Lecture 1: Introduction to Digital Communication

1 Introduction and Objectives

The digital communication industry is an enormous and rapidly growing industry, roughly comparable in size to the computer industry. The objective of this course is to study those aspects of digital communication systems that are unique to these systems. That is, rather than focusing on hardware and software for these systems, which is much like hardware and software for many other kinds of systems, we focus on the fundamental system aspects of modern digital communication.

This is the first subject in the two-term sequence, 6.450 and 6.451. The two subjects are in the process of being integrated into a coherent year-long sequence, and we hope that the rough edges do not show too much. Our intention is that 6.450 may be taken on a stand-alone basis and be accessible to well-prepared undergraduates.

Digital communication is a field in which theoretical ideas have had an unusually powerful impact on actual system design. The basis of the theory was developed 53 years ago by Claude Shannon, and is called information theory. For the first 25 years or so of its existence, information theory served as a rich source of academic research problems and as a tantalizing suggestion that communication systems could be made more efficient and more reliable by using these approaches. By the mid 1970's, mainstream systems using information theoretic ideas began to be widely implemented, for two reasons. First, by that time there were a sizable number of engineers who understood both information theory and communication system development. Second, the low cost and increasing processing power of digital hardware made it possible to implement increasingly sophisticated algorithms.

As you learn about digital communication systems and their conceptual basis in information theory, you will come to appreciate that these ideas require a fairly deep understanding of somewhat abstract concepts. Doing the problem sets is an important part of developing this understanding, but it is not sufficient. In many undergraduate engineering subjects, the emphasis (reinforced by the grading policy) is on learning to grind through the solution of a set of types of problems represented by prescribed equations ("plug and chug"). Here, as is appropriate for a graduate class, the focus is much more on the connections between engineering and theory.

The relationship between theory, problem sets, and engineering/design is rather complex. The theory deals with relationships and analysis for *models* of real systems. A good theory (and information theory is one of the best) allows for simple analysis of simplified models. It also provides structural principles that allow insights from these simple models to be applied to more complex and realistic models. Problem sets provide students with an opportunity to analyze these highly simplified models, and, with patience, to start to

understand the general principles. Engineering deals with making the approximations and judgment calls to create appropriate simple models, and from there to design workable systems.

The important point here is that engineering (at this level) cannot really be separated from theory. Engineering is necessary to theory in choosing appropriate models, and theory is necessary to engineering to create principles and quantitative results. Engineering sometimes becomes overly concerned with detail, and theory overly concerned with mathematical niceties, but we shall try to avoid both these excesses here.

We urge you, when you finish an exercise, to spend some time thinking about what it means, how the solution technique might be extended to more general problems, whether the model used in the exercise makes much sense physically, etc. We will try to encourage this both by keeping the number of exercises limited and by making it necessary to achieve this deeper understanding about one exercise in order to do the next.

In terms of depth, this subject is more like a graduate subject than undergraduate. However, it requires only a knowledge of elementary probability (6.041) and linear systems (6.003), so it can be understood at the undergraduate level. Learning to think more deeply about the material, in addition to solving the exercises, will be invaluable both in the workplace and in graduate work.

This is an experimental subject, both in terms of content and approach. Some undergraduate courses aim to make the student familiar with a large variety of different systems that have been implemented historically. In our opinion such an approach does not help much in understanding the new systems being designed currently, and provides little insight into the different design choices that might be made. Our objective here is to develop the relatively small number of underlying principles guiding all of these systems, with the hope that you can then understand the details of any system of interest on your own.

We need your help in this experiment. Let us know, by direct comment or by e-mail, what interests you, what turns you off, what is too hard, what seems too easy, etc. We think we know what you should learn, but are less certain about how to help you learn it.

2 Networks and Layering

The communication systems that you use every day— *e.g.*, the telephone system, the Internet— are networks of incredible complexity, made up of an enormous variety of equipment made by different manufacturers at different times following different design principles. Such complex networks need to be based on some simple architectural principles in order to be understood, managed and maintained.

Two fundamental architectural principles of communication networks are *standardized interfaces* and *layering*.

A standardized interface allows the user or equipment on one side of the interface to ignore all details about the other side of the interface except for certain specified characteristics. For example, the standard interface in the telephony system for decades has been the 4

KHz voice channel. You can plug a telephone into a wall plug anywhere in the world (subject to further electrical, mechanical and dialing interface standards), and expect to be able to send a nominal 4 KHz voice signal anywhere in the Public Switched Telephone Network (PSTN). Other standardized interfaces are defined at various levels within the telephone network.

The trend in recent decades has been to make all standardized interfaces digital, at least inside the telephone and other networks, and increasingly at the user interface as well. Basically, one sends and receives bits, regardless of the ultimate application. Thus electronic communication is becoming synonymous with digital communication.

The idea of layering in communication networks is to break up communication functions into separable modules, or "layers," which (a) communicate with higher and lower layers via standardized interfaces, and (b) communicate through networks (via lower layers) to counterpart modules ("peer layers") elsewhere in the network. Often peer modules occur in pairs, each containing both transmit and receive functions.

For example, a modem (modulator-demodulator) (a) receives a sequence of bits from a higher-level application or user; (b) generates from the received data a modulated signal for transmission through a standardized interface over a channel; (c) receives from the channel a signal that is a more-or-less faithful replica of a signal transmitted by a remote modem; and (d) generates from the received signal a demodulated bit sequence that is supposed to be a more-or-less faithful replica of the bit sequence that entered the remote modem.

Other examples of paired modules at the same layer are: telephones; encryptor/decryptors; speech encoder/decoders; image encoder/decoders; error-correction encoder/decoders; etc.

In more complicated situations, there may be more than two paired elements in a layer. For example, in a wireless system each receiver may be listening to many transmitters. The multi-access control issues that arise in such cases are beyond the scope of this course; different aspects of these issues are addressed in, *e.g.*, 6.451 (Principles of Digital Communication II), 6.263 (Data Networks), and 6.441 (Information Theory).

3 Block Diagram of a Generic Point-to-Point Digital Communication System

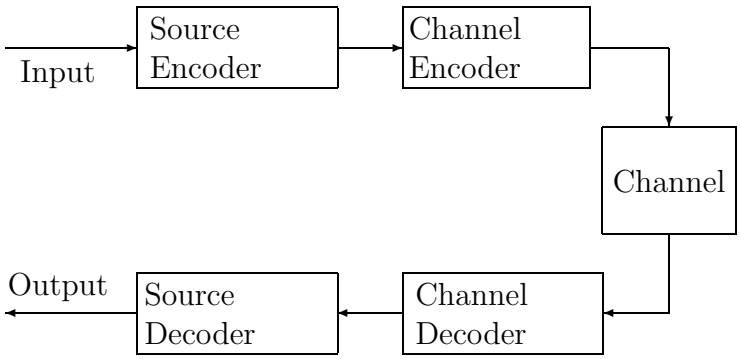
This course focuses on two fundamental problems of data communication: source coding and channel coding.

The source coding problem is the problem of efficient representation of source signals— *e.g.*, speech waveforms, image waveforms, text files— as a sequence of bits for transmission through a digital network. Of course we must keep in mind the paired problem of source decoding: conversion of the bit sequence (possibly corrupted) back into a more-or-less faithful replica of the original source signal.

The channel coding problem is the problem of efficient transmission of a sequence of bits

through a lower-layer channel— *e.g.*, a 4 KHz telephone channel or a wireless channel. Again we must very much keep in mind how we intend to recover a more-or-less faithful replica of the original bit sequence from the channel output in the remote receiver, despite the distortions that may be introduced by the channel.

In a simple point-to-point channel, illustrated below in the classic "Figure 1" of every information theory text, these are the only two layers. The output bit sequence of the source encoder is the input bit sequence of the channel encoder, and the output bit sequence of the channel decoder is the input bit sequence of the source decoder.



"Figure 1." Separation of source and channel coding.

The astute student may ask at this point whether anything is lost by separate source and channel coding as shown in "Figure 1." Could joint source-channel coding achieve better performance or reduced complexity?

One of the remarkable basic theorems of information theory is that, in a sense, nothing is lost in terms of performance. If a source signal can be communicated through a given point-to-point channel within some level of distortion by any means whatsoever, then separate source and channel coding can also be designed to stay within that same level of distortion. We will give some insight into this **source-channel coding separation theorem** during this course; 6.441 (Information Theory) explains it more fully.

The source-channel coding separation theorem does not rule out the possibility of smaller delay or lower complexity using joint source-channel coding. Also, there are more complex network scenarios in which this theorem does not hold.

However, the advantages of standardized binary interfaces and of layering, discussed in the previous section, are so overwhelming that joint source-channel coding is rarely attempted in practice.

3.1 Input

We now discuss the various elements of "Figure 1" in more detail.

The input is the source signal. It might be a sequence of symbols such as letters from the English or Chinese alphabet, binary symbols from a computer file, etc. Alternatively, the input might be a waveform, such as a voice signal from a microphone, the output of a sensor, a video waveform, etc. Or, it might be a sequence of images such as X-rays, photographs, etc.

Whatever the source signal is, we will model it as a sample function of a random process. This is one of the reasons why probability is an essential prerequisite for this subject. It is not obvious why inputs to communication systems should be modeled as random, and in fact this was not appreciated before Shannon developed information theory in 1948.

The study of communication before that time (and well after that time) was based on Fourier analysis, which basically studies the effect of passing sine waves through various kinds of systems and components. We will start our study of channels with this kind of analysis (often called Nyquist theory in the context of digital communication) to develop basic results about sampling, intersymbol interference, and bandwidth.

However, Shannon's view was that if the recipient knows that a sine wave of a given frequency is to be communicated, why not simply regenerate it at the output rather than send it over a long distance? Or, if the recipient knows that a sine wave of unknown frequency is to be communicated, why not simply send the frequency rather than the entire waveform?

The essence of Shannon's viewpoint is that we should focus on the set of possible inputs from the source rather than any particular input. The objective then is to transform each possible input into a transmitted signal in such a way that each possible transmitted signal can be distinguished from the others at the output. A probability measure is needed on this set of possible inputs to distinguish typical inputs from abnormal inputs. We shall see in the rest of this subject how this point of view drives the processing of the inputs as they pass through a communication system.

3.2 Source Coding

The source encoder in "Figure 1" has the function of converting the input from its original form into a sequence of bits. We have already discussed many of the reasons for conversion to a bit sequence: standardized interfaces, layering, and the source-channel coding separation theorem.

The simplest source coding techniques involve simply representing the source signal by a sequence of symbols from some finite alphabet, and then coding the alphabet symbols into fixed-length blocks of bits. For example, letters from the 27-symbol English alphabet (including a SPACE symbol) may be encoded into 5-bit blocks. Or, upper-case letters, lower-case letters, and a great many other special symbols may be converted into 8-bit blocks ("bytes") using the standard ASCII code.

The most straightforward approach to converting an analog waveform to a bit sequence, called analog to digital (A/D) conversion, is first to sample the source at a sufficiently high rate (called the "Nyquist rate"), and then to quantize it sufficiently finely for adequate

reproduction. For example, in standard voice telephony, the voice waveform is filtered to a bandwidth of less than 4 KHz and sampled 8000 times per second; each sample is then quantized into one of 256 levels and represented by an 8-bit byte. This yields a source coding bit rate of 64 kb/s.

Beyond the basic objective of conversion to bits, the source encoder often has the further objective of doing this as efficiently as possible— *i.e.*, transmitting as few bits as possible, subject to the need to reconstruct the input adequately at the output. In this case source encoding is often called data compression. For example, modern speech coders can encode telephone-quality speech at bit rates of the order of 6-16 kb/s rather than 64 kb/s.

We shall study data compression of discrete sources in the next few lectures.

3.3 Digital Interface

The interface between the source coding and channel coding layers is a sequence of bits. However, this simple characterization does not tell the whole story.

Some sources, such as voice or video, produce a virtually unending signal that must be encoded into a sequence of bits, usually at some constant rate measured in bits per second (b/s). These sources are often called *virtual circuit* sources. For other sources, such as email or data files, the data are encoded into packets, each consisting of a finite sequence of bits. These packets are usually presented to the interface as a unit. These sources are known as *packet sources*. Other sources are more complicated combinations of both virtual circuit and packet sources. Thus the bits coming into the interface from the source have some time structure, possibly constant rate, possibly irregularly arriving packets that must be queued, etc.

There are a number of protocols required to indicate the starting and stopping of packets and the synchronization of virtual circuit sources. These issues are treated in the Data Network course (6.263), but are somewhat disjoint from the topics here, so we simply recognize that these issues exist and assume they are resolved in some way. The issue of interest here is simply that of mapping the source output into a sequence of bits. Our objective in source coding, then, is to minimize the number of bits required (in some average sense to be discussed later). Typical packets are long enough that we can ignore their finite duration (to first order), and simply model all the sources of interest as unending waveforms or sequences of letters. As we discuss later, this source coding must be consistent with the required quality of service (*i.e.*, distortion, encoding delay, etc.).

The channel encoder at the other side of the digital interface must be capable of keeping up with the stream of incoming bits, encoding and transmitting them so that they can be recreated at the decoder with a suitably small error probability. In other words, the channel encoder and decoder must be capable of operating at the same bit rate as the source encoder.

In view of the source-channel separation theorem, we would like to look at the issue of understanding channel encoders and decoders as that of trying to *maximize* the rate at which bits can be transmitted reliably over the channel. At that point, the interface

problem becomes simple: if the channel encoder can successfully transmit bits at a rate at least as large as the rate produced by the source coder, then communication will be successful. Otherwise, we need to find a better channel or improve either the source or channel encoding.

It is unreasonable to expect that the channel coding layer will be able to transmit data without errors. The specification for this layer must include a data quality parameter, such as probability of error per bit or per packet. The usual objective for channel coding is to make this error probability very small, but it usually can not be reduced to 0.

The source coding layer must therefore be able to cope with a small but nonzero error probability. For certain kinds of source signals, such as speech or video, it may be acceptable simply to allow occasional errors in reconstruction of the signal, if they are sufficiently infrequent. Or, better, one can use an *error-detection code* to detect that an error has been made, and then perhaps perceptually mask the error.

For other kinds of sources, such as computer files, no transmission errors can be tolerated. The usual approach in such cases is to protect each packet with an error-detection code, and then at the receiver to request a retransmission of any packet in which an error has been detected. This kind of *automatic-repeat-request* (ARQ) system of course requires a feedback path and the ability to tolerate the consequent round-trip delay. However, if these are not problems, then an ARQ system can easily be engineered to provide nearly error-free transmission at the cost of a small amount of overhead and some variable delay.

Evidently *delay* is another important quality parameter of the channel coding layer—both absolute delay and, especially in networks, variability of delay.

In network scenarios, the encoded packets often contain other layers of protocol information, such as addressing data, ARQ overhead, and so forth. Again, we will not deal with such "network-layer" issues in this course.

3.4 Channels

We next discuss the channel in a generic digital communication system, before considering channel coding.

In general, the channel is that part of the communication medium that is given and not under the control of the designer. Thus, to a source code designer, the channel might be a digital channel with bits as input and output; to a telephone-line modem designer, it might be a 4 KHz voice channel; to a cable modem designer, it might be a physical coaxial cable of up to a certain length, with certain bandwidth restrictions.

For a channel code designer, the channel is often a "physical channel;" *e.g.*, a pair of wires, a coaxial cable, or an optical fiber going from the source location to the destination. It also might be open space between source and destination over which electromagnetic radiation can carry signals.

As in the study of signals and systems, we view a channel in terms of its input, its output, and some description of how the input affects the output, which in this course will usually

be a probabilistic description. If a channel were simply a linear time-invariant system (*e.g.*, a filter), then it could be completely characterized by its impulse response or frequency response. However, the channels that we look at here (and channels in practice) always have an extra ingredient— noise.

Suppose that there were no noise and a single input voltage level could be communicated exactly. Then, representing that voltage level by its infinite binary expansion, we would in principle be able to transmit an infinite number of binary digits by transmitting a single real number. This is ridiculous in practice, of course, precisely because noise limits the number of bits that can be reliably distinguished. Again, it was Shannon in 1948 who realized that noise provides the fundamental limitation to performance in communication systems.

The most common channel model involves a waveform input $X(t)$, an added noise waveform $Z(t)$, and a waveform output $Y(t) = X(t) + Z(t)$ that is the sum of the input and the noise, as shown in Figure 2. Each of these waveforms are viewed as stochastic processes. We study these stochastic processes later, but for now they can be viewed simply as waveforms. The noise $Z(t)$ is often modeled as white Gaussian noise (also to be studied and explained later). The input is usually constrained in power and in bandwidth.

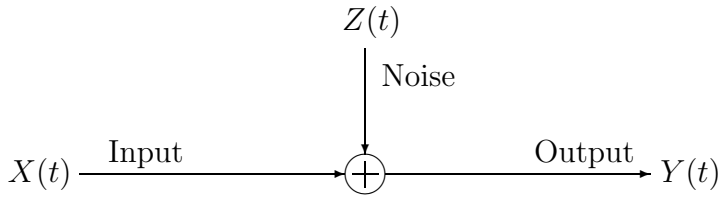


Figure 2. An additive white Gaussian noise (AWGN) channel.

Observe that for any channel with input $X(t)$ and output $Y(t)$, we could define the noise to be $Z(t) = Y(t) - X(t)$, so there must be something more to an additive-noise channel model than what is expressed in Figure 2. The missing ingredient is that the noise must be statistically independent of the input. Thus, whenever we say that a channel is subject to additive noise, we mean implicitly that the noise is statistically independent of the input.

In a somewhat more general model, called a *linear Gaussian channel*, the input waveform $X(t)$ is first filtered in a linear filter with impulse response $h(t)$, and then independent white Gaussian noise $Z(t)$ is added, as shown in Figure 3, so that the channel output is

$$Y(t) = X(t) * h(t) + Z(t),$$

where "$*$" denotes convolution. Note that Y at time t is a function of X over a range of times, *i.e.*,

$$Y(t) = \int_{\tau=-\infty}^{\infty} X(t-\tau)h(\tau)\,d\tau + Z(t)$$

The noise may even be non-white here, but, as we shall see later, this further "generalization" is actually not more general.

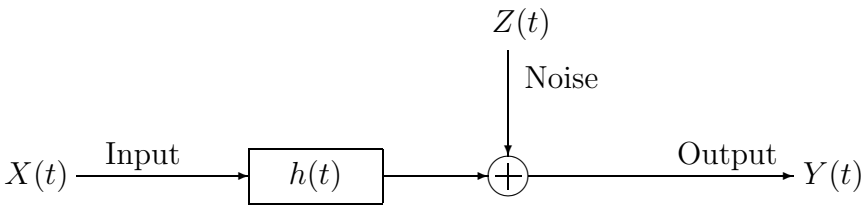


Figure 3. Linear Gaussian channel model.

This linear Gaussian channel model is typically not a bad model for wireline communication or for line-of-sight wireless communication. When engineers, journals, or texts fail to say what kind of channel they are talking about, this model is a safe bet.

The linear Gaussian channel model is rather poor for non-line-of-sight mobile communication. Here, multiple paths usually exist from source to destination, and these paths usually change in time in a way best modeled as random. A better model for mobile communication is to filter the input by a randomly-time-varying linear filter that represents the multiple paths as they change in time, as shown in Figure 4.

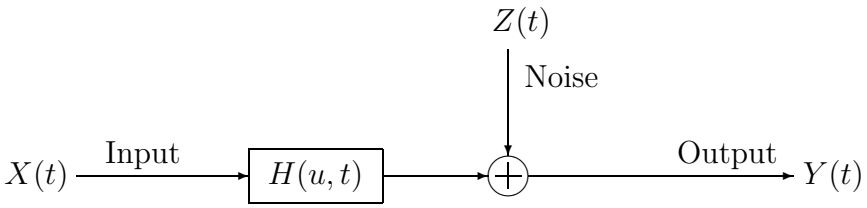


Figure 4. Time-varying linear Gaussian channel model.

Here the output is given by $Y(t) = \int_{u=-\infty}^{\infty} X(t-u)H(u,t)du + Z(t)$. These randomly varying channels will be studied in detail next term in 6.451. Students should be careful reading introductory literature about mobile communication, since it often ignores the time-varying nature of the channel.

3.5 Channel Coding

The channel encoder box in Figure 1 has the function of mapping the binary sequence at the source/channel interface into channel inputs. The channel inputs might be waveforms, or might be discrete sequences, but to be specific here, we view the channel as the linear Gaussian channel of the previous subsection. The general objective here is to map binary inputs at the maximum bit rate possible into waveforms such that the channel decoder can recreate the original bits with low probability of error. One simple approach to this problem is called modulation and demodulation.

In the simplest modulators, each bit is independently mapped into one of two possible waveforms. For example, in a binary frequency-shift-keyed (FSK) modem, each bit chooses one of two frequencies. On an optical channel, a laser beam may be turned ON or OFF.

In multi-level modulation, a string of m bits may select one of $M = 2^m$ waveforms. In eight-level pulse-amplitude modulation (8-PAM), for example, the input bit sequence is divided into successive triples of binary digits. Each of the eight possible combinations of three binary digits is then mapped into a different numerical signal level (*e.g.*, $-7, -5, -3, -1, 1, 3, 5, 7$). The resulting sequence of signal levels then modulates the amplitudes of a certain given modulation waveform. As another example, in eight-level phase-shift-keying (8-PSK), the eight signals might select one of eight phases $(0, \pi/4, \ldots)$ of a carrier sine wave in each signaling interval.

It is easy to think of many ways to map binary digits into signals and then signals into waveforms. We shall find that there is a simple geometric "signal-space" approach to looking at these various combinations in an integrated way.

Because of the noise on the channel, the received signal is almost certainly not equal to one of the possible transmitted signals. A major function of the demodulator is that of detection. The detector attempts to choose which possible input signal is most likely to have given rise to the given received signal. The geometric signal-space approach will be invaluable in understanding the detection problem.

When we look more carefully at demodulation, we will encounter a number of important details. One of these is that when a sequence of signals is modulated on a linear Gaussian channel, there will usually be intersymbol interference due to filtering. Dealing with intersymbol interference is called "equalization." Also, the demodulator must track the symbol timing of the transmitter in order to know when to detect each symbol. Finally, if the transmitter uses carrier modulation, then the receiver must track the carrier frequency and phase. We will look at these issues after clearly understanding the geometric structure of the basic modulation and demodulation problem.

3.6 Error Correction

Frequently the error probability incurred with simple modulation and demodulation is too high. One possible solution is to separate the channel coder into two layers, first an error-correcting code, and then a simple modulator.

As a very simple example, the bit rate into the channel encoder could be reduced by a factor of 3, and then each binary input could be repeated 3 times before entering the modulator. If at most one of the 3 binary digits coming out of the demodulator were incorrect, it could be corrected, thus reducing the error probability of the system at a considerable cost in data rate.

The scheme above is a very simple example of error-correction coding, namely repetition coding with majority-rule decoding. Unfortunately, with this scheme, it is possible to get very reliable communication only by greatly slowing down the rate at which the original bits are transmitted.

What Shannon showed was the very unintuitive fact that more sophisticated coding schemes can achieve arbitrarily low error probabilities without lowering the data rate below a certain data rate that depends on the channel being used, called the *channel*

capacity. In this subject, we will not prove this result, or even describe it very precisely, although we will provide various types of insight into coding and decoding. This *channel coding theorem* is the centerpiece of 6.441.

For example, for an AWGN channel with bandwidth W and input power P, if the one-sided power spectral density of the noise is N_0, then Shannon showed that the channel capacity in bits per second is

$$C = W \log_2 \left(1 + \frac{P}{N_0 W}\right).$$

Only in the past few years have channel coding schemes been developed that can closely approach this channel capacity. When we discuss channel coding, channel capacity will be our benchmark.

Until about 20 years ago, channel coding usually involved a two layer system similar to that above, where an error-correcting code is followed by a modulator. At the receiver, the waveform is first demodulated, and then the error correction code is decoded. More recently, it has been recognized that coding and modulation should be considered as a unit, in schemes called coded modulation. Moreover, coding for the AWGN channel is a problem that is best viewed in the geometric signal-space context. When we discuss channel coding, we will take a geometric approach.

In this course, we shall develop each of the above components of a digital communication system in detail. Of course, as we do so, we will understand better how all the pieces are related.