SSY130 - Applied Signal Processing
Lectures notes

Tomas McKelvey
Signal Processing
Electrical Engineering
Chalmers

2021

Preface

This document contains lecture notes associated with the lectures given in the course SSY130 - Applied Signal Processing. The 15 chapters in the document cover the material presented in the lectures in the course. The presentation of the material assumes basic knowledge of mathematical analysis, random variables, linear algebra, Fourier analysis and signals and systems.

The ordering of the material in the lectures is highly influenced by the two design projects which is part och the course. The scheduling of the lecture material is designed to be in sync with the project work.

Gothenburg, October 27, 2021

Tomas McKelvey

Contents

Chapter 1

Introduction and continuous time signals and systems

Introduction to the DSP subject

DSP means digital signal processing or digital signal processors. The former refers to the processing techniques and mathematical algorithms while the latter refers to specific electric hardware dedicated to signal processing.

Signals are everywhere around us:

- Electric: Voltages, currents and electromagnetic fields
- Acoustic: Sound waves in air and water
- Mechanic: Vibrations, angular motion, velocities, forces, moments and pressures
- Thermic: Temperatures

All signal above are continuous in time or in space (or both). Sampling of such signals enables algorithmic processing in a computer. The availability and rapid development of low-cost, low-power digital circuits, processors and large digital memory has lead to an enormous development of applications where DSP is an integral part. Examples are:

- Speech-Audio-Music-Video-Multimedia: noise-reduction, coding, compression, enhancement (CD, MPEG, MP3, DVB, DAB,DVD,Blue-Ray)
- Radio, Mobile phones: modulation, transmission, coding (QPSK, OFDM)
- Radar, Sonar: filtering, target detection, tracking
- Control: servomechanisms, setpoint control and disturbance rejection
- Biomedical: biomedical signal analysis (ECG), imaging (CT, MRI), monitoring, telemedicin, e-Health

What is a DSP system

- Processing of signals: filtering, modulation etc. ⇒ Change the signal
- Analysis of signals: transforms (Fourier analysis), model based analysis ⇒ Extract features from the signal
- Detection and Classification: Take final decisions based on the signal

Common SP - Structure

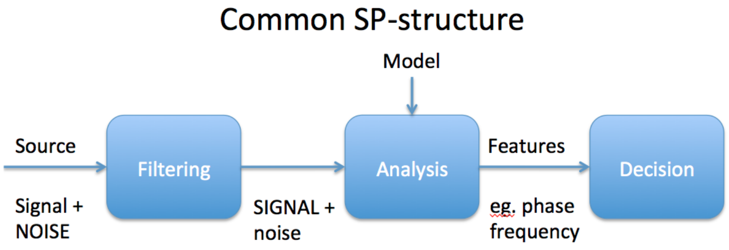


Figure 1.1: Common signal processing structure.

Examples:

- Digital radio receiver
- Radar system tracking air-traffic
- Electrocardiography (ECG) analysis

How is DSP implemented

The basic building block is gate logic. The same as for all other digitally based computing platforms. There exists different layers of abstraction:

- Desktop computers and software
- Dedicated signal processing processors and software
- Embedded processors and software
- Application specific integrated circuits (ASIC)
- Field programmable gate arrays (FPGA)

Properties of *D*SP

- (-) Signal is always sampled and quantized ⇒non-exact representation
- (-) Analog to digital (AD) and digital to analog (DA) converters necessary ⇒cost and power consumption

- (-) Limited bandwidth due to limits in clock frequency of the digital circuitry.
- (+) Control of accuracy and no aging (no need calibrate a digital algorithm!)
- (+) Very complex algorithms can be realized
- (+) High degree of flexibility (software + mass produced standardized components)
- (+) Adaptivity - “on-line” changes are possible to incorporate
- (+) Very low-frequencies are easily handled

A short review of Signals and Systems theory

Signal processing classically deals with exploiting the spectral content. The reason for this is that the complex exponential $e^{(\gamma+j\omega)t}$ form a basis for all solutions to linear dynamic systems. The Fourier and Laplace transforms are then the natural tools for analysis.

Fourier transform (FT)

Consider a signal $x(t)$ where the argument t denotes the signals dependency on time and has unit seconds. For a signal $x(t)$, the Fourier transform and its inverse are defined[1] as

$$\begin{aligned} X(\omega) \triangleq \mathrm{FT}[x(t)] &\triangleq \int_{-\infty}^{\infty} x(t)e^{-j\omega t}\,dt \\ x(t) = \mathrm{FT}^{-1}[X(\omega)] &\triangleq \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{j\omega t}\,d\omega \end{aligned} \tag{1.1}$$

here $j \triangleq \sqrt{-1}$ and where the argument ω denote frequency in unit radians/seconds and hence $\omega = 2\pi f$ where f denotes frequency with unit 1/seconds (Hertz). Note that the Fourier transform is a linear operator and we have the important property

$$\mathrm{FT}[\alpha_1 x_1(t) + \alpha_2 x_2(t)] = \alpha_1\mathrm{FT}[x_1(t)] + \alpha_2\mathrm{FT}[x_2(t)] \tag{1.2}$$

where α_1 and α_2 are real or complex scalar constants. The signal energy can be expressed as (Parseval's relation)

$$E = \int_{-\infty}^{\infty} |x(t)|^2\,dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |X(\omega)|^2\,d\omega \tag{1.3}$$

This means that $|X(\omega)|^2$ illustrates how the energy in the signal is distributed over the frequencies.

Using the theory of distributions simplifies the treatment of some infinite energy signals, such as sinusoidal signals, when using the Fourier transform. For such signals the classical FT does not exist. The Dirac delta function $\delta(t)$ play a key role in the analysis. A Dirac delta function is implicitly defined through the following relations where $f(t)$ is a "well behaved" function.

$$\begin{gathered} \delta(t) = 0, \quad \forall t \neq 0 \\ \int_{-\infty}^{\infty} \delta(t)f(t)\,dt = f(0) \end{gathered} \tag{1.4}$$

A possible constructive way of defining the delta function is by considering the function

$$d_\epsilon(t) = \begin{cases} 1/\epsilon & 0 \le t < \epsilon \\ 0 & \text{otherwise.} \end{cases} \tag{1.5}$$

[1] We use the notation $x \triangleq y/2$ to define the symbol x to be equal to $y/2$. A simple equality sign $=$ denotes that the expression on the left hand side is equal to the right hand side and is used in algebraic derivations.

	Time dom.	Fourier dom.
Time shift	$x(t-t_0)$	$e^{-j\omega t_0}X(\omega)$
Product	$x(t)y(t)$	$\frac{1}{2\pi}\int_{-\infty}^{\infty} X(\lambda)Y(\omega-\lambda)\,d\lambda$
Modulation	$e^{j\omega_0 t}x(t)$	$X(\omega-\omega_0)$
Differentiation	$\frac{d}{dt}x(t)$	$j\omega X(\omega)$
Constant	1	$2\pi\delta(\omega)$
Delta fun.	$\delta(t)$	1
Convolution	$\int_{-\infty}^{\infty} h(\tau)x(t-\tau)\,d\tau$	$H(\omega)X(\omega)$

Table 1.1: Fourier transform properties where $X(\omega) = \mathrm{FT}[x(t)]$

Then in the limit we have

$$\lim_{\epsilon\to 0} d_\epsilon(t) = \delta(t) \tag{1.6}$$

Scaling the argument of a delta function with a real constant $\alpha > 0$ yields the following identity

$$\int_{-\infty}^{\infty} f(t)\delta(\alpha t)\,dt = [t = \tau/\alpha] = \int_{-\infty}^{\infty} f(\tau/\alpha)\delta(\tau)\,\frac{d\tau}{\alpha} = \frac{f(0)}{\alpha} \tag{1.7}$$

Some of the Fourier transform pairs are listed in Table 1.1. As in illustration we proceed by showing the second result in Table 1.1. Let $p(t) = x(t)y(t)$ be the product signal. Then

$$\begin{aligned} P(\omega) &= \int_{-\infty}^{\infty} p(t)e^{-j\omega t}\,dt = \int_{-\infty}^{\infty} x(t)y(t)e^{-j\omega t}\,dt \\ &= \int_{-\infty}^{\infty}\left[\frac{1}{2\pi}\int_{-\infty}^{\infty} X(\lambda)e^{j\lambda t}\,d\lambda\right] y(t)e^{-j\omega t}\,dt \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\lambda)\left[\int_{\infty}^{\infty} y(t)e^{-j(\omega-\lambda)t}\,dt\right]d\lambda \\ &== \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\lambda)Y(\omega-\lambda)\,d\lambda \end{aligned} \tag{1.8}$$

Example 1 Let's show the first relation in Table 1.1. Using the definition we obtain

$$\begin{aligned} \mathrm{FT}[x(t-t_0)] &= \int_{-\infty}^{\infty} x(t-t_0)e^{-j\omega t}\,dt = [\tau = t-t_0] \\ &= \int_{-\infty}^{\infty} x(\tau)e^{-j\omega(\tau+t_0)}\,d\tau \\ &= e^{-j\omega t_0}\int_{-\infty}^{\infty} x(\tau)e^{-j\omega\tau}\,d\tau \\ &= e^{-j\omega t_0}X(\omega) \end{aligned} \tag{1.9}$$

□

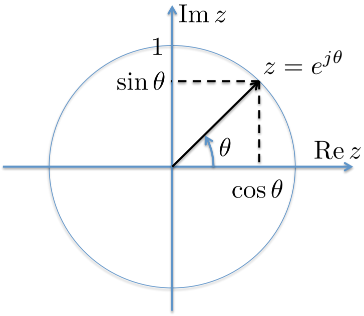


Figure 1.2: Unit circle with vector $z = e^{j\theta} = \cos\theta + j\sin\theta$.

Recall the Euler's relations for complex numbers:

$$\begin{aligned} e^{j\theta} &= \cos\theta + j\sin\theta \\ \cos\theta &= \frac{1}{2}(e^{j\theta} + e^{-j\theta}) = \operatorname{Re}(e^{j\theta}) \\ \sin\theta &= \frac{1}{2j}(e^{j\theta} - e^{-j\theta}) = \operatorname{Im}(e^{j\theta}) \end{aligned} \tag{1.10}$$

The relations can also graphically be illustrated as shown in Figure 1.2.

Example 2 With the properties of Euler's relations we obtain

$$\mathrm{FT}[\cos(\omega_0 t)] = \frac{1}{2}\mathrm{FT}[e^{j\omega_0 t} + e^{-j\omega_o t}] = \frac{2\pi}{2}\left(\delta(\omega-\omega_0) + \delta(\omega+\omega_0)\right) \tag{1.11}$$

The energy of a real valued sinusoidal signal is consequently represented in the Fourier domain by two delta functions located at $\pm\omega_0$.

□

Linear systems

Many signal processing operations, such as linear filtering, can be viewed as the result of a linear system operating on the input signal $x(t)$ and producing an output signal $y(t)$. Mathematically this operation is known as filtering or convolution and is defined by

$$y(t) = x(t) * h(t) \triangleq \int_{-\infty}^{\infty} h(\tau)x(t-\tau)\,d\tau \tag{1.12}$$

where $h(\tau)$ is the impulse response of the linear system, $x(t)$ is the input signal and $y(t)$ us the resulting output signal. Note that

$$\begin{aligned} x(t) * h(t) &= \int_{-\infty}^{\infty} h(\tau)x(t-\tau)\,d\tau = \\ &= \int_{-\infty}^{\infty} x(\tau)h(t-\tau)\,d\tau = h(t) * x(t) \end{aligned} \tag{1.13}$$

which means that the convolution operator is commutative (the order between x and h is indifferent). Inspecting (1.12) we notice that the output at time t

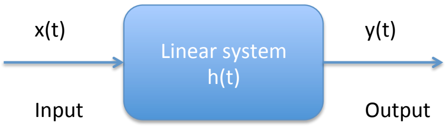


Figure 1.3: Linear system with impulse response $h(t)$

is a weighted integration of the input signal where the weighting function is given by the impulse response. If the impulse response is zero for all values of $\tau < 0$, then the output at time t only depends on past (and the present) values of the input signal. Such a system is called *causal*. For an *anti-causal* system the output depends only on the future values (and present) of the input while in a *non-causal* system the output can depend on both future and past values of the input.

As shown in Table 1.1 the convolution operation is particularly simple in the Fourier domain since

$$Y(\omega) = H(\omega)X(\omega) \tag{1.14}$$

The Fourier transform of the output of the linear dynamic system, which is a complex function, can be written as the (complex) product between the Fourier transforms of the input and impulse response respectively. Finally, we note that filtering/convolution is a linear operation and hence,

$$h(t) * (\alpha_1 x_1(t) + \alpha_2 x_2(t)) = \alpha_1(h(t) * x_1(t)) + \alpha_2(h(t) * x_2(t)). \tag{1.15}$$

It is clear from the definition of convolution in (1.12) that if $x_1(t) = x(t - t_0)$ is the input signal then the output will be $y_1(t) = y(t - t_0)$. This means that the dynamic behaviour of the system is invariant with respect to the absolute time, i.e. the impulse response function is unchanged. To highlight this property we call such systems *time invariant*. Together with the linear property we get linear time invariant (LTI) systems.

Complex exponential input

Consider the complex input signal $x(t) = e^{j\omega_0 t}$. Then $X(\omega) = \text{FT}[x(t)] = 2\pi\delta(\omega - \omega_0)$. If the signal $x(t)$ is filtered by a linear system with impulse response $h(t)$ we obtain

$$\begin{aligned} Y(\omega) &= H(\omega)X(\omega) = H(\omega)2\pi\delta(\omega - \omega_0) \quad \Rightarrow \\ y(t) &= \text{FT}^{-1}[Y(\omega)] = \frac{1}{2\pi}\int_{-\infty}^{\infty} H(\omega)2\pi\delta(\omega - \omega_0)e^{j\omega t}\, d\omega = \\ &= H(\omega_0)e^{j\omega_0 t} \end{aligned} \tag{1.16}$$

where $H(\omega_0)$ is a complex scalar which makes the output signal a scaled and phase shifted version of the input signal. The complex function $H(\omega)$ is called the transfer function or frequency function of the filter (linear system). The magnitude and the phase of this function are known as amplitude and phase functions of the filter

$$\begin{array}{ll} \text{Amplitude function:} & A(\omega) \triangleq |H(\omega)| \\ \text{Phase function:} & \phi(\omega) \triangleq \angle H(\omega) \end{array} \tag{1.17}$$

Note that both the amplitude and phase functions are real-valued. Hence we can rewrite the output in (1.16) as

$$y(t) = A(\omega_0)e^{j(\omega_0 t+\phi(\omega_0))} \tag{1.18}$$

Example 3 Assume $H(\omega)$ is the frequency function for a filter with a real-valued impulse response. Let $x(t) = A\cos(\omega_0 t + \gamma)$ be the signal filtered through the linear filter. Since

$$x(t) = \frac{A}{2}\left[e^{j(\omega_0 t+\gamma)} + e^{-j(\omega_0 t+\gamma)}\right] = \frac{A}{2}\left[e^{j\gamma}e^{j\omega_0 t} + e^{-j\gamma}e^{-j\omega_0 t}\right] \tag{1.19}$$

we obtain

$$Y(\omega) = H(\omega)\frac{2A\pi}{2}\left[e^{j\gamma}\delta(\omega-\omega_0) + e^{-j\gamma}\delta(\omega+\omega_0)\right]. \tag{1.20}$$

Using the inverse FT gives the time domain expression for the output

$$y(t) = FT^{-1}[Y(\omega)] = \frac{A}{2}\left[H(\omega_0)e^{j(\omega_0 t+\gamma)} + H(-\omega_0)e^{-j(\omega_0 t+\gamma)}\right]. \tag{1.21}$$

Since the impulse response is real-valued $|H(\omega_0)| = |H(-\omega_0)|$ and $\arg H(\omega) = -\arg H(-\omega)$. With this we can simplify the output expression to

$$y(t) = |H(\omega_0)|A\cos(\omega_0 t + \gamma + \arg H(\omega_0)) \tag{1.22}$$

□

An alternative derivation of the result in (1.16) can be obtained by directly considering the convolution equation

$$y(t) = \int_{-\infty}^{\infty} h(\tau)e^{j\omega_0(t-\tau)}\,d\tau = \int_{-\infty}^{\infty} h(\tau)e^{-j\omega_0\tau}\,d\tau e^{j\omega_0 t} = H(\omega_0)e^{j\omega_0 t} \tag{1.23}$$

Chapter 2

Discrete time systems and sampling

Discrete time systems

Using clocked digital circuits for processing requires analysis of sampled data also called discrete time signals. A discrete time signal is an enumerated collection of values $\{x(n)\}$ defined for integer indices $n = 0, \pm 1, \pm 2, \ldots$. A discrete time signal is often the result of sampling a continuous time signal $x_c(t)$ at regular time intervals

$$x(n) \triangleq x_c(n\Delta t), \quad n = 0, \pm 1, \pm 2, \ldots \tag{2.1}$$

where Δt is the sampling interval. Notation wise we make no distinction between continuous time signals and discrete time signals but use the convention that a continuous time signal have a real valued argument t or τ while the discrete time counterpart has the integer argument n, m or k.

Associated with the sampling period we also use several other definitions as listed below:

sampling period	Δt	[s]
sampling frequency:	$f_s \triangleq \frac{1}{\Delta t}$	[Hz=1/s]
sampling frequency:	$\omega_s \triangleq \frac{2\pi}{\Delta t}$	[rad/s]
Nyquist frequency:	$\frac{f_s}{2}$	[Hz]

Often we use dimensionless normalized frequencies f/f_s (or ω/ω_s). Normalized frequency 1 thus corresponds to the sampling frequency and 0.5 corresponds to the Nyquist frequency.

Discrete time Fourier transform (DTFT)

For discrete time signals the Fourier analysis tool is called Discrete time Fourier transform (DTFT) and is defined as

$$X(\omega) = \mathrm{DTFT}[x(n)] \triangleq \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n\Delta t} \tag{2.2}$$

where ω has unit radians per second. For integer values k we note that $X(\omega + k\omega_s) = X(\omega)$ since

$$e^{-j(\omega+k\omega_s)n\Delta t} = e^{-j\omega n\Delta t - j2\pi kn} = e^{-j\omega n\Delta t} \tag{2.3}$$

where the last equality follows from the fact that adding multiples of 2π to the argument (angle) of a complex number leaves the value unchanged. Consequently, the DTFT is a periodic function with a period of $\omega_s = 2\pi f_s = 2\pi/\Delta t$.

The inverse of the DTFT is given by the relation

$$x(n) = \mathrm{DTFT}^{-1}[X(\omega)] = \frac{1}{\omega_s}\int_0^{\omega_s} X(\omega)e^{j\omega n\Delta t}\, d\omega \tag{2.4}$$

Since $X(\omega)$ and $e^{j\omega n\Delta t}$ are periodic functions with period ω_s, the start of the integration interval can be arbitrary as long as the length of the interval is ω_s. Common choices are $[0, \omega_s]$ or $[\frac{-\omega_s}{2}, \frac{\omega_s}{2}]$.

Discrete time convolution Convolution between discrete time signals $x(n)$ and $h(n)$ is defined as

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \tag{2.5}$$

i.e. the output signal $y(n)$ is a weighted sum of the input signal samples $x(n)$ where each sample is weighted by the signal $h(n)$. If we consider $y(n)$ as the output from a discrete time filter we call $h(n)$ the impulse response of the filter. It can be shown that the DTFT of $y(n)$ has the simle form

$$Y(\omega) = H(\omega)X(\omega) \tag{2.6}$$

where $H(\omega)$ and $X(\omega)$ are the DTFT of the input and impulse response respectively. We can show this result by considering the inverse DTFT of $Y(\omega)$ and use the fact that $H(\omega) = \sum_{k=-\infty}^{\infty} h(k)e^{-j\omega k\Delta t}$. This leads to,

$$\begin{aligned} y(n) &= \frac{1}{\omega_s}\int_0^{\omega_s} \sum_{k=-\infty}^{\infty} h(k)e^{-j\omega k\Delta t} X(\omega)e^{j\omega n\Delta t}\, d\omega = \\ &= \sum_{k=-\infty}^{\infty} h(k)\frac{1}{\omega_s}\int_0^{\omega_s} X(\omega)e^{j\omega(n-k)\Delta t}\, d\omega = \\ &= \sum_{k=-\infty}^{\infty} h(k)x(n-k) \end{aligned} \tag{2.7}$$

Hence we have established that discrete time convolution in the time domain is equivalent to multiplication in the frequency domain:

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \quad \Leftrightarrow \quad Y(\omega) = H(\omega)X(\omega) \tag{2.8}$$

Sometimes the operator $*$ is used to denote convolution:

$$x(n) * h(n) \triangleq \sum_{k=-\infty}^{\infty} h(k)x(n-k) \tag{2.9}$$

	Time dom.	Fourier dom.
Delay	$x(n-k)$	$e^{-j\Delta t\omega k}X(\omega)$
Modulation	$e^{j\omega_0\Delta tn}x(n)$	$X(\omega-\omega_0)$
Constant	1	$\omega_s\tilde{\delta}(\omega) \triangleq \omega_s\sum_{k=-\infty}^{\infty}\delta(\omega+k\omega_s)$
Kronecker delta	δ_n	1
Convolution	$\sum_{k=-\infty}^{\infty}h(k)x(n-k)$	$H(\omega)X(\omega)$
Frequency convolution	$x(n)w(n)$	$\frac{1}{\omega_s}\int_0^{\omega_s}X(\lambda)W(\omega-\lambda)\,d\lambda$

Table 2.1: DTFT transform pairs

Some useful DTFT transform pairs are summarized in Table 2.1. In the table we denote by δ_n the Kronecker delta function

$$\delta_n \triangleq \begin{cases} 1, & n=0 \\ 0, & n\neq 0 \end{cases} \tag{2.10}$$

which is the discrete time counterpart to the Dirac delta function. Note that the convolution of the Kronecker delta function with the impulse response results in the impulse response, i.e.

$$\delta_n * h(n) = \sum_{k=-\infty}^{\infty}\delta_k h(n-k) = h(n). \tag{2.11}$$

Parseval's relation for a discrete time signal $x(n)$ with finite energy is given by

$$\sum_{n=-\infty}^{\infty}|x(n)|^2 = \frac{1}{\omega_s}\int_0^{\omega_s}|X(\omega)|^2\,d\omega \tag{2.12}$$

The unit delay Let z^{-1} denote the unit delay operator with a behavior defined by

$$z^{-1}x(n) \triangleq x(n-1). \tag{2.13}$$

In digital circuits, this element is one of the basic building blocks and is also one of the key operations in discrete time filtering (convolution). Employing the DTFT of the delayed signal yields

$$\begin{aligned}\mathrm{DTFT}[x(n-1)] &= \sum_{n=-\infty}^{\infty}x(n-1)e^{-j\omega n\Delta t} = [n'=n-1] = \\ &= \sum_{n'=-\infty}^{\infty}x(n')e^{-j\omega n'\Delta t}e^{-j\omega\Delta t} = e^{-j\omega\Delta t}X(\omega).\end{aligned} \tag{2.14}$$

A one step time delay consequently rotates $X(\omega)$ an angle of $-\omega\Delta t$ radians but leaves the magnitude unchanged. Using a natural extension of the notation above we have

$$z^{-k}x(n) = x(n-k) \quad \text{and} \quad \mathrm{DTFT}[x(n-k)] = e^{-j\omega k\Delta t}X(\omega). \tag{2.15}$$

Example 4 Consider the following linear operation defining the output $y(n)$ as

$$y(n) \triangleq \alpha x(n) + \beta x(n-1) \tag{2.16}$$

Transforming this relation with the DTFT yields

$$Y(\omega) = (\alpha + \beta e^{-j\omega\Delta t})X(\omega) = H(\omega)X(\omega) \tag{2.17}$$

Analogous to the continuous time filtering, we can interpret $H(\omega)$ as the frequency function of the discrete time filtering operation.

Clearly with the definition

$$h(n) = \begin{cases} \alpha, & n = 0 \\ \beta, & n = 1 \\ 0, & \text{otherwise} \end{cases} \tag{2.18}$$

we can write (2.16) as a convolution $y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k)$ and we conclude that $h(n)$ in (2.18) is the impulse response of the filter when we interpret the operation (2.16) as filtering the signal $x(n)$. □

Example 5 Consider the following linear recursive operation defining the output $y(n)$ as a linear combination between the input and past output

$$y(n) \triangleq -\alpha y(n-1) + \beta x(n) \tag{2.19}$$

Transforming this relation with the DTFT yields (assuming $|\alpha| < 1$ so stability is assured)

$$Y(\omega) = -\alpha e^{-j\omega\Delta t}Y(\omega) + \beta X(\omega) \tag{2.20}$$

a rearrangement of the terms above then finally yields

$$Y(\omega) = \frac{\beta}{1 + \alpha e^{-j\omega\Delta t}}X(\omega) = H(\omega)X(\omega) \tag{2.21}$$

□

A general difference equation In general a finite dimensional difference equation can be expressed as

$$y(n) \triangleq -\sum_{k=1}^{n_a} a_k y(n-k) + \sum_{k=0}^{n_b} b_k x(n-k) \tag{2.22}$$

and in the transform domain (again assuming stability)

$$Y(\omega) = \frac{\sum_{k=0}^{n_b} b_k e^{-j\omega\Delta tk}}{1 + \sum_{k=1}^{n_a} a_k e^{-j\omega\Delta tk}}X(\omega) = H(\omega)X(\omega) \tag{2.23}$$

Here we notice that the frequency function is a rational function in the variable $e^{-j\omega\Delta t}$ which is a characteristic for discrete time systems.

Z-transform, zeros and roots If we define $z \triangleq e^{j\omega\Delta t}$ we obtain

$$H(z) = \frac{\sum_{k=0}^{n_b} b_k z^{-k}}{1 + \sum_{k=1}^{n_a} a_k z^{-k}} = \frac{b(z)}{a(z)} \tag{2.24}$$

where $a(z)$ and $b(z)$ are polynomials. The object $H(z)$ is called the transfer function of the linear system and is the Z-transform of the associated impulse response. The solutions to $a(z) = 0$ are called poles and the solutions to $b(z) = 0$ are called zeros of the transfer function. If the system is causal and all poles have magnitude strictly smaller than one, the system is stable.

The impulse response If we return to the frequency function of the general difference equation in (2.22) and use a series expansion (here we assume this is allowed from a mathematical point of view) we obtain

$$H(\omega) = \frac{\sum_{k=0}^{n_b} b_k e^{-j\omega k \Delta t}}{1 + \sum_{k=1}^{n_a} a_k e^{-j\omega k \Delta t}} = \sum_{k=0}^{\infty} h(k) e^{-j\omega k \Delta t} \tag{2.25}$$

where obviously $h(k)$ is the impulse response of the linear filter.

Sampling and Reconstruction

A full signal processing chain can be described as sampling a continuous time signal, processing the sampled signal and finally reconstruct a continuous signal from the sampled version. In order to be able to analyse a full digital signal processing chain it is useful to describe the sampling and reconstruction operations using a common language. Here we focus on using the Fourier analysis tool and we hence need to connect the continuous time Fourier transform with the Discrete time Fourier transform. This will enable a full FT description including effects of sampling, filtering (in the discrete time) and reconstruction back to a continuous time signal.

Sampling As previously indicated sampling can be described simply as

$$x_d(n) = x(n\Delta t) \tag{2.26}$$

where Δt is the period time of the sampling operation. It is not possible to employ FT analysis of the discrete time signal x_d since it is not defined for all times t. Instead we model this signal with a continuous time signal $x_c(t)$ with the same information as $x_d(n)$. We accomplish this by using Dirac delta functions

$$x_c(t) \triangleq \sum_{n=-\infty}^{\infty} \delta(t - n\Delta t) x(t) \tag{2.27}$$

Clearly $x_c(t)$ is zero except at the sampling times, where the delta function is scaled with the value of the continuous time signal at that time instance. Now employing the FT of $x_c(t)$ yields

$$\begin{aligned} X_c(\omega) &= \int_{-\infty}^{\infty} x_c(t) e^{-j\omega t}\, dt = \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta(t - n\Delta t) x(t) e^{-j\omega t}\, dt \\ &= \ldots = \sum_{n=-\infty}^{\infty} x(n\Delta t) e^{-j\omega n \Delta t} = \sum_{n=-\infty}^{\infty} x_d(n) e^{-j\omega n \Delta t} \end{aligned} \tag{2.28}$$

which we recognize as the DTFT of $x_d(n)$. In summary we have the identity

$$\mathrm{FT}[x_c(t)] = \mathrm{DTFT}[x_d(n)]. \tag{2.29}$$

To get the full picture we also need to express $X_c(\omega)$ (and thus also $X_d(\omega)$) in terms of $X(\omega)$. Start by expressing $x(t)$ exactly at the sampling time instances using the inverse FT:

$$\begin{aligned}
x_d(n) = x(n\Delta t) &= \int_{-\infty}^{\infty} X(\omega) e^{j\omega n\Delta t}\,\frac{d\omega}{2\pi} \\
&= \sum_{k=-\infty}^{\infty} \int_{k\omega_s}^{(k+1)\omega_s} X(\omega) e^{j\omega n\Delta t}\,\frac{d\omega}{2\pi} \\
&= [\text{Variable change: } \omega = \omega' + k\omega_s] \\
&= \sum_{k=-\infty}^{\infty} \int_{0}^{\omega_s} X(\omega' + k\omega_s) e^{j(\omega'+k\omega_s)n\Delta t}\,\frac{d\omega'}{2\pi} \\
&= \frac{1}{\omega_s} \int_0^{\omega_s} \underbrace{\frac{1}{\Delta t} \sum_{k=-\infty}^{\infty} X(\omega' + k\omega_s)}_{X_d(\omega')} e^{j\omega' n\Delta t}\, d\omega'
\end{aligned} \tag{2.30}$$

where we recognize the last expression to be the inverse DTFT of the signal $x_d(n)$. In (2.30) we have shown

$$X_d(\omega) = X_c(\omega) = \frac{1}{\Delta t} \sum_{k=-\infty}^{\infty} X(\omega + k\omega_s) \tag{2.31}$$

which tell us that the transform of the sampled signal is formed from the summation of an infinite number of segements of the Fourier transform of $x(t)$, each with a length ω_s.

Chapter 3

Signal reconstruction

Ideal reconstruction of band-limited signals

The operation to convert a discrete time signal to a continuous time signal is known as *reconstruction.* Assume that a continuous signal $x(t)$ is band-limited such that $X(\omega) = 0$ for all $|\omega| > \omega_s/2$. Then the continuous signal $x(t)$ can be perfectly recovered from the discrete time samples $x_d(n) = x(n\Delta t)$. To see this consider the following derivation based on inverse FT of $X(\omega)$.

$$\begin{aligned}
x(t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{j\omega t}\,d\omega = \frac{1}{2\pi}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} X(\omega)e^{j\omega t}\,d\omega = [X_d(\omega) = \frac{1}{\Delta t}X(\omega)] \\
&= \frac{\Delta t}{2\pi}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} X_d(\omega)e^{j\omega t}\,d\omega = \frac{\Delta t}{2\pi}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} \left(\sum_{n=-\infty}^{\infty} x_d(n)e^{-j\omega\Delta tn}\right) e^{j\omega t}\,d\omega \\
&= \frac{\Delta t}{2\pi}\sum_{n=-\infty}^{\infty} x_d(n)\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} e^{j\omega(t-\Delta tn)}\,d\omega \\
&= \frac{\Delta t}{2\pi}\sum_{n=-\infty}^{\infty} x_d(n)\left(\frac{e^{j\frac{\pi}{\Delta t}(t-\Delta tn)} - e^{-j\frac{\pi}{\Delta t}(t-\Delta tn)}}{j(t-\Delta tn)}\right) \\
&= \frac{\Delta t}{\pi}\sum_{n=-\infty}^{\infty} x_d(n)\left(\frac{\sin(\frac{\pi}{\Delta t}(t-\Delta tn))}{(t-\Delta tn)}\right) = \sum_{n=-\infty}^{\infty} x_d(n)\left(\frac{\sin(\frac{\pi}{\Delta t}(t-\Delta tn))}{\frac{\pi}{\Delta t}(t-\Delta tn)}\right)
\end{aligned} \tag{3.1}$$

It is easy to verify that for all t and n such that $t = \Delta tn$ we have $x(t) = x_d(n)$. The continuous reconstructed signal $x(t)$, as expected, interpolates $x_d(n)$ at the sampling instances. This result is well known and is expressed in the following theorem.

Theorem 1. ***Nyquist sampling:*** *Assume $x_d(n) = x(\Delta tn)$. If $\forall|\omega| \geq \frac{\omega_s}{2}$, $X(\omega) \equiv 0$ then $X_d(\omega) = \frac{1}{\Delta t}X(\omega)$ and $x(t)$ can be reconstructed from $x_d(n)$ according to*

$$x(t) = \sum_{n=-\infty}^{\infty} x_d(n)\left(\frac{\sin(\frac{\omega_s}{2}(t-\Delta tn))}{\frac{\omega_s}{2}(t-\Delta tn)}\right). \tag{3.2}$$

The ideal reconstruction of the continuous signal $x(t)$ can be explicitly expressed in terms of the samples of the discrete time signal.

We can also describe the reconstruction as the result of a convolution/filtering of the mathematical continuous time representation of the sampled signal. To see this we start from the identity

$$x_d(n) = \int_{-\infty}^{\infty} x_d(n)\delta(\tau - \Delta t n)\, d\tau \tag{3.3}$$

and insert this into (3.2) to obtain

$$x(t) = \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} x_d(n)\delta(\tau - \Delta t n)\, d\tau \left(\frac{\sin(\frac{\omega_s}{2}(t - \Delta t n))}{\frac{\omega_s}{2}(t - \Delta t n)} \right) \tag{3.4}$$

By shifting the order of the sum and integration and noting that the integrand is zero whenever $\tau \neq \Delta t n$ we obtain

$$\begin{aligned} x(t) &= \int_{-\infty}^{\infty} \underbrace{\sum_{n=-\infty}^{\infty} x_d(n)\delta(\tau - \Delta t n)}_{\triangleq x_c(\tau)} \underbrace{\left(\frac{\sin(\frac{\omega_s}{2}(t - \tau))}{\frac{\omega_s}{2}(t - \tau)} \right)}_{\triangleq h_{\text{Ideal}}(t-\tau)} d\tau \\ &= \int_{-\infty}^{\infty} h_{\text{Ideal}}(\tau) x_c(t - \tau)\, d\tau \end{aligned} \tag{3.5}$$

which illustrates that the ideal reconstruction is a filtering operation where the discrete time signal is represented by the infinite sum of scaled delta pulses.

In the derivation found in (3.1) we implicitly showed that the frequency function of $h_{\text{Ideal}}(\tau)$ is

$$H_{\text{Ideal}}(\omega) = \begin{cases} \Delta t & |\omega| < \frac{\omega_s}{2} \\ 0 & \text{otherwise} \end{cases}$$

which of course is expected.

Note that the ideal reconstruction is non-causal. To reconstruct the signal $x(t)$ at time t, knowledge of all discrete time samples, i.e. both past as well as future samples of the signal is required. The ideal reconstruction hence has limited applicability in practice and brings us into the next topic of a more realistic way of how to approximately reconstruct the signal in a causal way.

ZOH Reconstruction

Ideal reconstruction, as illustrated above, is not practically possible as it requires non-causal filtering. Zero-order hold (ZOH) reconstruction is however easy to implement in electronic circuits and we will focus the analysis to this case. A ZOH reconstruction circuit provides a continuous time output signal $y(t)$ based on a discrete time signal $y_d(n)$ according to

$$y(t) \triangleq y_d(n), \quad n\Delta t \leq t < (n+1)\Delta t. \tag{3.6}$$

Hence, the continuous time signal $y(t)$ will be piecewise constant with levels given by the values of the discrete time signal. Practically, such signal a is generated by a digital to analog converter (DAC) which is periodically updated with the value $y_d(n)$. The ZOH operation can mathematically be modeled as filtering the continuous time representation of the discrete time signal

$$y_c(t) = \sum_{n=-\infty}^{\infty} y_d(n)\delta(t - n\Delta t) \tag{3.7}$$

through a linear filter with impulse response

$$h_{\text{ZOH}}(t) = \begin{cases} 1 & 0 \leq t < \Delta t \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

since

$$\int_{-\infty}^{\infty} h_{\text{ZOH}}(\tau) y_c(t-\tau)\, d\tau = \int_0^{\Delta t} \sum_{n=-\infty}^{\infty} y_d(n)\delta(t - \tau - n\Delta t)\, d\tau = y(t). \tag{3.9}$$

The impulse response of the ZOH filter has the frequency function

$$\begin{aligned} H_{\text{ZOH}}(\omega) = FT[h_{\text{ZOH}}(t)] &= \Delta t e^{-j\frac{\omega\Delta t}{2}} \frac{\sin(\frac{\omega\Delta t}{2})}{\frac{\omega\Delta t}{2}} \\ &= \Delta t e^{-j\pi\frac{\omega}{\omega_s}} \frac{\sin(\pi\frac{\omega}{\omega_s})}{\pi\frac{\omega}{\omega_s}}. \end{aligned} \tag{3.10}$$

We notice that since $\lim_{x\to 0} \sin(x)/x = 1$ (assuming argument to sin is radians) we obtain at zero frequency

$$H_{\text{ZOH}}(0) = \Delta t \tag{3.11}$$

and the frequency function is zero for all $\omega = k\omega_s$, $k = \pm 1, \pm 2, \ldots$. We can compare this frequency function with the ideal one given by

$$FT[h_{\text{Ideal}}(t)] = H_{\text{Ideal}}(\omega) = \begin{cases} \Delta t & |\omega| < \frac{\omega_s}{2} \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

where $h_{\text{Ideal}}(t)$ is given by (3.5). In Figure 3.1 the magnitude of the ZOH and ideal frequency functions are shown. The Fourier transform of the continuous time output is thus given by

$$Y(\omega) = H_{\text{ZOH}}(\omega) Y_d(\omega). \tag{3.13}$$

The full system

Now we are ready to present a Fourier domain description of the entire processing chain; sampling, digital processing and reconstruction. As each step can be represented with a frequency function in the Fourier domain, the total effect is the product of the individual frequency functions. Hence the FT of the output, after the reconstruction is described by

$$Y(\omega) = H_{\text{ZOH}}(\omega) P(\omega) \underbrace{\left[\frac{1}{\Delta t} \sum_{k=-\infty}^{\infty} X(\omega + \omega_s k)\right]}_{X_c(\omega) = X_d(\omega)} \tag{3.14}$$

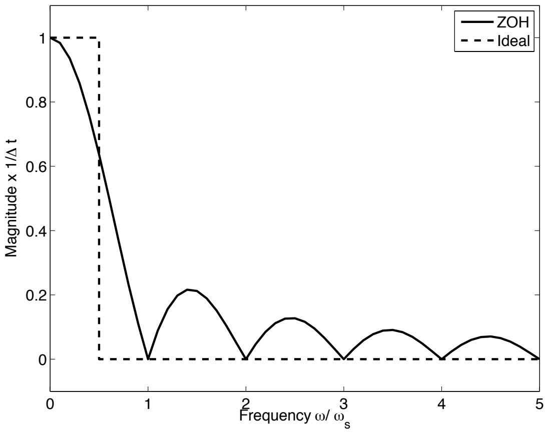


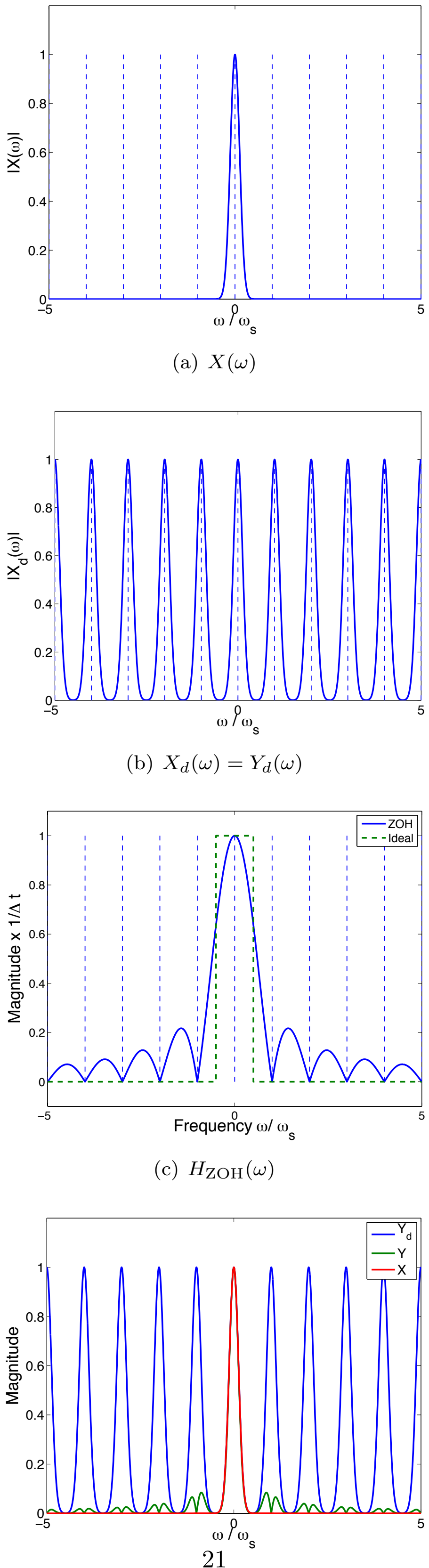
Figure 3.1: Magnitude plot of the frequency function for ZOH reconstruction and ideal reconstruction.

where $P(\omega)$ represent a discrete time linear filtering operation.

Both $P(\omega)$ as well as $X_d(\omega)$ are periodic with a period of ω_s and $H_{\text{ZOH}}(\omega)$ is also non-zero for $|\omega| > \omega_s/2$. Hence $Y(\omega)$ will also have energy beyond the Nyquist frequency when ZOH reconstruction is employed. Those parts can be regarded as distortions originating from the non-ideal reconstruction. A so called *reconstruction filter* of low-pass type can be used after the ZOH circuit to mitigate this effect. If instead ideal reconstruction is used (in theory since it cannot practically be realized) the reconstructed signal will be bandlimited to the frequency region $|\omega| < \omega_s/2$ due to the ideal low pass shape of $H_{\text{Ideal}}(\omega)$.

Aliasing distortion occur at the sampling step if not $X(\omega) \equiv 0$ for $|\omega| > \omega_s/2$. The effect is reduced by attaching a low-pass filter prior to the sampling circuit. Such a filter is called an *anti-aliasing filter (AAF)*. We will later return to the sampling and reconstruction cases when we discuss oversampled system architectures.

Example: Consider a continuous time signal $x(t)$ with a Fourier transform given by $X(\omega) = e^{-\gamma(\frac{\omega}{\omega_s})^2}$. The signal is first sampled with sampling frequency ω_s to yield $x(n)$. The signal is not processed any further so we set $y(n) \triangleq x(n)$. A continuous time signal $y(t)$ is reconstructed from the samples $y(n)$ using a ZOH circuit. In Figure 3.2 and Figure 3.3 the magnitude of the Fourier transforms for the different signals involved are shown for $\gamma = 32$ and $\gamma = 4$ respectively. The vertical dashed lines indicate integer multiples of the sampling frequency. When $\gamma = 32$ the original signal basically satisfy the Nyquist criterion and we notice that most of the distortions (the difference between the red and the green signals) is due to the ZOH reconstruction. When $\gamma = 4$ the original signal has significant energy beyond half the sampling frequency and the sampled signal is distorted by aliasing. This also leads to larger levels of high frequency distortions in the reconstructed signal.



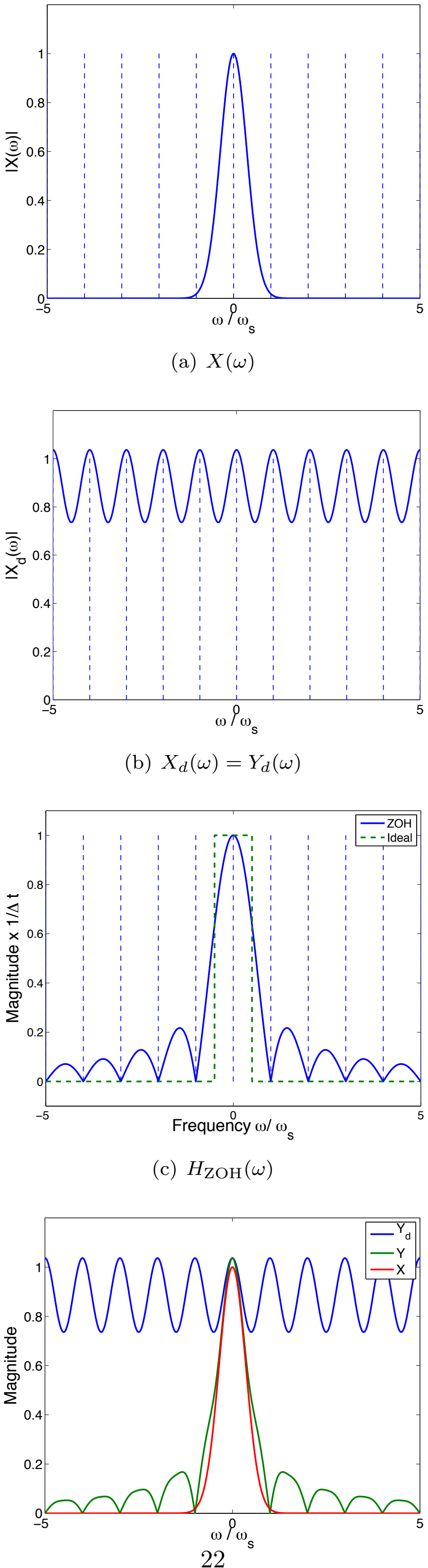
(a) $X(\omega)$

(b) $X_d(\omega) = Y_d(\omega)$

(c) $H_{\mathrm{ZOH}}(\omega)$

(d) $Y(\omega)$, $Y_d(\omega)\,X(\omega)$

Figure 3.2: Magnitude of Fourier transforms for signals in the reconstruction example when $\gamma = 32$.



(a) $X(\omega)$

(b) $X_d(\omega) = Y_d(\omega)$

(c) $H_{\mathrm{ZOH}}(\omega)$

(d) $Y(\omega)$, $Y_d(\omega)\, X(\omega)$

Figure 3.3: Magnitude of Fourier transforms for signals in the reconstruction example when $\gamma = 4$.

Example 6 In this example we consider a case including sampling, digital filtering and reconstruction using the ZOH method. The original continuous time signal is given by

$$x(n) = A\cos(2\pi f_0 t) = \frac{A}{2}(e^{j2\pi f_0 t} + e^{-j2\pi f_0 t})$$

where $f_0 = 22$ kHz. The signal is sampled with a sampling frequency of $f_s = 20$ kHz. The sampled signal is filtered through a discrete time filter with impulse response given by

$$h(n) = \begin{cases} 0.8^n, & n \geq 0 \\ 0 & n < 0 \end{cases}$$

and the resulting output is then fed to the ZOH reconstruction circuit. The problem to solve is to derive an expression for the reconstructed signal $y(n)$ in the time domain.

Solution: Since the sinusoidal signal is equivalently represented by the summation of two complex exponentials we directly see that

$$X(\omega) = \text{FT}[x(t)] = \pi A(\delta(\omega - \omega_0) + \delta(\omega + \omega_0))$$

where $\omega_0 = 2\pi f_0$. After the sampling the DTFT of the resulting discrete time signal $x_d(n)$ is given by

$$\begin{aligned} X_d(\omega) &= \frac{1}{\Delta t}\sum_{k=-\infty}^{\infty} X(\omega + \omega_s k) \\ &= \frac{\pi A}{\Delta t}(\tilde{\delta}(\omega - \omega_1) + \tilde{\delta}(\omega + \omega_1)) \end{aligned}$$

where $\omega_1 = 2\pi(f_0 - f_s) = 2\pi(22 - 20) \times 10^3 = 2\pi 2 \times 10^3$ rad/s . Hence, due to aliasing, the 22 kHz signal now appear as a discrete time cosine signal with frequency 2 kHz. Using the inverse DTFT we obtain

$$\begin{aligned} x_d(n) &= \frac{\Delta t}{2\pi}\int_0^{\omega_s} \frac{\pi A}{\Delta t}(\tilde{\delta}(\omega - \omega_1) + \tilde{\delta}(\omega + \omega_1))e^{j\omega\Delta t n}\, d\omega \\ &= A\cos(\omega_1 \Delta t n) \end{aligned}$$

The discrete time signal is then filtered through the filter given by the impulse response $h(n)$. It is easiest to evaluate the result of the filtering in the frequency domain. The DTFT of the filter is

$$H(\omega) = \sum_{n=0}^{\infty} h(n)e^{-j\omega\Delta t n} = \sum_{n=0}^{\infty} 0.8^n e^{-j\omega\Delta t n} = \frac{1}{1 - 0.8e^{-j\omega\Delta t}}$$

where the last equality follows from the property of infinite geometric series. Note also that $H(-\omega) = \overline{H(\omega)}$, i.e. for this filter the frequency function for negative frequencies is the complex conjugate of the frequency function for positive frequencies. Now we have

$$Y_d(\omega) = H(\omega)X_d(\omega) = H(\omega)\frac{\pi A}{\Delta t}(\tilde{\delta}(\omega - \omega_1) + \tilde{\delta}(\omega + \omega_1))$$

Using the IDFT on $Y_d(\omega)$ we obtain

$$y_d(n) = \frac{A}{2}\left(H(\omega_1)e^{j\omega_1\Delta tn} + H(-\omega_1)e^{-j\omega_1\Delta tn}\right)$$

Denote by $A_H \triangleq |H(\omega_1)|$ and $\phi_H \triangleq \angle H(\omega_1)$. We can rewrite y_d as

$$y_d(n) = \frac{AA_H}{2}\left(e^{j(\omega_1\Delta tn+\phi_H)} + e^{-j(\omega_1\Delta tn+\phi_H)}\right) = AA_H\cos(\omega_1\Delta tn+\phi_H)$$

and

$$Y_d(\omega) = \frac{\pi AA_H}{\Delta t}(e^{j\phi_H}\tilde{\delta}(\omega-\omega_1) + e^{-j\phi_H}\tilde{\delta}(\omega+\omega_1)) \tag{3.15}$$

A carful look at (3.15) reveals that we have delta functions located at $\omega_1 + k\omega_s$ scaled with $e^{j\phi_H}$ and delta functions located at $-\omega_1 + k\omega_s$ scaled with $e^{-j\phi_H}$ for all integer values k. Alternatively, for all positive frequencies we have delta functions at $(\omega_1 + k'\omega_s)$ scaled with $e^{j\phi_h}$ and at $(\omega_s - \omega_1 + k'\omega_s)$ scaled with $e^{-j\phi}$ for non-negative integer values of k', i.e. $k = 0, 1, 2, \ldots$. At the corresponding negative frequencies the scaling of the delta functions are conjugated. Thus we can reformulate (3.15) into

$$\begin{aligned}Y_d(\omega) = \frac{\pi AA_H}{\Delta t}\sum_{k=0}^{\infty}\left(e^{j\phi_H}\delta(\omega-\omega_1-k\omega_s) + e^{-j\phi_H}\delta(\omega+\omega_1+k\omega_s)\right) + \\ + \frac{\pi AA_H}{\Delta t}\sum_{k=0}^{\infty}\left(e^{-j\phi_H}\delta(\omega-\omega_2-k\omega_s) + e^{j\phi_H}\delta(\omega+\omega_2+k\omega_s)\right)\end{aligned} \tag{3.16}$$

where $\omega_2 \triangleq \omega_s - \omega_1 = 2\pi 18 \times 10^3$ rad/s.

The ZOH reconstruction can be interpreted as continuous time filtering the discrete time signal through the ZOH filter. In the Fourier domain we obtain

$$Y(\omega) = H_{ZOH}(\omega)Y_d(\omega) = \Delta te^{-j\pi\frac{\omega}{\omega_s}}\frac{\sin(\pi\frac{\omega}{\omega_s})}{\pi\frac{\omega}{\omega_s}}Y_d(\omega)$$

Applying the IFT we obtain the time-domain expression. For each k in (3.16) we obtain two cosine terms. The first term yields

$$\begin{aligned}&\frac{1}{2\pi}\int_{-\infty}^{\infty}\Delta te^{-j\pi\frac{\omega}{\omega_s}}\frac{\sin(\pi\frac{\omega}{\omega_s})}{\pi\frac{\omega}{\omega_s}}\frac{\pi AA_H}{\Delta t}\left(e^{j\phi_H}\delta(\omega-\omega_1-k\omega_s) + e^{-j\phi_H}\delta(\omega+\omega_1+k\omega_s)\right)e^{j\omega t}\,d\omega \\ &= \frac{\sin(\pi\frac{\omega_1+k\omega_s}{\omega_s})}{\pi\frac{\omega_1+k\omega_s}{\omega_s}}AA_H\cos((\omega_1+k\omega_s)t + \phi_H - \pi\omega_1/\omega_s - k\pi)\end{aligned} \tag{3.17}$$

similarly the second term results in

$$\begin{aligned}&\frac{1}{2\pi}\int_{-\infty}^{\infty}\Delta te^{-j\pi\frac{\omega}{\omega_s}}\frac{\sin(\pi\frac{\omega}{\omega_s})}{\pi\frac{\omega}{\omega_s}}\frac{\pi AA_H}{\Delta t}\left(e^{-j\phi_H}\delta(\omega-\omega_2-k\omega_s) + e^{j\phi_H}\delta(\omega+\omega_2+k\omega_s)\right)e^{j\omega t}\,d\omega \\ &= \frac{\sin(\pi\frac{\omega_2+k\omega_s}{\omega_s})}{\pi\frac{\omega_2+k\omega_s}{\omega_s}}AA_H\cos((\omega_2+k\omega_s)t - \phi_H - \pi\omega_2/\omega_s - k\pi)\end{aligned} \tag{3.18}$$

Putting it all together yields the time domain expression for the output of the ZOH circuit as

$$y(t) = AA_H \sum_{k=0}^{\infty} \left(\frac{\sin(\pi \frac{\omega_1 + k\omega_s}{\omega_s})}{\pi \frac{\omega_1 + k\omega_s}{\omega_s}} \cos((\omega_1 + k\omega_s)t + \phi_H - \pi\omega_1/\omega_s - k\pi) + \frac{\sin(\pi \frac{\omega_2 + k\omega_s}{\omega_s})}{\pi \frac{\omega_2 + k\omega_s}{\omega_s}} \cos((\omega_2 + k\omega_s)t - \phi_H - \pi\omega_2/\omega_s - k\pi) \right) \tag{3.19}$$

In summary the ZOH leads to a signal which contain an infinite number of cosine terms with increasing frequencies. The amplitude for these components are decreasing with increasing frequencies as the magnitude of the side lobes of the sinc function $\sin(x)/x$ decreases with increasing values of x. □

Chapter 4

Frequency analysis, Periodogram and DFT

Frequency analysis refers to techniques to obtain estimates of the DTFT for a measured discrete time signal. Since the definition of the DTFT involves an infinite number of samples of the signal $x(n)$ it is not in practice possible to determine the exact DTFT. What can be done practically is to analyze a *finite* number of samples and then interpret the results in comparison with the DTFT.

To carry the analysis over to a practical setting we will work with a *window* of the data, i.e. a finite number of samples at indices $n = 0, 1, \ldots, N-1$. Mathematically this can be described as

$$\hat{x}(n) \triangleq r_N(n)x(n), \quad \forall n \tag{4.1}$$

and the window function is defined as

$$r_N(n) = \begin{cases} 1 & n = 0, \ldots, N-1 \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

The question that arises is how is $\hat{X}(\omega)$ related to $X(\omega)$? Before establishing the relation we recall the dual result of convolution involving the point-wise multiplication of two discrete time signals. We have the result:

$$y(n) = x(n)w(n) \quad \Leftrightarrow \quad Y(\omega) = \frac{1}{\omega_s}\int_0^{\omega_s} X(\lambda)W(\omega-\lambda)\,d\lambda \tag{4.3}$$

A multiplication in the time domain is equal to a frequency domain convolution. Clearly (using the rules of a finite geometric series) the DTFT of the window function defined in (4.2) is

$$\begin{aligned} R_N(\omega) &= \sum_{n=-\infty}^{\infty} r_N(n)e^{-j\omega\Delta t n} = \sum_{n=0}^{N-1} e^{-j\omega\Delta t n} = \\ &= \frac{1-e^{-j\omega\Delta t N}}{1-e^{-j\omega\Delta t}} = e^{-j\frac{N-1}{2}\omega\Delta t}\frac{\sin(\frac{N\omega\Delta t}{2})}{\sin(\frac{\omega\Delta t}{2})} \end{aligned} \tag{4.4}$$

We note that $R_N(0) = N$ and $R_N(\omega) = 0$ for $\omega = \pm\omega_s/N, \pm 2\omega_s/N, \pm 3\omega_s/N, \ldots$. Since it is a DTFT, it is also periodic with a period ω_s. In Figure 4.1 the DTFT of the rectangular window function, $R_N(\omega)$, is plotted for $N = 10, 20, 40$. The magnitude peak around 0 is called the main lobe and has a width inversely proportional to N. The smaller peaks are called side lobes and decreases in size as frequency increases up to the Nyquist frequency. The DTFT of a windowed signal is then

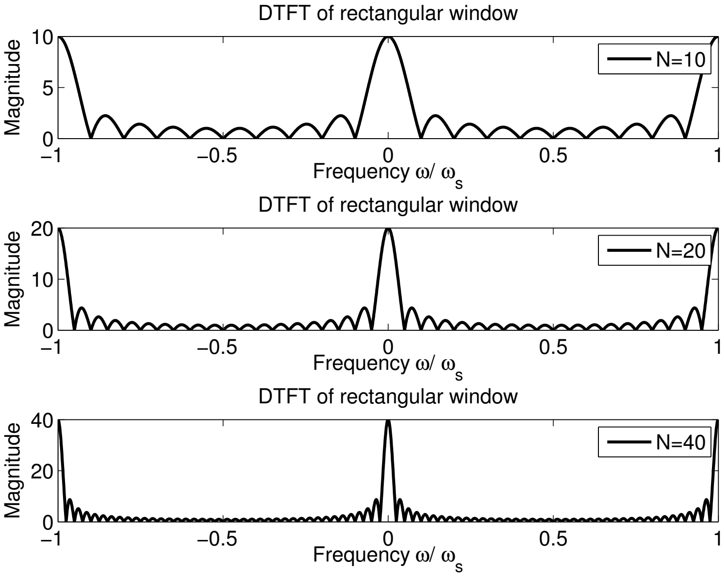


Figure 4.1: The discrete time Fourier transform of the rectangular window function for $N = 10, 20, 40$.

$$\hat{x}(n) = x(n)r_N(n) \quad \Leftrightarrow \quad \hat{X}(\omega) = \frac{1}{\omega_s}\int_0^{\omega_s} X(\lambda)R_N(\omega-\lambda)\,d\lambda. \tag{4.5}$$

If N is very large ($N \to \infty$) we can argue that $R_N(\omega) \approx 2\pi\tilde{\delta}(\omega)$. In such a case we obtain

$$\hat{X}(\omega) = \frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} X(\lambda)2\pi\delta(\omega-\lambda)\,d\lambda = X(\omega) \tag{4.6}$$

which of course is the expected result.

Example Consider a discrete time signal $x(n) \triangleq e^{j\omega_0 n}$ and $\Delta t = 1$ ($\Rightarrow \omega_s = 2\pi$) with DTFT $X(\omega) = 2\pi\tilde{\delta}(\omega-\omega_0)$. Assume N samples are available. The DTFT of this rectangularly windowed signal is the frequency convolution of $X(\omega)$ and $R_N(\omega)$

$$\begin{aligned}\hat{X}(\omega) &= \frac{1}{2\pi}\int_0^{2\pi} X(\omega-\lambda)R_N(\lambda)\,d\lambda = \\ &= \frac{1}{2\pi}\int_0^{2\pi} 2\pi\tilde{\delta}(\omega-\omega_0-\lambda)R_N(\lambda)\,d\lambda = R_N(\omega-\omega_0)\end{aligned} \tag{4.7}$$

Figure 4.2 illustrate the result when $\omega_0 = 0.2\omega_s$. The magnitude of the DTFT is

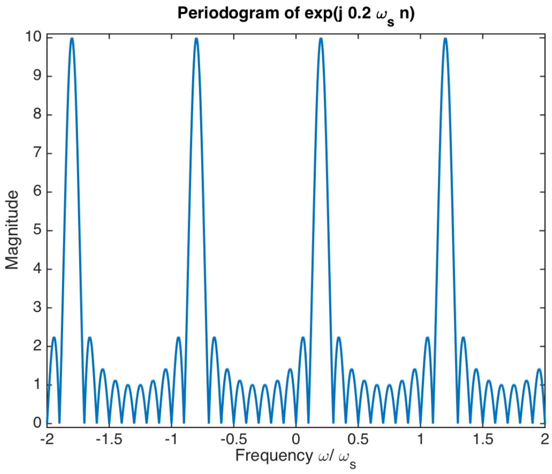


Figure 4.2: Periodogram based on $N = 10$ samples of the signal $x(n) = e^{j0.2\omega_s n}$

called the *periodogram* of the signal. Clearly the periodogram is centered around ω_0. However the periodogram is also non-zero for other large frequency intervals. This is due to the *leakage effect* where the energy leaks out to neighboring frequencies. A simple way to estimate the frequency of a complex sinusoidal signal is thus to find the location of the largest peak of the periodogram. Recall that the other peaks outside the interval $[0, 1]$ of relative frequencies are due to the periodic character of the DTFT. If the signal contain more than one complex exponential the resulting windowed DTFT will be equal to the summation of the windowed DTFT of each component. Hence, if frequencies are spaced far apart, in comparison with the size of N, we can expect to see two peaks located close to the correct frequencies. However, in general, a bias in the peak location will exist due to the leakage effects.

Discrete Fourier Transform (DFT)

If we "sample" the rectangularly windowed DTFT at equidistantly spaced frequencies $\omega_k \Delta t = \frac{2\pi k}{N}$ we obtain the transform known as the Discrete Fourier Transform (DFT)[1]:

$$\begin{aligned} \text{DFT:} \quad & X(k) \triangleq \hat{X}(\frac{2\pi k}{N\Delta t}) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi k n}{N}} \\ \text{IDFT:} \quad & x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi k n}{N}} \end{aligned} \tag{4.8}$$

Note that $X(k)$ is N-periodic, i.e. $X(k+N) = X(k)$. Also if we use the IDFT formulation we notice that $x(n+N) = x(n)$, i.e. if we use the IDFT relation then $x(n)$ is periodically repeated outside the interval $[0, N-1]$. The DFT and IDFT are calculated in MATLAB using the commands `X=fft(x);` and `x=ifft(X);`. The acronym FFT stands for the Fast Fourier Transform which is an efficient computational approach to calculate the DFT and IDFT. Later we will return to this technique.

The IDFT formula in (4.8) is the complex Fourier series expansion of a general N-periodic discrete time signal where $X(k)/N$ can be regarded as the weights of the Fourier series expansion.

We will now discuss three cases where the character of $x(n)$ outside the interval $n = 0, N-1$ defines the different cases. For simplicity we assume $\Delta t = 1$.

Case1: $x(n)$ is N-periodic This case we have discussed above when introducing the DFT. Here the signal is given by

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi k n}{N}}, \quad n = 0, \pm 1, \pm 2, \ldots \tag{4.9}$$

where $X(k)$ can be derived from the DFT based on one period of the discrete time signal $x(n)$. In this case the DTFT will be a sum of Dirac delta functions, scaled by $X(k)$

$$X(\omega) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) 2\pi \delta(\omega - \frac{2\pi k}{N}) \quad \omega \in [0, 2\pi], \tag{4.10}$$

and periodically repeated outside which gives the alternative representation (valid for all ω)

$$X(\omega) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) 2\pi \tilde{\delta}(\omega - \frac{2\pi k}{N}). \tag{4.11}$$

Case 2: $x(n)$ is zero outside If $x(n)$ is zero outside the observation interval we have full knowledge of the entire signal and hence

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n) e^{-j\omega n} = \sum_{n=0}^{N-1} x(n) e^{-j\omega n} \tag{4.12}$$

[1] Here the integer index k indicates that symbol $X(k)$ denotes the DFT.

In this case a multiplication with the rectangular window will change nothing. Calculating the DFT consequently means to calculate samples of the DTFT $X(\omega)$ at the specific frequencies $\omega_k = 2\pi k/N$ for $k = 0, \ldots, N-1$.

Case 3: $x(n)$ arbitrary outside This is the only case of the three where only partial knowledge of $x(n)$ is available. Hence we can only expect partial knowledge of the DTFT of the full signal. The analysis derived above then tell us that

$$\hat{x}(n) = x(n)r_N(n) \quad \Leftrightarrow \quad \hat{X}(\omega) = \frac{1}{2\pi}\int_0^{2\pi} X(\lambda)R_N(\omega - \lambda)\, d\lambda. \tag{4.13}$$

And again calculating the DFT will yield samples of $\hat{X}(\omega)$. That is

$$\hat{X}(\omega)\Big|_{\omega=2\pi k/N} = X(k). \tag{4.14}$$

Zero padding

To calculate the DTFT for arbitrary frequencies is straight-forward. It is just to implement the definition

$$\hat{X}(\omega) = \sum_{n=0}^{N-1} x(n)e^{-j\omega n}. \tag{4.15}$$

However, using an equidistantly spaced frequency grid the calculations can be made more efficiently by employing the FFT algorithm.

Assume the (windowed) signal $\hat{x}(n)$ is of length N and we wish to calculate the DTFT at a denser grid of frequencies, $\omega_k = 2\pi k/N_Z$, $k = 0, \ldots, N_Z - 1$ where $N_Z > N$. We obtain exactly this result if we define the N_Z long zero padded signal

$$\hat{x}_z(n) = \begin{cases} \hat{x}(n), & n = 0, \ldots, N-1 \\ 0 & n = N, \ldots N_Z - 1 \end{cases} \tag{4.16}$$

and calculate the DFT of this signal

$$\begin{aligned} \hat{X}_z(k) &= \sum_{n=0}^{N_Z-1} \hat{x}_z(n)e^{-j\frac{2\pi kn}{N_Z}} = \sum_{n=0}^{N-1} \hat{x}(n)e^{-j\frac{2\pi kn}{N_Z}}, \quad k = 0, \ldots, N_Z - 1 \\ \Rightarrow \hat{X}(\omega)\Big|_{\omega=\frac{2\pi k}{N_Z}} &= \hat{X}_z(k) \end{aligned} \tag{4.17}$$

In summary adding zeros to the end of the sequence and calculate the DFT will yield more sample of the *same* DTFT of the windowed signal. Figure 4.3 illustrates the effect for the signal $x(n) = 1$ for $n = 0, \ldots, 9$ and zero otherwise. In the top graph the magnitude of the DTFT of the signal is illustrated together with the 10 point DFT. In the middle and bottom graphs the DTFT and the DFT of the signal, when padded with zeros to a total length 30 and length 100 respectively, is shown. Notice that the DTFT is the *same* for all three cases.

The DFT with zero padding to a total length of `Nz` is calculated in Matlab using the command `X=fft(x,Nz);`.

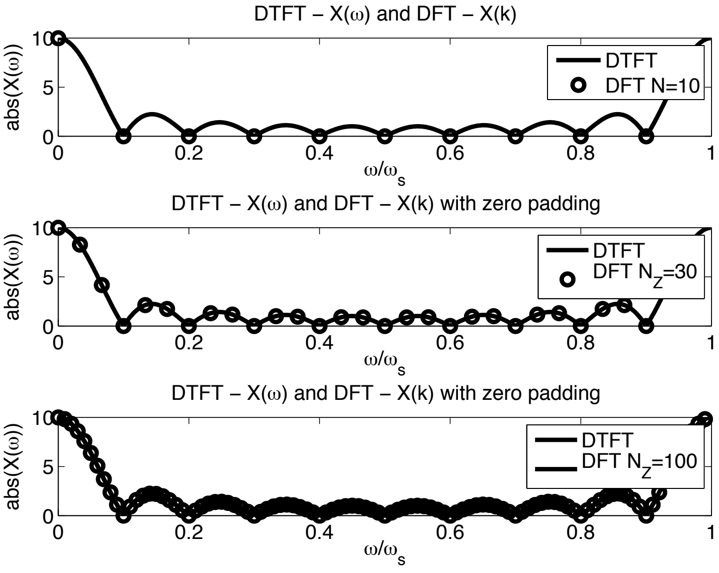


Figure 4.3: Illustration of zero padding.

Filtering using DFT - linear convolution

Given a finite length input signal $x(n)$ of length N and a finite length impulse response $h(n)$ of length M the linear convolution (filtering) is given by

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k), \quad n = 0, 1, \ldots, N_y - 1. \tag{4.18}$$

where $N_y = N+M-1$. The output is a signal of length N_y and is zero otherwise. We now seek a method to calculate the output based on the DFT. From DTFT theory we know that (4.18) is equivalent with $Y(\omega) = H(\omega)X(\omega)$ for all values of ω. Recall the Case 2, i.e. (4.12), from the DFT section. This case imply that $Y(\omega_k) = Y(k)$ where $Y(k)$ is the DFT of length N_y. We also know that given $Y(k)$ the inverse DFT will yield $y(n)$. Hence we need to calculate the product of $H(\omega)$ and $X(\omega)$ at the frequencies $\omega_k = \frac{2\pi k}{N_y}$ for $k = 0, \ldots, N_y - 1$. Since both N and M are less than N_y and both signals are zero outside their respective interval we can zero pad both signals to obtain the length N_y and then calculate the DFT of $x(n)$ and $h(n)$ to obtain the desired product $X(k)H(k)$ for $k = 0, 1, \ldots, N_y - 1$. In MATLAB this would look like

```
>> N = length(x);
>> M = length(h);
>> Ny = N+M-1;
>> X = fft(x,Ny);
>> H = fft(h,Ny);
>> Y = H.*X;
>> y = ifft(Y);
```

To make this execute fast N_y should be selected to be the nearest power of 2 that is larger or equal to $N + M - 1$.

Filtering using DFT - circular convolution

In this section we examine how the DFT can be used when the input signal to the filter can be regarded as a periodic signal with period N. From signals and systems theory we can thus express this signal using the Fourier series approach and we have:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi kn}{N}}, \quad n = 0, \pm 1, \pm 2, \ldots \tag{4.19}$$

where $X(k)$ is the DFT of one period of the input $x(n)$, i.e. for $n = 0, \ldots, N-1$. Filtering this signal through the filter with frequency function $H(\omega)$ consequently yields the output

$$y(n) = \frac{1}{N} \sum_{k=0}^{N-1} H(\omega_k) X(k) e^{j\frac{2\pi kn}{N}}, \quad n = 0, \pm 1, \pm 2, \ldots \tag{4.20}$$

where $\omega_k = 2\pi k/N$. If the filter has an impulse response which is shorter or equal to N, $H(\omega_k)$ is obtained by calculating the DFT of the impulse response zero-padded to a length N.

In MATLAB this can be formulated as (for the case when $M \leq N$):

```
>> N = length(x); % X assumed N-periodic
>> M = length(h);
>> if M>N, error('Filter length to long!'); end;
>> X = fft(x,N);
>> H = fft(h,N);
>> Y = X.*H; % Circular convolution
>> y = ifft(Y);
```

If the impulse response is longer than N then the DTFT of $h(t)$ can be evaluated for the frequencies $\omega_k = 2\pi k/N$. Samples at these frequencies can also be obtained by using DFT and a suitable length zero padding.

This technique is known as circular convolution. One reason for this become clear if we write this operation as a linear transformation. Since $x(n)$ is periodic we can write the output of the linear convolution as a matrix vector product

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N-1) \end{bmatrix} = \mathbf{H}_c \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} \tag{4.21}$$

where the matrix $\mathbf{H}_c$ has the structure

$$\mathbf{H}_c = \begin{bmatrix} h(0) & h(N-1) & h(N-2) & \cdots & h(1) \\ h(1) & h(0) & h(N-1) & \cdots & h(2) \\ h(2) & h(1) & h(0) & \cdots & h(3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ h(N-1) & h(N-2) & h(N-3) & \cdots & h(0) \end{bmatrix}. \tag{4.22}$$

This square matrix is called a circulant matrix since each row (and column) is a circularly shifted version of the previous one. The inverse of $\mathbf{H}_c$ is tightly connected to the DFT of $h(k)$. It is left as an exercise for the interested reader to explore the details.

Chapter 5

Equalization and FFT

Equalization Assume we have a finite signal $x(n)$, $n = 0, \ldots, N-1$ which we want to transmit to a receiver. The medium over which we send the signal, which we call the *channel*, changes the signal by the way of a linear convolution with an impulse response $h(n)$. Let us assume that $h(n)$ is of finite length M. The output of the convolution, i.e. the signal at the receiver end is hence

$$y(n) = \sum_{k=0}^{M-1} h(k)u(n-k) \tag{5.1}$$

clearly the received signal is of length $N+M-1$ and is hence zero for $n \geq N+M-1$ and $n < 0$. If the impulse response is known at the receiver we could (in the noise free case) recover the transmitted signal $x(n)$ by solving a system of linear equations defined by (5.1). Alternatively we could solve for x using the DFT representation of (5.1):

$$Y(k) = H(k)X(k), \quad k = 0, \ldots, N+M-2 \tag{5.2}$$

where the $Y(k), H(k)$ and $X(k)$ are the DFT of the signals $y(n), h(n)$ and $x(n)$ respectively, each zero padded to the length $N+M-1$. This is simpler as we can solve for $X(x)$ for each k simply by $X(k) = Y(k)/H(k)$ without having to solve a system of linear equations (which would involve a matrix inversion).

Equalization in OFDM In communication applications it is desired to recover the transmitted signal in the receiver in order to determine the message transmitted. This is known as *equalization.* The medium over which the signal is transmitted normally changes the signal. In communication applications the medium is often called the *channel*. Here we consider a medium which can be modeled as a linear dynamical system with a frequency function $H(\omega)$ and impulse response $h(n)$. If we ignore effects of noise or model errors the received signal can be modeled as

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k) \tag{5.3}$$

where $x(n)$ is the transmitted signal we wish to recover. If the transmitted signal is periodic with period N we have

$$Y(k) = H(k)X(k), \quad k = 0, \ldots, N-1 \tag{5.4}$$

where $Y(k)$ and $X(k)$ are the N-point DFT of $x(n)$ and $y(n)$ respectively. As a consequence, if we know $Y(k)$, we can recover the transmitted signal simply as

$$X(k) = Y(k)/H(k) \tag{5.5}$$

provided that $H(k)$ is non-zero for all k. This is in principle how the channel-equalization is performed in The Orthogonal Frequency-Division Multiplexing (OFDM) technique for information transmission. The advantage with the OFDM technique compared to the method above is that we only need a DTFT of length N which is much smaller than the required $N + M - 1$ in the method above.

Filter transients and quasi-periodic signals

When filtering a signal through a filter with a finte impulse response (FIR) of length M we can write the convolution equation as

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k). \tag{5.6}$$

Assume the input $x(n)$ is zero for $n < 0$ and periodic with a period P for all positive n. What character will the output $y(n)$ have? Clearly $x(n + mP) = x(n)$, for integers $m, n \geq 0$. Thus, for $n \geq M - 1$ and $m \geq 0$ the time index for x in equation (5.6) is then always positive and we obtain

$$y(n+mP) = \sum_{k=0}^{M-1} h(k)x(n+mP-k) = \sum_{k=0}^{M-1} h(k)x(n-k) = y(n), \quad n \geq M-1. \tag{5.7}$$

Hence, from time index $n = M-1$ and onwards the output $y(n)$ is also periodic. For time indices $n = 0$ to $M - 2$ the output is in a transient phase when the filter "fills up". Let us define

$$\tilde{x}(n) = x(n + n_0), \quad \tilde{y}(n) = y(n + n_0), \quad n = 0, \ldots, P - 1 \tag{5.8}$$

for any $n_0 \geq M - 1$. Since for these time indices the filtering acts in a periodic fashion we can regard it as a circular convolution, i.e.

$$\tilde{Y}(k) = H(\omega_k)\tilde{X}(k), \quad n = 0, \ldots, P - 1 \tag{5.9}$$

where $\omega_k = 2\pi k/P$, $\tilde{Y}(k)$ and $\tilde{X}(k)$ is the P-points DFT of $\tilde{y}$ and $\tilde{x}$ respectively.

The OFDM technique is based on the properties presented above. The part of the input signal $x(n)$ from $n = 0$ to $n \geq M - 2$ (and then followed by N samples) is known as the *cyclic-prefix* or *guard* of the OFDM block and need to be present to enable a recovery of the input as illustrated by (5.5).

Figure 5.1 illustrates the effect of filtering a quasi-periodic input (with a period time of 5), through a filter of length 4. After a transient of 4 samples (indices 0 to 3), the output is also periodic from sample 4 and onwards.

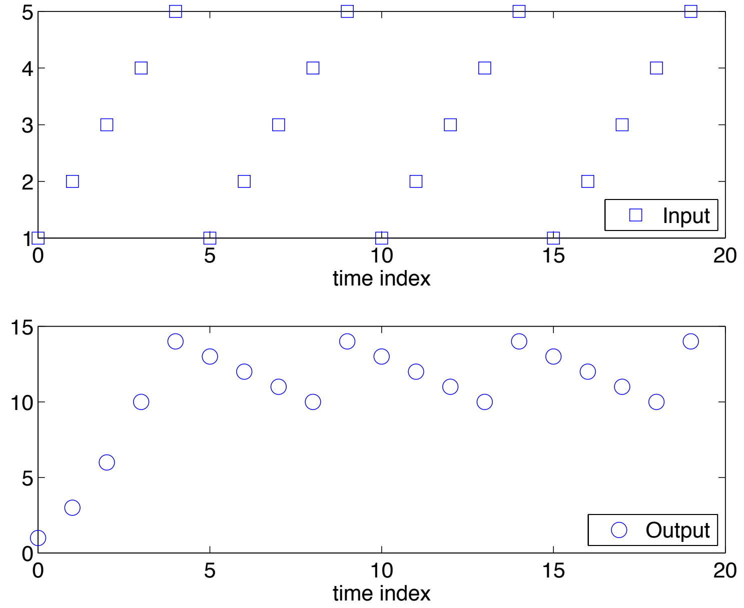


Figure 5.1: Filtering results using a quasi-periodic input signal and a filter length of $M = 4$.

Fast Fourier Transform (FFT)

Calculating the DFT in a computationally efficient way is instrumental in many signal processing applications. The Fast Fourier Transform (FFT) is a computational scheme for calculating the DFT which require less operations than a direct application of the definition. The algorithm is based on a recursive view of how to calculate the DFT and utilizing that the DFT is a periodic function. In digital circuitry a multiplication requires far more operations than addition. Hence, to discuss complexity, we will focus on the number of multiplications required.

Consider the DFT:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \quad k = 0, \ldots, N-1 \text{ and } W_N = e^{-j2\pi/N} \tag{5.10}$$

Assume $N = 2^p$ where $p = \log_2(N)$ is an integer. Hence N is an even number and we can divide the DFT into two separate sums where the first include all the even time indices and the second one involve all the odd time indices.

$$\begin{aligned} X(k) &= \sum_{n=0}^{N/2-1} x(2n) \underbrace{W_N^{k2n}}_{W_{N/2}^{kn}} + \sum_{n=0}^{N/2-1} x(2n+1) \underbrace{W_N^{k(2n+1)}}_{W_N^k W_{N/2}^{kn}} \\ &= \underbrace{\sum_{n=0}^{N/2-1} x(2n) W_{N/2}^{kn}}_{X_e(k)} + W_N^k \underbrace{\sum_{n=0}^{N/2-1} x(2n+1) W_{N/2}^{kn}}_{X_o(k)} \\ &= X_e(k) + W_N^k X_o(k) \end{aligned} \tag{5.11}$$

We note that $X_e(k)$ and $X_o(k)$ are DFT of length $N/2$ and to determine $X(k)$ for all values of k ($k = 0, \ldots, N-1$) we need to evaluate X_e and X_o for all these indices. However, since both of them are of length $N/2$ we can use the periodic property of the DFT. Hence X_e and X_o only need to be evaluated for $k = 0, \ldots, N/2-1$ and use the periodic property for $k = N/2, \ldots, N-1$. i.e.

$$\begin{aligned} X(k+N/2) &= X_e(k) + W_N^{k+N/2} X_o(k) \\ &= X_e(k) - W_N^k X_o(k), \quad k = 0, \ldots, N/2-1 \end{aligned} \tag{5.12}$$

Also note that $W_N^{k+N/2} = -W_N^k$ which implies that we, for each k, only need to derive the product $W_N^k X_o(k)$ once and then use it both in (5.12) as well as in (5.11). We can write the operations in (5.11) and (5.12) jointly as

$$\begin{aligned} X(k) &= X_e(k) + W_N^k X_o(k) \\ X(k+N/2) &= X_e(k) - W_N^k X_o(k), \quad k = 0, \ldots, N/2-1 \end{aligned} \tag{5.13}$$

The calculations above are know as a “butterfly operation” since a graphical representation has a butterfly geometry as illustrated in Figure 5.2.

In summary, to calculate $X(k)$ for $k = 0, \ldots, N-1$ we need $N/2$ multiplications and the two DFTs of length $N/2$. Recursively we can use the same

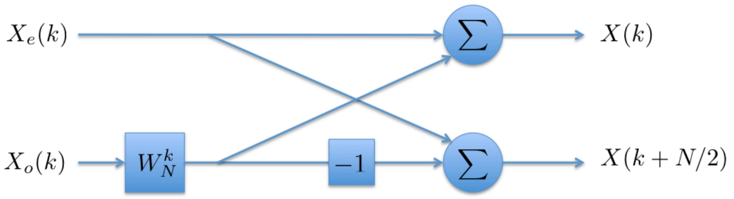


Figure 5.2: Graphical illustration of a butterfly operation.

Data length (N)	DFT complexity (N^2)	FFT complexity ($N/2\log_2 N$)
8	64	12
64	4096	192
128	16384	448
1024	1048576	5120

Table 5.1: Complexity of FFT in comparison with direct calculation of DFT.

argument on the two sub sequences and keep going until $N = 2$. For this case we obtain $W_2 = -1$ and the DFT of $x(0)$ and $x(1)$ are

$$\begin{aligned} X(0) &= x(0) + x(1) \\ X(1) &= x(0) - x(1) \end{aligned} \tag{5.14}$$

The total number of combination stages needed is $\log_2 N$ and in each of the combining stages we need a total of $N/2$ number of complex multiplications. The total complexity to calculate the DFT is thus $(N/2)\log_2 N$. The illustrated computational procedure is known as the Cooley-Tukey, radix 2, decimation in time (DIT) FFT algorithm. If we compare this complexity with a brute force application of the DFT equation we notice that for each k, N multiplications are needed and hence, to calculate the DFT for all N a total of N^2 multiplications are required. Since the core computational unit involves a two point DFT this algorithm is known as the Radix-2 FFT algorithm. Radix-4 and mixed radix algorithms are extensions which can provide even further reduction in computational effort needed. The efficiency of the FFT algorithm in comparison with direct calculation of DFT is illustrated in Table 5.1. For a data sequence length of 1024 samples the computational effort is 205 times less for the FFT algorithm. The final assembly of the result for a $N = 8$ FFT is illustrated in Figure 5.3.

Inverse Fast Fourier Transform (IFFT) The calculations involved to determine the inverse DFT very much resembles the calculation of the DFT. Comparing the expressions for DFT and IDFT in (4.8) we note that the differences are the scaling with the length of the sequence N and the complex exponential is conjugated. It is straight forward to derive a fast inverse DFT algorithm along the same lines as was done for the DFT algorithm.

Given that an implementation of the FFT algorithm exists we can actually make it perform the inverse DFT by feeding it with a reversed sequence and multiply each obtained time sample with $\frac{1}{N}e^{-j2\pi n/N}$. To illustrate this the following Matlab code shows how it is done:

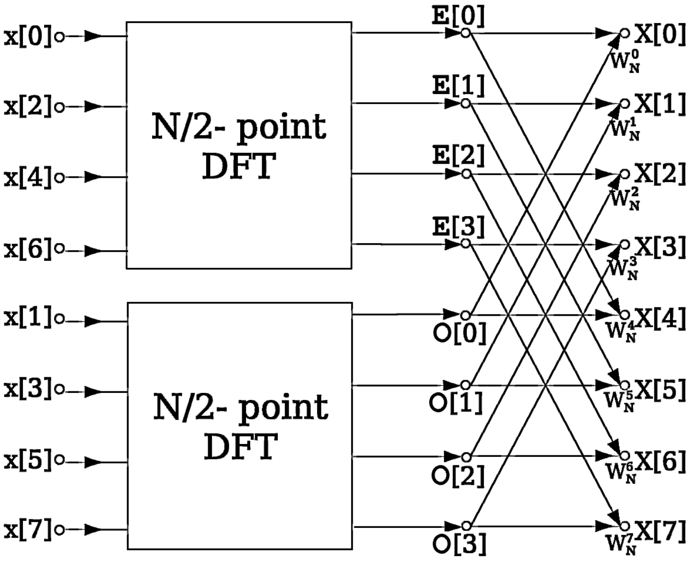


Figure 5.3: FFT calculation illustration. Data flow diagram for $N = 8$ illustrating the final step with Butterfly operations. (Source: Wikipedia)

```
N=128;
x = randn(N,1);
X = fft(x);
ix = 1/N*exp(-1i*2*pi*(0:N-1).'/N).*fft(X(end:-1:1));
norm(x-ix) % Check that the result is correct
```

We leave the derivation to the interested reader.

Overlap and add FFT We have up to now described a filtering case where the input as well as the filter have finite lengths. In many applications it is desired to filter an input of infinite length. This can also be achieved using the FFT by considering the input as an (infinite) sum of finite length signals organized in blocks and then proceed with the technique above for each individual block. Since the filtered output is longer (i.e. $N + M - 1$) than the input (N), the output must be assembled with care by adding the tail of one block to the beginning of the next block. This is method known as the overlap and add FFT algorithm.

Window effects

Before establishing the relation we recall the dual result of convolution involving the point-wise multiplication of two discrete time signals. We have the result:

$$y(n) = x(n)w(n) \quad \Leftrightarrow \quad Y(\omega) = \frac{1}{\omega_s}\int_0^{\omega_s} X(\lambda)W(\omega - \lambda)\, d\lambda \tag{5.15}$$

A multiplication in the time domain is equal to a frequency domain convolution. For the rectangular window ($w(n) = r_N(n)$) we have

$$R_N(\omega) = \sum_{n=-\infty}^{\infty} r_N(n)e^{-j\omega\Delta tn} = e^{-j\frac{N-1}{2}\omega\Delta t}\frac{\sin(\frac{N\omega\Delta t}{2})}{\sin(\frac{\omega\Delta t}{2})} \tag{5.16}$$

As illustrated above, observing only a window of the signal leads to a frequency domain convolution between the DTFT of the full signal and the window function. Previously we have simply used the rectangular window, defined by (4.2). However, there exist many different window functions with varying properties. In principle the different window functions yield different trade-offs regarding the width of the main lobe and the size of the side lobes. Recall that from a frequency resolution perspective the main lobe width should be small and from an accuracy point of view the side lobe should also be small. The properties of a few classical window functions are listed in Table 5.2. The list is ordered by increasing main lobe width. By allowing an increased main lobe width, and hence less resolution, the side lobe level can be significantly decreased, which in turn will reduce the leakage effects. By having a uniform side lobe level, as in the Dolph-Chebyshev window the peak level can be made quite small (-60 dB). This is at the expense of having no roll-off, i.e. all side lobes have equal level.

Window	-3dB bandwidth [Hz]	Peak side-lobe level [dB]	Side-lobe roll off [dB/octave]
Rectangular	$\frac{0.89}{N\Delta t}$	-13	-6
Hanning	$\frac{1.4}{N\Delta t}$	-32	-18
Hamming	$\frac{1.3}{N\Delta t}$	-43	-6
Dolph-Chebyshev	$\frac{1.44}{N\Delta t}$	-60	0

Table 5.2: Window characteristics

Example: Consider a signal $x(n)$ which is defined as

$$x(n) = \sin(2\pi f_0 n) + \sin(2\pi f_1 n) + 0.05\sin(2\pi f_2 n) \tag{5.17}$$

where $f_0 = 0.135$, $f_1 = 0.150$ and $f_2 = 0.3$. The signal thus contain 3 sinusoidal signals. Two closely spaced with unit amplitude and a third sinusoid with a small amplitude located further away. We perform a frequency analysis from 50 samples of the signal by using two alternative window functions:

1. Rectangular window
2. Chebyshev window

The resulting DTFT (calculated in Matlab using DFT employing zero padding to a total length of 10.000) are illustrated in Figure 5.4. The tradeoff between narrow main lobe and small side lobe levels is clearly seen. Using the rectangular window all three components are resolved. However, the third signal with a small amplitude is almost hidden among the side lobes originating from the large amplitude components. However since the Chebyshev window has small side lobe levels now the small amplitude signal component is clearly visible but on the other hand, since the main lobe is wide, the two closely spaced signal components are not resolved anymore. The reason for the result can be seen when comparing the DTFT of the two window functions, see Figure 5.5.

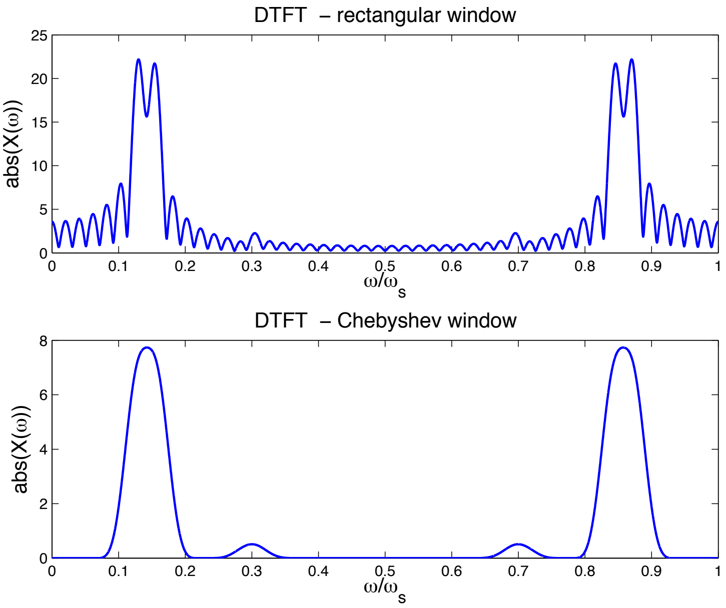


Figure 5.4: Frequency analysis using DTFT.

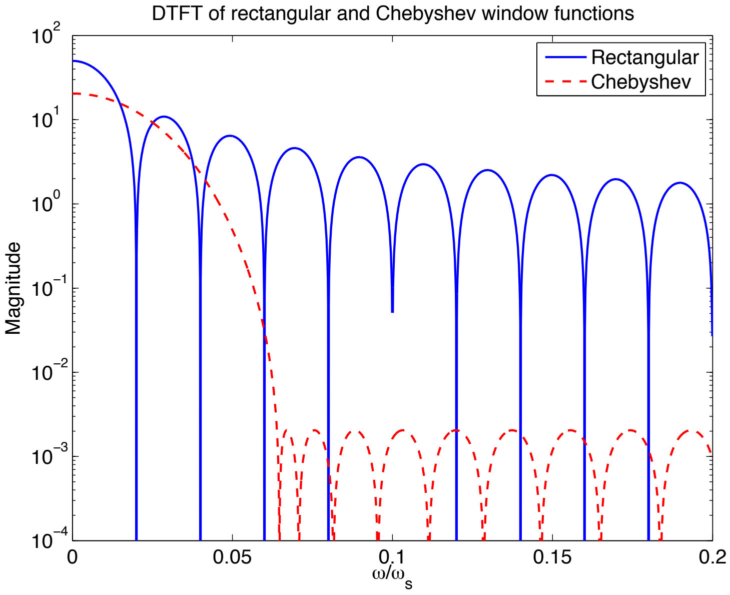


Figure 5.5: DTFT of the rectangular and Chebyshev window of length 50.

Chapter 6

FIR Filter Design

Filter design methods is a quite vast subject and we will here aim at presenting the basic ideas and illustrate fundamental trade-offs involved in the design process. The intended application of the filter is always the starting point in the design and normally involve both performance specifications and other constraints such as complexity in terms of memory space or available computational speed.

Filter structures

Practically filtering can be performed using either a finite impulse response (FIR) structure or within an infinite impulse response (IIR) structure. A causal FIR filter is defined by a finite impulse response and we can write the filter output directly as a convolution

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \tag{6.1}$$

where the impulse response of the filter, $h(k)$, $k = 0, \ldots, M-1$, are the coefficients of the filter. We denote by M the length of the filter, i.e. the length of the impulse response. The frequency function is given by the DTFT as

$$H(\omega) = \sum_{n=0}^{M-1} h(n)e^{-j\omega\Delta tn}. \tag{6.2}$$

An IIR filter is an extension of the FIR case where also past outputs are linearly combined to form the present filter output. An IIR filter is simply

$$y(n) = -\sum_{k=1}^{n_a} a(k)y(n-k) + \sum_{k=0}^{n_b} b(k)x(n-k) \tag{6.3}$$

where $a(k)$ and $b(k)$ are the filter coefficients. The frequency function of an IIR filter is given as the fraction between to polynomials

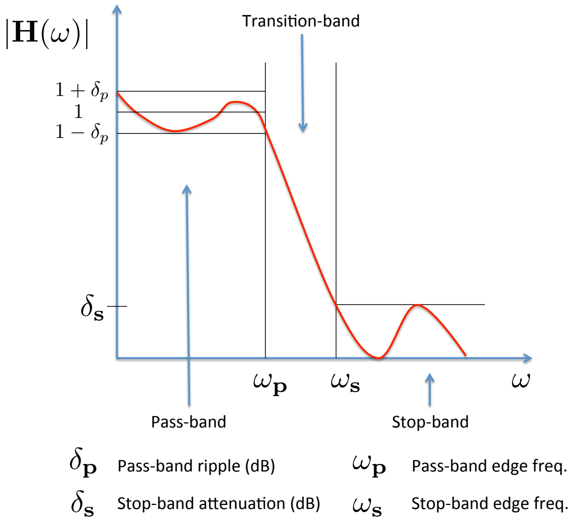
$$H(\omega) = \frac{\sum_{n=0}^{n_b} b(n)e^{-j\omega\Delta tn}}{1 + \sum_{n=1}^{n_a} a(n)e^{-j\omega\Delta tn}}. \tag{6.4}$$

Filter categories

Many signal processing algorithms results in operations which can be regarded as linear filtering and are implemented as (6.1) or (6.3). The objective of the processing algorithms is of course strongly dependent on the application and we can roughly classify filter design into two main categories

- *Frequency selective filters* where the objective of the operations are to enhance or suppress certain bands of the frequency content of the signal
- *Time domain filters* where the objective is defined in the time domain often as a signal matched filter or to minimize a time domain error function.

Frequency selective filters



$|\mathbf{H}(\omega)|$
Transition-band
$1+\delta_p$
1
$1-\delta_p$
$\delta_\mathbf{s}$
$\omega_\mathbf{p}$
$\omega_\mathbf{s}$
ω
Pass-band
Stop-band
$\delta_\mathbf{p}$ Pass-band ripple (dB)
$\omega_\mathbf{p}$ Pass-band edge freq.
$\delta_\mathbf{s}$ Stop-band attenuation (dB)
$\omega_\mathbf{s}$ Stop-band edge freq.

Figure 6.1: Illustration of filter specifications for a frequency selective filter.

Often filters are designed to suppress certain frequency regions while other regions are designed to let the signal pass more or less unaffected. Frequency selective filters can be classified according to their behavior. Common types are:

- Low-pass filter (LP)
- High-pass filter (HP)
- Band-bass filter (BP)
- Band-stop filter (BS)

Design specifications for frequency selective filters of low-pass type are illustrated in Figure 6.1. The specifications are given as specifications on how the amplitude function of the filter should behave. An amplitude function of a frequency selective filter can be divided into three regions; a pass-band region where the amplitude is close to 1, a stop-band region where the amplitude function is small to attenuate well and finally a transition region where the amplitude function moves from the pass-band to the stop-band. The pass-band ripple δ_p is a measure of how much the amplitude function deviates from the ideal 1 in the pass-band region. The stop-band ripple δ_s measure the maximum filter gain in the stop-band region. Finally, the pass-band and stop-band edge frequencies, ω_p and ω_s respectively, define the end of the pass-band region and the start of the stop-band region respectively. The difference $\omega_s - \omega_p$ is the width of the transition region. Obviously a good filter has small ripple and a narrow transition region. Note that for a given filter amplitude function you cannot derive the filter specifications uniquely, i.e. the pass-band ripple size and location of pass-band edge frequency are related via the amplitude function. Sometimes the term *crossover frequency* is used to denote at what frequency the division between the passband and transition region.

Ideal Filter

An ideal LP filter with pass-band edge frequency ω_c can be described in the Fourier domain as

$$H(\omega) = \begin{cases} 1 & |\omega| < \omega_c \\ 0 & \omega_c < |\omega| < \omega_s/2. \end{cases} \tag{6.5}$$

The impulse response is given by the inverse DTFT

$$h(n) = \begin{cases} \frac{2\omega_c}{\omega_s} & n = 0 \\ \frac{2}{\omega_s}\frac{\sin(\omega_c \Delta t n)}{\Delta t n} & n \neq 0 \end{cases} \tag{6.6}$$

and is a sinc function. Key properties are: 1) the impulse response is infinitely long and hence not possible to store in a computer 2) the impulse response is non-causal since $h(n)$ is non-zero also for $n < 0$. Filter design methods which produce realizable filters overcome theses two issues in various ways.

Linear phase, even symmetric and odd symmetric filters

Assume a periodic signal with period P given by

$$x(n) = \sum_{k=0}^{P-1} X(k) e^{j2\pi kn/P} = \sum_{k=0}^{P-1} X(k) e^{j\frac{k}{P}\omega_s \Delta t n} \tag{6.7}$$

is filtered through a linear filter with frequency function

$$H(\omega) = H_r(\omega) e^{-j\omega\tau} \tag{6.8}$$

where $H_r(\omega)$ is a real function. The phase function $\phi(\omega) = -\omega\tau$ of $H(\omega)$ is linear in ω. Such filters are called *linear-phase filters.* The filtered output $y(n)$ is then

$$\begin{aligned} y(n) &= \sum_{k=0}^{P-1} X(k) H_r(\frac{k}{P}\omega_s) e^{j\frac{k}{P}\omega_s \Delta t n} e^{-j\frac{k}{P}\omega_s \tau} \\ &= \sum_{k=0}^{P-1} X(k) H_r(\frac{k}{P}\omega_s) e^{j\frac{k}{P}\omega_s(\Delta t n - \tau)} \end{aligned} \tag{6.9}$$

which is a periodic signal where each component is scaled by the amplitude function of the filter and delayed τ seconds. If the signal $x(n)$ only have significant power in the frequency interval of the pass-band (i.e. where $H_r(\omega) \approx 1$) the output will be a time delayed version of the input. This means that such filters will preserve the pulse shapes.

Even symmetric filters Assume a real-valued finite length impulse response of a filter is even symmetric, i.e.

$$h(-n) = h(n) \quad n = 0, 1 \dots L. \tag{6.10}$$

The frequency function of this filter is given by

$$\begin{aligned} H(\omega) &= \sum_{n=-L}^{L} h(n)e^{-j\omega\Delta tn} = h(0) + \sum_{n=1}^{L} h(n)(e^{-j\omega\Delta tn} + e^{j\omega\Delta tn}) \\ &= h(0) + 2\sum_{n=1}^{L} h(n)\cos(\omega\Delta tn) \end{aligned} \tag{6.11}$$

and is real valued. It has a phase function which takes the value 0 or π depending on the sign of the real valued sum $h(0) + \sum_{n=1}^{L} h(n)\cos(\omega\Delta tn)$. If we now shift this filter to make it causal we obtain a FIR filter of length $M = 2L + 1$ with an impulse response $h(n) = h(M-1-n)$ (which is still often called even symmetric). The frequency function of this filter is given by (6.8) with $\tau = \Delta t(M-1)/2$ and the phase function is given by

$$\phi(\omega) = \angle H(\omega) = \begin{cases} -\omega\Delta t\frac{M-1}{2} & H_r(\omega) > 0 \\ -\omega\Delta t\frac{M-1}{2} + \pi & H_r(\omega) < 0 \end{cases} \tag{6.12}$$

Odd symmetric filters Assume the real-valued finite length impulse response of a filter is odd symmetric, i.e.

$$h(-n) = -h(n) \quad n = 0, 1 \dots L \tag{6.13}$$

which implies $h(0) = 0$. The frequency function of this filter is

$$H(\omega) = \sum_{n=-L}^{L} h(n)e^{-j\omega\Delta tn} = \sum_{n=1}^{L} h(n)(e^{-j\omega\Delta tn} - e^{j\omega\Delta tn}) = -2j\sum_{n=1}^{L} h(n)\sin(\omega\Delta tn) \tag{6.14}$$

which has a phase function which takes the value $\pi/2$ or $-\pi/2$ depending on the sign of the real valued sum $\sum_{n=1}^{L} h(n)\sin(\omega\Delta tn)$. Note also that $H(0) = \sum_{n=-L}^{L} h(n) = 0$ which mean that odd-symmetric filters are not suitable as LP-filters. The causal version of the length $M = 2L + 1$ odd symmetric filter is $h(n) = -h(M-1-n)$. This filter will have the phase response

$$\phi(\omega) = \angle H(\omega) = \begin{cases} -\omega\Delta t\frac{M-1}{2} + \frac{\pi}{2} & H_r(\omega) > 0 \\ -\omega\Delta t\frac{M-1}{2} - \frac{\pi}{2} & H_r(\omega) < 0 \end{cases} \tag{6.15}$$

Imposing even or odd symmetry as a design contraint will lead to a linear phase filter which in many applications is desirable. A symmetric FIR of length M

will consequently also delay the signal $(M-1)/2$ samples corresponding to $\Delta t(M-1)/2$ seconds. Furthermore since half the filter coefficients are identical (with a sign change for odd symmetry) only half the memory and half the number of multiplications are needed in an implementation of such filters.

FIR design with the window method

The window design method is based on modifying the ideal impulse response, e.g., (6.6), to make it both causal and finite. We start from a desired real amplitude response of the filter $H_D(\omega)$. We also assume $H_D(-\omega) = H_D(\omega)$, i.e., symmetric around frequency 0. From this we calculate M impulse response coefficients by solving the inverse DTFT integral for $n = 0, \pm 1, \ldots, \pm m_0$, where we assume M is an odd valued integer and $m_0 = (M-1)/2$.

$$\begin{aligned} h_D(n) &= \frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} H_D(\omega)e^{j\omega\Delta t n}\,d\omega \\ &= \frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} H_D(\omega)(\cos(\omega\Delta t n) + j\sin(\omega\Delta t n))\,d\omega \\ &= \frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} H_D(\omega)\cos(\omega\Delta t n)\,d\omega \end{aligned} \tag{6.16}$$

where the last identity follow from the odd symmetry of the sin function and that $H_D(-\omega) = H_D(\omega)$. This also implies that $h_D(-n) = h_D(n)$, i.e. the impulse response is also even symmetric.

The truncation of the impulse response involves a time domain multiplication with a user selectable window function $w(n)$. The truncated impulse response is defined as

$$h_T(n) = w(n)h_D(n), \quad n = 0, \pm 1, \ldots, \pm m_0 \tag{6.17}$$

The impulse response in (6.16) is non-causal. Finally we make it causal by time shifting it $m_0 = (M-1)/2$ samples leading to the final impulse response

$$h(n) = h_T(n - m_0), \quad n = 0, \ldots, M-1 \tag{6.18}$$

The effect of the truncation of the ideal impulse response can be characterized in the frequency domain by the convolution integral

$$H_T(\omega) = \frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} H_d(\lambda)W(\omega-\lambda)\,d\lambda \tag{6.19}$$

The time shift leads to a phase rotation and the final filter is given by

$$H(\omega) = e^{-j\omega\Delta t\frac{M-1}{2}}H_T(\omega) = e^{-j\omega\frac{M-1}{2}}\frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} H_d(\lambda)W(\omega-\lambda)\,d\lambda \tag{6.20}$$

The choice of window function $w(n)$ will influence the trade-off between the width of the transition region and the level of ripple in the pass- and stop-bands. The behavior is similar to the window effects in frequency analysis using DFT. Increasing the attenuation in the stop-band leads to a larger transition region and vice versa. When employing the window design method the pass-band and stop-band ripple are always equal, i.e. $\delta_p = \delta_s$ and the ideal crossover frequency ω_c is in the center of the transition region. Table 6.1 list some numbers regarding how the width of the transition band and the stop-band attenuation varies with window and filter length. Note that the size of the ripple is independent of the filter length and is only determined from the window type.

Window's name	Trans band	Peak $20\log_{10}\delta_s$
Rectangular	1/M	-21 dB
Hamming	3.3/M	-53 dB
Blackman	5.5/M	-74 dB

Table 6.1: Window function influence on FIR filter designed with the window method.

Example: We now employ the window design methodology to construct FIR low pass filters of order $M = 61$ with a cut off frequency of 0.25 (relative to the sampling frequency). Solving the integral (6.16) with $\Delta t = 1$ yields

$$\begin{aligned} h_D(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} H_D(\omega)\cos(\omega n)\,d\omega = \frac{1}{2\pi}\int_{-0.25\cdot 2\pi}^{0.25\cdot 2\pi}\cos(\omega n)\,d\omega \\ &= \frac{\sin(0.25\cdot 2\pi n)}{\pi n} \end{aligned} \tag{6.21}$$

Delaying the impulse response with $m_0 = (M-1)/2 = 30$ and multiplying with the delayed window function $w(n)$ yields

$$h(n) = w(n-30)\frac{\sin(0.25\times 2\pi(n-30))}{\pi(n-30)} \quad ,n = 0,\ldots,60. \tag{6.22}$$

With $m_0 = (M-1)/2$, the Hamming window function is defined by

$$w_H(n) = 0.54 + 0.46\cos\frac{2\pi(n+m_0)}{M-1} \tag{6.23}$$

and the Blackman window by

$$w_B(n) = 0.42 - 0.5\cos\frac{2\pi(n+m_0)}{M-1} + 0.08\cos\frac{4\pi(n+m_0)}{M-1}, \tag{6.24}$$

The three window functions are shown in Figure 6.2 and the magnitude of the final FIR filters are illustrated in Figure 6.3.

Optimal FIR design methods

Since the frequency function of the filter is a linear function of the parameters of the filter, i.e. the impulse response, several optimal methods can be used for designing filteres based on minimizing a functional which incorporates the design specification. Here we describe the Least-Squares method and the equiripple design method.

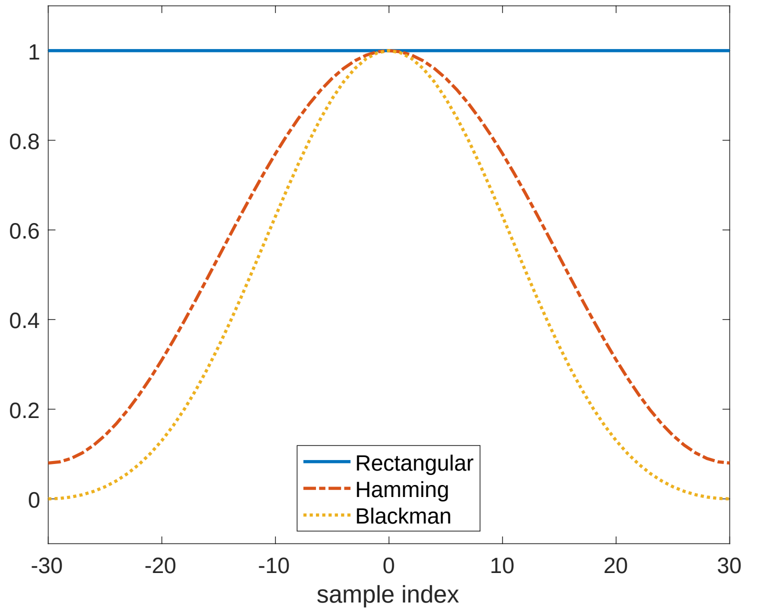


Figure 6.2: Illustration of the Rectangular, Hamming and Blackman window functions for $M = 61$.

FIR-LS The Least-Squares method minimizes the sum of the squared frequency function deviation at a given set of N_{spec} specification frequencies $\{\omega_k\}$

$$\min_{\mathbf{h}} \sum_{k=1}^{N_{\text{spec}}} W_K |H_D(\omega_k) - H(\omega_k, h)|^2 \tag{6.25}$$

where $H_D(\omega_k)$ is the desired frequency response, $H(\omega, \mathbf{h})$ is the frequency response of the designed filter and $\mathbf{h} = \left[{}_{h(0),h(1),\ldots,h(M-1)} \right]$ is the vector of filter coefficients. The optional positive weights W_k can be used to shape the error to better suit the specifications. The solution to (6.25) is easily obtained by solving the associated normal equations. In The MATLAB command for the design method is called `firls`.

FIR-PM The second design method focus on finding the filter solution which minimizes the maximum deviation of the filter from the desired specification. Formally the designed filter is the solution to the min-max problem

$$\min_{\mathbf{h}} \max_{\omega} |H_D(\omega_k) - H(\omega_k, h)|^2. \tag{6.26}$$

The solution can be found by employing the Remez exchange algorithm and will deliver a filter where all the ripples in the approximation error have an equal magnitude. The method is known as the Parks and McClellan filter design algorithm published By Thomas Parks and James McClellan in 1972 and was the result of a student project. In MATLAB the command for the design method is `firpm`.

Hence, if we desire to have a fixed phase $\pi/2$ in the passband we can obtain this by enforcing the impulse response to be odd symmetric. For example,

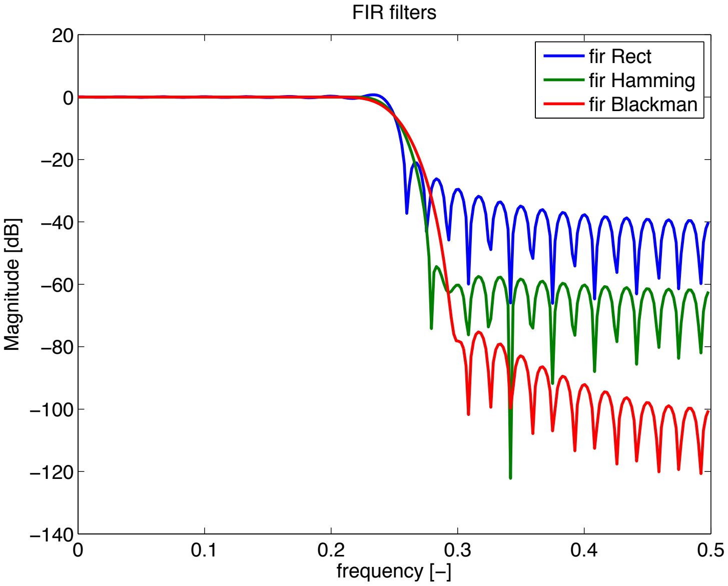


Figure 6.3: Illustration of the final FIR filters using the window design with Rectangular, Hamming and Blackman windows and $M = 61$.

filters which approximate the differentiation operator should have this property. In Matlab command `firpm` which designes equi-ripple filters odd symmetric impulse responses can be enforced by supplying the extra command argument `'differentiator'` (which also adds a weighting which makes the response more accurate for low frequencies) .

Chapter 7

IIR filter design

IIR filter design with bilinear transformation

The frequency function of IIR filters is a fraction of two polynomials.

$$H(\omega) = \frac{\sum_{n=0}^{n_b} b(n)e^{-j\omega\Delta tn}}{1 + \sum_{n=1}^{n_a} a(n)e^{-j\omega\Delta tn}}. \tag{7.1}$$

Hence, the frequency function is *not* a linear function of the denominator polynomial coefficients $a(n)$ which make the filter design less straight forward as compared to the FIR case. However, analog filter design is well known and several standard designs are available. Butterworth, Chebyshev and Elliptical filters are a few of the more common ones. With use of the bilinear transformation these design methodologies can be incorporated also when designing digital IIR filters.

Consider the following identity

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

where s and z are complex variables. This is known as the *bilinear transformation* or *Tustins transformation.* The inverse of this transformation is easily determined as

$$z = \frac{2 + s}{2 - s} \tag{7.3}$$

Now we interpret z as the Z-transform variable and s as the Laplace variable. In the discrete time domain we are interested in z values on the unit circle $z = e^{j\omega\Delta t}$ where the discrete time frequency ω is in rad/s. In the Laplace domain, s values on the imaginary axis ($s = j\Omega$) is of importance. If $H(s)$ is the Laplace transform of a filter then $H(j\omega)$ is the frequency function. The transformation and its inverse are well defined for all s on the imaginary axis where $s = j\Omega$. Here Ω denotes the analog frequencies in rad/s.

The bilinear transformation ties the two frequency scales together as follows. Starting from (7.2) and letting $z = e^{j\omega\Delta t}$ we obtain

$$\begin{aligned} s = j\Omega = \frac{2(1 - e^{-j\omega\Delta t})}{1 + e^{-j\omega\Delta t}} &= \frac{2(e^{j\omega\Delta t/2} - e^{-j\omega\Delta t/2})}{e^{j\omega\Delta t/2} + e^{-j\omega\Delta t/2}} \\ = 2j\frac{\sin(\omega\Delta t/2)}{\cos(\omega\Delta t/2)} &= 2j\tan(\omega\Delta t/2) \end{aligned} \tag{7.4}$$

and consequently the relation between the two frequency scales is given by

$$\Omega = 2\tan(\omega\Delta t/2) \tag{7.5}$$

From (7.5) we notice that $\omega = 0$ implies $\Omega = 0$ and for increasing ω the continuous frequencies will also increase and as $\omega \to \pi$ then $\Omega \to \infty$. Hence the entire continuous time frequency region $\Omega \in [0, \infty)$ is mapped to the upper half of the unit circle. The negative frequencies are mapped to the lower half of the unit circle.

One of the important applications of the bilinear transformation is that it provides a way of transforming a continuous time filter (transfer function) to a discrete time one while still preserving a number of important properties.

The transformation is applied as follows. Start from a Laplace domain filter $H_a(s)$ which is a rational function in the free variable s. A rational transfer function is the fraction of two polynomials. The bilinearly transformed discrete time (Z-domain) transfer function $H_d(z)$ is defined as

$$H_d(z) \triangleq H_a(s)\big|_{s=\frac{2(1-z^{-1})}{1+z^{-1}}} \tag{7.6}$$

Note that $H_d(z)$ is also a rational transfer function. The MATLAB command `bilinear` performs the bilinear transformation.

The following nice properties can be identified:

- It is easy to verify that the number of poles of $H_d(z)$ and $H_a(s)$ coincide. This means that the order of the filter is unchanged.
- If $H_a(s)$ is a stable transfer function so is also $H_d(z)$.
- The complex filter response, i.e. both magnitude and phase, coincides for frequencies related by (7.5).

$$H_a(j\Omega) = H_d(e^{j\omega\Delta t}) \tag{7.7}$$

The last feature implies that the bilinear transformation preserves the type of filter. If $H_a(s)$ is a low-pass filter then $H_d(z)$ defined by (7.6) is also a low-pass filter etc.

The design technique is summarized here:

1. Start by identifying the discrete time frequencies describing the desired edge frequencies.
2. Calculate the corresponding analog frequencies using $\Omega = 2\tan(\omega\Delta t/2)$.
3. Design an analog filter $H_a(s)$ based on the specifications.
4. Use the bilinear transformation to obtain the final digital (Z-domain) filter.

$$H_d(z) \triangleq H_a(s)\big|_{s=\frac{2(1-z^{-1})}{1+z^{-1}}} \tag{7.8}$$

Example 7 Let us design an discrete time low pass filter based on an continuous time first order Butterworth filter. The discrete time filter should have a -3 dB reduction of the gain at normalized frequency $f/f_s = 0.2$.

Property	IIR	FIR
Flexible design techniques	-	+
Complexity v.s. stop-band attenuation	+	-
Linear phase	-	+
Robustness to filter coefficient truncation	-	+
Guaranteed stability	-	+

Table 7.1: Comparison of FIR and IIR filter properties

A continuous time Butterworth filter has a frequency function

$$H_a(\Omega) = \frac{1}{1 + j\Omega/\Omega_c} \tag{7.9}$$

and for $\Omega = \Omega_c$ the magnitude will be $1/\sqrt{2}$ which corresponds to -3 dB gain. Based on the discrete time specification we obtain $\Omega_c = 2\tan(0.2\pi)$ and the Laplace transform of the filter is

$$H_a(s) = \frac{1}{1 + s/\Omega_c} \tag{7.10}$$

Finally the discrete time filter is obtained by the substitution in (7.8) and we obtain

$$H_d(z) = \frac{1}{1 + \frac{2(1-z^{-1})}{1+z^{-1}}\frac{1}{\Omega_c}} = \frac{\Omega_c(1+z^{-1})}{\Omega_c + 2 + z^{-1}(\Omega_c - 2)} \tag{7.11}$$

which is a first order discrete time LP-filter. It is easy check that the corresponding frequency function $H_d(\omega)$ is equal to $1/\sqrt{2}$ for $\omega = 2\pi 0.2$. □

Comparison of FIR and IIR filters

Table 7.1 compares how FIR and IIR filters perform regarding some filter properties.

Chapter 8

FIR filter design based on local parametric signal models

In the signal processing chain linear filtering is often applied to mitigate unwanted signal components or disturbances that clutters the desired signal. Previously we have designed frequency selective filters which are versatile when there is a clear separation between the desired and undesired signal in the frequency domain. In this chapter we will base the filter design on models of the desired signal formulated in the time domain. In some cases it is possible to view a portion or window of the desired signal as a signal which can be described by a parametric model. We will here discuss a FIR filter design technique which is based on a linearly parametrized local signal model.

A motivating example

A simplest example could be if the desired signal varies slowly. For a given window of data, i.e./ signal $x(n)$ around sample n we assume a signal model

$$\hat{y}(n+m, \alpha_0) = \alpha_0, \quad -M \leq m \leq M \tag{8.1}$$

i.e., we assume the signal has a fixed constant value α_0 in the analyzed window. The signal value of the model at sample n is trivially $\hat{y}(n|\alpha_0) = \alpha_0$. For this case it seems plausible that a good estimate of the value of α_0 is simply the average of the original signal $x(n)$ in a window around sample index n

$$\hat{\alpha}_0 = \frac{1}{2M+1} \sum_{k=-M}^{M} x(n+k) \tag{8.2}$$

and our local estimated signal model evaluated at sample n is now

$$y(n) = \hat{y}(n, \hat{\alpha}_0) = \hat{\alpha}_0 = \frac{1}{2M+1} \sum_{k=-M}^{M} x(n+k) \tag{8.3}$$

which we can interpret as a non-causal FIR filter with impulse response

$$h(k) = \begin{cases} \frac{1}{2M+1} & -M \leq k \leq M \\ 0 & \text{otherwise} \end{cases} \tag{8.4}$$

It is an easy exercise to show that (8.2) is the solution to the least-squares problem

$$\min_{\alpha_0} \sum_{k=-M}^{M} |x(n+k) - \hat{y}(n+k|\alpha_0)|^2 \,. \tag{8.5}$$

We have now illustrated a filter design procedure which is based on the following parts

1. a parametrized local signal model
2. a least-squares optimization step to find the parameters of the signal model using a window of the input data around sample n
3. use the model to calculate the filter output value at sample n

The end result is a non-causal linear FIR filter which is time invariant.

A general formulation

The basic idea outlined in the example above can be generalised to the class of linearly parametrized local signal models,

$$\hat{y}(n+m, \boldsymbol{\alpha}) = \sum_{i=0}^{p-1} \alpha_i f_i(m) \tag{8.6}$$

where $f_i(m)$ $i = 0, \ldots, p-1$ are p distinct given basis functions and the vector

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_0 & \alpha_1 & \ldots & \alpha_{p-1} \end{bmatrix}^T \tag{8.7}$$

collect the parameters. The least-squares problem is now given by

$$\min_{\boldsymbol{\alpha}} \sum_{k=-M}^{M} |x(n+k) - \hat{y}(n+k, \boldsymbol{\alpha})|^2 \tag{8.8}$$

which can be reformulated as

$$\min_{\boldsymbol{\alpha}} \|\mathbf{R}\boldsymbol{\alpha} - \mathbf{x}\|^2 \tag{8.9}$$

where

$$\mathbf{R} = \begin{bmatrix} f_0(-M) & f_1(-M) & \cdots & f_{p-1}(-M) \\ f_0(-M+1) & f_1(-M+1) & \cdots & f_{p-1}(-M+1) \\ \vdots & \vdots & \cdots & \vdots \\ f_0(0) & f_1(0) & \cdots & f_{p-1}(0) \\ \vdots & \vdots & \cdots & \vdots \\ f_0(M) & f_1(M) & \cdots & f_{p-1}(M) \end{bmatrix} \tag{8.10}$$

is the regressor matrix which is uniquely given by samples of the different basis functions. The input data in the window is collected in the vector

$$\mathbf{x} = \begin{bmatrix} x(n-M) & \cdots & x(n) & \cdots & x(n+M) \end{bmatrix}^T . \tag{8.11}$$

If $\mathbf{R}$ has full rank the solution to (8.9) is

$$\hat{\boldsymbol{\alpha}} = (\mathbf{R}^H\mathbf{R})^{-1}\mathbf{R}^H\mathbf{x} \tag{8.12}$$

where $(\cdot)^H$ denote complex conjugate and transpose. The value at a specific sample $n + m_0$ for the estimated signal model can then be formulated as

$$\begin{aligned} \hat{y}(n+m_0, \hat{\boldsymbol{\alpha}}) &= \sum_{i=0}^{p-1} \hat{\alpha}_i f_i(m_0) = \mathbf{f}^T(m_0)\hat{\boldsymbol{\alpha}} \\ &= \mathbf{f}^T(m_0)(\mathbf{R}^H\mathbf{R})^{-1}\mathbf{R}^H\mathbf{x} = \sum_{k=-M}^{M} h(k)x(n-k) \end{aligned} \tag{8.13}$$

where

$$\mathbf{f}^T(m) = \begin{bmatrix} f_0(m) & f_1(m) & \cdots & f_{p-1}(m) \end{bmatrix} \tag{8.14}$$

i.e. $\mathbf{f}^T(m)$ is one of the rows in the matrix $\mathbf{R}$ (if $|m| \leq M$) . The impulse response of the FIR filter is given by the relation

$$\begin{bmatrix} h(M) & \ldots & h(0) & \cdots & h(-M) \end{bmatrix} = \mathbf{f}^T(m_0)(\mathbf{R}^H\mathbf{R})^{-1}\mathbf{R}^H \tag{8.15}$$

i.e. the right hand of (8.15) is the time reversed impulse response. We notice that the impulse response $h(k)$ only depends on the selected basis functions and m_0 and is independent of the input signal $x(n)$.

It is clear that the parameters of the local model is determined by the input signal in the window and is independent of what choice of m_0 is made. The choice of m_0 control what sample index we want to evaluate the estimated model at. If $m_0 = 0$ then the midpoint index of the window is selected and the resulting filter will be non-causal since

$$y(n) = \sum_{k=-M}^{M} h(k)x(n-k) \tag{8.16}$$

which is known as a smoothing filter. If we select $m_0 = M$ we obtain

$$y(n+M) = \sum_{k=-M}^{M} h(k)x(n-k) \tag{8.17}$$

which is a causal filter since the value at index $n{+}M$ is based on sample $x(n{+}M)$ and older ones. If we select $m > M$ we ask the local signal model to predict a signal value in the future, i.e. for a time index outside the observation window. In fact, if the basis functions $f_i(m)$ are well defined for $m \in \mathbb{R}$ we can decide to use any real value m_0 to query the signal model. This opens up to define filters which estimates the signal values at arbitrary locations in time.

A weighted general formulation

Since the technique approximates a local model to a portion of the input signal it is quite natural to ask for a better model fit near the sample we want to predict (at index $n + m_0$) and accept more deviation further away. We can easily incorporate this behaviour by adding a positive weight function to the least-squares problem. The weighted LS formulation is

$$\min_{\boldsymbol{\alpha}} \sum_{k=-M}^{M} w_k \left| x(n+k) - \hat{y}(n+k, \boldsymbol{\alpha}) \right|^2 = \min_{\boldsymbol{\alpha}} \| \mathbf{W}^{1/2} (\mathbf{R}\boldsymbol{\alpha} - \mathbf{x}) \|^2 \tag{8.18}$$

and $\mathbf{W} = \mathbf{W}^{1/2}\mathbf{W}^{1/2}$ is a diagonal matrix

$$\mathbf{W} = \begin{bmatrix} w_{-M} & 0 & \cdots & 0 \\ 0 & w_{1-M} & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & w_M \end{bmatrix} \tag{8.19}$$

The FIR filter coefficients solution for the weighted problem is given by

$$\begin{bmatrix} h(M) & \ldots & h(0) & \cdots & h(-M) \end{bmatrix} = \mathbf{f}^T(m_0)(\mathbf{R}^H\mathbf{W}\mathbf{R})^{-1}\mathbf{R}^H\mathbf{W} \tag{8.20}$$

A possible choice of weight function is $w_k = \lambda^{|k-m_0|}$ where $0 < \lambda < 1$ which provides an exponential weight decrease away from the sample with index $n+m_0$.

Monomial basis functions - Savitzky-Golay filter

The simplest choice of basis functions f_i is to use monomials, i.e.

$$\begin{aligned} f_0(m) &= 1, \quad i = 0 \\ f_i(m) &= m^i, \quad i = 1, \ldots, p-1 \end{aligned} \tag{8.21}$$

for some order p. The initial example above had $p = 1$. if $p = 2$ the model is affine

$$\hat{y}(n+m, \boldsymbol{\alpha}) = \alpha_0 + \alpha_1 m \tag{8.22}$$

or if $p = 3$ we obtain a quadratic model

$$\hat{y}(n+m, \boldsymbol{\alpha}) = \alpha_0 + \alpha_1 m + \alpha_2 m^2 \tag{8.23}$$

etc. A model with a higher order is more flexible and will provide a better fit to the given data in the window. The efect is that the resulting filter will have a higher bandwidth. The choice of M which control the window size also influence the behaviour of the final filter. It is clear the total length of the FIR filter is $2M+1$. If M increases when the model order is fixed the bandwidth of the filter will decrease. For model order $p = 1$, the final filter is an averaging filter. The bandwidth of such a filter is inversely proportional to the length of the filter since the magnitude of the frequency response function is a sinc function (for a derivation see (4.2) - (4.4)). The filters derived using monomial basis functions have a long history but are commonly known as Savitzsky-Golay filters, named by the authors of a paper [5] which popularised the technique. The tutorial style contribution in [6] gives a nice signal processing flavoured introduction to the topic.

Applying linear operators on the local signal model

Once derived from the data samples in the window, the local model $\hat{y}(n+m, \boldsymbol{\alpha})$ can be viewed as a function of the continuous independent variable m and if we apply a linear operator $L(\cdot)$ on the signal model we obtain

$$L(\hat{y}(n+m, \boldsymbol{\alpha})) = \sum_{i=0}^{p-1} \alpha_i L(f_i(m)) \tag{8.24}$$

The linear operator applied to the signal can then also be expressed as a linear FIR filter where the filter coefficients are given by

$$\begin{bmatrix} h(M) & \dots & h(0) & \cdots & h(-M) \end{bmatrix} = L(\mathbf{f}^T(m_0))(\mathbf{R}^H\mathbf{W}\mathbf{R})^{-1}\mathbf{R}^H\mathbf{W} \tag{8.25}$$

and

$$L(\mathbf{f}^T(m)) = \begin{bmatrix} L(f_0(m)) & L(f_1(m)) & \cdots & L(f_{n-1}(m)) \end{bmatrix}. \tag{8.26}$$

We can interpret this in the following way. The input signal will provide an estimate of the parameter vector $\boldsymbol{\alpha}$. This estimate does not depend on the operator $L(\cdot)$. The operator however controls the result of applying the operator on the signal model and deliver the output $L(\hat{y}(n+m, \boldsymbol{\alpha}))$.

A common application where this approach is useful is when we want to determine the gradient, or rate of change, of the signal and how this rate changes over time. By employing the approach outline above we can construct a FIR filter that directly provides an estimate of the gradient of the input signal. As en example let as use the monomial basis functions $f_0(m) = 1$ and $f_1(m) = m$ and let $L(f_i) = \frac{\partial}{\partial m} f_i(m)$. This gives the FIR filter

$$\begin{aligned} \begin{bmatrix} h(M) & \dots & h(0) & \cdots & h(-M) \end{bmatrix} &= L(\mathbf{f}^T(0))(\mathbf{R}^H\mathbf{W}\mathbf{R})^{-1}\mathbf{R}^H\mathbf{W} \\ &= \begin{bmatrix} 0 & 1 \end{bmatrix} (\mathbf{R}^H\mathbf{W}\mathbf{R})^{-1}\mathbf{R}^H\mathbf{W}. \end{aligned} \tag{8.27}$$

Chapter 9

Multirate signal processing

Techniques which change the sample rate of a signal are known as Multirate Signal Processing and are key tools to efficiently process signals. In principle, the most efficient way, in terms of operations per second, is to use a sample rate as low as possible. The bandwidth of the processed signal determines this minimal rate. The key operations are

- up-sampling
- down-sampling
- filtering

which are combined to perform the desired operations.

Up-sampling The operation when the sample rate is increased by inserting zeros between the original signal values is know as up-sampling. If the rate change is L, the number of inserted zeros between the samples are $L-1$. If the original signal is $x(n')$ the up-sampled signal $x_u(n)$ can be described by

$$x_u(n) \triangleq \begin{cases} x(n/L), & n = 0, \pm, L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases} \tag{9.1}$$

For notational purposes we use the notation

$$x_u(n) = \Uparrow_L [x(n')] \tag{9.2}$$

to denote a factor L up-sampler operator. Since all orignal signal samples are retained in the up-sampled signal no information is lost in the operation. If the original sampling frequency is ω_s with corresponding sampling interval $\Delta t = 2\pi/\omega_s$, the up-sampled signal have sampling frequency $\omega_{su} = L\omega_s$ and sampling interval $\Delta t_u = \Delta t/L$. The DTFT of the up-sampled signal can be determined from the DTFT of the original signal as follows.

$$\begin{aligned}
X_u(\omega) &= \sum_{n=-\infty}^{\infty} x_u(n)e^{-j\omega n\Delta t_u} = [m = n/L \text{ and } \Delta t_u = \Delta t/L] \\
&= \sum_{m=-\infty}^{\infty} x(m)e^{-j\omega L m\Delta t/L} = [n = m] \\
&= \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n\Delta t} = X(\omega)
\end{aligned} \tag{9.3}$$

where ω is un-normalized, i.e. rad/s. The DTFT of the up-sampled signal is simply the DTFT of the original signal. The DTFT of the up-sampled signal, in the frequency range between 0 and ω_{su}, contain the original DTFT and $L-1$ images of it. A low-pass type signal which is up-sampled will thus also contain energy at frequencies higher than the Nyquist frequency (in Hz or rad/s) of the original signal.

Down-sampling The method of reducing the sample rate by only including a subset of the original signal samples is called down-sampling. A factor M down-sampling is simply defined as

$$x_d(n) \triangleq \Downarrow_M[x(n')] \triangleq x(Mn), \quad n = 0, \pm 1, \pm 2, \ldots \tag{9.4}$$

where $x_d(n)$ is the down-sampled signal associated with the new sample rate $\omega_{sd} = \omega_s/M$ assuming the original signal has sample frequency ω_s. The down-sampled signal has sampling interval $\Delta t_d = \Delta t M$.

Since only a subset of the original samples are retained in the produced signal, down-sampling can (possibly) lead to loss of information or distortion unless the bandwidth of the original signal is small enough. To see this let us derive the DTFT of the down-sampled signal. Consider expressing the down-sampled signal in terms of the inverse DTFT of the original signal:

$$\begin{aligned}
x_d(n) = x(nM) &= \frac{1}{\omega_s}\int_0^{\omega_s} X(\omega)e^{j\omega n M\Delta t}\,d\omega = [\Delta t_d = \Delta t M] \\
&= \frac{1}{\omega_s}\sum_{k=0}^{M-1}\int_{\omega_s k/M}^{\omega_s(k+1)/M} X(\omega)e^{j\omega n\Delta t_d}\,d\omega = [\omega = \lambda + \omega_s k/M] \\
&= \frac{1}{\omega_s/M}\int_0^{\omega_s/M} \underbrace{\frac{1}{M}\sum_{k=0}^{M-1} X\left(\lambda + \frac{\omega_s k}{M}\right)}_{X_d(\lambda)} e^{j\lambda n\Delta t_d}\,d\lambda.
\end{aligned} \tag{9.5}$$

Since the last expression is the inverse DTFT of x_d we have shown that

$$X_d(\omega) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(\omega + \frac{\omega_s k}{M}\right). \tag{9.6}$$

From the expression above we note that sampling distortion (i.e. loss of information) can occur if, for each value of ω more than one term in the summation (9.6) is non-zero. Distortion will *always* happen if the bandwidth of the original signal (counting the full band [0 to ω_s]) is larger than ω_s/M.

Ideal DT reconstruction

Assume a DT signal $x(n')$ has a DTFT such that

$$X(\omega) = 0, \quad \text{for} \quad \frac{\omega_s}{2M} \leq |\omega| \leq \frac{\omega_s}{2} \tag{9.7}$$

which mean it has low pass character. Define the factor M downsampled signal as

$$x_d(n) = x(nM). \tag{9.8}$$

The discrete time Fourier transform of this downsampled signal is given by (9.6) and due to (9.7) we have

$$X_d(\omega) = \frac{1}{M} X(\omega), \quad \text{for} \quad 0 \leq |\omega| \leq \frac{\omega_s}{2M} \tag{9.9}$$

It can be shown (in a similar fashion as the derivation of (3.2)) that the original signal $x(n)$ can be reconstructed exactly from the down-sampled signal by

$$x(n) = \sum_{k=-\infty}^{\infty} x_d(k) \frac{\sin(\frac{\pi}{M}(n - kM))}{\frac{\pi}{M}(n - kM)}. \tag{9.10}$$

We start the derivation by using the inverse DTFT to express the samples in $x(n)$ and we obtain

$$x(n) = \frac{1}{\omega_s} \int_{-\omega_s}^{\omega_s} X(\omega) e^{j\omega\Delta tn} \, d\omega = \frac{1}{\omega_s} \int_{-\omega_s/M}^{\omega_s/M} M X_d(\omega) e^{j\omega\Delta tn} \, d\omega \quad = \frac{1}{\omega_s/M} \int_{-\omega_s/M}^{\omega_s/M} \sum_{k=-\infty}^{\infty} x_d(k) e^{-j\omega\Delta t_d k} e^{j\omega\Delta} \tag{9.11}$$

In what follows we show that this ideal reconstruction is equivalent of first upsampling the signal $x_d(n)$ and then perform ideal LP-filtering where the end of the pass-band is at frequency $\omega_s/(2M)$. Let $x_u(n)$ be the rate M upsampled signal $x_d(n)$. Then we have

$$x(n) = \sum_{n'=-\infty}^{\infty} x_u(n) \frac{\sin(\frac{\pi}{M}(n - n'))}{\frac{\pi}{M}(n - n')} = \sum_{k=-\infty}^{\infty} \frac{\sin(\frac{\pi}{M}k)}{\frac{\pi}{M}k} x_u(n - k) \tag{9.12}$$

which we recognize as convolution with the impulse response

$$h(k) = \frac{\sin(\frac{\pi}{M}k)}{\frac{\pi}{M}k} \tag{9.13}$$

with the frequency function

$$H(\omega) = \begin{cases} M, & 0 \leq |\omega| < \frac{\omega_s}{2M} \\ 0, & \end{cases} \tag{9.14}$$

We can formulate the entire process as as the identity

$$x(n) = H(z) \Uparrow_M [\Downarrow_M [x(n)]] \tag{9.15}$$

where the operator $H(z)$ denotes DT convolution, i.e.

$$H(z) = \sum_{k=-\infty}^{\infty} h(k) z^{-k} \quad \Leftrightarrow \quad y(n) = H(z)x(n) = \sum_{k=-\infty}^{\infty} h(k)x(n - k) \tag{9.16}$$

since $z^{-k}x(n) = x(n - k)$. The ideal DT reconstruction is hence equivalent to LP filtering with an ideal filter. We finally note that the impulse response of the ideal filter is both infinitely long and non-causal.

Interpolation and decimation

By combining the up-sampling and down-sampling operations with filtering we obtain practically useful function blocks. As explained in the previous section ideal reconstruction can be seen as a combination of upsampling followed by an ideal LP-filter.

Interpolation An general *interpolation* function block consists of an up-sampler followed by a filter which removes the unwanted images of the signal. Mathematically we can summarize the function as

$$x_I(n) = H_I(z) \Uparrow_L [x(n')]. \tag{9.17}$$

Normally the interpolation filter $H_I(z)$ is a low-pass filter with a cut off frequency at around $\omega_c = \frac{\omega_s}{2L}$ (where ω_s is the sampling frequency of the up-sampled signal) and the resulting signal will be a smooth version of the original low rate signal. If the filter is an ideal LP filter we obtain ideal reconstruction. If, on the other hand, the filter selects some of the images higher up in frequency, the function will act like a modulator which moves the information from a base-band to a higher frequency band. Of course, the position of the modulated signal is limited to the location dictated by the location of the images resulting from the up-sampling step.

Decimation In order to reduce the sample rate in a useful manner the potentially harmful aliasing caused by the down-sampling must be reduced. A digital anti-aliasing filter is used to achieve the desired attenuation before the signal is down-sampled. After the filtering operation, the bandwidth of the signal should be limited to ω_s/M (counting all frequencies from 0 to ω_s, or from $-\omega_s/2$ to $\omega_s/2$), where M is the down-sampling factor and ω_s is the sampling frequency of the original signal. The combination of a filter and a down-sampling operation is called *decimation*. The operation can be described as

$$x_D(n) = \Downarrow_M [H_D(z)x(n')] \tag{9.18}$$

where $H_D(z)$ is the decimation filter. The most common decimation filter type is a low-pass filter assuming the information to retain is in the lowest frequency band. Then the decimation filter $H_D(z)$ is a low-pass filter with a cut off frequency at around $\omega_c = \frac{\omega_s}{2M}$ (where ω_s is the sampling frequency of original (high-rate) signal). If instead a band-pass or a high-pass filter is used, leads to a demodulation operation where the information in the passband will be moved to the baseband due to the sampling operation. See (9.6) for the exact relation.

Rate change By combining an L factor up-sampler, a filter and an M factor down sampler we obtain a *rate converter* which changes the sample rate to L/M times the old rate.

$$y(n) = \Downarrow_M [H(z) \Uparrow_L [x(n')]] \tag{9.19}$$

Here we can view the interpolation and decimation filter combined into one filter $H(z)$. Note that normally the up-sampler must be used *before* the down-sampler to yield the desired result. If the orders are reversed we are faced with possible sampling distortion problems unless the original signal $x(n)$ has a bandwidth limited to ω_s/M (counting all frequencies from 0 to ω_s) .

Cascaded implementation If M or L are needed to be large integers (and non-prime) it is from an implementation point of view better to cascade several interpolation or decimation steps, e.g. $L = L_1 L_2$.

$$y(n) = H_2(z)\Uparrow_{L_2}[H_1(z)\Uparrow_{L_1}[x(n')]] \tag{9.20}$$

Also if $L_1 = L_2$, it is also possible to reuse the filters, i.e. $H_1(z) = H_2(z)$ which means that only one filter need to be designed and stored.

Polyphase implementation

At first sight the interpolation and decimation operations seems costly since the filtering is performed at the higher sample rate. In the interpolation case the filter is feed with $L-1$ zeros at known time indices which of course we could take advantage of in the filter structure. In a similar fashion it is unnecessary to actually calculate the output of the decimation filter for those time indices which in the next step are discarded by the down-sampler. This can be described mathematically by utilizing a special result known as the "Noble identities". First consider a FIR filter in its operator form $H(z)$

$$H(z) = \sum_{k=0}^{K-1} h(k) z^{-k} \quad \Leftrightarrow \quad y(n) = H(z)x(n) = \sum_{k=0}^{K-1} h(k)x(n-k) \tag{9.21}$$

The filter $H(z^L)$ is defined by

$$H(z^L) = \sum_{k=0}^{K-1} h(k) z^{-kL} \quad \Leftrightarrow \quad y(n) = H(z^L)x(n) = \sum_{k=0}^{K-1} h(k)x(n-kL). \tag{9.22}$$

The noble identities are the following operator results

$$H(z)\Downarrow_M[x(n)] = \Downarrow_M[H(z^M)x(n)], \quad H(z^L)\Uparrow_L[x(n)] = \Uparrow_L[H(z)x(n)] \tag{9.23}$$

which show us how to move the filtering operations to the lower rate side. In order to achieve this for any decimation or interpolation filter we need to decompose an arbitrary filter into a sum of filters of the type $H(z^M)$ (or $H(z^L)$). Consider an arbitrary FIR filter which (if necessary) is zero padded to a length $K = LP$ where L and P are integers. Then

$$\begin{aligned} H(z) &= \sum_{k=0}^{K-1} h(k) z^{-k} = [k = pL + l] \\ &= \sum_{l=0}^{L-1} z^{-l} \underbrace{\sum_{p=0}^{P-1} h(pL+l) z^{-pL}}_{H_l(z^L)} \\ &= \sum_{l=0}^{L-1} z^{-l} H_l(z^L) \end{aligned} \tag{9.24}$$

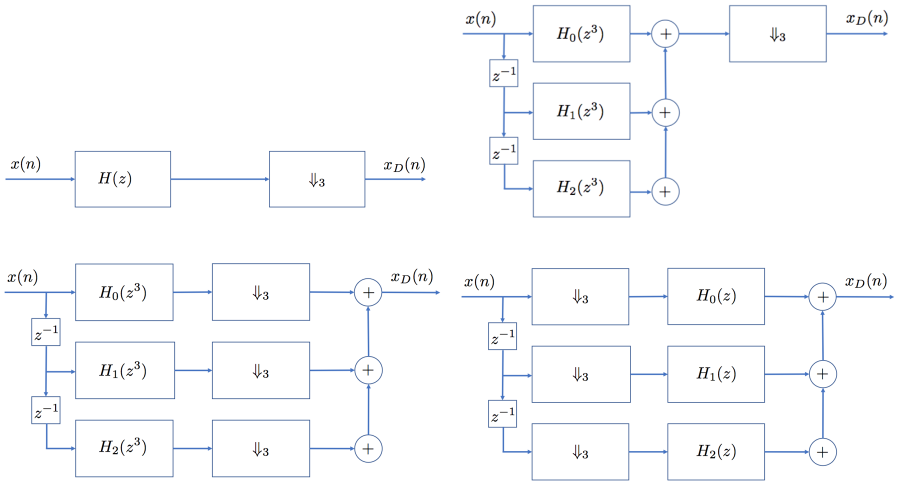


Figure 9.1: Example of processing structure for factor 3 decimation of signal $x(n)$. Top left graph illustrate the basic interpolation structure. Following graphs illustrate the steps to the polyphase implementation (bottom right graph)

Now assume the interpolation function given by (9.17). With the polyphase decomposition of $H_I(z) = \sum_{l=0}^{L-1} z^{-l} H_l(z^L)$ we have the identity

$$\begin{aligned} x_I(n) = H_I(z) \Uparrow_L [x(n')] &= \sum_{l=0}^{L-1} z^{-l} H_l(z^L) \Uparrow_L [x(n')] \\ &= \sum_{l=0}^{L-1} z^{-l} \Uparrow_L [H_l(z) x(n')]. \end{aligned} \tag{9.25}$$

All the L filters $H_l(z)$ are operated at the lower rate and are each of size P. The total complexity of the filtering is thus reduced by a factor L compared with a straightforward implementation according to (9.17), i.e. up-sample first and then filter. For the decimation application a dual approach can be applied in a similar manner. Graphical illustration of the polyphase identities for a factor 3 decimator can be seen in Figure 9.1. In Figure 9.2 a factor 3 interpolation is illustrated.

> **Example 8** Assume the LP filter $H(z)$ to be used in a factor 3 interpolation (like in the Figure 9.2) has impulse response $h_0, h_1, \ldots, h_5$. We can write
>
> $$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + \ldots + h_5 z^{-5} \tag{9.26}$$
>
> Since we want to use this filter for a factor 3 interpolator we want to split the filter into 3 parts according to (9.24). We obtain
>
> $$H(z) = H_0(z^3) + z^{-1} H_1(z^3) + z^{-2} H_2(z^3) \tag{9.27}$$

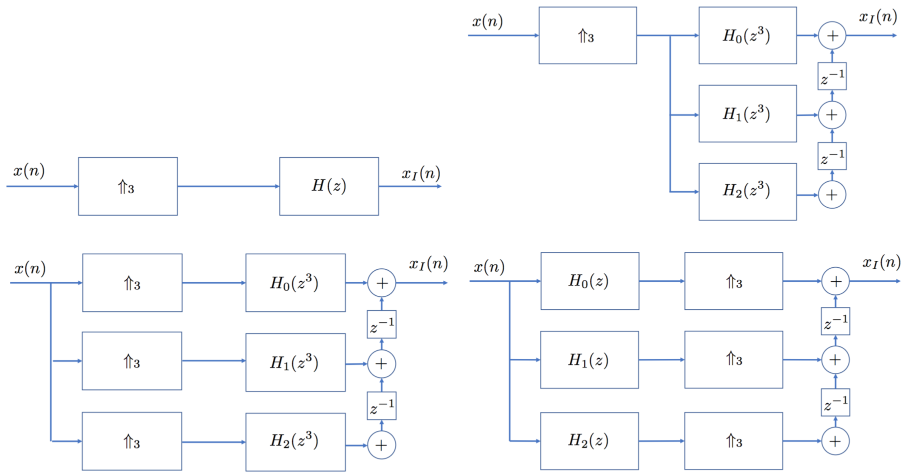


Figure 9.2: Example of processing structure for factor 3 interpolation of signal $x(n)$. Top left graph illustrate the basic interpolation structure. Following graphs illustrate the steps to the polyphase implementation (bottom right graph)

where

$$\begin{aligned} H_0(z^3) &= h_0 + h_3 z^{-3} && \Rightarrow H_0(z) = h_0 + h_3 z^{-1} \\ H_1(z^3) &= h_1 + h_4 z^{-3} && \Rightarrow H_1(z) = h_1 + h_4 z^{-1} \\ H_2(z^3) &= h_2 + h_5 z^{-3} && \Rightarrow H_2(z) = h_2 + h_5 z^{-1} \end{aligned} \qquad (9.28)$$

The orignal LP filter of length 6 has now been split into three FIR filters of length 2. Each of the 3 filters are employed at the low rate as illustrated in the bottom right graph in Figure 9.2. □

Oversampling techniques

Oversampling techniques can be used both when sampling continuous time signals as well as for reconstruction of analog signals from the digital samples. In both cases the oversampling, i.e. to increase the sample rate beyond the Nyquist rate dictated by the bandwidth of the signal, enable to perform the necessary anti-aliasing and reconstruction filtering in the digital domain instead of in the analog domain. Often this leads to inexpensive hardware solutions as long as the higher sample rates are possible to accomodate in the digital hardware.

Sampling When sampling a continuous signal alias distortion will occur if the signal sampled does not satisfy the Nyquist criteria. Higher frequency signal content will mix with lower frequency content according to (2.31). The classical way to mitigate alias distortion is to use an Anti-Aliasing Filter (AAF) prior to the sampling. For an illustration see 9.3 where the sample frequency f_s is selected to satisfy the Nyquist criteria for the desired signal. If the sampling unit is operated at a higher sample rate Lf_s, the aliasing distortion is reduced since most disturbances decrease in power with increasing frequency. The sample rate

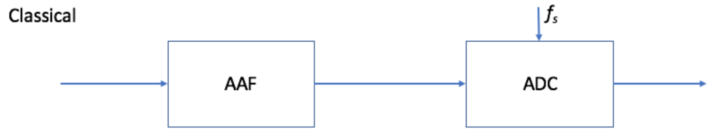


Figure 9.3: Classical processing chain including analog anti-aliasing filter (AAF) followed by sampling.

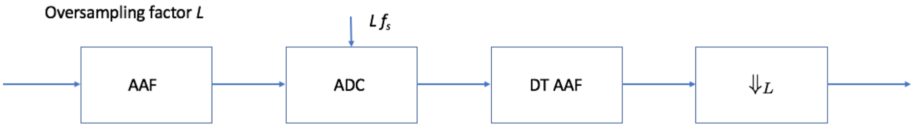


Figure 9.4: Oversampled data acquisition processing chain including analog anti-alias filter, sampling unit (ADC), discrete time anti-aliasing (DT AAF) filter and downsampling.

can then be reduced by a decimation step where the aliasing distortions can be controlled by the discrete time anti-aliasing filter. It is easier to implement a high performing discrete time filter compared to an analog continuous time filter. Figure 9.4 illustrates the oversampling processing steps.

Reconstruction Classical reconstruction employ a zero-order-hold (ZOH) circuit followed by a continuous time reconstruction filter, see Figure 9.5. The ZOH reconstruction method introduces reconstruction distortion which can be illustrated by the Fourier transform of the ZOH operation. The desired behaviour is an ideal LP filter. See Figure 3.1. The ZOH operation scales the magnitude of the transform of desired signal and allow higher frequencies due to the ripple behavior. If the sample rate increased to Lf_s the forst Zero location of the ZOH frequency function will move to Lf_s. This will make the frequency function more ideal in the frequency interval $[-f_s/2, f_s/2]$ and push the frequency distortions from the ripple to higher frequencies. An oversampled processing chain for signal reconstruction is illustrated in Figure 9.6. Since most physical system attenuates high frequencies due to inertia and natural damping, the analog reconstruction filter is often not necessary if L is selected large. This significantly reduces the cost of implementation.

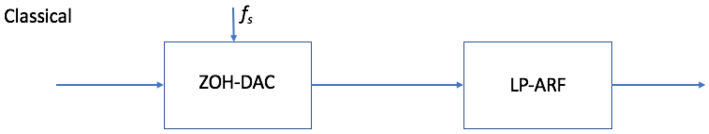


Figure 9.5: Classical reconstruction processing chain with a zero-order-hold circuit followed by a analog reconstruction filter.

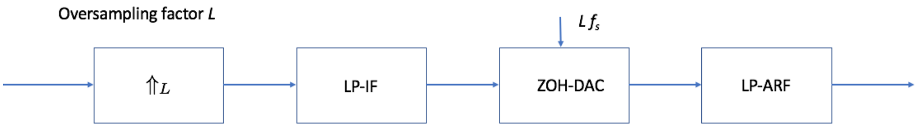


Figure 9.6: Oversampled reconstruction processing chain with an interpolator, ZOH circuit operating at Lf_s sampling frequency followed by an analog reconstruction filter.

Chapter 10

Statistical signal processing

A statistical view of signals provide a mathematical framework for analysis and optimal design. We limit the presentation to *real valued* stochastic signals.

For a random variabel x, the *cumulative distribution function* $F_x(x_0)$ give the probability of the event that the random variable x take a value less than x_0.

$$P(x < x_0) = F_x(x_0) \tag{10.1}$$

The probability of an outcome of the random variable x in the interval between x_l and x_h is

$$P(x_l < x < x_h) = F_x(x_h) - F_x(x_l) = \int_{x_l}^{x_h} p_x(x)\, dx \tag{10.2}$$

where p_x is the $p_x(x)$ is the *probability density function* where

$$\frac{d}{dx} F_x(x) = p_x(x) \tag{10.3}$$

From (10.2) it follows that $p_x(x) \geq 0$ and $\lim_{l \to \infty} P(-l < x < l) = 1$ so the area under $p_x(x)$ is equal to one.

Expectation of a function of the random variabel x, $f(x)$ is an operation defined as

$$\mathbf{E}\{f(x)\} = \int_{-\infty}^{\infty} f(x) p_x(x)\, dx \tag{10.4}$$

The mean value of a random variable x is obtained when $f(x) = x$, i.e.

$$m_x \triangleq \mathbf{E}\{x\} \triangleq \int_{-\infty}^{\infty} x p_x(x)\, dx \tag{10.5}$$

If the pdf of a random variable x is (even) symmetric around a value x_0 then $m_x = x_0$. A multidimensional pdf provide the information on how several variables are statistically related. Say x and y are two variables with a joint pdf $p_{x,y}(x, y)$. The pdf then gives the probability of the event that the two variables take values which fall inside the box defined by x_l, x_h, y_l and y_h;

$$P(x_l < x < x_h, y_l < y < y_h) = \int_{y_l}^{y_h} \int_{x_l}^{x_h} p_{x,y}(x, y)\, dxdy. \tag{10.6}$$

Expectation for two variables are defined as

$$\mathbf{E}\{f(x,y)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)p_{x,y}(x,y)\,dxdy. \tag{10.7}$$

If the joint probability distribution can be factored as $p_{x,y}(x,y) = p_x(x)p_y(y)$, the two variables are called *independent* and

$$\begin{aligned}\mathbf{E}\{xy\} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} xyp_{x,y}(x,y)\,dxdy = \int_{-\infty}^{\infty} xp_x(x)\,dx \int_{-\infty}^{\infty} yp_y(y)\,dy. \\ &= \mathbf{E}\{x\}\,\mathbf{E}\{y\}\end{aligned} \tag{10.8}$$

From this results we note that if two variables x and y are independent and at least one of them have zero mean value then $\mathbf{E}\{xy\} = 0$.

The *variance* σ_x^2 of a stochastic variable x is

$$\sigma_x^2 = \mathbf{E}(x - m_x)^2 = \mathbf{E}\,x^2 - 2m_x\,\mathbf{E}\,x + m_x^2 = \mathbf{E}\,x^2 - m_x^2. \tag{10.9}$$

If the stochastic variable is zero mean ($m_x = \mathbf{E}\,x = 0$) then the variance is the expected value of x^2. The square root of the variance $\sigma_x = \sqrt{\sigma_x^2}$ is called the standard deviation. The variance of a zero mean signal is proportional to the power of the signal while the standard deviation is proportional to the magnitude of the signal.

Stochastic processes $x(n)$

If we consider a sequence of stochastic variables and enumerate them with an integer index we can regard them as a stochastic process $x(n)$. Hence a discrete time signal can be viewed as a realization (outcome) of a stochastic process. In our notation we will not differentiate between the stochastic variable and the numerical outcome or realization of it. Hence $x(n)$ can both represent the stochastic variable or the numerical value the signal x takes at time index n. The context in which the symbol is used will most often clarify which interpretation is the intended one.

The statistical relation between different samples (random variables) are described by the joint pdf, $p_{x(n),x(n+k)}$. Here we only regard process which are *wide sense stationary* (WSS) which means that the mean value (first moment) is constant and the covariance (second moment) only depend on the time differences (the time lag) and is finite. Hence, for a WSS process $\mathbf{E}\,x(n) = m_x$ for all n, i.e. the mean value does not depend in the index n. For all n and k we have $\mathbf{E}\,x(n)x(n+k) = \mathbf{E}\,x(0)x(k)$. We also assume that the process is *wide sense ergodic* which mean that for WSS process $x(n)$ the sample means will converge to the expected value for the first and second moments, i.e.

$$\mathbf{E}\,x(n) = m_x = \lim_{M\to\infty} \frac{1}{2M+1} \sum_{n=-M}^{M} x(n) \tag{10.10}$$

$$\begin{aligned}\mathbf{E}\{x(n)x(n+k)\} &= \mathbf{E}\,x(0)x(k) \\ &= \lim_{M\to\infty} \frac{1}{2M+1} \sum_{n=-M}^{M} (x(n)x(n+k))\end{aligned} \tag{10.11}$$

where $x(n)$ in the summation represents the realization (outcome). Hence, for a finite M, the summation can be calculated for samples of a given signal and can be used to derive an *estimate* of the expected value. If a process $x(n)$ is *stationary* then

$$p_{x(n)x(n+k)} = p_{x(0)x(k)} \quad \text{for all } n \text{ and } k \tag{10.12}$$

if a process is stationary it then implies that it is also WSS. The converse is not always true.

The *auto-correlation function* is defined as

$$\phi_{xx}(n) \triangleq \mathbf{E}\{(x(0) - m_x)(x(n) - m_x)\} \tag{10.13}$$

and (trivially) satisfy $\phi_{xx}(-n) = \phi_{xx}(n)$, i.e. it is an even symmetric function. The auto-correlation function illustrates how samples at n lags (time indices) apart co-vary. The auto-correlation at lag 0, $\phi_{xx}(0)$ is the same as the variance. Notice that the definition in (10.13) in statistical literature is often denoted *auto-covariance* and the auto-correlation is defined as $\phi_{xx}(n)/\phi_{xx}(0)$.

The *cross-correlation* between two signals $x(n)$ and $y(n)$ is defined by

$$\phi_{xy}(n) \triangleq \mathbf{E}\{(x(0) - m_x)(y(n) - m_y)\} \tag{10.14}$$

and we see that $\phi_{yx}(n) = \phi_{xy}(-n)$.

From here and onwards, we assume that *signals have zero mean value and are wide sense stationary and ergodic* unless explicitly stated otherwise.

Frequency domain descriptions

By using the DTFT of the auto- and cross-correlation functions we obtain a spectral domain description of the stochastic processes. The *power spectral density* is given by

$$S_{xx}(\omega) \triangleq \sum_{n=-\infty}^{\infty} \phi_{xx}(n) e^{-j\omega n}. \tag{10.15}$$

Since the auto-correlation function is even symmetric, the power spectral density $S_{xx}(\omega)$ is a real and non-negative function. Since

$$\sigma_x^2 = \phi_{xx}(0) = \frac{1}{2\pi} \int_0^{2\pi} S_{xx}(\omega)\, d\omega \tag{10.16}$$

we can make the interpretation that $S_{xx}(\omega)$ let us know how the power in the signal is distributed over the frequency interval.

The *cross-spectral density* is defined in an analogous manner

$$S_{xy}(\omega) \triangleq \sum_{n=-\infty}^{\infty} \phi_{xy}(n) e^{-j\omega n}. \tag{10.17}$$

and we can derive the relation

$$\begin{aligned} S_{yx}(\omega) = \sum_{n=-\infty}^{\infty} \phi_{yx}(n) e^{-j\omega n} = \sum_{n=-\infty}^{\infty} \phi_{xy}(-n) e^{-j\omega n} \\ \sum_{n=-\infty}^{\infty} \phi_{xy}(n) \overline{e^{-j\omega n}} = \overline{S_{xy}(\omega)} \end{aligned} \tag{10.18}$$

since $\overline{e^{j\omega n}} = e^{-j\omega n}$ and $\phi_{xy}(n)$ is real valued.

The periodogram

The periodogram of a signal was previously introduced and is defined as the magnitude of the DTFT of the signal. We will now show the relation between the power spectral density of a stochastic process and the periodogram based on a realization of length N of the process. Let $P(\omega)$ denote the periodogram. Then

$$P(\omega) = |X(\omega)|^2 = X(\omega)X^*(\omega) \tag{10.19}$$

Since $X^*(\omega)$ is the DTFT of $x(-n)$ we can interpret the result as a convolution in the time domain between the signal $x(n)$ and $x(-n)$, i.e.

$$p(n) = \sum_{k=0}^{N-1} x(k)x(k-n), \quad n = 0, \pm 1, \ldots, \pm(N-1) \tag{10.20}$$

where $p(n)$ is the inverse DTFT of the periodogram. The expected value of $p(n)$ is

$$\mathbf{E}\, p(n) = \underbrace{(N-|n|)}_{w_{\text{tri}}(n)} \phi_{xx}(n) = w_{\text{tri}}(n)\phi_{xx}(n), \quad n = 0, \pm 1, \ldots, \pm(N-1) \tag{10.21}$$

which is a windowed version of the auto-correlation function and $w_{\text{tri}}(n)$ is the triangular window function. We can conclude that

$$\mathbf{E}\, P(\omega) = \sum_{k=-(N-1)}^{N-1} w_{\text{tri}}(n)\phi_{xx}(n)e^{-j\omega\Delta tn} = \frac{1}{\omega_s}\int_0^{\omega_s} W_{\text{tri}}(\lambda)S_{xx}(\omega-\lambda)\,d\lambda. \tag{10.22}$$

The expected value of the periodogram is hence equal to the power spectral density frequency convolved with the DTFT of the triangular window.

Filtering Stochastic Processes

Based on the characterization of the stochastic processes in terms of auto- and cross-correlations and power spectral densities it is natural to derive these properties also for signals which has undergone linear filtering.

Consider a stochastic process $x(n)$ which is filtered by a linear filter with impulse response $h(k)$ and frequency function $H(\omega)$. The output of the filter $y(n)$ is also a stochastic process given by

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \tag{10.23}$$

The cross-correlation between the input and output is given by

$$\begin{aligned} \phi_{xy}(n) &= \mathbf{E}\, x(0)y(n) = \mathbf{E} \sum_{k=-\infty}^{\infty} h(k)x(0)x(n-k) \\ &= \sum_{k=-\infty}^{\infty} h(k)\phi_{xx}(n-k) \end{aligned} \tag{10.24}$$

The cross-correlation between x and y is the convolution between the impulse response of the filter and the auto-correlation of the input x. In the DTFT domain we consequently can state

$$S_{xy}(\omega) = H(\omega)S_{xx}(\omega). \tag{10.25}$$

In turn the auto-correlation of the filter output y is

$$\begin{aligned} \phi_{yy}(n) = \mathbf{E}\, y(0)y(n) &= \mathbf{E} \sum_{k=-\infty}^{\infty} h(k)y(0)x(n-k) \\ &= \sum_{k=-\infty}^{\infty} h(k)\phi_{yx}(n-k) \end{aligned} \tag{10.26}$$

which in the DTFT domain is

$$\begin{aligned} S_{yy}(\omega) &= H(\omega)S_{yx}(\omega) = H(\omega)\overline{S_{xy}(\omega)} \\ &= H(\omega)\overline{H(\omega)S_{xx}(\omega)} = |H(\omega)|^2 S_{xx}(\omega) \end{aligned} \tag{10.27}$$

where we used the result from (10.18) and (10.25) and the fact that $S_{xx}(\omega)$ is a real valued function. By analyzing (10.27) we notice that the power spectrum of the output is the product of the power spectrum of the input and the squared magnitude (amplitude) function of the linear filter. The phase function of the filter does not influence the power spectrum.

Filtered white noise

Consider a signal $e(n)$ which is a stochastic process with zero mean value and an auto-correlation function $\phi_{ee}(n) = \sigma_e^2 \delta_n$ where $\delta_n = 1$ for $n = 0$ and zero everywhere else. The power spectrum of the signal $e(n)$ is

$$S_{ee}(\omega) = \sum_{n=-\infty}^{\infty} \sigma_e^2 \delta_n e^{-j\omega n} = \sigma_e^2. \tag{10.28}$$

The power spectral density $S_{ee}(\omega)$ is constant for all frequencies and is therefore referred to as a *white* spectrum and signals with a white power spectral density are called white noise signals. Consequently, signals with a non-constant spectrum, a colored spectrum, are called *colored.*

If the white noise signal e is filtered through a filter $H(\omega)$ the output power spectral density is given by

$$S_{yy}(\omega) = |H(\omega)|^2 \sigma_e^2. \tag{10.29}$$

Since only the amplitude function matters for the resulting spectrum the phase of the filter can be arbitrary.

Minimum phase spectral factors

Assume a power spectral density $S(\omega)$ which is strictly positive is specified in the form of a filter factor, i.e. $S(\omega) = |H(\omega)|^2 = H(\omega)\overline{H(\omega)}$ and $H(\omega)$ is an FIR filter with a real valued impulse response. We can express the filter as

$H(z) = \Pi_k(z - z_k)$ where z_k are the zeros of the polynomial. The zeros z_k are either real valued or are pairwise complex conjugated. With the z-transform notation (i.e. $z = e^{j\omega}$) we can write

$$S(z) = |H(z)|^2 = H(z)\overline{H(z)} = H(z)H(z^{-1}) \tag{10.30}$$

Notice that of z_k is a zero for $H(z)$ then $1/z_k$ is a zero for $H(z^{-1})$. If we collect all zeros of the product $H(z)H(z^{-1})$ which reside inside the unit circle into $H^+(z)$ we have by construction

$$S(z) = |H(z)|^2 = H^+(z)H^+(z^{-1}) = |H^+(z)|^2 \tag{10.31}$$

and we have constructed a filter $H^+(z)$ with all zeros inside the unit circle. The spectral factor $H^+(z)$ is called minimum phase and has the important property that the causal inverse of the filter, which is if IIR type, is stable since all zeros of $H^+(z)$ will be poles in $1/H^+(z)$. We can perform a similar factorization if the specified spectrum is a fraction of two polynomials which will yield a spectral factor with both poles and zeros inside the unit circle.

Noise whitening filter

In some applications it is desirable to use a filter to make the spectrum of a stochastic signal to become white. Such filters are called whitening filters. If the colored spectrum has a rational structure at can be written using the minimum phase factor $H^+(\omega)$

$$S_{xx}(\omega) = |H^+(\omega)|^2. \tag{10.32}$$

The whitening filter is given by $H^+(\omega)^{-1}$, the inverse of the spectral factor, which is always stable.

Inverse filtering

Equalization or inverse filtering is the process where a filter is used to recover a source signal from a second signal originating from the source but altered by some linear filter $g(n)$. Assume

$$y(n) = g(n) * x(n) + w(n) \tag{10.33}$$

where $y(n)$ is the received signal $x(n)$ is the signal we want to recover and $w(n)$ is a term modeling the noise in the receiver. To recover $x(n)$ from $y(n)$ we use a linear filter $h(n)$

$$\hat{x}(n) = \sum_{k=-\infty}^{\infty} h(k)y(n-k) \tag{10.34}$$

where $\hat{x}(n)$ is the *estimate* of the original input signal $x(n)$. In the DTFT domain we have

$$\hat{X}(\omega) = H(\omega)G(\omega)X(\omega) + H(\omega)W(\omega) \tag{10.35}$$

A possible choice could be to simply select $H(\omega) = G(\omega)^{-1}$. This choice could lead to some undesirable effects:

- Firstly, if $G(z)$ have zeros outside the unit circle then $H(z)$ would be an unstable filter and not possible to use in practice.
- Secondly, for frequencies where $G(\omega)$ is close to zero the filter $H(\omega)$ would be vey large and lead to a high gain of the noise.

The so called *Wiener filter* provide a solution which provide an optimal balance between inverting G and keeping the resulting noise levels low.

Chapter 11

Optimal filtering - Wiener filter

Previously we have designed frequency filters based on specifications in the frequency domain. In this section we will consider the filter design problem from a statistical signal point of view and use optimality based on a mean-square error criterion. At first this can be interpreted as a time-domain filter design technique. However since we will base the design on correlation and cross-correlation functions the optimal filtering problem have an equally valid interpretation in the frequency domain, i.e. in terms of minimizing the power spectrum of the error.

Equalization

Consider the filtering problem where a known signal $y(n)$ contain information of a signal $x(n)$ but also contain a disturbance $w(n)$. Assume

$$y(n) = G(z)x(n) + w(n) \tag{11.1}$$

where $G(z)$ is a linear filter, $x(n)$ and $w(n)$ are zero mean stochastic processes independent of each other. This signal model works well for a communication application where $y(n)$ is the received signal and $x(n)$ is the information which is transmitted over the channel modeled by $G(z)$. For this signal model the power spectrum of $y(n)$ and the cross spectral density are given by

$$\begin{aligned} S_{yy}(\omega) &= |G(\omega)|^2 S_{xx}(\omega) + S_{ww}(\omega) \\ S_{xy}(\omega) &= G(\omega)S_{xx}(\omega) \end{aligned} \tag{11.2}$$

where we have assumed w and x to be uncorrelated, an assumption which is most natural in many applications.

Filtering, prediction and smoothing

A second signal setup which is a variation on (11.1) can be written

$$y(n) = s(n) + w(n) \tag{11.3}$$

where $s(n)$ is a signal of importance and $w(n)$ is a disturbance. Three different scenarios can be identified.

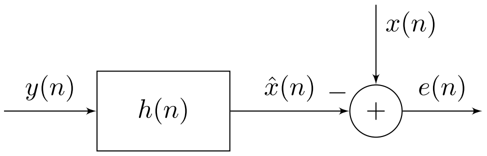


Figure 11.1: Block diagram for the general optimal filtering problem.

Filtering If we want to make the best causal estimate of the signal $s(n)$ from $y(n)$ we set the desired signal $x(n) = s(n)$.

Prediction If we want to use the signal $y(n)$ to produce estimates of future values of $s(n)$ we set $x(n) = s(n+k)$ for some positive prediction horizon k.

Smoothing If the objective is to obtain a good estimate of past values of $s(n)$ based on the observed values $y(n)$ we set $x(n) = s(n-k)$ for some positive smoothing lag k.

Optimal non-causal Wiener filter

The filtering objective for all cases listed above is to recover a good approximation of x by filtering the received signal y. Consider the general case, as illustrated in Figure 11.1, when the filter $H(\omega)$ has a double infinite length impulse response, i.e. it is a non-causal filter. Then the filter equation is

$$\hat{x}(n) = \sum_{k=-\infty}^{\infty} h(k)y(n-k), \tag{11.4}$$

where $\hat{x}$ is the estimate of the desired signal x. The mean squared error (MSE) is given by

$$\begin{aligned} \mathbf{E}\, e^2(n) &= \mathbf{E}(x(n) - \hat{x}(n))^2 = \mathbf{E}(x(n) - \sum_{k=-\infty}^{\infty} h(k)y(n-k))^2 \\ &= \mathbf{E}\left[x^2(n) - 2\sum_{k=-\infty}^{\infty} h(k)y(n-k)x(n) + \sum_{k=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} h(k)h(r)y(n-k)y(n-r) \right] \\ &= \phi_{xx}(0) - 2\sum_{k=-\infty}^{\infty} h(k)\phi_{yx}(k) + \sum_{k=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} h(k)h(r)\phi_{yy}(k-r) \end{aligned} \tag{11.5}$$

The gradient of the MSE w.r.t the filter coefficients are given by

$$\frac{d}{dh(k)}\,\mathbf{E}\, e^2(n) = -2\phi_{yx}(k) + 2\sum_{r=-\infty}^{\infty} h(r)\phi_{yy}(k-r) \tag{11.6}$$

When the filter is optimal the MSE is minimized and hence the gradient w.r.t. each of the filter coefficients must be zero. At the optimum, for each k we have

$$\phi_{yx}(k) = \sum_{r=-\infty}^{\infty} h(r)\phi_{yy}(k-r) \tag{11.7}$$

which gives an implicit characterization of the optimal filter. However, employing the DTFT on both sides of the equation and rearranging we obtain

$$H(\omega) = \frac{S_{yx}(\omega)}{S_{yy}(\omega)} = \frac{\overline{S_{xy}(\omega)}}{S_{yy}(\omega)} \tag{11.8}$$

Returning to the communication example before, e.g. (11.1), where a signal $x(n)$ should be recovered from noisy measurements $y(n)$. For this case the power spectrum and cross spectrum are detailed in (11.2). Hence, the filter which minimizes the MSE of the estimation error is given by

$$H(\omega) = \frac{\overline{G(\omega)}S_{xx}(\omega)}{|G(\omega)|^2 S_{xx}(\omega) + S_{ww}(\omega)} = \frac{1}{G(\omega)} \times \frac{1}{1 + \frac{S_{ww}(\omega)}{|G(\omega)|^2 S_{xx}(\omega)}}. \tag{11.9}$$

where the right hand side is only valid if $|G(\omega)| > 0$ for all ω. If we define $\mathrm{SNR}(\omega) \triangleq \frac{|G(\omega)|^2 S_{xx}(\omega)}{S_{ww}(\omega)}$ as the signal to noise ratio, we can also describe the optimal filter as

$$H(\omega) = \frac{1}{G(\omega)} \times \frac{\mathrm{SNR}(\omega)}{\mathrm{SNR}(\omega) + 1} \tag{11.10}$$

Notice that if the signal to noise ratio tends to infinity (for all frequencies) the optimal filter is given by $H(\omega) = 1/G(\omega)$, i.e. the optimal filter is the inverse of the transmission channel. On the other side of the scale, i.e. when $\mathrm{SNR}(\omega)$ approaches zero, the filter gain tends to zero. In conclusion (11.9) provide the optimal tradeoff between mitigating effects of noise and approximately inverting the transmission channel.

Optimal causal FIR Wiener filter

For many practical applications it is sufficient to use a finite FIR filter structure and the optimal filter refers to best causal FIR filter of a given length. Hence, the filter equation is simplified to

$$\hat{x}(n) = \sum_{k=0}^{M-1} h(k) y(n-k). \tag{11.11}$$

The derivation of the optimal filter of finite length follows along the lines above with the exemption that the summations are finite. The gradient of the MSE w.r.t. to $h(k)$ for $k = 0, 1, \ldots, M-1$ is given by

$$\frac{d}{dh(k)} \mathbf{E}\, e^2(n) = -2\phi_{yx}(k) + 2\sum_{r=0}^{M-1} h(r)\phi_{yy}(k-r) \tag{11.12}$$

and the optimal filter is defined by, for all k, setting all gradients to zero

$$\phi_{yx}(k) = \sum_{r=0}^{M-1} h(r)\phi_{yy}(k-r) \tag{11.13}$$

Define the matrix

$$\Phi_{yy} \triangleq \begin{bmatrix} \phi_{yy}(0) & \phi_{yy}(1) & \cdots & \phi_{yy}(M-1) \\ \phi_{yy}(1) & \phi_{yy}(0) & \cdots & \phi_{yy}(M-2) \\ \vdots & & & \\ \phi_{yy}(M-1) & \phi_{yy}(M-2) & \cdots & \phi_{yy}(0) \end{bmatrix}. \tag{11.14}$$

All elements on the diagonals in this square matrix have the same value. Such a matrix is known as a Toeplitz matrix. Also introduce the vectors

$$\Phi_{yx} = \begin{bmatrix}\phi_{yx}(0) & \phi_{yx}(1) & \cdots & \phi_{yx}(M-1)\end{bmatrix}^T \tag{11.15}$$

and

$$\mathbf{h} = \begin{bmatrix}h(0) & h(1) & \cdots & h(M-1)\end{bmatrix}^T \tag{11.16}$$

With the matrix and vector notation we can rewrite (11.13) as

$$\Phi_{yx} = \Phi_{yy}\mathbf{h} \tag{11.17}$$

where we have used the property that $\phi_{yy}(-k) = \phi_{yy}(k)$. All solutions $\mathbf{h}$ to equation (11.17) is an optimal causal FIR filter of order M. Furthermore, if Φ_{yy} has full rank the solution is unique and given by

$$\mathbf{h}_{\text{opt}} = \Phi_{yy}^{-1}\Phi_{yx}. \tag{11.18}$$

if we define

$$\mathbf{y} \triangleq \begin{bmatrix}y(n) & y(n-1) & \cdots & y(n-M+1)\end{bmatrix}^T \tag{11.19}$$

we obtain $\Phi_{yy} = \mathbf{E}\{\mathbf{y}\mathbf{y}^T\}$ and $\Phi_{yx} = \mathbf{E}\{\mathbf{y}x(n)\}$. With this notation we can derive the resulting MSE when using the optimal filter as

$$\begin{aligned}\mathbf{E}\,e^2(n) &= \mathbf{E}(x(n)-\hat{x}(n))^2 = \mathbf{E}(x(n)-\mathbf{h}^T\mathbf{y})^2\\ &= \mathbf{E}\left[x^2(n) - 2x(n)\mathbf{h}^T\mathbf{y} + (\mathbf{h}^T\mathbf{y})(\mathbf{y}^T\mathbf{h})\right]\\ &= \phi_{xx}(0) - 2\mathbf{h}^T\Phi_{yx} + \mathbf{h}^T\Phi_{yy}\mathbf{h}\\ &= \phi_{xx}(0) - 2\mathbf{h}^T\Phi_{yx} + \mathbf{h}^T\Phi_{yy}\Phi_{yy}^{-1}\Phi_{yx}\\ &= \phi_{xx}(0) - \mathbf{h}^T\Phi_{yx} = \phi_{xx}(0) - \Phi_{yx}^T\Phi_{yy}^{-1}\Phi_{yx}\end{aligned} \tag{11.20}$$

Finding the optimal filter from data

Many times in practice it might be difficult to obtain the auto-correlation and cross-correlation functions needed to derive the optimal Wiener filter. Secondly, in many applications we are faced with environments which slowly varies over time and we would like to track theses changes automatically. Both these issues are solved by adaptive filter algorithms. In principle we will consider:

- *The estimation problem:* How to derive the optimal filter based on measured data.
- *The adaption problem:* How to adaptively update the filter when the signal properties changes over time.

First we consider the estimation problem where we assume we have access to training samples of both $y(n)$ and $x(n)$. If we consider a sample based criterion as an approximation of the variance of $e(n)$ we have

$$\mathbf{E}\{e^2(n)\} \approx L_N(\mathbf{h}) \triangleq \frac{1}{N+1}\sum_{n=0}^{N} e^2(n) = \frac{1}{N+1}\sum_{n=0}^{N}\big(x(n)-\mathbf{h}^T\mathbf{y}(n)\big)^2 \tag{11.21}$$

where

$$\begin{aligned}\mathbf{y}(n) &= \begin{bmatrix} y(n) & y(n-1) & \cdots & y(n-M+1)\end{bmatrix}^T \\ \mathbf{h} &= \begin{bmatrix} h(0) & h(1) & \cdots h(M-1)\end{bmatrix}^T\end{aligned} \tag{11.22}$$

The criterion function L_N is a non-negative quadratic function and hence the minimum is attained when the gradient w.r.t. $\mathbf{h}$ is zero

$$\frac{d}{d\mathbf{h}}L_N(\mathbf{h}) = 0 \quad \Rightarrow \quad \frac{1}{N+1}\sum_{n=0}^{N}\mathbf{y}(n)\mathbf{y}^T(n)\mathbf{h} = \frac{1}{N+1}\sum_{n=0}^{N}\mathbf{y}(n)x(n) \tag{11.23}$$

Introducing the notation

$$\mathbf{R}_{yy}(N) \triangleq \sum_{n=0}^{N}\mathbf{y}(n)\mathbf{y}^T(n), \quad \mathbf{R}_{yx}(N) \triangleq \sum_{n=0}^{N}\mathbf{y}(n)x(n) \tag{11.24}$$

we note that the filters minimizing L_N should satisfy

$$\mathbf{R}_{yy}(N)\mathbf{h} = \mathbf{R}_{yx}(N) \tag{11.25}$$

and if $\mathbf{R}_{yy}(N)$ has full rank the solution is unique and can be written as

$$\hat{\mathbf{h}}(N) = \mathbf{R}_{yy}^{-1}(N)\mathbf{R}_{yx}(N). \tag{11.26}$$

If the signals $y(n)$ and $x(n)$ are ergodic processes we have that

$$\lim_{N\to\infty}\frac{1}{N+1}\mathbf{R}_{yy}(N) = \Phi_{yy}, \quad \lim_{N\to\infty}\frac{1}{N+1}\mathbf{R}_{yx}(N) = \Phi_{yx} \tag{11.27}$$

and, if for some N_0, $\mathbf{R}_{yy}(N)$ is non-singular for all $N > N_0$ we obtain

$$\lim_{N\to\infty}\hat{\mathbf{h}}(N) = \Phi_{yy}^{-1}\Phi_{yx} = \mathbf{h}_{\text{opt}} \tag{11.28}$$

Recursive least-squares (RLS) algorithm

For on-line applications it is desirable to be able to re-calculate the optimal filter base on new data. Clearly this can be directly be accomplished by simply use relations (11.24) and (11.26). However this would involve, for each time sample, calculating a matrix inverse of a matrix of size $M \times M$ which is computationally quite costly (order M^3 number of multiplications). However, the least-squares solution in (11.26) can be formulated as a recursion without calculation of the full inverse which saves computations.

Assume we have already calculated $\mathbf{R}_{yy}^{-1}(N-1)$ and $\hat{\mathbf{h}}(N-1)$. The filter output based on the filter $\hat{\mathbf{h}}(N-1)$ is

$$\hat{x}(N) = \hat{\mathbf{h}}^T(N-1)\mathbf{y}(N), \quad \text{with error} \quad e(N) = x(N) - \hat{x}(N). \tag{11.29}$$

We will now derive a recursion which provides us with an update of these two entities based on the new data $x(N)$ and $\mathbf{y}(N)$. Clearly from the definition in (11.24) we have

$$\mathbf{R}_{yy}(N) = \mathbf{R}_{yy}(N-1) + \mathbf{y}(N)\mathbf{y}^T(N) \tag{11.30}$$

and

$$\mathbf{R}_{yx}(N) = \mathbf{R}_{yx}(N-1) + \mathbf{y}(N)x(N) \tag{11.31}$$

Using relation (11.26) on the left hand side in (11.31) and relation (11.30) on the right hand side we obtain

$$\begin{aligned}\mathbf{R}_{yy}(N)\hat{\mathbf{h}}(N) &= \mathbf{R}_{yy}(N-1)\hat{\mathbf{h}}(N-1) + \mathbf{y}(N)x(N) \\ &= \left(\mathbf{R}_{yy}(N) - \mathbf{y}(N)\mathbf{y}^T(N)\right)\hat{\mathbf{h}}(N-1) + \mathbf{y}(N)x(N)\end{aligned} \tag{11.32}$$

Furthermore expanding the parenthesis yields

$$\begin{aligned}&\mathbf{R}_{yy}(N)\hat{\mathbf{h}}(N) \\ &\quad = \mathbf{R}_{yy}(N)\hat{\mathbf{h}}(N-1) - \mathbf{y}(N)\underbrace{\mathbf{y}^T(N)\hat{\mathbf{h}}(N-1)}_{\hat{x}(N)} + \mathbf{y}(N)x(N) \\ &\quad = \mathbf{R}_{yy}(N)\hat{\mathbf{h}}(N-1) + \mathbf{y}(N)\underbrace{\left(x(N) - \hat{x}(N)\right)}_{e(N)} \\ &\quad = \mathbf{R}_{yy}(N)\hat{\mathbf{h}}(N-1) + \mathbf{y}(N)e(N)\end{aligned} \tag{11.33}$$

Assuming $\mathbf{R}_{yy}(N)$ non-singular we obtain the recursion for the filter as

$$\hat{\mathbf{h}}(N) = \hat{\mathbf{h}}(N-1) + \mathbf{R}_{yy}^{-1}(N)\mathbf{y}(N)e(N) \tag{11.34}$$

The last issue to fix is to provide a technique to calculate the inverse of $R_{yy}(N)$ in an efficient way. To do so we will use the following matrix result. Consider the matrix A defined as

$$A = B + vv^T \tag{11.35}$$

where v is a column vector and B is a symmetric matrix. If B^{-1} exists and $1 + v^T B^{-1} v \neq 0$ then the inverse of A is given as

$$A^{-1} = B^{-1} - \frac{B^{-1}vv^T B^{-1}}{1 + v^T B^{-1} v} \tag{11.36}$$

and is a special case of the Sherman-Morrison-Woodbury formula. The update (11.35) is known as a matrix rank one update (since the vector outer product vv^T is a matrix with rank one). Since the formation of $\mathbf{R}_{yy}(N)$ in (11.30) is a rank one update, the inverse of $\mathbf{R}_{yy}(N)$ can be formulated as the recursion:

$$\mathbf{R}_{yy}^{-1}(N) = \mathbf{R}_{yy}^{-1}(N-1) - \frac{\mathbf{R}_{yy}^{-1}(N-1)\mathbf{y}(N)\mathbf{y}^T(N)\mathbf{R}_{yy}^{-1}(N-1)}{1 + \mathbf{y}^T(N)\mathbf{R}_{yy}^{-1}(N-1)\mathbf{y}(N)} \tag{11.37}$$

Note that the update in (11.37) only requires one matrix vector multiplication ($\mathbf{R}_{yy}^{-1}(N-1)\mathbf{y}(N)$) and one vector outer product. The computational complexity is hence proportional to M^2. The RLS algorithm is composed of the two recursions (11.37) and (11.34). The recursions must be initialized properly to give the true least-squares solution for each value of N. This can be accomplished by selecting an integer N_0 such that $\mathbf{R}_{yy}(N_0)$ is non-singular and then derive the least-squares solution

$$\hat{\mathbf{h}}(N_0) = \mathbf{R}_{yy}^{-1}(N_0)\mathbf{R}_{yx}(N_0) \tag{11.38}$$

and start the recursion from these values for $N > N_0$.

The RLS algorithm above will deliver the optimal estimate minimizing $\sum_{n=0}^{N} e^2(n)$, i.e. all errors are weighted equally. To obtain an algorithm which adaptively changes the filter when the conditions changes we can impose a time dependent weighting of the functional being minimized. If an exponential weighing is used

$$L_N(\mathbf{h}) = \sum_{n=0}^{N} e^2(n)\alpha^{N-n} \tag{11.39}$$

errors for time indices $n << N$ will hava a low weight if α is selected in the range

$$0 < \alpha < 1. \tag{11.40}$$

The weighting α is called the *forgetting factor* (when (11.40) is satisfied) and is commonly selected to be in the range of $[0.95, 0.999]$ depending on the application. A value close to 1 will yield an algorithm which adapts to changes slowly but is robust against measurement noise while a smaller value will lead to a fast adaptation to changes but with a higher sensitivity to noise.

It is interesting to note that the recursive representation of the least-squares solution is particularly well suited for the RLS algorithm with a forgetting factor. The only equation which need to be modified is the update of the matrix inverse in (11.37).

$$\mathbf{R}_{yy}^{-1}(N) = \frac{1}{\alpha}\left[\mathbf{R}_{yy}^{-1}(N-1) - \frac{\mathbf{R}_{yy}^{-1}(N-1)\mathbf{y}(N)\mathbf{y}^T(N)\mathbf{R}_{yy}^{-1}(N-1)}{\alpha + \mathbf{y}^T(N)\mathbf{R}_{yy}^{-1}(N-1)\mathbf{y}(N)}\right] \tag{11.41}$$

When using RLS with a forgetting factor it is natural to use a simplified initialization procedure for the recursions. Common choices are to select $\mathbf{R}_{yy}^{-1}(0) = \beta\mathbf{I}$ and $\hat{\mathbf{h}}(0) = 0$ with $\beta > 0$. The magnitude of β will influence the algorithm initial convergence but will not make any difference when the algorithm has reached a stationary operation.

Chapter 12

Kalman Filtering

Introduction

Kalman filtering [3, 4] is a commonly used technique for model based signal processing and estimation since it was presented around 1960. The technique is named after one of the key inventors Rudolf Kalman. Kalman filtering is part of the class of linear filtering techniques which are based on time domain design criteria.

First consider a simple filtering example to introduce the basic concept of Kalman filtering. Assume an airplane is moving along a straight line. The position of the airplane in some coordinate system is $p_c(t)$ [m]. The speed of the airplane is $s_c(t) = \frac{d}{dt}p_c(t)$ [m/s]. A discrete time model of the movement of the airplane is then

$$\begin{aligned} p(k+1) &= p(k) + \Delta t s(k) \\ s(k+1) &= s(k) \end{aligned} \tag{12.1}$$

where Δt is the sampling interval and $s(k) \triangleq s_c(k\Delta t)$ and $p(k) \triangleq p_c(k\Delta t)$ are the sampled speed and position respectively. In the model above the speed is assumed to be constant. This assumption is in practice not realistic. The pilot will change the speed of the airplane over time and the the airplane could be subject to turbulence etc. These changes can also be incorporated into the model by adding an extra input $w_1(k)$ and we obtain

$$\begin{aligned} p(k+1) &= p(k) + \Delta t s(k) \\ s(k+1) &= s(k) + w_1(k). \end{aligned} \tag{12.2}$$

It is natural to regard the extra input $w_1(k)$ as small perturbations which over time has a zero average. The model above is called a constant speed model and is a model of how the speed and position of the airplane changes over time.

A radar station can measure the position $p(n)$ of the airplane. However all measurements are subject to inaccuracies so a natural model of the measurement is

$$y(k) = p(k) + v(k) \tag{12.3}$$

where $v(k)$ is the measurement noise which we also assume is zero on average.

An air traffic controller need to know the speed and position of all airplanes in the airspace so the core signal processing problem is how to provide him with the best estimates of the position and speed of the airplane given only noisy measurements of the position obtained from the radar sensor.

In principle there are two sources of information involved in this estimation problem.

1. The equations which describes the movement of the airplane assuming constant speed (12.1)
2. The measurement of the position of the airplane (12.3)

A Kalman filter is a linear filtering algorithm which in an optimal way combines these two sources of information. It provides an estimate of both the speed and position of the airplane. To derive the Kalman filtering equations we will use a probabilistic view of the problem using multivariate random variables.

Some results on multivariate normal random variables

An n-dimensional real valued multivariate random variable Z with a normal distribution has the probability density function (PDF)

$$p_Z(z) = \frac{1}{\sqrt{(2\pi)^n \det Q}} \exp\left(-\frac{1}{2}(z-\mu)^T Q^{-1}(z-\mu)\right) \tag{12.4}$$

where $\mathbf{E}\, Z = \mu \in \mathbb{R}^n$ is the mean value and $\mathbf{E}(Z-\mu)(Z-\mu)^T = Q \in \mathbb{R}^{n\times n}$ is the positive definite covariance matrix. Often the multivariate normal distribution is called a multivariate Gaussian distribution. The distribution is uniquely defined by the mean value vector and the covariance matrix. Hence we use the notation

$$Z \sim N(\mu, Q) \tag{12.5}$$

to denote that the random variable Z has a normal distribution with mean value μ and covariance matrix Q.

An important property for random variables with a normal distribution is that the scaled sum of such variables also has a normal distribution. More specifically if the variables $Z_i \sim N(\mu_i, Q_i)$ are statistically independent and normally distributed then

$$\sum_i A_i Z_i \sim N\left(\sum_i A_i \mu_i, \sum_i A_i Q_i A_i^T\right) \tag{12.6}$$

where A_i are a scaling matrices such that $\sum_i A_i Q_i A_i^T$ is positive definite.

In Kalman filtering the conditional distribution plays a key role. This is connected with the case when we study a multivariate random variable and have exact knowledge about the outcome of some components of it. This knowledge can be used to infer more information about the remaining, unobserved, parts of the random vector. Particularly we seek the PDF of the unobserved part of the random vector. This is called the conditional distribution as it is a distribution which is conditioned on a specific numerical outcome of parts of the random variable. Assume variables X and Y have a joint PDF $p_{X,Y}(x,y)$. The distribution of X given that we know the outcome of $Y = y$ is the conditional distribution

$$p_{X|Y}(x|Y=y) = \frac{p_{X,Y}(x,y)}{p_Y(y)}. \tag{12.7}$$

Note that (12.7) is the PDF for X conditioned on the observed outcome $Y = y$, where y is the numerical outcome. For normally distributed variables we have the following properties.

Assume we partition a multivariate normally distributed random random variable into two parts $Z = \left[\begin{smallmatrix} X \\ Y \end{smallmatrix} \right]$ and write

$$Z = \begin{bmatrix} X \\ Y \end{bmatrix} \sim N \left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix} \right). \tag{12.8}$$

Then the conditional distribution of X given that we know $Y = y$ is also a normal distribution

$$p_{X|Y}(x|Y = y) \sim N \left(\mu_x + Q_{12} Q_{22}^{-1} (y - \mu_y), Q_{11} - Q_{12} Q_{22}^{-1} Q_{12}^T \right) \tag{12.9}$$

A proof of the result is given in Appendix A.1. The result can interpreted as follows. Before the outcome of $Y = y$ was observed, the variable X had a mean value of μ_x and a variance of Q_{11}. The observation of $Y = y$ provides partial knowledge about X. The mean value is changed to $\mu_x + Q_{12} Q_{22}^{-1} (y - \mu_y)$ and the variance decreases to $Q_{11} - Q_{12} Q_{22}^{-1} Q_{12}^T$. It is easy to see that Q_{12}, the cross-covariance between X and Y, control the amount of information the knowledge of the outcome $Y = y$ have regarding the variable X. For example if Q_{12} would be zero (which means that X and Y are uncorrelated and independent), then the conditional distribution is identical to the distribution of X only, i.e. the outcome of the variable Y would then carry *no* information regarding the variable X.

The Kalman Filter

First a data model is described and in the following section the filtering equations are derived based on on a probabilistic view where it is assumed the variables are described by multivariate normal distributions.

The data model

A linear discrete time stochastic process of order n can be written as a multidimensional first order difference equation

$$\begin{aligned} x(k+1) &= Ax(k) + w(k) \\ y(k) &= Cx(k) + v(k) \end{aligned} \tag{12.10}$$

where $x(k) \in \mathbb{R}^n$ is the state, $A \in \mathbb{R}^{n \times n}$ is the state-transition matrix, $y(k) \in \mathbb{R}^p$ is the measurement and $C \in \mathbb{R}^{p \times n}$ the measurement matrix. The index k normally refers to time samples but in general can refer to other units, e.g. distance samples. The variables $w(k)$ and $v(k)$ are the state noise and measurement noise respectively. We assume the state and measurement noises to be zero mean random variables with a normal distribution and independent of the state variable $x(k)$.

$$\begin{bmatrix} w(k) \\ v(k) \end{bmatrix} \sim N \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \right) \tag{12.11}$$

By employing a probabilistic view also for the state $x(k)$ and assume

$$x(k) \sim N\left(\hat{x}_k, P_k\right). \tag{12.12}$$

conditioned on the measurements up to $k-1$, where $\hat{x}_k$ is the mean value and P_k is the covariance matrix. Now considering the joint distribution of $x(k)$ and $y(k)$ and using (12.6), (12.10), (12.11) and (12.12) we obtain

$$\begin{bmatrix} x(k) \\ y(k) \end{bmatrix} \sim N\left(\begin{bmatrix} \hat{x}_k \\ C\hat{x}_k \end{bmatrix}, \begin{bmatrix} P_k & P_kC^T \\ CP_k & CP_kC^T + R \end{bmatrix}\right) \tag{12.13}$$

Since

$$\begin{aligned} &\mathbf{E}\, y(k) = \mathbf{E}(Cx(k) + v(k)) = C\hat{x}_k \\ &\mathbf{E}(x(k) - \hat{x}_k)(y(k) - C\hat{x}_k)^T = \\ &= \mathbf{E}(x(k) - \hat{x}(k))\left((x(k) - \hat{x}_k)^T C^T + v(k)^T\right) = P_kC^T \\ &\mathbf{E}(y(k) - C\hat{x}_k)(y(k) - C\hat{x}_k)^T = \\ &= \mathbf{E}\left(C(x(k) - \hat{x}_k) + v(k)\right)\left((x(k) - \hat{x}_k)^T C^T + v(k)^T\right) = CP_kC^T + R. \end{aligned} \tag{12.14}$$

The filtering equations

The probabilistic model (12.13) tell us how the measurement $y(k)$ is related to the state $x(k)$. As the goal of the filtering is to provide an estimate of $x(k)$ based on $y(k)$ it is then natural to determine the distribution of $x(k)$ conditional on knowledge of $y(k)$. By combining the results of (12.13), (12.8) and (12.9) the conditional distribution is

$$p_{X|Y}\left(x(k)|Y = y(k)\right) \sim N\left(\hat{x}_k^+, P_k^+\right) \tag{12.15}$$

where the mean value is

$$\hat{x}_k^+ = \hat{x}_k + P_kC^T(CP_kC^T + R)^{-1}(y(k) - C\hat{x}_k) \tag{12.16}$$

and covariance

$$P_k^+ = P_k - P_kC^T(CP_kC^T + R)^{-1}CP_k \tag{12.17}$$

The measurement $y(k)$ thus provide information about $x(t)$ leading to a new normally distributed random variable with a modified mean value $\hat{x}_k^+$ and reduced variance P_k^+. To complete the picture consider the time update in the state-equation (12.10). The state update equation is a linear combination of the current state and the independent noise where both quantities are Gaussian random variables. The distribution for the state variable time update, condistioned on all measurements up to time k, is thus

$$x(k+1) \sim N\left(\hat{x}_{k+1}, P_{k+1}\right) \tag{12.18}$$

where

$$\hat{x}_{k+1} = A\hat{x}_k^+ \tag{12.19}$$

and

$$P_{k+1} = AP_k^+A^T + Q. \tag{12.20}$$

We can see that the noise only influences the variance of the state-update distribution since the added noise has a zero mean value, see (12.11).

To this end it has been demonstrated that the state and measurement variables are normally distributed random variables. It has also been shown how the distribution of the state-variable is changed by considering the measurement $y(k)$. Hence, over time for $k = 1, 2, \ldots.$ we can track the distribution of the state-variable by recursively updating the mean values $\hat{x}_k$ and $\hat{x}_k^+$ and the corresponding covariances P_k and P_k^+. In summary the filtering equations are

$$\begin{aligned} \hat{x}_k^+ &= \hat{x}_k + P_k C^T (C P_k C^T + R)^{-1} (y(k) - C\hat{x}_k), \quad \text{measurement update} \\ P_k^+ &= P_k - P_k C^T (C P_k C^T + R)^{-1} C P_k \\ \hat{x}_{k+1} &= A\hat{x}_k^+, \quad \text{time update (prediction)} \\ P_{k+1} &= A P_k^+ A^T + Q. \end{aligned} \tag{12.21}$$

The recursion equations need to be initialized when started, i.e. values of $\hat{x}_0$ and P_0 must be supplied. The parameters are the mean value and variance of the initial state. If no knowledge is available a practical choice is $\hat{x}_0 = 0$ and $P_0 = \alpha I$ where α is a large scalar to reflect a large uncertainty (covariance) of $x(0)$.

The Kalman filter equations together with the assumption that the driving noise and the initial state are random variables with a known normal distribution provide means to calculate the distributions of $x(k_0)$ conditioned that $y(k)$ is known from $k = 0, 1, \ldots, k_0$. Access to the conditional distribution of the state $x(k_0)$ means that a point estimate of the variable $x(k_0)$ can be derived. A natural choice is to use the mean value of the conditional distribution, i.e. $\hat{x}_k^+$. Further analysis show that this value is also the minimum mean squared error estimate.

Remark 1. *Often the process noise is described as $w(k) = Pr(k)$ where the dimension of $r(k)$ is smaller then the dimension of $w(k)$. This imply that Q is singular. The net effect is that $w(k)$ is not a proper Gaussian random variable since Q is rank defect. However, what is needed for the Kalman filter is that P_k stays positive definite for all k which is guarenteed by* (12.20) *if $P_0 > 0$ is positive definite.*

Example again

Returning to the airplane example we identify that the state at time index k is

$$x(k) = \begin{bmatrix} p_c(k\Delta t) \\ s_c(k\Delta t) \end{bmatrix} \tag{12.22}$$

and the state transition matrix and measurement matrix are

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{12.23}$$

Remaining to define are the covariance matrices of the state and measurement noises. In a practical application it can be difficult to derive these from available data. Often they are used to tune the filter to a desired performance. A few key properties are though important. The relative magnitude between the covariance matrix of the state-noise and the measurement noise will control how much the filter will rely on the signal model as compared to the measurement. If the state-noise is small the measurements are given less weight and if the state-noise is large the measurements are given more weight.

Extensions

In the derivation above we have assumed that the matrices A and C and the covariance matrices Q and R are independent of the time index k. However, the derivation above is still valid if they also are modeled as time variant, i.e. each matrix can be assigned a time index e.g. A_k. This opens up to use the Kalman filter for a wide class of problems. For example with a very particular setup it is possible to show that the RLS and the LMS algorithms are special cases of the more general Kalman filter.

In many applications the state update and/or the measurement equations are nonlinear functions. A linearization approach to this case is known as the Extended Kalman filter [1] which is further developed into the Unscented Kalman Filter technique [2].

Chapter 13

LMS adaptive algorithm

The main disadvantage with the RLS algorithm is the computational complexity and storage requirement of the $\mathbf{R}_{yy}^{-1}$ matrix which is of size $M \times M$. Hence the complexity and storage requirement is a quadratic function of the filter length. Instead we can consider a iterative gradient descent method for optimizing (11.21). In each iteration the parameter vector $\hat{\mathbf{h}}^i$ is updated according to

$$\hat{\mathbf{h}}^{i+1}(N) = \hat{\mathbf{h}}^i(N) - \mu \nabla_{\mathbf{h}} L_N(\hat{\mathbf{h}}^i(N)) \tag{13.1}$$

where $\nabla_{\mathbf{h}} L_N(\mathbf{h})$ denotes the vector corresponding to the gradient of the scalar function L_N (actually a scalar field). The vector $\nabla_{\mathbf{h}} L_N(\mathbf{h})$ points in the direction where the function $L_N(\mathbf{h})$ increase most. In each iteration the vector is updated in the negative direction of the gradient in order to reduce the value of the function L_N. The positive scalar μ control the length of the step and must be selected small enough for the algorithm to converge. Employing the gradient descent method on the Wiener filter problem yields

$$\hat{\mathbf{h}}^{i+1}(N) = \hat{\mathbf{h}}^i(N) + 2\mu \sum_{n=0}^{N} \mathbf{y}(n) \left(x(n) - (\hat{\mathbf{h}}^i(N))^T \mathbf{y}(n) \right) \tag{13.2}$$

where index i is the gradient descent iteration index. The algorithm above is quite complex since for each new iteration index i and data index N the terms in the summation must be evaluated which means all previous data need to be stored. This is not a realistic alternative. A significant simplification is achieved if we merge the data sample index N and the iteration index i to be one. We can interpret this as, for each sample, taking one step using gradient descent on the function

$$L_{LMS}(N, \mathbf{h}) = e^2(N) = (x(N) - \mathbf{h}^T \mathbf{y}(N))^2 \tag{13.3}$$

which has gradient

$$\nabla_{\mathbf{h}} L_{LMS}(N, \mathbf{h}) = -2\mathbf{y}(N)e(N) \tag{13.4}$$

This simplification leads to the well known least-mean square algorithm (LMS) which can be summarized as

$$\begin{aligned} e(N) &= x(N) - \hat{\mathbf{h}}^T(N)\mathbf{y}(N) \\ \hat{\mathbf{h}}(N+1) &= \hat{\mathbf{h}}(N) + 2\mu \mathbf{y}(N)e(N) \end{aligned} \tag{13.5}$$

Comparing with the full gradient search method note that the update step is not in the true gradient direction but instead an estimate of it. Hence the method is also called a stochastic gradient descent method. However, if each update step is quite small, on average the movement in the parameters space will be in the negative gradient direction and the updated vector will move towards the global optimum. The rate of convergence towards the optimum is controlled by the value of the step length parameter μ. A too small value will lead to long convergence time while a large value will lead to a faster convergence. However, if selected too large the algorithm will diverge.

It is clear that the filter vector $\hat{\mathbf{h}}$ will be updated unless the error $e(N)$ is zero. In most applications it is highly unlikely that this would happen so the filter will, after approaching the optimum, continues to move around it. This movement will lead to a variance of the error which is larger then the optimum and is called a residual variance. The size of the residual variance is influenced by the step length choice. A large step length leads to larger residual variance and vice-versa. To see this consider the updating equation (13.5). If the algorithm has reached the convergence region at time index N_0 we have

$$\sum_{n=N_0}^{N_0+N} 2\mu\mathbf{y}(n)e(n) \approx 0 \tag{13.6}$$

i.e. on average the updating has stopped. The covariance, i.e. the variability of the filter coefficients will be proportional to

$$\operatorname{Cov}\hat{\mathbf{h}}(N) \approx \mathbf{E}\{4\mu^2\mathbf{y}(N)e(N)e(N)\mathbf{y}^T(N)\ \} = 4\mu^2\sigma_e^2\Phi_{yy} \tag{13.7}$$

The level of the variability (measured in terms if the covariance) is proportional to μ^2. This variability in the filter coefficients will be translated into an increased level of the filter error. To see this define

$$\tilde{\mathbf{h}}(n) = \hat{\mathbf{h}}(n) - \mathbf{h}_{\text{opt}} \tag{13.8}$$

as the deviation of the estimate from the optimal filter value. Now we can write the error signal as

$$e(n) = x(n) - \hat{\mathbf{h}}(n)^T\mathbf{y}(n) = \underbrace{x(n) - \mathbf{h}_{\text{opt}}^T\mathbf{y}(n)}_{e_{\text{opt}}(n)} - \underbrace{\tilde{\mathbf{h}}(n)^T\mathbf{y}(n)}_{\tilde{e}(n)} \tag{13.9}$$

where we call the last term the *residual error*. If we assume that the two error terms are uncorrelated we obtain

$$\mathbf{E}\,e(n)^2 = \mathbf{E}\,e_{\text{opt}}(n)^2 + \mathbf{E}\,\tilde{e}(n)^2 \tag{13.10}$$

The total error variance is the sum of the variance of the fixed optimal filter and the variance from the residual error. The variance of the residual error will be proportional to the covariance of $\hat{\mathbf{h}}(N)$ which illustrates that the adaptation of the LMS filter increases the error variance of the filter error signal. If the step length is increased the level of the residual error increases and vice versa.

In the LMS algorithm the continuous updating also results in that the algorithm adapts to changes automatically. When the signal properties of $x(n)$ or $y(n)$ changes, also the optimal solution is different.

Finally in comparison with RLS and the full gradient search, the LMS algorithm has low complexity. Both computation and memory requirements are linear in the filter length.

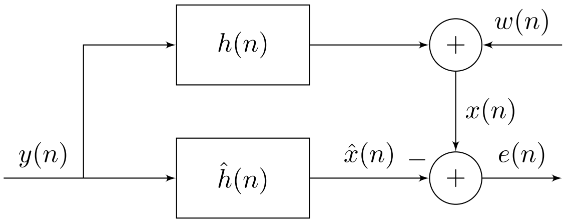


Figure 13.1: System modeling setup.

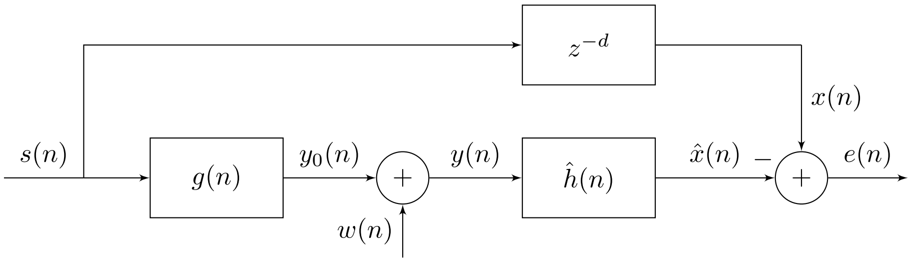


Figure 13.2: Inverse system modeling setup. The level of delay d will influence the quality of the solution.

Adaptive filter examples

In this section we discuss a few special cases where the optimal filtering methods can be used together with the LMS adaptation algorithm.

System modeling In systems modeling we ar interested in a parametric model $\hat{h}$ of the system h under study. Here we assume that the signals $x(n)$ and $y(n)$ are available for processing where $x(n)$ is the noisy version of the true system output. The setup is illustrated in Figure 13.1.

Inverse system modeling - Equalization In the equalization we are interested in a filter $\hat{h}$ which inverts the effect of the system g as well as suppress the noise signal $w(n)$, as illustrated in Figure 13.2 . The selected delay $d > 0$ will influence how well the optimal equalizer will work. In a communications application we can use the training samples of the message to learn the filter $\hat{h}$ and then turn off the adaptation to process the data part of the message.

Filtering, prediction and smoothing The block diagram illustrated in Figure 13.3 captures the application a the measured signal $y(n)$ is a sum of the desired signal $s(n)$ and a disturbance signal $w(n)$. By processing the signal in the filter $\hat{h}$ we try to recover $s(n-d)$. If $d = 0$ we causally try to recover s (filtering). If $d > 0$ we try to recover $d(n-d)$ from $y(n)$, i.e. smoothing. Finally if $d < 0$ the objective is to predict $s(n)$ d steps ahead. Practically we need access to example data $s(n)$ and $y(n)$ in order to train the filter.

Adaptive signal enhancement/separation Sometimes the signal $y(n)$ we process is composed of 2 signals with different spectral properties, e.g. one

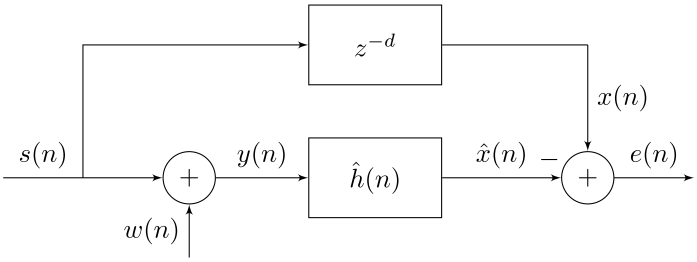


Figure 13.3: Setup which covers, filtering $d = 0$, prediction $d < 0$ and smoothing $d > 0$.

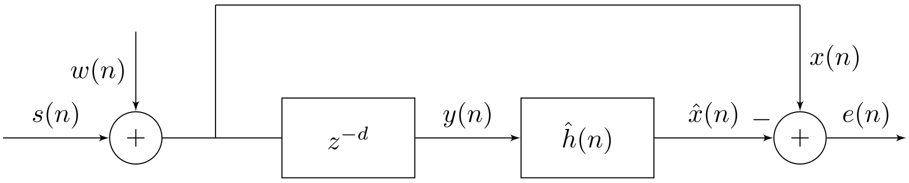


Figure 13.4: Adaptive signal enhancement with $d > 0$

component $s(n)$ have a line spectrum (sinusoidal signal), while the second component $w(n)$ have a wide bandwidth, e.g. white noise. The narrow band signal $s(n)$ can more easily be predicted from it's delayed version $s(n-d)$ by a suitable filter as compared to the wide band signal. If we run the adaptive filter according to the setup in Figure 13.4 we obtain a separation of the original signal where $\hat{x}(n)$ will be approximately $s(n)$ and consequently $e(n) \approx w(n)$.

Chapter 14

Analysis of LMS

Assume we are running an adaptive filter in a system modeling setup. This means that

$$x(n) = H_0(z)y(n) + w(n) \tag{14.1}$$

where $y(n)$ is a known (measured without error) input signal, the transfer function $H_0(z)$ is the unknown system to model and $w(n)$ is an unknown disturbance which is assumed to have zero mean. We connect the adaptive filter such that

$$\hat{x}(n) = H(z, \mathbf{h})y(n) \tag{14.2}$$

is the modeled output and where $H(z, \mathbf{h})$ is the model parametrized with the impulse response vector $\mathbf{h}$. The error between the measured output and model output is $e(n) = x(n) - \hat{x}(n)$. The optimal filter is defined as

$$\mathbf{h}_{\text{opt}} = \arg\min_{\mathbf{h}} \mathbf{E}\, e(n)^2 = \arg\min_{\mathbf{h}} \frac{1}{2\pi} \int_0^{2\pi} S_{ee}(\omega)\, d\omega \tag{14.3}$$

which means that we want to find the impulse response which will minimize the variance of the error which in turn is equivalent to minimize the integral of the power spectral density of the error $e(n)$ (Parseval's relation). The error is $e(n) = x(n) - \hat{x}(n) = (H_0(z) - H(z, \mathbf{h}))y(n) + w(n)$ which yields the power spectral density

$$S_{ee}(\omega) = |H_0(\omega) - H(\omega, \mathbf{h})|^2 S_{yy}(\omega) + S_{ww}(\omega) \tag{14.4}$$

where we have used the assumed fact that the signals y and w are uncorrelated. Since $S_{ww}(\omega)$ does not depend on $\mathbf{h}$ we have the alternative expression for the optimal filter

$$\mathbf{h}_{\text{opt}} = \arg\min_{\mathbf{h}} \frac{1}{2\pi} \int_0^{2\pi} |H_0(\omega) - H(\omega, \mathbf{h})|^2 S_{yy}(\omega)\, d\omega \tag{14.5}$$

The characterization in (14.5) indicate that the integrand has two factors which are non-negative functions. Clearly if the frequency function of the true system $H_0(\omega)$ can exactly be represented by some $\mathbf{h}^*$ such that $H_0(\omega) = H(\omega, \mathbf{h}^*)$ for all ω then the integral will be zero and $\mathbf{h}^*$ is an optimal solution. In practice this is often not possible and normally the function $H_0(\omega)$ can not be represented with a FIR filter with finite order M. In this case the optimal solution will yield a filter which makes the weighted error $|H_0(\omega) - H(\omega, \mathbf{h})|^2 S_{yy}(\omega)$ minimized when integrated. Clearly an optimal solution will be such that for frequencies where $S_{yy}(\omega)$ is large, the factor $|H_0(\omega) - H(\omega, \mathbf{h})|^2$ should be small, i.e. the FIR filter approximates the true H_o well. For frequencies where $S_{yy}(\omega)$ is small (or zero) then the approximation can be poor. Hence the shape of the power spectral density of the input signal will influence the character of the optimal solution whenever it is not possible to have $H_0(\omega) = H(\omega, \mathbf{h}_{\text{opt}})$ for all values of ω.

> **Example 9** Assume the input signal $y(n)$ is a single sinusoidal signal $y(n) = \cos(\omega_0 n) = \frac{1}{2}(e^{j\omega_0 n} + e^{-j\omega_0 n})$. In this case the power spectral density of the input will correspond to two delta functions at location ω_0 and $-\omega_0$ (or $2\pi - \omega_0$). Hence, all impulse responses $\mathbf{h}^*$ such that $H_0(\omega_0) = H(\omega_0, \mathbf{h}^*)$ will be an optimal filter for this input signal. The filter only need to match the amplitude and phase precisely at frequency ω_0. From another point of view we can say that with such an input signal we only obtain information about the unknown system at frequency ω_0. The behavior of the filter for all other frequencies will be unknown. If the frequency of the input signal changes to a frequency ω_1 then, in general, the set of optimal filter solutions will change, although the unknown system is still the same. □

Step length and convergence of LMS

A note on symmetric matrices Consider a real valued symmetric matrix $\mathbf{A} = \mathbf{A}^T \in \mathbb{R}^{M \times M}$. An eigenvalue λ and eigenvector $\mathbf{s}$ of the matrix $\mathbf{A}$ satisfy

$$\mathbf{A}\mathbf{s} = \lambda \mathbf{s} \tag{14.6}$$

For symmetric matrices there exist M unique eigenvectors $\mathbf{s}_m$ and corresponding real valued eigenvalues λ_m. The eigenvalue-eigenvector pairs are numbered according to the eigenvalue size, i.e. $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_M$. Denote by $\mathbf{S}$ the matrix constructed from all eigenvectors and $\mathbf{\Lambda}$ the diagonal matrix with the eigenvalues on the diagonal

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 & \ldots & \mathbf{s}_M \end{bmatrix}, \quad \mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_M \end{bmatrix}. \tag{14.7}$$

Since the matrix $\mathbf{A}$ is symmetric all eigenvectors are orthogonal, i.e. $\mathbf{s}_i^T \mathbf{s}_j = \delta(i-j)$. This means that $\mathbf{S}\mathbf{S}^T = \mathbf{I} = \mathbf{S}^T\mathbf{S}$ i.e. the matrix $\mathbf{S}$ is an orthogonal matrix (the inverse of the matrix is given by the transposed matrix).

By expressing (14.6) for all eigenvalues jointly we obtain

$$\mathbf{A}\mathbf{S} = \mathbf{S}\mathbf{\Lambda} \tag{14.8}$$

which le us write A as a product of three matrices known as the eigen-decomposition

$$A = \mathbf{S}\mathbf{\Lambda}\mathbf{S}^T. \tag{14.9}$$

If the matrix is positive definite (positive-semi definite) all eigenvalues are positive (non-negative).

Convergence analysis In principle the only adjustable parameter of the LMS filter is the step length μ besides the choice of the length of the filter M. In this section we will investigate theoretical bounds on its value in order to guarantee convergence of the filter and how the power spectrum of the input also influence the how fast the filter will converge to the optimum. The analysis is only approximately valid due to simplifying assumptions are employed.

Consider the LMS equations (13.5). Let us introduce the notation

$$\tilde{\mathbf{h}}(N) \triangleq \hat{\mathbf{h}}(N) - \mathbf{h}_{\text{opt}} \tag{14.10}$$

to denote the difference between the filter at time N and the optimal filter. Recall that the optimal filter is given by $\mathbf{h}_{\text{opt}} = \Phi_{yy}^{-1}\Phi_{yx}$. If we assume the step length is quite small the filter will only change slightly between samples. Hence, we could approximate these updates by the expected value taken over the signal samples assuming, when taking expectation, the filter coefficients are constant. By subtracting the optimal filter $\mathbf{h}_{\text{opt}}$ from the filter update equation of the LMS algorithm in (13.5) we obtain

$$\begin{aligned}\tilde{\mathbf{h}}(N) &= \tilde{\mathbf{h}}(N-1) + 2\mu\mathbf{y}(N)e(N)\\ &= \tilde{\mathbf{h}}(N-1) + 2\mu\mathbf{y}(N)(x(n) - \hat{\mathbf{h}}^T(N-1)\mathbf{y}(N))\\ &= \tilde{\mathbf{h}}(N-1) + 2\mu\mathbf{y}(N)(x(n) - \mathbf{y}^T(N)\hat{\mathbf{h}}(N-1))\end{aligned} \tag{14.11}$$

The expected value of the equation above is

$$\begin{aligned}\tilde{\mathbf{h}}(N) &= \tilde{\mathbf{h}}(N-1) + \mathbf{E}\,2\mu\mathbf{y}(N)(x(n) - \mathbf{y}^T(N)\hat{\mathbf{h}}(N-1))\\ &= \tilde{\mathbf{h}}(N-1) + 2\mu(\underbrace{\Phi_{yx}}_{\Phi yy\mathbf{h}_{\text{opt}}} - \Phi_{yy}\hat{\mathbf{h}}(N-1))\\ &= (\mathbf{I} - 2\mu\Phi_{yy})\tilde{\mathbf{h}}(N-1)\end{aligned} \tag{14.12}$$

which yields a recursion equation for the filter error. Clearly we have

$$\tilde{\mathbf{h}}(N) = (\mathbf{I} - 2\mu\Phi_{yy})^N\tilde{\mathbf{h}}(0) \tag{14.13}$$

and we can conclude that the filter error converges to the zero vector if all the elements of the matrix $(\mathbf{I} - 2\mu\Phi_{yy})^N$ converge to zero as $N \to \infty$. As Φ_{yy} is a positive semi-definite matrix we can use the eigen-decomposition $\Phi_{yy} = \mathbf{S\Lambda S}^T$

and write

$$
\begin{aligned}
\tilde{\mathbf{h}}(N) &= (\mathbf{I}-2\mu\mathbf{S}\mathbf{\Lambda}\mathbf{S}^T)^N\tilde{\mathbf{h}}(0) = (\mathbf{S}\mathbf{S}^T-2\mu\mathbf{S}\mathbf{\Lambda}\mathbf{S}^T)^N\tilde{\mathbf{h}}(0) = \mathbf{S}(\mathbf{I}-2\mu\mathbf{\Lambda})^N\mathbf{S}^T\tilde{\mathbf{h}}(0) \\
&= \mathbf{S}\begin{bmatrix} 1-2\mu\lambda_1 & 0 & \dots & 0 \\ 0 & 1-2\mu\lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1-2\mu\lambda_M \end{bmatrix}^N \mathbf{S}^T\tilde{\mathbf{h}}(0) \\
&= \mathbf{S}\begin{bmatrix} (1-2\mu\lambda_1)^N & 0 & \dots & 0 \\ 0 & (1-2\mu\lambda_2)^N & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (1-2\mu\lambda_M)^N \end{bmatrix} \mathbf{S}^T\tilde{\mathbf{h}}(0)
\end{aligned}
\tag{14.14}
$$

where λ_i are the eigenvalues of the matrix Φ_{yy}. By changing the basis of the initial error $\mathbf{S}^T\tilde{\mathbf{h}}(0) = [\tilde{h}_1, \tilde{h}_2, \cdots \tilde{h}_M]^T$ we can also write the error as a linear combinatin of the eigenvectors $\mathbf{s}_i$

$$\tilde{\mathbf{h}}(N) = \sum_{i=1}^{M} \mathbf{s}_i(1-2\mu\lambda_i)^N\tilde{h}_i \tag{14.15}$$

The desired convergence to zero is guaranteed if $|1-2\mu\lambda_i| < 1$ for all i. Since $\lambda_i \geq 0$ for all i, the step length μ must be positive. Hence the limiting eigenvalue is the largest one and the bounds on μ guaranteeing convergence are expressed as

$$0 < \mu < \frac{1}{\lambda_1} \tag{14.16}$$

where the λ_1 is the largest eigenvalue of Φ_{yy}. It is notable that the convergence only depends on the input $y(n)$ to the filter and the size of μ. It is important to note that the choice $\mu = \frac{1}{\lambda_1} - \epsilon$ for some small ϵ would make the factor corresponding to the largest eigenvalue to become $(-1+\epsilon)^N$ and the convergence rate of that parameter direction would go to zero as $\epsilon \to 0$. With this choice we are overshoting the optimium in each step, since $(-1+\epsilon) < 0$. This is of course not desirable. Hence, a more realistic bound for the step length is to the choice $\mu = \frac{1}{2\lambda_1}$ which for the largest eigenvalue then would yield the factor $(1-1)^N = 0$, i.e. convergence in one step.

> **Example 10** Assume the input to the filter is a white noise signal with variance σ_y^2. Then $\Phi_{yy} = \sigma_y^2\mathbf{I}$, i.e., a matrix proportional to the identity matrix. All eigenvalues are equal to σ_y^2 and the upper bound on μ is $1/\sigma_y^2$. This means that μ should be scaled with the inverse of the variance of the input. □

It should be remembered that the bound (14.16) is based on a simplifying approximation and should only be used as a guide to determine an appropriate value. We also know that a large μ leads to a high residual variance so from that perspective μ should be selected small.

Returning back to (14.14) we can interpret the result that the convergence rate in the directions in the parameter space given by the eigenvectors converge at different rates. The direction with the slowest convergence corresponds to the smallest eigenvalue and the convergence rate is given by the factor $(1-2\mu\lambda_M)$. If we for example set $\mu=\frac{1}{2\lambda_1}$ we can express the factor as $(1-\frac{\lambda_M}{\lambda_1})$. Clearly, if the fraction $\frac{\lambda_M}{\lambda_1}$ is small, i.e. the eigenvalue spread is large, there will be a large spread of convergence speeds in the different directions of the parameter space. If λ_M is zero then there will be no convergence in the corresponding parameter direction. This is the case when the optimal filter is non-unique since in such a case Φ_{yy} is singular. The opposite case is when the input $y(n)$ is a white noise signal since then (as before) all eigenvalues of Φ_{yy} are identical and equal to the variance of the signal. All directions in the parameter space will then converge with the same speed.

The eigenvalue spread of the matrix Φ_{yy} is closely linked to the shape of the power spectrum of the signal. A smooth spectrum will have a small spread while a peaky spectrum will have a larger spread. A white noise signal is one of the extreme cases which imply that all eigenvalues are identical and consequently have a zero spread between the eigenvalues. The other extreme side is when one or several eigenvalues are zero. A signal which contain only pure sinusoidal signals can have zero eigenvalues. Each sinusoidal signal will contribute with two non-zero eigenvalues and depending on the size of M the others (if any left) will be zero. If, for example, $M=3$ and $y(n)$ contain only one sinusoidal component, one eigenvalue of Φ_{yy} will be zero and the eigenvalue spread will be infinite.

Chapter 15

Quantization errors and signal detectors

So far we have disregarded the effects of using a finite precision representing signals and filters. Since each number in a computer is represented by a sequence of ones and zeros, i.e. bits, the amount of bits used will influence the quality of the processing but also how much memory and how expensive it is to perform arithmetic operations.

Here we will briefly discuss the effects of using a finite representation of the signal samples. We start with the natural analog frontend, the analog to digital converter. When an analog signal is sampled a hold circuits will lock the analog voltage at a constant level. This level is then converted to a digital word including B bits resulting in a quantized signal. The two steps are illustrated graphically in 15.1. The output of the ADC can be described by

$$x_q(n) = x_c(n\Delta t) + q(n) = x(n) + q(n) \tag{15.1}$$

where $x_q(n)$ is the quantized signal and $q(n)$ is the difference between the true sample $x(n)$ and the output $x_q(n)$. Assume the quantizer is uniform and the output word has B bits. Hence, the output of the quantizer has 2^B different

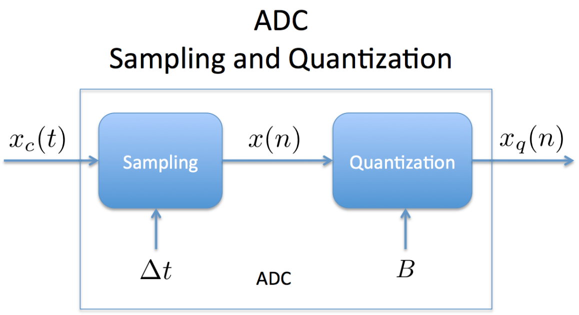


Figure 15.1: An analog to digital converter with a sampling circuit operating with sampling frequency $1/\Delta t$ and a quantization circuit converting the analog value into a digital word with B bits.

levels. Assume the quantizer is uniform and have max and min levels of A and $-A$ respectively. Then the quantization step is $S_q = \frac{2A}{2^B-1} \approx \frac{2A}{2^B}$ and the quantization error is bounded by $|q(n)| \leq \frac{S_q}{2} = \frac{A}{2^B}$. The quantization error is completely determined by the value of the input signal and the quantization levels. However, a good model of the error is to assume $q(n)$ is a zero mean random variable with a uniform distribution in the interval $[-\frac{A}{2^B}, \frac{A}{2^B}]$. The variance of the quantization error is

$$\sigma_q^2 = \int_{-A/2^B}^{A/2^B} x^2 \frac{1}{2A/2^B}\, dx = \frac{1}{3}\frac{A^2}{2^{2B}} \tag{15.2}$$

A common way to express the level of the quantization is to relate it to a sinusoidal signal with maximum amplitude A. The power of the sinusoidal signal is $A^2/2$ and the resulting signal to noise ratio (SNR) or dynamic range is given by

$$R_D = 10\log_{10} \frac{\frac{A^2}{2}}{\frac{A^2}{3\times 2^{2B}}} = 10\log_{10}\frac{3}{2} + 10\log_{10} 4^B = 1.76 + 6.02B \quad [\text{dB}]. \tag{15.3}$$

Every extra bit will provide an increased SNR with 6.02 dB. The quantization noise will have a constant power spectral density and will correspond to what is commonly referred to as a noise floor when analyzing the powers spectral density of a measured and sampled signal. If the power of the measured signal decreases the corresponding power spectrum will approach and finally disappear in the noise floor.

Oversampling and Quantization

Assume we are interested in sampling a real valued signal x where the information of interest is between frequency 0 and f_0. The signal is sampled at a rate which is M times the necessary rate according to the Nyquist theorem, i.e. $\omega_s = 2\pi M 2 f_0$ [Hz], using a total of B bits in the ADC. From the assumptions it follows that the power spectral density $S_{xx}(\omega)$ of the signal x is zero for $|\omega| > \frac{\omega_s}{2M}$. The power of the signal is hence

$$\sigma_x^2 = \frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} S_{xx}(\omega)\, d\omega = \frac{1}{\omega_s}\int_{-\frac{\omega_s}{2M}}^{\frac{\omega_s}{2M}} S_{xx}(\omega)\, d\omega \tag{15.4}$$

The sampled signal is filtered through an ideal LP filter with a cut off frequency of f_0 [Hz]. Before filtering the total power of the quantization error was $\sigma_q^2 = \frac{A^2}{3\times 2^{2B}}$. This corresponds to the power spectral density

$$S_q(\omega) = \sigma_q^2 = \frac{A^2}{3\times 2^{2B}}, \quad \omega \in [-\frac{\omega_s}{2}, \frac{\omega_s}{2}] \tag{15.5}$$

After filtering with the (ideal) low pass filter the quantization noise has PSD

$$S_{q,f}(\omega) \begin{cases} \frac{A^2}{3\times 2^{2B}}, & |\omega| < \frac{\omega_s}{2M} \\ 0, & \text{otherwise} \end{cases} \tag{15.6}$$

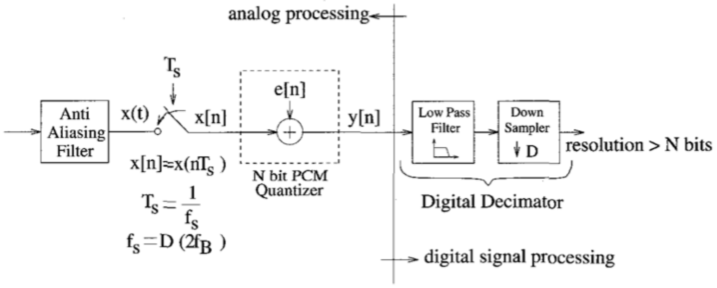


Figure 15.2: Block diagram of an oversampled ADC. Image reprinted from IEEE Signal Processing Magazine, Volume 13, Issue 1, September 1996, pages 61–84.

which corresponds to the noise variance

$$\sigma_{q,f}^2 = \frac{1}{\omega_s} \int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} S_{q,f}(\omega)\, d\omega = \sigma_q^2 / M \tag{15.7}$$

After the filtering the power of the quantization error has reduced a factor M while the PSD of the signal x (and hence, also its variance) is unchanged. After the LP-filtering we have increased the SNR a factor M.

Assume we downsample the filtered signal a factor M. Then the noise will have a flat power spectral density in the full frequency band $[0, \omega_s/M]$ and we can interpret the noise again as a quantization noise.

The SNR can then be expressed as

$$\text{SNR} = \frac{\sigma_x^2}{\frac{A^2}{3 \times 2^{2B}} \frac{1}{M}} = \frac{\sigma_x^2}{\frac{A^2}{3 \times 2^{2(B+(\log_2 M)/2)}}} \tag{15.8}$$

This corresponds to an effective increase in the number of bits of $\frac{\log_2 M}{2}$. Hence if we allow the word length to increase correspondingly during the LP filtering we can now downsample the signal a factor M. The resulting signal will have a dynamic range corresponding to $B + \frac{\log_2 M}{2}$ number of bits although the original ADC only used B bits. The price paid for the increased dynamic range is that the ADC must operate at a frequency M times higher then the required Nyquist rate. Even further gains can be obtained if so called noise shaping filtering is employed which form the back bone in the family of oversampled ADCs called Sigma-Delta ($\Sigma - \Delta$) converters.

Signal Matched Filtering

Signal matched filtering or simply matched filtering is a filtering method commonly applied for applications where the task is to determine if a signal is present or not. A classical application is radar signal processing where the receiver task is to detect the radar signal which are reflected on the target and accurately measure the time from transmission to reception.

Consider the following problem setup. Assume we receive a (possibly complex) signal

$$x(n) = s(n) + v(n) \tag{15.9}$$

where $s(n)$ is the signal to be detected and $v(n)$ is a noise signal which we assume has zero mean, an auto-correlation function $\phi_{vv}(n) = \sigma_v^2 \delta(n)$ and is uncorrelated with $s(n)$. The noise is hence uncorrelated in time, i.e. a white noise signal.

A signal matched filter is a filter which maximizes the signal to noise ratio of the received and filtered signal. For our derivation let us assume a finite signal length scenario where we observe samples $x(n)$, $n = 0, \ldots, N-1$ and we collect all observed samples in a vector

$$\mathbf{x} = \begin{bmatrix} x(0) & x(1) & \cdots & x(N-1) \end{bmatrix}^T \tag{15.10}$$

and we define $\mathbf{s}$ and $\mathbf{v}$ correspondingly. At time $N-1$ we can write the output $y(n)$ obtained by filtering the signal $x(n)$ through a FIR filter with impulse response $h(n)$ as

$$y(N-1) = \sum_{k=0}^{N-1} h(k)x(n-k) = \mathbf{h}^H \mathbf{x} \tag{15.11}$$

where

$$\mathbf{h} = \begin{bmatrix} h^*(N-1), & h^*(N-2), & \cdots, & h^*(0) \end{bmatrix}^T \tag{15.12}$$

and $(\cdot)^H$ denotes complex conjugation and transpose and $(\cdot)^*$ denotes complex conjugation. The filtered signal component in $y(N-1)$ is hence $\mathbf{h}^H\mathbf{s}$ while the noise component in the filtered signal is $\mathbf{h}^H\mathbf{v}$. The signal to noise ratio after the filtering is hence

$$\text{SNR} = \frac{|\mathbf{h}^H\mathbf{s}|^2}{\mathbf{E}\{|\mathbf{h}^H\mathbf{v}|^2\}} = \frac{|\mathbf{h}^H\mathbf{s}|^2}{\mathbf{E}\{\mathbf{h}^H\mathbf{v}\mathbf{v}^H\mathbf{h}\}} = \frac{|\mathbf{h}^H\mathbf{s}|^2}{\sigma_v^2\mathbf{h}^H\mathbf{h}} \tag{15.13}$$

since $\mathbf{E}\{\mathbf{v}\mathbf{v}^H\} = \sigma_v^2 I$. The inner product between any two vectors $\mathbf{h}$ and $\mathbf{s}$ satisfy the following inequality

$$|\mathbf{h}^H\mathbf{s}|^2 \leq (\mathbf{h}^H\mathbf{h})(\mathbf{s}^H\mathbf{s}) \tag{15.14}$$

with equality if $\mathbf{h} = \mathbf{s}$. Inserting this inequality in (15.13) results in

$$\text{SNR} = \frac{|\mathbf{h}^H\mathbf{s}|^2}{\sigma_v^2\mathbf{h}^H\mathbf{h}} \leq \frac{\mathbf{s}^H\mathbf{s}}{\sigma_v^2} \tag{15.15}$$

This shows that the filter which optimizes the SNR of the received signal is $\mathbf{h}_{\text{opt}} = \mathbf{s}$. The impulse response of the optimal matched filter is then given by

$$h_{\text{opt}}(n) = s(N-1-n)^*, \quad n = 0, \ldots, N-1, \tag{15.16}$$

i.e. the time reversed and complex conjugated signal we want to detect. In the frequency domain the optimal filter has the frequency function

$$H_{\text{opt}}(\omega) = S^*(\omega)e^{-j\omega(N-1)} \tag{15.17}$$

The bandwidth of the optimal filter matches the bandwidth of the signal to be detected.

Bibliography

[1] B D O Anderson and J B Moore. *Optimal Filtering.* Prentice-Hall, Englewood Cliffs, New Jersey, 1979.

[2] S J Julier and J K Uhlman. A new extension of the Kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls 3*, 1997.

[3] R E Kalman. A new approach to linear filtering and and prediction problems. *Trans. ASME Ser. D. J. Basic. Eng.*, 82:24–45, 1960.

[4] S M Kay. *Fundamentals of Statistical Signal Processin - Estimation Theory.* Prentice Hall, 1993.

[5] Abraham. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.

[6] Ronald W. Schafer. What is a savitzky-golay filter? [Lecture notes]. *IEEE Signal Processing Magazine*, 28(4):111–117, 2011.

Appendix A

Additional material

A.1 The conditional probability of a Gaussian random variable

First we note the following matrix identities for block matrices: The inverse of a triangular block matrix with identity blocks on the diagonal is simply a matrix with the same structure where the off diagonal block has a reversed sign.

$$\begin{bmatrix} I & A \\ 0 & I \end{bmatrix} \begin{bmatrix} I & -A \\ 0 & I \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} = I \tag{A.1}$$

Define the block triangular matrix P as

$$P = \begin{bmatrix} I & -Q_{12}Q_{22}^{-1} \\ 0 & I \end{bmatrix}. \tag{A.2}$$

If the covariance matrix Q is congruence transformed with P, i.e. PQP^T, we note the identity

$$PQP^T = \begin{bmatrix} I & -Q_{12}Q_{22}^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix} \begin{bmatrix} I & 0 \\ -Q_{22}^{-1}Q_{12}^T & I \end{bmatrix} = \begin{bmatrix} Q_{11} - Q_{12}Q_{22}^{-1}Q_{12}^T & 0 \\ 0 & Q_{22} \end{bmatrix} \tag{A.3}$$

Clearly the transformation block-diagonalized the covariance matrix Q. Also since $\det(AB) = \det A \det B$, and $\det P = 1$ we can conclude that

$$\det Q = \det(Q_{11} - Q_{12}Q_{22}^{-1}Q_{12}^T) \det Q_{22} \tag{A.4}$$

According to the definition of conditional distributions (see (12.7)) we obtain for the variables X and Y

$$p_{X|Y}(x|Y=y) = \frac{p_{X,Y}(x,y)}{p_Y(y)} = \frac{p_Z(z)}{p_Y(y)} \tag{A.5}$$

$$= \frac{\frac{1}{\sqrt{(2\pi)^n \det Q}} \exp\left(-\frac{1}{2}(z-\mu)^T Q^{-1}(z-\mu)\right)}{\frac{1}{\sqrt{(2\pi)^{n_y} \det Q_{22}}} \exp\left(-\frac{1}{2}(y-\mu_y)^T Q_{22}^{-1}(y-\mu_y)\right)} \tag{A.6}$$

where n_y and n_x is the length of the vector y and x respectively and $n_y + n_x = n$. By inserting identity matrices, in the form of $P^{-1}P = I$ and its transpose, right and left of Q in (A.5) the argument to the exponential function in the numerator becomes

$$-\frac{1}{2}(z-\mu)^T Q^{-1}(z-\mu) = \begin{bmatrix} x-\mu_x \\ y-\mu_y \end{bmatrix}^T P^T P^{-T} Q^{-1} P^{-1} P \begin{bmatrix} x-\mu_x \\ y-\mu_y \end{bmatrix} \tag{A.7}$$

Now using properties from (A.1) to (A.3), the expression in (A.7) turns into

$$-\frac{1}{2}(x-\mu_x-Q_{12}Q_{22}^{-1}(y-\mu_y))^T\tilde{Q}^{-1}(x-\mu_x-Q_{12}Q_{22}^{-1}(y-\mu_y))-\frac{1}{2}(y-\mu_y)^T Q_{22}^{-1}(y-\mu_y) \tag{A.8}$$

where $\tilde{Q} = Q_{11} - Q_{12}Q_{22}^{-1}Q_{12}^T$. To simplify the notation let us define $\tilde{\mu}_x(y) = \mu_x - Q_{12}Q_{22}^{-1}(y-\mu_y)$. Inserting (A.8) back into (A.5), use the property in (A.4) and the identity $\exp(a+b) = \exp(a)\exp(b)$, we can factor out $p_Y(y)$ from the numerator and obtain

$$p_{X|Y}(x|Y=y) = \frac{p_Y(y)\frac{1}{\sqrt{(2\pi)^{n_x}\det\tilde{Q}}}\exp\left(-\frac{1}{2}(x-\tilde{\mu}_x(y))^T\tilde{Q}^{-1}(x-\tilde{\mu}_x(y))\right)}{p_Y(y)} \tag{A.9}$$

Since $p_Y(y)$ in the numerator and denominator cancel each other we are left with a PDF for a Gaussian variable with mean $\mu_x - Q_{12}Q_{22}^{-1}(y-\mu_y)$ and covariance $Q_{11} - Q_{12}Q_{22}^{-1}Q_{12}^T$ which concludes the derivation.

A.2 Linear minimum mean square error estimator

In this section we consider the estimation problem of finding the linear minimum variance estimate of a vector x based on measurements of a correlated vector y.

Assume the a-priori knowledge regarding the two vectors, are

$$\mathbf{E}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} \tag{A.10}$$

and

$$\text{Cov}\begin{bmatrix} x \\ y \end{bmatrix} \triangleq \mathbf{E}\begin{bmatrix} x-\mu_x \\ y-\mu_y \end{bmatrix}\begin{bmatrix} x-\mu_x \\ y-\mu_y \end{bmatrix}^T = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix} \tag{A.11}$$

where we assume Q_{22} to be positive definite.

The linear minimum mean square error estimate $\hat{x}_{\text{LMMSE}}$ is defined as

$$\hat{x}_{\text{LMMSE}} = z_0 + K_0 y \tag{A.12}$$

where vector z_0 and matrix K_0 are the minimizers to the optimization problem

$$\min_{z,K} \mathbf{E}(x-z-Ky)^T(x-z-Ky). \tag{A.13}$$

The estimator (A.12) is actually affine whenever $z_0 \neq 0$. Using the property that $\operatorname{tr} AB = \operatorname{tr} BA$ we can rewrite the functional in (A.13) as

$$\begin{gathered}\mathbf{E}(x-z-Ky)^T(x-z-Ky) = \operatorname{tr}\mathbf{E}(x-z-Ky)(x-z-Ky)^T = \\ \operatorname{tr}\left(Q_{11} - Q_{12}K^T - KQ_{12}^T + KQ_{22}K^T\right) + (\mu_x - K\mu_y - z)^T(\mu_x - K\mu_y - z) = \\ \operatorname{tr}(Q_{11} - Q_{12}Q_{22}^{-1}Q_{12}^T) + \\ \operatorname{tr}(K - Q_{12}Q_{22}^{-1})Q_{22}(K - Q_{12}Q_{22}^{-1})^T + (\mu_x - K\mu_y - z)^T(\mu_x - K\mu_y - z)\end{gathered} \tag{A.14}$$

The first term is independent of z and K. The second and third terms in (A.14) are both non-negative and hence we minimize (A.13) by setting

$$K_0 = Q_{12}Q_{22}^{-1} \quad \text{and} \quad z_0 = \mu_x - K_0\mu_y \tag{A.15}$$

and we obtain the optimal estimator as

$$\hat{x}_{\text{LMMSE}} = \mu_x + Q_{12}Q_{22}^{-1}(y - \mu_y) \tag{A.16}$$

and the resulting variance of the optimal estimate

$$\mathbf{E}[(x - \hat{x}_{\text{LMMSE}})(x - \hat{x}_{\text{LMMSE}})^T] = Q_{11} - Q_{12}Q_{22}^{-1}Q_{12}^T. \tag{A.17}$$

We can derive an alternative formula for the LMMSE estimator based on the inverse of the full covariance matrix. To see this define the inverse covariance matrix (also known as the precision matrix)

$$S \triangleq \begin{bmatrix} S_{11} & S_{12} \\ S_{12}^T & S_{22} \end{bmatrix} \triangleq \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix}^{-1} \tag{A.18}$$

where sub-blocks of S have the same dimension as the sub-blocks of Q. Then since $SQ = I$ we obtain $0 = S_{11}Q_{12} + S_{12}Q_{22}$ which shows that $Q_{12}Q_{22}^{-1} = S_{11}^{-1}S_{12}$.

In Appendix A.1 we derived the conditional distribution for two joint Gaussian random vectors. If we compare the results there with the results for the Gaussian case we can conclude that the optimal LMSSE estimate is equal to the mean value of the conditional distribution and the variance of the LMMSE estimate is the variance of the conditional Gaussian distribution.