Information Theory
lecture notes

1st lecture (September 5, 2016):

Information theory deals with the theoretical limits of information transmission. To formulate meaningful statements about such limits we need a way to *measure the amount of information*. A way to quantify this amount is to consider the number of binary characters needed to describe some information. This way we can say that the specification of a month in the year contains more information that the specification of a day of the week as the former needs 4, while the latter only 3 binary digits (bits).

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 it 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 $f : \mathcal{X} \to \mathcal{Y}^*$ if $\forall \mathbf{u}, \mathbf{v} \in \mathcal{X}^*, \mathbf{u} = u_1 u_2 \dots u_k, \mathbf{v} = v_1 v_2 \dots v_m, \mathbf{u} \neq \mathbf{v}$ 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 $1 \cdot 1/2 + 2 \cdot 1/4 + 3 \cdot 1/8 + 3 \cdot 1/8 = 1.75 < 2 = \log_2 4$.

Kraft-McMillan inequality

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

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

Proof. 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. □

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

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

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

Proof. Arrange the lengths in nondecreasing order, i.e., $l_1 \leq \ldots \leq l_r$. 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. □

2nd lecture (September 12, 2016):

Def. The *entropy* $H(P)$ of the probability distribution $P = (p_1, \ldots, p_r)$ 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 $h(p) = -p \log p - (1-p)\log(1-p)$.

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. This is justified by so-called coding theorems we will see later. 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 *Let us have an information source emitting symbol $x_i \in \mathcal{X}$ with probability $p(x_i) = p_i, (i = 1, \ldots, r)$. 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 $(p_1, \ldots, p_r)$. 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.

bf Jensen's inequality: Let $g : [a, b] \to R$ be a convex function, that is, one satisfying for every $x, y \in [a, b]$ and $\lambda \in (0, 1)$ that

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

. Then for any $x_1, \ldots, x_k \in [a, b]$ and non-negative reals $\alpha_1, \ldots, \alpha_k$ satisfying $\sum_{i=1}^{k} \alpha_i = 1$, we have

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

Moreover, if g is strictly convex, that is we always have strict inequality in the first inequality above, then equality in the second inequality can only occur if either $x_1 = \ldots = x_k$, or if all but one of the α_i's is 0 (while the exceptional one is necessarily equal to 1).

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 3. 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 the Corollary 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. □

Thinking about $\log 1/p_i$ as information content measured in bits suggests the proof of the following result.

Theorem 4 *For the information source of the previous theorem an s-ary prefix code with average codeword length less than $\frac{H(P)}{\log s} + 1$ exists.*

Proof. Let $l_i = \lceil \log_s(1/p_i) \rceil$. Then $\sum_{i=1}^{r} s^{-l_i} \leq \sum_{i=1}^{r} s^{-\log_s(1/p_i)} = \sum_{i=1}^{r} p_i = 1$, thus by Kraft's theorem an s-ary prefix code with lengths l_i exists. But with these lengths we have $\sum_{i=1}^{r} p_i l_i = \sum_{i=1}^{r} p_i \lceil \log_s(1/p_i) \rceil \leq \sum_{i=1}^{r} p_i (\log_s(1/p_i) + 1) = \frac{1}{\log s} \sum_{i=1}^{r} p_i \log(1/p_i) + \sum_{i=1}^{r} p_i = \frac{H(P)}{\log s} + 1$. □

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.

3rd lecture (September 19, 2016):

We give a second proof of Theorem 4 to introduce another code construction, called the **Shannon-Fano code**:

We assume $p_1 \geq p_2 \geq \ldots \geq p_n$. Let $w_1 = 0$ and $w_j = \sum_{i=1}^{j-1} p_i$. Let the codeword $f(x_j)$ be the s-ary representation of the number w_j (which is always in $[0, 1)$ 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 the code being prefix.

This definition gives 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 is valid. 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

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

□

For $s = 2$, $r = 3, p_1 = p_2 = p_3 = 1/3$ the construction in the first proof of Theorem 4 gives average length 2. This is clearly not optimal, as the code $(0, 10, 11)$ would work and has average length $1/3$ less. So the question of how to find the optimal average length code comes up. This is answered by constructing the so-called Huffman code.

Huffman code

We are considering the binary case ($s = 2$). Assume $p_1 \geq \ldots \geq p_r$, $p_i = p(x_i)$ and having an optimal now binary code $C = (f(x_1), \ldots, f(x_r))$, $l_i := |f(x_i)|$.

Observe:
We may assume
1. $l_1 \leq l_2 \leq \ldots \leq l_r$; otherwise we can exchange codewords without increasing average length.
2. $l_n = l_{n-1}$ and $f(x_n)$, $f(x_{n-1})$ differ only in the last digit.
3. Cutting the last digit of these two codewords we obtain an optimal binary code for the distribution $(p_1, p_2, \ldots, p_{n-2}, p_{n-1} + p_n)$.

From these three the optimal code construction is immediate: add two smallest probabilities iteratively until only one value 1 probability remains. Give this probability 1 event the empty string as codeword, this is trivially best possible attaining average length 0. One step before there are two distinct probability values. Give these the (sub)words 0 and 1 (these length-1 codewords can be considered as continuations of the empty word) and then follow the previous "adding up two probabilities" process backwards and always put a 0 and a 1 at the end of the corresponding codeword.

Example:

$$P = (0.25, 0.14, 0.13, 0.12, 0.11, 0.10, 0.10, 0.05)$$

The in-between distributions:

$(0.25, 0.15 = 0.10 + 0.05, 0.14, 0.13, 0.12, 0.11, 0.10)$;
$(0.25, 0.21, 0.15, 0.14, 0.13, 0.12)$; $(0.25, 0.25, 0.21, 0.15, 0.14)$;
$(0.29, 0.25, 0.25, 0.21)$; $(0.46, 0.29, 0.25)$; $(0.54, 0.46)$; (1).

And the code obtained(writing it backwards for each stage of the construction:
$(\emptyset)$; $(0, 1)$; $(1, 00, 01)$; $(00, 01, 10, 11)$;
$(01, 10, 11, 000, 001)$; $(01, 11, 000, 001, 100, 101)$;
$(01, 000, 001, 100, 101, 110, 111)$, and finally

$$(01, 001, 100, 101, 110, 111, 0000, 0001).$$

Exercises:
1. Let $P = (p_1, p_2, p_3, p_4)$ be a probability distribution for which both the codes $(00, 01, 10, 11)$ and $(0, 10, 110, 1110)$ attains optimal average length. We know $p_1 \geq p_2 \geq p_3 \geq p_4$ and that $p_3 = \frac{1}{6}$. Determine the distribution P, that is, determine the other p_i values as well.

2. Let $p_1 \leq \ldots \leq p_r$, $r = 2^k$ and $k > 1$. Assume that the 2^k binary sequences of length k give optimal average length when giving an optimal average length prefix encoding of the source characters $x_1, \ldots, x_r$ that occur with probability $p_1, \ldots, p_r$, respectively. What is the maximum possible value of p_r among these circumstances?

4th lecture (September 26, 2016):

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).

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 contains separately. And equality means that they "do not contain information about each other", that is, they are independent.)

Proof of a): $0 \leq H(X)$ 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. □

Conditional entropy is defined as:

$$H(X|Y) = \sum_y p(y) H(X|Y = y) =$$

$$= -\sum_y p(y) \sum_x p(x|y) \log p(x|y) =$$

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

Some properties of conditional entropy are proven next.

Theorem 6 *a)*

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

b)

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

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. □

Exercises:

We have seen Maria's solution of Exercise 1 from last week.

For Exericise 2 of last week, we proved that for $r = 2^2 = 4$, the maximum possible value of p_r is $\frac{2}{5}$. The general case remained a homework. (It is suggested that the problem is solved first for $r = 2^3 = 8$.)

3. Let us have two dice both of which having two sides with 1 dot, two sides with 2 dots, and two sides with 3 dots on it. (When rolling such dice the chance for rolling a 1 is equal to the chance of rolling a 2 and same for rolling a 3: each has probability $\frac{1}{3}$. We roll the two dice together and consider the random variable X that is the sume of the two rolled values. Encode the possible outcomes of X in an optimal way, that is find the corresponding Huffman code.

5th lecture (October 3, 2016):

We proved the *Chain rule*:

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

Proof: 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 $X=(X_1,\dots,X_{n-1})$ in the form $H(X,Y) = H(X)+H(Y|X)$. It gives

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

□

We also proved a consequence of Theorem 6

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

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

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

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

Also from Theorem 6 we can 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

---

Exercises:

We solved Exercise 2 in the general case proving that the asked maximum value is $\frac{2}{r+1}$. We have also discussed the solution of Exercise 3.

We have solved the following exercise.

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 found that $H(X)=h(p)$ (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). We noticed that $H(X|Y)=h(p)=H(X)$ means that X and Y are also independent.

6th lecture (October 10, 2016):

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 link "Examples for the Lempel-Ziv data compression algorithms LZ77 and LZ78". 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 link "Examples for the Lempel-Ziv data compression algorithms LZ77 and LZ78".

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 inddex 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 link "Example for the Lempel-Ziv-Welch data compression algorithm".

7th lecture (October 15, 2016):

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 this way:

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

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

If the source is stationary and memoryless, then $H(X_1, X_2, \ldots, X_n) = H(X_1) + H(X_2) + \ldots + H(X_n)$ by the independence of the X_i's (i.e., by the memoryless property) and $H(X_1) + H(X_2) + \ldots + H(X_n) = nH(X_1)$ by the source being stationary, 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 $\lim_{n\to\infty} H(X_n|X_1, \ldots, X_{n-1})$ 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 $\lim_{n\to\infty} H(X_n|X_1, \ldots, X_{n-1}) = H(X_n|X_{n-1}) = h(0.9) < 1$.

Proof. By the source being stationary, we have

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

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

From the Chain rule we can write

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

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 $a_n := H(X_n|X_1, \ldots, X_{n-1})$ the statement follows. □

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

Markov chains, Markov source

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

The entropy of a homogenous Markov chain Z is $H(Z_2|Z_1)$, cf. the Remark above.

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 $H(Z_2|Z_1)$

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 $\mathbf{p}$ the probability distribution $(p_1, \ldots, p_r)$ for which $\mathbf{p}A = \mathbf{p}^T$.

8th lecture (October 17, 2016):

Encoding a stationary source

The following theorem shows that the entropy of a stationary source is in general the theoretical limit of how good average length can we achieve with a uniquely decodable code that encodes k-length blocks.

Theorem 8 *Let $f : \mathcal{X}^k \to \mathcal{Y}^*$ be a UD code of the stationary source $X = X_1, X_2, \ldots$. Then the per letter average length L of the code satisfies*

$$L \geq \frac{H(X)}{\log s},$$

(where $s = |\mathcal{Y}|$). On the other hand, for any $\varepsilon > 0$ and k large enough there exists a prefix code f for which

$$L \leq \frac{H(X)}{\log s} + \varepsilon$$

.

Proof. By Theorem 3 and the observation that $\frac{1}{k}H(X_1, \ldots, X_k)$ is non-increasing, we have

$$L \geq \frac{1}{k}\frac{H(X_1, \ldots, X_k)}{\log s} \geq \frac{H(X)}{\log s}.$$

Let now k_0 be large enough for both

$$\frac{1}{k_0}H(X_1, \ldots, X_{k_0}) - H(X) < \frac{\varepsilon}{2}\log s; \quad \frac{1}{k_0} < \frac{\varepsilon}{2}.$$

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

$$L' = E(f(X_1, \ldots, X_{k_0})) \leq \frac{H(X_1, \ldots, X_{k_0})}{\log s} + 1,$$

where L' stands for (not yet "per letter") average length.
Such a code satisfies our needs as for that we have $L = \frac{1}{k_0}E(f(X_1, \ldots, X_{k_0})) < \frac{H(X_1,\ldots,X_{k_0})}{\log s} + \frac{1}{k_0} < \frac{H(X)}{\log s} + \frac{\varepsilon}{2} + \frac{1}{k_0} < \frac{H(X)}{\log s} + \varepsilon$. This completes the proof. □

Source coding with negligible error probability

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.

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

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

Remark: 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.

9th lecture (October 24, 2016):

(Another) Remark: 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 $\boldsymbol{x} \in A$ implies

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

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

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

Let $N(k,\varepsilon)$ be the number N for which if $\boldsymbol{x}_1, \ldots, x_N$ are the N largest probability k-length output sequences of the source X, we have $\sum_{i=1} p(\boldsymbol{x}_i) > 1-\varepsilon$. The quantity relevant for us is $\frac{\log N(k,\varepsilon)}{k}$. The main result here is, that for the fairly general class of information stable 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 qualitative characterization of the best encoding rate one can achieve. This verifies the intuition that interprets the entropy of the source as a measure of information content in the source variables.

Theorem 9 *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 output sequences and for every $\delta \in (0,1)$ define

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

Then by

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

we have

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

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

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

and so

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

Since δ can be arbitrarily small, this also implies

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

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

$$\frac{1-\varepsilon}{2} < P(A_{k,\delta} \cap B_{k,\varepsilon}) \leq |B_{k,\varepsilon}| \max_{\boldsymbol{x} \in A_{k,\delta}} p(\boldsymbol{x}) \leq |B_{k,\varepsilon}| 2^{-k(H(X)-\delta)}$$

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

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. □

10th lecture (November 7, 2016):

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

Theorem 10 *Let X be a stationary source which is also information stable. 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 by 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 $\lim_{k\to\infty} \frac{1}{k} \log_s N(k,\varepsilon) = \frac{H(X)}{\log s}$.

The last coding theorem can be looked at from another point of view. Namely, we can say that we consider, $R = \frac{m_k}{k} \log s$ given, and ask about the error probability of the best possible code with these parameters. Then the theorem implies that the smallest possible error probability we can achieve as k goes to infinity (while $R = \frac{m_k}{k} \log s$ is fixed) can be arbitrarily close to 0 if $R > H(X)$, while for $R < H(X)$ it will tend to 1. (The latter means that the code is impossible to use with a tolerable error probability.)

Remark: It is also known that if the source is stationary and memoryless, then the above error probability will tend to 0 exponentially as a function $P_e(k,R)$ of the length k when $R > H(X)$. Similarly, for $R < H(X)$ the difference $1 - P_e(k,R)$ tends to 0 exponentially fast.

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. This way the measuring would belong to n, that is, it might depend on n, but since our X_i's are identically distributed, it will actually not. In fact, since our X_i's are identically distributed we can write

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

for the above distortion measure and this expression has no appearance of n in it. (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.) This condition 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 $B_i = \{x : a+(i-1)\frac{b-a}{N} \leq x \leq a + i\frac{b-a}{N}\}$ 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.)

11th lecture (November 14, 2016):

Distortion of the uniform quantizer

The simplest quantizer is the uniform quantizer, we investigate it a bit closer. For simplicity we assume 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

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

(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 11 *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 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. □

Differential entropy and quantization

If we quantize the stationary source $\mathbf{X}$ with quantizer $Q(.)$ then we get the sequence of random variables $Q(X_1), Q(X_2), \ldots$ which can be considered itself as a stationary source. If after quantization we want to encode the source into a binary sequence, we can certainly do it by using $\log_2 n$ bits for every $Q(X_i)$ (here N is the number of quantization levels). This is, however, not optimal if $Q(X_i)$ is not uniformly distributed. Rather, we could compress our encoding to $H(Q(X_i))$ bits approximately. How does this relate to the information content of the original continuous source. To be able for such a comparison, we would need some entropy-like measure of the continuous source. This exists and called differential entropy. It has analogous properties to the entropy of a discrete random variable. Those we will not prove, simply accept the intuition that it expresses a quantity we can interpret as the information content of the continuous source.

Def. The differential entropy of a random variable with density function $f(x)$ is defined as

$$h(X) = h(f) = \int_S f(x) \log f(x)dx,$$

where S is the support set of the random variable (i.e, the set on which it takes its values). We use the notations $h(X)$ and $h(f)$ interchangeably.

Now we prove the relation between the entropy of the uniformly quantized version of a random variable with its original differential entropy under some natural assumptions on its density function.

Theorem 12 *Let X be a continuous random variable with density function f, which is not positive outside the interval $[-A, A]$ and is continuous within this interval. Assume also that the integral defining $h(f)$ (on $S = [-A, A]$ is finite. Then for the entropy $H(Q_N(X))$ of the N-level uniformly quantized version $Q_N(X)$ of X we have the following asymptotic equality:*

$$\lim_{N\to\infty} (H(Q_N(X)) + \log q_N) = h(f),$$

where $q_N = \frac{2A}{N}$.

Thus the theorem states that for large N we can approximate $H(Q_N(X))$ with $h(f) - \log q_N$.

For the proof we will use Lagrange's mean value theorem that states the following. Let $f : [a, b] \to R$ be a continuous function on the closed interval $[a, b]$, and differentiable on the open interval (a, b), where $a < b$. Then there exists some c in (a, b) such that

$$f'(c) = \frac{f(b) - f(a)}{b - a}$$

.

Now we are ready to prove the above theorem.

Proof.: We use the notation of the previous proof. The Lagrange mean value theorem applied to the differentiable function $F(x) := \int_{-\infty}^{x} f(x)dx$ implies the existence of real numbers $\xi_{N,i} \in (y_{N,i-1}, y_{N,i}]$ that satisfy

$$\int_{y_{N,i-1}}^{y_{N,i}} f(x)dx = (y_{N.i} - y_{N,i-1})f(\xi_{N,i}) = q_N f(\xi_{N,i}).$$

Using this we can write

$$H(Q_N(X)) = -\sum_{i=1}^{N} P(Q_N(X) = x_{N,i}) \log(P(Q_N(X) = x_{N,i})) =$$

$$\sum_{i=1}^{N} \left(\int_{y_{N,i-1}}^{y_{N,i}} f(x)dx \log \int_{y_{N,i-1}}^{y_{N,i}} f(x)dx \right) =$$

$$-\sum_{i=1}^{N} q_N f(\xi_{N,i}) \log(q_N f(\xi_{N,i})) =$$

$$-\sum_{i=1}^{N} q_N f(\xi_{N,i}) \log q_N - \sum_{i=1}^{N} q_N f(\xi_{N,i}) \log(f(\xi_{N,i})) =$$

$$(-\log q_N) \sum_{i=1}^{N} \int_{y_{N,i-1}}^{y_{N,i}} f(x)dx - \sum_{i=1}^{N} q_N f(\xi_{N,i}) \log(f(\xi_{N,i})) =$$

$$-logq_N - \sum_{i=1}^{N} q_N f(\xi_{N,i}) \log(f(\xi_{N,i})).$$

Taking $N \to \infty$ for the second term of the last line we get

$$\lim_{N\to\infty} -\sum_{i=1}^{N} q_N f(\xi_{N,i}) \log(f(\xi_{N,i})) = -\int_{-A}^{A} f(x) \log f(x)dx,$$

since the left hand side is a Riemann approximation sum of the integral on the right hand side and thus its limit is equal to that integral.

So we obtained

$$H(Q_N(X)) = -logq_N - \int_{-A}^{A} f(x) \log f(x) dx = h(f) - \log q_N$$

as stated. □

12th lecture (November 21, 2016):

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.

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

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

2. "Z" channel: $\mathcal{X} = \mathcal{Y} = \{0,1\}$, $W(0|0) = 1$, $W(1|0) = 0$, $W(1|1) = 1-p$ $W(0|1) = p$.

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

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

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 $\{\mathbf{c_1}, \ldots, \mathbf{c_M}\}$.
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. (Certain discussions define codes as the pair of functions (f, φ).)

The probability of error if the message m_i, that is the codeword $\mathbf{c_i}$, was sent is

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

where $y^{(r)}$ and $c_i^{(r)}$ denote the rth character in the sequences $\mathbf{y}$ and $\mathbf{c_i}$, 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.)

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.

13th lecture (November 28, 2016):

Next we 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.

$$I(X^n, Y^n) = H(Y^n) - H(Y^n|X^n)$$

$$= H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_1, ..., Y_{i-1}, X^n)$$

$$= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \leq \sum_{i=1}^{n} H(Y_i) - \sum_{i=1}^{n} H(Y_i|X_i)$$

$$= \sum_{i=1}^{n} I(X_i, Y_i) \leq nC.$$

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 $Y_1, ..., Y_{i-1}, X_1, \ldots X_n$ and thus the used equality of conditional entropies. (The other relations should be clear: the first and fourth equalities follow 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 $M = \lceil 2^{nR} \rceil$ 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

$$nR \leq H(U) = H(U|Y^n) + I(U, Y^n) = I(U, Y^n) = I(X^n, Y^n)$$

$$\leq \sum_{i=1}^{n} I(X_i, Y_i) \leq nC.$$

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 ono-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. □

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 another lemma, known

as Fano's inequality. This will help us to bound $H(U|Y^n)$ from above in the setting when we cannot simply say that it is zero.

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, a random message, 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 $H(U|E,\hat{U}) = P(E = 0)H(U|\hat{U}, E = 0) + (1 - \bar{P}_e)H(U|\hat{U}, E = 1) \leq (1 - \bar{P}_e)0 + \bar{P}_e \log 2^{nR}$. □

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 Lemma 3, because $\hat{U}$ is 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 $H(U) - (U|Y^n) = H(X) - H(X^n|Y^n)$. (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 on-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. □

14th lecture (December 5, 2016):

The proof of the direct part of the channel coding theorem 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.)

Here we quote only 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

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

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 expected 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 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 $2^{(7-3)} = 16$.