§ 6. Variable-length Lossless Compression

The principal engineering goal of compression is to represent a given sequence $a_1, a_2, \ldots, a_n$ produced by a source as a sequence of bits of minimal possible length. Of course, reducing the number of bits is generally impossible, unless the source imposes certain restrictions. That is if only a small subset of all sequences actually occur in practice. Is it so for real world sources?

As a simple demonstration, one may take two English novels and compute empirical frequencies of each letter. It will turn out to be the same for both novels (approximately). Thus, we can see that there is some underlying structure in English texts restricting possible output sequences. The structure goes beyond empirical frequencies of course, as further experimentation (involving digrams, word frequencies etc) may reveal. Thus, the main reason for the possibility of data compression is the *experimental (empirical) law: real-world sources produce very restricted sets of sequences.*

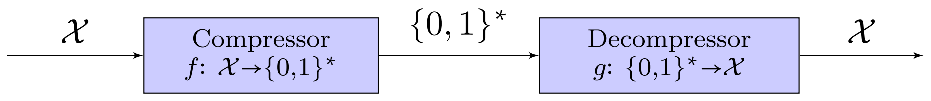
How do we model these restrictions? Further experimentation (with language, music, images) reveals that frequently, the structure may be well described if we assume that sequences are generated probabilistically. This is one of the main contributions of Shannon: *another empirical law states that real-world sources may be described probabilistically with increasing precision starting from i.i.d., 1-st order Markov, 2-nd order Markov etc.* Note that sometimes one needs to find an appropriate basis in which this "law" holds – this is the case of images (i.e. rasterized sequence of pixels won't appear to have local probabilistic laws, because of forgetting the 2-D constraints; wavelets and local Fourier transform provide much better bases).[1]

So our initial investigation will be about representing one random variable $X \sim P_X$ in terms of bits efficiently. Types of compression:

- Lossy
  $X \to W \to \hat{X}$ s.t. $\mathbb{E}[(X - \hat{X})^2] \le$ distortion.
- Lossless
  $P(X \ne \hat{X}) = 0$. variable-length code, uniquely decodable codes, prefix codes, Huffman codes
- Almost lossless
  $P(X \ne \hat{X}) \le \epsilon$. fixed-length codes

6.1 Variable-length, lossless, optimal compressor

Coding paradigm:


$\mathcal{X}$
Compressor
f: $\mathcal{X} \to \{0,1\}^*$
$\{0,1\}^*$
Decompressor
g: $\{0,1\}^* \to \mathcal{X}$
$\mathcal{X}$

[1]Of course, one should not take these "laws" too far. In regards to language modeling, (finite-state) Markov assumption is too simplistic to truly generate all proper sentences, cf. Chomsky [Cho56].

Remark 6.1.

- Codeword: $f(x) \in \{0,1\}^*$; Codebook: $\{f(x) : x \in \mathcal{X}\} \subset \{0,1\}^*$
- Since $\{0,1\}^* = \{\varnothing, 0, 1, 00, 01, \dots\}$ is countable, lossless compression is only possible for discrete R.V.;
- if we want $g \circ f = 1_{\mathcal{X}}$ (lossless), then f must be injective;
- relabel $\mathcal{X}$ such that $\mathcal{X} = \mathbb{N} = \{1, 2, \dots\}$ and order the pmf decreasingly: $P_X(i) \geq P_X(i+1)$.

Length function:

$$l : \{0,1\}^* \to \mathbb{N}$$

e.g., $l(01001) = 5$.

Objectives: Find the best compressor f to minimize

$$\mathbb{E}[l(f(X))]$$

$$\sup l(f(X))$$

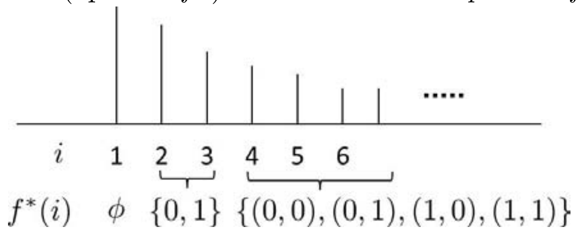
$$\text{median } l(f(X))$$

It turns out that there is a compressor f^* that minimizes all together!

Main idea: Assign longer codewords to less likely symbols, and reserve the shorter codewords for more probable symbols.

Aside: It is useful to introduce the partial order of *stochastic dominance*: For real-valued RV X and Y, we say Y stochastically dominates (or, is stochastically larger than) X, denoted by $X \overset{\text{st.}}{\leq} Y$, if $\mathbb{P}[Y \leq t] \leq \mathbb{P}[X \leq t]$ for all $t \in \mathbb{R}$. In other words, $X \overset{\text{st.}}{\leq} Y$ iff the CDF of X is larger than the CDF of Y pointwise. In particular, if X is dominated by Y stochastically, so are their means, medians, supremum, etc.

Theorem 6.1 (optimal f^*). *Consider the compressor f^* defined by*



Then

1. *length of codeword:*

$$l(f^*(i)) = \lfloor \log_2 i \rfloor$$

2. *$l(f^*(X))$ is stochastically the smallest: for any lossless f,*

$$l(f^*(X)) \overset{\text{st.}}{\leq} l(f(X))$$

i.e., for any k, $\mathbb{P}[l(f(X)) \leq k] \leq \mathbb{P}[l(f^*(X)) \leq k]$.

Proof. Note that

$$|A_k| \triangleq |\{x : l(f(x)) \le k| \le \sum_{i=0}^{k} 2^i = 2^{k+1} - 1 = |\{x : l(f^*(x)) \le k| \triangleq |A_k^*|.$$

where the inequality is because of f is lossless and $|A_k|$ exceeds the total number of binary strings of length less than k. Then

$$\mathbb{P}[l(f(X)) \le k] = \sum_{x \in A_k} P_X(x) \le \sum_{x \in A_k^*} P_X(x) = \mathbb{P}[l(f^*(X)) \le k],$$

since $|A_k| \le |A_k^*|$ and A_k^* contains all $2^{k+1} - 1$ most likely symbols. □

The following lemma is useful in bounding the expected code length of f^*. It says if the random variable is integer-valued, then its entropy can be controlled using its mean.

Lemma 6.1. *For any $Z \in \mathbb{N}$ s.t. $\mathbb{E}[Z] < \infty$, $H(Z) \le \mathbb{E}[Z] h(\frac{1}{\mathbb{E}[Z]})$, where $h(\cdot)$ is the binary entropy function.*

Theorem 6.2 (Optimal average code length: exact expression)**.** *Suppose $X \in \mathbb{N}$ and $P_X(1) \ge P_X(2) \dots$. Then*

$$\mathbb{E}[l(f^*(X))] = \sum_{k=1}^{\infty} \mathbb{P}[X \ge 2^k].$$

Proof. Recall that expectation of $U \in \mathbb{Z}_+$ can be written as $\mathbb{E}[U] = \sum_{k \ge 1} \mathbb{P}[U \ge k]$. Then by Theorem 6.1, $\mathbb{E}[l(f^*(X))] = \mathbb{E}[\lfloor \log_2 X \rfloor] = \sum_{k \ge 1} \mathbb{P}[\lfloor \log_2 X \rfloor \ge k] = \sum_{k \ge 1} \mathbb{P}[\log_2 X \ge k]$. □

Theorem 6.3 (Optimal average code length v.s. entropy)**.**

$$H(X)\ \texttt{bits} - \log_2[e(H(X)+1)] \le \mathbb{E}[l(f^*(X))] \le H(X)\ \texttt{bits}$$

Note: Theorem 6.3 is the first example of a coding theorem, which relates the fundamental limit $\mathbb{E}[l(f^*(X))]$ (operational quantity) to the entropy $H(X)$ (information measure).

Proof. Define $L(X) = l(f^*(X))$.

RHS: observe that since the pmf are ordered decreasingly by assumption, $P_X(m) \le 1/m$, so $L(m) \le \log_2 m \le \log_2(1/P_X(m))$, take exp., $\mathbb{E}[L(X)] \le H(X)$.

LHS:

$$\begin{aligned}
H(X) = H(X,L) &= H(X|L) + H(L) \\
&\le \mathbb{E}[L] + h\left(\frac{1}{1+\mathbb{E}[L]}\right)(1+\mathbb{E}[L]) && \text{(Lemma 6.1)} \\
&= \mathbb{E}[L] + \log_2(1+\mathbb{E}[L]) + \mathbb{E}[L]\log\left(1+\frac{1}{\mathbb{E}[L]}\right) \\
&\le \mathbb{E}[L] + \log_2(1+\mathbb{E}[L]) + \log_2 e && (x\log(1+1/x) \le \log e, \forall x > 0) \\
&\le \mathbb{E}[L] + \log(e(1+H(X))) && \text{(by RHS)}
\end{aligned}$$

where we have used $H(X|L=k) \le k\,\texttt{bits}$, since given $l(f^*(X))) = k$, X has at most 2^k choices. □

Note: (Memoryless source) If $X = S^n$ is an i.i.d. sequence, then

$$nH(S) \geq \mathbb{E}[l(f^*(S^n))] \geq nH(S) - \log n + O(1).$$

For iid sources, the exact behavior is found in [SV11, Theorem 4] as:

$$\mathbb{E}[\ell(f^*(S^n))] = nH(S) - \frac{1}{2}\log n + O(1)\,,$$

unless the source is uniform (in which case it is $nH(S) + O(1)$.

Theorem 6.3 relates the *mean* of $l(f^*(X)) \leq k$ to that of $\log_2 \frac{1}{P_X(X)}$ (entropy). The next result relates their *CDFs*.

Theorem 6.4 (Code length distribution of f^*)**.** $\forall \tau > 0, k \in \mathbb{Z}_+$,

$$\mathbb{P}\left[\log_2 \frac{1}{P_X(X)} \leq k\right] \leq \mathbb{P}\left[l(f^*(X)) \leq k\right] \leq \mathbb{P}\left[\log_2 \frac{1}{P_X(X)} \leq k + \tau\right] + 2^{-\tau+1}$$

Proof. LHS: easy, use $P_X(m) \leq 1/m$. Then similarly as in Theorem 6.3, $L(m) = \lfloor \log_2 m \rfloor \leq \log_2 m \leq \log_2 \frac{1}{P_X(m)}$. Hence $L(X) \leq \log_2 \frac{1}{P_X(X)}$ a.s.
RHS: (truncation)

$$\begin{aligned}
\mathbb{P}[L \leq k] &= \mathbb{P}\left[L \leq k, \log_2 \frac{1}{P_X(X)} \leq k + \tau\right] + \mathbb{P}\left[L \leq k, \log_2 \frac{1}{P_X(X)} > k + \tau\right] \\
&\leq \mathbb{P}\left[\log_2 \frac{1}{P_X(X)} \leq k + \tau\right] + \sum_{x \in \mathcal{X}} P_X(x) \mathbf{1}_{\{l(f^*(x)) \leq k\}} \mathbf{1}_{\{P_X(x) \leq 2^{-k-\tau}\}} \\
&\leq \mathbb{P}\left[\log_2 \frac{1}{P_X(X)} \leq k + \tau\right] + (2^{k+1} - 1) \cdot 2^{-k-\tau}
\end{aligned}$$

□

So far our discussion applies to an arbitrary random variable X. Next we consider the source as a random process $(S_1, S_2, \ldots)$ and introduce blocklength. We apply our results to $X = S^n$: the first n symbols. The following corollary states that the limiting behavior of $l(f^*(S^n))$ and $\log \frac{1}{P_{S^n}(S^n)}$ always coincide.

Corollary 6.1. *Let $(S_1, S_2, \ldots)$ be some random process and U be some random variable. Then*

$$\frac{1}{n}\log_2 \frac{1}{P_{S^n}(S^n)} \xrightarrow{\text{D}} U \quad \Leftrightarrow \quad \frac{1}{n} l(f^*(S^n)) \xrightarrow{\text{D}} U \tag{6.1}$$

and

$$\frac{1}{\sqrt{n}}\left(\log_2 \frac{1}{P_{S^n}(S^n)} - H(S^n)\right) \xrightarrow{\text{D}} V \quad \Leftrightarrow \quad \frac{1}{\sqrt{n}}(l(f^*(S^n)) - H(S^n)) \xrightarrow{\text{D}} V \tag{6.2}$$

Proof. The proof is simply logic. First recall: convergence in distribution is equivalent to convergence of CDF at any continuity point. $U_n \xrightarrow{\text{D}} U \Leftrightarrow \mathbb{P}[U_n \leq u] \to \mathbb{P}[U \leq u]$ for all u at which point the CDF of U is continuous (i.e., not an atom of U).

Apply Theorem 6.4 with $k = un$ and $\tau = \sqrt{n}$:

$$\mathbb{P}\left[\frac{1}{n}\log_2 \frac{1}{P_X(X)} \leq u\right] \leq \mathbb{P}\left[\frac{1}{n} l(f^*(X)) \leq u\right] \leq \mathbb{P}\left[\frac{1}{n}\log_2 \frac{1}{P_X(X)} \leq u + \frac{1}{\sqrt{n}}\right] + 2^{-\sqrt{n}+1}.$$

Apply Theorem 6.4 with $k = H(S^n) + \sqrt{n}u$ and $\tau = n^{1/4}$:

$$\mathbb{P}\left[\frac{1}{\sqrt{n}}\left(\log\frac{1}{P_{S^n}(S^n)} - H(S^n)\right) \le u\right] \le \mathbb{P}\left[\frac{l(f^*(S^n)) - H(S^n)}{\sqrt{n}} \le u\right]$$
$$\le \mathbb{P}\left[\frac{1}{\sqrt{n}}\left(\log\frac{1}{P_{S^n}(S^n)} - H(S^n)\right) \le u + n^{-1/4}\right] + 2^{-n^{1/4}+1} \square$$

Remark 6.2 (Memoryless source)**.** Now let us consider S^n that are i.i.d. Then $\log\frac{1}{P_{S^n}(S^n)} = \sum_{i=1}^n \log\frac{1}{P_S(S_i)}$.

1. By the Law of Large Numbers (LLN), we know that $\frac{1}{n}\log\frac{1}{P_{S^n}(S^n)} \xrightarrow{\mathbb{P}} \mathbb{E}\log\frac{1}{P_S(S)} = H(S)$. Therefore in (6.1) the limiting distribution U is degenerate, i.e., $U = H(S)$, and we have $\frac{1}{n}l(f^*(S^n)) \xrightarrow{\mathbb{P}} \mathbb{E}\log\frac{1}{P_S(S)} = H(S)$. [Note: convergence in distribution to a constant $\Leftrightarrow$ convergence in probability to a constant]

2. By the Central Limit Theorem (CLT), if $V(S) \triangleq \mathrm{Var}\left[\log\frac{1}{P_S(S)}\right] < \infty$,[2] then we know that V in (6.2) is Gaussian, i.e.,

$$\frac{1}{\sqrt{nV(S)}}\left(\log\frac{1}{P_{S^n}(S^n)} - nH(S)\right) \xrightarrow{\mathrm{D}} \mathcal{N}(0,1).$$

   Consequently, we have the following Gaussian approximation for the probability law of the optimal code length

$$\frac{1}{\sqrt{nV(S)}}(l(f^*(S^n)) - nH(S)) \xrightarrow{\mathrm{D}} \mathcal{N}(0,1),$$

   or, in shorthand,

$$l(f^*(S^n)) \sim nH(S) + \sqrt{nV(S)}\mathcal{N}(0,1) \text{ in distribution.}$$

   Gaussian approximation tells us the speed of $\frac{1}{n}l(f^*(S^n))$ to entropy and give us a good approximation at finite n. In the next section we apply our bounds to approximate the distribution of $\ell(f^*(S^n))$ in a concrete example:

6.1.1 Compressing iid ternary source

Consider the source outputing n ternary letters each independent and distributed as

$$P_X = [.445 \quad .445 \quad .11].$$

For iid source it can be shown

$$\mathbb{E}[\ell(f^*(X^n))] = nH(X) - \frac{1}{2}\log(2\pi eVn) + O(1),$$

where we denoted the *varentropy* of X by

$$V(X) \triangleq \mathrm{Var}\left[\log\frac{1}{P_X(X)}\right].$$

[2] V is often known as the *varentropy* of S.

The Gaussian approximation to $\ell(f^*(X))$ is defined as

$$nH(X) - \frac{1}{2}\log 2\pi e V n + \sqrt{nV} Z\,,$$

where $Z \sim \mathcal{N}(0,1)$.

On Fig. 6.1, 6.2, 6.3 we plot the distribution of the length of the optimal compressor for different values of n and compare with the Gaussian approximation.

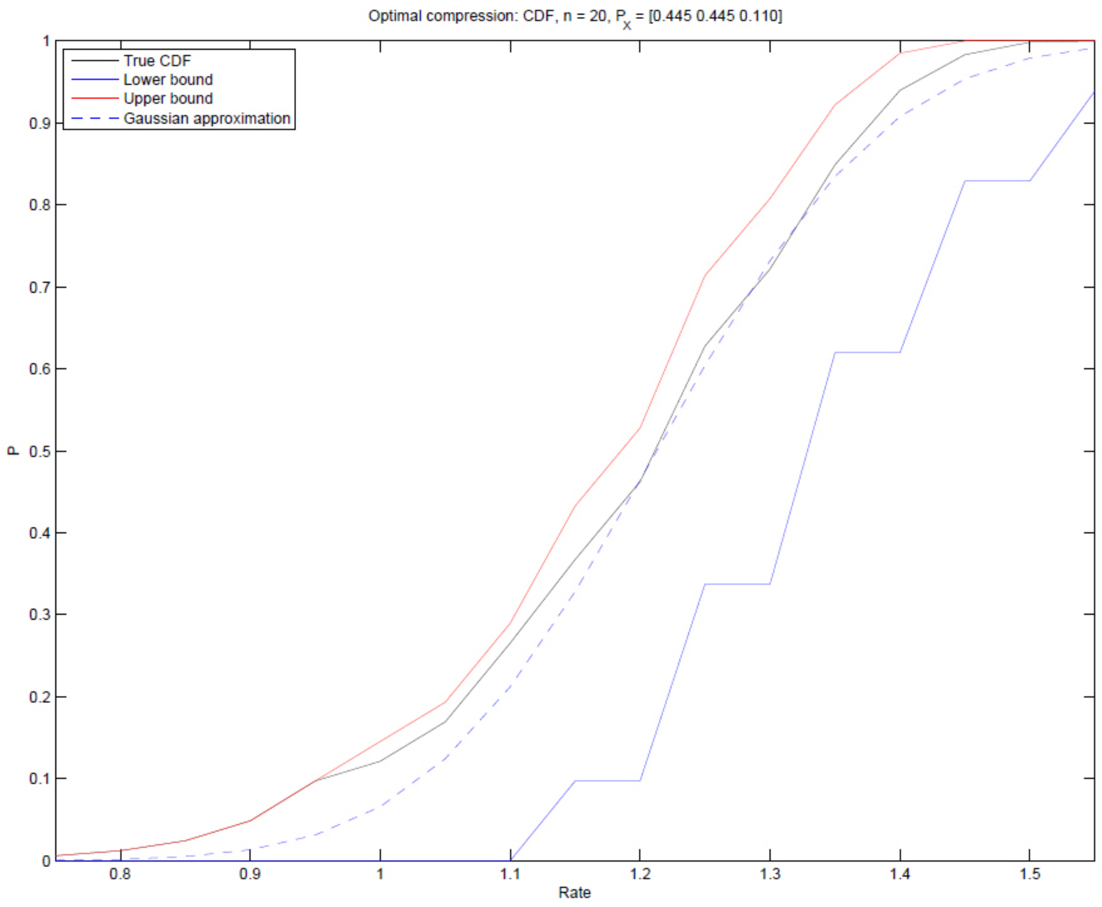
Upper/lower bounds on the expectation:

$$H(X^n) - \log(H(X^n)+1) - \log e \le \mathbb{E}[\ell(f^*(X^n))] \le H(X^n)$$

Here are the numbers for different n

$n = 20$	21.5	$\le$	24.3	$\le$	27.8
$n = 100$	130.4	$\le$	134.4	$\le$	139.0
$n = 500$	684.1	$\le$	689.2	$\le$	695.0

In all cases above $\mathbb{E}[\ell(f^*(X))]$ is close to a midpoint between the two.



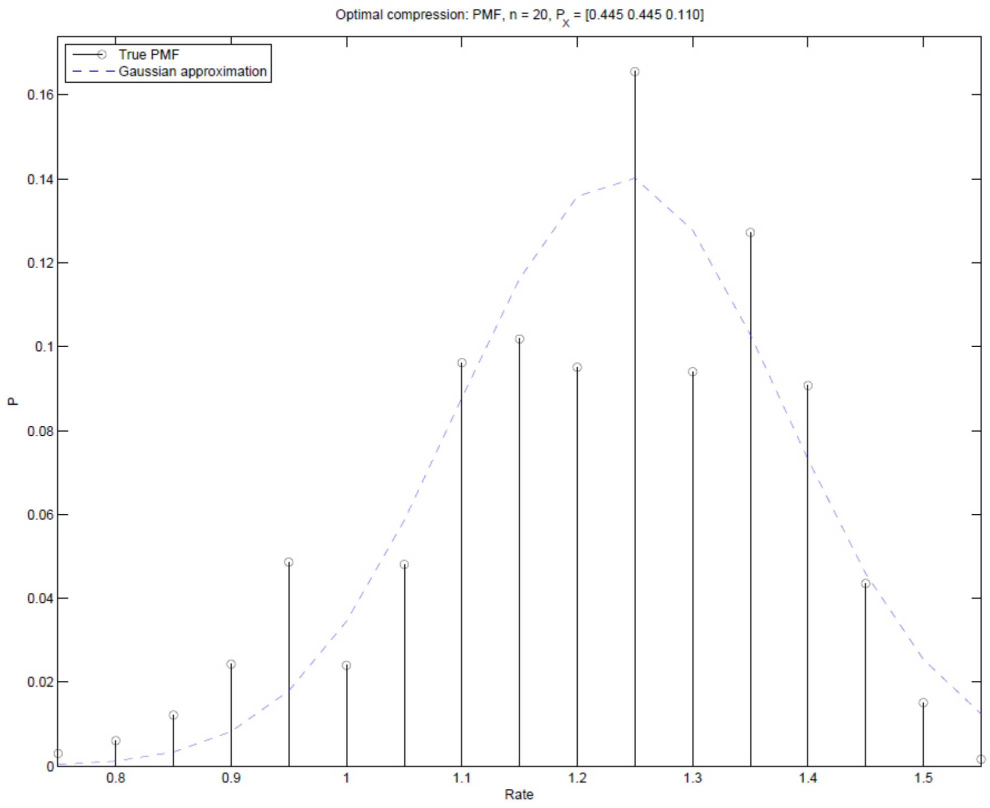
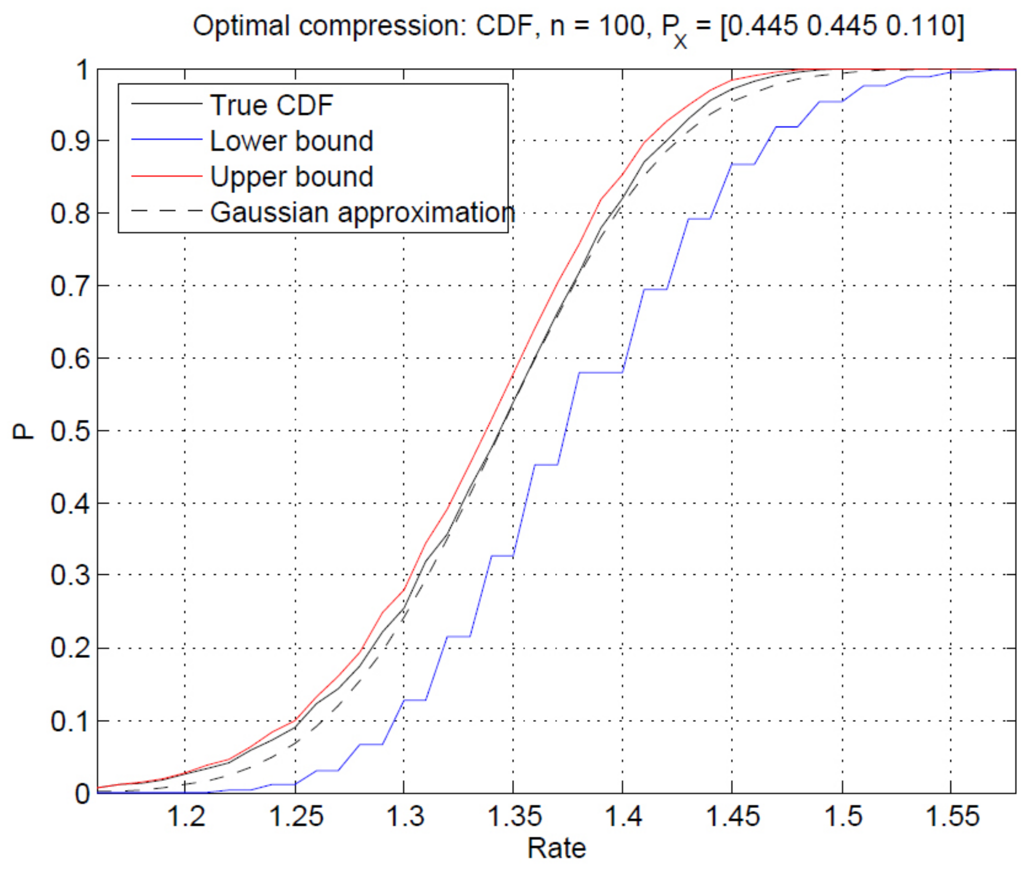


Figure 6.1: CDF and PMF of optimal compressor



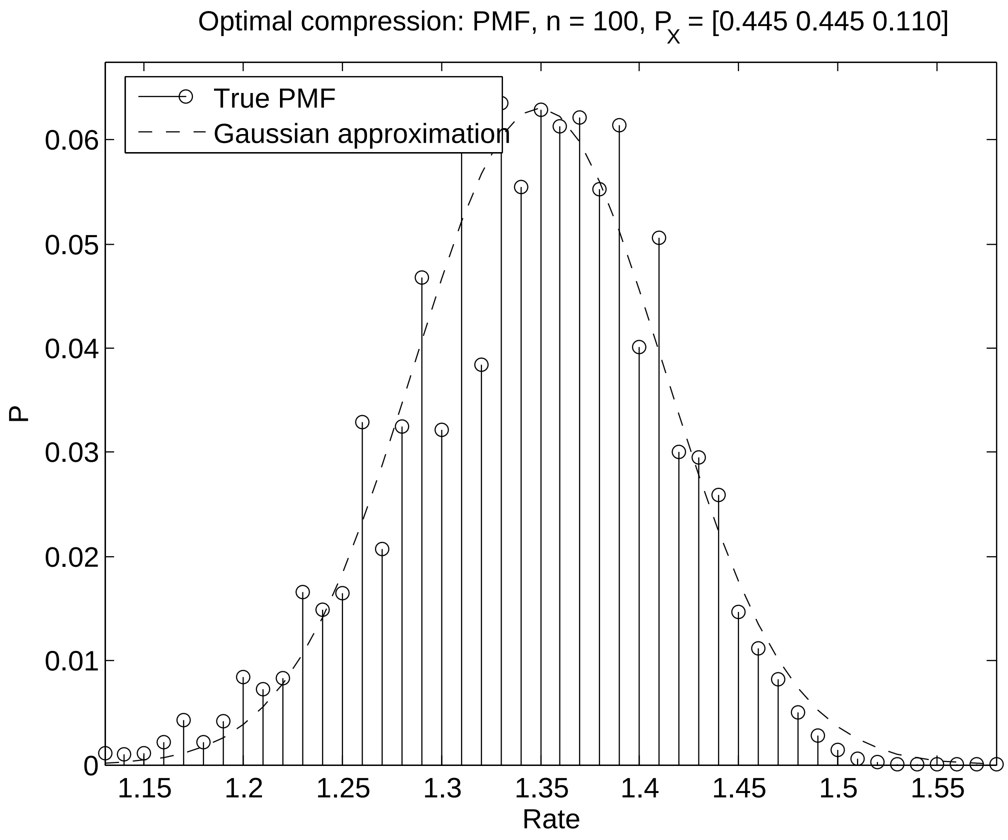
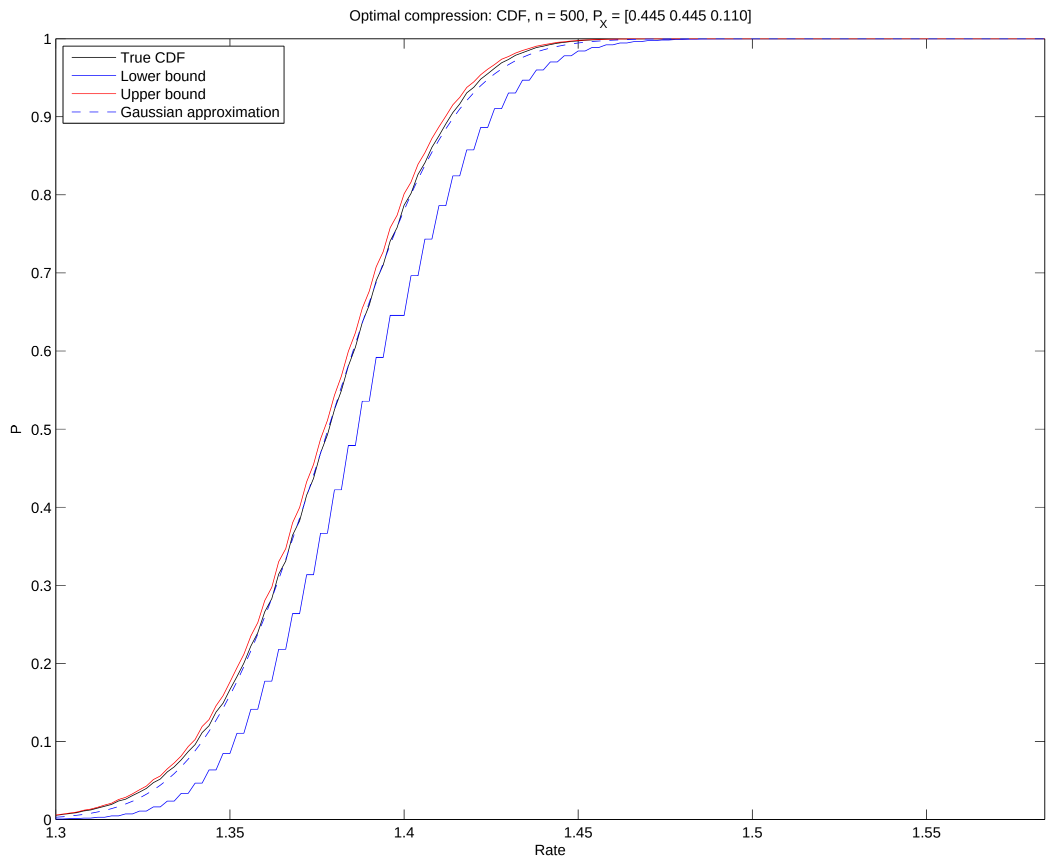


Figure 6.2: CDF and PMF, **Gaussian is shifted** to the true $\mathbb{E}[\ell(f^*(X))]$



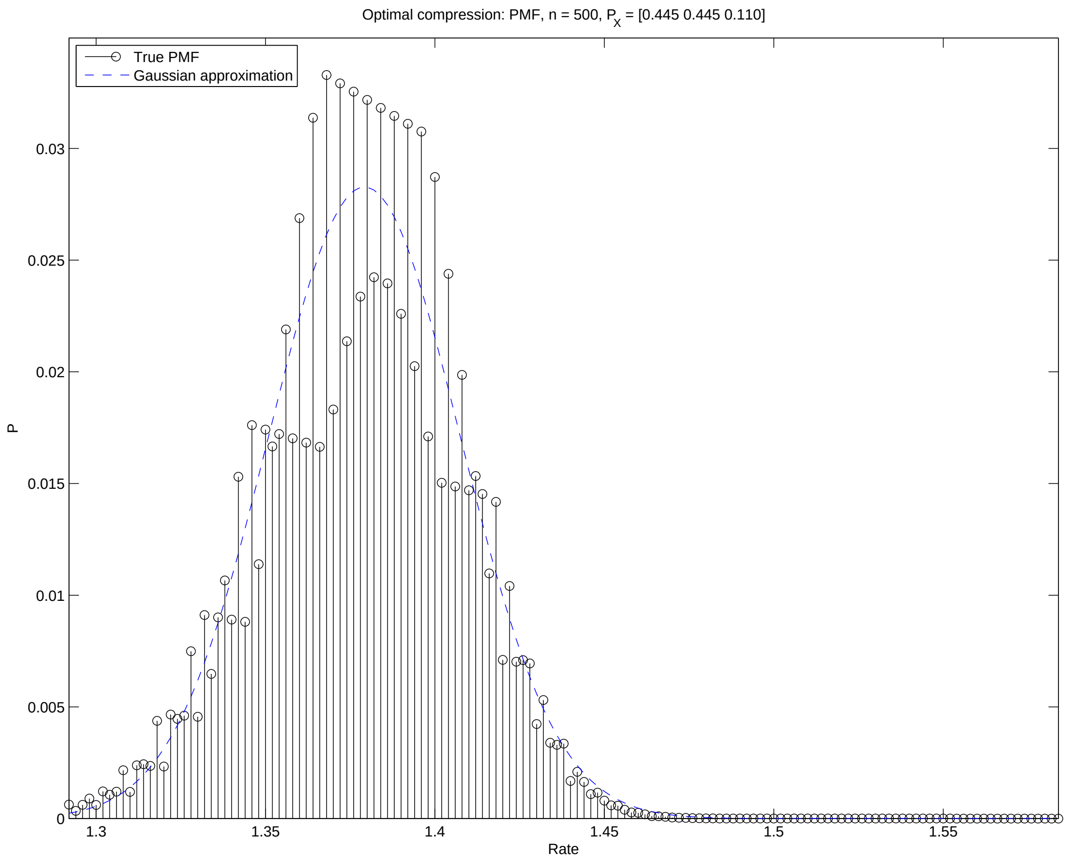
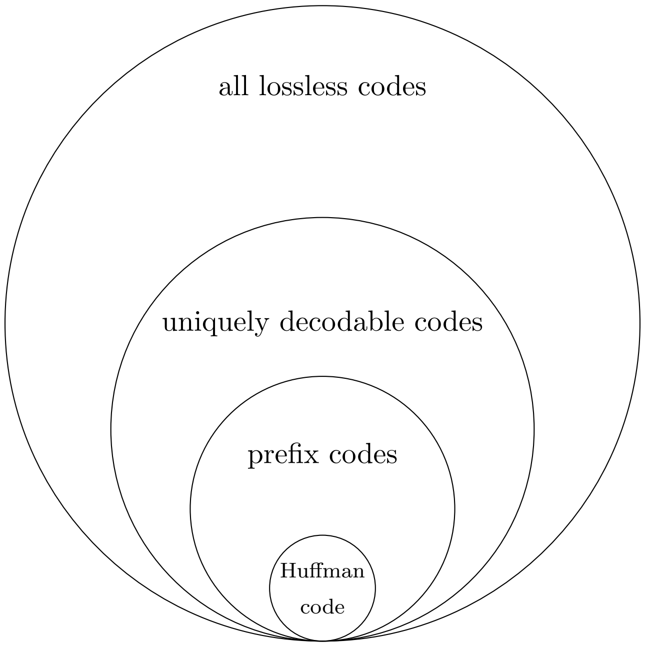


Figure 6.3: CDF and PMF of optimal compressor

6.2 Uniquely decodable codes, prefix codes and Huffman codes



We have studied f^*, which achieves the stochastically smallest code length among all variable-length compressors. Note that f^* is obtained by ordering the pmf and assigning shorter codewords to more likely symbols. In this section we focus on a specific class of compressors with good properties which lead to low complexity and short delay when decoding from a stream of compressed bits. This part is more combinatorial in nature.

We start with a few definition. Let $\mathcal{A}^+ = \bigcup_{n\geq 1} \mathcal{A}^n$ denotes all non-empty finite-length strings consisting of symbols from alphabet $\mathcal{A}$

Definition 6.1 (Extension of a code)**.** The extension of $f : \mathcal{A} \to \{0,1\}^*$ is $f : \mathcal{A}^+ \to \{0,1\}^*$ where $f(a_1, \ldots, a_n) = (f(a_1), \ldots, f(a_n))$ is defined by concatenating the bits.

Definition 6.2 (Uniquely decodable codes)**.** $f : \mathcal{A} \to \{0,1\}^*$ is *uniquely decodable* if its extension $f : \mathcal{A}^+ \to \{0,1\}^*$ is injective.

Definition 6.3 (Prefix codes)**.** $f : \mathcal{A} \to \{0,1\}^*$ is a *prefix code*[3] if no codeword is a prefix of another (e.g., 010 is a prefix of 0101).

Example:

- $f(a) = 0, f(b) = 1, f(c) = 10$ – not uniquely decodable, since $f(ba) = f(c) = 10$.
- $f(a) = 0, f(b) = 10, f(c) = 11$ – uniquely decodable and prefix.
- $f(a) = 0, f(b) = 01, f(c) = 011, f(d) = 0111$ – uniquely decodable but not prefix, since as long as 0 appears, we know that the last codeword has terminated.

Remark 6.3.

1. Prefix codes are uniquely decodable.

[3]Also known as prefix-free/comma-free/instantaneous code.

2. Similar to prefix-free codes, one can define suffix-free codes. Those are also uniquely decodable (one should start decoding in reverse direction).

3. By definition, any uniquely decodable code does not have the empty string as a codeword. Hence $f : \mathcal{X} \to \{0,1\}^+$ in both Definition 6.2 and Definition 6.3.

4. Unique decodability means that one can decode from a stream of bits without ambiguity, but one might need to look ahead in order to decide the termination of a codeword. (Think of the last example). In contrast, prefix codes allow the decoder to decode instantaneously without looking ahead.

5. Prefix code $\leftrightarrow$ binary tree (codewords are leaves) $\leftrightarrow$ strategy to ask "yes/no" questions

Theorem 6.5 (Kraft-McMillan)**.**

1. *Let $f : \mathcal{A} \to \{0,1\}^*$ be uniquely decodable. Set $l_a = l(f(a))$. Then f satisfies the* Kraft inequality

$$\sum_{a \in \mathcal{A}} 2^{-l_a} \le 1. \tag{6.3}$$

2. *Conversely, for any set of code length $\{l_a : a \in \mathcal{A}\}$ satisfying (6.3), there exists a prefix code f, such that $l_a = l(f(a))$.*

Note: The consequence of Theorem 6.5 is that as far as compression efficiency is concerned, we can forget about uniquely decodable codes that are not prefix codes.

Proof. We prove the Kraft inequality for prefix codes and uniquely decodable codes separately. The purpose for doing a separate proof for prefix codes is to illustrate the powerful technique of *probabilistic method*. The idea is from [AS08, Exercise 1.8, p. 12].

Let f be a prefix code. Let us construct a probability space such that the LHS of (6.3) is the probability of some event, which cannot exceed one. To this end, consider the following scenario: Generate independent Bern($\frac{1}{2}$) bits. Stop if a codeword has been written, otherwise continue. This process terminates with probability $\sum_{a\in\mathcal{A}} 2^{-l_a}$. The summation makes sense because the events that a given codeword is written are mutually exclusive, thanks to the prefix condition.

Now let f be a uniquely decodable code. The proof uses *generating function* as a device for counting. (The analogy in coding theory is the weight enumerator function.) First assume $\mathcal{A}$ is finite. Then $L = \max_{a\in\mathcal{A}} l_a$ is finite. Let $G_f(z) = \sum_{a\in\mathcal{A}} z^{l_a} = \sum_{l=0}^{L} A_l(f) z^l$, where $A_l(f)$ denotes the number of codewords of length l in f. For $k \ge 1$, define $f^k : \mathcal{A}^k \to \{0,1\}^*$ as the symbol-by-symbol extension of f. Then $G_{f^k}(z) = \sum_{a^k\in\mathcal{A}^k} z^{l(f^k(a^k))} = \sum_{a_1} \cdots \sum_{a_k} z^{l_{a_1}+\cdots+l_{a_k}} = [G_f(z)]^k = \sum_{l=0}^{kL} A_l(f^k) z^l$. By unique decodability of f, f^k is lossless. Hence $A_l(f^k) \le 2^l$. Therefore we have $G_f(1/2)^k = G_{f^k}(1/2) \le kL$ for all k. Then $\sum_{a\in\mathcal{A}} 2^{-l_a} = G_f(1/2) \le \lim_{k\to\infty} (kL)^{1/k} \to 1$. If $\mathcal{A}$ is countably infinite, for any finite subset $\mathcal{A}' \subset \mathcal{A}$, repeating the same argument gives $\sum_{a\in\mathcal{A}'} 2^{-l_a} \le 1$. The proof is complete by the arbitrariness of $\mathcal{A}'$.

Conversely, given a set of code lengths $\{l_a : a \in \mathcal{A}\}$ s.t. $\sum_{a\in\mathcal{A}} 2^{-l_a} \le 1$, construct a prefix code f as follows: First relabel $\mathcal{A}$ to $\mathbb{N}$ and assume that $l_1 \le l_2 \le \ldots$. For each i, $a_i \triangleq \sum_{k=1}^{i-1} 2^{-l_k} < 1$ by Kraft inequality. Thus we define the codeword $f(i) \in \{0,1\}^*$ as the first l_i bits in the binary expansion of a_i. Prove that f is a prefix code by contradiction: Suppose for some $j > i$, $f(i)$ is the prefix of $f(j)$, since $l_j \ge l_i$. Then $a_j - a_i \le 2^{-l_i}$. But $a_j - a_i = 2^{-l_i} + 2^{-l_{i+1}} + \ldots > 2^{-l_i}$, which is a contradiction. □

Open problems:

1. Find a probabilistic proof of Kraft inequality for uniquely decodable codes.
2. There is a conjecture of Ahslwede that for any sets of lengths for which $\sum 2^{-l_a} \le \frac{3}{4}$ there exists a fix-free code (i.e. one which is simultaneously prefix-free and suffix-free). So far, existence has only been shown when the Kraft sum is $\le \frac{5}{8}$, cf. [Yek04].

In view of Theorem 6.5, the optimal average code length among all prefix (or uniquely decodable) codes is given by the following optimization problem

$$\begin{aligned} L^*(X) \triangleq \min & \sum_{a\in\mathcal{A}} P_X(a) l_a \\ \text{s.t.} & \sum_{a\in\mathcal{A}} 2^{-l_a} \le 1 \\ & l_a \in \mathbb{N} \end{aligned} \tag{6.4}$$

This is an *integer programming* (IP) problem, which in general is hard to solve computationally. It is remarkable that this particular IP problem can be solved in *near-linear* time, thanks to the Huffman algorithm. Before describing the construction of Huffman codes, let us give bounds to $L^*(X)$ in terms of entropy:

Theorem 6.6.

$$H(X) \le L^*(X) \le H(X) + 1\,\texttt{bit}. \tag{6.5}$$

Proof. "$\le$" Consider the following length assignment $l_a = \left\lceil \log_2 \frac{1}{P_X(a)} \right\rceil$,[4] which satisfies Kraft since $\sum_{a\in\mathcal{A}} 2^{-l_a} \le \sum_{a\in\mathcal{A}} P_X(a) = 1$. By Theorem 6.5, there exists a prefix code f such that $l(f(a)) = \left\lceil \log_2 \frac{1}{P_X(a)} \right\rceil$ and $\mathbb{E}l(f(X)) \le H(X) + 1$.

"$\ge$" We give two proofs for the converse. One of the commonly used ideas to deal with combinatorial optimization is *relaxation*. Our first idea is to drop the integer constraints in (6.4) and *relax* it into the following optimization problem, which obviously provides a lower bound

$$L^*(X) \triangleq \min \sum_{a\in\mathcal{A}} P_X(a) l_a \tag{6.6}$$

$$\text{s.t.} \sum_{a\in\mathcal{A}} 2^{-l_a} \le 1 \tag{6.7}$$

This is a nice *convex programming* problem, since the objective function is affine and the feasible set is convex. Solving (6.6) by Lagrange multipliers (Exercise!) yields the minimum is equal to $H(X)$ (achieved at $l_a = \log_2 \frac{1}{P_X(a)}$).

Another proof is the following: For any f satisfying Kraft inequality, define a probability measure $Q(a) = \frac{2^{-l_a}}{\sum_{a\in\mathcal{A}} 2^{-l_a}}$. Then

$$\begin{aligned} \mathbb{E}l(f(X)) - H(X) &= D(P\|Q) - \log \sum_{a\in\mathcal{A}} 2^{-l_a} \\ &\ge 0 \end{aligned}$$

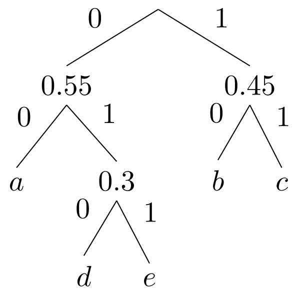
□

Next we describe the Huffman code, which achieves the optimum in (6.4). In view of the fact that prefix codes and binary trees are one-to-one, the main idea of Huffman code is to build the binary tree bottom-up: Given a pmf $\{P_X(a) : a \in \mathcal{A}\}$,

[4] Such a code is called a Shannon code.

1. Choose the two least-probable symbols in the alphabet
2. Delete the two symbols and add a new symbol (with combined weights). Add the new symbol as the parent node of the previous two symbols in the binary tree.

The algorithm terminates in $|\mathcal{A}| - 1$ steps. Given the binary tree, the code assignment can be obtained by assigning 0/1 to the branches. Therefore the time complexity is $O(|\mathcal{A}|)$ (sorted pmf) or $O(|\mathcal{A}| \log |\mathcal{A}|)$ (unsorted pmf).
Example: $\mathcal{A} = \{a, b, c, d, e\}, P_X = \{0.25, 0.25, 0.2, 0.15, 0.15\}$.
Huffman tree: codebook:



$$f(a) = 00$$
$$f(b) = 10$$
$$f(c) = 11$$
$$f(d) = 010$$
$$f(e) = 011$$

Theorem 6.7 (Optimality of Huffman codes)**.** *The Huffman code achieves the minimal average code length (6.4) among all prefix (or uniquely decodable) codes.*

Proof. [CT06, Sec. 5.8]. □

Remark 6.4 (Drawbacks of Huffman codes)**.**

1. Does not exploit memory. Solution: block Huffman coding. Shannon's original idea from 1948 paper: in compressing English text, instead of dealing with letters and exploiting the nonequiprobability of the English alphabet, working with pairs of letters to achieve more compression (more generally, n-grams). Indeed, compressing the block $(S_1, \ldots, S_n)$ using its Huffman code achieves $H(S_1, \ldots, S_n)$ within one bit, but the complexity is $|\mathcal{A}|^n$!
2. Non-universal (constructing the Huffman code needs to know the source distribution). This brings us the question: Is it possible to design universal compressor which achieves entropy for a class of source distributions? And what is the price to pay? – Homework!

There are much more elegant solutions, e.g.,

1. Arithmetic coding: sequential encoding, linear complexity in compressing $(S_1, \ldots, S_n)$ (see later).
2. Lempel-Ziv algorithm: low-complexity, universal, provably optimal in a very strong sense.

To sum up: Comparison of average code length (in bits):

$$H(X) - \log_2[e(H(X)+1)] \le \mathbb{E}[l(f^*(X))] \le H(X) \le \mathbb{E}[l(f_{\text{Huffman}}(X))] \le H(X) + 1.$$