Class Road Map - IT-I

- L1 (9/25): Overview, Communications, Information, Entropy
- L2 (9/30): Entropy, Mutual Information, KL-Divergence
- L3 (10/2): More KL, Jensen, more Venn, Log Sum, Data Proc. Inequality
- L4 (10/7): Data Proc. Ineq., thermodynamics, Stats, Fano,
- L5 (10/9): M. of Conv, AEP,
- L6 (10/14): AEP, Source Coding, Types
- LX (10/16): Makeup
- L7 (10/21): Types, Univ. Src Coding, Stoc. Procs, Entropy Rates
- L8 (10/23): Entropy rates, HMMs, Coding
- L9 (10/28): Kraft ineq., Shannon Codes, Kraft ineq. II, Huffman
- L10 (10/30): Huffman, Shannon/Fano/Elias
- L11 (11/4): Shannon/Fano/Elias, Games
- LXX (11/6): In class midterm exam
- L12 (11/11): Vet’s day, makeup lecture: Arith. Coding, Background On Channel Capacity
- L13 (11/13): Channel Capacity, Ex. DMC
- L15 (11/20): Joint AEP, Shannon’s 2nd Theorem,
- L16 (11/25): Zero Error Codes, 2nd Thm Conv, Zero Error, \( R = C \), Feedback, Joint Thm, Coding, Hamming Codes
- L17 (11/27):
- L18 (12/2):
- L19 (12/4):
- LXX (12/10): Final exam

Finals Week: December 9th–13th.
Cumulative Outstanding Reading TODOs

- Chapters 1 and 2 in our book (Cover & Thomas, “Information Theory”) (including Fano’s inequality).
- Chapters 3 and 4 in our book (Cover & Thomas, “Information Theory”).
- Sections 11.1 through 11.3 in our book (Cover & Thomas, “Information Theory”).
- Chapter 4 in our book (Cover & Thomas, “Information Theory”).
- Chapter 5 in our book (Cover & Thomas, “Information Theory”).
- Chapter 13 in our book (Cover & Thomas, “Information Theory”) (there is no chapter on arithmetic coding but the lecture slides will be complete, or see MacKay’s online text).
- Chapter 7 in our book (Cover & Thomas, “Information Theory”) on channel capacity.
Homework

- Homework 1 on our assignment dropbox (https://canvas.uw.edu/courses/1319497/assignments), was due Tuesday, Oct 8th, 11:55pm.

- Homework 2 on our assignment dropbox (https://canvas.uw.edu/courses/1319497/assignments), due Friday 10/18/2019, 11:45pm.

- Homework 3 on our assignment dropbox (https://canvas.uw.edu/courses/1319497/assignments), due Tuesday 10/29/2019, 11:45pm.


- Homework 5 on our assignment dropbox (https://canvas.uw.edu/courses/1319497/assignments), due Monday 12/2/2019, 11:45pm.
Properties of (Information) Channel Capacity $C$

- $C \geq 0$ since $I(X; Y) \geq 0$.
- $C \leq \log |\mathcal{X}|$ since $C = \max_{p(x)} I(X; Y) \leq \max H(X) = \log |\mathcal{X}|$.
- $C \leq \log |\mathcal{Y}|$ for same reason. Thus, the alphabet sizes limit the transmission rate.

$I(X; Y) = I_{p(x)}(X; Y)$ is a continuous function of $p(x)$.

Recall, $I(X; Y)$ is a concave function of $p(x)$ for fixed $p(y|x)$. Thus, $I_{\lambda p_1 + (1-\lambda)p_2}(X; Y) \geq \lambda I_{p_1}(X; Y) + (1 - \lambda) I_{p_2}(X; Y)$.

Interestingly, since concave, this makes computing something like the capacity easier. I.e., a local maximum is a global maximum, and computing the capacity for a general channel model is a convex optimization procedure.

Recall also, $I(X; Y)$ is a convex function of $p(y|x)$ for fixed $p(x)$.
Definition 16.2.2 \((M, n)\) code

An \((M, n)\) code for channel \((\mathcal{X}, p(y|x), \mathcal{Y})\) is:

1. An index set \(\{1, 2, \ldots, M\}\)
2. An encoding function \(X^n : \{1, 2, \ldots, M\} \rightarrow \mathcal{X}^n\) yielding codewords \(X^n(1), X^n(2), X^n(3), \ldots, X^n(M)\). Each source message has a codeword, and each codeword is \(n\) code symbols.
3. Decoding function, i.e., \(g : \mathcal{Y}^n \rightarrow \{1, 2, \ldots, M\}\) which makes a “guess” about original message given channel output.

- In an \((M, n)\) code, \(M = \) the number of possible messages to be sent, and \(n = \) number of channel uses by the codewords of the code.
Definition of Error

Definition 16.2.2 (Probability of Error $\lambda_i$ for message $i \in \{1, \ldots, M\}$)

$$
\lambda_i \triangleq \Pr(g(Y^n) \neq i | X^n = x^n(i)) = \sum_{y^n \in Y^n} p(y^n | x^n(i)) 1(g(y^n) \neq i)
$$

(16.3)

Definition 16.2.3 (Max probability of Error $\lambda^{(n)}$ for $(M, n)$ code)

$$
\lambda^{(n)} \triangleq \max_{i \in \{1, 2, \ldots, M\}} \lambda_i
$$

(16.4)
Definition 16.2.2 (Average probability of error $P_{e(n)}$)

\[ P_{e(n)} = \frac{1}{M} \sum_{i=1}^{M} \lambda_i = \Pr(I \neq g(Y^n)) \]  

(16.3)

where $I$ is a r.v. with probability $\Pr(I = i)$ according to a uniform source distribution . . .

\[ = E(1(I \neq g(Y^n))) = \sum_{i=1}^{M} \Pr(g(Y^n) \neq i | X^n = x^n(i))p(i) \]  

(16.4)

with $p(i) = 1/M$.

- A key Shannon’s result is that a small average probability of error means we must have a small maximum probability of error!
Rate

Definition 16.2.2 (Rate $R$ of an $(M, n)$ code)

$$R = \frac{\log M}{n} = \frac{\text{total num. of bits in a source message}}{\text{total num. of channel uses needed to send a message}}$$

(16.3)

- The rate $R$ is in units of bits per channel use, or bits per transmission.

Definition 16.2.3 (Achievability for a given channel)

A given rate $R$ is achievable for a given channel if $\exists$ a sequence of $(\lceil 2^{nR} \rceil, n)$ codes such that the maximal probability of error $\lambda^{(n)} \to 0$ as $n \to \infty$.
Definition 16.2.2 (Capacity of a DMC)

The capacity of a DMC is the largest possible achievable rate.

- So the capacity of a DMC is the rate beyond which the error won’t any longer go to zero with increasing $n$.
- Note: this is a different notion of capacity that we encountered before.
- Before we defined $C = \max_p p(x) I(X; Y)$ as the “information capacity”
- Here we are defining something called the “capacity of a DMC”.
- We have not yet compared the two (but of course we will 😊).
Theorem 16.2.1

All rates below $C \triangleq \max_p I(X; Y)$ are achievable. Specifically, $
forall R < C$, there exists a sequence of $(2^{nR}, n)$ codes with maximum probability of error $\lambda(n) \to 0$ as $n \to \infty$. Conversely, any $(2^{nR}, n)$ sequence of codes with $\lambda(n) \to 0$ as $n \to \infty$ must have that $R < C$.

- Implications: as long as we do not code above capacity we can, for all intents and purposes, code with zero error.
- This is true for all noisy channels representable under this model.
- We’re talking about discrete channels now, but we generalize this to continuous channels in the coming weeks and/or IT-II.
To summarize, random coding is the method of proof to show that if $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with $\lambda(n) \to 0$ as $n \to \infty$.

This might not be the best code, but it is sufficient. It is an existence proof.

Huge literature on coding theory. We’ll discuss Hamming codes.

But many good codes exist today: Turbo codes, Gallager (or low-density-parity-check) codes, and new ones are being proposed often.

Perhaps if there is enough demand, we’ll have a quarter class just on coding theory.

But we have yet to prove the converse . . .
More Discussion

- We next need to show that any sequence of \((2^{nR}, n)\) codes with 
  \(\lambda^{(n)} \to 0\) must have that \(R \leq C\).

- First let's consider the case if \(P_e^{(n)} = 0\), in such case it is easy to show that \(R \leq C\).
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over $\{1, 2, \ldots, M\}$). Sufficient since this is max rate under $M$ messages.
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty).
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over $\{1, 2, \ldots, M\}$). Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. 
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty).
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over $\{1, 2, \ldots, M\}$. Sufficient since this is max rate under $M$ messages.
- First, let's consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

\begin{align}
(16.5)
\end{align}
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over $\{1, 2, \ldots, M\}$. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

\[ nR \]

(16.5)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over $\{1, 2, \ldots, M\}$. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W)$$

(16.5)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W; Y^n)$$

(16.5)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n) \quad (16.1)$$

(16.5)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over $\{1, 2, \ldots, M\}$). Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

\[ nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n) \tag{16.1} \]
\[ \leq I(X^n; Y^n) \quad \text{//Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \tag{16.2} \]
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W;Y^n) = I(W;Y^n)$$

(16.1)

$$\leq I(X^n;Y^n) \quad /\!\!/\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.}$$

(16.2)

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

(16.5)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W;Y^n) = I(W;Y^n)$$ \hspace{1cm} (16.1)

$$\leq I(X^n;Y^n) \quad //\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \quad (16.2)$$

$$= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, X^n) \hspace{1cm} (16.3)$$

(16.5)
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n) \quad (16.1)$$

$$\leq I(X^n; Y^n) \quad //\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \quad (16.2)$$

$$= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, X^n) \quad (16.3)$$

But $Y_i \perp \perp \{Y_{1:i-1}, X_{1:i-1}, X_{i+1:n}\}|X_i$, 

$$\quad (16.5)$$
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n)$$  \hspace{1cm} (16.1)

$$\leq I(X^n; Y^n) \quad \text{//Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \quad \text{ (16.2)}$$

$$= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, X^n)$$ \hspace{1cm} (16.3)

But $Y_i \perp \perp \{Y_{1:i-1}, X_{1:i-1}, X_{i+1:n}\}|X_i$, so we can continue as

$$\text{ (16.5)}$$
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

\[
nR = H(W) = H(W|Y^n) + I(W;Y^n) = I(W;Y^n) \tag{16.1}
\]

\[
\leq I(X^n;Y^n) \quad //\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \tag{16.2}
\]

\[
= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1},X^n) \tag{16.3}
\]

But $Y_i \perp \perp \{Y_{1:i-1},X_{1:i-1},X_{i+1:n}\}|X_i$, so we can continue as

\[
= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \tag{16.5}
\]
Zero Error Codes

- If $P_e^n = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^n = 0$, in such case it is easy to show that $R \leq C$. Then we get

\[ nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n) \quad (16.1) \]
\[ \leq I(X^n; Y^n) \quad /\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \quad (16.2) \]
\[ = H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, X_{i+1:n}) \quad (16.3) \]

But $Y_i \perp \perp \{Y_{1:i-1}, X_{1:i-1}, X_{i+1:n}\}|X_i$, so we can continue as

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \leq \sum_{i} \left[ H(Y_i) - H(Y_i|X_i) \right] \quad (16.4) \]
\[ \leq nC \quad (16.5) \]
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First lets consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n) \quad (16.1)$$

$$\leq I(X^n; Y^n) \quad //\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \quad (16.2)$$

$$= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, X^n) \quad (16.3)$$

But $Y_i \perp \perp \{Y_{1:i-1}, X_{1:i-1}, X_{i+1:n}\}|X_i$, so we can continue as

$$= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \leq \sum_{i} \left[ H(Y_i) - H(Y_i|X_i) \right] \quad (16.4)$$

$$= \sum_{i=1}^{n} I(Y_i; X_i) \quad (16.5)$$
Zero Error Codes

- If $P_e^{(n)} = 0$, then $H(W|Y^n) = 0$ (no uncertainty)
- For simplicity, assume $H(W) = nR = \log M$ (i.e., uniform dist. over \{1, 2, \ldots, M\}. Sufficient since this is max rate under $M$ messages.
- First let's consider the case if $P_e^{(n)} = 0$, in such case it is easy to show that $R \leq C$. Then we get

$$nR = H(W) = H(W|Y^n) + I(W; Y^n) = I(W; Y^n) \quad (16.1)$$

$$\leq I(X^n; Y^n) \quad //\text{Since } W \rightarrow X^n \rightarrow Y^n \text{ and data proc. ineq.} \quad (16.2)$$

$$= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, X^n) \quad (16.3)$$

But $Y_i \perp \perp \{Y_{1:i-1}, X_{1:i-1}, X_{i+1:n}\}|X_i$, so we can continue as

$$= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \leq \sum_{i} \left[H(Y_i) - H(Y_i|X_i)\right] \quad (16.4)$$

$$= \sum_{i=1}^{n} I(Y_i; X_i) \leq nC \quad (16.5)$$
Zero Error Codes

- Thus, $nR \leq nC$ and $R \leq C$ when $P_e^{(n)} = 0$. 
Zero Error Codes

- Thus, \( nR \leq nC \) and \( R \leq C \) when \( P_e^{(n)} = 0 \).
- In fact, the proof shows \( H(W) \leq nC \), which means that \( \max_p H_p(W) \leq nC \) implying that

\[
H(W) \leq \max_p H_p(W) = nR \leq nC
\]

so we get \( R \leq C \) regardless of the source distribution.
Zero Error Codes

Thus, $nR \leq nC$ and $R \leq C$ when $P_e^{(n)} = 0$.

In fact, the proof shows $H(W) \leq nC$, which means that

$$\max_p H_p(W) \leq nC$$

implying that

$$H(W) \leq \max_p H_p(W) = nR \leq nC$$

so we get $R \leq C$ regardless of the source distribution.

It also shows a sub-lemma, namely that $I(X^n; Y^n) \leq nC$ that we’ll use later. Lets name it:
Zero Error Codes

- Thus, \( nR \leq nC \) and \( R \leq C \) when \( P_e(n) = 0 \).
- In fact, the proof shows \( H(W) \leq nC \), which means that \( \max_p H_p(W) \leq nC \) implying that

\[
H(W) \leq \max_p H_p(W) = nR \leq nC
\]

so we get \( R \leq C \) regardless of the source distribution.
- It also shows a sub-lemma, namely that \( I(X^n; Y^n) \leq nC \) that we’ll use later. Let’s name it:

**Lemma 16.3.1**

\[
I(X^n; Y^n) \leq nC
\]
Zero Error Codes

- Thus, $nR \leq nC$ and $R \leq C$ when $P_e^{(n)} = 0$.
- In fact, the proof shows $H(W) \leq nC$, which means that $\max_p H_p(W) \leq nC$ implying that

$$H(W) \leq \max_p H_p(W) = nR \leq nC$$  \hfill (16.6)

so we get $R \leq C$ regardless of the source distribution.
- It also shows a sub-lemma, namely that $I(X^n; Y^n) \leq nC$ that we’ll use later. Let’s name it:

**Lemma 16.3.1**

$$I(X^n; Y^n) \leq nC$$  \hfill (16.7)

- We also need Fano’s inequality. Recall, before it took the form

$$H(X|Y) \leq 1 + P_e \log |\mathcal{X}|$$  \hfill (16.8)
The next slide is from Lecture 4.
Fano’s Inequality

**Theorem 16.3.7**

\[
H(P_e) + P_e \log(|\mathcal{X}| - 1) \geq H(X|\hat{X}) \geq H(X|Y)
\]  \hspace{1cm} (16.28)

- So \( P_e = 0 \) requires that \( H(X|Y) = 0! \)
- Note, the theorem simplifies (and implies)
  \[
  1 + P_e \log(|\mathcal{X}|) \geq H(X|Y), \text{ or}
  \]
  \[
P_e \geq \frac{H(X|Y) - 1}{\log |\mathcal{X}|}
\]  \hspace{1cm} (16.29)

yielding a lower bound on the error.

- This will be used to prove the converse to Shannon’s coding theorem, i.e., that any code with probability of error \( \to 0 \) as the block length increases must have a rate \( R < C = \) the capacity of the channel (to be defined).
Fano’s Lemma (needed for current proof)

**Theorem 16.3.2 (Fano)**

For a DMC with codebook $C$ and uniformly distributed input messages ($H(W) = nR$) and $P_e^{(n)} = Pr(W \neq g(Y^n))$, then

$$H(X^n|Y^n) \leq 1 + P_e^{(n)}nR$$  \hspace{1cm} (16.9)
Fano’s Lemma (needed for current proof)

**Theorem 16.3.2 (Fano)**

For a DMC with codebook $C$ and uniformly distributed input messages $(H(W) = nR)$ and $P_e(n) = Pr(W \neq g(Y^n))$, then

$$H(X^n|Y^n) \leq 1 + P_e(n)nR \quad (16.9)$$

**Proof.**

Let $E \triangleq 1\{W \neq \hat{W}\}$. 

...
Fano’s Lemma (needed for current proof)

**Theorem 16.3.2 (Fano)**

For a DMC with codebook $C$ and uniformly distributed input messages ($H(W) = nR$) and $P_e^{(n)} = Pr(W \neq g(Y^n))$, then

$$H(X^n|Y^n) \leq 1 + P_e^{(n)}nR$$  \hspace{1cm} (16.9)

**Proof.**

Let $E \triangleq 1 \{ W \neq \hat{W} \}$. Then we get:

$$H(E, W|Y^n)$$

(16.11)
Fano’s Lemma (needed for current proof)

**Theorem 16.3.2 (Fano)**

For a DMC with codebook $C$ and uniformly distributed input messages ($H(W) = nR$) and $P_e(n) = Pr(W \neq g(Y^n))$, then

$$H(X^n|Y^n) \leq 1 + P_e(n)nR \quad (16.9)$$

**Proof.**

Let $E \triangleq 1\left\{ W \neq \hat{W} \right\}$. Then we get:

$$H(E, W|Y^n) = H(W|Y^n) + H(E|Y^n, W) = 0 \quad (16.10)$$

$$H(X^n|Y^n) \leq 1 + P_e(n)nR \quad (16.11)$$
Theorem 16.3.2 (Fano)

For a DMC with codebook $C$ and uniformly distributed input messages ($H(W) = nR$) and $P_e^{(n)} = Pr(W \neq g(Y^n))$, then

$$H(X^n|Y^n) \leq 1 + P_e^{(n)}nR \quad (16.9)$$

Proof.

Let $E \triangleq 1\{W \neq \hat{W}\}$. Then we get:

$$H(E, W|Y^n) = H(W|Y^n) + H(E|Y^n, W) = 0$$

$$= H(E|Y^n) + H(W|Y^n, E) \leq 1$$
Fano’s Lemma (needed for proof)

Proof continued.

\[
H(W|Y^n, E) = \Pr(E = 0) H(W|Y^n, E = 0) + \Pr(E = 1) H(W|Y^n, E = 1)
\]

\[
= 1 - P_e^{(n)} \log(2^{nR} - 1) + P_e^{(n)} \log(2^{nR} - 1)
\]

\[
= 0
\]

(16.12)
Fano’s Lemma (needed for proof)

proof continued.

\[
H(W|Y^n, E) = \Pr(E = 0) H(W|Y^n, E = 0)
\]

\[
1 - P_e^{(n)} = 0
\]

\[
+ \Pr(E = 1) H(W|Y^n, E = 1)
\]

\[
= P_e^{(n)} \log(2^{nR} - 1)
\]

\[
\leq P_e^{(n)} nR
\] (16.13)
proof continued.

\[ H(W|Y^n, E) = \Pr(E = 0) H(W|Y^n, E = 0) \]
\[ + \Pr(E = 1) H(W|Y^n, E = 1) \]
\[ \leq P_e^{(n)} nR \]
\[ \Rightarrow \quad H(W|Y^n) \leq 1 + P_e^{(n)} nR \]
Fano’s Lemma (needed for proof)

proof continued.

\[
H(W|Y^n, E) = \frac{\Pr(E = 0) \cdot H(W|Y^n, E = 0)}{1 - P_e^{(n)}} + \frac{\Pr(E = 1) \cdot H(W|Y^n, E = 1)}{P_e^{(n)}} \leq P_e^{(n)} nR
\]

\[
\leq P_e^{(n)} nR \implies H(W|Y^n) \leq 1 + P_e^{(n)} nR \tag{16.13}
\]

but \(X^n = X^n(W)\) and functions of random variables can only reduce entropy.
proof continued.

\[ H(W|Y^n, E) = \Pr(E=0) H(W|Y^n, E=0) \]
\[ + \Pr(E=1) H(W|Y^n, E=1) \]
\[ \leq P_e^{(n)} nR \Rightarrow H(W|Y^n) \leq 1 + P_e^{(n)} nR \quad (16.13) \]

but \( X^n = X^n(W) \) and functions of random variables can only reduce entropy. So we get:

\[ H(X^n|Y^n) \leq H(W|Y^n) \leq 1 + P_e^{(n)} nR \quad (16.14) \]
Sequence of codes w. vanishing error must have $R < C$.

The converse states: any sequence of $(2^{nR}, n)$ codes with $\lambda^{(n)} \to 0$ must have that $R \leq C$.

**Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.**

- Average prob. goes to zero if max probability does: 
  \[ \lambda^{(n)} \to 0 \Rightarrow P_e^{(n)} \to 0, \text{ where } P_e^{(n)} = \frac{1}{2^{nR}} \sum_{i=1}^{2^{nR}} \lambda_i \]

- Lets set $H(W) = nR$ for now (i.e., $W$ uniform on \{1, 2, \ldots, M = 2^{nR}\}). Again, makes the proof a bit easier and doesn’t affect relationship between $R$ and $C$.

- So, $\Pr(W = \hat{W}) = P_e^{(n)} = \frac{1}{M} \sum_{i=1}^{M} \lambda_i$ as we saw in last lecture. ...
Sequence of codes w. vanishing error must have $R < C$.

Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

\begin{equation}
R < C \tag{16.19}
\end{equation}
Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

$$nR = H(W)$$

(16.19)
Sequence of codes w. vanishing error must have $R < C$.

Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

\[ nR = H(W) = H(W|Y^n) + I(W;Y^n) \]  \hspace{1cm} (16.15)

\[ \leq H(W|Y^n) + I(X^n(W);Y^n) \] \hspace{1cm} (16.16)

\[ \leq 1 + P(n)e^{nR} + I(X^n(W);Y^n) \] \hspace{1cm} (16.17)

\[ \leq 1 + P(n)e^{nR} + nC \] \hspace{1cm} (16.18)

\[ R \leq P(n)e^{R} + 1/n + C \] \hspace{1cm} (16.19)

As $n \to \infty$, $P(n)e^{R} \to 0$, and $1/n \to 0$ as well.

Thus $\Rightarrow R < C$.  \hspace{1cm} (16.20)
Sequence of codes w. vanishing error must have $R < C$.

Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

\[
nR = H(W) = H(W|Y^n) + I(W; Y^n) \\ 
\leq H(W|Y^n) + I(X^n(W); Y^n) \\
\text{//Since } W \rightarrow X^n \rightarrow Y^n
\]

(16.15) (16.16) (16.19)
Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$. 

\[ nR = H(W) = H(W|Y^n) + I(W; Y^n) \]  \hspace{1cm} (16.15)

\[ \leq H(W|Y^n) + I(X^n(W); Y^n) \quad \text{//Since } W \rightarrow X^n \rightarrow Y^n \]  \hspace{1cm} (16.16)

\[ \leq 1 + P_e^{(n)} nR + I(X^n(W); Y^n) \quad \text{//by Fano} \]  \hspace{1cm} (16.17)

\[ \Rightarrow R < P_e^{(n)} + 1/n + C \]  \hspace{1cm} (16.19)
Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.  

\[ nR = H(W) = H(W|Y^n) + I(W; Y^n) \]  
\[ \leq H(W|Y^n) + I(X^n(W); Y^n) \quad \text{//Since } W \to X^n \to Y^n \]  
\[ \leq 1 + P_e^{(n)}nR + I(X^n(W); Y^n) \quad \text{//by Fano} \]  
\[ \leq 1 + P_e^{(n)}nR + nC \quad \text{//by lemma 16.3.1} \]
Sequence of codes w. vanishing error must have $R < C$.

Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

\[
nR = H(W) = H(W|Y^n) + I(W;Y^n) \tag{16.15}
\]
\[
\leq H(W|Y^n) + I(X^n(W);Y^n) \quad \text{//Since } W \to X^n \to Y^n \tag{16.16}
\]
\[
\leq 1 + P_e^{(n)}nR + I(X^n(W);Y^n) \quad \text{//by Fano} \tag{16.17}
\]
\[
\leq 1 + P_e^{(n)}nR + nC \quad \text{//by lemma 16.3.1} \tag{16.18}
\]
\[
\Rightarrow R \leq P_e^{(n)} R + 1/n + C \tag{16.19}
\]
Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

\[
nR = H(W) = H(W|Y^n) + I(W;Y^n) = H(W|Y^n) + I(X^n(W);Y^n) \quad /\text{Since } W \to X^n \to Y^n
\]

\[
\leq 1 + P_e^{(n)}nR + I(X^n(W);Y^n) \quad /\text{by Fano}
\]

\[
\leq 1 + P_e^{(n)}nR + nC \quad /\text{by lemma 16.3.1}
\]

\[
\Rightarrow R \leq P_e^{(n)}R + 1/n + C
\]

Now as $n \to \infty$, $P_e^{(n)} \to 0$, and $1/n \to 0$ as well.
Proof that $\lambda^{(n)} \to 0$ as $n \to \infty \Rightarrow R < C$.

\[ nR = H(W) = H(W|Y^n) + I(W;Y^n) \]
\[ \leq H(W|Y^n) + I(X^n(W);Y^n) \quad \text{//Since } W \to X^n \to Y^n \]
\[ \leq 1 + P_e^{(n)}nR + I(X^n(W);Y^n) \quad \text{//by Fano} \]
\[ \leq 1 + P_e^{(n)}nR + nC \quad \text{//by lemma 16.3.1} \]
\[ \Rightarrow R \leq P_e^{(n)}R + 1/n + C \]

Now as $n \to \infty$, $P_e^{(n)} \to 0$, and $1/n \to 0$ as well. Thus

\[ \Rightarrow R < C \]
Sequence of codes w. vanishing error must have $R < C$.

Also,

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

(16.21)

This means that:
Sequence of codes w. vanishing error must have $R < C$.

Also,

$$P_e(n) \geq 1 - \frac{C}{R} - \frac{1}{nR} \quad (16.21)$$

This means that:

- if $n \to \infty$ and $R > C$, then error lower bound is strictly positive, and depends on $1 - C/R$. 


Sequence of codes w. vanishing error must have $R < C$.

Also,

$$P_e(n) \geq 1 - \frac{C}{R} - \frac{1}{nR} \quad (16.21)$$

This means that:

- if $n \to \infty$ and $R > C$, then error lower bound is strictly positive, and depends on $1 - C/R$.

- Even for small $n$, $P_e(n) > 0$, since otherwise, if $P_e(n_0) = 0$ for some code, we can concatenate code to get large $n$ same rate code, contradicting $P_e > 0$. 
Sequence of codes w. vanishing error must have $R < C$.

Lower bound on error:

\[ P_e^{(n)} \geq \max(0, 1 - \frac{C}{R}) \quad (16.22) \]
Sequence of codes w. vanishing error must have $R < C$.

Lower bound on error:

$$P_e^{(n)} \geq \max(0, 1 - \frac{C}{R})$$  \hspace{1cm} (16.22)

generates this plot (lower bound on error):
Sequence of codes w. vanishing error must have $R < C$.

Lower bound on error:

$$P_e^{(n)} \geq \max(0, 1 - \frac{C}{R})$$  \hspace{1cm} (16.23)
Sequence of codes w. vanishing error must have $R < C$.

Lower bound on error:

$$
P_e^{(n)} \geq \max(0, 1 - \frac{C}{R})
$$

(16.23)

also generates this plot:

Meaning

$$
P_e \propto e^{-nE(R)}
$$

(16.24)
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

$nR$
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

\[ nR = H(W) \]
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

$$nR = H(W) = H(X^n(W)) \quad \text{//if codewords distinct} \quad (16.25)$$
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

$$nR = H(W) = H(X^n(W)) \quad \text{//if codewords distinct}$$  \hspace{3cm} (16.25)

$$= H(X^n|Y^n) + I(X^n; Y^n)$$

$$= 0 \text{ since } P_e=0$$  \hspace{3cm} (16.29)
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

$$nR = H(W) = H(X^n(W)) \quad \text{//if codewords distinct}$$

(16.25)

$$= H(X^n|Y^n) + I(X^n; Y^n) = I(X^n; Y^n)$$

(16.26)

=0 since $P_e=0$
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

\[
nR = H(W) = H(X^n(W)) \quad /\text{if codewords distinct} \quad (16.25)
\]

\[
= H(X^n|Y^n) + I(X^n; Y^n) = I(X^n; Y^n) \quad (16.26)
\]

\[
= 0 \quad \text{since } P_e=0
\]

\[
= H(Y^n) - H(Y^n|X^n) \quad (16.27)
\]
What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

\[
nR = H(W) = H(X^n(W)) \quad /\text{if codewords distinct} \quad (16.25)
\]

\[
= H(X^n|Y^n) + I(X^n; Y^n) = I(X^n; Y^n) \quad (16.26)
\]

=0 since $P_e=0$

\[
= H(Y^n) - H(Y^n|X^n) \quad (16.27)
\]

\[
= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \quad (16.28)
\]
What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

\[
nR = H(W) = H(X^n(W)) \quad \text{if codewords distinct} \quad (16.25)
\]

\[
= H(X^n|Y^n) + I(X^n; Y^n) = I(X^n; Y^n) \quad (16.26)
\]

\[
= 0 \quad \text{since } P_e = 0
\]

\[
= H(Y^n) - H(Y^n|X^n) \quad (16.27)
\]

\[
= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \quad (16.28)
\]

\[
= \sum_{i} H(Y_i) - \sum_{i} H(Y_i|X_i) \quad \text{if all } Y_i's \text{ are indep} \quad (16.29)
\]
Zero-error capacity

- What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

\[
nR = H(W) = H(X^n(W)) \quad \text{//if codewords distinct} \tag{16.25}
\]

\[
= H(X^n|Y^n) + I(X^n; Y^n) = I(X^n; Y^n) \tag{16.26}
\]

\[
= 0 \text{ since } P_e = 0
\]

\[
= H(Y^n) - H(Y^n|X^n) \tag{16.27}
\]

\[
= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \tag{16.28}
\]

\[
= \sum_{i} H(Y_i) - \sum_{i} H(Y_i|X_i) \quad \text{//if all } Y_i \text{'s are indep} \tag{16.29}
\]

\[
= \sum_{i} I(X_i; Y_i) \tag{16.30}
\]
Zero-error capacity

What if we insist on $R = C$ and $P_e = 0$. In such case, what are the requirements of and consequences for any such code.

\[ nR = H(W) = H(X^n(W)) \quad //\text{if codewords distinct} \hspace{1cm} (16.25) \]
\[ = H(X^n | Y^n) + I(X^n; Y^n) = I(X^n; Y^n) \quad (16.26) \]
\[ = 0 \quad \text{since } P_e = 0 \]
\[ = H(Y^n) - H(Y^n | X^n) \quad (16.27) \]
\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i | X_i) \quad (16.28) \]
\[ = \sum_{i} H(Y_i) - \sum_{i} H(Y_i | X_i) \quad //\text{if all } Y_i \text{'s are indep} \hspace{1cm} (16.29) \]
\[ = \sum_{i} I(X_i; Y_i) \quad (16.30) \]
\[ = nC \quad //\text{if we choose } p^*(x) \in \arg\max_{p(x)} I(X; Y) \hspace{1cm} (16.31) \]
\[ (16.32) \]
Zero-error capacity

So there are 3 conditions for equality, $R = C$, namely

1. all codewords must be distinct
So there are 3 conditions for equality, $R = C$, namely

1. all codewords must be distinct
2. $Y_i$'s are independent
So there are 3 conditions for equality, $R = C$, namely

1. all codewords must be distinct
2. $Y_i$’s are independent
3. distribution on $x$ is $p^*(x)$, a capacity achieving distribution.
Does feedback help for DMC

Consider a sequence of channel uses.
Does feedback help for DMC

Consider a sequence of channel uses.

Without Feedback

\[\begin{align*}
  X_1 &\rightarrow Y_1 \\
  X_2 &\rightarrow Y_2 \\
  X_3 &\rightarrow Y_3 \\
  \vdots &\vdots \\
  X_n &\rightarrow Y_n
\end{align*}\]
Does feedback help for DMC

Consider a sequence of channel uses.

Without Feedback

\[
\begin{align*}
X_1 & \rightarrow Y_1 \\
X_2 & \rightarrow Y_2 \\
X_3 & \rightarrow Y_3 \\
& \vdots \\
X_n & \rightarrow Y_n
\end{align*}
\]

With Feedback

\[
Y_i \perp \{\text{all else}\} \mid X_i
\]

Can this help? I.e., can this increase \( R \)?
Does feedback help for DMC

Consider a sequence of channel uses.

Without Feedback

\[ X_1 \rightarrow Y_1 \]
\[ X_2 \rightarrow Y_2 \]
\[ X_3 \rightarrow Y_3 \]
\[ \vdots \]
\[ X_n \rightarrow Y_n \]

With Feedback

\[ \text{Encoder} \]
\[ W \]
\[ X_i(W, Y_{1:i-1}) \]
\[ \text{Channel} \ p(y|x) \]
\[ Y_i \]
\[ \text{Decoder} \]
\[ \hat{W} \]

Another way of looking at it is:

\[ Y_i \perp \{\text{all else}\} | X_i \]
Does feedback help for DMC

Consider a sequence of channel uses.

Without Feedback

\[
\begin{align*}
X_1 &\rightarrow Y_1 \\
X_2 &\rightarrow Y_2 \\
X_3 &\rightarrow Y_3 \\
\vdots &\vdots \\
X_n &\rightarrow Y_n 
\end{align*}
\]

With Feedback

\[
\begin{align*}
X_1 &\rightarrow Y_1 \\
X_2 &\rightarrow Y_2 \\
X_3 &\rightarrow Y_3 \\
\vdots &\vdots \\
X_n &\rightarrow Y_n \\
Y_i \perp \{\text{all else}\} | X_i 
\end{align*}
\]

Another way of looking at it is:

\[
\begin{align*}
W &\rightarrow X_i(W, Y_{1:i-1}) \\
&\rightarrow Y_i \\
&\rightarrow \hat{W} \\
\end{align*}
\]

Can this help? I.e., can this increase \( C \)?
Does feedback help for DMC

A:

Does feedback help for DMC
Does feedback help for DMC

A: No.
Does feedback help for DMC

- A: No.
- Intuition: w/o memory, feedback tells us nothing more than what we already know, namely $p(y|x)$. 

Does feedback help for DMC

- A: No.
- Intuition: w/o memory, feedback tells us nothing more than what we already know, namely $p(y|x)$.
- Can feedback made decoding easier? Yes, consider binary erasure channel, when we get $Y = e$ we just re-transmit.
Does feedback help for DMC

- A: No.
- Intuition: w/o memory, feedback tells us nothing more than what we already know, namely $p(y|x)$.
- Can feedback made decoding easier? Yes, consider binary erasure channel, when we get $Y = e$ we just re-transmit.
- Can feedback help for channels with memory?
Does feedback help for DMC

- **A: No.**
- **Intuition:** w/o memory, feedback tells us nothing more than what we already know, namely $p(y|x)$.
- **Can feedback made decoding easier?** Yes, consider binary erasure channel, when we get $Y = e$ we just re-transmit.
- **Can feedback help for channels with memory?** In general, yes.
Feedback for DMC

**Definition 16.6.1** \((2^{nR}, n)\) feedback code

Such a code is the encoder \(X_i(W, Y_{1:i-1})\), a decoder \(g : Y^n \to \{1, 2, \ldots, 2^{nR}\}\), and \(P_e^{(n)} = \Pr(g(Y^n) \neq W)\) for \(H(W) = nR\) (uniform).

**Definition 16.6.2** (Capacity)

The capacity with feedback \(C_{FB}\) of a DMC is the max of all rates achievable by feedback codes.

**Theorem 16.6.3**

\[
C_{FB} = C \triangleq \max_{p(x)} I(X; Y) \quad \text{for a DMC} \quad (16.33)
\]
Feedback codes for DMC

Proof.

- Clearly, $C_{FB} \geq C$, since FB code is a generalization.
- Next, we use $W$ instead of $X$ and bound $R$.
- We have

$$H(W) = H(W|Y^n) + I(W;Y^n) \leq 1 + P_e(n)nR + I(W;Y^n)$$

//by Fano (16.35)

- We next bound $I(W;Y^n)$

...
... proof continued.

\[ I(W; Y^n) = H(Y) - H(Y|W) \]
Feedback codes for DMC

... proof continued.

\[ I(W; Y^n) = H(Y) - H(Y|W) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W) \]

(16.36)

\[ \sum_{i=1}^{n} I(X_i; Y_i) \leq nC \]

(16.39)
Proof continued:

\[ I(W; Y^n) = H(Y) - H(Y|W) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W) \]

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W, X_i) \quad //\text{note } X_i = f(W, Y_{1:i-1}) \]

(16.36)

(16.37)

(16.39)
Feedback codes for DMC

... proof continued.

\[ I(W; Y^n) = H(Y) - H(Y|W) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W) \]

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W, X_i) \]  
//note \( X_i = f(W, Y_{1:i-1}) \)

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

\[ (16.36) \]

\[ (16.37) \]

\[ (16.39) \]
Feedback codes for DMC

... proof continued.

\[ I(W; Y^n) = H(Y) - H(Y|W) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W) \]

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W, X_i) \quad \text{//note } X_i = f(W, Y_{1:i-1}) \]

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

\[ \leq \sum_{i} I(X_i; Y_i) \]
Feedback codes for DMC

\[ I(W; Y^n) = H(Y) - H(Y|W) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W) \]  
(16.36)

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W, X_i) \quad \text{//note } X_i = f(W, Y_{1:i-1}) \]  
(16.37)

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \leq \sum_{i} H(Y_i) - \sum_{i=1}^{n} H(Y_i|X_i) \]  
(16.38)

\[ = \sum_{i} I(X_i; Y_i) \]  
(16.39)
Feedback codes for DMC

... proof continued.

\[ I(W; Y^n) = H(Y) - H(Y|W) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W) \]

(16.36)

\[ = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{1:i-1}, W, X_i) \quad \text{//note } X_i = f(W, Y_{1:i-1}) \]

(16.37)

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

(16.38)

\[ = \sum_{i} I(X_i; Y_i) \leq nC \]

(16.39)
Thus we have

\[
H(W) \leq 1 + P_e^{(n)} nR + nC \tag{16.40}
\]

\[
\Rightarrow nR \leq 1 + P_e^{(n)} nR + nC \tag{16.41}
\]
Thus we have

\[ H(W) \leq 1 + P_e^{(n)} nR + nC \]  \hspace{1cm} (16.40)

\[ \Rightarrow nR \leq 1 + P_e^{(n)} nR + nC \]  \hspace{1cm} (16.41)

The implication follows since the first inequality holds for all \( H(W) \) including the maximum case at which \( H(W) = nR \).
Feedback codes for DMC

... proof continued.

Thus we have

\[ H(W) \leq 1 + P_e^{(n)} nR + nC \]  \hspace{1cm} (16.40)

\[ \Rightarrow nR \leq 1 + P_e^{(n)} nR + nC \]  \hspace{1cm} (16.41)

- The implication follows since the first inequality holds for all \( H(W) \) including the maximum case at which \( H(W) = nR \).
- This gives \( R \leq \frac{1}{n} + P_e^{(n)} R + C \) or \( R \leq C \) as \( n \rightarrow \infty \).
Feedback codes for DMC

... proof continued.

Thus we have

\[ H(W) \leq 1 + P_e^{(n)} nR + nC \]  \hspace{1cm} (16.40)

\[ \Rightarrow nR \leq 1 + P_e^{(n)} nR + nC \]  \hspace{1cm} (16.41)

- The implication follows since the first inequality holds for all \( H(W) \) including the maximum case at which \( H(W) = nR \).
- This gives \( R \leq \frac{1}{n} + P_e^{(n)} R + C \) or \( R \leq C \) as \( n \to \infty \).
- Thus feedback doesn’t help
Joint Source/Channel Theorem

- Data compression: We now know that it is possible to achieve error free compression if our average rate of compression, $R$, measured in units of bits per source symbol, is such that $R > H$ where $H$ is the entropy of the generating source distribution.
Joint Source/Channel Theorem

- Data compression: We now know that it is possible to achieve error-free compression if our average rate of compression, $R$, measured in units of bits per source symbol, is such that $R > H$ where $H$ is the entropy of the generating source distribution.

- Data Transmission: We now know that it is possible to achieve error-free communication and transmission of information if $R < C$, where $R$ is the average rate of information sent (units of bits per channel use), and $C$ is the capacity of the channel.
Data compression: We now know that it is possible to achieve error free compression if our average rate of compression, $R$, measured in units of bits per source symbol, is such that $R > H$ where $H$ is the entropy of the generating source distribution.

Data Transmission: We now know that it is possible to achieve error free communication and transmission of information if $R < C$, where $R$ is the average rate of information sent (units of bits per channel use), and $C$ is the capacity of the channel.

Q: Does this mean that if $H < C$, we can reliably send a source of entropy $H$ over a channel of capacity $C$?
Data compression: We now know that it is possible to achieve error free compression if our average rate of compression, $R$, measured in units of bits per source symbol, is such that $R > H$ where $H$ is the entropy of the generating source distribution.

Data Transmission: We now know that it is possible to achieve error free communication and transmission of information if $R < C$, where $R$ is the average rate of information sent (units of bits per channel use), and $C$ is the capacity of the channel.

Q: Does this mean that if $H < C$, we can reliably send a source of entropy $H$ over a channel of capacity $C$?

This seems intuitively reasonable.
Joint Source/Channel Theorem: process

The process would go something as follows:

1. Compress a source down to its entropy, using Huffman, LZ, arithmetic coding, etc.
Joint Source/Channel Theorem: process

The process would go something as follows:

1. Compress a source down to its entropy, using Huffman, LZ, arithmetic coding, etc.
2. Transmit it over a channel.
The process would go something as follows:

1. Compress a source down to its entropy, using Huffman, LZ, arithmetic coding, etc.
2. Transmit it over a channel.
3. If all sources could share the same channel, would be very useful.
Joint Source/Channel Theorem: process

The process would go something as follows:

1. Compress a source down to its entropy, using Huffman, LZ, arithmetic coding, etc.
2. Transmit it over a channel.
3. If all sources could share the same channel, would be very useful.
4. I.e., perhaps the same channel coding scheme could be used regardless of the source, if the source is first compressed down to the entropy. The channel encoder/decoder need not know anything about the original source (or how to encode it).
Joint Source/Channel Theorem: process

The process would go something as follows:

1. Compress a source down to its entropy, using Huffman, LZ, arithmetic coding, etc.
2. Transmit it over a channel.
3. If all sources could share the same channel, would be very useful.
4. I.e., perhaps the same channel coding scheme could be used regardless of the source, if the source is first compressed down to the entropy. The channel encoder/decoder need not know anything about the original source (or how to encode it).
5. Joint source/channel decoding as in the following figure:
Joint Source/Channel Theorem: process

The process would go something as follows:

1. Compress a source down to its entropy, using Huffman, LZ, arithmetic coding, etc.
2. Transmit it over a channel.
3. If all sources could share the same channel, would be very useful.
4. I.e., perhaps the same channel coding scheme could be used regardless of the source, if the source is first compressed down to the entropy. The channel encoder/decoder need not know anything about the original source (or how to encode it).
5. Joint source/channel decoding as in the following figure:

6. Maybe obvious now, but at the time (1940s) it was a revolutionary idea!
Joint Source/Channel Theorem

- Source: $V \in \mathcal{V}$ that satisfies AEP (e.g., stationary ergodic).
Joint Source/Channel Theorem

- **Source**: $V \in \mathcal{V}$ that satisfies AEP (e.g., stationary ergodic).
- **Send** $V_{1:n} = V_1, V_2, \ldots, V_n$ over channel, entropy rate $H(\mathcal{V})$ of stochastic process (if i.i.d., $H(\mathcal{V}) = H(V_i), \forall i$).
Joint Source/Channel Theorem

- Source: $V \in \mathcal{V}$ that satisfies AEP (e.g., stationary ergodic).
- Send $V_{1:n} = V_1, V_2, \ldots, V_n$ over channel, entropy rate $H(\mathcal{V})$ of stochastic process (if i.i.d., $H(\mathcal{V}) = H(V_i), \forall i$).
- $V_{1:n} \rightarrow \text{Encoder} \rightarrow X^n \rightarrow \text{Channel} \rightarrow Y^n \rightarrow \text{Decoder} \rightarrow \hat{V}_{1:n}$
Joint Source/Channel Theorem

- **Source:** \( V \in \mathcal{V} \) that satisfies AEP (e.g., stationary ergodic).
- **Send** \( V_1:n = V_1, V_2, \ldots, V_n \) over channel, entropy rate \( H(\mathcal{V}) \) of stochastic process (if i.i.d., \( H(\mathcal{V}) = H(V_i), \forall i \)).
- \( V_1:n \rightarrow \) Encoder \( \rightarrow X^n \rightarrow \) Channel \( \rightarrow Y^n \rightarrow \) Decoder \( \rightarrow \hat{V}_1:n \)
- **Error probability and setup:**

\[
P_e^{(n)} = P(V_1:n \neq \hat{V}_1:n) \quad (16.42)
= \sum_{y_1:n, v_1:n} \Pr(v_1:n) \Pr(y_1:n|X^n(v_1:n)) 1\{g(y_1:n) \neq v_1:n\} \quad (16.43)
\]
Joint Source/Channel Theorem

Theorem 16.7.1 (Source/Channel Coding Theorem)

if $V_1:n$ satisfies AEP, then $\exists$ a sequence of $(2^nR, n)$ codes with $P_e^{(n)} \to 0$ if $H(V) < C$. If $R < C$, then the error $\leq \epsilon$ which we can make as small as we wish.
Theorem 16.7.1 (Source/Channel Coding Theorem)

If $V_1:n$ satisfies AEP, then $\exists$ a sequence of $(2^{nR}, n)$ codes with $P_e^{(n)} \to 0$ if $H(V) < C$. Conversely, if $H(V) > C$, then $P_e^{(n)} > 0$ for all $n$ and cannot send with arbitrarily low probability of error.
Joint Source/Channel Theorem

Theorem 16.7.1 (Source/Channel Coding Theorem)

If $V_{1:n}$ satisfies AEP, then $\exists$ a sequence of $(2^{nR}, n)$ codes with $P_e^{(n)} \to 0$ if $H(V) < C$. Conversely, if $H(V) > C$, then $P_e^{(n)} > 0$ for all $n$ and cannot send with arbitrarily low probability of error.

Proof.

...
Joint Source/Channel Theorem

**Theorem 16.7.1 (Source/Channel Coding Theorem)**

If $V_{1:n}$ satisfies AEP, then $\exists$ a sequence of $(2^{nR}, n)$ codes with $P_e^{(n)} \to 0$ if $H(V) < C$. Conversely, if $H(V) > C$, then $P_e^{(n)} > 0$ for all $n$ and cannot send with arbitrarily low probability of error.

**Proof.**

- If $V$ satisfies AEP, then $\exists$ a set $A_\epsilon^{(n)}$ with $|A_\epsilon^{(n)}| \leq 2^n(H(V) + \epsilon)$ ($A_\epsilon^{(n)}$ has all the probability).

...
Theorem 16.7.1 (Source/Channel Coding Theorem)

If $V_{1:n}$ satisfies AEP, then $\exists$ a sequence of $(2^{nR}, n)$ codes with $P_e(n) \to 0$ if $H(V) < C$. Conversely, if $H(V) > C$, then $P_e(n) > 0$ for all $n$ and cannot send with arbitrarily low probability of error.

Proof.

- If $V$ satisfies AEP, then $\exists$ a set $A_\varepsilon(n)$ with $|A_\varepsilon(n)| \leq 2^{n(H(V) + \varepsilon)}$ ($A_\varepsilon(n)$ has all the probability).

- We only encode the typical set, and signal an error otherwise. This contributes $\varepsilon$ to $P_e$. ...
Joint Source/Channel Theorem

**Theorem 16.7.1 (Source/Channel Coding Theorem)**

if $V_1:n$ satisfies AEP, then $\exists$ a sequence of $(2^{nR}, n)$ codes with $P_e(n) \to 0$

if $H(V) < C$. Conversely, if $H(V) > C$, then $P_e(n) > 0$ for all $n$ and cannot send with arbitrarily low probability of error.

**Proof.**

- If $V$ satisfies AEP, then $\exists$ a set $A_{\epsilon}^{(n)}$ with $|A_{\epsilon}^{(n)}| \leq 2^{n(H(V)+\epsilon)}$ ($A_{\epsilon}^{(n)}$ has all the probability).
- We only encode the typical set, and signal an error otherwise. This contributes $\epsilon$ to $P_e$.
- We index elements of $A_{\epsilon}^{(n)}$ as $\{1, 2, \ldots, 2^{n(H+\epsilon)}\}$, so need $n(H + \epsilon)$ bits.
Joint Source/Channel Theorem

**Theorem 16.7.1 (Source/Channel Coding Theorem)**

If $V_{1:n}$ satisfies AEP, then $\exists$ a sequence of $(2^{nR}, n)$ codes with $P_e(n) \to 0$ if $H(V) < C$. Conversely, if $H(V) > C$, then $P_e(n) > 0$ for all $n$ and cannot send with arbitrarily low probability of error.

**Proof.**

- If $V$ satisfies AEP, then $\exists$ a set $A_{\epsilon}(n)$ with $|A_{\epsilon}(n)| \leq 2^{n(H(V)+\epsilon)}$ ($A_{\epsilon}(n)$ has all the probability).
- We only encode the typical set, and signal an error otherwise. This contributes $\epsilon$ to $P_e$.
- We index elements of $A_{\epsilon}(n)$ as $\{1, 2, \ldots, 2^{n(H+\epsilon)}\}$, so need $n(H + \epsilon)$ bits.
- This gives a rate of $R = H(V) + \epsilon$. If $R < C$ then the error $< \epsilon$ which we can make as small as we wish.

...
Joint Source/Channel Theorem

... proof continued.

\[ P(\epsilon) = \Pr(\mathbb{V}_1:n \neq \hat{\mathbb{V}}_1:n) \leq \Pr(\mathbb{V}_1:n \notin A(n)) + \Pr(g(Y_n) \neq V_n | V_n \in A(n)) < \epsilon \] since \( R < C \)

\[ \leq \epsilon + \epsilon = 2\epsilon \] (16.46)

And the first part of the theorem is proved.

To show the converse, show that \( P(\epsilon) \to 0 \Rightarrow H(\mathbb{V}) \leq C \) for source channel codes.
Joint Source/Channel Theorem

... proof continued.

Then

\[ P_e^{(n)} = \Pr(V_{1:n} \neq \hat{V}_{1:n}) \]
\[ \leq \Pr(V_{1:n} \notin A^{(n)}_\epsilon) + \Pr(g(Y^n) \neq V^n | V^n \in A^{(n)}_\epsilon) \]
\[ \leq \epsilon + \epsilon = 2\epsilon \] (16.44)

\[ \leq \epsilon + \epsilon = 2\epsilon \] (16.45)

<\epsilon \text{ since } R < C

≤ \epsilon + \epsilon = 2\epsilon (16.46)

...
Then

\[ P_e^{(n)} = \Pr(V_{1:n} \neq \hat{V}_{1:n}) \leq \Pr(V_{1:n} \not\in A_{\epsilon}^{(n)}) + \Pr(g(Y^n) \neq V^n | V^n \in A_{\epsilon}^{(n)}) \]  

\[ \leq \epsilon + \epsilon = 2\epsilon \]  

\[ \leq \epsilon + \epsilon = 2\epsilon \]  

And the first part of the theorem is proved.
Zero Error Codes

2nd Thm Conv. Zero Error, \( R = C \)

Feedback Joint Thm Coding

Hamming Codes

Joint Source/Channel Theorem

... proof continued.

- Then

\[
P_e^{(n)} = \Pr(V_{1:n} \neq \hat{V}_{1:n}) \leq \Pr(V_{1:n} \notin A^{(n)}_{\epsilon}) + \Pr(g(Y^n) \neq V^n | V^n \in A^{(n)}_{\epsilon}) \leq \epsilon + \epsilon = 2\epsilon
\]  

(16.44) (16.45) (16.46)

- And the first part of the theorem is proved.

- To show the converse, show that \( P_e^{(n)} \rightarrow 0 \Rightarrow H(\mathcal{V}) \leq C \) for source channel codes.
Joint Source/Channel Theorem

... proof continued.
Joint Source/Channel Theorem

... proof continued.

Define:

\[ X^n(V^n) : V^n \to X^n \]  //encoder \hspace{1cm} (16.47)

\[ g_n(Y^n) : Y^n \to V^n \]  //decoder \hspace{1cm} (16.48)
proof continued.

Define:

\[ X^n(V^n) : V^n \to \mathcal{X}^n \]  //encoder \hspace{1cm} (16.47)

\[ g_n(Y^n) : Y^n \to V^n \]  //decoder \hspace{1cm} (16.48)

Now recall, original Fano says \( H(X|Y) \leq 1 + P_e \log |\mathcal{X}|. \)
Joint Source/Channel Theorem

... proof continued.

Define:

\[ X^n(V^n) : V^n \rightarrow X^n \quad \text{//encoder} \]  \hspace{1cm} (16.47)

\[ g_n(Y^n) : Y^n \rightarrow V^n \quad \text{//decoder} \]  \hspace{1cm} (16.48)

Now recall, original Fano says

\[ H(X|Y) \leq 1 + P_e \log |X| \]  \hspace{1cm} (16.49)

Here we have

\[ H(V^n|\hat{V}^n) \leq 1 + P_e^{(n)} \log |V^n| = 1 + nP_e \log |X| \]  \hspace{1cm} (16.49)

...
Joint Source/Channel Theorem

\[ H(V) \leq H(V_1, V_2, \ldots, V_n) + 1_n \]

\[ \leq 1_n + 1 \frac{1}{n} I(V_1:V_n; \hat{V}_1:n) \]

\[ \leq 1_n + 1 \frac{1}{n} \left( 1 + P(e_n) \log |V| \right) + 1 \frac{1}{n} I(X_1:n; Y_1:n) \]

\[ \leq 2n + P(e_n) \log |V| + C \]

Letting \( n \to \infty \), \( 1/n \) and \( P(e_n) \to 0 \) which leaves us with \( H(V) < C \).
Joint Source/Channel Theorem

...proof continued.

- We get the following derivation

\[ H(V) \leq H(V_1, V_2, \ldots, V_{n+1}) = H(V_1: n) + 1 \]

\[ \leq \frac{1}{n+1} \left( 1 + P(e^n) \log |V| \right) + 1 I(V_1:n; \hat{V}_1:n) \quad \text{(16.51)} \]

\[ \leq \frac{1}{n+1} \left( 1 + P(e^n) \log |V| \right) + 1 I(X_1:n; Y_1:n) \quad \text{by Fano} \]

\[ \leq 2n + P(e^n) \log |V| + C \quad \text{(memoryless)} \quad \text{(16.54)} \]
Joint Source/Channel Theorem

... proof continued.

- We get the following derivation

\[ H(\mathcal{V}) \]
Joint Source/Channel Theorem

... proof continued.

We get the following derivation

\[ H(V) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} \]

(16.54)
Joint Source/Channel Theorem

... proof continued.

- We get the following derivation

\[ H(V) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} = \frac{H(V_{1:n})}{n} + \frac{1}{n} \quad (16.50) \]

\[ \leq \frac{1}{n} + \frac{1}{n} = \frac{2}{n} \quad (16.54) \]
Joint Source/Channel Theorem

... proof continued.

We get the following derivation

\[ H(V) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} = \frac{H(V_{1:n})}{n} + \frac{1}{n} \]  

(16.50)

\[ = \frac{1}{n} + \frac{1}{n} H(V_{1:n}|\hat{V}_{1:n}) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \]  

(16.51)

(16.54)
... proof continued.

We get the following derivation

\[
H(\mathcal{V}) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} = \frac{H(V_{1:n})}{n} + \frac{1}{n} \tag{16.50}
\]

\[
= \frac{1}{n} + \frac{1}{n} H(V_{1:n}|\hat{V}_{1:n}) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \tag{16.51}
\]

\[
\leq \frac{1}{n} + \frac{1}{n} (1 + P_e^{(n)} n \log |\mathcal{V}|) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \tag{16.52}
\]

\[
\leq \frac{1}{n} + \frac{1}{n} (1 + P_e^{(n)} n \log |\mathcal{V}|) + \frac{1}{n} I(X_{1:n}; Y_{1:n}) \tag{16.54}
\]
\[ H(\mathcal{V}) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} = \frac{H(V_{1:n})}{n} + \frac{1}{n} \]  

(16.50)

\[ = \frac{1}{n} + \frac{1}{n} H(V_{1:n} | \hat{V}_{1:n}) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \]  

(16.51)

\[ \leq \frac{1}{n} + \frac{1}{n} \left( 1 + P_e^{(n)} n \log |\mathcal{V}| \right) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \quad \text{//by Fano} \]  

(16.52)

\[ \leq \frac{1}{n} + \frac{1}{n} \left( 1 + P_e^{(n)} n \log |\mathcal{V}| \right) + \frac{1}{n} I(X_{1:n}; Y_{1:n}) \quad \text{//V → X → Y → \hat{V}} \quad \text{and DPP} \]  

(16.53)

(16.54)
Joint Source/Channel Theorem

... proof continued.

We get the following derivation

\[ H(\mathcal{V}) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} = \frac{H(V_{1:n})}{n} + \frac{1}{n} \]  

(16.50)

\[ = \frac{1}{n} + \frac{1}{n} H(V_{1:n}|\hat{V}_{1:n}) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \]  

(16.51)

\[ \leq \frac{1}{n} + \frac{1}{n} \left(1 + P_e(n) n \log |\mathcal{V}|\right) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \quad \text{//by Fano} \]  

(16.52)

\[ \leq \frac{1}{n} + \frac{1}{n} \left(1 + P_e(n) n \log |\mathcal{V}|\right) + \frac{1}{n} I(X_{1:n}; Y_{1:n}) \quad \text{//} V \to X \to Y \to \hat{V} \text{ and DPP} \]  

(16.53)

\[ \leq \frac{2}{n} + P_e(n) \log |\mathcal{V}| + C \quad \text{//memoryless} \]  

(16.54)
Joint Source/Channel Theorem

... proof continued.

We get the following derivation

\[ H(V) \leq \frac{H(V_1, V_2, \ldots, V_n)}{n} + \frac{1}{n} = \frac{H(V_{1:n})}{n} + \frac{1}{n} \quad (16.50) \]

\[ = \frac{1}{n} + \frac{1}{n} H(V_{1:n}|\hat{V}_{1:n}) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \quad (16.51) \]

\[ \leq \frac{1}{n} + \frac{1}{n} (1 + P_e^{(n)} n \log |V|) + \frac{1}{n} I(V_{1:n}; \hat{V}_{1:n}) \quad //by Fano \quad (16.52) \]

\[ \leq \frac{1}{n} + \frac{1}{n} (1 + P_e^{(n)} n \log |V|) + \frac{1}{n} I(X_{1:n}; Y_{1:n}) \quad //V \rightarrow X \rightarrow Y \rightarrow \hat{V} \quad (16.53) \]

\[ \leq \frac{2}{n} + P_e^{(n)} \log |V| + C \quad //memoryless \quad (16.54) \]

Letting \( n \rightarrow \infty, \frac{1}{n} \) and \( P_e \rightarrow 0 \) which leaves us with \( H(V) < C \).
Shannon’s theorem says that there exists a sequence of codes such that if $R < C$ the error goes to zero.
Shannon’s theorem says that there exists a sequence of codes such that if $R < C$ the error goes to zero.

It doesn’t provide such a code, nor does it offer much insight on how to find one.
Shannon’s theorem says that there exists a sequence of codes such that if $R < C$ the error goes to zero.

It doesn’t provide such a code, nor does it offer much insight on how to find one.

Typical set coding is not practical.
Shannon’s theorem says that there exists a sequence of codes such that if $R < C$ the error goes to zero.

It doesn’t provide such a code, nor does it offer much insight on how to find one.

Typical set coding is not practical. Why?
Shannon’s theorem says that there exists a sequence of codes such that if $R < C$ the error goes to zero.

It doesn’t provide such a code, nor does it offer much insight on how to find one.

Typical set coding is not practical. Why? Exponentially large sized table sizes.
Shannon’s theorem says that there exists a sequence of codes such that if $R < C$ the error goes to zero.

It doesn’t provide such a code, nor does it offer much insight on how to find one.

Typical set coding is not practical. Why? Exponentially large sized table sizes.

In all cases, we add enough redundancy to a message so that the original message can be decoded unambiguously.
It is possible to communicate more reliably by changing physical properties to decrease the noise (e.g., decrease $p$ in a BSC).
Physical Solution to Improve Coding

- It is possible to communicate more reliably by changing physical properties to decrease the noise (e.g., decrease $p$ in a BSC).
- Use more reliable and expensive circuitry
Physical Solution to Improve Coding

- It is possible to communicate more reliably by changing physical properties to decrease the noise (e.g., decrease $p$ in a BSC).
- Use more reliable and expensive circuitry
- improve environment (e.g., control thermal conditions, remove dust particles or even air molecules)
Physical Solution to Improve Coding

- It is possible to communicate more reliably by changing physical properties to decrease the noise (e.g., decrease $p$ in a BSC).
- Use more reliable and expensive circuitry
- Improve environment (e.g., control thermal conditions, remove dust particles or even air molecules)
- In compression, use more physical area/volume for each bit.
Physical Solution to Improve Coding

- It is possible to communicate more reliably by changing physical properties to decrease the noise (e.g., decrease $p$ in a BSC).
- Use more reliable and expensive circuitry
- Improve environment (e.g., control thermal conditions, remove dust particles or even air molecules)
- In compression, use more physical area/volume for each bit.
- In communication, use higher power transmitter, use more energy thereby making noise less of a problem.
Physical Solution to Improve Coding

- It is possible to communicate more reliably by changing physical properties to decrease the noise (e.g., decrease $p$ in a BSC).
- Use more reliable and expensive circuitry
- Improve environment (e.g., control thermal conditions, remove dust particles or even air molecules)
- In compression, use more physical area/volume for each bit.
- In communication, use higher power transmitter, use more energy thereby making noise less of a problem.

These are not IT solutions which is what we want.
Rather than send message $x_1 x_2 \ldots x_n$ we repeat each symbol $K$ times redundantly.
Rather than send message $x_1 x_2 \ldots x_n$ we repeat each symbol $K$ times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.
Rather than send message $x_1x_2\ldots x_n$ we repeat each symbol $K$ times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.

Message becomes $x_1x_1\ldots x_1 x_2x_2\ldots x_2\ldots$ with $k \times$.
Rather than send message \(x_1 x_2 \ldots x_n\) we repeat each symbol \(K\) times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.

Message becomes \(x_1 x_1 \ldots x_1\) \(\underbrace{x_2 x_2 \ldots x_2}_{k \times} \ldots\)

For many channels (e.g., BSC\((p < 1/2)\)), error goes to zero as \(k \to \infty\).
Rather than send message $x_1 x_2 \ldots x_n$ we repeat each symbol $K$ times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.

Message becomes $\underbrace{x_1 x_1 \ldots x_1}_{k \times} \underbrace{x_2 x_2 \ldots x_2}_{k \times} \ldots$

For many channels (e.g., BSC($p < 1/2$)), error goes to zero as $k \to \infty$.

Easy decoding: when $k$ is odd, take a majority vote (which is optimal for a BSC)
Rather than send message $x_1 x_2 \ldots x_n$ we repeat each symbol $K$ times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.

Message becomes $x_1 x_1 \ldots x_1 \underbrace{x_2 x_2 \ldots x_2 \ldots}_{k \times} \underbrace{k \times k \times}$

For many channels (e.g., BSC($p < 1/2$)), error goes to zero as $k \rightarrow \infty$.

Easy decoding: when $k$ is odd, take a majority vote (which is optimal for a BSC)

On the other hand, $R \propto 1/k \rightarrow 0$ as $k \rightarrow \infty$
Rather than send message $x_1 x_2 \ldots x_n$ we repeat each symbol $K$ times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.

Message becomes $\underbrace{x_1 x_1 \ldots x_1}_{k \times} \underbrace{x_2 x_2 \ldots x_2}_{k \times} \ldots$

For many channels (e.g., BSC($p < 1/2$)), error goes to zero as $k \to \infty$.

Easy decoding: when $k$ is odd, take a majority vote (which is optimal for a BSC)

On the other hand, $R \propto 1/k \to 0$ as $k \to \infty$

This is really a pre-1948 way of thinking code.
Rather than send message $x_1x_2 \ldots x_n$ we repeat each symbol $K$ times redundantly.

Recall our example of repeating each word in a noisy analog radio connection.

Message becomes $x_1x_1 \ldots x_1 x_2x_2 \ldots x_2 \ldots$

For many channels (e.g., BSC($p < 1/2$)), error goes to zero as $k \to \infty$.

Easy decoding: when $k$ is odd, take a majority vote (which is optimal for a BSC)

On the other hand, $R \propto 1/k \to 0$ as $k \to \infty$

This is really a pre-1948 way of thinking code.

Thus, this is not a good code.
Repetition Code Example

- (From D. Mackay) Consider sending message $s = 0 0 1 0 1 1 0$
Repetition Code Example

(From D. Mackay) Consider sending message \( s = 0 0 1 0 1 1 0 \)

One scenario . . .

\[
\begin{array}{cccccccc}
\text{s} & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\text{t} & 0 0 0 & 0 0 0 & 1 1 1 & 0 0 0 & 1 1 1 & 1 1 1 & 0 0 0 \\
\text{n} & 0 0 0 & 0 0 1 & 0 0 0 & 0 0 0 & 1 0 1 & 0 0 0 & 0 0 0 \\
\text{r} & 0 0 0 & 0 0 1 & 1 1 1 & 0 0 0 & 0 1 0 & 1 1 1 & 0 0 0 \\
\end{array}
\]

Thus, can only correct one bit error not two.
Repetition Code Example

(From D. Mackay) Consider sending message \( s = 0010110 \)

One scenario . . .

\[
\begin{array}{cccccccc}
\text{s} & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\text{t} & 000 & 000 & 111 & 000 & 111 & 111 & 000 \\
\text{n} & 000 & 001 & 000 & 000 & 101 & 000 & 000 \\
\text{r} & 000 & 001 & 111 & 000 & 010 & 111 & 000 \\
\end{array}
\]

. . . with decoded message

\[
\begin{array}{cccccccc}
\text{s} & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\text{t} & 000 & 000 & 111 & 000 & 111 & 111 & 000 \\
\text{n} & 000 & 001 & 000 & 000 & 101 & 000 & 000 \\
\text{r} & 000 & 001 & 111 & 000 & 010 & 111 & 000 \\
\hat{s} & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
\end{array}
\]

corrected errors \( \rightarrow \) *
detected but uncorrected errors \( \rightarrow \) *

Thus, can only correct one bit error not two.
Repetition Code Example

(From D. Mackay) Consider sending message \( s = 0 0 1 0 1 1 0 \)

One scenario . . .

\[
\begin{array}{cccccccccc}
\text{s} & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\text{t} & 0 0 0 & 0 0 0 & 1 1 1 & 0 0 0 & 1 1 1 & 1 1 1 & 0 0 0 \\
\text{n} & 0 0 0 & 0 0 1 & 0 0 0 & 0 0 0 & 1 0 1 & 0 0 0 & 0 0 0 \\
\text{r} & 0 0 0 & 0 0 1 & 1 1 1 & 0 0 0 & 0 1 0 & 1 1 1 & 0 0 0 \\
\end{array}
\]

. . . with decoded message

\[
\begin{array}{cccccccccc}
\text{s} & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\text{t} & 0 0 0 & 0 0 0 & 1 1 1 & 0 0 0 & 1 1 1 & 1 1 1 & 0 0 0 \\
\text{n} & 0 0 0 & 0 0 1 & 0 0 0 & 0 0 0 & 1 0 1 & 0 0 0 & 0 0 0 \\
\text{r} & 0 0 0 & 0 0 1 & 1 1 1 & 0 0 0 & 0 1 0 & 1 1 1 & 0 0 0 \\
\hat{s} & 0 0 1 0 0 1 0 \\
\end{array}
\]

corrected errors \( \rightarrow \)

detected but uncorrected errors \( \rightarrow \)

Thus, can only correct one bit error not two.
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$. 
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n - 1$ bits: $x_{1:n-1}$. 

Thus a necessary condition for valid code word is: 

$$\mod\left(\sum_{i=1}^{n-1} x_i, 2\right) = 0.$$ 

Any instance of an odd number of errors (bit swaps) won't pass this condition, and such an error is hence detected. Although an even number of errors will pass the condition (error goes undetected). 

Parity checks form the basis for many sophisticated coding schemes (e.g., low-density parity check (LDPC) codes, Hamming codes etc.). We study Hamming codes next.
Simple Parity Check Code

- Binary input/output alphabets \( \mathcal{X} = \mathcal{Y} = \{0, 1\} \).
- Block sizes of \( n - 1 \) bits: \( x_{1:n-1} \).
- \( n^{th} \) bit is an indicator of an odd number of 1 bits in \( x_{1:n-1} \).
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n - 1$ bits: $x_{1:n-1}$.
- $n^{th}$ bit is an indicator of an odd number of 1 bits in $x_{1:n-1}$.
- I.e., $x_n \leftarrow \text{mod} \left( \sum_{i=1}^{n-1} x_i, 2 \right)$. 

Thus a necessary condition for valid code word is:

\[
\text{mod} \left( \sum_{i=1}^{n-1} x_i, 2 \right) = 0. 
\]

Any instance of an odd number of errors (bit swaps) won't pass this condition, and such an error is hence detected. Although an even number of errors will pass the condition (error goes undetected). Parity checks can not correct all errors, and moreover only detects some of the kinds of errors (odd number of swaps).

On the other hand, parity checks form the basis for many sophisticated coding schemes (e.g., low-density parity check (LDPC) codes, Hamming codes etc.). We study Hamming codes next.
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n - 1$ bits: $x_{1:n-1}$.
- $n^{th}$ bit is an indicator of an odd number of 1 bits in $x_{1:n-1}$.
- I.e., $x_{n} \leftarrow \text{mod} \left( \sum_{i=1}^{n-1} x_{i}, 2 \right)$.
- Thus a necessary condition for valid code word is: $\text{mod} \left( \sum_{i=1}^{n} x_{i}, 2 \right) = 0$. 

Any instance of an odd number of errors (bit swaps) won’t pass this condition, and such an error is hence detected. Although an even number of errors will pass the condition (error goes undetected). 

Parity checks form the basis for many sophisticated coding schemes (e.g., low-density parity check (LDPC) codes, Hamming codes etc.). We study Hamming codes next.
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n - 1$ bits: $x_{1:n-1}$.
- $n^{th}$ bit is an indicator of an odd number of 1 bits in $x_{1:n-1}$.
- I.e., $x_n \leftarrow \text{mod} \left( \sum_{i=1}^{n-1} x_i, 2 \right)$.
- Thus a necessary condition for valid code word is: $\text{mod} \left( \sum_{i=1}^{n} x_i, 2 \right) = 0$.
- Any instance of an odd number of errors (bit swaps) won’t pass this condition, and such an error is hence detected.
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n - 1$ bits: $x_{1:n-1}$.
- $n^{th}$ bit is an indicator of an odd number of 1 bits in $x_{1:n-1}$.
- I.e., $x_n \leftarrow \text{mod}\left(\sum_{i=1}^{n-1} x_i, 2\right)$.
- Thus a necessary condition for valid code word is:
  $$\text{mod}\left(\sum_{i=1}^{n} x_i, 2\right) = 0.$$  
- Any instance of an odd number of errors (bit swaps) won’t pass this condition, and such an error is hence detected.
- Although an even number of errors will pass the condition (error goes undetected).
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n - 1$ bits: $x_{1:n-1}$.
- $n^{th}$ bit is an indicator of an odd number of 1 bits in $x_{1:n-1}$.
- I.e., $x_n \leftarrow \text{mod} \left( \sum_{i=1}^{n-1} x_i, 2 \right)$.
- Thus a necessary condition for valid code word is:
  $\text{mod} \left( \sum_{i=1}^{n} x_i, 2 \right) = 0$.
- Any instance of an odd number of errors (bit swaps) won’t pass this condition, and such an error is hence detected.
- although an even number of errors will pass the condition (error goes undetected).
- can not correct all errors, and moreover only detects some of the kinds of errors (odd number of swaps).
Simple Parity Check Code

- Binary input/output alphabets $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.
- Block sizes of $n-1$ bits: $x_{1:n-1}$.
- $n^{th}$ bit is an indicator of an odd number of 1 bits in $x_{1:n-1}$.
- I.e., $x_n \leftarrow \text{mod} \left( \sum_{i=1}^{n-1} x_i, 2 \right)$.
- Thus a necessary condition for valid code word is:
  $\text{mod} \left( \sum_{i=1}^{n} x_i, 2 \right) = 0$.
- Any instance of an odd number of errors (bit swaps) won’t pass this condition, and such an error is hence detected.
- although an even number of errors will pass the condition (error goes undetected).
- can not correct all errors, and moreover only detects some of the kinds of errors (odd number of swaps).
- On the other hand, parity checks form the basis for many sophisticated coding schemes (e.g., low-density parity check (LDPC) codes, Hamming codes etc.).
### Simple Parity Check Code

- Binary input/output alphabets \( \mathcal{X} = \mathcal{Y} = \{0, 1\} \).
- Block sizes of \( n - 1 \) bits: \( x_{1:n-1} \).
- \( n^{th} \) bit is an indicator of an odd number of 1 bits in \( x_{1:n-1} \).
- I.e., \( x_n \leftarrow \text{mod} \left( \sum_{i=1}^{n-1} x_i, 2 \right) \).
- Thus a necessary condition for valid code word is:
  \[
  \text{mod} \left( \sum_{i=1}^{n} x_i, 2 \right) = 0.
  \]
- Any instance of an odd number of errors (bit swaps) won’t pass this condition, and such an error is hence detected.
- although an even number of errors will pass the condition (error goes undetected).
- can not correct all errors, and moreover only detects some of the kinds of errors (odd number of swaps).
- On the other hand, parity checks form the basis for many sophisticated coding schemes (e.g., low-density parity check (LDPC) codes, Hamming codes etc.).
- We study Hamming codes next.
Best illustrated by an example.
Best illustrated by an example.

Let $\mathcal{X} = \mathcal{Y} = \{0, 1\}$. 

(7, 4, 3) Hamming Codes
Best illustrated by an example.

Let $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.

Fix the desired rate at $R = 4/7$ bit per channel use.
Best illustrated by an example.

Let \( X = Y = \{0, 1\} \).

Fix the desired rate at \( R = 4/7 \) bit per channel use.

Thus, in order to send 4 data bits, we need to use the channel 7 times.
Best illustrated by an example.

Let $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.

Fix the desired rate at $R = 4/7$ bit per channel use.

Thus, in order to send 4 data bits, we need to use the channel 7 times.

Let the four data bits be denoted $x_0, x_1, x_2, x_3 \in \{0, 1\}$. 

$(7, 4, 3)$ Hamming Codes
Best illustrated by an example.

Let $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.

Fix the desired rate at $R = 4/7$ bit per channel use.

Thus, in order to send 4 data bits, we need to use the channel 7 times.

Let the four data bits be denoted $x_0, x_1, x_2, x_3 \in \{0, 1\}$.

When we send these 4 bits, we are also going to send 3 additional parity or redundancy bits, named $x_4, x_5, x_6$.  

(7, 4, 3) Hamming Codes
Best illustrated by an example.

Let \( X = Y = \{0, 1\} \).

Fix the desired rate at \( R = 4/7 \) bit per channel use.

Thus, in order to send 4 data bits, we need to use the channel 7 times.

Let the four data bits be denoted \( x_0, x_1, x_2, x_3 \in \{0, 1\} \).

When we send these 4 bits, we are also going to send 3 additional parity or redundancy bits, named \( x_4, x_5, x_6 \).

Note: all arithmetic in the following will be mod 2. I.e. \( 1 + 1 = 0, 1 + 0 = 1, 1 = 0 - 1 = -1 \), etc.
Parity bits determined by the following equations:

\[
\begin{align*}
  x_4 &\equiv x_1 + x_2 + x_3 \mod 2 \\
  x_5 &\equiv x_0 + x_2 + x_3 \mod 2 \\
  x_6 &\equiv x_0 + x_1 + x_3 \mod 2
\end{align*}
\]
(7, 4, 3) Hamming Codes

- Parity bits determined by the following equations:

\[
x_4 \equiv x_1 + x_2 + x_3 \pmod{2} \quad (16.55)
\]

\[
x_5 \equiv x_0 + x_2 + x_3 \pmod{2} \quad (16.56)
\]

\[
x_6 \equiv x_0 + x_1 + x_3 \pmod{2} \quad (16.57)
\]

- I.e., if \((x_0, x_1, x_2, x_3) = (0110)\) then \((x_4, x_5, x_6) = (011)\) and complete 7-bit codeword sent over channel would be \((0110011)\).
(7, 4, 3) Hamming Codes

- Parity bits determined by the following equations:

\[ x_4 \equiv x_1 + x_2 + x_3 \pmod{2} \quad (16.55) \]
\[ x_5 \equiv x_0 + x_2 + x_3 \pmod{2} \quad (16.56) \]
\[ x_6 \equiv x_0 + x_1 + x_3 \pmod{2} \quad (16.57) \]

- I.e., if \((x_0, x_1, x_2, x_3) = (0110)\) then \((x_4, x_5, x_6) = (011)\) and complete 7-bit codeword sent over channel would be \((0110011)\).

- We can also describe this using linear equalities as follows (all mod 2).

\[
\begin{align*}
    x_1 + x_2 + x_3 + x_4 &= 0 \\
    x_0 + x_2 + x_3 + x_5 &= 0 \\
    x_0 + x_1 + x_3 + x_6 &= 0
\end{align*}
\]

\[ (16.58) \]
Hamming Codes

- Or alternatively, as $Hx = 0$ where $x^\top = (x_1, x_2, \ldots, x_7)$ and

$$H = \begin{pmatrix}
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{pmatrix}$$  \hspace{1cm} (16.59)
Hamming Codes

- Or alternatively, as $Hx = 0$ where $x^T = (x_1, x_2, \ldots, x_7)$ and
  \[
  H = \begin{pmatrix}
  0 & 1 & 1 & 1 & 1 & 0 & 0 \\
  1 & 0 & 1 & 1 & 0 & 1 & 0 \\
  1 & 1 & 0 & 1 & 0 & 0 & 1
  \end{pmatrix}
  \]  
  (16.59)

- Codewords lie in null-space of $H$
Hamming Codes

- Or alternatively, as $Hx = 0$ where $x^T = (x_1, x_2, \ldots, x_7)$ and

$$
H = \begin{pmatrix}
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1
\end{pmatrix}
$$

(16.59)

- Codewords lie in null-space of $H$
- Notice that $H$ is a column permutation of all seven non-zero length-3 column vectors.
Hamming Codes

- Or alternatively, as \( Hx = 0 \) where \( x^T = (x_1, x_2, \ldots, x_7) \) and

\[
H = \begin{pmatrix}
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1
\end{pmatrix}
\]  \hspace{1cm} (16.59)

- Codewords lie in null-space of \( H \)

- Notice that \( H \) is a column permutation of all seven non-zero length-3 column vectors.

- Thus the code words are defined by the null-space of \( H \). I.e., \( \{ x : Hx = 0 \} \).
Or alternatively, as \( Hx = 0 \) where \( x^\top = (x_1, x_2, \ldots, x_7) \) and

\[
H = \begin{pmatrix}
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1
\end{pmatrix}
\]  \hspace{1cm} (16.59)

- Codewords lie in null-space of \( H \)
- Notice that \( H \) is a column permutation of all seven non-zero length-3 column vectors.
- Thus the code words are defined by the null-space of \( H \). I.e., \( \{x : Hx = 0\} \).
- Since the rank of \( H \) is 3, the null-space is 4, and we expect there to be \( 16 = 2^4 \) binary vectors in this null space.
Hamming Codes

The 16 vectors in the nullspace (i.e., \{x : Hx = 0\}) are as follows:

\[
\begin{align*}
0000000 & \quad 0100101 & \quad 1000011 & \quad 1100110 & \quad (16.60) \\
0001111 & \quad 0101010 & \quad 1001100 & \quad 1101001 & \quad (16.61) \\
0010110 & \quad 0110011 & \quad 1010101 & \quad 1110000 & \quad (16.62) \\
0011001 & \quad 0111100 & \quad 1011010 & \quad 1111111 & \quad (16.63)
\end{align*}
\]
The 16 vectors in the nullspace (i.e., \( \{ x : Hx = 0 \} \)) are as follows:

\[
\begin{align*}
0000000 & \quad 0100101 & \quad 1000011 & \quad 1100110 \quad (16.60) \\
0001111 & \quad 0101010 & \quad 1001100 & \quad 1101001 \quad (16.61) \\
0010110 & \quad 0110011 & \quad 1010101 & \quad 1110000 \quad (16.62) \\
0011001 & \quad 0111100 & \quad 1011010 & \quad 1111111 \quad (16.63)
\end{align*}
\]

Note the first (highest order) four bits of these vectors range from 0 to 15 in binary (i.e., all bit strings of length 4).
Hamming Codes

- The 16 vectors in the nullspace (i.e., \( \{ x : Hx = 0 \} \)) are as follows:

\[
\begin{align*}
0000000 & \quad 0100101 & \quad 1000011 & \quad 1100110 & \quad (16.60) \\
0001111 & \quad 0101010 & \quad 1001100 & \quad 1101001 & \quad (16.61) \\
0010110 & \quad 0110011 & \quad 1010101 & \quad 1110000 & \quad (16.62) \\
0011001 & \quad 0111100 & \quad 1011010 & \quad 1111111 & \quad (16.63)
\end{align*}
\]

- Note the first (highest order) four bits of these vectors range from 0 to 15 in binary (i.e., all bit strings of length 4).

- These first four bits are the data bits, and the 2nd three bits are the redundancy bits.
The 16 vectors in the nullspace (i.e., \( \{ x : H x = 0 \} \)) are as follows:

\[
\begin{align*}
0000000 & \quad 0100101 & \quad 1000011 & \quad 1100110 \\
0001111 & \quad 0101010 & \quad 1001100 & \quad 1101001 \\
0010110 & \quad 0110011 & \quad 1010101 & \quad 1110000 \\
0011001 & \quad 0111100 & \quad 1011010 & \quad 1111111
\end{align*}
\] (16.60)

Note the first (highest order) four bits of these vectors range from 0 to 15 in binary (i.e., all bit strings of length 4).

These first four bits are the data bits, and the 2nd three bits are the redundancy bits.

The vectors constitute the codewords, any codeword must be one of the above.
Thus, any valid codeword is in \( C = \{ x : H x = 0 \} \).
Hamming Codes: weight

- Thus, any valid codeword is in $C = \{ x : Hx = 0 \}$.
- Thus, if $v_1, v_2 \in C$ then $H(v_1 + v_2) = Hv_1 + Hv_2 = 0$ and thus $v_1 + v_2 \in C$. 

- Minimum number of ones in any non-zero codeword is 3. This is called the weight of a code.
- Why weight 3? Suppose there was a weight-two code word with non-zeros at position $i$ and $j$. Thus, sum of columns $i$ and $j$ would be zero. But since columns of $H$ are all different, sum of any two columns is non-zero. Hence, can't have any weight-2 codeword.

Q: why can't we have a weight 1 code word? Can have weight 3 codeword, since sum of two columns will equal another column, and sum of two equal binary vectors is zero (mod 2).
Thus, any valid codeword is in $C = \{ x : Hx = 0 \}$.

Thus, if $v_1, v_2 \in C$ then $H(v_1 + v_2) = Hv_1 + Hv_2 = 0$ and thus $v_1 + v_2 \in C$.

Also, $v_1 - v_2 \in C$ due to linearity (codewords closed under addition and subtraction).
Hamming Codes: weight

- Thus, any valid codeword is in $C = \{x : Hx = 0\}$.
- Thus, if $v_1, v_2 \in C$ then $H(v_1 + v_2) = Hv_1 + Hv_2 = 0$ and thus $v_1 + v_2 \in C$.
- Also, $v_1 - v_2 \in C$ due to linearity (codewords closed under addition and subtraction).
- Minimum number of ones in any non-zero codeword is 3.
Hamming Codes : weight

- Thus, any valid codeword is in $C = \{x : Hx = 0\}$.
- Thus, if $v_1, v_2 \in C$ then $H(v_1 + v_2) = Hv_1 + Hv_2 = 0$ and thus $v_1 + v_2 \in C$.
- Also, $v_1 - v_2 \in C$ due to linearity (codewords closed under addition and subtraction).
- Minimum number of ones in any non-zero codeword is 3. This is called the weight of a code.
Hamming Codes: weight

- Thus, any valid codeword is in \( C = \{x : Hx = 0\} \).
- Thus, if \( v_1, v_2 \in C \) then \( H(v_1 + v_2) = Hv_1 + Hv_2 = 0 \) and thus \( v_1 + v_2 \in C \).
- Also, \( v_1 - v_2 \in C \) due to linearity (codewords closed under addition and subtraction).
- Minimum number of ones in any non-zero codeword is 3. This is called the weight of a code.
- Why weight 3? Suppose there was a weight-two code word with non-zeros at position \( i \) and \( j \). Thus, sum of columns \( i \) and \( j \) would be zero. But since columns of \( H \) are all different, sum of any two columns is non-zero. Hence, can’t have any weight-2 codeword.
Hamming Codes: weight

- Thus, any valid codeword is in \( C = \{ x : Hx = 0 \} \).
- Thus, if \( v_1, v_2 \in C \) then \( H(v_1 + v_2) = Hv_1 + Hv_2 = 0 \) and thus \( v_1 + v_2 \in C \).
- Also, \( v_1 - v_2 \in C \) due to linearity (codewords closed under addition and subtraction).
- Minimum number of ones in any non-zero codeword is 3. This is called the weight of a code.
- Why weight 3? Suppose there was a weight-two code word with non-zeros at position \( i \) and \( j \). Thus, sum of columns \( i \) and \( j \) would be zero. But since columns of \( H \) are all different, sum of any two columns is non-zero. Hence, can’t have any weight-2 codeword.
- Q: why can’t we have a weight 1 code word?
Hamming Codes: weight

- Thus, any valid codeword is in $C = \{x : Hx = 0\}$.
- Thus, if $v_1, v_2 \in C$ then $H(v_1 + v_2) = Hv_1 + Hv_2 = 0$ and thus $v_1 + v_2 \in C$.
- Also, $v_1 - v_2 \in C$ due to linearity (codewords closed under addition and subtraction).
- Minimum number of ones in any non-zero codeword is 3. This is called the weight of a code.
- Why weight 3? Suppose there was a weight-two code word with non-zeros at position $i$ and $j$. Thus, sum of columns $i$ and $j$ would be zero. But since columns of $H$ are all different, sum of any two columns is non-zero. Hence, can’t have any weight-2 codeword.
- Q: why can’t we have a weight 1 code word?
- Can have weight 3 codeword, since sum of two columns will equal another column, and sum of two equal binary vectors is zero (mod 2).
Hamming Codes: Distance

- Thus, any codeword is in $C = \{x : Hx = 0\}$. 

**Prof. Jeff Bilmes**

EE514a/Fall 2019/Info. Theory I – Lecture 16 - Nov 25th, 2019

L16 F50/51 (pg.202/218)
Hamming Codes: Distance

- Thus, any codeword is in $C = \{ x : Hx = 0 \}$.

- **Minimum distance** of a code is also 3, which is minimum number of differences between any two codewords.
Hamming Codes: Distance

- Thus, any codeword is in \( C = \{ x : Hx = 0 \} \).
- **minimum distance** of a code is also 3, which is minimum number of differences between any two codewords.
- Another way of saying this: if \( v_1, v_2 \in C \) then \( d_H(v_1, v_2) \geq 3 \) where \( d_H(x, y) = \sum_i 1_{\{x(i) \neq y(i)\}} \) is the Hamming distance.
Hamming Codes: Distance

- Thus, any codeword is in $C = \{x : Hx = 0\}$.
- minimum distance of a code is also 3, which is minimum number of differences between any two codewords.
- Another way of saying this: if $v_1, v_2 \in C$ then $d_H(v_1, v_2) \geq 3$ where $d_H(x, y) = \sum_i 1\{x(i) \neq y(i)\}$ is the Hamming distance.
- Why? Suppose $v_1, v_2 \in C$ differ in only two places. Then $H(v_1 - v_2)$ will be a difference or sum of two columns of $H$ (mod 2).
Hamming Codes: Distance

- Thus, any codeword is in $C = \{x : Hx = 0\}$.

- **minimum distance** of a code is also 3, which is minimum number of differences between any two codewords.

- Another way of saying this: if $v_1, v_2 \in C$ then $d_H(v_1, v_2) \geq 3$ where $d_H(x, y) = \sum_i 1\{x(i) \neq y(i)\}$ is the Hamming distance.

- Why? Suppose $v_1, v_2 \in C$ differ in only two places. Then $H(v_1 - v_2)$ will be a difference or sum of two columns of $H$ (mod 2). But given $v_1, v_2 \in C \Rightarrow (v_1 - v_2) \in C$. Can't have difference or sum, $(1+1 = 1-1 \mod 2)$ of any two columns equaling zero $H(v_1 - v_2) \neq 0$, contradiction. Hence, $v_1 - v_2$ can't differ in only two places.
Hamming Codes: Distance

- Thus, any codeword is in $C = \{x : Hx = 0\}$.
- minimum distance of a code is also 3, which is minimum number of differences between any two codewords.
- Another way of saying this: if $v_1, v_2 \in C$ then $d_H(v_1, v_2) \geq 3$ where $d_H(x, y) = \sum_i 1_{\{x(i) \neq y(i)\}}$ is the Hamming distance.
- Why? Suppose $v_1, v_2 \in C$ differ in only two places. Then $H(v_1 - v_2)$ will be a difference or sum of two columns of $H$ (mod 2). But given $v_1, v_2 \in C \Rightarrow (v_1 - v_2) \in C$. Can't have difference or sum, $(1+1 = 1-1 \mod 2)$ of any two columns equaling zero $H(v_1 - v_2) \neq 0$, contradiction. Hence, $v_1 - v_2$ can't differ in only two places.
- In general, codes with large minimum distance is good because then it is possible to correct errors. I.e., if $\hat{v}$ is received codeword, then we can find $i \in \arg\min_i d_H(\hat{v}, v_i)$ as the decoding procedure.
Hamming Codes: BSC

- Now a BSC\((p)\) (crossover probability \(p\)) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.
Now a BSC($p$) (crossover probability $p$) will change some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if $x = (x_0, x_2, \ldots, x_6)$ is transmitted, what is received is

$$y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6) \quad (16.64)$$

where $z = (z_0, z_2, \ldots, z_6)$ is the noise vector.
Now a BSC(p) (crossover probability p) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if \( x = (x_0, x_2, \ldots, x_6) \) is transmitted, what is received is

\[
y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)
\]  \hspace{1cm} (16.64)

where \( z = (z_0, z_2, \ldots, z_6) \) is the noise vector.

Receiver knows \( y \) but wants to know \( x \).
Hamming Codes: BSC

- Now a BSC($p$) (crossover probability $p$) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.
- So if $x = (x_0, x_2, \ldots, x_6)$ is transmitted, what is received is

$$y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)$$  \hspace{1cm} (16.64)

where $z = (z_0, z_2, \ldots, z_6)$ is the noise vector.
- Receiver knows $y$ but wants to know $x$. We then compute

$$s = Hy = H(x + z) = Hx + Hz =Hz$$  \hspace{1cm} (16.65)
Now a BSC\((p)\) (crossover probability \(p\)) will change some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if \(x = (x_0, x_2, \ldots, x_6)\) is transmitted, what is received is

\[
y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)
\]

(16.64)

where \(z = (z_0, z_2, \ldots, z_6)\) is the noise vector.

Receiver knows \(y\) but wants to know \(x\). We then compute

\[
s
\]

(16.65)
Hamming Codes: BSC

- Now a BSC($p$) (crossover probability $p$) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.
- So if $x = (x_0, x_2, \ldots, x_6)$ is transmitted, what is received is

$$y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6) \quad (16.64)$$

where $z = (z_0, z_2, \ldots, z_6)$ is the noise vector.
- Receiver knows $y$ but wants to know $x$. We then compute

$$s = Hy \quad (16.65)$$
Hamming Codes: BSC

Now a BSC\((p)\) (crossover probability \(p\)) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if \(x = (x_0, x_2, \ldots, x_6)\) is transmitted, what is received is

\[
y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)
\]

(16.64)

where \(z = (z_0, z_2, \ldots, z_6)\) is the noise vector.

Receiver knows \(y\) but wants to know \(x\). We then compute

\[
s = Hy = H(x + z)
\]

(16.65)
Hamming Codes: BSC

Now a BSC \(p\) (crossover probability \(p\)) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if \(x = (x_0, x_2, \ldots, x_6)\) is transmitted, what is received is

\[
y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)
\]

(16.64)

where \(z = (z_0, z_2, \ldots, z_6)\) is the noise vector.

Receiver knows \(y\) but wants to know \(x\). We then compute

\[
s = Hy = H(x + z) = Hx + Hz
\]

(16.65)
Now a BSC\((p)\) (crossover probability \(p\)) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice verse.  

So if \(x = (x_0, x_2, \ldots, x_6)\) is transmitted, what is received is

\[ y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6) \tag{16.64} \]

where \(z = (z_0, z_2, \ldots, z_6)\) is the noise vector.

Receiver knows \(y\) but wants to know \(x\). We then compute

\[ s = Hy = H(x + z) = Hx + Hz \]

\(= 0\)
Now a BSC\((p)\) (crossover probability \(p\)) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if \(x = (x_0, x_2, \ldots, x_6)\) is transmitted, what is received is

\[
y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)
\]

where \(z = (z_0, z_2, \ldots, z_6)\) is the noise vector.

Receiver knows \(y\) but wants to know \(x\). We then compute

\[
s = Hy = H(x + z) = Hx + Hz = Hz
\]
Now a BSC\((p)\) (crossover probability \(p\)) will chance some of the bits (noise), meaning a 0 might change to a 1 and vice versa.

So if \(x = (x_0, x_2, \ldots, x_6)\) is transmitted, what is received is

\[
y = x + z = (x_0 + z_0, x_1 + z_1, \ldots, x_6 + z_6)
\]  
(16.64)

where \(z = (z_0, z_2, \ldots, z_6)\) is the noise vector.

Receiver knows \(y\) but wants to know \(x\). We then compute

\[
s = Hy = H(x + z) = \underbrace{Hx}_{=0} + Hz
\]  
(16.65)

\(s\) is called the syndrome of \(y\). If \(s = 0\), then all parity checks are satisfied by \(y\) and is a necessary condition for a correct codeword.