Read chapters 1, and 2 in C&T.
Read chapter 3 in C&T.
Read section 11.1,11.3, method of types and universal source coding.
Read chapter 4.
Read chapter 5.
Read stream code chapter 6 in “Information Theory, Inference, and Learning Algorithms” by David J.C. MacKay (available online http://www.inference.phy.cam.ac.uk/mackay/itila/)
Read chapter 7 in Cover and Thomas, channel capacity
Announcements, Assignments, and Reminders

- Next homework won’t be available sometime this week, due next week.
- Late policy: 10% every 24 hour period that you are late, and no more than 3 days late accepted.
- Lowest grade out of all HW grades is not counted towards final grade (so you can skip one HW with impunity).
- Please do use our discussion board (https://catalyst.uw.edu/gopost/board/karna/25503/) for all questions, so that all will benefit from them being answered.
Class Road Map - IT-I

- L1 (1/3): Overview, Entropy
- L2 (1/5): Props. Entropy, Mutual Information, KL-Divergence
- L3 (1/10): KL-Divergence, Jensen, properties, Data Proc. Inequality
- L5 (1/17): Fano, AEP
- L6 (1/19): snow
- L6 (1/24): AEP, source coding
- L7 (1/26): Method of Types
- L9 (2/2): HMMs, coding
- L10 (2/7): Coding, Kraft,
- L11 (2/9): Huffman, midterm
- L12 (2/14): Midterm
- L13 (2/16): Shannon Games, Arithmetic
- L14 (2/21): Channel Capacity
- L16 (2/28): Shannon’s 2nd theorem.
- L18 (3/6):
- L19 (3/8):

Finals Week: March 12th–16th.
The sequence of guess numbers for the set of letters can be seen as a “code” for the string. I.e., a mapping from letters to integers:

\[ C : \{ 'A', 'B', 'C', \ldots, 'Z', ' ' \} \rightarrow \{1, 2, \ldots, 27\} \]  \hspace{1cm} (1)

Often, the guesses are immediate. So there are many more ones (1s), twos (2s), and so on then there are large integers.

Things that are more predictable have fewer guesses, or have higher probability. Things that require more guesses are less predictable, and have lower probability.

The redundancy in English is its predictability, the more low numbered integers, the more English is redundant.
Bits per guess

- Let $g_t$ be the number of guesses at position $t$ within the source string.
- $\log g_t$ is the number of bits to represent a number as large as $g_t$, or number of bits required to encode number of guesses at stage $t$.
- Then we can estimate the entropy rate of this process as follows:

$$\hat{H}(X) \approx \frac{1}{n} \sum_{t=1}^{n} \log g_t \approx \frac{1}{n} \sum_{t=1}^{n} \log \frac{1}{p(x_t|x_{t-1}, x_{t-2}, \ldots, x_1)} \quad (2)$$

- Suppose that $x_1, x_2, \ldots$ is a stochastic process with an entropy rate of the form $H(X_t|X_{t-1}, \ldots, X_1)$. Then $p(x_t|x_{t-1}, x_{t-2}, \ldots, x_1)$ is the probability of letter $x_t$ at time $t$. Then we should have approximately

$$g_t \approx \frac{1}{p(x_t|x_{t-1}, x_{t-2}, \ldots, x_1)} \quad (3)$$
A compression algorithm could transform the string into a string of numbers. I.e., we’d use 
\( C : \{\text{'A', 'B', 'C', \ldots, 'Z', ' '}\} \rightarrow \{1, 2, \ldots, 27\} \) to transform from source symbols into code symbols.

But the frequency of code symbols 1, 2, 3, etc. is much higher than any of the source (alpha) symbols. I.e., we’ll see a “1” much more frequently than an “e” since many letters, even if they are not an “e” are easily guessable, and sometimes “e” is not guessable.

So, rather than encode 
“there is no reverse on a motorcycle”, we would encode and compress 
“1,1,1,5,1,1,2,1,1,2,1,1,15,1,17,1,1,1,2,1,3,2,1,2,2,7,1,1,1,1,4,1,1,1,1”

Should compress well, many 1s.
Arithmetic Coding

- This is the method used by DjVu (adaptive image compression used for printed material, overtaken by PDF but probably certain PDF formats use this as well).

- Assume we are given a probabilistic model of the source. I.e.,

\[
p(x_{1:n}) = \prod_{i=1}^{n} p(x_i) \text{ would be simple i.i.d.} \tag{4}
\]

or alternatively

\[
p(x_{1:n}) = p(x_1) \prod_{i=2}^{n} p(x_i|x_{i-1}) \text{ would be a 1st order Markov model} \tag{5}
\]

- Higher order Markov models often used as well (as we’ll see).
Arithmetic Coding

- At each symbol, we use the conditional probability to provide the probability of the next symbol.
- Arithmetic coding can easily handle complex adaptive models of the source that produce context-dependent predictive distributions (so not nec. stationary).
- Best understood with an example. Let $\mathcal{X} = \{a, e, i, o, u, !\}$ so $|\mathcal{X}| = 6$.
- Source $X_1, X_2, \ldots$ need not be i.i.d.
- Assume that $p(x_n|x_1, x_2, \ldots, x_{n-1})$ is given to both encoder (sender, compressor) and receiver (decoder, uncompressor).
Arithmetic Coding

- Like in Shannon-Fano-Elias coding, we divide the unit interval up into segments of length according to the probabilities $p(X_1 = x)$ for $x \in \{a, e, i, o, u, !\}$.
- Consider the following figure:
Arithmetic Coding

- Each subinterval may be further divided into segments of (relatively) length $p(X_2 = x_2 | X_1 = x_1)$ or actual length $p(X_2 = x_2, X_1 = x_1)$.
- The following figure shows this, starting with $p(X_1 = a)$. 

```
  1
  !
  u
  o
  i
  e
  a

  p(a, e)
  p(a, e, i)
  p(a, e, i, o)
  p(a, e, i, o, a)
```
Length of interval for “ae” is
\[ p(X_1 = a, X_2 = e) = p(X_1 = a)p(X_2 = e | X_1 = a). \]
Intervals keep getting exponentially smaller with \( n \) larger.
Key: at each stage, relative lengths of the intervals can change depending on history. At \( t = 1 \), relative interval fraction for “a” is \( p(a) \), at \( t = 2 \), relative interval fraction for “a” is \( p(a | X_1) \), which might change depending on \( X_1 \), and so on.
This is different than Shannon-Fano-Elias coding which uses the same interval length at each step.
Thus, if a symbol gets very probable, it uses a long relative interval (few bits), and if it gets very improbable, it uses short relative interval (more bits).
How to code? Let $i$ be the current source symbol number for $X_i$.

We maintain a lower and an upper interval position.

\[
L_n(i | x_1, x_2, \ldots, x_{n-1}) = \sum_{j=1}^{i-1} p(x_n = j | x_1, x_2, \ldots, x_{n-1}) \quad (6)
\]

\[
U_n(i | x_1, x_2, \ldots, x_{n-1}) = \sum_{j=1}^{i} p(x_n = j | x_1, x_2, \ldots, x_{n-1}) \quad (7)
\]

on arrival of $n^{th}$ input symbol, we divide the $(n - 1)^{st}$ interval which is defined by $L_n$ and $U_n$ via the half-open interval $[L_n, U_n)$. 
Interval Divisions

Example: initial interval is \([0, 1)\) and we divide it depending on the symbol we receive.

\[
a \leftrightarrow [L_1(a), U_1(a)] = [0, p(X_1 = a)] \tag{9}
\]

\[
e \leftrightarrow [L_1(e), U_1(e)] = [p(X_1 = a), p(X_1 = a) + p(X_1 = e)] \tag{10}
\]

\[
i \leftrightarrow [L_1(i), U_1(i)] = [p(a) + p(e), p(a) + p(e) + p(i)] \tag{11}
\]

\[
o \leftrightarrow [L_1(o), U_1(o)] = [p(a) + p(e) + p(i), p(a) + p(e) + p(i) + p(o)]
\]

\[
u \leftrightarrow [L_1(u), U_1(u)] = [\sum_{x \in \{a,e,i,o\}} p(x), \sum_{x \in \{a,e,i,o,u\}} p(x)] \tag{12}
\]

\[
! \leftrightarrow [L_1(!), U_1(!)] = [\sum_{x \in \{a,e,i,o,u\}} p(x), 1] \tag{13}
\]
**Interval Divisions**

Example: initial interval is $[0, 1)$ and we divide it depending on the symbol we receive.

$$a \leftrightarrow [L_1(a), U_1(a)] = [0, p(X_1 = a)] \quad (9)$$

$$e \leftrightarrow [L_1(e), U_1(e)] = [p(X_1 = a), p(X_1 = a) + p(X_1 = e)] \quad (10)$$

$$i \leftrightarrow [L_1(i), U_1(i)] = [p(a) + p(e), p(a) + p(e) + p(i)] \quad (11)$$

$$o \leftrightarrow [L_1(o), U_1(o)] = [p(a) + p(e) + p(i), p(a) + p(e) + p(i) + p(o)]$$

$$u \leftrightarrow [L_1(u), U_1(u)] = \left[ \sum_{x \in \{a, e, i, o\}} p(x), \sum_{x \in \{a, e, i, o, u\}} p(x) \right] \quad (12)$$

$$! \leftrightarrow [L_1(!), U_1(!)] = \left[ \sum_{x \in \{a, e, i, o, u\}} p(x), 1 \right] \quad (13)$$

In general, we use an algorithm for the string $x_1, x_2, \ldots$ to derive the intervals $[\ell, u)$ at each time step where $\ell$ is the lower and $u$ is the upper range.
Suppose we want to send $N$ source symbols. Then we can follow the algorithm below.

1. $\ell \leftarrow 0$
2. $u \leftarrow 1$
3. $p \leftarrow u - \ell$
4. \textbf{for} $n = 1 \ldots N$ \textbf{do}
   \hspace{1em} /* First compute for all $i \in X$, $U_n$ and $L_n$ */
   \hspace{1em} $u \leftarrow \ell + pU_n(x_n|x_1, \ldots, x_{n-1})$
   \hspace{1em} $\ell \leftarrow \ell + pL_n(x_n|x_1, \ldots, x_{n-1})$
   \hspace{1em} $p \leftarrow u - \ell$
Once we have final interval, to encode we simply send any binary string that lives in the interval \([\ell, u]\) after running the algorithm.

On the other hand, we can make the algorithm online, so that it starts writing out bits in the interval once they are known unambiguously.

Analogous to Shannon-Fano-Elias coding, if the current interval is \([0.100101, 0.100110)\) then we can send the common prefix 1001 since that will not change.
Example

Here is an example (Let □ be a termination symbol):

<table>
<thead>
<tr>
<th></th>
<th>(p(a)) = 0.425</th>
<th>(p(b)) = 0.425</th>
<th>(p(□)) = 0.15</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(p(a</td>
<td>b)) = 0.28</td>
<td>(p(b</td>
</tr>
<tr>
<td></td>
<td>(p(a</td>
<td>bb)) = 0.21</td>
<td>(p(b</td>
</tr>
<tr>
<td></td>
<td>(p(a</td>
<td>bbb)) = 0.17</td>
<td>(p(b</td>
</tr>
<tr>
<td></td>
<td>(p(a</td>
<td>bbba)) = 0.28</td>
<td>(p(b</td>
</tr>
</tbody>
</table>

- With these probabilities, we will consider encoding the string \(bbba□\), and we’ll get the final interval

\[
\begin{align*}
-10011110 & 10011110 \\
10011111 & \\
10100000 & \\
100111101 &
\end{align*}
\]

- I.e, the final code word will be 100111101
- Lets look at the entire picture
Coding Example from D.J.C. MacKay’s 2001 book.

```
a
ba
b
b
bb
b

0000
0001
0001
0010
0011
0011
0100
0101
0101
0110
0111
0111
1000
1001
1001
1010
1011
1011
1100
1101
1101
1110
1111
1111
```

```
bb
b
b
bb
b

0
1
00
01
000
001
0000
0001
0001
0001
0001
0010
0011
0100
0101
0101
0101
0101
0110
0111
0111
0111
0111
```

```
bba
bbaa
bbba
bbbab
bbba
10011
10010111
10011000
10011001
10011010
10011011
10011100
10011101
10011110
10011111
10100000
```

```
bb
b
b
b
b

- 10010111
- 10011000
- 10011001
- 10011010
- 10011011
- 10011100
- 10011101
- 10011110
- 10011111
- 10100000
```
Coding Example from D.J.C. MacKay's 2001 book.

0 ≤ p < 1/4

1/4 ≤ v < 1/2

1/2 ≤ p < 1

3/4 ≤ v < 1

selected codeword
Q: Why can’t we use 1001111? A: Because its interval is too large. Codeword 100111101’s interval is entirely within bbba□’s interval, so we are prefix free.
To decode a binary string, say $\alpha = 0.z_1z_2z_3 \ldots$ we use algorithm:

1. $\ell \leftarrow 0$
2. $u \leftarrow 1$
3. $p \leftarrow u - \ell$
4. while special symbol $\square$ is not received do
5.   find $i$ such that:
6.     \[ L_n(i|x_1, \ldots, x_{n-1}) \leq \frac{\alpha - \ell}{u - \ell} < U_n(i|x_1, \ldots, x_{n-1}) \]
7.     $u \leftarrow \ell + pU_n(i|x_1, \ldots, x_{n-1})$
8.     $\ell \leftarrow \ell + pL_n(i|x_1, \ldots, x_{n-1})$
9.     $p \leftarrow u - \ell$
Number of bits

- Problem is, a given number in the final interval \([L_n, U_n]\) could be arbitrarily long (e.g., repeated or irrational number). We only need to send enough to uniquely identify string.
Problem is, a given number in the final interval \([L_n, U_n]\) could be arbitrarily long (e.g., repeated or irrational number). We only need to send enough to uniquely identify string.

How do we choose the number of bits to send?
Number of bits

- Problem is, a given number in the final interval \([L_n, U_n]\) could be arbitrarily long (e.g., repeated or irrational number). We only need to send enough to uniquely identify the string.
- How do we choose the number of bits to send?
- Define

\[
F_n(i|x_1, x_2, \ldots, x_{n-1}) = \frac{1}{2}[L_n(i) + U_n(i)]
\]  

(15)

and \([F_n(i|x_1, x_2, \ldots, x_{n-1})]_\ell\) which is \(F_n\) truncated to \(\ell\) bits.
Number of bits

- Problem is, a given number in the final interval \([L_n, U_n]\) could be arbitrarily long (e.g., repeated or irrational number). We only need to send enough to uniquely identify string.
- How do we choose the number of bits to send?
- Define

\[
F_n(i|x_1, x_2, \ldots, x_{n-1}) = \frac{1}{2}[L_n(i) + U_n(i)]
\]  

(15)

and \(\lfloor F_n(i|x_1, x_2, \ldots, x_{n-1}) \rfloor_\ell\) which is \(F_n\) truncated to \(\ell\) bits.
- We could use \(\ell(x_n|x_1, \ldots, x_{n-1}) = \lceil \log 1/p(x_n|x_1, \ldots, x_{n-1}) \rceil + 1\).
Problem is, a given number in the final interval $[L_n, U_n)$ could be arbitrarily long (e.g., repeated or irrational number). We only need to send enough to uniquely identify string.

How do we choose the number of bits to send?

Define

$$F_n(i| x_1, x_2, \ldots, x_{n-1}) = \frac{1}{2}[L_n(i) + U_n(i)]$$

(15)

and $[F_n(i| x_1, x_2, \ldots, x_{n-1})]_{\ell}$ which is $F_n$ truncated to $\ell$ bits.

We could use

$$\ell(x_n| x_1, \ldots, x_{n-1}) = \lceil\log 1/p(x_n| x_1, \ldots, x_{n-1})\rceil + 1$$

Instead, lets use the Shannon length of the entire code as

$$\ell(x_{1:n}) = \lceil\log 1/p(x_{1:n})\rceil + 1$$

(16)
By the same arguments we made for the Shannon-Fano-Elias codes, this is a prefix code and thus is uniquely decodable, etc.

Also, we have:

\[
E_\ell(x_{1:n}) = \sum_{x_{1:n}} p(x_{1:n}) \ell(x_{1:n})
\]

\[
= \sum_{x_{1:n}} p(x_{1:n}) \left( \lceil \log \frac{1}{p(x_{1:n})} \rceil + 1 \right)
\]

\[
\leq - \sum_{x_{1:n}} p(x_{1:n}) \log p(x_{1:n}) + 2
\]

\[
= H(x_{1:n}) + 2
\]

So the per symbol length \( \leq H(X_{1:n})/2 + 2/n \rightarrow H(X) \)

But this was not a block code.
Estimating $p(x_n | x_1, \ldots, x_{n-1})$

- We still have the problem that we need to estimate $p(x_n | x_1, \ldots, x_{n-1})$
Estimating $p(x_n|x_1, \ldots, x_{n-1})$

- We still have the problem that we need to estimate $p(x_n|x_1, \ldots, x_{n-1})$
- We’d like to use adaptive models.
Estimating $p(x_n | x_1, \ldots, x_{n-1})$

- We still have the problem that we need to estimate $p(x_n | x_1, \ldots, x_{n-1})$.
- We’d like to use adaptive models.
- One possibility is the Dirichlet model, having no independencies:

$$p(a | x_1:n-1) = \frac{N(a | x_1:n-1) + \alpha}{\sum_{a'}(N(a' | x_1:n-1) + \alpha)}$$

(21)

Small $\alpha$ means more responsive
Large $\alpha$ means more sluggish.

How do we derive this? We can do so in a Bayesian setting. In general the problem of density estimation is a topic in and of itself.
We still have the problem that we need to estimate
\[ p(x_n | x_1, \ldots, x_{n-1}) \]
We’d like to use adaptive models.
One possibility is the Dirichlet model, having no independencies:

\[
p(a | x_{1:n-1}) = \frac{N(a | x_{1:n-1}) + \alpha}{\sum_{a'} (N(a' | x_{1:n-1}) + \alpha)}
\]

Small \( \alpha \) means more responsive
Estimating $p(x_n|x_1, \ldots, x_{n-1})$

- We still have the problem that we need to estimate $p(x_n|x_1, \ldots, x_{n-1})$.
- We’d like to use adaptive models.
- One possibility is the Dirichlet model, having no independencies:

$$p(a|x_1:n−1) = \frac{N(a|x_1:n−1) + \alpha}{\sum_{a'}(N(a'|x_1:n−1) + \alpha)} \quad (21)$$

- Small $\alpha$ means more responsive.
- Large $\alpha$ means more sluggish.
Estimating $p(x_n|x_1,\ldots,x_{n-1})$

- We still have the problem that we need to estimate $p(x_n|x_1,\ldots,x_{n-1})$.
- We’d like to use adaptive models.
- One possibility is the Dirichlet model, having no independencies:

$$p(a|x_{1:n-1}) = \frac{N(a|x_{1:n-1}) + \alpha}{\sum_{a'}(N(a'|x_{1:n-1}) + \alpha)} \quad (21)$$

- Small $\alpha$ means more responsive.
- Large $\alpha$ means more sluggish.
- How do we derive this? We can do so in a Bayesian setting.
Estimating $p(x_n | x_1, \ldots, x_{n-1})$

- We still have the problem that we need to estimate $p(x_n | x_1, \ldots, x_{n-1})$.
- We'd like to use adaptive models.
- One possibility is the Dirichlet model, having no independencies:

$$p(a | x_{1:n-1}) = \frac{N(a | x_{1:n-1}) + \alpha}{\sum_{a'}(N(a' | x_{1:n-1}) + \alpha)}$$

- Small $\alpha$ means more responsive.
- Large $\alpha$ means more sluggish.
- How do we derive this? We can do so in a Bayesian setting.
- In general the problem of density estimation is a topic in and of itself.
Laplace’s rule: Bayesian derivation

- For simplicity, assume binary alphabet, so $\mathcal{X} = \{0, 1\}$. 
Laplace’s rule: Bayesian derivation

- For simplicity, assume binary alphabet, so $\mathcal{X} = \{0, 1\}$.
- $N_0 = N(0|x_{1:n})$ and $N_1 = N(1|x_{1:n})$ counts, and $N = N_0 + N_1$. 

Assumptions: there exists $p_0$, $p_1$ and $p_2$ for $0$, $1$, and the termination symbol.

Length of a given string has a geometric distribution. I.e.,

$p(\ell) = (1 - p_2)\ell p_2^{(\ell)}$ (22)

so that

$E\ell = \frac{1}{p_2}$

and

$\text{var}(\ell) = \frac{(1 - p_2)}{p_2^2}$.

Characters are drawn independently, so that

$p(x_{1:n}|p_0, N) = p_0^{N_0} p_1^{N_1}$ (23)

this is the “likelihood” function of the data.

Uniform priors, i.e.,

$\text{Pr}(p_0) = 1$ for $p_0 \in [0, 1]$. 
Laplace’s rule: Bayesian derivation

- For simplicity, assume binary alphabet, so \( \mathcal{X} = \{0, 1\} \).
- \( N_0 = N(0|x_{1:n}) \) and \( N_1 = N(1|x_{1:n}) \) counts, and \( N = N_0 + N_1 \).
- Assumptions: there exists \( p_0, p_1 \) and \( p_{\square} \) for 0, 1, and the termination symbol.
Laplace’s rule: Bayesian derivation

- For simplicity, assume binary alphabet, so $\mathcal{X} = \{0, 1\}$.
- $N_0 = N(0|x_{1:n})$ and $N_1 = N(1|x_{1:n})$ counts, and $N = N_0 + N_1$.
- Assumptions: there exists $p_0$, $p_1$ and $p_{\square}$ for 0, 1, and the termination symbol.
- Length of a given string has a geometric distribution. I.e.,

\[ p(\ell) = (1 - p_{\square})^\ell p_{\square} \quad (22) \]

so that $E\ell = 1/p_{\square}$ and $\text{var}(\ell) = (1 - p_{\square})/p_{\square}^2$. 
Laplace’s rule: Bayesian derivation

- For simplicity, assume binary alphabet, so \( \mathcal{X} = \{0, 1\} \).

- \( N_0 = N(0|x_{1:n}) \) and \( N_1 = N(1|x_{1:n}) \) counts, and \( N = N_0 + N_1 \).

- Assumptions: there exists \( p_0, p_1 \) and \( p_{\square} \) for 0, 1, and the termination symbol.

- Length of a given string has a geometric distribution. I.e.,

\[
p(\ell) = (1 - p_{\square})^\ell p_{\square}
\]

so that \( E\ell = 1/p_{\square} \) and \( \text{var}(\ell) = (1 - p_{\square})/p_{\square}^2 \).

- Characters are drawn independently, so that

\[
p(x_{1:n}|p_0, N) = p_0^{N_0} p_1^{N_1}
\]

this is the “likelihood” function of the data.
For simplicity, assume binary alphabet, so $\mathcal{X} = \{0, 1\}$.

- $N_0 = N(0|x_{1:n})$ and $N_1 = N(1|x_{1:n})$ counts, and $N = N_0 + N_1$.

- Assumptions: there exists $p_0$, $p_1$ and $p_\square$ for 0, 1, and the termination symbol.

- Length of a given string has a geometric distribution. I.e.,

$$p(\ell) = (1 - p_\square)^\ell p_\square \quad (22)$$

so that $E\ell = 1/p_\square$ and $\text{var}(\ell) = (1 - p_\square)/p_\square^2$.

- Characters are drawn independently, so that

$$p(x_{1:n}|p_0, N) = p_0^{N_0} p_1^{N_1} \quad (23)$$

this is the “likelihood” function of the data.

- Uniform priors, i.e., $\Pr(p_0) = 1$ for $p_0 \in [0, 1]$. 
Laplace’s rule: Bayesian derivation

Then

\[
\Pr(p_0|x_{1:n}, N) = \frac{\Pr(x_{1:n}|p_0, N)\Pr(p_0)}{\Pr(x_{1:n}|N)}
\]

\[
\Rightarrow \Pr(p_0|x_{1:n}, N) = \frac{p_0^{N_0} p_1^{N_1}}{\Pr(x_{1:n}|N)}
\]
Laplace’s rule: Bayesian derivation

Then

\[
\Pr(p_0|x_{1:n}, N) = \frac{\Pr(x_{1:n}|p_0, N)\Pr(p_0)}{\Pr(x_{1:n}|N)}
\]

\[
\Rightarrow \Pr(p_0|x_{1:n}, N) = \frac{p_0^{N_0} p_1^{N_1}}{\Pr(x_{1:n}|N)}
\]

Using the Beta integral, we can get:

\[
\Pr(x_{1:n}|N) = \int_0^1 \Pr(x_{1:n}, p_0|N)dp_0 = \int_0^1 p_0^{N_0} p_1^{N_1} \Pr(p_0)dp_0
\]

\[
= \frac{\Gamma(N_0 + 1)\Gamma(N_1 + 1)}{\Gamma(N_0 + N_1 + 2)} = \frac{N_0! N_1!}{(N_0 + N_1 + 1)!}
\]
Laplace’s rule: Bayesian derivation

- To make a prediction, we want:

\[
\Pr(X_{n+1} = 0| x_{1:n}, N) = \int_0^1 \Pr(0|p_0) \Pr(p_0|x_{1:n}, N)dp_0
\]  

(28)

\[
= \int_0^1 p_0^{N_0} p_1^{N_1} \Pr(x_{1:n}|N)dp_0
\]  

(29)

\[
= \int_0^1 p_0^{N_0+1} p_1^{N_1} \Pr(x_{1:n}|N)dp_0
\]  

(30)

\[
= \left[ \frac{(N + 0 + 1)!N_1!}{(N_0 + N_1 + 2)!} \right] / \left[ \frac{N_0!N_1!}{(N_0 + N_1 + 1)!} \right]
\]

(31)

\[
= \frac{N_0 + 1}{N_0 + N_1 + 2} = \text{Laplace's rule}
\]  

(32)
Laplace’s rule: Bayesian derivation

To make a prediction, we want:

$$\Pr(X_{n+1} = 0 | x_{1:n}, N) = \int_0^1 \underbrace{\Pr(0|p) \Pr(p|x_{1:n}, N)}_{=p_0} dp_0$$

(28)

$$= \int_0^1 p_0 \frac{p_0^{N_0} p_1^{N_1}}{\Pr(x_{1:n}|N)} dp_0$$

(29)

$$= \int_0^1 p_0^{N_0+1} p_1^{N_1} \Pr(x_{1:n}|N) dp_0$$

(30)

$$= \left[ \frac{(N + 0 + 1)!N_1!}{(N_0 + N_1 + 2)!} \right] / \left[ \frac{N_0!N_1!}{(N_0 + N_1 + 1)!} \right]$$

(31)

$$= \frac{N_0 + 1}{N_0 + N_1 + 2} = \text{Laplace’s rule}$$

(32)

Dirichlet’s approach is \((N_0 + \alpha)/(N_0 + N_1 + 2\alpha)\) so the only difference is the fractional \(\alpha\) (Exercise: derive Dirichlet’s formula).
So far, we have been talking about compression. I.e., we have some source $p(x)$ with information $H(X)$ (the limits of compression) and the goal is to compress it down to $H$ bits per source symbol in a representation $Y$. 

Recall efficiency: $H(X)/E\ell$. 

Towards Channels
Towards Channels

- So far, we have been talking about compression. I.e., we have some source \( p(x) \) with information \( H(X) \) (the limits of compression) and the goal is to compress it down to \( H \) bits per source symbol in a representation \( Y \).

- In some sense, the compressed representation has a “capacity” which is the total amount of bits that can be represented.
Towards Channels

- So far, we have been talking about compression. I.e., we have some source \( p(x) \) with information \( H(X) \) (the limits of compression) and the goal is to compress it down to \( H \) bits per source symbol in a representation \( Y \).

- In some sense, the compressed representation has a “capacity” which is the total amount of bits that can be represented. I.e., with \( n \) bits, we can obviously represent no more than \( n \) bits of information.
Towards Channels

- So far, we have been talking about compression. i.e., we have some source $p(x)$ with information $H(X)$ (the limits of compression) and the goal is to compress it down to $H$ bits per source symbol in a representation $Y$.

- In some sense, the compressed representation has a “capacity” which is the total amount of bits that can be represented. i.e., with $n$ bits, we can obviously represent no more than $n$ bits of information.

- Compression can be seen as a process where we want to fully utilize the capacity in the compressed representation, i.e., if we have $n$ bits of code word, we ideally (i.e., in an perfect compression scheme) would like there to be no less than $n$ bits of information being represented.
Towards Channels

- So far, we have been talking about compression. I.e., we have some source $p(x)$ with information $H(X)$ (the limits of compression) and the goal is to compress it down to $H$ bits per source symbol in a representation $Y$.

- In some sense, the compressed representation has a “capacity” which is the total amount of bits that can be represented. I.e., with $n$ bits, we can obviously represent no more than $n$ bits of information.

- Compression can be seen as a process where we want to fully utilize the capacity in the compressed representation, i.e., if we have $n$ bits of code word, we ideally (i.e., in an perfect compression scheme) would like there to be no less than $n$ bits of information being represented. Recall efficiency: $H(X)/E\ell$. 
Towards Channels

- We now want to transmit information over a channel.
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:

  *The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.*
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:

  The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.

  Frequently the messages have meaning . . . [which is] irrelevant to the engineering problem
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:
  
  *The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.*

  *Frequently the messages have meaning . . . [which is] irrelevant to the engineering problem*
Towards Channels

- We now want to transmit information over a channel.

- From Claude Shannon:
  
  *The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.*

  *Frequently the messages have meaning . . . [which is] irrelevant to the engineering problem*

- Is there a limit to the rate of communication over a channel?
Towards Channels

- We now want to transmit information over a channel.
- From Claude Shannon:
  \[\text{The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.}\]
  
  \[\text{Frequently the messages have meaning . . . [which is] irrelevant to the engineering problem}\]

- Is there a limit to the rate of communication over a channel?
- If the channel is noisy, can we achieve (essentially) perfect error-free transmission at a reasonable rate?
The 1930s America

- Stock-market crash of 1929
- The great depression
- Terrible conditions in textile and mining industries, deflated crop prices, soil depletion, farm mechanization
- The rise of fascism and the Nazi state in Europe.
- Analog radio, and urgent need for secure, precise, and efficient communications.
- Radio communication, noise always in data transmission (except for Morse code, which is slow).
The 1930s America

- Stock-market crash of 1929

From Wikipedia
The 1930s America

- Stock-market crash of 1929
- The great depression
The 1930s America

- Stock-market crash of 1929
- The great depression
- Terrible conditions in textile and mining industries

From Wikipedia
The 1930s America

- Stock-market crash of 1929
- The great depression
- Terrible conditions in textile and mining industries
- deflated crop prices, soil depletion, farm mechanization

From Wikipedia
The 1930s America

- Stock-market crash of 1929
- The great depression
- Terrible conditions in textile and mining industries
- Deflated crop prices, soil depletion, farm mechanization
- The rise of fascism and the Nazi state in Europe.

From Wikipedia
The 1930s America

- Stock-market crash of 1929
- The great depression
- Terrible conditions in textile and mining industries
- Deflated crop prices, soil depletion, farm mechanization
- The rise of fascism and the Nazi state in Europe.
- Analog radio, and urgent need for secure, precise, and efficient communications

From Wikipedia
The 1930s America

- Stock-market crash of 1929
- The great depression
- Terrible conditions in textile and mining industries
- Deflated crop prices, soil depletion, farm mechanization
- The rise of fascism and the Nazi state in Europe.
- Analog radio, and urgent need for secure, precise, and efficient communications
- Radio communication, noise always in data transmission (except for Morse code, which is slow).

From Wikipedia
Radio Communications

- Place yourself back in the 1930s.
Radio Communications

- Place yourself back in the 1930s.
- Analog communication model of the 1930s.

Q: Can we achieve perfect communication with an imperfect communication channel?
Q: Is there an upper bound on the information capable of being sent under different noise conditions?
If we increase the transmission rate over a noisy channel will the error rate increase?
Radio Communications

- Place yourself back in the 1930s.
- Analog communication model of the 1930s.

Q: Can we achieve perfect communication with an imperfect communication channel?
Radio Communications

- Place yourself back in the 1930s.
- Analog communication model of the 1930s.

Q: Can we achieve perfect communication with an imperfect communication channel?
Q: Is there an upper bound on the information capable of being sent under different noise conditions?
Radio Communications

- Place yourself back in the 1930s.
- Analog communication model of the 1930s.

Q: Can we achieve perfect communication with an imperfect communication channel?
Q: Is there an upper bound on the information capable of being sent under different noise conditions?
If we increase the transmission rate over a noisy channel will the error rate increase?
Radio Communications

- Key: If we increase the transmission rate over a noisy channel will the error rate increase?
Radio Communications

- Key: If we increase the transmission rate over a noisy channel will the error rate increase?
- Perhaps the only way to achieve error free communication is to have a rate of zero.
Radio Communications

- Key: If we increase the transmission rate over a noisy channel will the error rate increase?
- Perhaps the only way to achieve error free communication is to have a rate of zero.
- **The error profile we might expect to see is the following:**

![Error profile graph](image-url)
Radio Communications

- Key: If we increase the transmission rate over a noisy channel will the error rate increase?
- Perhaps the only way to achieve error free communication is to have a rate of zero.
- The error profile we might expect to see is the following:

Here, probability of error $P_e$ goes up linearly with the rate $R$, with an intercept at zero.
Radio Communications

- Key: If we increase the transmission rate over a noisy channel will the error rate increase?
- Perhaps the only way to achieve error free communication is to have a rate of zero.
- The error profile we might expect to see is the following:

\[ P_e \rightarrow R \]

Here, probability of error \( P_e \) goes up linearly with the rate \( R \), with an intercept at zero.
- This was the prevailing wisdom at the time. Shannon was critical in changing that.
Simple Example

Consider representing a signal by a sequence of numbers.
Simple Example

- Consider representing a signal by a sequence of numbers.
- We now know that any signal (either inherently discrete or continuous, under the right conditions) can be perfectly represented (or at least arbitrarily well) by a sequence of discrete numbers, and they can even be binary digits.
Simple Example

- Consider representing a signal by a sequence of numbers.
- We now know that any signal (either inherently discrete or continuous, under the right conditions) can be perfectly represented (or at least arbitrarily well) by a sequence of discrete numbers, and they can even be binary digits.
- Now consider speaking such a sequence over a noisy AM channel.
Simple Example

- Consider representing a signal by a sequence of numbers.
- We now know that any signal (either inherently discrete or continuous, under the right conditions) can be perfectly represented (or at least arbitrarily well) by a sequence of discrete numbers, and they can even be binary digits.
- Now consider speaking such a sequence over a noisy AM channel.
- Very possible one number will be masked by noise.
Consider representing a signal by a sequence of numbers.

We now know that any signal (either inherently discrete or continuous, under the right conditions) can be perfectly represented (or at least arbitrarily well) by a sequence of discrete numbers, and they can even be binary digits.

Now consider speaking such a sequence over a noisy AM channel.

Very possible one number will be masked by noise.

In such case, each number we repeat $k$ times, where $k$ is sufficiently large to ensure we can “decode” the original sequence with very small probability of error.
Simple Example

- Consider representing a signal by a sequence of numbers.
- We now know that any signal (either inherently discrete or continuous, under the right conditions) can be perfectly represented (or at least arbitrarily well) by a sequence of discrete numbers, and they can even be binary digits.
- Now consider speaking such a sequence over a noisy AM channel.
- Very possible one number will be masked by noise.
- In such case, each number we repeat $k$ times, where $k$ is sufficiently large to ensure we can “decode” the original sequence with very small probability of error.
- Rate of this code decreases but we can communicate reliably even if the channel is very noisy.
Simple Example

- Consider representing a signal by a sequence of numbers.
- We now know that any signal (either inherently discrete or continuous, under the right conditions) can be perfectly represented (or at least arbitrarily well) by a sequence of discrete numbers, and they can even be binary digits.
- Now consider speaking such a sequence over a noisy AM channel.
- Very possible one number will be masked by noise.
- In such case, each number we repeat $k$ times, where $k$ is sufficiently large to ensure we can “decode” the original sequence with very small probability of error.
- Rate of this code decreases but we can communicate reliably even if the channel is very noisy.
- Compare this idea to the figure on the following page.
A key idea

- If we choose the messages carefully at the sender, then with very high probability, they will be uniquely identifiable at the receiver.
A key idea

- If we choose the messages carefully at the sender, then with very high probability, they will be uniquely identifiable at the receiver.

- The idea is that we choose the source messages that (tend to) not have any ambiguity (or have any overlap) at the receiver end. I.e.,
A key idea

- If we choose the messages carefully at the sender, then with very high probability, they will be uniquely identifiable at the receiver.

- The idea is that we choose the source messages that (tend to) not have any ambiguity (or have any overlap) at the receiver end. I.e.,

  ![Diagram showing source messages and received messages with unique correspondences]

- This might restrict our possible set of source messages (in some cases severely, and thereby decrease our rate $R$), but if any message received in a region corresponds to only one source message, “perfect” communication can be achieved.
Definition 4.1 (discrete channel)

A discrete channel is one where there is an input alphabet $\mathcal{X}$, an output alphabet $\mathcal{Y}$, and a distribution $p(y|x)$ which is the probability of observing output $y$ after seeing $x$ as input.
Discrete Channels

Definition 4.1 (discrete channel)
A discrete channel is one where there is an input alphabet $\mathcal{X}$, an output alphabet $\mathcal{Y}$, and a distribution $p(y|x)$ which is the probability of observing output $y$ after seeing $x$ as input.

Definition 4.2 (memoryless channel)
A discrete channel is memoryless if $y_t$, the output at time $t$, is independent of all previous inputs, given $x_t$. I.e., $y_t \perp \perp x_{1:t-1} | x_t$. 

Discrete Channels

Definition 4.1 (discrete channel)
A discrete channel is one where there is an input alphabet $\mathcal{X}$, an output alphabet $\mathcal{Y}$, and a distribution $p(y|x)$ which is the probability of observing output $y$ after seeing $x$ as input.

Definition 4.2 (memoryless channel)
A discrete channel is memoryless if $y_t$, the output at time $t$, is independent of all previous inputs, given $x_t$. I.e., $y_t \perp \!\!\!\perp x_{1:t-1}|x_t$.

- We will see many instances of discrete memoryless channels (or just DMC).
Discrete Channels

**Definition 4.1 (discrete channel)**

A discrete channel is one where there is an input alphabet $\mathcal{X}$, an output alphabet $\mathcal{Y}$, and a distribution $p(y|x)$ which is the probability of observing output $y$ after seeing $x$ as input.

**Definition 4.2 (memoryless channel)**

A discrete channel is memoryless if $y_t$, the output at time $t$, is independent of all previous inputs, given $x_t$. I.e., $y_t \perp \perp x_{1:t-1} | x_t$.

- We will see many instances of discrete memoryless channels (or just DMC).
- Recall back from lecture 1 our general model of communications:
Model of Communication

- **Source** message $W$, one of $M$ messages.

\[
\text{source} \rightarrow \text{encoder} \rightarrow \text{channel} \rightarrow \text{decoder} \rightarrow \text{receiver}
\]

- **noise** $p(y|x)$

\[
W \quad X^n \quad Y^n \quad \hat{W}
\]

- $n = \log M$ bits
- $n \log |\mathcal{X}|$ bits
- $n \log |\mathcal{Y}|$ bits
Model of Communication

- **Source** message $W$, one of $M$ messages.
- **Encoder** transforms this into a length-$n$ string of source symbols $X^n$.

$$n = \log M \text{ bits} \quad n \log |X| \text{ bits} \quad n \log |Y| \text{ bits}$$
Model of Communication

- **Source** message $W$, one of $M$ messages.
- **Encoder** transforms this into a length-$n$ string of source symbols $X^n$.
- **Noisy channel** distorts this message into a length-$n$ string of receiver symbols $Y^n$.

\[
W \xrightarrow{n = \log M \text{ bits}} X^n \xrightarrow{Y^n \text{ bits}} \hat{W} \xrightarrow{n \log |\mathcal{X}| \text{ bits}} \hat{W} \xrightarrow{n \log |\mathcal{Y}| \text{ bits}} \text{receiver}
\]

\[\text{noise } p(y|x)\]
**Model of Communication**

- **Source** message $W$, one of $M$ messages.
- **Encoder** transforms this into a length-$n$ string of source symbols $X^n$.
- **Noisy channel** distorts this message into a length-$n$ string of receiver symbols $Y^n$.
- **Decoder** attempts to reconstruct original message as best as possible, comes up with $\hat{W}$, one of $M$ possible sent messages.

\[
p(y|x) = \text{noise}
\]

\[
\begin{align*}
W & \quad X^n & \quad Y^n & \quad \hat{W} \\
n & = \log M \text{ bits} & n \log |\mathcal{X}| \text{ bits} & n \log |\mathcal{Y}| \text{ bits}
\end{align*}
\]
Rates and Capacities

- So we have a source $X$ governed by $p(x)$ and channel that transforms $X$ symbols to $Y$ symbols and which is governed by the conditional distribution $p(y|x)$.
Rates and Capacities

- So we have a source $X$ governed by $p(x)$ and channel that transforms $X$ symbols to $Y$ symbols and which is governed by the conditional distribution $p(y|x)$
- These two items $p(x)$ and $p(y|x)$ is sufficient to compute the mutual information between $X$ and $Y$. 

$I(X;Y) = I_{p(x)}(X,Y)$, meaning implicitly the MI quantity is a function of the entire distribution $p(x)$, for a given fixed channel $p(y|x)$.

We will often be optimizing over the input distribution $p(x)$ for a given fixed channel $p(y|x)$. 

- Prof. Jeff Bilmes
Rates and Capacities

- So we have a source $X$ governed by $p(x)$ and channel that transforms $X$ symbols to $Y$ symbols and which is governed by the conditional distribution $p(y|x)$
- These two items $p(x)$ and $p(y|x)$ is sufficient to compute the mutual information between $X$ and $Y$. That is, we compute

$$I(X; Y) = I_{p(x)}(X, Y) = \sum_{x,y} p(x)p(y|x) \log \frac{p(y|x)}{p(y)}$$  \hspace{1cm} (33)$$

$$= \sum_{x,y} p(x)p(y|x) \log \frac{p(y|x)}{\sum_{x'} p(y|x')p(x')}$$  \hspace{1cm} (34)$$
Rates and Capacities

- So we have a source $X$ governed by $p(x)$ and channel that transforms $X$ symbols to $Y$ symbols and which is governed by the conditional distribution $p(y|x)$.

- These two items $p(x)$ and $p(y|x)$ is sufficient to compute the mutual information between $X$ and $Y$. That is, we compute

$$I(X; Y) = I_{p(x)}(X, Y) = \sum_{x,y} p(x)p(y|x) \log \frac{p(y|x)}{p(y)}$$

$$= \sum_{x,y} p(x)p(y|x) \log \frac{p(y|x)}{\sum_{x'} p(y|x')p(x')}$$

- We write this as $I(X; Y) = I_{p(x)}(X, Y)$, meaning implicitly the MI quantity is a function of the entire distribution $p(x)$, for a given fixed channel $p(y|x)$. 
Rates and Capacities

- So we have a source $X$ governed by $p(x)$ and channel that transforms $X$ symbols to $Y$ symbols and which is governed by the conditional distribution $p(y|x)$.

- These two items $p(x)$ and $p(y|x)$ is sufficient to compute the mutual information between $X$ and $Y$. That is, we compute

$$I(X;Y) = I_{p(x)}(X,Y) = \sum_{x,y} p(x)p(y|x) \log \frac{p(y|x)}{p(y)}$$

(33)

$$= \sum_{x,y} p(x)p(y|x) \log \frac{p(y|x)}{\sum_{x'} p(y|x')p(x')}$$

(34)

- We write this as $I(X;Y) = I_{p(x)}(X,Y)$, meaning implicitly the MI quantity is a function of the entire distribution $p(x)$, for a given fixed channel $p(y|x)$.

- We will often be optimizing over the input distribution $p(x)$ for a given fixed channel $p(y|x)$. 

---

Prof. Jeff Bilmes
Definition 4.3 (information flow)

The rate of information flow through a channel is given by $I(X; Y)$, the mutual information between $X$ and $Y$, in units of bits per channel use.
Rates and Capacities

Definition 4.3 (information flow)

The rate of information flow through a channel is given by $I(X; Y)$, the mutual information between $X$ and $Y$, in units of bits per channel use.

Definition 4.4 (capacity)

The information capacity of a channel is the maximum information flow.

$$C \triangleq \max_{p(x) \in \Delta} I(X; Y) \quad (35)$$

where $\Delta$ is the set of all possible probability distributions over source alphabet $\mathcal{X}$.
The rate of information flow through a channel is given by $I(X;Y)$, the mutual information between $X$ and $Y$, in units of bits per channel use.

The information capacity of a channel is the maximum information flow.

$$\mathcal{C} \triangleq \max_{p(x) \in \Delta} I(X;Y) \quad (35)$$

where $\Delta$ is the set of all possible probability distributions over source alphabet $\mathcal{X}$. Thus, $\mathcal{C}$ is the maximum number of bits sent over the channel per channel use.
Definition 4.3 (information flow)

The rate of information flow through a channel is given by \( I(X;Y) \), the mutual information between \( X \) and \( Y \), in units of bits per channel use.

Definition 4.4 (capacity)

The information capacity of a channel is the maximum information flow.

\[
C \triangleq \max_{p(x) \in \Delta} I(X;Y)
\]  

(35)

where \( \Delta \) is the set of all possible probability distributions over source alphabet \( \mathcal{X} \). Thus, \( C \) is the maximum number of bits sent over the channel per channel use.

Definition 4.5 (rate)

The rate \( R \) of a code is measured in the number of bits per channel use.
Fundamental Limits of Compression

- For compression, if error exponent is positive, then error → 0 exponentially fast as block length → ∞. Note, $P_e \propto e^{-nE(R)}$. 
For compression, if error exponent is positive, then error $\rightarrow 0$ exponentially fast as block length $\rightarrow \infty$. Note, $P_e \propto e^{-nE(R)}$.

That is,

![Graph showing the relationship between Error Exponent $E(R)$, Entropy $H$, and Rate $R$.](image)
Fundamental Limits of Compression

For compression, if error exponent is positive, then error $\rightarrow 0$ exponentially fast as block length $\rightarrow \infty$. Note, $P_e \propto e^{-nE(R)}$.

That is,

Only hope of reducing error was if $R > H$. Something “funny” happens at the entropy rate of the source distribution. Can’t compress below this without incurring error.
For communication, lower bound on probability of error becomes bounded away from 0 as the rate of the code $R$ goes above a fundamental quantity $C$. Note, $P_e \propto e^{-nE(R)}$. That is, $R \rightarrow C_0$ and $E(R)$ is the error exponent only possible way to get low error is if $R < C$. Something funny happens at the point $C$, the capacity of the channel. Note that $C$ is not 0, so can still achieve "perfect" communication over a noisy channel as long as $R < C$. 

Prof. Jeff Bilmes

For communication, lower bound on probability of error becomes bounded away from 0 as the rate of the code $R$ goes above a fundamental quantity $C$. Note, $P_e \propto e^{-nE(R)}$.

That is, $\log P_e$ only possible way to get low error is if $R < C$. Something funny happens at the point $C$, the capacity of the channel. Note that $C$ is not 0, so can still achieve "perfect" communication over a noisy channel as long as $R < C$. 
For communication, lower bound on probability of error becomes bounded away from 0 as the rate of the code $R$ goes above a fundamental quantity $C$. Note, $P_e \propto e^{-n E(R)}$.

That is,

\[ \log P_e \]

only possible way to get low error is if $R < C$. Something funny happens at the point $C$, the capacity of the channel.
For communication, lower bound on probability of error becomes bounded away from 0 as the rate of the code $R$ goes above a fundamental quantity $C$. Note, $P_e \propto e^{-nE(R)}$.

That is,

$$\log P_e \rightarrow \begin{cases} 0 & \text{if } R < C \\ \infty & \text{if } R \geq C \end{cases}$$

only possible way to get low error is if $R < C$. Something funny happens at the point $C$, the capacity of the channel.

Note that $C$ is not 0, so can still achieve “perfect” communication over a noisy channel as long as $R < C$. 
Examples of discrete memoryless channels (BSC)

- Noiseless binary channel, diagram shows $p(y|x)$
  
  $\mathcal{X} = \{0, 1\}$  
  $\mathcal{Y} = \{0, 1\}$

  $0 \xrightarrow{\text{}} 0$
  $1 \xrightarrow{\text{}} 1$

So, $p(y = 0 | x = 0) = 1 = 1 - p(y = 1 | x = 0)$ and $p(y = 1 | x = 1) = 1 = 1 - p(y = 0 | x = 1)$, so channel is just an input copy.

One bit sent at a time is received without error, so capacity should be 1 bit (intuitively, we can reliably send one bit per channel usage).

$I(X; Y) = H(X) - H(X | Y) = H(X)$ in this case, so $C = \max_p I(X; Y) = \max_p H(X) = 1$.

Clearly, $p(0) = p(1) = \frac{1}{2}$ achieves this capacity.

Also, $p(0) = 1 = 1 - p(1)$ has $I(X; Y) = 0$, so achieves zero information flow.
Examples of discrete memoryless channels (BSC)

- Noiseless binary channel, diagram shows $p(y|x)$

  \[ X = \{0, 1\} \quad X \rightarrow 0 \quad Y \quad Y = \{0, 1\} \]

  \[ 1 \rightarrow 1 \]

- So, $p(y = 0|x = 0) = 1 = 1 - p(y = 1|x = 0)$ and $p(y = 1|x = 1) = 1 = 1 - p(y = 0|x = 1)$, so channel is just an input copy.
Examples of discrete memoryless channels (BSC)

- Noiseless binary channel, diagram shows $p(y|x)$

  $X = \{0, 1\}$ \quad $Y = \{0, 1\}$

  $\begin{array}{c}
  0 \\
  1 \\
  \end{array} \quad \begin{array}{c}
  0 \\
  1 \\
  \end{array}$

- So, $p(y = 0|x = 0) = 1 = 1 - p(y = 1|x = 0)$ and $p(y = 1|x = 1) = 1 = 1 - p(y = 0|x = 1)$, so channel is just an input copy.

- One bit sent at a time is received without error, so capacity should be 1 bit (intuitively, we can reliably send one bit per channel usage).
Examples of discrete memoryless channels (BSC)

- Noiseless binary channel, diagram shows $p(y|x)$
  
  $\mathcal{X} = \{0, 1\}$
  
  $X \xrightarrow[]{} 0$
  
  $1 \xrightarrow[]{} 1$
  
  $\mathcal{Y} = \{0, 1\}$

- So, $p(y = 0|x = 0) = 1 = 1 - p(y = 1|x = 0)$ and
  
  $p(y = 1|x = 1) = 1 = 1 - p(y = 0|x = 1)$, so channel is just an input copy.

- One bit sent at a time is received without error, so capacity should be 1 bit (intuitively, we can reliably send one bit per channel usage).

- $I(X; Y) = H(X) - H(X|Y) = H(X)$ in this case, so
  
  $C = \max_{p(x)} I(X; Y) = \max_{p(x)} H(X) = 1.$
Examples of discrete memoryless channels (BSC)

- Noiseless binary channel, diagram shows $p(y|x)$

  $X = \{0, 1\}$  
  $Y = \{0, 1\}$

  $0 \rightarrow 0$  
  $1 \rightarrow 1$

  So, $p(y = 0|x = 0) = 1 = 1 - p(y = 1|x = 0)$ and $p(y = 1|x = 1) = 1 = 1 - p(y = 0|x = 1)$, so channel is just an input copy.

- One bit sent at a time is received without error, so capacity should be 1 bit (intuitively, we can reliably send one bit per channel usage).

- $I(X; Y) = H(X) - H(X|Y) = H(X)$ in this case, so $C = \max_{p(x)} I(X; Y) = \max_{p(x)} H(X) = 1$.

- Clearly, $p(0) = p(1) = 1/2$ achieves this capacity.
Examples of discrete memoryless channels (BSC)

- Noiseless binary channel, diagram shows $p(y|x)$

  $X = \{0, 1\}$
  $X \rightarrow 0$
  $Y \rightarrow 1$

So, $p(y = 0|x = 0) = 1 = 1 - p(y = 1|x = 0)$ and $p(y = 1|x = 1) = 1 = 1 - p(y = 0|x = 1)$, so channel is just an input copy.

- One bit sent at a time is received without error, so capacity should be 1 bit (intuitively, we can reliably send one bit per channel usage).

- $I(X;Y) = H(X) - H(X|Y) = H(X)$ in this case, so $C = \max_{p(x)} I(X;Y) = \max_{p(x)} H(X) = 1$.

- Clearly, $p(0) = p(1) = 1/2$ achieves this capacity.

- Also, $p(0) = 1 = 1 - p(1)$ has $I(X;Y) = 0$, so achieves zero information flow.
Consider the channel

\[
\begin{array}{ccc}
X & \rightarrow & Y \\
0 & \rightarrow & 0, 1/2 \rightarrow 1, 1/2 \rightarrow 2, 1/2 \rightarrow 3 \\
1 & \rightarrow & 2, 1/2 \rightarrow 1, 1/2 \rightarrow 3
\end{array}
\]

Here, \( p(Y = 0|X = 0) = p(Y = 1|X = 0) = 1/2 \) and \( p(Y = 2|X = 1) = p(Y = 3|X = 1) = 1/2 \).
Noisy Channel with non-overlapping outputs

Consider the channel

Here, \( p(Y = 0|X = 0) = p(Y = 1|X = 0) = 1/2 \) and \( p(Y = 2|X = 1) = p(Y = 3|X = 1) = 1/2 \).

If we receive a 0 or 1, we know 0 was sent. If we receive a 2 or 3, a 1 was sent.
Noisy Channel with non-overlapping outputs

Consider the channel

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

- Here, \( p(Y = 0|X = 0) = p(Y = 1|X = 0) = 1/2 \) and \( p(Y = 2|X = 1) = p(Y = 3|X = 1) = 1/2 \).
- If we receive a 0 or 1, we know 0 was sent. If we receive a 2 or 3, a 1 was sent.
- Thus, \( C = 1 \) since only two possible error free messages.
Noisy Channel with non-overlapping outputs

Consider the channel

\[
\begin{array}{c}
\text{X} \\
\downarrow \\
\text{0} \quad \frac{1}{2} \quad \text{1} \\
\downarrow \\
\text{Y} \\
\downarrow \\
\text{0} \\
\downarrow \\
\text{1} \quad \frac{1}{2} \quad \text{2} \\
\downarrow \\
\text{1} \quad \frac{1}{2} \quad \text{3} \\
\end{array}
\]

- Here, \( p(Y = 0|X = 0) = p(Y = 1|X = 0) = 1/2 \) and \( p(Y = 2|X = 1) = p(Y = 3|X = 1) = 1/2 \).
- If we receive a 0 or 1, we know 0 was sent. If we receive a 2 or 3, a 1 was sent.
- Thus, \( C = 1 \) since only two possible error free messages.
- Same argument applies
  \[
  I(X;Y) = H(X) - H(X|Y) = H(X).
  \]
  \[
  = 0
  \]
Consider the channel

\[
\begin{array}{c|c}
X & Y \\
\hline
0 & 0, 1/2 \\
1 & 1, 1/2 \\
2 & 2, 1/2 \\
3 & 3, 1/2 \\
\end{array}
\]

Here, \( p(Y = 0|X = 0) = p(Y = 1|X = 0) = 1/2 \) and \( p(Y = 2|X = 1) = p(Y = 3|X = 1) = 1/2 \).

If we receive a 0 or 1, we know 0 was sent. If we receive a 2 or 3, a 1 was sent.

Thus, \( C = 1 \) since only two possible error free messages.

Same argument applies
\[
I(X;Y) = H(X) - H(X|Y) = H(X).
\]

Again uniform distribution \( p(0) = p(1) = 1/2 \) achieves the capacity.
Permutation Channel

Consider the channel

\[
\begin{array}{c}
X \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \ Quad
Permutation Channel

Consider the channel

Here, \( p(Y = 1|X = 0) = p(Y = 0|X = 1) = 1 \).

So output is a binary permutation (swap) of input.
Permutation Channel

Consider the channel

\[ X \quad 0 \quad 1 \quad \rightarrow \quad 0 \quad 1 \quad Y \]

Here, \( p(Y = 1|X = 0) = p(Y = 0|X = 1) = 1 \).

So output is a binary permutation (swap) of input.

Thus, \( C = 1 \) no information lost.
Consider the channel

\[
\begin{array}{ccc}
X & \rightarrow & Y \\
0 & \rightarrow & 0 \\
1 & \rightarrow & 1 \\
\end{array}
\]

- Here, \( p(Y = 1|X = 0) = p(Y = 0|X = 1) = 1 \).
- So output is a binary permutation (swap) of input.
- Thus, \( C = 1 \) no information lost.

- In general, for alphabet of size \( k = |X| = |Y| \), let \( \sigma \) be a permutation, so that \( Y = \sigma(X) \).
Consider the channel

Here, \( p(Y = 1|X = 0) = p(Y = 0|X = 1) = 1 \).

So output is a binary permutation (swap) of input.

Thus, \( C = 1 \) no information lost.

In general, for alphabet of size \( k = |X| = |Y| \), let \( \sigma \) be a permutation, so that \( Y = \sigma(X) \).

Then \( C = \log k \).
Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$. 

We'll be doing this over the next few slides when we want to compute $C = \max_p p(x) I(X;Y)$ for fixed $p(y|x)$. The solution $p^*(x)$ that we find that achieves this maximum won’t necessarily be unique. Also, the solution $p^*(x)$ that we find won’t necessarily be the one that we end up, say, using when we wish to do channel coding. Right now $C$ is just the result of a given optimization. We’ll see that $C$, as computed, is also the critical point for being able to channel code with vanishingly small error probability. The resulting $p^*(x)$ that we obtain as part of the optimization in order to compute $C$ won’t necessarily be the one that we use for actual coding (example forthcoming).
Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$.

- We’ll be doing this over the next few slides when we want to compute $C = \max_{p(x)} I(X; Y)$ for fixed $p(y|x)$.

Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$.

- We’ll be doing this over the next few slides when we want to compute $C = \max p(x) I(X; Y)$ for fixed $p(y|x)$.

- The solution $p^*(x)$ that we find that achieves this maximum won’t necessarily be unique.
Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$.

- We’ll be doing this over the next few slides when we want to compute $C = \max_{p(x)} I(X;Y)$ for fixed $p(y|x)$.

- The solution $p^*(x)$ that we find that achieves this maximum won’t necessarily be unique.

- Also, the solution $p^*(x)$ that we find won’t necessarily be the one that we end up, say, using when we wish to do channel coding.
Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$.

- We’ll be doing this over the next few slides when we want to compute $C = \max_{p(x)} I(X; Y)$ for fixed $p(y|x)$.

- The solution $p^*(x)$ that we find that achieves this maximum won’t necessarily be unique.

- Also, the solution $p^*(x)$ that we find won’t necessarily be the one that we end up, say, using when we wish to do channel coding.

- Right now $C$ is just the result of a given optimization.
Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$.
- We’ll be doing this over the next few slides when we want to compute $C = \max_{p(x)} I(X; Y)$ for fixed $p(y|x)$.
- The solution $p^*(x)$ that we find that achieves this maximum won’t necessarily be unique.
- Also, the solution $p^*(x)$ that we find won’t necessarily be the one that we end up, say, using when we wish to do channel coding.
- Right now $C$ is just the result of a given optimization.
- We’ll see that $C$, as computed, is also the critical point for being able to channel code with vanishingly small error probability.
Asside: on the optimization to compute the value $C$

- To maximize a given function $f(x)$, it is sufficient to show that $f(x) \leq \alpha$ for all $x$, and then find an $x^*$ such that $f(x^*) = \alpha$.
- We’ll be doing this over the next few slides when we want to compute $C = \max_{p(x)} I(X; Y)$ for fixed $p(y|x)$.
- The solution $p^*(x)$ that we find that achieves this maximum won’t necessarily be unique.
- Also, the solution $p^*(x)$ that we find won’t necessarily be the one that we end up, say, using when we wish to do channel coding.
- Right now $C$ is just the result of a given optimization.
- We’ll see that $C$, as computed, is also the critical point for being able to channel code with vanishingly small error probability.
- The resulting $p^*(x)$ that we obtain as part of the optimization in order to compute $C$ won’t necessarily be the one that we use for actual coding (example forthcoming).
Consider the channel

So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor. I.e., $p(A \rightarrow A) = p(A \rightarrow B) = \frac{1}{2}$, etc. Each symbol always has some chance of error, so how can we communicate without error? Choose subset of symbols that can be uniquely disambiguated on receiver side. Choose every other source symbol, A, C, E, etc. Thus $A \rightarrow \{A, B\}$, $C \rightarrow \{C, D\}$, $E \rightarrow \{E, F\}$, etc. so that each received symbols has only one unique source symbol. Capacity $C = \log 13$, visualized on left:

Q: what happens to $C$ when probabilities are not all $\frac{1}{2}$?
Noisy Typewriter

Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.
Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.
- i.e., \( p(A \rightarrow A) = p(A \rightarrow B) = \frac{1}{2} \), etc.
Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.
- I.e., $p(A \rightarrow A) = p(A \rightarrow B) = 1/2$, etc.
- Each symbol always has some chance of error, so how can we communicate without error?

So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.

I.e., $p(A \rightarrow A) = p(A \rightarrow B) = 1/2$, etc.

Each symbol always has some chance of error, so how can we communicate without error?
Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.
- I.e., \( p(A \rightarrow A) = p(A \rightarrow B) = \frac{1}{2} \), etc.
- Each symbol always has some chance of error, so how can we communicate without error?
- Choose subset of symbols that can be uniquely disambiguated on receiver side. Choose every other source symbol, A, C, E, etc.
Noisy Typewriter

Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.

- I.e., \( p(A \rightarrow A) = p(A \rightarrow B) = 1/2 \), etc.

- Each symbol always has some chance of error, so how can we communicate without error?

- Choose subset of symbols that can be uniquely disambiguated on receiver side. Choose every other source symbol, A, C, E, etc.

- Thus \( A \rightarrow \{A, B\}, C \rightarrow \{C, D\}, E \rightarrow \{E, F\} \), etc. so that each received symbols has only one unique source symbol.
Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.
- I.e., \( p(A \to A) = p(A \to B) = 1/2 \), etc.
- Each symbol always has some chance of error, so how can we communicate without error?
- Choose subset of symbols that can be uniquely disambiguated on receiver side. Choose every other source symbol, A, C, E, etc.
- Thus \( A \to \{A, B\}, C \to \{C, D\}, E \to \{E, F\} \), etc. so that each received symbols has only one unique source symbol.
- Capacity \( C = \log 13 \), visualized on left:
Noisy Typewriter

Consider the channel

- So 26 input symbols, and each symbol maps probabilistically to itself or its lexicographic neighbor.
- I.e., \( p(A \rightarrow A) = p(A \rightarrow B) = \frac{1}{2} \), etc.
- Each symbol always has some chance of error, so how can we communicate without error?
- Choose subset of symbols that can be uniquely disambiguated on receiver side. Choose every other source symbol, A, C, E, etc.
- Thus \( A \rightarrow \{A, B\}, \ C \rightarrow \{C, D\}, \ E \rightarrow \{E, F\}, \) etc. so that each received symbols has only one unique source symbol.
- Capacity \( C = \log 13 \), visualized on left:
- Q: what happens to \( C \) when probabilities are not all \( 1/2 \)?
Noisy Typewriter

- We can also compute the capacity more mathematically.
Noisy Typewriter

- We can also compute the capacity more mathematically.
- For example:

$$C = \max_p p(x) I(X;Y) = \max_p p(x) \left( H(Y) - H(Y|X) \right)$$ (36)

$$= \max_p p(x) H(Y) - 1$$  

for $$X = x$$, there are two choices (37)

$$= \log_2 6 - 1 = \log_2 13$$ (38)

The $$\max_p p(x) H(Y) = \log_2 6$$ can be achieved by using the uniform distribution for $$p(x)$$, for which when we choose any $$x$$ symbol, there is equal likelihood of two $$Y$$s being received.

An alternatively $$p(x)$$ would put zero probability on the alternates (B, D, F, etc.). In this case, we still have $$H(Y) = \log_2 6$$ So the capacity is the same in each case (38) but only one is what we would use, say, for error free coding.
We can also compute the capacity more mathematically.

For example:

\[
C = \max_{p(x)} I(X; Y) = \max_{p(x)} \left( H(Y) - H(Y|X) \right)
\]

\[
= \max_{p(x)} H(Y) - 1 \quad \text{// for } X = x, \exists \text{ two choices}
\]

\[
= \log 26 - 1 = \log 13
\]
Noisy Typewriter

- We can also compute the capacity more mathematically.
- For example:

\[
C = \max_{p(x)} I(X; Y) = \max_{p(x)} \left( H(Y) - H(Y|X) \right) \tag{36}
\]

\[
= \max_{p(x)} H(Y) - 1 \quad // \text{for } X = x, \exists \text{ two choices} \tag{37}
\]

\[
= \log 26 - 1 = \log 13 \tag{38}
\]

- The \( \max_{p(x)} H(Y) = \log 26 \) can be achieved by using the uniform distribution for \( p(x) \), for which when we choose any \( x \) symbol, there is equal likelihood of two \( Y \)'s being received.
We can also compute the capacity more mathematically.

For example:

\[
C = \max_{p(x)} I(X; Y) = \max_{p(x)} \left( H(Y) - H(Y|X) \right) = \max_{p(x)} H(Y) - 1 \quad /\!/ \text{for } X = x, \exists \text{two choices} \quad (36)
\]

\[
= \log 26 - 1 = \log 13 \quad (37)
\]

The \( \max_{p(x)} H(Y) = \log 26 \) can be achieved by using the uniform distribution for \( p(x) \), for which when we choose any \( x \) symbol, there is equal likelihood of two \( Y \)s being received.

An alternatively \( p(x) \) would put zero probability on the alternates (B, D, F, etc.). In this case, we still have \( H(Y) = \log 26 \).
We can also compute the capacity more mathematically.

For example:

\[
C = \max_{p(x)} I(X; Y) = \max_{p(x)} \left( H(Y) - H(Y|X) \right)
\]

\[
= \max_{p(x)} H(Y) - 1 \quad \text{for } X = x, \exists \text{ two choices}
\]

\[
= \log 26 - 1 = \log 13
\]

The \( \max_{p(x)} H(Y) = \log 26 \) can be achieved by using the uniform distribution for \( p(x) \), for which when we choose any \( x \) symbol, there is equal likelihood of two \( Y \)s being received.

An alternatively \( p(x) \) would put zero probability on the alternates (B, D, F, etc.). In this case, we still have \( H(Y) = \log 26 \).

So the capacity is the same in each case (\( \exists \) two \( p(x) \) that achieved this) but only one is what we would use, say, for error free coding.
A bit that is sent will be flipped with probability $p$. $p(Y = 1 | X = 0) = p = 1 - p$.

$X$ is a binary symmetric channel (BSC) since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: Can we still achieve reliable ("guaranteed" error free) communication with this channel?

A: Yes, if $p < 1/2$ and if we do not ask for too high a transmission rate (which would be $R > C$), then we can. Actually, any $p \neq 1/2$ is sufficient.

Intuition: think about AEP and/or block coding.

But how to compute $C$, the capacity?
A bit that is sent will be flipped with probability $p$. 

\[ p(Y = 1 | X = 0) = 1 - p \]
\[ p(Y = 0 | X = 0) = p \]
\[ p(Y = 1 | X = 1) = p \]
\[ p(Y = 0 | X = 1) = 1 - p \]

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable ("guaranteed" error free) communication with this channel?

A: Yes, if $p < \frac{1}{2}$ and if we do not ask for too high a transmission rate (which would be $R > C$), then we can. Actually, any $p \neq \frac{1}{2}$ is sufficient.

Intuition: think about AEP and/or block coding.

But how to compute $C$, the capacity?
A bit that is sent will be flipped with probability $p$.

- $p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0)$. $p(Y = 0|X = 1) = p = p(Y = 1|X = 1)$. 

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable (“guaranteed” error free) communication with this channel?

A: Yes, if $p < 1/2$ and if we do not ask for too high a transmission rate (which would be $R > C$), then we can. Actually, any $p \neq 1/2$ is sufficient. Intuition: think about AEP and/or block coding.
Binary Symmetric Channel (BSC)

A bit that is sent will be flipped with probability $p$.

- $p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0)$.
- $p(Y = 0|X = 1) = p = p(Y = 1|X = 1)$.

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.
A bit that is sent will be flipped with probability $p$.

$$p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0). \quad p(Y = 0|X = 1) = p = p(Y = 1|X = 1).$$

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable (“guaranteed” error free) communication with this channel?
A bit that is sent will be flipped with probability $p$.

- $p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0)$.
- $p(Y = 0|X = 1) = p = p(Y = 1|X = 1)$.

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable ("guaranteed" error free) communication with this channel? A: Yes, if $p < 1/2$ and if we do not ask for too high a transmission rate (which would be $R > C$), then we can.
A bit that is sent will be flipped with probability $p$.

\[ p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0). \quad p(Y = 0|X = 1) = p = p(Y = 1|X = 1). \]

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable ("guaranteed" error free) communication with this channel? A: Yes, if $p < 1/2$ and if we do not ask for too high a transmission rate (which would be $R > C$), then we can. Actually, any $p \neq 1/2$ is sufficient.
Binary Symmetric Channel (BSC)

- A bit that is sent will be flipped with probability $p$.
- $p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0)$. $p(Y = 0|X = 1) = p = p(Y = 1|X = 1)$.

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable (“guaranteed” error free) communication with this channel? A: Yes, if $p < 1/2$ and if we do not ask for too high a transmission rate (which would be $R > C$), then we can. Actually, any $p \neq 1/2$ is sufficient.

Intuition: think about AEP and/or block coding.
Binary Symmetric Channel (BSC)

- A bit that is sent will be flipped with probability \( p \).
- \( p(Y = 1|X = 0) = p = 1 - p(Y = 0|X = 0) \).
- \( p(Y = 0|X = 1) = p = p(Y = 1|X = 1) \).

The BSC is an important channel since it is a simple model but at the same time captures some of the difficulties of more complicated channels.

Q: can we still achieve reliable ("guaranteed" error free) communication with this channel? A: Yes, if \( p < 1/2 \) and if we do not ask for too high a transmission rate (which would be \( R > C \)), then we can. Actually, any \( p \neq 1/2 \) is sufficient.

Intuition: think about AEP and/or block coding.

But how to compute \( C \) the capacity?
BSC Capacity

\[ I(X;Y) = H(Y) - H(Y|X) \]
BSC Capacity

\[ I(X; Y) = H(Y) - H(Y|X) = H(Y) - \sum_{x} p(x)H(Y|X = x) \]  \hspace{1cm} (39)

\[ \leq 1 - H(p) \]  \hspace{1cm} (40)
BSC Capacity

\[ I(X;Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x) \quad (39) \]

\[ = H(Y) - \sum_x p(x)H(p) \quad (40) \]
BSC Capacity

\[
I(X;Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x) \quad (39)
\]

\[
= H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \quad (40)
\]
BSC Capacity

\[ I(X; Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x) \]  

\[ = H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \leq 1 - H(p) \]
BSC Capacity

\[ I(X; Y) = H(Y) - H(Y|X) = H(Y) - \sum_{x} p(x)H(Y|X = x) \] (39)

\[ = H(Y) - \sum_{x} p(x)H(p) = H(Y) - H(p) \leq 1 - H(p) \] (40)

- When is \( H(Y) = 1 \)? Note that

\[ \Pr(Y = 1) \]
BSC Capacity

\[
I(X; Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x)
\]

\[
= H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \leq 1 - H(p)
\]

- **When is** \(H(Y) = 1\)? **Note that**

\[
\Pr(Y = 1) = \Pr(Y = 1|X = 1)\Pr(X = 1) + \Pr(Y = 1|X = 0)\Pr(X = 0)
\]
BSC Capacity

\[ I(X;Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x) \quad (39) \]

\[ = H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \leq 1 - H(p) \quad (40) \]

- **When is** \( H(Y) = 1? \) **Note that**

\[ Pr(Y = 1) = Pr(Y = 1|X = 1)Pr(X = 1) \quad (41) \]

\[ + Pr(Y = 1|X = 0)Pr(X = 0) \quad (42) \]

\[ = (1 - p)Pr(X = 1) + pPr(X = 1) \quad (43) \]
BSC Capacity

\[ I(X; Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x) \quad (39) \]

\[ = H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \leq 1 - H(p) \quad (40) \]

- **When is** \( H(Y) = 1? **Note that**

\[ \Pr(Y = 1) = \Pr(Y = 1|X = 1)\Pr(X = 1) \quad (41) \]

\[ + \Pr(Y = 1|X = 0)\Pr(X = 0) \quad (42) \]

\[ = (1 - p)\Pr(X = 1) + p\Pr(X = 1) \quad (43) \]

\[ = \Pr(X = 1) \quad (44) \]
BSC Capacity

\[ I(X; Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x) \]  

\[ = H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \leq 1 - H(p) \]  

- When is \( H(Y) = 1 \)? Note that

\[ \Pr(Y = 1) = \Pr(Y = 1|X = 1)\Pr(X = 1) \]
\[ = (1 - p)\Pr(X = 1) + p\Pr(X = 1) \]

\[ = \Pr(X = 1) \]

- So \( H(Y) = 1 \) if \( H(X) = 1 \).
**BSC Capacity**

\[
\begin{array}{c}
\begin{array}{c}
0 \\
1
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
0 \\
1
\end{array}
\end{array}
\]

\[I(X;Y) = H(Y) - H(Y|X) = H(Y) - \sum_x p(x)H(Y|X = x)\quad (39)\]

\[= H(Y) - \sum_x p(x)H(p) = H(Y) - H(p) \leq 1 - H(p)\quad (40)\]

- **When is** \(H(Y) = 1\)? Note that

\[
\Pr(Y = 1) = \Pr(Y = 1|X = 1)\Pr(X = 1) + \Pr(Y = 1|X = 0)\Pr(X = 0)\quad (41)
\]

\[
= (1 - p)\Pr(X = 1) + p\Pr(X = 1)\quad (42)
\]

\[
= \Pr(X = 1)\quad (43)
\]

- So \(H(Y) = 1\) if \(H(X) = 1\).

- **Thus, we get that** \(C = 1 - H(p)\) which happens when \(X\) is uniform.
Thus, we get that $C = 1 - H(p)$ which happens when $X$ is uniform.
Thus, we get that $C = 1 - H(p)$ which happens when $X$ is uniform.

If $p = 1/2$ then $C = 0$, so if it randomly flips bits, then no information can be sent.
Thus, we get that $C = 1 - H(p)$ which happens when $X$ is uniform.

If $p = 1/2$ then $C = 0$, so if it randomly flips bits, then no information can be sent.

If $p \neq 1/2$, then we can communicate, albeit potentially slowly. E.g., if $p = 0.499$ then $C = 2.8854 \times 10^{-6}$ bits per channel use. So to send one bit, need to use the channel quite a bit.
Thus, we get that $C = 1 - H(p)$ which happens when $X$ is uniform.

If $p = 1/2$ then $C = 0$, so if it randomly flips bits, then no information can be sent.

If $p \neq 1/2$, then we can communicate, albeit potentially slowly. E.g., if $p = 0.499$ then $C = 2.8854 \times 10^{-6}$ bits per channel use. So to send one bit, need to use the channel quite a bit.

If $p = 0$ or $p = 1$, then $C = 1$ and we can get maximum possible rate (i.e., the capacity is one bit per channel use).
Decoding

Let's temporarily look ahead to address this problem.
Decoding

- Lets temporarily look ahead to address this problem.
- We can “decode” the source using the received string, source distribution, and the channel model $p(y|x)$ via Bayes rule. I.e.

$$
\Pr(x|y) = \frac{\Pr(y|x) \Pr(x)}{\Pr(y)} = \frac{\Pr(y|x) \Pr(x)}{\sum_{x'} \Pr(y|x') \Pr(x')}
$$

(45)
Let’s temporarily look ahead to address this problem.

- We can “decode” the source using the received string, source distribution, and the channel model $p(y|x)$ via Bayes rule. I.e.

$$
\Pr(x|y) = \frac{\Pr(y|x) \Pr(x)}{\Pr(y)} = \frac{\Pr(y|x) \Pr(x)}{\sum_{x'} \Pr(y|x') \Pr(x')}
$$

- If we get a particular $y$, we can compute $p(x|y)$ and make a decision based on that. I.e., $\hat{x} = \arg\max_x p(x|y)$. 
Decoding

- Lets temporarily look ahead to address this problem.
- We can “decode” the source using the received string, source distribution, and the channel model \( p(y|x) \) via Bayes rule. I.e.

\[
\Pr(x|y) = \frac{\Pr(y|x) \Pr(x)}{\Pr(y)} = \frac{\Pr(y|x)\Pr(x)}{\sum_{x'} \Pr(y|x')\Pr(x')}
\]  

(45)

- If we get a particular \( y \), we can compute \( p(x|y) \) and make a decision based on that. I.e., \( \hat{x} = \arg\max_x p(x|y) \).
- This is optimal decoding in that it minimizes the error.
Decoding

- Lets temporarily look ahead to address this problem.
- We can “decode” the source using the received string, source distribution, and the channel model $p(y|x)$ via Bayes rule. I.e.

$$\Pr(x|y) = \frac{\Pr(y|x) \Pr(x)}{\Pr(y)} = \frac{\Pr(y|x)\Pr(x)}{\sum_{x'} \Pr(y|x')\Pr(x')} \quad (45)$$

- If we get a particular $y$, we can compute $p(x|y)$ and make a decision based on that. I.e., $\hat{x} = \arg\max_x p(x|y)$.
- This is optimal decoding in that it minimizes the error.
- Error if $x \neq \hat{x}$, and $\Pr(\text{error}) = \Pr(x \neq \hat{x})$. 
Decoding

- Lets temporarily look ahead to address this problem.
- We can “decode” the source using the received string, source distribution, and the channel model $p(y|x)$ via Bayes rule. I.e.

$$\Pr(x|y) = \frac{\text{channel} \cdot \text{source}}{\Pr(y)} = \frac{\Pr(y|x)\Pr(x)}{\sum_{x'} \Pr(y|x')\Pr(x')}$$  \hspace{1cm} (45)

- If we get a particular $y$, we can compute $p(x|y)$ and make a decision based on that. I.e., $\hat{x} = \arg\max_x p(x|y)$.
- This is optimal decoding in that it minimizes the error.
- Error if $x \neq \hat{x}$, and $\Pr(\text{error}) = \Pr(x \neq \hat{x})$.
- This is minimal if we chose $\arg\max_x p(x|y)$ since the error $1 - \Pr(\hat{x}|y)$ is minimal.
Minimum Error Decoding

- Note: Computing quantities such as \( \text{Pr}(x|y) \) is a task of probabilistic inference.
Minimum Error Decoding

- Note: Computing quantities such as $Pr(x|y)$ is a task of probabilistic inference.

- Often this problem is difficult (NP-hard, see Cooper and Herskovitz, 1990). This means that doing minimum error decoding might very well be exponentially expensive (unless $P = NP$).
Minimum Error Decoding

- Note: Computing quantities such as $\Pr(x|y)$ is a task of probabilistic inference.
- Often this problem is difficult (NP-hard, see Cooper and Herskovitz, 1990). This means that doing minimum error decoding might very well be exponentially expensive (unless $P = NP$).
- Many real world codes are such that computing the exact computation must be approximated (i.e., no known fast algorithm for minimum error or maximum likelihood decoding).
Minimum Error Decoding

- Note: Computing quantities such as $\Pr(x|y)$ is a task of probabilistic inference.
- Often this problem is difficult (NP-hard, see Cooper and Herskovitz, 1990). This means that doing minimum error decoding might very well be exponentially expensive (unless $P = NP$).
- Many real world codes are such that computing the exact computation must be approximated (i.e., no known fast algorithm for minimum error or maximum likelihood decoding).
- Instead we do approximate inference algorithms (e.g., loopy belief propagation, message passing, etc.). These algorithms tend still to work very well in practice (achieve close to the capacity $C$).
Minimum Error Decoding

- Note: Computing quantities such as $\Pr(x|y)$ is a task of probabilistic inference.

- Often this problem is difficult (NP-hard, see Cooper and Herskovitz, 1990). This means that doing minimum error decoding might very well be exponentially expensive (unless $P = NP$).

- Many real world codes are such that computing the exact computation must be approximated (i.e., no known fast algorithm for minimum error or maximum likelihood decoding).

- Instead we do approximate inference algorithms (e.g., loopy belief propagation, message passing, etc.). These algorithms tend still to work very well in practice (achieve close to the capacity $C$).

- But before doing that, we need first to study more channels and the theoretical properties of the capacity $C$. 
$e$ is an erasure symbol, if that happens we don’t have access to the transmitted bit.
Binary Erasure Channel

- $e$ is an erasure symbol, if that happens we don’t have access to the transmitted bit.

- The probability of dropping a bit is then $\alpha$.

\[\begin{array}{|c|}
\hline
X & 0 & 1 - \alpha & 0 \\
\hline
\alpha & e & Y \\
\hline
1 - \alpha & 1 & 1 \\
\hline
\end{array}\]
Binary Erasure Channel

- \( e \) is an erasure symbol, if that happens we don’t have access to the transmitted bit.
- The probability of dropping a bit is then \( \alpha \).
- We want to compute capacity. Obviously, \( C = 1 \) if \( \alpha = 0 \).
$e$ is an erasure symbol, if that happens we don’t have access to the transmitted bit.

- The probability of dropping a bit is then $\alpha$.
- We want to compute capacity. Obviously, $C = 1$ if $\alpha = 0$. 

\[ e \text{ is an erasure symbol, if that happens we don’t have access to the transmitted bit.} \]

\[ \text{The probability of dropping a bit is then } \alpha. \]

\[ \text{We want to compute capacity. Obviously, } C = 1 \text{ if } \alpha = 0. \]
A binary erasure channel is depicted with nodes $X$ and $Y$ and edges labeled $0$, $1$, and $e$. The edge $e$ represents an erasure symbol, which means that if this happens, we don't have access to the transmitted bit.

The probability of dropping a bit is $\alpha$. We want to compute the capacity $C$. Obviously, $C = 1$ if $\alpha = 0$.

The capacity is given by:

$$C = \max_{p(x)} I(X; Y)$$

(47)
$e$ is an erasure symbol, if that happens we don’t have access to the transmitted bit.

The probability of dropping a bit is then $\alpha$.

We want to compute capacity. Obviously, $C = 1$ if $\alpha = 0$.

$$C = \max_{p(x)} I(X; Y) = \max_{p(x)} (H(Y) - H(Y|X))$$  \hspace{1cm} (46)$$

$$\hspace{1cm} (47)$$
Binary Erasure Channel

- $e$ is an erasure symbol, if that happens we don’t have access to the transmitted bit.
- The probability of dropping a bit is then $\alpha$.
- We want to compute capacity. Obviously, $C = 1$ if $\alpha = 0$.

$$C = \max_{p(x)} I(X; Y) = \max_{p(x)} (H(Y) - H(Y|X))$$  \hspace{1cm} (46)$$

$$= \max_{p(x)} H(Y) - H(\alpha)$$  \hspace{1cm} (47)$$
Binary Erasure Channel

\[ e \text{ is an erasure symbol, if that happens we don’t have access to the transmitted bit.} \]

\[ \text{The probability of dropping a bit is then } \alpha. \]

\[ \text{We want to compute capacity. Obviously, } C = 1 \text{ if } \alpha = 0. \]

\[ C = \max_{p(x)} I(X; Y) = \max_{p(x)} (H(Y) - H(Y|X)) \]

\[ = \max_{p(x)} H(Y) - H(\alpha) \]

\[ \text{So while } H(Y) \leq \log 3, \text{ we want actual value of the capacity.} \]
let $E = \{Y = e\}$. Then

$$H(Y) = H(E) + H(Y|E)$$

This last equality follows since $H(E) = H(\alpha)$, and $H(Y|E) = \alpha H(Y|Y = e) + (1 - \alpha) H(Y|Y \neq e) = \alpha \cdot 0 + (1 - \alpha) H(Y|Y = e)$.
Binary Erasure Channel

- let \( E = \{ Y = e \} \). Then

\[
H(Y)
\]
Binary Erasure Channel

Let $E = \{Y = e\}$. Then

$$H(Y) = H(Y, E)$$
Binary Erasure Channel

Let $E = \{Y = e\}$. Then

$$H(Y) = H(Y, E) = H(E) + H(Y|E)$$
let \( E = \{ Y = e \} \). Then

\[
H(Y) = H(Y, E) = H(E) + H(Y | E)
\]
Binary Erasure Channel

- Let $E = \{Y = e\}$. Then

$$H(Y) = H(Y, E) = H(E) + H(Y|E)$$

- Let $\pi = \Pr(X = 1)$. 
**Binary Erasure Channel**

- Let $E = \{Y = e\}$. Then

  $$H(Y) = H(Y, E) = H(E) + H(Y|E)$$

- Let $\pi = \Pr(X = 1)$. Then

  $$H(Y) = H\left((1 - \pi)(1 - \alpha)\right) + H\left(\alpha, \pi(1 - \alpha)\right) \quad \text{(48)}$$

  $$H(Y) = H\left(1 - \pi\right) + H\left(\alpha, \pi\right) \quad \text{(49)}$$
Binary Erasure Channel

- Let $E = \{Y = e\}$. Then
  
  $$H(Y) = H(Y, E) = H(E) + H(Y|E)$$

- Let $\pi = \Pr(X = 1)$. Then
  
  $$H(Y) = H\left((1 - \pi)(1 - \alpha), \frac{\alpha}{\alpha}, \pi(1 - \alpha)\right)$$
  
  $$(48) = H(\alpha) + (1 - \alpha)H(\pi)$$
  
  $$(49)$$
Binary Erasure Channel

- Let $E = \{Y = e\}$. Then
  \[
  H(Y) = H(Y, E) = H(E) + H(Y|E)
  \]

- Let $\pi = \Pr(X = 1)$. Then
  \[
  H(Y) = H\left((1 - \pi)(1 - \alpha), \frac{\alpha}{\pi(1 - \alpha)}\right)
  = H(\alpha) + (1 - \alpha)H(\pi)
  \]

This last equality follows since $H(E) = H(\alpha)$, and
\[
H(Y|E) = \alpha H(Y|Y = e) + (1 - \alpha)H(Y|Y \neq e) = \alpha \cdot 0 + (1 - \alpha)H(\pi)
\]
Then we get

\[ C = \max_{p(x)} H(Y) - H(\alpha) \]  

\[ = \max_{\pi} \left( (1 - \alpha)H(\pi) + H(\alpha) \right) - H(\alpha) \]  

\[ = \max_{\pi} (1 - \alpha)H(\pi) = 1 - \alpha \]
Then we get

\[ C = \max_{p(x)} H(Y) - H(\alpha) \]  
(50)

\[ = \max_{\pi} \left( (1 - \alpha)H(\pi) + H(\alpha) \right) - H(\alpha) \]  
(51)

\[ = \max_{\pi} (1 - \alpha)H(\pi) = 1 - \alpha \]  
(52)

Best capacity when \( \pi = 1/2 = \Pr(X = 1) = \Pr(X = 0) \).
Then we get

\[ C = \max_{p(x)} H(Y) - H(\alpha) \]  

(50)

\[ = \max_\pi \left( (1 - \alpha)H(\pi) + H(\alpha) \right) - H(\alpha) \]  

(51)

\[ = \max_\pi (1 - \alpha)H(\pi) = 1 - \alpha \]  

(52)

Best capacity when \( \pi = 1/2 = \Pr(X = 1) = \Pr(X = 0) \).

This makes sense, loose \( \alpha \)% of the bits of original capacity.
Ternary Confusion Channel

Whenever the symbol '?' is input, the output is random. Other inputs are reliable. Thus, $C = 1$ bit.

- $P(Y = j|X = ?) = 1/2$. 
Ternary Confusion Channel

- $P(Y = j | X = ?) = 1/2$.
- Whenever the symbol “?” is input, the output is random.
Ternary Confusion Channel

\[ P(Y = j | X = ?) = \frac{1}{2}. \]

Whenever the symbol “?” is input, the output is random.

Other inputs are reliable.
Ternary Confusion Channel

Whenever the symbol “?” is input, the output is random.
Other inputs are reliable.
Thus, $C = 1$ bit.