PHYS 7895: Quantum Information Theory Spring 2017

Lecture 3

Prof. Mark M. Wilde *Scribe: Sushovit Adhikari*

1 Overview

In this lecture, we talk about the issue of transmission of information over a noisy classical channel. We will start with a simple error correction code example and look at a simple decoding algorithm for it. We will describe the information processing task for channel coding and at the end look at an overview of Shannon's Channel Capacity Theorem. Now we begin with a standard example—transmitting a single bit of information over a noisy bit-flip channel.

2 Simple Error Correction code (Repetition code)

We start with Alice (sender) and Bob (receiver). We also assume that a noisy classical channel connects them, so that information transfer is not reliable. Alice and Bob realize that a noisy channel is not as expensive as a noiseless one, but it still is expensive for them to use. For this reason, they would like to maximize the amount of information that Alice can communicate reliably to Bob, where reliable communication implies that there is a negligible probability of error when transmitting this information.

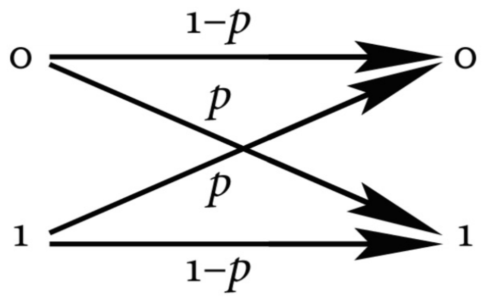

Figure 1: The figure depicts the action of the bit-flip channel. It preserves the input bit with probability $1 - p$ and flips it with probability p.

In this lecture, for simplicity, we assume that the channel is a simple noisy bit-flip channel as shown in figure above. This channel flips the input bit with probability p and leaves it unchanged with probability $1 - p$ (see Figure 1). When using the channel multiple times, we assume that it behaves independently from one use to the next. For this reason, we say that multiple uses of the channel is an i.i.d. channel. This assumption will again be important when we go to the asymptotic regime of a large number of uses of the channel.

Channel Output	**Probability**
000	$(1-p)^3$
001, 010, 100	$p\,(1-p)^2$
011, 110, 101	$p^2\,(1-p)$
111	p^3

Table 1: The first column gives the eight possible outputs of the noisy bit-flip channel when Alice encodes a “0” with the majority vote code. The second column gives the corresponding probability of Bob receiving the particular outputs.

Suppose that Alice and Bob just use the channel as is—Alice just sends plain bits to Bob. This scheme works reliably only if the probability of bit-flip error vanishes. So, Alice and Bob could invest their best efforts into engineering the physical channel to make it reliable. But, generally, it is not possible to engineer a classical channel this way for physical or logistical reasons. For example, Alice and Bob may only have local computers at their ends and may not have access to the physical channel because the telephone company may control the channel.

2.1 Encoding Algorithm

To reduce the probability error of transmitting, Alice and Bob can employ a “systems engineering” solution to this problem rather than an engineering of the physical channel. They can redundantly encode information in a way such that Bob can have a higher probability of determining what Alice is sending, effectively reducing the level of noise on the channel. A simple example of this systems engineering solution is the three-bit majority vote code. Alice and Bob employ the following encoding:

$$0 \to 000, \qquad 1 \to 111, \tag{1}$$

where both “000” and “111” are *codewords*. Alice transmits the codeword “000” with three independent uses of the noisy channel if she really wants to communicate a “0” to Bob and she transmits the codeword “111” if she wants to send a “1” to him. The *physical* or *channel* bits are the actual bits that she transmits over the noisy channel, and the *logical* or *information* bits are those that she intends for Bob to receive. In our example, “0” is a logical bit and “000” corresponds to the physical bits.

The rate of this scheme is 1/3 because it encodes one information bit. The term “rate” is perhaps a misnomer for coding scenarios that do not involve sending bits in a time sequence over a channel. We may just as well use the majority vote code to store one bit in a memory device that may be unreliable. Perhaps a more universal term is *efficiency*.

2.2 Decoding Algorithm

Even with the encoding algorithm which will reduce the error probability, it does not always transmit these codewords without error. So, Bob has to use a certain algorithm to decode in case of error. He simply takes a *majority vote* to determine the transmitted message—he decodes as “0” if the number of zeros in the codeword he receives is greater than the number of ones.

We now analyze the performance of this simple "systems engineering" solution. Table 1 enumerates the probability of receiving every possible sequence of three bits, assuming that Alice transmits a "0" by encoding it as "000." The probability of no error is $(1-p)^3$, the probability of a single-bit error is $3p(1-p)^2$, the probability of a double-bit error is $3p^2(1-p)$, and the probability of a total failure is p^3. The majority vote solution can "correct" for no error and it corrects for all single-bit errors, but it has no ability to correct for double-bit and triple-bit errors. In fact, it actually incorrectly decodes these latter two scenarios by "correcting" "011", "110", or "101" to "111" and decoding "111" as a "1." Thus, these latter two outcomes are errors because the code has no ability to correct them. We can employ similar arguments as above to the case where Alice transmits a "1" to Bob with the majority vote code.

When does this majority vote scheme perform better than no coding at all? It is exactly when the probability of error with the majority vote code is less than p, the probability of error with no coding. Letting e denote the event that an error occurs, the probability of error is equal to the following quantity:

$$\Pr(e) = \Pr(e|0)\Pr(0) + \Pr(e|1)\Pr(1). \tag{2}$$

Our analysis above suggests that the conditional probabilities $\Pr(e|0)$ and $\Pr(e|1)$ are equal for the majority vote code because of the symmetry in the noisy bit-flip channel. This result implies that the probability of error is

$$\Pr(e) = 3p^2(1-p) + p^3 \tag{3}$$
$$= 3p^2 - 2p^3, \tag{4}$$

because $\Pr(0) + \Pr(1) = 1$. We consider the following inequality to determine if the majority vote code reduces the probability of error:

$$3p^2 - 2p^3 < p. \tag{5}$$

This inequality simplifies as

$$0 < 2p^3 - 3p^2 + p \tag{6}$$
$$\therefore 0 < p(2p-1)(p-1). \tag{7}$$

The only values of p that satisfy the above inequality are $0 < p < 1/2$. Thus, the majority vote code reduces the probability of error only when $0 < p < 1/2$, i.e., when the noise on the channel is not too much. Too much noise has the effect of causing the codewords to flip too often, throwing off Bob's decoder.

The majority vote code gives a way for Alice and Bob to reduce the probability of error during their communication, but unfortunately, there is still a non-zero probability for the noisy channel to disrupt their communication. Is there any way that they can achieve reliable communication by reducing the probability of error to zero?

One simple approach to achieve this goal is to exploit the majority vote idea a second time. They can *concatenate* two instances of the majority vote code to produce a code with a larger number of physical bits. Concatenation consists of using one code as an "inner" code and another as an "outer" code. There is no real need for us to distinguish between the inner and outer code in this case because we use the same code for both the inner and outer code. The concatenation scheme for our case first encodes the message i, where $i \in \{0,1\}$, using the majority vote code. Let us label the codewords as follows:

$$\bar{0} \equiv 000, \qquad \bar{1} \equiv 111. \tag{8}$$

For the second layer of the concatenation, we encode $\bar{0}$ and $\bar{1}$ with the majority vote code again:

$$\bar{0} \to \bar{0}\bar{0}\bar{0}, \qquad \bar{1} \to \bar{1}\bar{1}\bar{1}. \tag{9}$$

Thus, the overall encoding of the concatenated scheme is as follows:

$$0 \to 000\ 000\ 000, \qquad 1 \to 111\ 111\ 111. \tag{10}$$

The rate of the concatenated code is 1/9 and smaller than the original rate of 1/3. A simple application of the above performance analysis for the majority vote code shows that this concatenation scheme reduces the probability of error as follows:

$$3[\Pr(e)]^2 - 2[\Pr(e)]^3 = O\left(p^4\right). \tag{11}$$

The error probability $\Pr(e)$ is in (4) and $O\left(p^4\right)$ indicates that the leading order term of the left-hand side is the fourth power in p.

The concatenated scheme achieves a lower probability of error at the cost of using more physical bits in the code. A first guess for achieving reliable communication is to continue concatenating. We can continue indefinitely with concatenating to make the probability of error arbitrarily small and achieve reliable communication, but the problem is that the rate approaches zero as the probability of error becomes arbitrarily small. In general, for n concatenations, the rate is equal to $\frac{1}{3^n}$ and the error probability is $O(p^{2n})$.

3 Shannon's Channel Coding Theorem

We now consider the case in which Alice wishes to transmit a larger set of messages with asymptotically perfect reliability, rather than merely sending "0" or "1." Suppose that she selects messages from a message set $[M]$ that consists of M messages:

$$[M] \equiv \{1, \ldots, M\}. \tag{12}$$

Before communication begins, Alice and Bob agree on a code: $\mathcal{C} \equiv \{x^n(m)\}_{m\in[M]}$ where $x^n(m)$ denotes each codeword corresponding to the message m and n is the length of the code. We do not really care much about the content of the actual message that she is transmitting. We just assume total ignorance of her message because we only really care about her ability to send any message reliably. The message set $[M]$ requires $\log(M)$ bits to represent it, where the logarithm is again base two. This number becomes important when we calculate the rate of a channel code.

Next, we would like to generalize the noisy channel that connects Alice to Bob. We used the bit-flip channel before, but this channel is not general enough for our purposes. A simple way to extend the channel model is to represent it as a conditional probability distribution involving an input random variable X and an output random variable Y:

$$\mathcal{N}: \qquad p_{Y|X}(y|x). \tag{13}$$

The last part of the model involves Bob receiving the corrupted codeword y^n over the channel and determining a potential codeword x^n with which it should be associated. Figure 2 displays Shannon's model of communication that we have described.

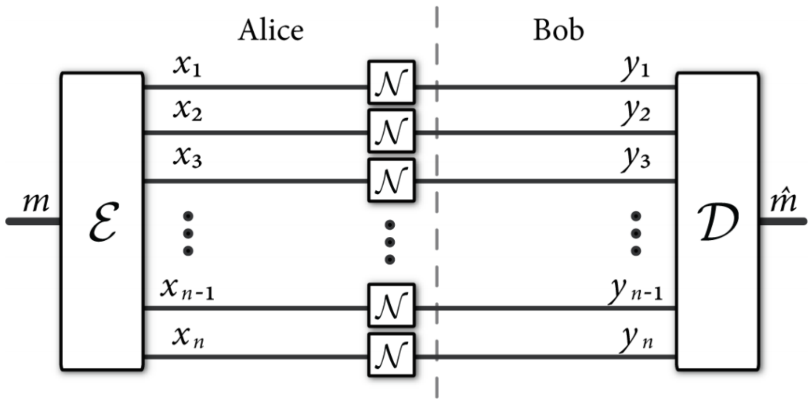

Figure 2: This figure depicts Shannon's idea for a classical channel code. Alice chooses a message m from a message set $[M] \equiv \{1, \ldots, M\}$. She encodes the message m with an encoding operation $\mathcal{E}$. This encoding operation assigns a codeword x^n to the message m and inputs the codeword x^n to a large number of i.i.d. uses of a noisy channel $\mathcal{N}$. The noisy channel randomly corrupts the codeword x^n to a sequence y^n. Bob receives the corrupted sequence y^n and performs a decoding operation $\mathcal{D}$ to estimate the codeword x^n. This estimate of the codeword x^n then produces an estimate $\hat{m}$ of the message that Alice transmitted. A reliable code has the property that Bob can decode each message $m \in [M]$ with a vanishing probability of error when the block length n becomes large.

We use the symbol $\mathcal{N}$ to represent this more general channel model. One assumption that we make about random variables X and Y is that they are discrete, but the respective sizes of their outcome sets do not have to match. The other assumption that we make concerning the noisy channel is that it is i.i.d. Let $X^n \equiv X_1 X_2 \cdots X_n$ and $Y^n \equiv Y_1 Y_2 \cdots Y_n$ be the random variables associated with respective sequences $x^n \equiv x_1 x_2 \cdots x_n$ and $y^n \equiv y_1 y_2 \cdots y_n$. If Alice inputs the sequence x^n to the n inputs of n respective uses of the noisy channel, a possible output sequence may be y^n. The i.i.d. assumption allows us to factor the conditional probability of the output sequence y^n:

$$\begin{aligned} p_{Y^n|X^n}(y^n|x^n) &= p_{Y_1|X_1}(y_1|x_1)\, p_{Y_2|X_2}(y_2|x_2) \cdots p_{Y_n|X_n}(y_n|x_n) && (14) \\ &= p_{Y|X}(y_1|x_1)\, p_{Y|X}(y_2|x_2) \cdots p_{Y|X}(y_n|x_n) && (15) \\ &= \prod_{i=1}^{n} p_{Y|X}(y_i|x_i)\,. && (16) \end{aligned}$$

We calculate the *rate* of a given coding scheme as follows:

$$\text{rate} \equiv \frac{\#\text{ of message bits}}{\#\text{ of channel uses}}. \tag{17}$$

In our model, the rate of a given coding scheme is

$$R = \frac{1}{n} \log(M)\,, \tag{18}$$

where $\log(M)$ is the number of bits needed to represent any message in the message set $[M]$ and n is the number of channel uses. The *capacity* of a noisy channel is the highest rate at which it can communicate information reliably.

3.1 Achievable Rate

Here we define an (n, R, ε) channel code for communication. Consider Figure 2 depicting a general protocol for communication over a classical channel $\mathcal{N} \equiv p_{Y|X}(y|x)$. Before communication begins, the sender Alice and receiver Bob have already established a codebook $\{x^n(m)\}_{m\in\mathcal{M}}$, where each codeword $x^n(m)$ corresponds to a message m that Alice might wish to send to Bob. If Alice wishes to send message m, she inputs the codeword $x^n(m)$ to the i.i.d. channel $\mathcal{N}^n \equiv p_{Y^n|X^n}(y^n|x^n)$. More formally, her map is some encoding $\mathcal{E}^n : \mathcal{M} \to \mathcal{X}^n$. She then exploits n uses of the channel to send $x^n(m)$. Bob receives some sequence y^n from the output of the channel, and he performs a decoding $\mathcal{D}^n : \mathcal{Y}^n \to \mathcal{M}$ in order to recover the message m that Alice transmits. The rate R of the code is equal to $\log_2|\mathcal{M}|/n$, measured in bits per channel use. The probability of error p_e for an (n, R, ε) channel code is bounded from above as

$$p_e \equiv \max_m \Pr\{\mathcal{D}^n(\mathcal{N}^n(\mathcal{E}^n(m))) \neq m\} \leq \varepsilon. \tag{19}$$

A communication rate R is *achievable* if there exists an $(n, R-\delta, \varepsilon)$ channel code for all $\varepsilon, \delta > 0$ and sufficiently large n. The channel capacity $C(\mathcal{N})$ of $\mathcal{N}$ is the supremum of all achievable rates. We can now state Shannon's channel capacity theorem:

Theorem 1 (Shannon Channel Capacity)**.** *The maximum mutual information $I(\mathcal{N})$ is equal to the capacity $C(\mathcal{N})$ of a channel $\mathcal{N} \equiv p_{Y|X}(y|x)$:*

$$C(\mathcal{N}) = I(\mathcal{N}) \equiv \max_{p_X(x)} I(X;Y), \tag{20}$$

where $I(X;Y) = H(X) + H(Y) - H(XY)$ is the mutual information of random variables X and Y.

4 Rough Sketch of the Achievability Proof

Alice chooses every symbol of every codeword independently at random according to random variable X with probability distribution $p_X(x)$. By the asymptotic equipartition theorem, it is highly likely that each of the codewords that Alice chooses is a typical sequence with sample entropy close to $H(X)$. In the coding scheme, Alice transmits a particular codeword x^n over the noisy channel and Bob receives a random sequence Y^n. The random sequence Y^n is a random variable that depends on x^n through the conditional probability distribution $p_{Y|X}(y|x)$. We would like a way to determine the number of possible output sequences that are likely to correspond to a particular input sequence x^n. A useful entropic quantity for this situation is the conditional entropy $H(Y|X) = H(XY) - H(X)$.

For now, just think of this conditional entropy as measuring the uncertainty of a random variable Y when one already knows the value of the random variable X. The conditional entropy $H(Y|X)$ is always less than the entropy $H(Y)$ unless X and Y are independent. This inequality holds because knowledge of a correlated random variable X does not increase the uncertainty about Y. It turns out that there is a notion of conditional typicality (depicted in Figure 3), similar to the notion of typicality, and a similar asymptotic equipartition theorem holds for conditionally typical sequences. This theorem also has three important properties. For each input sequence x^n, there is a corresponding conditionally typical set with the following properties:

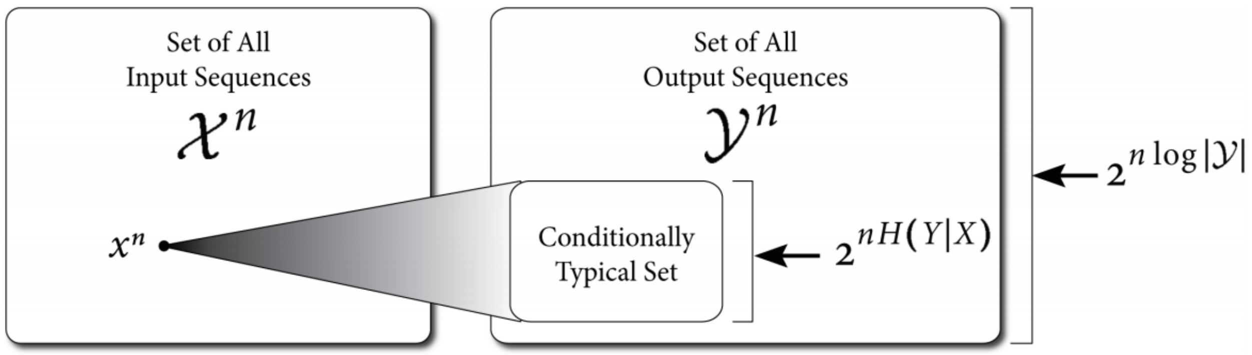

Figure 3: This figure depicts the notion of a conditionally typical set. Associated to every input sequence x^n is a conditionally typical set consisting of the likely output sequences. The size of this conditionally typical set is $\approx 2^{nH(Y|X)}$. It is exponentially smaller than the set of all output sequences whenever the conditional random variable is not uniform.

1. It has almost all of the probability—it is highly likely that a random channel output sequence is conditionally typical given a particular input sequence.

2. Its size is $\approx 2^{nH(Y|X)}$.

3. The probability of each conditionally typical sequence y^n, given knowledge of the input sequence x^n, is $\approx 2^{-nH(Y|X)}$.

If we disregard knowledge of the input sequence used to generate an output sequence, the probability distribution that generates the output sequences is

$$p_Y(y) = \sum_x p_{Y|X}(y|x)p_X(x). \tag{21}$$

We can think that this probability distribution is the one that generates all the possible output sequences. The likely output sequences are in an output typical set of size $2^{nH(Y)}$.

We are now in a position to describe the structure of a random code and the size of the message set. Alice generates 2^{nR} codewords according to the distribution $p_X(x)$ and suppose for now that Bob has knowledge of the code after Alice generates it. Suppose Alice sends one of the codewords over the channel. Bob is ignorant of the transmitted codeword, so from his point of view, the output sequences are generated according to the distribution $p_Y(y)$. Bob then employs typical sequence decoding. He first determines if the output sequence y^n is in the typical output set of size $2^{nH(Y)}$. If not, he declares an error. The probability of this type of error is small by the asymptotic equipartition theorem. If the output sequence y^n is in the output typical set, he uses his knowledge of the code to determine a conditionally typical set of size $2^{nH(Y|X)}$ to which the output sequence belongs. If he decodes an output sequence y^n to the wrong conditionally typical set, then an error occurs. This last type of error suggests how they might structure the code in order to prevent this type of error from happening. If they structure the code so that the output conditionally typical sets do not overlap too much, then Bob should be able to decode each output sequence y^n to a

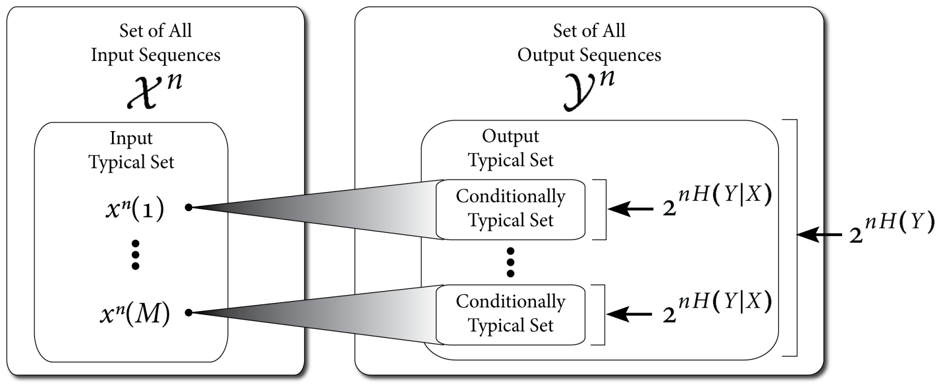

Figure 4: This figure depicts the packing argument that Shannon used. The channel induces a conditionally typical set corresponding to each codeword $x^n(i)$ where $i \in \{1, \ldots, M\}$. The size of each conditionally typical output set is $2^{nH(Y|X)}$. The size of the typical set of all output sequences is $2^{nH(Y)}$. These sizes suggest that we can divide the output typical set into M conditionally typical sets and be able to distinguish $M \approx 2^{nH(Y)}/2^{nH(Y|X)}$ messages without error.

unique input sequence x^n with high probability. This line of reasoning suggests that they should divide the set of output typical sequences into M sets of conditionally typical output sets, each of size $2^{nH(Y|X)}$. Thus, if they set the number of messages $M = 2^{nR}$ as follows:

$$2^{nR} \approx \frac{2^{nH(Y)}}{2^{nH(Y|X)}} = 2^{n(H(Y)-H(Y|X))}, \tag{22}$$

then our intuition is that Bob should be able to decode correctly with high probability. Such an argument is a "packing" argument because it shows how to pack information into the space of all output sequences. Figure 4 gives a visual depiction of the packing argument. It turns out that this intuition is correct—Alice can reliably send information to Bob if the quantity $H(Y) - H(Y|X) = I(X;Y)$ bounds the rate R:

$$R < H(Y) - H(Y|X) = I(X;Y). \tag{23}$$

A rate less than $H(Y) - H(Y|X)$ ensures that we can make the expectation of the average probability of error as small as we would like. We then employ a derandomization argument, in order to establish that there exists a code whose average probability of error vanishes as the number n of channel uses tends to infinity.