Work on this assignment either alone or in pairs. You may work with different partners on different assignments, but you can only have up to one partner for any one assignment. You may not talk about this assignment with others until all of you have handed in their work. See Mechanics → Homework on the class web page for details on the homework policy.

This assignment comes with auxiliary file `hw.tex`. Remember to put your name in the right place in this file before starting to work with it. Usual formatting instructions hold.

1. Below are shown some conditional, joint, and marginal probability functions for the compound experiment with outcomes $(X, Y)$. Outcome $X$ has sample space $S_X = \{1, 2, 3\}$ and outcome $Y$ has sample space $S_Y = \{0, 1\}$, so the compound experiment has sample space $S = S_X \times S_Y$. Fill the empty cells in the tables below with exact values. Represent values as irreducible fractions of integers (e.g., $2/3$ but not $4/6$) or as decimals (e.g., $0.40$). The decimal representation of $2/3$ is $0.666\ldots$. It is OK to mix and match representations as convenient.

   Recall that $P_{XY}(x, y) = \text{prob}((x, y))$, the probability that $X$ takes value $x$ and $Y$ takes value $y$. Then,

   $P_X(x) = \sum_{y\in S_Y} P_{XY}(x, y)$

   $P_Y(y) = \sum_{x\in S_X} P_{XY}(x, y)$

   $P_{Y|X}(y|x) = \frac{P_{XY}(x, y)}{P_X(x)}$

   $P_{X|Y}(x|y) = \frac{P_{XY}(x, y)}{P_Y(y)}$.

| $P_{X|Y}(x|y)$ | $x = 1$ | 2 | 3 |
|----------------|--------|---|---|
| $y = 0$        |        | 0.40 | 0.40 |
| 1              | 0.40   | 0.60 |

<table>
<thead>
<tr>
<th>$P_Y(y)$</th>
<th>$y = 0$</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.40</td>
<td>0.60</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P_{XY}(x, y)$</th>
<th>$x = 1$</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y = 0$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| $P_{Y|X}(y|x)$ | $x = 1$ | 2 | 3 |
|----------------|--------|---|---|
| $y = 0$        |        |   |   |
| 1              |        |   |   |

<table>
<thead>
<tr>
<th>$P_X(x)$</th>
<th>$x = 1$</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Are the two outcomes $U$ and $V$ whose joint probability $p_{UV}(u, v)$ is shown in the table below independent? Why or why not? Credit for your answer will decrease with the amount of computation used or implied.

<table>
<thead>
<tr>
<th>$p_{UV}(u, v)$</th>
<th>$u = 1$</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v = 0$</td>
<td>0.12</td>
<td>0.42</td>
</tr>
<tr>
<td>1</td>
<td>0.18</td>
<td>0.28</td>
</tr>
</tbody>
</table>

3. In a certain game, you throw two fair and independent dice once. If the outcome $D$ of this compound experiment is $(m, n)$, then you win $mn$ dollars if $mn$ is even, and you lose $mn$ dollars if $mn$ is odd. What is your expected gain or loss? Show your reasoning. If you write code to save time in your calculations (highly recommended!), do not show your code.

[Assignment continues on the next page]
Error-Correcting Codes. Whether you text a friend or download a picture from the web, digital information travels over communication channels as sequences of bits called messages (regardless of whether they represent a text message or something else). A communication channel can be described abstractly as a noisy transformation. Specifically, the channel takes a bit (a 0 or a 1) as input and returns a bit as output. Most of the time, the output bit is the same as the input bit, but not always, since noise on the channel can cause transmission errors. To model these errors we define a compound experiment with sample space

\[ \mathbb{S} = \mathbb{S}_T \times \mathbb{S}_R \quad \text{where} \quad \mathbb{S}_T = \mathbb{S}_R = \{0, 1\}. \]

Outcome \((t, r) \in \mathbb{S}\) occurs when bit \(t\) is transmitted \((T = t)\) and bit \(r\) is received \((R = r)\). We assume that transmitted zeros and ones are equally likely, so that the marginal probabilities of \(T\) are

\[ P_T(0) = P_T(1) = \frac{1}{2}. \]

We also assume that errors are symmetric, in the sense that a transmitted 0 is transformed into a received 1 with the same probability \(\epsilon\) that a transmitted 1 is transformed into a received 0. We collect the resulting conditional probabilities into a table as follows:

<table>
<thead>
<tr>
<th>(t)</th>
<th>(r = 0)</th>
<th>(r = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(1 - \epsilon)</td>
<td>(\epsilon)</td>
</tr>
<tr>
<td>1</td>
<td>(\epsilon)</td>
<td>(1 - \epsilon)</td>
</tr>
</tbody>
</table>

So both symbols 0 and 1 are transmitted correctly with probability \(1 - \epsilon\) and flipped during transmission with probability \(\epsilon\).

Finally, we assume that bits are both transmitted and perturbed by noise independently of each other. This means that if we send a sequence \(t = (t_1, \ldots, t_s)\) of \(s\) bits over the channel, all probabilities factor into single-bit probabilities:

\[
P_{T_{1,\ldots,1}}(t_1, \ldots, t_s) = P_T(t_1) \cdots P_T(t_s) \\
P_{R_{1,\ldots,1}}(r_1, \ldots, r_s | t_1, \ldots, t_s) = P_R(r_1 | t_1) \cdots P_R(r_s | t_s). 
\]

The channel bit error rate \(\epsilon\) is typically very small, so channel errors are infrequent. However, errors can cause text messages or pictures to be mangled, and even a small \(\epsilon\) is often unacceptably large. Therefore, the bit sequences are typically encoded with so-called error-correcting codes (ECCs) before transmission, and then decoded at the receiver. An ECC replaces a small block of consecutive bits in the message into a larger block of consecutive bits that are then sent over the channel. At the receiver, the resulting redundancy is used somehow to check if an error has occurred and, if so, to correct it, without having to re-transmit the sequence. The cascade of these transformations is illustrated in Figure 1. This problem compares two simple ECCs.

![Figure 1: A message is a sequence of bits divided into blocks \(m_1, m_2, \ldots, m_m\) of \(m\) bits each. At the transmitter, an encoder replaces each block \(m_i\) with a bigger block \(t_i\) of \(n\) code bits (with \(n > m\)), which is then transmitted over the noisy channel. The receiver receives a block \(r_i\) of \(n\) bits that are potentially different from those in \(t_i\) because of channel noise. A decoder then transforms \(r_i\) into a decoded message block \(d_i\) of \(m\) bits. The redundancy purchased at the cost of transmitting \(n\) bits instead of \(m\) in each block is used to make the end-to-end bit error rate \(\delta\) between message and decoded message much smaller than the channel bit error rate \(\epsilon\). This means that the probability \(\delta\) that each bit in \(d_i\) is different from the corresponding bit in \(m_i\) is much smaller than the channel bit error rate \(\epsilon\). The bit strings in the figure exemplify a \((7, 4)\) Hamming code, described in one of the problems below.

(a) Compute the joint probabilities \(P_{TR}(t, r)\) that symbol \(t\) is transmitted and symbol \(r\) is received, for all \(t \in \mathbb{S}_T\) and \(r \in \mathbb{S}_R\) and as functions of \(\epsilon\). Show your calculations.

(b) Compute the marginal probabilities \(P_R(r)\) that symbol \(r\) is received, for all \(r \in \mathbb{S}_R\). Show your calculations.

[Assignment continues on the next page]
The ECC that is simplest to understand is the *k*-repetition code, in which each message block \( m_i \) is a single bit and the corresponding code block \( t_i \) merely repeats that bit \( k \) times. For instance, for \( k = 3 \), the encoder implements the function represented by the following table:

<table>
<thead>
<tr>
<th>( m_i )</th>
<th>( t_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td>1</td>
<td>111</td>
</tr>
</tbody>
</table>

So if the message is \( m = 0010110 \), the transmitted message is\(^1\) \( t = 000 \ 000 \ 111 \ 000 \ 111 \ 111 \ 000 \). The received message may be different, since noise in the channel flips bits with probability \( \epsilon \). For instance, we may have

\[
r = 000 \ 001 \ 111 \ 000 \ 010 \ 111 \ 000
\]

where errors, in red, occurred in the second and fifth block.

It turns out that the optimal decoder takes a majority vote among the three bits in each block \( r_i \) of the received message \( r \) to produce the decoded blocks \( d_i \) in \( d \), according to the following table:

<table>
<thead>
<tr>
<th>( r_i )</th>
<th>( d_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
</tr>
<tr>
<td>011</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
</tr>
<tr>
<td>110</td>
<td>1</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
</tr>
</tbody>
</table>

With this rule, the decoded message in the example reads 0010001. In this example, the majority rule of the ECC managed to correct the lone channel error in the second block of \( r \) (blue). However, two channel errors occurred in the fifth block, and this overwhelmed the coding/decoding scheme, and resulted in an end-to-end error (red). The idea of repetition coding is that multiple channel errors in a block are much less frequent than single channel errors.

(a) Write a formula for the probability \( p(\epsilon, j, k) \) that exactly \( j \) channel errors occur in a block of \( k \) bits if the channel bit error rate is \( \epsilon \), for \( j \) between 0 and \( k \). Explain briefly. [Hint: Choosing which \( j \) bits gives a number of mutually exclusive outcomes. For each of these, the \( j \) chosen bits must be wrong and the other \( k - j \) must be right.]

(b) Write a formula for the probability \( p_+(\epsilon, \ell, k) \) that at least \( \ell \) channel errors occur in a block of \( k \) bits if the channel bit error rate is \( \epsilon \), for \( \ell \) between 0 and \( k \).

(c) Write a formula for the smallest number \( \ell_{\text{min}} \) of channel errors (individual bits flipped by the channel) that must occur in the same block for an end-to-end error to occur with a \( k \)-repetition code with \( k \) odd and greater than 1. Explain briefly.

(d) If the channel bit error rate is \( \epsilon = 10^{-4} \), what is the smallest odd repetition factor \( k \) needed in a \( k \)-repetition code to achieve an end-to-end bit error rate that is no worse than \( \delta = 10^{-12} \)? You may need to try various values of \( k \) to answer this question, using either Python or a calculator for your calculations. Explain briefly, but do not show your Python code, if any.

(e) For a channel with end-to-end bit error rate \( \delta \), give a formula for the probability \( p_b(\delta, s) \) that \( s \) bits are transmitted with no end-to-end error. Your formula should contain no summation over a variable number of terms (so nothing like \( \sum \ldots \)).

(f) What value of end-to-end bit error rate \( \delta \) do you need to transmit a 1 megabyte image with at least 99.9 percent probability that the whole image is transferred error-free?

(g) If the channel bit error rate \( \epsilon \) is \( 10^{-4} \), what is the smallest odd value of \( k \) that you need for a \( k \)-repetition ECC to transmit a 1 megabyte image with at least 99.9 percent probability that the whole image is transferred error-free? Explain briefly.

\[^1\]Spaces are inserted between blocks only for readability in this example. The transmitter has no way to send spaces.
The cost of a \( k \)-repetition ECC is high, as the transmitted sequence is much longer than the input message. If Verizon were to use this scheme, it would need to build a communication infrastructure with a bandwidth that is \( k \) times as high as the bandwidth its users see end-to-end. Hamming codes add redundancy to blocks of consecutive bits, rather than to individual bits, and can achieve end-to-end performance comparable to that of \( k \)-repetition ECCs at lower bandwidth costs. This problem looks at a particular Hamming code, the \((7, 4)\) Hamming code.

This encoder splits the input message \( m \) into blocks \( m_i \) of 4 bits each, and replaces each block with a code block \( t_i \) of 7 bits as follows:

- The first four bits in \( t_i \) are equal to the bits in \( m_i \):
  \[
  (t_1, t_2, t_3, t_4) = (m_1, m_2, m_3, m_4)
  \]
  (the subscript \( i \) is omitted from the bits for simplicity)

- The remaining three bits of \( t_i \) are the parity bits of groups of three bits out of \( m_i \):
  \[
  \begin{align*}
  t_5 &= t_1 \oplus t_2 \oplus t_3 \\
  t_6 &= t_2 \oplus t_3 \oplus t_4 \\
  t_7 &= t_1 \oplus t_3 \oplus t_4
  \end{align*}
  \]
  where the symbol ‘\( \oplus \)’ is addition modulo 2:
  \[
  x \oplus y = (x + y) \mod 2.
  \]

Thus, the complete encoding table is as follows:

<table>
<thead>
<tr>
<th>( m_i )</th>
<th>( t_i )</th>
<th>( m_i )</th>
<th>( t_i )</th>
<th>( m_i )</th>
<th>( t_i )</th>
<th>( m_i )</th>
<th>( t_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000000</td>
<td>0100</td>
<td>0100110</td>
<td>1000</td>
<td>1000101</td>
<td>1100</td>
<td>1100011</td>
</tr>
<tr>
<td>0001</td>
<td>0001011</td>
<td>0101</td>
<td>0101101</td>
<td>1001</td>
<td>1001110</td>
<td>1101</td>
<td>1101000</td>
</tr>
<tr>
<td>0010</td>
<td>0010111</td>
<td>0110</td>
<td>0110001</td>
<td>1010</td>
<td>1010010</td>
<td>1110</td>
<td>1110100</td>
</tr>
<tr>
<td>0011</td>
<td>0011100</td>
<td>0111</td>
<td>0111010</td>
<td>1011</td>
<td>1011001</td>
<td>1111</td>
<td>1111111</td>
</tr>
</tbody>
</table>

The received 7-bit block \( r_i \) is not necessarily one of the 16 blocks \( t_i \) in this table, because of possible channel errors. The optimal decoder turns out to be the one that chooses the row in the table above whose \( t_i \) is closest to \( r_i \) as measured by the Hamming distance, and makes \( d_i \) the corresponding \( m_i \) in the table.

The Hamming distance is the number of bits in which two blocks differ from each other. For instance, the Hamming distance between \( 0110001 \) and \( 0010101 \) is two, because these two blocks differ in the second and fifth position.

Writing the entire decoding table is unwieldy, because it would have \( 2^7 = 128 \) rows. Instead, one could compute the Hamming distance between the received block \( r_i \) and each \( t_i \) block in the table above, and then pick the row with the closest \( t_i \). For instance, if the received block is \( 0110101 \), the Hamming distances \( HD \) with the \( t_i \) blocks are as follows:

<table>
<thead>
<tr>
<th>( t_i )</th>
<th>( HD )</th>
<th>( d_i )</th>
<th>( t_i )</th>
<th>( HD )</th>
<th>( d_i )</th>
<th>( t_i )</th>
<th>( HD )</th>
<th>( d_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000000</td>
<td>4</td>
<td>0000</td>
<td>0100110</td>
<td>3</td>
<td>0100</td>
<td>1000101</td>
<td>3</td>
<td>1000</td>
</tr>
<tr>
<td>0001011</td>
<td>5</td>
<td>0001</td>
<td>0101101</td>
<td>2</td>
<td>0101</td>
<td>1001110</td>
<td>6</td>
<td>1001</td>
</tr>
<tr>
<td>0010111</td>
<td>2</td>
<td>0010</td>
<td>0110001</td>
<td>1</td>
<td>0110</td>
<td>1010010</td>
<td>5</td>
<td>1010</td>
</tr>
<tr>
<td>0011100</td>
<td>3</td>
<td>0011</td>
<td>0111010</td>
<td>4</td>
<td>0111</td>
<td>1011001</td>
<td>4</td>
<td>1011</td>
</tr>
</tbody>
</table>

The row in red has the smallest Hamming distance (1), so the decoder returns \( 0110 \) as the decoded block \( d_i \). This is a slow decoding method, and there turns out to be a much faster alternative, which is beyond the scope of this assignment.

[Assignment continues on the next page]
(a) Give the blocks $d_1$, $d_2$, $d_3$ that result from decoding the received blocks $r_1 = 1101011$, $r_2 = 0110110$, $r_3 = 0100111$.

[Hint: If you are impatient, it may be faster to write Python code to answer this question. Either way, just give your answers, don’t show how you determined them.]

(b) The $(7, 4)$ Hamming code is designed so that the smallest Hamming distance between any two distinct transmitted codes $t_i$ is 3. Because of this, the code will detect and correct any error in which one bit in a received block $r_i$ is wrong. If there are two or more wrong bits, the decoded message $d_i$ is different from the original message $m_i$, and a block error is said to occur. Thus, this code works well when the probability of multiple errors in a block is very small.

Give a formula for the probability $p_B(\epsilon)$ that a block error occurs with a $(7, 4)$ Hamming code used over a channel with channel bit error rate $\epsilon$. Explain your reasoning.

(c) If the channel bit error rate $\epsilon$ is small, we can approximate its square $\epsilon^2$ with zero. Under this approximation, it can be shown that the probability that any one bit in the four-bit decoded block $d_i$ is wrong is

$$p_b(\epsilon) = \frac{3}{7} p_B(\epsilon).$$

Use this fact and your formula for $p_B(\epsilon)$ to compute the numerical value of the approximate end-to-end bit error rate over a channel with channel bit error rate $\epsilon = 10^{-4}$ that uses a $(7, 4)$ Hamming code. [Hint to save you effort only: Since $\epsilon$ is small, if you add several powers of $\epsilon$, higher powers are negligible relative to the lowest one. For the same reason, $1 - \epsilon$ raised to some reasonably small power (say less than 30) is close to 1.]

(d) How much more bandwidth-efficient is the $(7, 4)$ Hamming code compared to the 3-repetition code? In other words, what is the ratio $\rho$ between the bandwidth required by the 3-repetition code and that required by the $(7, 4)$ Hamming code to transmit the same message? Explain briefly.

[Assignment continues on the next page]
7. An insurance company wants to study the relationship between the seriousness of injuries in car accidents and the type of safety restraint worn by the driver, when no other passengers are in the car. To this end, the company uses police accident reports from single-passenger accidents to compile the following contingency table:

\[
\begin{array}{c|ccc}
N_{SR}(s, r) & r = n & b & h \\
\hline
s = n & 39 & 45 & 36 \\
\quad l & 105 & 96 & 69 \\
\quad m & 81 & 60 & 39 \\
\quad d & 15 & 9 & 6 \\
\end{array}
\]

The sample space for the accident seriousness \( S \) is \( \mathbb{S}_S = \{n, l, m, d\} \) for no injury, light injury, major injury, and death. The sample space for the type of restraint \( R \) is \( \mathbb{S}_R = \{n, b, h\} \) for no restraint, seat belt, and seat harness. Belts and harnesses are completely different devices (so if you are wearing a harness you are not wearing a belt, and vice versa). It may be easiest to write Python code to answer the following questions, but that is up to you. Do not hand in your code. Report all probabilities rounded to three decimal digits, even when some of the trailing digits are zeros. When rounding a trailing 5 up or down, adjust the last digit (arbitrarily) so that table rows and/or columns add up to 1 as appropriate.

(a) Make a table for the joint probability \( p_{SR}(s, r) \) estimated from the contingency table.

(b) Make tables for the marginal probabilities \( p_S(s) \) and \( p_R(r) \).

(c) Make tables for the conditional probabilities \( p_{S|R}(s \mid r) \) and \( p_{R|S}(r \mid s) \).

(d) What is the probability that a person selected at random from those who were involved in an accident was wearing some restraint and survived with no major injury? Give your answer as a full sentence with the probability in the form of a percentage. Show your calculations.

(e) What is the probability that a person selected at random from those who were wearing no restraint during an accident suffered a major or fatal injury? Give your answer as a full sentence with the probability in the form of a percentage. Show your calculations.

(f) What is the probability that a person selected at random from those who were wearing some restraint during an accident survived with no major injury? Give your answer as a full sentence with the probability in the form of a percentage. Show your calculations.

(g) The average cost of an accident is $2,000 if no injury is involved. For light injuries, the cost is $5,000, and for major injuries it is $50,000. If a death is involved, then the cost is $500,000. However, the insurance company pays only 70 percent of the cost if the driver was wearing no restraint, 90 percent if the driver was wearing a seat belt, and the full cost if the driver was wearing a seat harness. What is the expected cost per accident for the insurance company? Show your reasoning, but do not give your code if you write code to save time.