the error thus incurred.

Its width is Δ and the quantization error is denoted by ϵ. To find the power in the quantization error, we note that no matter into which quantization interval the signal's value falls, the error will have the same characteristics. To calculate the rms value, we must square the error and average it over the interval.

$$\begin{aligned} \text{rms}(\epsilon) &= \sqrt{\frac{1}{\Delta}\int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} \epsilon^2 d\epsilon} \\ &= \left(\frac{\Delta^2}{12}\right)^{\frac{1}{2}} \end{aligned} \tag{1.69}$$

Since the quantization interval width for a B-bit converter equals $\frac{2}{2^B} = 2^{-(B-1)}$, we find that the signal-to-noise ratio for the analog-to-digital conversion process equals

$$\text{SNR} = \frac{\frac{1}{2}}{\frac{2^{-(2(B-1))}}{12}} = \frac{3}{2}2^{2B} = 6B + 10\text{log}1.5\text{dB} \tag{1.70}$$

Thus, every bit increase in the A/D converter yields a 6 dB increase in the signal-to-noise ratio. The constant term 10log1.5 equals 1.76.

Exercise 1.13.2 ***(Solution on p. 59.)***

This derivation assumed the signal's amplitude lay in the range $[-1, 1]$. What would the amplitude quantization signal-to-noise ratio be if it lay in the range $[-A, A]$?

Exercise 1.13.3 ***(Solution on p. 59.)***

How many bits would be required in the A/D converter to ensure that the maximum amplitude quantization error was less than 60 db smaller than the signal's peak value?

Exercise 1.13.4 ***(Solution on p. 59.)***

Music on a CD is stored to 16-bit accuracy. To what signal-to-noise ratio does this correspond?

Once we have acquired signals with an A/D converter, we can process them using digital hardware or software. It can be shown that if the computer processing is linear, the result of sampling, computer processing, and unsampling is equivalent to some analog linear system. Why go to all the bother if the same function can be accomplished using analog techniques? Knowing when digital processing excels and when it does not is an important issue.

1.14 Classic Fourier Series[24]

The classic Fourier series as derived originally expressed a periodic signal (period T) in terms of harmonically related sines and cosines.

$$s(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi kt}{T}\right) + \sum_{k=1}^{\infty} b_k \sin\left(\frac{2\pi kt}{T}\right) \tag{1.71}$$

The complex Fourier series and the sine-cosine series are identical, each representing a signal's spectrum. The **Fourier coefficients**, a_k and b_k, express the real and imaginary parts respectively of the spectrum while the coefficients c_k of the complex Fourier series express the spectrum as a magnitude and phase. Equating the classic Fourier series (1.71) to the complex Fourier series[25], an extra factor of two and complex conjugate become necessary to relate the Fourier coefficients in each.

$$c_k = \frac{1}{2}(a_k - jb_k)$$

Exercise 1.14.1 ***(Solution on p. 59.)***

Derive this relationship between the coefficients of the two Fourier series.

Just as with the complex Fourier series, we can find the Fourier coefficients using the **orthogonality** properties of sinusoids. Note that the cosine and sine of harmonically related frequencies, even the **same** frequency, are orthogonal.

$$\int_0^T \sin\left(\frac{2\pi kt}{T}\right)\cos\left(\frac{2\pi lt}{T}\right)dt = 0 \quad , \quad k \in \mathbb{Z} \quad l \in \mathbb{Z} \tag{1.72}$$

$$\int_0^T \sin\left(\frac{2\pi kt}{T}\right)\sin\left(\frac{2\pi lt}{T}\right)dt = \begin{cases} \frac{T}{2} \text{ if } (k = l) \;\; and \;\; (k \neq 0) \;\; and \;\; (l \neq 0) \\ 0 \text{ if } (k \neq l) \;\; or \;\; (k = 0 = l) \end{cases}$$

$$\int_0^T \cos\left(\frac{2\pi kt}{T}\right)\cos\left(\frac{2\pi lt}{T}\right)dt = \begin{cases} \frac{T}{2} \text{ if } (k = l) \;\; and \;\; (k \neq 0) \;\; and \;\; (l \neq 0) \\ T \text{ if } k = 0 = l \\ 0 \text{ if } k \neq l \end{cases}$$

These orthogonality relations follow from the following important trigonometric identities.

$$\begin{aligned} \sin(\alpha)\sin(\beta) &= \tfrac{1}{2}(\cos(\alpha-\beta) - \cos(\alpha+\beta)) \\ \cos(\alpha)\cos(\beta) &= \tfrac{1}{2}(\cos(\alpha+\beta) + \cos(\alpha-\beta)) \\ \sin(\alpha)\cos(\beta) &= \tfrac{1}{2}(\sin(\alpha+\beta) + \sin(\alpha-\beta)) \end{aligned} \tag{1.73}$$

These identities allow you to substitute a sum of sines and/or cosines for a product of them. Each term in the sum can be integrating by noticing one of two important properties of sinusoids.

- The integral of a sinusoid over an **integer** number of periods equals zero.
- The integral of the **square** of a unit-amplitude sinusoid over a period T equals $\frac{T}{2}$.

[24]This content is available online at <http://cnx.org/content/m0039/2.23/>.
[25]"Complex Fourier Series", (1) <http://cnx.org/content/m0042/latest/#complexfourierseries>

To use these, let's, for example, multiply the Fourier series for a signal by the cosine of the l^{th} harmonic $\cos\left(\frac{2\pi lt}{T}\right)$ and integrate. The idea is that, because integration is linear, the integration will sift out all but the term involving a_l.

$$\int_0^T s(t)\cos\left(\frac{2\pi lt}{T}\right)dt = \int_0^T a_0\cos\left(\frac{2\pi lt}{T}\right)dt + \sum_{k=1}^{\infty} a_k \int_0^T \cos\left(\frac{2\pi kt}{T}\right)\cos\left(\frac{2\pi lt}{T}\right)dt + \sum_{k=1}^{\infty} b_k \int_0^T \sin\left(\frac{2\pi kt}{T}\right)\cos\left(\frac{2\pi lt}{T}\right)dt \quad (1.74)$$

The first and third terms are zero; in the second, the only non-zero term in the sum results when the indices k and l are equal (but not zero), in which case we obtain $\frac{a_l T}{2}$. If $k = 0 = l$, we obtain $a_0 T$. Consequently,

$$a_l = \frac{2}{T}\int_0^T s(t)\cos\left(\frac{2\pi lt}{T}\right)dt \ , \ l \neq 0$$

All of the Fourier coefficients can be found similarly.

$$\begin{aligned} a_0 &= \tfrac{1}{T}\int_0^T s(t)\,dt \\ a_k &= \tfrac{2}{T}\int_0^T s(t)\cos\left(\tfrac{2\pi kt}{T}\right)dt \ , \ k \neq 0 \\ b_k &= \tfrac{2}{T}\int_0^T s(t)\sin\left(\tfrac{2\pi kt}{T}\right)dt \end{aligned} \quad (1.75)$$

Exercise 1.14.2 ***(Solution on p. 60.)***

The expression for a_0 is referred to as the **average value** of $s(t)$. Why?

Exercise 1.14.3 ***(Solution on p. 60.)***

What is the Fourier series for a unit-amplitude square wave?

Example 1.13

Let's find the Fourier series representation for the half-wave rectified sinusoid.

$$s(t) = \begin{cases} \sin\left(\frac{2\pi t}{T}\right) & \text{if } 0 \leq t < \frac{T}{2} \\ 0 & \text{if } \frac{T}{2} \leq t < T \end{cases} \quad (1.76)$$

Begin with the sine terms in the series; to find b_k we must calculate the integral

$$b_k = \frac{2}{T}\int_0^{\frac{T}{2}} \sin\left(\frac{2\pi t}{T}\right)\sin\left(\frac{2\pi kt}{T}\right)dt \quad (1.77)$$

Using our trigonometric identities turns our integral of a product of sinusoids into a sum of integrals of individual sinusoids, which are much easier to evaluate.

$$\begin{aligned} \int_0^{\frac{T}{2}} \sin\left(\tfrac{2\pi t}{T}\right)\sin\left(\tfrac{2\pi kt}{T}\right)dt &= \tfrac{1}{2}\int_0^{\frac{T}{2}} \cos\left(\tfrac{2\pi(k-1)t}{T}\right) - \cos\left(\tfrac{2\pi(k+1)t}{T}\right)dt \\ &= \begin{cases} \frac{1}{2} & \text{if } k = 1 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (1.78)$$

Thus,

$$b_1 = \frac{1}{2}$$

$$b_2 = b_3 = \cdots = 0$$

On to the cosine terms. The average value, which corresponds to a_0, equals $\frac{1}{\pi}$. The remainder of the cosine coefficients are easy to find, but yield the complicated result

$$a_k = \begin{cases} -\left(\frac{2}{\pi}\frac{1}{k^2-1}\right) & \text{if } k \in \{2, 4, \dots\} \\ 0 \text{ if k odd} \end{cases} \quad (1.79)$$

Thus, the Fourier series for the half-wave rectified sinusoid has non-zero terms for the average, the fundamental, and the even harmonics.

Solutions to Exercises in Chapter 1

Solution to Exercise 1.3.1 (p. 9)
2D Circular Convolution

$$\begin{aligned} \tilde{g}[m,n] &= \text{IDFT}(H[k,l]F[k,l]) \\ &= \text{circular convolution in 2D} \end{aligned} \tag{1.80}$$

Solution to Exercise 1.6.1 (p. 17)
There are $2N-1$ non-zero output points and each will be computed using N complex mults and $N-1$ complex adds. Therefore,

$$\text{Total Cost} = (2N-1)(N+N-1) \simeq O\left(N^2\right)$$

Solution to Exercise 1.8.1 (p. 26)

$$x[n+N] = ???$$

Solution to Exercise 1.8.2 (p. 30)
$S[n]$ is N-periodic, so it has the following Fourier Series[26]:

$$\begin{aligned} c_k &= \frac{1}{N}\int_{-\frac{N}{2}}^{\frac{N}{2}} \delta[n]\, e^{(-j)2\pi\frac{k}{N}n}dn \\ &= \frac{1}{N} \end{aligned} \tag{1.81}$$

$$S[n] = \sum_{k=-\infty}^{\infty} e^{(-j)2\pi\frac{k}{N}n} \tag{1.82}$$

where the DTFT of the exponential in the above equation is equal to $\delta\left(\omega - \frac{2\pi k}{N}\right)$.
Solution to Exercise 1.11.1 (p. 42)
Discretize (sample) $X(\omega)$ and $H(\omega)$. In order to do this, we should take the DFTs of $x[n]$ and $h[n]$ to get $X[k]$ and $X[k]$. Then we will compute

$$\tilde{y}[n] = \text{IDFT}(X[k]H[k])$$

Does $\tilde{y}[n] = y[n]$?
Solution to Exercise 1.13.1 (p. 54)
The plotted temperatures were quantized to the nearest degree. Thus, the high temperature's amplitude was quantized as a form of A/D conversion.
Solution to Exercise 1.13.2 (p. 55)
The signal-to-noise ratio does not depend on the signal amplitude. With an A/D range of $[-A, A]$, the quantization interval $\Delta = \frac{2A}{2^B}$ and the signal's rms value (again assuming it is a sinusoid) is $\frac{A}{\sqrt{2}}$.
Solution to Exercise 1.13.3 (p. 55)
Solving $2^{-B} = .001$ results in $B = 10$ bits.
Solution to Exercise 1.13.4 (p. 55)
A 16-bit A/D converter yields a SNR of $6 \times 16 + 10\log 1.5 = 97.8$ dB.
Solution to Exercise 1.14.1 (p. 56)
Write the coefficients of the complex Fourier series in Cartesian form as $c_k = A_k + jB_k$ and substitute into the expression for the complex Fourier series.

$$\sum_{k=-\infty}^{\infty} c_k e^{j\frac{2\pi kt}{T}} = \sum_{k=-\infty}^{\infty} (A_k + jB_k)\, e^{j\frac{2\pi kt}{T}}$$

[26]"Fourier Series: Eigenfunction Approach" <http://cnx.org/content/m10496/latest/>

Simplifying each term in the sum using Euler's formula,

$$\begin{aligned}(A_k + jB_k)\, e^{j\frac{2\pi kt}{T}} &= (A_k + jB_k)\left(\cos\left(\tfrac{2\pi kt}{T}\right) + j\sin\left(\tfrac{2\pi kt}{T}\right)\right) \\ &= A_k\cos\left(\tfrac{2\pi kt}{T}\right) - B_k\sin\left(\tfrac{2\pi kt}{T}\right) + j\left(A_k\sin\left(\tfrac{2\pi kt}{T}\right) + B_k\cos\left(\tfrac{2\pi kt}{T}\right)\right)\end{aligned}$$

We now combine terms that have the same frequency index **in magnitude**. Because the signal is real-valued, the coefficients of the complex Fourier series have conjugate symmetry: $c_{-k} = {c_k}^*$ or $A_{-k} = A_k$ and $B_{-k} = -B_k$. After we add the positive-indexed and negative-indexed terms, each term in the Fourier series becomes $2A_k\cos\left(\frac{2\pi kt}{T}\right) - 2B_k\sin\left(\frac{2\pi kt}{T}\right)$. To obtain the classic Fourier series (1.71), we must have $2A_k = a_k$ and $2B_k = -b_k$.

Solution to Exercise 1.14.2 (p. 57)

The average of a set of numbers is the sum divided by the number of terms. Viewing signal integration as the limit of a Riemann sum, the integral corresponds to the average.

Solution to Exercise 1.14.3 (p. 57)

We found that the complex Fourier series coefficients are given by $c_k = \frac{2}{j\pi k}$. The coefficients are pure imaginary, which means $a_k = 0$. The coefficients of the sine terms are given by $b_k = -(2\mathrm{Im}(c_k))$ so that

$$b_k = \begin{cases} \frac{4}{\pi k} & \text{if } k \text{ odd} \\ 0 & \text{if } k \text{ even} \end{cases}$$

Thus, the Fourier series for the square wave is

$$\mathrm{sq}(t) = \sum_{k \in \{1,3,\dots\}} \frac{4}{\pi k}\sin\left(\frac{2\pi kt}{T}\right) \quad (1.83)$$

Chapter 2

Random Signals

2.1 Introduction to Random Signals and Processes[1]

Before now, you have probably dealt strictly with the theory behind signals and systems, as well as look at some the basic characteristics of signals[2] and systems[3]. In doing so you have developed an important foundation; however, most electrical engineers do not get to work in this type of fantasy world. In many cases the signals of interest are very complex due to the randomness of the world around them, which leaves them noisy and often corrupted. This often causes the information contained in the signal to be hidden and distorted. For this reason, it is important to understand these random signals and how to recover the necessary information.

2.1.1 Signals: Deterministic vs. Stochastic

For this study of signals and systems, we will divide signals into two groups: those that have a fixed behavior and those that change randomly. As most of you have probably already dealt with the first type, we will focus on introducing you to random signals. Also, note that we will be dealing strictly with discrete-time signals since they are the signals we deal with in DSP and most real-world computations, but these same ideas apply to continuous-time signals.

2.1.1.1 Deterministic Signals

Most introductions to signals and systems deal strictly with **deterministic signals**. Each value of these signals are fixed and can be determined by a mathematical expression, rule, or table. Because of this, future values of any deterministic signal can be calculated from past values. For this reason, these signals are relatively easy to analyze as they do not change, and we can make accurate assumptions about their past and future behavior.

[1]This content is available online at <http://cnx.org/content/m10649/2.2/>.
[2]"Signal Classifications and Properties" <http://cnx.org/content/m10057/latest/>
[3]"System Classifications and Properties" <http://cnx.org/content/m10084/latest/>

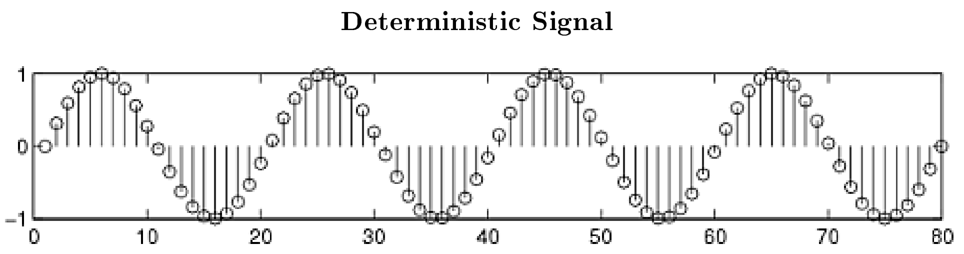


Figure 2.1: An example of a deterministic signal, the sine wave.

2.1.1.2 Stochastic Signals

Unlike deterministic signals, **stochastic signals**, or **random signals**, are not so nice. Random signals cannot be characterized by a simple, well-defined mathematical equation and their future values cannot be predicted. Rather, we must use probability and statistics to analyze their behavior. Also, because of their randomness, average values (Section 2.5) from a collection of signals are usually studied rather than analyzing one individual signal.

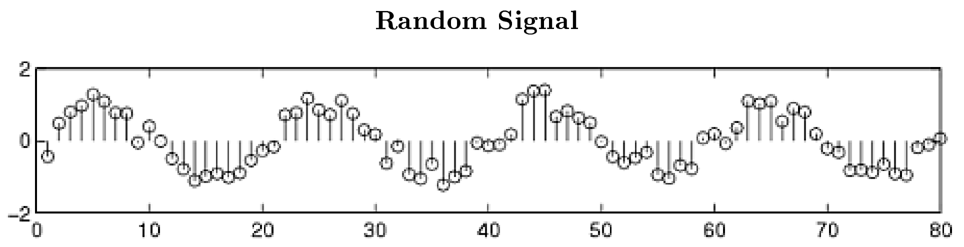


Figure 2.2: We have taken the above sine wave and added random noise to it to come up with a noisy, or random, signal. These are the types of signals that we wish to learn how to deal with so that we can recover the original sine wave.

2.1.2 Random Process

As mentioned above, in order to study random signals, we want to look at a collection of these signals rather than just one instance of that signal. This collection of signals is called a **random process**.

Definition 2.1: random process
A family or ensemble of signals that correspond to every possible outcome of a certain signal measurement. Each signal in this collection is referred to as a **realization** or **sample function** of the process.

Example
As an example of a random process, let us look at the Random Sinusoidal Process below. We use $f[n] = A\sin(\omega n + \phi)$ to represent the sinusoid with a given amplitude and phase. Note that the

phase and amplitude of each sinusoid is based on a random number, thus making this a random process.

Random Sinusoidal Process

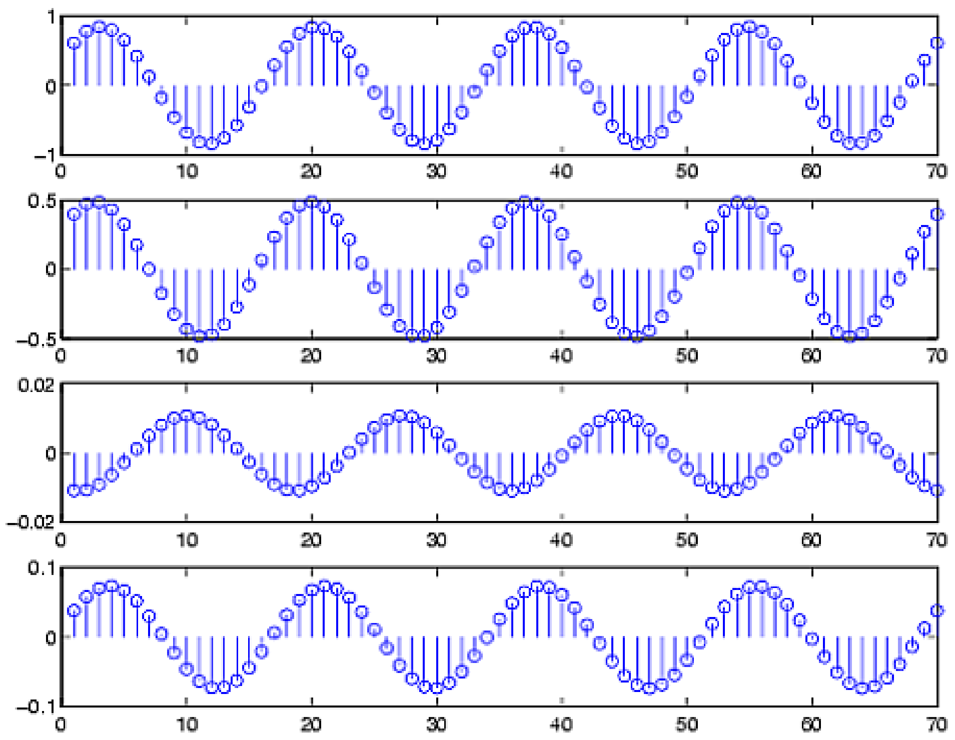


Figure 2.3: A random sinusoidal process, with the amplitude and phase being random numbers.

A random process is usually denoted by $X(t)$ or $X[n]$, with $x(t)$ or $x[n]$ used to represent an individual signal or waveform from this process.

In many notes and books, you might see the following notation and terms used to describe different types of random processes. For a **discrete random process**, sometimes just called a **random sequence**, t represents time that has a finite number of values. If t can take on any value of time, we have a **continuous random process**. Often times discrete and continuous refer to the amplitude of the process, and process or sequence refer to the nature of the time variable. For this study, we often just use **random process** to refer to a general collection of discrete-time signals, as seen above in Figure 2.3 (Random Sinusoidal Process).

2.2 Introduction to Stochastic Processes[4]

2.2.1 Definitions, distributions, and stationarity

Definition 2.2: Stochastic Process
Given a sample space, a stochastic process is an indexed collection of random variables defined for each $\omega \in \Omega$.

$$X_t(\omega) \quad , \quad t \in \mathbb{R} \tag{2.1}$$

Example 2.1
Received signal at an antenna as in Figure 2.4.

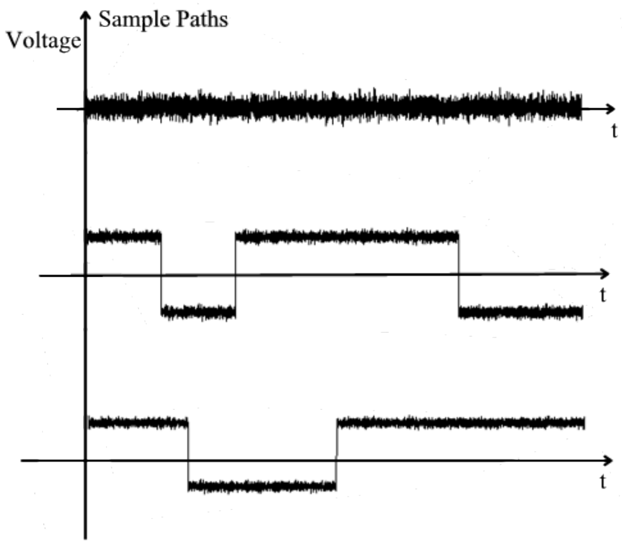


Figure 2.4

For a given t, $X_t(\omega)$ is a random variable with a distribution

First-order distribution

$$\begin{aligned} F_{X_t}(b) &= Pr[X_t \leq b] \\ &= Pr[\{\omega \in \Omega \mid X_t(\omega) \leq b\}] \end{aligned} \tag{2.2}$$

Definition 2.3: First-order stationary process
If $F_{X_t}(b)$ is not a function of time then X_t is called a first-order stationary process.

Second-order distribution

$$F_{X_{t_1},X_{t_2}}(b_1,b_2) = Pr[X_{t_1} \leq b_1, X_{t_2} \leq b_2] \tag{2.3}$$

for all $t_1 \in \mathbb{R}$, $t_2 \in \mathbb{R}$, $b_1 \in \mathbb{R}$, $b_2 \in \mathbb{R}$

[4]This content is available online at <http://cnx.org/content/m10235/2.15/>.

Nth-order distribution

$$F_{X_{t_1},X_{t_2},\ldots,X_{t_N}}\left(b_1,b_2,\ldots,b_N\right)=Pr\left[X_{t_1}\le b_1,\ldots,X_{t_N}\le b_N\right] \tag{2.4}$$

Nth-order stationary : A random process is stationary of order N if

$$F_{X_{t_1},X_{t_2},\ldots,X_{t_N}}\left(b_1,b_2,\ldots,b_N\right)=F_{X_{t_1+T},X_{t_2+T},\ldots,X_{t_N+T}}\left(b_1,b_2,\ldots,b_N\right) \tag{2.5}$$

Strictly stationary : A process is strictly stationary if it is Nth order stationary for all N.

Example 2.2

$X_t = \cos\left(2\pi f_0 t+\Theta(\omega)\right)$ where f_0 is the deterministic carrier frequency and $\Theta(\omega):\Omega\to\mathbb{R}$ is a random variable defined over $[-\pi,\pi]$ and is assumed to be a uniform random variable; *i.e.*, $f_\Theta(\theta)=\begin{cases}\frac{1}{2\pi} & \text{if } \theta\in[-\pi,\pi]\\ 0 & \text{otherwise}\end{cases}$

$$\begin{aligned}F_{X_t}(b) &= Pr\left[X_t\le b\right]\\ &= Pr\left[\cos\left(2\pi f_0 t+\Theta\right)\le b\right]\end{aligned} \tag{2.6}$$

$$F_{X_t}(b)=Pr\left[-\pi\le 2\pi f_0 t+\Theta\le -\arccos(b)\right]+Pr\left[\arccos(b)\le 2\pi f_0 t+\Theta\le\pi\right] \tag{2.7}$$

$$\begin{aligned}F_{X_t}(b) &= \int_{(-\pi)-2\pi f_0 t}^{(-\arccos(b))-2\pi f_0 t}\frac{1}{2\pi}d\theta+\int_{\arccos(b)-2\pi f_0 t}^{\pi-2\pi f_0 t}\frac{1}{2\pi}d\theta\\ &= \left(2\pi-2\arccos(b)\right)\frac{1}{2\pi}\end{aligned} \tag{2.8}$$

$$\begin{aligned}f_{X_t}(x) &= \frac{d}{dx}\left(1-\frac{1}{\pi}\arccos(x)\right)\\ &= \begin{cases}\frac{1}{\pi\sqrt{1-x^2}} & \text{if } |x|\le 1\\ 0 & \text{otherwise}\end{cases}\end{aligned} \tag{2.9}$$

This process is stationary of order 1.

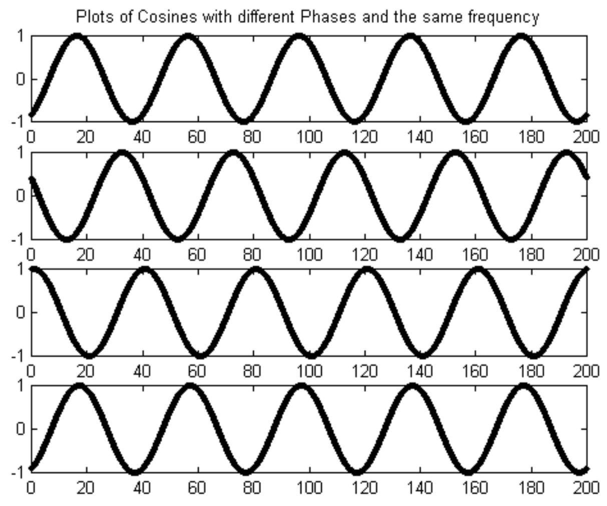


Figure 2.5

The second order stationarity can be determined by first considering conditional densities and the joint density. Recall that

$$X_t = \cos(2\pi f_0 t + \Theta) \tag{2.10}$$

Then the relevant step is to find

$$Pr\left[X_{t_2} \leq b_2 \;\mid X_{t_1} = x_1\right] \tag{2.11}$$

Note that

$$(X_{t_1} = x_1 = \cos(2\pi f_0 t + \Theta)) \Rightarrow (\Theta = \arccos(x_1) - 2\pi f_0 t) \tag{2.12}$$

$$\begin{aligned} X_{t_2} &= \cos(2\pi f_0 t_2 + \arccos(x_1) - 2\pi f_0 t_1) \\ &= \cos(2\pi f_0 (t_2 - t_1) + \arccos(x_1)) \end{aligned} \tag{2.13}$$

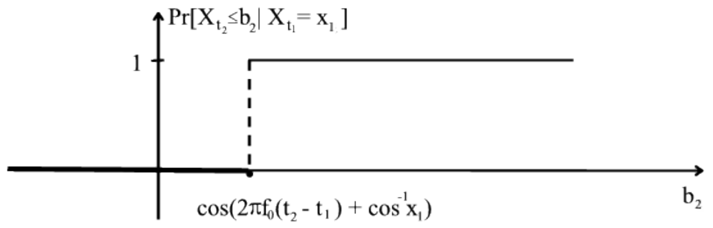


Figure 2.6

$$F_{X_{t_2},X_{t_1}}(b_2,b_1)=\int_{-\infty}^{b_1} f_{X_{t_1}}(x_1)\,Pr\left[X_{t_2}\le b_2\ \mid X_{t_1}=x_1\right]d\text{x}\,1 \tag{2.14}$$

Note that this is only a function of $t_2 - t_1$.

Example 2.3

Every T seconds, a fair coin is tossed. If heads, then $X_t = 1$ for $nT \le t < (n+1)T$. If tails, then $X_t = -1$ for $nT \le t < (n+1)T$.

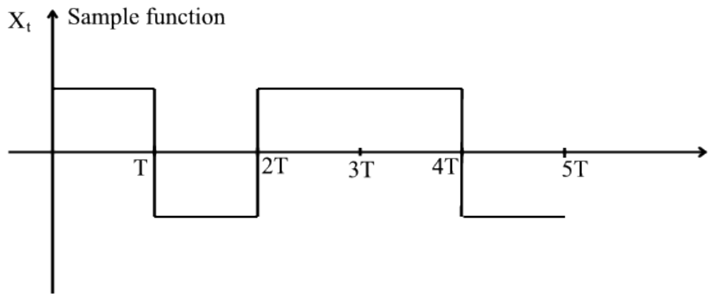


Figure 2.7

$$p_{X_t}(x)=\begin{cases}\frac{1}{2} & \text{if } x=1\\ \frac{1}{2} & \text{if } x=-1\end{cases} \tag{2.15}$$

for all $t \in \mathbb{R}$. X_t is stationary of order 1.

Second order probability mass function

$$p_{X_{t_1}X_{t_2}}(x_1,x_2)=p_{X_{t_2}|X_{t_1}}(x_2|x_1)\,p_{X_{t_1}}(x_1) \tag{2.16}$$

The conditional pmf

$$p_{X_{t_2}|X_{t_1}}(x_2|x_1)=\begin{cases}0 & \text{if } x_2\neq x_1\\ 1 & \text{if } x_2=x_1\end{cases} \tag{2.17}$$

when $nT \le t_1 < (n+1)T$ and $nT \le t_2 < (n+1)T$ for some n.

$$p_{X_{t_2}|X_{t_1}}(x_2|x_1)=p_{X_{t_2}}(x_2) \tag{2.18}$$

for all x_1 and for all x_2 when $nT \le t_1 < (n+1)T$ and $mT \le t_2 < (m+1)T$ with $n \neq m$

$$p_{X_{t_2}X_{t_1}}(x_2,x_1)=\begin{cases}0\ \text{ if }\ x_2\neq x_1 \text{for } nT\le t_1,t_2<(n+1)T\\ p_{X_{t_1}}(x_1)\ \text{ if }\ x_2=x_1\text{for } nT\le t_1,t_2<(n+1)T\\ p_{X_{t_1}}(x_1)\,p_{X_{t_2}}(x_2)\ \text{ if }\ n\neq m\text{for }\ (nT\le t_1<(n+1)T)\ \ and\ \ (mT\le t_2<(m+1\end{cases} \tag{2.19}$$

2.3 Random Signals[5]

Random signals are random variables which evolve, often with time (e.g. audio noise), but also with distance (e.g. intensity in an image of a random texture), or sometimes another parameter.

They can be described as usual by their cdf and either their pmf (if the amplitude is discrete, as in a digitized signal) or their pdf (if the amplitude is continuous, as in most analogue signals).

However a very important additional property is how rapidly a random signal fluctuates. Clearly a slowly varying signal such as the waves in an ocean is very different from a rapidly varying signal such as vibrations in a vehicle. We will see later in here[6] how to deal with these frequency dependent characteristics of randomness.

For the moment we shall assume that random signals are sampled at regular intervals and that each signal is equivalent to a sequence of samples of a given random process, as in the following examples.

[5]This content is available online at <http://cnx.org/content/m10989/2.5/>.
[6]"Approximation Formulae for the Gaussian Error Integral, Q(x)" <http://cnx.org/content/m11067/latest/>

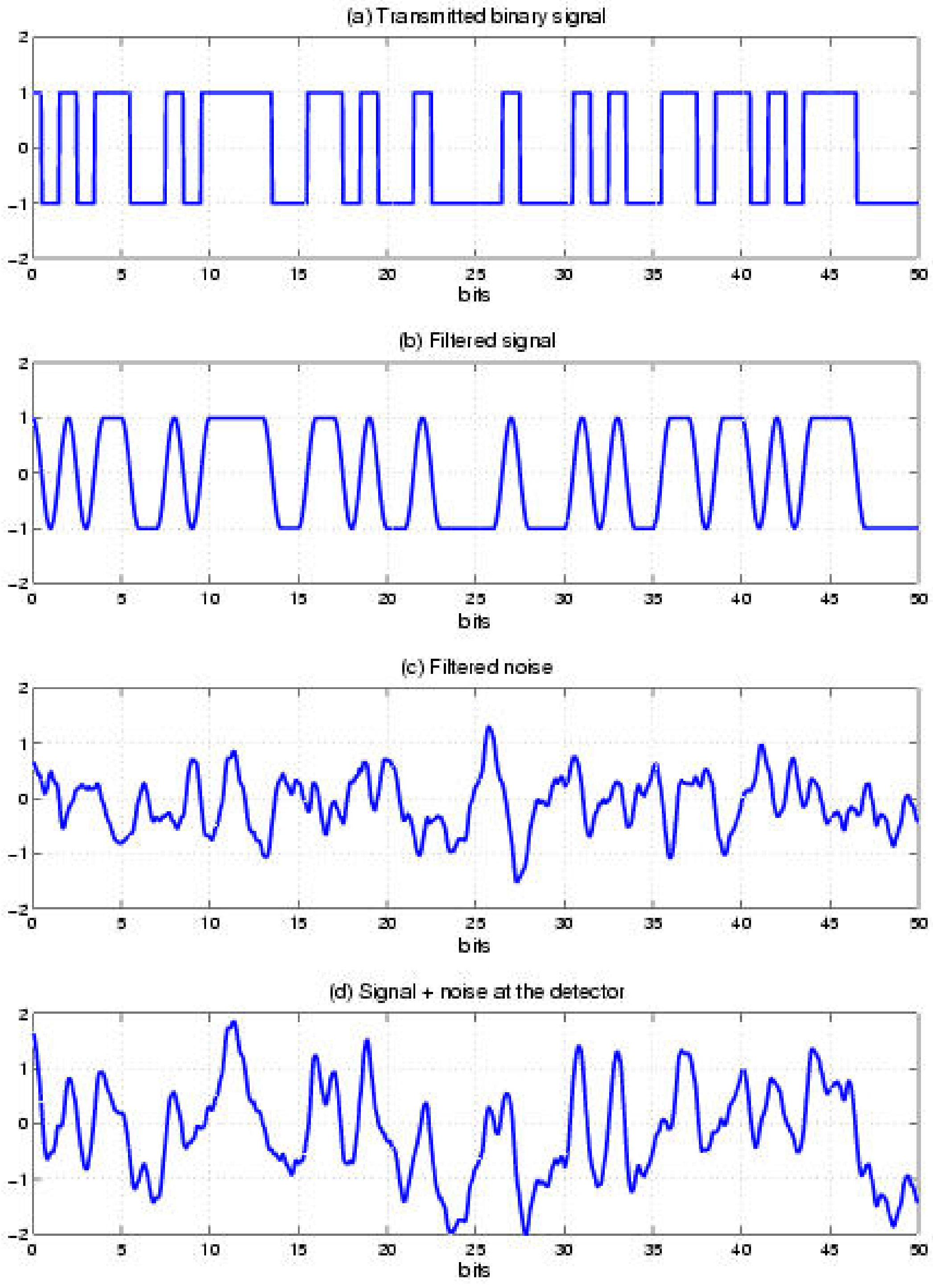


Figure 2.8: Detection of signals in noise: (a) the transmitted binary signal; (b) the binary signal after filtering with a half-sine receiver filter; (c) the channel noise after filtering with the same filter; (d) the filtered signal plus noise at the detector in the receiver.

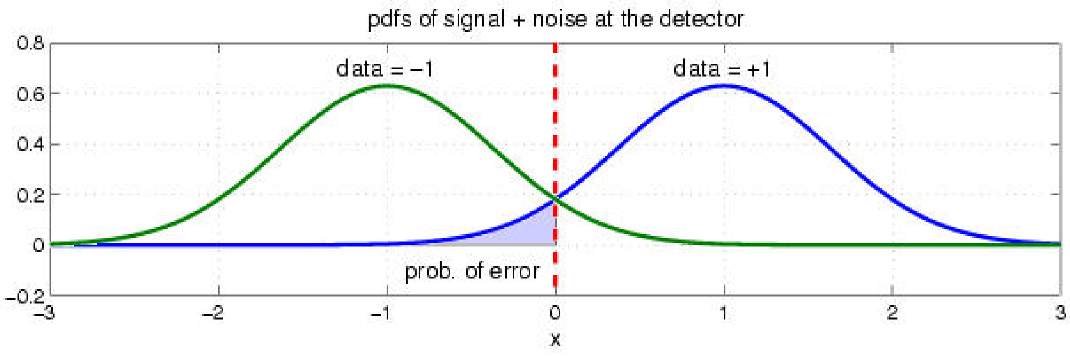


Figure 2.9: The pdfs of the signal plus noise at the detector for the two $\pm(1)$. The vertical dashed line is the detector threshold and the shaded area to the left of the origin represents the probability of error when data = 1.

2.3.1 Example - Detection of a binary signal in noise

We now consider the example of detecting a binary signal after it has passed through a channel which adds noise. The transmitted signal is typically as shown in (a) of Figure 2.8.

In order to reduce the channel noise, the receiver will include a lowpass filter. The aim of the filter is to reduce the noise as much as possible without reducing the peak values of the signal significantly. A good filter for this has a half-sine impulse response of the form:

$$h(t) = \begin{cases} \frac{\pi}{2T_b}\sin\left(\frac{\pi t}{T_b}\right) & \text{if } 0 \le t \le T_b \\ 0 & \text{otherwise} \end{cases} \tag{2.20}$$

Where T_b = bit period.

This filter will convert the rectangular data bits into sinusoidally shaped pulses as shown in (b) of Figure 2.8 and it will also convert wide bandwidth channel noise into the form shown in (c) of Figure 2.8. Bandlimited noise of this form will usually have an approximately Gaussian pdf.

Because this filter has an impulse response limited to just one bit period and has unit gain at zero frequency (the area under $h(t)$ is unity), the signal values at the center of each bit period at the detector will still be $\pm(1)$. If we choose to sample each bit at the detector at this optimal mid point, the pdfs of the signal plus noise at the detector will be shown in Figure 2.9.

Let the filtered data signal be $D(t)$ and the filtered noise be $U(t)$, then the detector signal is

$$R(t) = D(t) + U(t) \tag{2.21}$$

If we assume that $+(1)$ and -1 bits are equiprobable and the noise is a symmetric zero-mean process, the optimum detector threshold is clearly midway between these two states, i.e. at zero. The probability of error when the data $= +(1)$ is then given by:

$$\begin{aligned} Pr\left[\text{error } \mid D = +(1)\right] &= Pr\left[R(t) < 0 \mid D = +(1)\right] \\ &= F_U(-1) \\ &= \int_{-\infty}^{-1} f_U(u)\,du \end{aligned} \tag{2.22}$$

where F_U and f_U are the cdf and pdf of U. This is the shaded area in Figure 2.9.

Similarly the probability of error when the data $= -1$ is then given by:

$$\begin{aligned} Pr\left[\text{error} \;\mid D = -1\right] &= Pr\left[R\left(t\right) > 0 \;\mid D = -1\right] \\ &= 1 - F_U\left(+\left(1\right)\right) \\ &= \int_1^\infty f_U\left(u\right)du \end{aligned} \tag{2.23}$$

Hence the overall probability of error is:

$$\begin{aligned} Pr\left[\text{error}\right] &= Pr\left[\text{error} \;\mid D = +\left(1\right)\right]Pr\left[D = +\left(1\right)\right] + Pr\left[\text{error} \;\mid D = -1\right]Pr\left[D = -1\right] \\ &= \int_{-\infty}^{-1} f_U\left(u\right)duPr\left[D = +\left(1\right)\right] + \int_1^\infty f_U\left(u\right)duPr\left[D = -1\right] \end{aligned} \tag{2.24}$$

since f_U is symmetric about zero

$$Pr\left[\text{error}\right] = \int_1^\infty f_U\left(u\right)du\left(Pr\left[D = +\left(1\right)\right] + Pr\left[D = -1\right]\right) = \int_1^\infty f_U\left(u\right)du$$

To be a little more general and to account for signal attenuation over the channel, we shall assume that the signal values at the detector are $\pm\left(v_0\right)$ (rather than $\pm\left(1\right)$) and that the filtered noise at the detector has a zero-mean Gaussian pdf with variance σ^2:

$$f_U\left(u\right) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{u^2}{2\sigma^2}} \tag{2.25}$$

and so

$$\begin{aligned} Pr\left[\text{error}\right] &= \int_{v_0}^\infty f_U\left(u\right)du \\ &= \int_{\frac{v_0}{\sigma}}^\infty f_U\left(\sigma u\right)\sigma du \\ &= Q\left(\frac{v_0}{\sigma}\right) \end{aligned} \tag{2.26}$$

where

$$Q\left(x\right) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{u^2}{2}}du \tag{2.27}$$

This integral has no analytic solution, but a good approximation to it exists and is discussed in some detail in here[7].

From (2.26) we may obtain the probability of error in the binary detector, which is often expressed as the **bit error rate** or **BER**. For example, if $Pr\left[\text{error}\right] = 2 \times 10^3$, this would often be expressed as a bit error rate of 2×10^3, or alternatively as 1 error in 500 bits (on average).

The argument ($\frac{v_0}{\sigma}$) in (2.26) is the signal-to-noise voltage ratio (SNR) at the detector, and the BER rapidly diminishes with increasing SNR (see here[8]).

2.4 Stationary and Nonstationary Random Processes[9]

2.4.1 Introduction

From the definition of a random process (Section 2.1), we know that all random processes are composed of random variables, each at its own unique point in time. Because of this, random processes have all the properties of random variables, such as mean, correlation, variances, etc.. When dealing with groups of signals

[7]"Approximation Formulae for the Gaussian Error Integral, Q(x)" <http://cnx.org/content/m11067/latest/>

[8]"Approximation Formulae for the Gaussian Error Integral, Q(x)", Figure 1 <http://cnx.org/content/m11067/latest/#figure1>

[9]This content is available online at <http://cnx.org/content/m10684/2.2/>.

or sequences it will be important for us to be able to show whether of not these statistical properties hold true for the entire random process. To do this, the concept of **stationary processes** has been developed. The general definition of a stationary process is:

Definition 2.4: stationary process
a random process where all of its statistical properties do not vary with time

Processes whose statistical properties do change are referred to as **nonstationary**.

Understanding the basic idea of stationarity will help you to be able to follow the more concrete and mathematical definition to follow. Also, we will look at various levels of stationarity used to describe the various types of stationarity characteristics a random process can have.

2.4.2 Distribution and Density Functions

In order to properly define what it means to be stationary from a mathematical standpoint, one needs to be somewhat familiar with the concepts of distribution and density functions. If you can remember your statistics then feel free to skip this section!

Recall that when dealing with a single random variable, the **probability distribution function** is a simply tool used to identify the probability that our observed random variable will be less than or equal to a given number. More precisely, let X be our random variable, and let x be our given value; from this we can define the distribution function as

$$F_x(x) = Pr[X \leq x] \tag{2.28}$$

This same idea can be applied to instances where we have multiple random variables as well. There may be situations where we want to look at the probability of event X **and** Y both occurring. For example, below is an example of a second-order **joint distribution function**.

$$F_x(x, y) = Pr[X \leq x, Y \leq y] \tag{2.29}$$

While the distribution function provides us with a full view of our variable or processes probability, it is not always the most useful for calculations. Often times we will want to look at its derivative, the **probability density function (pdf)**. We define the the pdf as

$$f_x(x) = \frac{d}{dx} F_x(x) \tag{2.30}$$

$$f_x(x)\,\mathrm{dx} = Pr[x < X \leq x + \mathrm{dx}] \tag{2.31}$$

(2.31) reveals some of the physical significance of the density function. This equations tells us the probability that our random variable falls within a given interval can be approximated by $f_x(x)\,\mathrm{dx}$. From the pdf, we can now use our knowledge of integrals to evaluate probabilities from the above approximation. Again we can also define a **joint density function** which will include multiple random variables just as was done for the distribution function. The density function is used for a variety of calculations, such as finding the expected value or proving a random variable is stationary, to name a few.

NOTE: The above examples explain the distribution and density functions in terms of a single random variable, X. When we are dealing with signals and random processes, remember that we will have a set of random variables where a different random variable will occur at each time instance of the random process, $X(t_k)$. In other words, the distribution and density function will also need to take into account the choice of time.

2.4.3 Stationarity

Below we will now look at a more in depth and mathematical definition of a stationary process. As was mentioned previously, various levels of stationarity exist and we will look at the most common types.

2.4.3.1 First-Order Stationary Process

A random process is classified as **first-order stationary** if its first-order probability density function remains equal regardless of any shift in time to its time origin. If we let x_{t_1} represent a given value at time t_1, then we define a first-order stationary as one that satisfies the following equation:

$$f_x\left(x_{t_1}\right) = f_x\left(x_{t_1+\tau}\right) \tag{2.32}$$

The physical significance of this equation is that our density function, $f_x\left(x_{t_1}\right)$, is completely independent of t_1 and thus any time shift, τ.

The most important result of this statement, and the identifying characteristic of any first-order stationary process, is the fact that the mean is a constant, independent of any time shift. Below we show the results for a random process, X, that is a discrete-time signal, $x[n]$.

$$\begin{aligned} \bar{X} &= m_x[n] \\ &= E[x[n]] \\ &= \text{constant (independent of n)} \end{aligned} \tag{2.33}$$

2.4.3.2 Second-Order and Strict-Sense Stationary Process

A random process is classified as **second-order stationary** if its second-order probability density function does not vary over any time shift applied to both values. In other words, for values x_{t_1} and x_{t_2} then we will have the following be equal for an arbitrary time shift τ.

$$f_x\left(x_{t_1}, x_{t_2}\right) = f_x\left(x_{t_1+\tau}, x_{t_2+\tau}\right) \tag{2.34}$$

From this equation we see that the absolute time does not affect our functions, rather it only really depends on the time difference between the two variables. Looked at another way, this equation can be described as

$$Pr\left[X\left(t_1\right) \leq x_1, X\left(t_2\right) \leq x_2\right] = Pr\left[X\left(t_1+\tau\right) \leq x_1, X\left(t_2+\tau\right) \leq x_2\right] \tag{2.35}$$

These random processes are often referred to as **strict sense stationary (SSS)** when **all** of the distribution functions of the process are unchanged regardless of the time shift applied to them.

For a second-order stationary process, we need to look at the autocorrelation function (Section 2.7) to see its most important property. Since we have already stated that a second-order stationary process depends only on the time difference, then all of these types of processes have the following property:

$$\begin{aligned} R_{xx}(t, t+\tau) &= E[X(t+\tau)] \\ &= R_{xx}(\tau) \end{aligned} \tag{2.36}$$

2.4.3.3 Wide-Sense Stationary Process

As you begin to work with random processes, it will become evident that the strict requirements of a SSS process is more than is often necessary in order to adequately approximate our calculations on random processes. We define a final type of stationarity, referred to as **wide-sense stationary (WSS)**, to have slightly more relaxed requirements but ones that are still enough to provide us with adequate results. In order to be WSS a random process only needs to meet the following two requirements.

1. $\bar{X} = E[x[n]] = \text{constant}$
2. $E[X(t+\tau)] = R_{xx}(\tau)$

Note that a second-order (or SSS) stationary process will always be WSS; however, the reverse will not always hold true.

2.5 Random Processes: Mean and Variance[10]

In order to study the characteristics of a random process (Section 2.1), let us look at some of the basic properties and operations of a random process. Below we will focus on the operations of the random signals that compose our random processes. We will denote our random process with X and a random variable from a random process or signal by x.

2.5.1 Mean Value

Finding the average value of a set of random signals or random variables is probably the most fundamental concepts we use in evaluating random processes through any sort of statistical method. **The mean of a random process is the average of all realizations of that process.** In order to find this average, we must look at a random signal over a range of time (possible values) and determine our average from this set of values. The **mean**, or average, of a random process, $x(t)$, is given by the following equation:

$$\begin{aligned} m_x(t) &= \mu_x(t) \\ &= \bar{X} \\ &= E[X] \\ &= \int_{-\infty}^{\infty} x f(x)\, dx \end{aligned} \tag{2.37}$$

This equation may seem quite cluttered at first glance, but we want to introduce you to the various notations used to represent the mean of a random signal or process. Throughout texts and other readings, remember that these will all equal the same thing. The symbol, $\mu_x(t)$, and the X with a bar over it are often used as a short-hand to represent an average, so you might see it in certain textbooks. The other important notation used is, $E[X]$, which represents the "expected value of X" or the mathematical expectation. This notation is very common and will appear again.

If the random variables, which make up our random process, are discrete or quantized values, such as in a binary process, then the integrals become summations over all the possible values of the random variable. In this case, our expected value becomes

$$E[x[n]] = \sum_x \alpha Pr[x[n] = \alpha] \tag{2.38}$$

If we have two random signals or variables, their averages can reveal how the two signals interact. If the product of the two individual averages of both signals do **not** equal the average of the product of the two signals, then the two signals are said to be **linearly independent**, also referred to as **uncorrelated**.

In the case where we have a random process in which only one sample can be viewed at a time, then we will often not have all the information available to calculate the mean using the density function as shown above. In this case we must estimate the mean through the time-average mean (Section 2.5.4: Time Averages), discussed later. For fields such as signal processing that deal mainly with discrete signals and values, then these are the averages most commonly used.

2.5.1.1 Properties of the Mean

- The expected value of a constant, α, is the constant:

$$E[\alpha] = \alpha \tag{2.39}$$

- Adding a constant, α, to each term increases the expected value by that constant:

$$E[X + \alpha] = E[X] + \alpha \tag{2.40}$$

[10]This content is available online at <http://cnx.org/content/m10656/2.3/>.

- Multiplying the random variable by a constant, α, multiplies the expected value by that constant.

$$E[\alpha X] = \alpha E[X] \tag{2.41}$$

- The expected value of the sum of two or more random variables, is the sum of each individual expected value.

$$E[X+Y] = E[X] + E[Y] \tag{2.42}$$

2.5.2 Mean-Square Value

If we look at the second **moment** of the term (we now look at x^2 in the integral), then we will have the **mean-square value** of our random process. As you would expect, this is written as

$$\begin{aligned} \bar{X^2} &= E\left[X^2\right] \\ &= \int_{-\infty}^{\infty} x^2 f(x)\, dx \end{aligned} \tag{2.43}$$

This equation is also often referred to as the **average power** of a process or signal.

2.5.3 Variance

Now that we have an idea about the average value or values that a random process takes, we are often interested in seeing just how spread out the different random values might be. To do this, we look at the **variance** which is a measure of this spread. The variance, often denoted by σ^2, is written as follows:

$$\begin{aligned} \sigma^2 &= \mathrm{Var}(X) \\ &= E\left[(X - E[X])^2\right] \\ &= \int_{-\infty}^{\infty} \left(x - \bar{X}\right)^2 f(x)\, dx \end{aligned} \tag{2.44}$$

Using the rules for the expected value, we can rewrite this formula as the following form, which is commonly seen:

$$\begin{aligned} \sigma^2 &= \bar{X^2} - \left(\bar{X}\right)^2 \\ &= E\left[X^2\right] - (E[X])^2 \end{aligned} \tag{2.45}$$

2.5.3.1 Standard Deviation

Another common statistical tool is the standard deviation. Once you know how to calculate the variance, the standard deviation is simply **the square root of the variance**, or σ.

2.5.3.2 Properties of Variance

- The variance of a constant, α, equals zero:

$$\begin{aligned} \mathrm{Var}(\alpha) &= \sigma(\alpha)^2 \\ &= 0 \end{aligned} \tag{2.46}$$

- Adding a constant, α, to a random variable does not affect the variance because the mean increases by the same value:

$$\begin{aligned}\mathrm{Var}\left(X+\alpha\right) &= \sigma\left(X+\alpha\right)^2 \\ &= \sigma\left(X\right)^2\end{aligned} \tag{2.47}$$

- Multiplying the random variable by a constant, α, increases the variance by the square of the constant:

$$\begin{aligned}\mathrm{Var}\left(\alpha X\right) &= \sigma\left(\alpha X\right)^2 \\ &= \alpha^2\sigma\left(X\right)^2\end{aligned} \tag{2.48}$$

- The variance of the sum of two random variables only equals the sum of the variances if the variable are **independent**.

$$\begin{aligned}\mathrm{Var}\left(X+Y\right) &= \sigma\left(X+Y\right)^2 \\ &= \sigma\left(X\right)^2+\sigma\left(Y\right)^2\end{aligned} \tag{2.49}$$

  Otherwise, if the random variable are **not** independent, then we must also include the covariance of the product of the variables as follows:

$$\mathrm{Var}\left(X+Y\right)=\sigma\left(X\right)^2+2\mathrm{Cov}\left(X,Y\right)+\sigma\left(Y\right)^2 \tag{2.50}$$

2.5.4 Time Averages

In the case where we can not view the entire ensemble of the random process, we must use time averages to estimate the values of the mean and variance for the process. Generally, this will only give us acceptable results for independent and **ergodic** processes, meaning those processes in which each signal or member of the process seems to have the same statistical behavior as the entire process. The time averages will also only be taken over a finite interval since we will only be able to see a finite part of the sample.

2.5.4.1 Estimating the Mean

For the ergodic random process, $x\left(t\right)$, we will estimate the mean using the time averaging function defined as

$$\begin{aligned}\bar{X} &= E\left[X\right] \\ &= \tfrac{1}{T}\int_0^T X\left(t\right)dt\end{aligned} \tag{2.51}$$

However, for most real-world situations we will be dealing with discrete values in our computations and signals. We will represent this mean as

$$\begin{aligned}\bar{X} &= E\left[X\right] \\ &= \tfrac{1}{N}\sum_{n=1}^{N} X\left[n\right]\end{aligned} \tag{2.52}$$

2.5.4.2 Estimating the Variance

Once the mean of our random process has been estimated then we can simply use those values in the following variance equation (introduced in one of the above sections)

$$\sigma_x{}^2=\bar{X^2}-\left(\bar{X}\right)^2 \tag{2.53}$$

2.5.5 Example

Let us now look at how some of the formulas and concepts above apply to a simple example. We will just look at a single, continuous random variable for this example, but the calculations and methods are the same for a random process. For this example, we will consider a random variable having the probability density function described below and shown in Figure 2.10 (Probability Density Function).

$$f(x) = \begin{cases} \frac{1}{10} & \text{if } 10 \leq x \leq 20 \\ 0 & \text{otherwise} \end{cases} \quad (2.54)$$

Probability Density Function

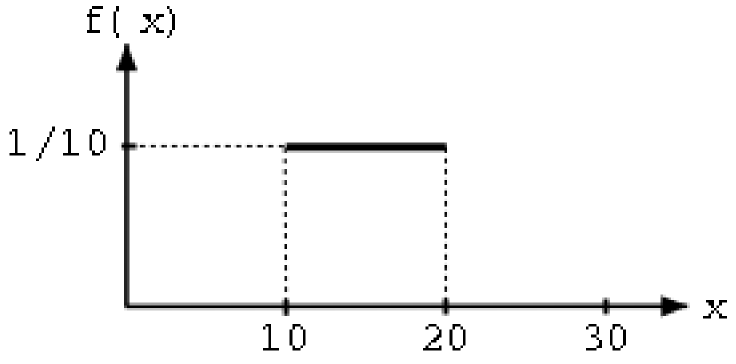


Figure 2.10: A uniform probability density function.

First, we will use (2.37) to solve for the mean value.

$$\begin{aligned} \bar{X} &= \int_{10}^{20} x \frac{1}{10} dx \\ &= \frac{1}{10} \frac{x^2}{2} \Big|_{x=10}^{20} \\ &= \frac{1}{10}(200 - 50) \\ &= 15 \end{aligned} \quad (2.55)$$

Using (2.43) we can obtain the mean-square value for the above density function.

$$\begin{aligned} \bar{X^2} &= \int_{10}^{20} x^2 \frac{1}{10} dx \\ &= \frac{1}{10} \frac{x^3}{3} \Big|_{x=10}^{20} \\ &= \frac{1}{10}\left(\frac{8000}{3} - \frac{1000}{3}\right) \\ &= 233.33 \end{aligned} \quad (2.56)$$

And finally, let us solve for the variance of this function.

$$\begin{aligned} \sigma^2 &= \bar{X^2} - \left(\bar{X}\right)^2 \\ &= 233.33 - 15^2 \\ &= 8.33 \end{aligned} \quad (2.57)$$

2.6 Correlation and Covariance of a Random Signal[11]

When we take the expected value (Section 2.5), or average, of a random process (Section 2.1.2: Random Process), we measure several important characteristics about how the process behaves in general. This proves to be a very important observation. However, suppose we have several random processes measuring different aspects of a system. The relationship between these different processes will also be an important observation. The covariance and correlation are two important tools in finding these relationships. Below we will go into more details as to what these words mean and how these tools are helpful. Note that much of the following discussions refer to just random variables, but keep in mind that these variables can represent random signals or random processes.

2.6.1 Covariance

To begin with, when dealing with more than one random process, it should be obvious that it would be nice to be able to have a number that could quickly give us an idea of how similar the processes are. To do this, we use the **covariance**, which is analogous to the variance of a single variable.

Definition 2.5: Covariance
A measure of how much the deviations of two or more variables or processes match.

For two processes, X and Y, if they are **not** closely related then the covariance will be small, and if they are similar then the covariance will be large. Let us clarify this statement by describing what we mean by "related" and "similar." Two processes are "closely related" if their distribution spreads are almost equal and they are around the same, or a very slightly different, mean.

Mathematically, covariance is often written as σ_{xy} and is defined as

$$\begin{aligned} \operatorname{cov}(X,Y) &= \sigma_{xy} \\ &= E\left[\left(X-\bar{X}\right)\left(Y-\bar{Y}\right)\right] \end{aligned} \tag{2.58}$$

This can also be reduced and rewritten in the following two forms:

$$\sigma_{xy} = \bar{(xy)} - \bar{x}\bar{y} \tag{2.59}$$

$$\sigma_{xy} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(X-\bar{X}\right)\left(Y-\bar{Y}\right)f(x,y)\,dx\,dy \tag{2.60}$$

2.6.1.1 Useful Properties

- If X and Y are independent and uncorrelated or one of them has zero mean value, then
$$\sigma_{xy} = 0$$
- If X and Y are orthogonal, then
$$\sigma_{xy} = -\left(E[X]\,E[Y]\right)$$
- The covariance is symmetric
$$\operatorname{cov}(X,Y) = \operatorname{cov}(Y,X)$$

[11]This content is available online at <http://cnx.org/content/m10673/2.3/>.

- Basic covariance identity
$$\mathrm{cov}\,(X+Y,Z)=\mathrm{cov}\,(X,Z)+\mathrm{cov}\,(Y,Z)$$
- Covariance of equal variables
$$\mathrm{cov}\,(X,X)=\mathrm{Var}\,(X)$$

2.6.2 Correlation

For anyone who has any kind of statistical background, you should be able to see that the idea of dependence/independence among variables and signals plays an important role when dealing with random processes. Because of this, the **correlation** of two variables provides us with a measure of how the two variables affect one another.

Definition 2.6: Correlation
A measure of how much one random variable depends upon the other.

This measure of association between the variables will provide us with a clue as to how well the value of one variable can be predicted from the value of the other. The correlation is equal to the average of the product of two random variables and is defined as

$$\begin{aligned}\mathrm{cor}\,(X,Y) &= E\,[XY]\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} xyf\,(x,y)\,dxdy\end{aligned} \tag{2.61}$$

2.6.2.1 Correlation Coefficient

It is often useful to express the correlation of random variables with a range of numbers, like a percentage. For a given set of variables, we use the **correlation coefficient** to give us the linear relationship between our variables. The correlation coefficient of two variables is defined in terms of their covariance and standard deviations (Section 2.5.3.1: Standard Deviation), denoted by σ_x, as seen below

$$\rho=\frac{\mathrm{cov}\,(X,Y)}{\sigma_x\sigma_y} \tag{2.62}$$

where we will always have

$$-1\leq\rho\leq 1$$

This provides us with a quick and easy way to view the correlation between our variables. If there is no relationship between the variables then the correlation coefficient will be zero and if there is a perfect positive match it will be one. If there is a perfect inverse relationship, where one set of variables increases while the other decreases, then the correlation coefficient will be negative one. This type of correlation is often referred to more specifically as the **Pearson's Correlation Coefficient**,or Pearson's Product Moment Correlation.

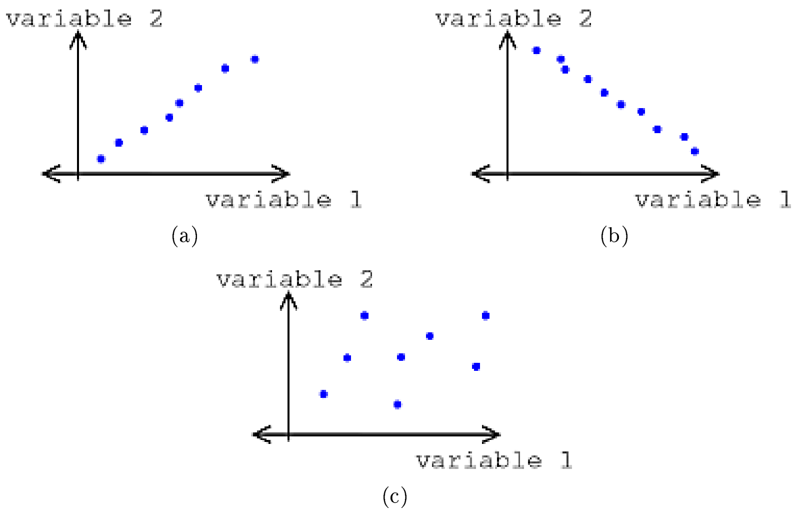


Figure 2.11: Types of Correlation (a) Positive Correlation (b) Negative Correlation (c) Uncorrelated (No Correlation)

NOTE: So far we have dealt with correlation simply as a number relating the relationship between any two variables. However, since our goal will be to relate random processes to each other, which deals with signals as a function of time, we will want to continue this study by looking at **correlation functions (Section 2.7)**.

2.6.3 Example

Now let us take just a second to look at a simple example that involves calculating the covariance and correlation of two sets of random numbers. We are given the following data sets:

$$x = \{3, 1, 6, 3, 4\}$$

$$y = \{1, 5, 3, 4, 3\}$$

To begin with, for the covariance we will need to find the expected value (Section 2.5), or mean, of x and y.

$$\overline{x} = \frac{1}{5}(3 + 1 + 6 + 3 + 4) = 3.4$$

$$\overline{y} = \frac{1}{5}(1 + 5 + 3 + 4 + 3) = 3.2$$

$$\overline{xy} = \frac{1}{5}(3 + 5 + 18 + 12 + 12) = 10$$

Next we will solve for the standard deviations of our two sets using the formula below (for a review click here (Section 2.5.3: Variance)).

$$\sigma = \sqrt{E\left[(X - E[X])^2\right]}$$

$$\sigma_x = \sqrt{\frac{1}{5}\left(0.16 + 5.76 + 6.76 + 0.16 + 0.36\right)} = 1.625$$

$$\sigma_y = \sqrt{\frac{1}{6}\left(4.84 + 3.24 + 0.04 + 0.64 + 0.04\right)} = 1.327$$

Now we can finally calculate the covariance using one of the two formulas found above. Since we calculated the three means, we will use that formula (2.59) since it will be much simpler.

$$\sigma_{xy} = 10 - 3.4 \times 3.2 = -0.88$$

And for our last calculation, we will solve for the correlation coefficient, ρ.

$$\rho = \frac{-0.88}{1.625 \times 1.327} = -0.408$$

2.6.3.1 Matlab Code for Example

The above example can be easily calculated using Matlab. Below I have included the code to find all of the values above.

```
x = [3 1 6 3 4];
y = [1 5 3 4 3];

mx = mean(x)
my = mean(y)
mxy = mean(x.*y)

% Standard Dev. from built-in Matlab Functions
std(x,1)
std(y,1)

% Standard Dev. from Equation Above (same result as std(?,1))
sqrt( 1/5 * sum((x-mx).^2))
sqrt( 1/5 * sum((y-my).^2))

cov(x,y,1)

corrcoef(x,y)
```

2.7 Autocorrelation of Random Processes[12]

Before diving into a more complex statistical analysis of random signals and processes (Section 2.1), let us quickly review the idea of correlation (Section 2.6). Recall that the correlation of two signals or variables is the expected value of the product of those two variables. Since our focus will be to discover more about a random process, a collection of random signals, then imagine us dealing with two samples of a random

[12]This content is available online at <http://cnx.org/content/m10676/2.4/>.

process, where each sample is taken at a different point in time. Also recall that the key property of these random processes is that they are now functions of time; imagine them as a collection of signals. The expected value (Section 2.5) of the product of these two variables (or samples) will now depend on how quickly they change in regards to **time**. For example, if the two variables are taken from almost the same time period, then we should expect them to have a high correlation. We will now look at a correlation function that relates a pair of random variables from the same process to the time separations between them, where the argument to this correlation function will be the time difference. For the correlation of signals from two different random process, look at the crosscorrelation function (Section 2.8).

2.7.1 Autocorrelation Function

The first of these correlation functions we will discuss is the **autocorrelation**, where each of the random variables we will deal with come from the same random process.

Definition 2.7: Autocorrelation
the expected value of the product of a random variable or signal realization with a time-shifted version of itself

With a simple calculation and analysis of the autocorrelation function, we can discover a few important characteristics about our random process. These include:

1. How quickly our random signal or processes changes with respect to the time function
2. Whether our process has a periodic component and what the expected frequency might be

As was mentioned above, the autocorrelation function is simply the expected value of a product. Assume we have a pair of random variables from the same process, $X_1 = X(t_1)$ and $X_2 = X(t_2)$, then the autocorrelation is often written as

$$\begin{aligned} R_{\text{xx}}(t_1, t_2) &= E[X_1 X_2] \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_1 x_2 f(x_1, x_2)\, d\text{x } 2d\text{x } 1 \end{aligned} \tag{2.63}$$

The above equation is valid for stationary and nonstationary random processes. For stationary processes (Section 2.4), we can generalize this expression a little further. Given a wide-sense stationary processes, it can be proven that the expected values from our random process will be independent of the origin of our time function. Therefore, we can say that our autocorrelation function will depend on the time difference and not some absolute time. For this discussion, we will let $\tau = t_2 - t_1$, and thus we generalize our autocorrelation expression as

$$\begin{aligned} R_{\text{xx}}(t, t+\tau) &= R_{\text{xx}}(\tau) \\ &= E[X(t) X(t+\tau)] \end{aligned} \tag{2.64}$$

for the continuous-time case. In most DSP course we will be more interested in dealing with real signal sequences, and thus we will want to look at the discrete-time case of the autocorrelation function. The formula below will prove to be more common and useful than (2.63):

$$R_{\text{xx}}[n, n+m] = \sum_{n=-\infty}^{\infty} x[n]\, x[n+m] \tag{2.65}$$

And again we can generalize the notation for our autocorrelation function as

$$\begin{aligned} R_{\text{xx}}[n, n+m] &= R_{\text{xx}}[m] \\ &= E[X[n] X[n+m]] \end{aligned} \tag{2.66}$$

2.7.1.1 Properties of Autocorrelation

Below we will look at several properties of the autocorrelation function that hold for **stationary** random processes.

- Autocorrelation is an even function for τ
$$R_{\mathrm{xx}}(\tau) = R_{\mathrm{xx}}(-\tau)$$
- The mean-square value can be found by evaluating the autocorrelation where $\tau = 0$, which gives us
$$R_{\mathrm{xx}}(0) = \bar{X}^2$$
- The autocorrelation function will have its largest value when $\tau = 0$. This value can appear again, for example in a periodic function at the values of the equivalent periodic points, but will never be exceeded.
$$R_{\mathrm{xx}}(0) \geq |R_{\mathrm{xx}}(\tau)|$$
- If we take the autocorrelation of a period function, then $R_{\mathrm{xx}}(\tau)$ will also be periodic with the same frequency.

2.7.1.2 Estimating the Autocorrleation with Time-Averaging

Sometimes the whole random process is not available to us. In these cases, we would still like to be able to find out some of the characteristics of the stationary random process, even if we just have part of one sample function. In order to do this we can **estimate** the autocorrelation from a given interval, 0 to T seconds, of the sample function.

$$\breve{R}_{\mathrm{xx}}(\tau) = \frac{1}{T-\tau}\int_0^{T-\tau} x(t)\, x(t+\tau)\, dt \tag{2.67}$$

However, a lot of times we will not have sufficient information to build a complete continuous-time function of one of our random signals for the above analysis. If this is the case, we can treat the information we do know about the function as a discrete signal and use the discrete-time formula for estimating the autocorrelation.

$$\breve{R}_{\mathrm{xx}}[m] = \frac{1}{N-m}\sum_{n=0}^{N-m-1} x[n]\, x[n+m] \tag{2.68}$$

2.7.2 Examples

Below we will look at a variety of examples that use the autocorrelation function. We will begin with a simple example dealing with Gaussian White Noise (GWN) and a few basic statistical properties that will prove very useful in these and future calculations.

Example 2.4
We will let $x[n]$ represent our GWN. For this problem, it is important to remember the following fact about the mean of a GWN function:

$$E[x[n]] = 0$$

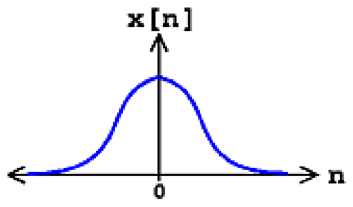


Figure 2.12: Gaussian density function. By examination, can easily see that the above statement is true - the mean equals zero.

Along with being **zero-mean**, recall that GWN is always **independent**. With these two facts, we are now ready to do the short calculations required to find the autocorrelation.

$$R_{\mathrm{xx}}[n, n+m] = E[x[n]\,x[n+m]]$$

Since the function, $x[n]$, is independent, then we can take the product of the individual expected values of both functions.

$$R_{\mathrm{xx}}[n, n+m] = E[x[n]]\,E[x[n+m]]$$

Now, looking at the above equation we see that we can break it up further into two conditions: one when m and n are equal and one when they are not equal. When they are equal we can combine the expected values. We are left with the following piecewise function to solve:

$$R_{\mathrm{xx}}[n, n+m] = \begin{cases} E[x[n]]\,E[x[n+m]] & \text{if } m \neq 0 \\ E\left[x^2[n]\right] & \text{if } m = 0 \end{cases}$$

We can now solve the two parts of the above equation. The first equation is easy to solve as we have already stated that the expected value of $x[n]$ will be zero. For the second part, you should recall from statistics that the expected value of the square of a function is equal to the variance. Thus we get the following results for the autocorrelation:

$$R_{\mathrm{xx}}[n, n+m] = \begin{cases} 0 & \text{if } m \neq 0 \\ \sigma^2 & \text{if } m = 0 \end{cases}$$

Or in a more concise way, we can represent the results as

$$R_{\mathrm{xx}}[n, n+m] = \sigma^2 \delta[m]$$

2.8 Crosscorrelation of Random Processes[13]

Before diving into a more complex statistical analysis of random signals and processes (Section 2.1), let us quickly review the idea of correlation (Section 2.6). Recall that the correlation of two signals or variables is the expected value of the product of those two variables. Since our main focus is to discover more about random processes, a collection of random signals, we will deal with two random processes in this discussion, where in this case we will deal with samples from two **different** random processes. We will analyze the expected value (Section 2.5.1: Mean Value) of the product of these two variables and how they correlate to one another, where the argument to this correlation function will be the time difference. For the correlation of signals from the same random process, look at the autocorrelation function (Section 2.7).

[13]This content is available online at <http://cnx.org/content/m10686/2.2/>.

2.8.1 Crosscorrelation Function

When dealing with multiple random processes, it is also important to be able to describe the relationship, if any, between the processes. For example, this may occur if more than one random signal is applied to a system. In order to do this, we use the **crosscorrelation function**, where the variables are instances from two different wide sense stationary random processes.

Definition 2.8: Crosscorrelation
if two processes are wide sense stationary, the expected value of the product of a random variable from one random process with a time-shifted, random variable from a different random process

Looking at the generalized formula for the crosscorrelation, we will represent our two random processes by allowing $U = U(t)$ and $V = V(t - \tau)$. We will define the crosscorrelation function as

$$\begin{aligned} R_{uv}(t, t-\tau) &= E[UV] \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} uvf(u,v)\,dvdu \end{aligned} \tag{2.69}$$

Just as the case with the autocorrelation function, if our input and output, denoted as $U(t)$ and $V(t)$, are at least jointly wide sense stationary, then the crosscorrelation does not depend on absolute time; it is just a function of the time difference. This means we can simplify our writing of the above function as

$$R_{uv}(\tau) = E[UV] \tag{2.70}$$

or if we deal with two real signal sequences, $x[n]$ and $y[n]$, then we arrive at a more commonly seen formula for the discrete crosscorrelation function. See the formula below and notice the similarities between it and the convolution[14] of two signals:

$$\begin{aligned} R_{xy}(n, n-m) &= R_{xy}(m) \\ &= \sum_{n=-\infty}^{\infty} x[n]\,y[n-m] \end{aligned} \tag{2.71}$$

2.8.1.1 Properties of Crosscorrelation

Below we will look at several properties of the crosscorrelation function that hold for two **wide sense stationary (WSS)** random processes.

- Crosscorrelation is **not** an even function; however, it does have a unique symmetry property:

$$R_{xy}(-\tau) = R_{yx}(\tau) \tag{2.72}$$

- The maximum value of the crosscorrelation is not always when the shift equals zero; however, we can prove the following property revealing to us what value the maximum cannot exceed.

$$|R_{xy}(\tau)| \leq \sqrt{R_{xx}(0)\,R_{yy}(0)} \tag{2.73}$$

- When two random processes are statistically independent then we have

$$R_{xy}(\tau) = R_{yx}(\tau) \tag{2.74}$$

[14]"Discrete Time Convolution" <http://cnx.org/content/m10087/latest/>

2.8.2 Examples

Exercise 2.8.1 ***(Solution on p. 87.)***

Let us begin by looking at a simple example showing the relationship between two sequences. Using (2.71), find the crosscorrelation of the sequences

$$x[n] = \{\dots, 0, 0, 2, -3, 6, 1, 3, 0, 0, \dots\}$$

$$y[n] = \{\dots, 0, 0, 1, -2, 4, 1, -3, 0, 0, \dots\}$$

for each of the following possible time shifts: $m = \{0, 3, -1\}$.

Solutions to Exercises in Chapter 2

Solution to Exercise 2.8.1 (p. 86)

1. For $m = 0$, we should begin by finding the product sequence $s[n] = x[n]\,y[n]$. Doing this we get the following sequence:
$$s[n] = \{\dots, 0, 0, 2, 6, 24, 1, -9, 0, 0, \dots\}$$
and so from the sum in our crosscorrelation function we arrive at the answer of
$$R_{xy}(0) = 22$$
2. For $m = 3$, we will approach it the same was we did above; however, we will now shift $y[n]$ to the right. Then we can find the product sequence $s[n] = x[n]\,y[n-3]$, which yields
$$s[n] = \{\dots, 0, 0, 0, 0, 0, 1, -6, 0, 0, \dots\}$$
and from the crosscorrelation function we arrive at the answer of
$$R_{xy}(3) = -6$$
3. For $m = -1$, we will again take the same approach; however, we will now shift $y[n]$ to the left. Then we can find the product sequence $s[n] = x[n]\,y[n+1]$, which yields
$$s[n] = \{\dots, 0, 0, -4, -12, 6, -3, 0, 0, 0, \dots\}$$
and from the crosscorrelation function we arrive at the answer of
$$R_{xy}(-1) = -13$$

Chapter 3

Filter Design I (Z-Transform)

3.1 Difference Equation[1]

3.1.1 Introduction

One of the most important concepts of DSP is to be able to properly represent the input/output relationship to a given LTI system. A linear constant-coefficient **difference equation** (LCCDE) serves as a way to express just this relationship in a discrete-time system. Writing the sequence of inputs and outputs, which represent the characteristics of the LTI system, as a difference equation help in understanding and manipulating a system.

Definition 3.1: difference equation
An equation that shows the relationship between consecutive values of a sequence and the differences among them. They are often rearranged as a recursive formula so that a systems output can be computed from the input signal and past outputs.
Example

$$y[n] + 7y[n-1] + 2y[n-2] = x[n] - 4x[n-1] \tag{3.1}$$

3.1.2 General Formulas for the Difference Equation

As stated briefly in the definition above, a difference equation is a very useful tool in describing and calculating the output of the system described by the formula for a given sample n. The key property of the difference equation is its ability to help easily find the transform, $H(z)$, of a system. In the following two subsections, we will look at the general form of the difference equation and the general conversion to a z-transform directly from the difference equation.

3.1.2.1 Difference Equation

The general form of a linear, constant-coefficient difference equation (LCCDE), is shown below:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \tag{3.2}$$

[1]This content is available online at <http://cnx.org/content/m10595/2.6/>.

We can also write the general form to easily express a recursive output, which looks like this:

$$y[n] = -\sum_{k=1}^{N} a_k y[n-k] + \sum_{k=0}^{M} b_k x[n-k] \tag{3.3}$$

From this equation, note that $y[n-k]$ represents the outputs and $x[n-k]$ represents the inputs. The value of N represents the **order** of the difference equation and corresponds to the memory of the system being represented. Because this equation relies on past values of the output, in order to compute a numerical solution, certain past outputs, referred to as the **initial conditions**, must be known.

3.1.2.2 Conversion to Z-Transform

Using the above formula, (3.2), we can easily generalize the **transfer function**, $H(z)$, for any difference equation. Below are the steps taken to convert any difference equation into its transfer function, *i.e.* z-transform. The first step involves taking the Fourier Transform[2] of all the terms in (3.2). Then we use the linearity property to pull the transform inside the summation and the time-shifting property of the z-transform to change the time-shifting terms to exponentials. Once this is done, we arrive at the following equation: $a_0 = 1$.

$$Y(z) = -\sum_{k=1}^{N} a_k Y(z) z^{-k} + \sum_{k=0}^{M} b_k X(z) z^{-k} \tag{3.4}$$

$$\begin{aligned} H(z) &= \frac{Y(z)}{X(z)} \\ &= \frac{\sum_{k=0}^{M} b_k z^{-k}}{1+\sum_{k=1}^{N} a_k z^{-k}} \end{aligned} \tag{3.5}$$

3.1.2.3 Conversion to Frequency Response

Once the z-transform has been calculated from the difference equation, we can go one step further to define the frequency response of the system, or filter, that is being represented by the difference equation.

NOTE: Remember that the reason we are dealing with these formulas is to be able to aid us in filter design. A LCCDE is one of the easiest ways to represent FIR filters. By being able to find the frequency response, we will be able to look at the basic properties of any filter represented by a simple LCCDE.

Below is the general formula for the frequency response of a z-transform. The conversion is simple a matter of taking the z-transform formula, $H(z)$, and replacing every instance of z with e^{jw}.

$$\begin{aligned} H(w) &= H(z)|_{z,z=e^{jw}} \\ &= \frac{\sum_{k=0}^{M} b_k e^{-(jwk)}}{\sum_{k=0}^{N} a_k e^{-(jwk)}} \end{aligned} \tag{3.6}$$

Once you understand the derivation of this formula, look at the module concerning Filter Design from the Z-Transform (Section 3.10) for a look into how all of these ideas of the Z-transform (Section 3.2), Difference Equation, and Pole/Zero Plots (Section 3.8) play a role in filter design.

3.1.3 Example

Example 3.1: Finding Difference Equation

Below is a basic example showing the opposite of the steps above: given a transfer function one can easily calculate the systems difference equation.

$$H(z) = \frac{(z+1)^2}{\left(z-\frac{1}{2}\right)\left(z+\frac{3}{4}\right)} \tag{3.7}$$

[2]"Derivation of the Fourier Transform" <http://cnx.org/content/m0046/latest/>

Given this transfer function of a time-domain filter, we want to find the difference equation. To begin with, expand both polynomials and divide them by the highest order z.

$$\begin{aligned} H(z) &= \frac{(z+1)(z+1)}{\left(z-\frac{1}{2}\right)\left(z+\frac{3}{4}\right)} \\ &= \frac{z^2+2z+1}{z^2+2z+1-\frac{3}{8}} \\ &= \frac{1+2z^{-1}+z^{-2}}{1+\frac{1}{4}z^{-1}-\frac{3}{8}z^{-2}} \end{aligned} \tag{3.8}$$

From this transfer function, the coefficients of the two polynomials will be our a_k and b_k values found in the general difference equation formula, (3.2). Using these coefficients and the above form of the transfer function, we can easily write the difference equation:

$$x[n] + 2x[n-1] + x[n-2] = y[n] + \frac{1}{4}y[n-1] - \frac{3}{8}y[n-2] \tag{3.9}$$

In our final step, we can rewrite the difference equation in its more common form showing the recursive nature of the system.

$$y[n] = x[n] + 2x[n-1] + x[n-2] + \frac{-1}{4}y[n-1] + \frac{3}{8}y[n-2] \tag{3.10}$$

3.1.4 Solving a LCCDE

In order for a linear constant-coefficient difference equation to be useful in analyzing a LTI system, we must be able to find the systems output based upon a known input, $x(n)$, and a set of initial conditions. Two common methods exist for solving a LCCDE: the **direct method** and the **indirect method**, the later being based on the z-transform. Below we will briefly discuss the formulas for solving a LCCDE using each of these methods.

3.1.4.1 Direct Method

The final solution to the output based on the direct method is the sum of two parts, expressed in the following equation:

$$y(n) = y_h(n) + y_p(n) \tag{3.11}$$

The first part, $y_h(n)$, is referred to as the **homogeneous solution** and the second part, $y_h(n)$, is referred to as **particular solution**. The following method is very similar to that used to solve many differential equations, so if you have taken a differential calculus course or used differential equations before then this should seem very familiar.

3.1.4.1.1 Homogeneous Solution

We begin by assuming that the input is zero, $x(n) = 0$. Now we simply need to solve the homogeneous difference equation:

$$\sum_{k=0}^{N} a_k y[n-k] = 0 \tag{3.12}$$

In order to solve this, we will make the assumption that the solution is in the form of an exponential. We will use lambda, λ, to represent our exponential terms. We now have to solve the following equation:

$$\sum_{k=0}^{N} a_k \lambda^{n-k} = 0 \tag{3.13}$$

We can expand this equation out and factor out all of the lambda terms. This will give us a large polynomial in parenthesis, which is referred to as the **characteristic polynomial**. The roots of this polynomial will be the key to solving the homogeneous equation. If there are all distinct roots, then the general solution to the equation will be as follows:

$$y_h(n) = C_1(\lambda_1)^n + C_2(\lambda_2)^n + \cdots + C_N(\lambda_N)^n \tag{3.14}$$

However, if the characteristic equation contains multiple roots then the above general solution will be slightly different. Below we have the modified version for an equation where λ_1 has K multiple roots:

$$y_h(n) = C_1(\lambda_1)^n + C_1 n(\lambda_1)^n + C_1 n^2(\lambda_1)^n + \cdots + C_1 n^{K-1}(\lambda_1)^n + C_2(\lambda_2)^n + \cdots + C_N(\lambda_N)^n \tag{3.15}$$

3.1.4.1.2 Particular Solution

The particular solution, $y_p(n)$, will be any solution that will solve the general difference equation:

$$\sum_{k=0}^{N} a_k y_p(n-k) = \sum_{k=0}^{M} b_k x(n-k) \tag{3.16}$$

In order to solve, our guess for the solution to $y_p(n)$ will take on the form of the input, $x(n)$. After guessing at a solution to the above equation involving the particular solution, one only needs to plug the solution into the difference equation and solve it out.

3.1.4.2 Indirect Method

The indirect method utilizes the relationship between the difference equation and z-transform, discussed earlier (Section 3.1.2: General Formulas for the Difference Equation), to find a solution. The basic idea is to convert the difference equation into a z-transform, as described above (Section 3.1.2.2: Conversion to Z-Transform), to get the resulting output, $Y(z)$. Then by inverse transforming this and using partial-fraction expansion, we can arrive at the solution.

$$Z\{y(n+1) - y(n)\} = zY(z) - y(0) \tag{3.17}$$

This can be interatively extended to an arbitrary order derivative as in Equation (3.18).

$$Z\{-\sum_{m=0}^{N-1} y(n-m)\} = z^n Y(z) - \sum_{m=0}^{N-1} z^{n-m-1} y^{(m)}(0) \tag{3.18}$$

Now, the Laplace transform of each side of the differential equation can be taken

$$Z\{\sum_{k=0}^{N} a_k \left[y(n-m+1) - \sum_{m=0}^{N-1} y(n-m)\, y(n) \right] = Z\{x(n)\}\} \tag{3.19}$$

which by linearity results in

$$\sum_{k=0}^{N} a_k Z\{y(n-m+1) - \sum_{m=0}^{N-1} y(n-m)\, y(n)\} = Z\{x(n)\} \tag{3.20}$$

and by differentiation properties in

$$\sum_{k=0}^{N} a_k \left(z^k Z\{y(n)\} - \sum_{m=0}^{N-1} z^{k-m-1} y^{(m)}(0) \right) = Z\{x(n)\}. \tag{3.21}$$

Rearranging terms to isolate the Laplace transform of the output,

$$Z\{y(n)\} = \frac{Z\{x(n)\} + \sum_{k=0}^{N} \sum_{m=0}^{k-1} a_k z^{k-m-1} y^{(m)}(0)}{\sum_{k=0}^{N} a_k z^k}. \tag{3.22}$$

Thus, it is found that

$$Y(z) = \frac{X(z) + \sum_{k=0}^{N} \sum_{m=0}^{k-1} a_k z^{k-m-1} y^{(m)}(0)}{\sum_{k=0}^{N} a_k z^k}. \tag{3.23}$$

In order to find the output, it only remains to find the Laplace transform $X(z)$ of the input, substitute the initial conditions, and compute the inverse Z-transform of the result. Partial fraction expansions are often required for this last step. This may sound daunting while looking at (3.23), but it is often easy in practice, especially for low order difference equations. (3.23) can also be used to determine the transfer function and frequency response.

As an example, consider the difference equation

$$y[n-2] + 4y[n-1] + 3y[n] = cos(n) \tag{3.24}$$

with the initial conditions $y^{'}(0) = 1$ and $y(0) = 0$ Using the method described above, the Z transform of the solution $y[n]$ is given by

$$Y[z] = \frac{z}{[z^2+1][z+1][z+3]} + \frac{1}{[z+1][z+3]}. \tag{3.25}$$

Performing a partial fraction decomposition, this also equals

$$Y[z] = .25\frac{1}{z+1} - .35\frac{1}{z+3} + .1\frac{z}{z^2+1} + .2\frac{1}{z^2+1}. \tag{3.26}$$

Computing the inverse Laplace transform,

$$y(n) = \left(.25z^{-n} - .35z^{-3n} + .1cos(n) + .2sin(n)\right)u(n). \tag{3.27}$$

One can check that this satisfies that this satisfies both the differential equation and the initial conditions.

3.2 The Z Transform: Definition[3]

3.2.1 Basic Definition of the Z-Transform

The **z-transform** of a sequence is defined as

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n} \tag{3.28}$$

Sometimes this equation is referred to as the **bilateral z-transform**. At times the z-transform is defined as

$$X(z) = \sum_{n=0}^{\infty} x[n] z^{-n} \tag{3.29}$$

which is known as the **unilateral z-transform**.

There is a close relationship between the z-transform and the **Fourier transform** of a discrete time signal, which is defined as

$$X\left(e^{j\omega}\right) = \sum_{n=-\infty}^{\infty} x[n] e^{-(j\omega n)} \tag{3.30}$$

[3]This content is available online at <http://cnx.org/content/m10549/2.10/>.

Notice that that when the z^{-n} is replaced with $e^{-(j\omega n)}$ the z-transform reduces to the Fourier Transform. When the Fourier Transform exists, $z = e^{j\omega}$, which is to have the magnitude of z equal to unity.

3.2.2 The Complex Plane

In order to get further insight into the relationship between the Fourier Transform and the Z-Transform it is useful to look at the complex plane or **z-plane**. Take a look at the complex plane:

Z-Plane

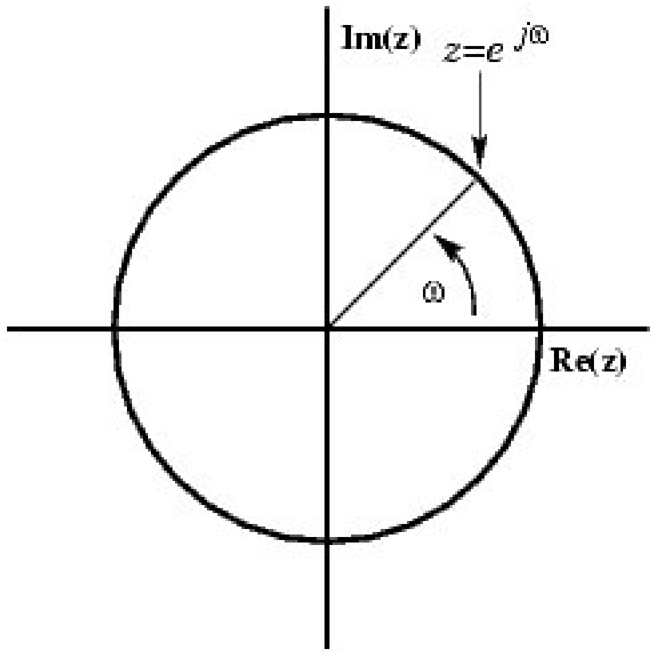


Figure 3.1

The Z-plane is a complex plane with an imaginary and real axis referring to the complex-valued variable z. The position on the complex plane is given by $re^{j\omega}$, and the angle from the positive, real axis around the plane is denoted by ω. $X(z)$ is defined everywhere on this plane. $X\left(e^{j\omega}\right)$ on the other hand is defined only where $|z| = 1$, which is referred to as the unit circle. So for example, $\omega = 1$ at $z = 1$ and $\omega = \pi$ at $z = -1$. This is useful because, by representing the Fourier transform as the z-transform on the unit circle, the periodicity of Fourier transform is easily seen.

3.2.3 Region of Convergence

The region of convergence, known as the **ROC**, is important to understand because it defines the region where the z-transform exists. The ROC for a given $x[n]$, is defined as the range of z for which the z-transform converges. Since the z-transform is a **power series**, it converges when $x[n]z^{-n}$ is absolutely summable.

Stated differently,

$$\sum_{n=-\infty}^{\infty} \left|x[n] z^{-n}\right| < \infty \tag{3.31}$$

must be satisfied for convergence. This is best illustrated by looking at the different ROC's of the z-transforms of $\alpha^n u[n]$ and $\alpha^n u[n-1]$.

Example 3.2
For

$$x[n] = \alpha^n u[n] \tag{3.32}$$

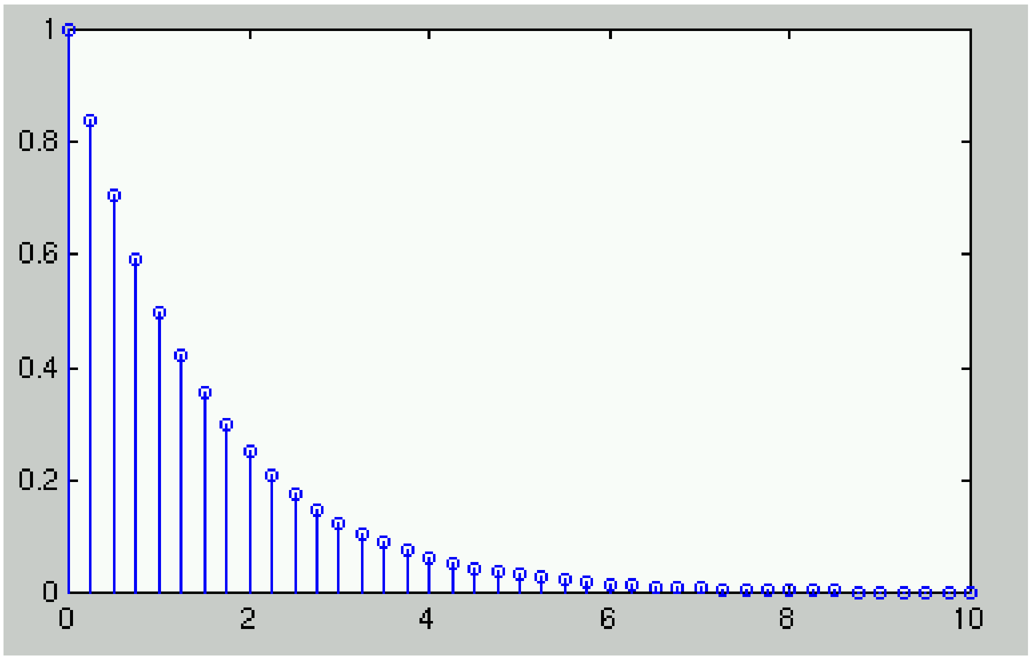


Figure 3.2: $x[n] = \alpha^n u[n]$ where $\alpha = 0.5$.

$$\begin{aligned} X(z) &= \sum_{n=-\infty}^{\infty} x[n] z^{-n} \\ &= \sum_{n=-\infty}^{\infty} \alpha^n u[n] z^{-n} \\ &= \sum_{n=0}^{\infty} \alpha^n z^{-n} \\ &= \sum_{n=0}^{\infty} \left(\alpha z^{-1}\right)^n \end{aligned} \tag{3.33}$$

This sequence is an example of a right-sided exponential sequence because it is nonzero for $n \geq 0$. It only converges when $|\alpha z^{-1}| < 1$. When it converges,

$$\begin{aligned} X(z) &= \frac{1}{1-\alpha z^{-1}} \\ &= \frac{z}{z-\alpha} \end{aligned} \tag{3.34}$$

If $|\alpha z^{-1}| \geq 1$, then the series, $\sum_{n=0}^{\infty} \left(\alpha z^{-1}\right)^n$ does not converge. Thus the ROC is the range of values where

$$|\alpha z^{-1}| < 1 \tag{3.35}$$

or, equivalently,

$$|z| > |\alpha| \tag{3.36}$$

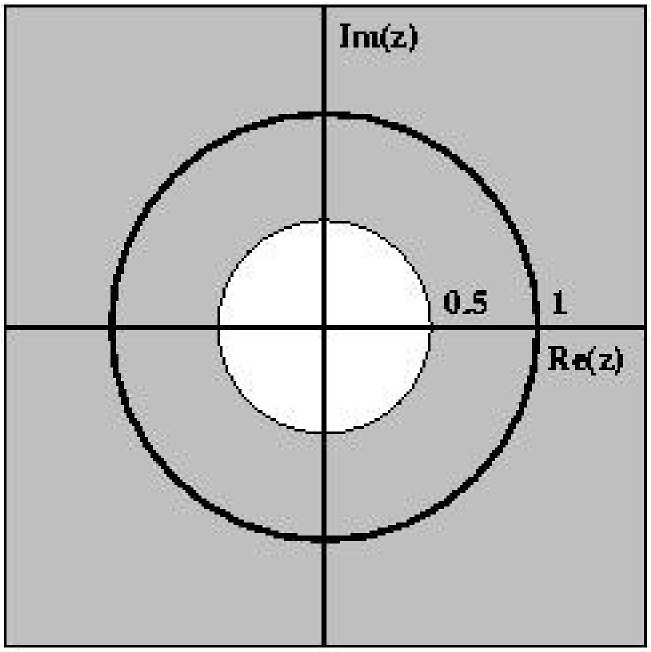


Figure 3.3: ROC for $x[n] = \alpha^n u[n]$ where $\alpha = 0.5$

Example 3.3
For

$$x[n] = (-\alpha^n)\, u[(-n) - 1] \tag{3.37}$$

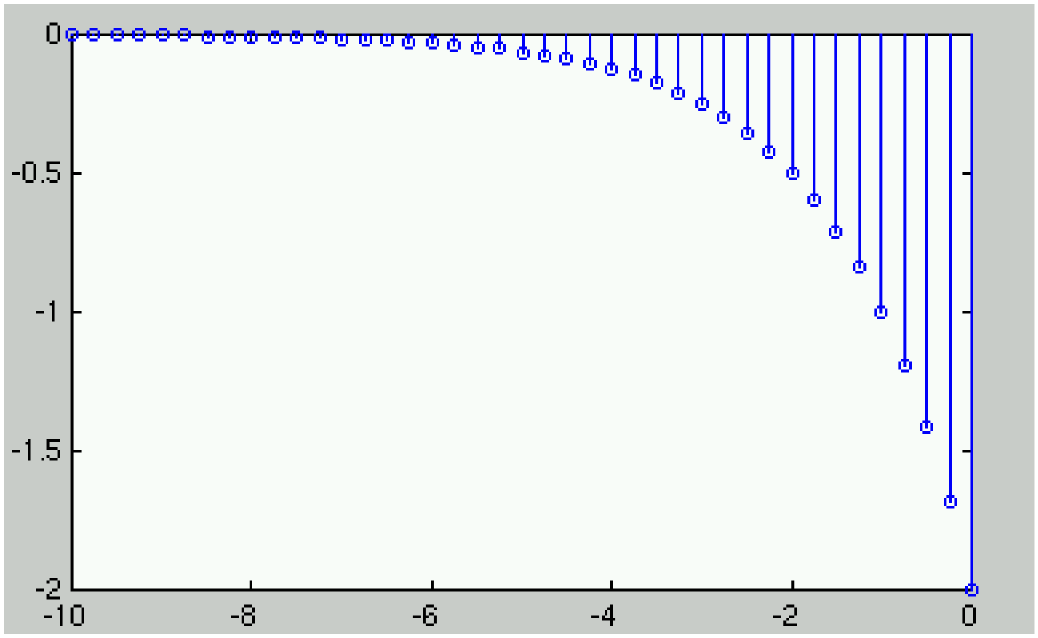


Figure 3.4: $x[n] = (-\alpha^n)\, u[(-n) - 1]$ where $\alpha = 0.5$.

$$\begin{aligned} X(z) &= \textstyle\sum_{n=-\infty}^{\infty} x[n]\, z^{-n} \\ &= \textstyle\sum_{n=-\infty}^{\infty} (-\alpha^n)\, u[-\mathrm{n} - 1]\, z^{-n} \\ &= -\textstyle\sum_{n=-\infty}^{-1} \alpha^n z^{-n} \\ &= -\textstyle\sum_{n=-\infty}^{-1} \left(\alpha^{-1} z\right)^{-n} \\ &= -\textstyle\sum_{n=1}^{\infty} \left(\alpha^{-1} z\right)^{n} \\ &= 1 - \textstyle\sum_{n=0}^{\infty} \left(\alpha^{-1} z\right)^{n} \end{aligned} \tag{3.38}$$

The ROC in this case is the range of values where

$$|\alpha^{-1} z| < 1 \tag{3.39}$$

or, equivalently,

$$|z| < |\alpha| \tag{3.40}$$

If the ROC is satisfied, then

$$\begin{aligned} X(z) &= 1 - \tfrac{1}{1-\alpha^{-1} z} \\ &= \tfrac{z}{z-\alpha} \end{aligned} \tag{3.41}$$

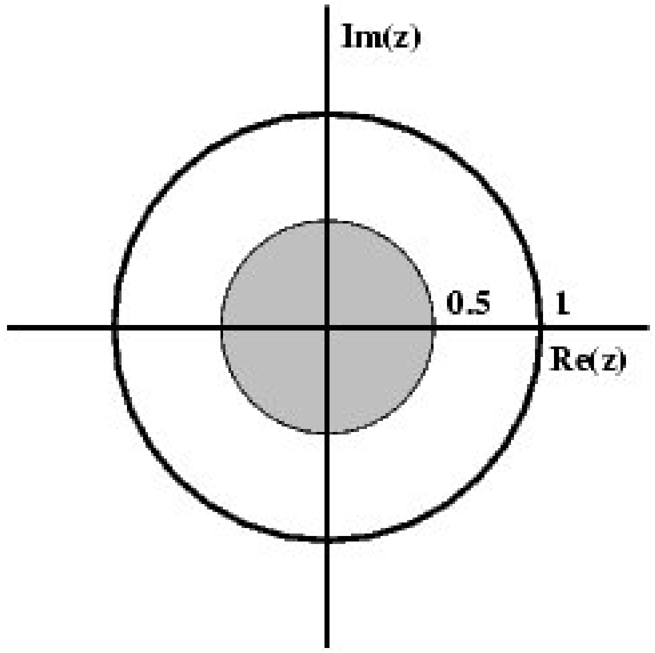


Figure 3.5: ROC for $x[n] = (-\alpha^n)\, u[(-n) - 1]$

3.3 Table of Common z-Transforms[4]

The table below provides a number of unilateral and bilateral **z-transforms (Section 3.2)**. The table also specifies the region of convergence (Section 3.7).

NOTE: The notation for z found in the table below may differ from that found in other tables. For example, the basic z-transform of $u[n]$ can be written as either of the following two expressions, which are equivalent:

$$\frac{z}{z-1} = \frac{1}{1-z^{-1}} \tag{3.42}$$

[4]This content is available online at <http://cnx.org/content/m10119/2.14/>.

Signal	**Z-Transform**	**ROC**
$\delta[n-k]$	z^{-k}	$\text{All}(z)$
$u[n]$	$\frac{z}{z-1}$	$\|z\| > 1$
$-u[(-n)-1]$	$\frac{z}{z-1}$	$\|z\| < 1$
$nu[n]$	$\frac{z}{(z-1)^2}$	$\|z\| > 1$
$n^2u[n]$	$\frac{z(z+1)}{(z-1)^3}$	$\|z\| > 1$
$n^3u[n]$	$\frac{z\left(z^2+4z+1\right)}{(z-1)^4}$	$\|z\| > 1$
$(-\alpha^n)u[(-n)-1]$	$\frac{z}{z-\alpha}$	$\|z\| < \|\alpha\|$
$\alpha^n u[n]$	$\frac{z}{z-\alpha}$	$\|z\| > \|\alpha\|$
$n\alpha^n u[n]$	$\frac{\alpha z}{(z-\alpha)^2}$	$\|z\| > \|\alpha\|$
$n^2\alpha^n u[n]$	$\frac{\alpha z(z+\alpha)}{(z-\alpha)^3}$	$\|z\| > \|\alpha\|$
$\frac{\prod_{k=1}^{m} n-k+1}{\alpha^m m!}\alpha^n u[n]$	$\frac{z}{(z-\alpha)^{m+1}}$	
$\gamma^n\cos(\alpha n)u[n]$	$\frac{z(z-\gamma\cos(\alpha))}{z^2-(2\gamma\cos(\alpha))z+\gamma^2}$	$\|z\| > \|\gamma\|$
$\gamma^n\sin(\alpha n)u[n]$	$\frac{z\gamma\sin(\alpha)}{z^2-(2\gamma\cos(\alpha))z+\gamma^2}$	$\|z\| > \|\gamma\|$

Table 3.1

3.4 Poles and Zeros[5]

3.4.1 Introduction

It is quite difficult to qualitatively analyze the Laplace transform[6] and Z-transform (Section 3.2), since mappings of their magnitude and phase or real part and imaginary part result in multiple mappings of 2-dimensional surfaces in 3-dimensional space. For this reason, it is very common to examine a plot of a transfer function's[7] poles and zeros to try to gain a qualitative idea of what a system does.

Given a continuous-time transfer function in the Laplace domain, $H(s)$, or a discrete-time one in the Z-domain, $H(z)$, a zero is any value of s or z such that the transfer function is zero, and a pole is any value of s or z such that the transfer function is infinite. To define them precisely:

Definition 3.2: zeros

1. The value(s) for z where the **numerator** of the transfer function equals zero
2. The complex frequencies that make the overall gain of the filter transfer function zero.

Definition 3.3: poles

1. The value(s) for z where the **denominator** of the transfer function equals zero
2. The complex frequencies that make the overall gain of the filter transfer function infinite.

3.4.2 Pole/Zero Plots

When we plot these in the appropriate s- or z-plane, we represent zeros with "o" and poles with "x". Refer to this module (Section 3.8) for a detailed looking at plotting the poles and zeros of a z-transform on the Z-plane.

[5]This content is available online at <http://cnx.org/content/m10112/2.12/>.
[6]"The Laplace Transform" <http://cnx.org/content/m10110/latest/>
[7]"Transfer Functions" <http://cnx.org/content/m0028/latest/>

Example 3.4

Find the poles and zeros for the transfer function $H(s) = \frac{s^2+6s+8}{s^2+2}$ and plot the results in the s-plane.

The first thing we recognize is that this transfer function will equal zero whenever the top, s^2+6s+8, equals zero. To find where this equals zero, we factor this to get, $(s+2)(s+4)$. This yields zeros at $s=-2$ and $s=-4$. Had this function been more complicated, it might have been necessary to use the quadratic formula.

For poles, we must recognize that the transfer function will be infinite whenever the bottom part is zero. That is when s^2+2 is zero. To find this, we again look to factor the equation. This yields $\left(s+j\sqrt{2}\right)\left(s-j\sqrt{2}\right)$. This yields purely imaginary roots of $j\sqrt{2}$ and $-\left(j\sqrt{2}\right)$

Plotting this gives Figure 3.6 (Pole and Zero Plot)

Pole and Zero Plot

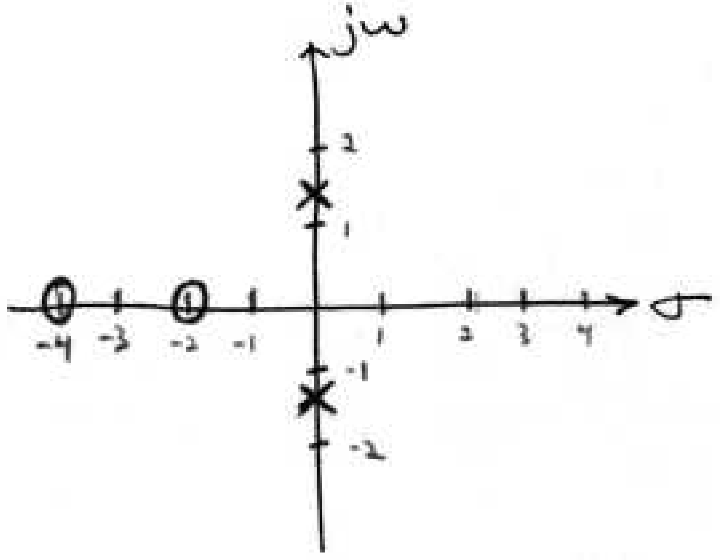


Figure 3.6: Sample pole-zero plot

Now that we have found and plotted the poles and zeros, we must ask what it is that this plot gives us. Basically what we can gather from this is that the magnitude of the transfer function will be larger when it is closer to the poles and smaller when it is closer to the zeros. This provides us with a qualitative understanding of what the system does at various frequencies and is crucial to the discussion of stability[8].

3.4.3 Repeated Poles and Zeros

It is possible to have more than one pole or zero at any given point. For instance, the discrete-time transfer function $H(z) = z^2$ will have two zeros at the origin and the continuous-time function $H(s) = \frac{1}{s^{25}}$ will have 25 poles at the origin.

[8]"BIBO Stability of Continuous Time Systems" <http://cnx.org/content/m10113/latest/>

3.4.4 Pole-Zero Cancellation

An easy mistake to make with regards to poles and zeros is to think that a function like $\frac{(s+3)(s-1)}{s-1}$ is the same as $s+3$. In theory they are equivalent, as the pole and zero at $s=1$ cancel each other out in what is known as **pole-zero cancellation**. However, think about what may happen if this were a transfer function of a system that was created with physical circuits. In this case, it is very unlikely that the pole and zero would remain in exactly the same place. A minor temperature change, for instance, could cause one of them to move just slightly. If this were to occur a tremendous amount of volatility is created in that area, since there is a change from infinity at the pole to zero at the zero in a very small range of signals. This is generally a very bad way to try to eliminate a pole. A much better way is to use **control theory** to move the pole to a better place.

3.5 Rational Functions and the Z-Transform[9]

3.5.1 Introduction

When dealing with operations on polynomials, the term **rational function** is a simple way to describe a particular relationship between two polynomials.

Definition 3.4: rational function
For any two polynomials, A and B, their quotient is called a rational function.
Example
Below is a simple example of a basic rational function, $f(x)$. Note that the numerator and denominator can be polynomials of any order, but the rational function is undefined when the denominator equals zero.

$$f(x) = \frac{x^2-4}{2x^2+x-3} \tag{3.43}$$

If you have begun to study the Z-transform (Section 3.2), you should have noticed by now they are all rational functions. Below we will look at some of the properties of rational functions and how they can be used to reveal important characteristics about a z-transform, and thus a signal or LTI system.

3.5.2 Properties of Rational Functions

In order to see what makes rational functions special, let us look at some of their basic properties and characteristics. If you are familiar with rational functions and basic algebraic properties, skip to the next section (Section 3.5.3: Rational Functions and the Z-Transform) to see how rational functions are useful when dealing with the z-transform.

3.5.2.1 Roots

To understand many of the following characteristics of a rational function, one must begin by finding the roots of the rational function. In order to do this, let us factor both of the polynomials so that the roots can be easily determined. Like all polynomials, the roots will provide us with information on many key properties. The function below shows the results of factoring the above rational function, (3.43).

$$f(x) = \frac{(x+2)(x-2)}{(2x+3)(x-1)} \tag{3.44}$$

Thus, the roots of the rational function are as follows:
Roots of the numerator are: $\{-2, 2\}$

[9]This content is available online at <http://cnx.org/content/m10593/2.8/>.

Roots of the denominator are: $\{-3, 1\}$

NOTE: In order to understand rational functions, it is essential to know and understand the roots that make up the rational function.

3.5.2.2 Discontinuities

Because we are dealing with division of two polynomials, we must be aware of the values of the variable that will cause the denominator of our fraction to be zero. When this happens, the rational function becomes undefined, *i.e.* we have a discontinuity in the function. Because we have already solved for our roots, it is very easy to see when this occurs. When the variable in the denominator equals any of the roots of the denominator, the function becomes undefined.

Example 3.5
Continuing to look at our rational function above, (3.43), we can see that the function will have discontinuities at the following points: $x = \{-3, 1\}$

In respect to the Cartesian plane, we say that the discontinuities are the values along the x-axis where the function is undefined. These discontinuities often appear as **vertical asymptotes** on the graph to represent the values where the function is undefined.

3.5.2.3 Domain

Using the roots that we found above, the **domain** of the rational function can be easily defined.

Definition 3.5: domain
The group, or set, of values that are defined by a given function.

Example
Using the rational function above, (3.43), the domain can be defined as any real number x where x does not equal 1 or negative 3. Written out mathematical, we get the following:

$$\{x \in \mathbb{R} \mid (x \neq -3) \quad and \quad (x \neq 1)\} \tag{3.45}$$

3.5.2.4 Intercepts

The **x-intercept** is defined as the point(s) where $f(x)$, *i.e.* the output of the rational functions, equals zero. Because we have already found the roots of the equation this process is very simple. From algebra, we know that the output will be zero whenever the numerator of the rational function is equal to zero. Therefore, the function will have an x-intercept wherever x equals one of the roots of the numerator.

The **y-intercept** occurs whenever x equals zero. This can be found by setting all the values of x equal to zero and solving the rational function.

3.5.3 Rational Functions and the Z-Transform

As we have stated above, all z-transforms can be written as rational functions, which have become the most common way of representing the z-transform. Because of this, we can use the properties above, especially those of the roots, in order to reveal certain characteristics about the signal or LTI system described by the z-transform.

Below is the general form of the z-transform written as a rational function:

$$X(z) = \frac{b_0 + b_1 z^{-1} + \dots + b_M z^{-M}}{a_0 + a_1 z^{-1} + \dots + a_N z^{-N}} \tag{3.46}$$

If you have already looked at the module about Understanding Pole/Zero Plots and the Z-transform (Section 3.8), you should see how the roots of the rational function play an important role in understanding the z-transform. The equation above, (3.46), can be expressed in factored form just as was done for the simple rational function above, see (3.44). Thus, we can easily find the roots of the numerator and denominator of the z-transform. The following two relationships become apparent:

Relationship of Roots to Poles and Zeros

- The roots of the numerator in the rational function will be the **zeros** of the z-transform
- The roots of the denominator in the rational function will be the **poles** of the z-transform

3.5.4 Conclusion

Once we have used our knowledge of rational functions to find its roots, we can manipulate a z-transform in a number of useful ways. We can apply this knowledge to representing an LTI system graphically through a Pole/Zero Plot (Section 3.8), or to analyze and design a digital filter through Filter Design from the Z-Transform (Section 3.10).

3.6 The Complex Plane[10]

3.6.1 Complex Plane

The **complex plane** provides a way to express complex numbers graphically. Any complex number can be expressed as a point on the complex plane. Before looking over the rest of this module, one should be very familiar with complex numbers. Please refer to the complex number[11] module for an explanation or review of these numbers.

Definition 3.6: Complex Plane
A two-dimensional graph where the horizontal axis maps the real part and the vertical axis maps the imaginary part of any complex number or function.

[10]This content is available online at <http://cnx.org/content/m10596/2.2/>.
[11]"Complex Numbers" <http://cnx.org/content/m0081/latest/>

Complex Plane

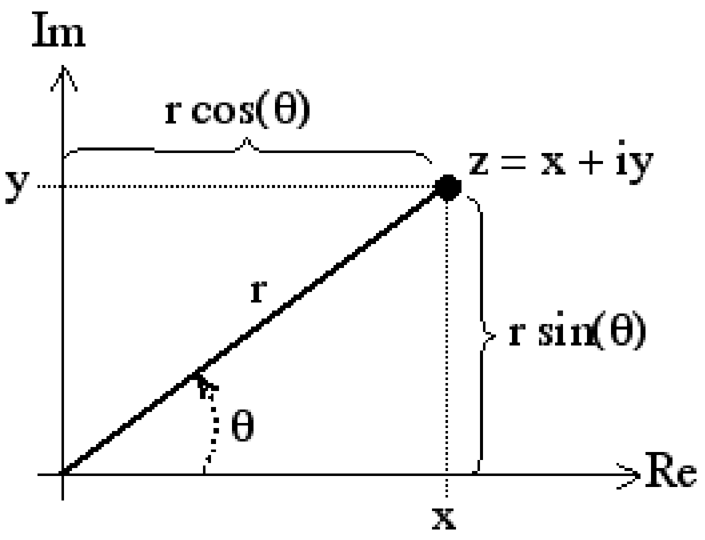


Figure 3.7: Plot of the complex number, z, as a point on the complex plane.

3.6.1.1 Rectangular Coordinates

Similar to the Cartesian plane, the complex plane allows one to plot ordered pairs of the form (a, b), where a and b are real numbers that describe a unique complex number through the following general form:

$$z = a + jb \tag{3.47}$$

This form is referred to as the **rectangular coordinate**.

3.6.1.2 Polar Form

The complex plane can also be used to plot complex numbers that are in **polar form**. Rather than using a and b, the polar coordinates use r and θ in their ordered pairs. The r is the distance from the origin to the complex number and θ is the angle of the complex number relative to the positive, real axis. Look at the figure above to see these variables displayed on the complex plane. The general form for polar numbers is as follows: $re^{j\theta}$

As a reminder, the following equations show the conversion between polar and rectangle coordinates:

$$r = \sqrt{a^2 + b^2} \tag{3.48}$$

$$\theta = \arctan\left(\frac{b}{a}\right) \tag{3.49}$$

3.7 Region of Convergence for the Z-transform[12]

3.7.1 Introduction

With the z-transform (Section 3.2), the s-plane represents a set of signals (complex exponentials[13]). For any given LTI[14] system, some of these signals may cause the output of the system to converge, while others cause the output to diverge ("blow up"). The set of signals that cause the system's output to converge lie in the **region of convergence (ROC)**. This module will discuss how to find this region of convergence for any discrete-time, LTI system.

3.7.2 The Region of Convergence

The region of convergence, known as the **ROC**, is important to understand because it defines the region where the z-transform (Section 3.2) exists. The **z-transform** of a sequence is defined as

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n} \tag{3.50}$$

The ROC for a given $x[n]$, is defined as the range of z for which the z-transform converges. Since the z-transform is a **power series**, it converges when $x[n] z^{-n}$ is absolutely summable. Stated differently,

$$\sum_{n=-\infty}^{\infty} |x[n] z^{-n}| < \infty \tag{3.51}$$

must be satisfied for convergence.

3.7.2.1 Properties of the Region of Convergencec

The Region of Convergence has a number of properties that are dependent on the characteristics of the signal, $x[n]$.

- **The ROC cannot contain any poles.** By definition a pole is a where $X(z)$ is infinite. Since $X(z)$ must be finite for all z for convergence, there cannot be a pole in the ROC.
- **If $x[n]$ is a finite-duration sequence, then the ROC is the entire z-plane, except possibly $z = 0$ or $|z| = \infty$.** A **finite-duration sequence** is a sequence that is nonzero in a finite interval $n_1 \leq n \leq n_2$. As long as each value of $x[n]$ is finite then the sequence will be absolutely summable. When $n_2 > 0$ there will be a z^{-1} term and thus the ROC will not include $z = 0$. When $n_1 < 0$ then the sum will be infinite and thus the ROC will not include $|z| = \infty$. On the other hand, when $n_2 \leq 0$ then the ROC will include $z = 0$, and when $n_1 \geq 0$ the ROC will include $|z| = \infty$. With these constraints, the only signal, then, whose ROC is the entire z-plane is $x[n] = c\delta[n]$.

[12]This content is available online at <http://cnx.org/content/m10622/2.8/>.
[13]"Continuous Time Complex Exponential" <http://cnx.org/content/m10060/latest/>
[14]"System Classifications and Properties" <http://cnx.org/content/m10084/latest/>

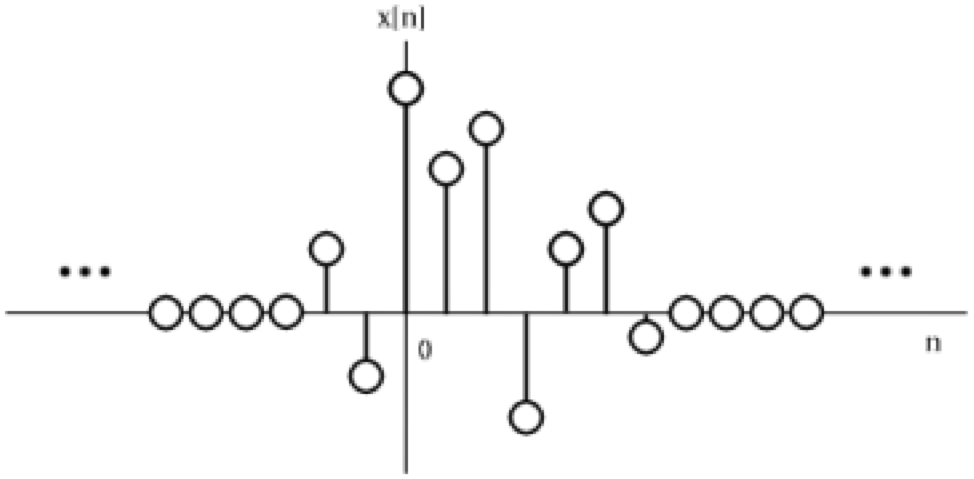


Figure 3.8: An example of a finite duration sequence.

The next properties apply to infinite duration sequences. As noted above, the z-transform converges when $|X(z)| < \infty$. So we can write

$$|X(z)| = |\sum_{n=-\infty}^{\infty} x[n] z^{-n}| \leq \sum_{n=-\infty}^{\infty} |x[n] z^{-n}| = \sum_{n=-\infty}^{\infty} |x[n]| (|z|)^{-n} \tag{3.52}$$

We can then split the infinite sum into positive-time and negative-time portions. So

$$|X(z)| \leq N(z) + P(z) \tag{3.53}$$

where

$$N(z) = \sum_{n=-\infty}^{-1} |x[n]| (|z|)^{-n} \tag{3.54}$$

and

$$P(z) = \sum_{n=0}^{\infty} |x[n]| (|z|)^{-n} \tag{3.55}$$

In order for $|X(z)|$ to be finite, $|x[n]|$ must be bounded. Let us then set

$$|x(n)| \leq C_1 {r_1}^n \tag{3.56}$$

for

$$n < 0$$

and

$$|x(n)| \leq C_2 {r_2}^n \tag{3.57}$$

for

$$n \geq 0$$

From this some further properties can be derived:

- **If $x[n]$ is a right-sided sequence, then the ROC extends outward from the outermost pole in $X(z)$.** A **right-sided sequence** is a sequence where $x[n] = 0$ for $n < n_1 < \infty$. Looking at the positive-time portion from the above derivation, it follows that

$$P(z) \leq C_2 \sum_{n=0}^{\infty} {r_2}^n (|z|)^{-n} = C_2 \sum_{n=0}^{\infty} \left(\frac{r_2}{|z|}\right)^n \tag{3.58}$$

Thus in order for this sum to converge, $|z| > r_2$, and therefore the ROC of a right-sided sequence is of the form $|z| > r_2$.

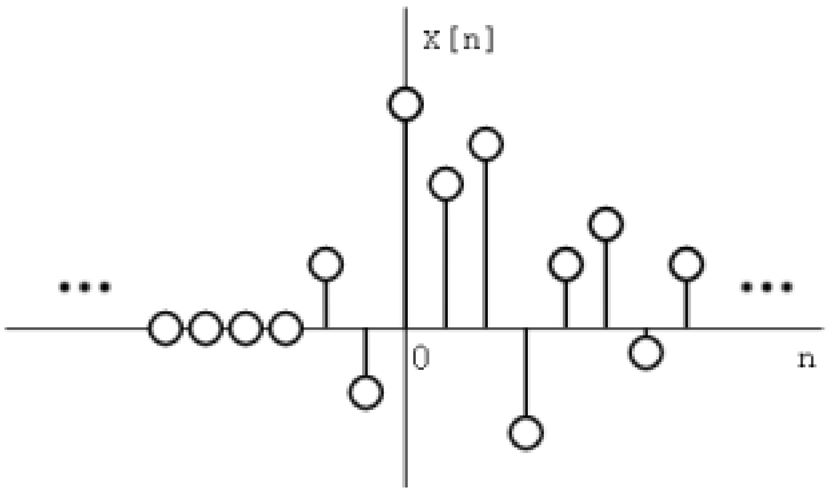


Figure 3.9: A right-sided sequence.

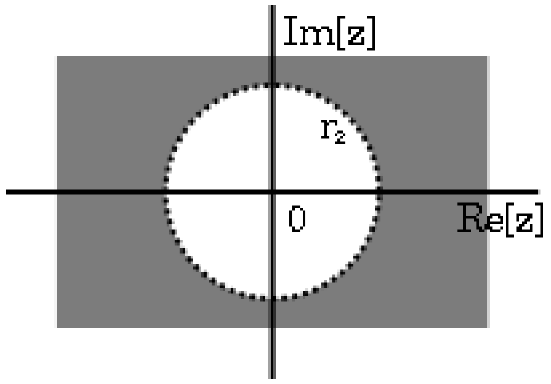


Figure 3.10: The ROC of a right-sided sequence.

- **If $x[n]$ is a left-sided sequence, then the ROC extends inward from the innermost pole in $X(z)$.** A **left-sided sequence** is a sequence where $x[n] = 0$ for $n > n_2 > -\infty$. Looking at the negative-time portion from the above derivation, it follows that

$$N(z) \leq C_1 \sum_{n=-\infty}^{-1} {r_1}^n (|z|)^{-n} = C_1 \sum_{n=-\infty}^{-1} \left(\frac{r_1}{|z|}\right)^n = C_1 \sum_{k=1}^{\infty} \left(\frac{|z|}{r_1}\right)^k \tag{3.59}$$

Thus in order for this sum to converge, $|z| < r_1$, and therefore the ROC of a left-sided sequence is of the form $|z| < r_1$.

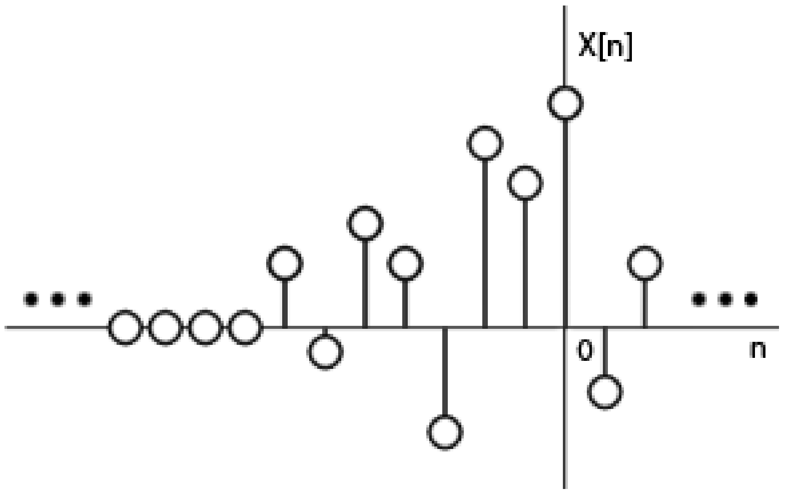


Figure 3.11: A left-sided sequence.

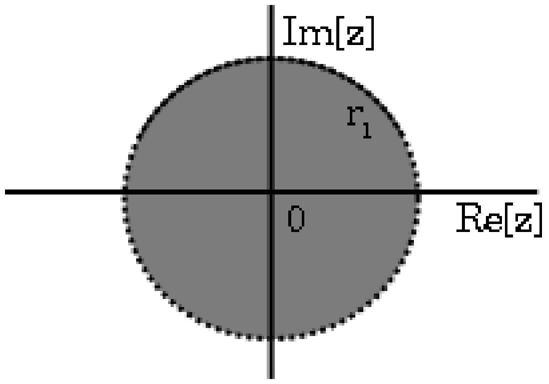


Figure 3.12: The ROC of a left-sided sequence.

- **If $x[n]$ is a two-sided sequence, the ROC will be a ring in the z-plane that is bounded on the interior and exterior by a pole.** A **two-sided sequence** is an sequence with infinite duration in the positive and negative directions. From the derivation of the above two properties, it follows that if $-\mathrm{r}_2 < |z| < r_2$ converges, then both the positive-time and negative-time portions converge and thus $X(z)$ converges as well. Therefore the ROC of a two-sided sequence is of the form $-\mathrm{r}_2 < |z| < r_2$.

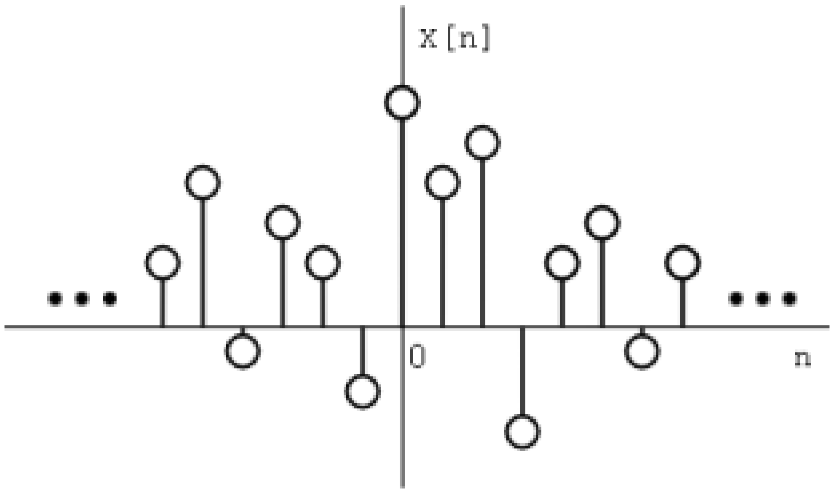


Figure 3.13: A two-sided sequence.

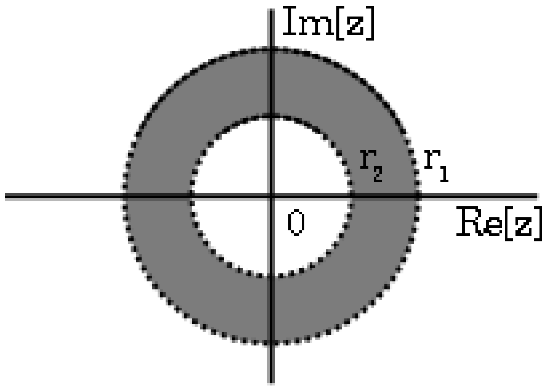


Figure 3.14: The ROC of a two-sided sequence.

3.7.2.2 Examples

Example 3.6

Lets take

$$x_1[n] = \left(\frac{1}{2}\right)^n u[n] + \left(\frac{1}{4}\right)^n u[n] \tag{3.60}$$

The z-transform of $\left(\frac{1}{2}\right)^n u[n]$ is $\frac{z}{z-\frac{1}{2}}$ with an ROC at $|z| > \frac{1}{2}$.

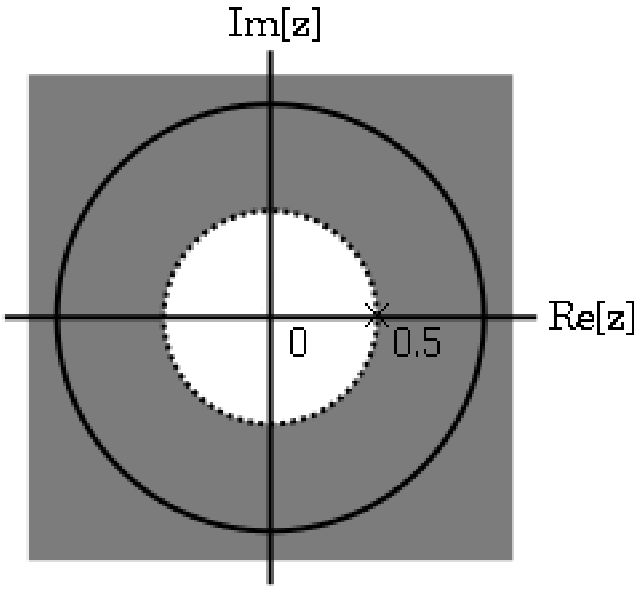


Figure 3.15: The ROC of $\left(\frac{1}{2}\right)^n u[n]$

The z-transform of $\left(\frac{-1}{4}\right)^n u[n]$ is $\frac{z}{z+\frac{1}{4}}$ with an ROC at $|z| > \frac{-1}{4}$.

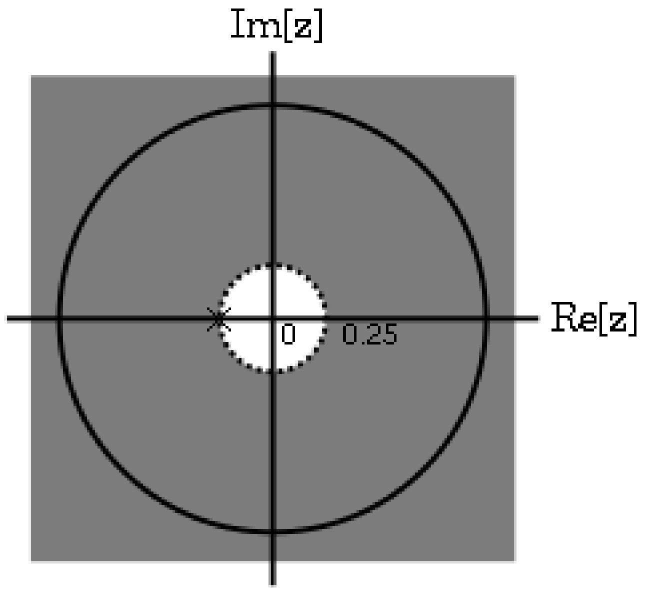


Figure 3.16: The ROC of $\left(\frac{-1}{4}\right)^n u[n]$

Due to linearity,

$$\begin{aligned} X_1[z] &= \frac{z}{z-\frac{1}{2}} + \frac{z}{z+\frac{1}{4}} \\ &= \frac{2z\left(z-\frac{1}{8}\right)}{\left(z-\frac{1}{2}\right)\left(z+\frac{1}{4}\right)} \end{aligned} \tag{3.61}$$

By observation it is clear that there are two zeros, at 0 and $\frac{1}{8}$, and two poles, at $\frac{1}{2}$, and $\frac{-1}{4}$. Following the obove properties, the ROC is $|z| > \frac{1}{2}$.

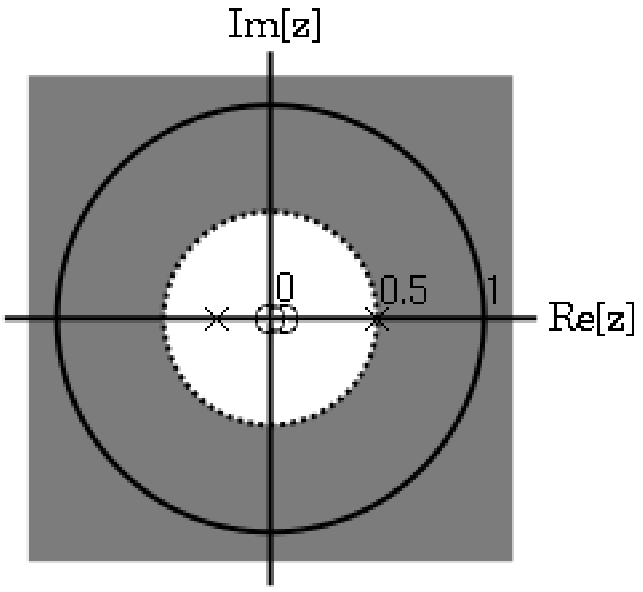


Figure 3.17: The ROC of $x_1[n] = \left(\frac{1}{2}\right)^n u[n] + \left(\frac{-1}{4}\right)^n u[n]$

Example 3.7
Now take

$$x_2[n] = \left(\frac{-1}{4}\right)^n u[n] - \left(\frac{1}{2}\right)^n u[(-n)-1] \tag{3.62}$$

The z-transform and ROC of $\left(\frac{-1}{4}\right)^n u[n]$ was shown in the example above (Example 3.6). The z-transorm of $\left(-\left(\frac{1}{2}\right)^n\right) u[(-n)-1]$ is $\frac{z}{z-\frac{1}{2}}$ with an ROC at $|z| > \frac{1}{2}$.

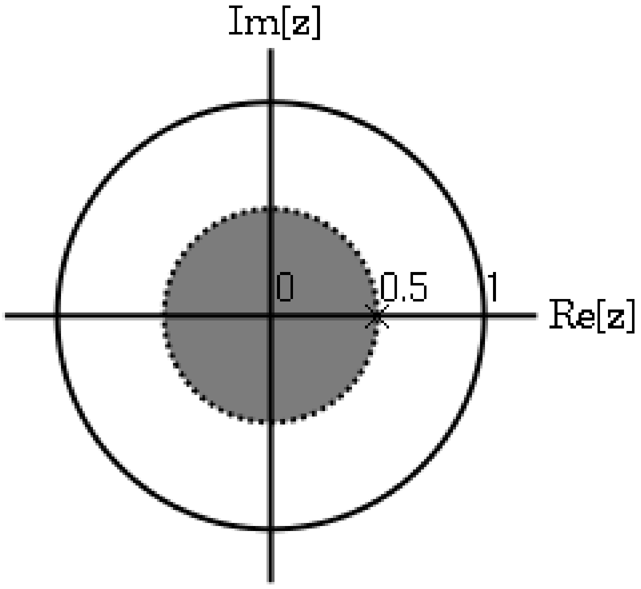


Figure 3.18: The ROC of $\left(-\left(\frac{1}{2}\right)^n\right) u\left[(-n)-1\right]$

Once again, by linearity,

$$\begin{aligned} X_2\left[z\right] &= \frac{z}{z+\frac{1}{4}}+\frac{z}{z-\frac{1}{2}} \\ &= \frac{z\left(2z-\frac{1}{8}\right)}{\left(z+\frac{1}{4}\right)\left(z-\frac{1}{2}\right)} \end{aligned} \qquad (3.63)$$

By observation it is again clear that there are two zeros, at 0 and $\frac{1}{16}$, and two poles, at $\frac{1}{2}$, and $\frac{-1}{4}$. in ths case though, the ROC is $|z| < \frac{1}{2}$.

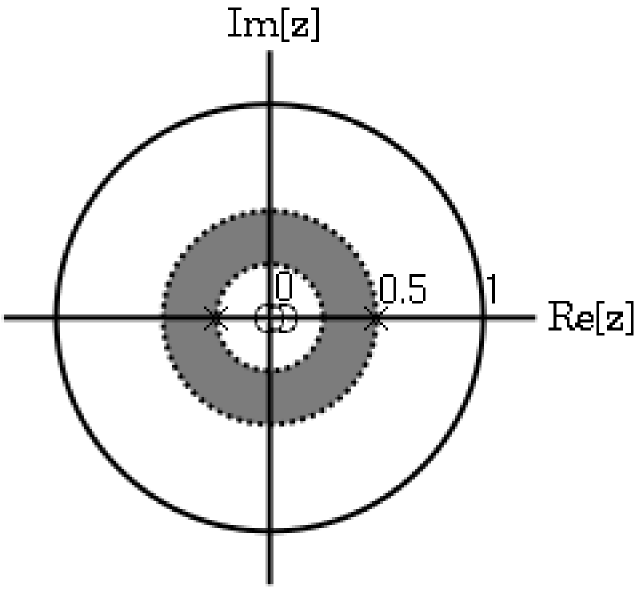


Figure 3.19: The ROC of $x_2[n] = \left(\frac{-1}{4}\right)^n u[n] - \left(\frac{1}{2}\right)^n u[(-n) - 1]$.

3.7.3 Graphical Understanding of ROC

Using the demonstration, learn about the region of convergence for the Laplace Transform.

3.7.4 Conclusion

Clearly, in order to craft a system that is actually useful by virtue of being causal and BIBO stable, we must ensure that it is within the Region of Convergence, which can be ascertained by looking at the pole zero plot. The Region of Convergence is the area in the pole/zero plot of the transfer function in which the function exists. For purposes of useful filter design, we prefer to work with rational functions, which can be described by two polynomials, one each for determining the poles and the zeros, respectively.

3.8 Understanding Pole/Zero Plots on the Z-Plane[15]

3.8.1 Introduction to Poles and Zeros of the Z-Transform

It is quite difficult to qualitatively analyze the Laplace transform[16] and Z-transform (Section 3.2), since mappings of their magnitude and phase or real part and imaginary part result in multiple mappings of 2-dimensional surfaces in 3-dimensional space. For this reason, it is very common to examine a plot of a transfer function's[17] poles and zeros to try to gain a qualitative idea of what a system does.

[15]This content is available online at <http://cnx.org/content/m10556/2.12/>.
[16]"The Laplace Transform" <http://cnx.org/content/m10110/latest/>
[17]"Transfer Functions" <http://cnx.org/content/m0028/latest/>

Once the Z-transform of a system has been determined, one can use the information contained in function's polynomials to graphically represent the function and easily observe many defining characteristics. The Z-transform will have the below structure, based on Rational Functions (Section 3.5):

$$X(z) = \frac{P(z)}{Q(z)} \tag{3.64}$$

The two polynomials, $P(z)$ and $Q(z)$, allow us to find the poles and zeros (Section 3.4) of the Z-Transform.

Definition 3.7: zeros
1. The value(s) for z where $P(z) = 0$.
2. The complex frequencies that make the overall gain of the filter transfer function zero.

Definition 3.8: poles
1. The value(s) for z where $Q(z) = 0$.
2. The complex frequencies that make the overall gain of the filter transfer function infinite.

Example 3.8
Below is a simple transfer function with the poles and zeros shown below it.

$$H(z) = \frac{z+1}{\left(z-\frac{1}{2}\right)\left(z+\frac{3}{4}\right)}$$

The zeros are: $\{-1\}$
The poles are: $\left\{\frac{1}{2}, -\frac{3}{4}\right\}$

3.8.2 The Z-Plane

Once the poles and zeros have been found for a given Z-Transform, they can be plotted onto the Z-Plane. The Z-plane is a complex plane with an imaginary and real axis referring to the complex-valued variable z. The position on the complex plane is given by $re^{j\theta}$ and the angle from the positive, real axis around the plane is denoted by θ. When mapping poles and zeros onto the plane, poles are denoted by an "x" and zeros by an "o". The below figure shows the Z-Plane, and examples of plotting zeros and poles onto the plane can be found in the following section.

Z-Plane

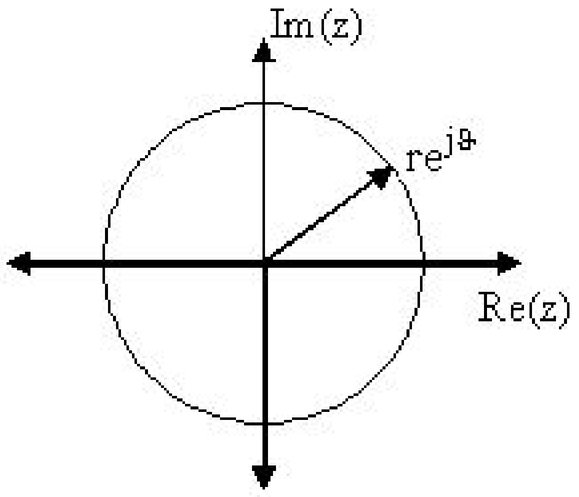


Figure 3.20

3.8.3 Examples of Pole/Zero Plots

This section lists several examples of finding the poles and zeros of a transfer function and then plotting them onto the Z-Plane.

Example 3.9: Simple Pole/Zero Plot

$$H(z) = \frac{z}{\left(z-\frac{1}{2}\right)\left(z+\frac{3}{4}\right)}$$

The zeros are: $\{0\}$
The poles are: $\left\{\frac{1}{2}, -\frac{3}{4}\right\}$

Pole/Zero Plot

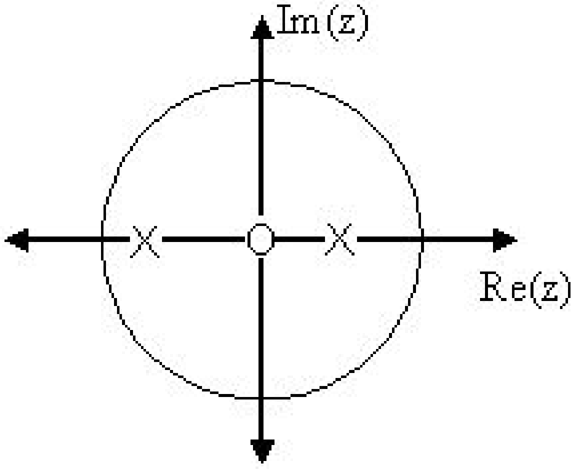


Figure 3.21: Using the zeros and poles found from the transfer function, the one zero is mapped to zero and the two poles are placed at $\frac{1}{2}$ and $-\frac{3}{4}$

Example 3.10: Complex Pole/Zero Plot

$$H(z) = \frac{(z-j)(z+j)}{\left(z-\left(\frac{1}{2}-\frac{1}{2}j\right)\right)\left(z-\frac{1}{2}+\frac{1}{2}j\right)}$$

The zeros are: $\{j, -j\}$
The poles are: $\left\{-1, \frac{1}{2}+\frac{1}{2}j, \frac{1}{2}-\frac{1}{2}j\right\}$

Pole/Zero Plot

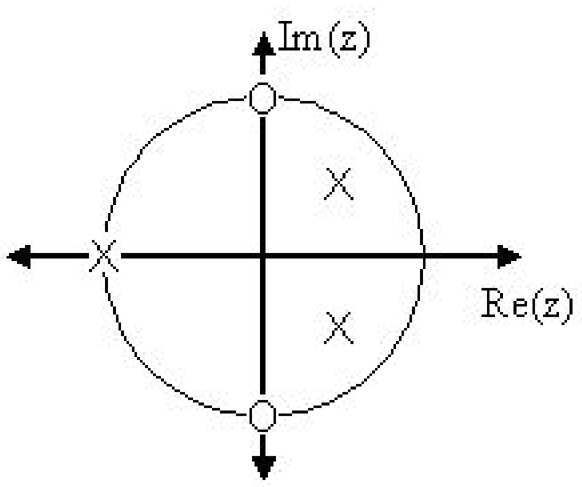


Figure 3.22: Using the zeros and poles found from the transfer function, the zeros are mapped to $\pm(j)$, and the poles are placed at -1, $\frac{1}{2}+\frac{1}{2}j$ and $\frac{1}{2}-\frac{1}{2}j$

Example 3.11: Pole-Zero Cancellation
An easy mistake to make with regards to poles and zeros is to think that a function like $\frac{(s+3)(s-1)}{s-1}$ is the same as $s+3$. In theory they are equivalent, as the pole and zero at $s=1$ cancel each other out in what is known as **pole-zero cancellation**. However, think about what may happen if this were a transfer function of a system that was created with physical circuits. In this case, it is very unlikely that the pole and zero would remain in exactly the same place. A minor temperature change, for instance, could cause one of them to move just slightly. If this were to occur a tremendous amount of volatility is created in that area, since there is a change from infinity at the pole to zero at the zero in a very small range of signals. This is generally a very bad way to try to eliminate a pole. A much better way is to use **control theory** to move the pole to a better place.

NOTE: It is possible to have more than one pole or zero at any given point. For instance, the discrete-time transfer function $H(z)=z^2$ will have two zeros at the origin and the continuous-time function $H(s)=\frac{1}{s^{25}}$ will have 25 poles at the origin.

MATLAB - If access to MATLAB is readily available, then you can use its functions to easily create pole/zero plots. Below is a short program that plots the poles and zeros from the above example onto the Z-Plane.

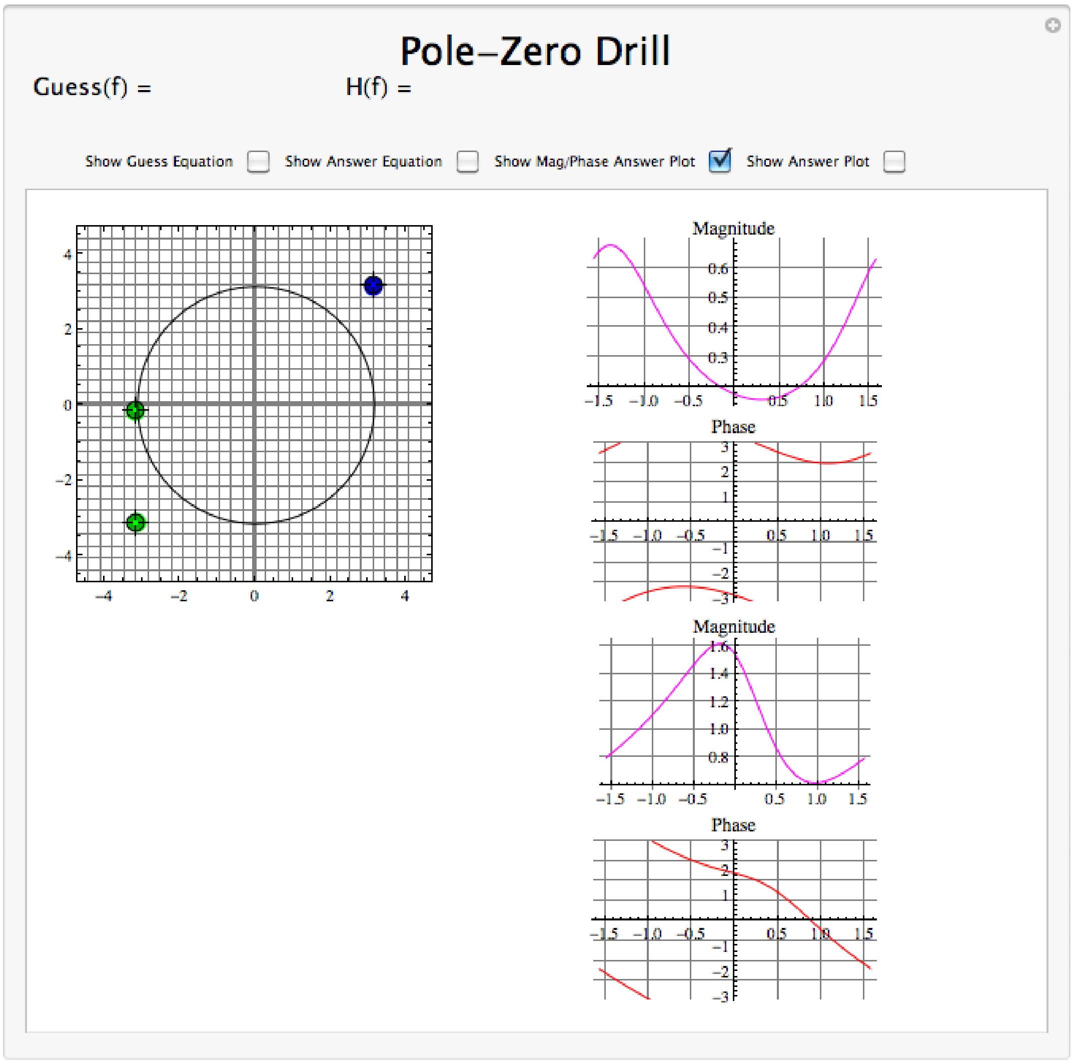
```
% Set up vector for zeros
z = [j ; -j];

% Set up vector for poles
p = [-1 ; .5+.5j ; .5-.5j];

figure(1);
```

```
zplane(z,p);
title('Pole/Zero Plot for Complex Pole/Zero Plot Example');
```

3.8.4 Interactive Demonstration of Poles and Zeros


Pole–Zero Drill

Guess(f) =

H(f) =

Show Guess Equation [ ] Show Answer Equation [ ] Show Mag/Phase Answer Plot [✓] Show Answer Plot [ ]

Figure 3.23: Interact (when online) with a Mathematica CDF demonstrating Pole/Zero Plots. To Download, right-click and save target as .cdf.

3.8.5 Applications for pole-zero plots

3.8.5.1 Stability and Control theory

Now that we have found and plotted the poles and zeros, we must ask what it is that this plot gives us. Basically what we can gather from this is that the magnitude of the transfer function will be larger when it is closer to the poles and smaller when it is closer to the zeros. This provides us with a qualitative understanding of what the system does at various frequencies and is crucial to the discussion of stability[18].

3.8.5.2 Pole/Zero Plots and the Region of Convergence

The region of convergence (ROC) for $X(z)$ in the complex Z-plane can be determined from the pole/zero plot. Although several regions of convergence may be possible, where each one corresponds to a different impulse response, there are some choices that are more practical. A ROC can be chosen to make the transfer function causal and/or stable depending on the pole/zero plot.

Filter Properties from ROC

- If the ROC extends outward from the outermost pole, then the system is **causal**.
- If the ROC includes the unit circle, then the system is **stable**.

Below is a pole/zero plot with a possible ROC of the Z-transform in the Simple Pole/Zero Plot (Example 3.9: Simple Pole/Zero Plot) discussed earlier. The shaded region indicates the ROC chosen for the filter. From this figure, we can see that the filter will be both causal and stable since the above listed conditions are both met.

Example 3.12

$$H(z) = \frac{z}{\left(z - \frac{1}{2}\right)\left(z + \frac{3}{4}\right)}$$

Region of Convergence for the Pole/Zero Plot

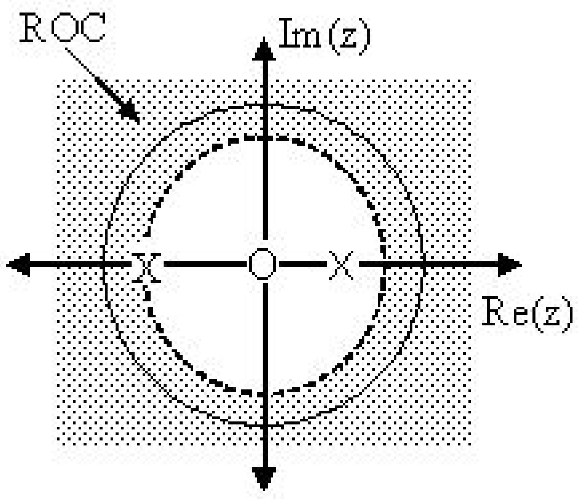


Figure 3.24: The shaded area represents the chosen ROC for the transfer function.

[18]"BIBO Stability of Continuous Time Systems" <http://cnx.org/content/m10113/latest/>

3.8.5.3 Frequency Response and Pole/Zero Plots

The reason it is helpful to understand and create these pole/zero plots is due to their ability to help us easily design a filter. Based on the location of the poles and zeros, the magnitude response of the filter can be quickly understood. Also, by starting with the pole/zero plot, one can design a filter and obtain its transfer function very easily.

3.9 Zero Locations of Linear-Phase FIR Filters[19]

3.9.1 Zero Locations of Linear-Phase Filters

The zeros of the transfer function $H(z)$ of a linear-phase filter lie in specific configurations.

We can write the symmetry condition

$$h(n) = h(N - (1 - n))$$

in the Z domain. Taking the Z-transform of both sides gives

$$H(z) = z^{-(N-1)} H\left(\frac{1}{z}\right) \tag{3.65}$$

Recall that we are assuming that $h(n)$ is real-valued. If z_0 is a zero of $H(z)$,

$$H(z_0) = 0$$

then

$$H({z_0}^*) = 0$$

(Because the roots of a polynomial with real coefficients exist in complex-conjugate pairs.)

Using the symmetry condition, (3.65), it follows that

$$H(z_0) = z^{-(N-1)} H\left(\frac{1}{z_0}\right) = 0$$

and

$$H({z_0}^*) = z^{-(N-1)} H\left(\frac{1}{{z_0}^*}\right) = 0$$

or

$$H\left(\frac{1}{z_0}\right) = H\left(\frac{1}{{z_0}^*}\right) = 0$$

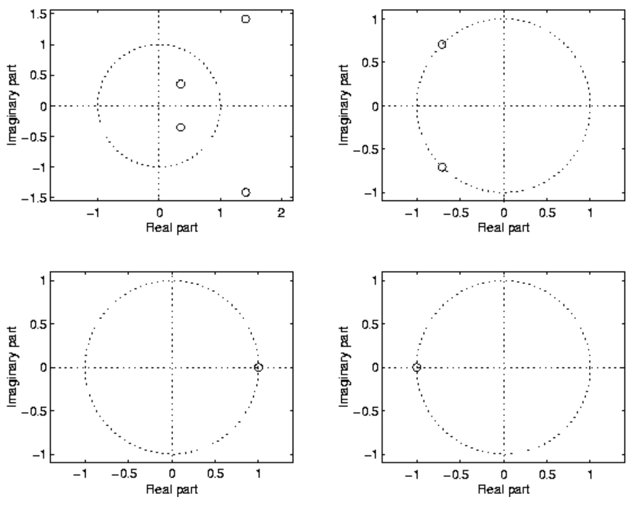
NOTE: If z_0 is a zero of a (real-valued) linear-phase filter, then so are ${z_0}^*$, $\frac{1}{z_0}$, and $\frac{1}{{z_0}^*}$.

3.9.2 ZEROS LOCATIONS

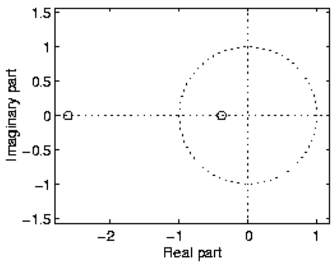
It follows that

1. generic zeros of a linear-phase filter exist in sets of 4.
2. zeros on the unit circle ($z_0 = e^{j\omega_0}$) exist in sets of 2. ($z_0 \neq \pm(1)$)
3. zeros on the real line ($z_0 = a$) exist in sets of 2. ($z_0 \neq \pm(1)$)
4. zeros at 1 and -1 do not imply the existence of zeros at other specific points.

[19]This content is available online at <http://cnx.org/content/m10700/2.2/>.



(a)



(b)

Figure 3.25: Examples of zero sets

3.9.3 ZERO LOCATIONS: AUTOMATIC ZEROS

The frequency response $H^f(\omega)$ of a Type II FIR filter always has a zero at $\omega = \pi$:

$$h(n) = [h_0, h_1, h_2, h_2, h_1, h_0]$$

$$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + h_2 z^{-3} + h_1 z^{-4} + h_0 z^{-5}$$

$$H(-1) = h_0 - h_1 + h_2 - h_2 + h_1 - h_0 = 0$$

$$H^f(\pi) = H\left(e^{j\pi}\right) = H(-1) = 0$$

NOTE: $H^f(\pi) = 0$ always for Type II filters.

Similarly, we can derive the following rules for Type III and Type IV FIR filters.

NOTE: $H^f(0) = H^f(\pi) = 0$ always for Type III filters.

NOTE: $H^f(0) = 0$ always for Type IV filters.

The automatic zeros can also be derived using the characteristics of the amplitude response $A(\omega)$ seen earlier.

Type	**automatic zeros**
I	—
II	$\omega = \pi$
III	$\omega = 0 \;\; or \;\; \pi$
IV	$\omega = 0$

Table 3.2

3.9.4 ZERO LOCATIONS: EXAMPLES

The Matlab command `zplane` can be used to plot the zero locations of FIR filters.

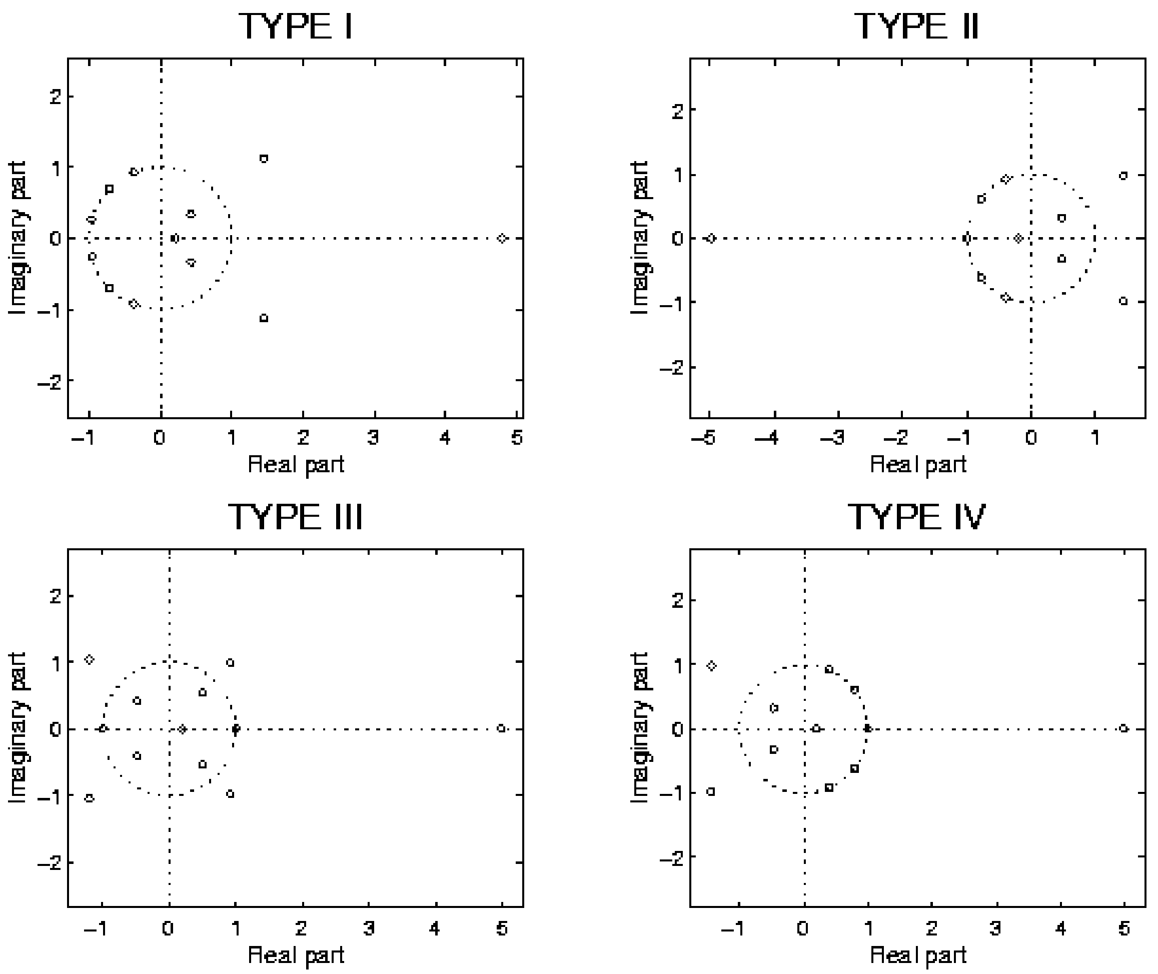


Figure 3.26

Note that the zero locations satisfy the properties noted previously.

3.10 Discrete Time Filter Design[20]

3.10.1 Estimating Frequency Response from Z-Plane

One of the primary motivating factors for utilizing the z-transform and analyzing the pole/zero plots is due to their relationship to the frequency response of a discrete-time system. Based on the position of the poles and zeros, one can quickly determine the frequency response. This is a result of the correspondence between the frequency response and the transfer function evaluated on the unit circle in the pole/zero plots. The

[20]This content is available online at <http://cnx.org/content/m10548/2.10/>.

frequency response, or DTFT, of the system is defined as:

$$\begin{aligned} H(w) &= H(z)\,|_{z,z=e^{jw}} \\ &= \frac{\sum_{k=0}^{M} b_k e^{-(jwk)}}{\sum_{k=0}^{N} a_k e^{-(jwk)}} \end{aligned} \tag{3.66}$$

Next, by factoring the transfer function into poles and zeros and multiplying the numerator and denominator by e^{jw} we arrive at the following equations:

$$H(w) = |\frac{b_0}{a_0}| \frac{\prod_{k=1}^{M} |e^{jw} - c_k|}{\prod_{k=1}^{N} |e^{jw} - d_k|} \tag{3.67}$$

From (3.67) we have the frequency response in a form that can be used to interpret physical characteristics about the filter's frequency response. The numerator and denominator contain a product of terms of the form $|e^{jw} - h|$, where h is either a zero, denoted by c_k or a pole, denoted by d_k. Vectors are commonly used to represent the term and its parts on the complex plane. The pole or zero, h, is a vector from the origin to its location anywhere on the complex plane and e^{jw} is a vector from the origin to its location on the unit circle. The vector connecting these two points, $|e^{jw} - h|$, connects the pole or zero location to a place on the unit circle dependent on the value of w. From this, we can begin to understand how the magnitude of the frequency response is a ratio of the distances to the poles and zero present in the z-plane as w goes from zero to pi. These characteristics allow us to interpret $|H(w)|$ as follows:

$$|H(w)| = |\frac{b_0}{a_0}| \frac{\prod \text{"distances from zeros"}}{\prod \text{"distances from poles"}} \tag{3.68}$$

3.10.2 Drawing Frequency Response from Pole/Zero Plot

Let us now look at several examples of determining the magnitude of the frequency response from the pole/zero plot of a z-transform. If you have forgotten or are unfamiliar with pole/zero plots, please refer back to the Pole/Zero Plots (Section 3.8) module.

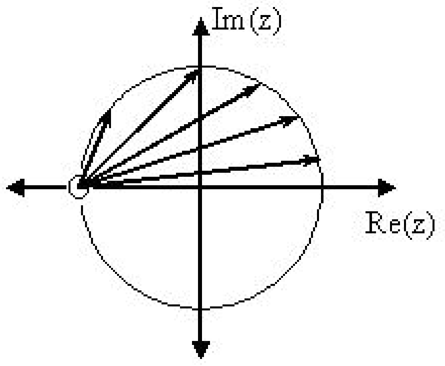
Example 3.13

In this first example we will take a look at the very simple z-transform shown below:

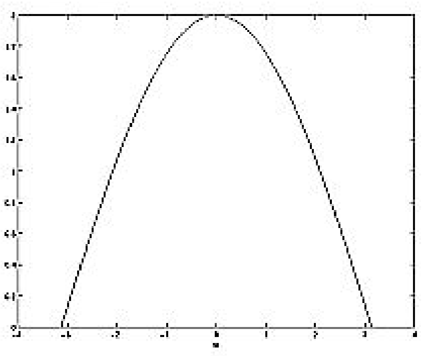
$$H(z) = z + 1 = 1 + z^{-1}$$

$$H(w) = 1 + e^{-(jw)}$$

For this example, some of the vectors represented by $|e^{jw} - h|$, for random values of w, are explicitly drawn onto the complex plane shown in the figure below. These vectors show how the amplitude of the frequency response changes as w goes from 0 to 2π, and also show the physical meaning of the terms in (3.67) above. One can see that when $w = 0$, the vector is the longest and thus the frequency response will have its largest amplitude here. As w approaches π, the length of the vectors decrease as does the amplitude of $|H(w)|$. Since there are no poles in the transform, there is only this one vector term rather than a ratio as seen in (3.67).



(a) Pole/Zero Plot



(b) Frequency Response: |H(w)|

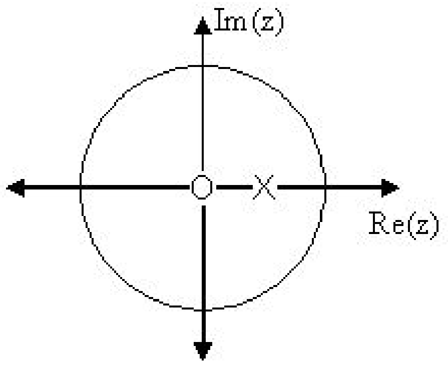
Figure 3.27: The first figure represents the pole/zero plot with a few representative vectors graphed while the second shows the frequency response with a peak at +2 and graphed between plus and minus π.

Example 3.14
For this example, a more complex transfer function is analyzed in order to represent the system's frequency response.

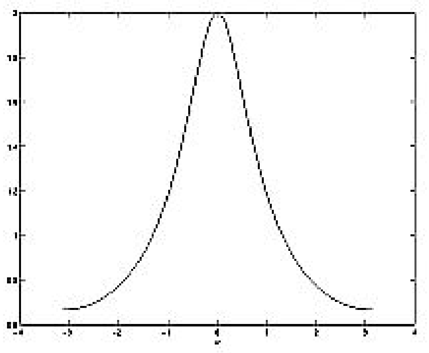
$$H(z) = \frac{z}{z - \frac{1}{2}} = \frac{1}{1 - \frac{1}{2}z^{-1}}$$

$$H(w) = \frac{1}{1 - \frac{1}{2}e^{-(jw)}}$$

Below we can see the two figures described by the above equations. The Figure 3.28(a) (Pole/Zero Plot) represents the basic pole/zero plot of the z-transform, $H(w)$. Figure 3.28(b) (Frequency Response: |H(w)|) shows the magnitude of the frequency response. From the formulas and statements in the previous section, we can see that when $w = 0$ the frequency will peak since it is at this value of w that the pole is closest to the unit circle. The ratio from (3.67) helps us see the mathematics behind this conclusion and the relationship between the distances from the unit circle and the poles and zeros. As w moves from 0 to π, we see how the zero begins to mask the effects of the pole and thus force the frequency response closer to 0.



(a) Pole/Zero Plot



(b) Frequency Response: |H(w)|

Figure 3.28: The first figure represents the pole/zero plot while the second shows the frequency response with a peak at +2 and graphed between plus and minus π.

3.10.3 Interactive Filter Design Illustration

This media object is a LabVIEW VI. Please view or download it at
<DFD_Utility.llb>

Figure 3.29: Digital filter design LabVIEW virtual instrument by NI from http://cnx.org/content/m13115/latest/[21].

3.10.4 Conclusion

In conclusion, using the distances from the unit circle to the poles and zeros, we can plot the frequency response of the system. As w goes from 0 to 2π, the following two properties, taken from the above equations, specify how one should draw $|H(w)|$.

While moving around the unit circle...

1. if close to a zero, then the magnitude is small. If a zero is on the unit circle, then the frequency response is zero at that point.
2. if close to a pole, then the magnitude is large. If a pole is on the unit circle, then the frequency response goes to infinity at that point.

[21]http://cnx.org/content/m13115/latest/

Chapter 4

Filter Design II

4.1 Bilinear Transform[1]

There **is** a way that we can make things a good bit easier for ourselves however. The only drawback is that we have to do some complex analysis first, and look at a **bilinear transform**! Let's do one more substitution, and define another complex vector, which we can call $r(s)$:

$$r(s) \equiv |\Gamma_\nu| e^{j(\theta_r - 2\beta s)} \quad (4.1)$$

The vector $r(s)$ is just the rotating part of the crank diagram which we have been looking at Figure 4.1 (The Vector r(s)). It has a magnitude equal to that of the reflection coefficient, and it rotates around at a rate $2\beta s$ as we move down the line. For every $r(s)$ there is a corresponding $Z(s)$ which is given by:

$$Z(s) = Z_0 \frac{1 + r(s)}{1 - r(s)} \quad (4.2)$$

The Vector r(s)

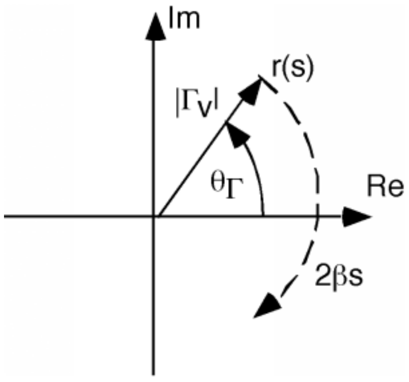


Figure 4.1

Now, it turns out to be easier if we talk about a **normalized impedance**, which we get by dividing

[1]This content is available online at <http://cnx.org/content/m1057/2.13/>.

$Z(s)$ by Z_0.

$$\frac{Z(s)}{Z_0} = \frac{1 + r(s)}{1 - r(s)} \tag{4.3}$$

which we can solve for $r(s)$

$$r(s) = \frac{\frac{Z(s)}{Z_0} - 1}{\frac{Z(s)}{Z_0} + 1} \tag{4.4}$$

This relationship is called a **bilinear transform**. For every $r(s)$ that we can imagine, there is one and only one $\frac{Z(s)}{Z_0}$ and for every $\frac{Z(s)}{Z_0}$ there is one and only one $r(s)$. What we would like to be able to do, is find $\frac{Z(s)}{Z_0}$, given an $r(s)$. The reason for this should be readily apparent. Whereas, as we move along in s, $\frac{Z(s)}{Z_0}$ behaves in a most difficult manner (dividing one phasor by another), $r(s)$ simply rotates around on the complex plane. Given one $r(s_0)$ it is **easy** to find another $r(s)$. We just rotate around!

We shall find the required relationship in a graphical manner. Suppose I have a complex plane, representing $\frac{Z(s)}{Z_0}$. And then suppose I have some point "A" on that plane and I want to know what impedance it represents. I just read along the two axes, and find that, for the example in Figure 4.2 (The Complex Impedance Plane), "A" represents an impedance of $\frac{Z(s)}{Z_0} = 4 + 2j$. What I would like to do would be to get a grid similar to that on the $\frac{Z(s)}{Z_0}$plane, but on the $r(s)$ plane instead. That way, if I knew one impedence (say $\frac{Z(0)}{Z_0} = \frac{Z_L}{Z_0}$ then I could find any other impedance, at any other s, by simply rotating $r(s)$around by $2\beta s$, and then reading off the new $\frac{Z(s)}{Z_0}$ from the grid I had developed. This is what we shall attempt to do.

The Complex Impedance Plane

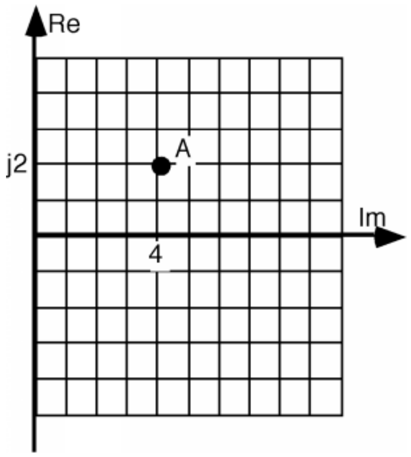


Figure 4.2

Let's start with (4.4) and re-write it as:

$$\begin{aligned} r(s) &= \frac{\frac{Z(s)}{Z_0}+1-2}{\frac{Z(s)}{Z_0}+1} \\ &= 1 + \frac{-2}{\frac{Z(s)}{Z_0}+1} \end{aligned} \tag{4.5}$$

In order to use (4.5), we are going to have to interpret it in a way which might seem a little odd to you. The way we will read the equation is to say: "Take $\frac{Z(s)}{Z_0}$ and add 1 to it. Invert what you get, and multiply by -2. Then add 1 to the result." Simple isn't it? The only hard part we have in doing this is inverting $\frac{Z(s)}{Z_0}+1$. This, it turns out, is pretty easy once we learn one very important fact.

The **one** fact about algebra on the complex plane that we need is as follows. Consider a vertical line, s, on the complex plane, located a distance d away from the imaginary axis Figure 4.3 (A Vertical Line, s, a Distance, d, Away From the Imaginary Axis). There are a lot of ways we could express the line s, but we will choose one which will turn out to be convenient for us. Let's let:

$$s = d\left(1 - j\tan\left(\phi\right)\right)\phi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \tag{4.6}$$

A Vertical Line, s, a Distance, d, Away From the Imaginary Axis

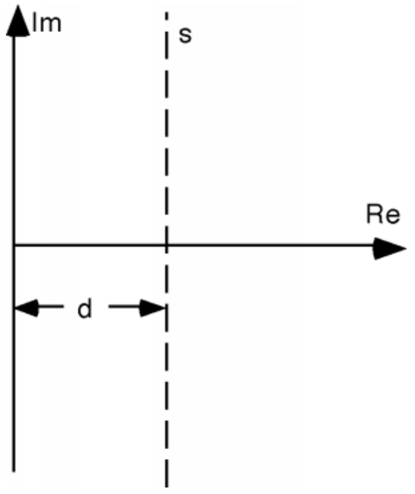


Figure 4.3

Now we ask ourselves the question: what is the inverse of s?

$$\frac{1}{s} = \frac{1}{d}\frac{1}{1 - j\tan\left(\phi\right)} \tag{4.7}$$

We can substitute for $\tan\left(\phi\right)$:

$$\begin{aligned}\frac{1}{s} &= \frac{1}{d}\frac{1}{1-j\frac{\sin(\phi)}{\cos(\phi)}} \\ &= \frac{1}{d}\frac{\cos(\phi)}{\cos(\phi)-j\sin(\phi)}\end{aligned} \tag{4.8}$$

And then, since $\cos\left(\phi\right) - j\sin\left(\phi\right) = e^{-(j\phi)}$

$$\begin{aligned}\frac{1}{s} &= \frac{1}{d}\frac{\cos(\phi)}{e^{-(j\phi)}} \\ &= \frac{1}{d}\cos\left(\phi\right)e^{j\phi}\end{aligned} \tag{4.9}$$

A Plot of 1/s

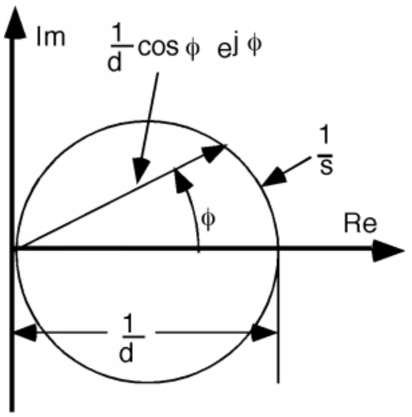


Figure 4.4

A careful look at Figure 4.4 (A Plot of 1/s) should allow you to convince yourself that (4.9) is an equation for a circle on the complex plane, with a diameter $\frac{1}{d}$. If s is not parallel to the imaginary axis, but rather has its perpendicular to the origin at some angle ϕ, to make a line s' Figure 4.5 (The Line s'). Since $s' = se^{j\phi}$, taking $\frac{1}{s}$ simply will give us a circle with a diameter of $\frac{1}{d}$, which has been rotated by an angle ϕ from the real axis Figure 4.6 (Inverse of a Rotated Line). And so we come to the **one** fact we have to keep in mind: **"The inverse of a straight line on the complex plane is a circle, whose diameter is the inverse of the distance between the line and the origin."**

The Line s'

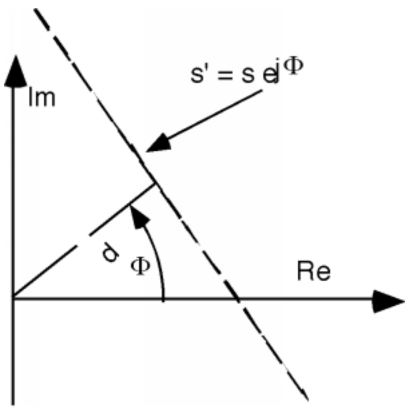


Figure 4.5: The line s multiplied by $e^{j\phi}$

Inverse of a Rotated Line

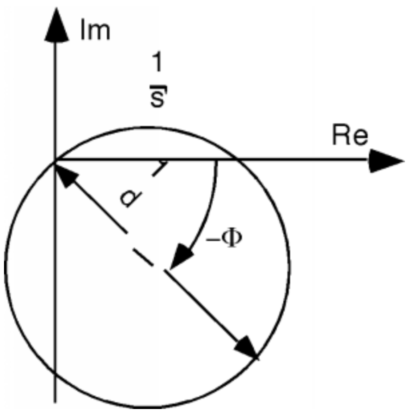


Figure 4.6

Chapter 5

Filter Design III

5.1 Linear-Phase FIR Filters[1]

5.1.1 THE AMPLITUDE RESPONSE

If the real and imaginary parts of $H^f(\omega)$ are given by

$$H^f(\omega) = Re(\omega) + j\text{Im}(\omega) \tag{5.1}$$

the magnitude and phase are defined as

$$|H^f(\omega)| = \sqrt{(Re(\omega))^2 + (\text{Im}(\omega))^2}$$

$$p(\omega) = \arctan\left(\frac{\text{Im}(\omega)}{Re(\omega)}\right)$$

so that

$$H^f(\omega) = |H^f(\omega)|e^{jp(\omega)} \tag{5.2}$$

With this definition, $|H^f(\omega)|$ is never negative and $p(\omega)$ is usually discontinuous, but it can be very helpful to write $H^f(\omega)$ as

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)} \tag{5.3}$$

where $A(\omega)$ can be positive and negative, and $\theta(\omega)$ continuous. $A(\omega)$ is called the **amplitude response**. Figure 5.1 illustrates the difference between $|H^f(\omega)|$ and $A(\omega)$.

[1]This content is available online at <http://cnx.org/content/m10705/2.3/>.

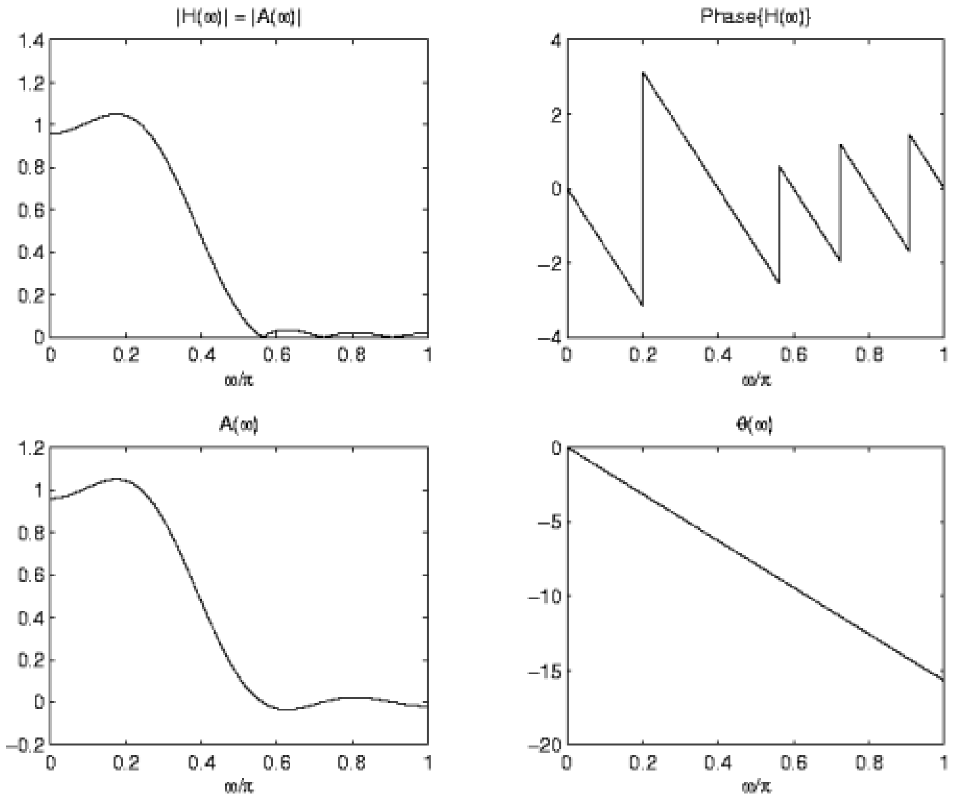


Figure 5.1

A linear-phase phase filter is one for which the continuous phase $\theta(\omega)$ is linear.

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)}$$

with

$$\theta(\omega) = M\omega + B$$

We assume in the following that the impulse response $h(n)$ is real-valued.

5.1.2 WHY LINEAR-PHASE?

If a discrete-time cosine signal

$$x_1(n) = \cos(\omega_1 n + \phi_1)$$

is processed through a discrete-time filter with frequency response

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)}$$

then the output signal is given by

$$y_1(n) = A(\omega_1)\cos(\omega_1 n + \phi_1 + \theta(\omega_1))$$

or

$$y_1(n) = A(\omega_1)\cos\left(\omega_1\left(n + \frac{\theta(\omega_1)}{\omega_1}\right) + \phi_1\right)$$

The LTI system has the effect of scaling the cosine signal and delaying it by $\frac{\theta(\omega_1)}{\omega_1}$.

Exercise 5.1.1 ***(Solution on p. 156.)***

When does the system delay cosine signals with different frequencies by the same amount?

The function $\frac{\theta(\omega)}{\omega}$ is called the **phase delay**. A linear phase filter therefore has constant phase delay.

5.1.3 WHY LINEAR-PHASE: EXAMPLE

Consider a discrete-time filter described by the difference equation

$$y(n) = 0.1821x(n) + 0.7865x(n-1) - 0.6804x(n-2) + x(n-3) + 0.6804y(n-1) - 0.7865y(n-2) + 0.1821y(n-3) \quad (5.4)$$

When $\omega_1 = 0.31\pi$, then the delay is $\frac{-\theta(\omega_1)}{\omega_1} = 2.45$. The delay is illustrated in Figure 5.2:

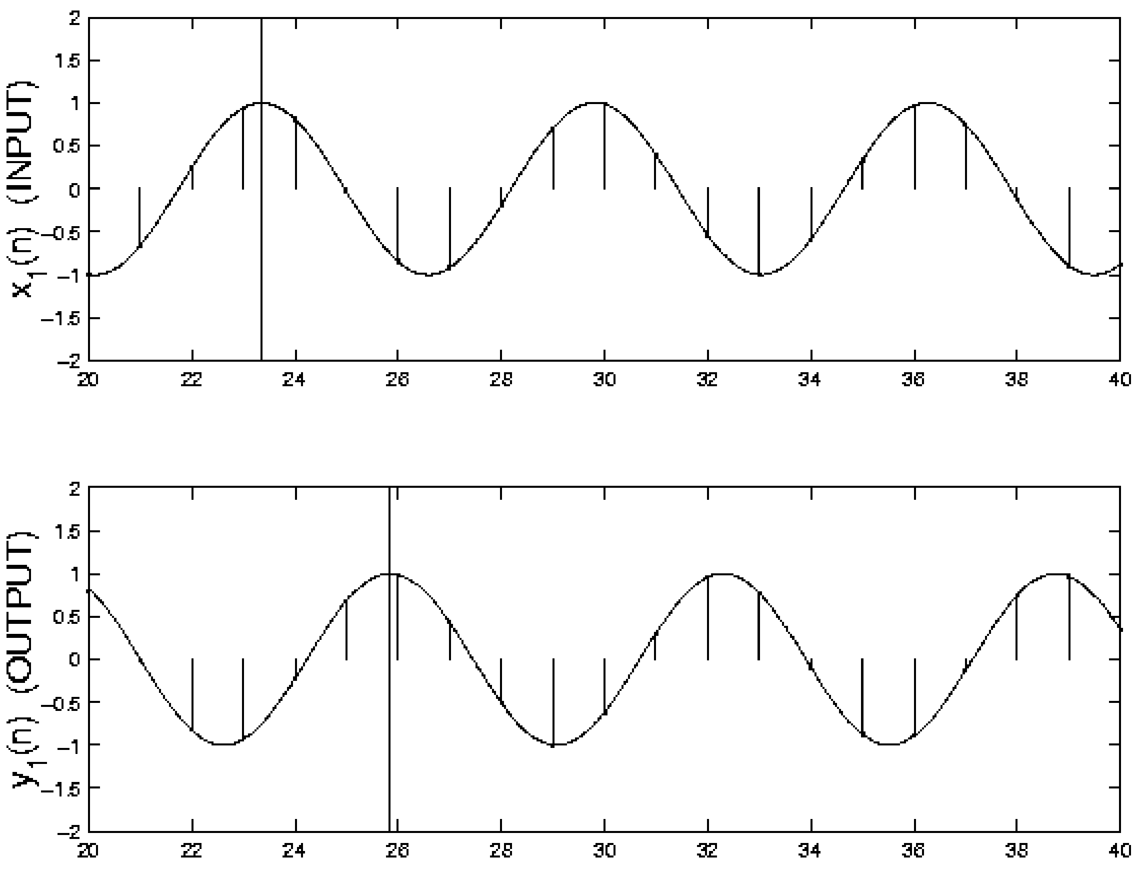


Figure 5.2

Notice that the delay is fractional — the discrete-time samples are not exactly reproduced in the output. The fractional delay can be interpreted in this case as a delay of the underlying continuous-time cosine signal.

5.1.4 WHY LINEAR-PHASE: EXAMPLE (2)

Consider the same system given on the previous slide, but let us change the frequency of the cosine signal.

When $\omega_2 = 0.47\pi$, then the delay is $\frac{-\theta(\omega_2)}{\omega_2} = 0.14$.

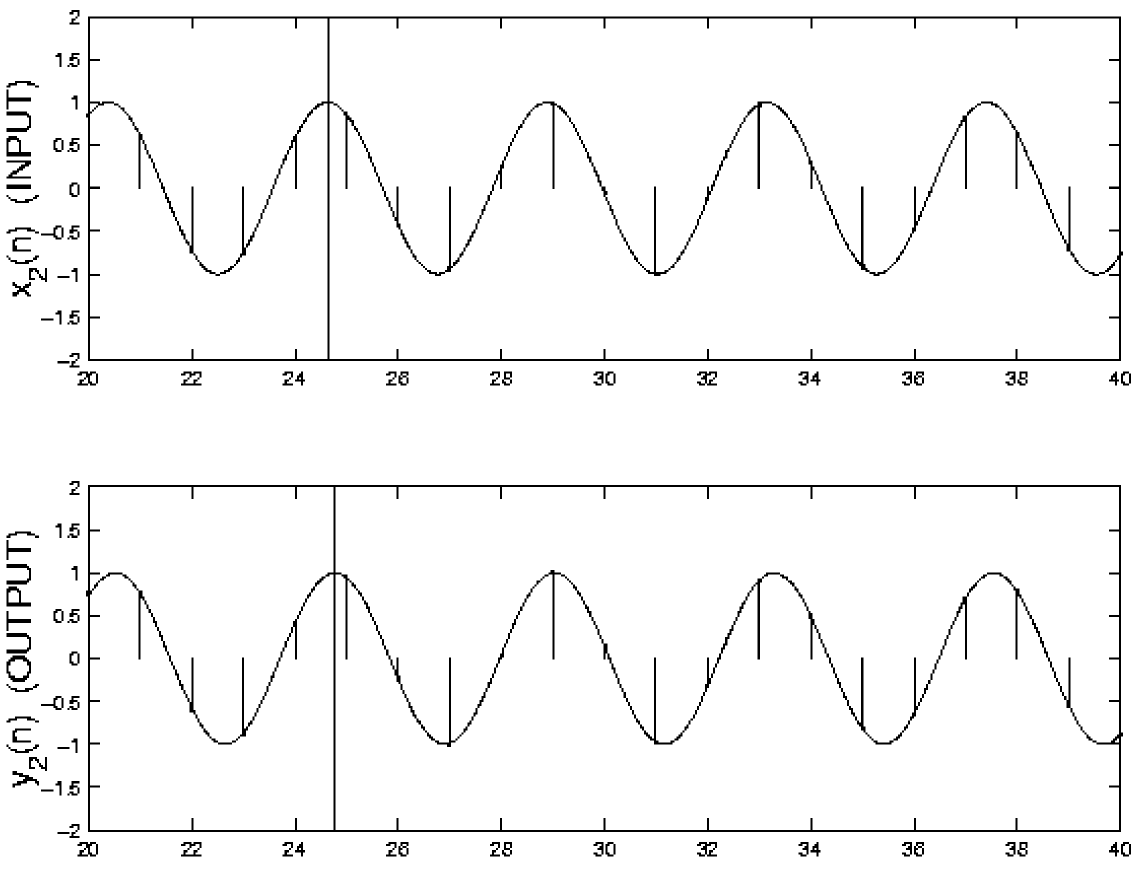


Figure 5.3

NOTE: For this example, the delay depends on the frequency, because this system does not have linear phase.

5.1.5 WHY LINEAR-PHASE: MORE

From the previous slides, we see that a filter will delay different frequency components of a signal by the same amount if the filter has linear phase (constant phase delay).

In addition, when a narrow band signal (as in AM modulation) goes through a filter, the envelop will be delayed by the **group delay** or **envelop delay** of the filter. The amount by which the envelop is delayed is independent of the carrier frequency only if the filter has linear phase.

Also, in applications like image processing, filters with non-linear phase can introduce artifacts that are visually annoying.

5.2 Four Types of Linear-Phase FIR Filters[2]

5.2.1 FOUR TYPES OF LINEAR-PHASE FIR FILTERS

Linear-phase FIR filter can be divided into four basic types.

Type	**impulse response**	
I	symmetric	length is odd
II	symmetric	length is even
III	anti-symmetric	length is odd
IV	anti-symmetric	length is even

Table 5.1

[2]This content is available online at <http://cnx.org/content/m10706/2.2/>.

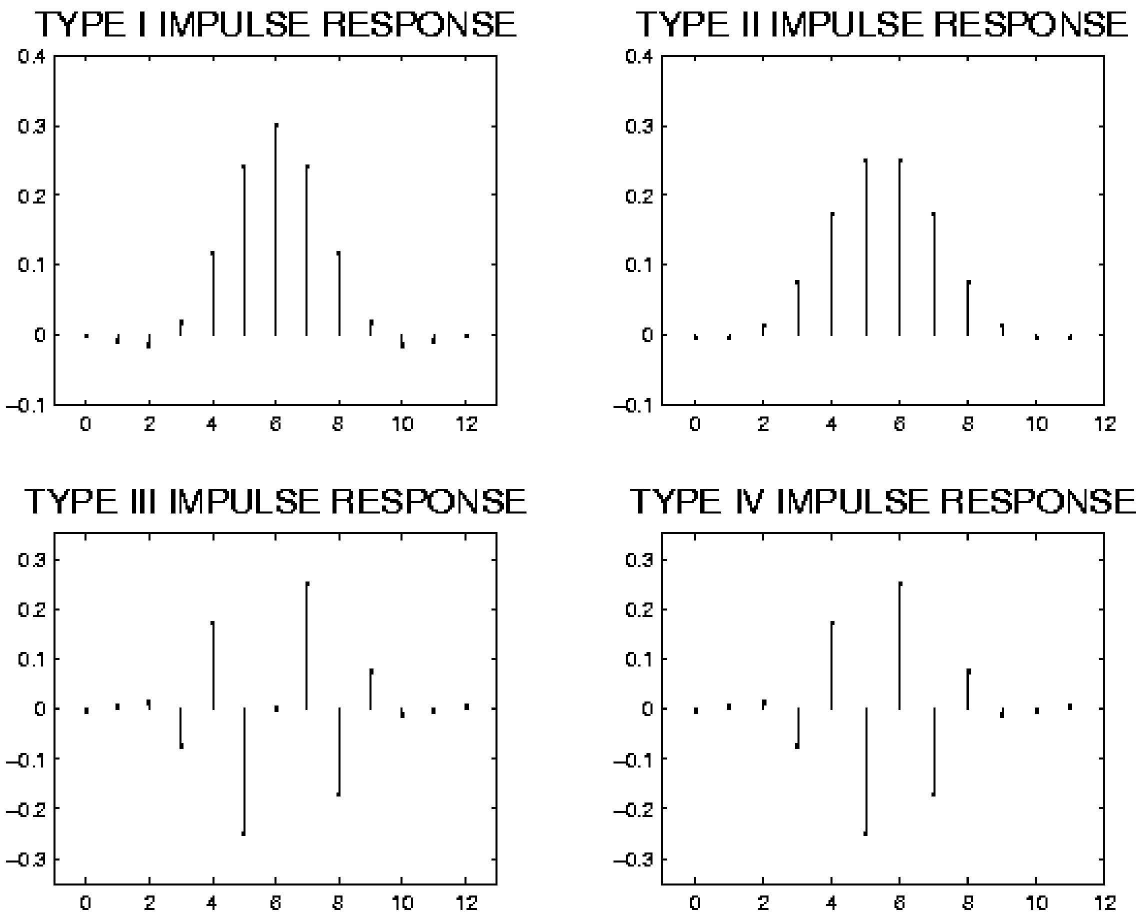


Figure 5.4

When $h(n)$ is nonzero for $0 \leq n \leq N-1$ (the length of the impulse response $h(n)$ is N), then the symmetry of the impulse response can be written as

$$h(n) = h(N-1-n) \tag{5.5}$$

and the anti-symmetry can be written as

$$h(n) = -h(N-1-n) \tag{5.6}$$

5.2.2 TYPE I: ODD-LENGTH SYMMETRIC

The frequency response of a length $N = 5$ FIR Type I filter can be written as follows.

$$H^f(\omega) = h_0 + h_1 e^{(-j)\omega} + h_2 e^{-2j\omega} + h_1 e^{-3j\omega} + h_0 e^{-4\omega} \tag{5.7}$$

$$H^f(\omega) = e^{-2j\omega}\left(h_0 e^{2j\omega} + h_1 e^{j\omega} + h_2 + h_1 e^{(-j)\omega} + h_0 e^{-2j\omega}\right) \tag{5.8}$$

$$H^f(\omega) = e^{-2j\omega}\left(h_0\left(e^{2j\omega} + e^{-2j\omega}\right) + h_1\left(e^{j\omega} + e^{(-j)\omega}\right) + h_2\right) \tag{5.9}$$

$$H^f(\omega) = e^{-2j\omega}\left(2h_0\cos(2\omega) + 2h_1\cos(\omega) + h_2\right) \tag{5.10}$$

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)} \tag{5.11}$$

where

$$\theta(\omega) = -2\omega$$

$$A(\omega) = 2h_0\cos(2\omega) + 2h_1\cos(\omega) + h_2$$

Note that $A(\omega)$ is real-valued and can be both positive and negative. In general, for a Type I FIR filters of length N:

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)}$$

$$A(\omega) = h(M) + 2\sum_{n=0}^{M-1} h(n)\cos((M-n)\omega) \tag{5.12}$$

$$\theta(\omega) = (-M)\,\omega$$

$$M = \frac{N-1}{2}$$

5.2.3 TYPE II: EVEN-LENGTH SYMMETRIC

The frequency response of a length $N = 4$ FIR Type II filter can be written as follows.

$$H^f(\omega) = h_0 + h_1 e^{(-j)\omega} + h_1 e^{-2j\omega} + h_0 e^{-3j\omega} \tag{5.13}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}\left(h_0 e^{\frac{3}{2}j\omega} + h_1 e^{\frac{1}{2}j\omega} + h_1 e^{\frac{-1}{2}j\omega} + h_0 e^{\frac{-3}{2}j\omega}\right) \tag{5.14}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}\left(h_0\left(e^{\frac{3}{2}j\omega} + e^{\frac{-3}{2}j\omega}\right) + h_1\left(e^{\frac{1}{2}j\omega} + e^{\frac{-1}{2}j\omega}\right)\right) \tag{5.15}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}\left(2h_0\cos\left(\frac{3}{2}\omega\right) + 2h_1\cos\left(\frac{1}{2}\omega\right)\right) \tag{5.16}$$

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)} \tag{5.17}$$

where

$$\theta(\omega) = \frac{-3}{2}\omega$$

$$A(\omega) = 2h_0\cos\left(\frac{3}{2}\omega\right) + 2h_1\cos\left(\frac{1}{2}\omega\right)$$

In general, for a Type II FIR filters of length N:

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)}$$

$$A(\omega) = 2\sum_{n=0}^{\frac{N}{2}-1} h(n)\cos((M-n)\omega) \tag{5.18}$$

$$\theta(\omega) = (-M)\omega$$

$$M = \frac{N-1}{2}$$

5.2.4 TYPE III: ODD-LENGTH ANTI-SYMMETRIC

The frequency response of a length $N = 5$ FIR Type III filter can be written as follows.

$$H^f(\omega) = h_0 + h_1 e^{(-j)\omega} - \left(h_1 e^{-3j\omega} - h_0 e^{-4\omega}\right) \tag{5.19}$$

$$H^f(\omega) = e^{-2j\omega}\left(h_0 e^{2j\omega} + h_1 e^{j\omega} - \left(h_1 e^{(-j)\omega} - h_0 e^{-2j\omega}\right)\right) \tag{5.20}$$

$$H^f(\omega) = e^{-2j\omega}\left(h_0\left(e^{2j\omega} - e^{-2j\omega}\right) + h_1\left(e^{j\omega} - e^{(-j)\omega}\right)\right) \tag{5.21}$$

$$H^f(\omega) = e^{-2j\omega}\left(2jh_0\sin(2\omega) + 2jh_1\sin(\omega)\right) \tag{5.22}$$

$$H^f(\omega) = e^{-2j\omega}j\left(2h_0\sin(2\omega) + 2h_1\sin(\omega)\right) \tag{5.23}$$

$$H^f(\omega) = e^{-2j\omega}e^{j\frac{\pi}{2}}\left(2h_0\sin(2\omega) + 2h_1\sin(\omega)\right) \tag{5.24}$$

$$H^f(\omega) = A(\omega)e^{j\theta(\omega)} \tag{5.25}$$

where

$$\theta(\omega) = 2\omega + \frac{\pi}{2}$$

$$A(\omega) = 2h_0\sin(2\omega) + 2h_1\sin(\omega)$$

In general, for a Type III FIR filters of length N:

$$H^f(\omega) = A(\omega)e^{j\theta(\omega)}$$

$$A(\omega) = 2\sum_{n=0}^{M-1} h(n)\sin((M-n)\omega) \tag{5.26}$$

$$\theta(\omega) = M\omega + \frac{\pi}{2}$$

$$M = \frac{N-1}{2}$$

5.2.5 TYPE IV: EVEN-LENGTH ANTI-SYMMETRIC

The frequency response of a length $N = 4$ FIR Type IV filter can be written as follows.

$$H^f(\omega) = h_0 + h_1 e^{(-j)\omega} - \left(h_1 e^{-2j\omega} - h_0 e^{-3j\omega}\right) \tag{5.27}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}\left(h_0 e^{\frac{3}{2}j\omega} + h_1 e^{\frac{1}{2}j\omega} - \left(h_1 e^{\frac{-1}{2}j\omega} - h_0 e^{\frac{-3}{2}j\omega}\right)\right) \tag{5.28}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}\left(h_0\left(e^{\frac{3}{2}j\omega} - e^{\frac{-3}{2}j\omega}\right) + h_1\left(e^{\frac{1}{2}j\omega} - e^{\frac{-1}{2}j\omega}\right)\right) \tag{5.29}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}\left(2jh_0\sin\left(\frac{3}{2}\omega\right) + 2jh_1\sin\left(\frac{1}{2}\omega\right)\right) \tag{5.30}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}j\left(2h_0\sin\left(\frac{3}{2}\omega\right) + 2h_1\sin\left(\frac{1}{2}\omega\right)\right) \tag{5.31}$$

$$H^f(\omega) = e^{\frac{-3}{2}j\omega}e^{j\frac{\pi}{2}}\left(2h_0\sin\left(\frac{3}{2}\omega\right) + 2h_1\sin\left(\frac{1}{2}\omega\right)\right) \tag{5.32}$$

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)} \tag{5.33}$$

where

$$\theta(\omega) = \frac{-3}{2}\omega + \frac{\pi}{2}$$

$$A(\omega) = 2h_0\sin\left(\frac{3}{2}\omega\right) + 2h_1\sin\left(\frac{1}{2}\omega\right)$$

In general, for a Type IV FIR filters of length N:

$$H^f(\omega) = A(\omega)\, e^{j\theta(\omega)}$$

$$A(\omega) = 2\sum_{n=0}^{\frac{N}{2}-1} h(n)\sin((M-n)\omega) \tag{5.34}$$

$$\theta(\omega) = M\omega + \frac{\pi}{2}$$

$$M = \frac{N-1}{2}$$

5.3 Design of Linear-Phase FIR Filters by DFT-Based Interpolation[3]

5.3.1 DESIGN OF FIR FILTERS BY DFT-BASED INTERPOLATION

One approach to the design of FIR filters is to ask that $A(\omega)$ pass through a specified set of values. If the number of specified interpolation points is the same as the number of filter parameters, then the filter is totally determined by the interpolation conditions, and the filter can be found by solving a system of linear equations. When the interpolation points are equally spaced between 0 and 2π, then this interpolation problem can be solved very efficiently using the DFT.

[3]This content is available online at <http://cnx.org/content/m10701/2.2/>.

To derive the DFT solution to the interpolation problem, recall the formula relating the samples of the frequency response to the DFT. In the case we are interested here, the number of samples is to be the **same** as the length of the filter ($L = N$).

$$\begin{aligned} H\left(\tfrac{2\pi}{N}k\right) &= \textstyle\sum_{n=0}^{N-1} h(n)\, e^{j\frac{2\pi}{N}nk} \\ &= \mathrm{DFT}_N\left(h(n)\right) \end{aligned} \tag{5.35}$$

5.3.1.1 Types I and II

Recall the relation between $A(\omega)$ and $H^f(\omega)$ for a Type I and II filter, to obtain

$$\begin{aligned} A\left(\tfrac{2\pi}{N}k\right) &= H\left(\tfrac{2\pi}{N}k\right) e^{jM\frac{2\pi}{N}k} \\ &= \mathrm{DFT}_N\left(h(n)\right) W_N^{Mk} \end{aligned} \tag{5.36}$$

Now we can related the N-point DFT of $h(n)$ to the samples of $A(\omega)$:

$$\mathrm{DFT}_N\left(h(n)\right) = A\left(\frac{2\pi}{N}k\right) W_N^{-Mk}$$

Finally, we can solve for the filter coefficients $h(n)$.

$$h(n) = \mathrm{DFT}_N{}^{-1}\left(A\left(\frac{2\pi}{N}k\right) W_N^{-Mk}\right) \tag{5.37}$$

Therefore, if the values $A\left(\frac{2\pi}{N}k\right)$are specified, we can then obtain the filter coefficients $h(n)$that satisfies the interpolation conditions by using the inverse DFT. It is important to note however, that the specified values $A\left(\frac{2\pi}{N}k\right)$must possess the appropriate symmetry in order for the result of the inverse DFT to be a real Type I or II FIR filter.

5.3.1.2 Types III and IV

For Type III and IV filters, we have

$$\begin{aligned} A\left(\tfrac{2\pi}{N}k\right) &= (-j)\, H\left(\tfrac{2\pi}{N}k\right) e^{jM\frac{2\pi}{N}k} \\ &= (-j)\, \mathrm{DFT}_N\left(h(n)\right) W_N^{Mk} \end{aligned} \tag{5.38}$$

Then we can related the N-point DFT of $h(n)$ to the samples of $A(\omega)$:

$$\mathrm{DFT}_N\left(h(n)\right) = jA\left(\frac{2\pi}{N}k\right) W_N^{Mk}$$

Solving for the filter coefficients $h(n)$ gives:

$$h(n) = \mathrm{DFT}_N{}^{-1}\left(jA\left(\frac{2\pi}{N}k\right) W_N^{-Mk}\right) \tag{5.39}$$

5.3.2 EXAMPLE: DFT-INTERPOLATION (TYPE I)

The following Matlab code fragment illustrates how to use this approach to design a length 11 Type I FIR filter for which $A\left(\frac{2\pi}{N}k\right) = (1,1,1,0,0,0,0,0,0,1,1)^T \quad , \quad (0 \leq k \leq N-1) \quad and \quad (N = 11) \quad .$

```
≫ N = 11;
≫ M = (N-1)/2;
≫ Ak = [1 1 1 0 0 0 0 0 0 1 1};   % samples of A(w)
≫ k = 0:N-1;
≫ W = exp(j*2*pi/N);
≫ h = ifft(Ak.*W.^(-M*k));
≫ h'

ans =

   0.0694 - 0.0000i
  -0.0540 - 0.0000i
  -0.1094 + 0.0000i
   0.0474 + 0.0000i
   0.3194 + 0.0000i
   0.4545 + 0.0000i
   0.3194 + 0.0000i
   0.0474 + 0.0000i
  -0.1094 + 0.0000i
  -0.0540 - 0.0000i
   0.0694 - 0.0000i
```

Observe that the filter coefficients `h` are real and symmetric; that a Type I filter is obtained as desired. The plot of $A(\omega)$ for this filter illustrates the interpolation points.

```
L = 512;
H = fft([h zeros(1,L-N)]);
W = exp(j*2*pi/L);
k = 0:L-1;
A = H .* W.^(M*k);
A = real(A);
w = k*2*pi/L;
plot(w/pi,A,2*[0:N-1]/N,Ak,'o')
xlabel('\omega/\pi')
title('A(\omega)')
```

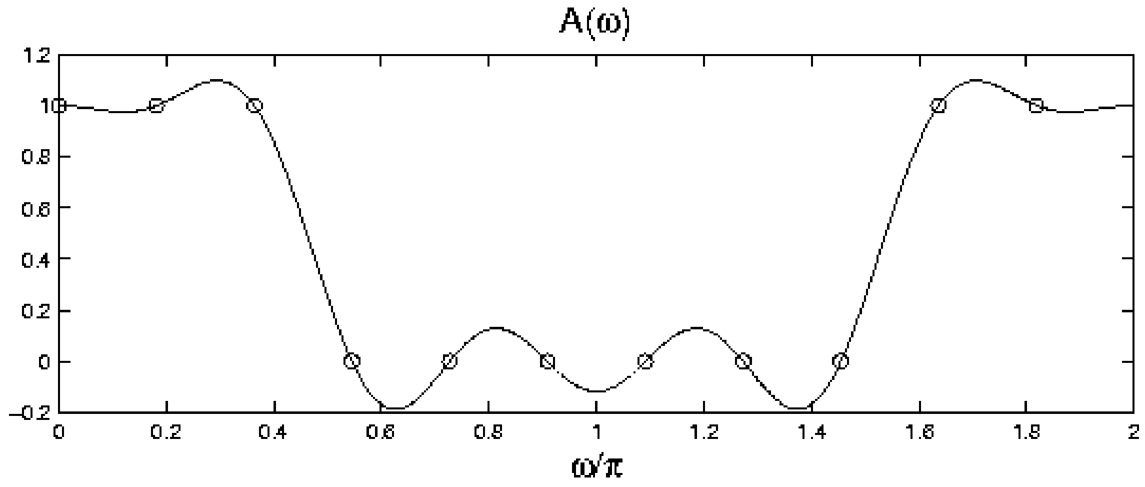


Figure 5.5

An exercise for the student: develop this DFT-based interpolation approach for Type II, III, and IV FIR filters. Modify the Matlab code above for each case.

5.3.3 SUMMARY: IMPULSE AND AMP RESPONSE

For an N-point linear-phase FIR filter $h(n)$, we summarize:

1. The formulas for evaluating the amplitude response $A(\omega)$ at L equally spaced points from 0 to 2π ($L \geq N$).
2. The formulas for the DFT-based interpolation design of $h(n)$.

TYPE I and II:

$$A\left(\frac{2\pi}{L}k\right) = \mathrm{DFT}_L\left(\left[h(n), 0_{L-N}\right]\right) W_L^{Mk} \tag{5.40}$$

$$h(n) = \mathrm{DFT}_N{}^{-1}\left(A\left(\frac{2\pi}{N}k\right) W_N^{-Mk}\right) \tag{5.41}$$

TYPE III and IV:

$$A\left(\frac{2\pi}{L}k\right) = (-j)\,\mathrm{DFT}_L\left(\left[h(n), 0_{L-N}\right]\right) W_L^{Mk} \tag{5.42}$$

$$h(n) = \mathrm{DFT}_N{}^{-1}\left(jA\left(\frac{2\pi}{N}k\right) W_N^{-Mk}\right) \tag{5.43}$$

5.4 Design of Linear-Phase FIR Filters by General Interpolation[4]

5.4.1 DESIGN OF FIR FILTERS BY GENERAL INTERPOLATION

If the desired interpolation points are not uniformly spaced between 0 and π then we can not use the DFT. We must take a different approach. Recall that for a Type I FIR filter,

$$A(\omega) = h(M) + 2\sum_{n=0}^{M-1} h(n)\cos((M-n)\omega)$$

For convenience, it is common to write this as

$$A(\omega) = \sum_{n=0}^{M} a(n)\cos(n\omega)$$

where $h(M) = a(0)$ and $h(n) = \frac{a(M-n)}{2}\quad,\quad 1 \leq n \leq N-1\quad$. Note that there are $M+1$ parameters. Suppose it is desired that $A(\omega)$ interpolates a set of specified values:

$$A(\omega_k) = A_k\quad,\quad 0 \leq k \leq M$$

To obtain a Type I FIR filter satisfying these interpolation equations, one can set up a linear system of equations.

$$\sum_{n=0}^{M} a(n)\cos(n\omega_k) = A_k\quad,\quad 0 \leq k \leq M$$

In matrix form, we have

$$\begin{pmatrix} 1 & \cos(\omega_0) & \cos(2\omega_0) & \dots & \cos(M\omega_0) \\ 1 & \cos(\omega_1) & \cos(2\omega_1) & \dots & \cos(M\omega_1) \\ \vdots & & & & \\ 1 & \cos(\omega_M) & \cos(2\omega_M) & \dots & \cos(M\omega_M) \end{pmatrix} \begin{pmatrix} a(0) \\ a(1) \\ \vdots \\ a(M) \end{pmatrix} = \begin{pmatrix} A(0) \\ A(1) \\ \vdots \\ A(M) \end{pmatrix}$$

Once $a(n)$ is found, the filter $h(n)$ is formed as

$$\{h(n)\} = 1/2\,\{a(M), a(M-1), \dots, a(1), 2\times a(0), a(1), \dots, a(M-1), a(M)\}$$

5.4.2 EXAMPLE

In the following example, we design a length 19 Type I FIR. Then $M = 9$ and we have 10 parameters. We can therefore have 10 interpolation equations. We choose:

$$A(\omega_k) = 1\quad,\quad \omega_k = \{0, 0.1\pi, 0.2\pi, 0.3\pi\}\quad 0 \leq k \leq 3 \tag{5.44}$$

$$A(\omega_k) = 0\quad,\quad \omega_k = \{0.5\pi, 0.6\pi, 0.7\pi, 0.8\pi, 0.8\pi, 1.0\pi\}\quad 4 \leq k \leq 9 \tag{5.45}$$

To solve this interpolation problem in Matlab, note that the matrix can be generated by a single multiplication of a column vector and a row vector. This is done with the command `C = cos(wk*[0:M]);` where `wk` is a column vector containing the frequency points. To solve the linear system of equations, we can use the Matlab backslash command.

[4]This content is available online at <http://cnx.org/content/m10704/2.2/>.

```
N = 19;
M = (N-1)/2;
wk = [0 .1 .2 .3 .5 .6 .7 .8 .9 1]'*pi;
Ak = [1 1 1 1 0 0 0 0 0 0]';
C = cos(wk*[0:M]);
a = C/Ak;
h = (1/2)*[a([M:-1:1]+1); 2*a([0]+1); a(1:M]+1)];

[A,w] = firamp(h,1);
plot(w/pi,A,wk/pi,Ak,'o')
title('A(\omega)')
xlabel('\omega/\pi')
```

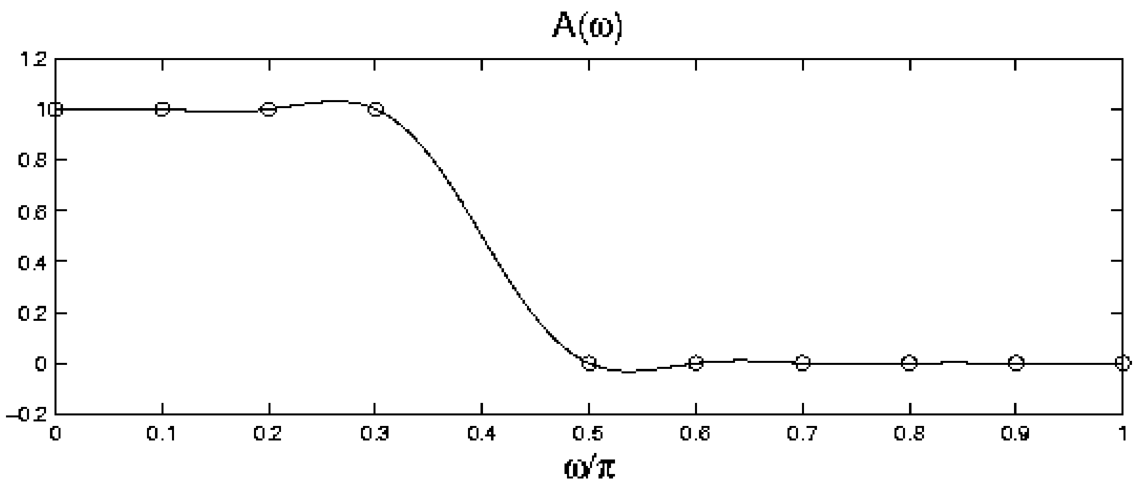


Figure 5.6

The general interpolation problem is much more flexible than the uniform interpolation problem that the DFT solves. For example, by leaving a gap between the pass-band and stop-band as in this example, the ripple near the band edge is reduced (but the transition between the pass- and stop-bands is not as sharp). The general interpolation problem also arises as a subproblem in the design of optimal minimax (or Chebyshev) FIR filters.

5.4.3 LINEAR-PHASE FIR FILTERS: PROS AND CONS

FIR digital filters have several desirable properties.

- They can have exactly linear phase.
- They can not be unstable.

- There are several very effective methods for designing linear-phase FIR digital filters.

On the other hand,

- Linear-phase filters can have long delay between input and output.
- If the phase need not be linear, then IIR filters can be more efficient.

5.5 Linear-Phase FIR Filters: Amplitude Formulas[5]

5.5.1 SUMMARY: AMPLITUDE FORMULAS

Type	$\theta(\omega)$	$A(\omega)$
I	$-(M\omega)$	$h(M)+2\sum_{n=0}^{M-1}h(n)\cos((M-n)\omega)$
II	$-(M\omega)$	$2\sum_{n=0}^{\frac{N}{2}-1}h(n)\cos((M-n)\omega)$
III	$-(M\omega)+\frac{\pi}{2}$	$2\sum_{n=0}^{M-1}h(n)\sin((M-n)\omega)$
IV	$-(M\omega)+\frac{\pi}{2}$	$2\sum_{n=0}^{\frac{N}{2}-1}h(n)\sin((M-n)\omega)$

Table 5.2

where $M=\frac{N-1}{2}$

5.5.2 AMPLITUDE RESPONSE CHARACTERISTICS

To analyze or design linear-phase FIR filters, we need to know the characteristics of the amplitude response $A(\omega)$.

Type	**Properties**	
I	$A(\omega)$ is even about $\omega=0$	$A(\omega)=A(-\omega)$
	$A(\omega)$ is even about $\omega=\pi$	$A(\pi+\omega)=A(\pi-\omega)$
	$A(\omega)$ is periodic with 2π	$A(\omega+2\pi)=A(\omega)$
II	$A(\omega)$ is even about $\omega=0$	$A(\omega)=A(-\omega)$
	$A(\omega)$ is odd about $\omega=\pi$	$A(\pi+\omega)=-A(\pi-\omega)$
	$A(\omega)$ is periodic with 4π	$A(\omega+4\pi)=A(\omega)$
III	$A(\omega)$ is odd about $\omega=0$	$A(\omega)=-A(-\omega)$
	$A(\omega)$ is odd about $\omega=\pi$	$A(\pi+\omega)=-A(\pi-\omega)$
	$A(\omega)$ is periodic with 2π	$A(\omega+2\pi)=A(\omega)$

continued on next page

[5]This content is available online at <http://cnx.org/content/m10707/2.5/>.

IV	$A(\omega)$ is odd about $\omega = 0$	$A(\omega) = -A(-\omega)$
	$A(\omega)$ is even about $\omega = \pi$	$A(\pi + \omega) = A(\pi - \omega)$
	$A(\omega)$ is periodic with 4π	$A(\omega + 4\pi) = A(\omega)$

Table 5.3

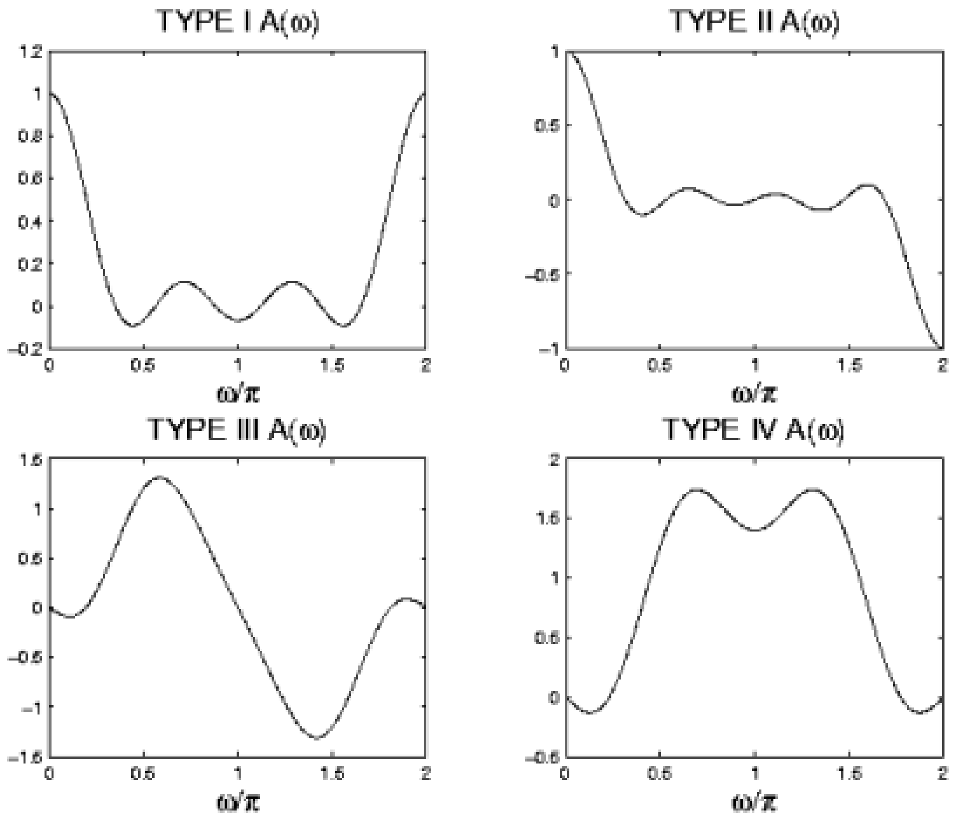


Figure 5.7

5.5.3 EVALUATING THE AMPLITUDE RESPONSE

The frequency response $H^f(\omega)$ of an FIR filter can be evaluated at L equally spaced frequencies between 0 and π using the DFT. Consider a causal FIR filter with an impulse response $h(n)$ of length-N, with $N \leq L$. Samples of the frequency response of the filter can be written as

$$H\left(\frac{2\pi}{L}k\right) = \sum_{n=0}^{N-1} h(n)\, e^{(-j)\frac{2\pi}{L}nk}$$

Define the L-point signal $\{g(n) \mid 0 \leq n \leq L-1\}$ as

$$g(n) = \begin{cases} h(n) & \text{if } 0 \leq n \leq N-1 \\ 0 & \text{if } N \leq n \leq L-1 \end{cases}$$

Then

$$H\left(\frac{2\pi}{L}k\right) = G(k) = \mathrm{DFT}_L(g(n))$$

where $G(k)$ is the L-point DFT of $g(n)$.

5.5.3.1 Types I and II

Suppose the FIR filter $h(n)$ is either a Type I or a Type II FIR filter. Then we have from above

$$H^f(\omega) = A(\omega)\, e^{(-j)M\omega}$$

or

$$A(\omega) = H^f(\omega)\, e^{jM\omega}$$

Samples of the real-valued amplitude $A(\omega)$ can be obtained from samples of the function $H^f(\omega)$ as:

$$A\left(\frac{2\pi}{L}k\right) = H\left(\frac{2\pi}{L}k\right) e^{jM\frac{2\pi}{L}k} = G(k)\, W_L^{Mk}$$

Therefore, the samples of the real-valued amplitude function can be obtained by zero-padding $h(n)$, taking the DFT, and multiplying by the complex exponential. This can be written as:

$$A\left(\frac{2\pi}{L}k\right) = \mathrm{DFT}_L([h(n), 0_{L-N}])\, W_L^{Mk} \tag{5.46}$$

5.5.3.2 Types III and IV

For Type III and Type IV FIR filters, we have

$$H^f(\omega) = je^{(-j)M\omega} A(\omega)$$

or

$$A(\omega) = (-j)\, H^f(\omega)\, e^{jM\omega}$$

Therefore, samples of the real-valued amplitude $A(\omega)$ can be obtained from samples of the function $H^f(\omega)$ as:

$$A\left(\frac{2\pi}{L}k\right) = (-j)\, H\left(\frac{2\pi}{L}k\right) e^{jM\frac{2\pi}{L}k} = (-j)\, G(k)\, W_L^{Mk}$$

Therefore, the samples of the real-valued amplitude function can be obtained by zero-padding $h(n)$, taking the DFT, and multiplying by the complex exponential.

$$A\left(\frac{2\pi}{L}k\right) = (-j)\,\mathrm{DFT}_L([h(n), 0_{L-N}])\, W_L^{Mk} \tag{5.47}$$

Example 5.1: EVALUATING THE AMP RESP (TYPE I)
In this example, the filter is a Type I FIR filter of length 7. An accurate plot of $A(\omega)$ can be obtained with zero padding.

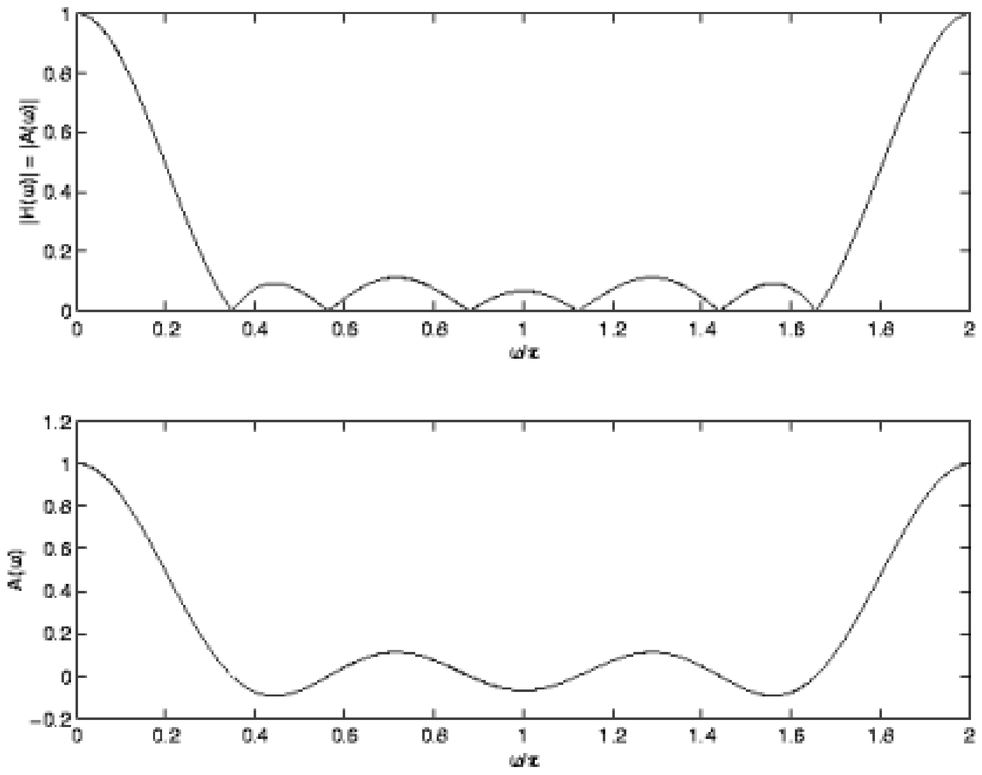


Figure 5.8

The following Matlab code fragment for the plot of $A(\omega)$ for a Type I FIR filter.

```
h = [3 4 5 6 5 4 3]/30;
N = 7;
M = (N-1)/2;
L = 512;
H = fft([h zeros(1,L-N)]);
k = 0:L-1;
W = exp(j*2*pi/L);
A = H .* W.^(M*k);
A = real(A);

figure(1)
w = [0:L-1]*2*pi/(L-1);
subplot(2,1,1)
plot(w/pi,abs(H))
ylabel('|H(\omega)| = |A(\omega)|')
xlabel('\omega/\pi')
subplot(2,1,2)
plot(w/pi,A)
ylabel('A(\omega)')
xlabel('\omega/\pi')
```

```
print -deps type1
```

The command `A = real(A)` removes the imaginary part which is equal to zero to within computer precision. Without this command, Matlab takes `A` to be a complex vector and the following plot command will not be right.

Observe the symmetry of $A(\omega)$ due to $h(n)$ being real-valued. Because of this symmetry, $A(\omega)$ is usually plotted for $0 \le \omega \le \pi$ only.

Example 5.2: EVALUATING THE AMP RESP (TYPE II)

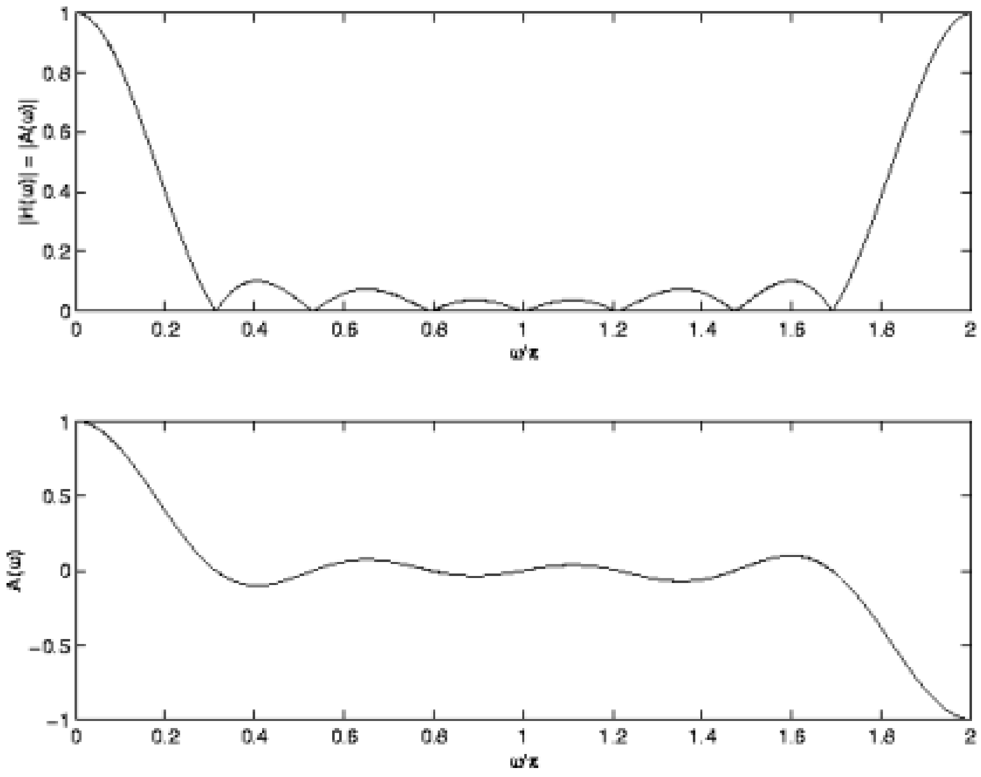


Figure 5.9

The following Matlab code fragment produces a plot of $A(\omega)$ for a Type II FIR filter.

```
h = [3 5 6 7 7 6 5 3]/42;
N = 8;
M = (N-1)/2;
L = 512;
H = fft([h zeros(1,L-N)]);
k = 0:L-1;
W = exp(j*2*pi/L);
A = H .* W.^(M*k);
A = real(A);
```

```
figure(1)
w = [0:L-1]*2*pi/(L-1);
subplot(2,1,1)
plot(w/pi,abs(H))
ylabel('|H(\omega)| = |A(\omega)|')
xlabel('\omega/\pi')
subplot(2,1,2)
plot(w/pi,A)
ylabel('A(\omega)')
xlabel('\omega/\pi')
print -deps type2
```

The imaginary part of the amplitude is zero. Notice that $A(\pi) = 0$. In fact this will always be the case for a Type II FIR filter.

An exercise for the student: Describe how to obtain samples of $A(\omega)$ for Type III and Type IV FIR filters. Modify the Matlab code above for these types. Do you notice that $A(\omega) = 0$ always for special values of ω?

5.5.4 Modules for Further Study

1. Zero Locations of Linear-Phase Filters (Section 3.9)
2. Design of Linear-Phase FIR Filters by Interpolation (Section 5.3)
3. Linear-Phase FIR Filter Design by Least Squares[6]

5.6 FIR Filter Design using MATLAB[7]

5.6.1 FIR Filter Design Using MATLAB

5.6.1.1 Design by windowing

The MATLAB function `fir1()` designs conventional lowpass, highpass, bandpass, and bandstop linear-phase FIR filters based on the windowing method. The command

```
b = fir1(N,Wn)
```

returns in vector `b` the impulse response of a lowpass filter of order `N`. The cut-off frequency `Wn` must be between 0 and 1 with 1 corresponding to the half sampling rate.

The command

```
b = fir1(N,Wn,'high')
```

[6]"Linear-Phase Fir Filter Design By Least Squares" <http://cnx.org/content/m10577/latest/>
[7]This content is available online at <http://cnx.org/content/m10917/2.2/>.

returns the impulse response of a highpass filter of order `N` with normalized cutoff frequency `Wn`.

Similarly, `b = fir1(N,Wn,'stop')` with `Wn` a two-element vector designating the stopband designs a bandstop filter.

Without explicit specification, the **Hamming window** is employed in the design. Other windowing functions can be used by specifying the windowing function as an extra argument of the function. For example, Blackman window can be used instead by the command `b = fir1(N, Wn, blackman(N))`.

5.6.1.2 Parks-McClellan FIR filter design

The MATLAB command

```
    b = remez(N,F,A)
```

returns the impulse response of the length `N+1` linear phase FIR filter of order `N` designed by Parks-McClellan algorithm. F is a vector of frequency band edges in ascending order between 0 and 1 with 1 corresponding to the half sampling rate. A is a real vector of the same size as F which specifies the desired amplitude of the frequency response of the points `(F(k),A(k))` and `(F(k+1),A(k+1))` for odd `k`. For odd `k`, the bands between `F(k+1)` and `F(k+2)` is considered as transition bands.

5.7 MATLAB FIR Filter Design Exercise[8]

5.7.1 FIR Filter Design MATLAB Exercise

5.7.1.1 Design by windowing

Exercise 5.7.1 ***(Solution on p. 156.)***

Assuming sampling rate at 48kHz, design an order-40 low-pass filter having cut-off frequency 10kHz by windowing method. In your design, use Hamming window as the windowing function.

5.7.1.2 Parks-McClellan Optimal Design

Exercise 5.7.2 ***(Solution on p. 156.)***

Assuming sampling rate at 48kHz, design an order-40 lowpass filter having transition band 10kHz-11kHz using the Parks-McClellan optimal FIR filter design algorithm.

5.8 Parks-McClellan Optimal FIR Filter Design[9]

[8]This content is available online at <http://cnx.org/content/m10918/2.2/>.
[9]This content is available online at <http://cnx.org/content/m10914/2.2/>.

Solutions to Exercises in Chapter 5

Solution to Exercise 5.1.1 (p. 137)

- $\frac{\theta(\omega)}{\omega} = \text{constant}$
- $\theta(\omega) = K\omega$
- The phase is linear.

Solution to Exercise 5.7.1 (p. 155)

```
b = fir1(40,10.0/48.0)
```

Solution to Exercise 5.7.2 (p. 155)

```
b = remez(40,[1 1 0 0],[0 10/48 11/48 1])
```

Chapter 6

Wiener Filter Design

Chapter 7

Adaptive Filtering

7.1 Adaptive Filtering: LMS Algorithm[1]

7.1.1 Introduction

Figure 7.1 is a block diagram of system identification using adaptive filtering. The objective is to change (adapt) the coefficients of an FIR filter, W, to match as closely as possible the response of an unknown system, H. The unknown system and the adapting filter process the same input signal $x[n]$ and have outputs $d[n]$(also referred to as the desired signal) and $y[n]$.

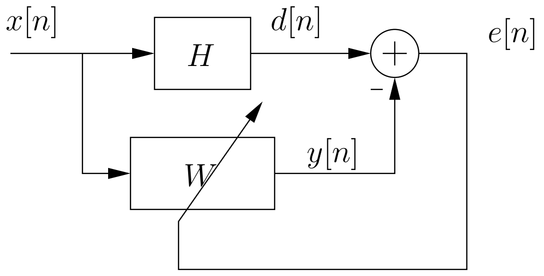


Figure 7.1: System identification block diagram.

7.1.1.1 Gradient-descent adaptation

The adaptive filter, W, is adapted using the least mean-square algorithm, which is the most widely used adaptive filtering algorithm. First the error signal, $e[n]$, is computed as $e[n] = d[n] - y[n]$, which measures the difference between the output of the adaptive filter and the output of the unknown system. On the basis of this measure, the adaptive filter will change its coefficients in an attempt to reduce the error. The

[1]This content is available online at <http://cnx.org/content/m10481/2.14/>.

coefficient update relation is a function of the error signal squared and is given by

$$h_{n+1}[i] = h_n[i] + \frac{\mu}{2}\left(-\frac{\partial(|e|)^2}{\partial h_n[i]}\right) \tag{7.1}$$

The term inside the parentheses represents the gradient of the squared-error with respect to the i^{th} coefficient. The gradient is a vector pointing in the direction of the change in filter coefficients that will cause the greatest increase in the error signal. Because the goal is to minimize the error, however, (7.1) updates the filter coefficients in the direction opposite the gradient; that is why the gradient term is negated. The constant μ is a step-size, which controls the amount of gradient information used to update each coefficient. After repeatedly adjusting each coefficient in the direction opposite to the gradient of the error, the adaptive filter should converge; that is, the difference between the unknown and adaptive systems should get smaller and smaller.

To express the gradient decent coefficient update equation in a more usable manner, we can rewrite the derivative of the squared-error term as

$$\begin{aligned} \frac{\partial(|e|)^2}{\partial h[i]} &= 2\frac{\partial e}{\partial h[i]}e \\ &= 2\frac{\partial(d-y)}{\partial h[i]}e \\ &= \left(2\frac{\partial\left(d-\sum_{i=0}^{N-1} h[i]x[n-i]\right)}{\partial h[i]}\right)(e) \end{aligned} \tag{7.2}$$

$$\frac{\partial(|e|)^2}{\partial h[i]} = 2(-x[n-i])e \tag{7.3}$$

which in turn gives us the final LMS coefficient update,

$$h_{\text{n}+1}[i] = h_n[i] + \mu e x[n-i] \tag{7.4}$$

The step-size μ directly affects how quickly the adaptive filter will converge toward the unknown system. If μ is very small, then the coefficients change only a small amount at each update, and the filter converges slowly. With a larger step-size, more gradient information is included in each update, and the filter converges more quickly; however, when the step-size is too large, the coefficients may change too quickly and the filter will diverge. (It is possible in some cases to determine analytically the largest value of μ ensuring convergence.)

7.1.2 MATLAB Simulation

Simulate the system identification block diagram shown in Figure 7.1.

Previously in MATLAB, you used the `filter` command or the `conv` command to implement shift-invariant filters. Those commands will not work here because adaptive filters are shift-varying, since the coefficient update equation changes the filter's impulse response at every sample time. Therefore, implement the system identification block on a sample-by-sample basis with a `do` loop, similar to the way you might implement a time-domain FIR filter on a DSP. For the "unknown" system, use the fourth-order, low-pass, elliptical, IIR filter designed for the IIR Filtering: Filter-Design Exercise in MATLAB[2].

Use Gaussian random noise as your input, which can be generated in MATLAB using the command `randn`. Random white noise provides signal at all digital frequencies to train the adaptive filter. Simulate the system with an adaptive filter of length 32 and a step-size of 0.02. Initialize all of the adaptive filter coefficients to zero. From your simulation, plot the error (or squared-error) as it evolves over time and plot the frequency response of the adaptive filter coefficients at the end of the simulation. How well does your adaptive filter match the "unknown" filter? How long does it take to converge?

Once your simulation is working, experiment with different step-sizes and adaptive filter lengths.

[2]"IIR Filtering: Filter-Design Exercise in MATLAB" <http://cnx.org/content/m10623/latest/>

7.1.3 Processor Implementation

Use the same "unknown" filter as you used in the MATLAB simulation.

Although the coefficient update equation is relatively straightforward, consider using the `lms` instruction available on the TI processor, which is designed for this application and yields a very efficient implementation of the coefficient update equation.

To generate noise on the DSP, you can use the PN generator from the Digital Transmitter: Introduction to Quadrature Phase-Shift Keying[3], but shift the PN register contents up to make the sign bit random. (If the sign bit is always zero, then the noise will not be zero-mean and this will affect convergence.) Send the desired signal, $d[n]$, the output of the adaptive filter, $y[n]$, and the error to the D/A for display on the oscilloscope.

When using the step-size suggested in the MATLAB simulation section, you should notice that the error converges very quickly. Try an extremely small μ so that you can actually watch the amplitude of the error signal decrease towards zero.

7.1.4 Extensions

If your project requires some modifications to the implementation here, refer to ***Haykin*** [1] and consider some of the following questions regarding such modifications:

- How would the system in Figure 7.1 change for different applications? (noise cancellation, equalization, *etc.*)
- What happens to the error when the step-size is too large or too small?
- How does the length of an adaptive FIR filters affect convergence?
- What types of coefficient update relations are possible besides the described LMS algorithm?

[3]"Digital Transmitter: Introduction to Quadrature Phase-Shift Keying" <http://cnx.org/content/m10042/latest/>

Chapter 8

Wavelets and Filterbanks

8.1 Haar Wavelet Basis[1]

8.1.1 Introduction

Fourier series[2] is a useful orthonormal representation[3] on $L^2([0,T])$ especiallly for inputs into LTI systems. However, it is ill suited for some applications, i.e. image processing (recall Gibb's phenomena[4]).

Wavelets, discovered in the last 15 years, are another kind of basis for $L^2([0,T])$ and have many nice properties.

8.1.2 Basis Comparisons

Fourier series - c_n give frequency information. Basis functions last the entire interval.

[1]This content is available online at <http://cnx.org/content/m10764/2.9/>.
[2]"Fourier Series: Eigenfunction Approach" <http://cnx.org/content/m10496/latest/>
[3]"Orthonormal Basis Expansions" <http://cnx.org/content/m10760/latest/>
[4]"Gibbs Phenomena" <http://cnx.org/content/m10092/latest/>

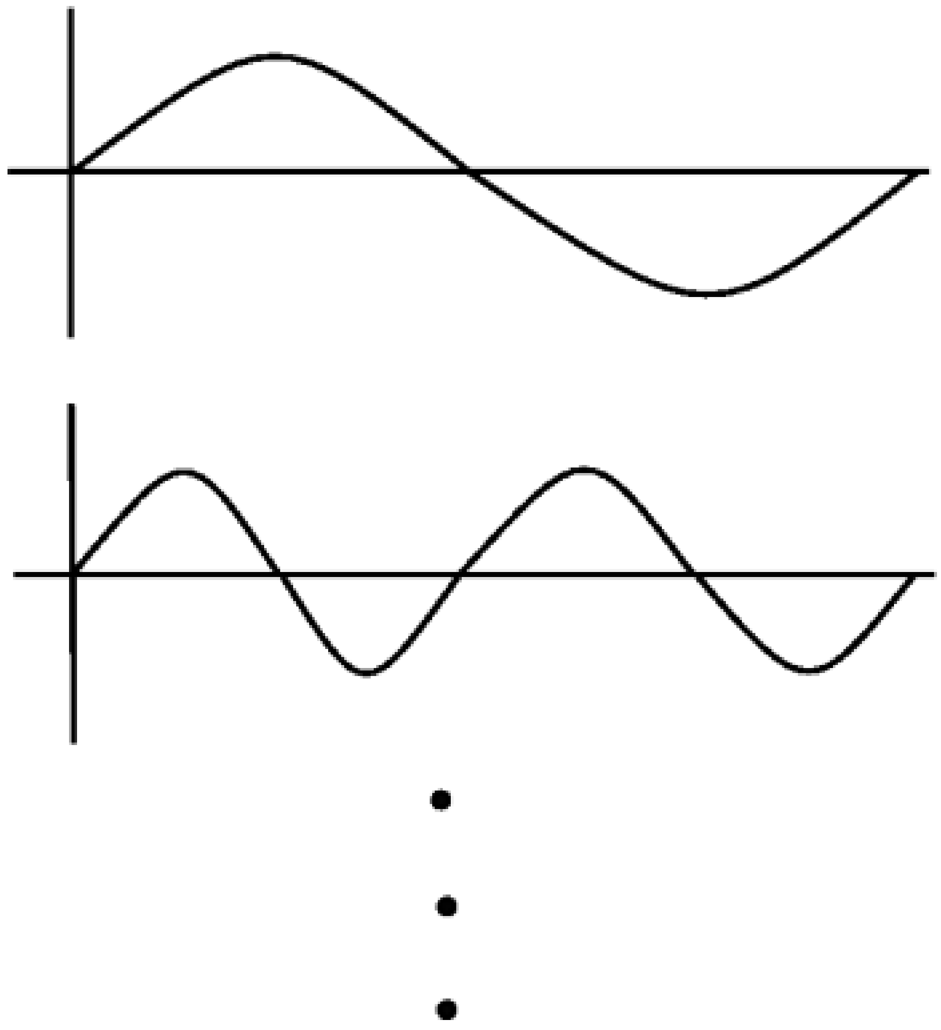


Figure 8.1: Fourier basis functions

Wavelets - basis functions give frequency info but are **local** in time.

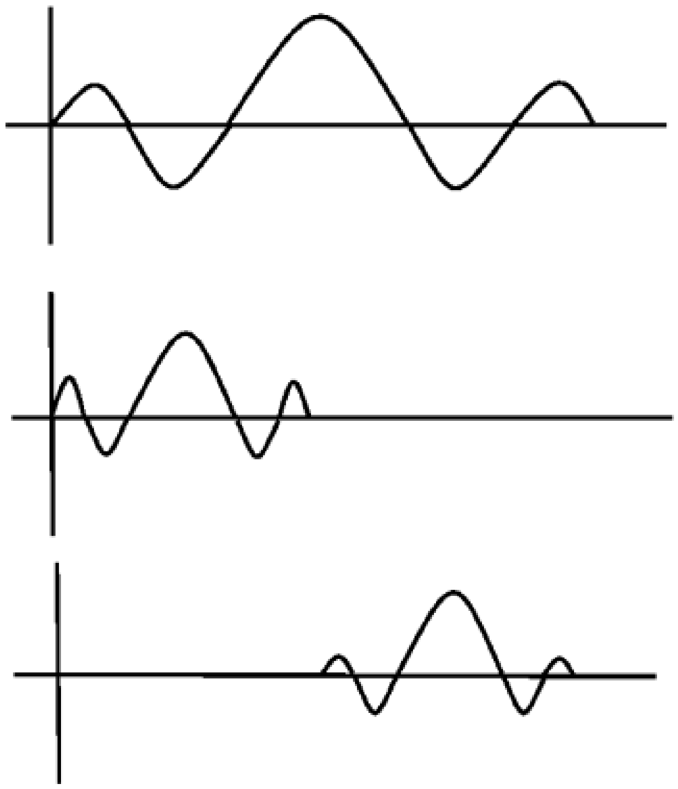


Figure 8.2: Wavelet basis functions

In Fourier basis, the basis functions are **harmonic multiples** of $e^{j\omega_0 t}$

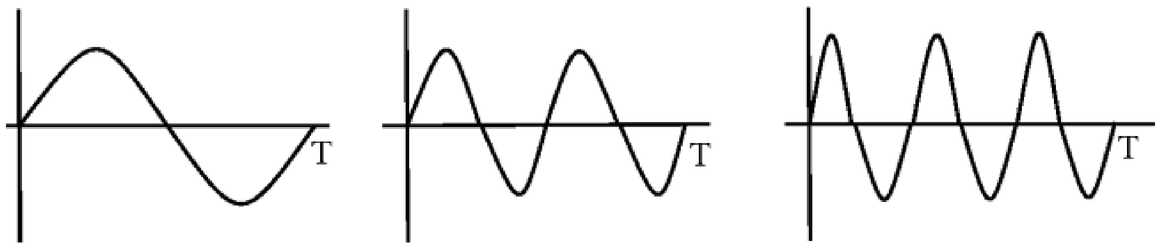


Figure 8.3: basis $= \left\{\frac{1}{\sqrt{T}} e^{j\omega_0 n t}\right\}$

In Haar wavelet basis (Section 8.5), the basis functions are **scaled and translated** versions of a "mother wavelet" $\psi(t)$.

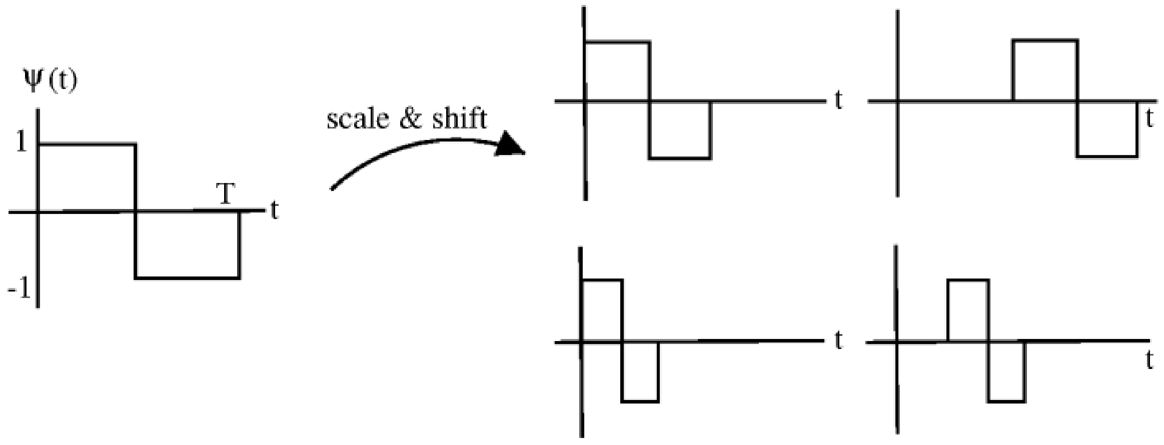


Figure 8.4

Basis functions $\{\psi_{j,k}(t)\}$ are indexed by a **scale** j and a **shift** k.
Let $\phi(t) = 1\ ,\ 0 \leq t < T$ Then $\left\{\phi(t), 2^{\frac{j}{2}}\psi\left(2^j t - k\right), \phi(t), 2^{\frac{j}{2}}\psi\left(2^j t - k\right) \mid j \in \mathbb{Z}\ \ and\ \ \left(k = 0, 1, 2, \dots, 2^j - 1\right)\right\}$

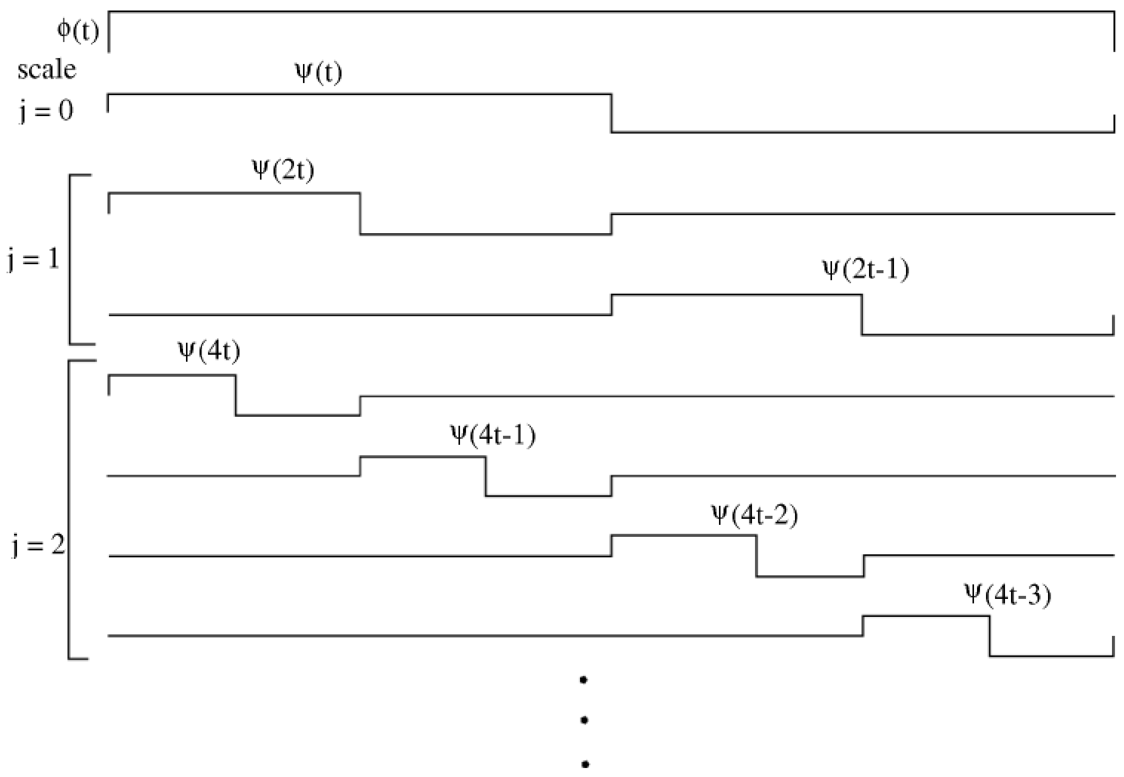


Figure 8.5

$$\psi(t) = \begin{cases} 1 \ \text{ if } \ 0 \leq t < \frac{T}{2} \\ -1 \ \text{ if } \ 0 \leq \frac{T}{2} < T \end{cases} \quad (8.1)$$

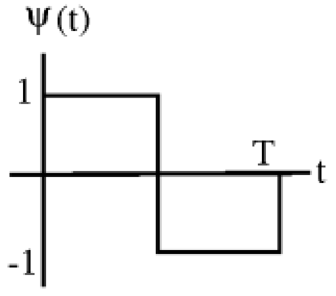


Figure 8.6

Let $\psi_{j,k}(t) = 2^{\frac{j}{2}}\psi\left(2^j t - k\right)$

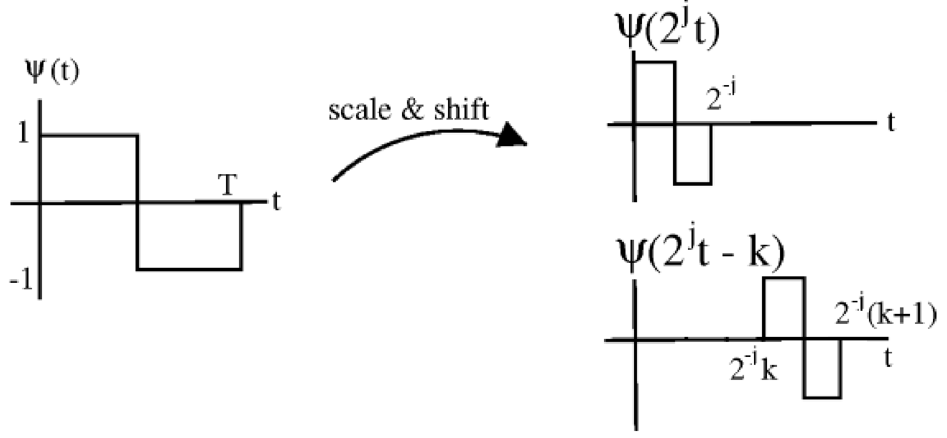


Figure 8.7

Larger $j \rightarrow$ "skinnier" basis function, $j = \{0, 1, 2, \dots\}$, 2^j shifts at each scale: $k = 0, 1, \dots, 2^j - 1$
Check: each $\psi_{j,k}(t)$ has unit energy

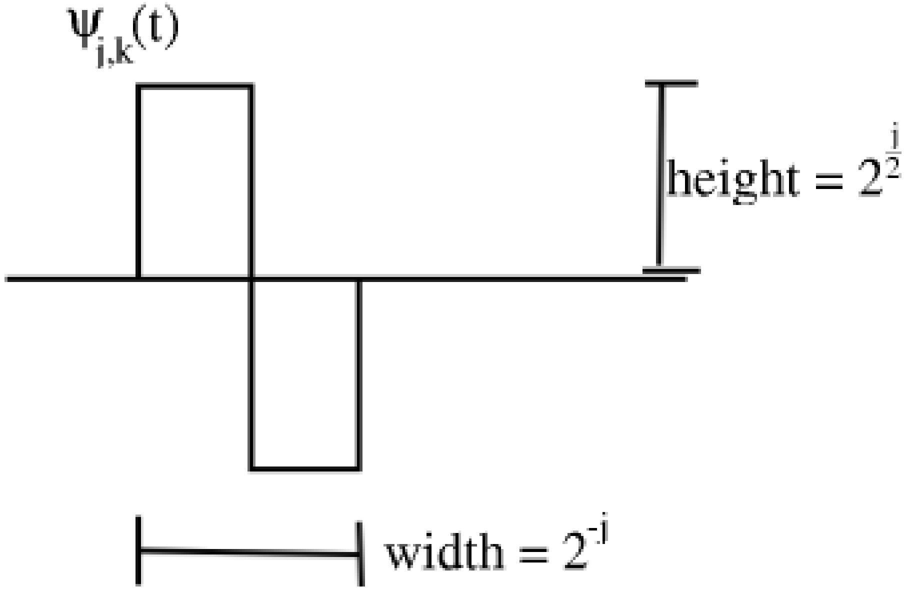


Figure 8.8

$$\left(\int \psi_{j,k}{}^2(t)\,dt = 1\right) \Rightarrow \left(\| \psi_{j,k}(t) \|_2 = 1\right) \tag{8.2}$$

Any two basis functions are orthogonal.

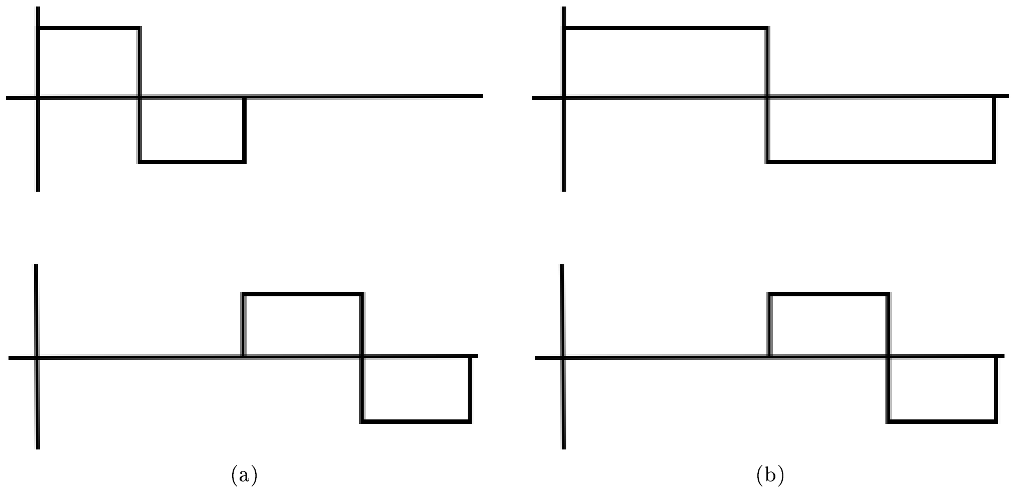


Figure 8.9: Integral of product = 0 (a) Same scale (b) Different scale

Also, $\{\psi_{j,k}, \phi\}$ span $L^2([0,T])$

8.1.3 Haar Wavelet Transform

Using what we know about Hilbert spaces[5]: For any $f(t) \in L^2([0,T])$, we can write

Synthesis

$$f(t) = \sum_j \sum_k w_{j,k} \psi_{j,k}(t) + c_0 \phi(t) \quad (8.3)$$

Analysis

$$w_{j,k} = \int_0^T f(t) \psi_{j,k}(t) \, dt \quad (8.4)$$

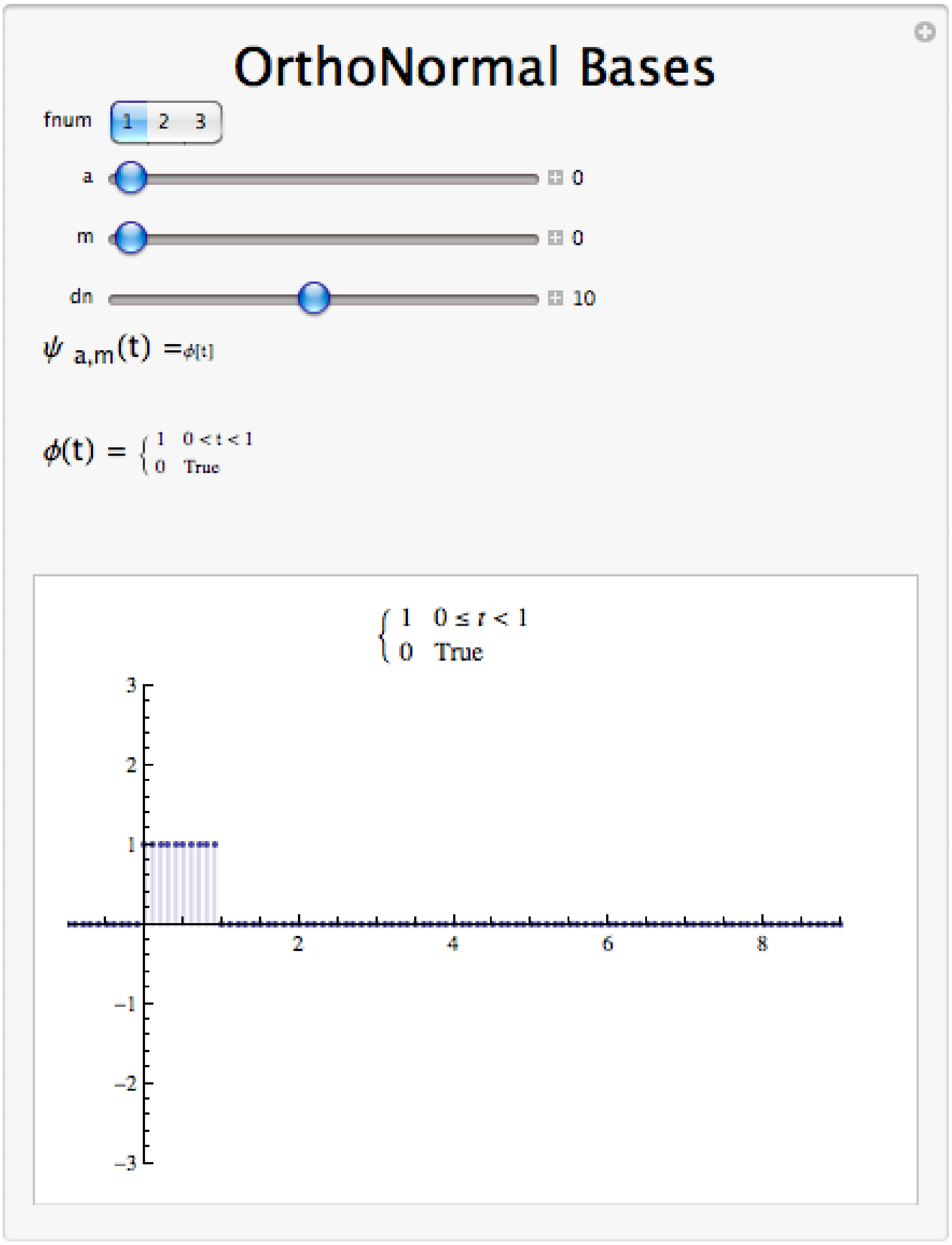
$$c_0 = \int_0^T f(t) \phi(t) \, dt \quad (8.5)$$

NOTE: the $w_{j,k}$ are **real**

The Haar transform is **super useful** especially in **image compression**

[5]"Inner Products" <http://cnx.org/content/m10755/latest/>

8.1.4 Haar Wavelet Demonstration


OrthoNormal Bases

fnum [✓ 1] 2 3

a 0

m 0

dn 10

$\psi_{a,m}(t) = \phi[t]$

$\phi(t) = \begin{cases} 1 & 0 < t < 1 \\ 0 & \text{True} \end{cases}$

$\begin{cases} 1 & 0 \le t < 1 \\ 0 & \text{True} \end{cases}$

Figure 8.10: Interact (when online) with a Mathematica CDF demonstrating the Haar Wavelet as an Orthonormal Basis.

8.2 Orthonormal Wavelet Basis[6]

An orthonormal wavelet basis is an orthonormal basis of the form

$$B = \left\{ 2^{\frac{j}{2}} \psi \left(2^j t - k \right) \mid j \in \mathbb{Z} \ \ and \ \ k \in \mathbb{Z} \right\} \quad (8.6)$$

The function $\psi(t)$ is called the **wavelet**.

The problem is how to find a function $\psi(t)$ so that the set B is an orthonormal set.

Example 8.1: Haar Wavelet

The Haar basis (Section 8.1) (described by Haar in 1910) is an orthonormal basis with wavelet $\psi(t)$

$$\psi(t) = \begin{cases} 1 \text{ if } 0 \leq t \leq 1/2 \\ -1 \text{ if } 1/2 \leq t \leq 1 \quad (8.7) \\ 0 \text{ otherwise} \end{cases}$$

For the Haar wavelet, it is easy to verify that the set B is an orthonormal set (Figure 8.11).

[6]This content is available online at <http://cnx.org/content/m10957/2.3/>.

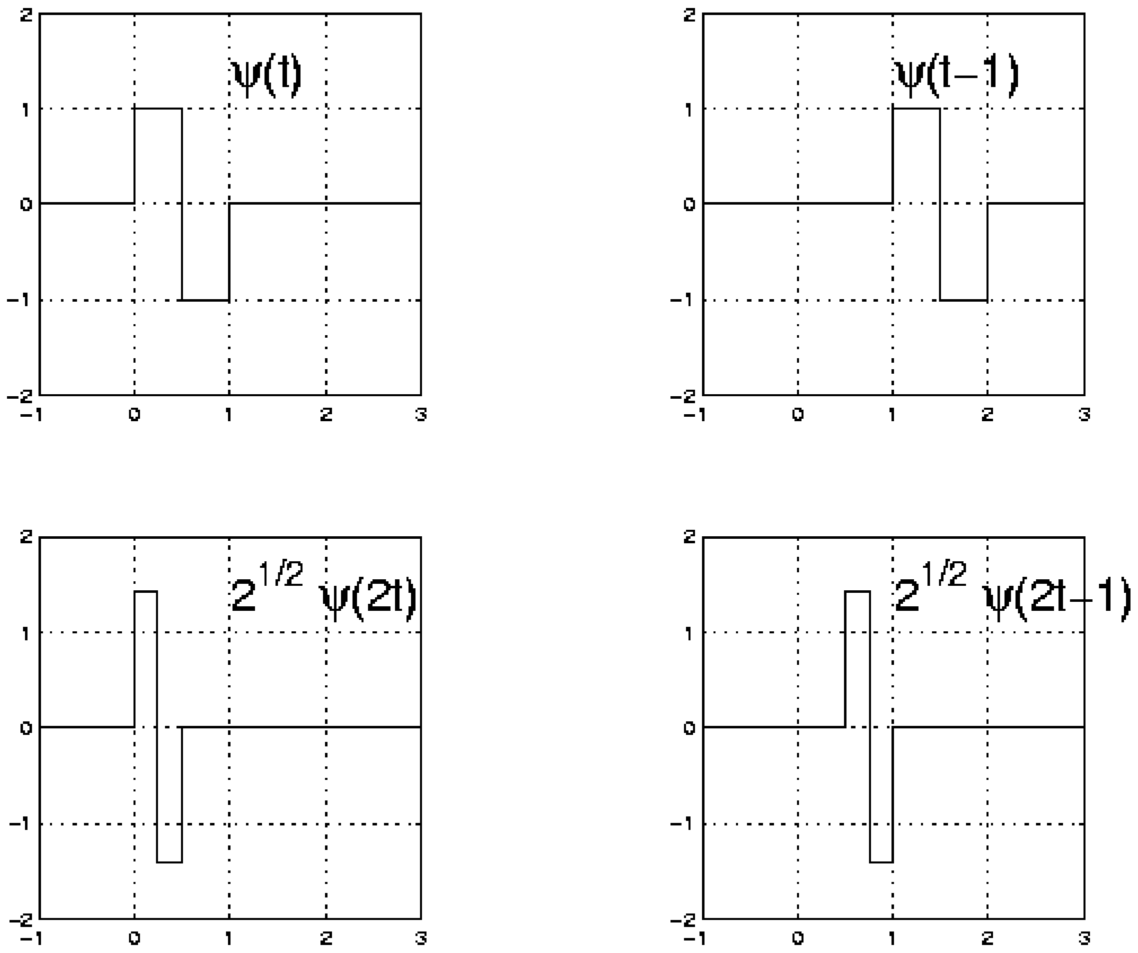


Figure 8.11

Notation:

$$\psi_{j,k}(t) = 2^{\frac{j}{2}} \psi\left(2^j t - k\right)$$

where j is an index of **scale** and k is an index of **location**.

If B is an orthonormal set then we have the wavelet series.

Wavelet series

$$x(t) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} d(j,k)\, \psi_{j,k}(t) \tag{8.8}$$

$$d(j,k) = \int_{-\infty}^{\infty} x(t)\, \psi_{j,k}(t)\, dt$$

8.3 Continuous Wavelet Transform[7]

The STFT provided a means of (joint) time-frequency analysis with the property that spectral/temporal widths (or resolutions) were the same for all basis elements. Let's now take a closer look at the implications of uniform resolution.

Consider two signals composed of sinusoids with frequency 1 Hz and 1.001 Hz, respectively. It may be difficult to distinguish between these two signals in the presence of background noise unless many cycles are observed, implying the need for a many-second observation. Now consider two signals with pure frequencies of 1000 Hz and 1001 Hz-again, a 0.1% difference. Here it should be possible to distinguish the two signals in an interval of much less than one second. In other words, good frequency resolution requires longer observation times as frequency decreases. Thus, it might be more convenient to construct a basis whose elements have larger temporal width at low frequencies.

The previous example motivates a multi-resolution time-frequency tiling of the form (Figure 8.12):

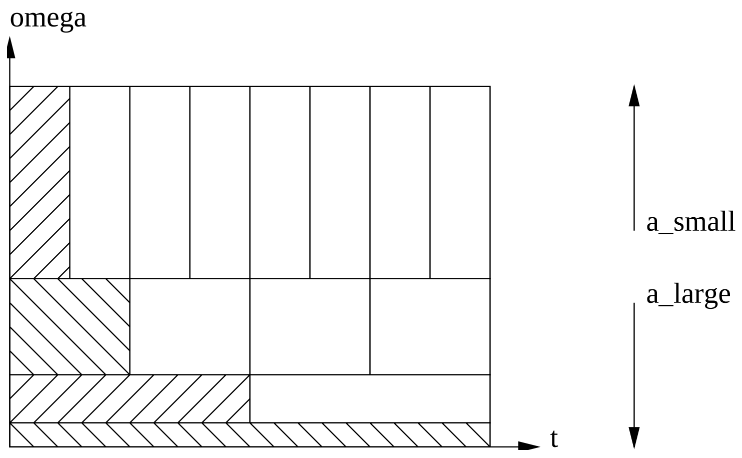


Figure 8.12

The Continuous Wavelet Transform (CWT) accomplishes the above multi-resolution tiling by time-scaling and time-shifting a prototype function $\psi(t)$, often called the **mother wavelet**. The a-scaled and τ-shifted basis elements is given by

$$\psi_{a,\tau}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-\tau}{a}\right)$$

where

$$a \ \ and \ \ \tau \in \mathbb{R}$$

$$\int_{-\infty}^{\infty} \psi(t)\, dt = 0$$

$$C_\psi = \int_{-\infty}^{\infty} \frac{\left(|\psi(\Omega)|\right)^2}{|\Omega|} d\Omega < \infty$$

[7]This content is available online at <http://cnx.org/content/m10418/2.14/>.

The conditions above imply that $\psi(t)$ is bandpass and sufficiently smooth. Assuming that $\| \psi(t) \| = 1$, the definition above ensures that $\| \psi_{a,\tau}(t) \| = 1$ for all a and τ. The CWT is then defined by the transform pair

$$X_{\mathrm{CWT}}(a,\tau) = \int_{-\infty}^{\infty} x(t)\,\psi_{a,\tau}(t)^{*} dt$$

$$x(t) = \frac{1}{C_{\psi}} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{X_{\mathrm{CWT}}(a,\tau)\,\psi_{a,\tau}(t)}{a^2} d\tau da$$

In basis terms, the CWT says that a waveform can be decomposed into a collection of shifted and stretched versions of the mother wavelet $\psi(t)$. As such, it is usually said that wavelets perform a "time-scale" analysis rather than a time-frequency analysis.

The **Morlet wavelet** is a classic example of the CWT. It employs a windowed complex exponential as the mother wavelet:

$$\psi(t) = \frac{1}{\sqrt{2\pi}} e^{-(j\Omega_0 t)} e^{-\frac{t^2}{2}}$$

$$\Psi(\Omega) = e^{-\frac{(\Omega-\Omega_0)^2}{2}}$$

where it is typical to select $\Omega_0 = \pi\sqrt{\frac{2}{\log 2}}$. (See illustration (Figure 8.13).) While this wavelet does not exactly satisfy the conditions established earlier, since $\Psi(0) \simeq 7 \times 10^{-7} \neq 0$, it can be corrected, though in practice the correction is negligible and usually ignored.

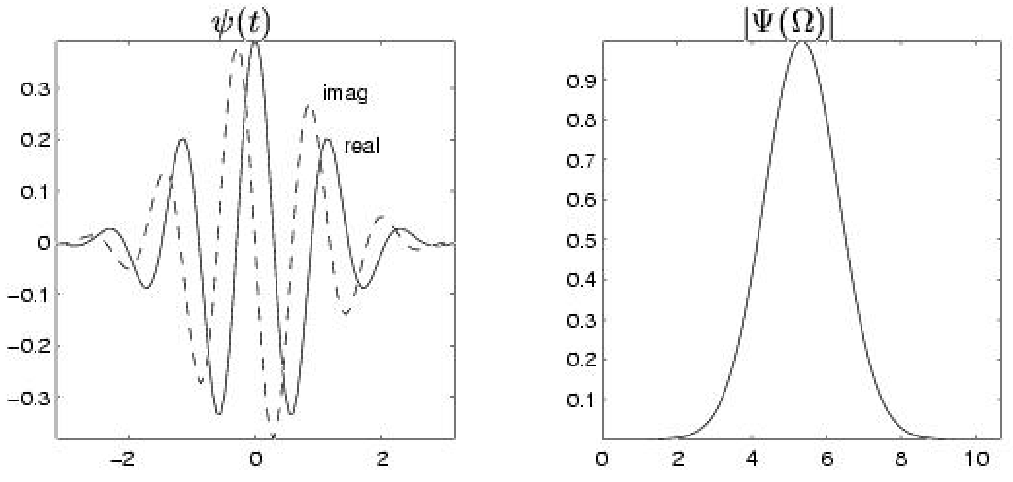


Figure 8.13

While the CWT discussed above is an interesting theoretical and pedagogical tool, the discrete wavelet transform (DWT) is much more practical. Before shifting our focus to the DWT, we take a step back and review some of the basic concepts from the branch of mathematics known as Hilbert Space theory (Vector

Space[8], Normed Vector Space[9], Inner Product Space[10], Hilbert Space[11], Projection Theorem[12]). These concepts will be essential in our development of the DWT.

8.4 Discrete Wavelet Transform: Main Concepts[13]

8.4.1 Main Concepts

The **discrete wavelet transform** (DWT) is a representation of a signal $x(t) \in \mathcal{L}_2$ using an orthonormal basis consisting of a countably-infinite set of **wavelets**. Denoting the wavelet basis as $\{\psi_{k,n}(t) \mid k \in \mathbb{Z} \ \ and \ \ n \in \mathbb{Z}\}$, the DWT transform pair is

$$x(t) = \sum_{k=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} d_{k,n} \psi_{k,n}(t) \tag{8.9}$$

$$\begin{aligned} d_{k,n} &= < \psi_{k,n}(t), x(t) > \\ &= \int_{-\infty}^{\infty} \psi_{k,n}(t)^* x(t)\, dt \end{aligned} \tag{8.10}$$

where $\{d_{k,n}\}$ are the wavelet coefficients. Note the relationship to Fourier series and to the sampling theorem: in both cases we can perfectly describe a continuous-time signal $x(t)$ using a countably-infinite (*i.e.*, discrete) set of coefficients. Specifically, Fourier series enabled us to describe **periodic** signals using Fourier coefficients $\{X[k] \mid k \in \mathbb{Z}\}$, while the sampling theorem enabled us to describe **bandlimited** signals using signal samples $\{x[n] \mid n \in \mathbb{Z}\}$. In both cases, signals within a limited class are represented using a coefficient set with a single countable index. The DWT can describe **any** signal in $\mathcal{L}_2$ using a coefficient set parameterized by two countable indices: $\{d_{k,n} \mid k \in \mathbb{Z} \ \ and \ \ n \in \mathbb{Z}\}$.

Wavelets are orthonormal functions in $\mathcal{L}_2$ obtained by shifting and stretching a **mother wavelet**, $\psi(t) \in \mathcal{L}_2$. For example,

$$\psi_{k,n}(t) = 2^{-\frac{k}{2}} \psi\left(2^{-k}t - n\right) \quad , \quad k \ \ and \ \ n \in \mathbb{Z} \tag{8.11}$$

defines a family of wavelets $\{\psi_{k,n}(t) \mid k \in \mathbb{Z} \ \ and \ \ n \in \mathbb{Z}\}$ related by power-of-two stretches. As k increases, the wavelet stretches by a factor of two; as n increases, the wavelet shifts right.

NOTE: When $\| \psi(t) \| = 1$, the normalization ensures that $\| \psi_{k,n}(t) \| = 1$ for all $k \in \mathbb{Z}$, $n \in \mathbb{Z}$.

Power-of-two stretching is a convenient, though somewhat arbitrary, choice. In our treatment of the discrete wavelet transform, however, we will focus on this choice. Even with power-of two stretches, there are various possibilities for $\psi(t)$, each giving a different flavor of DWT.

Wavelets are constructed so that $\{\psi_{k,n}(t) \mid n \in \mathbb{Z}\}$ (*i.e.*, the set of all shifted wavelets at fixed scale k), describes a particular level of 'detail' in the signal. As k becomes smaller (*i.e.*, closer to $-\infty$), the wavelets become more "fine grained" and the level of detail increases. In this way, the DWT can give a **multi-resolution** description of a signal, very useful in analyzing "real-world" signals. Essentially, the DWT gives us a **discrete multi-resolution description of a continuous-time signal in $\mathcal{L}_2$**.

In the modules that follow, these DWT concepts will be developed "from scratch" using Hilbert space principles. To aid the development, we make use of the so-called **scaling function** $\phi(t) \in \mathcal{L}_2$, which will be used to approximate the signal **up to a particular level of detail**. Like with wavelets, a family of scaling functions can be constructed via shifts and power-of-two stretches

$$\phi_{k,n}(t) = 2^{-\frac{k}{2}} \phi\left(2^{-k}t - n\right) \quad , \quad k \ \ and \ \ n \in \mathbb{Z} \tag{8.12}$$

[8]"Vector Space" <http://cnx.org/content/m10419/latest/>
[9]"Normed Vector Space" <http://cnx.org/content/m10428/latest/>
[10]"Inner Product Space" <http://cnx.org/content/m10430/latest/>
[11]"Hilbert Spaces" <http://cnx.org/content/m10434/latest/>
[12]"Projection Theorem" <http://cnx.org/content/m10435/latest/>
[13]This content is available online at <http://cnx.org/content/m10436/2.12/>.

given mother scaling function $\phi(t)$. The relationships between wavelets and scaling functions will be elaborated upon later via theory[14] and example (Section 8.5).

> NOTE: The inner-product expression for $d_{k,n}$, (8.10) is written for the general complex-valued case. In our treatment of the discrete wavelet transform, however, we will assume real-valued signals and wavelets. For this reason, we omit the complex conjugations in the remainder of our DWT discussions

8.5 The Haar System as an Example of DWT[15]

The Haar basis is perhaps the simplest example of a DWT basis, and we will frequently refer to it in our DWT development. Keep in mind, however, that **the Haar basis is only an example**; there are many other ways of constructing a DWT decomposition.

For the Haar case, the mother **scaling function** is defined by (8.13) and Figure 8.14.

$$\phi(t) = \begin{cases} 1 & \text{if } 0 \le t < 1 \\ 0 & \text{otherwise} \end{cases} \quad (8.13)$$

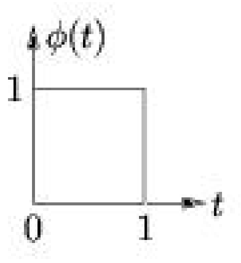


Figure 8.14

From the mother scaling function, we define a family of shifted and stretched scaling functions $\{\phi_{k,n}(t)\}$ according to (8.14) and Figure 8.15

$$\begin{aligned} \phi_{k,n}(t) &= 2^{-\frac{k}{2}}\phi\left(2^{-k}t - n\right) \quad , \quad k \in \mathbb{Z} \;\; n \in \mathbb{Z} \\ &= 2^{-\frac{k}{2}}\phi\left(\frac{1}{2^k}\left(t - n2^k\right)\right) \end{aligned} \quad (8.14)$$

[14]"The Scaling Equation" <http://cnx.org/content/m10476/latest/>
[15]This content is available online at <http://cnx.org/content/m10437/2.10/>.

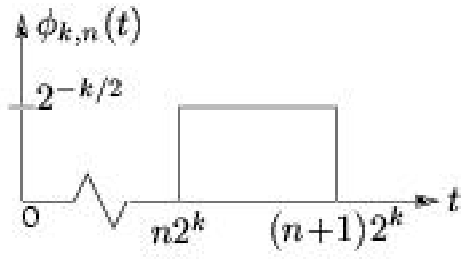


Figure 8.15

which are illustrated in Figure 8.16 for various k and n. (8.14) makes clear the principle that incrementing n by one shifts the pulse one place to the right. Observe from Figure 8.16 that $\{\phi_{k,n}(t) \mid n \in \mathbb{Z}\}$ is orthonormal for each k (*i.e.*, along each row of figures).

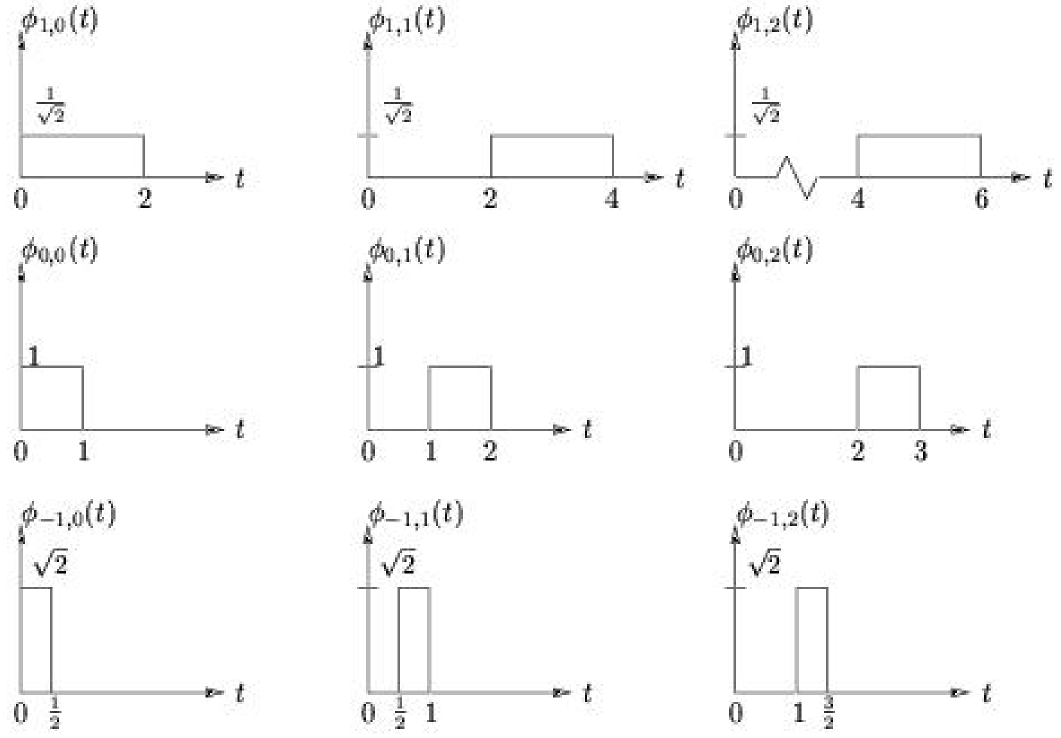


Figure 8.16

Glossary

A Autocorrelation

the expected value of the product of a random variable or signal realization with a time-shifted version of itself

C Complex Plane

A two-dimensional graph where the horizontal axis maps the real part and the vertical axis maps the imaginary part of any complex number or function.

Correlation

A measure of how much one random variable depends upon the other.

Covariance

A measure of how much the deviations of two or more variables or processes match.

Crosscorrelation

if two processes are wide sense stationary, the expected value of the product of a random variable from one random process with a time-shifted, random variable from a different random process

D difference equation

An equation that shows the relationship between consecutive values of a sequence and the differences among them. They are often rearranged as a recursive formula so that a systems output can be computed from the input signal and past outputs.

Example:

$$y[n] + 7y[n-1] + 2y[n-2] = x[n] - 4x[n-1] \tag{3.1}$$

domain

The group, or set, of values that are defined by a given function.

Example: Using the rational function above, [16] , the domain can be defined as any real number x where x does not equal 1 or negative 3. Written out mathematical, we get the following:

$$\{x \in \mathbb{R} \mid (x \neq -3) \quad and \quad (x \neq 1)\} \tag{3.45}$$

F FFT

(Fast Fourier Transform) An efficient **computational algorithm** for computing the DFT[17].

First-order stationary process

If $F_{X_t}(b)$ is not a function of time then X_t is called a first-order stationary process.

P poles

1. The value(s) for z where $Q(z) = 0$.

[16]http://cnx.org/content/m10593/latest/
[17]"Discrete Fourier Transform (DFT)" <http://cnx.org/content/m10249/latest/>

2. The complex frequencies that make the overall gain of the filter transfer function infinite.

poles

1. The value(s) for z where the **denominator** of the transfer function equals zero
2. The complex frequencies that make the overall gain of the filter transfer function infinite.

R random process

A family or ensemble of signals that correspond to every possible outcome of a certain signal measurement. Each signal in this collection is referred to as a **realization** or **sample function** of the process.

Example: As an example of a random process, let us look at the Random Sinusoidal Process below. We use $f[n] = A\sin(\omega n + \phi)$ to represent the sinusoid with a given amplitude and phase. Note that the phase and amplitude of each sinusoid is based on a random number, thus making this a random process.

rational function

For any two polynomials, A and B, their quotient is called a rational function.

Example: Below is a simple example of a basic rational function, $f(x)$. Note that the numerator and denominator can be polynomials of any order, but the rational function is undefined when the denominator equals zero.

$$f(x) = \frac{x^2 - 4}{2x^2 + x - 3} \tag{3.43}$$

S stationary process

a random process where all of its statistical properties do not vary with time

Stochastic Process

Given a sample space, a stochastic process is an indexed collection of random variables defined for each $\omega \in \Omega$.

$$X_t(\omega) \quad , \quad t \in \mathbb{R} \tag{2.1}$$

Z zeros

1. The value(s) for z where $P(z) = 0$.
2. The complex frequencies that make the overall gain of the filter transfer function zero.

zeros

1. The value(s) for z where the **numerator** of the transfer function equals zero
2. The complex frequencies that make the overall gain of the filter transfer function zero.

Bibliography

[1] S. Haykin. *Adaptive Filter Theory.* Prentice Hall, 3rd edition edition, 1996.

Index of Keywords and Terms

Keywords are listed by the section with that keyword (page numbers are in parentheses). Keywords do not necessarily appear in the text of the page. They are merely associated with that section. *Ex.* apples, § 1.1 (1) **Terms** are referenced by the page they appear on. *Ex.* apples, 1

Attributions

Intro to Digital Signal Processing
The course provides an introduction to the concepts of digital signal processing (DSP). Some of the main topics covered include DSP systems, image restoration, z-transform, FIR filters, adaptive filters, wavelets, and filterbanks.