Logistics

Review

Class Road Map - IT-I

- L1 (9/26): Overview, Communications, Information, Entropy
- L2 (10/1): Props. Entropy, Mutual Information,
- L5 (10/10): AEP, Compression
- L6 (10/15): Compression, Method of Types,
- L7 (10/17): Types, U. Coding., Stoc. Processes, Entropy rates,
- L8 (10/22): Entropy rates, HMMs, Coding, Kraft, Shannon Codes
- L9

- L10 (10/29):
- L12
- L13
- L14
- L15
- L16
- L17
- L18
- L19
Cumulative Outstanding Reading

- Read chapters 1 and 2 in our book (Cover & Thomas, “Information Theory”) (including Fano's inequality).
- Chapter 3 in our book (Cover & Thomas, “Information Theory”).
- Section 11.1 (method of types).
- Chapter 4 and 5 in our book (Cover & Thomas, “Information Theory”).

Homework

- Homework 3, due tonight, Tuesday, Oct 22nd, at 11:45pm. via our assignment dropbox (https://canvas.uw.edu/courses/847774/assignments)
- Homework 4 will be out today or tomorrow, due next Tuesday.
**Announcements**

- Office hours, every week, Tuesdays 4:30-5:30pm. Can also reach me at that time via a canvas conference.
- Midterm on Thursday, 10/31 in class. Covers everything up to and including homework 4 (today’s cumulative reading). We’ll have a review on 10/29 (one week from today).

---

**$(M, n)$ codes**

- Fixed rate block code of rate $R$.
- There are $M$ code words, $M =$ number of possible messages.
- There are $n$ source symbols encoded at a time in each code word.
- An encoder maps from length-$n$ strings of source symbols to length-$m$ bit strings.

\[
\begin{align*}
&M\text{ source symbols} \\
&\begin{array}{c}
(x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}) \\
(x_1^{(2)}, x_2^{(2)}, \ldots, x_n^{(2)}) \\
(x_1^{(3)}, x_2^{(3)}, \ldots, x_n^{(3)}) \\
\vdots
\end{array}
\rightarrow
\text{Encoder}
\rightarrow
\begin{array}{c}
M\text{ code words} \\
(y_1^{(1)}, y_2^{(1)}, \ldots, y_m^{(1)}) \\
(y_1^{(2)}, y_2^{(2)}, \ldots, y_m^{(2)}) \\
(y_1^{(3)}, y_2^{(3)}, \ldots, y_m^{(3)}) \\
\vdots
\end{array}
\end{align*}
\]

- The rate $R$ of the code depends on $M$ and $n$

\[
R = \frac{\log M}{n} = \log(\frac{\# \text{ of code words}}{\# \text{ of source symbols}})
\]
**Fixed rate block code of rate $R$**

- An $(M, n)$ code is one that uses $M$ code words for $n$ source symbols.
- Such a code thus has rate of $R = \frac{\log M}{n}$ bits per source symbol. So we need $\log M = nR$ bits to index this code.
- Then a code is defined as follows:

**Definition 8.2.2 (fixed rate block code of rate $R$)**

Let $X_1, X_2, \ldots, X_n \sim Q$, i.i.d. but $Q$ unknown. We have encoder and decoder functions as follows:

- Encoder: $f_n : \mathcal{X}^n \rightarrow \{1, 2, \ldots, 2^{nR}\}$ (8.8)
- Decoder: $\phi_n : \{1, 2, \ldots, 2^{nR}\} \rightarrow \mathcal{X}^n$ (8.9)

and probability of error

$$P_e^{(n)} = Q^n(\{x_{1:n} : \phi_n(f_n(x_{1:n})) \neq x_{1:n}\})$$ (8.10)

- Notation: $(M, n) = (2^{nR}, n)$ designates a series (in $n$) of such codes.

---

**Universal Code**

**Definition 8.2.2 (Universal rate $R$ block code)**

A rate $R$ block code for a source is universal if the functions $f_n$ and $\phi_n$ do not depend (rely directly) on the source distribution $Q$ and if

$$P_e^{(n)} \rightarrow 0 \text{ as } n \rightarrow \infty \text{ whenever } H(Q) < R$$ (8.8)

- So we require the “ability to code” at rate $R$, which really means code without error, or the error goes to zero for larger block length.
- We next state and prove one of Shannon’s main theorems.
- If $R > H(Q)$, then there exists a sequence (in $n$) of codes with the error of becoming vanishingly small.
- Conversely, if $R < H(Q)$, then the error goes to 1.
**Source Coding Theorem**

**Theorem 8.2.2 (Shannon’s Source Coding Theorem)**

\[ \exists \text{ a sequence } (2^{nR}, n) \text{ of universal source codes such that } P_e(n) \to 0 \text{ for all source distributions } Q \text{ such that } H(Q) < R. \]

**Proof.**

- Fix \( R > H(Q) \) to be strictly greater than entropy.
- Define a rate for \( n \) that is “fixed up” with a polynomial factor. I.e.,
  \[ R_n \triangleq R - \log \left( \frac{n + 1}{n} \right) < R \]  \hspace{1cm} (8.8)
- Define set of sequences that have type entropy less than this rate.
  \[ A_n \triangleq \{ x_{1:n} \in \mathcal{X}^n : H(P_{x_{1:n}}) \leq R_n \} \]  \hspace{1cm} (8.9)
  \[ = \left\{ \bigcup_{P \in \mathcal{P}_n} T(P) : H(P) \leq R_n \right\} \]  \hspace{1cm} (8.10)

---

**Stochastic Processes**

- So far we’ve been talking about i.i.d. random variables, \( X_1, X_2, \ldots \).
- In such case, each random variable has the same entropy.
- When the random variables are no longer i.i.d., how can we talk about the entropy of a process?
- We start to address that here.
Stochastic Process: Period and Periodicity

Definition 8.2.6

A Markov chain is periodic if \( d(i) > 1 \) with

\[
d(i) = \gcd\{n : p_{ii}(n) > 0\}
\]

and \( d(i) \) is called the period of state \( i \).

- Note that this is the gcd (greatest common divisor) of the epochs at which return to the same state is possible.
- State \( i \) is called periodic if \( d(i) > 1 \) and aperiodic if \( d(i) = 1 \).
- If \( p_{ii}(n) \) is the probability of returning to state \( i \) after \( n \) steps starting from state \( i \), then \( p_{ii}(n) = 0 \) unless \( n \) is a multiple of \( d(i) \). The value \( d(i) \) is maximal with this property.

Entropy rates or entropy rate

Theorem 8.2.9

We have that for stationary stochastic processes

\[
\lim_{n \to \infty} H(X_n|X_{n-1}, X_{n-2}, \ldots, X_1) \triangleq H'(\mathcal{X})
\]

\[
= H(\mathcal{X}) \triangleq \lim_{n \to \infty} \frac{1}{n} H(X_1, X_2, \ldots, X_n)
\]

Proof.

\[
b_n = \frac{H(X_1, X_2, \ldots, X_n)}{n} = \frac{1}{n} \sum_{i=1}^{n} \left( H(X_i|X_{i-1}, \ldots, X_1) = a_i \right)
\]

and \( a_n \to H'(\mathcal{X}) \) so \( b_n \to H'(\mathcal{X}) \) but by definition \( b_n \to H(\mathcal{X}) \)
Entropy rate and stationary Markov chain

- When the process is a stationary Markov chain, entropy rate has a nice form.
  - That is
    \[
    H(\mathcal{X}) = H'(\mathcal{X}) = \lim_{n \to \infty} H(X_n | X_{n-1}, \ldots, X_1) \tag{8.28}
    \]
    \[
    = \lim_{n \to \infty} H(X_n | X_{n-1}) = H(X_2 | X_1) \tag{8.29}
    \]
    \[
    = - \sum_{x_2, x_1} p(x_2, x_1) \log p(x_2 | x_1) = \sum_i \mu_i \left[ - \sum_j p_{ij} \log p_{ij} \right] \tag{8.30}
    \]
  - where again $\mu$ is the stationary distribution and $p_{ij}$ is the transition probability from $i$ to $j$.
  - Ex: previous figure $H(\mathcal{X}) = H(X_2 | X_1) = \frac{\beta}{\alpha+\beta} H(\alpha) + \frac{\alpha}{\alpha+\beta} H(\beta)$.

Ex: random walk on weighted undirected graph

- Assume irreducible and aperiodic ($d(i) = 1$ for all $i$) so unique stationary distribution.
- Graph $G = (V, E)$ with $m$ nodes labeled $\{1, 2, \ldots, m\}$ and edges with weight $w_{ij} \geq 0$.
- Random walk: start at a node, say $i$, and choose next node with probability proportional to edge weight, i.e., $p_{ij}$ as
  \[
  p_{ij} = \frac{w_{ij}}{\sum_j w_{ij}} = \frac{w_{ij}}{w_i} \tag{8.1}
  \]
  where $w_i \triangleq \sum_j w_{ij}$.
- Guess that stationary distribution has probability proportional to $w_i$.
- If $w = \sum_{i,j: j > i} w_{ij}$ then $\sum_i w_i = 2w$, so guess as stationary distribution $\mu$ with $\mu_i = w_i/2w$. 
Ex: random walk on weighted undirected graph

- This is stationary since
  \[ \forall j, \mu_j' = \sum_i \mu_i P_{ij} = \sum_i \frac{w_i}{2w} \frac{w_{ij}}{w_i} \quad (8.2) \]
  \[ = \sum_i \frac{1}{2w} w_{ij} = \frac{w_j}{2w} = \mu_j \quad (8.3) \]

- Can swap edges elsewhere (i.e., edges between nodes not including \( i \)), does not change the stationary condition which is local.
- Note chain is aperiodic since \( w_{ii} = 0 \). This is because
  \[ 2w = \sum_i w_i = \sum_{ij} w_{ij} = \sum_{ij:i=j} w_{ij} + \sum_{ij:j>i} w_{ij} \quad (8.4) \]
  \[ = \sum_{ij:i=j} w_{ij} + w + \sum_{ij:j<i} w_{ij} \quad (8.5) \]
  and \[ 2w - w = \sum_{ij:i=j} w_{ij} + \sum_{ij:j<i} w_{ij} \Rightarrow \sum_{ij:i=j} w_{ij} = 0 \Rightarrow w_{ii} = 0. \]

What is entropy of this random walk

\[ H(\mathcal{X}) = H(X_2|X_1) = - \sum_i \mu_i \sum_j p_{ij} \log p_{ij} \quad (8.6) \]
\[ = - \sum_i \frac{w_i}{2w} \sum_j \frac{w_{ij}}{w_i} \log \frac{w_{ij}}{w_i} = - \sum_{ij} \frac{w_{ij}}{2w} \log \frac{w_{ij}}{w_i} \quad (8.7) \]
\[ = - \sum_{ij} \frac{w_{ij}}{2w} \log \left[ \frac{w_{ij}}{2w \frac{w_i}{w_i}} \right] \quad (8.8) \]
\[ = - \sum_{ij} \frac{w_{ij}}{2w} \log \frac{w_{ij}}{2w} + \sum_{ij} \frac{w_{ij}}{2w} \log \frac{w_i}{2w} \quad (8.9) \]
\[ = H(\ldots, \frac{w_{ij}}{2w}, \ldots) - H(\ldots, \frac{w_i}{2w}, \ldots) \quad (8.10) \]
\[ = (\text{overall edge uncertainty}) \quad (8.11) \]
\[ - (\text{overall node uncertainty in stationary condition}) \quad (8.12) \]
What is entropy of this random walk

- So, the entropy of the random walk is
  \[ H(\mathcal{X}) = (\text{overall edge uncertainty}) - (\text{overall node uncertainty in stationary condition}) \]

- Intuition: As node entropy decreases while keeping edge uncertainty constant, the network becomes more concentrated,
- fewer nodes are hubs, and the hubs that remain are widely connected (since edge entropy is fixed).
- In such case (few well connected hubs), it is likely one will land on such a hub (in a random walk) and then will be faced with a wide variety of choice as to where to go next ⇒ increase in overall uncertainty of the walk.
- If node entropy goes up with edge entropy fixed, then many nodes are hubs all with relatively low connectivity, so hitting them doesn’t provide much choice ⇒ random walk entropy goes down.

Hidden Markov models (HMMs)

- An HMM is a distribution \( p(X_{1:n}, Y_{1:n}) \) over \( 2n \) random variables that factors in a particular way.
- Easiest way to depict all of the factorization properties is to use a graphical model, as in the below, where \( n = 5 \):

\[
\begin{align*}
Y_1 &\rightarrow X_1 \\
Y_2 &\rightarrow X_2 \\
Y_3 &\rightarrow X_3 \\
Y_4 &\rightarrow X_4 \\
Y_5 &\rightarrow X_5
\end{align*}
\]

- Let \( Y_1, Y_2, \ldots, Y_n \) be a stationary Markov chain.
- Let \( X_{1:n} \) be a random function of this Markov chain. I.e.,

\[
X_i = \begin{cases} 
\phi_1(Y_i) & \text{with probability } p_1 \\
\phi_2(Y_i) & \text{with probability } p_2 \\
\vdots & \\
\phi_m(Y_i) & \text{with probability } p_m 
\end{cases} = \phi_N(X_i) \quad (8.13)
\]

where \( N \in \{1, 2, \ldots, m\} \) itself is a random variable.
HMMs

- Note that the stochastic process $X_1, X_2 \ldots$ does not form a Markov chain in general. Why? because it does not satisfy the first order Markov assumption, nor any order Markov assumption in general.

- If $\{Y_i\}_i$ is stationary, then is $\{X_i\}_i$ a stationary stochastic process? Yes. Possible HW problem, so no more given here.

- We can compute the entropy rate of $\{X_i\}_i$, i.e.,
  
  $H(\mathcal{X}) = \lim_{n \to \infty} H(X_n|X_{n-1}, \ldots, X_1)$ but it is ugly, so instead we compute upper and lower bounds.

  - Upper bound(s):
    
    $H(X_n|X_{n-1}, \ldots, X_1) = H(X_{n+1}|X_n, \ldots, X_2) \geq H(X_{n+1}|X_1, \ldots, X_n)$ (8.14)
    
    $\geq H(X_{n+2}|X_{n+1}, \ldots, X_1) \geq \cdots \geq H(\mathcal{X})$ (8.15)

- A lower bound is given by $H(X_n|X_{n-1}, \ldots, X_2, Y_1) \leq H(\mathcal{X})$ because

  $H(X_n|X_{n-1}, \ldots, X_2, Y_1) = H(X_n|X_{n-1}, \ldots, X_2, X_1, Y_1)$ (8.16)
  
  $= H(X_n|X_{n-1}, \ldots, X_1, Y_1, Y_0, Y_1, \ldots, Y_{-k})$ (8.17)
  
  $= H(X_n|X_{n-1}, \ldots, X_1, Y_1, Y_0, Y_1, \ldots, Y_{-k}, X_0, \ldots, X_{-k})$ (8.18)
  
  $\leq H(X_n|X_{n-1}, \ldots, X_1, X_0, \ldots, X_{-k})$ (8.19)

  - So summarizing the bounds on the HMM information rates, we have

    $H(X_n|X_{n-1}, \ldots, X_1, Y_1) \leq H(\mathcal{X}) \leq H(X_n|X_{n-1}, \ldots, X_1)$ (8.20)
**Lemma 8.4.1 (ever shrinking sandwich)**

\[ H(X_n|X_{n-1}, \ldots, X_1) - H(X_n|X_{n-1}, \ldots, X_1, Y_1) \to 0 \]  \hspace{1cm} (8.21)

**Proof.**

\[
H(X_n|X_{n-1}, \ldots, X_1) - H(X_n|X_{n-1}, \ldots, X_1, Y_1)
\]

\[= I(X_n; Y_1|X_{n-1}, \ldots, X_1) \leq H(Y_1) \]  \hspace{1cm} (8.22)

Also, \( I(Y_1; X_1, \ldots, X_n) \leq H(Y_1) \) for all \( n \).

Now,

\[
\lim_{n \to \infty} I(Y_1; X_1, \ldots, X_n) = \lim_{n \to \infty} \sum_{i=1}^{n} I(Y_1; X_i|X_1:i-1)
\]

\[= \sum_{i=1}^{\infty} I(Y_1; X_i|X_1:i-1) \leq H(Y) < \infty \]  \hspace{1cm} (8.23)

So an infinite sum is constant, must mean the terms \( \to 0 \) as \( n \to \infty \).

Thus, each of the terms \( I(Y_1; X_i|X_1:i-1) \to 0 \) as \( n \to \infty \). \( \square \)

**HMM rate summary**

- Summarizing, we have

\[
\lim_{n \to \infty} H(X_n|X_{n-1}, \ldots, X_1, Y_1) = H(X) = \lim_{n \to \infty} H(X_n|X_{n-1}, \ldots, X_1)
\]  \hspace{1cm} (8.25)
Coding

- New topic: coding, meaning practical lossless coding of information sources governed by a distribution.
- Shannon’s source coding theorem said we can code using \( R > H(X) \) bits per source symbol if we use long enough block.
- These were “block” codes in that we store and/or transfer one block of symbols at a time, but such codes are not always practical.
- We seek other coding strategies.
- **Variable length symbol codes**: One symbol at a time is encoded. Variable length (rather than fixed length) code words. Ex: Huffman coding.
- **Stream codes**: codes that operate on data coming in as a stream and decide codeword depending on current symbol and history. Ex: Arithmetic codes, Lempel-Ziv code (the latter of which is Universal since it does not require \( p(x) \)).

Practical Coding

- We want to develop practical coding algorithms that still approach, or achieve, the entropy limit.
- They might use the distribution \( p(x) \) which is either given or is estimated in some way.
- We won’t get into any details on how to estimate \( p(x) \) (that is a density estimation problem) but we assume we either have it or some approximation.
- We will ultimately look, however, at what happens if the true distribution is \( p(x) \) and we use \( q(x) \) instead.
Source Code

Definition 8.5.1 (source code)

A source code $C$ for r.v. $X$ is a mapping

$$C : \mathcal{X} \rightarrow \mathcal{D}^* \tag{8.26}$$

from $\mathcal{X}$ to $\mathcal{D}^*$, the set of finite strings from a $D$-ary alphabet. $C(x)$ is the codeword corresponding to $x$, and $\ell(x)$ is the length of the codeword.

Example 8.5.2

Let $\mathcal{X} = \{ \text{red}, \text{blue} \}$. Then a code might be $C(\text{red}) = 00$ and $C(\text{blue}) = 11$, which would be a binary code for $D = \{0, 1\}$.

Definition 8.5.3 (expected length)

The expected length $L(C)$ of code $C$ for r.v. $X$ with distribution $p(x)$ is

$$L(C) = \sum_x p(x) \ell(x) \tag{8.27}$$

Assume $D = \{0, 1, 2, \ldots, D - 1\}$ in general (but often $D = 2$).

Another code,

Example 8.5.4

Let $\mathcal{X} = \{1, 2, 3, 4\}$ and $D = \{0, 1\}$. We can define the code with a table.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$p(x)$</th>
<th>$c(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1/2$</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>$1/4$</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>$1/8$</td>
<td>110</td>
</tr>
<tr>
<td>4</td>
<td>$1/8$</td>
<td>111</td>
</tr>
</tbody>
</table>

In this case, $H(X) = 1.75$. But $L(C) = E\ell(X) = 1.75$, so this code is pretty good.

Moreover, it is easy to code. What source symbols correspond to the string 0101101111110100? 1, 2, 3, 4, 4, 3, 2, 1

With punctuation: 0,10,110,111,111,110,10,0, so code in some sense is “self punctuating”
Aside: English

- isenglishselfpunctuating

Now does it take you to read this sentence that is written without any punctuation marks or even end of sentence marks such as a question mark or even inter-sentence spaces?

- nowhere = “now, here” or “no, where”?
Another code

- Here, $\mathcal{X} = \{1, 2, 3\}$ and $\mathcal{D} = \{0, 1\}$
- Code is:

<table>
<thead>
<tr>
<th>$x$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(x)$</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{1}{3}$</td>
</tr>
<tr>
<td>$C(X)$</td>
<td>0</td>
<td>10</td>
<td>11</td>
</tr>
</tbody>
</table>

- So $H = 1.58$ but $E\ell(X) = 1.66 > H$ bits.
- Can we easily decode? $10110010 = 2, 3, 1, 1, 2$

Ex: Morse Code

- Morse code, series of dots and dashes to represent letters
- most frequent letter sent with the shortest code, 1 dot
- Note: codewords might be prefixes of each other (e.g., “E” and “F”).
- uses only binary data (single current telegraph, size two “alphabet”), could use more (three, double current telegraph), but this is more susceptible to noise (binary in computer rather than ternary).
Set of codes

- For $\mathcal{X} = \{1, 2, 3, 4\}$ and binary code, consider the following 4 codes.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$p(x)$</th>
<th>$C_I$</th>
<th>$C_{II}$</th>
<th>$C_{III}$</th>
<th>$C_{IV}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0</td>
<td>1</td>
<td>10</td>
<td>01</td>
</tr>
<tr>
<td>3</td>
<td>0.125</td>
<td>1</td>
<td>00</td>
<td>110</td>
<td>011</td>
</tr>
<tr>
<td>4</td>
<td>0.125</td>
<td>10</td>
<td>11</td>
<td>111</td>
<td>0111</td>
</tr>
<tr>
<td>$H(X)$</td>
<td>1.75</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$E\ell(X)$</td>
<td>-</td>
<td>1.125</td>
<td>1.25</td>
<td>1.75</td>
<td>1.875</td>
</tr>
</tbody>
</table>

- Code efficiency = $H(X)/E\ell(X)$.
- Which code is best? The one with the shortest expected length is generally preferable, and so would we prefer $C_I$ or $C_{II}$?
- But how can the expected length be smaller than entropy?
- Consider $C_I$ and decode string: 00001. It could come from 1,2,1,2,3 or 2,1,2,1,3 or 1,1,1,1,3, or etc. This code is undesirable since we can’t decode, or if we try we will have errors with very high probability.

Consider $C_{II}$. How would we decode 0011? Could be either 1,1,2,2 or 3,4, so again we can’t decode without a probability of error (and the longer the sequence the probability of error goes to 1).

Consider $C_{III}$. This code seems at least feasible (since $E\ell \geq H$). Decoding seems easy: (e.g., 111110100 = 111,110,10,0 = 4,3,2,1) seems like every string we encounter easily gets to the end of a codeword before going to next codeword.
Set of codes

<table>
<thead>
<tr>
<th>x</th>
<th>p(x)</th>
<th>$C_I$</th>
<th>$C_{II}$</th>
<th>$C_{III}$</th>
<th>$C_{IV}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0</td>
<td>1</td>
<td>10</td>
<td>01</td>
</tr>
<tr>
<td>3</td>
<td>0.125</td>
<td>1</td>
<td>00</td>
<td>110</td>
<td>011</td>
</tr>
<tr>
<td>4</td>
<td>0.125</td>
<td>10</td>
<td>11</td>
<td>111</td>
<td>0111</td>
</tr>
<tr>
<td>$H(X)$</td>
<td>1.75</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$E\ell(X)$</td>
<td>-</td>
<td>1.125</td>
<td>1.25</td>
<td>1.75</td>
<td>1.875</td>
</tr>
</tbody>
</table>

Consider $C_{IV}$. Can we decode 00000000111? Yes, but if we only see a prefix, such as 01, we don’t know until we see more bits what source symbol it is.

Code types

### Definition 8.5.5 (non-singular)

A code is said to be non-singular if every element of the range of $X$ (i.e., all elements of $\mathcal{X}$) maps to a different string in $\mathcal{D}^*$. I.e.,

$$x_i \neq x_j \Rightarrow C(x_i) \neq C(x_j)$$

(8.28)

- We can view this as a mapping. It is less strict than onto but sufficient for being able to decode individual symbols.

- Note that $C_I$ above is singular.
Our goal

- Before going further, note: our goal is to send or store a sequence of code words for a sequence of symbols.
- A non-singular code could be unique if there is a comma between code words (e.g., Morse code is such that there is a space).
- In general, however, it is better to have a self punctuating or instantaneous code.

Definition 8.5.6 (code extension)

A code extension $C^*$ of $C$ is a mapping from finite length strings of $D$, defined as:

$$C(x_1, x_2, \ldots, x_n) = C(x_1)C(x_2) \ldots C(x_n) \quad (8.29)$$

- Note that there are no commas in the extension, rather concatenation.
- Ex: If $C(x_1) = 0$ and $C(x_2) = 1$ then $C(x_1, x_2) = 01$.

Code types

Definition 8.5.7 (uniquely decodable)

A code $C$ with extension $C^*$ is uniquely decodable if the extension $C^*$ is non-singular.

- $C_I$ singular. Extension to $C_{II}$ singular so $C_{II}$ not uniquely decodable.
- But how long must we wait until we know the source? In some even uniquely decodable cases, we might need to wait until the end.
- Ex: consider the code

<table>
<thead>
<tr>
<th>$x$</th>
<th>$C(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>00</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>110</td>
</tr>
</tbody>
</table>

- Code string: 1100000000 = 3,2,2,2,2
- Code string: 11000000000 = 4,2,2,2,2
- So we don’t know identity of first symbol until end of code string. ☺️.
Prefix codes

**Definition 8.5.8 (prefix code)**

A code is called a **prefix code** or an **instantaneous code** if no codeword is a prefix of any other codeword.

- We know the end of a codeword because it can’t be a prefix of any other codeword.
- Code in previous page is not prefix free, 11 was a prefix of 110 so we couldn’t decide between 11 or 110 until we could count the number of zeros.
- A prefix code is self-punctuating (since there are implicit punctuation marks between codewords).
- Prefix code $\Rightarrow$ uniquely decodable. But (as we saw) uniquely decodable $\nRightarrow$ prefix code.

**Code classes**

- Goal is to find a code with the shortest possible expected length.
- From the above code class, we might think that we want to use codes from the largest class possible (since we might think we’re more likely to get shorter codes).
- We can do better than entropy with non-singular codes, but we want lossless encoding $x = \text{ungzip}(\text{gzip}(x))$. 
### Kraft Inequality

**Theorem 8.6.1 (Kraft Inequality)**

For any instantaneous code (prefix code) over alphabet of size $D$, the codeword lengths $\ell_1, \ell_2, \ldots, \ell_m$ must satisfy

$$\sum_i D^{-\ell_i} \leq 1 \quad (8.30)$$

Conversely, given a set of codeword lengths satisfying the above inequality, there exists an instantaneous code with these word lengths.

- Note what the converse is saying: there exists a code with these lengths, not that all codes with these lengths will satisfy the inequality.
- Key point: for $\ell_i$ satisfying Kraft, no further restriction imposed by also wanting a prefix code, so we might as well use a prefix code (assuming it is easy to find given the lengths).

**Proof of Kraft Inequality.**

- Represent set of codes on a $D$-ary (not necessarily balanced) tree:

```
1 2 3 ... 
D D D ... 
1 2 3 ... 
D D D ... 
```

- Codewords correspond to leaves
- Path from root to leaf determines a codeword
- Prefix condition: won’t get to a codeword until we get to a leaf (no descendants of codewords are codewords)
Entropy rates
HMMs
Coding
Kraft ≤
Shannon Codes
Kraft ≤ II

Kraft inequality

...proof of Kraft inequality cont.

- $\ell_{\text{max}} = \max_i (\ell_i)$ is the length of the longest codeword.
- We can expand the full-tree down to depth $\ell_{\text{max}}$.
  Some nodes at that level $\ell_{\text{max}}$ are either:
  1. codewords,
  2. descendants of codewords, or
  3. neither
- Consider a codeword $i$ at level $\ell_i$ in tree (so it has length $\ell_i$).
- Then, there are $D^{\ell_{\text{max}} - \ell_i}$ descendants in the tree at level $\ell_{\text{max}}$.
- Because of prefix condition, descendants of code $i$ at level $\ell_i$ are disjoint from descendants of code $j$ at level $\ell_j$ when $i \neq j$ (i.e., descendant sets for different codewords are disjoint).
- Also, total number of nodes in set of all descendants is $\leq D^{\ell_{\text{max}}}$.

Conversely: given codeword lengths $\ell_1, \ell_2, \ldots, \ell_m$ satisfying Kraft inequality (we must construct a prefix code with these lengths).

- Consider a full $D$-ary tree of depth $\ell_{\text{max}}$ with $D^{\ell_{\text{max}}}$ terminal nodes.
  - @ level 0, $\exists$ fraction 1 of the descendants at each node at that level;
  - @ level 1, $\exists$ fraction $1/D$ descendants at each node at that level;
  - @ level 2, $\exists$ fraction $1/D^2$ ...
  - In general, at each level $i \in [0, \ell_{\text{max}}]$ in tree, there is a fraction $D^{-i}$ terminal nodes that are descendants that stem from each of the $D^i$ nodes at level $i$.

All of the above implies:

$$\sum_i D^{\ell_{\text{max}} - \ell_i} \leq D^{\ell_{\text{max}}} \Rightarrow \sum_i D^{-\ell_i} \leq 1 \quad (8.31)$$
Kraft inequality

... proof of Kraft inequality cont.

- Sort the lengths \((\ell_1, \ell_2, \ldots, \ell_m)\) ascending to \((s_1, s_2, \ldots, s_m)\) with \(s_1 \leq s_2 \leq \cdots \leq s_m\). Note there are as many lengths as there are codewords.

- For length \(s_1\) chose any node at level \(s_1\) to indicate the code.

- To ensure prefix free property, the node becomes a terminal node, thus eliminating a fraction \(D^{-s_1}\) of the terminal nodes at depth \(\ell_{\text{max}}\) (which would have been potential code words of longer length, but now they are out of the running).

- Next: chose any remaining node at level \(s_2\), thus eliminating a fraction \(D^{-s_2}\) of the nodes \((D^{s_1} - 1)D^{s_2-s_1}\) choices at this point).

- Total fraction eliminated is \(D^{-s_1} + D^{-s_2}\).

...
**Infinite Kraft**

**Theorem 8.6.2 (countably infinite Kraft)**

For any countably infinite set of codewords that form a prefix set, this satisfies the extended Kraft inequality, i.e.

\[ \sum_{i=1}^{\infty} D^{-\ell_i} \leq 1 \]  

(8.32)

Conversely, given \( \ell_i \) satisfying the above, \( \exists \) a prefix code with these lengths.

**proof of countably infinite Kraft.**

- Assume we have such a prefix code, and let the \( D \)-ary alphabet be \{0, 1, \ldots, D - 1\}.
- Consider the \( i^{th} \) codeword \( y_1, y_2, \ldots, y_{\ell_i} \).

... proof of infinite Kraft.

- Consider expansion of codeword using fractional digits:

\[ 0.y_1y_2y_3 \ldots y_{\ell_i} = \sum_{j=1}^{\ell_i} y_j D^{-j} \]  

(8.33)

- Examples: When \( D = \{0, 1\} \) then 0.1 = 1/2, 0.01 = 1/4, 0.11 = 3/4, and 0.001 = 1/8 (so bits are after the binary point).
- Associate each codeword \( y_1, \ell_i \) with the half-open interval on the real line \([0.y_1y_2 \ldots y_{\ell_i}, 0.y_1y_2 \ldots y_{\ell_i} + 1/D^{\ell_i}]\)
- Example: With \( D = 10 \), then if \( 0.y_1y_2y_3 = 0.157 \), the associated half-open interval is \([0.157, 0.158)\), and if \( 0.y_1y_2y_3 = 0.159 \), the associated half-open interval is \([0.159, 0.160)\).
Kraft inequality

...proof of infinite Kraft.

- So the interval for codeword $y_1y_2y_3 \ldots y_{\ell_i}$ corresponds to the set of all real numbers that begins with $0.y_1y_2y_3 \ldots y_{\ell_i}$ and is thus a sub-interval of the unit interval.
- Also $y_1y_2y_3 \ldots y_{\ell_i}$ is not a prefix of any other codeword, so the intervals must be disjoint.
- Length of interval for codeword $y_1y_2y_3 \ldots y_{\ell_i}$ is $D^{-\ell_i}$.
- And since all intervals live in $[0, 1)$ we must have

$$\sum_i D^{-\ell_i} \leq 1 \quad (8.34)$$

- Proof of converse is similar to finite case and also to arithmetic coding that we’ll soon see, so we skip the proof here.

Towards Optimal Codes

- Summarizing: Prefix code $\Leftrightarrow$ Kraft inequality.
- Thus, we need only find lengths that satisfy Kraft to find a prefix code.
- Goal: find a prefix code with minimum expected length

$$L(C) = \sum_i p_i \ell_i \quad (8.35)$$

- This is an constrained optimization problem:

$$\min_{\{\ell_1: \ldots \ell_m\} \in \mathbb{Z}^+} \sum_i p_i \ell_i \quad (8.36)$$

subject to $\sum_i D^{-\ell_i} \leq 1$

- Linear integer program is an NP-complete optimization, not likely to be efficiently solvable (unless P=NP).
Towards Optimal Codes

- Relax the integer constraints on $\ell_i$ for now, and consider Lagrangian

$$J = \sum_i p_i \ell_i + \lambda \left( \sum_i D^{-\ell_i} - 1 \right) \quad (8.37)$$

- Taking derivatives and setting to 0,

$$\frac{\partial J}{\partial \ell_i} = p_i - \lambda D^{-\ell_i} \ln D = 0 \quad (8.38)$$

$$\Rightarrow D^{-\ell_i} = \frac{p_i}{\lambda \ln D} \quad (8.39)$$

$$\frac{\partial J}{\partial \lambda} = \sum_i D^{-\ell_i} - 1 = 0 \quad \Rightarrow \quad \lambda = \frac{1}{\ln D} \quad (8.40)$$

$$\Rightarrow D^{-\ell_i} = p_i \quad \text{yielding} \quad \ell_i^* = -\log_D p_i \quad (8.41)$$

This implies that:

$$L^* = \sum_i p_i \ell_i^* = -\sum_i p_i \log_D p_i = H_D(X) = H(X)/\log D \quad (8.42)$$

- So the optimal expected code length, as a result of this optimization process, is the entropy assuming that we are allowed to have fractional code lengths

- Since $\ell_i^* = -\log_D p_i$, this means that optimal code “length” (while fractional) is the same as the information about the event. I.e., shortest possible coding length is the inherent information about an event. This is like the MDL (minimum description principle), tries to find the simplest explanation about a source.

- Compare fractional codeword lengths to long block codes, what is the relation?
Theorem 8.6.3

Entropy is the minimum expected length. That is, the expected length $L$ of any instantaneous $D$-ary code (which thus satisfies Kraft inequality) for a r.v. $X$ is such that

$$L \geq H_D(X)$$

(8.43)

with equality iff $D^{-\ell_i} = p_i$.

Proof of Theorem 8.6.3.

$$L - H_D(X) = \sum_i p_i \ell_i - \sum_i p_i \log_D 1/p_i$$

(8.44)

$$= -\sum_i p_i \log_D D^{-\ell_i} + \sum_i p_i \log_D p_i$$

(8.45)

$$= -\sum_i p_i \log_D D^{-\ell_i} + \log_D \left( \sum_i D^{-\ell_i} \right) - \log_D \left( \sum_i D^{-\ell_i} \right)$$

(8.46)

$$+ \sum_i p_i \log_D p_i$$  \quad \text{(now define } r_i = \frac{D^{-\ell_i}}{\sum_i D^{-\ell_i}} \text{)}

(8.47)

$$= \sum_i p_i \log \frac{p_i}{r_i} - \log_D \left( \sum_i D^{-\ell_i} \right) = D(p||r) + \log_D(1/c)$$

(8.48)

$$\geq 0 \quad \text{since } c \leq 1 \text{ by Kraft, where } c = \sum_i D^{-\ell_i}$$

(8.49)
Optimal Code Lengths

...Proof of Theorem 8.6.3.

- So we have that $L \geq H_D(X)$.
- Equality, $L = H$ is achieved iff $p_i = D^{-\ell_i}$ for all $i \Leftrightarrow -\log_D p_i$ is an integer . . .
- . . . in which case $c = \sum_i D^{-\ell_i} = 1$

Definition 8.6.4 ($D$-adic)

A probability distribution is called $D$-adic w.r.t. $D$ if each of the probabilities is $= D^{-n}$ for some $n$.

- Ex: when $D = 2$, the distribution $[\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}]$ is 2-adic.
- Thus, we have equality above iff the distribution is appropriately $D$-adic.

Summary of recent results

To summarize the conditions and relations:

- In the relaxed optimization problem, we relaxed the lengths so that they need not be integers, and we get $E\ell = H$.
- Assuming Kraft is true (and thus a prefix code exists), we have (with integer lengths) that $E\ell \geq H$.
- I.e., if we assume Kraft, and $\ell_i = -\log_D p_i$ is an integer, then $E\ell = H$.
- I.e., if we assume Kraft, and $\ell_i \neq -\log_D p_i$, but the lengths $\ell_i$ are still integers, then we have $E\ell > H$ strictly.
Shannon Codes

- \( L - H = D(p || r) + \log_D 1/c \), with \( c = \sum_i D^{-\ell_i} \)
- Thus, to code find closest (in the KL sense) \( D \)-adic distribution w.r.t. \( D \) to \( p \) and then construct the code as in the proof of the Kraft inequality converse.
- In general, however, unless P=NP, it is hard to find the KL closest \( D \)-adic distribution (integer linear programming problem).
- **Shannon codes**: consider \( \ell_i = \lceil \log_D 1/p_i \rceil \) as the code lengths
- \( \sum_i D^{-\ell_i} = \sum_i D^{-\lceil \log 1/p_i \rceil} \leq \sum_i D^{-\log 1/p_i} = \sum_i p_i = 1 \)
- This means Kraft inequality holds for these lengths, so there is a prefix code (if the lengths were too short there might be a problem but we're rounding up).
- Also, we have a bound on lengths in terms of real numbers
  \[
  \log_D \frac{1}{p_i} \leq \ell_i < \log_D \frac{1}{p_i} + 1 \quad (8.50)
  \]

- Taking expected values on both sides yields
  \[
  H_D(X) \leq L < H_D(X) + 1 \quad (8.51)
  \]
  Close to the entropy, only one extra bit! Is this good?
  Also, \( H \leq L^* \leq L \) where \( L^* \) is the optimal length for integer length codes (i.e., might not have optimal integer lengths satisfying Kraft).

**Theorem 8.7.1**

Let \( \ell_1^*, \ell_2^*, \ldots, \ell_m^* \) be the optimal integral codeword lengths for source \( p \) and \( D \)-ary alphabet. \( L^* \) is the expected length. Then

\[
H_D(X) \leq L^* < H_D(X) + 1 \quad (8.52)
\]

- So average overhead of using integers (rather than fractional) codeword lengths is no more than one bit per symbol.
How bad is one bit?

- How bad is this overhead?
- Depends on $H$. Efficiency of code

\[ 0 \leq \text{Efficiency} \triangleq \frac{H_D(X)}{E\ell(X)} \leq 1 \quad (8.53) \]

- If $E\ell(X) = H_D(X) + 1$, then efficiency $\to 1$ as $H(X) \to \infty$.
- Efficiency $\to 0$ as $H(X) \to 0$, so entropy would need to be very large for this to be good.
- For small alphabets, impossible to have good efficiency. E.g., $D = \{0, 1\}$ then $\max H(X) = 1$, so best possible efficiency is 50%.

Improving efficiency

- Such symbol codes are inherently disadvantaged, unless their distributions are $D$-adic.
- We can reduce overhead (improve efficiency) by coding > 1 symbol at a time (block code, or a vector code, the symbol is the vector).
- Let $L_n$ be the expected length of $n$ symbols $x_{1:n}$. $L_n$ is the expected per-symbol length, when encoding $n$ symbols.

\[ L_n = \frac{1}{n} \sum_{x_{1:n}} p(x_{1:n}) \ell(x_{1:n}) = \frac{1}{n} E\ell(x_{1:n}) \quad (8.54) \]

- Let's use Shannon coding lengths to get

\[ \sum_i p_i \left( \log 1/p_i \leq \ell_i \leq \log 1/p_i + 1 \right) \quad (8.55) \]

\[ \Rightarrow H(X_1, \ldots, X_n) \leq E\ell(X_{1:n}) < H(X_1, \ldots, X_n) + 1 \quad (8.56) \]
Improving efficiency

- If the $X_i$ are i.i.d. then $H(X_1, \ldots, X_n) = nH(X_i)$.
- $\Rightarrow$ we have that
  \[ H(X) \leq L_n \leq H(X) + \frac{1}{n} \]  
  (8.57)
- As $n$ gets big, per symbol penalty of a Shannon code decreases, and we approach the Entropy limit (per symbol), although once again we have to code a block at a time.
- Again, even if symbols are independent it is better to code jointly.

Stochastic processes

- Consider any stationary (ergodic) stochastic process. Then
  \[ H(X_1, \ldots, X_n) \leq E\ell(X_1, \ldots, X_n) < H(X_1, \ldots, X_n) + 1 \]  
  (8.58)
  \[ \Rightarrow \frac{H(X_1, \ldots, X_n)}{n} \leq L_n < \frac{H(X_1, \ldots, X_n)}{n} + \frac{1}{n} \]  
  (8.59)
- If stationary, then l.h.s. $\to H(X)$ as $n \to \infty$.
- Thus, as $n$ gets large, expected length of code goes to the entropy rate of the stochastic process.
- We can make penalty per source symbol as small as we want if we don’t mind long block lengths. This can be stated as a theorem

**Theorem 8.7.2**

*Minimum expected codeword lengths per symbol satisfy*

\[ \frac{H(X_1, \ldots, X_n)}{n} \leq L_n^* < \frac{H(X_1, \ldots, X_n)}{n} + \frac{1}{n} \]  
(8.60)

if $X_i$ is stationary. I.e., $L^* \to H(X)$
Coding with the wrong distribution

- In general, we don’t have the “true” distribution (if there is one).
- With the wrong distribution, we’ll make mistakes. I.e., Shannon code would use lengths \( \ell(x) = \lceil \log \frac{1}{q(x)} \rceil \) but the true probability is \( p(x) \neq q(x) \). How does this hurt us?

\[
E \ell(X) = \sum_x p(x) \lceil \log \frac{1}{q(x)} \rceil \leq \sum_x p(x) \left( \log \frac{1}{q(x)} + 1 \right) \quad (8.61)
\]

\[
= \sum_x p(x) \left( \log \frac{p(x)}{q(x)} \frac{1}{p(x)} + 1 \right) \quad (8.62)
\]

\[
= \sum_x p(x) \log \frac{p(x)}{q(x)} + \sum_x p(x) \log \frac{1}{p(x)} + 1 \quad (8.63)
\]

\[
= D(p||q) + H(p) + 1 \quad (8.64)
\]

- Thus, \( D(p||q) \) is per symbol bit penalty for using wrong distribution.

Theorem 8.7.3

*Expected length under \( p(x) \) of code with \( \ell(x) = \lceil \log \frac{1}{q(x)} \rceil \) satisfies*

\[
H(p) + D(p||q) \leq E_p \ell(X) \leq H(p) + D(p||q) + 1 \quad (8.65)
\]

- l.h.s. is the best we can do with the wrong distribution \( q \) when the true distribution is \( p \).
Goal is to find a code with the shortest possible expected length.

- From the above code class, we might think that we want to use codes from the largest class possible (since we might think we’re more likely to get shorter codes).
- We can do better than entropy with non-singular codes, but we want lossless encoding $x = \text{ungzip} (\text{gzip} (x))$.

Kraft revisited

- We proved Kraft inequality is true for instantaneous codes (and vice versa).
- Could it be true for all uniquely decodable codes?
- Could larger class of codes have shorter expected codeword lengths?
- Since larger, we might (naïvely) expect that we could do better.

**Theorem 8.8.1**

Codeword lengths of any uniquely decodable code (not nec. instantaneous) must satisfy Kraft inequality $\sum_i D^{-\ell_i} \leq 1$. Conversely, given a set of codeword lengths that satisfy Kraft, it is possible to construct a uniquely decodable code.

**Proof.**

Proof converse we already saw before (given lengths, we can construct a prefix code which is thus uniquely decodable). Thus we only need prove the first part.
Kraft and uniquely decodable

Proof of Theorem 8.8.1.

- Given: uniquely decodable (not necessarily instantaneous) code with lengths \( \ell(x) \), and length of \( k \)-extension \( \ell(x_1, \ldots, x_k) = \sum_{i=1}^{k} \ell(x_i) \) we wish to prove that \( \sum_x D^{-\ell(x)} \leq 1 \).
- Define \( S = \sum_{x \in X} D^{-\ell(x)} \), then

\[
S^k = \left[ \sum_x D^{-\ell(x)} \right]^k = \sum_{x_1:k \in X^k} D^{-\ell(x_1)} D^{-\ell(x_2)} \ldots D^{-\ell(x_k)} \quad (8.66)
\]

\[
= \sum_{x_1:k \in X^k} D^{-\sum_{i=1}^{k} \ell(x_i)} = \sum_{x_1:k \in X^k} D^{-\ell(x_1:k)} \quad (8.67)
\]

\[
= \sum_{m=1}^{\ell_{\text{max}}} a(m)D^{-m} \quad (8.68)
\]

where \( \ell_{\text{max}} = \max_x \ell(x) \) is the maximum codeword length.
- \( a(m) = \) number of source sequences \( x_1:k \) mapped into code words of length \( m \), i.e.,

\[
a(m) = \left| \left\{ x_1:k \in X^k : \ell(x_1:k) = m \right\} \right| \quad (8.69)
\]

- There are \( D^m \) codewords of length \( m \), and each of them can have (at most) one associated source sequence (since code is uniquely decodable). Hence, \( a(m) \leq D^m \).
...proof of Theorem 8.8.1.

- So continuing,

\[ S^k = \sum_{m=1}^{k\ell_{\text{max}}} a(m)D^{-m} \leq \sum_{m=1}^{k\ell_{\text{max}}} D^m D^{-m} = k\ell_{\text{max}} \quad \forall k \quad (8.70) \]

- So, \( S^k \) (exponential in \( k \)) never greater than \( k\ell_{\text{max}} \) (polynomial in \( k \)) \( \Rightarrow S \leq 1 \).

- Giving \( S = \sum_{x \in X} D^{-\ell(x)} \leq 1 \).

Summary: uniquely decodable vs. instantaneous codes

- Set of achievable codeword lengths the same for uniquely decodable codes and for instantaneous codes.
  \( \Rightarrow \) optimal codeword length bound still holds.

- In fact, this is not surprising since we can get arbitrarily close to entropy rate already using instantaneous code (e.g., Shannon code) with long block words.

- So we can then just consider instantaneous codes with relative impunity.