Information Theory
lecture notes, Fall 2018

First lecture (September 4, 2018)

We considered some "warming up" problems:

1. How many yes-no questions are needed (in the worst case, but for the best strategy) to determine one out of twenty numbers?

Answer: It is $\lceil \log_2 20 \rceil = 5$.

That many may be needed because each question partitions the set of possibilities into two subsets and if the answer always leaves us with the larger partition class then we will reduce the number of possibilities to 1 only after 5 questions.

That many is enough, because if we always ask questions that partition the set of still possible numbers into two subsets whose respective sizes are as close to each other as possible (i.e., they differ by at most 1), then after 5 answers we will have only one option remaining.

This argument shows that with 5 questions we can find one of at most 32 numbers with similar rules. In general, if we have n options, then $\lceil \log_2 n \rceil$ questions are needed.

2. How many questions do we need if we must ask all of them together, without knowing the answer to previous ones?

Answer: Somewhat surprisingly, the answer is the same. The situation is more restricted now, so we need at least as many questions as before. The point is that we do not need more. Here is an optimal strategy: write all the numbers (up to 20 or 32 or n, in general) into binary form and let the ith question be whether the ith binary digit is 0 (or 1). Clearly, this gives $\lceil \log_2 n \rceil$ questions and knowing the answer to all of them identifies the number we look for.

3. We have 27 identical looking coins, one of them lighter than the others. We should find it with a bascule that can compare the weight of two disjoint set of coins. How many measurements we need?

Answer: 3. Now each measurement has 3 possible outcomes, so $\log_3 27 = 3$ is a lower bound. It is not hard to plan a strategy that solves the problem with 3 measurements. But as before, we can do it also without using the result of previous measurements, that is by planning all measurements in advance. Here it is how: Label each coin with a 3-length ternary sequence from 000 to 222, there are exactly 27 such sequences. Let the ith measurement ($i = 1, 2, 3$) be this: put on one side of the bascule those coins whose label has a 0 as its ith digit and those with a 1 at the ith digit on the other side. Then we have $9 - 9$ coins on both sides and also 9 that are not in the balance. If the first side is lighter in the ith measurement, then the ith digit of the label of the lighter coin is 0, if the other side is lighter, then it is a 1, if they are equal, it is a 2. Thus after learning the three results, we will know the full label of the lighter coin.

4. How many pairwise comparisons are needed to find the order of 4 different objects. (Let us say, they are ordered according to their size, and any two of them have different size.)

Answer: One can easily find a strategy with 5 comparisons. The point is how we know we really need that many. Here is a proof. We look for one out of $4! = 24$ possible orders. Each comparison partitions the set of possible orders into 2, so if the outcome of a comparison is that the seeked order is in the larger (non-smaller) set of the two, then we need at least $\lceil \log_2 24 \rceil = 5$ comparisons.

We will measure information content by the number of binary digits ("bits") needed to "describe" the information. Thus the information contained in telling one out of 32 numbers is 5 bits.

We feel that the above is plausible only if the probability of the possibilities is close to each other. E.g., learning the fact that one did not hit the jackpot in the lottery game this week seems much less information than learning that the same person actually did hit the jackpot. So we will have to take probabilities also into account. A good guess seems to be if we measure the information content of an event of probability p by $\log_2 \frac{1}{p}$. It will indeed turn out that the expected value $\sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}$ of this quantity, called the *entropy* of the probabilility distribution $P = (p_1, \dots, p_n)$ plays a major role in information theory and can be interpreted as the information content of a random variable that has p as its distribution.

Historical remark: Information theory is a branch of science that has a surprisingly clear starting point: it is the publication of a two-part article by Claude Elwood Shannon in 1948 entitled "A Mathematical Theory of Communication". Many basic results are already contained in this article. Naturally, many other important results were obtained during the 70 years that passed since then, and information theory today is considered a flourishing field of both mathematics and electrical engineering.

Two main problems in information theory are:
Source coding
Channel coding

Goal of source coding: compressing data, that is encoding data with reduced redundancy.
Goal of channel coding: safe data transmission, that is encoding messages so that one can still correctly decode them after transmission in spite of channel noise. (This is achieved by increasing redundancy in some clever way.)

Variable length source coding

Notation: For a finite set V, the set of all finite length sequences of elements of V will be denoted by V^*.

Model: Source emits sequence of random symbols that are elements of the *source alphabet* $\mathcal{X} = \{x_1, \dots, x_r\}$.

Given code alphabet $\mathcal{Y} = \{y_1, \dots, y_s\}$ (with s elements) we seek for an encoding function $f : \mathcal{X} \to \mathcal{Y}^*$ which efficiently encodes the source.
meaning of "efficient": it uses as short sequences of y_i's as possible, while the original x_j will always be possible to be reproduced correctly.
meaning of "short": The average length of codewords should be small. The average is calculated according to the probability distribution characterizing the source: We assume that the emitted symbol is a random variable X and in the ideal situation we know the distribution of X that governs the behavior of the source.

Def. A uniquely decodable (UD) code is a function $f : \mathcal{X} \to \mathcal{Y}^*$ satisfying that $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathcal{X}^*, \boldsymbol{u} = u_1 u_2 \dots u_k, \boldsymbol{v} = v_1 v_2 \dots v_m, \boldsymbol{u} \neq \boldsymbol{v}$ implies $f(u_1)f(u_2)\dots f(u_k) \neq f(v_1)f(v_2)\dots f(v_m)$ (where $f(a)f(b)$ means the sequence obtained by concatenating the sequences $f(a)$ and $f(b)$).

Prefix code: No codeword $f(x_i)$ is a prefix of another. A prefix code is always UD.

Examples: (Codes given here with collection of codewords.) $C_1 = (0, 10, 110, 111)$ is UD, even prefix. $C_2 = (0, 10, 100, 101)$ is not prefix, not even UD, 100 can be $f(x_2)f(x_1)$ as well as $f(x_3)$. But $C_3 = (0, 01)$ is UD, although not prefix.

Question: Why do we care about variable length and not simply use $|\mathcal{X}|$ codewords of length $\lceil \log_s |\mathcal{X}| \rceil$ each?
Answer: Average length may be better, see this example. Let the probabilities of emitting the symbols be $p(x_1) = 1/2, p(x_2) = 1/4, p(x_3) = 1/8, p(x_4) = 1/8$. The code $f(x_1) = 0, f(x_2) = 10, f(x_3) = 110, f(x_4) = 111$ has average length $1 \cdot 1/2 + 2 \cdot 1/4 + 3 \cdot 1/8 + 3 \cdot 1/8 = 1.75 < 2 = \log_2 4$.

Next we state two basic theorems the proof of which will be given next week.

Kraft-McMillan inequality

Theorem 1 *(McMillan): If $C = (f(x_1), \dots, f(x_r))$ is a UD code over an s-ary alphabet, then*

$$\sum_{i=1}^{r} s^{-|f(x_i)|} \leq 1.$$

Theorem 2 *(Kraft): If the positive integers $l_1, \dots, l_r$ satisfy*

$$\sum_{i=1}^{r} s^{-l_i} \leq 1.$$

then there exists an s-ary prefix code with codeword lengths $l_1, \dots, l_r$.

Second lecture (September 11, 2018)

Proof of McMillan's theorem. Consider

$$\left(\sum_{i=1}^{r} s^{-|f(x_i)|} \right)^k = \sum_{\mathbf{v} \in C^k} s^{-|\mathbf{v}|} = \sum_{l=1}^{k \cdot l_{\max}} A_l s^{-l},$$

where A_l is the number of ways we can have an l length string of code symbols when using our code and $l_{\max}$ is the length of the longest codeword $f(x_i)$. Since the code is UD, we cannot have more than s^l different source strings resulting in such an l length string, so $A_l \leq s^l$. Thus the right hand side is at most $k \cdot l_{\max}$ giving $(\sum_{i=1}^{r} s^{-|f(x_i)|})^k \leq k \cdot l_{\max}$. Taking kth root and limit as $k \to \infty$, the result follows. □

Proof of Kraft's theorem. Arrange the lengths in nondecreasing order, i.e., $l_1 \leq \dots \leq l_r$. Define the numbers $w_1 := 0$ and for $j > 1$ let

$$w_j := \sum_{i=1}^{j-1} s^{l_j - l_i}.$$

This gives $w_j = s^{l_j} \sum_{i=1}^{j-1} s^{-l_i} < s^{l_j} \sum_{i=1}^{j} s^{-l_i} \leq s^{l_j}$, thus the s-ary form of w_j has at most l_j digits. Let $f(x_j)$ be the s-ary form of w_j "padded" with 0's at the beginning if necessary to make it have length exactly l_j for every j. This gives a code, we show it is prefix. Assume some $f(x_j)$ is just the continuation of another $f(x_h)$. (Then $l_j > l_h$, so $j > h$.) Thus cutting the last $l_j - l_h$ digits of $f(x_j)$ we get $f(x_h)$. This "cutting" belongs to division by $s^{l_j - l_h}$ (plus taking integer part), so this would mean $w_h = \left\lfloor \frac{w_j}{s^{l_j - l_h}} \right\rfloor = \left\lfloor s^{l_h} \sum_{i=1}^{j-1} s^{-l_i} \right\rfloor = s^{l_h} \sum_{i=1}^{h-1} s^{-l_i} + \left\lfloor s^{l_h} \sum_{i=h}^{j-1} s^{-l_i} \right\rfloor \geq w_h + 1$, a contradiction. □

Convention: When no basis for a logarithm is given, we mean it to be of base 2.

Idea: Kraft's theorem implies that there is a prefix code with codeword lengths $\left\lceil \log_s \frac{1}{p_1} \right\rceil, \dots, \left\lceil \log_s \frac{1}{p_m} \right\rceil$, since

$$1 = \sum_{i=1}^{m} p_i = \sum_{i=1}^{m} s^{\log_s p_i} = \sum_{i=1}^{m} s^{-\log_s(1/p_i)} \geq \sum_{i=1}^{m} s^{-\lceil \log_s(1/p_i) \rceil}.$$

Such a code has average length

$$\sum_{i=1}^{m} p_i \left\lceil \log_s \frac{1}{p_i} \right\rceil < \sum_{i=1}^{m} p_i (\log_s \frac{1}{p_i} + 1) \leq \sum_{i=1}^{m} p_i \log_s \frac{1}{p_i} + \sum p_i = \sum_{i=1}^{m} p_i \log_s \frac{1}{p_i} + 1.$$

The sum $\sum_{i=1}^{m} p_i \log_s \frac{1}{p_i}$ is an important quantity in information theory. Usually we define it with $s = 2$.

Def. The *entropy* $H(P)$ of the probability distribution $P = (p_1, \dots, p_r)$ is defined as

$$H(P) = -\sum_{i=1}^{r} p_i \log p_i.$$

For $r = 2$ we speak about the *binary entropy function* of the distribution $P = (p, 1-p)$ and denote it by $h(p)$. Thus $h(p) = -p \log p - (1-p) \log(1-p)$.

The quantity

$$H_s(P) := \sum_{i=1}^{m} p_i \log_s \frac{1}{p_i} = \frac{1}{\log s} H(P)$$

is sometimes called the s-ary or base s entropy of P. (Note however, that the binary entropy function mentioned above is not the $s = 2$ case of this. Rather the $s = 2$ case of the s-ary entropy we simply call entropy in accordance with the convention that logarithms are to the base 2 if not said otherwise. If we really want to emphasize that $s = 2$ we might say base-2 entropy.)

Thus above we have proved the following.

Theorem 3 *Let us have an information source emitting symbol $x_i \in \mathcal{X}$ with probability $p(x_i) = p_i, (i = 1, \dots, r)$. There exists an s-ary prefix code for this source with average codeword length less than $\frac{H(P)}{\log s} + 1$.*

Remark: The entropy function $H(P)$ is often interpreted as a measure of the information content in a random variable X that has distribution P. Intuitively, one can think about $\log \frac{1}{p_i} = -\log p_i$ as the information gained when observing that X just obtained its value having probability p_i. This interpretation would then mean that the average information per observation obtained during several observations is just $H(P)$. We think that the information content is measured in bits (binary digits) thus it has to do with the number of binary digits needed for an optimal encoding. Theorem 3 together with Theorem 4 below gives justification for this interpretation.

Theorem 4 *Let us have an information source emitting symbol $x_i \in \mathcal{X}$ with probability $p(x_i) = p_i, (i = 1, \dots, r)$. For any s-ary UD code $f : \mathcal{X} \to \mathcal{Y}^*$ of this source we have*

$$\sum_{i=1}^{r} p_i |f(x_i)| \geq \frac{1}{\log s} H(P) = \frac{1}{\log s}\left(-\sum_{i=1}^{r} p_i \log p_i\right) = -\sum_{i=1}^{r} p_i \log_s p_i,$$

where P stands for the distribution $(p_1, \dots, p_r)$. Thus, for a binary (this belongs to $s = 2$) UD code the average codeword length is bounded from below by the entropy of the distribution governing the system.

For the proof we will need the following simple tool from calculus that is often very useful when proving theorems in information theory. Recall the notion of convexity of a function first.

Def.: A function $g : [a, b] \to R$ is *convex* if for every $x, y \in [a, b]$ and $\lambda \in [0, 1]$ we have

$$g(\lambda x + (1 - \lambda)y) \leq \lambda g(x) + (1 - \lambda)g(y).$$

We say that g is *strictly convex* if we have strict inequality whenever $0 < \lambda < 1$ and $x \neq y$.

Jensen's inequality: *Let $g : [a, b] \to R$ be a convex function. Then for any $x_1, \dots, x_k \in [a, b]$ and non-negative reals $\alpha_1, \dots, \alpha_k$ satisfying $\sum_{i=1}^{k} \alpha_i = 1$, we have*

$$g\left(\sum_{i=1}^{k} \alpha_i x_i\right) \leq \sum_{i=1}^{k} \alpha_i g(x_i).$$

Moreover, if g is strictly convex, then equality holds if and only if all x_i's belonging to non-zero coefficients α_i are equal.

Below we present the simple induction proof of Jensen's inequality

Proof of Jensen's inequality: We argue by induction on k. The base case is $k = 2$, when the statement simply follows from the definition of convexity. (It is also obviously true for $k = 1$, in fact then the statement is simply trivial, stating that $g(x_1) \leq g(x_1)$. But we do need the $k = 2$ case in the proof, so it would not be enough to consider $k = 1$ alone as the base case.)

Now assume the statement is true for all $k < \ell$, we now prove it for $k = \ell$. We can write

$$g\left(\sum_{i=1}^{\ell} \alpha_i x_i\right) = g\left(\alpha_\ell x_\ell + \sum_{i=1}^{\ell-1} \alpha_i x_i\right) =$$

$$g\left(\alpha_\ell x_\ell + (1 - \alpha_\ell)\sum_{i=1}^{\ell-1} \frac{\alpha_i}{1 - \alpha_\ell} x_i\right) \leq \alpha_\ell g(x_\ell) + (1 - \alpha_\ell) g\left(\sum_{i=1}^{\ell-1} \frac{\alpha_i}{1 - \alpha_\ell} x_i\right) \leq$$

$$\alpha_\ell g(x_\ell) + (1 - \alpha_\ell)\sum_{i=1}^{\ell-1} \frac{\alpha_i}{1 - \alpha_\ell} g(x_i) =$$

$$\alpha_\ell g(x_\ell) + \sum_{i=1}^{\ell-1} \alpha_i g(x_i) = \sum_{i=1}^{\ell} \alpha_i g(x_i)$$

as claimed. Here the first inequality follows by the definition of convexity, while the second from the induction hypothesis applied to $k = \ell - 1$. The statement for the conditions of equality also follows: to have equality in the first inequality we need either that $\alpha_\ell = 0$ or that $\alpha_\ell = 1$, or if neither holds then $x_\ell = \sum_{i=1}^{\ell-1} \frac{\alpha_i}{1-\alpha_\ell} x_i$. To have equality also in the second inequality we know from the induction hypothesis that we need that all x_i's with $i \leq \ell - 1$ belonging to nonzero coefficients α_i are equal. Then these are equal also to their weighted average value $\sum_{i=1}^{\ell-1} \frac{\alpha_i}{1-\alpha_\ell} x_i$, so if α_ℓ is neither 0 nor 1, then x_ℓ must also be equal to this common value. □

Corollary 1 *If $P = (p_1, \ldots, p_k)$ and $Q = (q_1, \ldots, q_k)$ are two probability distributions, then*

$$\sum_{i=1}^{k} p_i \log \frac{p_i}{q_i} \geq 0,$$

and equality holds iff $p_i = q_i$ for every i.

Convention: To make the formulas above always meaningful, we use the "calculation rules" (for $a \geq 0, b > 0$) $0 \log \frac{0}{a} = 0 \log \frac{a}{0} = 0$ and $b \log \frac{b}{0} = +\infty, b \log \frac{0}{b} = -\infty$.

Proof. The function $-\log x$ is convex, thus by Jensen's inequality

$$\sum_{i=1}^{k} p_i \log \frac{p_i}{q_i} = \sum_{i=1}^{k} p_i \left(-\log \frac{q_i}{p_i} \right) \geq -\log \left(\sum_{i=1}^{k} p_i \frac{q_i}{p_i} \right) = -\log \left(\sum_{i=1}^{k} q_i \right) = 0.$$

The condition of equality also follows from the corresponding condition in Jensen's inequality. □

Proof of Theorem 4. We know from the McMillan theorem, that $\sum_{i=1}^{r} s^{-|f(x_i)|} \leq 1$. Set $b = \sum_{i=1}^{r} s^{-|f(x_i)|}$ and $q_i = \frac{s^{-|f(x)|}}{b} \geq s^{-|f(x)|}$. Then

$$\sum_{i=1}^{r} p_i |f(x_i)| = -\sum_{i=1}^{r} p_i \log_s (q_i b) \geq -\sum_{i=1}^{r} p_i \log_s q_i = -\frac{1}{\log s} \sum_{i=1}^{r} p_i \log q_i.$$

Observe that $\sum_{i=1}^{r} q_i = 1$ and $q_i \geq 0$ for every i (so $(q_1, \ldots, q_r)$ could be considered a probability distribution). Thus by Corollary 1 of Jensen's inequality above, we have that $-\sum_{i=1}^{r} p_i \log q_i \geq -\sum_{i=1}^{r} p_i \log p_i$ and the statement follows. □

If we encode m-length sequences of x_i's in place of only one x_i at a time, then the same upper estimation still has only a plus 1 over $H(\mathbf{X})$. If the source outputs are independent (the source is memoryless) and identically distributed, then $H(\mathbf{X}) = mH(X_1)$ (for $\mathbf{X}$ being the random variable belonging to an m length sequence of source outputs), so relative to one output the overhead in the coding is only $1/m$, which clearly tends to 0 as m goes to infinity.

Third lecture (September 18, 2018)

We give a second proof of Theorem 3 to introduce another code construction, called the **Shannon-Fano code**:

We assume $p_1 \geq p_2 \geq \ldots \geq p_n > 0$. Let $w_0 = 0$ and $w_j = \sum_{i=1}^{j-1} p_i$. Let the codeword $f(x_i)$ be the s-ary representation of the number w_j (which is always in the $[0, 1)$ interval) without the starting integer part digit 0, and with minimal such length that it is not a prefix of any other such codeword. The latter condition already ensures that the code is prefix.

This definition implies that the first $|f(x_j)| - 1$ digits of $f(x_j)$ is a prefix of another codeword and thus it must be the prefix of a codeword coming from a closest number w_h, thus w_{j-1} or w_{j+1}. This implies

$$p_j = p(x_j) = w_{j+1} - w_j \leq s^{-(|f(x_j)|-1)}$$

or

$$p_{j-1} = p(x_{j-1}) = w_j - w_{j-1} \leq s^{-(|f(x_j)|-1)}.$$

By $p_{j-1} \geq p_j$ in either case the first of the above two inequalities holds. Thus $\log_s p_j \leq -|f(x_j)| + 1$ implying

$$-p_j \log_s p_j \geq p_j(|f(x_j)| - 1),$$

and thus

$$-\sum_{j=1}^{r} p_j \log_s p_j + 1 \geq \sum_{j=1}^{r} p_j |f(x_j)|.$$

□

We have seen two constructions giving average codeword length close to the lower bound $H_s(P)$, but nothing guaranteed that any of these codes would be best possible. So the question of how to find an optimal average length code comes up. This is answered by constructing the so-called Huffman code. We study this only for the binaary case, i.e, when the size of the code alphabet is $s = 2$.

Huffman code

Assume $p_1 \geq \ldots \geq p_r > 0$, $p_i = p(x_i)$ and having an optimal binary code $C = (f(x_1), \ldots, f(x_r))$, $l_i := |f(x_i)|$. By the foregoing we can assume that the code is prefix. (Note that the $p_r > 0$ assumption is not a real restriction: if we have 0-probability events, they need not be encoded. Or they could even be encoded into long codewords, since their contribution to the average length will be zero anyway.)

Observe:
We may also assume

(1) $l_1 \leq l_2 \leq \ldots \leq l_r$. This is true, because if this is not satisfied, then we may exchange codewords without increasing the average length.

(2) $l_n = l_{n-1}$ and the two codewords $f(x_n)$ and $f(x_{n-1})$ differ only in the last digit. The first statement is true, because if not, then $l_n > l_{n-1}$ by (1) above and since the code is prefix, deleting the last digit of $f(x_n)$ would result in a prefix code with smaller average length, so the original code was not optimal. The second statement is true, because exchanging the last digit of the codeword $f(x_n)$ we should get another codeword (otherwise this last digit could have been deleted without ruining the prefix property), and if this codeword is not $f(x_{n-1})$ but some $f(x_i)$ with $i \neq n-1$, then we can simply exchange the two without effecting the average length as these two codewords both have the same length $|f(x_i)| = |f(x_n)| = |f(x_{n-1})|$.

(3) Cutting the last digit of the two codewords $f(x_{n-1})$ and $f(x_n)$ we obtain an optimal binary prefix code for the distribution $(p_1, p_2, \ldots, p_{n-2}, p_{n-1} + p_n)$. This is true because the average length L of our code is $L' + p_{n-1} + p_n$, where L' is the average length of the code obtained by identifying the codewords $f(x_{n-1})$ and $f(x_n)$ by cutting their last digit. If there was a better (i.e., one with smaller average length) prefix code for the distribution $(p_1, p_2, \ldots, p_{n-2}, p_{n-1}+p_n)$, then extending the codeword belonging to the probability $p_{n-1} + p_n$ source symbol once with a 0 digit and once with a 1 digit, we would obtain a better code than our original one, so its average length could have not been optimal.

From these three observations the optimal code construction is immediate: add two smallest probabilities iteratively until only two distinct ones remain. Give these the (sub)words 0 and 1 and then follow the previous "adding up two probabilities" process backwards and put a 0 and a 1 at the end of the corresponding codeword.

Example:

$$P = (0.05, 0.10, 0.10, 0.11, 0.12.0.13, 0.14, 0.25)$$

. The in-between distributions:
$(0.10, 0.11, 0.12, 0.13, 0.14, 0.15 = 0.10 + 0.05, 0.25)$;
$(0.12, 0.13, 0.14, 015, 0.21, 0.25)$; $(0.14, 0.15, 021, 0.25, 0.25)$;
$(0.21, 0.25, 0.25, 0.29)$; $(0.25, 0.29, 0.46)$; $(0.46, 0.54)$.

And the code obtained(writing it backwards for each stage of the construction:
$(0,1); (0,10,11); (00,01,10,11);$
$(110,111,00,01,10); (010,011,110,111,00,10);$
$(000,001,010,011,110,111,10)$, and finally

$$(1110,1111,000,001,010,011,100,10).$$

Exercises:

1. We have two dice with 1 dot on two faces, 2 dots on two faces, and 3 dots on two faces. We roll the two dice together and want to encode the total number of dots we see on the rolled faces of the two dice. Give the Shannon-Fano code for alphabet size 2 and also for alphabet size 3 for this problem. Construct also a binary code that has shortest average length, that is one, for which the expected number of bits needed to encode the result of many rolls is as small as possible.

Solution: The result can be $2,3,4,5$, or 6 dots on the two faces seen, and their probabilities can be calculated by the number of elementary events giving the corresponding number. So the probabilities are $1/9, 2/9, 3/9, 2/9, 1/9$, respectively. The corresponding w_i values are $0, 3/9, 5/9, 7/9, 8/9$ in the Shannon-Fano code construction. The binary Shannon-Fano code we obtain from these values is $(00,01,10,110,111)$, while the ternary Shannon-Fano code is $(0,10,12,21,22)$. To obtain shortest average length we have to construct a Huffman code for the above distribution. Doing this we can get $000,001,01,10,11$. (This shows that the binary Shannon-Fano code also has optimal average length in this case.) $\diamond$

The following two exercises remained homework problems. We will discuss their solution on the next class meeting.

2. Two people made two different Huffman codes for the distribution $p_1 \geq p_2 \geq p_3 \geq p_4$. The codewords of these codes are $0,10,110,111$ for one and $00,01,10,11$ for the other. Determine the distribution if we know that $p_3 = 1/6$.

3. Let $p_1 \geq p_2 \geq p_3 \geq p_4$ be a probability distribution. Assume that the code consisting of the four 2-length binary sequences gives optimal average length for this distribution. What is the maximum possible value of p_1?

(The solution of Exercise 2 and 3 can be seen at the end of the next lecture.)

Fourth lecture (September 25, 2018)

More on the entropy function

Notation:
$p(x) = Prob(X = x)$,
$p(y) = Prob(Y = y)$,
$p(x, y) = Prob(X = x, Y = y)$,
$p(x|y) = Prob(X = x|Y = y)$,
$p(y|x) = Prob(Y = y|X = x)$.

$$H(X, Y) = -\sum_{x,y} p(x, y) \log p(x, y)$$

is simply the entropy of the joint distribution of the variable (X, Y).

Conditional entropy is defined as:

$$H(X|Y) = \sum_{y} p(y) H(X|Y = y) =$$
$$= -\sum_{y} p(y) \sum_{x} p(x|y) \log p(x|y) =$$
$$= -\sum_{x,y} p(x, y) \log p(x|y).$$

Theorem 5 *a)*
$$0 \leq H(X) \leq \log n,$$
where $n = |\mathcal{X}|$. $H(X) = 0$ *iff* X *takes a fix value with probability* 1, $H(X) = \log n$ *iff* $p(x)$ *is uniform.*

b)
$$H(X, Y) \leq H(X) + H(Y)$$
with equality iff X *and* Y *are independent.*

Note the intuitive plausibility of these statements. (For example: The information content of the pair (X, Y) is not more than the sum of the information X and Y contain separately. And equality means that they "do not contain information about each other", that is, they are independent.)

Proof of a). $0 \leq H(X)$ is clear by $\log p(x) \leq 0$ for all x. Equality can occur iff $p(x) = 1$ for some x, then all other probabilities should be zero.

Applying Corollary 1 to $q_i = 1/n$ $\forall i$ gives $H(X) \leq \log n$ and also the condition for equality.

Proof of b). Follows by applying Corollary 1 for $p = p(x, y)$ and $q = p(x)p(y)$. In details:

$$H(X) + H(Y) - H(X, Y) =$$
$$-\sum_{x} \left(\sum_{y} p(x, y)\right) \log p(x) - \sum_{y} \left(\sum_{x} p(x, y)\right) \log p(y) + \sum_{x,y} p(x, y) \log p(x, y) =$$

$$\sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \geq 0.$$

Equality holds iff $p(x,y) = p(x)p(y) \forall x,y$, i.e. iff X and Y are independent. □

Some properties of conditional entropy are proven next.

Theorem 6 *a)*

$$H(X|Y) = H(X,Y) - H(Y).$$

b)

$$0 \leq H(X|Y) \leq H(X).$$

c) (Chain rule)

$$H(X_1,\ldots,X_n) = H(X_1)+H(X_2|X_1)+H(X_3|X_1,X_2)+\ldots+H(X_n|X_1,\ldots,X_{n-1}).$$

Proof of a). $H(X|Y) = -\sum_{x,y} p(x,y)\log p(x|y) = -\sum_{x,y} p(x,y)\log \frac{p(x,y)}{p(y)} = -\sum_{x,y} p(x,y)\log p(x,y) + \sum_{x,y} p(x,y)\log p(y) = H(X,Y) + \sum_y p(y)\log p(y) = H(X,Y) - H(Y)$.

Proof of b). $0 \leq H(X|Y)$ follows from observing that $H(X|Y)$ is the expected value of entropies that are non-negative by $0 \leq H(X)$ being valid in general. $H(X|Y) \leq H(X)$ follows from a) as it is equivalent to $H(X,Y) = H(X|Y) + H(Y)$, while we have already seen that $H(X,Y) \leq H(X) + H(Y)$. This also gives that the condition of equality is exactly the same as it is in Theorem 5 b), namely that X and Y are independent.

Proof of c). Goes by induction. Clear for $n = 1$, and it is just a) for $n = 2$. Having it for $n-1$, apply a) for $Y = X_n$ and $X = (X_1,\ldots,X_{n-1})$ in the form $H(X,Y) = H(X) + H(Y|X)$. It gives

$$H(X_1,\ldots,X_n) = H((X_1,\ldots,X_{n-1}),X_n) =$$

$$H(X_1,\ldots,X_{n-1}) + H(X_n|(X_1,\ldots,X_{n-1})) =$$

$$H(X_1) + H(X_2|X_1) + \ldots + H(X_{n-1}|X_1,\ldots,X_{n-2}) + H(X_n|X_1,\ldots,X_{n-1}).$$

□

Below are the solutions of the two problems that remained homeworks last time and we discussed the solution on this class meeting. The problem formulations are repeated here for the reader's convenience.

2. Two people made two different Huffman codes for the distribution $p_1 \geq p_2 \geq p_3 \geq p_4$. The codewords of these codes are $0, 10, 110, 111$ for one and $00, 01, 10, 11$ for the other. Determine the distribution if we know that $p_3 = 1/6$.

Solution of Exercise 2: When constructing the code, in the first step both people had to create the distribution (p_1, p_2, p_3+p_4). In the next step, however, one of them had to create (p_1+p_2, p_3+p_4), while the other created $(p_1, p_2+p_3+p_4)$. If both led to Huffman codes, that means both steps are optimal, so it did not matter whether we add p_1 or p_3+p_4 to p_2. This implies $p_1 = p_3+p_4$. All the probability values can be obtained from this as follows.
Since $p_4 \leq p_3 = 1/6$, we have $p_1 = p_3 + p_4 \leq 1/6 + 1/6 = 1/3$. On the other hand, $p_1 \geq \frac{1}{2}(p_1+p_2) = \frac{1}{2}(1-(p_3+p_4) \geq \frac{1}{2}(1-1/3) = 1/3$. So p_1 is not more and not less than $1/3$, thus $p_1 = 1/3$. Thus $p_4 = p_1 - p_3 = 1/3 - 1/6 = 1/6$ and $p_2 = 1 - 1/3 - 1/6 - 1/6 = 1/3$. So the distribution is $(1/3, 1/3, 1/6, 1/6)$. Note that the above two codes give indeed the same optimal average length 2 for this distribution. ◇

3. Let $p_1 \geq p_2 \geq p_3 \geq p_4$ be a probability distribution. Assume that the code consisting of the four 2-length binary sequences gives optimal average length for this distribution. What is the maximum possible value of p_1?

Solution of Exercise 3: To have optimal average length we have to construct a Huffman code. The condition means that in the second step of Huffman encoding we will create the distribution (p_1+p_2, p_3+p_4). In particular, $p_3+p_4 \geq p_1$, otherwise we would not add p_1 and p_2, but would add p_2 and p_3+p_4. Note that it does not matter how the total probability p_3+p_4 is divided between p_3 and p_4, so we may assume $p_3 = p_4$. Thus $p_1 \leq p_3 + p_4 = 2p_4$, from which we have $p_4 \geq \frac{p_1}{2}$. On the other hand, $p_1 = 1 - (p_2 + p_3 + p_4) \leq 1 - 3p_4$, so $1 \geq p_1 + 3p_4 \geq p_1 + \frac{3}{2}p_1$, which implies $p_1 \leq \frac{2}{5}$. Thus p_1 cannot be larger than $\frac{2}{5}$. Observe that p_1 can be that large, since we can consider the distribution, for which $p_1 = \frac{2}{5}, p_2 = p_3 = p_4 = \frac{1}{5}$ and the said code is indeed optimal for this distribution. $\diamond$

Exercises:

4. Let X and Z be independent random variables such that $Prob(X = 1) = p$, $Prob(X = 0) = 1 - p$ and $Prob(Z = 0) = Prob(Z = 1) = 1/2$. Let Y be the random variable that is given as the modulo 2 sum of X and Z. Calculate $H(X)$, $H(X|Z)$, $H(X|Y)$, $H(X|, Y, Z)$.

We have $H(X) = h(p)$ (directly follows), $H(X|Z) = h(p)$ (immediate by the independence of X and Z), $H(X|Y) = h(p)$ (follows by realizing that $H(X|Y = 0) = H(X|Y = 1) = h(p)$), and $H(X|Y, Z) = 0$ (since X is determined by the pair Y, Z). It is worth noticing that $H(X|Y) = h(p) = H(X)$ means that X and Y are also independent.

The following exercise is for homework:

5. Solve the question of Exercise 3 in the following more general setting.
We have $n = 2^k$ for some positive integer k. What is the maximum possible value of p_1 if $(p_1, p_2, \ldots, p_n)$ is a probability distribution satisfying $p_1 \geq p_2 \geq \ldots \geq p_n$ and the property that the 2^k binary sequences of length k provide an optimal encoding of the source emitting symbol i with probability $p_i (i = 1, \ldots, n)$?

Fifth lecture (October 2, 2018)

Recall the theorem we proved last time stating

$$0 \leq H(X|Y) = H(X, Y) - H(Y).$$

We prove a consequence of this next.

Corollary 2 *For any function $g(X)$ of a random variable X we have*

$$H(g(X)) \leq H(X).$$

Proof. Since $g(X)$ is determined by X we have $H(g(X)|X) = 0$. Thus using Theorem 6 a) we can write

$$H(X) = H(X)+H(g(X)|X) = H(X, g(X)) = H(g(X))+H(X|g(X)) \geq H(g(X)).$$

We also see that the condition of equality is $H(X|g(X)) = 0$, which is equivalent to $g(X)$ determining X, i.e. to g being invertible. $\square$

From the above we see that

$$H(X) - H(X|Y) = H(X) + H(Y) - H(X, Y) = H(Y) - H(Y|X).$$

This quantity is thus intuitively the difference of the amount of information X contains if we do and if we do not know Y. We can think about it as the amount of information Y carries about X. And we see that we get the same value if we exchange the role of X and Y. This interpretation is also consistent with the fact that the above value is 0 if and only if X and Y are independent. These thoughts motivate the following definition.

Def. For two random variables X and Y, their *mutual information* $I(X,Y)$ is defined as

$$I(X,Y) = H(X) + H(Y) - H(X,Y).$$

By the foregoing we also have $I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$.

Later we will see that mutual information is a basic quantity that also comes up as a central value in certain coding theorems

Def.: A source $X_1, X_2, \ldots$ is memoryless if the X_i's are independent.

Def.: A source is stationary if $X_1, \ldots, X_k$ and $X_{n+1}, \ldots, X_{n+k}$ has the same distribution for every n and k.

If the source is stationary and memoryless, then $H(X_1, X_2, \ldots, X_k) = kH(X_1)$.

Let $f : \mathcal{X}^k \to \mathcal{Y}^*$ be a UD code of the stationary and memoryless source $X = X_1, X_2, \ldots$.

We know there exists a (e.g. Shannon-Fano) code satisfying

$$L = E(f(X_1, \ldots, X_k)) \leq \frac{H(X_1, \ldots, X_k)}{\log s} + 1.$$

(Here $E(.)$ means expected value.) Then the *per letter* average length of this code is $\frac{1}{k}L \leq \frac{1}{k}\left(\frac{H(X_1,\ldots,X_k)}{\log s} + 1\right) = \frac{H(X_1)}{\log s} + \frac{1}{k}$. So $\frac{1}{k}L$ can get arbitrarily close to the lower bound $H_s(X_1) = \frac{H(X_1)}{\log s}$. (This was already hinted at the end of the second lecture.)

Ezercise 6. (Taken from the book: Thomas Cover and Joy Thomas *Elements of Information Theory*, Second Edition, Wiley, 2006.)

The World Series is a seven-game series that terminates as soon as either team wins four games. Let X be the random variable that represents the outcome of a World Series between teams A and B; examples of possible values of X are $AAAA, BABABAB$, and $BBBAAAA$. Let Y be the number of games played, which ranges from 4 to 7. Assuming that A and B are equally matched (that is, both has an 50% chance to win at each game) and that the games are independent, calculate $H(X)$, $H(Y)$, $H(Y|X)$, and $H(X|Y)$. Calculate also $I(X,Y)$.

Solution: Since X determines Y, we can immediately write $H(Y|X) = 0$. Also, since $H(X|Y) + H(Y) = H(Y|X) + H(X) = H(X)$, we will be able to calculate $H(X|Y)$ as the differenec between $H(X)$ and $H(Y)$. Furthermore, $I(X,Y) = H(Y) - H(Y|X) = H(Y)$. So it will be enough to calculate $H(X)$ and $H(Y)$. The value of Y can be 4 in two different ways ($AAAA$ and $BBBB$), both with probability $\frac{1}{2^4}$, so Prob$(Y = 4) = 1/8$. Y can be 5 in 8 different ways: there are 4 ways to have one game won by B among the first four games the fifth of which is won by A, and there are four other options when we change the role of A and B. Each of these options have probability $\frac{1}{2^5}$, thus Prob$(Y = 5) = 1/4$.

There are $\binom{5}{2} = 10$ ways to have exactly two games won by B among the first five while the sixth is won by A and we have 10 more options to have six games changing the role of A and B. Each of these have probability $\frac{1}{2^6}$. So Prob$(Y = 6) = \frac{20}{2^6} = 5/16$. Similarly, we have $2 \cdot \binom{6}{3} = 40$ options to have $Y = 7$, each with probability $\frac{1}{2^7}$. So Prob$(Y = 7) = \frac{40}{2^7} = 5/16$. Thus $H(Y) = \frac{1}{8}\log 8 + \frac{1}{4}\log 4 + 2 \cdot \frac{5}{16}\log\frac{16}{5} = \frac{3}{8} + \frac{1}{2} + \frac{40}{16} - \frac{5}{8}\log 5 = \frac{27}{8} - \frac{5}{8}\log 5$. Now to calculate $H(X)$, note that each particular outcome of i games have probability $\frac{1}{2^i}$. So the distribution of X contains the value $\frac{1}{16}$ twice, $\frac{1}{32}$ 8 times, $\frac{1}{64}$ 20 times, and $\frac{1}{128}$ 40 times. Thus $H(X) = \frac{2}{16}\log 16 + \frac{8}{32}\log 32 + \frac{20}{64}\log 64 + \frac{40}{128}\log 128 = 1/2 + 5/4 + 30/16 + 35/16 = \frac{93}{16}$.
Thus we obtained

$$H(X) = \frac{93}{16}, \quad H(Y) = \frac{27}{8} - \frac{5}{8}\log 5,$$

$$H(X|Y) = \frac{93}{16} - (\frac{27}{8} - \frac{5}{8}\log 5) = \frac{39}{16} + \frac{5}{8}\log 5,$$

$$H(Y|X) = 0, \quad I(X,Y) = H(Y) = \frac{27}{8} - \frac{5}{8}\log 5.$$

◇

Exercise 5. remained homework:

5. We have $n = 2^k$ for some positive integer k. What is the maximum possible value of p_1 if $(p_1, p_2, \ldots, p_n)$ is a probability distribution satisfying $p_1 \geq p_2 \geq \ldots \geq p_n$ and the property that the 2^k binary sequences of length k provide an optimal encoding of the source emitting symbol i with probability $p_i (i = 1, \ldots, n)$?

Sixth lecture (October 9, 2018)

Entropy of a source

The entropy of a source in general is defined as follows.
Definition. The entropy of a source emitting the sequence of random variables $X_1, X_2, \ldots$ is

$$\lim_{n\to\infty} \frac{1}{n} H(X_1, \ldots, X_n),$$

provided that this limit exists.

The above limit trivially exists for stationary memoryless sources defined above. Indeed, if the source is stationary and memoryless, then $H(X_1, X_2, \ldots, X_n) = nH(X_1)$, so we have $\lim_{n\to\infty} \frac{1}{n} H(X_1, \ldots, X_n) = \lim_{n\to\infty} \frac{1}{n} nH(X_1) = H(X_1)$.

In fact, once a source is stationary it always has an entropy, it need not be memoryless.

Theorem 7 *If a source $X_1, X_2, \ldots$ is stationary then its entropy exists and is equal to*

$$\lim_{n\to\infty} H(X_n | X_1, \ldots, X_{n-1}).$$

Remark: Note that $\lim_{n\to\infty} H(X_n|X_1,\dots,X_{n-1})$ can be much smaller than $H(X_1)$. Think about a source with source alphabet $\{0,1\}$ that emits the same symbol as the previous one with probability $9/10$ and the opposite with probability $1/10$. In the long run we have the same number of 0's and 1's, $Prob(X_1 = 1) = Prob(X_1 = 0) = 1/2$, so $H(X_1) = 1$, while $\lim_{n\to\infty} H(X_n|X_1,\dots,X_{n-1}) = H(X_n|X_{n-1}) = h(0.9) < 1$.

Proof. By the source being stationary, we have

$$H(X_n|X_1,\dots,X_{n-1}) = H(X_{n+1}|X_2,\dots,X_n) \geq H(X_{n+1}|X_1,X_2,\dots,X_n).$$

Thus the sequence $H(X_i|X_1,\dots,X_{i-1})$ is non-increasing and since all its elements are non-negative, it has a limit.

From the Chain rule we can write

$$\frac{1}{n}H(X_1,\dots,X_n) = \frac{1}{n}\left(H(X_1) + \sum_{i=2}^{n} H(X_i|X_1,\dots,X_{i-1})\right).$$

To complete the proof we refer to a lemma of Toeplitz that says that if $\{a_n\}_{n=1}^{\infty}$ is a convergent sequence of reals with $\lim_{n\to\infty} a_n = a$, then defining $b_n := \frac{1}{n}\sum_{i=1}^{n} a_i$, we have that $\{b_n\}_{n+1}^{\infty}$ is also convergent and $\lim_{n\to\infty} b_n = a$, too. Applying this to $a_n := H(X_n|X_1,\dots,X_{n-1})$ the statement follows. □

Note that the proof implies that the sequence $\frac{1}{n}H(X_1,\dots,X_n)$ is also non-increasing.

Markov chains, Markov source

Def. A stochastic process $Z = Z_1, Z_2, \dots$ is Markov (or Markovian) if for every i we have $P(Z_k|Z_1,\dots,Z_{k-1}) = P(Z_k|Z_{k-1})$. We say that the variables $Z_1, Z_2, \dots$form a Markov chain.

Intuitively the above definition means that knowing just the previous Z_i tells us everything we could know about the next one even if we knew the complete past. Such situations often occur.

Example. If Z_i denotes the number of heads we have when tossing a fair coin i times, then $Z_{i+1} = Z_i$ or $Z_{i+1} = Z_i + 1$ with probability $1/2 - -1/2$ and we cannot say more than this even if we know the values of Z_{i-2}, Z_{i-3}, etc. Thus $Z_1, Z_2, \dots$ is a Markov chain.
The pixels of a picture can be modeled by a Markov chain: After a black pixel we have another black pixel with high probability and a white one with small probability and this can be considered independent of earlier pixels (though clearly, this independence is not completely true).

A Markov chain Z is homogenous if $P(Z_n|Z_{n-1})$ is independent of n.

A Markov chain is stationary if all its stochastic parameters are invariant in "time", that is, they are not dependent on the indices of the Z_i's involved. In particular, a stationary Markov-chain is always homogenous. However, being stationary means more: it also requires that the distribution of Z_i (and not only the conditional probabilities $P(Z_i|Z_{i-1})$) are independent of i. In particular, for a stationary Markov chain already Z_1 is distributed according to the same stationary distribution according to which Z_n is distributed for a (very) large n.

The entropy of a homogenous Markov chain Z is $H(Z_2|Z_1)$, this follows from Theorem 7 above, cf. also the above Remark.

A general Markov source is a stochastic process X, for which each X_i can be written as a function of two random variables, namely $X_i = F(Z_i, Y_i)$ where Z is a homogenous Markov chain and Y is a stationary and memoryless source that is independent of Z.

A Markov source can model a situation where, for example, Z is a text or speech and Y is the noise. When the noise does not effect the outcome of the source, that is, $F(z, y) = z$ for every z and y, then the entropy of the Markov source is simply $H(Z_2|Z_1)$.

Universal source coding, Lempel-Ziv type algorithms

Huffmann code gives optimal average length but it assumes knowledge of source statistics: we expected we know the probabilities p_i with which the characters x_i are emitted by the source. When we have to compress information it may not be so. Or we want to compress earlier than we could know such statistics. (Think about compressing a text. In principle we could first read it through, make the source statistics and then encode. But we may prefer to encode right in the moment we proceed with its reading) The term *universal source coding* refers to coding the source in such a way that we do not have to know the source statistics in advance. The following algorithms are devised to such situations. Although they usually cannot provide as good compression as the Huffman code, they still do pretty well. Perhaps surprisingly, it can be shown that their compression rate approaches the entropy rate of the source, that is the theoretical limit. (We will learn the technique without proving this.) The examples provided in different files (see references to them below) are from the textbook written in Hungarian by László Györfi, Sándor Győri, and István Vajda "Információ- és kódelmélet" (Information Theory and Coding Theory), published by Typotex, 2000, 2002.

First version: LZ77

There is s sliding window in which we see $h_w = h_b + h_a$ characters, where h_b is the number of characters we see backwards and h_a is the number of characters we see ahead. The algorithm looks at the not yet encoded part of the character flow in the "ahead part" of the window and looks for the longest identical subsequence in the window that starts earlier. The output of the encoder is then a triple (t, h, c), where t is the number of characters we have to step backwards to the start of the longest subsequence identical to what comes ahead, h is the length of this longest identical subsequence, and c is the codeword for the first new character that is already not fitting in this longest subsequence. Note that the longest identical subsequence should start in the backward part of the window but may end in the ahead part, so t may be less than h.

For example, when we are encoding the sequence *...cabracadabrarrarrad...*, the *...cabraca* part is already encoded (so the coming part is *dabrarrarrad...*), and we have $h_b = 7, h_a = 6$, then the first triple sent is $(0, 0, f(d))$ (where $f(.)$ is the codeword for the character in the argument), the second triple sent is $(7, 4, f(r))$, etc., see the "LZexamples" file. Note that for the next triple we will have $(3, 5, f(d))$ showing an example when $t < h$.

Second version: LZ78

In LZ77 we build on the belief that similar parts of the text come close to each other. The LZ78 version needs only that substantial parts are repeated but they do not have to be right after each other. Also, LZ77 is sensitive to the window size. In LZ78 we do not have this disadvantage.

We build a codebook and each time we encode we look for the longest new segment that already appears in the codebook. The output is a pair (i, c) where i is the index of the longest coming segment that already has a codeword and c is the first new character after it. Apart from producing this output the algorithm also extends the codebook by putting into it the shortest not yet found segment, which is the concatenation of the segment with index i we found and the character c. This new, one longer segment gets the next index and then we go on with the encoding. For an example see again the "LZexamples" file

Third version: LZW

This is the most popular version of the algorithm that is a modification of LZ78 as suggested by Welch. We now start with a codebook that already contains all the one-character sequences. (They have an index which serves as a codeword for them; we can think about their codeword as the s-ary, or simply binary representation of this index.) We now read the longest new part p of the text that can be found in the codebook and the next character, let it be a. Then the output is simply the index of p, we extend the codebook with the new sequence pa (that we obtain by simply putting a to the end of p) giving it the next index, and we consider the extra character a as the beginning of the not yet encoded part of the text. For an example see the file "LZWexample".

Remark: Huffman encoding can also be adapted to the situation when we cannot create the optimal code in advance. Then we simply use the empirical source statistics. This means that the probability of a character is considered to be equal to the proportion of the number of its appearances up to a certain point to the number of characters seen altogether up to that point. This statistics is updated after every single character. For every new character we use the Huffman code that would belong to the source statistics we calculated right before the arrival of that character. We encode the character according to that code and then update the statistics and modify the code accordingly. This can be done simultaneously at the decoder (without communication), so the decoder will always be aware of the code the encoder is using. For the latter we need a rule how to handle equal probabilities that could result in different optimal encodings, but such rules are not hard to agree on. Also, the encoder and the decoder has to agree on a starting code. This can be one that assumes uniform distribution on the source characters or one that estimates the source statistics according to some a priori knowledge if that exists. (For example, if the source is an English text, we can use the more or less known frequency of each letter in the English language as the starting distribution and a Huffman encoding of that.)

Seventh lecture (October 16, 2018)

Homogenous Markov chains tend to a stationary distribution whose entropy might be much larger than the entropy of the Markov chain. When the homogenous Markov chain has r *states* its behavior is described by an $r \times r$ stochastic matrix (each row is a probability distribution) A defined by $A[i, j] = Prob(Z_2 = j|Z_1 = i)$. Then the stationary distribution $\boldsymbol{p}$ is the probability distribution $(p_1, \ldots, p_r)$ for which $\boldsymbol{p}A = \boldsymbol{p}$.

Example: Let A be the 2×2 matrix with first row $p, 1-p$ and second row $1-p, p$. Then the stationary distribution is $\frac{1}{2}, \frac{1}{2}$, while the entropy of the Markov chain is $h(p)$.

Exercise: Let $X_1, X_2, \dots$ be a Markov chain for which $\mathrm{Prob}(X_1 = 0) = \mathrm{Prob}(X_) = 1) = \frac{1}{2}$ and let the transition probabilities for $i \geq 1$ be given by $\mathrm{Prob}(X_{i+1} = 0|X_i = 0) = \mathrm{Prob}(X_{i+1} = 1|X_i = 0) = \frac{1}{2}$, while $\mathrm{Prob}(X_{i+1} = 0|X_i = 1) = 0$ and $\mathrm{Prob}(X_{i+1} = 1|X_i = 1) = 1$. Calculate the entropy of the source whose outcome is the resulting sequence of random variables $X_1, X_2, \dots$.

Intuitively the solution is quite clear: This source emits some number (perhaps zero) 0's first, but after the first 1 it will emit only 1's. As i gets larger and larger, the probbility of $X_i = 0$ is smaller and smaller (in fact it will be $\frac{1}{2^i}$), so if i is large, then X_i is almost certainly 1. Therefore the uncertainty about the value of X_i approaches zero, so the entropy of the source should be 0.

This intuition is easy to confirm by calculation: by Theorem 7

$$H(X) = \lim_{n\to\infty} H(X_n|X_1, \dots, X_{n-1}) = \lim_{n\to\infty} H(X_n|X_{n-1}) =$$

$$\lim_{n\to\infty} \mathrm{Prob}(X_{n-1} = 0)h(1/2) + \mathrm{Prob}(X_{n-1} = 1)h(0) =$$

$$\lim_{n\to\infty} \frac{1}{2^{n-1}} + (1 - \frac{1}{2^{n-1}})0 = 0.$$

Another example: Random variables $X_i, i = 1, 2\dots$ take two posible values, 0 or 1 and form a Markov chain. The transition probabilities are

$$P(X_{i+1} = 0|X_i = 0) = \frac{1}{2}, \quad P(X_{i+1} = 1|X_i = 0) = \frac{1}{2}$$

and

$$P(X_{i+1} = 0|X_i = 1) = \frac{1}{3}, \quad P(X_{i+1} = 1|X_i = 1) = \frac{2}{3}.$$

Give the entropy of the source that emits these variables.

Solution: The entropy of this source is, as we have learnt, $\lim_{n\to\infty} H(X_{n+1}|X_n)$, so we have to calculate this quantity. That needs that we know the stationary distribution $(p, 1-p)$, that should satisfy

$$p\frac{1}{2} + (1-p)\frac{1}{3} = p$$

(and equivalently $p\frac{1}{2} + (1-p)\frac{2}{3} = 1-p$).

Solving either of the above two equations gives $p = \frac{2}{5}$. Thus $\lim_{n\to\infty} H(X_{n+1}|X_n)$ is equal to $H(X_{n+1}|X_n)$ in the stationary distribution, i.e.,

$$\frac{2}{5}h\left(\frac{1}{2}\right) + \frac{3}{5}h\left(\frac{1}{3}\right) = \frac{2}{5} + \frac{3}{5}\log 3 - \frac{3}{5} \cdot \frac{2}{3}\log 2 = \frac{3}{5}\log 3.$$

Eighth lecture (October 30, 2018)

Source coding with negligible probability of error

A disadvantage of using variable length codewords is that if a codeword becomes erroneous causing a mistake in the decoding this mistake may propagate to the subsequent codewords. This will not happen if all codewords have the same length. Then, however, we cannot achieve any compression once we insist on error-free decoding. If the k-length codewords encode r messages over an s-ary alphabet then we must have $s^k \geq r$ and this is clearly enough, too. But this has nothing to do with the source statistics, so this way we encode everything with the same average length as if the r messages were equally likely thus providing maximum entropy. To overcome this problem we allow a negligible, but positive probability of error.

We will use block codes $f : \mathcal{X}^k \to \mathcal{Y}^m$.

Def. A code $f : \mathcal{X}^k \to \mathcal{Y}^m$ is possible to decode with error probability at most ε if there exists a function $\varphi : \mathcal{Y}^m \to \mathcal{X}^k$ such that

$$Prob(\varphi(f(X_1, \ldots, X_k)) \neq (X_1, \ldots, X_k)) < \varepsilon.$$

We can think about the code as the pair (f, φ) where we may select φ to be the decoding function achieving the smallest error probability for f. (The error probability is then defined as the above quantity on the left hand side, that is as $Prob(\varphi(f(X_1, \ldots, X_k)) \neq (X_1, \ldots, X_k))$.

We are interested in codes with small error probability and small rate m/k. We are able to give $|\mathcal{Y}|^m$ distinct codewords, so we may have that many elements of $\mathcal{X}^k$ decoded in an error-free manner. Thus the error probability is minimized if we choose the $|\mathcal{Y}|^m$ largest probability elements of $\mathcal{X}^k$ to encode in a one-to-one way, while all the rest of the elements of $\mathcal{X}^k$ will just be given some codeword which will not be decoded to them.

Let $N(k, \varepsilon)$ be the smallest number N for which if $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ are the N largest probability k-length outputs of the source X, we have $\sum_{i=1}^N p(\boldsymbol{x}_i) > 1 - \varepsilon$. The quantity relevant for us is $\frac{\log N(k,\varepsilon)}{k}$. The main result here is, that for a fairly general class of sources this quantity tends to the entropy of the source. So even in this setting it is the entropy of the source that gives the quantitative characterization of the best encoding rate one can achieve. This verifies again the intuition that interprets the entropy of the source as a measure of information content in the source variables.

Def. We call a stationary source *information stable* if for every $\delta > 0$

$$\lim_{k\to\infty} Prob\left\{\left|-\frac{1}{k}\log p(X_1, \ldots, X_k) - H(X)\right| > \delta\right\} = 0.$$

Some remarks:
1) If X is stationary and memoryless, then it is information stable. Here is the proof:
$Y_k := -\frac{1}{k}\log p(X_1, \ldots, X_k) = -\frac{1}{k}\log(p(X_1)p(X_2)\ldots p(X_k)) = \frac{1}{k}\sum_{i=1}^k(-\log p(X_i))$, where the $(-\log p(X_i))$'s are independent identically distributed (i.i.d.) random variables. Observe that their expected value is just the entropy $H(X) = H(X_i)$. By the weak law of large numbers the Y_i's converge in probability to this common expected value which means exactly that

$$\lim_{k\to\infty} Prob\left\{\left|-\frac{1}{k}\log p(X_1, \ldots, X_k) - H(X)\right| > \delta\right\} = 0,$$

i.e. the information stability of the source.
2) The intuitive meaning of information stability is that if k is large enough then there is a large probability set $A \subseteq \mathcal{X}^k$ for which $\mathbf{x} \in A$ implies

$$p(\mathbf{x}) \approx 2^{-kH(X)}.$$

If $Prob(A)$ is close to 1 this also implies

$$|A| \approx \frac{1}{p(\mathbf{x})} \approx 2^{kH(X)}.$$

Thus encoding the elements of A in a one-to-one manner will require about $kH(X)$ bits.

Theorem 8 *Let the stationary source X be information stable. Then for every $0 < \varepsilon < 1$*

$$\lim_{k\to\infty} \frac{1}{k} \log N(k,\varepsilon) = H(X).$$

Proof. Let $B_{k,\varepsilon}$ be the set of the $N(k,\varepsilon)$ highest probability k-length source outputs and for every $\delta \in (0,1)$ define

$$A_{k,\delta} := \left\{ \mathbf{x} \in \mathcal{X}^k : 2^{-k(H(X)+\delta)} \le p(\mathbf{x}) \le 2^{-k(H(X)-\delta)} \right\}.$$

Then by

$$1 \ge P(A_{k,\delta}) = \sum_{\mathbf{x}\in A_{k,\delta}} p(\mathbf{x}) \ge |A_{k,\delta}| 2^{-k(H(X)+\delta)}$$

we have

$$|A_{k,\delta}| \le 2^{k(H(X)+\delta)}.$$

By information stability, we know that $P(A_{k,\delta})$ is close to 1 (in particular, $P(A_{k,\delta}) > 1-\varepsilon$) for large enough k. Since $B_{k,\varepsilon}$ is the smallest cardinality set with probability at least $1-\varepsilon$, we get that for large enough k

$$N(k,\varepsilon) = |B_{k,\varepsilon}| \le |A_{k,\delta}| \le 2^{k(H(X)+\delta)},$$

and so

$$\frac{1}{k} \log N(k,\varepsilon) \le H(X) + \delta.$$

Since δ can be arbitrarily small, this also implies

$$\limsup_{k\to\infty} \frac{1}{k} \log N(k,\varepsilon) \le H(X).$$

For the reverse inequality let k be large enough for $P(A_{k,\delta}) > \frac{1+\varepsilon}{2}$. Then (denoting the complement of set U by U^c) we have

$$\frac{1+\varepsilon}{2} < P(A_{k.\delta}) = P(A_{k\delta} \cap B_{k,\varepsilon}) + P(A_{k,\delta} \cap B^c_{k,\varepsilon}) < P(A_{k,\delta} \cap B_{k,\varepsilon}) + \varepsilon,$$

that is

$$P(A_{k,\delta} \cap B_{k,\varepsilon}) > \frac{1-\varepsilon}{2}.$$

We can thus write

$$\frac{1-\varepsilon}{2} < P(A_{k,\delta} \cap B_{k,\varepsilon}) \le |B_{k,\varepsilon}| \max_{\mathbf{x}\in A_{k,\delta}} p(\mathbf{x}) \le |B_{k,\varepsilon}| 2^{-k(H(X)-\delta)} = N(k,\varepsilon) 2^{-k(H(X)-\delta)}.$$

So we get

$$N(k,\varepsilon) > \frac{1-\varepsilon}{2} 2^{k(H(X)-\delta)}.$$

Thus

$$\frac{1}{k} \log N(k,\varepsilon) > H(X) - \delta + \frac{1}{k} \log\left(\frac{1-\varepsilon}{2}\right).$$

Since $\delta > 0$ can be arbitrarily small and $\log\left(\frac{1-\varepsilon}{2}\right)$ is a constant independent of k, the latter implies

$$\liminf_{k\to\infty} \frac{1}{k} \log N(k,\varepsilon) \geq H(X)$$

and thus the statement. □

By the foregoing we have essentially proved the following coding theorem.

Theorem 9 *Let X be a stationary source which is also information stable and $0 < \varepsilon < 1$. Let us have a sequence of codes $f_k : \mathcal{X}^k \to \mathcal{Y}^{m_k}, (|\mathcal{Y}| = s)$ which encodes the source with less than ε probability of error. Then*

$$\liminf_{k\to\infty} \frac{m_k}{k} \geq \frac{H(X)}{\log s}.$$

On the other hand, for every $0 < \varepsilon < 1$ and $\delta > 0$ if k is large enough then there exists an $f_k : \mathcal{X}^k \to \mathcal{Y}^m$ code with probability of error less than ε and

$$\frac{m}{k} < \frac{H(X)}{\log s} + \delta.$$

The proof is immediate by realizing that if we want a code with probability of error less than ε, then the best we can do is to encode the $N(k,\varepsilon)$ largest probability elements of $\mathcal{X}^k$ in a one-to-one way to codewords of length $\log_s N(k,\varepsilon) = \frac{\log N(k,\varepsilon)}{\log s}$. Thus the compression rate will tend to $\lim_{k\to\infty} \frac{1}{k} \log_s N(k,\varepsilon) = \frac{H(X)}{\log s}$.

Ninth lecture (November 6, 2017)

Quantization

In many practical situations the source variables are real numbers, thus have a continuum range. If we want to use digital communication we have to discretize, which means that some kind of "rounding" is necessary.

Def. Let $X = X_1, X_2, \ldots$ be a stationary source, where the X_i's are real-valued random variables. A (1-dimensional) *quantized* version of this source is a sequence of discrete random variables (another source) $Q(X_1), Q(X_2), \ldots$ obtained by a map $Q : R \to R$ where the range of the map is finite. The function $Q(.)$ is called the *quantizer.*

Goal: Quantize a source so that the caused distortion is small.

How can we measure the distortion? We will do it by using the quadratic distortion measure $D(Q)$ defined for n-length blocks as

$$D(Q) = \frac{1}{n} E\left(\sum_{i=1}^{n} (X_i - Q(X_i))^2\right),$$

where $E(.)$ means expected value.
Since our X_i's are identically distributed we have

$$D(Q) = E((X - Q(X))^2).$$

(Here X is meant to have the same distribution as all the X_i's.)

Let the range of $Q(.)$ be the set $\{x_1, \ldots, x_N\}$, where the x_i's are real numbers. $Q(.)$ is uniquely defined by the values $x_1, \ldots, x_N$ and the sets $B_i = \{x : Q(x) =$

$x_i\}$. Once we fix $x_1, \ldots, x_N$, we will have the smallest distortion $D(Q)$ if every x is "quantized" to the closest x_i, i.e.,

$$B_i = \{x : |x - x_i| \leq |x - x_j| \ \forall j \neq i\}.$$

(Note that this rule puts some values into two neighboring B_i's (considering $x_1 < x_2 < \ldots < x_N$, we have $x = \frac{1}{2}(x_i + x_{i+1})$ in both B_i and B_{i+1}). This can easily be resolved by saying that all these values go to (say) the smaller indexed B_i.)

If now we consider the B_i's fixed then the smallest distortion $D(Q)$ is obtained if the x_i values lie in the barycenter of the B_i, which is $E(X|B_i) := E(X|X \in B_i) = \frac{\int_{B_i} xf(x)dx}{\int_{B_i} f(x)dx}$, where $f(x)$ is the density function of the random variable X. (We will always assume that $f(x)$ has all the "nice" properties needed for the existence of the integrals we mention.)

The previous claim (smallest distortion is achieved for given quantization intervals B_i when $Q(x) = E(X|B_i)$ for $x \in B_i$) can be seen as follows.

This holds for all B_i separately, so it is enough to show it for one of them. By the linearity of expectation

$$E((X - c)^2) = E(X^2) - c(2E(X) - c),$$

and this is smallest when $c(2E(X) - c)$ is largest. Since the sum of c and $2E(X) - c$ does not depend on c, one can see simply from the inequality between the arithmetic and geometric mean ($\frac{a+b}{2} \geq \sqrt{ab}$ with equality iff $a = b$) that this product is largest when $c = E(X)$.

Lloyd-Max algorithm

The above suggests an iterative algorithm to find a good quantizer: We fix some quantization levels $x_1 < \ldots < x_N$ first and optimize for them the B_i domains by defining them as above: let $y_i = \frac{x_i + x_{i+1}}{2}$ for $i = 1, \ldots, N-1$ and

$$B_1 := (-\infty, y_1], \quad B_i := (y_i, y_{i+1}], \ i = 2, \ldots, N-1, \quad B_N = (y_N, \infty).$$

Notice that in general there is no reason for the x_i's to be automatically the barycenters of the domains B_i obtained in the previous step. So now we can consider these domains B_i fixed and optimize the quantization levels with respect to them by redefining them as the corresponding barycenters:

$$x_i := \frac{\int_{B_i} xf(x)dx}{\int_{B_i} f(x)dx}.$$

Now we can consider again the so-obtained x_i's fixed and redefine the B_i's for them, and so on. After each step (or after each "odd" step when we optimize the B_i domains for the actual x_i's) we can check whether the current distortion is below a certain threshold. If yes we stop the algorithm, if no, then we continue with further iterations.

Remarks.
1) It should be clear from the above that if either of the two steps above changes the x_i quantization levels or the B_i domains, then the quantizer before that step was not optimal. It is possible, however, that no such change is attainable already and the quantizer is still not optimal. Here is an example. Let X be a random variable that takes its values on the finite set $\{1, 2, 3, 4\}$ with uniform distribution. (That is $P(X = 1) = P(X = 2) = P(X = 3) = P(X = 4) = 1/4$.) Let $N = 2$ that is we are allowed to use two values for the quantization. There

are three different quantizers for which neither of the above steps can cause any improvement, but only one of them is optimal. These three quantizers can be described by

$$Q_1(1) = 1, Q_1(2) = Q_1(3) = Q_1(4) = 3;$$

$$Q_2(1) = Q_2(2) = 1.5, Q_2(3) = Q_2(4) = 3.5;$$

$$Q_3(1) = Q_3(2) = Q_3(3) = 2, Q_3(4) = 4.$$

It takes an easy calculation to check that $D(Q_1) = D(Q_3) = 0.5$, while $D(Q_2) = 0.25$. Thus only Q_2 is optimal, although neither of Q_1 and Q_3 can be improved by the Lloyd-Max algorithm.
2) Let us call a quantizer a Lloyd-Max quantizer if the two steps of the Lloyd-Max algorithm have no effect on them. In the previous remark we have seen that a Lloyd-Max quantizer is not necessarily optimal. Fleischer gave a sufficient condition for the optimality of a Lloyd-Max quantizer. It is in terms of the density function $f(x)$ of the random variable to be quantized. In particular, it requires that $\log f(x)$ is concave.

Tenth lecture (November 13, 2018)

The above condition of Fleischer is satisfied by the density function of a random variable uniformly distributed in an interval $[a, b]$. Thus a corollary of Fleischer's theorem is that there is only one Lloyd-Max quantizer with N levels for the uniform distribution on $[a, b]$. It is not hard to see that this should be the uniform quantizer: the one belonging to $B_i = \{x : a+(i-1)\frac{b-a}{N} \le x \le a+i\frac{b-a}{N}\}$ and quantization levels at the middle of these intervals. (The extreme points of the intervals belonging to two neighboring B_i's can be freely decided to belong to either of them.)

Distortion of the uniform quantizer

The simplest quantizer is the uniform quantizer, we investigate it a bit closer. Note that we do not assume now that the distribution we work with is uniform. For simplicity we assume, however, that the density function of our random variable to be quantized is 0 outside the interval $[-A, A]$, and it is continuous within $[-A, A]$. The N-level uniform quantizer is defined by the function

$$Q_N(x) = -A + (2i-1)\frac{A}{N}$$

whenever

$$-A + 2(i-1)\frac{A}{N} < x \le -A + 2i\frac{A}{N}.$$

(To be precise: for $x = -A$ we also have $Q_N(-A) = -A + \frac{A}{N}$.)

The length of each interval for the elements of which we assign the same value is then $q_N = \frac{2A}{N}$. The following theorem gives the distortion of the uniform quantizer asymptotically (as N goes to infinity) in terms of q_N.

Theorem 10 *If the density function f of the random variable X staisfies the above requirements (continuous in $[-A, A]$ and 0 outside it) then for the distortion of the N-level uniform quantizer Q_N we have*

$$\lim_{N\to\infty} \frac{D(Q_N)}{q_N^2} = \frac{1}{12}.$$

Proof. We will use the following notation. The extreme points of the quantization intervals are

$$y_{N,i} = -A + 2i\frac{A}{N},\ i = 0, 1, \ldots, N,$$

while the quantization levels are

$$x_{N,i} = -A + (2i-1)\frac{A}{N},\ i = 1, 2, \ldots, N.$$

With this notation the distortion can be written by definition as

$$D(Q_n) = \sum_{i=1}^{N} \int_{y_{N,i-1}}^{y_{N,i}} (x - x_{N,i})^2 f(x)dx.$$

We define the auxiliary density function $f_N(x)$ as

$$f_N(x) := \frac{1}{q_N} \int_{y_{N,i-1}}^{y_{N,i}} f(z)dz \ \text{ if } x \in (y_{N,i-1}, y_{N,i}].$$

First we calculate the distortion $\hat{D}(Q_N)$ of Q_N with respect to this auxiliary density function.

$$\hat{D}(Q_N) = \sum_{i=1}^{N} \int_{y_{N,(i-1)}}^{y_{N,i}} (x - x_{N,i})^2 f_N(x)dx =$$

$$\sum_{i=1}^{N} \frac{1}{q_N} \int_{y_{N,(i-1)}}^{y_{N,i}} f(z)dz \int_{y_{N,(i-1)}}^{y_{N,i}} (x - x_{N,i})^2 dx =$$

$$\sum_{i=1}^{N} \frac{1}{q_N} \int_{y_{N,(i-1)}}^{y_{N,i}} f(z)dz \int_{-\frac{q_N}{2}}^{\frac{q_N}{2}} x^2 dx =$$

$$\frac{q_N^2}{12} \sum_{i=1}^{N} \int_{y_{N,(i-1)}}^{y_{N,i}} f(z)dz = \frac{q_N^2}{12}.$$

To finish the proof we will show that

$$\lim_{N\to\infty} \frac{\hat{D}(Q_N) - D(Q_N)}{\hat{D}(Q_N)} = \lim_{N\to\infty} \frac{\hat{D}(Q_N) - D(Q_N)}{q_N^2/12} = 0,$$

that is clearly enough.

Since f is continuous in the closed interval $[-A, A]$ it is also uniformly continuous. Thus for every $\varepsilon > 0$ there exists N_0 such that if $N \geq N_0$ then $|f(x) - f(x')| < \varepsilon$ whenever $x, x' \in (y_{N,(i-1)}, y_{N,i})$ (since $|y_{N,(i-1)} - y_{N,i}| < q_N$, and $q_N \to 0$ as $N \to \infty$).
So for $N \geq N_0$ we can write

$$\frac{|\hat{D}(Q_N) - D(Q_N)|}{q_N^2/12} =$$

$$\frac{12}{q_N^2} \left| \sum_{i=1}^{N} \int_{y_{N,(i-1)}}^{y_{N,i}} (x - x_{N,i})^2 f(x)dx - \sum_{i=1}^{N} \int_{y_{N,(i-1)}}^{y_{N,i}} (x - x_{N,i})^2 f_N(x)dx \right| \leq$$

$$\frac{12}{q_N^2} \sum_{i=1}^{N} \int_{y_{N,(i-1)}}^{y_{N,i}} (x - x_{N,i})^2 |f(x) - f_N(x)| dx \leq$$

$$\frac{12}{q_N^2}\sum_{i=1}^{N}\int_{-q_N/2}^{q_N/2} z^2\varepsilon dz = \frac{12}{q_N^2}N\frac{q_N^3}{12}\varepsilon = q_N N\varepsilon = \frac{2A}{N}N\varepsilon = 2A\varepsilon$$

that can be made arbitrarily small by choosing ε small enough. This completes the proof. $\square$

Channel coding

Channel model: stochastic matrix. Rows belong to input letters, columns belong to output letters. $W_{i,j} = W(y_j|x_i)$, which is the probability of receiving y_j when x_i was sent.

Example. (The input and output alphabets are denoted $\mathcal{X}, \mathcal{Y}$, respectively.)

Binary symmetric channel (BSC): $\mathcal{X} = \mathcal{Y} = \{0, 1\}$, $W(1|1) = W(0|0) = 1 - p$, $W(1|0) = W(0|1) = p$.

Goal: Communicating reliably and efficiently.
Reliably means: with small probability of error.
Efficiently means: with as few channel use as possible.

Warning: In the following we will use the letter n for codeword length unlike durnig the lecture where codeword length was denoted by t.

Code: A(n invertible) function $f : \mathcal{M} \to \mathcal{X}^n$, where $\mathcal{M}$ is the set of possible messages. The relevance of $\mathcal{M}$ will be its size $M := |\mathcal{M}|$. We can also think about the code as its codeword set $\{\boldsymbol{c}^{(1)}, \dots, \boldsymbol{c}^{(M)}\}$.
We also need a decoding function $\varphi : \mathcal{Y}^n \to \mathcal{M}$ that tells us which message we decode a certain received sequence to. (One can also define codes as the pairs of functions (f, φ).)

The probability of error if the message m_i, that is the codeword $\boldsymbol{c}^{(i)}$, was sent is

$$P_{e,i} = \sum_{\varphi(\mathbf{y})\neq m_i} Prob(\boldsymbol{y} \text{ was received}|\boldsymbol{c}^{(i)} \text{ was sent}) = \sum_{\varphi(\boldsymbol{y})\neq m_i}\prod_{r=1}^{n} W(y_r|c_r^{(i)}),$$

where y_r and $c_r^{(i)}$ denote the rth character in the sequences $\boldsymbol{y}$ and $\boldsymbol{c}^{(i)}$, respectively.

We want small error independently of the probability distribution on the messages. So we define the average error probability that is the average of the $P_{e,i}$ values on the M messages:

$$\bar{P}_e = \frac{1}{M}\sum_{i=1}^{M} P_{e,i}.$$

The efficiency of the code is measured by its rate:

$$R = \frac{\log_2 M}{n}.$$

Shannon's Channel Coding Theorem, one of the most fundamental results in information theory, says that discrete memoryless channels have a characteristic value, their *capacity*, with the property that one can communicate reliably with any rate below it, and one cannot, above it. Here "reliably" means "with arbitrary small probability of error".

First we define the capacity C_W of a discrete memoryless channel given by its matrix W.

Def.

$$C_W := \max I(X,Y),$$

where the maximization is over all joint distributions of the pair of random variables (X,Y) that satisfy that the conditional probability of Y given X is what is prescribed by W.

The above expression can be rewritten as

$$C_W = \max \left\{ \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \right\}$$

$$= \max \left\{ \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x)p(y|x) \log \frac{p(y|x)}{\sum_{x' \in \mathcal{X}} p(x')p(y|x')} \right\}$$

The advantage of the last expression is that it shows very clearly that when maximizing $I(X,Y)$ what we can vary is the distribution of X, that is the input distribution. (All other values in the last expression are conditional probabilities given by the channel matrix W.)

Example: For the binary symmetric channel we can calculate the capacity as follows. $I(X,Y) = H(Y) - H(Y|X)$ and it follows from the channel characteristics that $H(Y|X=0) = H(Y|X=1) = h(p)$, so $H(Y|X) = h(p)$ irrespective of the distribution of X. So $I(X,Y) = H(Y) - h(p) \leq \log 2 - h(p) = 1 - h(p)$. Observing that if we let X have uniform distribution, then Y will also have uniform distribution (that results in $H(Y) = 1$), we conclude that this upper bound can be achieved. Thus the *capacity of the binary symmetric channel* is $1 - h(p)$.

Now we state the **Channel Coding Theorem**:

For every rate $R < C$ there exists a sequence of codes with length n and number of codewords at least 2^{nR} such that the average probability of error $\bar{P}_e$ goes to zero as n tends to infinity.

Conversely, for any sequence of codes with length n, number of codewords at least 2^{nR} and average error probability tending to zero as n goes to infinity, we must have $R \leq C$.

In short one can say that all rates below capacity are achievable with an arbitrarily small error probability, and this is not true for any rate above capacity.

Here is another example for calculating channel capacity. The following channel is called the *binary erasure channel*: $\mathcal{X} = \{0,1\}$, $\mathcal{Y} = \{0,1,*\}$, $W(1|1) = W(0|0) = 1-p$, $W(1|0) = W(0|1) = 0$, $W(*|0) = W(*|1) = p$.

We need to calculate $C = \max I(X,Y) = \max\{H(Y) - H(Y|X)\} = \max\{H(X) - H(X|Y)\}$. First we use the second form, then also show how the same result can be obtained from the first one. Assume that at the input we have the distribution given by $Prob(X=0) = q, Prob(X=1) = 1-q$. Then $H(X) - H(X|Y) = h(q) - (Prob(Y=0)H(X|Y=0) + Prob(Y=1)H(X|Y=1) + Prob(Y=*)H(X|Y=*)$. Note that both $H(X|Y=0)) = 0$ and $H(X|Y=1) = 0$ since $Y=0$ and $Y=1$ leaves no uncertainty about the value of X. At the same time $Prob(Y=*) = p$ and $H(X|Y=*) = h\left(\frac{pq}{p}\right) = h(q)$, so we can write

$$C = \max\{H(X) - H(X|Y)\} = \max_q h(q) - ph(q) = 1 - p.$$

Notice that this is indeed what one would intuitively expect.

Now we show how the same value can be obtained via using the other formula. We need to calculate $C = \max I(X,Y) = \max\{H(Y) - H(Y|X)\} = \max\{H(Y) - h(p)\}$. So our task is to maximize $H(Y)$ (since $h(p)$ is a fixed value). Let E be the indicator variable taking value 1 when an erasure occurs, that is, when $Y = *$, and 0 otherwise. Note that E is a function of Y, so $H(Y) = H(E,Y)$. Then by the Chain rule $H(Y) = H(E,Y) = H(E) + H(Y|E) = h(p) + P(E=1)H(Y|E=1) + P(E=0)H(Y|E=0) = h(p) + p\cdot 0 + (1-p)\cdot H(Y|Y \neq *) \leq h(p) + 1 - p$. Note that if the input distribution is set to be uniform then the last inequality is an equality, so we get $\max H(Y) = h(p) + 1 - p$ and thus

$$C = \max I(X,Y) = \max\{H(Y) - h(p)\} = 1 - p.$$

Eleventh lecture (November 20, 2018)

First we will prove the converse statement. Even that we first show in a weaker form, namely we show that for zero-error codes we must have $R \leq C$. We will use the following lemma.

Lemma 1 *Let Y^n be the output of a discrete memoryless channel with capacity C resulting from the input X^n. Then*

$$I(X^n, Y^n) \leq nC.$$

Proof.

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

Here the second equality follows from the Chain rule, and the third equality used the discrete memoryless property of the channel, which implies that Y_i depends only on X_i among $Y_1,...,Y_{i-1},X_1,\dots X_n$ and thus the used equality of conditional entropies. (The other relations should be clear: the first and fourth equality follows from the definition of mutual information, the first "$\leq$" is a consequence of the standard property of the entropy of joint distributions, while the final inequality follows from the definition of channel capacity. □

Now assume that we communicate over a discrete memoryless channel of capacity C with zero-error, that is we have a code of length n with $M = 2^{nR}$ codewords and $\bar{P}_e = 0$. Then

$$R \leq C.$$

Here is the proof. Let the random variable that takes its values on the message set $\mathcal{M}$ (that is its value is the index of the message m_i to be sent) be denoted by U. (We assume that U is uniformly distributed, so its entropy is $\log M$.) Now we can write

$$nR = H(U) = H(U|Y^n) + I(U, Y^n) = I(U, Y^n) = I(X^n, Y^n)$$
$$\leq \sum_{i=1}^{n} I(X_i, Y_i) \leq nC.$$

Here we used that if the code has error probability zero then the message U sent is completely determined by the channel output Y^n, therefore $H(U|Y^n) = 0$. This explains the third equality above (the first two come from the appropriate definitions). The fourth equality $I(U, Y^n) = I(X^n, Y^n)$ follows from considering the coding function establishing a one-to-one correspondence between U and the input codeword X^n. (In some discussions X^n is considered as a "processed" version of U and then $I(U, Y^n) \leq I(X^n, Y^n)$ follows, which also properly fits the chain of inequalities above.) The last two inequalities are just proven in Lemma 1 above. Now dividing by n we just get the required $R \leq C$ inequality. $\square$

To strengthen the above proof so that we get $R \leq C$ also for negligible (but not necessarily zero) error probability codes we will need to estimate $H(U|Y^n)$ from above in the calculation just presented, since we no longer can say it is 0 if Y^n does not determine U with certainty. This will come next.
(Today we spent the rest of the class meeting with solving exercises.)

Twelfth lecture (November 27, 2018)

To have an upper bound on $H(U|Y^n)$ as promised last time, we need another lemma that is known as Fano's inequality.

Lemma 2 *(Fano's inequality) Let us have a discrete memoryless channel where the input message U is uniformly distributed over 2^{nR} possible messages. After sending the codeword belonging to U through the channel we receive the output Y^n from which we estimate U by $\hat{U}$. The error probability is $\bar{P}_e = P(\hat{U} \neq U) = \frac{1}{2^{nR}} \sum P_{e,i}$. Then we have*

$$H(U|\hat{U}) \leq 1 + \bar{P}_e nR.$$

Proof. Let E be the random variable defined by

$$E \in \{0, 1\}, E = 1 \Leftrightarrow \hat{U} \neq U,$$

i.e., the indicator variable for decoding the received word with an error. Clearly, E is determined by the pair $(U, \hat{U})$, so $H(E|U, \hat{U}) = 0$

Then using the Chain rule to expand $H(E, U|\hat{U})$ in two different ways, we can write

$$H(U|\hat{U}) = H(U|\hat{U}) + H(E|U, \hat{U}) = H(E, U|\hat{U}) = H(E|\hat{U}) + H(U|E, \hat{U})$$
$$\leq h(\bar{P}_e) + \bar{P}_e \log 2^{nR} \leq 1 + \bar{P}_e nR,$$

giving the statement. For the inequalities we used that conditioning cannot increase entropy and that $H(U|\hat{U}, E = 0) = 0$ since $E = 0$ means that $U = \hat{U}$, thus $H(U|E, \hat{U}) = P(E = 0)H(U|\hat{U}, E = 0) + P(E = 1)H(U|\hat{U}, E = 1) \leq (1 - \bar{P}_e)0 + \bar{P}_e \log 2^{nR}$. $\square$

In the proof below we will use the intuitively clear, but not yet explicitly stated property of conditional entropy expressed by the following lemma. It can be proven with a little work from Jensen's theorem.

Lemma 3 *If X, Y are two random variables and $Z = g(Y)$ is a function of Y, then*

$$H(X|Y) \leq H(X|Z).$$

Proof of the converse of the channel coding theorem. We follow the proof we have seen for zero-error codes and plug in Fano's inequality at the appropriate place. (The notation is identical to that used in Fano's inequality.)

$$nR = H(U) = H(U|\hat{U}) + I(U, \hat{U}) \leq 1 + \bar{P}_e nR + I(U, \hat{U}) \leq$$

$$1 + \bar{P}_e nR + I(X^n, Y^n) \leq 1 + \bar{P}_e nR + nC.$$

Here the first inequality is by Fano's inequality. The second inequality is a consequence of $\hat{U}$ being a function of Y^n and thus $I(U, \hat{U}) = H(U) - H(U|\hat{U}) \leq H(U) - H(U|Y^n)$. We consider X^n and U having a one-to-one correspondence between them, so we can write $H(U) - H(U|Y^n) = H(X^n) - H(X^n|Y^n)$. (In a more general way we can refer to the so-called data processing inequality that expresses that if the random variables A, B, Z form a Markov chain then $I(A, Z) \leq I(A, B)$. In that case we can write the inequality even if we consider X^n simply a function, not necessarily a one-to-one function of U.)

So dividing by n we have obtained above that

$$R(1 - \bar{P}_e) \leq C + \frac{1}{n}.$$

Now letting n go to infinity we know that $\bar{P}_e \to 0$ and $\frac{1}{n} \to 0$, so we get

$$R \leq C$$

as stated. □

Thirteenth lecture (December 4, 2018)

I showed some hints about the proof of the direct part of the Channel Coding Theorem. The proof of this part is well sketched at the wikipedia article at

`https://en.wikipedia.org/wiki/Noisy-channel_coding_theorem`

(See also the book Cover-Thomas: Elements of Information Theory, Section 7.7, pp. 199–205 for more details.)

A few hints:

The main idea of the proof is to randomly select $\lceil 2^{nR} \rceil$ codewords (for some $R < C$ according to the input distribution achieving the channel capacity C. Once the set of codewords is given there is an optimal decoding function belonging to it. However, for making the analysis more convenient a special decoding function is defined (which, though suboptimal, asymptotically still achieves the result we need). This is based on joint typicality. The set of jointly typical sequences (for some small $\varepsilon > 0$) is defined as

$$\begin{gathered} A_\varepsilon^{(n)} = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X}^n \times \mathcal{Y}^n \\ 2^{-n(H(X)+\varepsilon)} \leq p(\boldsymbol{x}) \leq 2^{-n(H(X)-\varepsilon)} \\ 2^{-n(H(Y)+\varepsilon)} \leq p(\boldsymbol{y}) \leq 2^{-n(H(Y)-\varepsilon)} \\ 2^{-n(H(X,Y)+\varepsilon)} \leq p(\boldsymbol{x}, \boldsymbol{y}) \leq 2^{-n(H(X,Y)-\varepsilon)}\}. \end{gathered}$$

When a codeword is sent and $\boldsymbol{y} \in \mathcal{Y}^n$ is received at the output, we decide on the codeword that is jointly typical with the received sequence, if there is a unique such codeword. Thus we make a mistake if either

1. the received word is not jointly typical with the codeword sent, or
2. there is another codeword, which is jointly typical with the received word.

It can be shown that the average value (over all codes) of the probability of both of these events goes to zero as n goes to infinity (when $R < C$ that we ensured). Thus there must exist a code for which the error probability tends to zero, while it has the required size, that is achieving the rate R. This proves the statement in the direct part of the Channel Coding Theorem.

Hamming codes

Constructing codes with rate close to channel capacity is a hard task and was out of reach for the first quite a few decades of information theory starting in 1948 with the publication of Shannon's fundamental paper that also contained the channel coding theorem. Here we just say a few words on one of the most basic (and most aesthetic) construction of error-correcting codes called Hamming codes. (These codes do not have rates close to channel capacity but at least show the basic ideas of error correction. We only mention the name of *turbo codes* defined in recent decades that achive rates close to channel capacity.)

A typical trick of error detection is to use a *parity bit.* Having a binary code in which every codeword contains an even number of ones, we will always be able to detect if at most one error occurred, because that will make the number of ones odd. With length n we have 2^{n-1} binary sequences with this property. Receiving an erroneous sequence, we will not be able to tell which codeword was sent even if we know that exactly one error occurred (that is only one bit was received erroneously). If, however, any two codewords would differ in at least 3 bits, then if we knew that at most one error occurred, we would be able to tell which codeword was sent as there is only one codeword within *Hamming distance* 1 from the received sequence. (The Hamming distance of two sequences is the number of positions where they differ.)

The Hamming code is a very simple and elegant construction of a binary code that has the ability of correcting one error: any two of its codewords differ at at least two positions. Here is how they are defined.

Let ℓ be a positive integer and $n = 2^\ell - 1$. Let H be the $n \times \ell$ matrix whose n columns are all the $2^\ell - 1$ nonzero binary sequences. A $0-1$ sequence $\boldsymbol{x}$ of length n is a codeword if and only $H\boldsymbol{x} = \mathbf{0}$. Thus, by elementary linear algebra, the words of this code form an $(n-\ell)$-dimensional subspace of the vector space $\{0,1\}^n$, the number of elements in it is thus $2^{n-\ell}$. And indeed, any two differs at at least three coordinates. This can be seen as follows. Let $\boldsymbol{c}$ be any codeword and $\boldsymbol{e}$ a vector of errors, that is the received word is $\boldsymbol{c}+\boldsymbol{e}$. Now this is another codeword if and only if $H(\boldsymbol{c}+\boldsymbol{e}) = H\boldsymbol{e} = \mathbf{0}$. That means that the modulo 2 sum of that many columns of H as many 1's are contained in $\boldsymbol{e}$ must be $\mathbf{0}$. Since no two columns of H are identical (and none of them is all-0), this requires that $\boldsymbol{e}$ have at least three 1's. So the minimum Hamming distance between two codewords of our code is at least three (in fact, it is three), thus the code can correct one error.

For $\ell = 1$ the above construction is not yet interesting, and it is not very sophisticated for $\ell = 2$ either: it contains two codewords of length $n = 3$, the all-0 and the all-1 vector. (Notice that these two words have Hamming distance three, indeed.) The simplest non-trivial Hamming code we obtain for $\ell = 3$. Then $n = 2^3 - 1 = 7$ and the number of codewords is $2^{(7-3)} = 16$.