Advanced Digital Communications

Lecture Notes — Fall 2015

Michael C. Gastpar

EPFL

Contents

Chapter 1

Introduction

1.1 Foreword

This is a set of lecture notes for an MS-level class on Advanced Digital Communication taught at EPFL (COM-510; Fall 2011, Fall 2012, Fall 2013, Fall 2014, Fall 2015) and at the University of California, Berkeley (EE 224A; Fall 2010).

There are many excellent books on the topic, including [1, 2, 3, 4, 5], and these lecture notes are in no way designed to replace any of these books.

The only reason these lecture notes got written is that I could not find a single book that contained all that I wanted to cover in this class at the desired level of detail. Switching between multiple books not only requires deep pockets on the part of the students, but poses additional challenges concerning the harmonization of notation. Therefore, after confusing students again and again with notational inconsistencies, ever so slight, I decided I needed a more or less consistent set of lecture notes.

Most importantly, these lecture notes are not suitable outside of the class — they are **not self-contained** and not meant to make sense without the lectures that go with them. They mostly contain many of the necessary definitions, bent in such a way as to fit my purpose in this class, as well as some of the more cumbersome derivations and proofs.

As a further apology, for the same reasons already mentioned, these notes contain essentially **no figures** — not because I do not like figures (I think figures are of crucial importance to the learning process!), but because those will be given during lecture, and it will be much more instructive for the students to redraw them themselves.

As a final word of caution, I am confident that there are plenty of typos and errors left, and I am immensely grateful if you can report them back to me.

M. C. Gastpar, Lausanne, Switzerland

1.2 Acknowledgments

The author thanks France Faille, Lionel Martin, Lucas Maystre, Giel Op 't Veld, Saeid Sahraei, Chien-Yi Wang, and Jingge Zhu for pointing out errors, typos, and inconsistencies.

1.3 Practical Information, Fall 2015, EPFL

Instructor:
Michael Gastpar, michael.gastpar@epfl.ch, Office: INR 130, Office Hours: TBA

Teaching Assistants:
Giel Op 't Veld, giel.optveld@epfl.ch, Office: INR 031, Office Hrs: TBA
Saeid Sahraei, saeid.sahraei@epfl.ch, Office: INR 034, Office Hrs: TBA
Jingge Zhu, jingge.zhu@epfl.ch, Office: INR 031, Office Hrs: TBA

Administrative Assistant:
France Faille, france.faille@epfl.ch, Office: INR 311

Class Meetings:
Mondays, 8:15-10, INM 10
Wednesdays, 11:15-13:00, INR 113
Wednesdays, 13:15-15:00, INR 113 (Exercises)

Course Web Page:
We will use http://moodle.epfl.ch

Official Prerequisites:
Principles of Digital Communications (EPFL class COM-302). Contact the instructor if you are in doubt whether you meet the prerequisite.

Grading:
Problem Sets (10%) (we only grade two of the Problem Sets, to be announced later),
Midterm **November 4** (11:15-13:00) (40%), Final Exam (50%)

Homework:
There will be weekly problem sets with solutions. Two of the problem sets will be graded.

Exam Rules:
Both exams are closed-book and closed-notes; calculators, computing and communication devices are *not* permitted. However, two handwritten and *not photocopied* double-sided sheets of notes are allowed for the midterm exam, and an additional two for the final exam.

1.4 Lecture Schedule, Fall 2015, EPFL

Date	Topics	Reading
Chapter 1		
Sept 14	Introduction, History, Overview, Outlook	
Chapter 2 Review of Basic Concepts		
Sept 16	Fourier Transform, Random Variables&Processes	Sections 2.1-2.4
Sept 23	Detection Theory; AWGN Detection Problem	Section 2.5
Chapter 3 The AWGN Channel		
Sept 28	Modulation; The vector equivalent	Sections 3.1-3.3
Sept 30	Vector AWGN: basics and geometry	Section 3.4
Oct 5	Vector AWGN: high-dimensional signaling	Sections 3.5-3.7
Chapter 4 The Band-limited AWGN Channel		
Oct 7	The discrete-time equivalent	Sections 4.1-4.2
Oct 12	ML decoding in the presence of ISI (Viterbi)	Section 4.3
Oct 14	How to avoid ISI at Tx and Rx; good or bad?	Sections 4.4-4.5
Oct 19	Better Equalizers	Section 4.5
Oct 21	Bandpass Signaling and complex-valued channels	Section 4.6
Oct 26	OFDM; FFT-OFDM implementation	Sections 4.7-4.8
Oct 28	OFDM: Allocation for parallel channels	Section 4.9
Chapter 5 Wireless Communication		
Nov 2	Wireless Channels: Physics	Section 5.1
Nov 4	*Midterm Exam*	Chapters 2-4
Nov 9	Wireless Channels: Statistics; Rayleigh fading	Section 5.2-5.3
Nov 11	Diversity; Maximum ratio combining	Section 5.4
Nov 16	Diversity; Alamouti scheme	Section 5.5
Nov 18	Non-coherent detection; MIMO	Sections 5.6-5.7
Nov 23	MIMO: V-BLAST and beyond	Section 5.7
Nov 25	Wireless Multi-user Communication	Section 5.8
Chapter 6 Coding		
Nov 30	Binary Linear Block Codes	Sections 6.1-6.2
Dec 2	Binary Linear Block Codes	Section 6.2
Dec 7	Binary Codes on Noisy Channels	Sections 6.3-6.4
Dec 9	Famous Binary Linear Codes	Sections 6.5-6.6
Dec 14	LDPC Codes and Message-Passing Decoding	Handout
Dec 16	Discussion, Extensions	Handout

Chapter 2

Deterministic and Stochastic Signals

2.1 Fourier Transform

2.1.1 Continuous-time Fourier Transform

For continuous-time signals $x(t)$, we define the following transform:

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}dt \tag{2.1}$$

$$x(t) = \int_{-\infty}^{\infty} X(f)e^{j2\pi ft}df \tag{2.2}$$

We note that the inverse transform only returns the original $x(t)$ under additional assumptions on $x(t)$. However, this discussion is beyond the scope of this class.

The Fourier transform has many desirable properties. We here mention only one, the so-called *Parseval theorem* asserting that

$$\int_{-\infty}^{\infty} x(t)y^*(t)dt = \int_{-\infty}^{\infty} X(f)Y^*(f)df, \tag{2.3}$$

where * denotes the complex conjugate.

2.1.2 Discrete-time Fourier Transform (DTFT)

For discrete-time signals $x[n]$, we define the following transform:

$$X(f) = \sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi fn} \tag{2.4}$$

$$x[n] = \int_{-1/2}^{1/2} X(f)e^{j2\pi fn}df \tag{2.5}$$

We note that the inverse transform only returns the original $x[n]$ under additional assumptions on $x[n]$. However, this discussion is beyond the scope of this class.

2.1.3 The Z-transform

For discrete-time signals $x[n]$, we define the following transform:

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n} \tag{2.6}$$

This transform also has an inverse, involving complex integration, but we will not discuss this to any detail in this class. Nevertheless, we will find the Z-transform useful.

The Z-transform has many desirable properties. For example, we have:

$$y[n] = x[-n] \quad \longleftrightarrow \quad Y(z) = X(1/z). \tag{2.7}$$

2.1.4 Discrete Fourier Transform (DFT)

For discrete-time signals $x[n]$ of length N (or simply, vectors of length N), we define the following transform:

$$X[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn} \tag{2.8}$$

$$x[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X[k]e^{j\frac{2\pi}{N}kn} \tag{2.9}$$

We note that the inverse transform *always* returns the original $x[n]$.

We also note that we have made a debatable choice here: Instead of placing a factor $1/N$ in the inverse transform, we have "split" the factor into two factors $1/\sqrt{N}$. Our choice of normalization factors leads to a representation by *unitary* matrices. We will see the advantage of this choice when we study communication systems such as the FFT-OFDM approach, but we note that Matlab, for example, makes a different choice of normalization factors.

2.2 Random Variables

We will denote random variables by upper case letters such as X and their realizations by lower case letters such as x. For continuous-valued (real or complex) random variables Y, we will consider their *probability density function* (pdf) and denote it by

$$f_Y(y), \tag{2.10}$$

and when it is clear from context, we will sometimes drop the subscript Y and simply write $f(y)$. For discrete-valued random variables H, we will consider their *probability mass function* (pmf) and denote it by

$$p_H(h), \tag{2.11}$$

and when it is clear from context, we will sometimes drop the subscript H and simply write $p(h)$.

For the purpose of this class, one distribution plays a very special role: The *normal* or *Gaussian* distribution, defined as

$$f_Y(y) \;=\; \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-m)^2}{2\sigma^2}}. \tag{2.12}$$

This is a valid distribution for all choices of m and $\sigma > 0$ since it is non-negative and integrates to one. Moreover, it is easy to show that the mean of this distribution is m and the variance is σ^2.

For the purpose of our considerations, it will often be necessary to determine the probability that a Gaussian random variable lands in a certain range, say, between 3 and 5. This probability is simply the integral of the Gaussian pdf from 3 to 5, but unfortunately, this integral cannot be calculated explicitly. Therefore, it is useful to introduce the so-called Q-function:

$$Q(x) \;=\; \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{y^2}{2}} dy. \tag{2.13}$$

2.3 Random Vectors

We will denote random vectors by bold-face letters $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)^T$. The mean (vector) $\mathbf{m}$ of the random vector $\mathbf{Y}$ is defined as

$$\mathbf{m} \;=\; \mathbb{E}[\mathbf{Y}] = \begin{pmatrix} \mathbb{E}[Y_1] \\ \mathbb{E}[Y_2] \\ \vdots \\ \mathbb{E}[Y_n] \end{pmatrix}, \tag{2.14}$$

and the covariance matrix C of the random vector $\mathbf{Y}$ is defined as

$$\begin{aligned} C \;&=\; \mathbb{E}\left[(\mathbf{Y}-\mathbf{m})(\mathbf{Y}-\mathbf{m})^H\right] \\ &=\; \begin{pmatrix} \mathbb{E}\left[|Y_1|^2\right] - |m_1|^2 & \mathbb{E}[Y_1Y_2^*] - m_1m_2^* & \ldots & \mathbb{E}[Y_1Y_n^*] - m_1m_n^* \\ \mathbb{E}[Y_2Y_1^*] - m_2m_1^* & \mathbb{E}\left[|Y_2|^2\right] - |m_2|^2 & \ldots & \mathbb{E}[Y_2Y_n^*] - m_2m_n^* \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[Y_nY_1^*] - m_nm_1^* & \mathbb{E}[Y_nY_2^*] - m_1m_2^* & \ldots & \mathbb{E}\left[|Y_n|^2\right] - |m_n|^2 \end{pmatrix}, \end{aligned} \tag{2.15}$$

where ^{H} denotes the Hermitian transpose and * denotes the complex conjugate. Covariance matrices will play a key role in several considerations in this class. One of the most useful properties in the context of digital communications is the fact that if we define the new random vector $\mathbf{Z} = A\mathbf{Y}$, for an arbitrary matrix A, then the covariance matrix of the random vector $\mathbf{Z}$ is given by ACA^H, where C is the covariance matrix of the random vector $\mathbf{Y}$.

A (real-valued) random vector $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)^T$ is Gaussian if the joint pdf of its components is of the form

$$f_{\mathbf{Y}}(\mathbf{y}) = \frac{1}{(2\pi)^{n/2}\sqrt{\det(C)}} e^{-\frac{1}{2}(\mathbf{y}-\mathbf{m})^T C^{-1}(\mathbf{y}-\mathbf{m})}, \tag{2.16}$$

where C is a positive definite (symmetric) matrix. Again, this is a valid pdf since it is non-negative and integrates to one. Moreover, it can be shown that the mean of the random vector $\mathbf{Y}$ is given by the vector $\mathbf{m}$, and the covariance matrix of the random vector $\mathbf{Y}$ is given by the invertible matrix C.

2.4 Random Processes

2.4.1 Sequences of random variables

A (discrete-time) random process is a (generally infinite) sequence of random variables. Specifying a random process fully means giving the full joint probability density (or mass) function of all of the involved random variables. Let us denote a sequence of random variables as

$$\ldots, X[-2], X[-1], X[0], X[1], X[2], \ldots \tag{2.17}$$

The IID process

Perhaps the simplest sequence of random variables is to assume that all the random variables $X[n]$ are *independent and identically distributed* (IID). These processes will be very important for us in this class.

Stationary processes

The key to obtaining useful random processes is to introduce some degree of *order* into the joint distribution of all the random variables. Perhaps the most important and successful approach is the concept of *stationarity.*

Here, we select a positive integer K and then select an arbitrary set of K distinct integers $n_1, n_2, \ldots, n_K$. Then, we select an arbitrary integer M and we consider the two random vectors

$$(X[n_1], X[n_2], \ldots, X[n_K]) \tag{2.18}$$
$$(X[n_1 + M], X[n_2 + M], \ldots, X[n_K + M]). \tag{2.19}$$

If, for *all* choices of K, M and the tuple $n_1, n_2, \ldots, n_K$, these two random vectors have the same probability density function (or probability mass function, if X is discrete-valued), then we call the random process $\{X[n]\}_{n=-\infty}^{\infty}$ *stationary.*

Wide-sense stationary processes

The following class of random processes is even more important for our course. We define the following two quantities:

$$m_X[n] = \mathbb{E}[X[n]] \tag{2.20}$$
$$R_X[n,m] = \mathbb{E}[X[n]X[m]]. \tag{2.21}$$

Then, if the mean function $m_X[n]$ is constant (for the purpose of this course, we will always have $m_X[n] = 0$) and, more importantly, if $R_X[n,m]$ depends only on the *difference* $(m-n)$, then we call the random process $\{X[n]\}_{n=-\infty}^{\infty}$ *wide-sense stationary,* and we write the autocorrelation function as

$$R_X[k] = \mathbb{E}[X[n]X[n+k]]. \tag{2.22}$$

In this case, we can also define the important concept of *power spectral density:*

$$S_X(f) = \sum_{k=-\infty}^{\infty} R_X[k]e^{-j2\pi fk}, \tag{2.23}$$

and it will also turn out useful to look at the power spectral density in the Z-transform domain:

$$S_X(z) = \sum_{k=-\infty}^{\infty} R_X[k]z^{-k}. \tag{2.24}$$

In this class, we will also encounter *complex-valued* random processes. In the complex-valued case, the autocorrelation function is customarily defined as

$$R_X[k] = \mathbb{E}[X^*[n]X[n+k]], \tag{2.25}$$

i.e., the first argument is complex-conjugated.

White noise

An important special case of a wide-sense stationary process is the so-called white noise (or perhaps better, white process). It is a wide-sense stationary process for which

$$R_X[k] = \sigma_X^2\delta[k], \tag{2.26}$$

where σ_X^2 is a constant. The reason we use the symbol σ_X^2 is because by definition, $R_X[0] = \mathbb{E}[(X[n])^2]$, and if we assume a zero mean, then this is simply the variance of $X[n]$. For this process, we find that

$$S_X(f) = \sigma_X^2, \tag{2.27}$$

that is, a completely "flat" power spectral density.

Wide-sense stationary processes and LTI systems; the whitening filter

Consider a stable linear time-invariant (LTI) system with frequency response $G(f)$ (or, in z-transform domain, $G(z)$). If the input to the system is a wide-sense stationary process $X[n]$ with power spectral density $S_X(f)$ (or, in z-transform domain, $S_X(z)$), then the output random process $Y[n]$ is also wide-sense stationary and has power spectral density given by

$$S_Y(f) = |G(f)|^2 S_X(f), \tag{2.28}$$

$$S_Y(z) = G(z)G^*(1/z^*)S_X(z). \tag{2.29}$$

One important implication of this is that we can turn a wide-sense stationary random process $X[n]$ into a white process simply by passing it through an LTI system (a filter) whose frequency response has magnitude

$$|G(f)| = \frac{1}{\sqrt{S_X(f)}}, \tag{2.30}$$

if a stable system with this frequency response exists. This is called the *whitening filter.*

To find the whitening filter, it is instructive to consider the power spectrum in the z-domain. A fundamental theorem ensures that without essential loss of generality, it is possible to write this power spectrum in the following shape:

$$S_X(z) = \sigma_0^2 \left(\frac{B(z)}{A(z)}\right)\left(\frac{B^*(1/z^*)}{A^*(1/z^*)}\right), \tag{2.31}$$

where σ_0^2 is a constant, $B(z)$ is a monic polynomial $B(z) = 1 + b_1 z^{-1} + \ldots + b_q z^{-q}$ with the special property that all of the roots of this polynomial are inside the unit circle (i.e., have magnitude smaller than 1), and $A(z)$ is a monic polynomial $A(z) = 1 + a_1 z^{-1} + \ldots + a_p z^{-p}$ also with the special property that all of the roots of this polynomial are inside the unit circle. A method for finding the polynomials $A(z)$ and $B(z)$ will be discussed in the homework, but it should be clear that if such a representation can be found, a whitening filter is given simply by

$$G(z) = \frac{A(z)}{\sigma_0 B(z)}. \tag{2.32}$$

Moreover, since the roots of both polynomials are inside the unit circle, we are guaranteed that this $G(z)$ represents a system that is both *stable* and *causal.*

The Central Limit Theorem

The Central Limit Theorem states that when we add up many *independent* random variables and normalize the sum to have unit variance, then in the limit, the resulting random variable behaves like a Gaussian. This is often used in Communications to justify the assumption that noise processes behave like Gaussians since they tend to be the sum of the effects of many "independent" electrons. Of course, the last statement is only approximate and would need to be justified much more carefully, but we will not do this in our class.

Formally, the Central Limit Theorem can be stated as follows: Let $\{Y_k\}_{k=1}^n$ be a sequence of independent and identically distributed (real-valued) random variables with mean m and variance σ^2. Let

$$S_n = Y_1 + Y_2 + \cdots + Y_n. \tag{2.33}$$

Then, for every β,

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{S_n - nm}{\sigma\sqrt{n}} < \beta\right) = 1 - Q(\beta), \tag{2.34}$$

where the $Q(\cdot)-$function was defined in Equation (2.13).

We point out that more general Central Limit Theorems can be proved. In particular, while the independence of the random variables Y_k is rather important and can only be relaxed slightly, these random variables do not need to all have the same distribution, as long as they behave reasonably well (see e.g. [6, Section X.5]).

2.4.2 Continuous-time Random Processes

The extension of the above concepts to continuous time is at once trivial and technically rather challenging. That is, if we ignore potential technicalities, we can directly extend the above definitions to continuous time. However, to deal in full generality with continuous-time random processes would require a much more sophisticated mathematical machinery. For the purpose of this class, we will only deal with a single continuous-time random process, namely, the *white Gaussian noise,* in Section 3.2, and we will skip most of the technicalities. The interested reader may find a treatment e.g. in [5, Chapter 25].

2.5 Detection

2.5.1 Basic Concepts

A detection problem[1] has two basic components: An underlying discrete random variable, which we will call H, and an arbitrary random variable (or vector, or continuous-time signal) Y representing the observations made. You could think of H as representing the transmitted bit stream, and Y as representing the noisy output of our communication channel.

The main object of interest is the *joint distribution* of these two random variables. Since H is discrete-valued and Y is often continuous-valued (certainly for the purpose of this course), it is convenient to write this (without loss of generality) as

$$f_{Y|H}(y|h)p_H(h). \tag{2.35}$$

The probability mass function $p_H(h)$ is referred to as the *priors.*

[1]In our view of the detection problem, we assume that a full probabilistic model is defined. This is often referred to as *Bayesian detection,* and is the most relevant model for communication systems. For signal processing, there is also an alternative formulation where only the observations Y are modeled as random variables.

The goal of detection theory is to find good detectors. A detector is simply a function that takes as its input the noisy observation Y, and provides an estimate $\hat{H}(Y)$ of the data H. How should we judge the quality of a detector? In this class, we will exclusively consider the *error probability*, namely, the probability that the estimate $\hat{H}(Y)$ is *not* equal to the data H :

$$P_e \quad = \quad \mathbb{P}(\hat{H}(Y) \neq H). \tag{2.36}$$

The quest is to find the best detector, namely, the one that leads to the smallest error probability.

It turns out to be simple to give a general formula for the best detector. It is often called the *maximum a posteriori* (MAP) estimator and is given by the formula

$$\hat{H}_{\text{MAP}}(Y = y) \quad = \quad \arg\max_h p_{H|Y}(h|y). \tag{2.37}$$

This could be the end of the story. But we want to gain some further insight. From Bayes' rule, we can write

$$p_{H|Y}(h|y) \quad = \quad \frac{f_{Y|H}(y|h)p_H(h)}{f_Y(y)} = \frac{f_{Y|H}(y|h)p_H(h)}{\sum_{\tilde{h}} f_{Y|H}(y|\tilde{h})p_H(\tilde{h})}. \tag{2.38}$$

Note that for every pair (h, y), it is in principle a simple matter to calculate this.

We want to make a few interesting observations about the MAP estimator. First of all, we can observe that

$$\hat{H}_{\text{MAP}}(Y = y) \quad = \quad \arg\max_h f_{Y|H}(y|h)p_H(h). \tag{2.39}$$

This is because the denominator in Equation (2.38) does not depend on h, and thus has no influence on the maximization.

Second, in the special case where all realizations h of the data H are equally likely, we note that we can further simplify the MAP estimator. This simplified version is often referred to as the *maximum likelihood* (ML) estimator, and can be written as

$$\hat{H}_{\text{ML}}(Y = y) \quad = \quad \arg\max_h f_{Y|H}(y|h). \tag{2.40}$$

2.5.2 The Likelihood Ratio

In general, there is not much we can say about the MAP maximization problem in Equation (2.39); it all depends on the specific structure of the probability distributions. One intermediate formalization that turns out to be useful in many applications is the so-called *likelihood ratio.* To understand this concept, we think of the case where H has only two possible values: $h \in \{1, 2\}$. In this case, the MAP rule from Equation (2.39) will decide $\hat{H}(y) = 1$ for those y that satisfy

$$f_{Y|H}(y|h = 1)p_H(h = 1) \quad \geq \quad f_{Y|H}(y|h = 2)p_H(h = 2). \tag{2.41}$$

Now, define the likelihood ratio to be

$$\Lambda_{12}(Y=y) \quad = \quad \frac{f_{Y|H}(y|h=1)}{f_{Y|H}(y|h=2)}. \tag{2.42}$$

Then, we can observe that the MAP rule will decide $\hat{H}(y)=1$ for those y that satisfy

$$\Lambda_{12}(Y=y) \quad \geq \quad \frac{p_H(h=2)}{p_H(h=1)}. \tag{2.43}$$

In the important special case where $p_H(h=1)=p_H(h=2)=1/2$, the likelihood ratio test thus becomes $\Lambda_{12}(Y=y)\geq 1$. Moreover, since in many interesting probabilistic models, $f_{Y|H}(y|h)$ are from exponential families (such as Gaussian distributions), it is often good to take (natural) logarithms and consider instead the so-called *log-likelihood ratio:*

$$\mathrm{LLR}_{12}(Y=y) \quad = \quad \log\left(\frac{f_{Y|H}(y|h=1)}{f_{Y|H}(y|h=2)}\right). \tag{2.44}$$

Then, we can observe that the MAP rule will decide $\hat{H}(y)=1$ for those y that satisfy $\mathrm{LLR}_{12}(Y=y)\geq \log\frac{p_H(h=2)}{p_H(h=1)}$. In the important special case where $p_H(h=1)=p_H(h=2)=1/2$, the log-likelihood ratio test thus becomes $\mathrm{LLR}_{12}(Y=y)\geq 0$.

2.5.3 Calculation of the Error Probability

For a fixed detector $\hat{H}_0(Y=y)$, defined for every possible realization y, we would like to calculate the associated error probability:

$$P_e \quad = \quad \mathbb{P}(\hat{H}_0(Y)\neq H). \tag{2.45}$$

Conceptually, this is a straightforward exercise, but we want to make a few remarks. First, it is often convenient to consider the error probability for a particular realization h:

$$P_{e,h} \quad = \quad \mathbb{P}(\hat{H}_0(Y)\neq h|H=h). \tag{2.46}$$

Second, it is also convenient to introduce the *decoding regions* for each value h, that is, those values of y for which our detector decides for h. Formally:

$$\mathcal{D}_h \quad = \quad \{y : \hat{H}_0(Y)=h\}, \tag{2.47}$$

and we will also use the complement of this set, denoted by

$$\mathcal{D}_h^c \quad = \quad \{y : \hat{H}_0(Y)\neq h\}, \tag{2.48}$$

which is the set of all y that do *not* map back to h. Then, we can write

$$P_{e,h} \quad = \quad \int_{y\in\mathcal{D}_h^c} f_{Y|H}(y|h)dy. \tag{2.49}$$

Thus, we can express the total error probability as

$$P_e \quad = \quad \sum_h p_H(h)\int_{y\in\mathcal{D}_h^c} f_{Y|H}(y|h)dy. \tag{2.50}$$

2.5.4 Detection in Gaussian Probabilistic Models

In a Gaussian model, the conditional probability density functions $f_{Y|H}(y|h)$ are assumed to be Gaussian (for each value of h). For the scalar case, this can be expressed in full generality as:

$$f_{Y|H}(y|h) = \frac{1}{\sqrt{2\pi\sigma_h^2}} e^{-\frac{1}{2\sigma_h^2}(y-x_h)^2}, \tag{2.51}$$

where the mean x_h and the variance σ_h^2 may depend on the value of h.

It is useful to observe that we can write this same model also in an algebraic form as

$$Y = x_h + Z_h, \tag{2.52}$$

where Z_h is a Gaussian random variable of mean zero and variance σ_h^2.

In most communication applications, the noise variance will be constant $\sigma_h^2 \equiv \sigma^2$ (the variance of the background additive noise), and in this case, the ML detector takes an intuitively pleasing form:

$$\begin{aligned} \hat{H}_{\mathrm{ML}}(Y = y) &= \arg\max_h \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-x_h)^2} \qquad (2.53)\\ &= \arg\min_h \, (y - x_h)^2. \qquad (2.54) \end{aligned}$$

We will call this the *minimum distance detector*: The ML estimate is simply the one "signal point" x_h that is closest to the received signal y.

2.5.5 The example you should know by heart...

Consider $H \in \{1, 2\}, p_H(1) = p_H(2) = 1/2$ (uniform priors), and

$$f_{Y|H}(y|h) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-x_h)^2}. \tag{2.55}$$

As we have just seen, in this case, the optimum decoder is the minimum distance decoder, which means we simply slice halfway between x_1 and x_2. Assuming that $x_1 < x_2$, we can write this explicitly as

$$\hat{H}_{\mathrm{ML}}(Y = y) = \begin{cases} 1, & \text{if } y \le (x_1 + x_2)/2, \\ 2, & \text{if } y > (x_1 + x_2)/2. \end{cases} \tag{2.56}$$

That is, we can write the corresponding decoding regions as

$$\begin{aligned} \mathcal{D}_{h=1} &= \{y : y \le (x_1 + x_2)/2\}, \qquad (2.57)\\ \mathcal{D}_{h=2} &= \{y : y > (x_1 + x_2)/2\}. \qquad (2.58) \end{aligned}$$

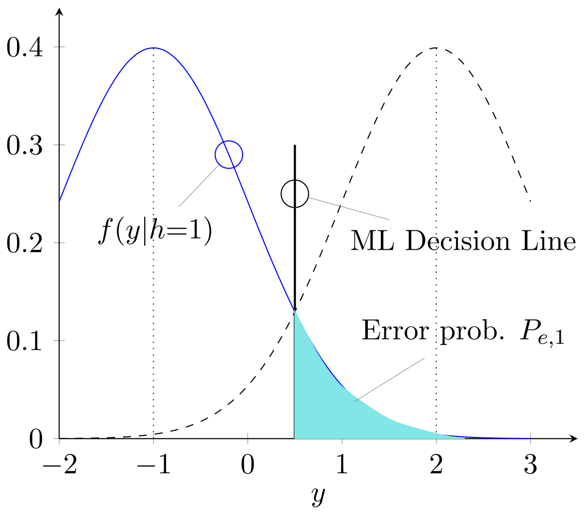


Figure 2.1: The one example you should know by heart. Here, $x_1 = -1$ and $x_2 = 2$, hence, the ML decision line is at $y = 0.5$.

To find the corresponding error probability, let us consider $H = 1$. We make an error when $y > (x_1 + x_2)/2$ under $f_{Y|H}(y|h = 1)$, which is

$$\begin{aligned} P_{e,1} &= \int_{(x_1+x_2)/2}^{\infty} f_{Y|H}(y|h = 1)dy && (2.59)\\ &= \int_{(x_1+x_2)/2}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-x_1)^2} dy && (2.60)\\ &= Q\left(\frac{d_{12}}{2\sqrt{\sigma^2}}\right), && (2.61) \end{aligned}$$

where $d_{12} = x_2 - x_1$ is the distance between the two message points, and σ^2 is the variance of the additive noise. By symmetry, you can show that $P_{e,2} = P_{e,1}$, and thus, the total error probability is

$$P_e = Q\left(\frac{d_{12}}{2\sqrt{\sigma^2}}\right). \qquad (2.62)$$

2.5.6 Sufficient Statistics

In many applications, the original observation Y is quite large and cumbersome. For example, think of a wireless channel: the original observation Y is the full voltage trace over all time at the receiving antenna. However, it often turns out that we can directly "throw away" most of the observation and only keep a reduced version Z of Y. Informally, if Z contains all the information about H that Y had (that is, if we did not throw away anything important), then we call Z a *sufficient statistic* for H given Y.

More formally, Z is called a *sufficient statistic* for H given Y if

$$f_{Y,Z|H}(y,z|h) \quad = \quad f_{Z|H}(z|h) f_{Y|Z}(y|z). \tag{2.63}$$

Dividing both sides by $f_{Z|H}(z|h)$, we can rewrite this condition equivalently as

$$f_{Y|Z,H}(y|z,h) \quad = \quad f_{Y|Z}(y|z). \tag{2.64}$$

We note that the sufficient statistic is not unique. Trivially, for example, Y itself is always a sufficient statistic. There is also a notion of *minimal sufficient statistic,* which refers to the case where Z does not contain any irrelevant data, but we will not discuss this further.

For the case of two hypotheses, one can show that the *likelihood ratio* is a sufficient statistic, i.e., we can select

$$Z \quad = \quad \Lambda_{12}(Y) = \frac{f_{Y|H}(Y|h=1)}{f_{Y|H}(Y|h=2)}. \tag{2.65}$$

In direct extension, this means that any "projection" of the original data Y that still permits to calculate the likelihood ratio is also a sufficient statistic.

The main theorem says that the best detection based on Z is just as good (in terms of error probability) as the best decision based on Y. In practical systems, it is often convenient to determine a simple sufficient statistic. The received signal is then first pre-processed into the sufficient statistic Z, from which we detect H. This can help find low-complexity implementations of optimum detection. We will see several important examples of sufficient statistics, including an orthogonal projection argument in Section 3.3.1, the Matched Filter in Section 3.3.2, and the *maximum ratio combiner* in the consideration of coherent detection over fading channels (Section 5.4.1).

Chapter 3

The AWGN Channel Model

One of the most fruitful models for the design of digital communication systems is the AWGN channel model.

3.1 The Channel Model

In this chapter, we consider the standard AWGN channel model, given by the following description:

$$Y(t) \quad = \quad x(t) + Z(t), \tag{3.1}$$

where $Z(t)$ is (real-valued) additive white Gaussian noise of power spectral density $N_0/2$.

3.2 Additive White Gaussian Noise

The definition of the noise process $Z(t)$ in the AWGN model is a little subtle. First, the easy part: The noise is called "Gaussian" because we assume that for an arbitrary collection of measurable functions $\{g_i(t)\}_{i=1}^N$, the random variables

$$Z_i \quad = \quad \int_{-\infty}^{\infty} Z(t)g_i(t)dt, \tag{3.2}$$

for $i = 1, 2, \ldots, N$, are *jointly Gaussian* random variables.

The slightly more tricky part follows now. We assume that the process $Z(t)$ satisfies

$$\begin{aligned} \mathbb{E}[Z(t)] &= 0 & (3.3)\\ R_Z(\tau) &= \mathbb{E}[Z(t)Z(t+\tau)] = \frac{N_0}{2}\delta(\tau). & (3.4) \end{aligned}$$

Clearly, there is a problem with the second expression since it is unbounded at $\tau = 0$. However, the formula will be good enough as a proxy in our applications. Nevertheless, we do encourage the interested student to go and consult the mathematically more sophisticated literature (e.g., [5, Chapter 25].

These assumptions directly imply the following important fact about the random variables in Equation (3.2):

$$\mathbb{E}[Z_i Z_j] = \frac{N_0}{2}\int_{-\infty}^{\infty} g_i(t)g_j(t)dt. \tag{3.5}$$

This is the key equation to remember about the noise.

3.3 Optimum Detection for Waveforms

Let us now suppose that the transmitted signal $x(t)$ is selected from the following M choices:

$$x_1(t), x_2(t)\ldots, x_M(t). \tag{3.6}$$

There are two versions of the sufficient statistic that both turn out to be important. The first works via an orthonormal basis expansion, and the second is referred to as the *matched filter.*

3.3.1 Sufficient Statistic, take 1: Orthonormal Basis Expansion

Let us first construct an orthonormal basis for our M signal waveforms:

$$\varphi_1(t), \varphi_2(t), \ldots, \varphi_N(t), \tag{3.7}$$

where we point out that $N \le M$, but the precise value of N depends on the dimensionality of the set of signal waveforms. This means that we can write, for $m = 1, 2, \ldots, M$,

$$x_m(t) = \sum_{n=1}^{N} \underbrace{\left(\int_{-\infty}^{\infty} x_m(u)\varphi_n(u)du\right)}_{x_{m,n}} \varphi_n(t), \tag{3.8}$$

and we will find it convenient to identify the waveform $x_m(t)$ with its corresponding vector of coefficients (or coordinates in the φ-basis) $\mathbf{x}_m = (x_{m,1}, x_{m,2}, \ldots, x_{m,N})^T$.

Then, we can prove from first principles (see Appendix 3.A) that the following is a sufficient statistic:

$$\begin{aligned} Y_1 &= \int_{-\infty}^{\infty} Y(t)\varphi_1(t)dt \\ Y_2 &= \int_{-\infty}^{\infty} Y(t)\varphi_2(t)dt \\ \vdots \; &\vdots \; \vdots \\ Y_N &= \int_{-\infty}^{\infty} Y(t)\varphi_N(t)dt. \end{aligned} \tag{3.9}$$

That is, instead of keeping the full continuous-time waveform $Y(t)$, we can simply retain these N real numbers and base all further processing on these. This is clearly a significantly simpler scenario.

Supposing that signal $x_m(t)$ was chosen at the transmitter, we can write:

$$\begin{aligned} Y_n &= \int_{-\infty}^{\infty} Y(t)\varphi_n(t)dt = \int_{-\infty}^{\infty} (x_m(t) + Z(t))\varphi_n(t)dt \\ &= \underbrace{\int_{-\infty}^{\infty} x_m(t)\varphi_n(t)dt}_{=x_{m,n}} + \underbrace{\int_{-\infty}^{\infty} Z(t)\varphi_n(t)dt}_{=Z_n}, \end{aligned} \tag{3.10}$$

where the noises $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_N)^T$ are independent zero-mean Gaussian of variance $N_0/2$, and each transmitted waveform $x_m(t)$ is characterized by the vector of length N given by $\mathbf{x}_m = (x_{m,1}, x_{m,2}, \ldots, x_{m,N})^T$. Thus, when the input signal $x(t)$ can only assume one of M different signals, the original waveform problem of Equation (3.1) is fully captured by the following vector problem:

$$\mathbf{Y} = \mathbf{x}_m + \mathbf{Z}. \tag{3.11}$$

3.3.2 Sufficient Statistic, take 2: The Matched Filter

Alternatively, we can retain the following:

$$\begin{aligned} U_1 &= \int_{-\infty}^{\infty} Y(t)x_1(t)dt \\ U_2 &= \int_{-\infty}^{\infty} Y(t)x_2(t)dt \\ \vdots &\ \vdots\ \vdots \\ U_M &= \int_{-\infty}^{\infty} Y(t)x_M(t)dt. \end{aligned} \tag{3.12}$$

Again, one can prove that $\mathbf{U} = (U_1, U_2, \ldots, U_M)$ is a sufficient statistic for the detection problem. For example, this can be shown by relating to the sufficient statistic given in Equation (3.9), as we will do below in Equation (3.22). This version is often referred to as the *matched filter*: we are "matching" the received waveform $Y(t)$ against each of the possible transmitted waveforms.

Supposing that signal $x_m(t)$ was chosen at the transmitter, we can write:

$$\begin{aligned} U_j &= \int_{-\infty}^{\infty} Y(t)x_j(t)dt = \int_{-\infty}^{\infty} (x_m(t) + Z(t))x_j(t)dt \\ &= \int_{-\infty}^{\infty} x_m(t)x_j(t)dt + \underbrace{\int_{-\infty}^{\infty} Z(t)x_j(t)dt}_{=V_j}, \end{aligned} \tag{3.13}$$

where it is important to note that the noises V_j are still Gaussian, but they are *not* independent of each other. Therefore, in order to find an expression for the ML detector or to calculate error probabilities, the orthonormal basis expansion ("take 1" of the sufficient statistic) is often easier to deal with. But it is still instructive to express the vector $\mathbf{U}$ in the following shape:

$$\mathbf{U} = A\mathbf{x}_m + \mathbf{V}, \tag{3.14}$$

where A is the $M \times N$ matrix whose rows are given by $\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_M^T$, and the vector $\mathbf{V}$ is additive Gaussian noise of mean zero and covariance matrix $\frac{N_0}{2}AA^T$. To derive this formula, it is convenient to first realize that we can rewrite:

$$\int_{-\infty}^{\infty} x_m(t)x_j(t)dt = \langle \mathbf{x}_m, \mathbf{x}_j \rangle. \tag{3.15}$$

Along the same lines, one can also prove that

$$U_j = \langle \mathbf{Y}, \mathbf{x}_j \rangle, \tag{3.16}$$

where $\mathbf{Y}$ is the vector with components given in Equation (3.10).

3.4 AWGN Vector Channels

We have seen that when the input signal $x(t)$ can only assume one of M different signals, the waveform problem of Equation (3.1) reduces to the vector problem

$$\mathbf{Y} = \mathbf{x} + \mathbf{Z}, \tag{3.17}$$

where all vectors are of length N, the vector $\mathbf{x}$ is known to be selected uniformly from a given set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, and the noise vector $\mathbf{Z}$ is a vector of independent, zero-mean Gaussian random variables of variance $N_0/2$.

3.4.1 Optimal Detection

For such AWGN vector channels, it is very simple to find the ML detector. Note that we are assuming that all signal points are selected with equal probability, therefore, the ML is also the optimal (MAP) detector. To find the ML, we write out the conditional probability density function:

$$f_{\mathbf{Y}|H}(\mathbf{y}|h) = \frac{1}{(\pi N_0)^{N/2}} e^{-\frac{1}{N_0}\|\mathbf{y}-\mathbf{x}_h\|^2}, \tag{3.18}$$

where $h \in \{1, 2, \ldots, M\}$. Then,

$$H_{\mathrm{ML}}(\mathbf{Y} = \mathbf{y}) = \arg\max_h \frac{1}{(\pi N_0)^{N/2}} e^{-\frac{1}{N_0}\|\mathbf{y}-\mathbf{x}_h\|^2} \tag{3.19}$$

$$= \arg\min_h \|\mathbf{y} - \mathbf{x}_h\|^2. \tag{3.20}$$

Thus, amongst our M signal points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, we simply find the one that is closest to the received vector $\mathbf{y}$, which we call *minimum distance detection.*

It is also instructive to further develop this expression. Namely,

$$H_{\mathrm{ML}}(\mathbf{Y}=\mathbf{y}) = \arg\min_h \|\mathbf{y}\|^2 + \|\mathbf{x}_h\|^2 - 2\langle \mathbf{y}, \mathbf{x}_h\rangle \quad (3.21)$$

$$= \arg\max_h \ \langle \mathbf{y}, \mathbf{x}_h\rangle - \frac{1}{2}\|\mathbf{x}_h\|^2. \quad (3.22)$$

To understand the significance of this formula, note that $\|\mathbf{x}_h\|^2$ is just a constant, independent of the received signal $\mathbf{y}$. The key is the expression $\langle \mathbf{y}, \mathbf{x}_h\rangle$ which measures the *correlation* between the received signal and each of the possible transmitted waveforms. That is, with respect to the received signal, the only thing we need to keep to make optimal decisions is the vector $(\langle \mathbf{y}, \mathbf{x}_1\rangle, \langle \mathbf{y}, \mathbf{x}_2\rangle, \ldots, \langle \mathbf{y}, \mathbf{x}_M\rangle)$, which implies that this vector is a sufficient statistic. Note that this is exactly the vector $\mathbf{U}$ given in Equation (3.12), which shows that indeed, $\mathbf{U}$ is a sufficient statistic.

3.4.2 The one example you should know by heart...

Let us now suppose that $M = 2$, i.e., there are only two possible signal points $\{\mathbf{x}_1, \mathbf{x}_2\}$. We will allow the dimensionality N to be arbitrary. The optimal detector is minimum distance, thus we will choose $\mathbf{x}_1$ if

$$\|\mathbf{y} - \mathbf{x}_1\|^2 \ \le\ \|\mathbf{y} - \mathbf{x}_2\|^2. \quad (3.23)$$

Expanding this, we can rewrite:

$$\|\mathbf{y}\|^2 + \|\mathbf{x}_1\|^2 - 2\langle \mathbf{y}, \mathbf{x}_1\rangle \ \le\ \|\mathbf{y}\|^2 + \|\mathbf{x}_2\|^2 - 2\langle \mathbf{y}, \mathbf{x}_2\rangle, \quad (3.24)$$

or, equivalently,

$$\langle \mathbf{y}, \mathbf{x}_2 - \mathbf{x}_1\rangle \ \le\ \frac{\|\mathbf{x}_2\|^2 - \|\mathbf{x}_1\|^2}{2}. \quad (3.25)$$

Thus, this formula proves that $\langle \mathbf{y}, \mathbf{x}_2 - \mathbf{x}_1\rangle$ is a sufficient statistic for h given $\mathbf{y}$. This is intuitive: $\langle \mathbf{y}, \mathbf{x}_2 - \mathbf{x}_1\rangle$ is just the (unnormalized) projection of $\mathbf{y}$ into the direction $\mathbf{x}_2 - \mathbf{x}_1$, meaning that all that matters for our decision is the contribution of $\mathbf{y}$ in the direction of the line connecting the two message points $\mathbf{x}_1$ and $\mathbf{x}_2$. All other components of $\mathbf{y}$ are orthogonal to this direction and thus do not matter for the decision.

The corresponding error probability can easily be found to be

$$P_e = Q\left(\frac{d_{12}}{2\sqrt{N_0/2}}\right), \quad (3.26)$$

where $d_{12} = \|\mathbf{x}_1 - \mathbf{x}_2\|$ is the distance between the two message points, and $N_0/2$ is the variance of the noise.

3.4.3 The Signal-to-Noise Ratio (SNR)

Communication engineers often find it convenient to express the error probability as a function of the so-called signal-to-noise ratio. The average signal power is $\mathbb{E}\left[\int_{-\infty}^{\infty} x^2(t)dt\right]$, which can be written in terms of the message point representation of the waveforms as

$$\mathcal{E} = \frac{1}{M}\sum_{m=1}^{M} \|\mathbf{x}_m\|^2. \tag{3.27}$$

The noise power is $N_0/2$, and thus, we define the *signal-to-noise ratio* as

$$\gamma = \frac{\mathcal{E}}{N_0/2}. \tag{3.28}$$

In order to compare different communication strategies, it is often interesting to consider the *energy-per-bit.* With M different signals, we can represent $\log_2 M$ bits, hence, the energy-per-bit is

$$\mathcal{E}_b = \frac{\mathcal{E}}{\log_2 M}. \tag{3.29}$$

Communications engineers then often like to express their results in terms of $\mathcal{E}_b/N_0$, which is pronounced “Ebno.”

As an example, consider the signal set $\{\mathbf{x}_1 = \sqrt{\mathcal{E}}, \mathbf{x}_2 = -\sqrt{\mathcal{E}}\}$. In this case, the average signal power is precisely $\mathcal{E}$, and we have $d_{12} = 2\sqrt{\mathcal{E}}$. Therefore, we can rewrite Equation (3.26) as

$$P_e = Q\left(\frac{d_{12}}{2\sqrt{N_0/2}}\right) = Q\left(\frac{2\sqrt{\mathcal{E}}}{2\sqrt{N_0/2}}\right) = Q(\sqrt{\gamma}). \tag{3.30}$$

Alternatively, we can also express this in terms of the “Ebno.” Since in this example, we only transmit 1 bit, we simply obtain

$$P_e = Q(\sqrt{2\,\mathcal{E}_b/N_0}). \tag{3.31}$$

3.4.4 The general Gaussian vector channel

Finally, we also mention the general Gaussian vector problem, defined as

$$\mathbf{U} = A\mathbf{x} + \mathbf{V}, \tag{3.32}$$

where A is an arbitrary $K \times N$ matrix, $\mathbf{x}$ is a vector of length N selected uniformly from a given set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, and the noise vector $\mathbf{V}$ is a vector of length K of jointly Gaussian random variables of mean zero and arbitrary covariance matrix Σ_V. Such a model arises for example in Equation (3.14).

There are many tricks that one can play with such models. Let us first assume that the matrix Σ_V is a full-rank matrix. (Note that this is not generally the case for the model in

Equation (3.14).) To understand optimum detection, a useful approach is to first render the noise white. By this, we mean that we define

$$\mathbf{S} \quad = \quad B\mathbf{U} = BA\mathbf{x} + B\mathbf{V}, \tag{3.33}$$

where we select the $K \times K$ full-rank matrix B in such a way that the new equivalent noise vector $\mathbf{W} = B\mathbf{V}$ is a vector of *independent and identically distributed* Gaussian random variables of mean zero and variance $N_0/2$ (meaning that the equivalent noise $\mathbf{W}$ is white Gaussian noise). One can show that such a matrix B exists when Σ_V is a full-rank matrix.

With this, we can rewrite our probabilistic model as

$$\mathbf{S} \quad = \quad \tilde{\mathbf{x}} + \mathbf{W}, \tag{3.34}$$

where all vectors are of length K, the vector $\tilde{\mathbf{x}}$ is known to be selected uniformly from the given set $\{BA\mathbf{x}_1, BA\mathbf{x}_2, \ldots, BA\mathbf{x}_M\}$, and the noise vector $\mathbf{W}$ is a vector of independent, zero-mean Gaussian random variables of variance $N_0/2$. That is, we have reduced the general model to the model in Equation (3.17), which we know how to solve.

The case where the matrix Σ_V is *not* full rank is left as an exercise.

3.5 Bounds on the Error Probability for Many Messages

So far, we have analyzed the case of two messages only. Let us now return to the original setting and analyze the case of M messages. That is, we consider again

$$\mathbf{Y} \quad = \quad \mathbf{x} + \mathbf{Z}, \tag{3.35}$$

where all vectors are of length N, the vector $\mathbf{x}$ is known to be selected uniformly from a given set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, and the noise vector $\mathbf{Z}$ is a vector of independent, zero-mean Gaussian random variables of variance $N_0/2$.

The error probability can be expressed as follows:

$$P_e \quad = \quad \frac{1}{M}\sum_{m=1}^{M} \mathbb{P}(E_m), \tag{3.36}$$

where E_m denotes the event that a different message was detected, given that message m was transmitted, and thus, an error occurred. Unfortunately, it is generally difficult to determine this probability exactly. Instead, we now derive an upper bound to the error probability. The trick is to split this event up into a *union* of "smaller" events:

$$E_m \quad = \quad \bigcup_{m'=1, m'\neq m}^{M} E_{m,m'}, \tag{3.37}$$

where now, $E_{m,m'}$ denotes the event that message m' was detected, given that message m was transmitted. Clearly, we have

$$P_e \quad = \quad \frac{1}{M}\sum_{m=1}^{M} \mathbb{P}\left(\bigcup_{m'=1, m'\neq m}^{M} E_{m,m'} \right). \tag{3.38}$$

Now, we can apply the *union bound* to get the following upper bound:

$$P_e \;\leq\; \frac{1}{M}\sum_{m=1}^{M}\sum_{m'=1,m'\neq m}^{M}\mathbb{P}(E_{m,m'}). \tag{3.39}$$

Finally, we observe that

$$\mathbb{P}(E_{m,m'}) \;=\; Q\left(\frac{d_{mm'}}{2\sqrt{N_0/2}}\right), \tag{3.40}$$

where $d_{mm'} = \|\mathbf{x}_m - \mathbf{x}_{m'}\|$ is the distance between the two message points, and $N_0/2$ is the variance of the noise. Thus, we can write our upper bound on the error probability as

$$P_e \;\leq\; \frac{1}{M}\sum_{m=1}^{M}\sum_{m'=1,m'\neq m}^{M} Q\left(\frac{d_{mm'}}{2\sqrt{N_0/2}}\right). \tag{3.41}$$

One simplification that leads to a weaker, but often still interesting bound, is to define the *minimum distance* of our signal constellation as

$$d_{min} \;=\; \min_{m\neq m'}\|\mathbf{x}_m - \mathbf{x}_{m'}\|. \tag{3.42}$$

Then, we get the following upper bound:

$$P_e \;<\; (M-1)Q\left(\frac{d_{min}}{2\sqrt{N_0/2}}\right). \tag{3.43}$$

3.6 Constellation Design

Throughout this section, we assumed that the transmitted signal $x(t)$ is selected from M choices:

$$x_1(t), x_2(t)\ldots, x_M(t), \tag{3.44}$$

or, equivalently, from M signal constellations points:

$$\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}. \tag{3.45}$$

Every constellation can be characterized by four fundamental parameters:

1. The number of signals M;
2. The dimensionality N of the signal set (which essentially corresponds to the bandwidth required by the given signal set);
3. The average energy $\mathcal{E}$ needed:

$$\mathcal{E} \;=\; \frac{1}{M}\sum_{m=1}^{M}\left(\int_{-\infty}^{\infty} x_m^2(t)dt\right) = \frac{1}{M}\sum_{m=1}^{M}\|\mathbf{x}_m\|^2; \tag{3.46}$$

4. The average error probability incurred.

In line with this perspective, a considerable part of the communications literature of past decades has attempted to solve the following problem: Fix M, N, and $\mathcal{E}$ and find the signal constellation (that is, the signal set) that minimizes the incurred average probability of error.

In this class, we will only briefly touch upon this problem. The main issue is that contemporary communication systems use *error-correcting codes.* Therefore, the "raw" error probability of the signal constellation is not by itself the dominant feature; a much more involved optimization would have to be carried out. However, for reasons of implementation complexity (since the constellation shaping requires analog electronics), most systems use simple constellations and put all the optimization and cleverness into the design of the error-correcting code.

3.7 Behavior when the number of messages becomes large

Let us now study a very particular choice of message points, namely

$$\begin{aligned}
\mathbf{x}_1 &= \sqrt{\mathcal{E}}(1,0,0,0,\ldots 0) \\
\mathbf{x}_2 &= \sqrt{\mathcal{E}}(0,1,0,0,\ldots 0) \\
\mathbf{x}_3 &= \sqrt{\mathcal{E}}(0,0,1,0,\ldots 0) \\
\mathbf{x}_4 &= \sqrt{\mathcal{E}}(0,0,0,1,\ldots 0) \\
\vdots & \quad \vdots \quad \vdots \\
\mathbf{x}_M &= \sqrt{\mathcal{E}}(0,0,0,0,\ldots 1).
\end{aligned} \tag{3.47}$$

This is referred to as *orthogonal signaling* — since all signal points are orthogonal to each other. Clearly, the dimensionality of this signal set is $N = M$. A basic analysis of the error probability of this signal set (assuming uniform priors) is easy. Simply note that the distance between any two signal points is $\sqrt{2\mathcal{E}}$. Thus, using Equation (3.43), the error probability can be upper bounded by

$$P_e \;<\; (M-1)Q\left(\frac{\sqrt{2\mathcal{E}}}{2\sqrt{N_0/2}}\right). \tag{3.48}$$

To gain insight, let us further upper bound the Q-function by $Q(x) \leq e^{-x^2/2}$, and thus

$$P_e \;<\; (M-1)e^{-\frac{\mathcal{E}}{2N_0}}. \tag{3.49}$$

To rewrite this expression, let us define the *energy per bit.* We are transmitting a total energy of $\mathcal{E}$ and communicating $\log_2 M$ bits. Thus, the energy per bit is $\mathcal{E}_b = \mathcal{E}/\log_2 M$. Plugging in, we can write

$$P_e \;<\; (M-1)e^{-\frac{\log_2 M}{2}\mathcal{E}_b/N_0}, \tag{3.50}$$

where we have used the "Ebno." Now, since we can write $M = 2^{\log_2 M} = e^{(\ln 2)\log_2 M}$,

$$P_e \quad < \quad e^{(\ln 2)\log_2 M} e^{-\frac{\log_2 M}{2}\mathcal{E}_b/N_0}, \tag{3.51}$$

and thus, finally,

$$P_e \quad < \quad e^{-\frac{1}{2}\log_2 M(\mathcal{E}_b/N_0 - 2\ln 2)}. \tag{3.52}$$

The astonishing insight is that as soon as our energy per bit investment is large enough so that the "Ebno" satisfies $\mathcal{E}_b/N_0 > 2\ln 2$, we can make the resulting error probability as small as we want simply by increasing $\log_2 M$, the number of bits that we transmit! Such behavior is definitely surprising and needs to be digested thoroughly.

Appendix 3.A Proof of Sufficient Statistic (take 1)

Formal proofs of this fact can be found throughout the literature. One of the best sources for formal arguments in this context is [5].

We here choose to ignore the mathematical subtleties and focus only on the main line of the proof. In particular, consider the orthonormal basis

$$\varphi_1(t), \varphi_2(t), \ldots, \varphi_N(t), \tag{3.53}$$

and suppose that we extend this basis:

$$\varphi_1(t), \varphi_2(t), \ldots, \varphi_N(t), \varphi_{N+1}(t), \varphi_{N+2}(t), \ldots, \tag{3.54}$$

indefinitely.

Without a proof, we are going to make the following leap of faith: The noise waveform can be represented in this (infinite-dimensional) basis, namely,

$$Z(t) \quad = \quad \sum_{k=1}^{\infty} Z_k \varphi_k(t), \tag{3.55}$$

where, as always, we have to set

$$Z_k \quad = \quad \langle Z(t), \varphi_k(t) \rangle = \int_{-\infty}^{\infty} Z(t)\varphi_k(t)dt. \tag{3.56}$$

We should point out that this leap of faith does depend on our noise process $Z(t)$ being sufficiently well behaved, and we also point out that the claimed equality signs are valid only in a certain sense. However, this "certain sense" is good enough for what we want to show.

By contrast, what does *not* require any leap of faith is that the noise variables Z_k are all IID — they are independent zero-mean Gaussians of variance $N_0/2$.

Let us now turn to the received waveform $Y(t)$. The leap of faith we made about the noise waveform implies that we can represent the received waveform $Y(t)$ in this (infinite-dimensional) basis, namely,

$$Y(t) \quad = \quad \sum_{k=1}^{\infty} Y_k \varphi_k(t), \tag{3.57}$$

where, again

$$Y_k \quad = \quad \langle Y(t), \varphi_k(t) \rangle = \int_{-\infty}^{\infty} Y(t)\varphi_k(t)dt. \tag{3.58}$$

Thus, from here onwards, we can work with Y_k, for $k = 1, 2, \ldots, N, N+1, N+2, \ldots$: It is trivial to see that these are a sufficient statistic since (under our leap of faith) they uniquely specify the full waveform $Y(t)$. (But recall that there are infinitely many Y_k.)

Now, supposing that $x_m(t)$ was transmitted, we can write

$$\begin{aligned} Y_k &= \langle Y(t), \varphi_k(t)\rangle = \int_{-\infty}^{\infty} (x_m(t) + Z(t))\varphi_k(t)dt && (3.59)\\ &= \int_{-\infty}^{\infty} x_m(t)\varphi_k(t)dt + \int_{-\infty}^{\infty} Z(t))\varphi_k(t)dt && (3.60)\\ &= \int_{-\infty}^{\infty} x_m(t)\varphi_k(t)dt + Z_k. && (3.61)\end{aligned}$$

Now, by definition, for $k = 1, 2, \ldots, N$, we have

$$\int_{-\infty}^{\infty} x_m(t)\varphi_k(t)dt = x_{m,k}. \tag{3.62}$$

However, since all the basis vectors $\varphi_{N+1}(t), \varphi_{N+2}(t), \ldots$ are orthogonal to the space in which the transmitted signals live, we have

$$\int_{-\infty}^{\infty} x_m(t)\varphi_k(t)dt = 0, \tag{3.63}$$

for $k = N + 1, N + 2, \ldots$ So, we can write out explicitly:

$$\begin{aligned} Y_1 &= x_{m,1} + Z_1\\ Y_2 &= x_{m,2} + Z_2\\ \vdots &\ \vdots\ \vdots\\ Y_N &= x_{m,N} + Z_N\\ Y_{N+1} &= Z_{N+1}\\ Y_{N+2} &= Z_{N+2}\\ \vdots &\ \vdots\ \vdots \end{aligned}$$

It is very important to recall that the noises Z_k are *independent* of each other. But then it is trivial to observe that the samples $Y_{N+1}, Y_{N+2}, \ldots$ do not contain any relevant information — they contain only noise, independent of everything else. This proves that $(Y_1, Y_2, \ldots, Y_N)$ is indeed a sufficient statistic for our problem.

Chapter 4

The Bandlimited AWGN Channel Model

4.1 The Channel Model

In this chapter, we consider an important generalization of the standard AWGN channel model, given by the following description:

$$Y(t) \quad = \quad \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau + Z(t), \tag{4.1}$$

where $Z(t)$ is (real-valued) additive white Gaussian noise of power spectral density $N_0/2$, and $h(t)$ is a stable filter (meaning that $\int |h(t)|dt < \infty$).

Throughout this chapter, we assume that the channel input signal is of the following form:

$$x(t) \quad = \quad \sum_{k=-\infty}^{\infty} I[k]g(t-kT), \tag{4.2}$$

where:

- $I[k]$ will be referred to as the information symbols; each $I[k]$ takes values in a finite set, a typical example being binary antipodal modulation, meaning that $I[k] = \pm\sqrt{\mathcal{E}_s}$. Typically, we assume that $\{I[k]\}_k$ is a sequence of independent and identically distributed random variables.
- $g(t)$ is a real-valued function called the "pulse shape," to be designed carefully. It typically needs to respect spectral constraints. Its choice is usually a trade-off between performance and implementation complexity.
- T is a positive real number representing the interval between consecutive pulses.

Then, we can express the channel output in the following fashion:

$$Y(t) \quad = \quad \sum_{k=-\infty}^{\infty} I[k]c(t-kT) + Z(t), \tag{4.3}$$

where

$$c(t) \;=\; (g * h)(t) = \int_{-\infty}^{\infty} g(\tau)h(t-\tau)d\tau. \tag{4.4}$$

4.2 A Discrete-time Equivalent

The most important feature of this channel is that it has an insightful discrete-time equivalent. To develop this, we first preprocess the received signal $Y(t)$ to obtain:

$$U(t) \;=\; \int_{-\infty}^{\infty} \tilde{c}(\tau)Y(t-\tau)d\tau, \tag{4.5}$$

where $\tilde{c}(t) = c(-t)$, that is, $\tilde{c}(t)$ is the matched filter for the signal $c(t)$. The key observation is that the samples of $U(t)$ taken at intervals T, let us call them $U[n] = U(nT)$, are a sufficient statistic for the information-carrying signal $I[n]$, given the full original channel output $Y(t)$ (see Appendix 4.A). But we can write those samples as

$$U[n] \;=\; \sum_{k=-\infty}^{\infty} I[n-k]d[k] + V[n], \tag{4.6}$$

$$\;=\; d[0]I[n] + \underbrace{\sum_{k=-\infty, k\neq 0}^{\infty} I[n-k]d[k]}_{\text{ISI}} + V[n], \tag{4.7}$$

where the function $d[k]$ characterizes the *inter-symbol interference* (ISI), that is, the mixing of the information symbols $I[n]$ corresponding to different time instants. The ingredients of Equation (4.6) are the following:

- The function $d[k]$ is given by

  $$d[k] \;=\; (\tilde{c} * c)(kT), \tag{4.8}$$

  where we are using the shorthand

  $$(\tilde{c} * c)(t) \;=\; \int_{-\infty}^{\infty} \tilde{c}(\tau)c(t-\tau)d\tau. \tag{4.9}$$

  Note that the function $(\tilde{c} * c)(t)$ is symmetric in the sense $(\tilde{c} * c)(t) = (\tilde{c} * c)(-t)$.

- It is important to realize that in Equation (4.6), the additive noise $V[n]$ is *not* white. That is, subsequent noise symbols are not independent of each other. However, $V[n]$ is still Gaussian and wide-sense stationary, thus uniquely specified by its mean (which is zero) and its autocorrelation function

  $$R_V[k] \;=\; \frac{N_0}{2}d[-k]. \tag{4.10}$$

4.3 Optimum Detection

To understand optimum detection for this channel, it is a good idea to first *filter* the (discrete-time) signal $U[n]$ such as to make the noise white. Let us denote the impulse response of the (noise) whitening filter by $d_W[k]$. The corresponding output signal can be expressed as

$$\begin{aligned} S[n] &= \sum_{k=-\infty}^{\infty} d_W[k]U[n-k] \\ &= \sum_{k=-\infty}^{\infty} I[n-k]f[k] + W[n], \end{aligned} \tag{4.11}$$

where now, the noise $W[n]$ is additive white Gaussian noise and has variance $N_0/2$. The new equivalent filter $f[k]$ is simply the convolution of $d[k]$ with the noise whitening filter $d_W[k]$.

Now, for each realization of the information-carrying signal $I[n]$, consider the resulting sequence $\sum_{k=-\infty}^{\infty} I[n-k]f[k]$. The received signal is the sum of one of these sequences and white Gaussian noise. But we know from Chapter 3 that the ML detector under white Gaussian noise is simply the minimum (Euclidean) distance decoder. That is, the ML detector will decide for the one information sequence $\{\hat{I}[n]\}_n$ that minimizes

$$\min_{\{\hat{I}[n]\}_n} \sum_n \left(S[n] - \sum_{k=-\infty}^{\infty} \hat{I}[n-k]f[k] \right)^2 . \tag{4.12}$$

This is often referred to as the *Maximum Likelihood Sequence Estimator* (MLSE).

Owing to the convolutional structure of the sum $\sum_{k=-\infty}^{\infty} I[n-k]f[k]$, it is possible to evaluate this minimum sequentially over time n, thus reducing the complexity. This is referred to as the Viterbi algorithm. It has linear complexity in time n, but still exponential complexity as a function of the depth of the ISI (i.e., the distance, in number of taps, between the furthest-spaced non-zero taps in $f[n]$).

4.4 Avoiding ISI

By careful design of the transmitted pulse shape $g(t)$, it is possible to avoid the problem of ISI altogether. In particular, it is clear that if $d[k] = 0$ for $k \neq 0$, then Equation (4.6) simplifies to $U[n] = d[0]I[n] + V[n]$, where $V[n]$ is simply additive white Gaussian noise. This is precisely the channel model that we have studied in the previous chapter.

So, why bother? For several reasons. First, for many channels of interest, the pulse shape that avoids ISI turns out to be essentially not implementable — something like a sinc function, for example. Second, in many communication problems, we may not know the precise shape of the channel impulse response at the transmitter, and thus, cannot perfectly choose the pulse shape. Third, the channel impulse response may change over time. Thus, we would have to update the pulse shape used by, say, the cell phone or the

DSL modem, every so often. But to implement such a reconfigurable pulse shape generator is very expensive, if it is even possible.

Thus, for all practical purposes, we have to design receiver architectures that can deal with some amount of ISI — banking on completely avoiding ISI would be much too optimistic.

4.5 Suboptimum Detection

4.5.1 Zero-forcing Equalization

Moving away from optimum detection, a first tempting approach is to simply remove all of the ISI. This is often referred to as *zero-forcing:* Forcing the ISI to be zero. More precisely, let us take the signal $U[n]$ from Equation (4.6) and filter it by the *inverse* of the filter $d[k]$. Clearly, this leads to a new signal of the form

$$\hat{I}_{ZF}[n] \quad = \quad I[n] + \tilde{V}[n], \tag{4.13}$$

where the noise $\tilde{V}[n]$ is *not* white. The idea here is that we decode $I[n]$ using only one received sample, namely $\hat{I}_{ZF}[n]$. It is clear that this is suboptimal: The noise is not white, which means that it can be predicted (to some accuracy) from its past. However, the appeal of this approach is its really low complexity. Its main drawback is that the zero-forcing filter generally amplifies the noise.

4.5.2 LMMSE Equalization

Reconsidering the approach taken in the previous section, we realize that the error probability will be governed by the equivalent per-sample noise, by which we mean that we filter the signal $U[n]$ to obtain

$$\hat{I}_{LMMSE}[n] \quad = \quad \sum_{k=-\infty}^{\infty} a[k]U[n-k], \tag{4.14}$$

and we take a decision concerning the information symbol $I[n]$ based on $\hat{I}_{LMMSE}[n]$. Thus, the effective noise[1] that affects our decision is $\hat{I}_{LMMSE}[n] - I[n]$, and its variance is given by

$$\mathbb{E}\left[\left(\hat{I}_{LMMSE}[n] - I[n]\right)^2\right]. \tag{4.15}$$

[1]We point out that this effective noise is *not* Gaussian, and thus, strictly speaking, the variance alone does not determine the resulting error probability. However, it is customary to pretend that the noise is Gaussian — and in an order-of-magnitude sense, this is often appropriate. First of all, we can use the Central Limit Theorem to argue that we expect the noise to be close to Gaussian. More interestingly, as long as the noise is additive and independent of the signal, in an information-theoretic sense (meaning that we assume that optimal codes are used), for fixed variance, the worst-case distribution is indeed Gaussian. A second issue is that the effective noise is generally *not independent* of the signal. Again, however, we can argue that it is “almost” independent.

Now, we want to select the filter $a[k]$ such as to *minimize* the equivalent noise variance in Equation (4.15). This is what gives this method its name: Linear Minimum Mean-Squared Error (LMMSE) equalization. To solve the problem, we can simply take derivatives of the expression in Equation (4.15) with respect to each of the filter coefficients $a[k]$ and set all these derivatives to zero.

The solution can be found in an elegant (and intuitively pleasing) two-step procedure. In the first step, we can show that all derivatives are zero if and only if

$$\mathbb{E}\left[\left(\hat{I}_{LMMSE}[n] - I[n]\right) U[n-m]\right] = 0, \quad \text{for } m = -\infty, \ldots, \infty. \tag{4.16}$$

This is often referred to as the *orthogonality principle:* If the coefficients $a[k]$ are optimally chosen in the mean-squared error sense (Equation (4.15)), the resulting error $(\hat{I}_{LMMSE}[n] - I[n])$ must be orthogonal to all of the data samples $U[k]$ that were used in order to produce the estimate $\hat{I}_{LMMSE}[n]$. In the second step, we merely plug in from Equation (4.14) to obtain

$$\sum_{k=-\infty}^{\infty} a[k]\mathbb{E}\left[U[n-k]U[n-m]\right] = \mathbb{E}\left[I[n]U[n-m]\right], \quad \text{for } m = -\infty, \ldots, \infty. \tag{4.17}$$

From now on, let us assume that the information symbols $I[n]$ are a wide-sense stationary random process with autocorrelation function denoted by $R_I[k]$ and power spectral density denoted by $S_I(z)$. (For example, they could be independent and identically distributed random variables of mean zero and variance $\mathcal{E}$, in which case $R_I[k] = \mathcal{E}\delta[k]$ and $S_I(z) = \mathcal{E}$.) Combining with Equation (4.6), we can then evaluate

$$\begin{aligned}
\mathbb{E}\left[I[n]U[n-m]\right] &= \mathbb{E}\left[I[n]\left(\sum_{k=-\infty}^{\infty} I[n-m-k]d[k] + V[n-m]\right)\right] \\
&= \sum_{k=-\infty}^{\infty} d[k]\mathbb{E}\left[I[n]I[n-m-k]\right] + \mathbb{E}\left[I[n]V[n-m]\right] \\
&= \sum_{\ell=-\infty}^{\infty} d[-\ell]R_I[m-\ell], \qquad (4.18)
\end{aligned}$$

where, for the last step, we have substituted $\ell = -k$ and used the fact that $R_I[\cdot]$ is a symmetric function. Hence, all derivatives are zero if and only if

$$\sum_{k=-\infty}^{\infty} a[k]R_U[m-k] = \sum_{k=-\infty}^{\infty} \tilde{d}[k]R_I[m-k], \quad \text{for } m = -\infty, \ldots, \infty, \tag{4.19}$$

where $\tilde{d}[n] = d[-n]$. Transforming into the Z-domain, we can rewrite

$$A(z)S_U(z) = D(1/z)S_I(z). \tag{4.20}$$

We can find $S_U(z)$ using Equation (2.29) and the fact that all signals are real-valued (which implies that $D^*(1/z^*) = D(1/z)$), thus

$$A(z)\left(D(z)D(1/z)S_I(z) + \frac{N_0}{2}D(1/z)\right) = D(1/z)S_I(z). \tag{4.21}$$

Then, the solution to this problem can be expressed in the following simple formula:

$$A_{LMMSE}(z) \;=\; \frac{S_I(z)}{S_I(z)D(z)+\frac{N_0}{2}}. \tag{4.22}$$

Finally, consider the special case where the information symbols are independent and identically distributed random variables of mean zero and variance $\mathcal{E}$. In this case, $A_{LMMSE}(z) = \mathcal{E}/(\mathcal{E}D(z)+\frac{N_0}{2})$. Here, we observe that as the signal power $\mathcal{E}$ becomes large, this converges simply to $A_{LMMSE}(z) = 1/D(z)$, which is precisely the zero-forcing solution. Thus, one can think of the LMMSE as balancing between the noise and the ISI power.

4.5.3 Finite-order LMMSE Equalization

In applications, it might not be possible to implement the filter $D_{LMMSE}(z)$. In order to have control over the complexity of the filter, one can impose a constraint on its length (in number of taps) and thus consider

$$\hat{I}_{KLMMSE}[n] \;=\; \sum_{k=-K}^{K} a[k]U[n-k]. \tag{4.23}$$

The $2K+1$ taps of this filter are now selected to minimize the mean-squared error in Equation (4.15).

Again, taking derivatives leads to the same orthogonality principle,

$$\mathbb{E}\left[\left(\hat{I}_{LMMSE}[n]-I[n]\right)U[n-m]\right] \;=\; 0, \tag{4.24}$$

with the difference that this time, this only has to hold for $-K \le m \le K$. This means that the equivalent of Equation (4.19) now becomes

$$\sum_{k=-K}^{K} a[k]R_U[m-k] \;=\; \mathcal{E}d[-m], \quad \text{for } m=-K,\ldots,K. \tag{4.25}$$

These are $2K+1$ linear conditions on $2K+1$ variables. Collecting the variables into the vector $\mathbf{a} = (a[-K], a[-K+1], \ldots, a[K-1], a[K])^T$, it is instructive to write Equation (4.25) in matrix form as:

$$R_U\mathbf{a} \;=\; \mathbf{r}, \tag{4.26}$$

and thus, the optimal coefficient vector $\mathbf{a} = R_U^{-1}\mathbf{r}$ can be found by matrix inversion, provided that R_U is invertible. Here $\mathbf{r} = \mathcal{E}(d[K], d[K-1], \ldots, d[-K+1], d[-K])^T$, and the autocorrelation matrix R_U is given by

$$R_U \;=\; \begin{pmatrix} R_U[0] & R_U[-1] & R_U[-2] & \ldots & R_U[-2K+1] & R_U[-2K] \\ R_U[1] & R_U[0] & R_U[-1] & \ldots & R_U[-2K+2] & R_U[-2K+1] \\ R_U[2] & R_U[1] & R_U[0] & \ldots & R_U[-2K+3] & R_U[-2K+2] \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ R_U[2K-1] & R_U[2K-2] & R_U[2K-3] & \ldots & R_U[0] & R_U[-1] \\ R_U[2K] & R_U[2K-1] & R_U[2K-2] & \ldots & R_U[1] & R_U[0] \end{pmatrix} \tag{4.27}$$

This matrix is special in that along any diagonal, the entries are constant. Such a matrix is called a *Toeplitz* matrix. We also note that for real-valued processes, $R_U[k] = R_U[-k]$, and for complex-valued processes, $R_U[k] = R_U^*[-k]$. Hence, the matrix R_U is a *symmetric* Topelitz matrix.

4.5.4 Decision-feedback Equalization

To be completed for the next version.

4.6 Bandpass Signaling and Complex-Valued Channel Models

So far, we have considered the general case of band-limited signaling across the AWGN channel model, characterized by Equation (4.2). For the remainder of this chapter, we will further specialize this to the case of so-called *bandpass signaling,* which means that we are using two *base-band, band-limited* information-carrying signals, $x_I(t)$ and $x_Q(t)$, band-limited to $(-W, W)$, from which we form the transmitted waveform as

$$x(t) \quad = \quad x_I(t)\cos(2\pi f_c t) - x_Q(t)\sin(2\pi f_c t), \tag{4.28}$$

where we assume that $W \ll f_c$. We leave it as an exercise for the reader to prove that indeed, this equation represents a special case of the more general formula given in Equation (4.2) (but note that in this case, $I[n]$ cannot be chosen to be independent and identically distributed). It is convenient to write this as

$$x(t) \quad = \quad Re\left(x_b(t)e^{j2\pi f_c t}\right), \tag{4.29}$$

where we call $x_b(t)$ the complex base-band equivalent of the transmitted signal $x(t)$. Note that $x_b(t)$ is also band-limited to the frequency interval $(-W, W)$. It can be written as

$$x_b(t) \quad = \quad x_I(t) + jx_Q(t). \tag{4.30}$$

Let us now discuss the corresponding channel output signal. To this end, we will first *ignore* the noise. The noiseless channel output is given by

$$\begin{aligned}
Y(t) \quad &= \quad \int_{-\infty}^{\infty} h(\tau)Re\left(x_b(t-\tau)e^{j2\pi f_c(t-\tau)}\right)d\tau && (4.31)\\
&= \quad Re\left(\int_{-\infty}^{\infty} h(\tau)x_b(t-\tau)e^{j2\pi f_c(t-\tau)}d\tau\right) && (4.32)\\
&= \quad Re\left(\underbrace{\left(\int_{-\infty}^{\infty} h(\tau)e^{-j2\pi f_c\tau}x_b(t-\tau)d\tau\right)}_{=Y_b(t)} e^{j2\pi f_c t}\right). && (4.33)
\end{aligned}$$

With this formalism, we can now equivalently think of the channel as a *complex* channel, and consider

$$Y_b(t) \;=\; \int_{-\infty}^{\infty} \underbrace{h(\tau)e^{-j2\pi f_c \tau}}_{=h_c(\tau)} x_b(t-\tau)d\tau, \tag{4.34}$$

where $h_c(\tau)$ is the equivalent complex-valued channel.

Finally, we recall that both $x_b(t)$ and $Y_b(t)$ are band-limited to $(-W, W)$. This means that they are uniquely specified by their Nyquist samples, meaning that we arrive at the equivalent channel model

$$Y_b[n] \;=\; \sum_{k=-\infty}^{\infty} h_c[k]x_b[n-k]. \tag{4.35}$$

Let us now finally bring the noise back in to obtain (see Appendix 4.B):

$$Y_b[n] \;=\; \sum_{k=-\infty}^{\infty} h_c[k]x_b[n-k] + Z_b[n], \tag{4.36}$$

where $Z_b[n]$ is *circularly symmetric* complex-valued additive white Gaussian noise of variance N_0, which means that its real and imaginary parts are independent with mean zero and variance $N_0/2$. That is, plugging into Equation (2.16), we can write the pdf of this complex-valued random variable as

$$f(Re(z_b), Im(z_b)) \;=\; \frac{1}{\pi N_0} e^{-\frac{1}{N_0}(Re(z_b)^2 + Im(z_b)^2)}. \tag{4.37}$$

A more convenient way of writing the same is

$$f(z_b) \;=\; \frac{1}{\pi N_0} e^{-\frac{1}{N_0}|z_b|^2}. \tag{4.38}$$

We could now again proceed to discuss ML detection and the suboptimal approaches, with the only modification of having a complex-valued channel. We will not do this — our main motivation for introducing the complex-valued channel model is to have a nice discussion of OFDM. The interested student can find the complex-valued version of the LMMSE in Appendix 4.C.

4.7 An FFT Implementation of OFDM

We start with a simple example:[2]

$$y[n] \;=\; h_0 x[n] + h_1 x[n-1] + w[n]. \tag{4.39}$$

[2] In this subsection, we slightly deviate from our notational convention and write all time-domain signals in lower case so as to be able to distinguish from the DFT domain, as shown in the sequel.

The trick is to use a so-called *cyclic prefix.* That is, we transmit a block of N information symbols, which we will denote as $x_0, x_1, x_2, \ldots, x_{N-1}$, using *more* than N channel uses. For the example at hand, we use $N+1$ channel uses. Specifically, the FFT-OFDM system will transmit:

$$\begin{aligned} x[-1] &= x_{N-1} \\ x[0] &= x_0 \\ x[1] &= x_1 \\ \vdots \;&\vdots\; \vdots \\ x[N-1] &= x_{N-1}. \end{aligned} \tag{4.40}$$

The transmitted sample $x[-1]$ is referred to as the "cyclic prefix," and the whole sequence of $N+1$ transmitted symbols is referred to as one *OFDM symbol.*

What is the received signal? We can express it as follows:

$$\begin{aligned} y[-1] &= h_0 x_{N-1} + h_1 x[-2] + w[-1] \\ y[0] &= h_0 x_0 + h_1 x_{N-1} + w[0] \\ y[1] &= h_0 x_1 + h_1 x_0 + w[1] \\ \vdots \;&\vdots\; \vdots \\ y[N-1] &= h_0 x_{N-1} + h_1 x_{N-2} + w[N-1], \end{aligned} \tag{4.41}$$

which depends on the transmitted signal $x[-2]$. We have not specified this signal (actually, if you want, this signal would belong to the *previous* OFDM symbol). The key trick is to simply disregard ("throw away," if you prefer) the received sample $y[-1]$, which is the price we pay for this beautiful implementation. The remaining received samples can then be collected into the following nice matrix form:

$$\begin{pmatrix} y[0] \\ y[1] \\ y[2] \\ \vdots \\ y[N-2] \\ y[N-1] \end{pmatrix} = \begin{pmatrix} h_0 & 0 & 0 & \ldots & 0 & h_1 \\ h_1 & h_0 & 0 & \ldots & 0 & 0 \\ 0 & h_1 & h_0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & h_0 & 0 \\ 0 & 0 & 0 & \ldots & h_1 & h_0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-2} \\ x_{N-1} \end{pmatrix} + \begin{pmatrix} w[0] \\ w[1] \\ w[2] \\ \vdots \\ w[N-2] \\ w[N-1] \end{pmatrix}. \tag{4.42}$$

Let us introduce the N-dimensional Fourier matrix F_N using $\omega = e^{-j2\pi/N}$ as

$$F_N = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 & 1 \\ 1 & \omega & \omega^2 & \ldots & \omega^{N-2} & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \ldots & \omega^{2(N-2)} & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \omega^{N-2} & \omega^{2(N-2)} & \ldots & \omega^{(N-2)^2} & \omega^{(N-2)(N-1)} \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \ldots & \omega^{(N-2)(N-1)} & \omega^{(N-1)^2} \end{pmatrix}. \tag{4.43}$$

In Equation (4.42), we multiply both sides (from the left) by the N-dimensional Fourier matrix F_N. Moreover, letting $\mathbf{X} = (X_0, X_1, X_2, \dots, X_{N-1})^T$ be the DFT of our information vector $\mathbf{x} = (x_0, x_1, x_2, \dots, x_{N-1})^T$, we can write $\mathbf{x} = F_N^{-1}\mathbf{X}$. Thus, the equation now looks as follows:

$$\underbrace{F_N \begin{pmatrix} y[0] \\ y[1] \\ y[2] \\ \vdots \\ y[N-2] \\ y[N-1] \end{pmatrix}}_{\text{This is } \mathbf{Y}} = \underbrace{F_N \begin{pmatrix} h_0 & 0 & 0 & \dots & 0 & h_1 \\ h_1 & h_0 & 0 & \dots & 0 & 0 \\ 0 & h_1 & h_0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & h_0 & 0 \\ 0 & 0 & 0 & \dots & h_1 & h_0 \end{pmatrix} F_N^{-1}}_{\text{This is a diagonal matrix}} \mathbf{X} + \underbrace{F_N \begin{pmatrix} w[0] \\ w[1] \\ w[2] \\ \vdots \\ w[N-2] \\ w[N-1] \end{pmatrix}}_{\text{Still IID Gaussian}}. \tag{4.44}$$

We know that there is a diagonal matrix in the middle because circulant matrices are diagonalized by the Fourier matrix. Furthermore, we know that the equivalent new noise is still IID Gaussian because the Fourier matrix is unitary. Thus, we obtain the desired form:

$$\begin{pmatrix} Y_0 \\ Y_1 \\ Y_2 \\ \vdots \\ Y_{N-2} \\ Y_{N-1} \end{pmatrix} = \begin{pmatrix} H_0 & 0 & 0 & \dots & 0 & 0 \\ 0 & H_1 & 0 & \dots & 0 & 0 \\ 0 & 0 & H_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & H_{N-2} & 0 \\ 0 & 0 & 0 & \dots & 0 & H_{N-1} \end{pmatrix} \mathbf{X} + \begin{pmatrix} Z_0 \\ Z_1 \\ Z_2 \\ \vdots \\ Z_{N-2} \\ Z_{N-1} \end{pmatrix}. \tag{4.45}$$

The elements H_i on the diagonal are simply the eigenvalues of the original channel matrix:

$$\begin{pmatrix} h_0 & 0 & 0 & \dots & 0 & h_1 \\ h_1 & h_0 & 0 & \dots & 0 & 0 \\ 0 & h_1 & h_0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & h_0 & 0 \\ 0 & 0 & 0 & \dots & h_1 & h_0 \end{pmatrix}. \tag{4.46}$$

Because this matrix is circulant, we can write the diagonal elements H_i in closed form, for $m = 0, 1, 2, \dots, N-1$,

$$H_m = \sum_{\ell=0}^{N-1} h_\ell \omega^{\ell m}. \tag{4.47}$$

That is, through the FFT-OFDM implementation, we turn the original ISI channel into N parallel ISI-free channels, each of different quality:

$$Y_m[k] = H_m X_m[k] + Z_m[k], \tag{4.48}$$

for $m = 0, 1, 2, \ldots, N-1$, where we use the index k to denote the OFDM symbol. That is, each tick of k represents ($N +$ length of cyclic prefix) uses of the original channel.

We have discussed the example given in Equation (4.39), but it should be clear that the approach works for *any* channel of this type, as long as the filter $h[n]$ only has a finite number of "taps", namely, filter coefficients. To this end, let us introduce the term "depth of the ISI" as the total time span of the filter governing the ISI, expressed in number of time steps. For example, in Equation (4.39), the depth of the ISI is two; and for the filter with $h_0 = 1, h_2 = 0.5$ and $h_5 = 0.25$ (and all other values of h_n equal to zero), the depth of the ISI is 6. We then simply select an appropriate length of cyclic prefix to end up with a representation as in Equation (4.42), where the matrix is circulant.

Length of cyclic prefix. To end this discussion, let us mention that N is a completely free design parameter, but the cyclic prefix is not: if we want to obtain the beautiful circulant channel form given in Equation (4.42), we must use a cyclic prefix whose length is no smaller than the depth of the ISI, minus one. In our example of Equation (4.39), the depth of the ISI is 2, and we need a cyclic prefix of length at least 1. The price is that we will disregard ("throw away") as many received samples per OFDM symbol as the length of the cyclic prefix. Thus, the effective symbol rate of an FFT-OFDM system is

$$R = \frac{N}{N + \text{ length of cyclic prefix}} = \frac{N}{N + \text{ depth of the ISI } - 1}. \tag{4.49}$$

For fixed ISI, by selecting N large enough, one can make the symbol rate as close to 1 as desired, but at the expense of large delay — all ($N +$ length of cyclic prefix) symbols have to be transmitted (and received) before anything can be decoded.

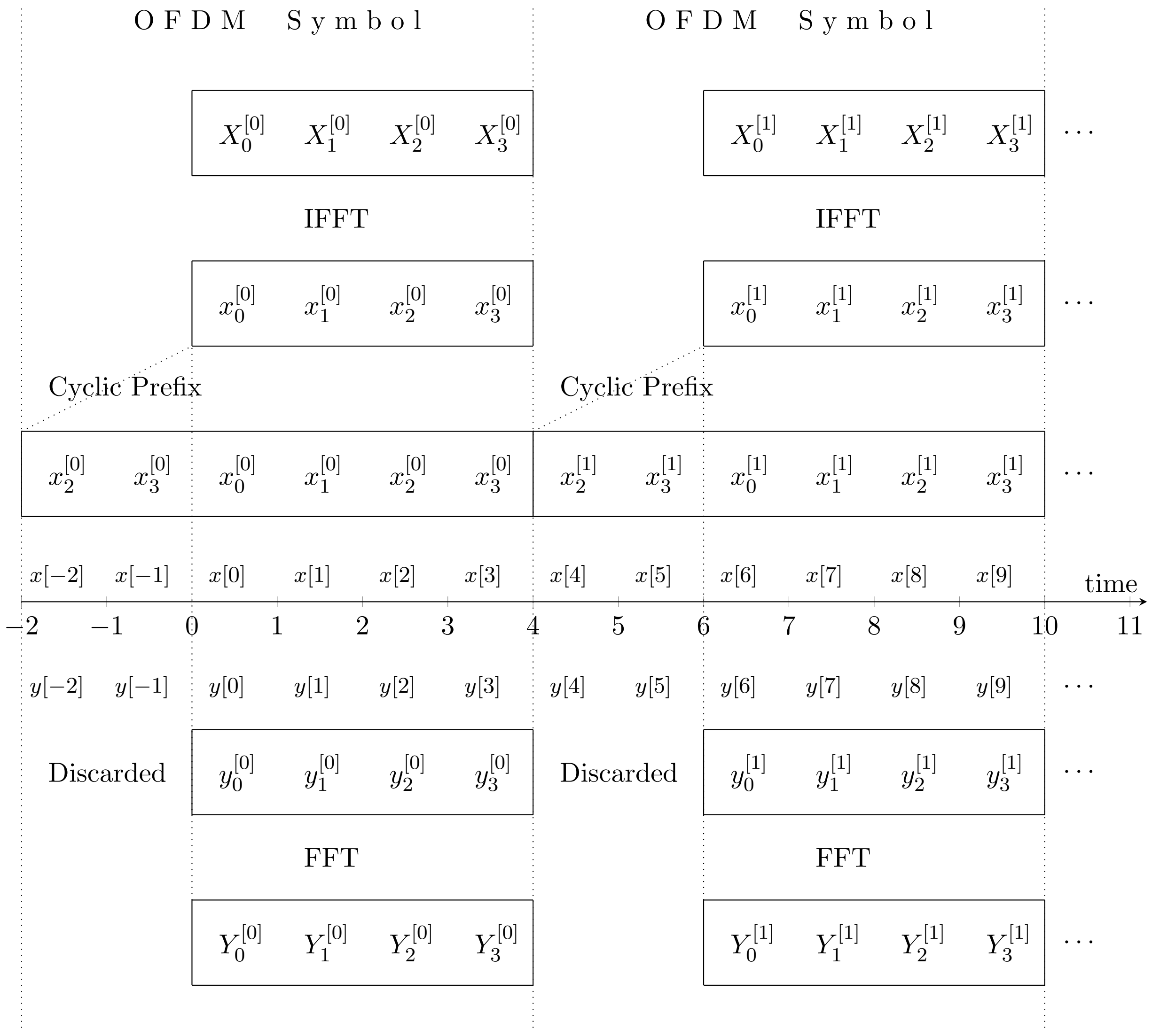
4.8 Resource Allocation in OFDM systems

Once we have removed ISI and obtained parallel channels, there is one key question left. To illustrate this issue, consider the running example of Section 4.7 and set $N = 2$ with $h_0 = 1$ and $h_1 = 0.5$. The FFT-OFDM approach will lead to two parallel channels with $H_0 = 1.5$ and $H_1 = 0.5$. But this means that at equal transmit power, the better channel has a signal-to-noise ratio that is 9 times higher than the worse channel!

Should we really assign equal power to both channels? And if we do, should we send more bits through the better channel? Using which signal constellation?

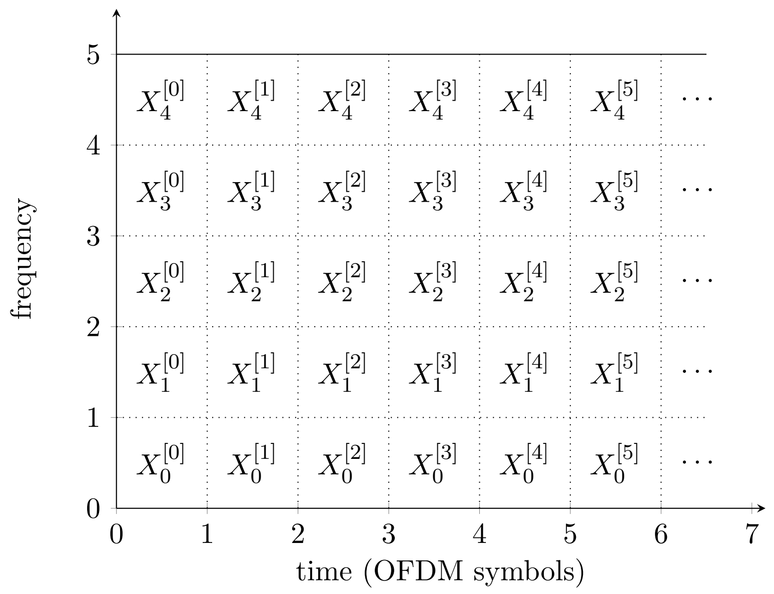
To answer this question, we need to specify what we want to optimize. Clearly, a first candidate would be the end-to-end error probability for the entire message (part of which is transmitted via the better channel and part via the worse channel). However, this problem does not appear to have an instructive solution as it crucially depends on exactly how we modulate in each of the parallel channels.

An alternative approach that has had significant impact on practical systems is via the concept of *capacity.* A formal discussion of this concept is beyond the scope of this class. However, the resulting allocation problem has a very interesting structure that appears in several other related problems and is of high importance in communication engineering.



OFDM Symbol
OFDM Symbol
$X_0^{[0]}$ $X_1^{[0]}$ $X_2^{[0]}$ $X_3^{[0]}$
$X_0^{[1]}$ $X_1^{[1]}$ $X_2^{[1]}$ $X_3^{[1]}$ ...
IFFT
IFFT
$x_0^{[0]}$ $x_1^{[0]}$ $x_2^{[0]}$ $x_3^{[0]}$
$x_0^{[1]}$ $x_1^{[1]}$ $x_2^{[1]}$ $x_3^{[1]}$ ...
Cyclic Prefix
Cyclic Prefix
$x_2^{[0]}$ $x_3^{[0]}$ $x_0^{[0]}$ $x_1^{[0]}$ $x_2^{[0]}$ $x_3^{[0]}$ $x_2^{[1]}$ $x_3^{[1]}$ $x_0^{[1]}$ $x_1^{[1]}$ $x_2^{[1]}$ $x_3^{[1]}$ ...
$x[-2]$ $x[-1]$ $x[0]$ $x[1]$ $x[2]$ $x[3]$ $x[4]$ $x[5]$ $x[6]$ $x[7]$ $x[8]$ $x[9]$
time
-2 -1 0 1 2 3 4 5 6 7 8 9 10 11
$y[-2]$ $y[-1]$ $y[0]$ $y[1]$ $y[2]$ $y[3]$ $y[4]$ $y[5]$ $y[6]$ $y[7]$ $y[8]$ $y[9]$ ...
Discarded
Discarded
$y_0^{[0]}$ $y_1^{[0]}$ $y_2^{[0]}$ $y_3^{[0]}$
$y_0^{[1]}$ $y_1^{[1]}$ $y_2^{[1]}$ $y_3^{[1]}$ ...
FFT
FFT
$Y_0^{[0]}$ $Y_1^{[0]}$ $Y_2^{[0]}$ $Y_3^{[0]}$
$Y_0^{[1]}$ $Y_1^{[1]}$ $Y_2^{[1]}$ $Y_3^{[1]}$ ...

Figure 4.1: FFT-OFDM implementation. Each tick on the time axis represents one channel use. In this example, $N = 4$ and the cyclic prefix is of length 2.



frequency
time (OFDM symbols)
0
1
2
3
4
5
6
7
$X_4^{[0]}$ $X_4^{[1]}$ $X_4^{[2]}$ $X_4^{[3]}$ $X_4^{[4]}$ $X_4^{[5]}$ ...
$X_3^{[0]}$ $X_3^{[1]}$ $X_3^{[2]}$ $X_3^{[3]}$ $X_3^{[4]}$ $X_3^{[5]}$ ...
$X_2^{[0]}$ $X_2^{[1]}$ $X_2^{[2]}$ $X_2^{[3]}$ $X_2^{[4]}$ $X_2^{[5]}$ ...
$X_1^{[0]}$ $X_1^{[1]}$ $X_1^{[2]}$ $X_1^{[3]}$ $X_1^{[4]}$ $X_1^{[5]}$ ...
$X_0^{[0]}$ $X_0^{[1]}$ $X_0^{[2]}$ $X_0^{[3]}$ $X_0^{[4]}$ $X_0^{[5]}$ ...

Figure 4.2: A time-frequency picture of the FFT-OFDM approach: Each tick on the time axis represents one OFDM symbol, meaning $N + L$ uses of the original underlying ISI channel, where $N = 5$ in the example shown, and L is the length of the cyclic prefix.

For a single (complex-valued) channel with gain H_i and noise level N_0, used with input power P_i, the so-called capacity is given by

$$C_i = \log_2\left(1 + \frac{|H_i|^2 P_i}{N_0}\right). \tag{4.50}$$

If we have K parallel channels, the total capacity is

$$\sum_{i=1}^{K} \log_2\left(1 + \frac{|H_i|^2 P_i}{N_0}\right). \tag{4.51}$$

The goal is now to allocate the total system power P in such a way to the K separate channels as to maximize the sum rate:

$$C_{total} = \max \sum_{i=1}^{K} \log_2\left(1 + \frac{|H_i|^2 P_i}{N_0}\right), \tag{4.52}$$

where the maximum is over all possible power allocations that satisfy $P_1+P_2+\ldots+P_K = P$. Clearly, we must observe $P_i \geq 0$.

To solve this problem, we find it useful to first rewrite it in "standard form." We will say that an optimization problem is in standard form if it is expressed as[3]

$$\begin{aligned} \text{minimize} \quad & f_0(x) && (4.53)\\ \text{subject to} \quad & f_i(x) \leq 0, \ \text{for } i = 1,2,\ldots,m, && (4.54)\\ & h_j(x) = 0, \ \text{for } j = 1,2,\ldots,p. && (4.55) \end{aligned}$$

Here, x is an n-dimensional vector, $f_0(x)$ is called the *objective* function, $f_i(x)$ are called the *inequality constraints* and $h_j(x)$ are called the *equality constraints*. The so-called *Lagrangian* for a problem in standard form is given as

$$L(x,\lambda,\nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{p} \nu_j h_j(x). \tag{4.56}$$

When the problem is convex (as it is in our case), the key advantage of the Lagrange formulation is that we have the KKT (Karush-Kuhn-Tucker) conditions, saying that the optimal solution is characterized by:[4]

$$\begin{aligned} \frac{\partial}{\partial x_\ell} L(x,\lambda,\nu) &= 0, \ \text{for all } \ell = 1,2,\ldots,n && (4.57)\\ f_i(x) &\leq 0, \ \text{for all } i = 1,2,\ldots,m && (4.58)\\ h_j(x) &= 0, \ \text{for all } j = 1,2,\ldots,p && (4.59)\\ \lambda_i &\geq 0, \ \text{for all } i = 1,2,\ldots,m && (4.60)\\ \lambda_i f_i(x) &= 0, \ \text{for all } i = 1,2,\ldots,m. && (4.61) \end{aligned}$$

[3]Here, we follow the terminology of the standard reference book: S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004, p.127.

[4]If you want to learn more, you may turn to Chapter 5 of S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004. Here, we are using p.243 ff.

If we apply this technique to our rate allocation problem, we can show that the KKT conditions imply

$$\begin{aligned} \text{Either } P_i \text{ satisfies:} \qquad & P_i + \frac{N_0}{|H_i|^2} = \text{ constant} && (4.62) \\ \text{or} \qquad & P_i = 0. && (4.63) \end{aligned}$$

This formula can be interpreted in an instructive way via a "water-filling" picture, which is left as an exercise.

Appendix 4.A Proof of Discrete-time Equivalent

In this appendix, we prove that for the band-limited AWGN channel model, if the input signal is of the form given in Equation (4.2), then we can equivalently express the channel in the shape given in Equation (4.6).

Looking at Equation (4.3), consider the following set of functions:

$$\psi_k(t) \;=\; \frac{1}{C} c(t-kT), \tag{4.64}$$

where $C^2 = \int |c(t)|^2 dt$ ensures that the functions $\psi_k(t)$ all have unit norm. In all but trivial cases, the functions $\psi_k(t)$ are linearly independent and thus constitute a basis of some space of functions, though *not* generally an orthogonal basis. We now complement this basis with additional basis functions, denoted by $\eta_\ell(t)$, for $\ell = 1, 2, 3, \ldots$. We assume that these basis functions are *orthogonal* to the space spanned by all the $\psi_k(t)$, that they are orthogonal with respect to each other and that they are normalized.

Next, we use the same leap of faith as in Equation (3.55): we assume that any waveform can be written as a linear combination of the $\psi_k(t)$ and the $\eta_\ell(t)$. This means that we can express the received waveform in Equation (4.3) as

$$Y(t) \;=\; \sum_{k=-\infty}^{\infty} \tilde{A}_k \psi_k(t) + \sum_{\ell=1}^{\infty} B_k \eta_\ell(t). \tag{4.65}$$

Because the $\eta_\ell(t)$ form an orthonormal basis and are orthogonal to all of the $\psi_k(t)$, we can express

$$B_k \;=\; \int_{-\infty}^{\infty} Y(t)\eta_\ell(t)dt. \tag{4.66}$$

The more difficult part is to find the coefficients $\tilde{A}_k$. For this, we need to understand a little more about non-orthogonal bases. We will use the notion of a *Riesz* pair of bases: it can be shown that there exists a set of functions

$$\tilde{\psi}_k(t) \tag{4.67}$$

which is *also* a basis for the space spanned by the $\psi_k(t)$ and which satisfies the following condition:

$$\int_{-\infty}^{\infty} \psi_n(t)\tilde{\psi}_k(t)dt \;=\; \begin{cases} 0, & \text{if } k \neq n, \\ 1, & \text{if } k = n. \end{cases} \tag{4.68}$$

Using this, we can now also represent the received waveform as

$$Y(t) \;=\; \sum_{k=-\infty}^{\infty} A_k \tilde{\psi}_k(t) + \sum_{\ell=1}^{\infty} B_k \eta_\ell(t), \tag{4.69}$$

and using the relation given in Equation (4.68), you can establish that

$$\begin{aligned} A_k &= \int_{-\infty}^{\infty} Y(\tau)\psi_k(\tau)d\tau = \int_{-\infty}^{\infty} \frac{1}{C}c(\tau - kT)Y(\tau)d\tau \\ &= \frac{1}{C}\int_{-\infty}^{\infty} \tilde{c}(\tau)Y(kT-\tau)d\tau. \end{aligned} \tag{4.70}$$

To conclude the proof, we show that the B_k are independent of everything else and thus constitute irrelevant information. Therefore, the A_k are a sufficient statistic. But the A_k are (up to the irrelevant constant scaling factor C) precisely the samples that we are retaining, as a comparison with Equation (4.5) reveals.

Appendix 4.B The complex base-band representation

In this appendix, we prove that for the band-limited AWGN channel model, if the input signal is of the form given in Equation (4.28), then we can equivalently express the channel in the shape given in Equation (4.36). Define

$$\varphi_k(t) = \gamma\,\mathrm{sinc}(t/T-k)\cos(2\pi f_c t) \quad \text{and} \quad \psi_k(t) = \gamma\,\mathrm{sinc}(t/T-k)\sin(2\pi f_c t), \tag{4.71}$$

where $T = 1/(2W)$ and γ is a constant chosen such that these functions have unit norm. We observe that $\{\varphi_k(t), \psi_k(t)\}_{k=-\infty}^{\infty}$ is an *orthonormal basis* for the space of functions whose spectrum is band-limited to the interval $(f_c - W, f_c + W)$ (and its negative complement). Since $x(t)$ is assumed to be band-limited exactly to this frequency interval, we can represent

$$x(t) = \sum_{k=-\infty}^{\infty} x_I[k]\varphi_k(t) - x_Q[k]\psi_k(t), \tag{4.72}$$

which alternatively can be understood simply as the sampling theorem. Next, we use the same leap of faith as in Equation (3.55): we complement our basis with many more basis vectors, orthogonal to all the basis vectors we have already chosen, in such a way as to obtain a basis for all waveforms. Let us call these additional basis elements $\eta_\ell(t)$. Then, we can represent the received waveform as

$$Y(t) = \sum_{n=-\infty}^{\infty} Y_I[n]\varphi_n(t) - Y_Q[n]\psi_n(t) + \sum_\ell C_\ell \eta_\ell(t). \tag{4.73}$$

Again, we can show that the random variables C_ℓ are independent of the transmitted signal and independent of $Y_I[n]$ and $Y_Q[n]$, so they represent irrelevant data and can be dropped from our consideration. The remaining terms can be rewritten as follows:

$$\tilde{Y}(t) = Re\left(\left(\underbrace{\sum_{n=-\infty}^{\infty} (Y_I[n] + jY_Q[n])\gamma\,\mathrm{sinc}(t/T-n)}_{=Y_b(t)}\right) e^{j2\pi f_c t}\right). \tag{4.74}$$

We point out that $\tilde{Y}(t)$ is simply $Y(t)$ passed through a bandpass filter that only retains the frequency band $(f_c - W, f_c + W)$ (and its negative complement). It should be intuitively clear that $\tilde{Y}(t)$ is a sufficient statistic since outside of this frequency band, there is only noise, and since the noise is white. Moreover, we observe that the filtered channel output $\tilde{Y}(t)$ is completely characterized by the (complex-valued) samples:

$$Y_b[n] \;=\; Y_I[n] + jY_Q[n] \tag{4.75}$$

The main bulk of the argument is now to calculate these two quantities and relate them to the channel input and noise. Because we are dealing with orthonormal bases, it is easy to get started:

$$\begin{aligned}
Y_I[n] &= \int_{-\infty}^{\infty} Y(t)\varphi_n(t)dt \\
&= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau + Z(t) \right) \varphi_n(t)dt \\
&= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau \right) \varphi_n(t)dt + \int_{-\infty}^{\infty} Z(t)\varphi_n(t)dt \\
&= \int_{-\infty}^{\infty} h(\tau) \left(\int_{-\infty}^{\infty} x(t-\tau)\varphi_n(t)dt \right) d\tau + Z_n^{\varphi} \\
&= \sum_{k=-\infty}^{\infty} x_I[k] \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)\varphi_k(t-\tau)\varphi_n(t)dtd\tau \right) \\
&\quad -x_Q[k] \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)\psi_k(t-\tau)\varphi_n(t)dtd\tau \right) + Z_n^{\varphi} \\
&= \sum_{k=-\infty}^{\infty} x_I[k]h_1[n-k] - x_Q[k]h_2[n-k] + Z_n^{\varphi}
\end{aligned} \tag{4.76}$$

where

$$\begin{aligned}
h_1[m] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)\varphi_0(t-\tau)\varphi_m(t)dtd\tau \\
h_2[m] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)\psi_0(t-\tau)\varphi_m(t)dtd\tau.
\end{aligned} \tag{4.77}$$

Similarly,

$$\begin{aligned}
Y_Q[n] &= -\int_{-\infty}^{\infty} Y(t)\psi_n(t)dt \\
&= -\int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau + Z(t)\right)\psi_n(t)dt \\
&= -\int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau\right)\psi_n(t)dt + \int_{-\infty}^{\infty} Z(t)\psi_n(t)dt \\
&= -\int_{-\infty}^{\infty} h(\tau)\left(\int_{-\infty}^{\infty} x(t-\tau)\psi_n(t)dt\right)d\tau + Z_n^{\psi} \\
&= -\sum_{k=-\infty}^{\infty} x_I[k]\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)\varphi_k(t-\tau)\psi_n(t)dtd\tau\right) \\
&\quad +x_Q[k]\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)\psi_k(t-\tau)\psi_n(t)dtd\tau\right) + Z_n^{\psi} \\
&= \sum_{k=-\infty}^{\infty} x_I[k]h_2[n-k] + x_Q[k]h_1[n-k] + Z_n^{\psi}. \qquad (4.78)
\end{aligned}$$

We point out that one technicality would deserve further scrutiny, namely, interchanging the order of the integrals and the summation. This is beyond the scope of the current course; it follows from the fact that $h(t)$ is stable and Fubini's theorem.

Finally, we find

$$Y_b[n] = \sum_{k=-\infty}^{\infty} \underbrace{(x_I[k] + jx_Q[k])}_{=x_b[k]}\underbrace{(h_1[n-k] + jh_2[n-k])}_{=h_c[n-k]} + Z_n^{\varphi} + jZ_n^{\psi}. \qquad (4.79)$$

The key to observe is that since the noise on our channel is Gaussian, the random variables Z_n^{φ} and Z_n^{ψ} are all independent for all n and have mean zero and variance $N_0/2$.

Appendix 4.C LMMSE Equalization for Complex-Valued Signals

In the complex-valued case, we have

$$U(t) = \int_{-\infty}^{\infty} \tilde{c}(\tau)Y(t-\tau)d\tau, \qquad (4.80)$$

where $\tilde{c}(t) = c^*(-t)$, that is, $\tilde{c}^*(t)$ is the matched filter for the signal $c(t)$.

As in Equation (4.6), we have

$$U[n] = \sum_{k=-\infty}^{\infty} I[n-k]d[k] + V[n], \qquad (4.81)$$

but we have to recalculate the autocorrelation function of the noise $V[n]$ for the complex-valued case. We find

$$R_V[k] = \frac{N_0}{2} d^*[-k]. \tag{4.82}$$

and hence,

$$S_V(z) = \frac{N_0}{2} D^*(1/z^*). \tag{4.83}$$

In the complex-valued case, the orthogonality principle takes the shape

$$\mathbb{E}\left[\left(\hat{I}_{LMMSE}[n] - I[n]\right) U^*[n-m]\right] = 0, \text{for } m = -\infty, \ldots, \infty. \tag{4.84}$$

Thus

$$\sum_{k=-\infty}^{\infty} a[k] \mathbb{E}\left[U[n-k]U^*[n-m]\right] = \mathbb{E}\left[I[n]U^*[n-m]\right], \text{for } m = -\infty, \ldots, \infty. \tag{4.85}$$

As in the real-valued case, let us assume that the information symbols $I[n]$ are from a (complex-valued) wide-sense stationary random process with autocorrelation function $R_I[k]$ and power spectrum $S_I(z)$. Combining with Equation (4.6), we can then evaluate

$$\begin{aligned}\mathbb{E}\left[I[n]U^*[n-m]\right] &= \mathbb{E}\left[I[n]\left(\sum_{k=-\infty}^{\infty} I^*[n-m-k]d^*[k] + V^*[n-m]\right)\right] \\ &= \sum_{k=-\infty}^{\infty} d^*[k]\mathbb{E}\left[I[n]I^*[n-m-k]\right] + \mathbb{E}\left[I[n]V^*[n-m]\right] \\ &= \sum_{\ell=-\infty}^{\infty} d^*[-\ell]R_I[m-\ell],\end{aligned} \tag{4.86}$$

where, for the last step, we have substituted $\ell = -k$. Hence, all derivatives are zero if and only if

$$\sum_{k=-\infty}^{\infty} a[k]R_U[m-k] = \sum_{\ell=-\infty}^{\infty} d^*[-\ell]R_I[m-\ell], \text{for } m = -\infty, \ldots, \infty. \tag{4.87}$$

Transforming into the Z-domain, we can rewrite

$$A(z)S_U(z) = D^*(1/z^*)S_I(z). \tag{4.88}$$

We can find $S_U(z)$ using Equation (2.29), thus

$$A(z)\left(D(z)D^*(1/z^*)S_I(z) + \frac{N_0}{2}D^*(1/z^*)\right) = D^*(1/z^*)S_I(z). \tag{4.89}$$

Hence, interestingly, the final solution does not change, and still takes the shape

$$A_{LMMSE}(z) = \frac{S_I(z)}{D(z)S_I(z) + \frac{N_0}{2}}. \tag{4.90}$$

Chapter 5

Wireless Communication

5.1 Physical Channel Models

The main effects are

- Free-space signal (amplitude) decay is as 1/distance.
- Reflections (include a phase shift).
- Linear superposition of many reflected versions, constructive and destructive interference, channel coherence (in space).
- Mobility and Doppler effect, channel coherence (in time and frequency).
- "Rural tragedy": with ideal reflections off a ground plane, energy decays like $1/d^4$, where d is the distance from the transmitter.

5.2 Statistical Channel Models

To model multi-path propagation from the wireless transmitter to the receiver, we assume that there are K paths:

$$Y(t) \quad = \quad \sum_{k=1}^{K} \alpha_k(t) x(t - d_k(t)) + Z(t), \tag{5.1}$$

where the sum is over all signal propagations paths, $\alpha_k(t)$ is the strength of path k at time t and $d_k(t)$ is the delay incurred along path k at time t. Finally, $Z(t)$ is additive white Gaussian noise.

5.2.1 Bandpass Signalling

We suppose a particular way of signaling across the wireless channel. Namely, we assume that we are using two *base-band, band-limited* information-carrying signals, $x_I(t)$ and $x_Q(t)$.

We suppose they are band-limited to the interval $(-W, W)$. From these, we form the transmitted waveform as

$$x(t) \quad = \quad x_I(t)\cos(2\pi f_c t) - x_Q(t)\sin(2\pi f_c t), \tag{5.2}$$

where we assume that $f_c \gg W$. It is convenient to write this as

$$x(t) \quad = \quad Re\left(x_b(t)e^{j2\pi f_c t}\right), \tag{5.3}$$

where we call $x_b(t) = x_I(t) + jx_Q(t)$ the *complex base-band equivalent* of the transmitted signal $x(t)$.

Plugging this into the wireless model of (5.1), and for now ignoring the additive noise, we can write

$$Y(t) \quad = \quad Re\left(e^{j2\pi f_c t}\underbrace{\sum_{k=1}^{K}\alpha_k(t)e^{-j2\pi f_c d_k(t)}x_b(t-d_k(t))}_{=Y_b(t)}\right), \tag{5.4}$$

where we call $Y_b(t)$ the complex base-band equivalent of the received signal $Y(t)$.

That is, we can now switch to considering only complex base-band equivalent signals, and obtain the following equivalent channel model:

$$Y_b(t) \quad = \quad \sum_{k=1}^{K}\alpha_k(t)e^{-j2\pi f_c d_k(t)}x_b(t-d_k(t)). \tag{5.5}$$

5.2.2 Complex Discrete-time Model

The complex baseband equivalent signal $x_b(t)$ is band-limited to $(-W, W)$. This does not imply that the (noiseless) received signal $Y_b(t)$ is also band-limited: the wireless channel model is a linear *time-varying* system and new frequencies can be "added" (Doppler effect). Nevertheless, in contemporary considerations, it is customary to assume that the received signal $Y(t)$ is first band-pass filtered and sampled. In general, this comes at a certain loss of optimality, but assuming that the Doppler shift is small compared to the signal bandwidth, the loss is negligible. Then, the discrete-time equivalent of the wireless channel takes the following shape:

$$Y[n] \quad = \quad \sum_{\ell=-\infty}^{\infty} H_n[\ell]x[n-\ell] + Z[n], \tag{5.6}$$

where $x[n]$ are the complex-valued (Nyquist) samples of the signal $x_b(t)$. The noise process $Z[n]$ is a sequence of independent and identically distributed (iid) circularly symmetric complex-valued Gaussian random variables of mean zero and variance N_0, which means that real and imaginary parts of $Z[n]$ are independent of each other and have variance $N_0/2$ each. For the case where the channel is not time-varying, we have derived the discrete-time representation in Appendix 4.B.

5.2.3 Channel Estimation

An obvious question is how we would know the precise characteristics of the channel at hand. In principle, one could attempt to identify the geometry of the scene, and hence, determine all K paths separately, along with their strength $\alpha_k(t)$ and delay $d_k(t)$. Knowing this, one could then calculate the channel coefficients $H_n[\ell]$ in Equation (5.6).

In practice, this procedure is very difficult to implement. Instead, one usually assumes that the channel coefficients $H_n[\ell]$ do not change too much over time n. Then, one can simply estimate them, for example by sending so-called *pilot signals,* which are known ahead of time to the receiver: if the receiver knows the transmitted signal $x[n]$ (at least for some portion of time) in Equation (5.6), it can use the channel outputs to estimate the channel coefficients $H_n[\ell]$. In this fashion, at the end of the *training period,* the receiver has knowledge of the channel and can use that knowledge in subsequent channel uses to decode the information-carrying signal (assuming that the channel coefficients do not change over time n). We will not study the problem of channel estimation in much detail in this class. Rather, we will consider the channel model given in Equation (5.6) and mostly assume that the channel coefficients are known precisely to the receiver.

5.3 Rayleigh Flat-Fading

To get started, we consider perhaps the simplest possible case of a fading channel, Rayleigh Flat-Fading (also called Rayleigh *frequency non-selective* fading):

$$Y[n] \;=\; H[0]x[n] + Z[n], \tag{5.7}$$

where $H[0]$ is a circularly-symmetric complex Gaussian random variable of unit variance (meaning that its real and imaginary parts are independent and each have variance $1/2$) and $Z[n]$ is circularly-symmetric complex additive white Gaussian noise whose real and imaginary parts each have variance $N_0/2$.

For notational convenience, we will simply write this as

$$Y[n] \;=\; Hx[n] + Z[n]. \tag{5.8}$$

It will be convenient to recall the following facts about the distribution of H :

$$\begin{aligned}
\mathbb{E}[H] &= \mathbb{E}[Re(H)] + j\mathbb{E}[Im(H)] = 0 && (5.9)\\
\mathbb{E}[|H|^2] &= \mathbb{E}[(Re(H))^2 + (Im(H))^2] = \mathbb{E}[(Re(H))^2] + \mathbb{E}[(Im(H))^2] = 1, && (5.10)
\end{aligned}$$

both of which can be easily proved by noting that $Re(H)$ and $Im(H)$ are independent Gaussian random variables of mean zero and variance $1/2$. Next, it is not hard to show that the random variable $R = |H|$ has a Rayleigh distribution:

$$f_R(r) \;=\; \begin{cases} 2re^{-r^2}, & \text{for } r \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.11}$$

Finally, perhaps the most useful fact is that the random variable $V = |H|^2$ is exponentially distributed:

$$f_V(v) = \begin{cases} e^{-v}, & \text{for } v \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.12}$$

5.3.1 Antipodal Signaling over Rayleigh Flat-Fading

Let us now assume that the transmitted signal is $\pm\sqrt{\mathcal{E}_c}$ (with uniform priors). Moreover, we assume *channel state information at the receiver (CSIR)*, meaning that the realization h of the fading state H is known to the receiver.

The next important (and simple) observation is that for a fixed fading state $H = h$, it is easy to find the error probability $P_e(h)$. First, we observe that without loss of optimality, we can start by removing the *phase* of the fading state, as follows:

$$\tilde{Y} = e^{-j\arg(h)}Y = e^{-j\arg(h)}hx[n] + e^{-j\arg(h)}Z[n] \tag{5.13}$$

$$= |h|x[n] + \tilde{Z}[n]. \tag{5.14}$$

Since we are only changing the phase, it is easy to show that $\tilde{Z}[n]$ is again circularly-symmetric complex additive white Gaussian noise whose real and imaginary parts each have variance $N_0/2$. Finally, note that the transmitted signal is real-valued and the noise has independent real and imaginary parts. Hence, the imaginary part of $\tilde{Y}$ is irrelevant information, and the real part of $\tilde{Y}$ is a sufficient statistic. Therefore, this is simply the standard problem of antipodal detection in Gaussian noise, where the two message points are now located at $\pm\sqrt{|h|^2\mathcal{E}_c}$. Thus, their distance is given by $d(h) = 2\sqrt{|h|^2\mathcal{E}_c}$, and the error probability can be written as (using Equation (3.26))

$$P_e(h) = Q\left(\frac{d(h)}{2\sqrt{N_0/2}}\right) = Q\left(\sqrt{\frac{|h|^2\mathcal{E}_c}{N_0/2}}\right). \tag{5.15}$$

To gain intuition for the behavior of Rayleigh fading, it is convenient to introduce the *effective signal-to-noise ratio for channel state h* as

$$\gamma = \frac{|h|^2\mathcal{E}_c}{N_0/2}. \tag{5.16}$$

Then, we can write the error probability as

$$P_e(\gamma) = Q\left(\sqrt{\gamma}\right). \tag{5.17}$$

Let us consider the *average* error probability, averaged over all fading states H :

$$\overline{P_e} = \mathbb{E}_H[P_e(H)] = \int_h P_e(h) f_H(h) dh. \tag{5.18}$$

It is convenient to switch coordinates and express:

$$\overline{P_e} = \int_0^\infty P_e(\gamma) f_\Gamma(\gamma) d\gamma. \tag{5.19}$$

To compute this, we have to find the pdf of the effective SNR, $f_\Gamma(\gamma)$. Clearly, since $\gamma = \frac{|h|^2\mathcal{E}_c}{N_0/2}$, and since the random variable $|H|^2$ is exponentially distributed, we know that the random variable Γ is also exponentially distributed, and thus,

$$f_\Gamma(\gamma) = \begin{cases} \frac{1}{\overline{\gamma}}e^{-\gamma/\overline{\gamma}}, & \text{for } \gamma \geq 0, \\ 0, & \text{otherwise,} \end{cases} \tag{5.20}$$

where $\overline{\gamma}$ denotes the *average* effective SNR:

$$\overline{\gamma} = \mathbb{E}[\Gamma] = \mathbb{E}[|H|^2]\frac{\mathcal{E}_c}{N_0/2} = \frac{\mathcal{E}_c}{N_0/2}. \tag{5.21}$$

With this in hand, we can write:

$$\overline{P_e} = \int_0^\infty Q\left(\sqrt{\gamma}\right)\frac{1}{\overline{\gamma}}e^{-\gamma/\overline{\gamma}}d\gamma. \tag{5.22}$$

Perhaps somewhat surprisingly, this integral can actually be solved in closed form! In the homework, you will show that

$$\overline{P_e} = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{\gamma}}{2+\overline{\gamma}}}\right). \tag{5.23}$$

It is interesting to consider how this expression behaves as we increase the transmit energy $\mathcal{E}_c$, meaning that the average SNR $\overline{\gamma}$ becomes large. To do so, we can study

$$\sqrt{\frac{\overline{\gamma}}{2+\overline{\gamma}}} = \sqrt{1 - \frac{2}{2+\overline{\gamma}}}. \tag{5.24}$$

Now, for small ϵ, we can use the approximation $\sqrt{1-\epsilon} \approx 1 - \frac{\epsilon}{2}$, thus we find

$$\overline{P_e} \approx \frac{1}{2\overline{\gamma}}. \tag{5.25}$$

This formula is rather disappointing because it says that the error probability only decays *inversely proportional* to the transmitted energy. By comparison, for any fixed AWGN channel, we saw earlier that the error probability decays *exponentially* with the transmitted energy.

Where does this disappointing behavior come from? There is an interesting observation concerning the intuitively pleasing concept of a *deep fade.* To this end, we think of $\mathcal{E}_c$ as being large, and we say that the channel is in a deep fade if the resulting effective SNR is small even though we made $\mathcal{E}_c$ large, meaning that the channel fading robbed us pretty much of all the power we invested. More formally, we consider the event where $\Gamma < 1$ to be a deep fade event. This means at a given $\mathcal{E}_c$, we are in a deep fade if

$$|h|^2 < \frac{N_0/2}{\mathcal{E}_c}. \tag{5.26}$$

Now, let us look at the *probability* (over the fading state, for fixed transmit energy) of ending up in a deep fade:

$$\mathbb{P}(\Gamma < 1) = \mathbb{P}\left(|H|^2 < \frac{N_0/2}{\mathcal{E}_c}\right) \tag{5.27}$$

$$= \int_0^{\frac{N_0/2}{\mathcal{E}_c}} f_V(v)dv, \tag{5.28}$$

where $V = |H|^2$ and $f_V(v)$ is given in Equation (5.12). But because the exponential distribution is essentially constant (and non-zero) for small values, we find that the probability of a deep fade event is

$$\mathbb{P}(\Gamma < 1) \approx \frac{N_0/2}{\mathcal{E}_c} = \frac{1}{\gamma}. \tag{5.29}$$

Finally, we would like to connect the deep fade event to the overall error probability. To do so, we can write (using E to denote the error event, and exploiting the law of total probability):

$$\begin{aligned}\overline{P_e} &= \mathbb{P}(E|\Gamma < 1)\mathbb{P}(\Gamma < 1) + \mathbb{P}(E|\Gamma \geq 1)\mathbb{P}(\Gamma \geq 1) \\ &\geq Q(1)\mathbb{P}(\Gamma < 1). \end{aligned} \tag{5.30}$$

This means that the error probability is lower bounded by the probability of a deep fade event (multiplied by the constant $Q(1)$); hence, these events are responsible for the disappointing behavior.

5.3.2 Orthogonal Signaling over Rayleigh Flat-Fading

Let us now consider a different way of signaling over this channel. This time, we consider *two* consecutive channel uses:

$$Y[0] = Hx[0] + Z[0] \tag{5.31}$$

$$Y[1] = Hx[1] + Z[1], \tag{5.32}$$

where all assumptions are as above, except the signaling: now, we assume orthogonal signaling, i.e., we transmit one of the following two signal vectors:

$$\begin{pmatrix} x[0] \\ x[1] \end{pmatrix} = \begin{pmatrix} \sqrt{\mathcal{E}_c} \\ 0 \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} x[0] \\ x[1] \end{pmatrix} = \begin{pmatrix} 0 \\ \sqrt{\mathcal{E}_c} \end{pmatrix}. \tag{5.33}$$

Note that in this case, we transmit real-valued signals only, but we receive two complex-valued signals.

First, for a fixed fading state $H = h$, we can again undo the phase term to obtain the two samples

$$\tilde{Y}[0] = |h|x[0] + \tilde{Z}[0] \tag{5.34}$$

$$\tilde{Y}[1] = |h|x[1] + \tilde{Z}[1], \tag{5.35}$$

where we now observe that it is a sufficient statistic to only keep the real parts — the imaginary parts only contain independent noise. Thus, we arrive at the model

$$\begin{aligned}\tilde{Y}_R[0] &= |h|x[0] + \tilde{Z}_R[0] &\quad (5.36)\\ \tilde{Y}_R[1] &= |h|x[1] + \tilde{Z}_R[1], &\quad (5.37)\end{aligned}$$

This is again just a binary hypothesis testing problem in Gaussian noise, and the two possible message points are given by

$$\begin{pmatrix} x[0] \\ x[1] \end{pmatrix} = \begin{pmatrix} \sqrt{|h|^2\mathcal{E}_c} \\ 0 \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} x[0] \\ x[1] \end{pmatrix} = \begin{pmatrix} 0 \\ \sqrt{|h|^2\mathcal{E}_c} \end{pmatrix}. \qquad (5.38)$$

The distance between these points is simply $d(h) = \sqrt{2}\sqrt{|h|^2\mathcal{E}_c}$, meaning that (using Equation (3.26))

$$P_e(h) = Q\left(\frac{d(h)}{2\sqrt{N_0/2}}\right) = Q\left(\sqrt{\frac{|h|^2\mathcal{E}_c}{N_0}}\right). \qquad (5.39)$$

Proceeding along similar steps as in the derivation of Equation (5.23), we find, not surprisingly,

$$\overline{P_e} = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{\gamma}}{4+\overline{\gamma}}}\right), \qquad (5.40)$$

showing exactly the same deep-fade behavior.

5.4 Diversity

5.4.1 The Basic Picture — Maximum Ratio Combining

Suppose that instead of just a single faded noisy observation, we have L *independently faded* noisy observations of one and the same information symbol x:

$$\begin{aligned} Y_1 &= H_1 x + Z_1 \\ Y_2 &= H_2 x + Z_2 \\ \vdots \;&\vdots\; \vdots \\ Y_L &= H_L x + Z_L, &\quad (5.41)\end{aligned}$$

where H_ℓ are independent circularly-symmetric complex Gaussian random variables of unit variance and Z_ℓ are independent circularly-symmetric complex additive white Gaussian noise, whose real and imaginary parts each have variance $N_0/2$.

The key observation is that irrespective of any assumptions about x, if the realizations of the fading coefficients are known to the decoder, then the following is a *sufficient statistic* for x given $Y_1, Y_2, \ldots, Y_L$:

$$Y = \sum_{\ell=1}^{L} h_\ell^* Y_\ell. \qquad (5.42)$$

This is called *maximum ratio combining* — we linearly combine the L received values with the best possible relative weights ("ratios").

To prove that Y is indeed a sufficient statistic, we simply write out the conditional probability density function of all channel outputs given the channel input, $f(y_1, y_2, \ldots, y_L|x)$, and notice that $Y_1, Y_2, \ldots, Y_L$ are conditionally independent given x (see Appendix 5.A).

That is, we can write

$$Y = \left(\sum_{\ell=1}^{L} |h_\ell|^2\right) x + \underbrace{\sum_{\ell=1}^{L} h_\ell^* Z_\ell}_{Z}, \tag{5.43}$$

where the equivalent noise Z is circularly symmetric complex-valued Gaussian with mean zero and variance $N_0 \sum_{\ell=1}^{L} |h_\ell|^2$.

Now, assuming binary antipodal modulation (of energy $\mathcal{E}_c$), for a fixed realization $(h_1, h_2, \ldots, h_L)$, the equivalent message points are located at $\pm(\sum_{\ell=1}^{L} |h_\ell|^2)\sqrt{\mathcal{E}_c}$, and thus, $d(h_1, h_2, \ldots, h_L) = 2(\sum_{\ell=1}^{L} |h_\ell|^2)\sqrt{\mathcal{E}_c}$. However, the noise is also modified: its variance is $(\sum_{\ell=1}^{L} |h_\ell|^2)N_0/2$. Thus, the error probability is (using Equation (3.26))

$$P_e(h_1, h_2, \ldots, h_L) = Q\left(\frac{d(h_1, h_2, \ldots, h_L)}{2\sqrt{\left(\sum_{\ell=1}^{L} |h_\ell|^2\right) N_0/2}}\right) \tag{5.44}$$

$$= Q\left(\sqrt{\frac{2\left(\sum_{\ell=1}^{L} |h_\ell|^2\right)\mathcal{E}_c}{N_0}}\right). \tag{5.45}$$

This needs to be averaged over the randomness in the fading process. Again, somewhat surprisingly, this can actually be solved in closed form. Eventually, we find the following formula for the error probability:

$$\overline{P_e}(\overline{\gamma}) = \left(\frac{1}{2}\left(1 - \sqrt{\frac{\overline{\gamma}}{2+\overline{\gamma}}}\right)\right)^L \sum_{\ell=0}^{L-1} \binom{L-1+\ell}{\ell} \left(\frac{1}{2}\left(1 + \sqrt{\frac{\overline{\gamma}}{2+\overline{\gamma}}}\right)\right)^\ell. \tag{5.46}$$

The dependence of this formula on $\overline{\gamma}$ can be made explicit (for large $\overline{\gamma}$) via the following approximation:

$$\overline{P_e}(\overline{\gamma}) \approx \left(\frac{1}{2\overline{\gamma}}\right)^L \binom{2L-1}{L}. \tag{5.47}$$

This is called *diversity of order L* because the error probability depends on the average SNR $\overline{\gamma}$ like $1/\overline{\gamma}^L$.

5.4.2 Formal Definition of Diversity

More formally, we can define the diversity order of a communication strategy as

$$d = -\lim_{\overline{\gamma}\to\infty} \frac{\log \overline{P_e}(\overline{\gamma})}{\log \overline{\gamma}}. \tag{5.48}$$

Thus, diversity captures the high-SNR behavior of a communication strategy. (This is not to say that high SNR is the only interesting operating regime!)

5.4.3 Sources of Diversity

As we have seen, diversity is obtained if we can observe one and the same signal multiple times, each time *independently faded.* The question is, why and when would we get to observe one and the same signal multiple times?

- *Diversity in time.* A first answer is to wait long enough: After a sufficient amount of time, the channel will have changed to a new fading state that is independent of the previous one. That is, the strategy would be to transmit the information symbol x at time 0, then wait long enough and transmit it again at a future time, again wait long enough and transmit again at a future time, and so on.

  The *coherence time* of the channel gives a certain basic measure of how long one has to wait to see an independent new channel. In the literature, the following two models are popular:

  – A simple abstraction is the so-called fast-fading channel:

  $$Y[n] = H_n[0]x[n] + Z[n], \tag{5.49}$$

  where we now assume that $\{H_n[0]\}_n$ is an IID sequence of circularly-symmetric complex Gaussian random variables of unit variance.

  – A more realistic model is to assume a so-called *block fading model.* We can write the channel again simply as:

  $$Y[n] = H_n[0]x[n] + Z[n], \tag{5.50}$$

  but the difference lies in the model according to which $H_n[0]$ changes as a function of n. Specifically, under the block fading model, we assume that $H_n[0]$ stays constant for a block of N transmissions, after which it *independently* assumes a new value, again drawn from a circularly-symmetric complex Gaussian distribution of unit variance.

  Beyond this, one could also assume that the process $\{H_n[0]\}_n$ is a Markov process (or similar), but in this class, we will not discuss such models to any level of detail.

- *Diversity in frequency.* Another way to obtain diversity is to use *multiple frequency bands* in such a way that each frequency band is independently faded.

- *Diversity in space.* Perhaps the most obvious source of diversity is to use multiple antennas at the receiver. If we use L antennas and assume that they are sufficiently far apart from each other so that the multi-path patterns are very different for each antenna, then we directly receive L independently faded copies, and thus, attain a diversity order of L. A rule of thumb found often in the literature stipulates that an antenna distance of at least a quarter or a half wavelength (of the carrier frequency) is sufficient to obtain independent fades (though the specifics depend on the multi-path propagation environment at hand).

5.5 Transmit Diversity and the Alamouti Trick

It is clear that L receive antennas give a diversity level of L (assuming independent fades on each receive antenna). What if we have L *transmit* antennas? Can we get diversity out of these?

Let us consider $L = 2$. The channel model is now given by

$$Y[n] \quad = \quad H_1 x_1[n] + H_2 x_2[n] + Z[n], \tag{5.51}$$

where we assume that H_1 and H_2 are independent circularly symmetric complex Gaussians with unit variance. The signal $x_1[n]$ is the signal transmitted from the first antenna and the signal $x_2[n]$ from the second antenna.

It should be immediately clear that we can get diversity order of 2 by *repetition coding.* To see this, denote the information symbol by u and let:

$$x_1[0] = u \quad \text{and} \quad x_2[0] = 0, \tag{5.52}$$
$$x_1[1] = 0 \quad \text{and} \quad x_2[1] = u. \tag{5.53}$$

This seems very wasteful: we are using both antennas *and* two time slots for just a single information symbol. To fix this, a famous trick due to Alamouti is to consider *two* information symbols, let us denote them by u_1 and u_2, and transmit

$$x_1[0] = u_1 \quad \text{and} \quad x_2[0] = u_2, \tag{5.54}$$
$$x_1[1] = -u_2^* \quad \text{and} \quad x_2[1] = u_1^*. \tag{5.55}$$

Analyzing ML detection reveals that we get diversity order 2 for *both* information symbols.

5.6 Non-coherent Detection

So far, we have only talked about the case where the channel fading state realization is known to the receiver (but not to the transmitter). In this section, we look at the case where the channel fading state realization is *completely unknown* to both the transmitter and the receiver. The receiver's operation in this case is often referred to as *non-coherent detection.*

First of all, it should be clear that antipodal signaling is now completely useless: Even if there is no noise, the channel output is statistically independent of the signal, because the phase of the channel state is uniform over the full interval from 0 to 2π. By contrast, one example of a signaling scheme that still works reasonably well even if we do not know the channel state at the receiver is the orthogonal signaling scheme analyzed in Section 5.3.2. You will study this in detail in the homework.

5.7 MIMO Communication

Let us consider a system with T transmit antennas and R receive antennas:

$$\begin{pmatrix} Y_1[n] \\ Y_2[n] \\ \vdots \\ Y_R[n] \end{pmatrix} = \begin{pmatrix} H_{11} & H_{12} & \dots & H_{1T} \\ H_{21} & H_{22} & \dots & H_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ H_{R1} & H_{R2} & \dots & H_{RT} \end{pmatrix} \begin{pmatrix} X_1[n] \\ X_2[n] \\ \vdots \\ X_T[n] \end{pmatrix} + \begin{pmatrix} Z_1[n] \\ Z_2[n] \\ \vdots \\ Z_R[n] \end{pmatrix}, \qquad (5.56)$$

where $Z_j[n]$ are IID circularly symmetric complex Gaussians of variance N_0.

We will also write this equation more compactly, in terms of vectors and matrices, as

$$\mathbf{Y} = H\mathbf{X} + \mathbf{Z}, \qquad (5.57)$$

where, for notational convenience, we have also suppressed the time index n.

We consider this model mostly for a *fixed* channel matrix H, but later also for the case where all H_{ij} are IID circularly symmetric complex Gaussians of unit variance, again assuming *coherent* detection, i.e., that the receiver knows the realizations of all of the H_{ij}. However, we point out that it is also of interest to consider models in which there is dependence between the individual fading coefficients.

5.7.1 Channel Known at The Transmitter

Although this case is not as frequent in practice, let us first assume that the transmitter knows the channel matrix H. Then, it can apply what is often referred to as *precoding:* It transmits the signal

$$\mathbf{X} = \tilde{V}\mathbf{W}, \qquad (5.58)$$

where $\tilde{V}$ is the *precoding matrix* and $\mathbf{W}$ is the data-carrying vector (containing the modulated symbols). How should we select the matrix $\tilde{V}$? To understand this, we need to introduce the so-called *singular-value decomposition.* This decomposition asserts that any $R \times T$ matrix H can be written as

$$H = U\Sigma V^H, \qquad (5.59)$$

where U is a unitary[1] $R \times R$ matrix, V is a unitary $T \times T$ matrix, and Σ is a diagonal $R \times T$ matrix containing the *singular values* of the matrix H (where we start the diagonal with

[1]unitary means that $UU^H = U^H U = I_R$, where I_R is the R-dimensional identity matrix.

the upper left element and proceed as far as we can, which means that there are at most $\min\{R,T\}$ non-zero entries in Σ). Now, we see that if we select $\tilde{V}=V$, we get

$$\mathbf{Y} \;=\; U\Sigma\mathbf{W}+\mathbf{Z}. \tag{5.60}$$

Finally, without losing anything, we let the decoder first multiply the received signal by U^H, leading to the model

$$\mathbf{Y}' \;=\; U^H\mathbf{Y}=\Sigma\mathbf{W}+U^H\mathbf{Z}. \tag{5.61}$$

Since Σ is a diagonal matrix, and since the noise vector $U^H\mathbf{Z}$ still has independent components[2], we thus obtain $\min\{R,T\}$ completely independent, parallel channels:

$$\begin{aligned} Y_1' &= \sigma_1 W_1 + Z_1' \\ Y_2' &= \sigma_2 W_2 + Z_2' \\ \vdots &= \vdots \\ Y'_{\min\{R,T\}} &= \sigma_{\min\{R,T\}} W_{\min\{R,T\}} + Z'_{\min\{R,T\}}, \end{aligned} \tag{5.62}$$

where, to be precise, we should point out that some of the singular values could be zero, hence, there could be *fewer* than $\min\{R,T\}$ parallel channels.

5.7.2 V-BLAST: Signaling

In the remainder of this chapter, we will assume that the transmitter does *not* know the matrix H. In this case, how should we signal across the T transmit antennas? In the V-BLAST (Vertical Bell Labs Space Time) architecture, we transmit *independent data streams* across each antenna. Therefore, in our analysis of V-BLAST, we will assume that for $\ell \neq k$,

$$X_\ell[n] \text{ and } X_k[m] \text{ are independent random variables for all } n, m. \tag{5.63}$$

Moreover, we assume that modulation is balanced and that the energy (per symbol) is $\mathcal{E}$:

$$\mathbb{E}\left[X_\ell[n]\right] \;=\; 0, \tag{5.64}$$

$$\mathbb{E}\left[|X_\ell[n]|^2\right] \;=\; \mathcal{E}, \tag{5.65}$$

for all ℓ, n.

5.7.3 V-BLAST: ML Detection

Since the additive noise is Gaussian, we can express the ML detector simply as a minimum distance detector, exactly as in the discussion in Section 3.4.4. Searching over all possible transmitted sequences, we find that the complexity is exponential in the product of the number of transmit antennas and the number of channel uses. In practice, however, many systems decouple the spatial from the temporal processing. That is, they first disentangle the T transmitted signals, and then decode each one separately against the noise. In the next few subsections, we discuss a few common approaches to disentangle the T transmitted signals.

[2]This is left as an exercise for the reader.

5.7.4 Suboptimal V-BLAST Detection: The Zero-Forcer

The rationale of this architecture is to first multiply the channel output vector by a matrix to suppress the interference between the T data streams, namely:

$$\begin{pmatrix} \tilde{Y}_1[n] \\ \tilde{Y}_2[n] \\ \vdots \\ \tilde{Y}_T[n] \end{pmatrix} = \begin{pmatrix} B_{11} & B_{12} & \dots & B_{1R} \\ B_{21} & B_{22} & \dots & B_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ B_{T1} & B_{T2} & \dots & B_{TR} \end{pmatrix} \begin{pmatrix} Y_1[n] \\ Y_2[n] \\ \vdots \\ Y_R[n] \end{pmatrix}. \tag{5.66}$$

For simplicity, let us start with the case where $R = T$ and the matrix H is invertible. Then, the Zero-Forcer chooses $B = H^{-1}$, thus obtaining

$$\begin{pmatrix} \tilde{Y}_1[n] \\ \tilde{Y}_2[n] \\ \vdots \\ \tilde{Y}_T[n] \end{pmatrix} = \begin{pmatrix} X_1[n] \\ X_2[n] \\ \vdots \\ X_T[n] \end{pmatrix} + \begin{pmatrix} \tilde{Z}_1[n] \\ \tilde{Z}_2[n] \\ \vdots \\ \tilde{Z}_R[n] \end{pmatrix}, \tag{5.67}$$

where now we detect $X_1[n]$ using only $\tilde{Y}_1[n]$; we detect $X_2[n]$ using only $\tilde{Y}_2[n]$; and so on.

An alternative and perhaps more instructive way of understanding the Zero-Forcer is to consider the *projection* of the received vector $\mathbf{Y} = (Y_1, Y_2, \dots, Y_R)$ into judiciously chosen directions. (We suppress the time index n for ease of notation.) In particular, we form a first projection

$$\tilde{Y}_1 = \mathbf{b}_1^H \mathbf{Y}, \tag{5.68}$$

where we choose the vector $\mathbf{b}_1$ to be *orthogonal* to columns $2, 3, \dots, T$ of the matrix H. By doing so, $\tilde{Y}_1$ will only depend on X_1, but not on $X_2, X_3, \dots, X_T$. As a second projection, we use

$$\tilde{Y}_2 = \mathbf{b}_2^H \mathbf{Y}, \tag{5.69}$$

where we choose the vector $\mathbf{b}_2$ to be *orthogonal* to columns $1, 3, 4, \dots, T$ of the matrix H. By doing so, $\tilde{Y}_2$ will only depend on X_2, but not on $X_1, X_3, X_4, \dots, X_T$. We keep going until we have extracted all T data streams.

How well does the Zero-Forcer perform? At a first glance, everything looks fine, until we notice that we have done something to the noise: The new noise vector $\tilde{\mathbf{Z}}[n] = H^{-1}\mathbf{Z}[n]$ is still circularly symmetric[3] complex Gaussian with mean zero, but has covariance matrix

$$\mathbb{E}[\tilde{\mathbf{Z}}\tilde{\mathbf{Z}}^H] = N_0 H^{-1} H^{-H}. \tag{5.70}$$

Rather than dwelling on this formula, let us consider an example with $T = R = 2$, namely

$$H = \begin{pmatrix} 1 & 0.99 \\ 0.99 & 1 \end{pmatrix}, \tag{5.71}$$

[3]Note that it is not entirely trivial that in the vector $\tilde{\mathbf{Z}}[n]$, real and imaginary parts are still independent (as required to qualify as "circularly symmetric") — the proof is a bit tedious and left as an exercise.

for which we find

$$H^{-1} \approx \begin{pmatrix} 50.25 & -49.75 \\ -49.75 & 50.25 \end{pmatrix}. \tag{5.72}$$

Thus, we find

$$\mathbb{E}[\tilde{\mathbf{Z}}\tilde{\mathbf{Z}}^H] = N_0 H^{-1} H^{-H} \approx N_0 \begin{pmatrix} 5000 & 4999 \\ 4999 & 5000 \end{pmatrix}. \tag{5.73}$$

Thus, the noise variance in each stream has increased by a factor of 5000 in this unfortunate example! This is very undesirable, and we will therefore explore alternative receiver architectures.

The cases where H is not invertible and where $R \neq T$ are left as exercises.

5.7.5 Suboptimal V-BLAST Detection: MMSE

The main idea behind the Zero-Forcer was to project the received signal vector into appropriately chosen directions. Are there other projection directions that might be of interest? To study this question, reconsider the projection

$$\mathbf{b}_1^H \mathbf{Y}. \tag{5.74}$$

If we use this projection to extract the first data stream, X_1, then our equivalent noise is simply given by $X_1 - \mathbf{b}_1^H \mathbf{Y}$. The noise power is thus given by

$$\mathbb{E}\left[\left|X_1 - \mathbf{b}_1^H \mathbf{Y}\right|^2\right], \tag{5.75}$$

and a reasonable goal is to choose $\mathbf{b}_1$ such as to minimize this noise power. This problem can be solved by taking derivatives, leading to the so-called *orthogonality principle: The error has to be orthogonal to all of the data that was used.* For the case at hand, the error is $\left(X_1 - \sum_{r=1}^R b_{1r}^* Y_r\right)$, and the data used is $Y_1, Y_2, \ldots, Y_R$. Thus, we obtain R orthogonality conditions:

$$\begin{aligned} \mathbb{E}\left[\left(X_1 - \sum_{r=1}^R b_{1r}^* Y_r\right)^* Y_1\right] &= 0, \\ \mathbb{E}\left[\left(X_1 - \sum_{r=1}^R b_{1r}^* Y_r\right)^* Y_2\right] &= 0, \\ \vdots \quad & \vdots \ \vdots \\ \mathbb{E}\left[\left(X_1 - \sum_{r=1}^R b_{1r}^* Y_r\right)^* Y_R\right] &= 0. \end{aligned} \tag{5.76}$$

A somewhat tedious (but simple) calculation shows that we can rewrite this in the following shape:

$$R_{YY}\mathbf{b}_1 = \mathbf{r}_1, \tag{5.77}$$

where the matrix R_{YY} is the autocorrelation matrix of the channel output vector, defined as follows:

$$\begin{aligned} R_{YY} &= \begin{pmatrix} \mathbb{E}[Y_1Y_1^*] & \mathbb{E}[Y_1Y_2^*] & \dots & \mathbb{E}[Y_1Y_R^*] \\ \mathbb{E}[Y_2Y_1^*] & \mathbb{E}[Y_2Y_2^*] & \dots & \mathbb{E}[Y_2Y_R^*] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[Y_RY_1^*] & \mathbb{E}[Y_RY_2^*] & \dots & \mathbb{E}[Y_RY_R^*] \end{pmatrix} && (5.78) \\ &= \mathbb{E}\left[\mathbf{Y}\mathbf{Y}^H\right] && (5.79) \\ &= \mathbb{E}\left[(H\mathbf{X}+\mathbf{Z})(H\mathbf{X}+\mathbf{Z})^H\right] && (5.80) \\ &= H\mathbb{E}\left[\mathbf{X}\mathbf{X}^H\right]H^H + \mathbb{E}\left[\mathbf{Z}\mathbf{Z}^H\right]. && (5.81) \end{aligned}$$

Now, we recall that in V-BLAST, the signals in each of the transmitted streams are independent and of energy $\mathcal{E}$. Therefore, we can express R_{YY} as

$$R_{YY} = \mathcal{E}HH^H + N_0 I_R, \qquad (5.82)$$

where I_R denotes the R-dimensional identity matrix. Moreover, the vector $\mathbf{r}_1$ is given by

$$\mathbf{r}_1 = \begin{pmatrix} \mathbb{E}[Y_1X_1^*] \\ \mathbb{E}[Y_2X_1^*] \\ \vdots \\ \mathbb{E}[Y_RX_1^*] \end{pmatrix} = \mathcal{E}\begin{pmatrix} h_{11} \\ h_{21} \\ \vdots \\ h_{R1} \end{pmatrix}. \qquad (5.83)$$

From Equation (5.77), and assuming that R_{YY} is invertible, we can express the optimal equalization coefficients for the first stream as

$$\mathbf{b}_1 = \mathcal{E}\left(\mathcal{E}HH^H + N_0 I_R\right)^{-1}\begin{pmatrix} h_{11} \\ h_{21} \\ \vdots \\ h_{R1} \end{pmatrix}. \qquad (5.84)$$

By analogy, we can find the appropriate vectors to extract streams $t = 2, 3, \dots, T$ as

$$\mathbf{b}_t = \mathcal{E}\left(\mathcal{E}HH^H + N_0 I_R\right)^{-1}\begin{pmatrix} h_{1t} \\ h_{2t} \\ \vdots \\ h_{Rt} \end{pmatrix}. \qquad (5.85)$$

Finally, as a convenient short-hand, we can summarize all T operations as taking the channel output vector and multiplying it by the following matrix:

$$\mathcal{E}H^H\left(\mathcal{E}HH^H + N_0 I_R\right)^{-1}. \qquad (5.86)$$

Now, let us get a sense for how good this MIMO detector is. That is, how large is the resulting effective noise for each stream? Consider the detection problem in stream 1. The effective noise in this detection problem has variance[4] given by

$$\mathbb{E}\left[\left|X_1 - \sum_{r=1}^{R} B_{1r}^* Y_r\right|^2\right]. \tag{5.87}$$

We can rewrite this as

$$\mathbb{E}\left[\left(X_1 - \sum_{r=1}^{R} B_{1r}^* Y_r\right)^* X_1\right] - \sum_{\tilde{r}=1}^{R} B_{1\tilde{r}}^* \mathbb{E}\left[\left(X_1 - \sum_{r=1}^{R} B_{1r}^* Y_r\right)^* Y_{\tilde{r}}\right]. \tag{5.88}$$

If we plug in the optimum coefficients from Equation (5.84), then we know from the orthogonality conditions in Equation (5.76) that the second term is zero, and hence, that the effective noise in the detection problem has variance

$$\begin{aligned}\mathbb{E}\left[\left(X_1 - \sum_{r=1}^{R} B_{1r}^* Y_r\right)^* X_1\right] &= \mathcal{E} - \mathbf{r}_1^H R_{YY}^{-1} \mathbf{r}_1 &(5.89)\\ &= \mathcal{E}\left(1 - \mathcal{E}\mathbf{h}_1^H (\mathcal{E} H H^H + N_0 I)^{-1} \mathbf{h}_1\right). &(5.90)\end{aligned}$$

Again, rather than dwelling on this formula, let us go back to our example and plug in

$$H = \begin{pmatrix} 1 & 0.99 \\ 0.99 & 1 \end{pmatrix}, \tag{5.91}$$

which gives us

$$\begin{aligned}&\mathbb{E}\left[\left|X_1 - \sum_{r=1}^{R} B_{1r} Y_r\right|^2\right] &(5.92)\\ &= \mathcal{E}\left(1 - \mathcal{E}(1 \quad 0.99)\begin{pmatrix} 1.9801\mathcal{E} + N_0 & 1.98\mathcal{E} \\ 1.98\mathcal{E} & 1.9801\mathcal{E} + N_0 \end{pmatrix}^{-1}\begin{pmatrix} 1 \\ 0.99 \end{pmatrix}\right). &(5.93)\end{aligned}$$

For example, plugging in a signal energy of $\mathcal{E} = 2$ and $N_0 = 1$, we find

$$\mathbb{E}\left[\left|X_1 - \sum_{r=1}^{R} B_{1r} Y_r\right|^2\right] \approx 1.1119. \tag{5.94}$$

That is, we see that there is a significant difference compared to the zero-forcer: there, the effective noise was 5000 (plugging in $N_0 = 1$) while here, it is only 1.1119.

As a final remark, we point out that as $\mathcal{E}$ tends to infinity, the MMSE detector tends to the zero-forcing solution.

[4]We point out that this effective noise is *not* independent of the signal and *not* Gaussian, and thus, strictly speaking, the variance alone does not determine the resulting error probability. However, it is customary to pretend that the noise is Gaussian — and in an order-of-magnitude sense, this is often appropriate. More interestingly, as long as the noise is additive and independent of the signal, in an information-theoretic sense (meaning that we assume that optimal codes are used), for fixed variance, the worst-case distribution is indeed Gaussian.

5.7.6 Suboptimal V-BLAST Detection: Successive Interference Cancellation (SIC)

To be written up for the next version.

5.8 MIMO Communication under Fading

So far, we have studied the MIMO communication channel in Equation (5.56) for a fixed channel matrix H. Returning to the fading channel models, it should be clear that it is also of interest to study the scenario where the channel matrix H is random. Starting from the Rayleigh fading model we have studied in detail, a first MIMO model with fading is simply to assume that all entries of the matrix H are independent of each other, and that each entry by itself is standard Rayleigh fading, exactly as considered earlier.

For this model, let us simply use V-BLAST signaling (which does not require knowing anything about the actual fading matrix). Then, we can ask about the resulting diversity. A simple calculation reveals that we can obtain a diversity of R for each transmitted symbol, where R is the number of receive antennas, as before.

An interesting follow-up question is whether we can obtain a higher diversity. If we assume that the channel matrix stays fixed for at least T time slots, then we could send each information symbol using T time slots: we send that information symbol in turn separately over each of the T transmit antennas. Via maximum-ratio combining, it is straightforward to show that we thus attain a diversity of RT for this symbol.

Clearly, this is a high price to pay: over T channel uses and occupying all T transmit antennas, we have transmitted just one single information symbol. To increase the number of information symbols, we can again use codes like the Alamouti trick discussed earlier. This is called *space-time coding.* A full treatment of space-time coding is beyond the scope of this class (but we will see some constructions in class and in the homework).

5.9 Wireless Multi-user Communication

5.9.1 Wireless Broadcast Channels

There are many aspects of wireless broadcasting that have been intensively studied over the last decade. In this course, we can only scratch the surface.

As a first case, we consider the so-called *broadcast* channel (or "downlink"). In this scenario, there is a base station that serves L users. The base station has a separate message for each of the L users. The goal is to deliver all L messages in an efficient fashion. A baseline strategy is for the base station to take turns and serve all L users. That is, it starts by serving the first user, ignoring the fact that all other $L-1$ users can overhear the communication. Then, it serves the second user, and so on. The question is whether we can do better than this simple, static scheme.

The answer to this question is a resounding yes, and there are multiple techniques and tricks to do better. In this class, we will only study one such technique, often referred to as *multi-user* diversity. For this purpose, we will assume that the base station knows ahead of time which user has the best channel state (that is, the largest magnitude of the channel coefficient). We will also assume that the channel states change over time. To make matters more concrete, let us suppose that there is a single transmitted signal $x[n]$ received by L users. User ℓ receives

$$Y_\ell[n] \quad = \quad H_\ell[n]x[n] + Z_\ell[n], \tag{5.95}$$

where the Rayleigh fading coefficients $H_\ell[n]$ and additive white Gaussian noises $Z_\ell[n]$ of different users are assumed to be independent. Since we assume that the transmitter knows all the channel states ahead of time, in every time step, it can select the one user with the best channel. It is then a simple matter to show that every user will attain diversity L.

There are a few things to observe here. First of all, how would a particular user know that in a certain time slot, she has the best channel state? To fix this, one solution would be to assume that the channel state stays fixed over a block of time slots, and we could include a header sequence into the transmitted message in that block, identifying which user it is destined for. Note that this revised fading model would also make it more realistic to assume channel state information at the transmitter (measured by the users and fed back to the transmitters).

A second issue concerns the effective symbol rate of each user, which is now a random variable, depending on how many times a particular user turned out to be the best user. Over long time intervals, assuming symmetric fading conditions, this would even out and lead to a fair overall allocation. Of course, if we deviate from pure Rayleigh fading and bring in, say, the geometry, meaning that users "closer" to the transmitter tend to have better channels, then we have to be more careful to obtain a fair solution. A similar observation applies to the delay experienced by each user.

Finally, we emphasize that this example only serves to illustrate the issue of multi-user diversity (which is indeed exploited in certain practical wireless systems). By contrast, a fundamental treatment of the problem of broadcasting would involve many more issues, including the idea of "superposition coding," where, in one and the same time slot and frequency band, we transmit information to multiple users.

5.9.2 Wireless Multi-access Channels

In a multi-access scenario, multiple users communicate to a single receiver and the transmitted signals interfere. This leads to many new issues. In classical systems (such as GSM), multi-accessing is solved simply by scheduling: the users take turns so that at no time more than one transmitter is active.

While a full treatment of how to deal with this scenario is beyond this course, we discuss two ideas. First, consider L users with transmit signals $x_\ell[n]$ transmitting to a single destination according to

$$Y[n] \quad = \quad \sum_{\ell=1}^{L} H_\ell[n]x_\ell[n] + Z[n], \tag{5.96}$$

where H_ℓ is Rayleigh fading, independent across users. We can now use the idea of multi-user diversity: In every time slot, only one user transmits, namely, the one with the best channel. While this again attains diversity L for each user, it requires significant overheads (users must be informed when to transmit).

As a second example, let us go back to the ISI channel and follow the treatment in Section 4.7, except that we add more users. That is, our channel model is now, by analogy to Equation (4.39), given by

$$y[n] = \sum_{\ell=1}^{L} \sum_{k=-\infty}^{\infty} h_\ell[k] x_\ell[n-k] + w[n], \tag{5.97}$$

where we have used the same notation as in Section 4.7 because we now apply the FFT-OFDM approach. In particular, we consider data length N plus a sufficient cyclic prefix (chosen such as to accommodate the "worst" of the L users). For a single user, we have seen in Equation (4.48) that FFT-OFDM transforms the original ISI channel into N parallel channels of the form

$$Y_m[k] = H_m X_m[k] + Z_m[k]. \tag{5.98}$$

Due to the linearity of our multi-access model, we can show exactly along the same lines that the FFT-OFDM approach transforms our multi-access channel into N parallel channels of the form:

$$Y_m[k] = \sum_{\ell=1}^{L} H_{\ell,m} X_{\ell,m}[k] + Z_m[k]. \tag{5.99}$$

There are LN coefficients $H_{\ell,m}$, and engineers have found it instructive to draw these as a two-dimensional $L \times N$ sudoku-style grid of "OFDM chips." Then, one can allocate these chips between the L users, again exploiting an idea like multi-user diversity: for every m, we could assign unique transmitting rights to the one user who has the best channel. Or we can mix and match. We will discuss this further in the homework.

5.9.3 Wireless Relay Channels

[TO BE COMPLETED for the next version]

Appendix 5.A Proof of Equation (5.42)

The conditional pdf $f_{Y_1,Y_2,\ldots,Y_L|x}(y_1, y_2, \ldots, y_L|x)$ for any fixed (complex) number x is simply

$$\begin{aligned} f_{Y_1,Y_2,\ldots,Y_L|x}(y_1, y_2, \ldots, y_L|x) &= \prod_{\ell=1}^{L} \frac{1}{\pi N_0} e^{-\frac{1}{N_0}(y_\ell - h_\ell x)^*(y_\ell - h_\ell x)} && (5.100) \\ &= \left(\frac{1}{\pi N_0}\right)^L e^{-\frac{1}{N_0}\sum_{\ell=1}^{L}(y_\ell - h_\ell x)^*(y_\ell - h_\ell x)} && (5.101) \end{aligned}$$

Now, let us suppose that x is selected from the set $\{x_a, x_b\}$, where x_a and x_b are completely arbitrary complex numbers. Based on the noisy observations $(Y_1, Y_2, \ldots, Y_L)$, we want to detect whether x_a or x_b was transmitted. The log-likelihood ratio is

$$LLR_{ab}(y_1, y_2, \ldots, y_L) = \frac{1}{N_0}\left(\sum_{\ell=1}^{L}(y_\ell - h_\ell x_b)^*(y_\ell - h_\ell x_b) - \sum_{\ell=1}^{L}(y_\ell - h_\ell x_a)^*(y_\ell - h_\ell x_a)\right) \tag{5.102}$$

Rewriting the expression in parentheses, we find

$$\begin{aligned} &\sum_{\ell=1}^{L}(y_\ell - h_\ell x_b)^*(y_\ell - h_\ell x_b) - \sum_{\ell=1}^{L}(y_\ell - h_\ell x_a)^*(y_\ell - h_\ell x_a) \\ &= \sum_{\ell=1}^{L} \left(|y_\ell|^2 - h_\ell x_b y_\ell^* - h_\ell^* x_b^* y_\ell - |h_\ell|^2|x_b|^2\right) - \left(|y_\ell|^2 - h_\ell x_a y_\ell^* - h_\ell^* x_a^* y_\ell - |h_\ell|^2|x_a|^2\right) \\ &= -\left(\sum_{\ell=1}^{L} h_\ell^* y_\ell\right)^* x_b - \left(\sum_{\ell=1}^{L} h_\ell^* y_\ell\right) x_b^* + \left(\sum_{\ell=1}^{L} h_\ell^* y_\ell\right)^* x_a + \left(\sum_{\ell=1}^{L} h_\ell^* y_\ell\right) x_a^* \\ &\quad + \sum_{\ell=1}^{L} |h_\ell|^2|x_a|^2 - \sum_{\ell=1}^{L} |h_\ell|^2|x_b|^2. \end{aligned} \tag{5.103}$$

Clearly, as soon as we have $\sum_{\ell=1}^{L} h_\ell^* y_\ell$, we can calculate the LLR — we do not need to retain any other information about the received signal vector. In other words, this very expression is a sufficient statistic for the given problem.

Since we did not make any assumptions about x_a and x_b, this result holds irrespective of the constellation — even if there are more than just two alternatives: any pairwise LLR can be calculated simply from knowing $\sum_{\ell=1}^{L} h_\ell^* y_\ell$.

Therefore, let us define

$$Y = \sum_{\ell=1}^{L} h_\ell^* Y_\ell. \tag{5.104}$$

By plugging in, we find that

$$\begin{aligned} Y &= \sum_{\ell=1}^{L} h_\ell^*(h_\ell x + Z_\ell) && (5.105) \\ &= \sum_{\ell=1}^{L} h_\ell^* h_\ell x + \sum_{\ell=1}^{L} h_\ell^* Z_\ell && (5.106) \\ &= \left(\sum_{\ell=1}^{L} |h_\ell|^2 \right) x + Z, && (5.107) \end{aligned}$$

where the noise Z is again circularly symmetric AWGN, but its variance is given by

$$N_0 \sum_{\ell=1}^{L} |h_\ell|^2. \tag{5.108}$$

To prove this, we can first show that $\mathbb{E}[Re(Z)Im(Z)] = 0$, which ensures the "circularly symmetric" part of the claim. Then, we can show that $\mathbb{E}[Re(Z)^2] = \mathbb{E}[Im(Z)^2] = \frac{N_0}{2} \sum_{\ell=1}^{L} |h_\ell|^2$.

Chapter 6

Classical Linear Codes

6.1 High-dimensional Signaling, Revisited

In Section 3.7, we have considered a set of M orthogonal messages and have observed remarkable behavior in terms of error probability as the number of messages becomes large. The price to pay was a sharp increase in the number of signaling dimensions. Now, looking back at the M message points considered in Section 3.7, it is tempting to add more messages points *without* increasing the number of signal dimensions. In Equation (3.47), what happens to the error probability if we add another message point with coordinates $\mathbf{x}_{M+1} = \sqrt{\mathcal{E}}(1, 1, 1, 0, \dots 0)$? How many more message points can we add with negligible effect on the error probability? In this chapter, we consider a systematic way of designing constellations of message points. These constellations are intuitively pleasing and of obvious practical appeal.

A deeper question would concern the *optimality* of message point constellations for certain channel models. For this, we refer the student to the class on Information Theory as well as to advanced classes on Coding Theory.

6.2 Binary Linear Codes: An Informal Introduction

6.2.1 Binary Codes

A binary (block) code of length N is simply any subset $\mathcal{C}$ of the binary strings of length N. We will denote the number of codewords by M. The *rate* of the code is defined as $R = (\log_2 M)/N$.

Error Correction and Detection; Minimum Distance of the code

There is a natural fashion of thinking about error correction and detection with respect to a block code. Suppose you are given a block code $\mathcal{C}$ of length N and any binary string $\mathbf{z}$ of length N. Then:

1. Either $\mathbf{z}$ belongs to the code, in which case we would conclude that no error happened.

2. Or $\mathbf{z}$ *does not* belong to the code. Then, we have *detected* an error: something went wrong.

   In this second case, we can push things further: we can say that we map the string $\mathbf{z}$ to the *closest*[1] string $\mathbf{x}$ that is part of the code $\mathcal{C}$. In this sense, we have *corrected* errors.

   We can even give a *quantitative measure* as to how many errors we have "corrected:" Simply compare the original string $\mathbf{z}$ to the decoded string $\mathbf{x}$ and count how many disagreements there are (i.e., in how many positions these two strings differ). This is how many errors were "corrected."

How good is a given code $\mathcal{C}$? Our discussion of error correction and detection gives us one set of criteria to answer this question. Typically, we would like the code to correct *as many errors as possible.* To this end, we introduce the *minimum Hamming distance* of the code $\mathcal{C}$ as

$$d_{H,min} \quad = \quad \min_{\mathbf{x},\mathbf{y}\in\mathcal{C},\mathbf{x}\neq\mathbf{y}} d_H(\mathbf{x},\mathbf{y}), \tag{6.1}$$

where the expression $d_H(\mathbf{x},\mathbf{y})$ denotes the number of positions in which the two strings $\mathbf{x}$ and $\mathbf{y}$ differ (the so-called *Hamming distance*).

It is now clear that our code $\mathcal{C}$ can correct *any pattern* of t errors if and only if

$$2t \quad < \quad d_{H,min}. \tag{6.2}$$

This is *one* figure of merit for a given code $\mathcal{C}$; we will see more refined (and more relevant) criteria later on.

6.2.2 Binary Linear Codes

In order to efficiently implement codes, we need them to have some structure. That is, instead of picking *any* M sequences of length N, we now consider one clever way of finding good sets of sequences. Namely, we consider binary *linear* codes of block length N. We represent codewords by row vectors.[2] A binary linear code is specified by a binary matrix G of dimension $K \times N$. This matrix is called the (code) *generator matrix.* The task of the code designer is to find a good generator matrix.

Given the generator matrix, all the codewords can then be generated by

$$\mathbf{b} \quad = \quad \mathbf{a}G, \tag{6.3}$$

where $\mathbf{a}$ is a binary (row) vector of length K, and we let $\mathbf{a}$ vary over all of its 2^K possibilities. We will refer to the code as a binary linear (N, K) block code.

Example. Consider the binary linear $(3, 2)$ block code given by the generator matrix

$$G \quad = \quad \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

[1]Note that there could be multiple closest strings, in which case this does not work.

[2]In coding theory, it is customary to use row vectors. We here choose not to deviate from this tradition.

Using Equation (6.3), we successively plug in $(0,0),(1,0),(0,1)$ and $(1,1)$ for $\mathbf{a}$ and thus identify the $2^K = 4$ codewords as $(0,0,0),(1,0,1),(0,1,1)$, and $(1,1,0)$, respectively.

In such a binary linear code, there cannot be more than 2^K codewords. There could be fewer, depending on the rows of the generator matrix. Let us denote the number of codewords by M. The *rate* of the code is defined as $R = (\log_2 M)/N$.

6.2.3 The Algebra of Binary Linear Codes

Here are two key observations about binary linear codes that we will use over and over again:

P1: In a binary linear code, the all-zero sequence is always a codeword.

P2: In a binary linear code, the sum of any two codewords is again a codeword.

It is easy to verify this property for our running example.

6.2.4 The Geometry of Binary Linear Codes

Let us reconsider our running example, namely, the binary linear $(3,2)$ block code given by the generator matrix

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

If we write $\mathbf{a} = (a_1, a_2)$, then we can write Equation (6.3) as

$$\mathbf{b} = a_1(1,0,1) + a_2(0,1,1). \tag{6.4}$$

This is just a linear combination of two vectors.

For a minute, let us pretend that a_1 and a_2 are real numbers, and the addition is usual real-valued addition. Then, as you have seen in your math classes, Equation (6.4) describes a two-dimensional plane in three-dimensional space, and this plane goes through the origin. Equivalently, you can describe this plane by the equation $b_1 + b_2 = b_3$.

As you have also seen in your math classes, it is often convenient to describe such a plane by the vector that is *orthogonal* to this plane.

Can we do this in the binary world, too? As it turns out, the answer is yes. For our example, a vector that is orthogonal (in the binary world) to both of the binary vectors $(1,0,1)$ and $(0,1,1)$ is

$$\mathbf{h} = (1,1,1), \tag{6.5}$$

and indeed, you can convince yourself that a binary sequence $\mathbf{x}$ of length 3 is a codeword in our binary linear $(3,2)$ block code if and only if it satisfies

$$\mathbf{b}\mathbf{h}^T = 0. \tag{6.6}$$

In a similar fashion, we can show that any binary linear (N, K) block code is just a K-dimensional subspace of N-dimensional space. Again, we can characterize this subspace by $N - K$ binary vectors that are *orthogonal* to the space spanned by the code. Call these vectors $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{N-K}$. Then, we can show that a binary sequence $\mathbf{b}$ of length n belongs to our code if and only if it satisfies

$$\mathbf{b}\mathbf{h}_i^T = 0, \tag{6.7}$$

for $i = 1, 2, \ldots, N-K$. It is often convenient to collect this condition into a matrix condition by forming the so-called *parity check matrix*

$$H = \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \vdots \\ \mathbf{h}_{N-K} \end{pmatrix}, \tag{6.8}$$

and writing the condition as

$$\mathbf{b}H^T = \mathbf{0}, \tag{6.9}$$

where, here, $\mathbf{0}$ is the all-zero (row) vector of length $N - K$.

In general, it is not straightforward to find a parity check matrix H for a code defined via its generator matrix G. However, there is a special case where it is easy. This is the case of *systematic codes,* which are codes with generator matrix of the form

$$G = (I_K \,|\, P), \tag{6.10}$$

where I_K is the K-dimensional identity matrix and P is an arbitrary binary $K \times (N - K)$ matrix. In this case, a parity check matrix can be found as

$$H = \left(P^T \,|\, I_{N-K}\right), \tag{6.11}$$

where I_{N-K} is the $(N - K)$-dimensional identity matrix.

6.2.5 Cosets and Syndromes

An important concept to grasp the structure of binary linear codes comes in the shape of so-called *cosets*. Namely, for each binary sequence $\mathbf{r}$ of length N, we define its corresponding coset as the set of those binary sequences obtained by adding, in turn, each of the codewords in our code to the sequence $\mathbf{r}$.

Formally, we can express this as

$$\mathbf{e} + \mathcal{C} = \{\mathbf{x} : \mathbf{x} = \mathbf{e} + \mathbf{b} \text{ for some } \mathbf{b} \in \mathcal{C}\}, \tag{6.12}$$

where $\mathbf{e}$ is any binary sequence of length N.

Here are two important facts about cosets:

1. Every binary sequence $\mathbf{r}$ of length N is in *exactly one* coset.
2. It is easy to identify which coset simply by calculating the so-called syndrome $\mathbf{s} = \mathbf{r}H^T$, where H is the parity check matrix of our code.

To prove these two facts requires further insights about the underlying (so-called *finite-field*) algebra.

Moreover, since the code has 2^K (distinct) codewords, we know that the coset corresponding to the all-zero sequence (meaning that in Equation (6.12), we choose $\mathbf{e}$ to be the all-zero sequence) contains exactly 2^K sequences. By the same token, we can infer that *every* coset must contain exactly 2^K sequences. Finally, since every sequence is in exactly one coset, this means that there are exactly 2^{N-K} different cosets.

Example. Consider again the binary linear $(3, 2)$ block code with codewords

$$(0, 0, 0), (1, 0, 1), (0, 1, 1), (1, 1, 0).$$

Recall that here, the parity check matrix is simply the vector $(1, 1, 1)$. We obtain the following table of cosets:

binary sequence $\mathbf{r}$	Corresponding Coset				Syndrome $\mathbf{s} = \mathbf{r}H^T$
000	000	101	011	110	0
001	001	100	010	111	1
010	010	111	001	100	1
011	011	110	000	101	0
100	100	001	111	010	1
101	101	000	110	011	0
110	110	011	101	000	0
111	111	010	100	001	1

To understand the meaning of this table, it is important to note that for a set, the ordering does not matter. That is, the cosets on lines 1, 4, 6, and 7 are all exactly the same set (they have exactly the same elements), and the same comment applies to lines 2, 3, 5, and 8. So, in this simple example, there are only two possible cosets. In class, we will see the coset structure of a longer code.

6.2.6 Syndrome Decoding

As we stated, for any binary sequence $\mathbf{r}$ of length N, one can determine which coset it belongs to by calculating the syndrome

$$\mathbf{s} = \mathbf{r}H^T. \tag{6.13}$$

Clearly, by Equation (6.9), the syndrome is $\mathbf{s} = \mathbf{0}$ if and only if $\mathbf{r}$ is a codeword. Indeed, the code itself is the coset corresponding to choosing $\mathbf{e}$ as the all-zero sequence.

For all other sequences, there is an insightful observation as follows: Trivially, we can write $\mathbf{r} = \mathbf{b} + \mathbf{e}$, where $\mathbf{b}$ is a codeword. But then,

$$\mathbf{s} = \mathbf{r}H^T = (\mathbf{b} + \mathbf{e})H^T = \mathbf{b}H^T + \mathbf{e}H^T = \mathbf{e}H^T. \tag{6.14}$$

That is, the syndrome $\mathbf{s}$ identifies the coset in which the error pattern $\mathbf{e}$ lies.

This leads to the idea of *syndrome decoding:* For any binary sequence $\mathbf{r}$, determine first the coset in which it lies (with respect to a given code $\mathcal{C}$). Then, inside the coset, find the sequence of smallest Hamming weight, which we denote by $\mathbf{e}_{min}(\mathbf{r})$. We decode the sequence $\mathbf{r}$ to the sequence $\mathbf{r} - \mathbf{e}_{min}(\mathbf{r})$, that is, we simply subtract the error pattern from the sequence $\mathbf{r}$. (Recall that over the binary field, adding and subtracting is the same.)

To understand the appeal of this decoder, let us imagine that we transmit a codeword over an imperfect channel. Suppose that we are guaranteed that out of the N components of the codeword, the channel only affects at most t and leaves the remaining $N-t$ components perfectly intact. Then, as long as all binary sequences of Hamming weight up to t are in *different* cosets, we are guaranteed that the syndrome decoder returns the correct answer.

6.2.7 Distance Properties and Minimum Distance

The power of a code results from its distance properties, i.e., the mutual distances between all pairs of codewords. For binary sequences, there is only one (intuitively pleasing) way to measure distance, the so-called Hamming distance:

$$d_H(\mathbf{x}, \mathbf{y}) \quad = \quad \text{Number of positions where } \mathbf{x} \text{ and } \mathbf{y} \text{ differ.} \tag{6.15}$$

Now, take an arbitrary codeword that is a member of our binary linear (N, K) block code, call it $\mathbf{b}$. Define the *distance enumerator* for codeword $\mathbf{b}$ as

$$D_k(\mathbf{b}) \quad = \quad (\text{ Number of codewords } \mathbf{y} \text{ that satisfy } d_H(\mathbf{b}, \mathbf{y}) = k\ )\,. \tag{6.16}$$

Example: For the binary linear $(3, 2)$ block code considered above, let us take, for example, the codeword $\mathbf{b} = (0, 1, 1)$. Then, we can see that $D_0(\mathbf{b} = (0, 1, 1)) = 1, D_1(\mathbf{b} = (0, 1, 1)) = 0, D_2(\mathbf{b} = (0, 1, 1)) = 3$, and $D_3(\mathbf{b} = (0, 1, 1)) = 0$.

The next fundamental fact about binary linear codes is the following:

P3: For a binary linear code, $D_k(\mathbf{b}) = D_k$, i.e., it does *not* depend on $\mathbf{b}$.

This fundamental insight follows straightforwardly from **P2** (i.e., the fact that the sum of any two codewords is again a codeword) and the observation that

$$d_H(\mathbf{x}, \mathbf{y}) \quad = \quad d_H(\mathbf{x} + \mathbf{z}, \mathbf{y} + \mathbf{z}), \tag{6.17}$$

for any binary vector $\mathbf{z}$, where the addition is component-wise modulo-2.

We know that any binary linear code always contains the all-zero codeword. Therefore, to explore the distance spectrum D_k, we can simply start from the all-zero codeword. This is particularly easy if we introduce the Hamming weight of a binary sequence:

$$w_H(\mathbf{x}) \quad = \quad (\text{ Number of positions where } \mathbf{x} \text{ is one }) = d_H(\mathbf{x}, \mathbf{0}). \tag{6.18}$$

With this, it follows that D_k is simply equal to the *weight spectrum* of the code, defined as

$$A_k \quad = \quad (\text{ Number of codewords } \mathbf{b} \text{ that have Hamming weight } k\ )\,. \tag{6.19}$$

We can then also state the following more convenient version of **P3**:

P3': For a binary linear code, $D_k(\mathbf{b}) = A_k$ for every codeword $\mathbf{b}$.

One interesting fact about a code is its *minimum distance,* that is, the smallest Hamming distance between any two codewords of the code, which we can formally define as

$$d_{H,min} = \min_{\mathbf{b},\mathbf{c}} d_H(\mathbf{b},\mathbf{c}), \tag{6.20}$$

where both $\mathbf{b}$ and $\mathbf{c}$ are codewords. From our theorem, we can directly conclude that

$$d_{H,min} = (\text{ smallest Hamming weight of any codeword, except the all-zero codeword }). \tag{6.21}$$

However, we have to caution the reader that it is not always easy to directly find the minimum distance of a code.

Example: For the binary linear $(3,2)$ block code considered above, we have seen that $D_0 = 1, D_1 = 0, D_2 = 3, D_3 = 0$. Thus, the minimum Hamming weight of any non-zero codeword is 2 (and there are 3 codewords of weight 2). Thus, the minimum distance of the code must be 2.

6.2.8 Minimum Distance via the Parity Check Matrix

We have seen that any binary sequence of length n is a codeword if it satisfies

$$\mathbf{b}H^T = \mathbf{0}. \tag{6.22}$$

To find the minimum distance of the code, we must find the one sequence with the *smallest* Hamming weight, i.e., the *smallest* number of ones, that satisfies this condition, and we cannot choose the all-zero sequence.

This problem has a nice structure. Namely, we can observe that we are simply adding up a subset of the *columns* of the parity check matrix H, precisely those columns that correspond to the 1's in the candidate codeword $\mathbf{b}$.

Therefore, we obtain the following fundamental insight:

- The minimum distance of the code is equal to the smallest integer d such that there is a set of d columns in the parity check matrix that adds up to the all-zero sequence.

Example: For the binary linear $(3,2)$ block code considered above, we have seen that the parity check matrix is simply the (row) vector $(1,1,1)$. This vector has 3 columns. Any single column is not equal to zero, so the minimum distance is definitely larger than 1. Are there two columns that can be added up to arrive at zero? Clearly, this is true. Therefore, the minimum distance is 2, in line with our earlier result.

6.2.9 Minimum Distance: Singleton Bound

For a binary code of block length N, it is clear that the minimum distance can never be larger than N. Now, suppose that the code has dimension K. Intuitively, it should be clear that the larger K (which means: the more codewords there are in the code), the smaller the minimum distance.

A simple bound called the *Singleton Bound* shows that *any* (N, K) linear code with minimum distance $d_{H,min}$ must satisfy

$$d_{H,min} \quad \leq \quad N - K + 1. \tag{6.23}$$

This follows trivially from the main insight of Section 6.2.8: Since the parity check matrix is of dimension $(N-K) \times N$, we know that *any* collection of $N-K+1$ columns must be linearly dependent, meaning that it must contain a subset that adds up to the all-zero sequence. But this directly means that the minimum distance cannot be larger than $N - K + 1$.

A code that attains equality in the Singleton bound is called an *MDS code* (maximum distance separable code).

6.2.10 Hamming Codes

Hamming's construction is a simple way of exploiting the main result of Subsection 6.2.8 in order to construct a code that has minimum distance $d_{H,min} = 3$. Namely, we directly design the parity check matrix H in such a way that *no two columns* are equal, and no column is equal to the all-zero sequence.

From Subsection 6.2.8, this means that we need to add up at least *three* columns of the parity check matrix in order to arrive at the all-zero sequence. Thus, the code has a minimum distance of at least $d_{H,min} = 3$.

Now, we would like to select *as many columns* as possible: the more columns, the more codewords we have in our code. So, Hamming codes fix the length $N - K$ of the columns in the parity check matrix and simply include *all* possible binary vectors of length $N - K$, except the all-zero vector. For example, if we set $N - K = 3$, there are 7 different non-zero binary vectors, and we obtain the following parity check matrix for the $(7, 4)$ Hamming code:

$$H \quad = \quad \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{6.24}$$

It is easy to show that a generator matrix is given by

$$G \quad = \quad \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{6.25}$$

Here are all $2^K = 16$ codewords of the (7,4) Hamming code:

$$\begin{aligned}
\mathbf{b}_1 &= (0,0,0,0,0,0,0)\\
\mathbf{b}_2 &= (1,0,0,0,1,1,0)\\
\mathbf{b}_3 &= (0,1,0,0,1,0,1)\\
\mathbf{b}_4 &= (0,0,1,0,0,1,1)\\
\mathbf{b}_5 &= (0,0,0,1,1,1,1)\\
\mathbf{b}_6 &= (1,1,0,0,0,1,1)\\
\mathbf{b}_7 &= (1,0,1,0,1,0,1)\\
\mathbf{b}_8 &= (1,0,0,1,0,0,1)\\
\mathbf{b}_9 &= (0,1,1,0,1,1,0)\\
\mathbf{b}_{10} &= (0,1,0,1,0,1,0)\\
\mathbf{b}_{11} &= (0,0,1,1,1,0,0)\\
\mathbf{b}_{12} &= (1,1,1,0,0,0,0)\\
\mathbf{b}_{13} &= (1,1,0,1,1,0,0)\\
\mathbf{b}_{14} &= (1,0,1,1,0,1,0)\\
\mathbf{b}_{15} &= (0,1,1,1,0,0,1)\\
\mathbf{b}_{16} &= (1,1,1,1,1,1,1)
\end{aligned}$$

We can also write the weight enumerator function explicitly (just by counting codewords of a particular Hamming weight in the above list):

$$A_0 = 1,\ A_1 = 0,\ A_2 = 0,\ A_3 = 7,\ A_4 = 7,\ A_5 = 0,\ A_6 = 0,\ A_7 = 1, \tag{6.26}$$

which, for example, shows that the minimum (Hamming) distance of the code $d_{H,min} = 3$.

6.3 Binary Linear Block Codes on Noisy Channels

In this section, we characterize the channel by a conditional probability density function

$$f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}), \tag{6.27}$$

where we use $\mathbf{B}$ to denote the random vector representing the codeword (and $\mathbf{b}$ its realization) and we use $\mathbf{Y}$ to represent the channel output vector (and $\mathbf{y}$ its realization).

6.3.1 ML Decoding, Bhattacharyya bound

By analogy to Section 3.5, let us consider $E_{m,m'}$, the event that message m' was detected, given that message m was transmitted. That is, we suppose that the codeword $\mathbf{b}_m$ is transmitted, but the decoder decided for codeword $\mathbf{b}_{m'}$. The ML decoder decides in favor of $\mathbf{b}_{m'}$ if the received vector $\mathbf{y}$ satisfies $f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_{m'}) > f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_m)$. Therefore,

$$\begin{aligned}\mathbb{P}\left(E_{m,m'}\right) &= \mathbb{P}\left(\text{ decode to } \mathbf{b}_{m'} \;\middle|\; \mathbf{b}_m \text{ is transmitted }\right) \\ &= \mathbb{P}\left(\ln\left(\frac{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{Y}|\mathbf{b}_{m'})}{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{Y}|\mathbf{b}_m)}\right) \geq 0 \;\middle|\; \mathbf{b}_m \text{ is transmitted}\right). \qquad (6.28)\end{aligned}$$

Next, we use the Chernoff bound to upper bound this, for $\lambda > 0$, as

$$\begin{aligned}&\mathbb{P}\left(\ln\left(\frac{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{Y}|\mathbf{b}_{m'})}{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{Y}|\mathbf{b}_m)}\right) \geq 0 \;\middle|\; \mathbf{b}_m \text{ is transmitted}\right) \\ &\leq \mathbb{E}\left[e^{\lambda \ln\left(\frac{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{Y}|\mathbf{b}_{m'})}{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{Y}|\mathbf{b}_m)}\right)} \;\middle|\; \mathbf{b}_m \text{ is transmitted}\right] \qquad (6.29) \\ &= \int_{\mathbf{y}} e^{\lambda \ln\left(\frac{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_{m'})}{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_m)}\right)} f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_m) d\mathbf{y} \qquad (6.30) \\ &= \int_{\mathbf{y}} f^{\lambda}_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_{m'}) f^{1-\lambda}_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_m) d\mathbf{y}. \qquad (6.31)\end{aligned}$$

Next, a particularly appealing choice appears to be $\lambda = 1/2$ (though we challenge the reader to try other choices), from which we find

$$\mathbb{P}\left(E_{m,m'}\right) \leq \int_{\mathbf{y}} \sqrt{f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_{m'}) f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}_m)} d\mathbf{y}. \qquad (6.32)$$

Note that so far, we have not used a single property of our code... To bring this to bearing, we now assume that the channel is *memoryless,* by which we mean that

$$f_{\mathbf{Y}|\mathbf{B}}(\mathbf{y}|\mathbf{b}) = \prod_{i=1}^{N} f_{Y|B}(y_i|b_i). \qquad (6.33)$$

Then, we find

$$\begin{aligned}\mathbb{P}\left(E_{m,m'}\right) &\leq \int_{\mathbf{y}} \sqrt{\prod_{i=1}^{N} f_{Y|B}(y_i|b_{m',i}) f_{Y|B}(y_i|b_{m,i})} d\mathbf{y} \\ &= \prod_{i=1}^{N} \int_{y_i} \sqrt{f_{Y|B}(y_i|b_{m',i}) f_{Y|B}(y_i|b_{m,i})} dy_i. \qquad (6.34)\end{aligned}$$

However, the integral inside the product evaluates to 1 whenever $b_{m',i} = b_{m,i}$, and when $b_{m',i} \neq b_{m,i}$, it evaluates to

$$\int_{y} \sqrt{f_{Y|B}(y|1) f_{Y|B}(y|0)} dy \overset{\text{def}}{=} \Delta, \qquad (6.35)$$

where Δ is called the *Bhattacharyya parameter*[3] of the (binary-input) channel $f_{Y|B}(y|b)$. Using this, we can express our bound as

$$\mathbb{P}\left(E_{m,m'}\right) \quad \leq \quad \Delta^{d_H(\mathbf{b}_{m'},\mathbf{b}_m)}. \tag{6.36}$$

Using the union bound as in Equation (3.39), we find

$$P_e \quad \leq \quad \frac{1}{M}\sum_{m=1}^{M}\sum_{m'=1,m'\neq m}^{M} \Delta^{d_H(\mathbf{b}_{m'},\mathbf{b}_m)}. \tag{6.37}$$

Next, using the distance enumerator $D_k(\mathbf{b})$ defined in Equation (6.16), we can write

$$P_e \quad \leq \quad \frac{1}{M}\sum_{m=1}^{M}\left(D_1(\mathbf{b}_m)\Delta + D_2(\mathbf{b}_m)\Delta^2 + \ldots + D_N(\mathbf{b}_m)\Delta^N\right). \tag{6.38}$$

Now, exploiting the structure of the linear code, we recall that $D_k(\mathbf{b})$ does not depend on which codeword $\mathbf{b}$ we consider, and since the all-zero sequence is itself a codeword, we have $D_k(\mathbf{b}) = A_k$, where A_k is the weight enumerator of the code, that is, the number of codewords in the code that have Hamming weight exactly equal to k. Hence,

$$P_e \quad \leq \quad \frac{1}{M}\sum_{m=1}^{M}\left(A_1\Delta + A_2\Delta^2 + \ldots + A_N\Delta^N\right), \tag{6.39}$$

but now, the expression in parentheses no longer depends on m, and hence,

$$P_e \quad \leq \quad \sum_{\ell=d_{H,min}}^{N} A_\ell\Delta^\ell, \tag{6.40}$$

where Δ is the Bhattacharyya parameter of the channel.

It can be shown that $0 \leq \Delta \leq 1$ for all channel laws $p_{Y|X}(y|x)$ (for example via Cauchy-Schwarz). Because of this, the above upper bound on P_e can be weakened to

$$P_e \quad \leq \quad \left(2^K - 1\right)\Delta^{d_{H,min}}, \tag{6.41}$$

which tends to be much easier to deal with.

6.4 Binary Linear Block Codes on AWGN Channels

We reconsider the standard AWGN channel model, given by the following description:

$$Y(t) \quad = \quad x(t) + Z(t), \tag{6.42}$$

where $Z(t)$ is (real-valued) additive white Gaussian noise of power spectral density $N_0/2$.

[3]If the channel output alphabet of $\mathbf{y}$ is discrete-valued, all integrals in the derivation as well as in the definition of Δ are naturally replaced with sums.

6.4.1 Modulation

The basic idea is to consider the binary codeword of length N, given by

$$\mathbf{b} \;=\; (b_1, b_2, \ldots, b_N), \tag{6.43}$$

and to map it symbol-by-symbol to a waveform, for example, simply by transmitting the signal

$$x(t) \;=\; \sum_{\ell=1}^{n} A(b_\ell) g(t-(\ell-1)T), \tag{6.44}$$

where we could choose the pulse shape $g(t)$ to be a box of width T and height $1/\sqrt{T}$, starting at $t=0$, and

$$A(b_\ell) \;=\; \begin{cases} \sqrt{\mathcal{E}}, & \text{if } b_\ell = 1, \\ -\sqrt{\mathcal{E}}, & \text{if } b_\ell = 0. \end{cases} \tag{6.45}$$

Now, we can proceed exactly as in Chapter 3: Without loss of optimality, we can convert this scenario into a vector AWGN channel with 2^K message points: one message point for each codeword. Let us consider this for the case of a simple example:

Example 6.1. For the (7,4) Hamming code, modulated onto $\pm\sqrt{\mathcal{E}}$, we end up with the following $M=16$ message points:

$$\begin{aligned}
\mathbf{x}_1 &= \sqrt{\mathcal{E}}(-1,-1,-1,-1,-1,-1,-1)\\
\mathbf{x}_2 &= \sqrt{\mathcal{E}}(1,-1,-1,-1,1,1,-1)\\
\mathbf{x}_3 &= \sqrt{\mathcal{E}}(-1,1,-1,-1,1,-1,1)\\
\mathbf{x}_4 &= \sqrt{\mathcal{E}}(-1,-1,1,-1,-1,1,1)\\
\mathbf{x}_5 &= \sqrt{\mathcal{E}}(-1,-1,-1,1,1,1,1)\\
\mathbf{x}_6 &= \sqrt{\mathcal{E}}(1,1,-1,-1,-1,1,1)\\
\mathbf{x}_7 &= \sqrt{\mathcal{E}}(1,-1,1,-1,1,-1,1)\\
\mathbf{x}_8 &= \sqrt{\mathcal{E}}(1,-1,-1,1,-1,-1,1)\\
\mathbf{x}_9 &= \sqrt{\mathcal{E}}(-1,1,1,-1,1,1,-1)\\
\mathbf{x}_{10} &= \sqrt{\mathcal{E}}(-1,1,-1,1,-1,1,-1)\\
\mathbf{x}_{11} &= \sqrt{\mathcal{E}}(-1,-1,1,1,1,-1,-1)\\
\mathbf{x}_{12} &= \sqrt{\mathcal{E}}(1,1,1,-1,-1,-1,-1)\\
\mathbf{x}_{13} &= \sqrt{\mathcal{E}}(1,1,-1,1,1,-1,-1)\\
\mathbf{x}_{14} &= \sqrt{\mathcal{E}}(1,-1,1,1,-1,1,-1)\\
\mathbf{x}_{15} &= \sqrt{\mathcal{E}}(-1,1,1,1,-1,-1,1)\\
\mathbf{x}_{16} &= \sqrt{\mathcal{E}}(1,1,1,1,1,1,1)
\end{aligned}$$

6.4.2 ML Decoding, Standard Gaussian bound

After modulating our binary linear code onto waveforms and doing the usual conversion to the vector AWGN channel, we have the standard scenario

$$\mathbf{Y} \;=\; \mathbf{x}+\mathbf{Z}, \tag{6.46}$$

where all vectors are of length N, the vector $\mathbf{x}$ is known to be selected uniformly from a given set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$ and the noise vector $\mathbf{Z}$ is a vector of independent, zero-mean Gaussian random variables of variance $N_0/2$.

The only new thing is that there is some **structure** to the set of message points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$.

As a first step, we can use the tools developed in Section 3.5. There, we found the following upper bound on the error probability (Equation (3.41)):

$$P_e \;\le\; \frac{1}{M}\sum_{m=1}^{M}\sum_{m'=1,m'\neq m}^{M} Q\left(\frac{d_{mm'}}{2\sqrt{N_0/2}}\right), \tag{6.47}$$

where, as before,

$$d_{mm'} \;=\; \|\mathbf{x}_m - \mathbf{x}_{m'}\|. \tag{6.48}$$

The key is to connect this to the Hamming distance between the corresponding codewords. Let us now suppose that we have simply modulated a code digit of “1” to $\sqrt{\mathcal{E}}$ and a code digit of “0” to $-\sqrt{\mathcal{E}}$. It is then easy to show that

$$d_{mm'} \;=\; 2\sqrt{\mathcal{E}d_H(\mathbf{b}_m, \mathbf{b}_{m'})}, \tag{6.49}$$

Hence, we can give the upper bound

$$P_e \;\le\; \frac{1}{M}\sum_{m=1}^{M}\sum_{m'=1,m'\neq m}^{M} Q\left(\sqrt{\frac{2d_H(\mathbf{b}_m, \mathbf{b}_{m'})\mathcal{E}}{N_0}}\right). \tag{6.50}$$

Next, using the distance enumerator $D_k(\mathbf{b})$ defined in Equation (6.16), we can write

$$P_e \;\le\; \frac{1}{M}\sum_{m=1}^{M}\left(D_1(\mathbf{b}_m)Q\left(\sqrt{\frac{2\mathcal{E}}{N_0}}\right)+D_2(\mathbf{b}_m)Q\left(\sqrt{\frac{4\mathcal{E}}{N_0}}\right)+\ldots+D_N(\mathbf{b}_m)Q\left(\sqrt{\frac{2N\mathcal{E}}{N_0}}\right)\right) \tag{6.51}$$

Now, exploiting the structure of the linear code, we recall that $D_k(\mathbf{b})$ does not depend on which codeword $\mathbf{b}$ we consider, and since the all-zero sequence is itself a codeword, $D_k(\mathbf{b}) = A_k$. Hence,

$$P_e \;\le\; \frac{1}{M}\sum_{m=1}^{M}\left(A_1Q\left(\sqrt{\frac{2\mathcal{E}}{N_0}}\right)+A_2Q\left(\sqrt{\frac{4\mathcal{E}}{N_0}}\right)+\ldots+A_NQ\left(\sqrt{\frac{2N\mathcal{E}}{N_0}}\right)\right) \tag{6.52}$$

but now, the expression in parentheses no longer depends on m, and hence,

$$P_e \leq \sum_{\ell=d_{H,min}}^{N} A_\ell Q\left(\sqrt{\frac{2\ell\mathcal{E}}{N_0}}\right), \tag{6.53}$$

where $d_{H,min}$ is the minimum (Hamming) distance of the code and A_ℓ is the weight enumerator function of the code.

Finally, it is sometimes instructive to further upper bound this. To this end we note that the largest term in the above sum is for $\ell = d_{H,min}$, thus,

$$P_e \leq \left(2^K - 1\right) Q\left(\sqrt{\frac{2\mathcal{E}d_{H,min}}{N_0}}\right). \tag{6.54}$$

If you like, you may use the standard exponential upper bound $Q(x) \leq e^{-x^2/2}$ to obtain

$$P_e \leq \left(2^K - 1\right) e^{-\frac{\mathcal{E}}{N_0}d_{H,min}}. \tag{6.55}$$

6.4.3 ML Decoding, Bhattacharyya bound

For the AWGN channel used with antipodal modulation $\pm\sqrt{\mathcal{E}}$ and noise of variance $N_0/2$, we find that the Bhattacharyya parameter is given by

$$\Delta = e^{-\frac{\mathcal{E}}{N_0}}. \tag{6.56}$$

Hence, using Equation (6.40), we find

$$P_e \leq \sum_{\ell=d_{H,min}}^{N} A_\ell e^{-\ell\frac{\mathcal{E}}{N_0}}. \tag{6.57}$$

Comparing to Equation (6.53), we observe that the Bhattacharyya approach leads to a slightly weaker bound — easily obtained by combining Equation (6.53) with the standard upper bound $Q(x) \leq e^{-x^2/2}$.

6.4.4 "Hard-decision decoding"

Consider again the standard vector AWGN channel:

$$\mathbf{Y} = \mathbf{x} + \mathbf{Z}, \tag{6.58}$$

where all vectors are of length N, the vector $\mathbf{x}$ is known to be selected uniformly from a given set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$ and the noise vector $\mathbf{Z}$ is a vector of independent, zero-mean Gaussian random variables of variance $N_0/2$.

Now, suppose that we have simply modulated a code digit of "1" to $\sqrt{\mathcal{E}}$ and a code digit of "0" to $-\sqrt{\mathcal{E}}$.

A popular suboptimal decoder is to first do a "hard-decision" step. That is, the receiver takes the received vector $\mathbf{y}$ and first forms the vector $\mathbf{c}$, where

$$c_i = \begin{cases} 0, & \text{if } y_i < 0, \\ 1, & \text{if } y_i \geq 0. \end{cases} \tag{6.59}$$

Intuitively, we are taking every received digit and separately guessing whether the corresponding code bit was a "0" or a "1," without exploiting the structure of the codebook. This is clearly generally suboptimal.

If we start the decoding process with this suboptimal step, we are equivalently turning the original channel into a new channel:

$$p_{\mathbf{C}|\mathbf{B}}(\mathbf{c}|\mathbf{b}), \tag{6.60}$$

where now both $\mathbf{b}$ and $\mathbf{c}$ are binary vectors. To analyze the error probability associated with decoding from this channel, we can apply the approach in Section 6.3.1. This allows to develop an understanding of the loss due to hard-decision decoding, which is left as an exercise.

6.5 Some Famous Binary Codes

Hamming Codes

For every integer $m \geq 3$, there exists a Hamming code with the following properties:

$$\begin{aligned} \text{Code Length:} \quad & N = 2^m - 1 \\ \text{Code Dimension:} \quad & K = 2^m - m - 1 \\ \text{Minimum distance:} \quad & d_{H,min} = 3. \end{aligned}$$

To construct such a Hamming code, it is easiest to start from the parity check matrix. This matrix is of dimension $m \times (2^m - 1)$ and contains as its rows all possible distinct binary sequences of length m *except* the all-zero sequence.

Reed-Muller Codes

For all integers m and r with $0 \leq r \leq m$, there exists a binary Reed-Muller code with the following properties:

$$\begin{aligned} \text{Code Length:} \quad & N = 2^m \\ \text{Code Dimension:} \quad & K = 1 + \binom{m}{1} + \binom{m}{2} + \ldots + \binom{m}{r} \\ \text{Minimum distance:} \quad & d_{H,min} = 2^{m-r}. \end{aligned}$$

One way of constructing these codes is through their generator matrix. To do so, we start with the matrix

$$G_{(2,2)} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}. \tag{6.61}$$

Next, we take the m-fold Kronecker product of this matrix with itself, thus arriving at a matrix of dimenions $2^m \times 2^m$. Finally, of this matrix, we only retain those rows that have a Hamming weight of at least 2^{m-r}. This is the generator matrix of the Reed-Muller code with parameters m and r. (*Exercise:* Verify that this matrix has dimension $(1 + \binom{m}{1} + \binom{m}{2} + \ldots + \binom{m}{r}) \times 2^m$.)

Recently, a promising code construction called *Polar Codes* has been proposed. These codes are closely related to Reed-Muller codes. The difference is that instead of retaining those rows that have a Hamming weight of at least 2^{m-r}, a more involved selection of rows is carried out, based on *information measures.*

The (23,12) Golay Code

This code was invented by Marcel J. E. Golay, born on May 3, 1902, in Neuchâtel, Switzerland. It has many special properties. The basics are:

$$\begin{aligned} \text{Code Length:} \quad & N = 23 \\ \text{Code Dimension:} \quad & K = 12 \\ \text{Minimum distance:} \quad & d_{H,min} = 7. \end{aligned}$$

A generator matrix for this code is given by

$$G \quad = \quad \begin{pmatrix} I_{12} & A \end{pmatrix}, \tag{6.62}$$

where I_{12} denotes the identity matrix of dimension 12×12 and A is the matrix

$$A \quad = \quad \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1
\end{pmatrix} \tag{6.63}$$

Chapter 7

Polar Codes

7.1 Motivation

It took about 70 years since Shannon's seminal 1948 paper for coding theorists and practitioners to come up with coding schemes that have low complexity and perform close to the Shannon limit of the variety of channels seen in practice.

In this chapter we will talk about one of those – they are called *polar codes* and were only introduced in 2008. As we will see these codes are extremely elegant and closely connected to the information-theoretic relevant quantities, in particular mutual information. Since this is not a class on information theory, but rather on digital communication, and since so far we have not talked about quantities such as entropy and mutual information we will keep our discussion more or less exclusively to the binary erasure channel. For this case the notion of capacity is very intuitive and the expression is trivial. But once you absorbed the material in this section it takes very little to understand the general case (binary symmetric channels, Gaussian channels).

So whenever there is no real difference between the specific case and the general case, we will discuss polar codes in general. But all examples and calculations will be done for the binary erasure channel of erasure probability ϵ.

7.2 Summary

Before we discuss the specifics of polar codes let us just summarize their main characteristics and strong points.

1. A polar code of length N can be encoded and decoded in complexity $O(N \ln(N))$, i.e., in *almost* linear time. It is therefore quite easy to implement with today's hardware for quite long lengths (many thousands of bits).

2. A better way to measure complexity is to measure the number of operations per decoded bit and to measure it as a function of the *gap to capacity* δ. We say that we are transmitting at a gap δ with block-error probability P_B if we are transmitting at a rate $R = C - \delta$, where C is the capacity of the channel and we have a scheme that

has block-error probability P_B. For polar codes the complexity per decoded bit then scales like $\ln(1/\delta)$. In more detail, this complexity tends to infinity as we approach capacity but only very slowly.

3. The block-length that we require to transmit at a gap δ to capacity at a fixed block-error probability scales roughly $1/\delta^4$. Note that *optimal* codes have a significantly shorter block-length, namely $1/\delta^2$.

4. All proofs are extremely simple (even for general channels), considerably simpler than for other coding schemes.

7.3 Notation and Channel

In the following we will write W to denote a "channel." We will always think of W as a binary erasure channel with some erasure probability, call it ϵ, $0 \leq \epsilon \leq 1$. The channel takes as inputs bits, i.e., 0s and 1s. It "acts" on each bit indepedently. This is called a *memoryless* channel. In particular, the BEC(ϵ) takes each input bit and passes it unchanged to the output with probability $1-\epsilon$, and replaces it with a "?" with probability ϵ. The special symbol "?" means that the original input symbol was erased.

What is the capacity of the channel BEC(ϵ). It is $C_{\text{BEC}}(\epsilon) = 1-\epsilon$. The bound $C_{\text{BEC}}(\epsilon) \leq 1-\epsilon$ is easy to see. If we were told in advance what fraction $1-\epsilon$ of all bits would not be erased, then the best we could do is to load the information into those positions and to ignore the remaining positions (which will get erased). So even with this additional side information our capacity would be upper bounded by this quantity. It is perhaps more surprising that we can achieve this upper bound. We will shortly see how we can do this.

7.4 The Basic Polarization Step

Consider the diagram in Fig. **??**. We take two identical channels W. Our input is the binary pair $U^T = (U_1, U_2)$. This input is converted via the 2×2 binary full-rank matrix G_2 to the "codeword" $X^T = (X_1, X_2)$ according to $X = G_2 U$. We have

$$G_2 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

Note that the matrix G_2 is full rank. We therefore do not loose any information via this map. But also we do not gain any. So what is the point of this transformation?

Consider now the following *successive* decoder. We first decode the bit U_1 given the output $Y^T = (Y_1, Y_2)$. We call this "channel" W^- (it will turn out that this channel is worse than the original channel W, hence the minus sign). We think of U_2 in this respect as being "noise."

$$W^- : U_1 \to Y.$$

Note that $X_1 = U_1 + U_2$, which, since we are dealing with the binary case is equivalent to $U_1 = X_1 + U_2$. Further $X_2 = U_2$. Combining these two we see that $U_1 = X_1 + X_2$. In other words, the bit which we want to decode is the sum of two unknown bits X_1 and X_2. Finally, X_1 is transmitted over the channel W and we get to observe the output Y_1 and X_2 is sent over an indepedent channel W and we get to see the output Y_2.

Let us summarize. The channel W^- is the following. We have two unknown bits X_1 and X_2 which are sent over two independent channels W and we observe the two outputs Y_1 and Y_2. We are interested in the bit $U_1 = X_1 + X_2$. If we think of the erasure channel then the resulting channel is again an erasure channel but with an erasure probability $1-(1-\epsilon)^2$, since we get an erasure unless *both* bits have been passed through unerased. Note that $1-(1-\epsilon)^2 = \epsilon(2-\epsilon) \geq \epsilon$. So W^- is *worse* than the original channel W.

Consider now the view point seen from bit U_2, *given that we know* U_1. Why is this relevant? Imagine now that we have a very good code at our disposal and that we code over long instances so that we can decode the bit U_1 with high probability. In other words, our estimate $\hat{U}_1$ agrees with high probability with the true value U_1. We will see shortly (using the union bound) that this is essentially as good as knowing U_1 exactly. Therefore, to separate out things clearly, we can assume for the moment that at the decoder we have U_1 as a side information.

We have $U_2 = X_1 + U_1$ and $U_2 = X_2$. As before, we observe Y_1 and Y_2. If we did not have the extra term U_1 then we would simply have two conditionally independent observations of the same bit U_2. But note that U_1 is known, and since we always assume that the channel is symmetric this extra constant does not matter. E.g., if you consider the BEC. If we received Y_1 then we know that $Y_1 = U_1 + U_2$ and hence we can determine U_2 by adding the known term U_1. We call the channel seen by bit U_2 given the side information U_1, W^+ (as we will see this channel is better than the original channel, hence the plus).

$$W^+ : U_2 \to Y, U_1.$$

Let us summarize. The channel W^+ is equivalent to a channel where we send the same bit U_2 over two independent instances of the original channel W and we get to see both outputs. If we think of the erasure channel then the resulting channel is again an erasure channel but with an erasure probability ϵ^2, since we only get an erasure if *both* bits have been erased. Note that $\epsilon^2 \leq \epsilon$. So W^- is *better* than the original channel W.

What is the sum capacity of these two channels. I.e., what is $C(W^-) + W(W^+)$? For the BEC(ϵ) we have

$$C(W^-) + C(W^+) = 1 - [1-(1-\epsilon)^2] + 1 - [\epsilon^2] = 2(1-\epsilon) = 2C(W).$$

In words, the sum of the capacities of these two channels is equal to two times the capacity of the original channel. So no capacity was lost. This is not entirely surprising. The matrix G_2 is invertible after all. We did this computation for the BEC. But the same computation can be done in general, using the chain rule of the so-called *mutual information.* If you have taken a course in information theory, here is the general proof:

$$\begin{aligned}2I(W) &= I(X_1;Y_1) + I(X_2;Y_2)\\ &= I(X_1,X_2;Y_1,Y_2)\\ &= I(U_1,U_2;Y_1,Y_2)\\ &= I(U_1;Y_1,Y_2) + I(U_2;Y_1,Y_2 \mid U_1)\\ &= I(W^-) + I(W^+).\end{aligned}$$

In the third step we have used that the map $(U_1, U_2) \leftrightarrow (X_1, X_2)$ is 1-1, in the fourth step we used the chainn rule of mutual information and in the last step we used the definitions of the two channels W^- and W^+.

Let us now write down a little bit more formally why we can assume in the second step that we have U_1 as side information. Let $\hat{U}_1((Y_1, Y_2)$ and $\hat{U}_2((Y_1, Y_2, U_1)$ denote the functions which estimate the two bits based on the input. Then we have

$$\begin{aligned}&\mathbb{P}\{\hat{U}_1(Y_1,Y_2) \neq U_1 \vee \hat{U}_2(Y_1,Y_2,\hat{U}_1(Y_1,Y_2)) \neq U_2\}\\ =&\mathbb{P}\{\hat{U}_1(Y_1,Y_2) \neq U_1 \vee \hat{U}_2(Y_1,Y_2,U_1) \neq U_2\}\\ \leq&\mathbb{P}\{\hat{U}_1(Y_1,Y_2) \neq U_1\} + \mathbb{P}\{\hat{U}_2(Y_1,Y_2,U_1) \neq U_2\}.\end{aligned}$$

So we see that if we are interested in the total probability of making a mistake in either estimating U_1 or U_2 the two models are in fact equivalent and the union bound is an upper bound on this probability. In the above lines we wrote down the estimates as functions of the instantaneous outputs. But of course, as we mentioned, what we are really doing is to code each branch over long blocks, so you should think of each of these quantities are long vectors. But the above statement stays true if we replace the scalars for vectors.

We will need only one extra ingredient. We have already discussed that the *sum* capacity is equal to two times the capacity of the original channel. In other words, $\frac{1}{2}(C(W^+) + C(W^-)) = C(W)$. What about the *difference* $\frac{1}{2}(C(W^+) + C(W^-))$? We have

$$\begin{aligned}\frac{1}{2}(C(W^+) - C(W^-)) &= \frac{1}{2}((C(W^+) - C(W)) + (C(W) - C(W^-)))\\ &= \frac{1}{2}(2(C(W^+) - C(W)))\\ &= C(W^+) - C(W).\end{aligned}$$

For the BEC this quantity, call it $f(\epsilon)$, is equal to $1 - \epsilon^2 - 1 + \epsilon = \epsilon(1 - \epsilon)$. This is shown in Fig 7.1. Note that this function is strictly positive for $\epsilon \in (0, 1)$ with a derivative of 1 and -1 at $\epsilon = 0$ and $\epsilon = 1$, respectively. For the general case one can also show that the equivalent function (as a function of the entropy (entropy is one minus capacity) of the channel) is strictly positive for any channel as long as the entropy is in $(0, 1)$ and one can derive a universal lower bound. This lower bound has the same derivatives at the two end-points as the function $f(\epsilon)$ in Fig. 7.1, namely 1 and -1, respectively. Let us now compare the second moments. I.e., let us compare $C(W)^2$ to $\frac{1}{2}(C(W^-)^2 + C(W^+)^2)$.

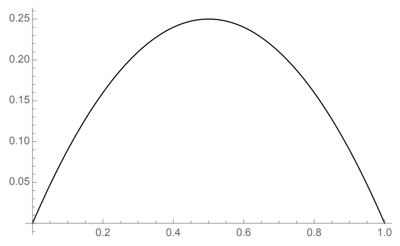


Figure 7.1: The function $f(\epsilon)$ corresponding to $\frac{1}{2}(C(W^+) - W(W^-))$ for the BEC.

Note that $(a^2+b^2) = (a+b)^2 + (a-b)^2$. Hence, we have

$$\begin{aligned}\frac{1}{2}(C(W^-)^2 + C(W^+)^2) &= (\frac{1}{2}(C(W^-) + C(W^+))^2 + (\frac{1}{2}(C(W^-) - C(W^+))^2 \\ &= C(W)^2 + \epsilon^2(1-\epsilon)^2,\end{aligned}$$

where we assumed that W is the BEC(ϵ). In words, the second moment is increased by the quantity $\epsilon^2(1-\epsilon)^2$, which is a strictly positive quantity unless the starting channel is trivial (perfect or completely useless channel).

Let us summarize what we have done at this point. We have started with a channel W that was a BEC(ϵ). We have first *combined* two such channels and then *split* them again into two channels by performing *successive decoding*. We called the two channels W^- and W^+ and the were both BECs, the first being a BEC($\epsilon(2-\epsilon)$) and the second being a BEC(ϵ^2). We have seen that the *average* capacity stays preserved but that *second moment* increases by strictly positive quantity unless the original parameter ϵ is either 0 or 1.

7.5 Repeating the Basic Step

Repeat now the basic step for the two channels W^- and W^+. This way we get four channels W^{--}, W^{-+}, W^{+-}, and W^{++}. Using the same arguments we see that

$$\frac{1}{4}[C(W^{--}) + C(W^{-+}) + C(W^{+-}) + C(W^{++})] = \frac{1}{2}[C(W^-) + C(W^+)] = C(W),$$

i.e., the average capacity of these four channels is equal to the average capacity of the two channels, which in turn is equal to the capacity of the original channel. But

$$\frac{1}{4}[C(W^{--})^2 + C(W^{-+})^2 + C(W^{+-})^2 + C(W^{++})^2] > \frac{1}{2}[C(W^-)^2 + C(W^+)^2] > C(W)^2,$$

i.e., the second moment is strictly increasing at every step.

At this point the idea is hopefully clear. Repeat the basic constructing n times. In this way we have transformed $N = 2^n$ original channels into N "synthetic" channels. The average capacity of these N sythetic channels is equal to the capacity of the original channel but their second moment is strictly increasing at every step of the process.

Let ν_i, $i \geq 0$, denote the sequence of average second moments. As we discussed this sequence is strictly increasing,

$$\nu_0 < \nu_1 < \nu_2 < \cdots .$$

But note that this sequence is also upper bounded by 1 since each term is upper bounded by 1 (each term represents the capacity of a binary-input channel). Therefore this sequence converges and so $\nu_{n+1} - \nu_n$ converges to 0.

Let $\sigma_n(\delta)$ denote the fraction of channels after n polarization steps whose parameters are in the range $(\delta, 1-\delta)$. Then

$$\nu_{n+1} \geq \nu_n + \delta^2(1-\delta)^2\sigma_n(\delta).$$

Rewriting this expression we see that

$$\frac{\nu_{n+1} - \nu_n}{\epsilon^2(1-\epsilon)^2} \geq \sigma_n(\delta).$$

Since, as we stated, $\nu_{n+1} - \nu_n$ converges to 0, it follows that $\sigma_n(\delta)$ converges to 0.

We say that (almost) all channels polarize to either perfect or useless channels.

How many channels converge to essentially perfect channels, i.e., their erasure parameter is vanishingly small? Since the average capacity stays preserved it is easy to see that this fraction is equal to $1-\epsilon$, i.e., roughly $(1-\epsilon)N$ out of all N channels have a vanishingly small erasure probability.

7.6 Encoding

7.7 Decoding

7.8 Historical Notes and References

Bibliography

[1] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering.* John Wiley, 1965. Reprinted by Waveland Press starting 1990.

[2] J. G. Proakis and M. Salehi, *Digital Communications.* McGraw-Hill, 5 ed., 2007.

[3] R. G. Gallager, *Principles of Digital Communications.* Cambridge University Press, 2008.

[4] U. Madhow, *Fundamentals of Digital Communication.* Cambridge University Press, 2008.

[5] A. Lapidoth, *A Foundation in Digital Communications.* Cambridge University Press, 2009.

[6] W. Feller, *An Introduction to Probability Theory and Its Applications, Volume 1.* New York: Wiley, 1950.