EE 376A/Stat 376A
Information Theory
Prof. T. Cover[1]

Handout #33
Friday, March 13, 2009

EE376A WINTER 08-09 FINAL EXAM REVIEW SESSION NOTES

BY WILLIAM WU

Note: Updated frequently. The most recent upload occurred on 2009-03-16 04:11.

Contents

[1]The author greatly thanks Professor Thomas Cover for correcting many typos, clarifying several proofs, and giving me words of encouragement and support.
Date: Opened 2009-03-10. Last updated 2009-03-16 04:11. Author's e-mail: `willywutang AT gmail DOT com` .

1. Introduction and General Announcements

- Exam is in Kresge Auditorium on Monday, March 16th 2009, fro 3:30 pm to 6:30 pm.
- The exam has comprehensive coverage.

- If you picked up your homework on Thursday 03/12/2009 and the grader signature was "jykim", then you need to e-mail him your grade.
- Check all your grades on `eeclass`.

  `http://www.stanford.edu/class/ee376a/files/ee376a_win0809_MTreview_willywu.pdf`

  Please briefly fill out the TA evaluation form as well if you have time; I would appreciate it.
- The focus of this review will be on Chapters 7 and 9.1 and 9.2. I won't review almost anything in Chapters 2 through 6. For a review of these previous chapters, please refer to the Midterm Exam Review session Notes (MERN):

  `http://www.stanford.edu/class/ee376a/files/ee376a_win0809_MTreview_willywu.pdf`
- Open for general questions about anything.
- Please inform me of all errors you find in this handout, in MERN, or any others. They must inevitably exist. My legal disclaimer about these errors is in Section J.

2. Clarifications

- (Mahesh Srinivasan) On page 120 of [CTWI06], it says that the optimal way to play 20 questions is to do Huffman coding:

  > Since the Huffman code is the best source code for a random variable, the optimal series of questions is that determined by the Huffman code.

  Some argument may be needed to make this clear. The Kraft-McMillan inequality only proves that Huffman codes are the best source codes amongst all uniquely decodable codes, which is a subset of nonsingular codes. And if you're playing a game 20 questions, you're only going to play once, so one might think that we don't need a uniquely decodable code.

  Resolution (Paul Cuff, Lei Zhao):

  > When you stop in 20 questions, you know what the item is, and all the information you have received is the answers to the questions you asked (and you also know the questions that you asked). We can prove that a random strategy will not improve over a deterministic strategy, so we'll assume deterministic. Now, suppose there is more than on item with the same answers to a sequence of questions in your strategy. Because the strategy is deterministic, we know that you would ask the same sequence of questions and receive the same answers for both. Therefore, you cannot possibly know which of the two items it is. Your strategy cannot terminate at this point. It cannot terminate even for one of them because it doesn't know which one it is yet.
  > The bottom line is, assume a deterministic strategy and a non-prefix code, and you can find a contradiction. Namely, for two codewords that violate the prefix condition, the shorter one should not have terminated.
- (Midterm) When we start the random walk in a stationary distribution, then every X_i has an identically distributed marginal distribution. However, they are not independent.

And if the stochastic process you are betting on is not independent, then it doesn't make sense to bet on each X_i solely according to the marginals. For example, if $X_1 = A$, it is impossible for $X_2 = D$. So given that $X_1 = A$, one should not place any money on the even that $X_2 = D$. Bottom line: bets should be set according to the conditional distribution given all the data we have seen so far:

$$b(x_k|x_{k-1}, \ldots, x_1) = p(x_k|x_{k-1}, \ldots, x_1)$$

This is log-optimal gambling.

- HW 4, Problem 6b:

  ... Thus his wealth S_n at time n is given by

  $$S_n = \prod_{i=1}^{n} \left(\frac{X_i}{c} \right).$$

  Question: Why is it $S_n = \prod_{i=1}^{n} \left(\frac{X_i}{c} \right)$?

  Answer:

  – Firstly, there is only one gambler, not a group of gamblers splitting the returns.

  – Say it costs c dollars to play the game.

  – And say he gets a certain multiplicative payoff X from playing once.

  – Thus, if he starts with one dollar, the payoff is $\frac{X}{c}$.

  – We want ratios of payoffs per (one) dollar because we want to multiply payoff ratios.

  – And we are multiplying payoff ratios because we are reinvesting our wealth.

  Big Picture: The notion of reinvesting (like compound interest) is why the wealth S_n is expressed as a product, not a sum. The product is the reason why the logarithm shows up. After applying the Weak Law of Large Numbers (WLLN), the analysis shows that we should maximize *not the expectation of X, but the expected logarithm of X.*

- HW 5, Problem 5d

  Let $X_1, X_2, \ldots,$ i.i.d. with $X = \begin{cases} 1, & \text{with probability } 1/2 \\ 2, & \text{with probability } 1/4 \\ 3, & \text{with probability } 1/4. \end{cases}$

  Consider the code assignment $C(x) = \begin{cases} 0, & \text{if } x = 1 \\ 01, & \text{if } x = 2 \\ 11, & \text{if } x = 3. \end{cases}$

  What is the entropy rate of the process $Z_1 Z_2 Z_3 \ldots = C(X_1)C(X_2)C(X_3)\ldots$?

  Solution: The expected codeword length is

  $$L(C(x)) = 0.5 \times 1 + 0.25 \times 2 + 0.25 \times 2 = \frac{3}{2}.$$

  Further, the entropy rate of the i.i.d. X^n is

  $$H(\mathcal{X}) = H(X) = H(.5, .25, .25) = \frac{3}{2}.$$

So the code is a uniquely decodable code with $L = H(\mathcal{X})$, and therefore the sequence is maximally compressed with $H(\mathcal{Z}) = 1$ bit. If $H(\mathcal{Z})$ were less than its maximum of 1 bit then the Z^n sequence could be further compressed to its entropy rate, and X^m could also be compressed further by blockcoding. However, this would result in $L_m < H(\mathcal{X})$ which contradicts theorem 5.4.2 of the text. So $H(\mathcal{Z}) = 1$ bit.

Note that the Z^n sequence is not i.i.d. $\sim \text{Br}(\frac{1}{2})$, even though $H(\mathcal{Z}) = 1$ bit. For example, $P\{Z_1 = 1\} = \frac{1}{4}$, and a sequence starting $10\ldots$ is not allowed. However, once $Z_i = 0$ for some i then Z_k is Bernoulli($\frac{1}{2}$) for $k > i$, so Z^n is asymptotically Bernoulli($\frac{1}{2}$) and gives the entropy rate of 1 bit.

3. Algebra of Information Theory

First, we established definitions for $H(X), H(X|Y), I(X;Y), I(X;Y|Z)$, and $D(p||q)$, along with chain rules for H, I, D. Then we showed the relations in Figure 1.

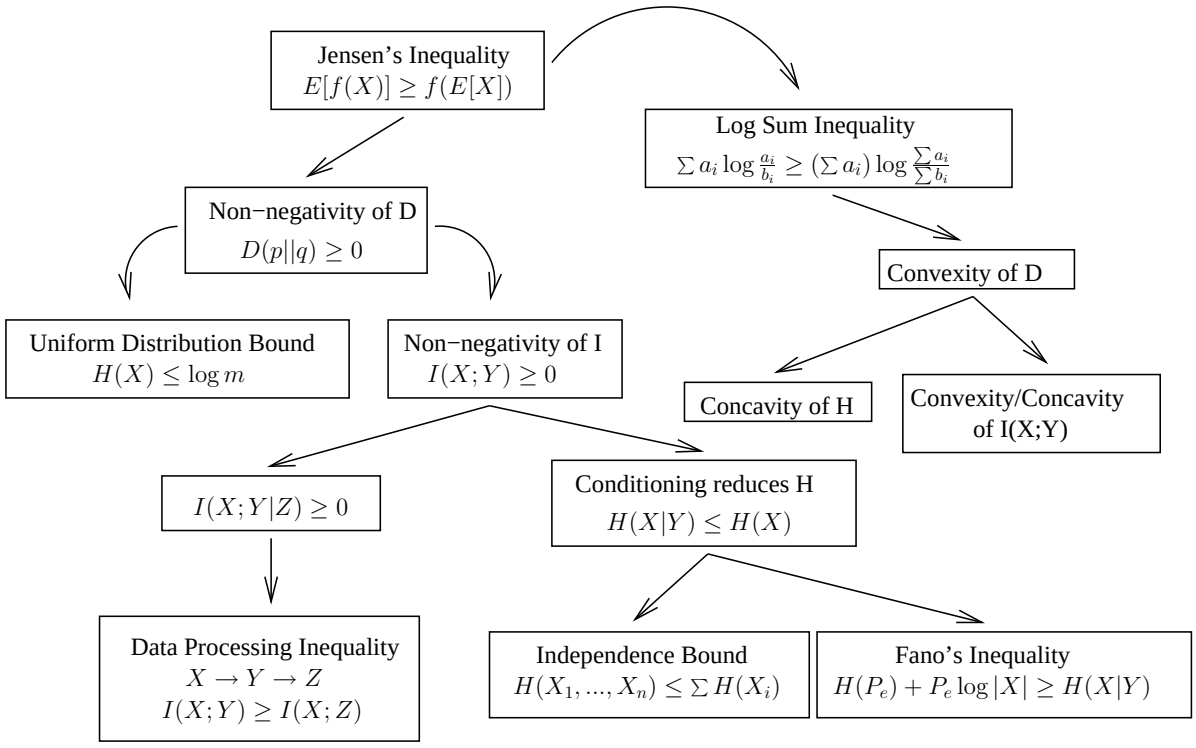


Figure 1. Chain of ideas in Chapter 2.

4. Asymptotic Equipartition Property (AEP)

- **Definition:** We define the typical set as

$$A_\epsilon^{(n)}(X) = \left\{ x^n : \left| -\frac{1}{n} \log p(x^n) - H(X) \right| < \epsilon \right\}.$$

The definition of this set is motivated by the fact that in the i.i.d. case, the Weak Law of Large Numbers (WLLN) says that $-\frac{1}{n} \log p(X^n) \to H(X)$ in probability.

- **The AEP.**

  Theorem 4.1 (AEP). *(a) If* $(x_1, \ldots, x_n) \in A_\epsilon^{(n)}$*, then* $H(X) - \epsilon \leq -\frac{1}{n} \log p(x_1, x_2, \ldots, x_n) \leq H(X) + \epsilon$.

  (b) $\mathbf{P}\left[A_\epsilon^{(n)}\right] > 1 - \epsilon$ *for* n *sufficiently large.*

  (c) $|A_\epsilon^{(n)}| \leq 2^{n(H(X)+\epsilon)}$

  (d) $|A_\epsilon^{(n)}| \geq (1-\epsilon)2^{n(H(X)-\epsilon)}$.

- If we are dealing with continuous random variables, everything is the same, but you replace p with a density f, H with h, and set cardinalities with volumes.
- More generally: The AEP also holds for stationary ergodic processes (Shannon-McMillan-Breiman Theorem, EE376B)
- Examples of processes that satisfy the AEP:
  (a) sequence of i.i.d. random variables
  (b) sequence of states in a stationary irreducible Markov chain
  (c) any stationary ergodic source (see optional Appendix F for more clarification on this topic)
- Why We Care: We can use the AEP (as a theoretical tool) for data compression. By repeatedly generating the random variables, we'll see that amongst all the possible realizations of $(X_1, \ldots, X_n)$ when n is large, there will be a subset of those sequences of size $2^{nH(X)}$ upon which almost all the probability mass lies. This subset is the typical set $A_\epsilon^{(n)}(X)$. So in encoding a random message source, we can throw out everything outside of this typical set and still preserve all the information in the source with high probability.

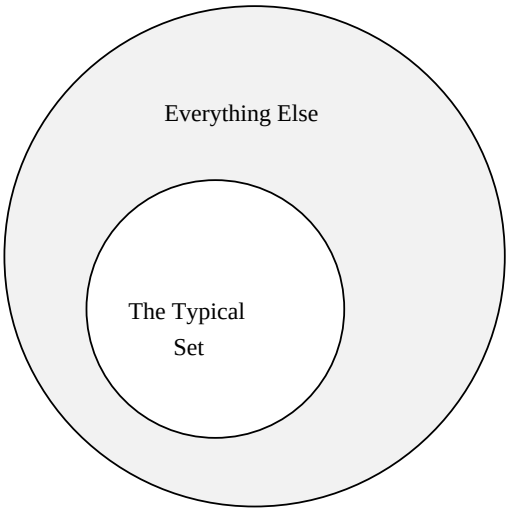


FIGURE 2. The typical set is a subset of all possible sequences, and yet it holds almost all the probability. So forget the rest.

5. Kolmogorov Complexity

- The Kolmogorov complexity of an binary string x is the length of the smallest program required to have a computer (universal Turing machine[1]) create x. We denote this length by $K(x)$.
- Note that images can be thought of as binary strings – take the rows of the matrix of pixels, and concatenate them into one big string.
- The Kolmogorov Complexity of any object is not computable. Roughly speaking, if you want to know what the best program is, you have to try all possible programs, but there's no way of knowing which programs will stop and which won't. Maybe there is a very short program that accomplishes the task, but it has some `for` loops that cause it to run for a very long time. It has been proven that there is generally no way to determine whether a program is stuck in an infinite loop and will never halt, or whether it actually will halt eventually and is just taking a long time. In short, being able to compute the Kolmogorov Complexity requires being able to solve the halting problem, which is undecidable over Turing machines.
- So all we can do is give an upper bound by thinking up an explicit program that will generate the object. The goal, for the purposes of this course, is to simply give the lowest upper bound you can think of.
- We usually only ask for the conditional Kolmogorov complexity $K(x|n)$, where $n = l(x)$ is the length of the string x that I want to construct with a program. This is the length of smallest program needed to create x assuming that the length of x is somehow supplied in advance. See Appendix B for more details about why this convention is adopted.
- **Bottom Line:** Try to think of the most efficient parameterization possible for the object, and then give the number of bits needed to describe these parameters.

  For example, in specifying a rectangle on a computer screen, we only need to specify four numbers corresponding to the two corners, and not eight numbers corresponding to all four corners.
- Sometimes a brute-force enumeration scheme does well (e.g., if we have an enumeration of all possible squares which can be drawn in a bounding box, then any particular square is parameterized by the index of that square in my list)
- Do some random examples.

6. Inequalities

A huge fraction of the arguments we have used in this class have had to do with inequalities. Applying the right inequalities at the right time, and knowing conditions under which the bounds are met. We can use inequalities to compute information capacities, prove capacity converses, and prove many general information-theoretic statements.

Inequalities can be tricky and difficult to think about, as there is always a vague gap floating around, and we have to assure that our inequalities are pointing in the right directions. Also,

[1]For more about Turing machines and the theory of computational complexity, I recommend [Sip05].

they often have to be applied gradually in the right order. It's possible to start with an inequality that throws out too much to prove a desired result.

6.1. **Inequalities Seen in EE376A.**

- Jensen's Inequality: For convex f, $\mathbf{E}[f(X)] \geq f(\mathbf{E}[X])$, with equality if and only if either X is a constant (degenerate random variable) or f is linear. See Appendix G for more exercises on using this.

  Another way of stating Jensen's inequality: for convex $f : [a, b] \to \mathbf{R}$, if you have nonnegative real numbers p_j such that $\sum p_j = 1$, then for all $x_j \in [a, b]$, we have

$$f\left(\sum_{j=1}^{n} p_j x_j\right) \leq \sum_{j=1}^{n} p_j f(x_j).$$

- Conditional entropy: $H(X|Y) \leq H(X)$ with equality if and only if X and Y are independent.
- Entropy under mappings: $H(f(X)) \leq H(X)$ with equality if and only if f is a 1-to-1 function.
- Relative Entropy Inequality: $D(\mathbf{p}||\mathbf{q}) \geq 0$ with equality if and only if $\mathbf{p} = \mathbf{q}$.
- Independence Bound: $H(X_1, \ldots, X_n) \leq \sum H(X_i)$ with equality if and only if the X_i's are independent.
- Union Bound: For events $A_1, \ldots, A_n$ $\mathbf{P}[A_1 \cup A_2 \cup \ldots \cup A_n] \leq \sum_{i=1}^{n} \mathbf{P}[A_i]$, with equality if and only if the events are disjoint. (We used the union bound in the achievability proof for the noisy coding theorem.)
- Kraft's Inequality: The codeword lengths for any uniquely decodable D-ary code must satisfy the Kraft inequality

$$\sum D^{-l_i} \leq 1$$

  Conversely, given a set of codeword lengths that satisfy this inequality, it is possible to construct a uniquely decodable code with these codeword lengths.

- Log-Sum Inequality: For nonnegative real sequences (a_i) and (b_i), $\sum a_i \log \frac{a_i}{b_i} \geq (\sum a_i) \log \frac{\sum a_i}{\sum b_i}$, with equality if and only if $\frac{a_i}{b_i}$ equals the same constant for all i.

For some more common inequalities that were not explicitly covered in EE376A but nevertheless are very useful, check out Appendix A.

6.2. **General Inequalities.** Here are some high school level but nevertheless very important inequalities that get constantly used behind the scenes without further mention:

- If $a \leq b$, then $-a \geq -b$.
- If $a \geq b$ and $b \geq c$, then $a \geq c$ with equality if and only if $a = b = c$.
- If $a_1 \geq b_1$ and $a_2 \geq b_2$, then $a_1 + a_2 \geq b_1 + b_2$.
- If $a_1 \geq b_1 > 0$ and $a_2 \geq b_2 > 0$, then $a_1 a_2 \geq b_1 b_2$ with equality if and only if $a_1 = b_1$ and $a_2 = b_2$.

- If $a \geq b > 0$ then $\frac{1}{b} \geq \frac{1}{a}$.
- If $a \geq b > 0$ and $\alpha > 0$ then $a^\alpha \geq b^\alpha$.
- $a^2 \geq 0$.
- Suppose $a, b, c \in R$. If $a + b = c$ and b is positive, then $a < c$.
- If $f(x) \leq M$ and we see by direct computation that $f(x_0) = M$, then x_0 achieves the maximum possible value of f.
- $\inf_x f(x) \leq \sup_x f(x)$.
- $\sup_{x \in U} f(x) \leq \sup_{x \in V} f(x)$ where $U \subseteq V$
- If $x_i \geq M$ for all i, then $\sum_{i=1}^n x_i \geq nM$. Similarly, if $x_i \leq M$ for all i, then $\sum_{i=1}^n x_i \leq nM$.
- A useful technique: If we are comparing two different infimums/supremums, and we know that one of them is achieved, give a name to the x^* that achieves it, and plug that x^* into the other infimum/supremum to establish a comparison.

7. High-Level Overview of EE376A

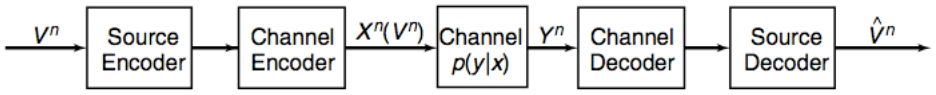
7.1. Executive Summary.

- Information theory is about the fundamental *limits* of
  (a) data compression (source coding), and
  (b) reliable data transmission (channel coding).

  These problems are like duals. One is about removing all redundancy. The other is about adding redundancy in a controlled fashion to combat noise.
- The two major theorems, respectively, are:

  Theorem 7.1 (Source Coding Theorem)**.** *The entropy $H(\mathcal{X})$ of the source X is the minimum rate at which data can be compressed losslessly.*

  Theorem 7.2 (Channel Coding Theorem)**.** *The capacity $C = \max_X I(X;Y)$ of the channel $p(y|x)$ is the maximum rate at which data can be reliably transmitted.*
- By the Source-Channel separation theorem, these problems can be considered separately, resulting in the following block diagram:



- The first half of EE376A was concerned with source coding, or data compression. The goal is to remove redundancy. Given a message source, we want to get to the bottom of it and determine how much information is really contained inside it. Huffman coding, the AEP, and Kolmogorov complexity are all concerned with this; in fact, even log-optimal gambling can be thought of as data compression (see Section 6.5: Gambling and Data

Compression). The main result is that the entropy of the source is the compression limit.

- The second half of EE376A was concerned with channel coding. The goal is to add back redundancy in a controlled fashion, so as to transmit across a channel at optimal rates. The main result is there exists a number C, called the capacity, such that for all rates R less than C, there exists a code that can achieve arbitrarily low probabilities of error for sufficiently large blocklengths. C is written in terms of mutual information, and can be thought of as the capacity of a physical pipe.
- Information theory establishes the general approach and unifying framework for communications theory. Communications theory is the implementation of information theory.

7.2. **History.** Preliminary remarks: For further reading reflecting on the history of information theory, I recommend "Claude E. Shannon: A Retrospective on His Life, Work, and Impact" by Robert Gallager [Gal01], from which I have drawn heavily in the following several paragraphs. Other papers to look at are [Gal05], [Gal03], [Bat] , [GBC$^+$02], [RW02]. You may also be interested in my writeup, "About Claude Shannon", on the ee376a course website:

`http://www.stanford.edu/class/ee376a/files/ee376a_win0809_aboutClaudeShannon_willywu.pdf`.

Prior to Shannon's groundbreaking 1948 paper "A Mathematical Theory of Communication", there was no unified theory of communication. There was a hodgepodge of different modes of communication, such as telegraphs, AM radio, or FM radio, and engineers who specialized in each of the different modes. It was not entirely clear what all these modes of communication had in common. In Gallager's words, "before 1948, there was only the fuzziest idea of what a message was." All they had were waveforms – different kinds of analog electromagnetic waveforms being sent through the air, whose structures depended on what kind of communication device was being used. It was unclear how to generally transform any particular message, such a letter, into a waveform that captured all the information. It was also unclear how to compare the different modes of communication.

Shannon defined a message as a choice between alternatives. He also had the keen insight that it is irrelevant what the actual values of the alternatives are. The only thing that matters is the choice between distinct alternatives, and the probabilities associated with these alternatives. Upon successful decoding on the receiver side, these alternatives can be given any interpretation we like, whether they be distinct images, letters, sounds, or numbers. Notice how this insight is included in the definition of entropy, which Shannon uses to characterize the amount of information in a source. The entropy function takes probabilities between distinct states into account, and does not consider the values associated with each state.

Shannon then proceeded to model message sources as random processes. It is not intuitive to think of message sources as random processes – after all, I don't like to think of anything I say, write, or do as being randomly generated! However, from the point of view of an engineer designing a communications system, perhaps he is forced to model the source as random, because he does not know a priori what messages users will be transmitted over the system.

So, although our users are (hopefully!) not random message sources – much less stationary ergodic stochastic processes – we model them as such. But is there any other justification for this modeling decision beside a lack of alternative options?

One justification can be provided by changing our viewpoint about randomness as being not undesirable, but rather, informative. Consider a teacher who, each day in class, simply reads directly off a series of Powerpoint slides with almost no embellishment. Students would probably quickly stop attending class because they could just stay in bed and read the slides later. Attending lecture does not tell them anything new. On the other hand, if a teacher deviates from his notes, adding more to them, and offers fun digressions every once in a while, then the lecture is far more interesting because students don't know exactly what's going to happen. The uncertainty is actually attractive. *Uncertainty makes the lecture more informative.*

Another justification is provided by experiments. In his 1948 paper, Shannon also did an analysis of English. He created Markov models that randomly generate alphabetic characters, either based on only the previous letter, previous two letters, previous three letters, previous word, etc. Some results from Shannon's experiments can be seen on pages 169 through 170 of the text [CTWI06]. In the seventh example, we see that by using merely a second-order Markov word model, we get a sentence that closely resembles English:

> THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED

This suggests that modeling sources as random may not be a bad idea. Whether it also suggests that we humans are actually probabilistic finite automata with unexpectedly small state spaces will not be digressed upon here.

Having established what a message source is, Shannon then characterized channels as conditional probability distributions that produce different outputs based on the input. He formalized the notion of encoding, in which the messages from the source are first converted into a form appropriate for transmission across the channel, and then a decoder on the receiver side inverts the encoding.

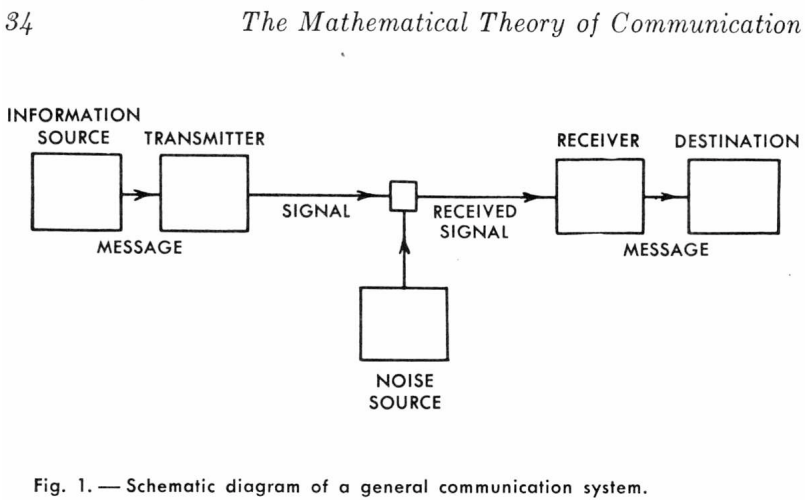


FIGURE 3. Shannon's Figure 1.

Thus, Shannon envisioned the general framework for how communications should be conducted. In Figure 3, we see the original block diagram presented as the first figure in Shannon's

1948 paper. This block diagram can now be found in the first chapter of any communications textbook.

8. Channel Capacity

8.1. **What The Channel Capacity Theorem Is Saying, and Why It Is Surprising.** Suppose you are trying to communicate with me using a BSC with a bit flip probability p that is less than $\frac{1}{2}$. You want to ensure that I decode every bit correctly, with arbitrarily low probability of error. The most natural idea you would first think of is to use a repetition code. If you want to send me a 1, then you send me a string of n consecutive 1s. I will receive a mixture of 0s and 1s, and I'll decode using majority vote: if I receive more 1s than 0s, then I conclude you probably sent me a 1.

Using the Chernoff bound (Appendix C), we can show that the probability of me decoding your bit incorrectly goes to zero as n goes to infinity. But this also causes the rate of our communication to go to zero, because in order to force the error probability for just this one bit to go to zero, you need to add an infinite amount of redundancy.

Intuitively, it seems that if we want to force the probability of error to go to zero, we have to add an infinite amount of redundancy for each bit, and thus the rate ought to go to zero.

The amazing thing is that Shannon proved that for a channel with nonzero information capacity such as the BSC, there must exist a coding scheme that has both arbitrarily low probability of error AND nonzero rate![2]

Furthermore, for any channel, there is a threshold number C, called the capacity, such that all rates $R < C$ are *achievable*, meaning that we can communicate at that rate with arbitrarily low probability of error. And all rates $R > C$ are not achievable, meaning the probability of error can no longer be forced as low as possible – it will be strictly bounded away from zero.

8.2. **Summary of Impressive Facts and Clever Ideas.**

8.2.1. *Impressive Facts.*

- The fact that there do indeed exist codes for noisy channels such as the BEC, with nonzero rate and arbitrarily low probability of error.[3]
- The fact that source coding and channel coding can be separated.[4]

8.2.2. *Clever Ideas.*

- The fact that Shannon even imagined that there should exist a code with nonzero rate and arbitrarily low probability of error. (Formulating the problem.)
- The way Shannon circumvented the complexities of analyzing any particular channel, instead producing a sweeping statement that holds for all channels in general.

[2]All coding schemes thrown at the BSC up till 1972 had an asymptotic rate of zero. [CTWI06]

[3]In the case of BEC, Tornado codes, Turbo codes, and LDPC codes achieve the capacity.

[4]This is not that intuitive. Sometimes, one might suspect that sources should not be coded since they are already somehow naturally suited for the channel. If the information in this handout was sent through an erasure channel that kills a fourth of the letters, maybe you could still decode it perfectly from context. But if we compress the handout first and then drop a fourth of the symbols, that human decoding scheme doesn't work anymore.

- The way Shannon circumvented the problem of figuring out what is the optimal code, which should be a channel-dependent task as well. Instead, Shannon used the probabilistic method, conducting a random experiment over all possible codebooks and transmitted codewords. It so happens that the average probability of error over all codebooks and transmitted codewords is small. This demonstrates the existence of a codebook with low average probability of error.
- The way Shannon circumvented the complexities of analyzing the performance of an optimal Maximum A Posteriori (MAP) decoder, instead choosing to use the idea of jointly typical decoding, making the analysis far easier.
- The analysis technique of allowing for an arbitrarily small (ϵ) but nonzero probability of error.
- Using the channel many times in succession, to exploit the structure that must arise thanks to the Law of Large Numbers, thereby allowing for the appearance of typical sets, which then allows for typical set decoding.
- In the achievability, using the fact that $P_e^{(n)} \to 0$ implies that $\lambda^{(n)} \to 0$. This is achieved as follows: given a codebook with very low average probability of error, throw away the worst half of these codewords to come up with a truncated codebook whose maximal probability of error is small as well. Thus, Shannon shifts his position from arbitrarily good average performance to arbitrarily good worst-case performance.
- In the converse, using the fact that $\lambda^{(n)} \to 0$ implies that $P_e^{(n)} \to 0$, and $P_e^{(n)}$ can be thought of $\mathbf{P}\left[\hat{W} \neq W\right]$ if we assume W to be uniformly distributed.

8.3. **Preliminary Definitions.**

8.3.1. *Characterizing Channels.*

- Memoryless = probability distribution of output at time n only depends on the input at time n, and is conditionally independent of past inputs or outputs.
- A channel is a 3-tuple $(\mathcal{X}, p(y|x), \mathcal{Y})$. (input set, conditional probability distributions, and output set)
- Discrete Memoryless Channel (DMC) $p(\mathbf{y}|\mathbf{x}) = \prod_i p(y_i|x_i)$.
- nth extension of the DMC is the channel $(\mathcal{X}^n, p(y^n|x^n), \mathcal{Y}^n)$ that you get by grouping in blocks, where $p(y_k|x^k, y^{k-1}) = p(y_k|x_k)$. If there is no feedback, this reduces to

$$p(y^n|x^n) = \prod_{i=1}^{n} p(y_i|x_i)$$

8.3.2. *What Is a Code?* The procedure:

(a) A message W is drawn from $\{1, 2, \ldots, M\}$, according to some distribution.

(b) W is encoded into the codeword $X^n(W)$, a sequence of n symbols.

(c) $X^n(W)$ is sent across a probabilistic channel, where it can get corrupted.

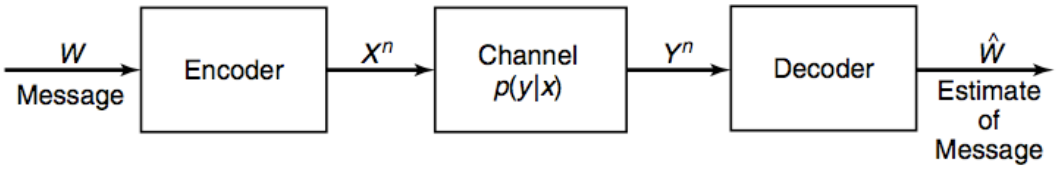


FIGURE 4. The channel coding block diagram.

(d) The receiver makes a guess $\hat{W}$ as to what the original message was, using a decoding rule $\hat{W} = g(Y^n)$.

(e) An error occurs if $\hat{W} \neq W$.

Thus a "$(2^{nR}, n)$" code consists of three things:

1. Set of possible messages $\mathcal{W} = \{1, 2, \ldots, M\}$, where $M := 2^{nR}$.
2. The encoding function $X^n(\cdot)$.
3. The decoding function $g(\cdot)$

8.3.3. *Measuring Performance.*

- Rate: $R = \frac{\log_2 M}{n}$
- Conditional probability of error *for a fixed codebook and a fixed codeword*:

$$\lambda_i = \mathbf{P}\left[\hat{W} \neq w \mid W = w\right] = \mathbf{P}\left[g(Y^n) \neq i | X^n = x^n(i)\right] = \sum_{y^n} \mathbf{P}\left[y^n | x^n(i)\right] I(g(y^n) \neq i)$$

  This is the conditional probability of error given that index i was sent, where $I(\cdot)$ is the indicator function. Note that we can express this either with respect to the codeword index i or the corresponding source message symbol w.

- Maximal probability of error *for a fixed codebook*:

$$\lambda^{(n)} = \max_{w \in \{1,2,\ldots,M\}} \lambda_w$$

  This is what we actually want to minimize in good codebooks.

- Average probability of error *for a fixed codebook*:

$$P_e^{(n)} = \frac{1}{M} \sum_{w=1}^{M} \underbrace{\mathbf{P}\left[\hat{W} \neq w \mid W = w\right]}_{\lambda_w}$$

  This is not what we want to minimize, but it serves simply as a vehicle to symmetrize the mathematics and help us get rolling.

  In the achievability, after the analysis for $P_e^{(n)}$ is done, we will be actually able to conclude things about $\lambda^{(n)}$ by conducting the "final improvements".

  In the converse, we assume that $\lambda^{(n)}$ goes to zero. But that implies that $P_e^{(n)}$ goes to zero. And that will make the analysis easier as well because $P_e^{(n)}$ corresponds to

$\mathbf{P}\left[W \neq \hat{W}\right]$ and we'll be able to apply Fano's inequality assuming that W is uniformly distributed.

More details on the interplay between $\lambda^{(n)}$ and $P_e^{(n)}$ will be described in the sequel.

- **Achievable Rate**:

  A rate is said to be achievable if there exists a sequence of $(2^{nR}, n)$ codes such that $\lambda^{(n)}$ goes to 0 as $n \to \infty$.

- **The Goal:**

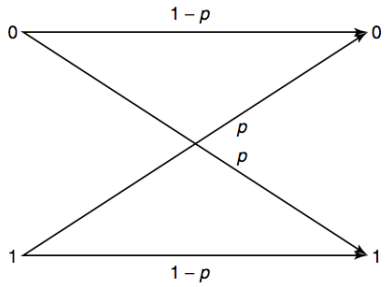
  Find the capacity C, the supremum of all achievable rates.

8.4. **Summary of Ideas.**

- **The Answer:**

$$\boxed{C = \max_{p(x)} I(X;Y)}$$

- **Standard channels**[5]:
  - Binary Symmetric Channel (BSC):



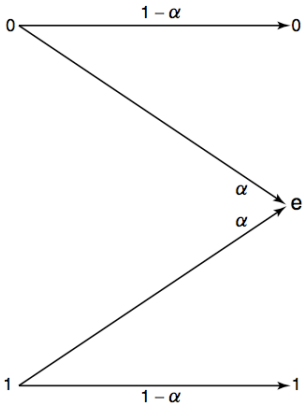
    $C = 1 - H(p)$, achieved with uniform distribution on input.
  - Binary Erasure Channel (BEC):

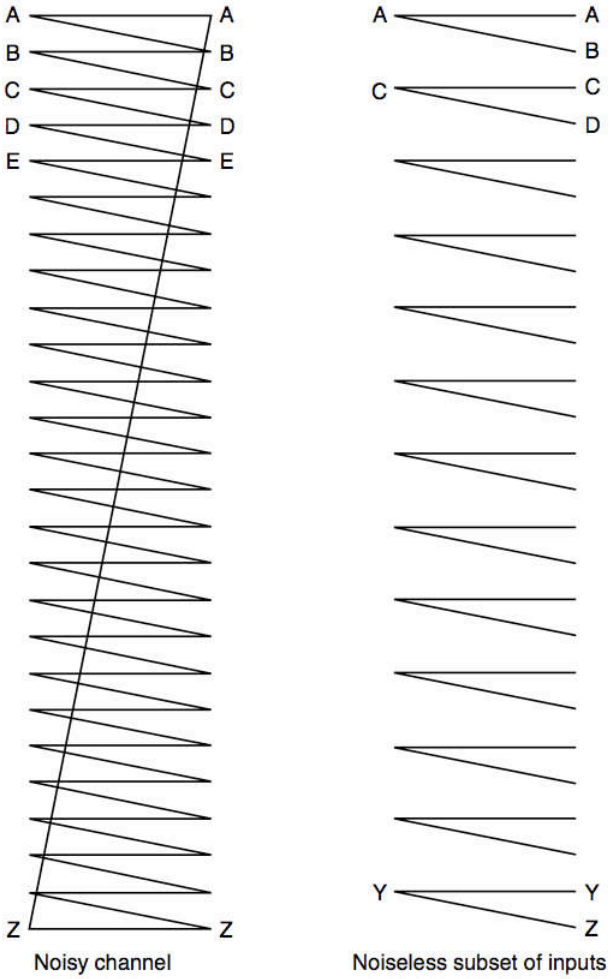
    $C = 1 - \alpha$, achieved with uniform distribution on input.
  - Symmetric Channel: $C = \log|\mathcal{Y}| - H(\text{row of transition matrix})$, achieved with uniform distribution on input.
  - Noisy Typewriter (HW7, Problem 2):
  - Z-Channel (HW7, Problem 3):

$$Q = \begin{bmatrix} 1 & 0 \\ \gamma & 1-\gamma \end{bmatrix}$$

  - Gaussian Channel: $Y_i = X_i + Z_i$, where $Z_i \sim \mathcal{N}(0, N)$, and power constraint $\sum_{i=1}^{n} x_i^2 \leq nP$

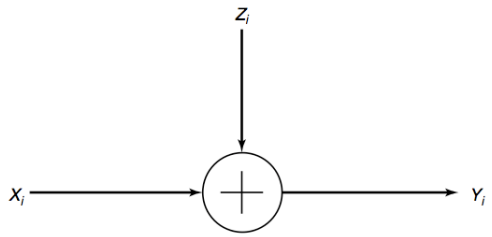
[5]Channels in future classes (EE376B, EE478): Multiple Access Channel (MAC), Broadcast Channel, Interference Channel, Relay Channel, Two-way Channel, Channels with state





$C = \frac{1}{2}\log\left(1 + \frac{P}{N}\right)$.

- Properties:
  - $0 \leq C \leq \min\{\log|\mathcal{X}|, \log|\mathcal{Y}|\}$
  - $I(X;Y)$ is a continuous concave function of $p(x)$.



- Joint Typicality and Joint AEP (will say more about these ideas later in Section 8.5)
- **Channel coding theorem:**

  > All rates below capacity C are achievable, and all rates above capacity are not; that is, for all rates $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with probability of error $\lambda^{(n)} \to 0$. Conversely, for rates $R > C$, $\lambda^{(n)}$ is bounded away from 0.

- **Feedback capacity:** Feedback does not increase capacity for DMCs.
- **Source-channel Theorem:** A stochastic process with entropy rate H cannot be sent reliably over a discrete memoryless channel if $H > C$. Conversely, if the process satisfies the AEP, then the source can be transmitted reliably if $H < C$.

8.5. **Joint Typicality and Joint AEP.**

- **Joint Typicality:** The set $A_\epsilon^{(n)}(X,Y)$ of **jointly typical sequences** $\{(x^n, y^n)\}$ with respect to the distribution $p(x,y)$ is given by

$$\begin{aligned} A_\epsilon^{(n)}(X,Y) = \{(x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n : \\ \left| -\frac{1}{n} \log p(x^n) - H(X) \right| < \epsilon, \\ \left| -\frac{1}{n} \log p(y^n) - H(Y) \right| < \epsilon, \\ \left| -\frac{1}{n} \log p(x^n, y^n) - H(X,Y) \right| < \epsilon \} \end{aligned}$$

  where $p(x^n, y^n) = \prod_{i=1}^n p(x_i) p(y_i|x_i)$.

- **Joint AEP:** Let (X^n, Y^n) be sequences of length n drawn iid according to $p(x^n, y^n) = \prod_{i=1}^n p(x_i, y_i)$. Then:

  (a) $\mathbf{P}\left[(X^n, Y^n) \in A_\epsilon^{(n)}\right] \to 1$ as $n \to \infty$.

  (b) $|A_\epsilon^{(n)}| \leq 2^{nH(X,Y)+\epsilon}$

  (c) If $(\tilde{X}^n, \tilde{Y}^n) \sim p(x^n)p(y^n)$, then $\mathbf{P}\left[(\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}\right] \leq 2^{-nI(X;Y)-3\epsilon}$.

This picture explains it all:

Remarks:

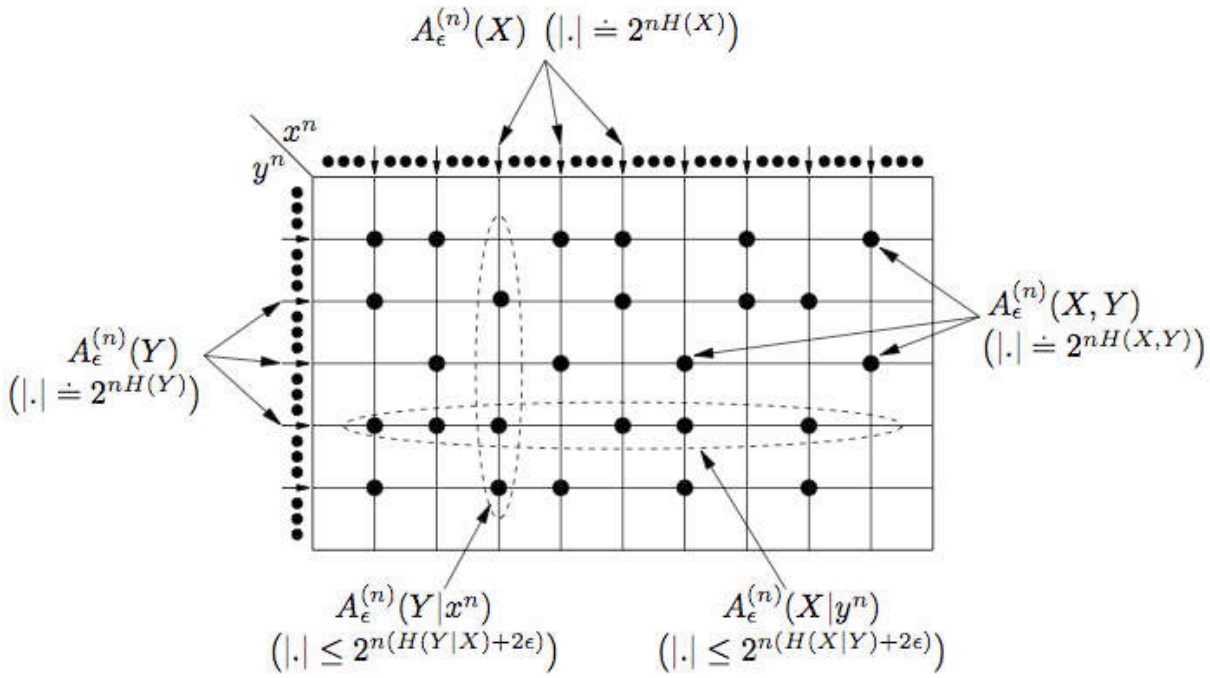


FIGURE 5. Joint Typicality (Source: EE478 Notes by Abbas El Gamal)

(a) Joint typicality is a property of *pairs of sequences.*

(b) The three conditions in the definition of joint typicality correspond to applying the AEP three times – once for X^n, once for Y^n, and once for (X^n, Y^n).

(c) We will see pairs from the jointly typical set with high probability. Each of the three individual conditions in the the definition of the jointly typical set $A_\epsilon^{(n)}(X,Y)$ are conditions that will hold with high probability thanks to the Weak Law of Large Numbers (WLLN). Thus, most of the probability mass of the joint distribution $P_{X^nY^n}$ is concentrated on the jointly typical sequences. The number of jointly typical sequences is roughly $2^{nH(X,Y)}$, each with roughly the same $P_{X^nY^n}$-probability of $2^{-nH(X,Y)}$. (These last two statements are just consequences of applying the standard AEP to a sequence of random 2-vectors (X,Y).)

(d) Suppose we have a pair of sequences (x^n, y^n) that belongs to $A_\epsilon^{(n)}(X,Y)$. Then by the first two conditions in the definition of joint typicality, the x^n sequence alone is typical for the marginal distribution P_{X^n}, and the y^n sequence alone is typical for the marginal distribution P_{Y^n}. That is, $x^n \in A_\epsilon^{(n)}(X)$ and $y^n \in A_\epsilon^{(n)}(Y)$.

(e) **Important Point #1:** However, the converse is not necessarily true! Suppose we pick a random $x^n \in A_\epsilon^{(n)}(X)$ and $y^n \in A_\epsilon^{(n)}(Y)$. We then pair these two sequences up to make (x^n, y^n). This pair is not necessarily in the jointly typical set $A_\epsilon^{(n)}(X,Y)$.

Why? Well, presumably the channel is not so bad that X^n and Y^n are completely independent. That is, there is a channel that specifies a conditional distribution $p(y^n|x^n)$

such that given x^n, only certain y^n's are possible. Thus, if we draw x^n and y^n independently from $A_\epsilon^{(n)}(X)$ and $A_\epsilon^{(n)}(Y)$, respectively, then in the extreme case, you may end up choosing a y^n that is impossible for the channel to produce given your choice of x^n.

Make up a concrete example of this.

(f) **Important Point #2:** Let's compute the probability of *accidentally creating a jointly typical* (x^n, y^n) *pair when* x^n *and* y^n *are independently drawn from* $A_\epsilon^{(n)}(X)$ *and* $A_\epsilon^{(n)}(Y)$. The total number of ways to create a pair out of x^n sequences drawn from $A_\epsilon^{(n)}(X)$, and y^n sequences drawn from $A_\epsilon^{(n)}(Y)$, is roughly $2^{nH(X)} \cdot 2^{nH(Y)} = 2^{n(H(X)+H(Y))}$. On the other hand, we know that the number of jointly typical sequences, due to the application of AEP to (X^n, Y^n), is roughly $2^{nH(X,Y)}$. Thus, the probability we end up with a jointly typical pair by sheer accident is roughly

$$\frac{2^{nH(X,Y)}}{2^{n(H(X)+H(Y))}} = 2^{-nI(X;Y)}.$$

This is handwaving due to the repeated occurrences of the adjective "roughly" but it can be made precise; see Appendix E for a rigorous presentation of this.

This result is perhaps the most important part of the analysis of error for jointly typical decoding in Shannon's capacity theorem proof.

8.6. **Three Different Intuitions.** These do not constitute proofs; just intuitions.

- Scatterplot Intuition

  In Figure 5, the probability of a randomly chosen pair of sequences being typical is $2^{-nI(X;Y)}$. So we can expect to have to look at $2^{nI(X;Y)}$ such pairs before finding a jointly typical pair. This suggests that the number of distinguishable X^n messages is X^n, particularly if we are to be use jointly typical decoding.

- Noisy Typewriter From X to Y

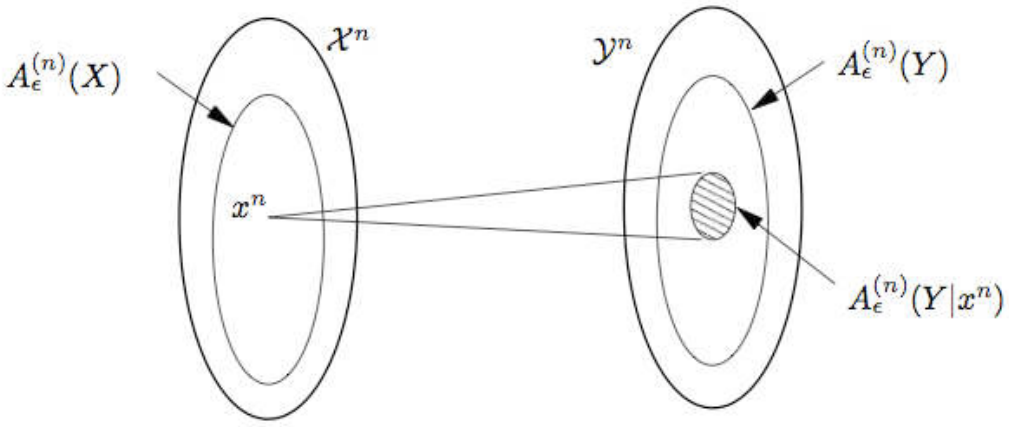


FIGURE 6. Everything is like a noisy typewriter. (Source: EE478 Notes by Abbas El Gamal)

See Figure 6. One can show that if you fix x^n, then the number of y^n sequences which are jointly typical with x^n is roughly $2^{nH(Y|X)}$.[6] These sequences are represented by the expanded part of the fan, which should be thought of as confused letters as in the noisy typewriter channel. Since we don't want any confusions, we hope to pick x^n sequences that are "widely spaced apart" such that their corresponding fans do not overlap on the right-hand side. Assuming we can do that, then the most number of fans we could pack into the right-hand side oval is $\frac{2^{nH(Y)}}{2^{nH(Y|X)}} = 2^{nI(X;Y)}$. Hence, sometimes people say that every channel looks like a noisy typewriter channel eventually, as long as the blocklength n is long enough.

- Fans Running From Y to X

  Same argument, but reverse the direction of the fans. Fixing a (typical) output y^n, there are only certain x^n sequences that are jointly typical with y^n (read: have a good chance of having given rise to y^n after channel corruption), and the number of such sequences is roughly $2^{nH(X|Y)}$. Since we don't want confusion as to what was originally sent, then for any given y^n, we can only pick one x^n sequence in the corresponding fan. Since the total number of typical X sequences is about $2^{nH(X)}$, the number of codewords we can play with is thus

$$M \approx \frac{2^{nH(X)}}{2^{nH(X|Y)}} = 2^{nI(X;Y)}$$

  and the rate is thus $R = \frac{\log_2 M}{n} = I(X;Y)$.

8.7. **Illustrations of The Probabilistic Method.**

8.7.1. *Using Expectation To Demonstrate Existence.* Here's a puzzle for you to consider:

> *Using probability theory*, show that if 101 items are randomly thrown among 10 boxes, prove that at least one of the boxes must contain more than 10 items.

The emphasis is to use probability theory to show this, and not the pigeonhole principle.

Here is how to do it. Let X be a random variable corresponding to the number of balls in a box. This is a binomial random variable with parameters 101 and 1/10. Its expectation is thus 10.1, which is bigger than 10. Thus, there must exist at least one box which has more than 10 balls in it – for otherwise, how could the average be greater than 10?

Shannon uses this idea, which could be called the first principle of the probabilistic method[7] in his random coding argument. The idea could be summarized as follows:

> A random variable must assume at least one value no smaller than its expectation, and at least one value no greater than its expectation.

[6] Actually, what can be shown is that $|A_\epsilon^{(n)}(Y|x^n)| \leq 2^{n(H(Y|X)+2\epsilon)}$ for all n, but there's no lower bound. The compromise that allows us to at least be able to wave our hands in this direction is to take an averaged lower bound: $E_X^n|A_\epsilon^{(n)}(Y|X^n)| \geq (1-\epsilon)2^{nH(Y|X)-2\epsilon}$.

[7] The second principle, which we don't use, is that if an object chosen randomly from a collection satisfies a certain property with positive probability, then there must be an object in that collection that satisfies that property. The general idea in this line of work is that we are using probabilities and expectations to make some very concrete deterministic statements.

By showing that the average value of $P_e^{(n)}$ is small over all codebooks in existence, it follows that there exists a good codebook with small $P_e^{(n)}$ as well.

8.7.2. *Shifting from the Average Case to the Worst-Case.* Shannon also uses the fact that small average probability of error for a codebook also corresponds to small maximal probability of error over the *best* half of the codebook. This idea is illustrated in the following example, which also strikes of the probabilistic method:

> If the average amount of money in the pockets of 100 ISL graduate students is no more than 1 dollar, then the 50 *poorest* ISL students must all have less than 2 dollars in their pockets.

If statements like these are not obvious to you, you're not alone because I don't think they're that obvious either. For proofs, see Appendix D.

To spell it out, the analogy being drawn with the random coding argument is

random coding argument	ISL
$M = 2^{nR}$	number of students
λ_i	each student's cash
$P_e^{(n)} \leq 2\epsilon$	average cash $\leq$ \$1
$\lambda^{(n)} \leq 4\epsilon$	max cash amongst the poorest half $\leq$ \$2

8.8. **Achievability.**

8.8.1. *The Objective, and The Interplay Between $P_e^{(n)}$ and $\lambda^{(n)}$, and Why W is Uniformly Distributed.* First, let us make clear what the objective is.

Problem For $R < C$, demonstrate the existence of a sequence of $(2^{nR}, n)$ codes with maximum probability of error $\lambda^{(n)} \to 0$.

We want to find codes with small $\lambda^{(n)}$. Observe that the objective function we're trying to minimize, $\lambda^{(n)}$, has nothing to do with the distribution of W at all. $\lambda^{(n)}$ is a maximum of *conditional* probabilities of error.[8] So we're asking for something very strong – a sequence of codes such that, no matter what codeword you decide to send, the probability of decoding it incorrectly on the other side is arbitrarily low.

In the proof however, we will assume that W, the random variable that determines which codeword is sent, is uniformly distributed. This corresponds to computing, for any fixed codebook, the average probability of error over all codewords, which we will end up showing is arbitrarily small. Then in the final stages, we will use the "throwing away codewords" technique mentioned in Section 8.7.2 to argue that the maximal probability of error also becomes arbitrarily small, thereby accomplishing the objective.

So, in summary, the chain of ideas is as follows:

(a) We really want to minimize $\lambda^{(n)}$.

[8]So perhaps one might prefer to describe $\lambda^{(n)}$ as "maximal *conditional* probability of error".

(b) Temporarily assume W is uniform, which corresponds to analyzing the performance of $P_e^{(n)}$ instead.

(c) Set up random coding argument which is facilitated by the fact that we are analyzing $P_e^{(n)}$ instead, since the form of $P_e^{(n)}$ is symmetric.

(d) Discover that $P_e^{(n)}$ goes to zero.

(e) Use the "throwing away codewords" technique to show that $\lambda^{(n)}$ also goes to zero.

(f) So we have found a sequence of truncated codebooks, which asymptotically have the same rate, that achieves the objective.

As Thomas Cover says, "$P_e^{(n)} = \frac{1}{M}\sum_{i=1}^{M} \lambda_i$ is merely a construct of the λ_i's that we put into the argument and then withdraw at the appropriate time."

8.8.2. *Coding Scheme.*

- A codebook is just a tall, skinny matrix with $M := 2^{nR}$ rows and n columns. Each row is a codeword corresponding to some message $w \in \{1, \ldots, M\}$.
- We generate a random codebook $\mathcal{C}$, where every matrix entry is i.i.d. according to $p(x)$.

  Note that any codebook is simply a matrix, where each row is a codeword. So for any fixed codebook $\tilde{\mathcal{C}}$, there is a nonzero probability that this random matrix generation experiment will create the codebook $\tilde{\mathcal{C}}$, as long as the input distribution $p(x)$ is adjusted accordingly – and recall that we can manipulate this distribution however we wish. In this way, Shannon considers all possible codes in existence through his random coding argument.
- This codebook is revealed to both sender and receiver. To send message w, you send the wth row of the matrix, which we denote by $x^n(w)$.
- The receiver gets y^n, and uses Jointly Typical Decoding:

  i. If there exists exactly one $\hat{x}^n$ sequence such that the pair $(\hat{x}^n, y^n)$ is jointly typical, then he declares that $\hat{x}^n$ was sent.

  ii. Otherwise, he declares an error.

  So there are two possibilities for error:

  1. The transmitted codeword x^n and received sequence y^n are not jointly typical.

     By the Joint AEP, the probability of this happening goes to zero.

  2. There is a wrong codeword (not the sent one) that is jointly typical with the received sequence. (This is "confusion.")

     The probability of any individual wrong codeword being jointly typical with the received sequence is about $2^{-nI(X;Y)}$.

8.8.3. *Error Analysis.* Let the error event be $\mathcal{E} := \{\hat{W}(Y^n) \neq W\}$. We now compute the **average** probability of error, $\mathbf{P}[\mathcal{E}]$.

Subtle things to notice:

- $\mathbf{P}[\mathcal{E}]$ is *not* computed from the perspective of the point in time when the sender and receiver have already fixed their codebook. It is computed from the perspective of *before* that point in time, when the codebook has not been randomly generated yet.
- Thus, $\mathbf{P}[\mathcal{E}]$ will involve an average over all possible randomly generated codebooks.
- Furthermore, once you have fixed a codebook, the codeword that you decide to send is chosen uniformly at random, and different codewords might have different resulting probabilities of error. Any fixed codebook thus has its own average probability of error, averaged over which codeword was sent.
- Thus, $\mathbf{P}[\mathcal{E}]$ actually takes **two** random experiments into account – which codebook was generated, and which codeword was sent.
- So $\mathbf{P}[\mathcal{E}]$ is an iterated average. In fact, it is an average of averages. Perhaps you will find this ghastly equation illuminating:

$$\mathbf{P}[\mathcal{E}] = E_{\mathcal{C}}\left[E_W\left[\Pr(\hat{W}(Y^n) \neq W|W)\right]\right]$$

- The fact that we have two expectations foreshadows that Shannon will make two improvements, both in the style of the probabilistic method.

The computation:

$$\begin{aligned}
\mathbf{P}[\mathcal{E}] &\overset{(a)}{=} \sum_{\mathcal{C}} p(\mathcal{C}) P_e^{(n)}(\mathcal{C}) \\
&\overset{(b)}{=} \sum_{\mathcal{C}} p(\mathcal{C}) \sum_{w=1}^{M} \frac{1}{M} \lambda_w(\mathcal{C}) \\
&\overset{(c)}{=} \frac{1}{M} \sum_{w=1}^{M} \sum_{\mathcal{C}} p(\mathcal{C}) \lambda_w(\mathcal{C}) \\
&\overset{(d)}{=} \frac{1}{M} \sum_{w=1}^{M} \sum_{\mathcal{C}} p(\mathcal{C}) \lambda_1(\mathcal{C}) \\
&\overset{(e)}{=} \sum_{\mathcal{C}} p(\mathcal{C}) \lambda_1(\mathcal{C}) \\
&\overset{(f)}{=} \mathbf{P}[\mathcal{E}|W=1].
\end{aligned}$$

where

(a) by definition of average probability of error over all codebooks

(b) again by definition, since $P_e^{(n)}$ is an average probability of error for a fixed codebook, and the message is chosen uniformly at random with probability $\frac{1}{M}$

(c) interchanging the order of summation

(d) by symmetry of the codebook construction and assumed symmetry in the decoding algorithm[9], $\sum_{\mathcal{C}} p(\mathcal{C})\lambda_w(\mathcal{C})$ should not depend on what message was sent, so we can assume WLOG that $W = 1$

[9]One possibility is that the codebook performance should be invariant of permutations of the codebook.

(e) the summation over w can now be dropped, since no summands involve w, and there is a normalization factor of $\frac{1}{M}$

(f) total probability theorem

Define $E_i = \{(X^n(i), Y^n) \in A_\epsilon^{(n)}(X)\}$ for $i \in \{1, \ldots, M\}$. Then

$$\begin{aligned}
\mathbf{P}\left[\mathcal{E}|W=1\right] &\overset{(a)}{=} \mathbf{P}\left[E_1^c \cup E_2 \cup E_3 \ldots \cup E_M | W=1\right] \\
&\overset{(b)}{\leq} \mathbf{P}\left[E_1^c|W=1\right] + \sum_{i=2}^{M} \mathbf{P}\left[E_i|W=1\right] \\
&\overset{(c)}{\leq} \epsilon + \sum_{i=2}^{M} \mathbf{P}\left[E_i|W=1\right] \\
&\overset{(d)}{\leq} \epsilon + \sum_{i=2}^{M} 2^{-n(I(X;Y)-3\epsilon)} \\
&\overset{(e)}{=} \epsilon + (M-1)2^{-n(I(X;Y)-3\epsilon)} \\
&= \epsilon + (2^{nR}-1)2^{-n(I(X;Y)-3\epsilon)} \\
&\leq \epsilon + 2^{nR}2^{-n(I(X;Y)-3\epsilon)} \\
&= \epsilon + 2^{n(R-I(X;Y)+3\epsilon)} \\
&= \epsilon + \frac{1}{2^{n(-R+I(X;Y)-3\epsilon)}} \\
&\overset{(f)}{\leq} 2\epsilon
\end{aligned}$$

where each step is justified as follows:

(a) two types of error, as mentioned before

(b) union bound

(c) by the joint AEP, the first term is arbitrarily small for sufficiently large n

(d) applying the upper bound that was emphasized in the Joint AEP section

(e) all terms in the summation do not depend on i

(f) as long as $R < I(X;Y) - 3\epsilon$, then the second term goes to zero as n grows large.

- Thus, the average probability of error averaged over all codebooks (the outermost expectation in the double expectation) can be made really small. By the first principle of the probabilistic method, there must exist a good codebook $\mathcal{C}^*$ whose average probability of error is at most 2ϵ; that is, $\mathbf{P}\left[\mathcal{E}|\mathcal{C}^*\right] \leq 2\epsilon$.
- If the average probability of error for $\mathcal{C}^*$ is no more than 2ϵ, then the best half of the codewords in $\mathcal{C}^*$ must each have conditional probabilities of error no greater than 4ϵ. This follows from the result proven in Appendix D. The number of codewords we are left with is

$$\frac{M}{2} = \frac{2^{nR}}{2} = 2^{nR-1} = 2^{n(R-\frac{1}{n})}$$

thereby changing the rate from R to $R - \frac{1}{n}$, a negligible loss as n grows large.

So getting rid of the outer expectation allowed us to find a good codebook, and getting rid of the inner expectation allowed us to find a great codebook.

- Whenever we have an inequality, we try to strengthen it. The inequality
$$R < I(X;Y)$$
has a constant on the left-hand side, and an expression in terms of X and Y on the right. And we can control $p(x)$, the input distribution. So we should maximize the right-hand side of the inequality over all possible $p(x)$ so as to allow the rate R to be as high as possible. This yields $R < \max_{p(x)} I(X;Y) = C$.

Thus we have demonstrated the existence of a code of rate $R' = R - 1/n$ with maximal probability of error below 4ϵ. This proves the achievability of any rate below capacity.

Note that random coding is only the method of proof. It is not how we achieve the capacity. The actual code is a deterministic object, but we just don't know what it is. Just as we don't know which of the 10 boxes in Subsection 8.7 has more than 10 balls in it.

8.9. **Weak Converse.**

8.9.1. *The Objective, The Interplay Between $\lambda^{(n)}$ and $P_e^{(n)}$, and Why W is Uniformly Distributed.* Again, let us make clear what the objective is.

Problem Given a sequence of $(2^{nR}, n)$ codes such that $\lambda^{(n)} \to 0$, show that $R \leq C$.

So it is given to us in the premise that $\lambda^{(n)} \to 0$. Now we use the following fact:

Lemma 8.1. *If $\lambda^{(n)} \to 0$, then $P_e^{(n)} \to 0$.*

Proof.
$$0 \leq P_e^{(n)} = \frac{1}{M}\sum_{i=1}^{M} \lambda_i \leq \frac{1}{M}\sum_{i=1}^{M} \max_{i \in \{1,2,\ldots,M\}} \lambda_i = \frac{1}{M}\sum_{i=1}^{M} \lambda^{(n)} = \lambda^{(n)}.$$
□

Our approach in proving the converse will be to show that $R \leq C$ assuming only that $P_e^{(n)} \to 0$. This is a stronger statement, since our new premise is weaker than our original one.

Having established this, now observe that if W is uniformly distributed, two nice equations would hold:

(a) $nR = H(W)$

(b) $P_e^{(n)} = \mathbf{P}\left[\hat{W} \neq W\right]$

Why do we like the first equation, $nR = H(W)$? Think about it from a problem-solving perspective. There is a blank piece of paper in front of us, and we need to prove that $R \leq C$. So, somehow starting with the letter R, we're going to play with inequalities, and finally end up with the letter C. But what's the first equation that can give us an R to start with? $nR = H(W)$ provides us with an option, and it holds if W is uniformly distributed. It is also

a natural choice since 2^{nR} is defined to be the number of rows in the codebook matrix, and a non-uniform distribution on W would result in $H(W) = g(R)$ where g is an equation probably not nearly as simple as $nR = H(W)$.

As for the second equation, again we can think about it from a problem-solving perspective. We need to leverage the one piece of information we have, which is $P_e^{(n)} \to 0$. How can we incorporate this? It so happens that under a uniform distribution, $P_e^{(n)} = \frac{1}{M}\sum_{i=1}^{M} \lambda_i = \sum_{i=1}^{M} \frac{1}{M}\lambda_i = \mathbf{P}\left[\hat{W} \neq W\right]$. (This is like taking the $\frac{1}{M}$ scale factor and reinterpreting it as a probability distribution for uniform W.) And this is great because we know how to deal with expressions like $\mathbf{P}\left[\hat{W} \neq W\right]$ – we apply Fano's inequality.

In summary, the decision to make W uniformly distributed is purely algebraically motivated. Remember that it's all for the sake of proving the mathematical relationship $R \leq C$ – we don't care how we have to go about doing it. Assuming that W is uniform allows us to (1) write down $nR = H(W)$ and (2) make the connection with $P_e^{(n)} \to 0$ and bust out Fano's inequality. As Thomas Cover puts it, "assuming W to be uniform is just an artifice to make the probability of error equal to the constructed sum of the λ_i's."

8.9.2. *Preliminary Lemmas.* There are two preliminary tools needed:

- Fano's inequality: $H(W|\hat{W}) \leq 1 + P_e^{(n)} nR$
- DMC mutual information:

Lemma 8.2. *For a DMC without feedback: $I(X^n; Y^n) \leq nC$.*

Proof.

$$\begin{aligned} I(X^n; Y^n) &= H(Y^n) - H(Y^n|X^n) \\ &= H(Y^n) - \sum H(Y_i|Y_1, \ldots, Y_{i-1}, X^n) \\ &= H(Y^n) - \sum H(Y_i|X_i) \\ &\leq \sum H(Y_i) - \sum H(Y_i|X_i) \\ &= \sum I(X_i; Y_i) \\ &\leq nC \end{aligned}$$

where step 3 is the key step. □

Theorem 8.3. *If $P_e^{(n)} \to 0$ for a sequence of $(2^{nR}, n)$ codes, then $R < C$.*

$$\begin{aligned} nR &\overset{(a)}{=} H(W) \\ &\overset{(b)}{=} H(W|\hat{W}) + I(W;\hat{W}) \\ &\overset{(c)}{\leq} 1 + P_e^{(n)} nR + I(W;\hat{W}) \\ &\overset{(d)}{\leq} 1 + P_e^{(n)} nR + I(X^n;Y^n) \\ &\overset{(e)}{\leq} 1 + P_e^{(n)} nR + nC \\ R &\overset{(f)}{\leq} P_e^{(n)} R + \frac{1}{n} + C \end{aligned}$$

where

(a) W is uniformly distributed over $\{1, 2, \ldots, M = 2^{nR}\}$

(b) chain rule

(c) Fano

(d) data processing: $W \to X^n \to Y^n \to \hat{W}$

(e) Lemma 8.2, DMC mutual information

(f) division by n

By the premise, $P_e^{(n)} \to 0$ as $n \to \infty$. Thus $R \leq C$.

Note that we can also rewrite inequality (f) above as

$$P_e^{(n)} \geq 1 - \frac{C}{R} - \frac{1}{nR}$$

which shows that if $R > C$, then the probability of error is bounded away from 0.

8.10. **Error Exponents.** (not covered in class)

The Weak Converse is called the Weak Converse is because one can prove a stronger result that also talks about convergence rates. It can be shown that when $R < C$, the probability of error decays exponentially fast in n (see [Gal68]). However, when $R > C$, the probability of error is bounded away from 0, and it can also be shown that the probability of error blows up to 1 exponentially as $n \to \infty$. Thus, C is the dividing line marking a dramatic phase transition.

8.11. **Feedback Capacity.**

Theorem 8.4.

$$C_{FB} = C.$$

The approach in the proof is to show that $C \leq C_{FB}$ and $C \geq C_{FB}$.

To show $C \leq C_{FB}$, we can use the same achievability proof that we did without feedback. This shows that we can achieve all rates up to C, simply by ignoring the feedback line.

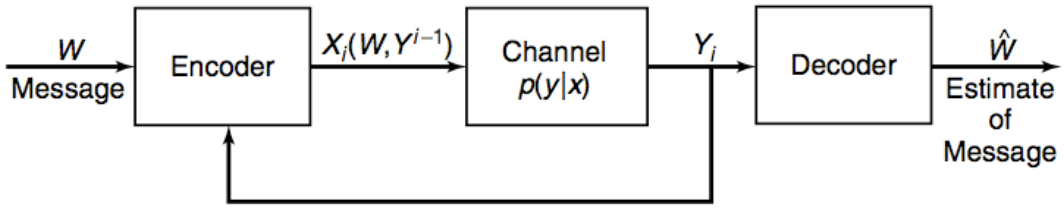


FIGURE 7. DMC with feedback.

The converse is to show that $C \geq C_{FB}$. The key problem now is that we cannot use the "DMC mutual information Lemma" anymore because the crux move in the proof of the Lemma, step 3, is no longer valid when there is feedback. So we have to proceed differently. See the book for the proof. The difference is a more delicate application of the data processing inequality. Given the Markov chain $W \to X^n \to Y^n \to \hat{W}$, the previous converse proof uses $I(W; \hat{W}) \leq I(X^n; Y^n)$, whereas the converse proof with feedback doesn't jump so far at first: $I(W; \hat{W}) \leq I(W; Y^n)$.

8.12. **Source-Channel Separation.**

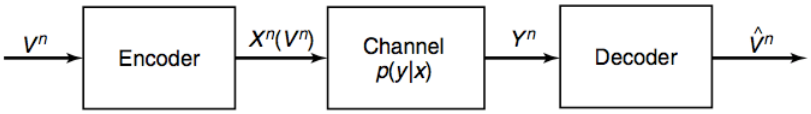


FIGURE 8. Joint coding.

8.12.1. *The Setup.*

Problem Given a *stochastic process* (V_i), send the symbols V^n over the channel so that the receiver can reconstruct the sequence.

(a) Take a block of symbols V^n and encode it as $X^n(V^n)$.

(b) Send the codeword $X^n(V^n)$ across the noisy channel.

(c) Receiver gets Y^n and makes an estimate $\hat{V}^n = g(Y^n)$.

(d) Error occurs if $\hat{V}^n \neq V^n$.

The point of Figure 8 is that we considering the possibility of doing things in less stages. Rather than first compressing the stochastic process and adding redundancy back to prepare it for transmission over the channel, we are open to the possibility of doing it all in one stage, with one code. Hence, when using this approach, we call it a *source-channel* code, with a hyphen.

However, it turns out that a one-stage approach does not outperform a two-stage approach – the achievable rate region is no different. The plan of attack for arguing this is as follows:

- Converse: Demonstrate that there is cutoff capacity C such that no sequence of source-channel codes can achieve rates higher than C.
- Achievability: Achieve all rates below C by using a particular kind of source-channel code – a two-stage scheme that separates source coding and channel coding. (The two-stage scheme can be thought of as a special case of the one-stage scheme.)

8.12.2. *The Source-Channel Separation Theorem.*

Theorem 8.5. *Source-Channel Separation Theorem:*

- *If (X_i) is a finite alphabet stochastic process satisfying the AEP and $H(\mathcal{X}) < C$, then there exists a source-channel code such that $\mathbf{P}\left[\hat{V}^n \neq V^n\right] \to 0$.*
- *If $H(\mathcal{X}) > C$, then the probability of error is bounded away from zero. (Or contrapositive form: If $\mathbf{P}\left[\hat{V}^n \neq V^n\right] \to 0$, then $H(\mathcal{X}) < C$.)*

Proof. Achievability: Two stages. First, use AEP to do compression. Only encode source sequences that are in the typical set, and let everything else incur an error – it doesn't matter since the probability of anything atypical goes to zero. The number of source sequences is $2^{nH(\mathcal{V})}$. Now do channel coding, and apply Shannon's random coding argument. Since $H(\mathcal{V}) < C$, we win.

Converse: We'll show the contrapositive. Suppose we have $\mathbf{P}\left[\hat{V}^n \neq V^n\right] \to 0$ for a sequence of source-channel codes.

$$\begin{aligned}
H(\mathcal{V}) &\overset{(a)}{\leq} \frac{H(V_1, \ldots, V_n)}{n} \\
&= \frac{H(V^n)}{n} \\
&= \frac{1}{n} H(V^n|\hat{V}^n) + \frac{1}{n} I(V^n; \hat{V}^n) \\
&\overset{(b)}{\leq} \frac{1}{n}(1 + \mathbf{P}\left[\hat{V}^n \neq V^n\right] n \log |\mathcal{V}|) + \frac{1}{n} I(V^n; \hat{V}^n) \\
&\overset{(c)}{\leq} \frac{1}{n}(1 + \mathbf{P}\left[\hat{V}^n \neq V^n\right] n \log |\mathcal{V}|) + \frac{1}{n} I(X^n; Y^n) \\
&\overset{(d)}{\leq} \frac{1}{n} + \mathbf{P}\left[\hat{V}^n \neq V^n\right] \log |\mathcal{V}| + C
\end{aligned}$$

where each step is justified as follows:

(a) Recall that for a stationary process, it was shown on pages 75 and 76 that

- $H(\mathcal{V}) = \lim_{n\to\infty} \frac{H(V_1,\ldots,V_n)}{n} = \lim_{n\to\infty} H(V_n|V_1, V_2, \ldots, V_{n+1})$, and
- $H(X_n|X_{n-1}, \ldots, X_1)$ is non-increasing in n.

However, the inequality in this step claims that $H(\mathcal{V}) \leq \frac{H(V_1,\ldots,V_n)}{n}$, which does not directly follow from either of the above two facts. So more argument is needed.

Let $H_n := H(V_1, \ldots, V_n)$. We want to argue that H_n is non-increasing; that is, $H_{n+1} \leq H_n$. By the chain rule,

$$\begin{aligned}
H_{n+1} &\overset{(i)}{=} \frac{1}{n+1} \sum_{i=1}^{n+1} H(V_i|V^{i-1}) \\
&\overset{(ii)}{=} \frac{1}{n+1} \left(H(V_{n+1}|V^n) + \sum_{i=1}^{n} H(V_i|V^{i-1}) \right) \\
&\overset{(iii)}{=} \frac{1}{n+1} H(V_{n+1}|V^n) + \frac{n}{n+1} \cdot \frac{1}{n} \left(\sum_{i=1}^{n} H(V_i|V^{i-1}) \right) \\
&\overset{(iv)}{=} \frac{1}{n+1} H(V_{n+1}|V^n) + \frac{n}{n+1} \cdot H_n \\
&\overset{(v)}{\leq} \frac{1}{n+1} H_n + \frac{n}{n+1} \cdot H_n \\
&= H_n.
\end{aligned}$$

where each step is justified as follows:

(i) chain rule

(ii) first crux move: peel off the last term in the summation

(iii) multiply and divide by n

(iv) definition of H_n

(v) the second, and main, crux move: Recall from the textbook that the sequence of conditional entropies $H(V_n|V^{n-1})$ is non-increasing, and thus $H(V_n|V^{n-1})$ is smaller than all of the terms $H(V_i|V^{i-1})$ for $i \in \{1, 2, \ldots, n\}$. Since $H_n = \sum_{i=1}^{n} \frac{1}{n} H(V_i|V^{i-1})$, the average of a bunch of terms all larger than $H(V_{n+1}|V^n)$, it follows that $H(V_{n+1}|V^n) \leq H_n$.

(b) Fano's inequality

(c) data processing inequality

(d) Lemma 8.2: $I(X^n; Y^n) \leq nC$

By the premise, $\mathbf{P}\left[\hat{V}^n \neq V^n\right]$ goes to zero. Thus $H(\mathcal{V}) \leq C$.

□

8.12.3. *The Take-Home Message:* Instead of doing joint encoding, as shown in Figure 8, we can do separate source and channel coding, as show in Figure 9.

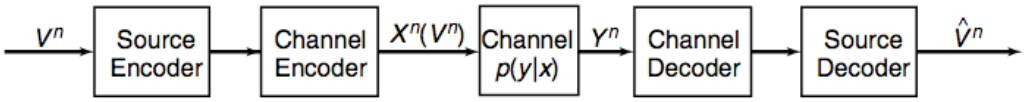


FIGURE 9. Separate source and channel coding.

8.13. **Computing Capacities.**

8.13.1. *General Procedure.*

(a) Write the expression for $I(X;Y)$.

(b) At this point, you'll probably have to decide to either break it as $H(X) - H(X|Y)$ or $H(Y) - H(Y|X)$. Depending on the structure of the channel, one way probably results in less work than the other.

(c) Simplify. For example, maybe $H(Y|X)$ has a simple form that does not involve $p(x)$ at all, as in the BEC.

(d) Now maximize the expression over all possible input distributions $p(x)$. If the input distribution is binary, you are maximizing over all $\pi \in [0,1]$ where π is the probability that $X = 1$.

8.13.2. *Tricks for simplifying $I(X;Y)$.*

(a) Cascades of BSCs are BSCs under different parameters.

(b) Introduce an auxiliary random variable to simplify your life (e.g., the erasure indicator variable E in the BEC)

8.13.3. *Tricks for maximization.*

(a) Pull out all terms that don't involve the variable you are maximizing over

(b) Use calculus to determine the input distribution π^*, and then plug in π^* to get the maximum value of $I(X;Y)$.

(c) If you have to differentiate something involving logarithms and it is in log base 2, you can ignore the fact that it is base 2 and use the standard calculus rules for differentiating logarithms as if it were in base e. The reason is, if you pull out a base change factor to make the logarithms base e, and then do calculus, after the smoke has cleared you will just absorb the base change factor back in to make it base 2 again.

Useful fact: $\frac{d}{dp}H(p) = \frac{1}{\log_e 2}\log_e \frac{1-p}{p} = \log_2 \frac{1-p}{p}$.

(d) Inequalities: Apply inequalities, and apply conditions for equality if possible. See Section 6.

8.13.4. *General short-circuit tricks.*

(a) If you cascade channels, then the capacity of the cascade is upper bounded by the minimum of the individual capacities: $C \leq \min(C_1, C_2)$. (HW8, Problem 2)

(b) If you cascade channels and allow intermediate encoding and decoding, the capacity equals the minimum of the individual capacities. $C = \min(C_1, C_2)$.(HW8, Problem 2)

(c) If you take the product of two channels, the capacities add. $C = C_1 + C_2$. (HW7, Problem 7)

8.14. **Random Channel Capacity Exercises.**

- Sum channel / choice of channels. Take two channels, and rather than sending along both simultaneously, you are only allowed to send on one of the two). What is the capacity of the new channel in terms of the capacities of the original two channels?
- Random problems from the book (originally planned to do a whole bunch of these, but by then it was 7PM and apparently people had lives to attend to or food to eat)

9. Differential Entropy

Almost everything is the same. Replace sums with integrals and PMFs with densities. You have algebraic relations, a typical set, and so forth. Use a lower case h.

But here are a few important differences from the discrete cases:

- Normal Distribution: $h(\mathcal{N}(0,\sigma^2)) = \frac{1}{2}\log 2\pi e\sigma^2$.
- Maximum Entropy Subject to Power Constraint: Given a variance constraint, the normal distribution maximizes the entropy. In general, for a length n random vector $\mathbf{X}$,
$$\max_{E\mathbf{X}\mathbf{X}^t=K} h(\mathbf{X}) = \frac{1}{2}\log(2\pi e)^n|K|$$
where $|K|$ is the determinant. When $n = 1$, we have the normal distribution entropy directly above:
$$h(X) \leq \frac{1}{2}\log 2\pi e\sigma^2 \qquad \text{if } \mathbf{var}(X) \leq \sigma^2$$
- Scaling: $h(aX) = h(X) + \log|a|$
- Differential entropy can be negative, but 2^h, the "effective alphabet size", is always nonnegative.

  What do we mean by "effective alphabet size"? Firstly, recall that for a uniformly distributed discrete random variable U, $H(U) = \log|\mathcal{X}|$, the number of bits required to describe the size of the alphabet set $\mathcal{X}$. Thus $2^{H(U)} = |\mathcal{X}|$, the alphabet size.

  Now, if you have an arbitrary random variable X with a certain amount of entropy in it, one can ask, what uniform random variable U would have the same amount of entropy as X? The answer is: a uniform random variable whose alphabet size is $2^{H(X)}$.

  Thus, at least in this respect, we need not be bothered by negative differential entropy. For a uniform continuous random variable U whose density has support of length a, $H(U)$ may be negative, but $2^{H(U)}$ is still positive, as it is the length of an interval.

10. Gaussian Channel Capacity

10.1. **Information Capacity.**

Theorem 10.1. *The information capacity (defined as simply* $\max_{p(x)} I(X;Y)$*) of the additive Gaussian noise channel with a power constraint* P *and noise variance* N *is*
$$C = \frac{1}{2}\log\left(1 + \frac{P}{N}\right)$$
and is achieved by setting the input distribution to the normal distribution $\mathcal{N}(0,P)$*.*

Proof.

$$\begin{aligned} I(X;Y) &= h(Y) - h(Y|X) \\ &= h(Y) - h(X+Z|X) \\ &= h(Y) - h(Z|X) \\ &\overset{(a)}{=} h(Y) - h(Z) \\ &\overset{(b)}{=} h(Y) - \frac{1}{2}\log 2\pi eN \\ &\overset{(c)}{\leq} \frac{1}{2}\log 2\pi e(P+N) - \frac{1}{2}\log 2\pi eN \\ &= \frac{1}{2}\log\left(1+\frac{P}{N}\right). \end{aligned}$$

(a) Z is independent of X.

(b) Z has a Gaussian distribution.

(c) Since $Y = X + Z$, $\mathbf{E}\left[Y^2\right] = \mathbf{E}\left[X^2 + 2XZ + Z^2\right] = \mathbf{E}\left[X^2\right] + 2\underbrace{\mathbf{E}\left[X\right]}_{=0}\underbrace{\mathbf{E}\left[Z\right]}_{=0} + \mathbf{E}\left[Z^2\right] = P + N$. We know that the differential entropy under a variance constraint is maximized by the normal distribution, and thus the inequality follows.

□

We have not shown that this information capacity corresponds to the operational definition of capacity – the maximum achievable rate. We cannot just quote the achievability and converse theorems from Chapter 7 because two things are different:

- now we are using continuous probability distributions, whereas the theory before pertained to discrete distributions, and
- we have a cost constraint on the covariance matrix of the input distribution.

10.2. **Achievability.** Everything is the same as the generalized discrete achievability proof, except for two things.

- Codewords are generated i.i.d. according to a *normal* distribution $\mathcal{N}(0, P-\epsilon)$. By the WLLN, $\frac{1}{n}\sum X_i^2 \xrightarrow{P} P - \epsilon$, and thus $\sum X_i^2 \leq n(P-\epsilon)$ with high probability for large n, satisfying the power constraint.
- When decoding, we also declare an error if the power constraint is violated.

Regarding the first fix, without having some background in analysis, it may be unclear why we should set the variance to $P - \epsilon$ instead of just P. If you are interested in a more detailed explication of why this is, please see Appendix H.

10.3. **Converse.** This is different from the DMC case, but the outline is the same.

Theorem 10.2. *Suppose $P_e^{(n)} \to 0$ for a sequence of $(2^{nR}, n)$ codes for a Gaussian channel with power constraint P. Then $R \leq C$.*

Proof.

$$
\begin{aligned}
nR &= H(W) \\
&\overset{(a)}{=} H(W|\hat{W}) + I(W;\hat{W}) \\
&\overset{(b)}{\leq} 1 + P_e^{(n)} nR + I(W;\hat{W}) \\
&\overset{(c)}{\leq} 1 + P_e^{(n)} nR + I(X^n;Y^n) \\
&= 1 + P_e^{(n)} nR + h(Y^n) - h(Y^n|X^n) \\
&\overset{(d)}{\leq} 1 + P_e^{(n)} nR + \sum_i h(Y_i) - h(Y^n|X^n) \\
&\overset{(e)}{=} 1 + P_e^{(n)} nR + \sum_i h(Y_i) - \sum_{i=1}^n h(Y_i|X^n, Y_1^{i-1}) \\
&\overset{(f)}{=} 1 + P_e^{(n)} nR + \sum_i h(Y_i) - \sum_{i=1}^n h(Y_i|X_i) \\
&= 1 + P_e^{(n)} nR + \sum_i I(X_i;Y_i) \\
&\overset{(g)}{\leq} 1 + P_e^{(n)} nR + \sum_i \frac{1}{2} \log\left(1 + \frac{P_i}{N}\right) \\
R &\overset{(h)}{=} \frac{1}{n} + P_e^{(n)} R + \sum_i \frac{1}{n} \cdot \frac{1}{2} \log\left(1 + \frac{P_i}{N}\right) \\
&\overset{(i)}{\leq} \frac{1}{n} + P_e^{(n)} R + \frac{1}{2} \log\left(1 + \frac{1}{n} \sum_{i=1}^n \frac{P_i}{N}\right) \\
&\overset{(j)}{\leq} \frac{1}{n} + P_e^{(n)} R + \frac{1}{2} \log\left(1 + \frac{P}{N}\right)
\end{aligned}
$$

where

(a) chain rule

(b) Fano

(c) data processing inequality

(d) independence bound

(e) chain rule

(f) DMC + no feedback

(g) divide both sides by n

(h) Define P_i to be the *average* power of the ith entry over all the x^n codewords; that is, $P_i = \frac{1}{2^{nR}} \sum_w x_i^2(w)$. Note that $Y_i = X_i + Z_i$. Apply the result from Theorem 10.1.

(i) Use Jensen's inequality. Think of $\frac{1}{n}$ as the distribution which makes the summation look like an expectation. Then Jensen says that for concave functions such as $f(x) = \frac{1}{2}\log(1+x)$, $\mathbf{E}[f(X)] \leq f(\mathbf{E}[X])$.

(j) Let ϕ_i be a random variable denoting the power of the ith entry of a uniformly chosen codeword that is sent. Then $P_i = E_W \phi_i$ and

$$\frac{1}{n}\sum_{i=1}^{n} P_i = \frac{1}{n}\sum_{i=1}^{n} E_W[\phi_i] = E_W\left[\underbrace{\frac{1}{n}\sum_{i=1}^{n}\phi_i}_{\leq P}\right] \leq P$$

where the last inequality holds because the summation framed inside the expectation brackets is the sum power of a single randomly chosen codeword, and the premise is that any single codeword satisfies the power constraint.

Thus, $\frac{1}{n}\sum P_i \leq P$. We can plug this upper bound into the argument of the logarithm because the logarithm is an increasing function.

By the premise, $P_e^{(n)} \to 0$. Thus, $R \leq \frac{1}{2}\log(1 + \frac{P}{N})$. □

Appendix A. Additional Inequalities

(not covered in class)

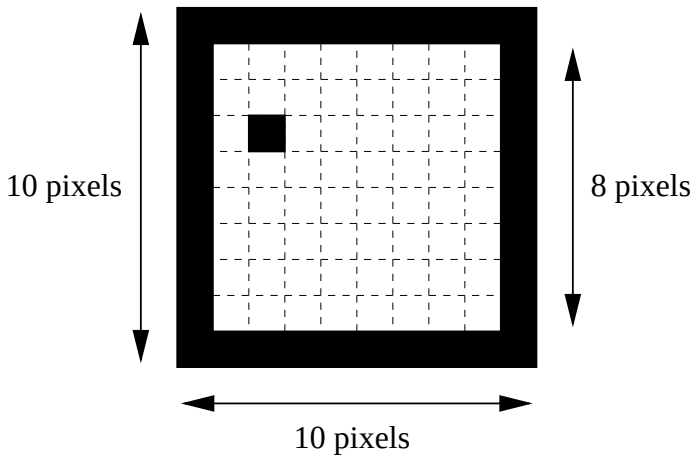
- Triangle Inequality: For two real numbers, $|x+y| \leq |x|+|y|$. Reverse triangle inequality: $||x| - |y|| \leq |x - y|$.
- Cauchy-Schwarz Inequality: For sequences (a_n) and (b_n), $\sum a_n b_n \leq (\sum a_n^2)^{\frac{1}{2}}(\sum b_n^2)^{\frac{1}{2}}$ with equality if and only if one sequence is a scalar multiple of the other.
- AM-GM Inequality: For nonnegative real numbers $(x_i)_{i=1}^n$, $\frac{x_1+x_2+\ldots+x_n}{n} \leq (x_1 x_2 \cdots x_n)^{\frac{1}{n}}$, with equality if and only if the sequence is constant.
- Holder's Inequality: If $p > 1$ and $\frac{1}{p} + \frac{1}{q} = 1$, and the sequences (a_i) and (b_i) have only non-negative terms, then $\sum_{i=1}^n a_i b_i \leq (a_1^p + \ldots + a_n^p)^{1/p}(b_1^q + \ldots + b_n^q)^{1/q}$. We have equality for a nonzero sequence (a_i) if and only if there exists a constant $\lambda \in \mathbf{R}$ such that $\lambda a_k^{1/p} = b_k^{1/q}$ for all $1 \leq k \leq n$.

Appendix B. Kolmogorov Complexity Conditioned On Knowing $l(x)$

Let me explain by example why it is convenient to assume that we know the length of x in advance. As we did in the homework, suppose we wish to generate the following image:

The arrows, ASCII characters, and dashed lines are only there for visual reference and annotative purposes; we do not want to generate them. The goal is to construct the image corresponding to a single black pixel turned on inside an $m \times m$ grid of otherwise all white pixels, which are framed by a one-pixel width bounding square of black pixels. In the figure, $m = 8$.

First, we must tell the computer to draw the bounding square, which has length $m + 2$. The instructions corresponding to "add 2" are of constant length; in fact, it requires at least 1 bit, to specify the number 2. Given these instructions, it suffices to tell the computer what m is. The binary representation of m has $\log m$ bits. However, if you feed these bits into the



computer, how does the computer know that it has reached the end of the description for m? Maybe more bits will be coming along the way.

One way is to use a 2-repetition code. If $m = 100$ in binary, then we duplicate every digit to make 110000, and then append 01 to the end to indicate to the computer that the binary sequence corresponding to the description of m has terminated. This requires $2 \log m$ bits.

Alternatively, a more efficient way to describe m is to do so recursively. We could specify the number of bits in the binary representation of m, which is $\log m$, and then specify the actual bits of m. But then we have the same problem of specifying $\log m$, which could be done in the same manner by first specifying $\log \log m$. And so on. This results in a program of length $\log m + \log \log m + \log \log \log m + \ldots$ bits, which we denote as $\log^* m$ in short.

Now that the computer knows m, it will be easier to describe the isolated black pixel. Clearly, we should give the computer two numbers corresponding to the coordinates of the isolated black pixel: $x \in \{1, \ldots, m\}$, and $y \in \{1, \ldots, m\}$. However, if we just write out the binary representations for both x and y and naively feed them into the computer, the computer has the same problem as before of not knowing where the x sequence ends and the y sequence begins, and then where the y sequence ends. But since m is known to the computer, so is $\log m$. Thus, we could specify both x and y with two zero-padded bit sequences of each length exactly $\log m$, and then feed these $2 \log m$ bits right into the computer. The computer could then properly partition this stream of bits as two blocks each of length $\log m$, since the computer knows the blocklength.

Thus, the Kolmogorov complexity upper bound we have now is

$$K(x) \leq 2 \log m + \log^* m + 1 \approx 2 \log m + \log^* m$$

But having to add terms like $\log^* m$ all the time is annoying. So we just assume m is given, in which case,

$$K(x|m) \leq 2 \log m.$$

Lastly, although the length of the binary string x corresponding to the $m \times m$ images is actually m^2, since the computer can compute m^2 given m, and the extra instruction required to inform the computer that it should compute the square of m has constant length (need only 1 bit to specify the exponent 2), it's sufficient to say that we supply the computer with m, the side length of the bounding square. This reconciles the discrepancy between using

$K(x|l(x)) = K(x|n)$ in the textbook and using $K(x|n)$ in the homework when n was actually the side length of the square.

Bottom line: $K(x|n)$ just means we assume that whatever numbers the computer needs to know to specify the "bounding box" of the object, whether it be the length of the entire string or the side length of an image's frame, are given.

Appendix C. Chernoff Bounds for Binomial Random Variables

(not on the exam)

Let $X = B(n, p)$ denote a binomial random variable with parameter p.

Then $\mathbf{P}[X \leq k] \leq \exp -\frac{1}{2p}\frac{(np-k)^2}{n}$.

Also, if $p > 1/2$, then $\mathbf{P}[X \geq n/2 + 1] = \sum_{i=\frac{n}{2}+1}^{n} \binom{n}{i} p^i (1-p)^{n-i} \geq 1 - e^{-2n(p-1/2)^2}$.

Appendix D. "Throwing Away Codewords" And Similar Arguments

Here's a fact:

> If the average amount of money in the pockets of 100 ISL graduate students is no more than 1 dollar, then the 50 *poorest* ISL students must all have less than 2 dollars in their pockets.

Statements like these are mentioned in many classes and passed off as obvious, but to be honest, I still find them very confusing due to all the inequalities floating around. I've also never seen anyone actually bother to prove it. So let's prove the general result here once and for all.

To connect the following notation with a real-world scenario, imagine enumerating n ISL students using the integers 1 through n, and let $x_1, x_2, \ldots, x_n$ be their corresponding pocket cash *sorted from lowest to highest*. $\bar{x}$ is the average cash amongst all students. Let S be a subset of $[n] := \{1, \ldots, n\}$, where $|S| = \frac{n}{d}$ and d divides into n, where $d \geq 2$.

Theorem D.1. *Let $(x_1, x_2, \ldots, x_n)$ be non-negative real numbers sorted from lowest to highest. If $\bar{x} := \frac{\sum_{i=1}^{n} x_i}{n} \leq \epsilon$, then $\max_{i \in \{1,2,\ldots,n/d\}} x_i \leq d\epsilon$.*

Proof. We will prove the result by contradiction. Suppose that $\max_{i \in \{1,2,\ldots,n/d\}} x_i > d\epsilon$. Then,

$$\begin{aligned}
\bar{x} &= \frac{1}{n} \sum_{i=1}^{n} x_i \\
&= \frac{1}{n} \left(\sum_{i=1}^{\frac{n}{d}} x_i + \sum_{i=\frac{n}{d}+1}^{n} x_i \right) \\
&\overset{(a)}{\geq} \frac{1}{n} \left(\sum_{i=\frac{n}{d}+1}^{n} x_i \right) \\
&\overset{(b)}{>} \frac{1}{n} \left(\sum_{i=\frac{n}{d}+1}^{n} d\epsilon \right) \\
&\overset{(c)}{=} \frac{1}{n} \left(\frac{d-1}{d} \right) \cdot n \cdot d\epsilon \\
&\overset{(d)}{=} (d-1)\epsilon
\end{aligned}$$

where each step is justified as follows:

(a) the x_i's are nonnegative[10], so throwing out the lowest n/d terms gives a lower bound

(b) (crux move) since the x_i's are *sorted*, and the largest value amongst the first $\frac{n}{d}$ is greater than $d\epsilon$, it follows that all x_i's with indices past $\frac{n}{d}$ are also greater than $d\epsilon$

(c) the number of terms in the summation is $\left(\frac{d-1}{d}\right) \cdot n$

(d) cancelling fractions

Thus, $\bar{x} > (d-1)\epsilon \geq \epsilon$, since $d \geq 2$. This contradicts the premise that $\bar{x} \leq \epsilon$. □

For the sake of completeness in killing off allegedly obvious statements of this type, here is a similar type of argument that often shows up:

> If the average amount of money in the pockets of 100 ISL graduate students is 1 dollar, then any subset of 50 ISL students cannot all have more than 2 dollars in their pockets.

Shannon does not use this, but nevertheless it could be useful. Stated as a general result,

Theorem D.2. *Let $(x_1, x_2, \ldots, x_n)$ be non-negative real numbers sorted from lowest to highest. If $\bar{x} := \frac{\sum_{i=1}^{n} x_i}{n} \leq \epsilon$, then $\forall S \subseteq [n]$ such that $|S| = \frac{n}{d}$, it holds that $\min_{i \in S} x_i \leq d\epsilon$.*

[10]Probably an unrealistic assumption for ISL students.

Proof. We proceed by contradiction. Suppose there exists a size $\frac{n}{d}$ subset S such that $\min_{i \in S} x_i > d\epsilon$. Then,

$$\begin{aligned}
\text{average C.R.E.A.M.} = \epsilon &= \frac{1}{n} \sum_{i=1}^{n} x_i \\
&\overset{(a)}{=} \frac{1}{n} \left(\sum_{i \in S} x_i + \sum_{i \in S^c} x_i \right) \\
&\overset{(b)}{\geq} \frac{1}{n} \sum_{i \in S} x_i \\
&\overset{(c)}{>} \frac{1}{n} \sum_{i \in S} d\epsilon \\
&\overset{(d)}{=} \frac{1}{n} \cdot \frac{n}{d} \cdot d\epsilon \\
&= \epsilon
\end{aligned}$$

where each step is justified as follows:

(a) break the sum into two parts

(b) since $x_i \geq 0$, we can throw out the summation for S^c to get a lower bound

(c) follows from the assumption that $\min_{i \in S} x_i > d\epsilon$

(d) the size of S is $\frac{n}{d}$

Thus, $\epsilon > \epsilon$, which is a contradiction. □

Appendix E. Accidentally Jointly Typical

More rigorously, here is how to sandwich $\mathbf{P}\left[(\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}(X,Y)\right]$ from above and below by terms arbitrarily close to $2^{-nI(X;Y)}$. For the upper bound, which applies for all n,

$$\begin{aligned}
\mathbf{P}\left[(\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}(X,Y)\right] &= \sum_{(x^n,y^n) \in A_\epsilon^{(n)}(X,Y)} p(x^n)p(y^n) \\
&\overset{(a)}{\leq} 2^{n(H(X,Y)+\epsilon)} 2^{-n(H(X)-\epsilon)} 2^{-n(H(Y)-\epsilon)} \\
&= 2^{-n(I(X;Y)-3\epsilon)}
\end{aligned}$$

where in (a), we have upper bounded $p(x^n)$ and $p(y^n)$ individually using the facts that $x^n \in A_\epsilon^{(n)}(X)$ and $y^n \in A_\epsilon^{(n)}(Y)$, and then upper bounded the number of sequences in $A_\epsilon^{(n)}(X,Y)$. For the lower bound, note that *for sufficiently large* n,

$$\begin{aligned}
1 - \epsilon &\leq \sum_{(x^n,y^n) \in A_\epsilon^{(n)}(X,Y)} p(x^n, y^n) \\
&\leq |A_\epsilon^{(n)}(X,Y)| 2^{-n(H(X,Y)-\epsilon)} \\
|A_\epsilon^{(n)}(X,Y)| &\geq (1-\epsilon) 2^{n(H(X,Y)-\epsilon)}.
\end{aligned}$$

Thus,

$$\begin{aligned}\mathbf{P}\left[(\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}(X,Y)\right] &= \sum_{(x^n,y^n)\in A_\epsilon^{(n)}(X,Y)} p(x^n)p(y^n) \\ &\overset{(b)}{\geq} (1-\epsilon)2^{n(H(X,Y)-\epsilon)}2^{-n(H(X)+\epsilon)}2^{-n(H(Y)+\epsilon)} \\ &= 2^{-n(I(X;Y)+3\epsilon)}\end{aligned}$$

where in (b), we separately lower bounded both $p(x^n)$ and $p(y^n)$, and also lower bounded the number of terms in the sum using the newly found bound $|A_\epsilon^{(n)}(X,Y)| \geq (1-\epsilon)2^{n(H(X,Y)-\epsilon)}$.

Appendix F. Ergodicity

(not covered in class)

F.1. **Intuitions.** It's beyond the scope of these notes to flesh out what ergodicity is in full technical detail. But here are some general ideas.

- The basic law of large numbers presented in elementary statistics classes requires the random variables to be i.i.d.. If the random variables are not quite i.i.d., you may still want to draw conclusions similar to the ones you would draw if they actually were i.i.d.. The condition of ergodicity allows you to do this. In this class, the conclusion we would like to draw is the AEP. Although the AEP was proven for the i.i.d. case as an immediate consequence of the WLLN, it turns out that ergodicity also allows you to make an AEP statement even when the variables are not i.i.d..
- Ergodicity means that rather than looking at the statistics of an entire ensemble, you can just look at the behavior of one sample path over a long period of time and determine the general statistics of the ensemble from that alone.

  To expound on this point, many fallacious conclusions can be attributed to assuming that certain ensembles are ergodic when actually they are not. For example, if a single particular Martian mugs me on the street today, robs a bank a year later, burns my house down two years later, embezzles 50 billion dollars three years later, and files for bankruptcy four years later, I might conclude that all Martians are unscrupulous law-breaking hooligans that ought to be thrown in jail. However, such a conclusion relies on the assumption that the ensemble of all Martians is ergodic. That is, observing the behavior of one particular Martian for a very long time will allow me to infer the statistical properties of the entire Martian race as a whole.

F.2. **Definitions.**

- Time average:
  - continuous: $\langle X_t \rangle := \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} x(t)dt$.
  - discrete: $\langle X_k \rangle := \lim_{N\to\infty} \frac{1}{2N+1}\sum_{k=-N}^{N} X_k$
- Ensemble average, at time t:
  - continuous: $\mathbf{E}\left[x(t)\right] = \int x f_t(x)dx$

– discrete: $\mathbf{E}[x_k] = \sum_x x P_{X_k}(x)$

Ergodicity requires that the time average equals the ensemble average.

Note that a time average is generally a random variable, whereas an ensemble average is a deterministic function of time. Thus, if a process is ergodic, we require a deterministic time average, and an ensemble average that is constant over all time.

The notion of stationarity also addresses shift-invariance over time. However, ergodicity is stronger than stationarity.

F.3. **Example of a Process That Is Not Ergodic.** A coin is flipped. If it comes up heads, then $x(t) = 1$ for all time. If it comes up tails, then $x(t) = -1$ for all time.

The time average will be either 1 or -1, depending on the sample path. However, the ensemble average is 0.

F.4. **Example of a Process That Is Ergodic.** $X(t) = \cos(\omega t + \theta)$ where θ is a uniformly distributed phase in $[0, 2\pi)$.

Using calculus, we can show that both the time average and ensemble average are 0.

Appendix G. "Fun" Exercises Using Jensen's Inequality

(a) Use Jensen's Inequality to show that for all $x > 1$,

$$\frac{1}{x-1} + \frac{1}{x} + \frac{1}{x+1} > \frac{3}{x}.$$

(b) Show that for $x, y, z > 0$ and $x + y + z = 1$ then

$$64 \leq (1 + 1/x)(1 + 1/y)(1 + 1/z).$$

Appendix H. The Epsilon Trick for Power Constraints

Recall that in the achievability proof for the Gaussian channel capacity, we generated a random matrix (codebook) where every entry was i.i.d., distributed $\mathcal{N}(0, P - \epsilon)$. But why $P - \epsilon$?

Suppose that we go ahead and live on the edge, setting the variance of our i.i.d. Gaussians to P instead. Let S_n correspond to the amount of power in a randomly generated codeword of length n; that is, $S_n = \frac{1}{n}\sum_{i=1}^{n} X_i^2$. Thus S_n is a sequence of random partial sums. The WLLN then says that $S_n \xrightarrow{P} \mathbf{E}\left[X^2\right] = P$, which means, by definition of convergence in probability[11], the following:

> Fix any $\gamma > 0$ you wish. Then for all $\delta > 0$, there exists an N such that for all $n > N$, $\mathbf{P}[|S_n - P| > \gamma] < \delta$.

[11] See Appendix I for more about probabilistic convergence.

We can equivalently rewrite the last clause of the statement directly above as follows:

$$\begin{aligned}\mathbf{P}\left[|S_n - P| > \gamma\right] &< \delta \\ \mathbf{P}\left[|S_n - P| < \gamma\right] &\geq 1 - \delta \\ \mathbf{P}\left[-\gamma < S_n - P < \gamma\right] &\geq 1 - \delta \\ \mathbf{P}\left[P - \gamma < S_n < P + \gamma\right] &\geq 1 - \delta.\end{aligned}$$

Thus, S_n is sandwiched in the interval $(P-\gamma, P+\gamma)$ with probability that goes to 1 as n tends to infinity, where γ is an arbitrarily small tolerance factor that is independent of n. This means there is a high chance – perhaps a $\frac{1}{2}$ chance – that S_n will actually exceed the power constraint P by lying in the subinterval $(P, P+\gamma)$.

However, we can fix all this by setting the variance of each i.i.d. Gaussian random variable to be $P - \epsilon$ instead, in which case the WLLN says that S_n is sandwiched between $(P - \epsilon - \gamma, P - \epsilon + \gamma)$. Setting $\gamma = 2\epsilon$ then says that S_n is sandwiched in $(P - 3\epsilon, P - \epsilon)$ with arbitrarily high probability, thereby satisfying the power constraint with arbitrarily high probability.

Appendix I. Notions of Probabilistic Convergence

(not covered in class)

Suppose we have a sequence of random variables X_n, and we wish to describe how X_n becomes close to some other random variable X as n tends to infinity. To make clear what closeness might mean, here are four different commonly used notions of convergence from probability theory that we could consider.

(a) in rth mean ($X_n \xrightarrow{L^r} X$):

$$\lim_{n\to\infty} \mathbf{E}\left[|X_n - X|^r\right] = 0$$

(b) almost surely ($X_n \xrightarrow{a.s.} X$):[12]

$$\mathbf{P}\left[\lim_{n\to\infty} X_n = X\right] = \mathbf{P}\left[\{\omega \in \Omega : \lim X_n(\omega) = X(\omega)\}\right] = 1$$

(c) in probability ($X_n \xrightarrow{P} X$):

$$\forall \epsilon > 0 : \quad \lim_{n\to\infty} \mathbf{P}\left[|X_n - X| > \epsilon\right] = 0$$

(d) in distribution ($X_n \xrightarrow{D} X$):[13]

$$\lim_{n\to\infty} \mathbf{P}\left[X_n \leq x\right] = \mathbf{P}\left[X < x\right]$$

More extensive notes written by the author regarding probabilistic convergence and the differences between these various notions are available upon request.

[12]This is also called "with probability 1".

[13]Also goes by the names "weak convergence", and"convergence in law".

Appendix J. Legal Disclaimer

These review session notes were mostly typed over the course of four or five days, admittedly with little sleep. Therefore, it is entirely possible – and in fact highly likely – that they contain some errors. I believe the likelihood of conceptual errors is very low, but the likelihood of stupid typos is probably high. It's just not reasonable for me to promise that there are absolutely none. Even most standard textbooks have lots of errors in them, regardless of whether they have been around for decades.

These notes have been provided as a complimentary service for the students, and are not part of my required duties as a teaching assistant. They should not be put up to the same standard as the required textbook, which was written by top authorities in the field, has been read by many people, and has withstood the tests of time.

In short, I am humbly requesting to not bear any culpability. The reader relinquishes all rights to litigate the author should an error in these notes somehow adversely affect him or her on the final examination or, God forbid, even later in life.

References

[Bat] G. Battail. Tribute to Shannon.

[CTWI06] T.M. Cover, J.A. Thomas, J. Wiley, and W. InterScience. *Elements of Information Theory, 2nd Edition*. Wiley New York, 2006.

[Gal68] R.G. Gallager. *Information Theory and Reliable Communication*. John Wiley & Sons, Inc. New York, NY, USA, 1968.

[Gal01] RG Gallager. Claude E. Shannon: A retrospective on his life, work, and impact. *IEEE Transactions on Information Theory*, 47(7):2681–2695, 2001.

[Gal03] R.G. Gallager. Claude Elwood Shannon, 30 April 1916· 24 February 2001. *Proceedings of the American Philosophical Society*, pages 188–191, 2003.

[Gal05] Robert Gallager. First Kailath Lecture: The Golden Years of Information Theory 1955-1975, 2005.

[GBC+02] S.W. Golomb, E. Berlekamp, T.M. Cover, R.G. Gallager, J.L. Massey, and A.J. Viterbi. Claude Elwood Shannon. *Notices of American Mathematical Society*, 292:8–16, 2002.

[RW02] D. Ramsey and M. Weber. *Claude Shannon Father of the Information Age.* University of California Television San Diego, 2002.

[Sip05] M. Sipser. *Introduction to the theory of Computation, 2nd Edition*. Course Technology, 2005.