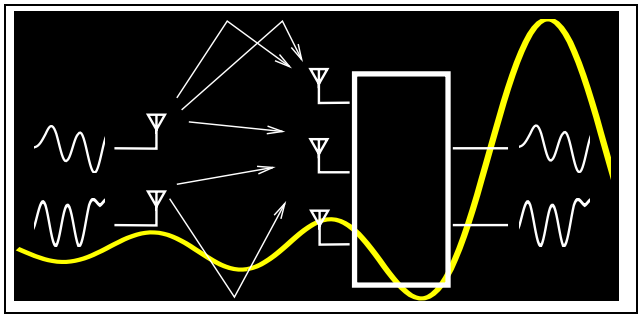
SIGNAL PROCESSING FOR COMMUNICATIONS

Delft University of Technology
Faculty of Electrical Engineering, Mathematics, and Computer Science
Circuits and Systems

SIGNAL PROCESSING FOR COMMUNICATIONS

ET4 147
Spring 2005

Alle-Jan van der Veen and Geert Leus



Contents

PREFACE

This reader contains the course material for the 4-th/5-th year undergraduate course on signal processing for communications at TU Delft, ET4 147. The course centers around two main topics: wireless communications and array signal processing. Both areas are currently experiencing a rapid evolution, caused, e.g., by new possibilities from the use multiple sensors and by the fact that computationally intensive algorithms are no longer a bottleneck.

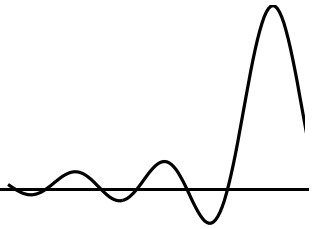
An example of this, in mobile communications, is the recent "discovery" that the use of adaptive antennas enables the separation of multiple signals that have the same frequency, as long as these arrive from different locations or otherwise have distinctive propagation paths. In the same manner, echos caused by multipath can be detected and suppressed. By using the strong structural properties of digital signals, an array calibration is not necessary, and only a few dozen samples are needed. Algebraic techniques play an important role in the separation of signals.

The leading theme in the course is to demonstrate how modern mathematical techniques, mostly from linear algebra, can be applied to solve emerging problems in these areas. It is assumed that the participants already have a fair background in linear algebra, although one lecture is spent to refresh this knowledge. The course material is derived in part from previously published papers, by several authors, but edited to fit the course.

Alle-Jan van der Veen

Chapter 1

THE WIRELESS CHANNEL



Contents

Wireless communications is currently experiencing a dramatic evolution. The unprecedented growth of existing services, and the emergence of many new ones, creates interesting and challenging opportunities for signal processing. In this course, a review is given of a number of signal processing algorithms that will be of value in this area. A good understanding of the applicability and potential of such algorithms, however, requires a suitable data model to start with. The model should be based on reality but not be overly detailed. The purpose of this chapter is to derive such a model.

1.1 INTRODUCTION[1]

The radio age began just over one hundred years ago with the invention of the radio telegraph by Guglielmo Marconi. After steady but mostly unremarkable progress, an important change of gears occured in the 1990s. The rapid evolution in radio technology results in new and improved services at lower costs. This in turn generates an expanding number of subscribers to wireless services. Wireless revenues are currently growing at around 40% per year and these trends are likely to continue for several years. An (incomplete) list of wireless applications is given in table 1.1. In this course, we will be mostly concerned with signal processing techniques for cellular applications.

[1] From Paulraj and Papadias [1].

Table 1.1. Applications of wireless technology

Personal Communications Systems (PCS)	*Satellite Communications and Navigation*
Mobile communications (GSM, CDMA)	Mobile satellite services
Analog and digital cellular systems	Global positioning system (GPS)
Cordless phones	VSAT
Messaging systems	Cellular communications
Hand pagers	
Wireless Data	*Dual-Use Applications*
Wireless PDAs (GPRS, EDGE, UMTS)	Direction-finding radar
Wireless local-area networks (WLANs)	Commercial spread-spectrum
Wide-area networks	Radio astronomy
RF Identification (RFID)	*Automotive and Transportation Electronics*
Inventory control and tracking	Collision-avoidance/warning systems
Personnel and plant security	In-vehicle navigation systems
Automatic vehicle identification (toll)	Intelligent-vehicle highway systems (IVHS)
Debit/credit systems	

A wireless designer is faced with a number of challenges. These include a complicated and rapidly varying propagation environment, limited availability of radio spectrum, limited energy storage capability of batteries in portable units, user demands for higher data rates, voice quality and longer talk times. The increasing demand for capacity in wireless systems traditionally translates directly into a demand for more bandwidth, which however is already quite restricted. At the same time, the infrastructure investment costs are often a limiting factor when deploying a new system aimed at wide area coverage. Increasing the range of current systems is therefore of great interest as well.

A number of technologies are being employed to meet such diverse requirements. These include advanced multiple access techniques, efficient voice and channel coding, and improved signal processing. For signal processing, a critical issue to be investigated for future systems is advanced interference reduction, or more ambitiously the joint detection of multiple signals at the same frequency.

In figure 1.1, the layout of a cellular network is schematically shown. Each user is assigned to a base station, conceptually the one that is closest in physical distance, but in practice the one that gives strongest reception. Within a logical cell, there is only one base station. Each base station has a collection of carrier frequencies at which it can communicate to its users. In traditional systems, the base stations in directly neighboring cells do not use the same set of frequencies. A user in a neigboring cell that does use the same frequency can cause interference: this is called *adjacent cell interference (ACI)*. Similarly, a user in the same cell that happens to use the same frequency causes *co-channel interference (CCI)*.[2] If there is multipath, the base station can receive a number of echos of the signal of the desired user, from different directions

[2]Often, no distiction is made between ACI and CCI.

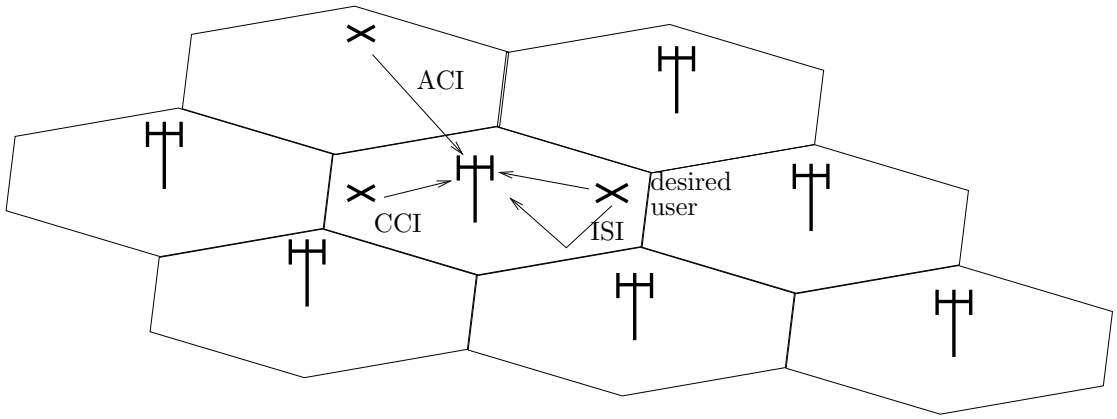


Figure 1.1. Cellular network concept

and displaced in time, which causes *intersymbol interference (ISI)*.

There are two critical factors in the design of wide area mobile communication systems: coverage and capacity. These factors have a direct impact on the cost and quality of the services since spectral resources are limited and spectral efficiency is necessary. The spatial dimension is to a large extent unexplored in wireless systems. Traditional telecommunication schemes multiplex channels in frequency and time, but use the spatial dimension in general only in a very rudimentary fashion. By incorporating adaptive antenna arrays at the base station, co-channel interference can be reduced to a large extent by efficient *space-time (ST)* processing techniques. This may increase both the capacity, coverage, and quality of future systems. With proper processing, it is also possible to multiplex channels in the spatial dimension just as in the frequency and time dimensions. Spatially selective reception and transmission can reduce interference in the system significantly, allowing frequencies to be reused more often and thereby increasing capacity.

The key leverage of space-time signal processing is that this provides significant co-channel interference reduction, in a way that is not possible with single antenna modems. This is because co-channel interference and the desired signal almost always arrive at the base station antenna array with distinct and well separated spatial-temporal signatures. By identifying this signature for the user-to-base station communication link (uplink or reverse link), the signal of interest may be extracted from the noise while suppressing interference. Furthermore, with knowledge of the spatial-temporal signature describing the base station-to-user (downlink or forward link) channel, transmission schemes may be devised which maximize the power of the signal of interest at the user while minimizing cochannel interference to other mobiles as well as overall radiated power. This offers substantial capacity increases over current wireless system implementations.

The spatial dimension can also be used to enhance other aspects of space-time modem perfor-

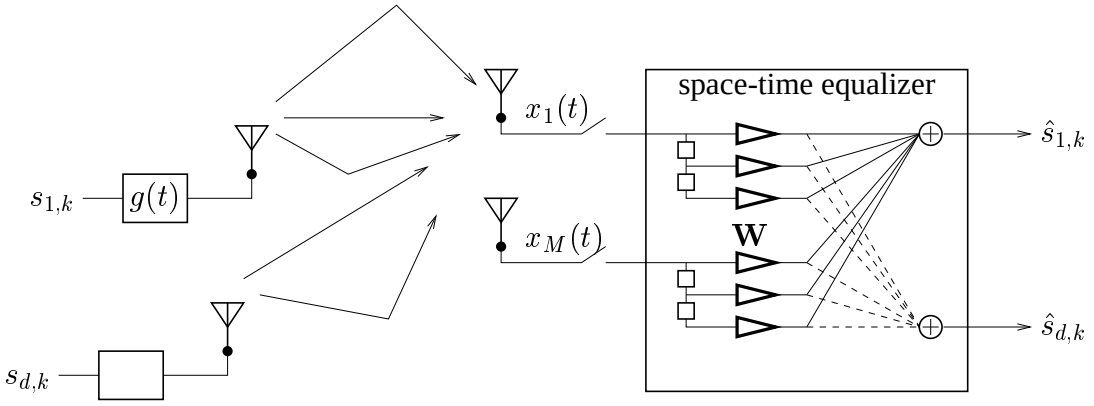


Figure 1.2. Wireless communication scenario

mance, in particular with regard to *diversity*. In receive mode, the complicated radio channel causes the signal arriving at a single antenna to exhibit significant fluctuations in power, both in the time domain and in frequency. This fading, however, might be quite independent in space, even over short distances. Thus, properly combining multiple antenna outputs at the base station (i.e., space-time equalization) can be used to enhance overall reception and improve signal to thermal noise ratio, enhance diversity gain and even improve inter-symbol interference suppression. In transmit mode (downlink), the spatial dimension can enhance array gain, improve transmit diversity and reduce delay spread at the subscriber's end.

In this chapter, we develop a wireless channel model that can form the basis for the design of signal processing algorithms for space-time modems. This forms the preparation for the next few chapters, which go into the details of some of these algorithms. Extensive background information on channel models and signal modulation can be found in Jakes [2], William Lee [3], Lee and Messerschmidt [4], Steele [5], Proakis [6], as well as others.

A data model for wireless communication consists of the following parts (see figure 1.2):

1. Source model: signal alphabet, data packets, and modulation by a pulse shape function
2. Physical channel: multipath propagation over the wireless channel
3. Receiver model: reception of multiple signals at multiple antennas, sampling, beamforming, equalization and decision making.

1.2 ANTENNA ARRAY RECEIVER MODEL

For several reasons, it is simplest to discuss the data model in reverse order. Hence, in this section we start out with making simple models for the receiver, emphasizing the reception of a

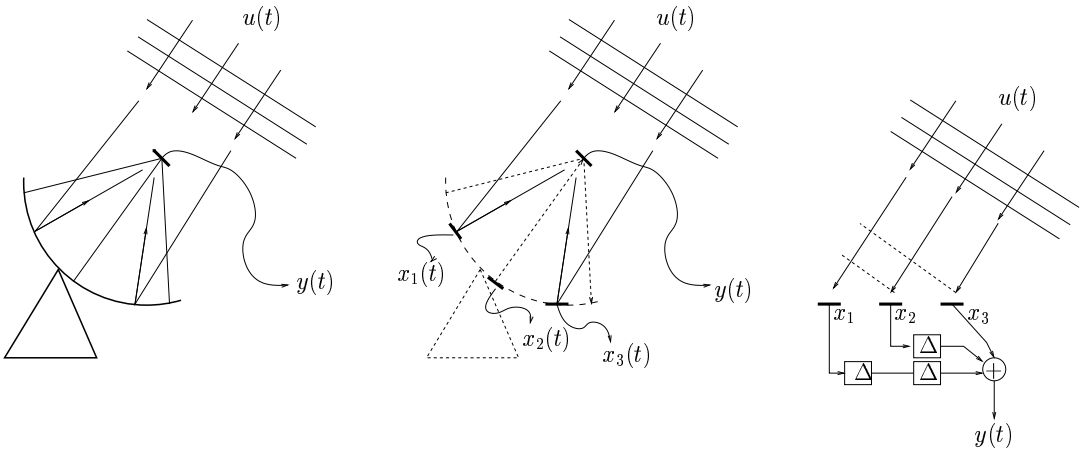


Figure 1.3. Coherent adding of signals

signal at an antenna array consisting of multiple antennas. We only consider linear receivers at this point.

1.2.1 Introduction

An antenna array may be employed for several reasons. A traditional one is *signal enhancement*. If the same signal is received at multiple antennas and can be coherently added, then additive noise is averaged out. For example, suppose we have a signal $u(t)$ received at M antennas $x_1, \cdots, x_M$,

$$x_m(t) \;=\; u(t) + n_m(t)\,, \qquad m = 1, \cdots, M$$

where $u(t)$ is the desired signal and $n_m(t)$ is noise. Let us suppose that the noise variance is $\mathrm{E}[|n_m(t)|^2] = \sigma^2$. If the noise is uncorrelated from each antenna to the others, then by averaging we obtain

$$y(t) \;=\; \frac{1}{M}\sum_{m=1}^{M} x_m(t) \;=\; u(t) + \frac{1}{M}\sum_{m=1}^{M} n_m(t) \;=\; u(t) + n(t)$$

The noise variance on $y(t)$ is given by $\mathrm{E}[|n(t)|^2] = \frac{1}{M}\sigma^2$. We thus see that there is an *array gain* equal to a factor M, the number of antennas.

The reason that we could simply average, or add up the received signals $x_m(t)$, is that the desired signal entered coherently, with the same delay at each antenna. More general, the desired signal is received at unequal delays and we have to introduce compensating delays to be able to coherently add them. This requires knowledge on these delays, or the direction at which the signal was received. The operation of delay-and-sum is known as *beamforming*, since it can be regarded as forming a beam into the direction of the source. The delay-and-sum beamformer acts like an equivalent of a parabolic dish, which physically inserts the correct delays to look in the desired direction. See figure 1.3.

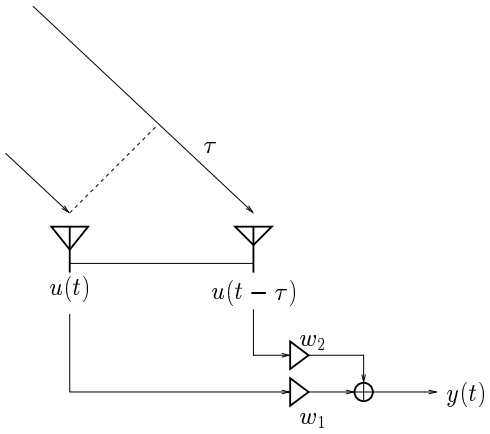


Figure 1.4. Nulling out a signal

A second reason to use an antenna array is to introduce a form of *spatial filtering*. Filtering can be done in the frequency domain —very familiar—, but similarly in the spatial domain. Spatial filtering can just be regarded as taking (often linear) combinations of the antenna outputs, and perhaps delays of them, to reach a desired spatial response.

A prime application of spatial filtering is *null steering*: the linear combinations are chosen such that a signal (interferer) is completely cancelled out. Suppose the first antenna receives the signal $u(t)$, whereas the second antenna receives the signal $u(t-\tau)$, see figure 1.4. It is easy to see how the received signals can be processed to produce a zero output, by inserting a proper delay and taking the difference. However, even without a delay we can do something. By weighting and adding the antenna outputs, we obtain a signal $y(t)$ at the output of the beamformer,

$$y(t) = w_1 u(t) + w_2 u(t-\tau)$$

In the frequency domain, this is

$$Y(\omega) = U(\omega)(w_1 + w_2 e^{-j\omega\tau})$$

Thus, we can make sure that the signal is cancelled, $Y(\omega) = 0$, at a certain frequency ω_0, if we select the weights such that

$$w_2 = -w_1 e^{j\omega_0\tau}$$

Note that (1) if we do not delay the antenna outputs but only scale them before adding, then we need complex weights, (2) with an implementation using weights, we can cancel the signal only at a specific frequency, but not at all frequencies.

Thus, for signals that consist of a single frequency, or are narrow band around a carrier frequency, we can do null steering by means of a *phased array*. In more general situations, with broadband signals, we need a beamformer structure consisting of weights *and* delays. See figure 1.5. How narrow is narrow-band depends on the maximal delay across the antenna array, as is discussed next.

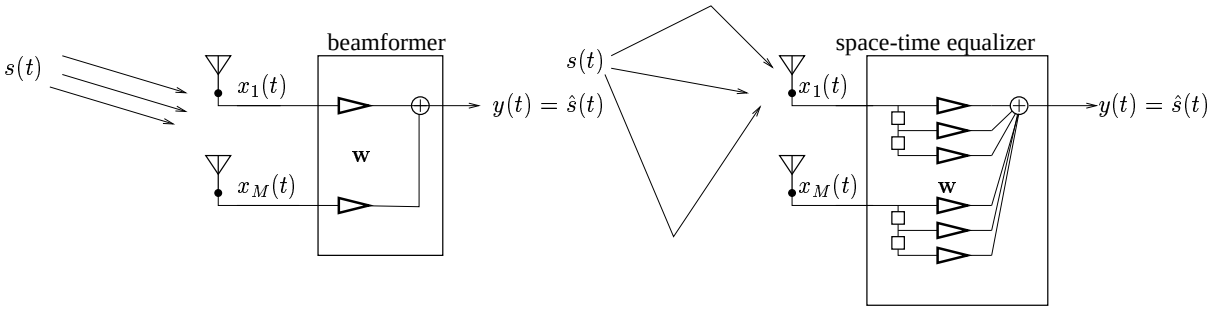


Figure 1.5. (a) Narrowband beamformer (spatial filter); (b) broadband beamformer (spatial-temporal filter)

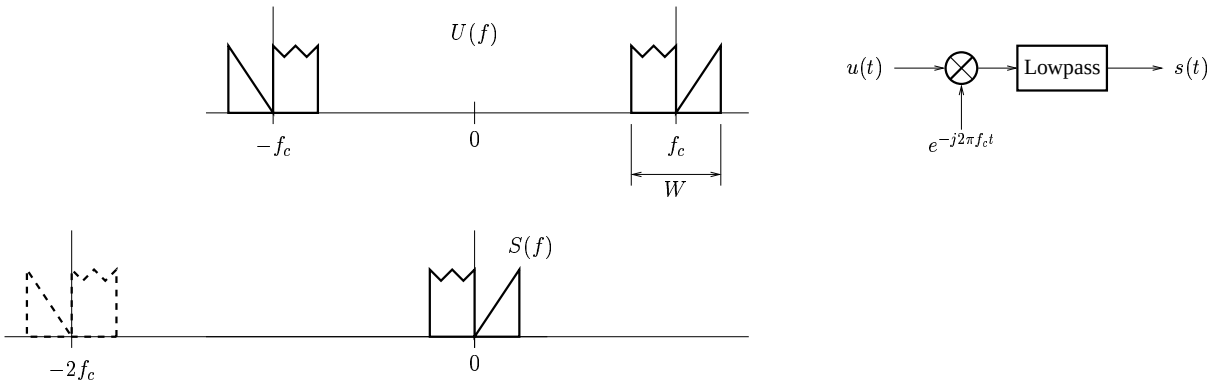


Figure 1.6. Transmitted real signal $u(t)$ and complex baseband signal $s(t)$.

1.2.2 Complex baseband signal representation

Let us recall the following facts. In signal processing, signals are usually represented by their lowpass equivalents, see e.g., [7]. This is a suitable representation for narrowband signals in a digital communication system. A real valued bandpass signal with center frequency f_c may be written as

$$u(t) = \text{real}\{s(t)e^{j2\pi f_c t}\} = x(t)\cos 2\pi f_c t - y(t)\sin 2\pi f_c t \tag{1.1}$$

where $s(t) = x(t) + jy(t)$ is the *complex envelope* of the signal $u(t)$, also called the *baseband signal*. The real and imaginary parts, $x(t)$ and $y(t)$, are called the in-phase and quadrature components of the signal $u(t)$. In practice, they are generated by multiplying the received signal with $\cos 2\pi f_c t$ and $\sin 2\pi f_c t$ followed by low-pass filtering. See figure 1.6. An alternative is to apply a Hilbert transformation.

Suppose that the bandpass signal $u(t)$ is delayed by a time τ. This can be written as

$$u_\tau(t) := u(t-\tau) = \text{real}\{s(t-\tau)e^{j2\pi f_c(t-\tau)}\} = \text{real}\{s(t-\tau)e^{-j2\pi f_c\tau}e^{j2\pi f_c t}\}\,.$$

The complex envelope of the delayed signal is thus $s_\tau(t) = s(t-\tau)e^{-j2\pi f_c\tau}$. Let W be the bandwidth of the complex envelope $s(t)$ and let $S(f)$ be its Fourier transform. We then have

$$s(t-\tau) = \int_{-W/2}^{W/2} S(f)e^{-j2\pi f\tau}e^{j2\pi ft}df$$

If $|2\pi f\tau| \ll 1$ for all frequencies $|f| \leq \frac{W}{2}$ we can approximate $e^{-j2\pi f\tau} \approx 1$ and get

$$s(t-\tau) \approx \int_{-W/2}^{W/2} S(f)e^{j2\pi ft}df = s(t)$$

Thus, we have for the complex envelope $s_\tau(t)$ of the delayed bandpass signal $u_\tau(t)$ that

$$s_\tau(t) \approx s(t)e^{-j2\pi f_c\tau} \qquad \text{for } W\tau \ll 1$$

The conclusion is that, for narrowband signals, time delays smaller than the inverse bandwidth may be represented as phase shifts of the complex envelope. This is fundamental in direction estimation using phased antenna arrays.

For propagation across an antenna array, the maximal delay depends on the maximal distance across the antenna array — the aperture, which is usually denoted in wavelengths related to the carrier frequency f_c. If the wavelength is $\lambda = c/f_c$ and the aperture is Δ wavelengths, then the maximal delay is $\tau = \Delta\lambda/c = \Delta/f_c$. In this context, narrowband means $W\Delta \ll f_c$. For mobile communications, the wavelength for $f_c = 1$ GHz is about 30 cm. For practical purposes, Δ is small, say $\Delta < 5$ wavelengths, and narrow band means $W \ll 200$ MHz. This condition is always satisfied.

For outdoor multipath propagation, the situation may be different. If there is a reflection on a distant object, there may be path length differences of a few km, or $\Delta = \mathcal{O}(1000)$ at 1 GHz. In this context, narrow band means $W \ll 1$ MHz, and for example for GSM this is not really satisfied. In this case, delays of the signal cannot be represented by mere phase shifts, and we need to do broadband beamforming.

1.2.3 Antenna array response

Consider an array consisting of M antenna elements placed along a line in space, also known as a linear array. See figure 1.7. For narrowband signals with carrier frequency f_c, the corresponding wavelength λ is given by $\lambda = c/f_c$ where c is the speed of propagation. The distance of the i-th element to the first is denoted by δ_i, or $\Delta_i = \delta_i/\lambda$ wavelengths.

Suppose that a point source is present in the far field. Let $s_0(t)$ be the transmitted baseband signal. If the distance between the source and the array is large enough in comparison to the extent of the array, the wave incident on the array is approximately planar. The received plane wave can then be charactarized by an attenuation A and a direction of arrival θ. Let $a_i(\theta)$ be the gain pattern of the i-th antenna element. The signal received at the i-th antenna then is

$$x_i(t) = a_i(\theta)As_0(t-T_i)e^{-j2\pi f_cT_i} \tag{1.2}$$

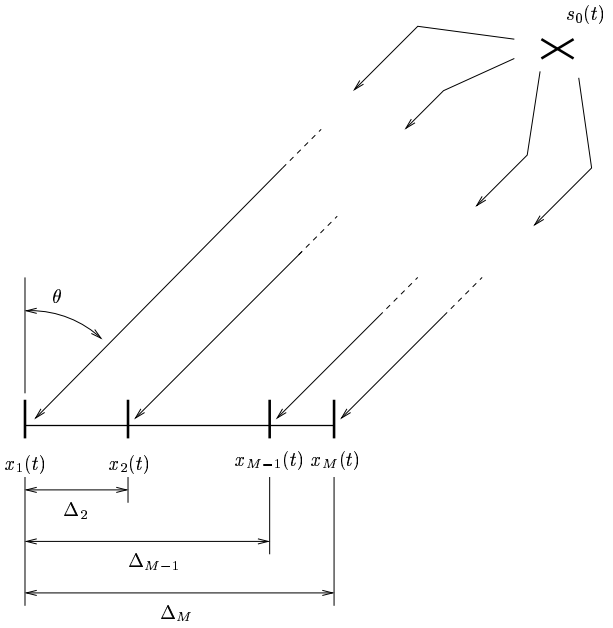


Figure 1.7. A linear array receiving a far field point source

where T_i is the time it takes the signal to travel from the source to the i-th element of the array. Generally, an array with identical elements is assumed. In that case, all antennas have the same gain pattern: $a_i(\theta) = a(\theta)$. Defining $s(t) = s_0(t - T_1)$, $\tau_i = T_i - T_1$, and $\beta = Ae^{-j2\pi f_c T_1}$, we can then write (1.2) as

$$x_i(t) = a(\theta)\beta s(t - \tau_i)e^{-j2\pi f_c \tau_i} \tag{1.3}$$

Note that the relation between τ_i and θ is given by

$$2\pi f_c \tau_i = -2\pi f_c \frac{\delta_i \sin(\theta)}{c} = -2\pi \frac{\delta_i}{\lambda} \sin(\theta) = -2\pi \Delta_i \sin(\theta)$$

If the τ_i are small compared to the inverse bandwidth of $s(t)$, we may set $s_{\tau_i}(t) := s(t - \tau_i)e^{-j2\pi f_c \tau_i} = s(t)e^{-j2\pi f_c \tau_i} = s(t)e^{j2\pi \Delta_i \sin(\theta)}$. Collecting the signals received by the individual elements into a vector $\mathbf{x}(t)$, we obtain from (1.3)

$$\mathbf{x}(t) = a(\theta)\beta \begin{bmatrix} s_{\tau_1}(t) \\ s_{\tau_2}(t) \\ \vdots \\ s_{\tau_M}(t) \end{bmatrix} = \begin{bmatrix} 1 \\ e^{j2\pi \Delta_2 \sin(\theta)} \\ \vdots \\ e^{j2\pi \Delta_M \sin(\theta)} \end{bmatrix} a(\theta)\beta s(t) := \mathbf{a}(\theta)\beta s(t)$$

where the *array response vector* $\mathbf{a}(\theta)$ is the response of the array to a planar wave with DOA θ. The *array manifold* $\mathcal{A}$ is the curve that $\mathbf{a}(\theta)$ describes in the M-dimensional complex vector

space $\mathbb{C}^M$ when θ is varied over the domain of interest:

$$\mathcal{A} = \{\mathbf{a}(\theta) : 0 \leq \theta < 2\pi\}$$

For a uniform linear array, we have that all distances between two consecutive array elements are the same, so that $\Delta_i = (i-1)\Delta$. In this case, the array response vector can be written as

$$\mathbf{a}(\theta) = \begin{bmatrix} 1 \\ e^{j2\pi\Delta\sin(\theta)} \\ \vdots \\ e^{j2\pi(M-1)\Delta\sin(\theta)} \end{bmatrix} a(\theta)$$

In many algorithms, the common factor (gain pattern) $a(\theta)$ does not play a role and is often omitted: each antenna is assumed to have a gain of one in all directions (we assume "omnidirectional" and "normalized" antennas), although this assumption is not always necessary.

1.2.4 Array manifold and parametric direction finding

Data models of the form

$$\mathbf{x}(t) = \mathbf{a}(\theta)\beta s(t)$$

play an important role throughout this book. Note that for varying source samples $s(t)$, the data vector $\mathbf{x}(t)$ is only scaled in length, but its direction $\mathbf{a}(\theta)$ is constant. Thus, $\mathbf{x}(t)$ is confined to a line. If we know the array manifold, i.e., the function $\mathbf{a}(\theta)$, then we can determine θ by intersecting the line with the curve traced by $\mathbf{a}(\theta)$ for varying θ, or "fitting" the best $\mathbf{a}(\theta)$ to the direction of the $\mathbf{x}(t)$, see figure 1.8.

For two sources, the data model becomes a superposition,

$$\mathbf{x}(t) = \mathbf{a}(\theta_1)\beta_1 s_1(t) \; + \; \mathbf{a}(\theta_2)\beta_2 s_2(t) = [\mathbf{a}(\theta_1) \quad \mathbf{a}(\theta_2)] \begin{bmatrix} \beta_1 & \\ & \beta_2 \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \end{bmatrix}$$

or

$$\mathbf{x}(t) = \mathbf{A}_\theta \mathbf{B}\mathbf{s}(t)\,, \quad \mathbf{A}_\theta = [\mathbf{a}(\theta_1) \quad \mathbf{a}(\theta_2)]\,, \quad \mathbf{B} = \begin{bmatrix} \beta_1 & \\ & \beta_2 \end{bmatrix}, \quad \mathbf{s}(t) = \begin{bmatrix} s_1(t) \\ s_2(t) \end{bmatrix}$$

When $s_1(t)$ and $s_2(t)$ both vary with t, $\mathbf{x}(t)$ is confined to a *plane*. Direction finding now amounts to intersecting this plane with the array manifold, see figure 1.8.

Considering multipath propagation, as will be explained in section 1.3.1, we actually obtain a linear combination of the same source via different paths. If the relative delays between the different paths are small compared to the inverse bandwidth, the delays can be represented by phase shifts. Thus, the data model is

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{a}(\theta_1)\beta_1 s(t) \; + \; \mathbf{a}(\theta_2)\beta_2 s(t) \; + \; \ldots \; + \; \mathbf{a}(\theta_r)\beta_r s(t) \\ &= \{\beta_1\mathbf{a}(\theta_1) \; + \; \beta_2\mathbf{a}(\theta_2) \; + \; \ldots \; + \; \beta_r\mathbf{a}(\theta_r)\}\, s(t) \quad = \; \mathbf{a}\, s(t) \end{aligned} \tag{1.4}$$

In this case, the combined vector $\mathbf{a}$ is **not** on the array manifold and direction finding is more complicated. At any rate, $\mathbf{x}(t)$ contains an *instantaneous multiple* $\mathbf{a}$ of $s(t)$.

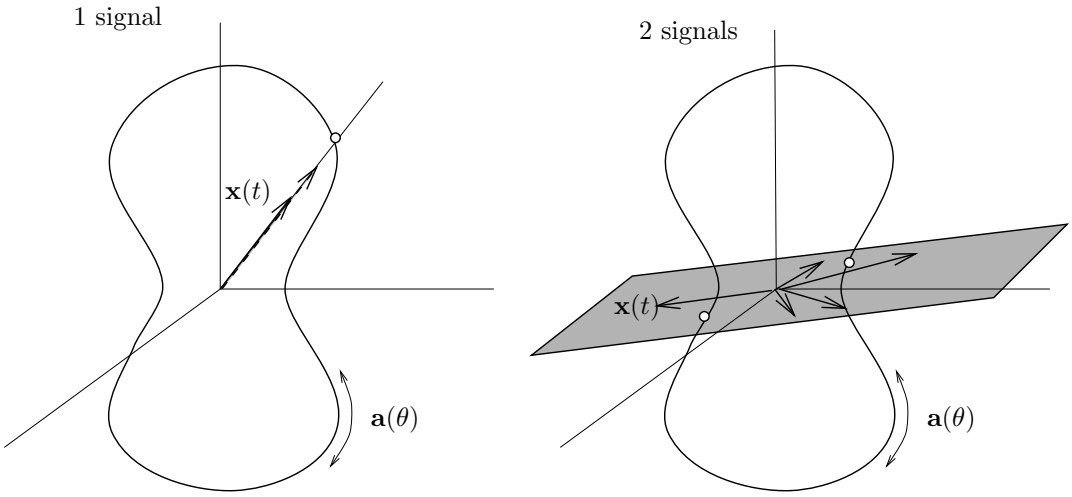


Figure 1.8. Direction finding means intersecting the array manifold with the line or plane spanned by the antenna output vectors

1.2.5 Beamforming and source separation

With two narrowband sources and multipath, we receive a linear mixture of these sources,

$$\mathbf{x}(t) = \mathbf{a}_1 s_1(t) + \mathbf{a}_2 s_2(t) = \mathbf{A}\mathbf{s}(t)$$

The objective of *source separation* is to estimate beamformers $\mathbf{w}_1$ and $\mathbf{w}_2$ to separate and recover the individual sources:

$$y_1(t) = \mathbf{w}_1^{\mathrm{H}}\mathbf{x}(t) = \hat{s}_1(t)\,, \qquad y_2(t) = \mathbf{w}_2^{\mathrm{H}}\mathbf{x}(t) = \hat{s}_2(t)\,,$$

or in matrix form, with $\mathbf{W} = [\mathbf{w}_1 \quad \mathbf{w}_2]$,

$$\mathbf{W}^{\mathrm{H}}\mathbf{x}(t) = \mathbf{s}(t) \qquad \Leftrightarrow \qquad \mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I}$$

Thus, we have to obtain an estimate of the mixing matrix $\mathbf{A}$ and invert it to separate the sources. There are several ways to estimate $\mathbf{A}$. One method we have seen before: if there is no multipath, then

$$\mathbf{A} = [\mathbf{a}(\theta_1) \quad \mathbf{a}(\theta_2)] \begin{bmatrix} \beta_1 & \\ & \beta_2 \end{bmatrix} = \mathbf{A}_\theta \mathbf{B}$$

By estimating the directions of the sources, we find estimates of θ_1 and θ_2, and hence $\mathbf{A}_\theta$ becomes known. $\mathbf{B}$ is not known, but its estimation can be omitted by employing differential modulation, see section 1.4.

In other situations, in wireless communications, we may know the values of $s_1(t)$ and $s_2(t)$ for a short time interval $t = [0, N)$, the data contains a "training period". We thus have a data

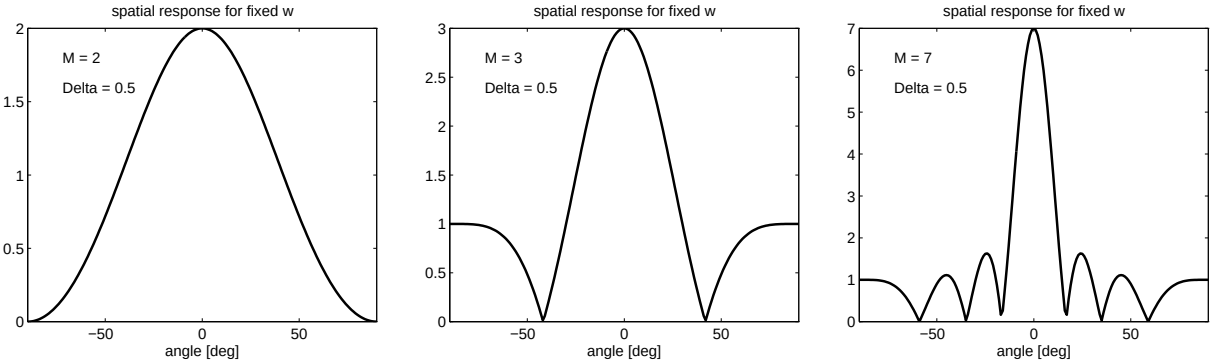


Figure 1.9. Spatial responses to a beamformer $\mathbf{w} = [1, \cdots, 1]^T$.

model

$$\mathbf{X} = \mathbf{AS}\,, \qquad \mathbf{X} = [\mathbf{x}(0)\,, \cdots, \mathbf{x}(N-1)]\,, \qquad \mathbf{S} = [\mathbf{s}(0)\,, \cdots, \mathbf{s}(N-1)]\,.$$

This allows to set up a least squares problem

$$\min_{\mathbf{A}} \; \| \mathbf{X} - \mathbf{AS} \|_F^2$$

with $\mathbf{X}$ and $\mathbf{S}$ known. The solution is given by

$$\mathbf{A} = \mathbf{X}\mathbf{S}^{\mathrm{H}}(\mathbf{S}\mathbf{S}^{\mathrm{H}})^{-1}$$

and subsequently $\mathbf{W} = \mathbf{A}^{-H}$.

1.2.6 Array response graph

Let us know consider in some more detail the properties of the array response vector $\mathbf{a}(\theta)$. For simplicity, we will look at uniform linear arrays. Suppose we have an antenna spacing of $\lambda/2$, and select a simple beamformer of the form

$$\mathbf{w} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

i.e., we simply sum the outputs of the antennas. The response of the array to a unit-amplitude signal ($|\beta s(t)| = 1$) from direction θ is then characterized by

$$|y(t)| = |\mathbf{w}^{\mathrm{H}}\mathbf{a}(\theta)|$$

Graphs of this response for $M = 2,\ 3,\ 7$ antennas are shown in figure 1.9. Note that the response is maximal for a signal from the direction $0°$, or broadside from the array. This is natural since

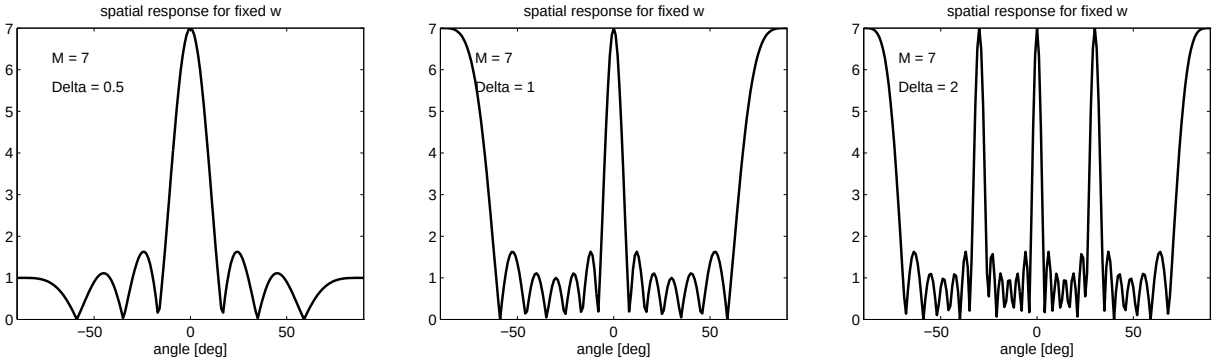


Figure 1.10. Grating lobes

a signal from this direction is summed coherently, as we have seen in the begining of the section. The gain in this direction is equal to M, the array gain. From all other directions, the signal is not summed coherently. For some directions, the response is even zero, where the delayed signals add destructively. The number of zeros is equal to $M-1$. In between the zeros, sidelobes occur. The width of the main beam is also related to the number of antennas, and is about $180^\circ/M$. With more antennas, the beamwidth gets smaller.

Ambiguity and grating lobes Let us now consider what happens if the antenna spacing increases beyond $\lambda/2$. We have an array response vector

$$\mathbf{a}(\theta) = \begin{bmatrix} 1 \\ \phi \\ \vdots \\ \phi^{M-1} \end{bmatrix}, \qquad \phi = e^{j2\pi\Delta\sin(\theta)}$$

Since $\sin(\theta) \in [-1, 1]$, we have that $2\pi\Delta\sin(\theta) \in [-2\pi\Delta, 2\pi\Delta]$. If $\Delta > 1/2$, then this interval extends beyond $[-\pi, \pi]$. In that case, there are several values of θ that give rise to the same argument of the exponent, or to the same ϕ. The effect is two-fold:

- *spatial aliasing* occurs: we cannot recover θ from knowledge of ϕ, and
- in the array reponse graph, *grating lobes* occur, see figure 1.10. This is because coherent addition is now possible for several values of θ.

Grating lobes prevent a unique estimation of θ. However, we can still estimate $\mathbf{A}$ and it does not prevent the possibility of null steering or source separation. Sometimes, grating lobes can be suppressed by using directional antennas rather than omnidirectional ones: the spatial response is then multiplied with the directional response of the antenna and if it is sufficiently narrow, only a single lobe is left.

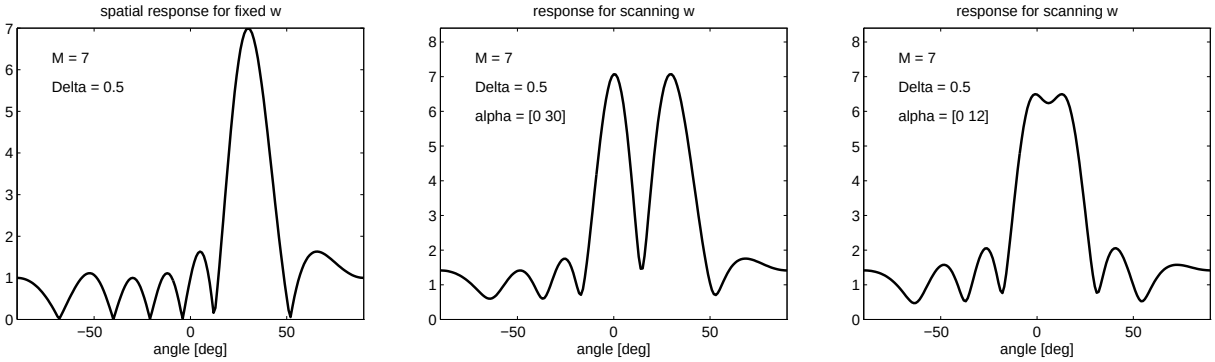


Figure 1.11. Beam steering. (a) response to $\mathbf{w} = \mathbf{a}(30°)$; ($b$) response for scanning $\mathbf{w} = \mathbf{a}(\theta)$, in a scenario with two sources, well separated, and (c) separated less than a beam width.

Beam steering Finally, let us consider what happens when we change the beamforming vector $\mathbf{w}$. Although we are free to choose anything, let us choose a structured vector, e.g., $\mathbf{w} = \mathbf{a}(30°)$. Figure 1.11 shows the response to this beamformer. Note that now the main peak shifts to 30°, signals from this direction are coherently added. By scanning $\mathbf{w} = \mathbf{a}(\theta)$, we can place the peak at any desired θ. This is called *classical beamforming*.

This also provides a simple way to do direction estimation. Suppose we have a single unit-amplitude signal, arriving from broadside (0°),

$$\mathbf{x}(t) = \mathbf{a}(0)\beta s(t) = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \beta s(t) = \mathbf{1}\beta s(t), \quad |\beta s(t)| = 1$$

If we compute $y(t) = \mathbf{w}^{\mathrm{H}}\mathbf{x}(t)$ and scan $\mathbf{w} = \mathbf{a}(\theta)$ over all values of theta and monitor the output power of the beamformer,

$$|y(t)| = |\mathbf{w}^{\mathrm{H}}\mathbf{x}(t)| = |\mathbf{a}(\theta)^{\mathrm{H}}\mathbf{1}|, \qquad -\pi \leq \theta \leq \pi$$

then we obtain essentially the same array graph as in 1.9 before (it is the same functional). Thus, there will be a main peak at 0°, the direction of arrival, and the beam width is related to the number of antennas. In general, if the source is coming from direction θ_0, then the graph will have a peak at θ_0.

If we consider two unit-amplitude signals: $\mathbf{x}(t) = \mathbf{a}(\theta_1)\beta_1 s_1(t) + \mathbf{a}(\theta_2)\beta_2 s_2(t)$, with $|\beta_1 s_1(t)| = 1$ and $|\beta_2 s_2(t)| = 1$, the array graph will show two peaks, at θ_1 and θ_2, at least if the two sources are well separated. If the sources are close, then the two peaks will merge and at some point we will not recognize that there are in fact two sources.

The choice $\mathbf{w} = \mathbf{a}(\theta)$ is one of the simplest forms of beamforming.[3] It is data independent and

[3]It is known as maximum ratio combining or matched filtering in communications.

non-parametric (i.e., the explicit structure of $\mathbf{a}(\theta)$ is not exploited), and optimal only for a single source in white noise. One can show that for more than 1 source, the parameter estimates for the directions θ_i will be biased: the peaks have a tendency to move a little bit to each other. Unbiased estimates are obtained only for a single source.

There are other ways of beamforming, in which the beamformer is selected depending on the data, with higher resolution (sharper peaks) and better statistical properties in the presence of noise. Alternatively, we may follow a parametric approach in which we pose the model $\mathbf{x}(t) = \mathbf{a}(\theta_1)\beta_1 s_1(t) + \mathbf{a}(\theta_2)\beta_2 s_2(t)$ and try to compute the parameters θ_1 and θ_2 that best fit the observed data, as we discussed in section 1.2.4.

1.3 PHYSICAL CHANNEL PROPERTIES[4]

1.3.1 Wide-area multipath propagation model

The propagation of signals through the wireless channel is fairly complicated to model. A correct treatment would require a complete description of the physical environment and involve Maxwells equations, and would not be very useful for the design of signal processing algorithms. To arrive at a more useful parametric model, we have to make simplifying assumptions regarding the wave propagation. Provided this model is reasonably valid, we can, in a second stage, try to derive statistical models for the parameters to obtain reasonable agreement with measurements.

The number of parameters in an accurate model can be quite high, and from a signal processing point of view, they might not be very well identifiable. For this reason, another model used in signal processing is a much less sophisticated *unparametrized* model. The radio channel is simply modeled as an FIR (finite impulse response) filter, with main parameters the impulse response length (in symbols) and the total attenuation or signal-to-noise ratio (SNR). This model is described in section 1.5. The parametrized model is a special case, giving structure to the FIR coefficients.

Jakes' model A commonly used parametric model is a multiray scattering model, also known as Jakes' model (after Jakes [2], see also [1, 3, 10–12]). In this model, the signal follows on its way from the source to the receiver a number of distinct paths, referred to as multipaths. These arise from scattering, reflection, or diffraction of the radiated energy on objects that lie in the environment. The received signal from each path is much weaker than the transmitted signal due to various scattering and fading effects. Multipath propagation also results in the spreading of the signal in various dimensions: delay spread in time, Doppler spread in frequency, and angle spread in space. Each of them has a significant effect on the signal. The mean path loss, shadowing, fast fading, delay, Doppler spread and angle spread are the main channel characteristics and form the parameters of the multiray model.

[4] Adapted from Paulraj and Papadias [1], B. Ottersten [8], and D. Aztely [9].

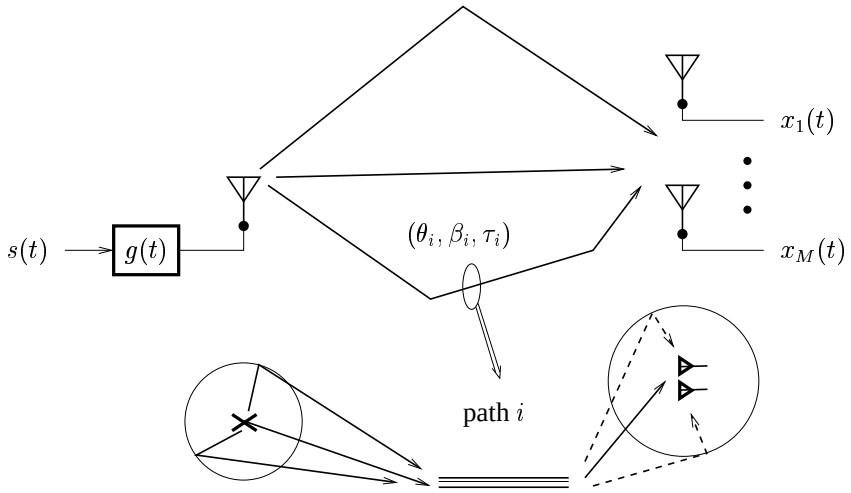


Figure 1.12. Multipath propagation model

The scattering of the signal in the environment can be specialized into three stages: scattering local to the source at surrounding objects, reflections on distant objects of the few dominant rays that emerge out of the local clutter, and scattering local to the receiver. See figure 1.12.

Scatterers local to the mobile Scattering local to the mobile is caused by buildings and other objects in the direct vicinity of the mobile (at, say, a few tens of meters). Motion of the mobile and local scattering give rise to Doppler spread which causes "time-selective fading": the signal power can have significant fluctuations over time. While local scattering contributes to Doppler spread, the delay spread will usually be insignificant because of the small scattering radius. Likewise, the angle spread will also be small.

Remote scatterers Away from the cluster of local scatterers, the emerging wavefronts may then travel directly to the base or may be scattered toward the base by remote *dominant scatterers*, giving rise to specular multipath. These remote scatterers can be either terrain features (distant hills) or high rise building complexes. Remote scattering can cause significant delay and angle spreads.

Scatterers local to the base Once these multiple wavefronts reach the base station, they may be scattered further by local structures such as buildings or other structures that are in the vicinity of the base. Such scattering will be more pronounced for low elevation and below-roof-top antennas. The scattering local to the base can cause significant angle spread which can cause space-selective fading: different antennas at the base station can receive totally different signal

powers. This fading is time invariant, unlike the time varying space-selective fading caused by remote scattering.

The forward link channel is effected in similar ways by these scatterers, but in a reverse order.

Model overview Let us ignore the local scattering for the moment, and assume that there are r rays bouncing off remote objects such as hills or tall buildings. The received parametric signal model is then usually written as the convolution

$$\mathbf{x}(t) = \left[\sum_{i=1}^{r} \mathbf{a}(\theta_i)\beta_i g(t-\tau_i)\right] * s(t) \tag{1.5}$$

where $\mathbf{x}(t)$ is a vector consisting of the M antenna outputs, $\mathbf{a}(\theta)$ is the array response vector, and the impulse response $g(t)$ collects all temporal aspects, such as pulse shaping and transmit and receive filtering. In the previous models, $g(t)$ was lumped into $s(t)$, whereas in this more complete model, we explicitly bring out the temporal filtering effects. The model parameters of each ray are its (mean) direction of arrival θ_i, (mean) delay τ_i, and complex gain β_i. The latter parameter lumps the overall attenuation and all phase shifts.

Each of the rays is itself composed of a large number of "mini-rays" due to scattering local to the mobile: all with roughly equal angles and delays, but arbitrary phases. This can be described by extending the model with additional parameters such as the standard deviations from the mean direction of arrivel θ_i and mean delay τ_i, which depend on the radius (aspect ratio) of the scattering region and its distance to the remote scattering object [8, 13]. For macroscopic models, the standard deviations are generally small (less than a few degrees, and a fraction of τ_i) and are usually but not always ignored. The scattering local to the mobile has however a major effect on the statistics and stationarity of β_i, as will be discussed later on.

Sattering local to the base will complicate the model a lot, and will make us loose the identifiability of the directions in which the rays are entering at the base station array. However, the receiving antennas at the base station are usually placed high above the ground so that the effects of waves scattered in the vicinity of the receiving antennas can be neglected.

1.3.2 Fast fading

Let us focus on a single ray of the multipath model, and omit the subscript i. The complex gain β of this ray can actually be decoupled into three factors. The first factor is the path loss, which is due to the fact that we loose energy when the signal travels from the mobile to the base station. The second factor is the slow or long term fading, which is due to slowly varying shadowing effects of buildings or natural features. The third and final factor is the fast or short term fading, which we focus on in this section. It refers to the rapid fluctuations of a received multipath ray and is due to the scattering of the signal by a cluster of objects near the moving source. The assumption is that each ray in the multipath model is the sum of a large number of locally scattered wavefronts, with randomized directions of arrival, i.e., with phases uniformly

distributed in $[0, 2\pi)$. As a result, the in-phase and quadrature phase components of the vertical electrical field are Gaussian processes.

Remember that the expression of the complex gain for a single mini-ray is given by $Ae^{-j2\pi f_c T}$, where A is the attenuation and T is the propagation delay. Each complex gain β in (1.5) is the result of many such mini-rays, and can therefore be written as

$$\beta = \sum_{n=1}^{N} A_n e^{-j2\pi f_c T_n},$$

where A_n and T_n are respectively the attenuation and the delay related to the n-th mini-ray of the considered ray. If the relative propagation delays are approximately independent and uniformly distributed over an interval ΔT defined as

$$\Delta T = \max_n T_n - \min_n T_n$$

it is reasonable to assume that the phases ϕ_n defined as $\phi_n = 2\pi f_c T_n \bmod 2\pi$ are random variables independent and uniformly distributed in the interval $[0, 2\pi)$. The A_n's are usually assumed to be i.i.d. as well. Writing β as $\beta = a + jb$ it is then clear that both a and b are the sums of N i.i.d. random variables. From the central limit theorem we can therefore conclude that they are asymptotically Gaussian distributed. Thus, the complex gain β is approximately complex Gaussian distributed,

$$\beta \sim \mathcal{CN}(0, \sigma_\beta^2) \qquad \Leftrightarrow \qquad p(\beta) = \frac{1}{\sqrt{2\pi}\sigma_\beta} e^{-\frac{|\beta|^2}{\sigma_\beta^2}}$$

where the variance σ_β^2 is given by

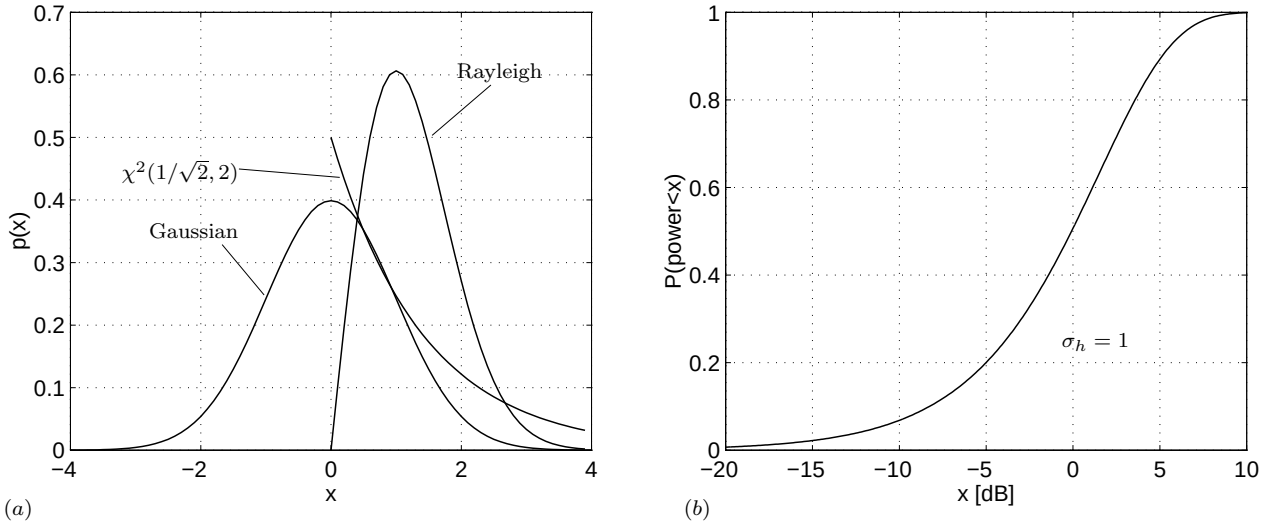
$$\sigma_\beta^2 = \mathrm{E}[|\beta|^2] = N\mathrm{E}[|A_n|^2]$$

The absolute value (amplitude) of a complex random number with zero mean i.i.d. Gaussian distributed real and imaginary parts has a distribution known as the Rayleigh density function [2]: let $r = |beta|$, then

$$p(r) \;=\; \frac{r}{\sigma^2} e^{-r^2/2\sigma^2}, \qquad r \geq 0 \tag{1.6}$$

where σ is the standard deviation of each component. The distribution has mean $\sigma\sqrt{\pi/2}$ and variance $(2 - \pi/2)\sigma^2$. The distribution is plotted in figure 1.13(a)). Essential for the derivation of this Rayleigh model is that we have a large number of i.i.d. waves, with relative time delays (i.e., delay spread) small enough to model them as phase shifts.

The sum of two squared Gaussian variables has a Chi-square distribution with two degrees of freedom. This is the distribution of $|\beta|^2 = a^2 + b^2$, which is the power of the resulting ray (mean is σ_β^2, standard deviation is also σ_β^2). The corresponding cumulative density function is shown in figure 1.13(b), and specifies the probability that the power of the ray is smaller than the ordinate. Fades of 10 dB or more are not unlikely!



(a) (b)

Figure 1.13. (*a*) Zero mean Gaussian, and Rayleigh probability distributions, for $\sigma = 1$, (*b*) the probability that $|\beta|^2 < x$ is given by the CDF of $\chi^2(\sigma_\beta/\sqrt{2}, 2)$.

Note that β is actually time-varying: if the source is in motion, then the Doppler shifts and/or the varying location change the phase differences among the mini-rays, so that the sum can be totally different from one time instant to the next. The maximal Doppler shift f_D is given by the speed of the source (in m/s) divided by the wavelength of the carrier. The *coherence time* of the channel is inversely proportional to f_D, roughly by a factor of 0.2: β can be considered approximately constant for time intervals smaller than this time [3, 14, 15]. Angles and delays are generally assumed to be stationary over much longer periods. For simplicity, we will generally ignore the time selectivity.

1.3.3 Multiray parameters

Now that we have seen the properties of a single ray, it remains to describe how the collection of rays adds up at a receiving antenna.

Delay spread and frequency selective fading Due to the multipath propagation, each ray experiences a different time delay, so that several time-shifted and scaled copies of the transmitted signal will arrive at the receiver. This spreading in time is called *delay spread*. Precise models are hard to give as they strongly depend on the type of surroundings, but typically, a double negative exponential distribution is observed: the delay separation between paths increases negative exponentially with path delay, and the path amplitudes also fall off negative exponentially with delay.

The delay spread determines the maximal symbol rate for which no equalization is required. The

Table 1.2. Typical delay, angle and Doppler spreads in cellular applications.

Environment	delay spread	angle spread	Doppler spread
Flat rural (macro)	0.5 μs	1°	190 Hz
Urban (macro)	5 μs	20°	120 Hz
Hilly (macro)	20 μs	30°	190 Hz
Mall (micro)	0.3 μs	120°	10 Hz
Indoors (pico)	0.1 μs	360°	5 Hz

inverse of the delay spread is proportional to the *coherence bandwidth* [2, 3, 15]: the distance $|f_1 - f_2|$ in frequency domain over which $|X(f_1)|$ is significantly correlated to $|X(f_2)|$. Narrowband signals with a bandwidth sufficiently smaller than the inverse of the delay spread experience a flat channel (in the frequency domain) that does not require equalization: $g(t)$ is essentially a scalar and can be lumped with β_i.

For a delay spread of $\tau = 1\,\mu$s (a typical value), the coherence bandwidth is approximately $1/2\pi\tau = 150$ kHz [2, ch.1]. The larger the delay spread, the smaller the coherence bandwidth.

If there is only a single dominant ray, or if there are a few rays but with approximately the same delays (as compared to the inverse of the signal bandwidth), then there is no delay spread and the channel frequency response is flat. Because of the scattering local to the source, Rayleigh fading can still occur, but will affect all frequencies equally. Such a scenario is known as a "flat fading channel".

Angle spread and space selective fading *Angle spread* refers to the spread in arrival angles of the different rays at the antenna array. The inverse of the angle spread determines the *coherence distance*, which gives an indication of the minimal distance by which two antennas have to be spaced to enable separation of two disparate rays within this spread by (classical) spatial separation techniques.

Angle spreads are strongly dependent on the geometry of the environment and have not been studied as thoroughly yet as delay spreads. Current research suggests that most outdoor channels can be modeled adequately by a small number of dominant rays and that in open or suburban terrain, most energy is often concentrated in only a single ray in the direction of the mobile [22]. Moreover, multiple rays usually have widely separated angles.

Typical channel parameters Angle spread, delay spread, and Doppler spread are important characterizations of a mobile channel, as it determines the amount of equalization that is required, but also the amount of diversity that can be obtained. Measurements in macrocells indicate that up to 6 to 12 dominant paths may be present. Typical channel delay and Doppler spreads (1800 MHz) are given in table 1.2 [3, 14] (see also references in [12]). Typical angle spreads are not well known; the given values are suggested by [1].

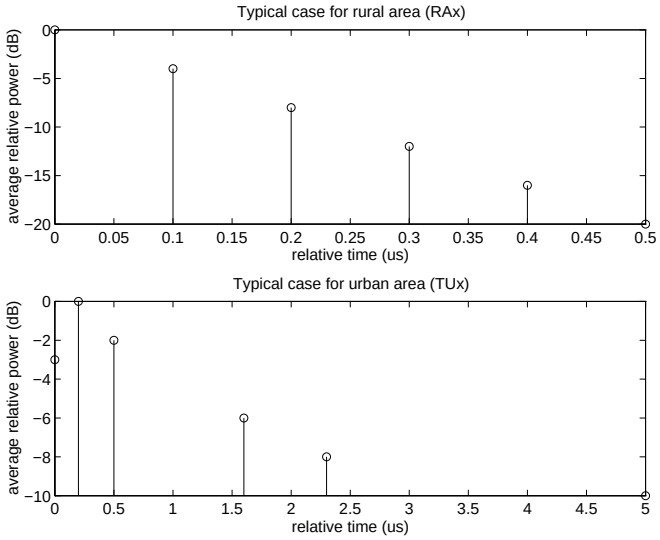


Figure 1.14. GSM specified "propagation models". The taps are considered to be independent Rayleigh distributed random variables, except for the first tap of the rural area model which is Rice distributed.

The European Telecommunication Standard Institute (ETSI) has in [23] specified some typical discretized impulse responses to be used in simulations. A conventional single receiving antenna is assumed, so angle spread is not specified. Figure 1.14 shows two such impulse response. The validity of these channel models is under discussion. E.g., angle spread and directivity is not part of the specification, and the relative time delays should of course be randomized as well. To put the given delays into perspective, note that the data transmission rate in GSM is 270 kb/s, so that the duration of each bit is $1/270000 \approx 3.7\,\mu$s.

Several measurements with sine-waves have been done to estimate the geometrical properties of the local scatterers [10, 11, 16, 18]. Since the bandwidth of a sinewave is zero, no temporal aspects could be extracted from these experiments, and it is not possible to tell whether different rays come from local scatterers or from other ray paths corresponding to much larger time delays. Not many measurements have been carried out so far that perform a joint estimation of both time delay and directions of arrival.

One case in which the spatial channel model has been validated against experimental data collected by Ericsson Radio Systems is described in [8]. In the field experiments, a transmitter has been placed in urban areas with non line of sight approximately 1 km from the receiving array which was elevated 30 meters above the ground [24]. The data has been processed to gain insight into propagation effects as well as into the behavior of some receiving algorithms. In [19, 25] the angular spread is found to be between 2° and 6° in the experiments for a transmitter placed 1 km from the receiving array. In figure 1.15, the estimated directions and angular spreads along with their associated standard deviations are displayed for a number of trials at one location.

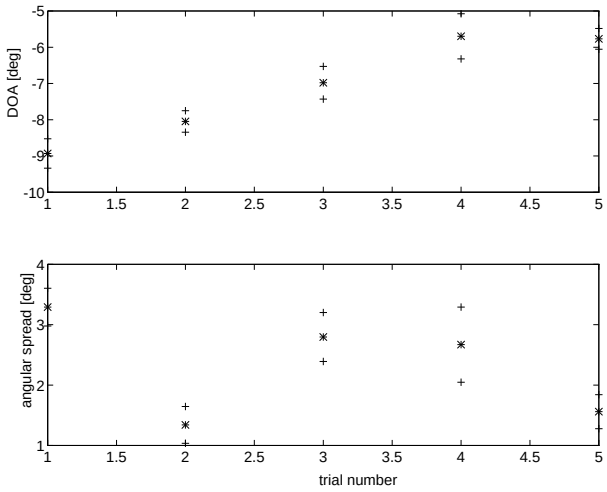


Figure 1.15. Estimated directions and angular spreads for 5 different trials in an experiment by Ericsson

1.4 SIGNAL MODULATION

Before a digital bit sequence can be transmitted over a radio channel, it has to be prepared: among other things, it has to be transformed into an analog signal in continuous time and modulated onto a carrier frequency. The various steps are shown in figure 1.16. The *coding* step, in its simplest form, maps the binary sequence $\{b_k\} \in \{0, 1\}$ into a sequence $\{s_k\}$ with another alphabet, such as $\{-1, +1\}$. A digital filter may be part of the coder as well. In linear modulation schemes, the resulting sequence is then convolved with a pulse shape function $p(t)$, whereas in phase modulation, it is convolved with some other pulse shape function $q(t)$ to yield the phase of the modulated signal. The resulting baseband signal $s(t)$ is modulated by the carrier frequency f_c to produce the RF signal that will be broadcast.

In this section, a few examples of coding alphabets and pulse shape functions are presented, for future reference. We do not go into the properties and reasons why certain modulation schemes are chosen; see e.g., [4] for more details.

1.4.1 Digital alphabets

The first step in the modulation process is the coding of the binary sequence $\{b_k\}$ into some other sequence $\{s_k\}$. The $\{s_k\}$ are chosen from an *alphabet* or *constellation*, which might be real or complex. There are many possibilities; common examples are BPSK (binary phase shift keying), QPSK (quadrature phase shift keying), m-PSK, m-PAM (pulse amplitude modulation), m-QAM (quadrature amplitude modulation), defined as in table 1.3. See also figure 1.17. Smaller constellations are more robust in the presence of noise, because of the larger distance between

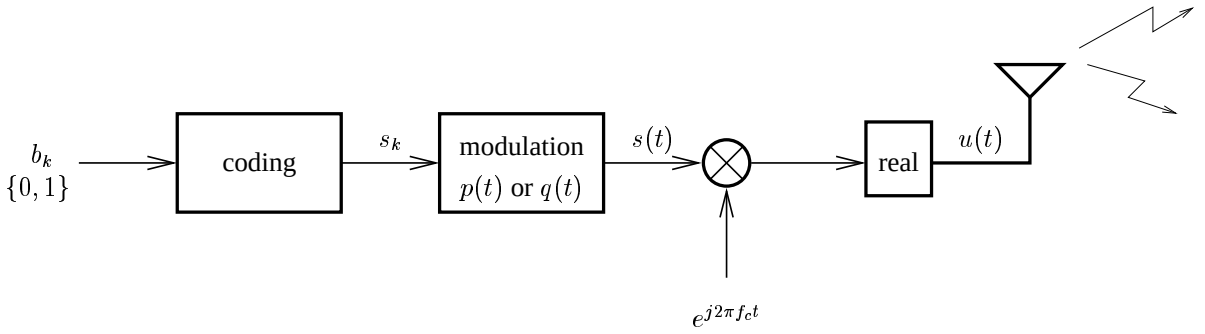


Figure 1.16. Modulation process

Table 1.3. Common digital constellations

	s_k chosen from (up to a possible scaling):
BPSK	$\{1, -1\}$
QPSK	$\{1, j, -1, -j\}$
m-PSK	$\{1, e^{j2\pi/m}, e^{j2\pi 2/m}, \ldots, e^{j2\pi(m-1)/m}\}$
m-PAM	$\{\pm 1, \pm 3, \ldots, \pm(m-1)\}$
m-QAM	$\{\pm 1 \pm j, \pm 1 \pm 3j, \ldots, \pm 1 \pm (\sqrt{m}-1)j, \pm 3 \pm j, \ldots, \pm(\sqrt{m}-1) \pm (\sqrt{m}-1)j\}$

the symbols. Larger constellations may lead to higher bitrates, but are harder to detect in noise.

A filtering operation might be part of the coder. Differential encoding is often used. In differential PSK (DPSK), for instance, the phase is shifted relative to the previous phase, e.g., over 0^o and 180^o for DBPSK and over 0^o, 90^o, 180^o, and 270^o for DQPSK. Filtering operations are done for *spectral shaping*, to obtain better spectral properties of the modulated signal. It can also be used to build in some amount of redundancy, or to simplify demodulation.

It is possible that the symbol rate of the output of the coder is different than the input bit rate. E.g, if a binary sequence is coded into m-PSK, m-PAM, or m-QAM, the rate drops by a factor of $\log_2 m$. (The opposite is also possible, e.g., in CDMA systems, where each bit is coded into a sequence of 31 or more "chips".)

1.4.2 Raised cosine pulseshape

The coded digital signal s_k can be described in analog form as a sequence of dirac pulses,

$$s_\delta(t) = \sum_{k=-\infty}^{\infty} s_k \delta(t-k)\,,$$

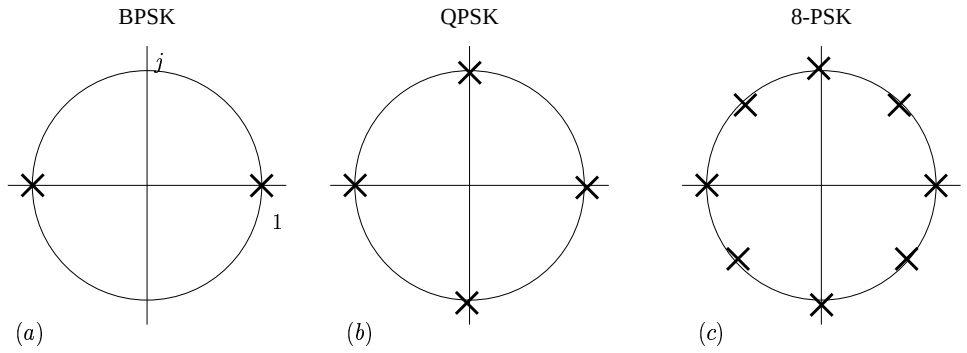


Figure 1.17. Digital constellations

where, for convenience, the symbol rate is normalized to $T = 1$. In linear modulation schemes, the digital dirac-pulse sequence is convolved by a pulse shape function $p(t)$:

$$s(t) = p(t) * s_\delta(t) = \sum_{k=-\infty}^{\infty} s_k p(t-k)\,.$$

Again, there are many possibilities. The optimum wave form is one that is both localized in time (to lie within a pulse period of length $T = 1$) and in frequency (to satisfy the Nyquist criterion when sampled at a rate $1/T = 1$). This is of course impossible, but good approximations exist. A pulse with perfect frequency localization is the sinc-pulse, defined by

$$p(t) = \frac{\sin \pi t}{\pi t}\,, \qquad P(f) = \begin{cases} 1\,, & |f| < \frac{1}{2} \\ 0\,, & \text{otherwise} \end{cases} \tag{1.7}$$

However, the pulse has very long tails in the time-domain. A modification of this pulse leads to the family of *raised-cosine* pulseshapes, with better localization properties. They are defined, for $\alpha \leq 1$, by [4, ch.6]

$$p(t) = \frac{\sin \pi t}{\pi t} \cdot \frac{\cos \alpha \pi t}{1 - 4\alpha^2 t^2}$$

with corresponding spectrum

$$P(f) = \begin{cases} 1\,, & |f| < \frac{1}{2}(1-\alpha) \\ \frac{1}{2} - \frac{1}{2}\sin(\frac{\pi}{\alpha}(|f| - \frac{1}{2}))\,, & \frac{1}{2}(1-\alpha) < |f| < \frac{1}{2}(1+\alpha) \\ 0\,, & \text{otherwise} \end{cases}$$

The spectrum is limited to $|\,f\,| \leq \frac{1}{2}(1+\alpha)$, so that α represents the excess bandwidth. For $\alpha = 0$, the pulse is identical to the sinc pulse (1.7). For other values of α, the amplitude decays more smoothly in frequency, so it is also known as the *rolloff factor*. The shape of the rolloff is that of a cosine, hence the name. In the time domain, the pulses are still infinite in extent.

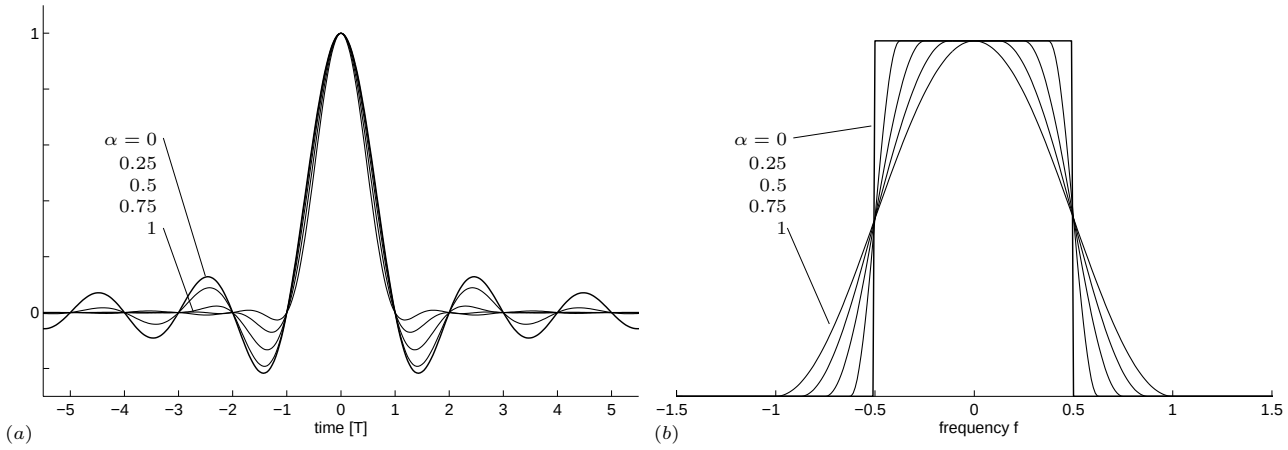


Figure 1.18. (a) Family of raised-cosine pulse shape functions, (b) corresponding spectra.

However, as α increases, the size of the tails diminishes. A common choice is $\alpha = 0.35$, and to truncate $p(t)$ outside the interval $[-3, 3]$.

The raised-cosine pulses are designed such that, when sampled at integer time instants, the only nonzero sample occurs at $t = 0$. Thus, $s_k = s(k)$, and to recover $\{s_k\}$ from $s(t)$ is simple, provided we are synchronized: any fractional delay $0 < \tau < 1$ results in intersymbol interference. Sometimes, we assume that $p(t)$ is part of the filter $g(t)$ in (1.5), in which case (1.5) transforms into

$$\begin{aligned} \mathbf{x}(t) &= \left[\sum_{i=1}^{r} \mathbf{a}(\theta_i)\beta_i g(t-\tau_i)\right] * s_\delta(t) \\ &= \sum_{k=-\infty}^{\infty} \left[\sum_{i=1}^{r} \mathbf{a}(\theta_i)\beta_i g(t-k-\tau_i)\right] s_k. \end{aligned} \tag{1.8}$$

1.4.3 Phase modulation

An alternative to linear modulation is phase modulation,

$$s(t) = e^{j\phi(t)}\,, \qquad \phi(t) = q(t) * s_\delta(t) = \sum_{k=-\infty}^{\infty} s_k q(t-k)\,,$$

In this case, the digital symbols are modulated with some phase function $q(t)$, which gives the phase of the modulated signal. A simple choice for $q(t)$ is

$$q(t) = \begin{cases} 0 & t < 0\,, \\ \pi t & 0 \le t < 1\,. \\ \pi & t \ge 1 \end{cases}$$

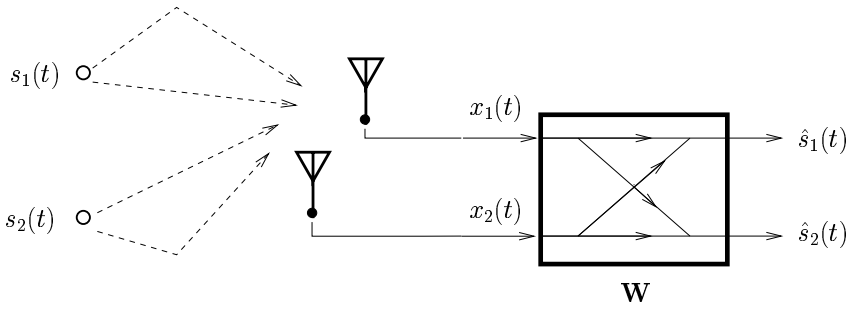


Figure 1.19. Spatial beamformer with an I-MIMO channel.

Note that the phase changes are very abrupt at integer times, leading to excess bandwidth. To mitigate this, we can use a smoother function for $q(t)$, and/or a smaller *modulation index* (scaling of $q(t)$ so that $q(\infty) < \pi$), and/or a *partial response modulation*, where $q(t)$ is nonconstant in a larger interval than just $[0, 1]$. However, this study is beyond the scope of this course, and we will generally consider only linear modulation schemes.

1.5 DATA MODEL FOR SIGNAL PROCESSING

In section 1.3, we have derived a channel model based on physical properties of the radio channel. Though useful for generating simulated data, it is not always a suitable model for identification purposes, e.g., if the number of parameters is large, if the angle spreads within a cluster are large so that parametrization in terms of directions is not possible, or if there is a large and fuzzy delay spread. In these situations, it is more appropriate to work with an unstructured model, where the channel impulse responses are posed simply as arbitrary multichannel finite impulse response (FIR) filters. It is a generalization of the physical channel model considered earlier, in the sense that at a later stage we can still specify the structure of the coefficients.

1.5.1 I-MIMO model

Assume that d source signals $s_1(t), \cdots, s_d(t)$ are transmitted from d independent sources at different locations. If the delay spread is small, then what we receive at the antenna array will be a simple linear combination of these signals:

$$\mathbf{x}(t) = \mathbf{a}_1 s_1(t) + \cdots + \mathbf{a}_d s_d(t)$$

where as before $\mathbf{x}(t)$ is a stack of the output of the M antennas. We will usually write this in matrix form:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)\,, \quad \mathbf{A} = [\mathbf{a}_1 \cdots \mathbf{a}_d]\,, \quad \mathbf{s}(t) = \begin{bmatrix} s_1(t) \\ \vdots \\ s_d(t) \end{bmatrix}.$$

Suppose we sample with a period T, normalized to $T = 1$, and collect a batch of N samples into a matrix $\mathbf{X}$, then

$$\mathbf{X} = \mathbf{AS}$$

where $\mathbf{X} = [\mathbf{x}(0) \cdots \mathbf{x}(N-1)]$ and $S = [\mathbf{s}(0) \cdots \mathbf{s}(N-1)]$. The resulting $[\mathbf{X} = \mathbf{AS}]$ model is called an *instantaneous multi-input multi-output* model, or I-MIMO for short. It is a generic linear model for source separation, valid when the delay spread of the dominant rays is much smaller than the inverse bandwidth of the signals, e.g., for narrowband signals, in line-of-sight situations or in scenarios where there is only local scattering. Even though this appears to limit its applicability, it is important to study it in its own right, since more complicated convolutive models can often be reduced (after equalization) to $\mathbf{X} = \mathbf{AS}$.

The objective of beamforming for source separation is to construct a left-inverse $\mathbf{W}^{\mathrm{H}}$ of $\mathbf{A}$, such that $\mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I}$ and hence $\mathbf{W}^{\mathrm{H}}\mathbf{X} = \mathbf{S}$: see figure 1.19. This will recover the source signals from the observed mixture. It immediately follows that in this scenario it is necessary to have $d \leq M$ to ensure interference-free reception, i.e., not more sources than sensors. If we know already (part of) $\mathbf{S}$, e.g., because of training, then $\mathbf{W}^{\mathrm{H}} = \mathbf{S}\mathbf{X}^{\dagger} = \mathbf{S}\mathbf{X}^{\mathrm{H}}(\mathbf{X}\mathbf{X}^{\mathrm{H}})^{-1}$, where $\mathbf{X}^{\dagger}$ denotes the Moore-Penrose pseudo-inverse of $\mathbf{X}$, here equal to its right inverse (see chapter 2). With noise, other beamformers may be better.

If we adopt the multipath propagation model, then $\mathbf{A}$ is endowed with a parametric structure: every column $\mathbf{a}_i$ is a sum of direction vectors $\mathbf{a}(\theta_{ij})$, with different fadings β_{ij}. If the i-th source is received through r_i rays, then

$$\mathbf{a}_i \;=\; \sum_{j=1}^{r_i} \mathbf{a}(\theta_{ij})\beta_{ij} \;=\; [\mathbf{a}(\theta_{i1}) \cdots \mathbf{a}(\theta_{i,r_i})] \begin{bmatrix} \beta_{i1} \\ \vdots \\ \beta_{i,r_i} \end{bmatrix} \qquad (i = 1, \cdots, d)\,.$$

If each source has only a single ray to the receiver array (a line-of-sight situation), then each $\mathbf{a}_i$ is a vector on the array manifold, and identification will be relatively straightforward. The more general case amounts to decomposing a given $\mathbf{a}$-vector into a sum of vectors on the manifold, which makes identification much harder.

To summarize the parametric structure in a compact way, we usually collect all $\mathbf{a}(\theta_{ij})$-vectors and path attenuation coefficients β_{ij} of all rays of all sources in single matrices $\mathbf{A}_\theta$ and $\mathbf{B}$,

$$\mathbf{A}_\theta = [\mathbf{a}(\theta_{11}) \cdots \mathbf{a}(\theta_{d,r_d})]\,, \qquad \mathbf{B} = \begin{bmatrix} \beta_{11} & & \\ & \ddots & \\ & & \beta_{d,r_d} \end{bmatrix}.$$

To sum the rays belonging to each source into the single $\mathbf{a}_i$-vector of that source, we define a selection matrix

$$\mathbf{J} = \begin{bmatrix} \mathbf{1}_{r_1} & & 0 \\ & \ddots & \\ 0 & & \mathbf{1}_{r_d} \end{bmatrix} : \quad r \times d \tag{1.9}$$

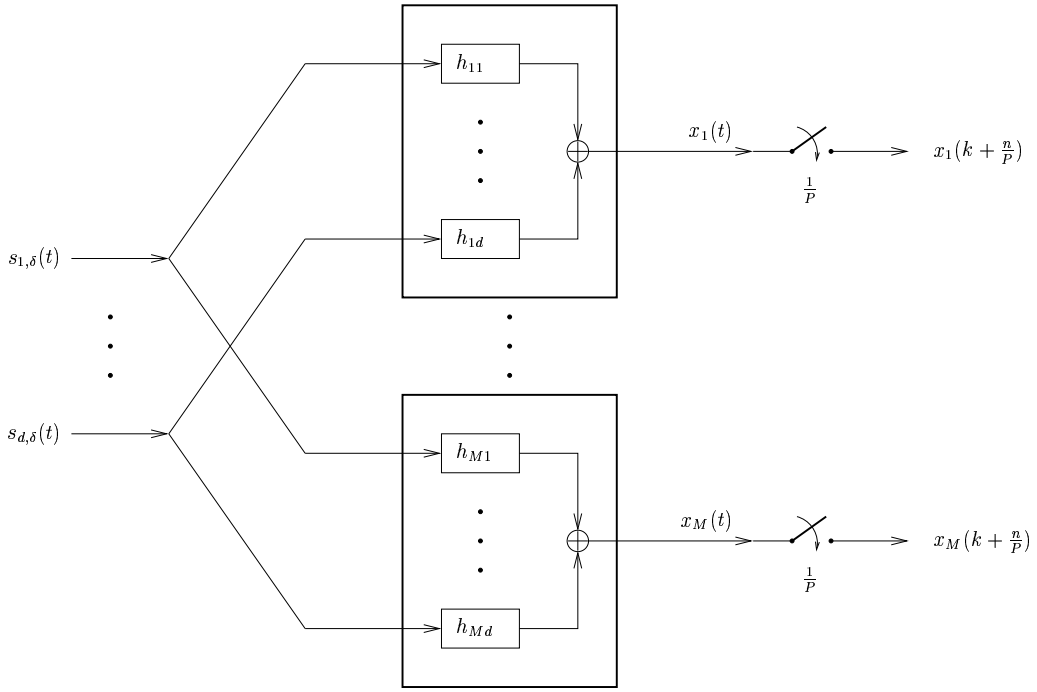


Figure 1.20. Channel model.

where $r = \sum_1^d r_i$ and $\mathbf{1}_m$ denotes an $m \times 1$ vector consisting of 1's. Together, this allows to write the full (noise-free) I-MIMO data model as

$$\mathbf{X} = \mathbf{AS}\,, \qquad \mathbf{A} = \mathbf{A}_\theta \mathbf{BJ}\,. \tag{1.10}$$

1.5.2 FIR-MIMO model

Assume that d source signals are transmitted from d independent sources, but moreover that they are now received through a convolutive channel. This scenario is as in figure 1.2, a schematic representation of the same is in figure 1.20. To limit ourselves to a practical and interesting case, let us assume that the signals are digital with a common pulse period, so that they can be described by a sequence of dirac pulses,

$$s_{j,\delta}(t) = \sum_{k=-\infty}^{\infty} s_{j,k}\delta(t - kT)\,.$$

For convenience, we normalize the symbol period to $T = 1$. The signal emitted by a source is a convolution of $s_{j,\delta}(t)$ by the pulse shape function $p(t)$, e.g., a raised cosine (generalized sinc function), which gives

$$s_j(t) = p(t) * s_{j,\delta}(t) = \sum_{k=-\infty}^{\infty} p(t-k)s_{j,k}$$

After propagation through the channel, the signal is received by an array of M sensors, with outputs $x_1(t), \cdots, x_M(t)$. The impulse response of the channel from source j to the i-th sensor, $h_{i,j}(t)$, is a convolution of the pulse shaping filter $p(t)$ and the actual channel response from $s_{j,\delta}(t)$ to $x_i(t)$. We can include any propagation delays and delays due to unsynchronized sources in $h_{i,j}(t)$ as well. The data model is written compactly as the convolution

$$\mathbf{x}(t) = \mathbf{H}(t) * \mathbf{s}_\delta(t)\,,$$

where

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_M(t) \end{bmatrix},\ \mathbf{H}(t) = \begin{bmatrix} h_{11}(t) \cdots h_{1d}(t) \\ \vdots \qquad \vdots \\ h_{M1}(t) \cdots h_{Md}(t) \end{bmatrix},\ \mathbf{s}_\delta(t) = \begin{bmatrix} s_{1,\delta}(t) \\ \vdots \\ s_{d,\delta}(t) \end{bmatrix}.$$

At this point, we make the assumption that the M channels $h_{i,j}(t)$ associated to each source j are FIR filters of (integer) length at most L symbols:

$$h_{i,j}(t) = 0\,, \quad t \notin [0, L) \qquad i = 1, \cdots, M;\ j = 1, \cdots, d\,.$$

An immediate consequence of the FIR assumption is that, at any given moment, at most L consecutive symbols of signal j play a role in $\mathbf{x}(t)$. Indeed, for $t = k + \tau$, where $k \in \mathbb{Z}$ and $0 \le \tau < 1$, the convolution $x_i(t) = \sum_j h_{i,j}(t) * s_{j,\delta}(t)$ can be expressed as

$$x_i(k+\tau) = \sum_{l=0}^{L-1} h_{i,1}(l+\tau) s_{1,k-l} \ + \cdots + \ \sum_{l=0}^{L-1} h_{i,d}(l+\tau) s_{d,k-l}\,. \tag{1.11}$$

Suppose that we sample each $x_i(t)$ at a rate of P times the symbol rate, and collect samples during K symbol periods. Then we can construct a data matrix $\mathbf{X}$ containing all samples as

$$\mathbf{X} = [\mathbf{x}_0 \quad \cdots \quad \mathbf{x}_{N-1}] := \begin{bmatrix} \mathbf{x}(0) & \mathbf{x}(1) & \cdots & \mathbf{x}(N-1) \\ \mathbf{x}(\frac{1}{P}) & \mathbf{x}(1+\frac{1}{P}) & \cdots & \mathbf{x}(N-1+\frac{1}{P}) \\ \vdots & \vdots & & \vdots \\ \mathbf{x}(\frac{P-1}{P}) & \mathbf{x}(1+\frac{P-1}{P}) & \cdots & \mathbf{x}(N-1+\frac{P-1}{P}) \end{bmatrix}. \tag{1.12}$$

$\mathbf{X}$ has size $MP \times N$; its k-th column $\mathbf{x}_k$ contains the MP spatial and temporal samples taken during the k-th interval. Based on the FIR assumption, it follows that $\mathbf{X}$ has a factorization

$$\mathbf{X} = \mathbf{H}\mathcal{S}_L \tag{1.13}$$

where

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} \mathbf{H}(0) & \mathbf{H}(1) & \cdots & \mathbf{H}(L-1) \\ \mathbf{H}(\frac{1}{P}) & \mathbf{H}(1+\frac{1}{P}) & \cdots & \mathbf{H}(L-1+\frac{1}{P}) \\ \vdots & \vdots & & \vdots \\ \mathbf{H}(\frac{P-1}{P}) & \mathbf{H}(1+\frac{P-1}{P}) & \cdots & \mathbf{H}(L-1+\frac{P-1}{P}) \end{bmatrix} : \quad MP \times dL \\ \mathcal{S}_L &= \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_{N-1} \\ \mathbf{s}_{-1} & \mathbf{s}_0 & \cdots & \mathbf{s}_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_{-L+1} & \mathbf{s}_{-L+2} & \cdots & \mathbf{s}_{N-L} \end{bmatrix} : \quad dL \times N\,, \end{aligned} \tag{1.14}$$

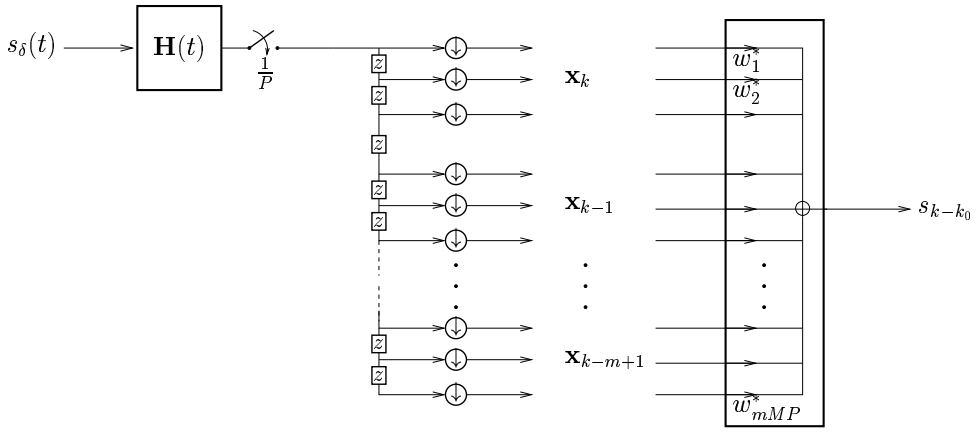


Figure 1.21. Space-time equalizer

and in this context $\mathbf{s}_i = [s_{1,i} \ \cdots \ s_{d,i}]^T$, a d-dimensional vector. The matrix $\mathbf{H}$ represents the unknown space-time channel, whereas $\mathcal{S} = \mathcal{S}_L$ contains the transmitted symbols.[5] $\mathcal{S}$ has a *block-Toeplitz* structure: it is constant along the diagonals. This structure is a consequence of the time-invariance of the channel. For generality, we have assumed that the measured block of data starts while the transmission of each of the signals was already in progress, i.e., $\mathbf{x}_0$ is determined by previous symbols $\mathbf{s}_{-L+1}, \cdots, \mathbf{s}_{-1}$ as well as $\mathbf{s}_0$. Note that if the channels do not all have the same length L, then certain columns of $\mathbf{H}$ are equal to zero.

A linear equalizer in this context can be written as a vector $\mathbf{w}$ which combines the rows of $\mathbf{X}$ to generate an output $\mathbf{w}^{\mathrm{H}}\mathbf{X}$. In the model so far, we can only equalize among the antenna outputs (simple beamforming) and among the P samples within one sample period (polyphase combining). More generally, we would want to filter over multiple sample periods, leading to a *space-time* equalizer. For a linear equalizer with a length of m symbol periods, we have to augment $\mathbf{X}$ with $m-1$ horizontally shifted copies of itself:

$$\mathcal{X}_m = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{N-1} \\ \mathbf{x}_{-1} & \mathbf{x}_0 & \cdots & \mathbf{x}_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{-m+1} & \mathbf{x}_{-m+2} & \cdots & \mathbf{x}_{N-m} \end{bmatrix} : \quad mMP \times N \,.$$

Each column of $\mathcal{X}_m$ is a regression vector: the memory of the filter. Using $\mathcal{X}_m$, a general space-time linear equalizer can be written as $\mathbf{w}^{\mathrm{H}}\mathcal{X}_m$, which combines mP snapshots of M antennas:

[5] The subscript L denotes the number of block rows in $\mathcal{S}$. We usually omit the subscript if this does not lead to confusion.

see figure 1.21. The augmented data matrix $\mathcal{X}_m$ has a factorization

$$\mathcal{X}_m \;=\; \mathcal{H}_m \mathcal{S}_{L+m-1} \;=\; \begin{bmatrix} \boxed{\mathbf{H}} & & \mathbf{0} \\ & \boxed{\mathbf{H}} & \\ & & \ddots \\ \mathbf{0} & & \boxed{\mathbf{H}} \end{bmatrix} \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_{N-1} \\ \mathbf{s}_{-1} & \mathbf{s}_0 & \cdots & \mathbf{s}_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_{-L-m+2} & \mathbf{s}_{-L-m+3} & \cdots & \mathbf{s}_{N-L-m+1} \end{bmatrix} \tag{1.15}$$

where $\mathcal{H} = \mathcal{H}_m$ has size $mMP \times d(L+m-1)$ and the m shifts of $\mathbf{H}$ to the right are each over d positions. $\mathcal{H}$ also has a block-Toeplitz structure. $\mathcal{S}_{L+m-1}$ has the same structure as $\mathcal{S}_L$. A *necessary* condition for space-time equalization (the output $\mathbf{w}^{\mathrm{H}}\mathcal{X}_m$ is equal to a row of $\mathcal{S}_{L+m-1}$) is that $\mathcal{H}$ is tall, which gives minimal conditions on m in terms of M, P, d, L:

$$mMP \geq d(L+m-1) \qquad \Rightarrow \qquad m(MP-d) \geq d(L-1)$$

which implies

$$MP > d\,, \qquad m \geq \frac{d(L-1)}{MP-d}$$

Unlike spatial beamforming, it will not be necessary to find $\mathcal{H}^{\dagger}$: it suffices to reconstruct a single block row of $\mathcal{S}$, which can be done with d space-time equalizers $\mathbf{w}_i$. Nonlinear equalizer structures are possible, e.g., by using feedback, but are not discussed here.

1.5.3 Connection to the parametric multipath model

For a *single* source, recall the multipath propagation model (1.8), valid for specular multipath with small cluster angle and delay spread:

$$\mathbf{h}(t) = \sum_{i=1}^{r} \mathbf{a}(\theta_i)\beta_i g(t-\tau_i) \tag{1.16}$$

where $g(t)$ includes all the filtering at transmitter and receiver, as well as the pulse shape function $p(t)$. In this model, there are r distinct propagation paths, each parameterized by $(\theta_i, \tau_i, \beta_i)$, where θ_i is the direction-of-arrival (DOA), τ_i is the path delay, and $\beta_i \in \mathbb{C}$ is the complex path attenuation (fading). The vector-valued function $\mathbf{a}(\theta)$ is the array response vector for an array of M antenna elements to a signal from direction θ.

Suppose as before that $\mathbf{h}(t)$ has finite duration and is zero outside an interval $[0, L)$. Consequently, $g(t-\tau_i)$ has the same support for all τ_i. At this point, we can define a parametric "time manifold" vector function $\mathbf{g}(\tau)$, collecting LP samples of $g(t-\tau)$:

$$\mathbf{g}(\tau) = \begin{bmatrix} g(0-\tau) \\ g(\frac{1}{P}-\tau) \\ \vdots \\ g(L-\frac{1}{P}-\tau) \end{bmatrix}, \quad 0 \leq \tau \leq \max \tau_i\,.$$

If we also construct a vector $\mathbf{h}$ with samples of $\mathbf{h}(t)$,

$$\mathbf{h} = \begin{bmatrix} \mathbf{h}(0) \\ \mathbf{h}(\frac{1}{P}) \\ \vdots \\ \mathbf{h}(L - \frac{1}{P}) \end{bmatrix}$$

then it is straightforward to verify that (1.16) gives

$$\mathbf{h} = \sum_{i=1}^{r} (\mathbf{g}_i \otimes \mathbf{a}_i)\beta_i = [\mathbf{g}_1 \otimes \mathbf{a}_1, \cdots, \mathbf{g}_r \otimes \mathbf{a}_r] \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_r \end{bmatrix}$$
$$\mathbf{g}_i = \mathbf{g}(\tau_i)\,, \quad \mathbf{a}_i = \mathbf{a}(\theta_i)\,,$$

where '$\otimes$' denotes a Kronecker product, defined for vectors $\mathbf{a}$ and $\mathbf{b}$ as

$$\mathbf{a} \otimes \mathbf{b} = \begin{bmatrix} a_1\mathbf{b} \\ \vdots \\ a_m\mathbf{b} \end{bmatrix}.$$

Thus, the multiray channel vector is a weighted sum of vectors on the *space-time manifold* $\mathbf{g}(\tau) \otimes \mathbf{a}(\theta)$. Because of the Kronecker product, this is a vector in an LPM-dimensional space, with more distinctive characteristics than the M-dimensional $\mathbf{a}(\theta)$-vector in a scenario without delay spread. The connection of $\mathbf{h}$ with $\mathbf{H}$ as in (1.14) is that $\mathbf{h} = \mathrm{vec}(\mathbf{H})$, i.e., $\mathbf{h}$ is a stacking of all columns of $\mathbf{H}$ in a single vector.

We can define, much as before, parametric matrix functions

$$\mathbf{A}_\theta = [\mathbf{a}(\theta_1) \cdots \mathbf{a}(\theta_r)]\,, \quad \mathbf{G}_\tau = [\mathbf{g}(\tau_1) \cdots \mathbf{g}(\tau_r)]\,, \quad \mathbf{B} = \begin{bmatrix} \beta_1 & & \\ & \ddots & \\ & & \beta_r \end{bmatrix}$$

$$\mathbf{G}_\tau \circ \mathbf{A}_\theta := [\mathbf{g}_1 \otimes \mathbf{a}_1, \cdots, \mathbf{g}_r \otimes \mathbf{a}_r]$$

$(\mathbf{G}_\tau \circ \mathbf{A}_\theta)$ is a columnwise Kronecker product known as the *Khatri-Rao product*. this gives $\mathbf{h} = (\mathbf{G}_\tau \circ \mathbf{A}_\theta)\mathbf{B}\mathbf{1}_r$. Extending now to d sources, we get that the $MP \times dL$-sized matrix $\mathbf{H}$ in (1.14) can be rearranged into an $MPL \times d$ matrix

$$\mathbf{H}' = [\mathbf{h}_1 \cdots \mathbf{h}_d] = (\mathbf{G}_\tau \circ \mathbf{A}_\theta)\mathbf{B}\mathbf{J}\,.$$

where $\mathbf{J}$ is the selection matrix defined in (1.9) that sums the rays into channel vectors. $(\mathbf{G}_\tau \circ \mathbf{A}_\theta)$ now plays the same role as $\mathbf{A}_\theta$ in the previous section. Each of its columns is a vector on the space-time manifold.

Summary A summary of the noise-free data models developed so far is

$$\begin{array}{ll} \text{I-MIMO:} & \mathbf{X} = \mathbf{AS}\,, \quad \mathbf{A} = \mathbf{A}_\theta\mathbf{BJ} \\ \text{FIR-MIMO:} & \mathcal{X} = \mathcal{HS}\,, \quad \mathcal{H} \leftrightarrow \mathbf{H}' = (\mathbf{G}_\tau \circ \mathbf{A}_\theta)\mathbf{BJ} \end{array} \tag{1.17}$$

The first part of these model equations is generally valid for linear time-invariant channels, whereas the second part is a consequence of the adopted multiray model.

Based on this model, the received data matrix $\mathbf{X}$ or $\mathcal{X}$ has several *structural properties*. In several combinations, these are often strong enough to allow to find the factors $\mathbf{A}$ (or $\mathbf{H}$) and $\mathbf{S}$ (or $\mathcal{S}$), even from knowledge of $\mathbf{X}$ (or $\mathcal{X}$) alone. Very often, this will be in the form of a collection of beamformers (or space-time equalizers) $\{\mathbf{w}_i\}_{i=1}^d$ such that each beamformed output $\mathbf{w}_i^{\mathrm{H}}\mathbf{X}$ (or $\mathbf{w}_i^{\mathrm{H}}\mathcal{X}$) is equal to one of the source signals, so that it must have the properties of that signal.

One of the most powerful "structures", on which most systems today rely to a large extent, is knowledge of part of the transmitted message (a training sequence), so that several columns of $\mathcal{S}$ are known. Along with the received signal $\mathcal{X}$, this allows to estimate $\mathbf{H}$. Very often, an unparameterized FIR model is assumed here. The algorithms are using a *temporal reference*. Algorithms that do not use this are called *blind*. Examples of this will be discussed in the coming chapters.

1.6 APPLICATIONS OF SPACE-TIME PROCESSING[6]

To end this chapter, we describe a few applications of antennas and space-time processing in cellular base stations.

1.6.1 Simple space-time filtering

Switched beam systems Switched Beam Systems (SBS) use an antenna array, a switch and a beam selector or sniffer. The SBS operates by forming a set of preformed beams, which are then scanned by the sniffer to determine the best, or sometimes, the two best beams. These beams are then switched through to the radio receiver. Beam selection is made by picking the beam with the highest signal level (by use of a *Received Signal Strength Indicator* (RSSI)). Signal fading and interference can cause incorrect beam selection and therefore the signal is often first validated with the identifying color code prior to RSSI averaging.

The main advantage of the SBS is the improvement in cell coverage on the reverse link due to the improved array gain of the antenna array. Also, since the SBS beamwidth is smaller than that of a sector antenna, a reduction in the average interference power is also likely [26].

Adaptive spatial filtering Adaptive spatial filtering implements single user space-time processing discussed earlier. The goals is to maximize the signal delivered power while minimizing

[6]From Paulraj and Papadias [1].

ISI and CCI. These methods offer significant improvement in capacity and link quality in cellular base stations. Details of implementation depend on the air-interface, propagation environment and traffic characteristics. Adaptive spatial filtering offers both coverage and capacity improvements.

1.6.2 Increasing capacity using space-time processing

Consider the simple low rank channel model in (1.4) which may be used to characterize the downlink spatial channel statistics as well. In most current frequency division duplex (FDD) systems the up and downlink fading may be considered independent. If the main objective is increased range, this does not pose a major problem. However, the unobservable downlink channel is one of the main obstacles if the intention is to also increase system capacity. An array could be employed at the mobile site as well, but in many applications this is not considered a feasible solution.[7]

There are two main approaches for increasing capacity with antenna arrays:

1. The frequency reuse distance may be decreased, or
2. multiple mobiles in the same cell may be allocated to the same channel.

Channel Reuse within Cell (CRC) (or space division multiple access, SDMA) refers to the reuse of a channel *within* a cell by exploiting differences in directions of users. In CRC, the entire spectrum resource of a cell is reused by exploiting separation in the directions of the users. This is similar to spectrum reuse in cellular systems, where a channel used in one cell is reused in another cell that is separated by a sufficient *distance* such that the co-channel interference is sufficiently small.

Increase of the capacity of the uplink requires space-time processing to separate and equalize multiple received signals. In the first case (smaller frequency reuse distance), the objective is to detect only the desired user, but jointly estimating the strong adjacent cell interferers might be beneficial for this. In the second case (co-channel users), all received signals have to be separated and detected. This is a much more ambitious feat, since we have to ensure that we can detect users, even when they are in a deep fade among strong interferers. Many algorithms are currently devised to solve such joint estimation problems.

In any case, a capacity increase can only be realised if we can handle the downlink as well. The idea is to have the base station transmit signals directed to the intended user, while sending nulls in the direction of co-channel users. The problem with directional transmission in the downlink is that it is truly blind: we have to rely on a channel model that was estimated from the uplink channel.

In time division duplex (TDD) systems, the up- and downlink channels can be considered reciprocal if there is limited movement between receive and transmit, i.e., as long as the "ping-pong"

[7] This might change once higher frequencies are used.

time is small compared to the channel coherence time. Uplink channel information may then be used to achieve spatially selective transmission and thus increase the capacity [27]. We do not have to use the parametric multipath model: the unstructured FIR model suffices. The estimated uplink channel may be inverted in the sense that the signals are appropriately pre-equalized and spatially multiplexed at the base station to minimize inter symbol and co-channel interference at the users [28].

However, most current systems are frequency division duplex (FDD). If we are attempting to increase capacity in current FDD systems in the downlink, the information gained from the signal separation techniques in the uplink cannot be used directly. In most systems, the separation between the forward and reverse link frequencies is about 5% of the mean carrier frequency. This implies that the principle of reciprocity cannot be used: the up- and downlink flat fading may be considered independent. The unobservable downlink channel is one of the main obstacles if the intention is to also increase system capacity.

Two classes of solutions are known to date to attempt to solve this problem.

Parametric channel estimation and inversion The transformation of the space-time receive weights to transmit weights is not a well conditioned problem unless the parametric multipath model is introduced. Since the frequency separation is 5%, the forward and reverse channel will share many common features. If the specular multipath channel model as described earlier holds true, the paths used by both links can assumed to be identical: the path delays and paths angles are the same for both links, and remain approximately constant over long periods in time. However, the path amplitudes and phases (β_i) are not the same on both links, and will in fact be largely uncorrelated and independently fading. Hence, the overall channel response appears to be uncorrelated between the forward and reverse links.

Channel estimation using feedback A direct approach to estimating the forward channel is to arrange for the subscriber unit to feedback part of the received signal to the base station, enabling it to estimate the channel [29]. This requires a complete redesign of protocols and signaling and is probably only possible in environments which vary very slowly in time. However, it may be feasible for movable (rather than mobile) systems such as indoor wireless local area networks.

In time varying channels which need frequent tracking, more efficient probing methods can be used to reduce the overhead of multiple training signals. The common subspaces shared by the two channels can be used with great effectiveness to minimize the amount of training needed [30].

Practical considerations CRC is very unlikely to be of value in current TDMA air interfaces. FDD systems pose significant problems since the forward channel is only partially known. Even in TDD systems rapid channel variations limit performance. In CDMA, because of inherent quasi-orthogonality of user signals, CRC is much simpler to implement and in fact is used in the current CDMA air interfaces.

Core space-time processing algorithms discussed above serve as a basis for practical techniques that can work in real mobile channels. The space-time processing solutions are influenced by the multiple access technique, modulation, bit rate, slot duration and slot repetition rate, delay spread, angle spread and number of paths, and the mobile speed. These core algorithms will need to be merged in several ways to suit specific situations. Some examples are:

- *Non-blind and blind.* Most mobile radio interfaces support some form of periodic training. However, the channel may vary between the training bursts and we need blind techniques to track the channel between bursts. Therefore, hybrid blind/non-blind methods are needed in many mobile applications.
- *Single and multi-user.* While some mobile applications may result in true multi-user situations, most applications have only one user of interest. However, since joint demodulation of signal and interference can outperform single-user demodulation, multi-user algorithms are preferred. This, however, needs knowledge of the interference channel which can be hard to estimate due to weak interference power. In practice, some combination of single and multi-user solutions may be needed to use only partial knowledge of the interference channel.
- *Spatial and temporal structure.* While signal and temporal structure is robust to most channels, large delay spread and high speed mobiles may stress such techniques to a breaking point. Spatial structure methods, on the other hand, can work in high Doppler situations but are affected by delay and angle spreads. Therefore, a combination of spatial and temporal structure algorithms may be needed in complex environments.

Bibliography

[1] A. Paulraj and C. Papadias, "Space-time processing for wireless communications," *IEEE Signal Processing Magazine*, vol. 14, pp. 49–83, Nov. 1997.

[2] W. Jakes, ed., *Microwave Mobile Communications*. New York: John Wiley, 1974.

[3] W. Lee, *Mobile Communications Design Fundamentals*. New York: John Wiley, 1993.

[4] E. Lee and D. Messerschmitt, *Digital Communication*. Boston: Kluwer Publishers, 1988.

[5] R. Steele, *Mobile radio communications*. Pentech Press, 1992.

[6] J. Proakis, *Digital Communications*. New York: McGraw-Hill, 1983.

[7] J. Proakis and M. Salehi, *Communication Systems Engineering*. Prentice-Hall, 1994.

[8] B. Ottersten, "Array processing for wireless communications," in *Proc. IEEE workshop on Stat. Signal Array Proc.*, (Corfu), pp. 466–473, June 1996.

[9] D. Asztély, "On antenna arrays in mobile communication systems, fast fading and GSM base station receiver algorithms," Tech. Rep. IR-S3-SB-9603, Signals, Sensors and Systems, Royal Inst. Sciences, Stockholm, Mar. 1996.

[10] F. A. etal, "Cross correlation between the envelopes of 900 MHz signals received at a mobile radio base station site," *IEE Proceedings*, vol. 133, pp. 506–512, Oct. 1986.

[11] W. Lee, "Effects on correlation between two mobile radio basestation antennas," *IEEE Tr. Comm.*, vol. 21, pp. 1214–1224, Nov. 1973.

[12] B. Sklar, "Rayleigh fading channels in mobile digital communication systems, part I: Characterization," *IEEE Communications Magazine*, vol. 35, pp. 90–100, July 1997.

[13] T. Trump and B. Ottersten, "Estimation of nominal direction of arrival and angular spread using an array of sensors," *Signal Processing*, vol. 50, pp. 57–69, Apr. 1996.

[14] T. Rappaport, *Wireless Communications: Principles and Practice*. Upper Saddle River, NJ: Prentice Hall, 1996.

[15] K. Pahlavan and A. Levesque, "Wireless data communications," *Proceedings of the IEEE*, vol. 82, pp. 1398–1430, Sept. 1994.

[16] F. Lotse, "Macrocell propagation measurements at 900 MHz using a linear antenna array," Tech. Rep. T/B 94:065, Ericsson, 1994.

[17] P. Zetterberg and B. Ottersten, "The spectrum efficiency of a basestation antenna array system for spatially selective transmission," *IEEE Tr. Veh. Techn.*, vol. 44, pp. 651–660, Aug. 1995.

[18] T. Trump and B. Ottersten, "Maximum likelihood estimation of nominal direction of arrival and angular spread using an array of sensors," in *Proc. COST 229 workshop*, (Vigo (Spain)), Oct. 1994.

[19] T. Trump and B. Ottersten, "Estimation of nominal direction of arrival and angular spread using an array of sensors," *to appear in Signal Processing*, 1996.

[20] J. Parsons and A. Turkmani, "Characterization of mobile radio signals: model description," *IEE Proceedings-I*, vol. 138, pp. 549–555, Dec. 1991.

[21] F. Kronestedt, "A study of local scattering," Tech. Rep. T/B 94:050, Ericsson, 1994.

[22] A. Klein, W. Mohr, R. Thomas, P. Weber, and B. Wirth, "Direction of arrival of partial waves in wideband mobile radio channels for intelligent antenna concepts," in *Proc. IEEE Veh. Techn. Conf.*, pp. 849–853, Nov. 1996.

[23] E. T. S. Institute, "European digital cellular telecommunications system (phase 2): Radio transmission and reception (GSM 05.05)," tech. rep., ETSI, Sophia Antipolis, France, 1994.

[24] U. F. et al., "Adaptive antenna arrays for GSM900/DCS1800," in *Proc. IEEE Veh. Technol. Conf.*, 1994.

[25] P. Zetterberg, "Mobile communication with base station antenna arrays: Propagation modeling and system capacity," Tech. Rep. TRITA-SB-9502, Signals, Sensors and Systems, Royal Inst. Science, Stockholm, Feb. 1995.

[26] T. Matsumoto, S. Nishioka, and D. Hodder, "Beam selection performance analysis of a switched multi-beam antenna system in mobile communications environments," in *Second workshop on Smart Antennas in Wireless Mobile Communications*, (Stanford, CA 94305, USA), July 1995.

[27] P. Mogensen, F. Frederiksen, J. Wigard, and S. Petersen, "A research study of antenna diversity and data receivers for DECT," in *Proc. Nordic Radio Symposium*, (Saltsjöbaden, Sweden), Apr. 1995.

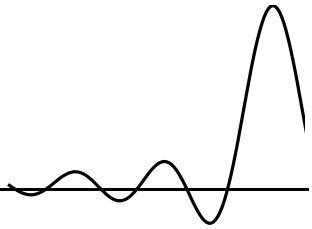
[28] H. Liu and G. Xu, "Multiuser blind channel estimation and spatial channel pre-equalization," in *Proc. IEEE ICASSP*, (Detroit), pp. 1756–1759 vol.3, May 1995.

[29] D. Gerlach and A. Paulraj, "Adaptive transmitting antenna arrays with feedback," *IEEE SP Letters*, vol. 1, pp. 150–152, Oct. 1994.

[30] D. Gerlach, *Adaptive transmitting antenna arrays at the base station in mobile radio networks*. PhD thesis, Stanford University, Stanford, 1995.

Chapter 2

LINEAR ALGEBRA BACKGROUND



Contents

Throughout the course, several linear algebra concepts such as subspaces, QR factorizations, singular value decompositions (SVDs) and eigenvalue decompositions (EVDs) play an omni-important role. This chapter gives a brief review of the most important properties as needed here. Tutorial information as well as related technical papers on the subject of SVD and signal processing can be found in [1] and the series [2, 3]. Suitable reference books on advanced matrix algebra are Golub and Van Loan [4], and Horn and Johnson [5].

2.1 DEFINITIONS

Notation A bold-face letter, such as $\mathbf{x}$, denotes a vector (usually a column vector, but occasionally a row vector). Matrices are written with capital bold letters. A matrix $\mathbf{A}$ has entries a_{ij}, and columns $\mathbf{a}_j$, and we can write

$$\mathbf{A} = [a_{ij}] = [\mathbf{a}_1 \quad \mathbf{a}_2 \ \cdots \ \mathbf{a}_N] \, .$$

The $N \times N$ identity matrix is denoted by $\mathbf{I}_N$.

Complex conjugate is denoted by an overbar, the transpose of a matrix is denoted by $\mathbf{A}^{\mathrm{T}} = [a_{ji}]$. For complex matrices, the complex conjugate (= hermitian) transpose is $\mathbf{A}^{\mathrm{H}} := \overline{\mathbf{A}}^{\mathrm{T}}$.

For two matrices $\mathbf{A}$ and $\mathbf{B}$, the Kronecker product is defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1N}\mathbf{B} \\ \vdots & & \vdots \\ a_{M1}\mathbf{B} & \cdots & a_{MN}\mathbf{B} \end{bmatrix},$$

and the Schur-Hadamard product as

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & \cdots & a_{1N}b_{1N} \\ \vdots & & \vdots \\ a_{M1}b_{M1} & \cdots & a_{MN}b_{MN} \end{bmatrix},$$

provided $\mathbf{A}$ and $\mathbf{B}$ have the same size.

Vector norm Let $\mathbf{x} \in \mathbb{C}^N$ be an N-dimensional complex vector. The Euclidean norm (2-norm) of $\mathbf{x}$ is

$$\| \mathbf{x} \| \;:=\; \Big(\sum_{i=1}^{N} |x_i|^2\Big)^{1/2} = \; \Big(\sum_{i=1}^{N} x_i^{\mathrm{H}} x_i\Big)^{1/2}$$

Matrix norms Let $\mathbf{A} \in \mathbb{C}^{M\times N}$ be an $M \times N$ complex matrix. The *induced matrix 2-norm* (also called the spectral norm, or the operator norm) is

$$\| \mathbf{A} \| := \max_{\mathbf{x}} \frac{\|\mathbf{A}\mathbf{x}\|}{\|\mathbf{x}\|}$$

It represents the largest magnification that can be obtained by applying $\mathbf{A}$ to any vector. Another expression for this is

$$\| \mathbf{A} \|^2 = \max_{\mathbf{x}} \frac{\mathbf{x}^{\mathrm{H}}\mathbf{A}^{\mathrm{H}}\mathbf{A}\mathbf{x}}{\mathbf{x}^{\mathrm{H}}\mathbf{x}}$$

The *Frobenius norm* of $\mathbf{A}$ represents the energy contained in its entries:

$$\| \mathbf{A} \|_{\mathrm{F}} = \Big(\sum_{i=1}^{M}\sum_{j=1}^{M} |a_{ij}|^2\Big)^{1/2}$$

Subspace The space $\mathcal{H}$ spanned by a collection of vectors $\{\mathbf{x}_i\}$

$$\mathcal{H} := \{\alpha_1\mathbf{x}_1 + \cdots + \alpha_N\mathbf{x}_N \mid \alpha_i \in \mathbb{C}\,,\ \forall i\}$$

is called a *linear subspace*

Important examples of subspaces are

Range (column span) of $\mathbf{A}$:	$\mathrm{ran}(\mathbf{A}) = \{\mathbf{A}\mathbf{x} :\ \mathbf{x} \in \mathbb{C}^N\}$
Kernel (row nullspace) of $\mathbf{A}$:	$\ker(\mathbf{A}) = \{\mathbf{x} \in \mathbb{C}^N :\ \mathbf{A}\mathbf{x} = 0\}$

One routinely shows that

$$\begin{array}{lcll} \mathrm{ran}(\mathbf{A}) & \oplus & \ker(\mathbf{A}^{\mathrm{H}}) & = & \mathbb{C}^M \\ \mathrm{ran}(\mathbf{A}^{\mathrm{H}}) & \oplus & \ker(\mathbf{A}) & = & \mathbb{C}^N \end{array}$$

Here, $\mathcal{H}_1 \oplus \mathcal{H}_2$ denotes the direct sum of two linearly independent subspaces, namely $\{\mathbf{x}_1 + \mathbf{x}_2 \mid \mathbf{x}_1 \in \mathcal{H}_1,\ \mathbf{x}_2 \in \mathcal{H}_2\}$.

Linear independence A collection of vectors $\{\mathbf{x}_i\}$ is called linearly independent if

$$\alpha_1\mathbf{x}_1 + \cdots + \alpha_N\mathbf{x}_N = 0 \quad \Leftrightarrow \quad \alpha_1 = \cdots = \alpha_N = 0\,.$$

Basis An independent collection of vectors $\{\mathbf{x}_i\}$ that together span a subspace is called a *basis* for that subspace. If the vectors are orthogonal ($\mathbf{x}_i^{\mathrm{H}}\mathbf{x}_j = 0,\ i \neq j$), it is an *orthogonal basis*. If the vectors are orthonormal ($\mathbf{x}_i^{\mathrm{H}}\mathbf{x}_j = 0,\ i \neq j$ and $\|\mathbf{x}_i\| = 1$) it is an *orthonormal basis*.

Rank The *rank* of a matrix $\mathbf{A}$ is the number of independent columns (or rows) of $\mathbf{A}$. A prototype rank-1 matrix is $\mathbf{A} = \mathbf{a}\mathbf{b}^{\mathrm{H}}$, a prototype rank-2 matrix is $\mathbf{A} = \mathbf{a}\mathbf{b}^{\mathrm{H}} + \mathbf{c}\mathbf{d}^{\mathrm{H}}$, etc. The rank cannot be larger than the smallest size of the matrix (when it is equal, the matrix is full rank, otherwise it is rank deficient). A tall matrix is said to have full column rank if the rank is equal to the number of columns: the columns are independent. Similary, a wide matrix has full row rank if its rank equals the number of rows.

Unitary matrix A real (square) matrix $\mathbf{U}$ is called an orthogonal matrix if $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}$, and $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{I}$. Likewise, a complex matrix $\mathbf{U}$ is unitary if $\mathbf{U}^{\mathrm{H}}\mathbf{U} = \mathbf{I}$, $\mathbf{U}\mathbf{U}^{\mathrm{H}} = \mathbf{I}$. A unitary matrix looks like a rotation and/or a reflection. Its norm is $\|\,\mathbf{U}\,\| = 1$, and its columns are orthonormal.

Isometry A tall matrix $\hat{\mathbf{U}}$ is called an isometry if $\hat{\mathbf{U}}^{\mathrm{H}}\hat{\mathbf{U}} = \mathbf{I}$. Its columns are an orthonormal basis of a subspace (not the complete space). Its norm is $\|\,\hat{\mathbf{U}}\,\| = 1$. There is an orthogonal complement $\hat{\mathbf{U}}^{\perp}$ of $\hat{\mathbf{U}}$ such that $\mathbf{U} = [\hat{\mathbf{U}}\ \ \hat{\mathbf{U}}^{\perp}]$ is square and unitary.

Projection A square matrix $\mathbf{P}$ is a projection if $\mathbf{P}\mathbf{P} = \mathbf{P}$. It is an orthogonal projection if also $\mathbf{P}^{\mathrm{H}} = \mathbf{P}$.

The norm of an orthogonal projection is $\|\,\mathbf{P}\,\| = 1$. For an isometry $\hat{\mathbf{U}}$, the matrix $\mathbf{P} = \hat{\mathbf{U}}\hat{\mathbf{U}}^{\mathrm{H}}$ is an orthogonal projection onto the space spanned by the columns of $\hat{\mathbf{U}}$. This is the general form of an orthogonal projection.

Suppose $\mathbf{U} = [\underbrace{\hat{\mathbf{U}}}_{d} \quad \underbrace{\hat{\mathbf{U}}^{\perp}}_{M-d}]$ is unitary. Then,

1. from $\mathbf{U}^{\mathrm{H}}\mathbf{U} = \mathbf{I}_M$:

$$\hat{\mathbf{U}}^{\mathrm{H}}\hat{\mathbf{U}} = \mathbf{I}_d\,, \qquad \hat{\mathbf{U}}^{\mathrm{H}}\hat{\mathbf{U}}^{\perp} = 0\,, \qquad (\hat{\mathbf{U}}^{\perp})^{\mathrm{H}}\hat{\mathbf{U}}^{\perp} = \mathbf{I}_{M-d}\,.$$

2. from $\mathbf{U}\mathbf{U}^{\mathrm{H}} = \mathbf{I}_M$:

$$\hat{\mathbf{U}}\hat{\mathbf{U}}^{\mathrm{H}} + \hat{\mathbf{U}}^{\perp}(\hat{\mathbf{U}}^{\perp})^{\mathrm{H}} = \mathbf{I}_M\,, \qquad \hat{\mathbf{U}}\hat{\mathbf{U}}^{\mathrm{H}} = \mathbf{P}_c\,, \qquad \hat{\mathbf{U}}^{\perp}(\hat{\mathbf{U}}^{\perp})^{\mathrm{H}} = \mathbf{P}_c^{\perp} \;=\; \mathbf{I}_M - \mathbf{P}_c$$

This shows that any vector $\mathbf{x} \in \mathbb{C}^M$ can be decomposed into $\mathbf{x} = \hat{\mathbf{x}} + \hat{\mathbf{x}}^{\perp}$, where $\hat{\mathbf{x}} \perp \hat{\mathbf{x}}^{\perp}$,

$$\hat{\mathbf{x}} = \mathbf{P}_c\mathbf{x} \in \mathrm{ran}(\hat{\mathbf{U}})\,, \qquad \hat{\mathbf{x}}^{\perp} = \mathbf{P}_c^{\perp}\mathbf{x} \in \mathrm{ran}(\hat{\mathbf{U}}^{\perp})$$

The matrices $\hat{\mathbf{U}}\hat{\mathbf{U}}^{\mathrm{H}} = \mathbf{P}_c$ and $\hat{\mathbf{U}}^{\perp}(\hat{\mathbf{U}}^{\perp})^{\mathrm{H}} = \mathbf{P}_c^{\perp}$ are the orthogonal projectors onto the column span of $\mathbf{X}$ and its orthogonal complement in $\mathbb{C}^M$ respectively.

Similarly, we can find a matrix $\hat{\mathbf{V}}^{\mathrm{H}}$ whose rows span the row span of $\mathbf{X}$, and augment it with a matrix $\hat{\mathbf{V}}^{\perp}$ to a unitary matrix $\mathbf{V}$:

$$\mathbf{V} = {}_N\!\updownarrow \overset{\overset{d}{\leftrightarrow}}{\left[\hat{\mathbf{V}}\right.} \quad \overset{\overset{N-d}{\leftrightarrow}}{\left.\hat{\mathbf{V}}^{\perp}\right]}.$$

The matrices $\hat{\mathbf{V}}\hat{\mathbf{V}}^{\mathrm{H}} = \mathbf{P}_r$ and $\hat{\mathbf{V}}^{\perp}(\hat{\mathbf{V}}^{\perp})^{\mathrm{H}} = \mathbf{P}_r^{\perp}$ are orthogonal projectors onto the original subspaces in $\mathbb{C}^N$ spanned by the columns of $\hat{\mathbf{V}}$ and $\hat{\mathbf{V}}^{\perp}$, respectively. The columns of $\hat{\mathbf{V}}^{\perp}$ span the kernel (or nullspace) of $\mathbf{X}$, i.e., the space of vectors $\mathbf{a}$ for which $\mathbf{X}\mathbf{a} = \mathbf{0}$.

2.2 THE QR FACTORIZATION

Let $\mathbf{X}:\ N \times N$ be a square matrix of full rank. Then there is a decomposition $\mathbf{X} = \mathbf{Q}\mathbf{R}$,

$$\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N\end{bmatrix} = \begin{bmatrix}\mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_N\end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1N} \\ 0 & r_{22} & \cdots & r_{2N} \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & r_{NN} \end{bmatrix}$$

The interpretation is that $\mathbf{q}_1$ is a normalized vector with the same direction as $\mathbf{x}_1$, similarly $[\mathbf{q}_1 \;\; \mathbf{q}_2]$ is an isometry spanning the same space as $[\mathbf{x}_1 \;\; \mathbf{x}_2]$, etc.

In general, if $\mathbf{X}:\ M \times N$ is a tall matrix ($M \geq N$), then there is a decomposition

$$\mathbf{X} = \mathbf{Q}\mathbf{R} = [\hat{\mathbf{Q}} \quad \hat{\mathbf{Q}}^{\perp}] \begin{bmatrix} \hat{\mathbf{R}} \\ 0 \end{bmatrix} = \hat{\mathbf{Q}}\hat{\mathbf{R}}.$$

Here, $\mathbf{Q}$ is a unitary matrix, $\hat{\mathbf{R}}$ is upper triangular and square. $\mathbf{R}$ is upper triangular with $M - N$ zero rows added. $\mathbf{X} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$ is called an "economy-size" QR.

If $\hat{\mathbf{R}}$ is nonsingular (all entries on the main diagonal are invertible), then $\mathbf{X}$ has full column rank N, the columns of $\hat{\mathbf{Q}}$ form a basis of the column span of $\mathbf{X}$, and $\mathbf{P}_c = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^{\mathrm{H}}$. If $\hat{\mathbf{R}}$ is rank-deficient, then this is not true: the column span of $\hat{\mathbf{Q}}$ is too large. However, the QR

factorization can be used as a start in the estimation of an orthogonal basis for the column span of $\mathbf{X}$. Although this has sometimes been attempted, it is numerically not very robust to use the QR directly to estimate the rank of a matrix. (Modifications such as a "rank-revealing QR" do exist.)

Likewise, for a "wide" matrix ($M \leq N$) we can define an RQ factorization

$$\mathbf{X} = \mathbf{RQ} = [\hat{\mathbf{R}} \quad 0] \begin{bmatrix} \hat{\mathbf{Q}} \\ \hat{\mathbf{Q}}^{\perp} \end{bmatrix}$$

(for different $\mathbf{Q}$ and $\mathbf{R}$). Now, $\mathbf{X}$ and $\hat{\mathbf{R}}$ have the same singular values and *left* singular vectors.

2.3 THE SINGULAR VALUE DECOMPOSITION (SVD)

Any $M \times N$ matrix $\mathbf{X}$ of rank d admits the following factorization, which is called the *singular value decomposition* [4]

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}} = [\hat{\mathbf{U}} \quad \hat{\mathbf{U}}^{\perp}] \left[\begin{array}{ccc|ccc} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_d & & & \\ \hline & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \\ \hline 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{array}\right] \begin{bmatrix} \hat{\mathbf{V}}^{\mathrm{H}} \\ (\hat{\mathbf{V}}^{\perp})^{\mathrm{H}} \end{bmatrix}$$

where $\mathbf{U}: \ M \times M$ and $\mathbf{V}: \ N \times N$ are unitary, and $\mathbf{\Sigma}$ is an $M \times N$ diagonal matrix[1] containing the so-called singular values σ_i of $\mathbf{X}$. These are positive real numbers ordered such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d > \sigma_{d+1} = \cdots = \sigma_N = 0\,.$$

Note that only d singular values are non-zero. The d columns of $\hat{\mathbf{U}}$ corresponding to these non-zero singular values span the column space of $\mathbf{X}$ and are called the left singular vectors. Similarly, the d columns of $\hat{\mathbf{V}}$ are called the right singular vectors and span the row space of $\mathbf{X}$ (or the column space of $\mathbf{X}^{\mathrm{H}}$). In terms of these (sometimes much) smaller matrices, the SVD of $\mathbf{X}$ can also be written in 'economy' size:

$$\mathbf{X} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^{\mathrm{H}}\,, \tag{2.1}$$

where $\hat{\mathbf{\Sigma}}$ is a $d \times d$ diagonal matrix containing $\sigma_1, \cdots, \sigma_d$. This form of the SVD better reveals that $\mathbf{X}$ is actually of rank d: it is constructed from a product of rank-d matrices.

The SVD of $\mathbf{X}$ makes the various spaces (range and kernel) associated with $\mathbf{X}$ explicit. So does any decomposition of $\mathbf{X}$ as $\mathbf{X} = \hat{\mathbf{U}}\mathbf{E}_x\hat{\mathbf{V}}^{\mathrm{H}}$, where $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ are any matrices whose columns

[1] The equation shows $\mathbf{\Sigma}$ as a tall matrix ($M \geq N$); if $M \leq N$ the structure is transposed.

span the column and row spaces of $\mathbf{X}$, respectively, and where $\mathbf{E}_x$ is an invertible $d \times d$ matrix. The property that makes the SVD special is the fact that $\mathbf{E}_x$ is a diagonal matrix, so that a decoupling is obtained: with $\mathbf{u}_i$ the i-th column of $\mathbf{U}$, and $\mathbf{v}_i$ likewise for $\mathbf{V}$, $\mathbf{X}$ can be written as a sum of rank-1 isometric matrices $\mathbf{u}_i\mathbf{v}_i^{\mathrm{H}}$, scaled by σ_i:

$$\mathbf{X} = \sum_{i=1}^{d} \sigma_i(\mathbf{u}_i\mathbf{v}_i^{\mathrm{H}}) \,,$$

and we also have

$$\sigma_i\mathbf{u}_i = \mathbf{X}\mathbf{v}_i \,, \qquad \sigma_i\mathbf{v}_i = \mathbf{X}^{\mathrm{H}}\mathbf{u}_i \,.$$

This makes it possible to order the vectors in the column span and row span of $\mathbf{X}$: the most important direction in the column space is $\mathbf{u}_1$, with scale σ_1, and is reached by applying $\mathbf{X}$ to the vector $\mathbf{v}_1$. The second most important direction is $\mathbf{u}_2$, etc. This ranking will in turn lead to optimal low-rank approximants of $\mathbf{X}$ (see below). In the mapping $\mathbf{a} \in \mathbb{C}^N \rightarrow \mathbf{b} \in \mathbb{C}^M : \mathbf{b} = \mathbf{X}\mathbf{a}$, $\mathbf{b}$ will automatically be a vector in the column range of $\mathbf{X}$, and will be non-zero if and only if $\mathbf{a}$ has a component in the row space of $\mathbf{X}$; i.e., if and only if $\mathbf{P}_r\mathbf{a}$ is non-zero. On the other hand, $\mathbf{b}$ will be identically zero if and only if $\mathbf{a}$ is orthogonal to the row space of $\mathbf{X}$. Therefore, the space spanned by the vectors $\mathbf{v}_{d+1}, \cdots, \mathbf{v}_n$ in $\hat{\mathbf{V}}^{\perp}$ is called the null space (or kernel) of $\mathbf{X}$. Vectors $\mathbf{a}$ in this space are mapped to zero by one of the zero singular values of $\mathbf{X}$. The SVD of $\mathbf{X}$ reveals the behavior of the map $\mathbf{b} = \mathbf{X}\mathbf{a}$: $\mathbf{a}$ is rotated in N-space (by $\mathbf{V}^{\mathrm{H}}$), then scaled (by the entries of $\mathbf{\Sigma}$: $M - d$ components are projected to zero), and finally rotated in M-space (by $\mathbf{U}$) to give $\mathbf{b}$.

A summary of some useful properties of the SVD:

- The rank of $\mathbf{X}$ is d, the number of nonzero singular values. The SVD is one of the numerically most reliable techniques to establish the rank of a matrix.
- $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}} \quad \Leftrightarrow \quad \mathbf{X}\mathbf{V} = \mathbf{U}\mathbf{\Sigma}\,, \quad \hat{\mathbf{X}}\hat{\mathbf{V}} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}.$

  The columns of $\hat{\mathbf{U}}$ are an orthonormal basis for the range of $\mathbf{X}$. The columns of $\hat{\mathbf{V}}^{\perp}$ are an orthonormal basis for the kernel of $\mathbf{X}$.
- The norm of $\mathbf{X}$ or $\mathbf{X}^{\mathrm{H}}$ is $\| \mathbf{X} \| = \| \mathbf{X}^{\mathrm{H}} \| = \sigma_1$, the largest singular value. The norm is attained on the corresponding singular vectors $\mathbf{u}_1$ and $\mathbf{v}_1$:

  $$\mathbf{X}\mathbf{v}_1 = \mathbf{u}_1\sigma_1, \qquad \mathbf{X}^{\mathrm{H}}\mathbf{u}_1 = \mathbf{v}_1\sigma_1.$$

- For an arbitrary matrix $\mathbf{X}$, perhaps of full rank, the best rank-d approximant $\hat{\mathbf{X}}$ is obtained by computing $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}}$, and then setting all but the first d singular values in $\mathbf{\Sigma}$ equal to zero:

  $$\hat{\mathbf{X}} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^{\mathrm{H}} \,,$$

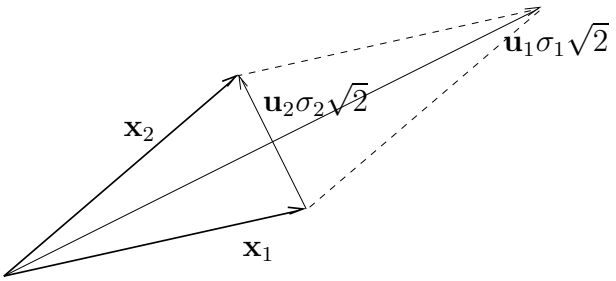


Figure 2.1. Construction of the left singular vectors and values of the matrix $X = [\mathbf{x}_1 \;\; \mathbf{x}_2]$, where $\mathbf{x}_1$ and $\mathbf{x}_2$ have equal length.

The approximation error in Frobenius norm and operator norm is given by

$$\begin{aligned} \| \mathbf{X} - \hat{\mathbf{X}} \|_{\mathrm{F}}^2 &= \sum_{i=d+1}^{N} \sigma_i^2 \\ \| \mathbf{X} - \hat{\mathbf{X}} \|^2 &= \sigma_{d+1}^2 \end{aligned}$$

The QR factorization can be used as a start in the computation of the SVD of a tall matrix $\mathbf{X}$. We first compute

$$\mathbf{X} = \hat{\mathbf{Q}}\hat{\mathbf{R}}\,.$$

The next step is to continue with an SVD of $\hat{\mathbf{R}}$:

$$\hat{\mathbf{R}} = \hat{\mathbf{U}}_R\hat{\mathbf{\Sigma}}_R\hat{\mathbf{V}}_R^{\mathrm{H}}\,,$$

so that the SVD of $\mathbf{X}$ is

$$\mathbf{X} = (\hat{\mathbf{Q}}\hat{\mathbf{U}}_R)\hat{\mathbf{\Sigma}}_R\hat{\mathbf{V}}_R^{\mathrm{H}}\,,$$

The preprocessing by QR in computing the SVD is useful because it reduces the size from $\mathbf{X}$ to that of $\hat{\mathbf{R}}$, and obviously, $\mathbf{X}$ and $\hat{\mathbf{R}}$ have the same singular values and *right* singular vectors.

Example 2.1. Figure 2.1 shows the construction of the left singular vectors of a matrix $\mathbf{X} = [\mathbf{x}_1 \;\; \mathbf{x}_2]$, whose columns $\mathbf{x}_1$ and $\mathbf{x}_2$ are of equal length. The largest singular vector $\mathbf{u}_1$ is in the direction of the sum of $\mathbf{x}_1$ and $\mathbf{x}_2$, i.e., the "common" direction of the two vectors, and the corresponding singular value σ_1 is equal to $\sigma_1 = \| \mathbf{x}_1 + \mathbf{x}_2 \|/\sqrt{2}$. On the other hand, the smallest singular vector $\mathbf{u}_2$ is dependent on the difference $\mathbf{x}_2 - \mathbf{x}_1$, as is its corresponding singular value: $\sigma_2 = \| \mathbf{x}_2 - \mathbf{x}_1 \|/\sqrt{2}$. If $\mathbf{x}_1$ and $\mathbf{x}_2$ become more aligned, then σ_2 will be smaller and $\mathbf{X}$ will be closer to a singular matrix. Clearly, $\mathbf{u}_2$ is the most sensitive direction for perturbations on $\mathbf{x}_1$ and $\mathbf{x}_2$.

An example of such a matrix could be $\mathbf{A} = [\mathbf{a}(\theta_1) \;\; \mathbf{a}(\theta_2)]$, where θ is the direction at which a signal hits an antenna array. If two directions are close together, then

Table 2.1. Singular values of $\mathbf{X}_{M,N}$.

$M = 3,$	$\sigma_1 = 3.44$	$M = 3,$	$\sigma_1 = 4.86$
$N = 3$	$\sigma_2 = 0.44$	$N = 6$	$\sigma_2 = 0.63$
$M = 6,$	$\sigma_1 = 4.73$		
$N = 3$	$\sigma_2 = 1.29$		

$\theta_1 \approx \theta_2$ and $\mathbf{a}(\theta_1)$ points in about the same direction as $\mathbf{a}(\theta_2)$, which will be the direction of $\mathbf{u}_1$. The smallest singular value, σ_2, is dependent on the difference of the directions of $\mathbf{a}(\theta_1)$ and $\mathbf{a}(\theta_2)$.

For further illustration, consider the following small numerical experiment. Consider a uniform linear array with $\Delta \sin(\theta_1) = 0$ and $\Delta \sin(\theta_2) = 0.05$, and construct $M \times N$ matrices $\mathbf{X} = [\mathbf{a}(\theta_1) \;\; \mathbf{a}(\theta_2)]\mathbf{S}$, where $\mathbf{S}\mathbf{S}^{\mathrm{H}} = N\mathbf{I}$. Since $(1/\sqrt{N})\mathbf{S}$ is co-isometric, the singular values of $\mathbf{X}$ are those of $\mathbf{A} = [\mathbf{a}(\theta_1) \;\; \mathbf{a}(\theta_2)]$ times $\sqrt{N}$. The two non-zero singular values of $\mathbf{X}$ for some values of M, N are given in Table 2.1. It is seen that doubling M almost triples the smallest singular value, whereas doubling N only increases the singular values by a factor $\sqrt{2}$, which is because the matrices have larger size.

2.4 PSEUDO-INVERSE

Consider a rank-d $M \times N$ matrix $\mathbf{X}$. In general, since $\mathbf{X}$ may be rank-deficient or non-square, the inverse of $\mathbf{X}$ does not exist; i.e., for a given vector $\mathbf{b}$, we cannot always find a vector $\mathbf{a}$ such that $\mathbf{b} = \mathbf{X}\mathbf{a}$.

If $\mathbf{X}$ is tall but of full rank, the *pseudo-inverse* of $\mathbf{X}$ is $\mathbf{X}^\dagger = (\mathbf{X}^{\mathrm{H}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{H}}$. It satisfies

$$\begin{aligned} \mathbf{X}^\dagger\mathbf{X} &= \mathbf{I}_N \\ \mathbf{X}\mathbf{X}^\dagger &= \mathbf{P}_c \end{aligned}$$

Thus, $\mathbf{X}^\dagger$ is an inverse on the "short space", and $\mathbf{X}\mathbf{X}^\dagger$ is a projection onto the column span of $\mathbf{X}$. It is easy to verify that the solution to $\mathbf{b} = \mathbf{X}\mathbf{a}$ is given by $\mathbf{a} = \mathbf{X}^\dagger\mathbf{b}$.

If $\mathbf{X}$ is rank deficient, then $\mathbf{X}^{\mathrm{H}}\mathbf{X}$ is not invertible, and there is no exact solution to $\mathbf{b} = \mathbf{X}\mathbf{a}$. In this case, we can resort to the Moore-Penrose pseudo-inverse of $\mathbf{X}$, also denoted by $\mathbf{X}^\dagger$. It can be defined in terms of the "economy size" SVD $\mathbf{X} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^{\mathrm{H}}$ (equation (2.1)) as

$$\mathbf{X}^\dagger = \hat{\mathbf{V}}\hat{\mathbf{\Sigma}}^{-1}\hat{\mathbf{U}}^{\mathrm{H}} \, .$$

This pseudo-inverse satisfies the properties

1. $\mathbf{X}\mathbf{X}^\dagger\mathbf{X} = \mathbf{X}$
2. $\mathbf{X}^\dagger\mathbf{X}\mathbf{X}^\dagger = \mathbf{X}^\dagger$
3. $\mathbf{X}\mathbf{X}^\dagger = \mathbf{P}_c$
4. $\mathbf{X}^\dagger\mathbf{X} = \mathbf{P}_r$

which constitute the Moore-Penrose inverse in the traditional way.

These equations show that, in order to make the problem $\mathbf{b} = \mathbf{X}\mathbf{a}$ solvable, a solution can be forced to an approximate problem by projecting $\mathbf{b}$ onto the column space of $\mathbf{X}$:

$$\mathbf{b}' = \mathbf{P}_c\mathbf{b}\,,$$

after which $\mathbf{b}' = \mathbf{X}\mathbf{a}$ has solution

$$\mathbf{a} = \mathbf{X}^\dagger\mathbf{b}'\,.$$

The projection is in fact implicitly done by just taking $\mathbf{a} = \mathbf{X}^\dagger\mathbf{b}$: from properties 1 and 3 of the list above, we have that

$$\mathbf{a} = \mathbf{X}^\dagger\mathbf{b}' = \mathbf{X}^\dagger\mathbf{X}\mathbf{X}^\dagger\mathbf{b} = \mathbf{X}^\dagger\mathbf{b}$$

It can be shown that this solution $\mathbf{a}$ is the solution of the (Least Squares) minimization problem

$$\min_{\mathbf{a}} \| \mathbf{b} - \mathbf{X}\mathbf{a} \|^2\,,$$

where $\mathbf{a}$ is chosen to have minimal norm if there is more than one solution (the latter requirement translates to $\mathbf{a} = \mathbf{P}_r\mathbf{a}$).

Some other properties of the pseudo-inverse are

- The norm of $\mathbf{X}^\dagger$ is $\| \mathbf{X}^\dagger \| = \sigma_d^{-1}$.
- The *condition number* of $\mathbf{X}$ is $c(\mathbf{X}) := \frac{\sigma_1}{\sigma_d}$.

  If it is large, then $\mathbf{X}$ is hard to invert ($\mathbf{X}^\dagger$ is sensitive to small changes).

Total Least Squares Now, suppose that instead of a single vector $\mathbf{b}$ we are given an $(M \times N)$-dimensional matrix $\mathbf{Y}$, the columns of which are not all in the column space of the matrix $\mathbf{X}$. We want to force solutions to $\mathbf{X}\mathbf{A} = \mathbf{Y}$. Clearly, we can use a least squares approximation $\hat{\mathbf{Y}} = \mathbf{P}_\mathbf{X}\mathbf{Y}$ to force the columns of $\hat{\mathbf{Y}}$ to be in the d-dimensional column space of $\mathbf{X}$. This is reminiscent to the LS application above, but just one way to arrive at $\mathbf{X}$ and $\mathbf{Y}$ having a common column space, in this case by only modifying $\mathbf{Y}$. There is another way, called Total Least Squares (TLS) which is effectively described as projecting both $\mathbf{X}$ and $\mathbf{Y}$ onto some subspace that lies between them, and that is "closest" to the column spaces of the two matrices. To implement this method, we compute the SVD

$$[\mathbf{X} \quad \mathbf{Y}] \;=\; [\hat{\mathbf{U}} \quad \hat{\mathbf{U}}^\perp]\,\mathbf{\Sigma}\begin{bmatrix} \hat{\mathbf{V}}^{\mathrm{H}} \\ (\hat{\mathbf{V}}^\perp)^{\mathrm{H}} \end{bmatrix} \;=\; \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\,[\hat{\mathbf{V}}_1^{\mathrm{H}} \quad \hat{\mathbf{V}}_2^{\mathrm{H}}] + \hat{\mathbf{U}}^\perp\hat{\mathbf{\Sigma}}^\perp(\hat{\mathbf{V}}^\perp)^{\mathrm{H}}$$

and define the projection $\mathbf{P}_c = \hat{\mathbf{U}}\hat{\mathbf{U}}^{\mathrm{H}}$. We now take the TLS (column space) approximations to be $\hat{\mathbf{X}} = \mathbf{P}_c\mathbf{X} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}_1^{\mathrm{H}}$ and $\hat{\mathbf{Y}} = \mathbf{P}_c\mathbf{Y} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}_2^{\mathrm{H}}$, where $\hat{\mathbf{V}}_1$ and $\hat{\mathbf{V}}_2$ are the partitions of $\hat{\mathbf{V}}$

corresponding to $\mathbf{X}$ and $\mathbf{Y}$ respectively. $\hat{\mathbf{X}}$ and $\hat{\mathbf{Y}}$ have the same column span defined by $\hat{\mathbf{U}}$, and are in fact solutions to

$$\min_{[\hat{\mathbf{X}}\ \hat{\mathbf{Y}}]\ \text{rank}\ N} \| [\mathbf{X}\ \ \mathbf{Y}] - [\hat{\mathbf{X}}\ \ \hat{\mathbf{Y}}] \|_{\mathrm{F}}^2$$

and $\mathbf{A}$ satisfying $\hat{\mathbf{X}}\mathbf{A} = \hat{\mathbf{Y}}$ is obtained as $\mathbf{A} = \hat{\mathbf{X}}^{\dagger}\hat{\mathbf{Y}}$. This $\mathbf{A}$ is the TLS solution of $\mathbf{X}\mathbf{A} \approx \mathbf{Y}$. Instead of asking for rank N, we might even insist on a lower rank d.

2.5 THE EIGENVALUE PROBLEM

The *eigenvalue problem* for a matrix $\mathbf{A}$ is

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \quad \Leftrightarrow \quad (\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = \mathbf{0}$$

Any λ that makes $\mathbf{A} - \lambda\mathbf{I}$ singular is called an eigenvalue, the corresponding $\mathbf{x}$ is the eigenvector (invariant vector). It has an arbitrary norm usually set equal to 1.

We can collect the eigenvectors in a matrix:

$$\begin{aligned} \mathbf{A}[\mathbf{x}_1\ \ \mathbf{x}_2\ \cdots] &= [\mathbf{x}_1\ \ \mathbf{x}_2\ \cdots] \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \ddots \end{bmatrix} \\ \Leftrightarrow \qquad \mathbf{A}\mathbf{T} &= \mathbf{T}\boldsymbol{\Lambda} \end{aligned}$$

A *regular* matrix $\mathbf{A}$ has an *eigenvalue decomposition:*

$$\mathbf{A} = \mathbf{T}\boldsymbol{\Lambda}\mathbf{T}^{-1}\,,$$

where $\mathbf{T}$ is invertible and $\boldsymbol{\Lambda}$ is diagonal. This decomposition might not exist if eigenvalues are repeated.

Schur decomposition Suppose $\mathbf{T}$ has a QR factorization $\mathbf{T} = \mathbf{Q}\mathbf{R}_T$, so that $\mathbf{T}^{-1} = \mathbf{R}_T^{-1}\mathbf{Q}^{\mathrm{H}}$. Then

$$\mathbf{A} \ = \ \mathbf{Q}\mathbf{R}_T\boldsymbol{\Lambda}\mathbf{R}_T^{-1}\mathbf{Q}^{\mathrm{H}} \ = \ \mathbf{Q}\mathbf{R}\mathbf{Q}^{\mathrm{H}}.$$

The factorization

$$\mathbf{A} = \mathbf{Q}\mathbf{R}\mathbf{Q}^{\mathrm{H}}\,,$$

with $\mathbf{Q}$ unitary and $\mathbf{R}$ upper triangular, is called a *Schur decomposition.* One can show that this decomposition always exists. $\mathbf{R}$ has the eigenvalues of $\mathbf{A}$ on the diagonal. $\mathbf{Q}$ gives information about "eigen-subspaces" (invariant subspaces), but doesn't contain eigenvectors.

Connection to the SVD Suppose we compute the SVD of a matrix $\mathbf{X}$, and then consider $\mathbf{X}\mathbf{X}^{\mathrm{H}}$:

$$\begin{aligned} \mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}} \quad \Rightarrow \quad \mathbf{X}\mathbf{X}^{\mathrm{H}} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}}\mathbf{V}\mathbf{\Sigma}\mathbf{U}^{\mathrm{H}} \\ &= \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^{\mathrm{H}} \\ &= \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}} \end{aligned}$$

This shows that the eigenvalues of $\mathbf{X}\mathbf{X}^{\mathrm{H}}$ are the singular values of $\mathbf{X}$, squared (hence real). The eigenvectors of $\mathbf{X}\mathbf{X}^{\mathrm{H}}$ are equal to the left singular vectors of $\mathbf{X}$ (hence $\mathbf{U}$ is unitary). Since the SVD always exists, the eigenvalue decomposition of $\mathbf{X}\mathbf{X}^{\mathrm{H}}$ always exists. (In fact it exists for any Hermitian matrix $\mathbf{C} = \mathbf{C}^{\mathrm{H}}$.)

Bibliography

[1] L. Scharf, "The SVD and reduced-rank signal processing," in *SVD and Signal Processing, II: Algorithms, Analysis and Applications* (R. Vaccaro, ed.), pp. 3–31, Elsevier, 1991.

[2] E. Deprettere, ed., *SVD and Signal Processing: Algorithms, Applications and Architectures*. North-Holland, 1988.

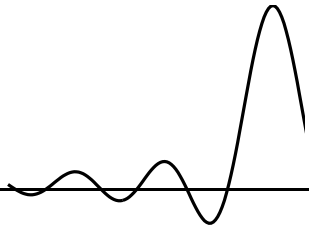
[3] R. Vaccaro, ed., *SVD and Signal Processing, II: Algorithms, Analysis and Applications*. Elsevier, 1991.

[4] G. Golub and C. Van Loan, *Matrix Computations*. The Johns Hopkins University Press, 1989.

[5] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge, NY: Cambridge Univ. Press, 1985.

Chapter 3

SPATIAL PROCESSING TECHNIQUES



Contents

In this chapter, we look at elementary receiver schemes: the matched filter and Wiener filter in their non-adaptive forms. They are suitable if we have a good estimate of the channel, or if we know a segment of the transmitted data, e.g., because of a training sequence. These receivers are most simple in the context of narrowband antenna array processing, and hence we place the discussion first in this scenario. The matched filter is shown to maximize the output signal-to-noise ratio (in the case of a single signal in noise), whereas the Wiener receiver maximizes the output signal-to-interference plus noise (in the case of several sources in noise). We also look at the application of these receivers as non-parametric beamformers for direction-of-arrival estimation. Improved accuracy is possible using parametric data models and subspace-based techniques: a prime example is the MUSIC algorithm.

General references to this chapter are [1–7].

3.1 DETERMINISTIC APPROACH TO MATCHED AND WIENER FILTERS

3.1.1 Data model and assumptions

In this chapter, we consider a simple array signal processing model of the form

$$\mathbf{x}(t) \;=\; \sum_{i=1}^{d} \mathbf{a}_i s_i(t) + \mathbf{n}(t) \;=\; \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t).$$

Remember that if we are perfectly synchronized, sampling $s_i(t)$ at integer time instants results in the transmitted data symbols, i.e., $s_i(k) = s_{i,k}$. Hence, sampling $\mathbf{x}(t)$ at integer time instants, we obtain

$$\mathbf{x}_k \;:=\; \mathbf{x}(k) \;=\; \sum_{i=1}^{d} \mathbf{a}_i s_i(k) + \mathbf{n}(k) \;:=\; \sum_{i=1}^{d} \mathbf{a}_i s_{i,k} + \mathbf{n}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k \,. \tag{3.1}$$

We assume that signals are received by M antennas, and that the antenna outputs (after demodulation, sampling, A/D conversion) are stacked into vectors $\mathbf{x}_k$. According to the model, $\mathbf{x}_k$ is a linear combination of d narrowband source signals $s_{i,k}$ and noise $\mathbf{n}_k$. Often, we will consider an even simpler case where there is only one signal in noise. In all cases, we assume that the noise covariance matrix

$$\mathbf{R}_n := \mathrm{E}[\mathbf{n}_k \mathbf{n}_k^{\mathrm{H}}]$$

is known, up to a scalar which represents the noise power. The most simple situation is spatially white noise, for which

$$\mathbf{R}_n = \sigma^2 \mathbf{I} \,.$$

Starting from the data model (3.1), let us assume that we have collected N sample vectors. If we store the samples in an $M \times N$ matrix $\mathbf{X} = [\mathbf{x}_0 \,, \cdots \,, \mathbf{x}_{N-1}]$, then we obtain that $\mathbf{X}$ has a decomposition

$$\mathbf{X} = \mathbf{A}\mathbf{S} + \mathbf{N} \tag{3.2}$$

where the rows of $\mathbf{S} \in \mathbb{C}^{d \times N}$ contain the samples of the source signals. Note that we can choose to put the source powers in either $\mathbf{A}$ or $\mathbf{S}$, or even in a separate factor $\mathbf{B}$. Here we will assume they are absorbed in $\mathbf{A}$, thus the sources have unit powers. Sources may be considered either stochastic (with probability distributions) or deterministic. If they are stochastic, we assume they are zero mean, independent and hence uncorrelated,

$$\mathrm{E}[\mathbf{s}_k \mathbf{s}_k^{\mathrm{H}}] = \mathbf{I} \,.$$

If they are considered deterministic ($\mathrm{E}[\mathbf{s}_k] = \mathbf{s}_k$), we will assume similarly that

$$\lim_{N \to \infty} \; \tfrac{1}{N} \mathbf{S}\mathbf{S}^{\mathrm{H}} \;=\; \mathbf{I} \,.$$

The objective of beamforming is to construct a receiver weight vector $\mathbf{w}_i$ such that the output is

$$\mathbf{w}_i^{\mathrm{H}} \mathbf{x}_k = \hat{s}_{i,k}. \tag{3.3}$$

In other words, we want the output of the weight vector $\mathbf{w}_i$ to be an estimate of the i-th source. Which beamformer is "the best" depends on the optimality criterion, of which there are many. It also makes a difference if we wish to receive only a single signal, as in (3.3), or all d signals jointly,

$$\mathbf{W}^{\mathrm{H}} \mathbf{x}_k = \hat{\mathbf{s}}_k, \tag{3.4}$$

where $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_d]$.

We will first look at purely deterministic techniques to estimate the beamformers: here no explicit statistical assumptions are made on the data. The noise is viewed as a perturbation on the noise-free data $\mathbf{AS}$, and the perturbations are assumed to be small and equally important on all entries. Then we will look at more statistically oriented techniques. The noise will be modeled as a stochastic sequence with a joint Gaussian distribution. Still, we have a choice whether we consider $\mathbf{s}_k$ to be a deterministic sequence (known or unknown), or if we associate a probabilistic distribution to it, for example Gaussian or belonging to a certain alphabet such as $\{+1, -1\}$. In the latter case, we can often improve on the linear receiver (3.3) or (3.4) by taking into account that the output of the beamformer should belong to this alphabet (or should have a certain distribution). The resulting receivers will then contain some non-linear components.

In this chapter, we only consider the most simple cases, resulting in the classical linear beamformers.

3.1.2 Algebraic (purely deterministic) approach

Noiseless case Let us first consider the noiseless case, and a situation where we have collected N samples. Our data model thus is

$$\mathbf{X} = \mathbf{AS}\,.$$

Our objective will be to construct a linear beamforming matrix $\mathbf{W}$ such that

$$\mathbf{W}^{\mathrm{H}}\mathbf{X} = \mathbf{S}\,.$$

We consider two cases:

1. $\mathbf{A}$ is known, for example we know the directions and the complex gains of the sources and have set $\mathbf{A} = [\mathbf{a}(\theta_1)\ \cdots\ \mathbf{a}(\theta_d)]\mathrm{diag}[\beta_1\ \cdots\ \beta_d]$,
2. $\mathbf{S}$ is known, for example we have selected a segment of the data which contains a training sequence for all sources. Alternatively, for discrete alphabet sources (e.g., $s_{i,k} \in \{\pm 1\}$) we can be in this situation via *decision feedback*.

In both cases, the problem is easily solved. If $\mathbf{A}$ is known, then we set

$$\mathbf{W}^{\mathrm{H}} = \mathbf{A}^{\dagger}\,, \qquad \mathbf{S} = \mathbf{W}^{\mathrm{H}}\mathbf{X}\,.$$

Here, $\mathbf{A}^{\dagger}$ is the Moore-Penrose pseudo-inverse of $\mathbf{A}$. If $M \geq d$ and the columns of $\mathbf{A}$ are linearly independent, then $\mathbf{A}^{\dagger}$ is equal to the left inverse

$$\mathbf{A}^{\dagger} = (\mathbf{A}^{\mathrm{H}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{H}}\,.$$

Note that, indeed, under these assumptions $\mathbf{A}^{\mathrm{H}}\mathbf{A}$ is invertible and $\mathbf{A}^{\dagger}\mathbf{A} = \mathbf{I}$. If $M < d$ then we cannot recover the sources exactly: $\mathbf{A}^{\mathrm{H}}\mathbf{A}$ is not invertible (it is a $d \times d$ matrix with maximal rank M), so that $\mathbf{A}^{\dagger}\mathbf{A} \neq \mathbf{I}$.

If $\mathbf{S}$ is known, then we take

$$\mathbf{W}^{\mathrm{H}} = \mathbf{S}\mathbf{X}^{\dagger}\,, \qquad \mathbf{A} = (\mathbf{W}^{\mathrm{H}})^{\dagger}\,.$$

where $\mathbf{X}^{\dagger}$ is a right inverse of $\mathbf{X}$. If $N \geq d$ and the rows of $\mathbf{X}$ are linearly independent,[1] then

$$\mathbf{X}^{\dagger} = \mathbf{X}^{\mathrm{H}}(\mathbf{X}\mathbf{X}^{\mathrm{H}})^{-1}\,.$$

This is verified by $\mathbf{X}\mathbf{X}^{\dagger} = \mathbf{I}$. In both cases, we obtain a beamformer which exactly cancels all interference, i.e., $\mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I}$.

Noisy case In the presence of additive noise, we have $\mathbf{X} = \mathbf{A}\mathbf{S} + \mathbf{N}$. Two types of linear least-squares (LS) minimization problems can now be considered. The first is based on minimizing the model fitting error,

$$\min_{\mathbf{S}} \ \|\mathbf{X} - \mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2\,, \qquad \text{or} \qquad \min_{\mathbf{A}} \ \|\mathbf{X} - \mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2\,, \tag{3.5}$$

with $\mathbf{A}$ or $\mathbf{S}$ known, respectively. The second type of minimization problem is based on minimizing the output error,

$$\min_{\mathbf{W}} \ \|\mathbf{W}^{\mathrm{H}}\mathbf{X} - \mathbf{S}\|_{\mathrm{F}}^2\,, \tag{3.6}$$

also with $\mathbf{A}$ or $\mathbf{S}$ known, respectively. The minimization problems are straightforward to solve, and in the same way as before.

Deterministic model matching For (3.5) with $\mathbf{A}$ known we obtain

$$\hat{\mathbf{S}} = \arg\min_{\mathbf{S}} \ \|\mathbf{X} - \mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2 \qquad \Rightarrow \qquad \hat{\mathbf{S}} = \mathbf{A}^{\dagger}\mathbf{X}\,, \tag{3.7}$$

so that again $\mathbf{W}^{\mathrm{H}} = \mathbf{A}^{\dagger}$. This is known as the *Zero-Forcing solution*, because $\mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I}$: all interfering sources are canceled. It is clear that the ZF beamformer maximizes the Signal-to-Interference power Ratio (SIR) at the output (it is actually infinity). Note however that

$$\mathbf{W}^{\mathrm{H}}\mathbf{X} = \mathbf{S} + \mathbf{A}^{\dagger}\mathbf{N}\,.$$

The noise contribution at the output is $\mathbf{A}^{\dagger}\mathbf{N}$, and if $\mathbf{A}^{\dagger}$ is large, the output noise will be large. To get a better insight for this, introduce the "economy-size" singular value decomposition of $\mathbf{A}$,

$$\mathbf{A} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}$$

where we take $\mathbf{U}_A : m \times d$ with orthonormal columns, $\mathbf{\Sigma}_A : d \times d$ diagonal containing the nonzero singular values of $\mathbf{A}$, and $\mathbf{V}_A : d \times d$ unitary. Since

$$\mathbf{A} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}} \qquad \Rightarrow \qquad \mathbf{A}^{\dagger} = \mathbf{V}_A\mathbf{\Sigma}_A^{-1}\mathbf{U}_A^{\mathrm{H}}\,,$$

[1] In the present noiseless case, note that there are only d linearly independent rows in $\mathbf{S}$ and $\mathbf{X}$, so for linear independence of the rows of $\mathbf{X}$ we need $M = d$. With noise, $\mathbf{X}$ will have full row rank M.

$\mathbf{A}^\dagger$ is large if $\mathbf{\Sigma}_A^{-1}$ is large, i.e., if $\mathbf{A}$ is ill conditioned.

Similarly, for (3.5) with $\mathbf{S}$ known we obtain

$$\hat{\mathbf{A}} = \arg\min_{\mathbf{A}} \|\mathbf{X} - \mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2 \qquad \Rightarrow \qquad \hat{\mathbf{A}} = \mathbf{X}\mathbf{S}^\dagger = \mathbf{X}\mathbf{S}^{\mathrm{H}}(\mathbf{S}\mathbf{S}^{\mathrm{H}})^{-1}\,. \tag{3.8}$$

This does not specify the beamformer, but staying in the same context of minimizing $\|\mathbf{X}-\mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2$, it is natural to take again a Zero-Forcing beamformer so that $\mathbf{W}^{\mathrm{H}} = \hat{\mathbf{A}}^\dagger$. Asymptotically for zero mean noise independent of the sources, this gives $\hat{\mathbf{A}} \to \mathbf{A}$: we converge to the true $\mathbf{A}$-matrix.

Example 3.1. The ZF beamformer satisfies $\mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I}$. Let $\mathbf{w}_1$ be the first column of $\mathbf{W}$, it is the beamformer to receive the first signal. Then

$$\mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I} \qquad \Rightarrow \qquad \mathbf{w}_1^{\mathrm{H}}[\mathbf{a}_2\,, \cdots, \mathbf{a}_d] = [\mathbf{0}\,, \cdots, \mathbf{0}]$$

so that

$$\mathbf{w}_1 \perp \{\mathbf{a}_2\,,\ \cdots\,,\ \mathbf{a}_d\}\,.$$

Thus, $\mathbf{w}_1$ projects out all other sources, except source 1,

$$\begin{aligned} \mathbf{w}_1^{\mathrm{H}}\mathbf{x}(t) &= \textstyle\sum_{i=1}^d \mathbf{w}_1^{\mathrm{H}}\mathbf{a}_i s_i(t) + \mathbf{w}_1^{\mathrm{H}}\mathbf{n}(t) \\ &= s_1(t) + \mathbf{w}_1^{\mathrm{H}}\mathbf{n}(t)\,. \end{aligned}$$

The effect on the noise is not considered. In ill-conditioned cases ($\mathbf{A}$ is ill-conditioned so that its inverse $\mathbf{W}$ may have large entries), $\mathbf{w}_1$ might give a large amplification of the noise.

Deterministic output error minimization The second optimization problem (3.6) minimizes the difference of the output signals to $\mathbf{S}$. For known $\mathbf{S}$, we obtain

$$\mathbf{W}^{\mathrm{H}} = \arg\min_{\mathbf{W}} \|\ \mathbf{W}^{\mathrm{H}}\mathbf{X} - \mathbf{S}\ \|_{\mathrm{F}}^2 = \mathbf{S}\mathbf{X}^\dagger\,. \tag{3.9}$$

Note that $\mathbf{X}^\dagger = \mathbf{X}^{\mathrm{H}}(\mathbf{X}\mathbf{X}^{\mathrm{H}})^{-1}$, so that

$$\mathbf{W}^{\mathrm{H}} = \tfrac{1}{N}\mathbf{S}\mathbf{X}^{\mathrm{H}}(\tfrac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}})^{-1} \;=\; \hat{\mathbf{R}}_{xs}^{\mathrm{H}}\hat{\mathbf{R}}_x^{-1}\,, \qquad \mathbf{W} \;=\; \hat{\mathbf{R}}_x^{-1}\hat{\mathbf{R}}_{xs}\,.$$

$\hat{\mathbf{R}}_x := \frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}}$ is the sample data covariance matrix, and $\hat{\mathbf{R}}_{xs} := \frac{1}{N}(\mathbf{X}\mathbf{S}^{\mathrm{H}})$ is the sample correlation matrix between the sources and the received data.

With known $\mathbf{A}$, note that we cannot solve the minimization problem (3.6) since we can fit any $\mathbf{S}$. We have to put certain assumptions on $\mathbf{S}$ and $\mathbf{N}$, for example the fact that the rows of $\mathbf{S}$ and $\mathbf{N}$ are statistically independent from each other, and hence for large N

$$\tfrac{1}{N}\mathbf{S}\mathbf{S}^{\mathrm{H}} \to \mathbf{I}, \quad \tfrac{1}{N}\mathbf{N}\mathbf{N}^{\mathrm{H}} \to \sigma^2\mathbf{I}, \quad \tfrac{1}{N}\mathbf{S}\mathbf{N}^{\mathrm{H}} \to \mathbf{0},$$

(we assumed that the source powers are incorporated in $\mathbf{A}$), so that

$$\hat{\mathbf{R}}_x = \tfrac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}} = \tfrac{1}{N}\mathbf{A}\mathbf{S}\mathbf{S}^{\mathrm{H}}\mathbf{A}^{\mathrm{H}} + \tfrac{1}{N}\mathbf{N}\mathbf{N}^{\mathrm{H}} + \tfrac{1}{N}\mathbf{A}\mathbf{S}\mathbf{N}^{\mathrm{H}} + \tfrac{1}{N}\mathbf{N}\mathbf{S}^{\mathrm{H}}\mathbf{A}^{\mathrm{H}} \quad \rightarrow \quad \mathbf{R}_x = \mathbf{A}\mathbf{A}^{\mathrm{H}} + \sigma^2\mathbf{I}$$
$$\hat{\mathbf{R}}_{xs} = \tfrac{1}{N}\mathbf{X}\mathbf{S}^{\mathrm{H}} = \tfrac{1}{N}\mathbf{A}\mathbf{S}\mathbf{S}^{\mathrm{H}} + \tfrac{1}{N}\mathbf{N}\mathbf{S}^{\mathrm{H}} \quad \rightarrow \quad \mathbf{R}_{xs} = \mathbf{A}$$

Asymptotically,

$$\mathbf{W} \;=\; \mathbf{R}_x^{-1}\mathbf{R}_{xs} = (\mathbf{A}\mathbf{A}^{\mathrm{H}} + \sigma^2\mathbf{I})^{-1}\mathbf{A}\,,$$

where $\mathbf{R}_x = \mathrm{E}[\mathbf{x}\mathbf{x}^{\mathrm{H}}]$ is the true data covariance matrix and $\mathbf{R}_{xs} = \mathrm{E}[\mathbf{x}\mathbf{s}^{\mathrm{H}}]$ is the true correlation matrix between the sources and the received data.[2] This is known as the Linear Minimum Mean Square Error (LMMSE) or Wiener receiver. This beamformer maximizes the Signal-to-Interference-plus-Noise Ratio (SINR) at the output. Since it does not cancel all interference, $\mathbf{W}^{\mathrm{H}}\mathbf{A} \neq \mathbf{I}$, the output source estimates are not unbiased. However, it produces estimates of $\mathbf{S}$ with minimal deviation, which is often more relevant.

3.2 STOCHASTIC APPROACH TO MATCHED AND WIENER FILTERS

3.2.1 Performance criteria

Let us now define some performance criteria, based on elementary stochastic assumptions on the data. For the case of a single signal in noise,

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k\,, \qquad y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k = (\mathbf{w}^{\mathrm{H}}\mathbf{a})s_k + (\mathbf{w}^{\mathrm{H}}\mathbf{n}_k)\,.$$

We make the assumptions

$$\mathrm{E}[|s_k|^2] = 1\,, \qquad \mathrm{E}[s_k\mathbf{n}_k^{\mathrm{H}}] = 0\,, \qquad \mathrm{E}[\mathbf{n}_k\mathbf{n}_k^{\mathrm{H}}] = \mathbf{R}_n\,,$$

so that

$$\mathrm{E}[|y_k|^2] = (\mathbf{w}^{\mathrm{H}}\mathbf{a})(\mathbf{a}^{\mathrm{H}}\mathbf{w}) + \mathbf{w}^{\mathrm{H}}\mathbf{R}_n\mathbf{w}\,.$$

The Signal-to-Noise Ratio (SNR) at the output can then be defined as

$$\mathrm{SNR}_{out}(\mathbf{w}) = \frac{\mathrm{E}[|(\mathbf{w}^{\mathrm{H}}\mathbf{a})s_k|^2]}{\mathrm{E}[|\mathbf{w}^{\mathrm{H}}\mathbf{n}_k|^2]} = \frac{\mathbf{w}^{\mathrm{H}}\mathbf{a}\mathbf{a}^{\mathrm{H}}\mathbf{w}}{\mathbf{w}^{\mathrm{H}}\mathbf{R}_n\mathbf{w}}\,.$$

With d signals (signal 1 of interest, the others considered interferers), we can write

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k = \mathbf{a}_1 s_{1,k} + \mathbf{A}'\mathbf{s}'_k + \mathbf{n}_k\,, \qquad y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k = (\mathbf{w}^{\mathrm{H}}\mathbf{a}_1)s_{1,k} + \mathbf{w}^{\mathrm{H}}\mathbf{A}'\mathbf{s}'_k + (\mathbf{w}^{\mathrm{H}}\mathbf{n}_k)\,,$$

[2] We thus see that even if we adopt a deterministic framework, we cannot avoid to make certain stochastic assumptions on the data and noise.

where $\mathbf{A}'$ contains the columns of $\mathbf{A}$ except for the first one, and similarly for $\mathbf{s}'_k$. Now we can define two criteria: the Signal-to-Interference Ratio (SIR), and the Signal-to-Interference-plus-Noise Ratio (SINR):

$$\begin{aligned} \mathrm{SIR}_1(\mathbf{w}) &:= \frac{\mathbf{w}^{\mathrm{H}}(\mathbf{a}_1\mathbf{a}_1^{\mathrm{H}})\mathbf{w}}{\mathbf{w}^{\mathrm{H}}\mathbf{A}'\mathbf{A}'^{\mathrm{H}}\mathbf{w}} = \frac{\mathbf{w}^{\mathrm{H}}(\mathbf{a}_1\mathbf{a}_1^{\mathrm{H}})\mathbf{w}}{\mathbf{w}^{\mathrm{H}}(\mathbf{A}\mathbf{A}^{\mathrm{H}} - \mathbf{a}_1\mathbf{a}_1^{\mathrm{H}})\mathbf{w}} \\ \mathrm{SINR}_1(\mathbf{w}) &:= \frac{\mathbf{w}^{\mathrm{H}}(\mathbf{a}_1\mathbf{a}_1^{\mathrm{H}})\mathbf{w}}{\mathbf{w}^{\mathrm{H}}(\mathbf{A}'\mathbf{A}'^{\mathrm{H}} + \mathbf{R}_n)\mathbf{w}} = \frac{\mathbf{w}^{\mathrm{H}}(\mathbf{a}_1\mathbf{a}_1^{\mathrm{H}})\mathbf{w}}{\mathbf{w}^{\mathrm{H}}(\mathbf{A}\mathbf{A}^{\mathrm{H}} - \mathbf{a}_1\mathbf{a}_1^{\mathrm{H}} + \mathbf{R}_n)\mathbf{w}}\,. \end{aligned} \tag{3.10}$$

For the Zero-Forcing receiver, we have by definition (for known $\mathbf{A}$)

$$\mathbf{W}^{\mathrm{H}}\mathbf{A} = \mathbf{I} \qquad \Rightarrow \qquad \mathbf{w}_1^{\mathrm{H}}\mathbf{A} = [1,\ 0,\ \cdots,\ 0] \qquad \Rightarrow \qquad \mathbf{w}_1^{\mathrm{H}}\mathbf{a}_1 = 1\,, \quad \mathbf{w}_1^{\mathrm{H}}\mathbf{A}' = [0,\ \cdots,\ 0]\,,$$

and it follows that $\mathrm{SIR}_1(\mathbf{w}_1) = \infty$. When $\mathbf{W}$ is estimated from a known $\mathbf{S}$, we can only construct an approximation of the ZF receiver. This approximate ZF receiver still maximizes the SIR, but it is not infinity anymore.

Note that (3.10) defines only the performance with respect to the first signal. If we want to receive all signals, we need to define a performance vector, with entries for each signal,

$$\begin{aligned} \mathrm{SIR}(\mathbf{W}) &:= [\mathrm{SIR}_1(\mathbf{w}_1) \quad \cdots \quad \mathrm{SIR}_d(\mathbf{w}_d)] \\ \mathrm{SINR}(\mathbf{W}) &:= [\mathrm{SINR}_1(\mathbf{w}_1) \quad \cdots \quad \mathrm{SINR}_d(\mathbf{w}_d)]\,. \end{aligned}$$

In graphs, we would usually plot only the worst performance of each vector, or the average of each vector.

3.2.2 Stochastic derivations (white noise)

We now show how the same ZF and Wiener receivers can be derived when starting from a stochastic formulation.

Stochastic model matching Assume a model with d sources,

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k \quad (k = 1, \cdots, N) \qquad \Leftrightarrow \qquad \mathbf{X} = \mathbf{A}\mathbf{S} + \mathbf{N}\,.$$

Suppose that $\mathbf{A}$ and $\mathbf{S}$ are deterministic, and that the noise samples are independent and identically distributed in time (temporally white), and spatially white ($\mathbf{R}_n = \mathbf{I}$) and jointly complex Gaussian distributed, so that $\mathbf{n}_k$ has a probability density

$$\mathbf{n}_k \sim \mathcal{CN}(0, \sigma^2\mathbf{I}) \qquad \Leftrightarrow \qquad p(\mathbf{n}_k) = \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{\|\mathbf{n}_k\|^2}{\sigma^2}}\,.$$

Because of temporal independence, the probability distribution of N samples is the product of the individual probability distributions,

$$p(\mathbf{N}) = \prod_{k=1}^{N} \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{\|\mathbf{n}_k\|^2}{\sigma^2}}\,.$$

Since $\mathbf{n}_k = \mathbf{x}_k - \mathbf{A}\mathbf{s}_k$, the probability to receive a certain vector $\mathbf{x}_k$, given $\mathbf{A}$ and $\mathbf{s}_k$ is thus

$$p(\mathbf{x}_k|\mathbf{A},\mathbf{s}_k) = \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{\|\mathbf{x}_k - \mathbf{A}\mathbf{s}_k\|^2}{\sigma^2}}$$

and hence

$$p(\mathbf{X}|\mathbf{A},\mathbf{S}) = \prod_{k=1}^{N} \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{\|\mathbf{x}_k - \mathbf{A}\mathbf{s}_k\|^2}{\sigma^2}} = \left(\frac{1}{\sqrt{\pi}\sigma}\right)^N e^{-\frac{\sum_{k=1}^{N}\|\mathbf{x}_k - \mathbf{A}\mathbf{s}_k\|^2}{\sigma^2}} = \left(\frac{1}{\sqrt{\pi}\sigma}\right)^N e^{-\frac{\|\mathbf{X}-\mathbf{A}\mathbf{S}\|_F^2}{\sigma^2}}.$$

$p(\mathbf{X}|\mathbf{S})$ is called the *likelihood* of receiving a certain data matrix $\mathbf{X}$, for a certain mixing matrix $\mathbf{A}$ and a certain transmitted data matrix $\mathbf{S}$. It is of course a probability density function, but in the likelihood interpretation we regard it as a function of $\mathbf{A}$ and $\mathbf{S}$, for an actual received data matrix $\mathbf{X}$. The *Deterministic Maximum Likelihood* technique estimates $\mathbf{A}$ and/or $\mathbf{S}$ as that matrix that maximizes the likelihood of having received the actual received $\mathbf{X}$, thus

$$(\hat{\mathbf{A}},\hat{\mathbf{S}}) = \arg\max_{\mathbf{A},\mathbf{S}} \left(\frac{1}{\sqrt{\pi}\sigma}\right)^N e^{-\frac{\|\mathbf{X}-\mathbf{A}\mathbf{S}\|_F^2}{\sigma^2}}. \tag{3.11}$$

If we take the negative logarithm of $p(\mathbf{X}|\mathbf{A},\mathbf{S})$, we obtain what is called the negative log-likelihood function. Since it is a monotonously growing function, taking the logarithm does not change the location of the maximum. The maximization problem then becomes a minimization over const $+ \|\mathbf{X} - \mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2/\sigma^2$, or

$$(\hat{\mathbf{A}},\hat{\mathbf{S}}) = \arg\min_{\mathbf{A},\mathbf{S}} \|\mathbf{X} - \mathbf{A}\mathbf{S}\|_{\mathrm{F}}^2. \tag{3.12}$$

This is the same model fitting problem as we had before in (3.5). Thus, the deterministic ML problem is equivalent to the LS model fitting problem in the case of white (temporally and spatially) Gaussian noise.

Stochastic output error minimization In a statistical framework, the output error problem (3.6) becomes

$$\min_{\mathbf{w}_i} \mathrm{E}[|\mathbf{w}_i^{\mathrm{H}}\mathbf{x}_k - s_{i,k}|^2].$$

The cost function is known as the Linear Minimum Mean Square Error. Note that the full equalizer matrix $\mathbf{W}$ can then be obtained by stacking the solutions for the different sources: $\mathbf{W} = [\mathbf{w}_1 \ \cdots \ \mathbf{w}_d]$. The above cost function can be worked out as follows:

$$\begin{aligned} J(\mathbf{w}_i) &= \mathrm{E}[|\mathbf{w}_i^{\mathrm{H}}\mathbf{x}_k - s_{i,k}|^2] \\ &= \mathbf{w}_i^{\mathrm{H}}\mathrm{E}[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}]\mathbf{w}_i \ - \ \mathbf{w}_i^{\mathrm{H}}\mathrm{E}[\mathbf{x}_k\bar{s}_{i,k}] \ - \ \mathrm{E}[s_{i,k}\mathbf{x}_k^{\mathrm{H}}]\mathbf{w} \ + \ \mathrm{E}[|s_{i,k}|^2] \\ &= \mathbf{w}_i^{\mathrm{H}}\mathbf{R}_x\mathbf{w}_i \ - \ \mathbf{w}_i^{\mathrm{H}}\mathbf{r}_{xs,i} \ - \ \mathbf{r}_{xs,i}^{\mathrm{H}}\mathbf{w}_i + 1 \end{aligned}$$

Note that we have assumed that $\mathrm{E}[|s_{i,k}|^2] = 1$ and that $\mathbf{r}_{xs,i} = \mathrm{E}[\mathbf{x}_k\bar{s}_{i,k}]$ is the i-th column of $\mathbf{R}_{xs} = \mathrm{E}[\mathbf{x}_k\mathbf{s}^{\mathrm{H}}]$.

If $s_{i,k}$ is deterministic, then $J = J_k$ depends on $s_{i,k}$, and we need to work with an average over N samples, $\bar{J} = \frac{1}{N}\sum_{k=0}^{N-1} J_k$. For large N and i.i.d. assumptions on $s_{i,k}$, the result will be the same.

Now differentiate with respect to $\mathbf{w}_i$. This is a bit tricky since $\mathbf{w}_i$ is complex and functions of complex variables may not be differentiable (a simple example of a non-analytic function is $f(z) = \bar{z}$). There are various approaches (e.g. [1, 8]). A consistent approach is to regard $\mathbf{w}_i$ and $\bar{\mathbf{w}}_i$ as *independent* variables. Let $\mathbf{w}_i = \mathbf{u} + j\mathbf{v}$ with $\mathbf{u}$ and $\mathbf{v}$ real-valued, then the complex gradients to $\mathbf{w}_i$ and $\bar{\mathbf{w}}_i$ are defined as [8]

$$\nabla_{\mathbf{w}_i} J = \frac{1}{2}\left(\nabla_{\mathbf{u}} J - j\nabla_{\mathbf{v}} J\right) = \frac{1}{2}\begin{bmatrix} \frac{\partial}{\partial u_1} J \\ \vdots \\ \frac{\partial}{\partial u_d} J \end{bmatrix} - \frac{1}{2} j \begin{bmatrix} \frac{\partial}{\partial v_1} J \\ \vdots \\ \frac{\partial}{\partial v_d} J \end{bmatrix}$$

$$\nabla_{\bar{\mathbf{w}}_i} J = \frac{1}{2}\left(\nabla_{\mathbf{u}} J + j\nabla_{\mathbf{v}} J\right) = \frac{1}{2}\begin{bmatrix} \frac{\partial}{\partial u_1} J \\ \vdots \\ \frac{\partial}{\partial u_d} J \end{bmatrix} + \frac{1}{2} j \begin{bmatrix} \frac{\partial}{\partial v_1} J \\ \vdots \\ \frac{\partial}{\partial v_d} J \end{bmatrix}$$

with properties

$$\nabla_{\mathbf{w}_i} \mathbf{w}_i^{\mathrm{H}} \mathbf{r}_{xs,i} = \mathbf{0}\,, \quad \nabla_{\mathbf{w}_i} \mathbf{r}_{xs,i}^{\mathrm{H}} \mathbf{w}_i = \bar{\mathbf{r}}_{xs,i}\,, \quad \nabla_{\mathbf{w}_i} \mathbf{w}_i^{\mathrm{H}} \mathbf{R}_x \mathbf{w}_i = \mathbf{R}_x^T \bar{\mathbf{w}}_i$$

$$\nabla_{\bar{\mathbf{w}}_i} \mathbf{w}_i^{\mathrm{H}} \mathbf{r}_{xs,i} = \mathbf{r}_{xs,i}\,, \quad \nabla_{\bar{\mathbf{w}}_i} \mathbf{r}_{xs,i}^{\mathrm{H}} \mathbf{w}_i = \mathbf{0}\,, \quad \nabla_{\bar{\mathbf{w}}_i} \mathbf{w}_i^{\mathrm{H}} \mathbf{R}_x \mathbf{w}_i = \mathbf{R}_x \mathbf{w}_i.$$

It can further be shown that for a stationary point, it is necessary and sufficient that either $\nabla_{\mathbf{w}} J = \mathbf{0}$ or that $\nabla_{\bar{\mathbf{w}}} J = \mathbf{0}$: the two are equivalent. Since the latter expression is more simple, and because it specifies the maximal rate of change, we keep from now on the definition for the gradient

$$\nabla J(\mathbf{w}) \equiv \nabla_{\bar{\mathbf{w}}} J(\mathbf{w})\,, \tag{3.13}$$

and we obtain

$$\nabla J(\mathbf{w}_i) = \mathbf{R}_x \mathbf{w}_i - \mathbf{r}_{xs,i}\,.$$

The minimum of $J(\mathbf{w}_i)$ is attained for

$$\nabla_{\mathbf{w}_i} J = \mathbf{0} \qquad \Rightarrow \qquad \mathbf{w}_i = \mathbf{R}_x^{-1} \mathbf{r}_{xs,i}\,,$$

or if all sources have to be detected

$$\mathbf{W} = \mathbf{R}_x^{-1}[\mathbf{r}_{xs,1} \ \cdots \ \mathbf{r}_{xs,d}] = \mathbf{R}_x^{-1} \mathbf{R}_{xs}.$$

We thus obtain the Wiener receiver.

The LMMSE cost function is also called Minimum Variance. This is in fact a misnomer: the expression is not really that of a variance because the error $\mathrm{E}[\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k] \neq 0$. In fact, for the

Wiener receiver, a single signal in noise, and s_k considered deterministic ($\mathrm{E}[s_k] = s_k$),

$$\begin{aligned}\mathrm{E}[y_k] &= \mathrm{E}[\mathbf{w}^{\mathrm{H}}\mathbf{x}_k] \\ &= \mathrm{E}[\mathbf{r}_{xs}^{\mathrm{H}}\mathbf{R}_x^{-1}(\mathbf{a}s_k + \mathbf{n}_k)] \\ &= \mathbf{a}^{\mathrm{H}}(\mathbf{a}\mathbf{a}^{\mathrm{H}} + \sigma^2\mathbf{I})^{-1}\mathbf{a}s_k \\ &= \mathbf{a}^{\mathrm{H}}\mathbf{a}(\mathbf{a}^{\mathrm{H}}\mathbf{a} + \sigma^2)^{-1}s_k \\ &= \tfrac{\mathbf{a}^{\mathrm{H}}\mathbf{a}}{\mathbf{a}^{\mathrm{H}}\mathbf{a}+\sigma^2}s_k\,.\end{aligned}$$

Thus, the expected value of the output is not s_k, but a scaled-down version of it.

3.2.3 Colored noise

Let us now see what changes in the above when the noise is not white, but has a variance

$$\mathrm{E}[\mathbf{n}_k\mathbf{n}_k^{\mathrm{H}}] = \mathbf{R}_n\,.$$

We assume that we know the variance. In that case, we can *prewhiten* the data with a square-root factor $\mathbf{R}_n^{-1/2}$:

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k \qquad \Rightarrow \qquad \begin{array}{ccccccc} \underbrace{\mathbf{R}_n^{-1/2}\mathbf{x}_k} & = & \underbrace{\mathbf{R}_n^{-1/2}\mathbf{A}}\,\mathbf{s}_k & + & \underbrace{\mathbf{R}_n^{-1/2}\mathbf{n}_k} \\ \underline{\mathbf{x}}_k & = & \underline{\mathbf{A}}\mathbf{s}_k & + & \underline{\mathbf{n}}_k \end{array}$$

Note that now

$$\underline{\mathbf{R}}_n = \mathrm{E}[\underline{\mathbf{n}}_k\underline{\mathbf{n}}_k{}^{\mathrm{H}}] = \mathbf{R}_n^{-1/2}\mathbf{R}_n\mathbf{R}_n^{-1/2} = \mathbf{I}$$

so that the noise $\underline{\mathbf{n}}_k$ is white. At this point, we are back on familiar grounds. The ZF equalizer becomes

$$\begin{aligned}\mathbf{s}_k = \underline{\mathbf{A}}^{\dagger H}\underline{\mathbf{x}}_k = (\underline{\mathbf{A}}^{\mathrm{H}}\mathbf{A})^{-1}\underline{\mathbf{A}}^{\mathrm{H}}\underline{\mathbf{x}}_k = (\mathbf{A}^{\mathrm{H}}\mathbf{R}_n^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{H}}\mathbf{R}_n^{-1}\mathbf{x}_k \\ \Rightarrow \qquad \mathbf{W} = \mathbf{R}_n^{-1}\mathbf{A}(\mathbf{A}^{\mathrm{H}}\mathbf{R}_n^{-1}\mathbf{A})^{-1}. \end{aligned} \tag{3.14}$$

The Wiener receiver on the other hand will be the same, since $\mathbf{R}_n$ is not used at all in the derivation. This can also be checked:

$$\underline{\mathbf{W}} = \underline{\mathbf{R}}_x^{-1}\underline{\mathbf{R}}_{xs} = (\mathbf{R}_n^{-1/2}\mathbf{R}_x\mathbf{R}_n^{-1/2})^{-1}\mathbf{R}_n^{-1/2}\mathbf{R}_{xs} = \mathbf{R}_n^{1/2}\mathbf{R}_x^{-1}\mathbf{R}_{xs}$$

$$\Rightarrow \quad \mathbf{W} = \mathbf{R}_n^{-1/2}\underline{\mathbf{W}} = \mathbf{R}_x^{-1}\mathbf{R}_{xs}.$$

3.3 OTHER INTERPRETATIONS OF MATCHED FILTERING

3.3.1 Maximum Ratio Combining

Consider a special case of the previous, a single signal in white noise,

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k\,, \qquad \mathrm{E}[\mathbf{n}_k\mathbf{n}_k^{\mathrm{H}}] = \sigma^2\mathbf{I}\,.$$

As we showed before, the ZF beamformer is given by

$$\mathbf{w} = \mathbf{a}(\mathbf{a}^{\mathrm{H}}\mathbf{a})^{-1} = \gamma_1 \mathbf{a}$$

where γ_1 is a scalar. Since a scalar multiplication does not change the output SNR, the optimal beamformer for s in this case is given by

$$\mathbf{w}_{MF} = \mathbf{a}$$

which is known as a *matched filter* or a *classical beamformer*. It is also known as *Maximum Ratio Combining* (MRC).

With non-white noise,

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k\,, \qquad \mathrm{E}[\mathbf{n}\mathbf{n}^{\mathrm{H}}] = \mathbf{R}_n$$

we have seen in (3.14) that

$$\mathbf{w} = \mathbf{R}_n^{-1}\mathbf{a}(\mathbf{a}^{\mathrm{H}}\mathbf{R}_n^{-1}\mathbf{a})^{-1} = \gamma_2 \mathbf{R}_n^{-1}\mathbf{a}\,.$$

Thus, the matched filter in non-white noise is

$$\mathbf{w}_{MF} = \mathbf{R}_n^{-1}\mathbf{a}\,.$$

We can proceed similarly with the Wiener receiver. In white noise,

$$\begin{aligned} \mathbf{w} &= \mathbf{R}_x^{-1}\mathbf{r}_{xs} \\ &= (\mathbf{a}\mathbf{a}^{\mathrm{H}} + \sigma^2\mathbf{I})^{-1}\mathbf{a} \\ &= \mathbf{a}(\mathbf{a}^{\mathrm{H}}\mathbf{a} + \sigma^2)^{-1} \\ &\sim \mathbf{a}\,. \end{aligned}$$

It is equal to a multiple of the matched filter. In colored noise, we obtain similarly:

$$\begin{aligned} \mathbf{w} &= \mathbf{R}_x^{-1}\mathbf{r}_{xs} \\ &= (\mathbf{a}\mathbf{a}^{\mathrm{H}} + \mathbf{R}_n)^{-1}\mathbf{a} \\ &= \mathbf{R}_n^{-1}\mathbf{a}(\mathbf{a}^{\mathrm{H}}\mathbf{R}_n^{-1}\mathbf{a} + 1)^{-1} \\ &\sim \mathbf{R}_n^{-1}\mathbf{a}\,. \end{aligned}$$

This is equal to a multiple of the matched filter for colored noise.

The colored noise case is relevant also for the following reason: with more than one signal, we can write the model as

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k = \mathbf{a}_1 s_{1,k} + (\mathbf{A}'\mathbf{s}'_k + \mathbf{n}_k)\,.$$

This is of the form

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k\,, \qquad \mathbf{R}_n = \mathbf{A}'\mathbf{A}'^H + \sigma^2\mathbf{I}$$

where the noise is now not white, but colored due to the contribution of the interfering sources.

The conclusion is quite interesting:

For the reception of a single source out of interfering sources plus noise, the ZF receiver, matched filter or MRC: $\mathbf{w} = \mathbf{R}_n^{-1}\mathbf{a}$, and the Wiener receiver: $\mathbf{w} = \mathbf{R}_x^{-1}\mathbf{r}_{xs}$, are all equal to a scalar multiple of each other, and hence will asymptotically give the same performance.

The above are examples of *non-joint receivers*: the interference is lumped together with the noise, and there might as well be many more interferers than antennas. Intuitively, one could think that performing a *joint* estimation of all the sources would yield an improved performance. However, since the Wiener filter optimally balances interference suppression and noise suppression, it does not make a difference whether we design our Wiener filter by lumping the interference together with the noise, or not. Hence, the non-joint and joint Wiener filter lead to the same performance. This is not true for the ZF receiver.

Example 3.2. Consider a single source in white noise:

$$\mathbf{x}(t) = \mathbf{a}s(t) + \mathbf{n}(t)\,, \qquad \mathbf{R}_n = \sigma^2\mathbf{I}\,.$$

Suppose the signal is normalized to have unit-power $\mathrm{E}[|s(t)|^2] = 1$. Then

$$\mathrm{SNR}_{in} = \frac{1}{\sigma^2}\,.$$

This is the SNR at each element of the array.

If we choose the matched filter, or MRC, i.e., $\mathbf{w} = \mathbf{a}$, then

$$y(t) = \mathbf{w}^H\mathbf{x}(t) = \mathbf{a}^{\mathrm{H}}\mathbf{a}s(t) + \mathbf{a}^{\mathrm{H}}\mathbf{n}(t) = \|\mathbf{a}\|^2 s(t) + \mathbf{a}^{\mathrm{H}}\mathbf{n}(t)$$

then

$$\mathrm{SNR}_{out} = \frac{\|\mathbf{a}\|^4}{\mathbf{a}^{\mathrm{H}}\sigma^2\mathbf{I}\mathbf{a}} = \frac{\|\mathbf{a}\|^4}{\|\mathbf{a}\|^2\sigma^2} = \|\mathbf{a}\|^2 \cdot \mathrm{SNR}_{in}\,.$$

The factor $\|\mathbf{a}\|^2$ is the array gain.

3.3.2 Maximizing the output SNR

For a single signal in noise, the matched filter $\mathbf{w} = \mathbf{R}_n^{-1}\mathbf{a}$ maximizes the output SNR. This is derived as follows. Similarly as in the preceding example, we have

$$\mathbf{x}(t) = \mathbf{a}s(t) + \mathbf{n}(t)\,.$$

Assume again that $\mathrm{E}[|s(t)|^2] = 1$, then $\mathbf{R}_x = \mathbf{R}_a + \mathbf{R}_n$, with

$$\mathbf{R}_a = \mathbf{a}\mathbf{a}^{\mathrm{H}}\,, \qquad \mathbf{R}_n = \mathrm{E}[\mathbf{n}_k\mathbf{n}_k^{\mathrm{H}}]\,.$$

The output SNR after beamforming is equal to

$$\mathrm{SNR}_{out}(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{H}}\mathbf{R}_a\mathbf{w}}{\mathbf{w}^{\mathrm{H}}\mathbf{R}_n\mathbf{w}}\,.$$

We now would like to find the beamformer that maximizes SIR_{out}, i.e.,

$$\mathbf{w} = \arg\max_{\mathbf{w}} \frac{\mathbf{w}^{\mathrm{H}}\mathbf{R}_a\mathbf{w}}{\mathbf{w}^{\mathrm{H}}\mathbf{R}_n\mathbf{w}}\,.$$

The expression is known as a Rayleigh quotient, and the solution is known to be given by the solution of the eigenvalue equation

$$\mathbf{R}_n^{-1}\mathbf{R}_a\mathbf{w} = \lambda_{\max}\mathbf{w}\,. \tag{3.15}$$

This can be seen as follows: suppose that $\mathbf{R}_n = \mathbf{I}$, then the equation is

$$\max_{\mathbf{w}}\ \mathbf{w}^{\mathrm{H}}\mathbf{R}_a\mathbf{w}\,.$$

Introduce an eigenvalue decomposition for $\mathbf{R}_a = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}}$, then

$$\max_{\mathbf{w}} (\mathbf{w}^{\mathrm{H}}\mathbf{U})\mathbf{\Lambda}(\mathbf{U}^{\mathrm{H}}\mathbf{w})\,.$$

Let λ_1 be the largest eigenvalue ($(1,1)$-entry of $\mathbf{\Lambda}$), then it is clear that the maximum of the expression is given by choosing $\mathbf{w}^{\mathrm{H}}\mathbf{U} = [1 \quad 0 \cdots 0]$. Thus, the optimal $\mathbf{w}$ is the eigenvector corresponding to the largest eigenvalue, and satisfies the eigenvalue equation $\mathbf{R}_a\mathbf{w} = \lambda_1\mathbf{w}$. If $\mathbf{R}_n \neq \mathbf{I}$, then we can first whiten the noise to obtain the result in (3.15).

The solution of (3.15) can be found in closed form, by inserting $\mathbf{R}_a = \mathbf{a}\mathbf{a}^{\mathrm{H}}$. We obtain

$$\begin{aligned}
& & \mathbf{R}_n^{-1}\mathbf{R}_a\mathbf{w} &= \lambda_{\max}\mathbf{w} \\
\Leftrightarrow\ & & \mathbf{R}_n^{-1}\mathbf{a}\mathbf{a}^{\mathrm{H}}\mathbf{w} &= \lambda_{\max}\mathbf{w} \\
\Leftrightarrow\ & (\mathbf{R}_n^{-1/2}\mathbf{a})(\mathbf{a}^{\mathrm{H}}\mathbf{R}_n^{-1/2})(\mathbf{R}_n^{1/2}\mathbf{w}) &= \lambda_{\max}(\mathbf{R}_n^{1/2}\mathbf{w}) \\
\Leftrightarrow\ & \underline{\mathbf{a}}\underline{\mathbf{a}}^{\mathrm{H}}\underline{\mathbf{w}} &= \lambda_{\max}\underline{\mathbf{w}} \\
\Leftrightarrow\ & \underline{\mathbf{w}} &= \underline{\mathbf{a}}\,, \quad \lambda_{\max} = \underline{\mathbf{a}}^{\mathrm{H}}\underline{\mathbf{a}}
\end{aligned}$$

and it follows that

$$\mathbf{w} = \mathbf{R}_n^{-1}\mathbf{a}$$

which is, as promised, the matched filter in colored noise.

3.3.3 LCMV – MVDR – GSC – Capon

A related technique for beamforming is the so-called *Linearly constrained Minimum Variance* (LCMV), also known as *Minimum Variance Distortionless Response* (MVDR), *Generalized Sidelobe Canceling* (GSC), and *Capon beamforming* (in the French literature). In this technique, it is again assumed that we have a single source in colored noise (this might contain other interferers as well),

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k\,.$$

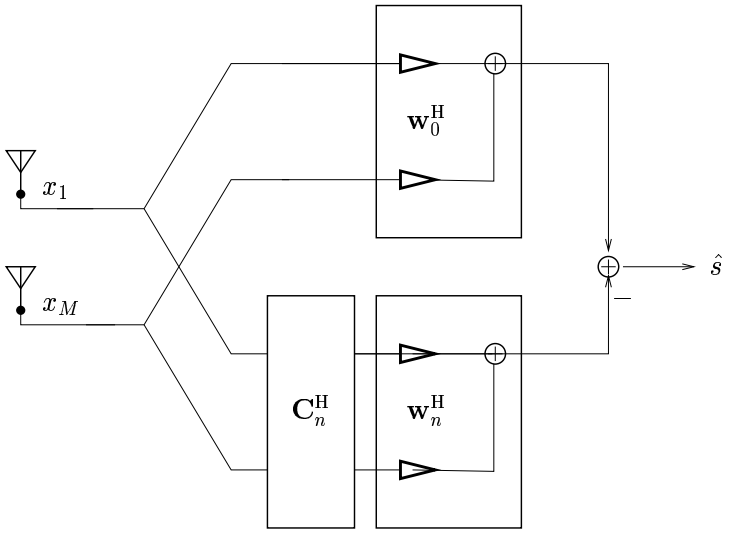


Figure 3.1. The Generalized Sidelobe Canceler

If $\mathbf{a}$ is known, then the idea is that we constrain the beamformer $\mathbf{w}$ to

$$\mathbf{w}^{\mathrm{H}}\mathbf{a} = 1$$

i.e., we have a fixed response towards the source. The remaining freedom is used to minimize the total output power ("response" or "variance") after beamforming:

$$\min_{\mathbf{w}} \ \mathbf{w}^{\mathrm{H}}\mathbf{R}_x\mathbf{w} \qquad \text{such that} \quad \mathbf{w}^{\mathrm{H}}\mathbf{a} = 1\,.$$

The solution can be found in closed form using Lagrange multipliers and is given by

$$\mathbf{w} = \mathbf{R}_x^{-1}\mathbf{a}(\mathbf{a}^{\mathrm{H}}\mathbf{R}_x^{-1}\mathbf{a})^{-1}\,.$$

Thus, $\mathbf{w}$ is a scalar multiple of the Wiener receiver.

This case may be generalized by introducing a constraint matrix $\mathbf{C}:\ M \times L$ $(M > L)$ and an L-dimensional vector $\mathbf{f}$, and asking for $\mathbf{C}^{\mathrm{H}}\mathbf{w} = \mathbf{f}$. The solution to

$$\min_{\mathbf{w}} \ \mathbf{w}^{\mathrm{H}}\mathbf{R}_x\mathbf{w} \qquad \text{such that} \quad \mathbf{C}^{\mathrm{H}}\mathbf{w} = \mathbf{f}$$

is given by

$$\mathbf{w} = \mathbf{R}_x^{-1}\mathbf{C}(\mathbf{C}^{\mathrm{H}}\mathbf{R}_x^{-1}\mathbf{C})^{-1}\mathbf{f}\,.$$

Generalized Sidelobe Canceler The generalized sidelobe canceler (GSC) represents an alternative formulation of the LCMV problem, which provides insight, is useful for analysis, and can

simplify LCMV beamformer implementation. Essentially, it is a technique to convert a constrained minimization problem into an unconstrained form. Suppose we decompose the weight vector $\mathbf{w}$ into two orthogonal components, $\mathbf{w}_0$ and $-\mathbf{v}$ ($\mathbf{w} = \mathbf{w}_0 - \mathbf{v}$), that lie in the range and null space of $\mathbf{C}$ and $\mathbf{C}^{\mathrm{H}}$, respectively. These subspaces span the entire space so this decomposition can be used to represent any $\mathbf{w}$. Since $\mathbf{C}^{\mathrm{H}}\mathbf{v} = \mathbf{0}$, we must have

$$\mathbf{w}_0 = \mathbf{C}(\mathbf{C}^{\mathrm{H}}\mathbf{C})^{-1}\mathbf{f} \tag{3.16}$$

if $\mathbf{w}$ is to satisfy the constraints. (3.16) is the minimum norm solution to the under-determined system $\mathbf{C}^{\mathrm{H}}\mathbf{w}_0 = \mathbf{f}$. The vector $\mathbf{v}$ is a linear combination of the columns of an $M \times (M - L)$ matrix $\mathbf{C}_n$, and $\mathbf{v} = \mathbf{C}_n\mathbf{w}_n$; provided the columns of $\mathbf{C}_n$ form a basis for the null space of $\mathbf{C}^{\mathrm{H}}$. The matrix $\mathbf{C}_n$ can be obtained from $\mathbf{C}$ using any of several orthogonalization procedures, for example the QR factorization or the SVD. The structure of the beamformer using the weight vector $\mathbf{w} = \mathbf{w}_0 - \mathbf{C}_n\mathbf{w}_n$ is depicted in figure 3.1. The choice for $\mathbf{w}_0$ and $\mathbf{C}_n$ implies that $\mathbf{w}$ satisfies the constraints independent of $\mathbf{w}_n$ and reduces the LCMV problem to the unconstrained problem

$$\min_{\mathbf{w}_n} \left[\mathbf{w}_0 - \mathbf{C}_n\mathbf{w}_n\right]^{\mathrm{H}}\mathbf{R}_x[\mathbf{w}_0 - \mathbf{C}_n\mathbf{w}_n] \,.$$

The solution is

$$\mathbf{w}_n = (\mathbf{C}_n^{\mathrm{H}}\mathbf{R}_x\mathbf{C}_n)^{-1}\mathbf{C}_n^{\mathrm{H}}\mathbf{R}_x\mathbf{w}_0 \,.$$

The primary advantage of this implementation stems from the fact that the weights $\mathbf{w}_n$ are unconstrained and a data independent beamformer $\mathbf{w}_0$ is implemented as an integral part of the adaptive beamformer. The unconstrained nature of the adaptive weights permits much simpler adaptive algorithms to be employed and the data independent beamformer is useful in situations where adaptive signal cancellation occurs.

Example 3.3. *Reference channels – Multiple sidelobe canceler*

A special case of the LCMV is that where there is a primary channel $x_0(t)$, receiving a signal of interest plus interferers and noise, and a collection of reference antennas $\mathbf{x}(t)$, receiving only interference and noise. For example, in hands-free telephony in a car, we may have a microphone close to the speaker, and other microphones further away from the speaker and closer to the engine and other noise sources. Or we may have a directional antenna (parabolic dish) and an array of omnidirectional antennas. The objective is to subtract from the primary channel a linear combination of the reference antennas such that the output power is minimized. If indeed the signal of interest is not present on the reference antennas, the SINR of this signal will be improved. (If the signal *is* present at the reference antennas, then of course it will be canceled as well!)

Call the primary sensor signal x_0 and the reference signal vector $\mathbf{x}$. Then the objective is

$$\min_{\mathbf{w}} \mathrm{E}\|x_0 - \mathbf{w}^{\mathrm{H}}\mathbf{x}\|^2 \,.$$

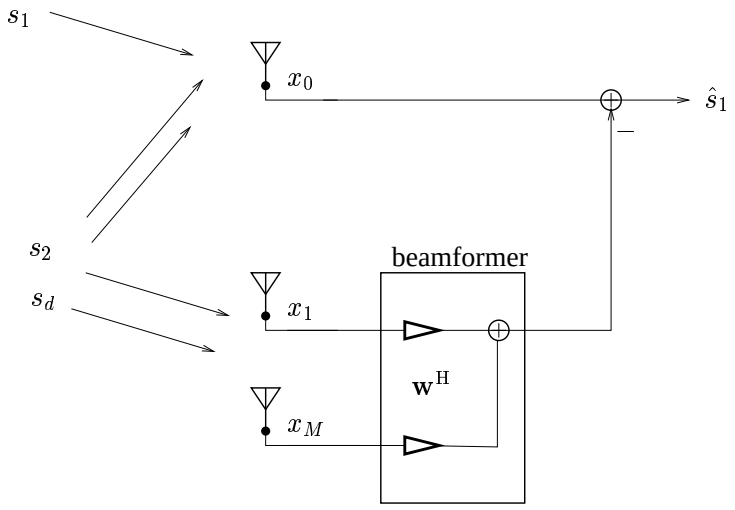


Figure 3.2. The Multiple Sidelobe Canceller: interference is estimated from a reference antenna array and subtracted from the primary antenna x_0.

The solution of this problem is given by

$$\mathbf{w} = \mathbf{R}_x^{-1}\mathbf{r}, \quad \mathbf{R}_x = \mathrm{E}[\mathbf{x}\mathbf{x}^{\mathrm{H}}], \quad \mathbf{r} := \mathrm{E}[\mathbf{x}\bar{x}_0]\,.$$

This technique is called the Multiple Sidelobe Canceler (Applebaum 1976). It is a special case of the LCMV beamformer, which becomes clear if we construct a joint data vector

$$\mathbf{x}' = \begin{bmatrix} x_0 \\ \mathbf{x} \end{bmatrix}, \qquad \mathbf{w}' = \begin{bmatrix} 1 \\ -\mathbf{w} \end{bmatrix}, \qquad \mathbf{c} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}$$

The constraint is $(\mathbf{w}')^{\mathrm{H}}\mathbf{c} = 1$.

3.4 PREWHITENING FILTER STRUCTURE

Subspace-based prefiltering In the noise-free case with less sources than sensors, $\mathbf{X} = \mathbf{AS}$ is rank deficient: its rank is d (the number of signals) rather than m (the number of sensors). As a consequence, once we have found a beamformer $\mathbf{w}$ such that $\mathbf{w}^{\mathrm{H}}\mathbf{X} = \mathbf{s}$, one of the source signals, then we can add any vector $\mathbf{w}_0$ such that $\mathbf{w}_0^{\mathrm{H}}\mathbf{X} = \mathbf{0}$ to $\mathbf{w}$, and obtain the same output. The beamforming solutions are not unique.

The desired beamforming solutions are all in the column span of $\mathbf{A}$. Indeed, any component orthogonal to this span will not contribute at the output. The most easy way to ensure that our solutions will be in this span is by performing a *dimension-reducing* prefiltering. Let $\mathbf{F}$ be any

$M \times d$ matrix such that $\text{span}(\mathbf{F}) = \text{span}(\mathbf{A})$. Then all beamforming matrices $\mathbf{W}$ in the column span of $\mathbf{A}$ are given by

$$\mathbf{W} = \mathbf{F}\underline{\mathbf{W}}$$

where $\underline{\mathbf{W}}$ is a $d \times d$ matrix, nonsingular if the beamformers are linearly independent. We will use the underscore to denote prefiltered variables. Thus, the prefiltered noisy data matrix is

$$\underline{\mathbf{X}} := \mathbf{F}^{\mathrm{H}}\mathbf{X}$$

with structure

$$\underline{\mathbf{X}} = \underline{\mathbf{A}}\mathbf{S} + \underline{\mathbf{N}}\,, \qquad \text{where} \quad \underline{\mathbf{A}} := \mathbf{F}^{\mathrm{H}}\mathbf{A}\,, \;\; \underline{\mathbf{N}} := \mathbf{F}^{\mathrm{H}}\mathbf{N}\,.$$

$\underline{\mathbf{X}}$ has only d channels, and is such that $\mathbf{W}^{\mathrm{H}}\mathbf{X} = \underline{\mathbf{W}}^{\mathrm{H}}\underline{\mathbf{X}}$. Thus, the columns of $\underline{\mathbf{W}}$ are d-dimensional beamformers on the prefiltered data $\underline{\mathbf{X}}$, and for any choice of $\underline{\mathbf{W}}$ the columns of the effective beamformer $\mathbf{W}$ are all in the column span of $\mathbf{A}$, as desired.

To describe the column span of $\mathbf{A}$, introduce the "economy-size" singular value decomposition of $\mathbf{A}$,

$$\mathbf{A} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}$$

where we take $\mathbf{U}_A : m \times d$ with orthonormal columns, $\mathbf{\Sigma}_A : d \times d$ diagonal containing the nonzero singular values of $\mathbf{A}$, and $\mathbf{V}_A : d \times d$ unitary. Also let $\mathbf{U}_A^{\perp}$ be the orthonormal complement of $\mathbf{U}_A$. The columns of $\mathbf{U}_A$ are an orthonormal basis of the column span of $\mathbf{A}$. The point is that even if $\mathbf{A}$ is unknown, $\mathbf{U}_A$ can be estimated from the data, as described below (and in more detail in section 3.5).

We assume that the noise is spatially white, with covariance matrix $\sigma^2\mathbf{I}$. Let $\hat{\mathbf{R}}_x = \frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}}$ be the noisy sample data covariance matrix, with eigenvalue decomposition

$$\hat{\mathbf{R}}_x \;=\; \hat{\mathbf{U}}\hat{\mathbf{\Lambda}}\hat{\mathbf{U}}^{\mathrm{H}} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}^2\hat{\mathbf{U}}^{\mathrm{H}}\,. \tag{3.17}$$

Here, $\hat{\mathbf{U}}$ is $M \times M$ unitary, and $\hat{\mathbf{\Sigma}}$ is $M \times M$ diagonal. Equivalently, these factors follow from an SVD of the data matrix $\mathbf{X}$ directly:

$$\tfrac{1}{\sqrt{N}}\mathbf{X} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^{\mathrm{H}}$$

We collect the d largest singular values into a $d \times d$ diagonal matrix $\hat{\mathbf{\Sigma}}_s$, and collect the corresponding d eigenvectors into $\hat{\mathbf{U}}_s$. Asymptotically, $\mathbf{R}_x$ satisfies $\mathbf{R}_x = \mathbf{A}\mathbf{A}^{\mathrm{H}} + \sigma^2\mathbf{I}$, with eigenvalue decomposition

$$\mathbf{R}_x \;=\; \mathbf{U}_A\mathbf{\Sigma}_A^2\mathbf{U}_A^{\mathrm{H}} + \sigma^2\mathbf{I} \;=\; \mathbf{U}_A(\mathbf{\Sigma}_A^2 + \sigma^2 I)\mathbf{U}_A^{\mathrm{H}} \;+\; \sigma^2\mathbf{U}_A^{\perp}\mathbf{U}_A^{\perp H}\,. \tag{3.18}$$

Since $\hat{\mathbf{R}}_x \to \mathbf{R}_x$ as the number of samples N grows, we have that $\hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_s^2\hat{\mathbf{U}}_s^{\mathrm{H}} \to \mathbf{U}_A(\mathbf{\Sigma}_A^2 + \sigma^2\mathbf{I})\mathbf{U}_A^{\mathrm{H}}$, so that $\hat{\mathbf{U}}_s$ is an asymptotically unbiased estimate of $\mathbf{U}_A$. Thus $\mathbf{U}_A$ and also $\mathbf{\Sigma}$ and $\mathbf{\Lambda}$ can be estimated consistently from the data, by taking sufficiently many samples. In contrast, $\mathbf{V}_A$ cannot be estimated like this: this factor is on the "inside" of the factorization $\mathbf{A}\mathbf{S} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}\mathbf{S}$ and as long as $\mathbf{S}$ is unknown, any unitary factor can be exchanged between $\mathbf{V}_A$ and $\mathbf{S}$.

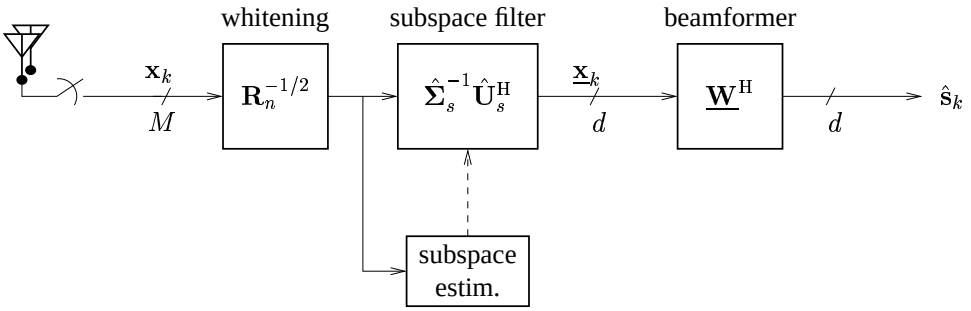


Figure 3.3. Beamforming prefiltering structure

Even if we choose $\mathbf{F}$ to have the column span of $\hat{\mathbf{U}}_s$, there is freedom left. As we will show, a natural choice is to combine the dimension reduction with a whitening of the data covariance matrix, i.e., such that $\underline{\mathbf{R}}_x := \frac{1}{N}\underline{\mathbf{X}}\underline{\mathbf{X}}^{\mathrm{H}}$ becomes unity: $\underline{\mathbf{R}}_x = \mathbf{I}$. This is achieved if we define $\mathbf{F}$ as

$$\mathbf{F} = \hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_s^{-1}\,. \tag{3.19}$$

Without dimension reduction, $\mathbf{F} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}^{-1}$ is a square root factor[3] of $\hat{\mathbf{R}}_x^{-1}$, i.e., $\hat{\mathbf{R}}_x^{-1} = \mathbf{F}\mathbf{F}^{\mathrm{H}}$.

After this preprocessing, the Wiener filter is simply given by

$$\underline{\mathbf{W}} = \underline{\mathbf{A}}$$

at least asymptotically. Indeed,

$$\mathbf{W} = \mathbf{F}\underline{\mathbf{W}} = \mathbf{F}\mathbf{F}^{\mathrm{H}}\mathbf{A}$$

and asymptotically $\mathbf{F}\mathbf{F}^{\mathrm{H}} = (\mathbf{A}^{\mathrm{H}}\mathbf{A} + \sigma^2\mathbf{I})^{-1}\mathbf{P}_A$. Since $\mathbf{P}_A\mathbf{A} = \mathbf{A}$, the result follows. For finite samples, the dimension reduction gives a slight difference.

If the noise is colored with covariance matrix $\sigma^2\mathbf{R}_n$, where we know $\mathbf{R}_n$ but perhaps not the noise power σ^2, then we first whiten the noise by computing $\mathbf{R}_n^{-1/2}\mathbf{X}$, and continue as in the white noise case. The structure of the resulting beamformer is shown in figure 3.3.

Direct matched filtering Another choice for $\mathbf{F}$ that reduces dimensions and that is often taken if (an estimate of) $\mathbf{A}$ is known is by simply setting

$$\mathbf{F} = \mathbf{A}$$

The output after this filter becomes

$$\underline{\mathbf{X}} = \mathbf{A}^{\mathrm{H}}\mathbf{X} = (\mathbf{A}^{\mathrm{H}}\mathbf{A})\mathbf{S} + \mathbf{A}^{\mathrm{H}}\mathbf{N}$$

[3]Square root factors are usually taken symmetric, i.e., $\hat{\mathbf{R}}_x^{1/2}\hat{\mathbf{R}}_x^{1/2} = \hat{\mathbf{R}}_x$ and $\hat{\mathbf{R}}_x^{1/2\,H} = \hat{\mathbf{R}}_x^{1/2}$, but this is not necessary. $\mathbf{F}$ is a non-symmetric factor.

The noise is now non-white, it has covariance $\mathbf{A}^{\mathrm{H}}\mathbf{A}$.

We can whiten it by multiplying by a factor $(\mathbf{A}^{\mathrm{H}}\mathbf{A})^{-1/2}$. It is more convenient to introduce an SVD $\mathbf{A} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}$, and use a non-symmetrical factor $\mathbf{\Sigma}_A^{-1}\mathbf{V}_A^{\mathrm{H}}$. Note that $\mathbf{\Sigma}_A^{-1}\mathbf{V}_A^{\mathrm{H}}\mathbf{A}^{\mathrm{H}} = \mathbf{U}_A^{\mathrm{H}}$. This gives

$$\underline{\mathbf{X}} = \mathbf{U}_A^{\mathrm{H}}\mathbf{X} = (\mathbf{U}_A^{\mathrm{H}}\mathbf{A})\mathbf{S} + \mathbf{U}_A^{\mathrm{H}}\mathbf{N} = (\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}})\mathbf{S} + \mathbf{U}_A^{\mathrm{H}}\mathbf{N}\,.$$

The noise is white again, and $\underline{\mathbf{A}} = \mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}$. If we subsequently want to apply a Wiener receiver in this prefiltered domain, it is given by

$$\underline{\mathbf{W}} = (\underline{\mathbf{A}}\underline{\mathbf{A}}^{\mathrm{H}} + \sigma^2\mathbf{I})^{-1}\underline{\mathbf{A}} = (\mathbf{\Sigma}_A^2 + \sigma^2\mathbf{I})^{-1}\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}$$

Conclusion

We can do the following forms of prefiltering:

- $\mathbf{F} = \mathbf{A}$. After this the noise is nonwhite.
- $\mathbf{F} = \mathbf{A}(\mathbf{A}^{\mathrm{H}}\mathbf{A})^{-1/2} = \mathbf{U}_A = \mathbf{U}_s$. After this the noise is white, the Wiener receiver is obtained by setting $\underline{\mathbf{W}} = (\underline{\mathbf{A}}\underline{\mathbf{A}}^{\mathrm{H}} + \sigma^2\mathbf{I})^{-1}\underline{\mathbf{A}}$.
- $\mathbf{F} = \hat{\mathbf{\Sigma}}_s^{-1}\hat{\mathbf{U}}_s$. The noise becomes nonwhite, but the data is whitened, $\underline{\hat{\mathbf{R}}}_x = \mathbf{I}$. The Wiener receiver is obtained by $\underline{\mathbf{W}} = \underline{\mathbf{A}}$.

3.5 EIGENVALUE ANALYSIS OF $\mathbf{R}_X$

So far, we have looked at the receiver problem from a rather restricted viewpoint: the beamformers were based on the situation where there is a single source in noise. In the next section we will also consider beamforming algorithms that can handle more sources. These are based on an eigenvalue analysis of the data covariance matrix, which is introduced in this section.

Let us first consider the covariance matrix due to d sources and no noise,

$$\mathbf{R}_x = \mathbf{A}\mathbf{R}_s\mathbf{A}^{\mathrm{H}}$$

where $\mathbf{R}_x$ has size $M \times M$, $\mathbf{A}$ has size $M \times d$ and $\mathbf{R}_s$ has size $d \times d$. If $d < M$, then the rank of $\mathbf{R}_x$ is d since $\mathbf{A}$ has only d columns. Thus, we can estimate the number of narrow-band sources from a rank analysis. This is also seen from an eigenvalue analysis: let

$$\mathbf{R}_x = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}}$$

be an eigenvalue decomposition of $\mathbf{R}_x$, where the $M \times M$ matrix $\mathbf{U}$ is unitary ($\mathbf{U}\mathbf{U}^{\mathrm{H}} = \mathbf{I}$, $\mathbf{U}^{\mathrm{H}}\mathbf{U} = \mathbf{I}$) and contains the eigenvectors, and the $M \times M$ diagonal matrix $\mathbf{\Lambda}$ contains the corresponding eigenvalues in non-increasing order ($\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M \geq 0$). Since the rank is d, there are only d nonzero eigenvalues. We can collect these in a $d \times d$ diagonal matrix $\mathbf{\Lambda}_s$, and the corresponding eigenvectors in a $M \times d$ matrix $\mathbf{U}_s$, so that

$$\mathbf{R}_x = \mathbf{U}_s\mathbf{\Lambda}_s\mathbf{U}_s^{\mathrm{H}}\,. \tag{3.20}$$

The remaining $M-d$ eigenvectors from $\mathbf{U}$ can be collected in a matrix $\mathbf{U}_n$, and they are orthogonal to $\mathbf{U}_s$ since $\mathbf{U} = [\mathbf{U}_s \;\; \mathbf{U}_n]$ is unitary. The subspace spanned by the columns of $\mathbf{U}_s$ is called the *signal subspace*, the orthogonal complement spanned by the columns of $\mathbf{U}_n$ is known as the *noise subspace* (although this is a misnomer since here there is no noise yet and later the noise will be everywhere and not confined to the subspace). Thus, in the noise-free case,

$$\mathbf{R}_x = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}} = [\mathbf{U}_s \quad \mathbf{U}_n] \left[\begin{array}{c|c} \mathbf{\Lambda}_s & 0 \\ \hline 0 & 0 \end{array}\right] \left[\begin{array}{c} \mathbf{U}_s^{\mathrm{H}} \\ \mathbf{U}_n^{\mathrm{H}} \end{array}\right]$$

In the presence of white noise,

$$\mathbf{R}_x \;=\; \mathbf{A}_s\mathbf{R}_s\mathbf{A}_s^{\mathrm{H}} + \sigma^2\mathbf{I}_M \,.$$

In this case, $\mathbf{R}_x$ is full rank: its rank is always M. However, we can still detect the number of sources by looking at the eigenvalues of $\mathbf{R}_x$. Indeed, the eigenvalue decomposition is derived as (expressed in terms of the previous decomposition (3.20) and using the fact that $\mathbf{U} = [\mathbf{U}_s \quad \mathbf{U}_n]$ is unitary: $\mathbf{U}_s\mathbf{U}_s^{\mathrm{H}} + \mathbf{U}_n\mathbf{U}_n^{\mathrm{H}} = \mathbf{I}_M$)

$$\begin{aligned} \mathbf{R}_x &= \mathbf{A}_s\mathbf{R}_s\mathbf{A}_s^{\mathrm{H}} + \sigma^2\mathbf{I}_M \\ &= \mathbf{U}_s\mathbf{\Lambda}_s\mathbf{U}_s^{\mathrm{H}} + \sigma^2(\mathbf{U}_s\mathbf{U}_s^{\mathrm{H}} + \mathbf{U}_n\mathbf{U}_n^{\mathrm{H}}) \\ &= \mathbf{U}_s(\mathbf{\Lambda}_s + \sigma^2\mathbf{I}_q)\mathbf{U}_s^{\mathrm{H}} + \mathbf{U}_n(\sigma^2\mathbf{I}_{M-d})\mathbf{U}_n^{\mathrm{H}} \\ &= [\mathbf{U}_s \quad \mathbf{U}_n] \left[\begin{array}{c|c} \mathbf{\Lambda}_s + \sigma^2\mathbf{I}_q & 0 \\ \hline 0 & \sigma^2\mathbf{I}_{M-d} \end{array}\right] \left[\begin{array}{c} \mathbf{U}_s^{\mathrm{H}} \\ \mathbf{U}_n^{\mathrm{H}} \end{array}\right] \\ &=: \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}} \end{aligned} \tag{3.21}$$

hence $\mathbf{R}_x$ has $M-d$ eigenvalues equal to σ^2, and d that are larger than σ^2. *Thus, we can detect the number of signals d by comparing the eigenvalues of $\mathbf{R}_x$ to a threshold defined by σ^2.*

A physical interpretation of the eigenvalue decomposition can be as follows. The eigenvectors give an orthogonal set of "directions" (spatial signatures) present in the covariance matrix, sorted in decreasing order of dominance. The eigenvalues give the power of the signal coming from the corresponding directions, or the power of the output of a beamformer matched to that direction. Indeed, let the i'th eigenvector be $\mathbf{u}_i$, then this output power will be

$$\mathbf{u}_i^{\mathrm{H}}\mathbf{R}\mathbf{u}_i = \lambda_i \,.$$

The first eigenvector, $\mathbf{u}_1$, is always pointing in the direction from which most energy is coming. The second one, $\mathbf{u}_2$, points in a direction orthogonal to $\mathbf{u}_1$ from which most of the remaining energy is coming, etcetera.

If only (spatially white) noise is present but no sources, then there is no dominant direction, and all eigenvalues are equal to the noise power. If there is a single source with unit power and spatial signature $\mathbf{a}$, then the covariance matrix is $\mathbf{R}_x = \mathbf{a}\mathbf{a}^{\mathrm{H}} + \sigma^2\mathbf{I}$. It follows from the previous that there is only one eigenvalue larger than σ^2. The corresponding eigenvector is $\mathbf{u}_1 = \mathbf{a}\frac{1}{\|\mathbf{a}\|}$, and is in the direction of $\mathbf{a}$. The power coming from that direction is

$$\lambda_1 = \mathbf{u}_1^{\mathrm{H}}\mathbf{R}_x\mathbf{u}_1 = \|\mathbf{a}\|^2 + \sigma^2 \,.$$

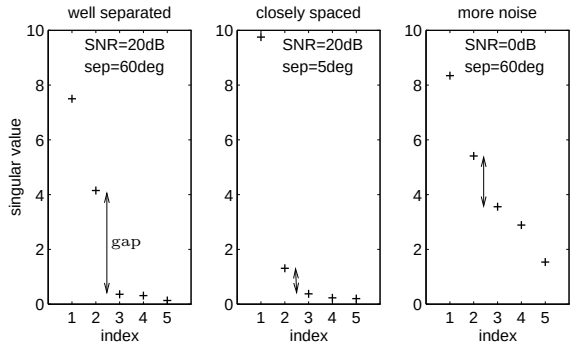


Figure 3.4. Behavior of singular values.

Since there is only one source, the power coming from any other direction orthogonal to $\mathbf{u}_1$ is σ^2, the noise power. Since $\mathbf{u}_1 = \mathbf{a}\frac{1}{\|\mathbf{a}\|}$,

$$\frac{\mathbf{a}^{\mathrm{H}}\mathbf{R}_x\mathbf{a}}{\mathbf{a}^{\mathrm{H}}\mathbf{a}} = \frac{\mathbf{u}_1^{\mathrm{H}}\mathbf{R}_x\mathbf{u}_1}{\mathbf{u}_1^{\mathrm{H}}\mathbf{u}_1} = \lambda_1 \,.$$

Thus, the result of using the largest eigenvector as a beamformer is the same as the output power of a matched filter where the $\mathbf{a}$-vector of the source is known.

With more than one source, this generalizes. Suppose there are two sources with unit powers, and spatial signatures $\mathbf{a}_1$ and $\mathbf{a}_2$. If the spatial signatures are orthogonal, $\mathbf{a}_1^{\mathrm{H}}\mathbf{a}_2 = 0$, then $\mathbf{u}_1$ will be in the direction of the strongest source, number 1 say, and λ_1 will be the corresponding power, $\lambda_1 = \|\mathbf{a}_1\|^2 + \sigma^2$. Similarly, $\lambda_2 = \|\mathbf{a}_2\|^2 + \sigma^2$.

In general, the spatial signatures are not orthogonal to each other. In that case, $\mathbf{u}_1$ will point into the direction that is common to both $\mathbf{a}_1$ and $\mathbf{a}_2$, and $\mathbf{u}_2$ will point in the remaining direction orthogonal to $\mathbf{u}_1$. The power λ_1 coming from direction $\mathbf{u}_1$ will be larger than before because it combines power from both sources, whereas λ_2 will be smaller.

Example 3.4. Instead of the eigenvalue decomposition of $\hat{\mathbf{R}}_x$, we may also compute the singular value decomposition of $\mathbf{X}$:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}}$$

Here, $\mathbf{U} : M \times M$ and $\mathbf{V} : N \times N$ are unitary, and $\mathbf{\Sigma} : M \times N$ is diagonal. Since

$$\mathbf{R}_x = \tfrac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}} = \tfrac{1}{N}\mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^{\mathrm{H}}$$

it is seen that $\mathbf{U}$ contains the eigenvectors of $\hat{\mathbf{R}}_x$, whereas $\frac{1}{N}\mathbf{\Sigma}^2 = \mathbf{\Lambda}$ are the eigenvalues. Thus, the two decompositions give the same information (numerically, it is often better to compute the SVD).

Figure 3.4 shows singular values of $\mathbf{A}$ for $d = 2$ sources, a uniform linear array with $M = 5$ antennas, and $N = 10$ samples, for

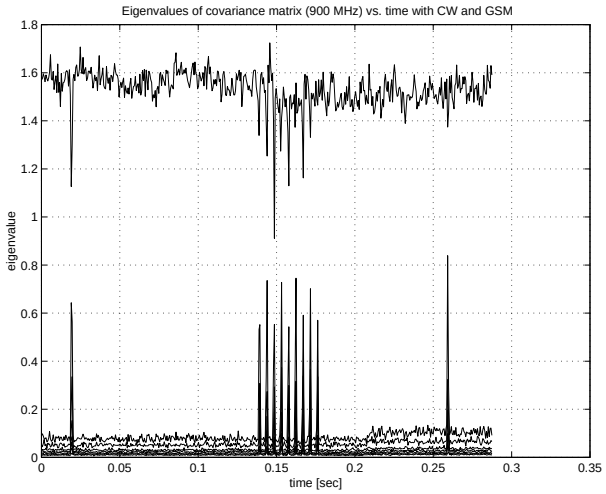


Figure 3.5. Eigenstructure as a function of time

1. well separated angles: large gap between signal and noise singular values,
2. signals from close directions, resulting in a small signal singular value,
3. increased noise level, increasing noise singular values.

Example 3.5. The covariance matrix eigenvalue structure can be nicely illustrated on data collected at the Westerbork telescope array. We selected a narrow band slice (52 kHz) of a GSM uplink data file, around 900 MHz. In this subband we have two sources: a continuous narrow band (sine wave) signal which leaked in from a local oscillator, and a weak GSM signal. From this data we computed a sequence of short term data covariance matrices $\hat{\mathbf{R}}_x^{0.5ms}$ based on 0.5 ms averages. Figure 3.5 shows the time evolution of the eigenvalues of these matrices. The largest eigenvalue is due to the CW signal and is always present. The GSM source is intermittent: at time intervals where it is present the number of large eigenvalues increases to two. The remaining eigenvalues are at the noise floor, σ^2. The small step in the noise floor after 0.2 s is due to a periodically switched calibration noise source at the input of the telescope front ends.

3.6 BEAMFORMING AND DIRECTION ESTIMATION

In the previous sections, we have assumed that the source matrix $\mathbf{S}$ or the array matrix $\mathbf{A}$ is known. We can now generalize the situation and only assume that the array response is known as a function of the direction parameter θ. Then the directions of arrival (DOA's) of the signals are estimated and used to generate the beamformer weights. The beamformers are in fact the

same as we derived in the previous section, except that we specify them in terms of $\mathbf{a}(\theta)$ and subsequently scan θ to find directions where there is "maximal response" (e.g., in the sense of maximal output SNR).

3.6.1 The classical beamformer

The weights in a data independent beamformer are designed so the beamformer response approximates a desired response independent of the array data or data statistics. The design objective—approximating a desired response—is the same as that for classical FIR filter design.

In spatial filtering one is often interested in receiving a signal arriving from a known location point θ_0. Assuming the signal is narrowband, a common choice for the beamformer weight vector is the array response vector $\mathbf{a}(\theta_0)$. This is called the *classical beamformer*, or the Bartlett beamformer; it is precisely the same as the matched filter assuming spatially white noise.

In direction finding using classical beamforming, we estimate the directions of the sources as those that maximize the output power of the beamformer when pointing in a scanning direction θ (and normalizing the output by the array gain):

$$\hat{\theta} = \max_{\theta} \frac{\mathbf{a}(\theta)^{\mathrm{H}} \mathbf{R}_x \mathbf{a}(\theta)}{\mathbf{a}(\theta)^{\mathrm{H}} \mathbf{a}(\theta)} \,.$$

The expression is a *spatial spectrum* estimator. An example of the spectrum obtained this way is shown in figure 3.6, see also chapter 1. With only N samples available, we replace $\mathbf{R}_x$ by the sample covariance matrix, $\hat{\mathbf{R}}_x$. For multiple signals we choose the d largest local maxima.

This technique is equivalent to maximizing the output SNR in case there is only 1 signal in white noise. If the noise is colored, the denominator should actually be replaced by $\mathbf{a}(\theta)^{\mathrm{H}} \mathbf{R}_n \mathbf{a}(\theta)$. If the noise is white but there are interfering sources, our strategy before was to lump the interferers with the noise. However, in the present situation we do not know the interfering directions or $\mathbf{a}(\theta_2), \cdots, \mathbf{a}(\theta_d)$, so this is impossible. This shows that with multiple sources, the classical beamforming technique gives a bias to the direction estimate.

3.6.2 The MVDR

As discussed before, in the MVDR technique we try to minimize the output power, while constraining the power towards the direction θ:

$$\hat{\theta} = \max_{\theta} \{ \min_{\mathbf{w}} \; \mathbf{w}^{\mathrm{H}} \hat{\mathbf{R}}_x \mathbf{w} \qquad \text{subject to} \quad \mathbf{w}^{\mathrm{H}} \mathbf{a}(\theta) = 1 \} \,.$$

This yields

$$\mathbf{w} = \frac{\hat{\mathbf{R}}_x^{-1} \mathbf{a}(\theta)}{\mathbf{a}(\theta)^{\mathrm{H}} \hat{\mathbf{R}}_x^{-1} \mathbf{a}(\theta)}$$

and thus the direction estimate is

$$\hat{\theta} = \max_{\theta} \; \frac{1}{\mathbf{a}(\theta)^{\mathrm{H}} \hat{\mathbf{R}}_x^{-1} \mathbf{a}(\theta)} \,.$$

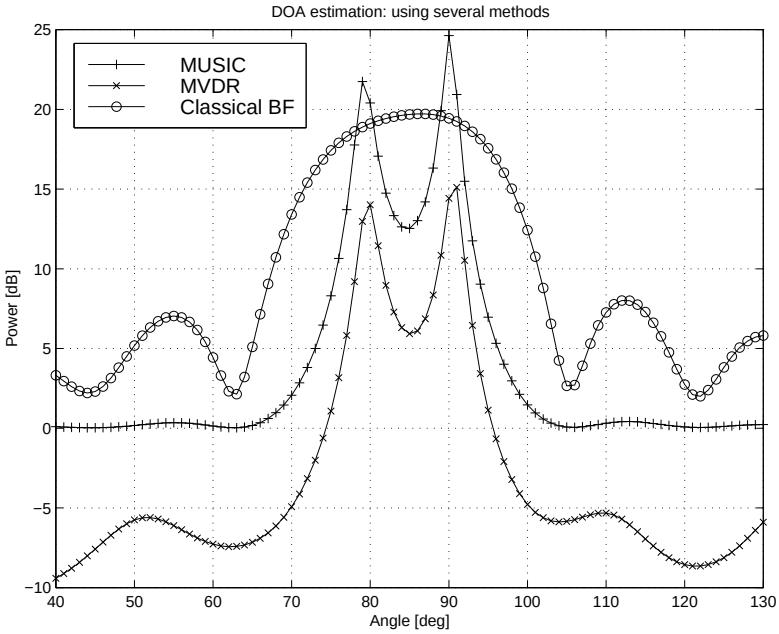


Figure 3.6. Spatial spectra corresponding to the classical beamformer, MVDR, and MUSIC. The DOA's are estimated as the maxima of the spectra.

For multiple signals choose again the d largest local maxima. The MVDR is also illustrated in figure 3.6.

3.6.3 The MUSIC algorithm

The classical beamformer and the MVDR have a poor performance in cases where there are several closely spaced sources. We now consider more advanced techniques based on the eigenvalue decomposition of the covariance matrix, viz. equation (3.21),

$$\begin{aligned} \mathbf{R}_x &= \mathbf{A}_s \mathbf{R}_s \mathbf{A}_s^{\mathrm{H}} + \sigma^2 \mathbf{I}_M \\ &= \mathbf{U}_s (\mathbf{\Lambda}_s + \sigma^2 \mathbf{I}_q) \mathbf{U}_s^{\mathrm{H}} + \mathbf{U}_n (\sigma^2 \mathbf{I}_{M-d}) \mathbf{U}_n^{\mathrm{H}} \end{aligned}$$

As discussed before, the eigenvalues give information on the number of sources (by counting how many eigenvalues are larger than σ^2). However, the decomposition shows more than just the number of sources. Indeed, *the columns of* $\mathbf{U}_s$ *span the same subspace as the columns of* $\mathbf{A}$. This is clear in the noise-free case (3.20), but the decomposition (3.21) shows that the eigenvectors contained in $\mathbf{U}_s$ and $\mathbf{U}_n$ respectively are the same as in the noise-free case. Thus,

$$\mathrm{span}(\mathbf{U}_s) = \mathrm{span}(\mathbf{A})\,, \qquad \mathbf{U}_n^{\mathrm{H}} \mathbf{A} = 0\,. \tag{3.22}$$

Given a correlation matrix $\hat{\mathbf{R}}_x$ estimated from the data, we compute its eigenvalue decomposition. From this we can detect the rank d from the number of eigenvalues larger than σ^2, and

we can estimate $\mathbf{U}_s$ and hence the subspace spanned by the columns of $\mathbf{A}$. Although we cannot directly identify each individual column of $\mathbf{A}$, its subspace estimate can nonetheless be used to determine the directions, since we know that

$$\mathbf{A} = [\mathbf{a}(\theta_1)\,,\ \cdots\,,\ \mathbf{a}(\theta_d)]$$

If $\mathbf{a}(\theta)$ is known as a function of θ, then we can select the unknown parameters $[\theta_1\,,\ \cdots\,,\ \theta_d]$ to make the estimate of $\mathbf{A}$ *fit* the subspace $\mathbf{U}_s$. Several algorithms are based on this idea. Below we discuss an effective algorithm that is widely used, the MUSIC (Multiple SIgnal Classification) algorithm.

Note that it is crucial that the noise is spatially white. For colored noise, an extension (whitening) is possible but we have to know the coloring.

Assume that $d < M$. Since $\mathrm{col}(\mathbf{U}_s) = \mathrm{col}\{\mathbf{a}(\theta_1), \cdots, \mathbf{a}(\theta_d)\}$, we have

$$\mathbf{U}_n^{\mathrm{H}}\,\mathbf{a}(\theta_i)\ =\ 0\,, \qquad (1 \le i \le d) \tag{3.23}$$

The MUSIC algorithm estimates the directions of arrival by choosing the d lowest local minima of the cost function

$$J_{MUSIC}(\theta) = \frac{\|\hat{\mathbf{U}}_n^{\mathrm{H}}\mathbf{a}(\theta)\|^2}{\|\mathbf{a}(\theta)\|^2} = \frac{\mathbf{a}(\theta)^{\mathrm{H}}\hat{\mathbf{U}}_n\hat{\mathbf{U}}_n^{\mathrm{H}}\mathbf{a}(\theta)}{\mathbf{a}(\theta)^{\mathrm{H}}\mathbf{a}(\theta)} \tag{3.24}$$

where $\hat{\mathbf{U}}_n^{\mathrm{H}}$ is the sample estimate of the noise subspace, obtained from an eigenvalue decomposition of $\hat{\mathbf{R}}_x$. To obtain a 'spectral-like' graph as before (it is called a pseudo-spectrum), we plot the inverse of $J_{MUSIC}(\theta)$. See figure 3.6. Note that this eigenvalue technique gives a higher resolution than the original classical spectrum, also because its sidelobes are much more flat.

Note, very importantly, that as long as the number of sources is smaller than the number of sensors ($d < M$), the eigenvalue decomposition of the true $\mathbf{R}_x$ allows to estimate *exactly* the DOAs. This means that if the number of samples N is large enough, we can obtain estimates with arbitrary precision. Thus, in contrast to the beamforming techniques, the MUSIC algorithm provides *statistically consistent* estimates.

An important limitation is still the failure to resolve closely spaced signals in small samples and at low SNR scenarios. This loss of resolution is more pronounced for highly correlated signals. In the limiting case of coherent signals, the property (3.23) is violated because the rank of $\mathbf{R}_x$ becomes smaller than the number of sources (the dimension of $\mathbf{U}_n$ is too large), and the method fails to yield consistent estimates. To remedy this problem, techniques such as "spatial smoothing" as well as extensions of the MUSIC algorithm have been derived.

3.7 APPLICATIONS TO TEMPORAL MATCHED FILTERING

In the previous sections, we have looked at matched filtering in the context of array signal processing. Let us now look at how this applies to temporal filtering.

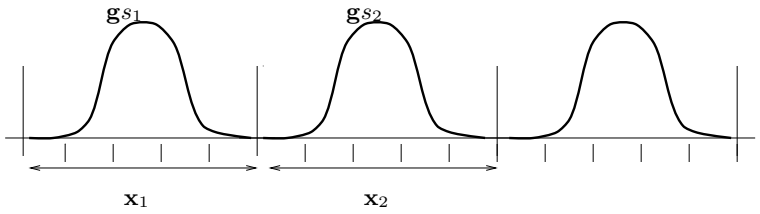


Figure 3.7. No intersymbol interference

No intersymbol interference We start with a fairly simple case, namely the reception of a symbol sequence $s_\delta(t)$ convolved with a pulse shape function $g(t)$:

$$x(t) = g(t) * s_\delta(t)$$

The symbol sequence is modeled as a sequence of delta pulses, $s_\delta(t) = \sum s_k \delta(t-k)$. Note that the symbol period is normalized to 1. We will first assume that the pulse shape function has a duration of less than 1, so that $g(t)$ has support only on the interval $[0, 1)$. We sample $x(t)$ at a rate P, where P is the (integer) *oversampling rate*. The samples of $x(t)$ are stacked in vectors

$$\mathbf{x}_k = \begin{bmatrix} x(k) \\ x(k+\frac{1}{P}) \\ \vdots \\ x(k+\frac{P-1}{P}) \end{bmatrix}$$

See also figure 3.7. If we are sufficiently synchronized, we obtain

$$\mathbf{x}_k = \mathbf{g}s_k \quad \Leftrightarrow \quad \begin{bmatrix} x(k) \\ x(k+\frac{1}{P}) \\ \vdots \\ x(k+\frac{P-1}{P}) \end{bmatrix} = \begin{bmatrix} g(0) \\ g(\frac{1}{P}) \\ \vdots \\ g(\frac{P-1}{P}) \end{bmatrix} s_k \tag{3.25}$$

or

$$\mathbf{X} = \mathbf{g}\mathbf{s}\,, \qquad \mathbf{X} = [\mathbf{x}_0 \quad \mathbf{x}_1 \quad \cdots \quad \mathbf{x}_{N-1}]\,, \quad \mathbf{s} = [s_0 \quad s_1 \quad \cdots \quad s_{N-1}]\,.$$

The matched filter in this context is simply $\mathbf{g}^{\mathrm{H}}$. It has a standard interpretation as a convolution or *integrate-and-dump filter*. Indeed, $y_k = \mathbf{g}^{\mathrm{H}}\mathbf{x}_k = \sum_{i=0}^{P-1} g(i)x(k+\frac{i}{P})$. This can be viewed as a convolution by the reverse filter $g_r(t) := g(1-t)$:

$$y_k \;=\; \mathbf{g}^{\mathrm{H}}\mathbf{x}_k \;=\; \sum_{i=1}^{P} g_r(\tfrac{i}{P})x(k+1-\tfrac{i}{P})$$

If P is very large, the summation becomes an integral

$$y_k = \int_0^1 g(t)x(k+t)\,dt = \int_0^1 g_r(t)x(k+1-t)\,dt\,.$$

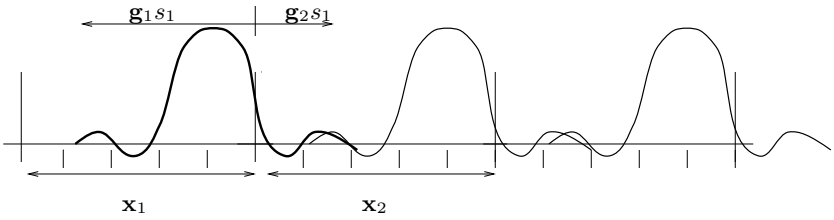


Figure 3.8. With intersymbol interference

With intersymbol interference In practise, pulse shape functions are often a bit larger than 1 symbol period. Also, we might not be able to achieve perfect synchronization. Thus let us define a shift of $\mathbf{g}$ over some delay τ and assume for simplicity that the result has support on $[0, 2T)$ (although with pulse shapes longer than a symbol period, it would in fact be more correct to have a support of $[0, 3T)$):

$$\mathbf{g}(\tau) := \begin{bmatrix} g(0-\tau) \\ g(\frac{1}{P}-\tau) \\ \vdots \\ g(2-\frac{1}{P}-\tau) \end{bmatrix}.$$

Now, $\mathbf{g}(\tau)$ is spread over two symbol periods, and we can define

$$\mathbf{g}(\tau) = \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix}.$$

After convolution of $g(t-\tau)$ by the symbol sequence $s_\delta(t)$, sampling at rate P, and stacking, we obtain that the resuling sample vectors $\mathbf{x}_k$ are the sum of two symbol sequences (see figure 3.8):

$$\mathbf{x}_k = \mathbf{g}_1 s_k + \mathbf{g}_2 s_{k-1} \qquad \Leftrightarrow \qquad \begin{bmatrix} x(k) \\ x(k+\frac{1}{P}) \\ \vdots \\ x(k+\frac{P-1}{P}) \end{bmatrix} = \begin{bmatrix} g(0-\tau) \\ g(\frac{1}{P}-\tau) \\ \vdots \\ g(1-\frac{1}{P}-\tau) \end{bmatrix} s_k + \begin{bmatrix} g(1-\tau) \\ g(\frac{1}{P}-\tau) \\ \vdots \\ g(2-\frac{1}{P}-\tau) \end{bmatrix} s_{k-1}$$

or in matrix form

$$\mathbf{X} = \mathbf{G}_\tau \mathbf{S} \qquad \Leftrightarrow \qquad [\mathbf{x}_0 \quad \mathbf{x}_1 \quad \cdots \quad \mathbf{x}_{N-1}] \;=\; [\mathbf{g}_1 \quad \mathbf{g}_2] \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ s_{-1} & s_0 & \cdots & s_{N-2} \end{bmatrix}.$$

In this case, there is *intersymbol interference*: a sample vector $\mathbf{x}_k$ contains the contributions of more than a single symbol.

A matched filter in this context would be $\mathbf{G}_\tau^{\mathrm{H}}$, at least if $\mathbf{G}_\tau$ is tall: $P \geq 2$. In the current situation (impulse response length including fractional delay shorter than 2 symbols) this is the case as soon as we do any amount of oversampling. After matched filtering, the output $\mathbf{y}_k$ has

two entries, each containing a mixture of the symbol sequence and one shift of this sequence. The mixture is given by

$$\mathbf{G}_\tau^{\mathrm{H}}\mathbf{G}_\tau = \begin{bmatrix} \mathbf{g}_1^{\mathrm{H}}\mathbf{g}_1 & \mathbf{g}_1^{\mathrm{H}}\mathbf{g}_2 \\ \mathbf{g}_2^{\mathrm{H}}\mathbf{g}_1 & \mathbf{g}_2^{\mathrm{H}}\mathbf{g}_2 \end{bmatrix}$$

Thus, if $\mathbf{g}_1$ is not orthogonal to $\mathbf{g}_2$, the two sequences will be mixed and further equalization ('beamformer' on $\mathbf{y}_k$) will be necessary. The matched filter in this case only serves to make the output more compact (2 entries) in case P is large.

More in general, we can stack the sample vectors to obtain

$$\mathcal{X} = \mathcal{G}_\tau \mathcal{S} \qquad \Leftrightarrow \qquad \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{N-1} \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{g}_1 & \mathbf{g}_2 \\ \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{0} \end{bmatrix} \begin{bmatrix} s_1 & s_2 & \cdots & s_N \\ s_0 & s_1 & \cdots & s_{N-1} \\ s_{-1} & s_0 & \cdots & s_{N-2} \end{bmatrix}$$

$\mathcal{G}_\tau$ is tall if $2P \geq 3$. It is clear that for any amount of oversampling ($P > 1$) this is satisfied.

We can imagine several forms of filtering based on this model.

1. *Matched filtering by* $\mathcal{G}_\tau$. The result after matched filtering is $\mathbf{y}_k = \mathcal{G}_\tau^{\mathrm{H}}[\begin{smallmatrix}\mathbf{x}_k \\ \mathbf{x}_{k+1}\end{smallmatrix}]$, a vector with 3 entries, and containing the contributions of 3 symbols, mixed via $\mathcal{G}_\tau^{\mathrm{H}}\mathcal{G}_\tau$ (a 3×3 matrix).

2. *Matched filtering by* $\mathbf{g}(\tau)$. This is a more common operation, and equal to performing integrate-and-dump filtering after a synchronization delay by τ. The data model is regarded as a signal of interest (the center row of $\mathcal{S}$, premultiplied by $\mathbf{g}(\tau)$: the center column of $\mathcal{G}_\tau$),

$$\mathcal{X} \;=\; \begin{bmatrix}\mathbf{g}_1 \\ \mathbf{g}_2\end{bmatrix} [s_0 \quad s_1 \quad \cdots \quad s_{N-1}] \;+\; \begin{bmatrix} \mathbf{0} & \mathbf{g}_2 \\ \mathbf{g}_1 & \mathbf{0} \end{bmatrix} \begin{bmatrix} s_1 & s_2 & \cdots & s_N \\ s_{-1} & s_0 & \cdots & s_{N-2} \end{bmatrix}$$

   The second term is regarded as part of the noise. As such, it has a covariance matrix

$$\mathbf{R}_{\mathcal{N}} = \begin{bmatrix} \mathbf{g}_2\mathbf{g}_2^{\mathrm{H}} & \mathbf{0} \\ \mathbf{0} & \mathbf{g}_1\mathbf{g}_1^{\mathrm{H}} \end{bmatrix}$$

   The result after matched filtering is a 1-dimensional sequence $\{y_k\}$,

$$y_k = \mathbf{g}(\tau)^{\mathrm{H}} \begin{bmatrix}\mathbf{x}_k \\ \mathbf{x}_{k+1}\end{bmatrix} = [\mathbf{g}(\tau)^{\mathrm{H}}\mathbf{g}(\tau)]s_k + \mathbf{g}(\tau)^{\mathrm{H}} \begin{bmatrix}\mathbf{n}_k \\ \mathbf{n}_{k+1}\end{bmatrix}$$

   where the noise at the output due to ISI has variance

$$\mathbf{g}(\tau)^{\mathrm{H}}\mathbf{R}_{\mathcal{N}}\mathbf{g}(\tau) = [\mathbf{g}_1^{\mathrm{H}} \quad \mathbf{g}_2^{\mathrm{H}}] \begin{bmatrix} \mathbf{g}_2\mathbf{g}_2^{\mathrm{H}} & \mathbf{0} \\ \mathbf{0} & \mathbf{g}_1\mathbf{g}_1^{\mathrm{H}} \end{bmatrix} \begin{bmatrix}\mathbf{g}_1 \\ \mathbf{g}_2\end{bmatrix} = 2|\mathbf{g}_1^{\mathrm{H}}\mathbf{g}_2|^2$$

   If $\mathbf{g}_1$ is not orthogonal to $\mathbf{g}_2$, then the noise due to ISI is not zero. Since these vectors are dependent on τ, this will generally be the case. With temporally white noise added to the samples, there will also be a contribution $\sigma^2(\mathbf{g}_1^{\mathrm{H}}\mathbf{g}_1 + \mathbf{g}_2^{\mathrm{H}}\mathbf{g}_2)$ to the output noise variance.[4]

[4] In actuality, the noise will not be white but shaped by the receiver filter.

3. *Zero-forcing filtering* and selection of one output. This solution can be regarded as the matched filter of item 1, followed by a de-mixing step (multiplication by $(\mathcal{G}_\tau^{\mathrm{H}}\mathcal{G}_\tau)^{-1}$), and selection of one of the outputs. The resulting filter is

$$\mathbf{w} = \mathcal{G}_\tau(\mathcal{G}_\tau^{\mathrm{H}}\mathcal{G}_\tau)^{-1}\begin{bmatrix}0\\1\\0\end{bmatrix} = (\mathcal{G}_\tau\mathcal{G}_\tau^{\mathrm{H}})^\dagger\mathbf{g}(\tau)$$

   and the output will be $[s_0 \;\; s_1 \;\; \cdots \;\; s_{N-1}]$. Note that in principle we could select also one of the other outputs, this would give only a shift in the output sequence ($[s_1 \;\; s_2 \;\; \cdots \;\; s_N]$ or $[s_{-1} \;\; s_0 \;\; \cdots \;\; s_{N-2}]$). With noise, however, reconstructing the center sequence is likely to give the best performance since it carries the most energy.

4. *Wiener filtering.* This is

$$\mathbf{w} = \hat{\mathbf{R}}_{\mathcal{X}}^{-1}\hat{\mathbf{R}}_{\mathcal{X}\mathcal{S}}\begin{bmatrix}0\\1\\0\end{bmatrix}.$$

   Under noise-free conditions, this is asympotically equal to $\mathbf{w} = (\mathcal{G}_\tau\mathcal{G}_\tau^{\mathrm{H}})^\dagger\mathbf{g}(\tau)$, i.e., the zero-forcing filter. In the presence of noise, however, it is more simply implemented by direct inversion of the data covariance matrix. Among the linear filtering schemes considered here, the Wiener filter is probably the preferred filter since it maximizes the output SINR.

   As we have seen before, the Wiener filter is asymptotically also equal to a scaling of $\mathbf{R}_{\mathcal{N}}^{-1}\mathbf{g}(\tau)$, i.e., the result of item 2, taking the correlated ISI-noise into account. (This equivalence can however only be shown if there is some amount of additive noise as well, or else $\mathbf{R}_{\mathcal{N}}$ and $\mathbf{R}_{\mathcal{X}}$ are not invertible.)

Delay estimation In general, the delay τ by which the data is received is unknown and has to be estimated from the data as well. This is a question very related to that of the DOA estimation considered in the previous section. Indeed, in an ISI-free model $\mathbf{x}_k = \mathbf{g}(\tau)s_k$, the problem is similar to $\mathbf{x}_k = \mathbf{a}(\theta)s_k$, but for a different functional. The traditional technique in communications is to use the "classical beamformer": scan the matched filter over a range of τ, and take that τ that gives the peak response. As we have seen in the previous sections, this is optimal if there is only a single component in noise, i.e., no ISI. With ISI, the technique relies on a sufficient orthogonality of the columns of $\mathcal{G}_\tau$. This is however not guaranteed, and the resolution may be poor.

We may however also use the MUSIC algorithm. This is implemented here as follows: compute the SVD of $\mathcal{X}$, or the eigenvalue decomposition of $\mathbf{R}_{\mathcal{X}}$. In either case, we obtain a basis $\mathbf{U}_s$ for the column span of $\mathcal{X}$. In noise-free conditions or asymptotically for a large number of samples, we know that the rank of $\mathcal{X}$ is 3, so that $\mathbf{U}_s$ has 3 columns, and that

$$\mathrm{span}\{\mathbf{U}_s\} = \mathrm{span}\{\mathcal{G}_\tau\} = \mathrm{span}\{\begin{bmatrix}\mathbf{0} & \mathbf{g}_1 & \mathbf{g}_2\\ \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{0}\end{bmatrix}\}$$

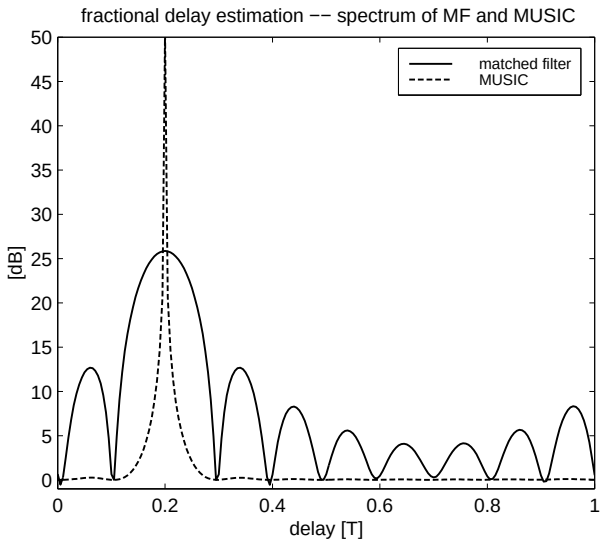


Figure 3.9. Delay estimation: spectra corresponding to the matched filter and MUSIC. The true delay is $0.2T$.

Thus, $\mathbf{g}(\tau)$ is in the span of $\mathbf{U}_s$. Therefore,

$$\mathbf{g}(\tau) \ \perp \ \mathbf{U}_n = (\mathbf{U}_s)^{\perp}$$

Thus, if we look at the MUSIC cost function (viz. (3.24))

$$J_{MUSIC}(\tau) = \frac{\mathbf{g}(\tau)^{\mathrm{H}}\hat{\mathbf{U}}_n\hat{\mathbf{U}}_n^{\mathrm{H}}\mathbf{g}(\tau)}{\mathbf{g}(\tau)^{\mathrm{H}}\mathbf{g}(\tau)}$$

it will be exactly zero when τ matches the true delay. Figure 3.9 shows the inverse of $J_{MUSIC}(\tau)$, compared to scanning the matched filter. It is obvious that the MUSIC provides a much higher resolution.

Bibliography

[1] D. Johnson and D. Dudgeon, *Array Signal Processing: Concepts and Techniques*. Prentice-Hall, 1993.

[2] S. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993.

[3] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs (NJ): Prentice-Hall, 1992.

[4] R. A. Monzingo and T. W. Miller, *Introduction to Adaptive Arrays.* New-York: Wiley-Interscience, 1980.

[5] H. Krim and M. Viberg, "Two decades of array signal processing research: The parametric approach," *IEEE Signal Processing Magazine*, vol. 13, pp. 67–94, July 1996.

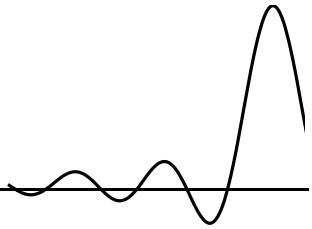
[6] B. van Veen and K. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP Magazine*, vol. 5, pp. 4–24, Apr. 1988.

[7] L. Scharf, *Statistical Signal Processing.* Reading, MA: Addison-Wesley, 1991.

[8] D. Brandwood, "A complex gradient operator and its application in adaptive array theory," *IEE Proc., parts F and H*, vol. 130, pp. 11–16, Feb. 1983.

Chapter 4

ADAPTIVE FILTERING



Contents

In chapter 3, we considered various beamforming and receiver strategies for the case where (part of) the transmitted signal is known. The objective was either to construct a receiver to recover the same signal from the observed data, or to estimate the channel (array response matrix), which is called model matching. In chapter 3, a data block was assumed to be available and we could use all of the samples to construct the solutions. In many cases, however, data is coming in sample by sample and the objective is to *update* the estimates as more data becomes known. Such algorithms are called *adaptive*. Adaptive algorithms are based on feedback: the error using the current beamforming weights or model is estimated, and then the weights are modified such that the error becomes smaller. In this chapter, we study the most well-known adaptive algorithms: the LMS and the RLS.

General references for this chapter are (Haykin 1992 [1], Widrow and Stearns 1985 [2]). A historical perspective and examples can be found in [3]. A more recent overview of adaptive filtering algorithms and acoustic echo cancellation can be found in [4].

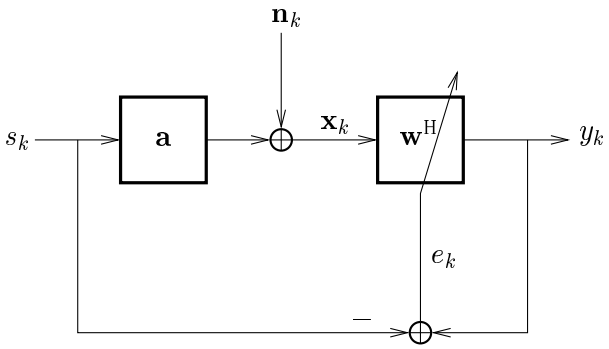


Figure 4.1. Adaptive output error minimization

4.1 WIENER FILTERING

Let us first briefly summarize some results from chapter 3. The scenario that we consider is depicted in figure 4.1. We consider a single source-of-interest in noise,

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k\,, \qquad k = 1, 2, \cdots\,.$$

The noise may contain also the effect of interfering sources. We assume that the source of interest has unit power,

$$\mathrm{E}(|s_k|^2) = 1\,,$$

and absorbe the scaling in $\mathbf{a}$. We define We further define the received data covariance matrix as $\mathbf{R}_x = \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})$ and the correlation between the received data and the symbols as $\mathbf{r}_{xs} = \mathrm{E}(\mathbf{x}_k\bar{s}_k)$.

In the present scenario, we consider a beamformer at the output that is to produce a replica of the signal. The objective of the beamformer is to minimize the output error. The corresponding cost function is

$$J(\mathbf{w}) \;=\; \mathrm{E}(|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k|^2)\,.$$

It can be worked out as

$$\begin{aligned} J(\mathbf{w}) &= \mathrm{E}[(\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k)(\mathbf{x}_k^{\mathrm{H}}\mathbf{w} - \bar{s}_k)] \\ &= \mathbf{w}^{\mathrm{H}}\mathbf{R}_x\mathbf{w} - \mathbf{w}^{\mathrm{H}}\mathbf{r}_{xs} - \mathbf{r}_{xs}^{\mathrm{H}}\mathbf{w} + 1\,. \end{aligned}$$

The gradient of $J(\mathbf{w})$ is (recall (3.13))

$$\nabla J(\mathbf{w}) = \mathbf{R}_x\mathbf{w} - \mathbf{r}_{xs}\,.$$

Let us denote the optimum that minimizes $J(\mathbf{w})$ by $\mathbf{w}_0$. At the optimum, $\nabla J(\mathbf{w}_0) = 0$, so that

$$\mathbf{R}_x\mathbf{w}_0 = \mathbf{r}_{xs} \qquad \Rightarrow \qquad \mathbf{w}_0 = \mathbf{R}_x^{-1}\mathbf{r}_{xs}\,.$$

The first expression is called the *normal equations*, and it leads to the second expression for the Wiener filter. Note that

$$\nabla J(\mathbf{w}) = 0 \qquad \Rightarrow \qquad \mathrm{E}(\mathbf{x}_k \mathbf{x}_k^{\mathrm{H}} \mathbf{w} - \mathbf{x}_k \bar{s}_k) = \mathbf{0} \qquad \Rightarrow \qquad \mathrm{E}(\mathbf{x}_k \bar{e}_k) = \mathbf{0}$$

where

$$y_k = \mathbf{w}^{\mathrm{H}} \mathbf{x}_k \,, \qquad e_k = y_k - s_k \,.$$

Thus, at the optimum, the output error e_k is uncorrelated to the input vector. Indeed, any correlated part can be used to reduce the error. This is called the *orthogonality principle*.

At the optimum, the remaining cost is

$$J(\mathbf{w}_0) = 1 - \mathbf{r}_{xs}^{\mathrm{H}} \mathbf{R}_x^{-1} \mathbf{r}_{xs} \;=:\; J_0,$$

and it is straightforward to verify that for any other $\mathbf{w}$,

$$J(\mathbf{w}) = J_0 + (\mathbf{w} - \mathbf{w}_0)^{\mathrm{H}} \mathbf{R}_x (\mathbf{w} - \mathbf{w}_0) \,.$$

This shows that $J(\mathbf{w})$ is a quadratic function (in the entries of $\mathbf{w}$) and that $\mathbf{w}_0$ is really the minimizer.

With finite data, all expectations are estimated from the available data:

$$\begin{aligned} \hat{\mathbf{R}}_x &= \tfrac{1}{N} \sum_{k=1}^{N} \mathbf{x}_k \mathbf{x}_k^{\mathrm{H}} \;=\; \tfrac{1}{N} \mathbf{X} \mathbf{X}^{\mathrm{H}} \\ \hat{\mathbf{r}}_{xs} &= \tfrac{1}{N} \sum_{k=1}^{N} \mathbf{x}_k \bar{s}_k \;=\; \tfrac{1}{N} \mathbf{X} \mathbf{s}^{\mathrm{H}} \,. \end{aligned}$$

If we define also the finite-sample cost function as

$$\begin{aligned} \hat{J}(\mathbf{w}) &= \tfrac{1}{N} \sum_{k=1}^{N} |e_k|^2 \\ &= \tfrac{1}{N} \sum_{k=1}^{N} |\mathbf{w}^{\mathrm{H}} \mathbf{x}_k - s_k|^2 \\ &= \tfrac{1}{N} \|\mathbf{w}^{\mathrm{H}} \mathbf{X} - \mathbf{s}\|^2 \\ &= \tfrac{1}{N} \left[\mathbf{w}^{\mathrm{H}} \mathbf{X} \mathbf{X}^{\mathrm{H}} \mathbf{w} - \mathbf{w}^{\mathrm{H}} \mathbf{X} \mathbf{s}^{\mathrm{H}} - \mathbf{s} \mathbf{X}^{\mathrm{H}} \mathbf{w} + \mathbf{s}^{\mathrm{H}} \mathbf{s} \right] \\ &= \mathbf{w}^{\mathrm{H}} \hat{\mathbf{R}}_x \mathbf{w} - \mathbf{w}^{\mathrm{H}} \hat{\mathbf{r}}_{xs} - \hat{\mathbf{r}}_{xs}^{\mathrm{H}} \mathbf{w} + \text{const.} \end{aligned}$$

then it is clear that the optimal finite-sample solution is $\hat{\mathbf{w}}_0 = \hat{\mathbf{R}}_x^{-1} \hat{\mathbf{r}}_{xs}$, and that at the optimum

$$\tfrac{1}{N} \sum \mathbf{x}_k \bar{e}_k = \mathbf{0} \qquad \Leftrightarrow \qquad \mathbf{X} \mathbf{e}^{\mathrm{H}} = \mathbf{0} \,, \qquad \mathbf{e} := [e_1 \quad e_2 \quad \cdots \quad e_N] \,.$$

Thus, the orthogonality principle also holds for finite samples.

4.2 STEEPEST GRADIENT DESCENT ALGORITHM

The optimal Wiener solution asks for the inversion of $\mathbf{R}_x$. In some cases, $\mathbf{R}_x$ is large and its inversion would be time-consuming. A common technique in optimization, used to find the minimum of all kinds of cost functions, is to resort to iterative techniques. A popular and robust technique (though not particularly fast) is based on the steepest descent algorithm. Given a function $f(x)$ of which we want to find the minimum, and an initial point $x^{(1)}$ with value $f^{(1)} := f(x^{(1)})$ and gradient $\nabla f^{(1)} = \nabla f(x^{(1)})$. For another point $x^{(2)}$ close to $x^{(1)}$, we can then write

$$\nabla f^{(1)} \;\approx\; \frac{f^{(2)} - f^{(1)}}{x^{(2)} - x^{(1)}}$$

or $f^{(2)} \approx f^{(1)} + (x^{(2)} - x^{(1)})\nabla f^{(1)}$. If we now choose

$$x^{(2)} \;=\; x^{(1)} - \mu \nabla f^{(1)}$$

where μ is a small number (called the *step size*), then $f^{(2)} \approx f^{(1)} - \mu(\nabla f^{(1)})^2$, which is making $f^{(1)}$ smaller. At the minimum, $\nabla f^{(1)} = 0$ and $x^{(2)} = x^{(1)}$. Thus, this algorithm can be viewed as making small steps in the direction of the negative gradient, for which the value of the function will get smaller.

In our application, we have a cost function $J(\mathbf{w})$, with complex gradient $\nabla J(\mathbf{w}) = \mathbf{R}_x\mathbf{w} - \mathbf{r}_{xs}$. The Steepest Gradient Descent algorithm becomes

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \mu(\mathbf{R}_x\mathbf{w}^{(k)} - \mathbf{r}_{xs})\,.$$

Since our cost function is a nice quadratic function in the entries of $\mathbf{w}$, with only a single minimum, we can arbitrarily select an initial point for the algorithm, typically $\mathbf{w}^{(0)} = \mathbf{0}$.

Figure 4.2(a) shows the cost function $J(\mathbf{w})$ for two dimensions, $\mathbf{w} = [w_1, w_2]^{\mathrm{T}}$. The horizontal axes are w_1 and w_2, the vertical axis is the corresponding $J(\mathbf{w})$. Since it is quadratic in the entries of $\mathbf{w}$, the cost function is a paraboloid. The contours are ellipses for which the cost is constant at a certain level. The curved lines show the convergence of the steepest gradient algorithm from certain starting points, and two different values of μ. For small μ, the algorithm follows precisely the direction of the negative gradient, and we obtain a curve that is orthogonal to all the contour lines. For larger μ, the algorithm tends to overshoot.[1]

Figure 4.2(b) shows the convergence as a function of time, and for various values of μ. The graphs are known as the *learning curves*. In this example, we took $d = 2$ sources coming from directions $[-10°,\ 20°]$ and with amplitudes $[1,\ 0.8]$, $M = 2$ antennas spaced at half-wavelength, and an SNR = 10 dB. Notice that if μ is larger, then the algorithm converges faster. However, there is a critical point beyond which μ is too large: the algorithm becomes unstable. Hence, there is a trade-off between stability and convergence rate.

[1] In this example, we took real-valued data to be able to show the cost function in two dimensions.

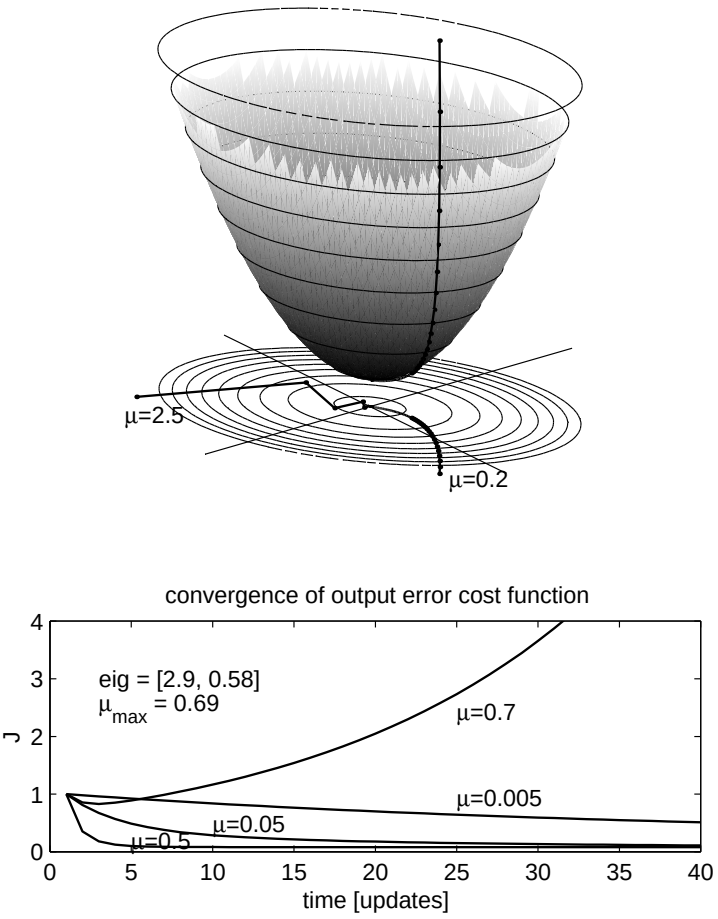


Figure 4.2. Convergence of the Steepest Gradient Descent Algorithm

4.2.1 Stability

We now analyze the stability properties of the algorithm. Let $\mathbf{w}_0$ denote the optimum, and define

$$\mathbf{c}^{(k)} = \mathbf{w}^{(k)} - \mathbf{w}_0 \,.$$

Thus, $\mathbf{c}^{(k)}$ is the error in the weight vector at the k-th iteration. To analyze the stability of the algorithm, we derive the following recursion for $\mathbf{c}^{(k)}$:

$$\begin{array}{rcll} \mathbf{w}^{(k+1)} & = & \mathbf{w}^{(k)} & - \quad \mu(\mathbf{R}_x \mathbf{w}^{(k)} - \mathbf{r}_{xs}) \\ \mathbf{w}_0 & = & \mathbf{w}_0 & - \quad \mu(\mathbf{R}_x \mathbf{w}_0 - \mathbf{r}_{xs}) \\ \hline \mathbf{c}^{(k+1)} & = & \mathbf{c}^{(k)} & - \quad \mu \mathbf{R}_x \mathbf{c}^{(k)} \,. \end{array}$$

Hence

$$\mathbf{c}^{(k+1)} \;=\; (\mathbf{I} - \mu \mathbf{R}_x)\mathbf{c}^{(k)} \;=\; \cdots \;=\; (\mathbf{I} - \mu \mathbf{R}_x)^{k+1}\mathbf{c}^{(0)} \,. \tag{4.1}$$

We thus see that the stability of the recursion depends on whether $(\mathbf{I} - \mu \mathbf{R}_x)^k$ converges to zero. This in turn depends on the eigenvalues of $(\mathbf{I} - \mu \mathbf{R}_x)$: introduce the eigenvalue decomposition

$$\mathbf{I} - \mu \mathbf{R}_x \;=:\; \mathbf{U} \mathbf{\Lambda}_\mu \mathbf{U}^{\mathrm{H}}$$

then it follows that

$$(\mathbf{I} - \mu \mathbf{R}_x)^k \;=\; \mathbf{U} (\mathbf{\Lambda}_\mu)^k \mathbf{U}^{\mathrm{H}}$$

since $\mathbf{U}$ is unitary. If we now introduce a change of variables, $\mathbf{v}^{(k)} := \mathbf{U}^{\mathrm{H}} \mathbf{c}^{(k)}$ then

$$\mathbf{v}^{(k)} \;=\; (\mathbf{\Lambda}_\mu)^k \mathbf{v}^{(0)} \,.$$

Note that $\|\mathbf{v}^{(k)}\| = \|\mathbf{c}^{(k)}\|$, so that the change of variables does not change the norm. From this it is clear that $\|\mathbf{c}^{(k)}\| \to 0$ if and only if all eigenvalues $\lambda_{\mu,i}$ satisfy $|\lambda_{\mu,i}| < 1$. Under this condition the recursion is stable.

Let us now compute these eigenvalues. Introduce the eigenvalue decomposition of $\mathbf{R}_x$:

$$\mathbf{R}_x \;=\; \mathbf{U} \mathbf{\Lambda} \mathbf{U}^{\mathrm{H}}$$

then

$$\mathbf{I} - \mu \mathbf{R}_x \;=\; \mathbf{U}\mathbf{U}^{\mathrm{H}} - \mu \mathbf{U} \mathbf{\Lambda} \mathbf{U}^{\mathrm{H}} \;=\; \mathbf{U}(\mathbf{I} - \mu \mathbf{\Lambda}) \mathbf{U}^{\mathrm{H}} \,.$$

We thus see that $\mathbf{\Lambda}_\mu = \mathbf{I} - \mu \mathbf{\Lambda}$. It follows that the recursion is stable if and only if

$$\begin{array}{rcll} & |1 - \mu\lambda_i| & < & 1 \,, \qquad i = 1, \cdots, M \\ \Leftrightarrow & 0 \;<\; \mu\lambda_i & < & 2 \,. \end{array}$$

The largest value that μ may take is constrained by $\lambda_{\max}$, hence we finally obtain the result that *the steepest gradient descent algorithm is stable if*

$$0 < \mu < \frac{2}{\lambda_{\max}} \tag{4.2}$$

In the example in figure 4.2(b), the largest eigenvalue was $\lambda_{\max} = 2.9$, so that $\mu_{\max} = 0.69$. Indeed, for $\mu = 0.7$, the algorithm is seen to be instable. Note that

- The stability is independent of the initialization of the recursion.
- The maximal value of μ is data dependent. This is a large disadvantage in practical implementations.

4.2.2 Convergence rate

Let us now consider the convergence speed. From

$$\mathbf{v}^{(k)} \;=\; (\mathbf{\Lambda}_\mu)^k \mathbf{v}^{(0)} \;=\; (\mathbf{I} - \mu\mathbf{\Lambda})^k \mathbf{v}^{(0)}\,,$$

it is seen that each entry of $\mathbf{v}^{(k)}$ converges with a rate determined by $|1-\mu\lambda_i|$. The entry that converges the *slowest* is determined by that $|1-\mu\lambda_i|$ that is closest to 1.

If μ satisfies $1-\mu\lambda_{\max} > 0$, i.e., $\mu < 1/\lambda_{\max}$, then the slowest mode is determined by $\lambda_{\min}$. We can define a corresponding time constant τ for which $\|\mathbf{v}^{(\tau)}\| = \|\mathbf{v}^{(0)}\|/\mathrm{e}$, i.e.,

$$(1-\mu\lambda_{\min})^\tau = \frac{1}{\mathrm{e}}\,.$$

For sufficiently small μ, it follows that

$$\tau \;=\; \frac{-1}{\ln(1-\mu\lambda_{\min})} \;\approx\; \frac{1}{\mu\lambda_{\min}}\,.$$

Thus, the convergence rate is inversely proportional to μ and the smallest eigenvalue of $\mathbf{R}_x$. If we select μ large, e.g., $\mu = \frac{1}{\lambda_{\max}}$, then

$$\tau \;\approx\; \frac{\lambda_{\max}}{\lambda_{\min}} \;=:\; \mathrm{cond}(\mathbf{R}_x)\,. \tag{4.3}$$

Thus, we see that the maximal convergence speed that can be achieved is limited by the conditioning of the matrix $\mathbf{R}_x$. If the eigenvalues of $\mathbf{R}_x$ are widely spread, e.g., because the signals come from close directions or because one signal is much weaker than the others, then the convergence will be slow.

4.3 THE LMS ALGORITHM

As we have seen, the steepest gradient descent algorithm is based on taking small steps into the opposite direction of the complex gradient of the cost function,

$$\nabla J(\mathbf{w}) \;=\; \mathbf{R}_x\mathbf{w} - \mathbf{r}_{xs}\,.$$

Until now, we had assumed that the matrix $\mathbf{R}_x$ and vector $\mathbf{r}_{xs}$ are perfectly known,

$$\begin{aligned}\mathbf{R}_x &= \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\\ \mathbf{r}_{xs} &= \mathrm{E}(\mathbf{x}_k\bar{s}_k)\end{aligned}$$

where s_k is the desired response. In practical situations, these quantities have to be estimated from the data. The Least-Mean-Square algorithm (LMS; Widrow 1975) is obtained by making the following extremely simple estimates,

$$\begin{aligned} \hat{\mathbf{R}}_x &= \mathbf{x}_k \mathbf{x}_k^{\mathrm{H}} \\ \hat{\mathbf{r}}_{xs} &= \mathbf{x}_k \bar{s}_k \end{aligned}$$

The estimates are based just on the current samples, and not averaged at all over several samples.[2] The resulting instantaneous gradient estimate is

$$\begin{aligned} \widehat{\nabla J}(\mathbf{w}) &= \mathbf{x}_k \mathbf{x}_k^{\mathrm{H}} \mathbf{w} - \mathbf{x}_k \bar{s}_k \\ &= \mathbf{x}_k (\mathbf{x}_k^{\mathrm{H}} \mathbf{w} - \bar{s}_k) \\ &= \mathbf{x}_k \bar{e}_k \,, \qquad e_k = \mathbf{w}^{\mathrm{H}} \mathbf{x}_k - s_k = y_k - s_k \,. \end{aligned}$$

Note that y_k is the output of the beamformer for the present value of $\mathbf{w}$, and e_k is the output error. The LMS algorithm thus becomes

$$\begin{aligned} y_k &:= \hat{\mathbf{w}}^{(k)H} \mathbf{x}_k \\ e_k &:= y_k - s_k \\ \hat{\mathbf{w}}^{(k+1)} &:= \hat{\mathbf{w}}^{(k)} - \mu \mathbf{x}_k \bar{e_k} \end{aligned}$$

It is usually initialized by setting $\hat{\mathbf{w}}^{(0)} = \mathbf{0}$. The large advantage of the LMS algorithm is its small complexity: only $2M + 1$ complex multiplications per update step.

Figure 4.3 shows an example of the convergence of the LMS algorithm.[3] It is seen that for small values of μ, the algorithm stays close to the Steepest Gradient Descent algorithm, but for larger values of μ, the convergence becomes erratic. This is because the algorithm acts on noisy instantaneous values of the data which are insufficiently damped by a large step size μ.

4.4 ANALYSIS OF THE LMS

4.4.1 Convergence in the mean

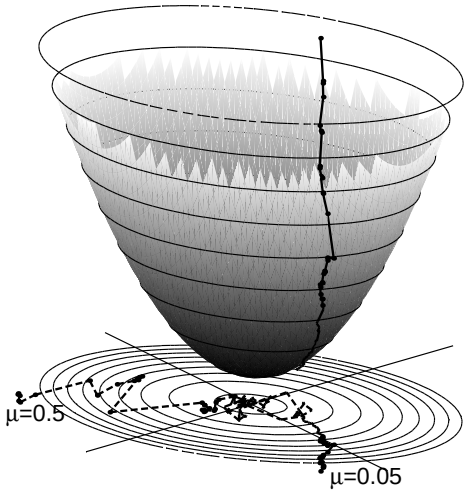
Consider the error in the weight vector computed by LMS, when compared to the optimal (Wiener) weight, $\mathbf{w}_0 = \mathbf{R}_x^{-1} \mathbf{r}_{xs}$,

$$\boldsymbol{\epsilon}_k = \mathbf{w}^{(k)} - \mathbf{w}_0$$

If $k \to \infty$, does the weight error go to zero? This cannot be answered directly like this, because the algorithm acts on instantaneous (noisy) data and hence never converges. However, we can

[2] The averaging is obtained implicitly by choosing a sufficiently small step size.

[3] The data used in part (a) and (b) of the figure is not the same; part (a) is based on real data, whereas the conditions for part (b) are the same as in figure 4.2(b).



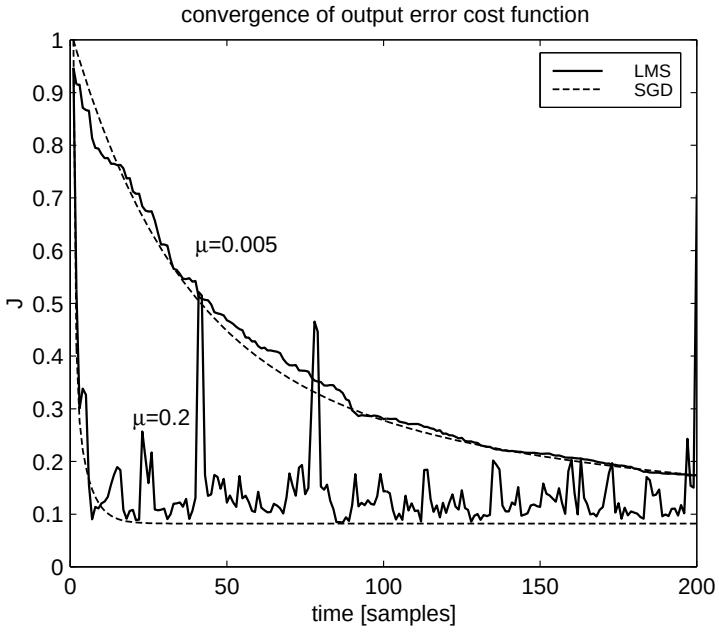


Figure 4.3. Convergence of the LMS

consider the convergence of $\mathrm{E}(\mathbf{w}^{(k)})$, the ensemble-averaged weight vector (obtained by averaging $\mathbf{w}^{(k)}$ over many independent realizations of $\mathbf{x}$). For this we can say that

$$\begin{array}{rcl|l|l|l|l} \hat{\mathbf{w}}^{(k+1)} & = & \hat{\mathbf{w}}^{(k)} & - & \mu[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w}^{(k)} & - & \mathbf{x}_k\bar{s}_k] \\ \mathbf{w}_0 & = & \mathbf{w}_0 & - & \mu[\mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathbf{w}_0 & - & \mathrm{E}(\mathbf{x}_k\bar{s}_k)] \\ \hline \boldsymbol{\epsilon}_{k+1} & = & \boldsymbol{\epsilon}_k & - & \mu[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w}^{(k)} - \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathbf{w}_0 & - & (\mathbf{x}_k\bar{s}_k - \mathrm{E}(\mathbf{x}_k\bar{s}_k))] \end{array}$$

Since $\mathbf{x}_k$ is independent of $\mathbf{w}^{(k)}$, it follows that $\mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w}^{(k)}) = \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathrm{E}(\mathbf{w}^{(k)})$, so that

$$\begin{aligned} \mathrm{E}(\boldsymbol{\epsilon}_{k+1}) &= \mathrm{E}(\boldsymbol{\epsilon}_k) - \mu(\mathrm{E}[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}]\mathrm{E}(\boldsymbol{\epsilon}_k) - \mathbf{0}) \\ &= \mathrm{E}(\boldsymbol{\epsilon}_k) - \mu\mathbf{R}_x\mathrm{E}(\boldsymbol{\epsilon}_k) \\ &= (\mathbf{I} - \mu\mathbf{R}_x)\mathrm{E}(\boldsymbol{\epsilon}_k) \end{aligned}$$

Hence $\mathrm{E}(\boldsymbol{\epsilon}_k)$ satisfies exactly the same recursion as the weight error $\mathbf{c}^{(k)}$ of the Steepest Gradient Descent algorithm (4.1). It follows that the (ensemble-)average behavior of the weight vector convergence of LMS is the same as that of the SGD. Also the conditions for convergence are the same as (4.2):

$$0 \;<\; \mu \;<\; \frac{2}{\lambda_{\max}}.$$

Under this condition, LMS is said to be *convergent in the mean.*

4.4.2 Convergence in mean-square

Similar as for $\hat{\mathbf{w}}^{(k)}$, the cost function $\hat{J}_k = e_k\bar{e}_k$ also is a stochastic quantity. Let us therefore consider the (ensemble-)average value of it,

$$\begin{aligned} J_k &:= \mathrm{E}(e_k\bar{e}_k) \\ &= \mathrm{E}([\mathbf{w}^{(k)H}\mathbf{x}_k - s_k]^{\mathrm{H}}[\mathbf{w}^{(k)H}\mathbf{x}_k - s_k]) \qquad (\mathbf{w}^{(k)} = \mathbf{w}_0 + \boldsymbol{\epsilon}_k) \\ &= \mathrm{E}([\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k]^{\mathrm{H}}[\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k] + \boldsymbol{\epsilon}_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\boldsymbol{\epsilon}_k + \boldsymbol{\epsilon}_k^{\mathrm{H}}\mathbf{x}_k(\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k) + (\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k)^{\mathrm{H}}\mathbf{x}_k^{\mathrm{H}}\boldsymbol{\epsilon}_k) \\ &= J_{\min} + \mathrm{E}(\boldsymbol{\epsilon}_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\boldsymbol{\epsilon}_k) \end{aligned}$$

where $J_{\min}$ is the output error of the Wiener receiver. The second term can be regarded as the *excess mean-squared error.* This error due to misadjustment is the price paid for adaptivity. It can be further written as

$$\begin{aligned} J_{ex}(k) &= \mathrm{E}(\boldsymbol{\epsilon}_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\boldsymbol{\epsilon}_k) \\ &= \mathrm{E}(\mathrm{tr}[\boldsymbol{\epsilon}_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\boldsymbol{\epsilon}_k]) \\ &= \mathrm{E}(\mathrm{tr}[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\boldsymbol{\epsilon}_k\boldsymbol{\epsilon}_k^{\mathrm{H}}]) \\ &= \mathrm{tr}(\mathbf{R}_x\mathbf{K}_k)\,, \qquad \mathbf{K}_k := \mathrm{E}(\boldsymbol{\epsilon}_k\boldsymbol{\epsilon}_k^{\mathrm{H}}) \end{aligned}$$

where we have used the property of the trace operator: $\mathrm{tr}(\mathbf{AB}) = \mathrm{tr}(\mathbf{BA})$. Thus, the excess mean-squared error is dependent on the data covariance matrix and the covariance of the weight vector error. It is possible to derive a recursion for this error, see [1]. This then describes the

transient behavior of the averaged algorithm as it converges to its final value, $J_\infty = J_{\min} + J_{ex}(\infty)$.

Since the algorithm never converges but keeps responding to new data samples, the final weight vector jitters around $\mathbf{w}_0$ with a variance $\mathbf{K}_\infty$, and thus $J_{ex}(\infty)$ is nonzero. It can be shown [1] that, if μ satisfies

$$\gamma \;:=\; \sum_{i=1}^{M} \frac{\mu\lambda_i}{2-\mu\lambda_i} \;<\; 1$$

then the mean-squared error of the LMS algorithm converges and this asymptotic excess error satisfies

$$J_{ex}(\infty) \;=\; J_{\min}\frac{\gamma}{1-\gamma}$$

Suppose that $\mu\lambda_i \ll 1$, $i = 1, \cdots, M$, then this can be simplified to

$$J_{ex}(\infty) \;\approx\; J_{\min}\gamma \;\approx\; J_{\min}\tfrac{1}{2}\mu\sum_{i=1}^{M}\lambda_i$$

Moreover, $\sum_{i=1}^{M}\lambda_i$ has an interpretation as the average input power,

$$\begin{aligned}\sum_{i=1}^{M}\lambda_i \;=\; \mathrm{tr}(\mathbf{R}_x) \;&=\; \mathrm{tr}(\mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})) \\ &=\; \mathrm{tr}(\mathrm{E}(\mathbf{x}_k^{\mathrm{H}}\mathbf{x}_k)) \;=\; \mathrm{E}(\|\mathbf{x}_k\|^2)\end{aligned}$$

In summary, if

$$0 \;<\; \mu \;<\; \frac{2}{\mathrm{E}(\|\mathbf{x}_k\|^2)}$$

then the LMS converges in the mean-square, and the cost function converges to

$$J(\infty) \;\approx\; J_{\min}[1 + \tfrac{1}{2}\mu\mathrm{E}(\|\mathbf{x}_k\|^2)]$$

We make the following remarks:

1. The misadjustment, defined as $J_{ex}(\infty)/J_{\min}$, is in this approximation linearly dependent on the step size μ. A smaller step size gives a smaller error.

2. On the other hand, the convergence speed is inversely proportional to the step size (as in the steepest gradient descent algorithm, see (4.3)). Thus, there is a trade-off between speed and accuracy. For small μ, the adaptation is slow, but the excess error (asymptotic variance of $\hat{\mathbf{w}}$) is smaller.

3. This trade-off (choice of μ) is data dependent. It depends on the input power (average eigenvalue of the data covariance matrix) and the smallest eigenvalue of the data covariance matrix. Hence, note that also the number of antennas (filter taps) plays a role.

4.5 NORMALIZED LMS

As discussed above, the value of μ in the LMS algorithm depends on the *scaling* of the data. Indeed,

$$\hat{\mathbf{w}}^{(k+1)} \;=\; \hat{\mathbf{w}}^{(k)} \;-\; \mu \mathbf{x}_k \bar{e}_k\,, \qquad e_k \;=\; \mathbf{w}^{(k)H}\mathbf{x}_k \;-\; s_k\,,$$

and if $\mathbf{x}_k$ is scaled by some factor α (e.g., because of a different antenna gain or propagation conditions), then e_k also scales with α, and μ has to be scaled by α^{-2} to obtain the same weights.

To obtain a scaling-invariant algorithm, a modification of the LMS called the Normalized LMS (NLMS) has been derived (Goodwin; 1984). It is defined by the following recursion:

$$\hat{\mathbf{w}}^{(k+1)} \;=\; \hat{\mathbf{w}}^{(k)} \;-\; \frac{\tilde{\mu}}{\|\mathbf{x}_k\|^2}\mathbf{x}_k \bar{e}_k$$

We can make the following remarks:

1. In comparison to the original LMS, we see that we have set $\mu_k = \tilde{\mu}/\|\mathbf{x}_k\|^2$. The scaling by the instantaneous input power makes the modified step size $\tilde{\mu}$ scale-independent. At the same time, the step size can be regarded as *time-varying*.

2. The NLMS is convergent in the mean-square if and only if

   $$0 \;<\; \tilde{\mu} \;<\; 2,$$

   and the MSE is given by

   $$J(\infty) \;\approx\; J_{\min}[1 + \tfrac{1}{2}\tilde{\mu}].$$

3. To avoid division by zero, one often adds a small positive number to $\|\mathbf{x}_k\|^2$.

4. For stationary inputs, the convergence properties of LMS and NLMS are quite similar, if μ and $\tilde{\mu}$ are related as $\mu_k = \tilde{\mu}/\mathrm{E}(\|\mathbf{x}_k\|^2)$. The advantage of NLMS is that the input power doesn't have to be known, and that it is allowed to vary in time (non-stationary inputs). This is essential in wireless communications with fading channels.

The NLMS can be derived as follows. Consider the usual cost function

$$\begin{aligned} J(\mathbf{w}) &= \mathrm{E}(\|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k\|^2) \\ &= \mathbf{w}^{\mathrm{H}}\mathbf{R}_x\mathbf{w} - \mathbf{w}^{\mathrm{H}}\mathbf{r}_{xs} - \mathbf{r}_{xs}^{\mathrm{H}}\mathbf{w} + 1 \end{aligned}$$

and the steepest gradient descent update rule

$$\hat{\mathbf{w}}^{(k+1)} = \hat{\mathbf{w}}^{(k)} - \mu_k \boldsymbol{\nabla}_k\,, \qquad \boldsymbol{\nabla}_k = \mathbf{R}_x\mathbf{w}^{(k)} - \mathbf{r}_{xs}$$

where $\boldsymbol{\nabla}_k$ is the gradient of $J(\mathbf{w}^{(k)})$ based on the current estimate $\mathbf{w}^{(k)}$. Previously, we have taken a fixed step size $\mu_k = \mu$, which gave the Steepest Gradient Descent algorithm. LMS followed by inserting instantaneous estimates. The idea now is to consider a varying μ_k and

to compute which value of the step size optimizes the cost at time $k+1$. This line search optimization problem leads to

$$\frac{\mathrm{d}J_{k+1}}{\mathrm{d}\mu_k} \;=\; 0\,.$$

The derivative is given by

$$\begin{aligned}\frac{\mathrm{d}J_{k+1}}{\mathrm{d}\mu_k} &= \frac{\mathrm{d}}{\mathrm{d}\mu_k}\Big[\,(\hat{\mathbf{w}}^{(k)}-\mu_k\boldsymbol{\nabla}_k)^{\mathrm{H}}\mathbf{R}_x(\hat{\mathbf{w}}^{(k)}-\mu_k\boldsymbol{\nabla}_k)\\ &\qquad -(\hat{\mathbf{w}}^{(k)}-\mu_k\boldsymbol{\nabla}_k)^{\mathrm{H}}\mathbf{r}_{xs} \;-\; \mathbf{r}_{xs}^{\mathrm{H}}(\hat{\mathbf{w}}^{(k)}-\mu_k\boldsymbol{\nabla}_k) \;+\; 1\,\Big]\\ &= -\boldsymbol{\nabla}_k^{\mathrm{H}}\mathbf{R}_x(\hat{\mathbf{w}}^{(k)}-\mu_k\boldsymbol{\nabla}_k) \;-\; (\hat{\mathbf{w}}^{(k)}-\mu_k\boldsymbol{\nabla}_k)^{\mathrm{H}}\mathbf{R}_x\boldsymbol{\nabla}_k \;+\; \boldsymbol{\nabla}_k^{\mathrm{H}}\mathbf{r}_{xs} \;+\; \mathbf{r}_{xs}^{\mathrm{H}}\boldsymbol{\nabla}_k\\ &= 2\mu_k(\boldsymbol{\nabla}_k^{\mathrm{H}}\mathbf{R}_x\boldsymbol{\nabla}_k) \;-\; 2\boldsymbol{\nabla}_k^{\mathrm{H}}\boldsymbol{\nabla}_k\,.\end{aligned}$$

Thus, the optimal value for μ_k is given by

$$\mu_{k,opt} \;=\; \frac{\boldsymbol{\nabla}_k^{\mathrm{H}}\boldsymbol{\nabla}_k}{\boldsymbol{\nabla}_k^{\mathrm{H}}\mathbf{R}_x\boldsymbol{\nabla}_k}$$

Inserting as in LMS instantaneous values for the gradient, $\hat{\boldsymbol{\nabla}}_k = \mathbf{x}_k\bar{e}_k$, and for $\mathbf{R}_x$, i.e., $\hat{\mathbf{R}}_x = \mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}$, we obtain

$$\mu_{k,opt} \;=\; \frac{e_k\mathbf{x}_k^{\mathrm{H}}\mathbf{x}_k\bar{e}_k}{e_k\mathbf{x}_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{x}_k\bar{e}_k} \;=\; \frac{1}{\mathbf{x}_k^{\mathrm{H}}\mathbf{x}_k}$$

In practice, we would not use this optimal value but a fraction of it,

$$\mu_k \;=\; \tilde{\mu}\mu_{k,opt} \;=\; \tilde{\mu}\frac{1}{\mathbf{x}_k^{\mathrm{H}}\mathbf{x}_k}$$

where usually $0 < \tilde{\mu} < 1$. For $1 < \tilde{\mu} < 2$, the algorithm is still stable, but we continuously overshoot the optimal location, and the result is rather noisy. See figure 4.4.

4.6 THE RLS ALGORITHM

The LMS algorithm was inspired by a desire to iteratively compute an estimate of the Wiener receiver, without explicitly inverting the data covariance matrix. In this section we look at an alternative approach, where we keep track of the inverse as more and more data becomes available. The algorithm is based on the following matrix equality, known as Woodbury's identity or the Matrix Inversion Lemma.

4.6.1 Matrix inversion lemma

LEMMA 4.1. *For matrices of compatible sizes, and assuming all inverses exist,*

$$(\,\mathbf{A}-\mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B}\,)^{-1} = \mathbf{A}^{-1}+\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}}(\mathbf{C}-\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}})^{-1}\mathbf{B}\mathbf{A}^{-1}\,.$$

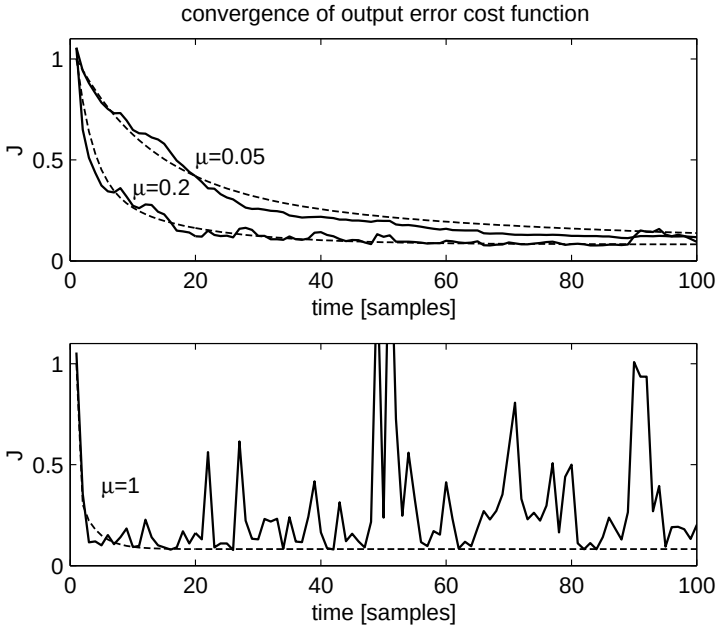


Figure 4.4. Convergence of NLMS

PROOF We can factor the following block-partitioned matrix in two ways (UDL or LDU):

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{H}} \\ \mathbf{B} & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} - \mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B} & \\ & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{C}^{-1}\mathbf{B} & \mathbf{I} \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{B}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \\ & \mathbf{C} - \mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}} \\ 0 & \mathbf{I} \end{bmatrix}.$$

Thus, the inverse can be written as

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{H}} \\ \mathbf{B} & \mathbf{C} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & 0 \\ -\mathbf{C}^{-1}\mathbf{B} & \mathbf{I} \end{bmatrix} \begin{bmatrix} (\mathbf{A} - \mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B})^{-1} & 0 \\ 0 & \mathbf{C}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1} \\ 0 & \mathbf{I} \end{bmatrix}$$
$$= \begin{bmatrix} (\mathbf{A} - \mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B})^{-1} & * \\ * & * \end{bmatrix}$$

but also as

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{H}} \\ \mathbf{B} & \mathbf{C} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{-1} & \\ & (\mathbf{C} - \mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ -\mathbf{B}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}}(\mathbf{C} - \mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}})^{-1}\mathbf{B}\mathbf{A}^{-1} & * \\ * & * \end{bmatrix}.$$

□

4.6.2 Infinite-horizon RLS algorithm

Suppose at time k we have collected k data samples $\mathbf{X}_k = [\mathbf{x}_1, \cdots, \mathbf{x}_k]$. Given the desired response $\mathbf{s}_k = [s_1, \cdots, s_k]$, the best solution to the linear least squares problem

$$\hat{\mathbf{w}}_k = \arg\min_{\mathbf{w}} \| \mathbf{w}^{\mathrm{H}}\mathbf{X}_k - \mathbf{s}_k \|^2$$

is

$$\hat{\mathbf{w}}_k = \mathbf{X}_k^{\dagger}\mathbf{s}_k^{\mathrm{H}} = (\mathbf{X}_k\mathbf{X}_k^{\mathrm{H}})^{-1}\mathbf{X}_k\mathbf{s}_k^{\mathrm{H}}$$

This is also the best estimate of the Wiener receiver $\hat{\mathbf{w}}_k = \hat{\mathbf{R}}_k^{-1}\hat{\mathbf{r}}_k$ where

$$\hat{\mathbf{R}}_k = \frac{1}{k}\sum_{i=1}^{k}\mathbf{x}_i\mathbf{x}_i^{\mathrm{H}}, \qquad \hat{\mathbf{r}}_k = \frac{1}{k}\sum_{i=1}^{k}\mathbf{x}_i\bar{s}_i$$

It is clear that we can drop the scaling $\frac{1}{k}$ in both $\hat{\mathbf{R}}_k$ and $\hat{\mathbf{a}}_k$ without changing the result $\hat{\mathbf{r}}_k$. Thus define

$$\mathbf{\Phi}_k := \mathbf{X}_k\mathbf{X}_k^{\mathrm{H}} = \sum_{i=1}^{k}\mathbf{x}_i\mathbf{x}_i^{\mathrm{H}}, \qquad \boldsymbol{\theta}_k := \mathbf{X}_k\mathbf{s}_k^{\mathrm{H}} = \sum_{i=1}^{k}\mathbf{x}_i\bar{s}_i\,.$$

Then $\hat{\mathbf{w}}_k = \mathbf{\Phi}_k^{-1}\boldsymbol{\theta}_k$.

Now we consider the computation of $\hat{\mathbf{w}}_{k+1}$. For this we require the inverse of $\mathbf{\Phi}_{k+1}$. However, note that $\mathbf{\Phi}_{k+1} = \mathbf{\Phi}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}}$. We can thus obtain $\mathbf{\Phi}_{k+1}$ from $\mathbf{\Phi}_k$ via a rank-1 update. Using the matrix inversion lemma, we can compute an update rule for $\mathbf{\Phi}_{k+1}^{-1}$ as well. Indeed, if we identify in lemma 4.1

$$\begin{array}{rcl} \mathbf{A} & \rightarrow & \mathbf{\Phi}_k \\ \mathbf{B} & \rightarrow & \mathbf{x}_{k+1}^{\mathrm{H}} \\ \mathbf{C} & \rightarrow & -1 \end{array}$$

then we find that

$$\begin{aligned} \mathbf{\Phi}_{k+1}^{-1} &= (\mathbf{\Phi}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}})^{-1} \\ &= \mathbf{\Phi}_k^{-1} - \frac{\mathbf{\Phi}_k^{-1}\mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{\Phi}_k^{-1}}{1 + \mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{\Phi}_k^{-1}\mathbf{x}_{k+1}} \end{aligned}$$

If we define a matrix $\mathbf{P}_k = \mathbf{\Phi}_k^{-1}$, then we can simply update this inverse factor. The result is the *Recursive Least Squares* (RLS) algorithm, (with infinite horizon, as we discuss later):

$$\begin{aligned} \mathbf{P}_{k+1} &:= \mathbf{P}_k - \frac{\mathbf{P}_k\mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{P}_k}{1 + \mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{P}_k\mathbf{x}_{k+1}} \\ \boldsymbol{\theta}_{k+1} &:= \boldsymbol{\theta}_k + \mathbf{x}_{k+1}\bar{s}_{k+1} \\ \hat{\mathbf{w}}_{k+1} &:= \mathbf{P}_{k+1}\boldsymbol{\theta}_{k+1}\,. \end{aligned}$$

The algorithm is usually initialized with $\mathbf{P}_0 = \delta^{-1}\mathbf{I}$, where δ is a very small positive constant, and with $\boldsymbol{\theta}_0 = \mathbf{0}$. As usual, the filter output is given by

$$y_{k+1} := \hat{\mathbf{w}}_{k+1}^{\mathrm{H}}\mathbf{x}_{k+1}.$$

The RLS algorithm provides at any time the best possible estimate of the Wiener receiver, based on the available data up to that point. The number of operations needed to update the weight vector is (if we compute $\mathbf{P}_k\mathbf{x}_{k+1}$ once) about $2M^2 + 3M$ multiplications. Compare this to a direct inversion of $\mathbf{\Phi}_k$, which is an order M^3 operation. Thus, RLS is an order M more efficient than a direct matrix inversion, but an order M slower than the LMS algorithm. (There are several faster or numerically more attractive implementations of RLS, in particular for equalization purposes [1].)

4.6.3 Finite horizon

The preceding RLS algorithm has infinite memory. Indeed, $\mathbf{\Phi}_k$ contains all preceding samples until time 0, equally weighted. For adaptive purposes in which the data is non-stationary, this is not desirable, in this case we want an effective window of data. There are two popular techniques for this:

1. *Sliding window:* we base $\mathbf{\Phi}_k$ and $\boldsymbol{\theta}_k$ on only the last n samples. This involves an update rule
$$\begin{aligned}\mathbf{\Phi}_{k+1} &= \mathbf{\Phi}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}} - \mathbf{x}_{k-n}\mathbf{x}_{k-n}^{\mathrm{H}},\\ \boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k + \mathbf{x}_{k+1}\bar{s}_{k+1} - \mathbf{x}_{k-n}\bar{s}_{k-n}^{\mathrm{H}}.\end{aligned}$$
It is clear that the *downdate* term $\mathbf{x}_{k-n}\mathbf{x}_{k-n}^{\mathrm{H}}$ can be treated just as the update term in deriving the update rule for $\mathbf{P}_{k+1}$. Since the complexity is doubled and we have to keep n previous data samples in memory, this technique is not used very often.

2. *Exponential window*: here the downdate is performed by simply scaling down $\mathbf{\Phi}_k$ and $\boldsymbol{\theta}_k$ by a factor λ close to but smaller than 1:
$$\begin{aligned}\mathbf{\Phi}_{k+1} &= \lambda\mathbf{\Phi}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}},\\ \boldsymbol{\theta}_{k+1} &= \lambda\boldsymbol{\theta}_k + \mathbf{x}_{k+1}\bar{s}_{k+1}.\end{aligned}$$
It is as if we have scaled the data $\mathbf{x}_k$ and s_k progressively, as if we would have used
$$\begin{aligned}\mathbf{X}_{k+1} &= [\mathbf{x}_{k+1} \quad \lambda^{1/2}\mathbf{x}_k \quad \lambda\mathbf{x}_{k-1} \quad \lambda^{3/2}\mathbf{x}_{k-2} \quad \cdots],\\ \mathbf{s}_{k+1} &= [s_{k+1} \quad \lambda^{1/2}s_k \quad \lambda s_{k-1} \quad \lambda^{3/2}s_{k-2} \quad \cdots].\end{aligned}$$
Thus, data is never quite forgotten but becomes less and less important. Typical values for λ are in the range 0.95 – 0.999.

If we work out the updating rules, we obtain the *RLS algorithm with finite horizon*:
$$\begin{aligned}\mathbf{P}_{k+1} &:= \lambda^{-1}\mathbf{P}_k - \lambda^{-2}\frac{\mathbf{P}_k\mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{P}_k}{1+\lambda^{-1}\mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{P}_k\mathbf{x}_{k+1}}\\ \boldsymbol{\theta}_{k+1} &:= \lambda\boldsymbol{\theta}_k + \mathbf{x}_{k+1}\bar{s}_{k+1}\\ \hat{\mathbf{w}}_{k+1} &:= \mathbf{P}_{k+1}\boldsymbol{\theta}_{k+1}\,.\end{aligned}$$

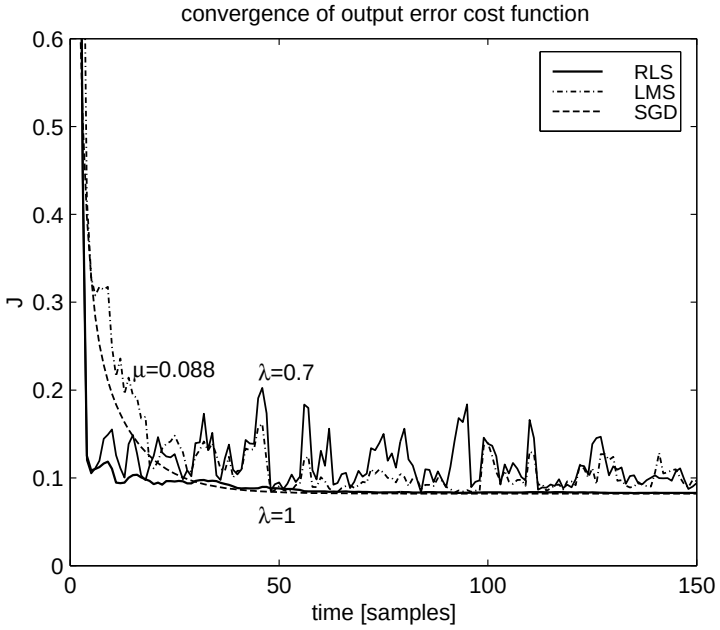


Figure 4.5. Convergence of RLS

4.6.4 Comparison of RLS to LMS

With exponential windowing, one often defines the *effective window length* as the value for n such that

$$\lambda^n = \frac{1}{\mathrm{e}} \qquad \Rightarrow \qquad n \;=\; \frac{-1}{\ln \lambda} \;\approx\; \frac{1}{1-\lambda}\,.$$

Thus, $1-\lambda$ plays an analogous role as $\mu\lambda_{\min}$ in the LMS algorithm.

A comparison is shown in figure 4.5. It is based on the same data as earlier in figure 4.3(b). For the case $\lambda = 1$, we find that the algorithm converges in 4 steps, and stays flat after that. For $\lambda = 0.7$ (this is an impractically small value that gives a memory length of only 3.3 samples), the algorithm converges just as fast but has a large excess error. The excess error is comparable to the LMS algorithm with $\mu = (1-\lambda)/\mathrm{E}(\|\mathbf{x}\|^2) = 0.088$.

Further comparisons give rise to the following conclusions [1]:

1. RLS converges faster. One shows that the algorithm converges in about $2M$ iterations. It is relatively insensitive to the eigenvalue spread of $\mathbf{R}_x$, and has a negligible misadjustment (zero for a stationary environment without disturbances, when $\lambda = 1$).
2. As LMS, RLS converges in the mean: $\mathrm{E}(\mathbf{w}_k) = \mathbf{w}_0$.
3. If $\lambda = 1$, then the RLS achieves asymptotically the Wiener solution, hence it has asymptotically zero excess mean-squared error.

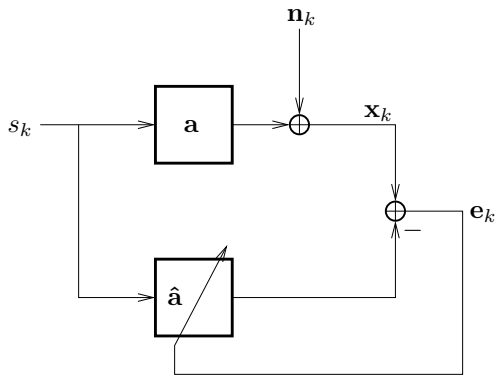


Figure 4.6. Adaptive model matching

4. LMS has a superior performance in tracking *non-stationary* environments, for the same level of misadjustment.

4.7 EXAMPLES

4.7.1 Model matching

In the previous sections, we have looked at output error models: the objective was to construct a beamformer that recovers the signal and minimizes the output error. It is however also possible to use the same type of algorithms on *model error* problems, as sketched in figure 4.6. The objective here is to estimate the channel rather than the receiver, and to adapt the estimate until the model error is minimal. It is thus a form of channel identification.

In equations: we consider a single source in noise (plus interference),

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k \,.$$

The objective is to find an estimate $\hat{\mathbf{a}}$ such that the model error

$$\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{a}}s_k$$

is minimized. Note that here the model error is a vector. The corresponding cost function is

$$J(\hat{\mathbf{a}}) \;=\; \mathrm{E}(\|\mathbf{e}_k\|^2) \;=\; \mathrm{E}(\|\mathbf{x}_k - \hat{\mathbf{a}}s_k\|^2)\,.$$

This can be worked out as

$$\begin{aligned} J(\hat{\mathbf{a}}) &= \mathrm{E}([\mathbf{x}_k - \hat{\mathbf{a}}s_k]^{\mathrm{H}}[\mathbf{x}_k - \hat{\mathbf{a}}s_k]) \\ &= \mathbf{R}_x - \hat{\mathbf{a}}^{\mathrm{H}}\mathbf{r}_{xs} - \mathbf{r}_{xs}\hat{\mathbf{a}} + \hat{\mathbf{a}}^{\mathrm{H}} r_s \hat{\mathbf{a}}, \quad r_s = \mathrm{E}(|s_k|^2), \end{aligned}$$

and the gradient is

$$\nabla J(\hat{\mathbf{a}}) = r_s\hat{\mathbf{a}} - \mathbf{r}_{xs} = r_s(\hat{\mathbf{a}} - \mathbf{a})\,,$$

where the latter equality is due to $\mathrm{E}(\mathbf{x}_k \bar{s}_k) = r_s \mathbf{a}$.

The steepest gradient algorithm in this context becomes

$$\begin{aligned} \hat{\mathbf{a}}^{(k+1)} &= \hat{\mathbf{a}}^{(k)} - \mu \nabla J(\hat{\mathbf{a}}^{(k)}) \\ &= \hat{\mathbf{a}}^{(k)} - \mu r_s (\hat{\mathbf{a}}^{(k)} - \mathbf{a}) \\ &= \mu r_s \mathbf{a} + (1 - \mu r_s) \hat{\mathbf{a}}^{(k)} . \end{aligned}$$

It is straightforward to check the convergence of this recursion. Indeed, we can compare to the similar recursion that occurs in systems theory,

$$\begin{aligned} x_k &= b + a x_{k-1} \\ &= b + ba + a^2 x_{k-2} \\ &= b + ba + \cdots + b a^{k-1} + a^k x_0 \\ &= b \frac{1 - a^k}{1 - a} + a^k x_0 . \end{aligned}$$

In our case, we obtain similarly

$$\begin{aligned} \hat{\mathbf{a}}^{(k)} &= \mathbf{a} \mu r_s \frac{1 - (1 - \mu r_s)^k}{1 - (1 - \mu r_s)} + (1 - \mu r_s)^k \hat{\mathbf{a}}^{(0)} \\ &= \mathbf{a}[1 - (1 - \mu r_s)^k] + (1 - \mu r_s)^k \hat{\mathbf{a}}^{(0)} . \end{aligned}$$

Thus, if $|1 - \mu r_s| < 1$, i.e., $0 < \mu < \frac{2}{r_s}$, then the algorithm converges to $\mathbf{a}$, independent of the initial condition.

The LMS algorithm is derived from the steepest gradient algorithm by replacing the estimates by instantaneous values. This gives

$$\begin{aligned} \hat{\mathbf{a}}^{(k+1)} &= \hat{\mathbf{a}}^{(k)} - \mu(\hat{\mathbf{a}}^{(k)} s_k \bar{s}_k - \mathbf{x}_k \bar{s}_k) \\ &= \hat{\mathbf{a}}^{(k)} + \mu \mathbf{e}_k \bar{s}_k . \end{aligned} \tag{4.4}$$

The similarity to the LMS algorithm of the previous sections is obvious. We can conclude immediately that the algorithm converges if $0 < \mu < \frac{2}{r_s}$, and that $\mathrm{E}(\hat{\mathbf{a}}^{(k)}) = \mathbf{a}$. It is also possible to extend the algorithm for the case with multiple (known) signals, although the derivation becomes more involved since the gradient has to be defined with respect to an unknown *matrix* $\mathbf{A}$.

Interference cancellation in radio astronomy A practical example of the previous scenario is the situation in radio astronomy. In Westerbork (NL), 14 telescope dishes are pointing in the sky towards astronomical sources of interest. However, local interfering signals from GSM telephones, television, airplane communication satellites, etc, enter into the dishes via the side lobes and completely mask the sky sources. A current proposal is to put up a reference antenna with an omni-directional response, which would mainly capture the interfering signal and have almost no gain towards the astronomical sources. By correlating the reference antenna signal

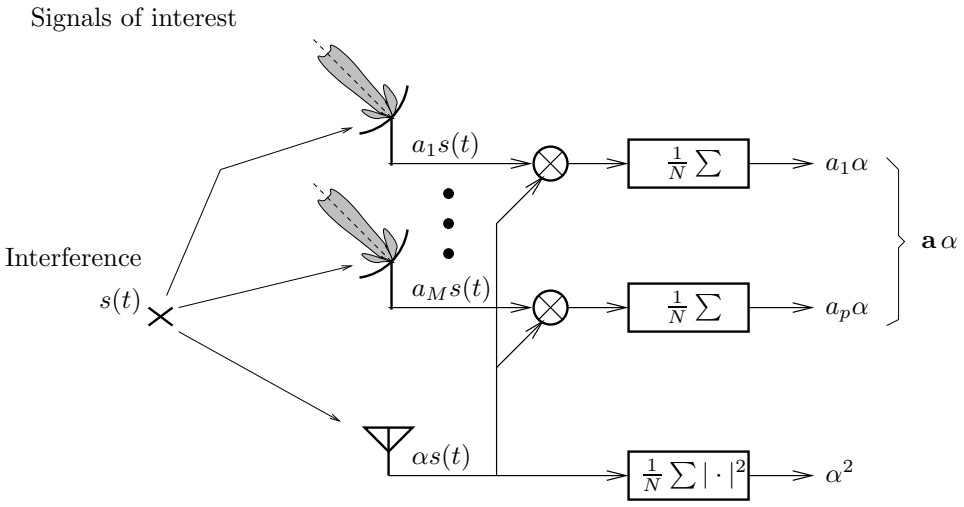


Figure 4.7. Interference cancellation in radio astronomy using a reference antenna

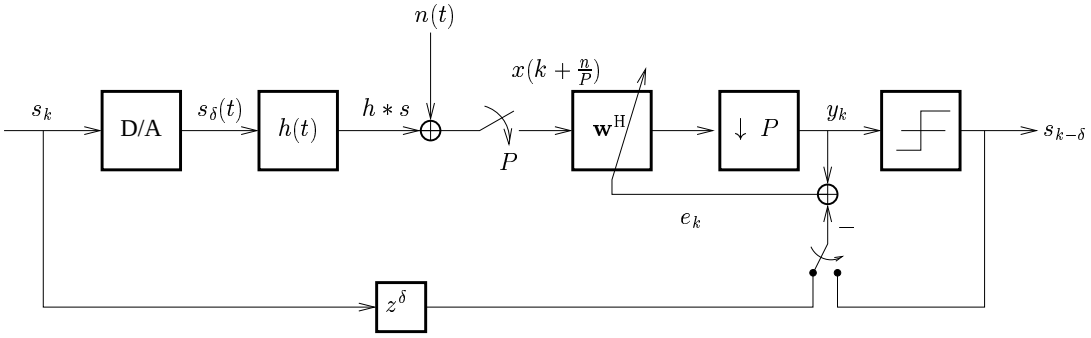


Figure 4.8. Adaptive filter

with the telescope signals, we can obtain an estimate of the array response $\mathbf{a}$ of the interfering signal at the main telescopes. The correlation can be done directly by acting on blocks of N data samples (as shown in figure 4.7), or adaptively using the LMS algorithm in (4.4). After $\mathbf{a}$ has been obtained, there are various techniques to cancel the interference, e.g., by subtracting $\hat{\mathbf{a}}\hat{s}(t)$, or by projecting out this dimension via a projection $\mathbf{P}_{\hat{\mathbf{a}}} = \mathbf{I} - \hat{\mathbf{a}}(\hat{\mathbf{a}}^{\mathrm{H}}\hat{\mathbf{a}})^{-1}\hat{\mathbf{a}}^{\mathrm{H}}$ acting on $\mathbf{x}_k$.

4.7.2 Adaptive equalization

Until now we have considered adaptive filtering in the spatial domain, i.e., adaptive beamforming. However, the same theory applies to adaptive filtering in the time domain as well. In fact, the theory of adaptive temporal filtering is more rich because the model has more structure, and this can be exploited to derive more efficient algorithms.

To illustrate this, consider the scenario depicted in figure 4.8. The source s_k (a symbol sequence) is transformed to the pulse sequence $s_\delta(t)$ in the analog domain. This sequence is then convolved with the channel $h(t)$, which we assume to be an FIR channel with length at most L symbols. Noise $n(t)$ is added, and the result is sampled at a rate P times faster than the symbol rate. We normalize the symbol period to $T = 1$. The data model is thus (cf. chapter 1, equation (1.15))

$$\begin{aligned}
\mathcal{X}_m &= \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{N-1} \\ \mathbf{x}_{-1} & \mathbf{x}_0 & \cdots & \mathbf{x}_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{-m+1} & \mathbf{x}_{-m+2} & \cdots & \mathbf{x}_{N-m} \end{bmatrix} \\
&= \begin{bmatrix} \boxed{\mathbf{H}} & & \mathbf{0} \\ & \boxed{\mathbf{H}} & \\ & \ddots & \\ \mathbf{0} & & \boxed{\mathbf{H}} \end{bmatrix} \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_{N-1} \\ \mathbf{s}_{-1} & \mathbf{s}_0 & \cdots & \mathbf{s}_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_{-L-m+1} & \mathbf{s}_{-L-m+2} & \cdots & \mathbf{s}_{N-L-m} \end{bmatrix} = \mathcal{H}_m \mathcal{S}_{L+m-1}
\end{aligned}$$

With noise, we obtain $\mathcal{X}_m = \mathcal{H}_m \mathcal{S}_{L+m-1} + \mathcal{N}_m$. Note that the model $\mathcal{X}_m = \mathcal{H}_m \mathcal{S}_{L+m-1} + \mathcal{N}_m$ is of the same form as $\mathbf{X} = \mathbf{AS} + \mathbf{N}$. The difference is that each row of $\mathcal{S}$ is a shift of another: they are not independent, and the model has more structure than before.

A linear equalizer can be written as a vector $\mathbf{w}$ which combines the rows of $\mathcal{X}_m$ to generate an output $\mathbf{y} = \mathbf{w}^{\mathrm{H}} \mathcal{X}_m$. Note that we equalize among m symbol periods, with P samples per symbol (polyphase combining).

A *necessary* condition for equalization (the output $\mathbf{y}$ is equal to a row of $\mathcal{S}$) is that $\mathcal{H}$ is tall, which gives conditions on m in terms of P and L:

$$mP \geq L + m - 1 \qquad \Rightarrow \qquad m(P-1) \geq L - 1$$

which implies

$$P > 1\,, \qquad m \geq \frac{(L-1)}{P-1}$$

With $P = 2$, a customary value, we obtain $m \geq L - 1$. Ideally, without noise, we can reconstruct any row of $\mathcal{S}$: each row is a valid symbol sequence. There are $L + m - 1$ rows, hence $L + m - 1$ valid equalizers. Equivalently, we would like to find $\mathbf{w}$ such that

$$\mathbf{w}^{\mathrm{H}} \mathcal{H} = [1,\ 0,\ 0,\ \cdots,\ 0] \quad \text{or} \quad [0,\ 1,\ 0,\ \cdots,\ 0] \quad \text{or} \quad [0,\ 0,\ 1,\ \cdots,\ 0] \quad \text{or} \quad \cdots$$

Each equalizer $\mathbf{w}^{\mathrm{H}}$ is another row of the left inverse $\mathcal{H}^\dagger$ of $\mathcal{H}$. These are the zero-forcing equalizers.

The symbol sequences that we can recover are all shifts of each other. Thus, we have to specify at which delay δ we would like the symbol sequence to match the original sequence. Usually, we specify the 'center tap': $\delta = \frac{1}{2}(L + m - 1)$. With $P = 2$ and $m = L - 1$, we obtain $\delta = L - 1$.

Without noise, all equalizers are equivalent. With noise, however, it is not guaranteed that the center tap delay will result in the best equalizer: this depends on the channel. Equalizers have

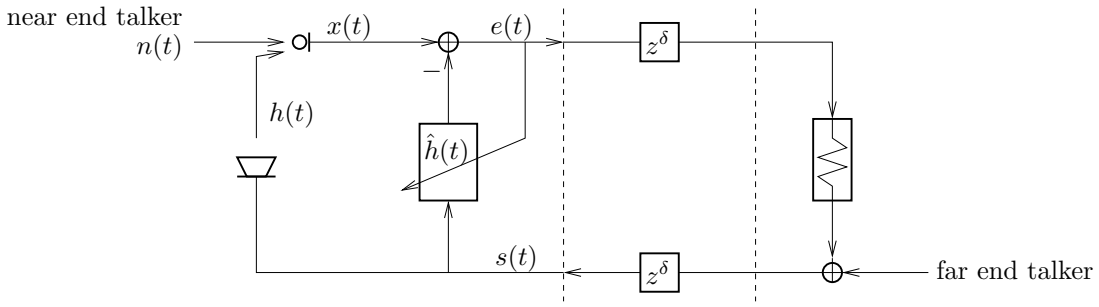


Figure 4.9. Echo cancellation

different norms $\|\mathbf{w}\|$, and we prefer the equalizer that gives the smallest noise amplification, thus has the smallest norm. Without estimating all equalizers, there is no way to know which one is best.

In the presence of noise, we would also prefer to compute the Wiener equalizers rather than the zero-forcing equalizers, i.e., columns of $\mathbf{R}_{\mathcal{X}}^{-1}\mathbf{R}_{\mathcal{XS}}$. The i-th column of $\mathbf{R}_{\mathcal{X}}^{-1}\mathbf{R}_{\mathcal{XS}}$ is a Wiener equalizer reconstructing the signal at a delay i.

For an adaptive filter, we have to select a reference signal. This is equal to the original signal, after introducing the preferred delay δ. At this point, we have a data model $\mathcal{X} = \mathcal{HS}$, with a reference signal (one row of $\mathcal{S}$), and we can apply the same LMS, NLMS and RLS algorithms as in the previous sections.

In practical systems, the reference signal is available for only a limited period (the training period). After this period, the equalizer is assumed to have converged. At this point, we can use the output of the equalizer to estimate the original source symbols, by comparing it to the closest matching symbol. This is a quantization operation. E.g., if the source alphabet is simply $\{-1, +1\}$, BPSK, then we simply look at the sign of y_k and decide if it is negative or positive. The idea behind *decision-directed* adaptive equalization is that if the noise is small, the decisions are likely to be correct, and in that case we can generate the noise-free reference signal from the output of the equalizer, without needing the original source. Decision-directed equalizers are quite successful and often applied in telephone modems.

Echo cancellation A related example in which model matching in the temporal domain plays a role is that of *echo cancellation*. In this case, (figure 4.9), we have a telephone system with two speakers, one on the near-end and one on the far-end. On both sides, the system is not perfect in the sense that an incoming signal $s(t)$ from the far-end is put on the loud-speaker, received again by the microphone, put back into the channel towards the far-end talker and looped back to the near-end again. (This problem occurs in particular in hands-free telephony. An important part of the feedback loop in conventional telephony is formed by imperfect hybrids at either side of the telephone lines.) The signal is perceived as an echo.

To mitigate this feedback loop, an adaptive echo canceller is introduced. This system consists of an adaptive channel estimator, of the same principle as in section 4.7.1, but now in the temporal domain. The objective is to estimate $h(t)$ and to subtract $h(t) * s(t)$, such as to minimize the error signal $e(t)$, which ideally only consists of the near-end signal $n(t)$.

Bibliography

[1] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs (NJ): Prentice-Hall, 1992.

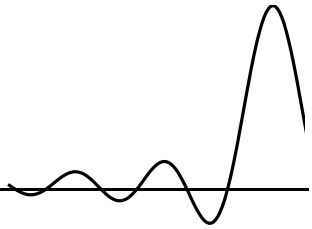
[2] B. Widrow and S. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1985.

[3] C. Johnson, "On the interaction of adaptive filtering, identification, and control," *IEEE Signal Processing Mag.*, vol. 12, pp. 22–37, Mar. 1995.

[4] "Spec. issue on adaptive algorithms and echo cancellation," *IEEE Signal Processing Mag.*, vol. 16, Apr. 1999.

Chapter 5

THE ESPRIT ALGORITHM



Contents

In chapter 3, we have looked at the MVDR and MUSIC algorithms for direction finding. It was seen that MUSIC provides high-resolution estimates for the directions-of-arrival (DOAs). However, these algorithms need a search over the parameter α, and extensive calibration data (i.e., the function $\mathbf{a}(\alpha)$ for a finely sampled range of α). In this present chapter, we look at the ESPRIT algorithm for direction estimation. This algorithm does not require a search or calibration data, but assumes a special array configuration that allows to solve for the DOAs algebraically, by solving an eigenvalue problem. The same algorithm applies to delay estimation and to frequency estimation.

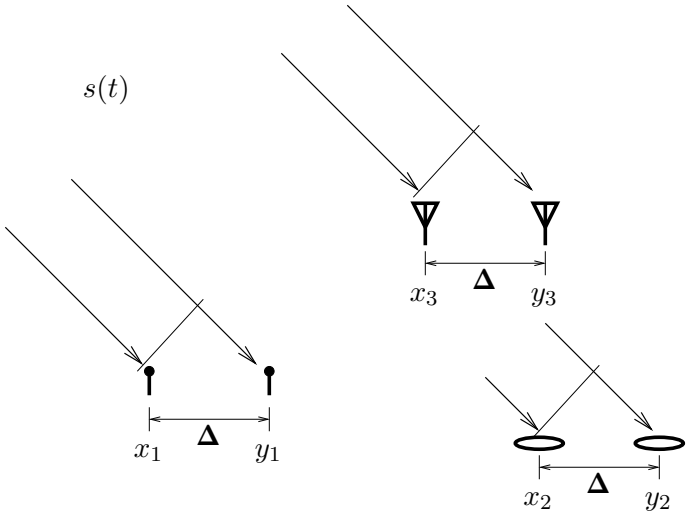
5.1 DIRECTION ESTIMATION USING THE ESPRIT ALGORITHM

As in previous chapters, we assume that all signals are narrowband with respect to the propagation delay across the array, so that this delay translates to a phase shift. We consider a simple propagation scenario, in which there is no multipath and sources have only one ray towards the receiving antenna array. Since no delays are involved, all measurements are simply instantaneous linear combinations of the source signals. Each source has only one ray, so that the data model is

$$\mathbf{X} = \mathbf{AS}\,.$$

$\mathbf{A} = [\mathbf{a}(\alpha_1), \cdots, \mathbf{a}(\alpha_d)]$ contains the array response vectors. The rows of $\mathbf{S}$ contain the signals, multiplied by the fading parameters (amplitude scalings and phase rotations).

Computationally attractive ways to compute $\{\alpha_i\}$ and hence $\mathbf{A}$ are possible for certain regular antenna array configurations for which $\mathbf{a}(\alpha)$ becomes a *shift-invariant* or similar recursive structure. This is the basis for the ESPRIT algorithm (Roy, Kailath and Paulraj 1987 [1]).



5.1.1 Array geometry

The constraint on the array geometry imposed by ESPRIT is that of *sensor doublets*: the array consists of two subarrays, denoted by

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_M(t) \end{bmatrix}, \qquad \mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_M(t) \end{bmatrix}$$

where each sensor y_i has an identical response as x_i, and is spaced at a constant displacement vector $\mathbf{\Delta}$ (wavelengths) from x_i. It is important that the displacement vector is the same for all sensor pairs (both in length and in direction). The antenna response $a_i(\alpha)$ for the pair (x_i, y_i) is arbitrary and may be different for other pairs.

For the pair $(x_i(t), y_i(t))$, we have the model

$$\begin{aligned} x_i(t) &= \sum_{k=1}^{d} a_i(\alpha_k) s_k(t) \\ y_i(t) &= \sum_{k=1}^{d} a_i(\alpha_k) e^{j2\pi\Delta\sin(\alpha_k)} s_k(t) \;=\; \sum_{k=1}^{d} a_i(\alpha_k)\theta_k s_k(t) \end{aligned}$$

where $\theta_k = e^{j2\pi\Delta\sin(\alpha_k)}$ is the phase rotation due to the propagation of the signal from the x-antenna to the corresponding y-antenna.

In terms of the vectors $\mathbf{x}$ and $\mathbf{y}$, we have

$$\begin{aligned} \mathbf{x}(t) &= \sum_{k=1}^{d} \mathbf{a}(\alpha_k) s_k(t) \\ \mathbf{y}(t) &= \sum_{k=1}^{d} \mathbf{a}(\alpha_k) \theta_k s_k(t) \end{aligned} \tag{5.1}$$

The ESPRIT algorithm does not assume any structure on $\mathbf{a}(\alpha)$. It will instead use the phase relation between $\mathbf{x}(t)$ and $\mathbf{y}(t)$.

If we collect N samples in matrices $\mathbf{X}$ and $\mathbf{Y}$, we obtain the data model

$$\begin{aligned} \mathbf{X} &= \mathbf{AS} \\ \mathbf{Y} &= \mathbf{A\Theta S} \end{aligned} \tag{5.2}$$

where

$$\mathbf{A} = \left[\mathbf{a}(\alpha_1) \quad \cdots \quad \mathbf{a}(\alpha_d) \right], \qquad \mathbf{\Theta} = \begin{bmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_d \end{bmatrix}, \qquad \theta_i = e^{j2\pi\Delta \sin(\alpha_i)}$$

One special case in which the shift-invariant structure occurs is that of a uniform linear array (ULA) with $M+1$ antennas. For such an array, with interelement spacing Δ wavelengths, we have seen that

$$\mathbf{a}(\theta) = \begin{bmatrix} 1 \\ \theta \\ \vdots \\ \theta^M \end{bmatrix}, \quad \theta = e^{j2\pi\Delta \sin(\alpha)} \tag{5.3}$$

If we now split the array into two overlapping subarrays, the first ($\mathbf{x}$) containing antennas 1 to M, and the second ($\mathbf{y}$) antennas 2 to $M+1$, we obtain

$$\mathbf{a}_x = \begin{bmatrix} 1 \\ \theta \\ \vdots \\ \theta^{M-1} \end{bmatrix}, \qquad \mathbf{a}_y = \begin{bmatrix} \theta \\ \theta^2 \\ \vdots \\ \theta^{M} \end{bmatrix} = \begin{bmatrix} 1 \\ \theta \\ \vdots \\ \theta^{M-1} \end{bmatrix} \theta$$

which gives precisely the model (5.1), where $\mathbf{a}$ in (5.1) is one entry shorter than in (5.3).

5.1.2 Algorithm

Given the data $\mathbf{X}$ and $\mathbf{Y}$, we first stack all data in a single matrix $\mathbf{Z}$ of size $2M \times N$ with model

$$\mathbf{Z} = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix} = \mathbf{A}_z \mathbf{S}, \qquad \mathbf{A}_z = \begin{bmatrix} \mathbf{A} \\ \mathbf{A\Theta} \end{bmatrix}$$

(In the case of a ULA with $M+1$ antennas, we stack the available antenna outputs vertically but do not duplicate the antennas; $\mathbf{Z}$ will then have size $M+1 \times N$). Since $\mathbf{Z}$ has rank d, we compute an (economy-size) SVD

$$\mathbf{Z} = \hat{\mathbf{U}}_z \hat{\mathbf{\Sigma}}_z \hat{\mathbf{V}}_z^{\mathrm{H}}, \tag{5.4}$$

where $\hat{\mathbf{U}}_z : 2M \times d$ has d columns which together span the column space of $\mathbf{Z}$. The same space is spanned by the columns of $\mathbf{A}_z$, so that there must exist a $d \times d$ invertible matrix $\mathbf{T}$ that maps one basis into the other, i.e., such that

$$\hat{\mathbf{U}}_z = \mathbf{A}_z \mathbf{T} = \begin{bmatrix} \mathbf{AT} \\ \mathbf{A\Theta T} \end{bmatrix} \tag{5.5}$$

If we now split $\hat{\mathbf{U}}_z$ into two $M \times d$ matrices in the same way as $\mathbf{Z}$,

$$\hat{\mathbf{U}}_z = \begin{bmatrix} \hat{\mathbf{U}}_x \\ \hat{\mathbf{U}}_y \end{bmatrix}$$

then we obtain that

$$\begin{cases} \hat{\mathbf{U}}_x &= \mathbf{AT} \\ \hat{\mathbf{U}}_y &= \mathbf{A\Theta T} \end{cases}$$

For $M \geq d$, $\hat{\mathbf{U}}_x$ is "tall", and if we assume that $\mathbf{A}$ has full column rank, then $\hat{\mathbf{U}}_x$ has a left-inverse

$$\hat{\mathbf{U}}_x^{\dagger} := (\hat{\mathbf{U}}_x^{\mathrm{H}} \hat{\mathbf{U}}_x)^{-1} \hat{\mathbf{U}}_x^{\mathrm{H}}$$

It is straightforward to verify that

$$\hat{\mathbf{U}}_x^{\dagger} = (\mathbf{T}^{\mathrm{H}} \mathbf{A}^{\mathrm{H}} \mathbf{A} \mathbf{T})^{-1} \mathbf{T}^{\mathrm{H}} \mathbf{A}^{\mathrm{H}} = \mathbf{T}^{-1} \mathbf{A}^{\dagger}$$

so that

$$\hat{\mathbf{U}}_x^{\dagger} \hat{\mathbf{U}}_y = \mathbf{T}^{-1} \mathbf{\Theta} \mathbf{T}.$$

The matrix on the left hand side is known from the data. Since $\mathbf{\Theta}$ is a diagonal matrix, the matrix product on the right hand side is recognized as an eigenvalue equation: $\mathbf{T}^{-1}$ contains the eigenvectors of $\hat{\mathbf{U}}_x^{\dagger} \hat{\mathbf{U}}_y$ (scaled arbitrarily to unit norm), and the entries of $\mathbf{\Theta}$ on the diagonal are the eigenvalues. Hence we can simply compute the eigenvalue decomposition of $\hat{\mathbf{U}}_x^{\dagger} \hat{\mathbf{U}}_y$, take the eigenvalues $\{\theta_i\}$ (they should be on the unit circle), and compute the DOAs α_i from each of them. This comprises the ESPRIT algorithm.

Note that the SVD of $\mathbf{Z}$ in (5.4) along with the definition of $\mathbf{T}$ in (5.5) as $\hat{\mathbf{U}}_z = \mathbf{A}_z \mathbf{T}$ implies that

$$\begin{aligned} & \mathbf{Z} = \hat{\mathbf{U}}_z \hat{\mathbf{\Sigma}}_z \hat{\mathbf{V}}_z^{\mathrm{H}}, && \mathbf{Z} = \mathbf{A}_z \mathbf{S} = \mathbf{A}_z \mathbf{T} \mathbf{T}^{-1} \mathbf{S} \\ \Rightarrow \quad & \mathbf{T}^{-1} \mathbf{S} = \hat{\mathbf{\Sigma}}_z \hat{\mathbf{V}}_z^{\mathrm{H}} = \hat{\mathbf{U}}_z^{\mathrm{H}} \mathbf{Z} \\ \Rightarrow \quad & \mathbf{S} = \mathbf{T} \hat{\mathbf{U}}_z^{\mathrm{H}} \mathbf{Z} \end{aligned}$$

Hence, after having obtained $\mathbf{T}$ from the eigenvectors, a zero-forcing beamformer on $\mathbf{Z}$ is given by

$$\mathbf{W} \;=\; \hat{\mathbf{U}}_z \mathbf{T}^{\mathrm{H}}$$

Thus, source separation is straightforward in this case and essentially reduced to an SVD and an eigenvalue problem.

If the two subarrays are spaced by at most half a wavelength, then the DOAs are directly recovered from the diagonal entries of $\mathbf{\Theta}$, otherwise they are ambiguous (two different values of α give the same θ). Such an ambiguity does not prevent the construction of the beamformer $\mathbf{W}$ from $\mathbf{T}$, and source separation is possible. Because the rows of $\mathbf{T}$ are determined only up to a scaling, the correct scaling of the rows of $\mathbf{S}$ cannot be recovered unless we know the average power of each signal or the array manifold $\mathbf{A}$. This is of course inherent in the problem definition.

With noise, essentially the same algorithm is used. If we assume that the number of sources d is known, then we compute the SVD of the noisy $\mathbf{Z}$, and set $\hat{\mathbf{U}}_Z$ equal to the principal d left singular vectors. This is the best estimate of the subspace spanned by the columns of $\mathbf{A}$, and asymptotically (infinite samples) identical to it. Thus, for infinitely many samples we obtain the correct directions: the algorithm is asymptotically unbiased (*consistent*). For finite samples, an estimated eigenvalue $\hat{\theta}$ will not be on the unit circle, but we can easily map it to the unit circle by dividing by $|\hat{\theta}|$.

5.1.3 Extension for a ULA

There are many important refinements and extensions to this algorithm. If we have a uniform linear array, we can use the fact that the solutions θ should be on the unit circle, i.e.,

$$\bar{\theta} = \theta^{-1}$$

along with the structure of $\mathbf{a}(\theta)$ in (5.3):

$$\mathbf{a}(\theta) = \begin{bmatrix} 1 \\ \theta \\ \vdots \\ \theta^M \end{bmatrix} \quad \Rightarrow \quad \mathbf{\Pi}\bar{\mathbf{a}}(\theta) =: \begin{bmatrix} & & & 1 \\ & & 1 & \\ & \iddots & & \\ 1 & & & \end{bmatrix} \begin{bmatrix} 1 \\ \bar{\theta} \\ \vdots \\ \bar{\theta}^M \end{bmatrix} = \begin{bmatrix} \bar{\theta}^M \\ \bar{\theta}^{M-1} \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ \theta \\ \vdots \\ \theta^M \end{bmatrix} \theta^{-M} = \mathbf{a}(\theta)\theta^{-M}$$

Thus, if we construct an extended data matrix

$$\mathbf{Z}_e \;=\; [\mathbf{Z}\,, \quad \mathbf{\Pi}\bar{\mathbf{X}}]$$

then this will double the number of observations but will not increase the rank, since

$$\mathbf{Z}_e \;=\; \mathbf{A}_z[\mathbf{S}\,, \quad \mathbf{\Theta}^{-1}\mathbf{S}]$$

Using this structure, it is also possible to transform $\mathbf{Z}_e$ to a real-valued matrix, by simple linear operations on its rows and columns [2, 3]. As we saw in chapter 3, there are many other direction

finding algorithms that are applicable. For the case of a ULA in fact a better algorithm is known to be MODE [4]. Although ESPRIT is statistically suboptimal, its performance is usually quite adequate. Its interest lies also in its straightforward generalization to more complicated estimation problems in which shift-invariance structure is present.

5.1.4 Performance

Figure 5.1 shows the results of a simulation with 2 sources with directions $-10°$, $10°$, a ULA($\frac{\lambda}{2}$) with 6 antennas, and $N = 40$ samples. The first graph shows the mean value, the second the standard deviation (averaged over the two sources), which indicates the accuracy of an individual estimate. For sufficient SNR, the performance of both algorithms is approximately the same.

Figure 5.2 shows the same for varying separation of the two sources, with an SNR of 10 dB. For small separation, the performance of ESPRIT drops because the matrix $\mathbf{A}$ drops in rank: it appears to have only 1 independent column rather than 2. If we select two singular vectors, then this subspace will not be shift-invariant, and the algorithm produces bad estimates: both the mean value and the standard deviation explode. MUSIC, on the other hand, selects the null space and scans for vectors orthogonal to it. If we ask for 2 vectors, it will in this case produce two times the same vector since there is only a single maximum in the MUSIC spectrum. It is seen that the estimates become biased towards a direction centered between the two sources ($= 0°$), but that the standard deviation gets smaller since the algorithm consistenly picks this center.

The performance of both ESPRIT and MUSIC is noise limited: without noise, the correct DOAs are obtained. With noise and asymptotically many samples, $N \to \infty$, the correct DOAs are obtained as well, since the subspace spanned by $\hat{\mathbf{U}}_z$ is asymptotically identical to that obtained in the noise-free case, the span of the columns of $\mathbf{A}$.

5.2 DELAY ESTIMATION USING ESPRIT

A channel matrix $\mathbf{H}$ can be estimated from training sequences, or sometimes "blindly" (without training). Very often, we do not need to know the details of $\mathbf{H}$ if our only purpose is to recover the signal matrix $\mathbf{S}$. But there are several situations as well where it is interesting to pose a multipath propagation model, and try to resolve the individual propagation paths. This would give information on the available delay and angle spread, for the purpose of diversity. It is often assumed that the directions and delays of the paths do not change quickly, only their powers (fading parameters), so that it makes sense to estimate these parameters. If the channel is well-characterized by this parametrized model, then fitting the channel estimate to this model will lead to a more accurate receiver. Another application would be mobile localization for emergency services.

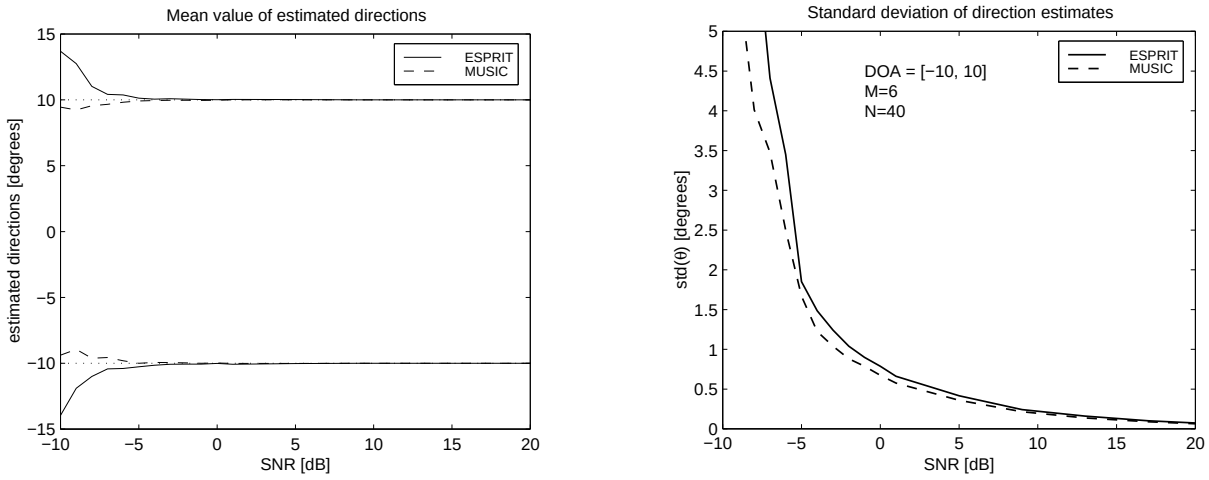


Figure 5.1. Mean and standard deviations of ESPRIT and MUSIC estimates as function of SNR

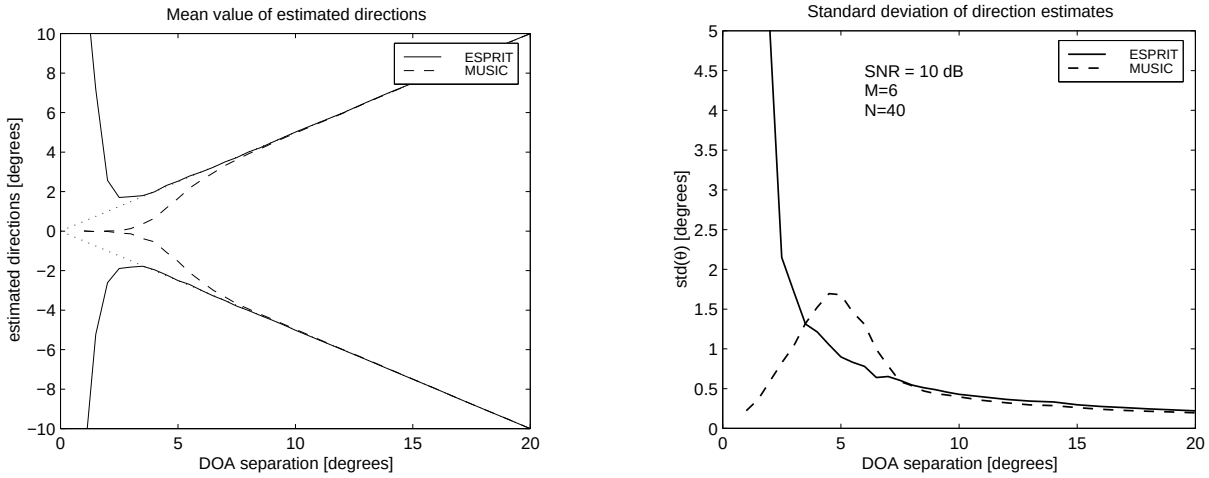


Figure 5.2. Mean and standard deviations of ESPRIT and MUSIC estimates as function of DOA separation

5.2.1 Principle

Let us consider first the simple case already introduced in section 3.7. Assume we have a vector $\mathbf{g}_0$ corresponding to samples of an FIR pulse shape function $g(t)$, nonzero only in a finite interval $[0, L_g)$, sampled at a rate P times the symbol rate:

$$g(t) \quad \leftrightarrow \quad \mathbf{g}_0 = \begin{bmatrix} g(0) \\ g(\frac{1}{P}) \\ \vdots \\ g(L - \frac{1}{P}) \end{bmatrix}$$

Similarly, we can consider a delayed version of $g(t)$:

$$g(t-\tau) \quad \leftrightarrow \quad \mathbf{g}_\tau = \begin{bmatrix} g(0-\tau) \\ g(\frac{1}{P}-\tau) \\ \vdots \\ g(L - \frac{1}{P} - \tau) \end{bmatrix}$$

The length L is chosen such that at the maximal possible delay, $g(t-\tau)$ has support only on the interval $[0, L\rangle$ symbols, i.e., $L \geq L_g + \lceil \max(\tau) \rceil$.

Given $\mathbf{g}_\tau$ and knowing $\mathbf{g}_0$, how do we estimate τ? Note here that τ does not have to be a multiple of $\frac{1}{P}$, so that $\mathbf{g}_\tau$ is not exactly a shift of the samples in $\mathbf{g}_0$. A simple 'pattern matching' with entry-wise shifts of $\mathbf{g}_0$ will thus not give an exact result.

We can however make use of the fact that a Fourier transformation maps a delay to a certain phase progression. Let

$$\tilde{g}(\omega_i) = \sum_k e^{-j\omega_i k} g(\frac{k}{P}), \qquad \omega_i = i\frac{2\pi}{LP}, \qquad i = 0, 1, \cdots, LP-1, \;\; k = 0, 1, \cdots, LP-1$$

In matrix-vector form, this can be written as

$$\tilde{\mathbf{g}}_0 = \mathcal{F}\,\mathbf{g}_0, \qquad \tilde{\mathbf{g}}_\tau = \mathcal{F}\,\mathbf{g}_\tau$$

where $\mathcal{F}$ denotes the DFT matrix of size $LP \times LP$, defined by

$$\mathcal{F} := \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \phi & \cdots & \phi^{LP-1} \\ \vdots & \vdots & & \vdots \\ 1 & \phi^{LP-1} & \cdots & \phi^{(LP-1)^2} \end{bmatrix}, \quad \phi = e^{-j\frac{2\pi}{LP}}. \tag{5.6}$$

If τ is an integer multiple of $\frac{1}{P}$, then it is straightforward to see that the Fourier transform $\tilde{\mathbf{g}}_\tau$

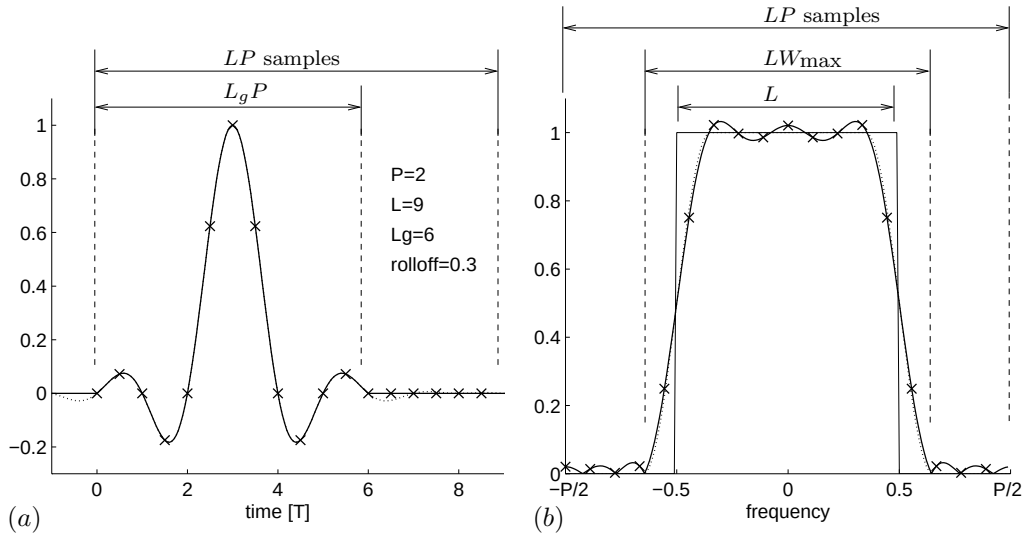


Figure 5.3. Definition of parameters: (a) time domain, (b) frequency domain.

of the sampled version of $g(t-\tau)$ is given by

$$\tilde{\mathbf{g}}_\tau \;=\; \tilde{\mathbf{g}}_0 \odot \begin{bmatrix} 1 \\ \phi^{\tau P} \\ (\phi^{\tau P})^2 \\ \vdots \\ (\phi^{\tau P})^{LP-1} \end{bmatrix} \;=\; \mathrm{diag}(\tilde{\mathbf{g}}_0) \cdot \begin{bmatrix} 1 \\ \phi^{\tau P} \\ (\phi^{\tau P})^2 \\ \vdots \\ (\phi^{\tau P})^{LP-1} \end{bmatrix} \tag{5.7}$$

where $\odot$ represents entrywise multiplication of the two vectors. The same holds true for any τ if $g(t)$ is bandlimited and sampled at or above the Nyquist rate.[1]

Thus, we will assume that $g(t)$ is bandlimited and sampled at such a rate that (5.7) is valid even if τ is not an integer multiple of $\frac{1}{P}$. The next step is to do a deconvolution of $g(t)$ by entrywise dividing $\tilde{\mathbf{g}}_\tau$ by $\tilde{\mathbf{g}}_0$. Obviously, this can be done only on intervals where $\tilde{\mathbf{g}}_0$ is nonzero.

To be specific, assume that $g(t)$ is bandlimited with normalized bandwidth $W_{\max}$ (that is, its Fourier transform is nonzero only for angular frequencies $|\omega| \le \frac{1}{2}W_{\max}$), and assume that $P > W_{\max}$. Then $\tilde{\mathbf{g}}_0$ has at most $LW_{\max}$ nonzero entries, and we can limit our attention to this interval. For a raised-cosine pulse shape with roll-off factor (excess bandwidth) ρ, we have $W_{\max} = 1+\rho$, see figure 5.3. Usually, however, we would select a somewhat smaller number, W say, since the entries at the border can be relatively small as well, and their inversion can blow

[1]This is not in full agreement with the FIR assumption we made earlier. Because of the truncation to length L, the spectrum of $g(t)$ widens and sampling at a rate $\frac{1}{P}$ introduces some aliasing due to spectral folding. This will eventually lead to a small bias in the delay estimate.

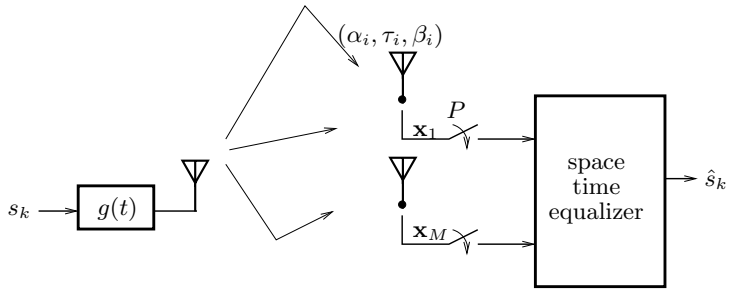


Figure 5.4. Multiray propagation channel

up the noise. Indeed, in the case of a raised-cosine pulse, we would set $W = 1$ and select only the L center frequency samples.

Let $J_{\tilde{\mathbf{g}}} : LW \times LP$ be the corresponding selection matrix for $\tilde{\mathbf{g}}$, such that $J_{\tilde{\mathbf{g}}}\tilde{\mathbf{g}}$ has the desired entries. For later use, we require that the selected frequencies appear in increasing order, which with the definition of the DFT in (5.6) usually means that the final $\lceil LW/2 \rceil$ samples of $\tilde{\mathbf{g}}_0$ should be moved up front: $J_{\tilde{\mathbf{g}}}$ has the form

$$J_{\tilde{\mathbf{g}}} = \begin{bmatrix} 0 & 0 & I_{\lceil LW/2 \rceil} \\ I_{\lfloor LW/2 \rfloor} & 0 & 0 \end{bmatrix} : \quad LW \times LP\,.$$

If there are no other (intermittent) zero entries, we can factor $\mathrm{diag}(J_{\tilde{\mathbf{g}}}\tilde{\mathbf{g}}_0)$ out of $J_{\tilde{\mathbf{g}}}\tilde{\mathbf{g}}_\tau$ and obtain

$$\mathbf{z} := \{\mathrm{diag}(J_{\tilde{\mathbf{g}}}\tilde{\mathbf{g}})\}^{-1} J_{\tilde{\mathbf{g}}}\tilde{\mathbf{g}}_\tau\,, \qquad (LW \times 1) \tag{5.8}$$

which satisfies the model

$$\mathbf{z} = \mathbf{f}(\phi)\,, \qquad \mathbf{f}(\phi) := \begin{bmatrix} 1 \\ \phi \\ \phi^2 \\ \vdots \\ \phi^{LW-1} \end{bmatrix}, \qquad \phi := e^{j2\pi\tau/L} \tag{5.9}$$

Note that $\mathbf{f}(\phi)$ has the same structure as $\mathbf{a}(\theta)$ for a ULA. Hence, we can apply the ESPRIT algorithm in the same way as before to estimate ϕ from $\mathbf{z}$, and subsequently τ. In the present case, we simply split $\mathbf{z}$ into two subvectors $\mathbf{x}$ and $\mathbf{y}$, one a shift of the other, and from the model $\mathbf{y} = \mathbf{x}\phi$ we can obtain $\phi = \mathbf{x}^\dagger\mathbf{y}$, $\tau = \mathrm{imag}(\log(\phi)) \cdot \frac{L}{2\pi}$.

5.2.2 Multipath channel model estimation

Single antenna case We will now build on the above principle. Consider a multipath channel which consists of r delayed copies of $g(t)$, as in figure 5.4, so that the impulse response is

$$h(t) = \sum_{i=1}^{r} \beta_i g(t-\tau_i) \qquad \Leftrightarrow \qquad \mathbf{h} = \sum_{i=1}^{r} \mathbf{g}_{\tau_i}\beta_i = [\mathbf{g}_{\tau_1}, \cdots, \mathbf{g}_{\tau_r}] \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_r \end{bmatrix} =: \mathbf{G}_\tau \boldsymbol{\beta}$$

We assume that we know $\mathbf{h}$ (e.g., from a channel identification using a training sequence). Also $g(t)$ is known. The unknowns are the parameters $\{\tau_i\}$ and $\{\beta_i\}$. Our objective is to estimate these parameters.

As before, we can introduce the DFT transformation and the deconvolution by the known pulse shape,

$$\mathbf{z} := \{\mathrm{diag}(J_{\tilde{\mathbf{g}}}\tilde{\mathbf{g}})\}^{-1} J_{\tilde{\mathbf{g}}} \mathcal{F}\mathbf{h}\,, \qquad (LW \times 1)$$

The vector $\mathbf{z}$ has model

$$\mathbf{z} = \mathbf{F}\boldsymbol{\beta}\,, \qquad \mathbf{F} = [\mathbf{f}(\phi_1), \cdots, \mathbf{f}(\phi_r)]\,, \qquad \mathbf{f}(\phi) := \begin{bmatrix} 1 \\ \phi \\ \phi^2 \\ \vdots \\ \phi^{LW-1} \end{bmatrix}$$

Since there are now multiple components in $\mathbf{F}$ and only a single vector $\mathbf{z}$, we cannot simply estimate the parameters from this single vector by splitting it in $\mathbf{x}$ and $\mathbf{y}$: this would allow only to estimate a model with a single component. However, we can use the shift-invariance of the vectors $\mathbf{f}(\cdot)$ to construct a matrix out of $\mathbf{z}$ as

$$\mathbf{Z} = [\mathbf{z}^{(0)}\,,\ \mathbf{z}^{(1)}\,,\ \cdots,\ \mathbf{z}^{(m-1)}]\,, \quad (LW-m+1 \times m)\,, \qquad \mathbf{z}^{(i)} := \begin{bmatrix} z_{i+1} \\ z_{i+2} \\ \vdots \\ z_{LW-m+i} \end{bmatrix}$$

where $\mathbf{z}^{(i)}$ is a subvector of $\mathbf{z}$ containing the $i+1$-st till the $LW-m+i$-th entry. If we define $\mathbf{f}(\phi)^{(i)}$ similarly, then

$$\mathbf{f}(\phi)^{(i)} = \begin{bmatrix} \phi^i \\ \phi^{i+1} \\ \phi^{i+2} \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 \\ \phi \\ \phi^2 \\ \vdots \end{bmatrix} \phi^i =: \mathbf{f}'(\phi)\phi^i$$

Thus, $\mathbf{Z}$ has the model

$$\mathbf{Z} = \mathbf{F}'\mathbf{B}\,, \qquad \mathbf{F}' = [\mathbf{f}'(\phi_1), \cdots, \mathbf{f}'(\phi_r)]$$

$$\mathbf{B} = [\boldsymbol{\beta} \quad \boldsymbol{\Phi}\boldsymbol{\beta} \quad \boldsymbol{\Phi}^2\boldsymbol{\beta} \quad \cdots \quad \boldsymbol{\Phi}^{m-1}\boldsymbol{\beta}], \qquad \boldsymbol{\Phi} = \begin{bmatrix} \phi_1 & & \\ & \ddots & \\ & & \phi_r \end{bmatrix}$$

where $\mathbf{F}'$ is a submatrix of $\mathbf{F}$ of size $LW - m + 1 \times r$, and $\mathbf{B}$ has size $r \times m$. Since each column of $\mathbf{F}'$ has the required shift-invariant structure, this is a model of the form that can be used by ESPRIT: split $\mathbf{Z}$ into $\mathbf{X}$ and $\mathbf{Y}$,

$$\mathbf{Z} = \left[\begin{array}{c} \mathbf{X} \\ \hline * * * \end{array}\right] = \left[\begin{array}{c} * * * \\ \hline \mathbf{Y} \end{array}\right]$$

where $\mathbf{X}$ contains all but the last rows of $\mathbf{Z}$, and $\mathbf{Y}$ contains all but the first. Subsequently compute the eigenvalue decomposition

$$\mathbf{X}^\dagger\mathbf{Y} = \mathbf{T}^{-1}\boldsymbol{\Phi}\mathbf{T}\,.$$

This determines $\boldsymbol{\Phi}$ as the eigenvalues of $\mathbf{X}^\dagger\mathbf{Y}$, from which the delays $\{\tau_i\}$ can be estimated.

This algorithm produces high-resolution estimates of the delays, in case the parametrized model holds with good accuracy for $\mathbf{h}$. There is however one condition to check. ESPRIT requires that the factorization $\mathbf{Z} = \mathbf{F}'\mathbf{B}$ is a low-rank factorization, i.e., if $\mathbf{F}'$ is strictly tall ($LW - m + 1 > r$) and $\mathbf{B}$ is square or wide ($r \leq m$). These conditions imply

$$r \leq \frac{1}{2}LW$$

and if we assume $W = 1$ (not much excess bandwidth), then

$$r \leq \frac{1}{2}L$$

Thus, there is a limit on the number of rays that can be estimated: not more than half the length of the channel. If this condition cannot be satisfied, we need to use multiple antennas.

Multiple antennas Assume that we have an array with M antennas and have a multipath propagation channel of the form

$$\mathbf{h}(t) = \sum_{i=1}^{r} \mathbf{a}(\alpha_i)\beta_i g(t - \tau_i) \tag{5.10}$$

In this case, $\mathbf{h}(t)$ is a vector with M entries. Assume that we have estimated the channel $\mathbf{h}(t)$ and know the pulse shape function $g(t)$. If we stack samples of $\mathbf{h}(t)$ into a vector as before, we obtain

$$\mathbf{h} = \begin{bmatrix} \mathbf{h}(0) \\ \mathbf{h}(\frac{1}{P}) \\ \vdots \\ \mathbf{h}(L - \frac{1}{P}) \end{bmatrix} = \sum_{i=1}^{r} [\mathbf{g}_{\tau_i} \otimes \mathbf{a}(\alpha_i)]\beta_i \;=:\; [\mathbf{G}(\boldsymbol{\tau}) \circ \mathbf{A}(\boldsymbol{\alpha})]\boldsymbol{\beta}$$

where $\otimes$ denotes the kronecker product and $\circ$ the Khatri-Rao product,

$$\mathbf{a} \otimes \mathbf{b} := \begin{bmatrix} a_1\mathbf{b} \\ \vdots \\ a_N\mathbf{b} \end{bmatrix}, \qquad \mathbf{A} \circ \mathbf{B} := [\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \cdots \quad \mathbf{a}_r \otimes \mathbf{b}_r]$$

We can now continue and construct $\mathbf{z}$ from the DFT-transformation of $\mathbf{h}$, and subsequently $\mathbf{Z}$ from (block-)shifts of $\mathbf{z}$ as before,

$$\mathbf{Z} = [\mathbf{z}^{(0)}\,,\ \mathbf{z}^{(1)}\,,\ \cdots\,,\ \mathbf{z}^{(m-1)}]\,, \qquad (M(LW - m + 1) \times m)$$

$\mathbf{Z}$ has a model

$$\mathbf{Z} = [\mathbf{F}'(\boldsymbol{\phi}) \circ \mathbf{A}(\boldsymbol{\alpha})]\mathbf{B}\,, \qquad \mathbf{F}' = [\mathbf{f}'(\phi_1)\,, \cdots\,, \mathbf{f}'(\phi_r)]$$

$$\mathbf{B} = [\boldsymbol{\beta} \quad \boldsymbol{\Phi}\boldsymbol{\beta} \quad \boldsymbol{\Phi}^2\boldsymbol{\beta} \cdots \boldsymbol{\Phi}^{m-1}\boldsymbol{\beta}]\,, \qquad \boldsymbol{\Phi} = \begin{bmatrix} \phi_1 & & \\ & \ddots & \\ & & \phi_r \end{bmatrix}$$

Since each column of $\mathbf{F}'$ has the required shift-invariance structure, the ESPRIT algorithm can be applied to $\mathbf{Z}$, defining $\mathbf{X}$ and $\mathbf{Y}$ as submatrices of $\mathbf{Z}$. The shift is now over a block of M rows: $\mathbf{X}$ is all but the last M rows of $\mathbf{Z}$, and $\mathbf{Y}$ is all but the first M rows of $\mathbf{Z}$. This will then produce estimates of $\{\tau_i\}$.

The conditions for identifiability now become

$$\begin{cases} r & \leq & m \\ r & \leq & M(LW - m) \end{cases} \qquad \Rightarrow \qquad r \leq \frac{M}{M+1} LW$$

Thus, with many antennas we can estimate almost LW delays, two times more than before. Note that there are no conditions on the antenna array: it can be an arbitrary collection of antennas.

If the antenna array has the shift-invariance structure required by ESPRIT, for example it is a ULA, then we can also estimate the angles in quite the same way, by applying the ESPRIT algorithm to other submatrices of $\mathbf{Z}$. We can even connect the angles and delays so obtained, because the eigenvector matrix $\mathbf{T}$ is the same in both eigenvalue problems. We omit the details here, see [5].

5.3 FREQUENCY ESTIMATION

The ESPRIT algorithm can also be used to estimate frequencies. Consider a signal $x(t)$ which is the sum of d harmonic components,

$$x(t) = \sum_{i=1}^{d} \beta_i e^{j\omega_i t} \tag{5.11}$$

Suppose that we uniformly sample this signal with period T (satisfying the Nyquist criterion, here $-\pi \leq \omega_i T < \pi$), and have available $x(T),\ x(2T),\ \cdots,\ x_(NT)$. We can then collect the samples in a data matrix $\mathbf{Z}$ with m rows,

$$\mathbf{Z} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots \\ x_2 & x_3 & x_4 & \cdots \\ \vdots & \vdots & & \vdots \\ x_m & x_{m+1} & \cdots & x_N \end{bmatrix}, \qquad x_k = x(kT)\,.$$

From (5.11), we see that this matrix satisfies the model

$$\mathbf{Z} = \mathbf{A}(\boldsymbol{\omega})\mathbf{S} := \begin{bmatrix} 1 & \cdots & 1 \\ \phi_1 & \cdots & \phi_d \\ \phi_1^2 & \cdots & \phi_d^2 \\ \vdots & & \vdots \\ \phi_1^{m-1} & \cdots & \phi_d^{m-1} \end{bmatrix} \begin{bmatrix} \beta_1\phi_1 & \beta_1\phi_1^2 & \cdots \\ \vdots & \vdots & \\ \beta_d\phi_d & \beta_d\phi_d^2 & \cdots \end{bmatrix}$$

where $\phi_i = e^{j\omega_i T}$. Since the model is the same as before, we can estimate the phase factors $\{\phi_i\}$ as before using ESPRIT, and from these the frequencies $\{\omega_i\}$ follow uniquely, since the Nyquist condition was assumed to hold.

The parameter m has to be chosen larger than d. A larger m will give more accurate estimates, however if N is fixed then the number of columns of $\mathbf{Z}$ $(= N - m + 1)$ will get smaller and there is a tradeoff. For a single sinusoid in noise, one can show that the most accurate estimate is obtained by making $\mathbf{Z}$ rectangular with 2 times more columns than rows, $m = \frac{N}{3}$.

It is straightforward to extend the algorithm with multiple antennas, enabling joint angle-frequency estimation [6].

Bibliography

[1] R. Roy and T. Kailath, "ESPRIT – Estimation of Signal Parameters via Rotational Invariance Techniques," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. 37, pp. 984–995, July 1989.

[2] M. Haardt and J.A. Nossek, "Unitary ESPRIT: how to obtain increased estimation accuracy with a reduced computational burden," *IEEE Trans. Signal Proc.*, vol. 43, pp. 1232–1242, May 1995.

[3] M.D. Zoltowski, M. Haardt, and C.P. Mathews, "Closed-form 2-D angle estimation with rectangular arrays in element space or beamspace via Unitary ESPRIT," *IEEE Trans. Signal Processing*, vol. 44, pp. 316–328, Feb. 1996.

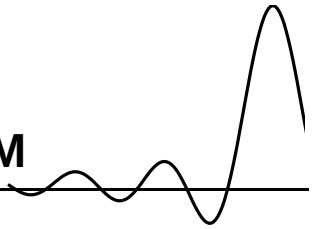
[4] P. Stoica and K.C. Sharman, "Maximum Likelihood methods for direction-of-arrival estimation," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. 38, pp. 1132–1143, July 1990.

[5] A.J. van der Veen, M.C. Vanderveen, and A. Paulraj, "Joint angle and delay estimation using shift-invariance properties," *subm. IEEE Signal Processing Letters*, Aug. 1996.

[6] M.D. Zoltowski and C.P. Mathews, "Real-time frequency and 2-D angle estimation with sub-Nyquist spatio-temporal sampling," *IEEE Trans. Signal Processing*, vol. 42, pp. 2781–2794, Oct. 1994.

Chapter 6

THE CONSTANT MODULUS ALGORITHM



Contents

6.1 INTRODUCTION

In this chapter we introduce an elementary form of *blind beamforming* (see figure 6.1). In wireless communications, this problem arises when a number of sources at distinct locations transmit signals at the same frequency and in the same time slot. The signals are received by the base station, which contains an array of antennas. By linearly combining the antenna outputs, the objective is to separate the signals and remove the interference from the other signals. In chapter 3 we have considered this situation, but assumed that we had available a training sequence: a segment of the signal of interest which is known. In this chapter, we consider a "blind" algorithm: a blind beamformer is to compute the proper weight vectors $\mathbf{w}_i$ from the measured data only, *without* detailed knowledge of the signals and the channel. It can do so by comparing properties of the signal at the output of the beamformer to properties that the desired source signal would have at this point.

For example, if we know that the desired source has an amplitude $|s_k|$ constant to 1 for every sample (such a signal is called *constant modulus*), then we can test this property for the output signal y_k of the beamformer, and define an error equal to the modulus difference $|y|^2 - 1$. See figure 6.1(a). Alternatively, we can estimate the best signal that has this property based on the output of the beamformer, i.e., $\hat{s}_k = \frac{y_k}{|y_k|}$, and give an error equal to $\hat{s}_k - y_k$. Here, $\hat{s}_k$ is regarded as a good estimate of the source signal, it is used as a reference signal instead of s_k. It is an elementary form of *decision feedback*. If the source belongs to a certain alphabet, e.g., $s_k \in \{\pm 1\}$ for BPSK of $s_k \in \{\pm 1, \pm j\}$ for QPSK, then we can make estimates of the symbols by rounding y_k to the closest constellation point, and use the resulting symbol sequence as a reference signal. See figure 6.1(b).

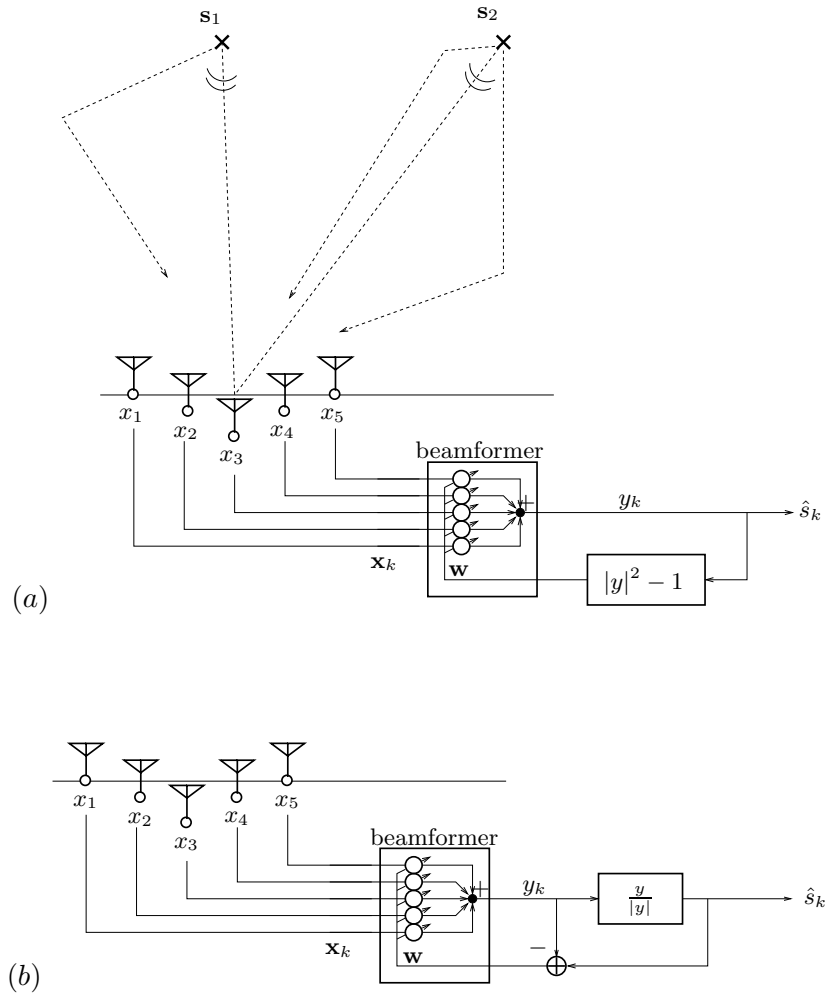


Figure 6.1. Blind beamforming scenarios: (a) based on modulus error, (b) based on estimated output error.

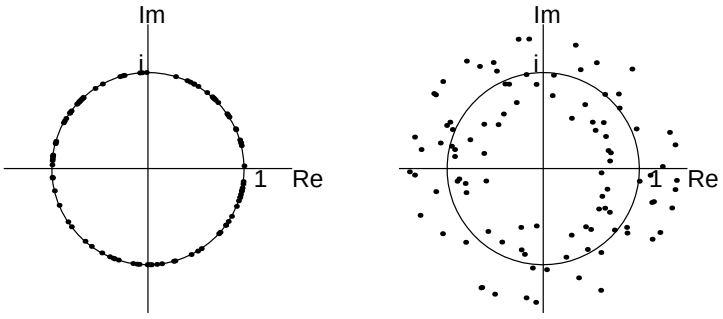


Figure 6.2. (a) single CM signal, (b) sum of two CM signals

Throughout this chapter we assume as before a stationary situation with essentially no delay spread (as compared to the inverse of the signal bandwidths), so that no equalization is required. Any angle spread is acceptable. The situation is described by the simple data model

$$\mathbf{x}(k) = \mathbf{A}\mathbf{s}(k) \tag{6.1}$$

where the vector $\mathbf{x}(k)$ is a stacking of the m antenna outputs $x_i(k)$ at discrete time k, $\mathbf{s}(k)$ is a stacking of the d source signals $s_i(k)$, and A is the array response matrix which describes the linear combinations of the signals as received by the antennas. The beamforming problem is to find weight vectors $\mathbf{w}_i$, one for each source, such that $\mathbf{w}_i^{\mathrm{H}}\mathbf{x}(k) = s_i(k)$ is equal to one of the original sources, without interference from the others.

Although we will be concerned with blind beamforming, it is useful to note that a quite similar problem arises in the context of *blind equalization* of a single source observed through an unknown time-dispersive FIR channel. In that situation, the received signal $x(k)$ is a linear combination of shifts of the original source $s(k)$. By feeding $x(k)$ through a tapped delay line, we can construct a vector of received signals and we will arrive at the same model as (6.1), be it with more structure since $s_i(k) = s(k-i)$ and $x_i(k) = x(k-i)$. Another aspect that distinguishes blind equalization from blind beamforming is that in the latter we try to receive *all* independent sources.

In chapter 5, we have looked at beamforming algorithms that focussed on properties of $\mathbf{A}$. We assumed that the columns of $\mathbf{A}$ are vectors on the array manifold, each associated to a certain direction-of-arrival (DOA). By finding these directions, we obtain an estimate of $\mathbf{A}$, and subsequently we can construct a beamformer $\mathbf{W}$ to separate the sources. This approach requires a calibrated array, and a scenario with very limited multipath propagation (since all DOAs have to be estimated).

In this chapter, we look at a second class of approaches, exploiting structural properties of the source vector that should hold and be reconstructed by the beamformer. This is more promising in the presence of unstructured multipath and useful in the context of blind equalization as well.

6.2 THE CONSTANT MODULUS ALGORITHM

One widely used property, and the property considered here, is the *constant modulus* of many communication signals (e.g. FM and PM in the analog domain, and FSK, PSK, 4-QAM for digital signals). For such signals, the amplitude $|s(k)|$ is a constant, typically normalized to 1, and all information is carried in the phase. If we have a single source $s(k)$, and plot the (complex) samples in the complex plane, then all samples will lie on the unit circle, see figure 6.2. On the other hand, if we have the sum of two sources, $s_1(k) + \alpha s_2(k)$, then the samples will in general not lie on a circle, unless $\alpha = 0$ (or if there are very special relations between the two sources—this is not possible if the two sources are independent). If $\alpha \neq 0$, then the received samples will be on a donut-shaped annulus.

The idea of modulus restoral is to play with the weights of a beamformer $\mathbf{w}$ until the output $y(k) = \hat{s}(k) = \mathbf{w}^{\mathrm{H}}\mathbf{x}(k)$ has the same property, $|\hat{s}(k)| = 1$, for all k. If that is the case, the output signal will be equal to one of the original sources [1].

6.2.1 CMA cost function

A popular implementation of such a property restoral algorithm is found by writing down a suitable cost function and minimizing it using stochastic gradient-descent techniques. For example, for a sample vector $\mathbf{x}_k$ we can consider as cost function the expected deviation of the squared modulus of the output signal $y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k$ to a constant, say 1:

$$J(\mathbf{w}) \;=\; \mathrm{E}(|y_k|^2 - 1)^2 = \mathrm{E}(|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k|^2 - 1)^2 \,. \tag{6.2}$$

This is simply a positive measure of the average amount that the beamformer output y_k deviates from the unit modulus condition. The objective in choosing $\mathbf{w}$ is to minimize J and hence to make y_k as close to a constant modulus signal as possible. Without additive noise, if we manage to achieve $J(\mathbf{w}) = 0$ then $\mathbf{w}$ reconstructs one of the sources.

In the derivation of the Wiener receiver, we could compute the minimizer $\mathbf{w}$ of the cost function analytically. For the CM cost function (6.2) this is not possible because it is a fourth-order function with a more complicated structure. However, there are many ways in which we can iteratively search for the minimum of J. The simplest algorithm again follows from a stochastic gradient-descent, similar to the derivation of the LMS algorithm. In this case, we update $\mathbf{w}$ iteratively, with small steps into the direction of the negative gradient,

$$\mathbf{w}^{(k+1)} \;=\; \mathbf{w}^{(k)} - \mu\nabla(J_k)$$

where μ is a small step size, and $\nabla(J_k) \equiv \nabla_{\bar{\mathbf{w}}} J(\mathbf{w}^{(k)})$ is the gradient vector of $J(\mathbf{w})$ with respect to the entries of $\bar{\mathbf{w}}$ (treated independently from $\mathbf{w}$), evaluated at the current value of $\mathbf{w}$. Using complex calculus and the fact that $|y_k|^2 = y_k\bar{y}_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}\mathbf{x}^{\mathrm{H}}\mathbf{w}$, it can be verified that the gradient is given by

$$\begin{aligned}
\nabla_{\mathbf{w}} J &= \mathrm{E}\{(|y_k|^2 - 1)\cdot\nabla(\mathbf{w}^{\mathrm{H}}\mathbf{x}\mathbf{x}^{\mathrm{H}}\mathbf{w})\} \\
&= 2\mathrm{E}\{(|y_k|^2 - 1)\cdot\mathbf{x}\mathbf{x}^{\mathrm{H}}\mathbf{w}\} \\
&= 2\mathrm{E}\{(|y_k|^2 - 1)y_k\mathbf{x}\}\,.
\end{aligned}$$

Thus, we can find a minimizer $\mathbf{w}$ iteratively via

$$\mathbf{w}^{(k+1)} \;=\; \mathbf{w}^{(k)} - \mu \mathbf{x}_k \, \mathbf{x}_k^{\mathrm{H}} \mathbf{w}^{(k)} \, (|\mathbf{w}^{(k)H} \mathbf{x}_k|^2 - 1) \tag{6.3}$$

(absorbing the factor 2 in μ). This iteration is called the Constant Modulus Algorithm (CMA, Treichler Agee and Larimore 1983 [1, 2]) and was first introduced for the case of blind equalization. It has its roots in the work of Sato [3] and Godard [4]. See [5, 6] for recent overviews and a historical perspective.

In comparison to the LMS algorithm, we see that the role of the update error (in the LMS equal to the output error $\epsilon_k = y_k - s_k$) is here played by

$$\epsilon_k = (|y_k|^2 - 1) y_k$$

In the LMS, we need a reference signal s_k to which we want to have the output converge. In CMA, however, the reference signal is not necessary, we use the a priori information that $|y_k| = 1$ in the absence of interfering signals.

6.2.2 Properties of the CMA

The CMA cost function as a function of y is shown in figure 6.3(a). For clarity, the plot is shown for a real-valued y. Note that there are minima for $|y| = 1$ and a local maximum for $y = 0$. In the complex case, the minima will be the unit circle. Thus, there is no unique minimum. Indeed, if $\mathbf{w}$ is a beamformer such that $y_k = \mathbf{w}^{\mathrm{H}} \mathbf{x}_k$ has the constant-modulus property, then another beamformer is $\alpha \mathbf{w}$, for any scalar α with $|\alpha| = 1$. This would scale the output by $\bar{\alpha}$, but this represents only a constant phase shift. It is clear that without further information on the source we cannot remove this ambiguity. Very often, the ambiguity is not important since the information is coded in the phase changes (such as for frequency-modulated sources).

Figure 6.3(b) shows the cost function in the $\mathbf{w}$-domain, again for a real-valued case. There are two antennas; the horizontal axes are w_1 and w_2, respectively; the vertical axis is $J([w_1 \;\; w_2]^T)$. There are four minima, corresponding to beamformers that reconstruct $\mathbf{s}_1$, $-\mathbf{s}_1$, $\mathbf{s}_2$ and $-\mathbf{s}_2$, respectively. For complex signals, there would be two classes of beamformers, reconstructing $\alpha \mathbf{s}_1$ and $\alpha \mathbf{s}_2$, for arbitrary α with $|\alpha| = 1$. The graph also shows the convergence of the CMA algorithm for two different initializations. Note that depending on the initial point, the algorithm converges to different minima.

Figure 6.3(c) shows the corresponding costs as function of the number of updates. As with LMS, a larger step size makes the convergence faster but also more noisy. The top graph shows the convergence of the CMA cost as used by the algorithm, the bottom graph the convergence in terms of the output error, $J(\mathbf{w}) = \mathbf{w}^{\mathrm{H}} \mathbf{R} \mathbf{w} - \mathbf{a}^{\mathrm{H}} \mathbf{w} - \mathbf{w}^{\mathrm{H}} \mathbf{a} + 1$. Here, $\mathbf{a}$ is the $\mathbf{a}$-vector of the source to which the algorithm converges. Note that the algorithm is *not* trying to minimize this cost function.

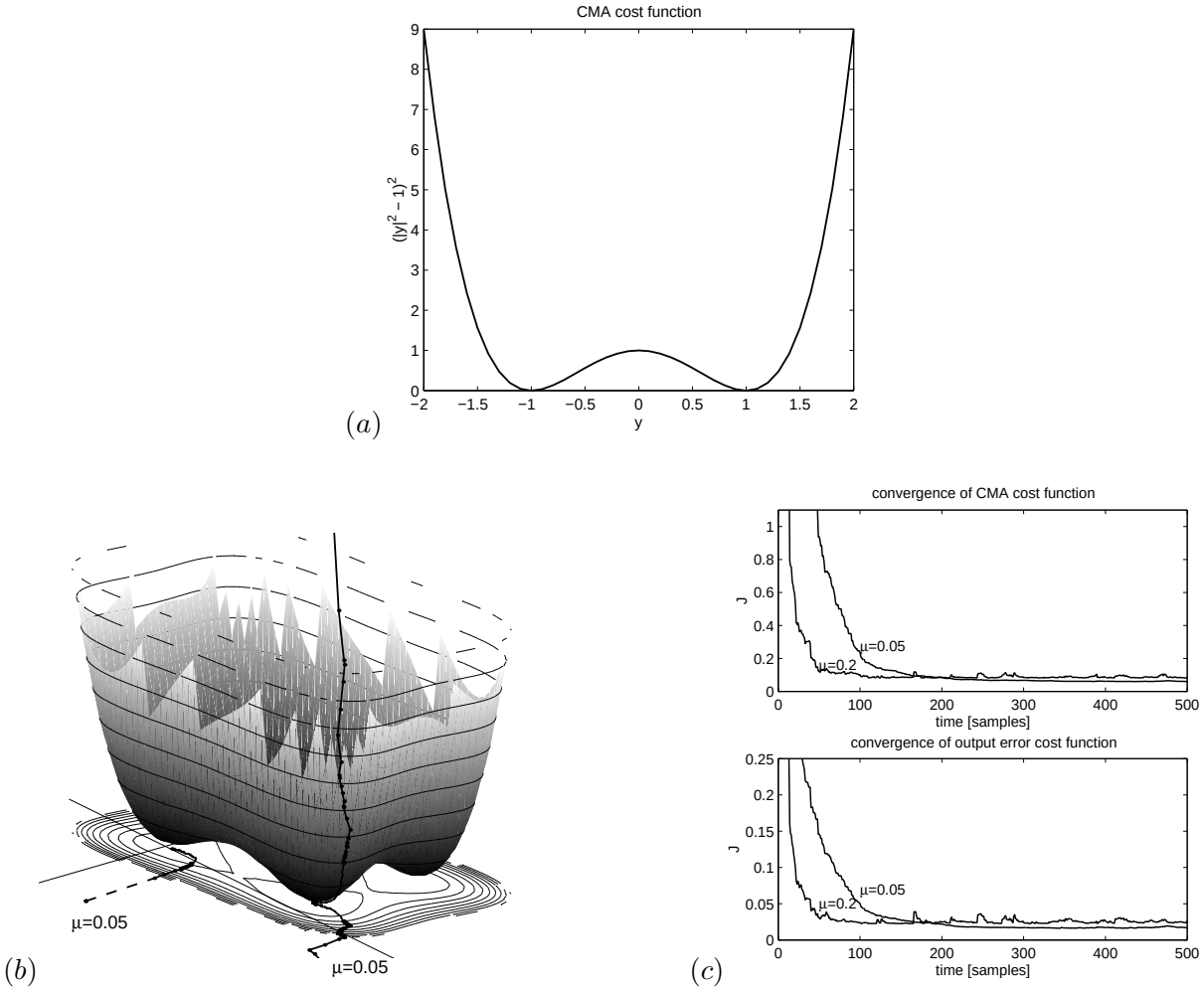


Figure 6.3. CMA cost function and convergence

Remarks

- We need to select a suitable step size μ and an initial point $\mathbf{w}^{(0)}$ for the iteration. Unlike LMS, we cannot choose $\mathbf{w}^{(0)} = 0$ since this precisely selects the local maxima of the cost function, but any other random vector will do.

  The maximal step size has not been theoretically derived. Because the cost function involves fourth order moments, the gradient is much steeper away from the optimum in comparison to LMS, and the maximal μ that still guarantees stability is smaller.

- One can show that the minimizing points of the CMA cost function are usually quite close (but not precisely identical) to the Wiener beamformers for each of the sources (up to scaling by α, $|\alpha| = 1$). Since it is a stochastic algorithm, the beamformer computed at each step of the CM algorithm will never completely converge but jitter around the minimizing solution with a variance depending on the step size.

- In chapter 5, we have found the factorization of $\mathbf{X}$ using properties of $\mathbf{A}$, in particular its Vandermonde structure. The CMA uses only the properties of $\mathbf{S}$. This is potentially more robust, since these properties can be controlled very well, whereas the propagation model is determined by nature and may or may not be close to our presumed specular multipath model. In addition, the estimation of DOAs requires the use of calibrated antenna arrays (the function $\mathbf{a}(\theta)$ must be known up to a scalar), or arrays with a specific structure (such as a ULA). Finally, DOA estimation is hard if two rays of the same user have identical delays, because in that case the $\mathbf{a}$-vectors of the two rays add up and the rank of the $\mathbf{X}$-matrix does not increase.

- The constant modulus property holds for all phase-modulated and frequency modulated signals, and for several types of signals in the digital domain (such as frequency-shift keying (FSK), phase-shift keying (PSK), binary-shift keying (BPSK) and 4-QAM. For digital signals, the fact that the source symbols are selected from a finite alphabet is an even stronger property that can very well be exploited.

  Even for sources that are not constant modulus, such as multi-level constellations (higher order QAM), the CMA can be successfully applied.

Variants of CMA

There exists a very similar CMA update rule [7],

$$y := \mathbf{w}^{(k)H}\mathbf{x}_k\,, \qquad \epsilon = \frac{y}{|y|} - y\,, \qquad \mathbf{w}^{(k+1)} := \mathbf{w}^{(k)} + \mu\mathbf{x}_k\,\overline{\epsilon} \tag{6.4}$$

(the overbar denotes complex conjugation). In this case, the update error that controls the iteration is $(\frac{y}{|y|} - y)$. Compared to the LMS, we see that $\frac{y}{|y|}$ plays the role of desired signal, see

also figure 6.1. Ideally, y is constant modulus and the error is zero. This update rule is derived from a slightly different cost function, namely the "CMA(1,2)" cost function

$$J(\mathbf{w}) \;=\; \mathrm{E}(|y_k|-1)^2 = \mathrm{E}(|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k|-1)^2\,. \tag{6.5}$$

An advantage is that the role of μ is more closely related to that of LMS, facilitating its analysis close to convergence.

It also allows to pose a "Normalized CMA" similar to the Normalized LMS (section 4.5),

$$y := \mathbf{w}^{(k)H}\mathbf{x}_k\,, \qquad \epsilon = \frac{y}{|y|} - y\,, \qquad \mathbf{w}^{(k+1)} \;:=\; \mathbf{w}^{(k)} + \frac{\mu}{\|\mathbf{x}_k\|^2}\mathbf{x}_k\,\overline{\epsilon} \tag{6.6}$$

where μ is made data scaling independent by dividing by the instantaneous input power. A better but slightly more complicated 'orthogonalization' version of CMA was considered in [7] and became known as orthogonal CMA (OCMA, see also [8]),

$$y := \mathbf{w}^{(k)H}\mathbf{x}_k\,, \qquad \epsilon = \frac{y}{|y|} - y\,, \qquad \mathbf{w}^{(k+1)} \;:=\; \mathbf{w}^{(k)} + \mu\mathbf{R}_k^{-1}\mathbf{x}_k\,\overline{\epsilon} \tag{6.7}$$

where $\mathbf{R}_k$ is the data covariance matrix estimated from the available data, iteratively given by

$$\mathbf{R}_k = \frac{1}{k}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}} + \frac{k-1}{k}\mathbf{R}_{k-1}$$

or possibly a sliding window or exponential window estimate as in the RLS.

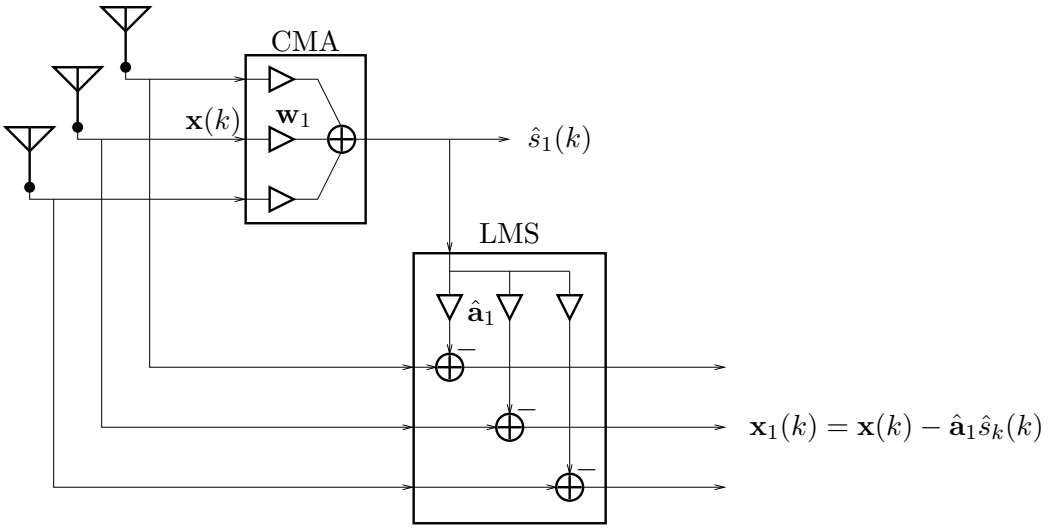
Finally, we mention the Least Squares CMA (LSCMA) [9], see also [10]. It is a *block* updating scheme acting iteratively on the complete data matrix $\mathbf{X} = [\mathbf{x}_1\,, \cdots\,,\, \mathbf{x}_N]$, assuming N samples are available. It is derived from optimizing the LS cost function $\min_{\mathbf{w}}\ \|\mathbf{s} - \mathbf{w}^{\mathrm{H}}\mathbf{X}\|$, substituting at time k the best blind estimate of the desired source vector $\mathbf{s} = [s_1\,,\ \cdots\,,\ s_N]$ available at that moment, i.e.,

$$\hat{\mathbf{s}} := [\frac{y_1}{|y_1|}\,,\ \frac{y_2}{|y_2|}\,,\ \cdots\,,\ \frac{y_N}{|y_N|}]$$

where $y_i = \mathbf{w}^{(k)H}\mathbf{x}_i$ is the output of the beamformer based on the estimate of $\mathbf{w}$ at iteration step k. The solution of the LS problem $\min_{\mathbf{w}}\ \|\hat{\mathbf{s}} - \mathbf{w}^{\mathrm{H}}\mathbf{X}\|$ is given by

$$\mathbf{w}^{(k+1)} := (\hat{\mathbf{s}}\,\mathbf{X}^{\dagger})^{\mathrm{H}}$$

One can show that this iteration converges to a minimum of the CMA(1,2) cost function (6.5). The advantage of this scheme is that the convergence is quickly and does not depend on a step size μ. It also requires much less temporal samples (N) than the CMA, since the same block of data is reused until convergence.



6.3 THE CM ARRAY

The CMA gives only a single beamformer vector. This is sufficient for blind equalization applications ($\mathbf{X} = \mathbf{H}\mathcal{S}$), since the rows of $\mathcal{S}$ are merely shifts of each other, and we don't have to find all of them. In contrast, the beamforming problem asks for all possible weight vectors that give back linearly independent CM signals, which is usually much harder.

If we initialize the CMA with a random vector $\mathbf{w}^{(0)}$, then the CMA tends to converge to the strongest signal. However, this cannot be guaranteed: for an initialization close enough to a weak signal, the algorithm converges to that weaker signal.[1] This gives one way to find all signals: use various initializations and determine if independent signals have been found.

A more secure algorithm is the so-called multi-stage CMA, also called the CM Array [11–14]. The output of a first CMA stage results in the detection of the first CM signal, and gives an estimate $\hat{s}_1(k)$, say. This signal can be used as a reference signal for the LMS algorithm in section 4.7.1 to estimate the corresponding array response vector $\hat{\mathbf{a}}_1$. The update rule is given in equation (4.4):

$$\hat{\mathbf{a}}_1^{(k+1)} = \hat{\mathbf{a}}_1^{(k)} \; + \; \mu_{lms}[\mathbf{x}(k) - \hat{\mathbf{a}}_1^{(k)}\hat{s}_1(k)]\bar{\hat{s}}_1(k)\,.$$

We can then subtract the estimated source signal from the original data sequence,

$$\mathbf{x}_1(k) = \mathbf{x}(k) - \hat{\mathbf{a}}_1^{(k)}\hat{s}_1(k)$$

and feed the resulting filtered data to a second CMA stage in order to detect a possible second CM signal. This can be repeated until all signals have been found. See figure 6.3.[2]

[1] During the early days of CMA, only the reception of the strongest signal was desired, and convergence to another signal was regarded as mis-convergence.

[2] The algorithm used in the CM Array in [12–14] is in fact the alternative CMA in (6.4).

A problem with this scheme is that the LMS algorithm can converge only after the first CMA has sufficiently converged. In the mean time, the second CMA may have converged to the same signal as the first CMA (especially if it is strong), and if the first LMS is not completely removing this signal, the second stage will stay at this signal. Thus, it may happen that the same signal is found twice, and/or that not all signals are found. A related problem is that the CMA converges to a point close to the Wiener solution. Hence, the estimate $\hat{s}_1(k)$ will always contain components of the other signals as well. An analysis of the situation is given in [12, 14].

A second problem is that the convergence speed may be slow (several hundreds of samples), since we have a cascade of adaptive algorithms. It has been proposed to use direction-finding algorithms such as MUSIC first to initialize the cascade. An alternative approach is to augment the cost function with additional terms that express the independence of the output signals, e.g., by putting a constraint on the cross-correlation of these signals [15, 16].

It has also been proposed to force the linear independence of the beamforming vectors (or even their orthogonality after prewhitening). However, if the number of sensors is larger than the number of sources, there exist vectors $\mathbf{w}_0$ in the left null space of $\mathbf{A}$ such that $\mathbf{w}_0^{\mathrm{H}}\mathbf{A} = \mathbf{0}$. These vectors can be added to any solution $\mathbf{w}$ without changing the output signal. It is thus possible that independent beamforming vectors give rise to the same output signals, and hence it is not sufficient to require the independence of the $\mathbf{w}$.

In summary, iterative CMAs are straightforward to implement and computationally of modest complexity. They can however converge slowly, with unpredictable convergence speed, and the recovering of all independent sources remains a problem. It is thus interesting to note that the problem admits an elegant and algebraic solution, the Algebraic CMA (ACMA) [17].

Bibliography

[1] J.R. Treichler and B.G. Agee, "A new approach to multipath correction of constant modulus signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 31, pp. 459–471, Apr. 1983.

[2] M.G. Larimore and J.R. Treichler, "Convergence behavior of the constant modulus algorithm," in *Proc. IEEE ICASSP*, pp. 13–16 vol.1, 1983.

[3] Y. Sato, "A method of self-recovering equalization for multilevel amplitude-modulation systems," *IEEE Trans. Communications*, vol. 23, pp. 679–682, June 1975.

[4] D.N. Godard, "Self-recovering equalization and carrier tracking in two-dimensional data communication systems," *IEEE Trans. Communications*, vol. 28, pp. 1867–1875, Nov. 1980.

[5] S. Haykin, ed., *Blind Deconvolution*. Englewood Cliffs: Prentice Hall, 1994.

[6] R. Johnson, P. Schniter, T.J. Endres, J.D. Behm, D.R. Brown, and R.A. Casas, "Blind equalization using the constant modulus criterion: a review," *Proc. IEEE*, vol. 86, pp. 1927–1950, Oct. 1998.

[7] R. Gooch and J. Lundell, "The CM array: An adaptive beamformer for constant modulus signals," in *Proc. IEEE ICASSP*, (Tokyo), pp. 2523–2526, 1986.

[8] R. Pickholtz and K. Elbarbary, "The recursive constant modulus algorithm: A new approach for real-time array processing," in *27-th Asilomar Conf. Signals, Syst. Comp.*, pp. 627–632, IEEE, 1993.

[9] B.G. Agee, "The least-squares CMA: A new technique for rapid correction of constant modulus signals," in *Proc. IEEE ICASSP*, (Tokyo), pp. 953–956, 1986.

[10] T.E. Biedka, W.H. Tranter, and J.H. Reed, "Convergence analysis of the Least Squares Constant Modulus Algorithm in interference cancellation applications," *IEEE Trans. Communications*, vol. 48, pp. 491–501, Mar. 2000.

[11] J.R. Treichler and M.G. Larimore, "New processing techniques based on constant modulus adaptive algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, pp. 420–431, Apr. 1985.

[12] J.J. Shynk and R.P. Gooch, "The constant modulus array for cochannel signal copy and direction finding," *IEEE Trans. Signal Proc.*, vol. 44, pp. 652–660, Mar. 1996.

[13] A. Mathur, A.V. Keerthi, J.J. Shynk, and R.P. Gooch, "Convergence properties of the multistage constant modulus array for correlated sources," *IEEE Tr. Signal Processing*, vol. 45, pp. 280–286, Jan. 1997.

[14] A.V. Keerthi, A. Mathur, and J.J. Shynk, "Misadjustment and tracking analysis of the constant modulus array," *IEEE Trans. Signal Processing*, vol. 46, pp. 51–58, Jan. 1998.

[15] C.B. Papadias and A.J. Paulraj, "A constant modulus algorithm for multiuser signal separation in presence of delay spread using antenna arrays," *IEEE Signal Proc. Letters*, vol. 4, pp. 178–181, June 1997.

[16] T. Nguyen and Z. Ding, "Blind CMA beamforming for narrowband signals with multipath arrivals," *Int. J. Adaptive Control and Signal Processing*, vol. 12, pp. 157–172, Mar. 1998.

[17] A.J. van der Veen and A. Paulraj, "An analytical constant modulus algorithm," *IEEE Trans. Signal Processing*, vol. 44, pp. 1136–1155, May 1996.

GLOSSARY

Abbreviations

BPSK	Binary Phase-Shift Keying
CDF	cumulative distribution function
C/I	Carrier to Interferer ratio
CM	Constant modulus
CMA	Constant modulus algorithm
DOA	Direction Of Arrival
FA	Finite alphabet
FIR	Finite duration Impulse Response
GMSK	Gaussian filtered Minimum-Shift Keying
GSM	Global System for Mobile communication
IF	Interferer
i.i.d.	independent and identically distributed
ISI	Inter Symbol Interference
LOS	Line Of Sight
ML	Maximum Likelihood
MIMO	Multi-input multi-output
MLSE	Maximum Likelihood Sequence Estimation
MSK	Minimum-Shift Keying
PDF	probability density function
PSK	Phase-Shift Keying
SISO	Single-input single-output
SNR	Signal to Noise Ratio
SOI	Signal Of Interest
ULA	Uniform Linear Array

Notation

$\mathrm{real}(z)$	The real part of z
$\mathrm{imag}(z)$	The imaginary part of z
$\bar{z}$	The complex conjugate of z
$\mathbf{I}$	The identity matrix
$\mathbf{A}^{\mathrm{T}}$	The transpose of $\mathbf{A}$
$\mathbf{A}^{\mathrm{H}}$	The complex conjugate transpose of $\mathbf{A}$
$\lvert\mathbf{A}\rvert$	The determinant of$\mathbf{A}$
$\mathbf{P}_A = \mathbf{A}(\mathbf{A}^{\mathrm{H}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{H}}$	The projection matrix onto the range of $\mathbf{A}$
$\mathbf{P}_A^{\perp} = \mathbf{I} - \mathbf{P}_A$	The projection matrix onto the orthogonal complement of the range of $\mathbf{A}$
$\mathrm{vec}(\mathbf{A})$	Stacking of the columns of $\mathbf{A}$ into a vector
$\mathbf{A} \otimes \mathbf{B}$	Kronecker product: matrix with block entries $a_{ij}\mathbf{B}$
$\mathbf{A} \odot \mathbf{B}$	Pointwise multiplication of matrices $\mathbf{A}$ and $\mathbf{B}$
$h * s(t)$	convolution

Common meaning of symbols

d	Number of sources (users)
L	Channel length (integer, in symbol periods)
M	Number of antennas
N	Number of snapshots
P	Oversampling rate
r	Number of rays in multipath model
$\mathbf{A}$	Array response matrix
$\mathbf{X}$	Data matrix
$\mathbf{S}$	Signal matrix
$\mathbf{H}$	Channel matrix