Info Theory 1 Notes

Grant Yang and Rohan Ramkumar

January 13, 2025

Contents

1 Counting & Probability

1.1 Intro & Counting (2024.08.19)

$$\begin{tikzcd}
\boxed{\text{Info Src}} \arrow[r] & \boxed{\text{Transmit}} \arrow[r, "\text{Channel}", "\text{(+ Noise)}"'] & \boxed{\text{Receive}} \arrow[r] & \boxed{\text{Dest}}
\end{tikzcd}$$

- Infor**m**ation sources can be m**o**deled as random variables, probability distributions (LLMs and Chat**G**PT model language this way).
- **Fundamental Counting Principle** (for consec**u**tive choices): If one event can occur in n ways and a 2nd event can happen in m ways after event one, there are $m \times n$ total possibilitie**s**.
- **Subsets of a set**: A set of n elements contains 2^n subsets.
  - For each element of a set, it can either be in the subset or not. 2 possibilities for each of the n elements results in 2^n possibilities.
- **Permutations**: There are $n!$ permutations of a set of n elements.
- **Distinguishable Permutations**: Suppose a set of n objects has k distinguishable kinds of objects (e.g. a set of 4 As, 2 Bs, etc.), such that there are $n_1, n_2, \dots n_k$ items in each of the k groups, respectively, with $n_1 + n_2 + \cdots + n_k = n$ Then the number of ways to arrange the objects is
$$\frac{n!}{n_1!n_2!\cdots n_k!}$$
- $$P(n,r) = \frac{n!}{(n-r)!}$$
- $$C(n,k) = \binom{n}{k} = \frac{n!}{k!(n-k)!}$$
- Additional reading: Stirling Numbers of the Second Kind

1.2 Probability (2024.08.21)

Sample Space: Set of all possible outcomes called $\mathcal{S}$.

Axioms: Let E be the event of interest.
The complement (the event of E *not* happening) is denoted E^c

$$0 \leq P(E) \leq 1$$
$$P(\mathcal{S}) = 1$$

Probability of E_1 or E_2: $P(E_1 \cup E_2) = P(E_1) + P(E_2) - P(E_1 \cap E_2)$

Mutually exclusive: $E_1 \cap E_2 = \emptyset$.
Not to be confused with "independent": If events are mutually exclusive, they are completely dependent on each other (E_1 happening means E_2 did not happen).
If E_1 and E_2 are independent, then $P(E_1 \cap E_2) = P(E_1) \cdot P(E_2)$

1.2.1 Exercises

Exercise 1: 3 balls are drawn from a bowl of 3 white, 5 black without replacement. What is the probability that 1 ball is white and 2 balls are black?

$$3\left[\frac{6 \cdot 5 \cdot 4}{11 \cdot 10 \cdot 9}\right] = \frac{C(6,1)C(5,2)}{C(11,3)}$$

Exercise 3: 5-card hand in poker, P(straight)?

$$\frac{10\,(4^5 - 4)}{C(52,5)}$$

1.3 Discrete Random Variables (2024.08.23)

1.3.1 Random Variable:

Mapping from sample space $\mathcal{S}$ to real numbers $\mathbb{R}$. Denoted by later capital letters.

Exercise 1: Toss 3 fair coins, then Y: # of heads

y	$P(Y = y)$
0	1/8
1	3/8
2	3/8
3	1/8

Describes a discrete **Probability Mass Function** (vs a continuous **Probability Density Function, PDF**).

Exercise 2: Choose 3 balls randomly selected without replacement from an urn of 20 balls, numbered in order.

$$P(X = i) = \binom{i-1}{2} \Big/ \binom{20}{3}$$

1.3.2 Cumulative Distribution Functions (CDFs)

Let F be a (discrete) cumulative distribution function and P be the probability mass function. F approaches 1 as $a \to \infty$.

$$F(a) = \sum_{x \leq a} P(x)$$

1.3.3 Expected Value

The expected value of a random variable is given by:

$$E[X] = \sum_{x \in \mathcal{S}} xP(X = x)$$

Exercise 4: For a fair die,

$$E[X] = \sum_{i=1}^{6} i\frac{1}{6} = 3.5$$

Notice that $E[X] \notin \mathcal{S}$.

Exercise 5: Let A be an event with probability $P[A]$. Let

$$I = \begin{cases} 1 & A \text{ occurs} \\ 0 & A^c \text{ occurs.} \end{cases}$$

Then, $E[I] = P[A]$.

x	$P(X = x)$
-1	0.2
0	0.5
1	0.3.

Notice: $E[X^2] = 0.5$. However, $E[X]^2 = 0.01$

1.3.4 Functions of Random Variables:

If $g(X)$ is a function of the random variable, then

$$E[g(X)] = \sum_{x \in \mathcal{S}} g(x)P(X = x).$$

Proof: Let $\mathcal{G}$ be the image (i.e. range) of $g(X)$. Then,

$$\begin{aligned} E[g(X)] &= \sum_{y\in\mathcal{G}} \sum_{x\in\mathcal{S}, g(x)=y} g(x)P(X=x) \\ &= \sum_{y\in\mathcal{G}} y \sum_{x\in\mathcal{S}, g(x)=y} P(X=x) \\ &= \sum_{y\in\mathcal{G}} yP(g(x)=y). \end{aligned}$$

Thus, we have the **Linearity of Expectation**

$$E[aX+b] = aE[X]+b$$

Variance:

$$\mathrm{Var}[X] = E\left[(X-\mu)^2\right],$$

where $\mu = E[X]$. As a result, by expanding and the linearity of expectation,

$$\mathrm{Var}[X] = E[X^2] - E[X]^2.$$

$$\mathrm{Var}[aX+b] = a^2 \cdot \mathrm{Var}[X].$$

Also, stdev=$\sqrt{\mathrm{Var}}$

1.4 Discrete Random Variables 2 (2024.08.27)

- **Bernoulli Random Variable**: 1 with probability p and 0 with probability $1-p$.

$$E[X] = p \qquad \mathrm{Var}[X] = E[X^2] - E[X]^2 = p - p^2$$

- **Binomial Rand Var**: Sum of n independent Bernoulli random variables, each with probability p of being 1. In other words, X is the number of successes in the n trials.

$$P(X=k) = \binom{n}{k} p^k (1-p)^{n-k}$$

$$E[X] = np \qquad \mathrm{Var}[x] = np(1-p)$$

When $p = 0.5$, it is symmetrical about the halfway point: $P(X=k) = P(X=n-k)$

Exercise 1: A player bets on a number from 1-6 and 3 dice are rolled. If the number betted on appears i times for $i \in \{1,2,3\}$, then they win i units, otherwise losing 1 unit.

$$\begin{aligned} E[X] = &-1\cdot\binom{3}{0}\left(\frac{1}{6}\right)^0\left(\frac{5}{6}\right)^3 \\ &+1\cdot\binom{3}{1}\left(\frac{1}{6}\right)\left(\frac{5}{6}\right)^2 + 2\cdot\binom{3}{2}\left(\frac{1}{6}\right)^2\left(\frac{5}{6}\right) + 3\cdot\binom{3}{3}\left(\frac{1}{6}\right)^3\left(\frac{5}{6}\right)^0 \\ = &-\frac{17}{216} < 0. \end{aligned}$$

So it is unfair.

Exercise 2: An airplane engine fails in flight with a probability of $1-p$. If at least 50% of engines work, then the airplane is successful. For what values of p is a 4-engine plane better than a 2-engine plane?

$$\begin{aligned}
P[X_4 < 2] &= \binom{4}{0}(1-p)^4 + \binom{4}{1}p(1-p)^3 \\
P[X_2 < 1] &= \binom{2}{0}(1-p)^2.
\end{aligned}$$

Then,

$$\begin{aligned}
P[X_4 < 2] - P[X_2 < 1] &= (1-p)^2((1-p)^2 + 4p(1-p) - 1) \\
&= (1-p)^2(p^2 - 2p + 1 - 4p^2 + 4p - 1) \\
&= (1-p)^2(-3p^2 + 2p) \\
&= p(p-1)^2(-3p+2).
\end{aligned}$$

So, $P[X_4 \text{ fails}] > P[X_2 \text{ fails}]$ when $p < \frac{2}{3}$, in which case the 2-engine plane is better. Otherwise, when $p > \frac{2}{3}$, a 4-engine plane is better.

Exercise 3: Let X be a binomial random variable. Then, what are $E[X]$ and $\operatorname{Var}[X]$?

$$\begin{aligned}
E[X] &= \sum_{k=1}^{n} E[X_k] \\
&= \sum_{k=1}^{n} p \\
&= np,
\end{aligned}$$

where X_k is Bernoulli. Also, for independent events $E[XY] = E[X]E[Y]$, so:

$$\begin{aligned}
\operatorname{Var}[X+Y] &= E[(X+Y)^2] - E[X+Y]^2 \\
&= E[X^2] + E[Y^2] + 2E[XY] - E[X]^2 - E[Y]^2 - 2E[X]E[Y] \\
&= \operatorname{Var}[X] + \operatorname{Var}[Y] + 2(E[XY] - E[X]E[Y]) \\
&= \operatorname{Var}[X] + \operatorname{Var}[Y],
\end{aligned}$$

so

$$\begin{aligned}
\operatorname{Var}[X] &= \sum_{k=1}^{n} \operatorname{Var}[X_k] \\
&= \sum_{k=1}^{n} p(1-p) \\
&= np(1-p).
\end{aligned}$$

1.5 More Random Vars (2024.08.29)

1.5.1 Poisson Random Variables

$$P(X = k) = e^{-\lambda}\frac{\lambda^k}{k!} \qquad k \in \mathbb{Z}_{\geq 0}, \lambda \in \mathbb{R}$$

Models the probability an event happens X times if it has a probability p of happening every time interval, over the time period of n intervals in the continuous limit as $n \to \infty$ and $\lambda = np$ is a sensible rate. That is, the continuous process where the event has some infinitesimally small probability of happening every instant, yielding a reasonable overall rate of the event happening.

Examples `https://en.wikipedia.org/wiki/Poisson_point_process`. The number of misprints in a page of a book, the number of people entering a queue in a fixed period of time. This distribution can be constructed from binomial distributions:

$$\begin{aligned} &X: \text{ Binomial rand var } (n, p) \\ &\lambda = np \qquad (n \to \infty, p \to 0) \end{aligned}$$

We have

$$\begin{aligned} P(X = k) &= \frac{n!}{(n-k)!k!}p^k(1-p)^{n-k} \\ &= \frac{n!}{(n-k)!k!}\left(\frac{\lambda}{n}\right)^k\left(1-\frac{\lambda}{n}\right)^{n-k} \\ &= \frac{n(n-1)\cdots(n-k+1)}{n^k}\frac{\lambda^k}{k!}\frac{\left(1-\frac{\lambda}{n}\right)^n}{\left(1-\frac{\lambda}{n}\right)^k}. \end{aligned}$$

Applying the facts that:

$$\lim_{n\to\infty}\left(1-\frac{\lambda}{n}\right)^n = e^{-\lambda},$$

$$\lim_{n\to\infty}\left(1-\frac{\lambda}{n}\right)^k = 1,$$

$$\lim_{n\to\infty}\frac{n(n-1)\cdots(n-k+1)}{n^k} = 1,$$

we have

$$P(X = k) \to e^{-\lambda}\frac{\lambda^k}{k!}.$$

Let an item be defective with probability 0.1. Then, sample 10 items and calculate $P(\text{at most 1 defective item})$. Binomial:

$$\binom{10}{0}(1-p)^{10} + \binom{10}{1}p(1-p)^9 \approx 0.7361.$$

Poisson: $\lambda = 10 \cdot 0.1 = 1$,
$$\frac{e^{-1} \cdot 1^0}{0!} + \frac{e^{-1} \cdot 1^1}{1!} \approx 0.7358.$$

$$\begin{aligned}
E[X] &= \sum_{k=0}^{\infty} \lambda e^{-\lambda} \frac{\lambda^k}{k!} \\
&= \lambda e^{-\lambda} \sum_{k=0}^{\infty} \frac{\lambda^k}{k!} \\
&= \lambda e^{-\lambda} e^{\lambda} \\
&= \lambda.
\end{aligned}$$

Also,
$$\mathrm{Var}[X] = E[X^2] - E[X]^2 = \lambda.$$
(do the math yourself)

Poisson variables can model the probability of an event happening k times at a constant rate of λ in a continuous interval of time/space. In this case, sometimes a Poisson R.V. is written as
$$e^{-\lambda t} \frac{(\lambda t)^k}{k!},$$
where t is the time/space interval.

Exercise: Let an earthquake happen at an average rate of
$$\lambda = \frac{2}{\text{week}}.$$

Then,
$$P(\text{at least 3 earthquakes in 2 weeks}) = 1 - (P(0) + P(1) + P(2)).$$

We have:
$$P(0) = e^{-\lambda t} = e^{-4},$$
$$P(1) = e^{-\lambda t} \frac{\lambda t}{1!} = 4e^{-4},$$
and
$$P(2) = e^{-\lambda t} \frac{(\lambda t)^2}{2!} = 8e^{-4}.$$

1.5.2 Geometric Random Variables

Given p probability of success, X is the number of trials until the first success.

Then,
$$P(X = k) = (1 - p)^{k-1} p, k \in \mathbb{Z}_{>0}.$$

And,

$$
\begin{aligned}
E[X] &= \sum_{k=1}^{\infty} k(1-p)^{k-1}p \\
&= \frac{p}{1-p}\sum_{k=1}^{\infty} k(1-p)^{k} \\
&= \frac{p}{1-p}\frac{1-p}{p^2} \\
&= \frac{1}{p}.
\end{aligned}
$$

1.6 Continuous Random Variables (2024.09.03)

1.6.1 Summary of Discrete R.V.

Know Expected Value and Variance:

- Bernoulli (1 with probability p, 0 otherwise)

$$X = \begin{cases} 0 & (1-p) \\ 1 & p, \end{cases}$$

$$E[X] = p \qquad \mathrm{Var}[X] = (1-p)p$$

- Binomial (sum of n Bernoulli)

$$X = \sum_{n} X_{\mathrm{Bernoulli}(p)}$$

$$P(X=k) = \binom{n}{k} p^k (1-p)^{n-k}$$

$$E[X] = np \qquad \mathrm{Var}[X] = n(1-p)p$$

- Poisson (# of occurrences with infinitesimal probability of happening every instant)

$$X = \lim_{n\to\infty} X_{\mathrm{Binomial}(n, p=\frac{\lambda}{n})}$$

$$P(X=k) = \frac{e^{-\lambda}(\lambda)^k}{k!} = \frac{e^{-\lambda t}(\lambda t)^k}{k!}$$

$$E[X] = \lambda \qquad \mathrm{Var}[X] = \lambda$$

- Geometric

$$P(X=k) = (1-p)^{k-1}p$$

$$E[X] = \frac{1}{p} \qquad \mathrm{Var}[X] = \frac{1-p}{p^2}$$

Extra ones for fun:

- Negative Binomial Random Variables

  How many trials necessary to get k successes?

  $$P(X = n) = \binom{n-1}{k-1} p^k (1-p)^{n-k}, \qquad k \in \{r, r+1, \dots\}.$$

  Geometric R.V. is case where $k = 1$.

- Hypergeometric Random Variable: m white balls, $N - m$ black. Number of white balls out of n chosen.

  $$P(X = k) = \frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}}, \qquad k \in \{0, 1, \dots, n\}.$$

  As $N \to \infty$, approaches Binomial/Poisson

- look at Zeta distribution

1.6.2 Continuous Random Variables

$$P(a \le X \le b) = \int_a^b f(x)\,\mathrm{d}x,$$

where PDF(x) is called the probability density function (pdf). Also,

$$\int_{\mathcal{S}} f(x)\mathrm{d}x = \text{usually} \int_{-\infty}^{\infty} f(x)\,\mathrm{d}(x) = 1.$$

is this a lebesgue integral lol

Importantly, the height of a pdf isn't a real probability since there are infinite values (and thus a single value has a probability of zero in some sense), so $P(X = k)$ is nonsensical (or the probability is 0, but the event can still occur). What you can measure is the probability that X is in a specific region.

Exercise 1

$$f(x) = \begin{cases} C(4x - 2x^2) & 0 < x < 2 \\ 0 & \text{otherwise.} \end{cases}$$

The total probability is

$$1 = \int_0^2 C(4x - 2x^2)\mathrm{d}x = C\left(2x^2 - \frac{2}{3}x^3\right)\Big|_0^2 = C\left(8 - \frac{16}{3}\right),$$

so

$$C = \frac{3}{8}.$$

Then,

$$P(X > 1) = \int_1^2 f(x)\mathrm{d}x = \frac{3}{8}\int_1^2 f(x)\mathrm{d}x = \frac{1}{2}.$$

Exercise 2 The amount of time in hours that a computer functions before it breaks down is given by

$$f(x) = \begin{cases} \lambda e^{-x/100} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

$$\int_{\mathcal{S}} f(x)\,\mathrm{d}x = 1$$

$$\therefore \lambda = 1/100$$

$$P(50 < X < 150) = \int_{50}^{150} f(x)\,\mathrm{d}x = e^{-1/2} - e^{-3/2}.$$

Exercise 3 Lifetime in hours of a certain light bulb is a random variable with pdf

$$f(x) = \begin{cases} 0 & x \leq 100 \\ \frac{100}{x^2} & x > 100 \end{cases}$$

$$P(\text{Exactly 2 of 5 replaced in 150 hours}) = \binom{5}{2} p^2 (1-p)^3,$$

where

$$p = \int_{100}^{150} f(x)\,\mathrm{d}x = \frac{1}{3}.$$

So,

$$P(\text{Exactly 2 of 5 replaced in 150 hours}) = \frac{80}{243}.$$

Cumulative Distributions: Cumulative distribution function $F(x)$ is the probability that the random variable is less than x:

$$F(x) = \int_{-\infty}^{x} f(t)\,\mathrm{d}t.$$

Going the other way,

$$\frac{\mathrm{d}}{\mathrm{d}a} F(a) = f(a).$$

Exercise 4 Let X have pdf $f_X(x)$, and $Y = 2X$. Letting

$$F_Y(a) = P(Y \leq a)$$

and

$$F_X(a) = P(X \leq a),$$

we have

$$P(2X \leq a) = P(X \leq a/2) = F_X(a/2),$$

so, by chain rule,

$$f_Y(a) = f_X(a/2) \cdot \frac{1}{2}.$$

Expected value

$$\int_{\mathcal{S}} x f(x)\,\mathrm{d}x,$$

$$E[g(X)] = \int_{\mathcal{S}} g(x) f(x)\,\mathrm{d}x.$$

Variance

$$E[X^2] - E[X]^2 = \int_{\mathcal{S}} x^2 f(x)\,\mathrm{d}x - \left(\int_{\mathcal{S}} x f(x)\,\mathrm{d}x\right)^2.$$

Exercise 5 Given **uniform random variable**

$$f(x) = \begin{cases} 1 & 0 \le x \le 1 \\ 0 & \text{otherwise,} \end{cases}$$

then

$$E\left[e^X\right] = \int_0^1 e^x\,\mathrm{d}x = e - 1.$$

Now, let $Y = e^X$. Find f_Y. We have

$$F_Y(a) = P(Y \le a) = P(e^X \le a) = P(X \le \log(a)) = F_x(\log(a)).$$

By chain rule,

$$f_y(a) = \frac{f_x(\log(a))}{a} = \begin{cases} \frac{1}{a} & 1 \le a \le e \\ 0 & \text{otherwise} \end{cases}.$$

1.6.3 Continuous Distributions

Uniform Random Variables

$$f(x) = \begin{cases} 1 & 0 < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

or

$$f(x) = \begin{cases} \frac{1}{\beta - \alpha} & \alpha < x < \beta \\ 0 & \text{otherwise} \end{cases}.$$

$$E[X] = \frac{\beta + \alpha}{2} \qquad \mathrm{Var}[X] = \frac{1}{12}(\beta - \alpha)^2$$

1.7 More Continuous Random Variables (2024.09.05)

1.7.1 Gaussian Distribution & Central Limit Theorem

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)} \qquad x \in \mathbb{R}.$$

$$E[X] = \mu \qquad \mathrm{Var}[X] = \sigma^2$$

Ignore this:

$$\mathcal{N}(\mu_1,\, \sigma_1^2) * \mathcal{N}(\mu_2,\, \sigma_2^2) = \mathcal{N}\left(\mu_1 + \mu_2,\, \sigma_1^2 + \sigma_2^2\right)$$

According Laplace, binomial distributions approach a Gaussian distribution as $n \to \infty$. X being a random variable from a normal/gaussian distribution is written as $X \sim \mathcal{N}(\mu, \sigma^2)$.

Do the Gaussian integral by squaring and using polar:

$$\int_{-\infty}^{\infty} e^{-x^2}\ \mathrm{d}x = \sqrt{\pi},$$

it follows that $f(x)$ is normalized with u-sub. By integrating, letting $X \sim N(0,1)$, then

$$E[X] = 0, \mathrm{Var}[X] = 1.$$

Then, let

$$Y = aX + b,$$

$$E[Y] = \mu,$$

and

$$\mathrm{Var}[Y] = \sigma^2.$$

Exercise 1 Let us transmit a binary message (0 or 1) from point A to point B. If I want to send a 1, I send a signal of height 2, and if I want to send a 0, I send a signal of height -2. However, this signal might become disturbed by noise. So, you will receive

$$R = x + N.$$

If $R \geq 0.5$, we assume that 1 was sent, otherwise we assume a 0 was sent. Let N be distributed with $N(0,1)$. If I send a 1, then

$$R_1 \sim N(2,1),$$

so

$$P[\text{fail}_1] = P[R_1 < 0.5] \approx 0.0067.$$

If I send a 0, then

$$R_2 \sim N(-2,1)$$

$$P[\text{fail}_2] = P[R_2 \geq -0.5] \approx 0.0730.$$

Then,

$$P[\text{fail} = \frac{1}{2}(P[\text{fail}_1] + P[\text{fail}_2]) \approx 0.0365.$$

1.7.2 Exponential Distribution

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

$$E[X] = \lambda^{-1} \qquad \mathrm{Var}[X] = \lambda^{-2}$$

Models the amount of time that passes before an event happens. The relationship between Exponential and Poisson is the same as the relationship between Geometric and Binomial.

More precisely, the probability that the event happens at least once after time t in a Poisson process with rate λ is 1 - probability it never happens:

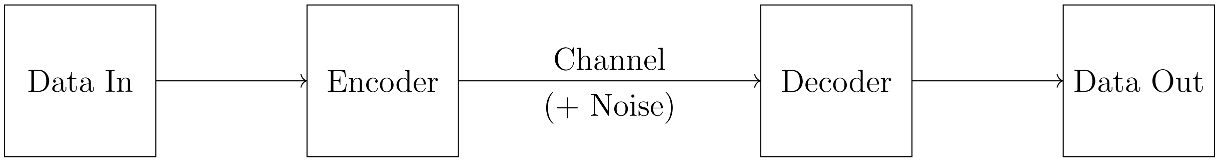
$$1 - \sum_{k=0}^{0} e^{-\lambda t} \frac{(\lambda t)^k}{k!} = 1 - e^{-\lambda t}$$

Which equals exactly the CDF of the corresponding exponential distribution! The CDF evaluated at t has the interpretation of "the probability the event happened at least once between t=0 and now", which matches up with our calculation from the Poisson distribution.

$$\int_0^t \lambda e^{-\lambda x} \, \mathrm{d}x = 1 - e^{-\lambda t}$$

2 Information

2.1 Intro to Information (2024.09.11)



Mathematical formalism: "Data In" is the information source (random variable), and the data is a sequence of k symbols where each symbol is the outcome of a random variable.

$$S : \{s_1, s_2, \ldots, s_k\}.$$

Encoder acts on blocks of symbols (vectors) and does $\vec{x} = g(\vec{s})$. The encoder's output is a sequence of codewords, which is modeled as another random variable with m different values

$$X : \{x_1, x_2, \ldots, x_m\}$$

In transit, noise may be added to the codewords, and this noise is another random variable. The decoder undoes what the encoder did and may also try to correct for noise and errors.

2.1.1 Shannon's Axioms

- Continuity: The amount of information associated with an outcome increases/decreases smoothly (is continuous) as the probability of the outcome changes.
- Symmetry: The amount of information associated with a sequence of outcomes does not depend on the order in which the outcomes occur.
- Maximal Value: The amount of information associated with a set of outcomes cannot be increased if those outcomes are already equally probably.
- Additive: The information associated with a set of outcomes is obtained by adding the information of individual outcomes. Info(1 and 2 and 3) = Info(1) + Info(2) + Info(3)

Information measures amount of surprise, e.g. if a coin that lands on heads 90% of the time ends up landing on tails, then you are surprised. So we guess that the surprise amount is $1/p$. But, if $p = 1$ then surprise is 1 (which should be zero) and this isn't additive, so we take the $\log_2$:

$$\log_2\left(\frac{1}{p}\right).$$

Assume that all logs are base 2 from here on.

$$h(x) = -\log(p(x))$$

Units are bits (this is not a regular cs bit that you nerds know). One bit is the amount of information required to choose between 2 equally likely outcomes. A binary digit (computer science bit), on the other hand, is the value of a binary variable, which is zero or one.

We are interested in the expected/average amount of surprise from an event:

$$X : \{x_1, x_2, \dots, x_,\} \qquad P(X) : \{P(x_1), P(x_2) \dots, P(x_m)\}$$

$$\begin{aligned} H(X) &= \sum_{i=1}^{m} P(x_i) \log\left(\frac{1}{P(x_i)}\right) \\ &= E\left[\log\left(\frac{1}{P(X)}\right)\right]. \end{aligned}$$

$H(x)$ is the **entropy** of the random variable.

2.1.2 Properties of Entropy

- $H(X) \geq 0$.
- $H_b(X) = \log_b a \cdot H_a(X)$.
- Doubling the number of outcomes increases $H(X)$ by one bit. (Equivalent to flipping a coin + doing the original thing, counting heads + outcome and tails + outcome as different outcomes).

2.2 Entropy Properties (9.13.2024)

$$H(X) = \sum P(X) \log\left(\frac{1}{P(X)}\right).$$

Example 1 Fair die with 8 sides.

$$H(X) = \sum_{i=1}^{8} \frac{1}{8} \log(8) = 3.$$

Example 2 Let

$$X = \begin{cases} a & p = \frac{1}{2} \\ b & p = \frac{1}{4} \\ c & p = \frac{1}{8} \\ d & p = \frac{1}{8}. \end{cases}$$

$$\begin{aligned} H(X) &= \frac{1}{2}\log(2) + \frac{1}{4}\log(4) + \frac{1}{8}\log(8) + \frac{1}{8}\log(8) \\ &= \frac{1}{2} + \frac{1}{2} + \frac{3}{8} + \frac{3}{8} = \frac{7}{4}. \end{aligned}$$

Example 3

$$X = \begin{cases} 1 & p \\ 0 & 1-p \end{cases}$$

$$H(p) := H(X) = -p\log(p) - (1-p)\log(1-p)$$

$H(p)$ is maximized when $p = \frac{1}{2}$ with $H\left(\frac{1}{2}\right) = 1$. At the endpoints $H(0) = 0$ and $H(1) = 0$ because you get no information from certain outcomes.

2.2.1 Jensen's inequality

$f(x)$ is convex over the interval (a, b) if for every $x_1, x_2 \in (a, b)$ and any $0 \leq \lambda \leq 1$, then

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2).$$

If $f(x)$ is convex, then $-f(x)$ is concave and obeys the opposite inequality.

Jensen's inequality states that if $f(x)$ is convex over the sample space of X and X is a random variable, then

$$E[f(X)] \geq f(E[X]).$$

`https://en.wikipedia.org/wiki/H%C3%B6lder%27s_inequality`

"Proof" Why is this true? If $g(x)$ is convex then for some $x_1 < c < x_2$, then for all $x \in [x_1, x_2]$ we have

$$g(x) \geq g(c) + g'(c)(x - c).$$

Letting $c = E[X]$, we have

$$g(X) \geq g(E[X]) + g'(E[X])(X - E[X]).$$

Taking the expectation of both sides,

$$E[g(X)] \geq E[g(E[X])] + g'(E[X])E[X - E[X]] = g(E[X]).$$

`https://journalofinequalitiesandapplications.springeropen.com/counter/pdf/10.1186/s13660-016-0985-4.pdf`

Exercise 1 We know that

$$E[X^2] = \text{Var}[X] + E[X]^2 \implies E[X^2] \geq E[X]^2.$$

This is just Jensen's!!!!!

Exercise 2 $f(x) = -\log(x)$ is convex:

$$E[-\log X] \geq -\log(E[X]).$$

$$\log E[X] \geq E[\log X].$$

2.2.2 Theorem

Let X be an R.V. with values $\{x_1, x_2, \ldots, x_r\}$. Prove that

$$0 \leq H(X) \leq \log r,$$

and $H(X) = 0$ if all the $p_i \in \{0, 1\}$. Since each $p_i \leq 1$, we have

$$p_i \log\left(\frac{1}{p_i}\right) \geq 0 \implies H(X) \geq 0,$$

and

$$p_i \log\left(\frac{1}{p_i}\right) \iff p_i = 0 \text{ or } p_i = 1,$$

so $H(X) = 0$ iff each $p_i \in \{0, 1\}$.

We have, letting $Y = \frac{1}{p_i}$ with probability p_i,

$$H(X) = \sum_i p_i \log \frac{1}{p_i} \leq \log \sum_i p_i \frac{1}{p_i} = \log r,$$

applying Jensen's with $-\log$ and Y. When $p_i = \frac{1}{r}$, equality occurs, for example.

Exercise π Let

$$A = \sum_{n=2}^{\infty} \left(n \log^2 n\right)^{-1},$$

and let X be a Random Variable such that

$$P(X = n) = \frac{1}{An \log^2 n}$$

for $n = 2, 3, \ldots$ Prove that $H(X) = \infty$.

$$\begin{aligned} H(X) &= \frac{1}{A} \sum_{i=2}^{\infty} \frac{1}{n \log^2 n} \log(An \log^2 n) \\ &\geq \frac{1}{A} \sum_{i=2}^{\infty} \frac{\log n}{n \log^2 n} \\ &= \frac{1}{A} \sum_{n=2}^{\infty} \frac{1}{n \log n} = \infty. \end{aligned}$$

Exercise 3 Given $(p_1, p_2, \ldots, p_n)$ and some m with $0 \leq m \leq n$, define

$$q_m = 1 - \sum_{j=1}^{m} p_j.$$

Prove that

$$H(p_1, p_2, \ldots, p_n) \leq H(p_1, p_2, \ldots, p_m, q_m) + q_m \log(n - m).$$

$$H(p_, \leq H(q) + q \log(r)$$

2.3 Differential Entropy (2024.09.17)

Agastya's 13b solution: From 13a, we know that this is just barely possible: At maximum the scale gives us $\log_2(3)$ bits of entropy (from reading left heavy, right heavy, or equal), and the problem to be solved has $\log_2(2 \cdot 12 + 1)$ bits of entropy. The counterfeit could be any of the 12 coins, either heavier of lighter, or there might not be a counterfeit. $3 \log_2(3) = 4.75$ while $\log_2(2 \cdot 12 + 1) = 4.64$! We only have about 0.1 bits of entropy to spare (27 possibilities vs. 25)!

Strategy: Find the number of coins to weigh by considering what will give you the maximum entropy. The probabilities that the scale reads equal, left side heavy, and right side heavy should all be as close to 1/3 as possible. We must also then use the information about which side was heavier from all weighings.

- Weigh 4v4. In the case they are equal, these 8 are known good. Then,
  - Weigh 3 known good vs. 3 unknown. If equal, counterfeit must be the one left out (if there is a counterfeit). Weigh that one left out coin with a known good coin to see if it is counterfeit.
    - If the unknown coins are heavier or lighter, we know one is counterfeit and that the counterfeit is heavier or lighter correspondingly. Weigh 1 of the 3 unknowns against another of the 3 unknowns. If equal, the 3rd unknown is counterfeit, and if unequal, counterfeit is the one agreeing with the previous weighing.
- If the 4v4 was unequal, a counterfeit exists. Let's denote the 4 heavier coins as HHHH and 4 lighter as LLLL and 4 known good coins as CCCC. Weigh LLHHH vs. HCCCC
  - if $LLHHH > HCCCC$, one of the 3 Hs on the left-hand-side is a heavy counterfeit. Weigh 2 of them in a 1v1 and see which is heavier (if equal, the one left out must be heavier).
  - if $LLHHH = HCCCC$, counterfeit is light and is one of the 2 Ls we left out. Do a 1v1 to see which is the lighter counterfeit.
  - If $LLHHH < HCCCC$, counterfeit is either one of the 2 Ls on the left-hand-side or the H on the right-hand-side. Weigh the 2 Ls in a 1v1 to see if one is the lighter counterfeit. If they are equal, the heavy one must be the counterfeit.

2.3.1 Differential Entropy

We have

$$H(X) = \sum p(X) \log \frac{1}{p(X)}.$$

Just like in a Riemann sum, we will split a continuous distribution into many "bins" of width Δx.

$$P(\text{random value X is in a given bin}) = \text{proportion of all values in that bin}.$$

Define

$$X^\Delta$$

to be the discretized version of X, which is a continuous R.V. We have

$$H\left(X^\Delta\right) = \sum_i P(X \in \text{bin}_i) \log \frac{1}{P(X \in \text{bin}_i)}.$$

The probability of X being in the ith bin is the ratio between a_i, which is the the area of the ith bin, to the total area A :

$$a_i = n_i \Delta x, \qquad A = \sum a_i, \qquad P_i = \frac{a_i}{A}, \qquad \sum P_i = 1.$$

We have

$$P_i = P(x_i)\Delta x.$$

$$\begin{aligned} H\left(X^\Delta\right) &= \sum P_i \log \frac{1}{P_i} \\ &= \sum P(x_i)\Delta x \log \frac{1}{P(x_i)\Delta x} \\ &= \sum P(x_i)\Delta x \log \frac{1}{P(x_i)} + \sum P(x_i)\Delta x \log \frac{1}{\Delta x}. \end{aligned}$$

Taking the limit, we get

$$\int P(x) \log \frac{1}{P(x)} \, \mathrm{d}x + \log \frac{1}{\Delta x} \int P(x) \, \mathrm{d}x.$$

Unfortunately, this goes to negative infinity... So, Differential Entropy is necessary, which would yield

$$h(X) = \int f(x) \log \frac{1}{f(x)} \, \mathrm{d}x,$$

where $f(x)$ is the pdf of X. (we just ignore the second term lol)

Properties:

- If $Y = X + c$, then $h_y = h_x$.
- If $Y = cX$, then $h_y = h_x + \log c$.

Example 1 Let $X \sim U(0, a)$. Then

$$h(X) = \int_0^a \frac{1}{a} \log a = \log a.$$

If you double the sample size, then h increases by 1, just like discrete entropy.

Since h can be negative, we treat it with respect to $U(0,1)$ as $h(X \sim U(0,1)) = 0$. We treat h to mean having more or less information than $U(0,1)$.

Differential entropy of an exponential distribution:

$$h(X) = -\int_0^\infty \lambda e^{-\lambda x} \log(\lambda e^{-\lambda x}) \, \mathrm{d}x = \log\left(\frac{e}{\lambda}\right)$$

Claim:

- Given a fixed lower and upper bound, a no pdf has a larger entropy than a uniform
- Given a positive R.V. X with a mean μ, then the distribution with maximum entropy is an exponential random variable.

2.4 More Entropy (9.19.2024)

2.4.1 Aarush Integrates Simplices

Probability Tuples: First, we must define the object we are working with.

Definition We define a *probability tuple* of size n and total probability a as a real tuple $\boldsymbol{p}_{n,a} = (p_1, \ldots, p_n)$ where $0 \leq a \leq 1$, and $0 \leq p_i \leq 1$ for integers $1 \leq i \leq n$, and $\sum_{i=1}^{n} p_i = a$.

We need some method to count probability tuples.

Definition We denote the number (measure) of valid probability tuples $\boldsymbol{p}_{n,a}$ as the number $S(n, a)$.

Note that $S(1, a) = 1$, since the only tuple is (a). Similarly, $S(2, a) = a$, since p_1 can be any value from 0 through a and p_1 uniquely fixes p_2. Moreover, for $n \geq 2$, note that there is a bijection between $\boldsymbol{p}_{n,a}$, and the space of probability tuples $\boldsymbol{p}_{n,b}$ as b varies from 0 to a. More specifically,

$$S(n, a) = \int_0^a S(n-1, a-x)\, dx = \int_0^a S(n-1, x)\, dx.$$

Using this equation we get that

$$S(n, a) = \frac{a^{n-1}}{(n-1)!}.$$

We define a random variable $X_n = (x_1, \ldots, x_n)$ to be a random probability tuple $\boldsymbol{p}_{n,1}$, where X_n is equally likely to be every distinct probability tuple. Our goal is to find the probability distribution for $P(x_1 = k)$ for some $0 \leq k \leq 1$.

To do this, we will find the cumulative probability distribution by finding the probability $P(x_1 \leq k)$. Say a tuple $\boldsymbol{p}_{n,a}$ has $p_1 \geq k$. Then, we can subtract k from p_1 to get a probability tuple $\boldsymbol{p}'_{n,a-k}$. In other words, there is a bijection between these two objects. Thus, to count the tuples $\boldsymbol{p}_{n,a}$ where $p_1 \leq k$, we can use complementary counting, which then gets the cumulative distribution of

$$P(x_1 \leq k) = \frac{S(n,1) - S(n, 1-k)}{S(n,1)} = 1 - (1-k)^{n-1},$$

The $S(n, 1-k)$ counts the number of probability tuples where we can add k to one of the probabilities $p_1 < 1-k$ so that the tuple sums to 1, so $S(n, 1-k)$ counts the number of n-tuples summing to 1 where $p_1 > k$. Alternatively, we can integrate over all possibilities for $p_1 < k$ and write this as

$$P(x_1 \leq k) = \frac{1}{S(n,1)} \int_0^k S(n-1, 1-p)\, \mathrm{d}p = 1 - (1-k)^{n-1}.$$

Thus, we can calculate the probability distribution function as

$$P(x_1 = k) = \frac{d}{dk}P(x_1 \leq k) = (n-1)(1-k)^{n-2}.$$

Expected Information: Let $H(x) = -x\log_2(x)$, and $H(x_1, \ldots, x_n) = \sum_{i=1}^n H(x_i)$. Then, the expected amount of information from a probability distribution is

$$\begin{aligned} E(H(X_n)) &= E\left(\sum_{i=1}^n H(x_i)\right) = \sum_{i=1}^n E(H(x_i)) = n \cdot E(H(x_1)) \\ &= n \cdot \int_0^1 P(x_1 = k) \cdot H(k)\, dk \\ &= -n(n-1) \cdot \int_0^1 (1-k)^{n-2} \cdot k \cdot \log_2(k)\, \mathrm{d}x, \end{aligned}$$

which we simplify using Mathematica to get

$$E(H(X_n)) = \frac{H_n - 1}{\ln(2)},$$

where $H_n = \sum_{i=1}^n \frac{1}{i}$.

2.4.2 Funny Discrete Problem

Go back to discrete: Given $(p_1, p_2, \ldots, p_n)$ and some m with $0 \leq m \leq n$, define

$$q_m = 1 - \sum_{i=1}^m p_i = \sum_{i=m+1}^n p_i.$$

Prove that

$$H(p_1, p_2, \ldots, p_n) \leq H(p_1, p_2, \ldots, p_m, q_m) + q_m \log(n-m).$$
$$H(p_{m+1}, \ldots, p_n) \leq H(q_m) + q_m \log(n-m)$$

Let

$$z = H(p_{m+1}, \ldots, p_n) = \sum_{i=m+1}^n p_i \log\left(\frac{1}{p_i}\right)$$

Divide by q_m :

$$\frac{z}{q_m} = \sum_{i=m+1}^n \frac{p_i}{q_m} \log\left(\frac{1}{p_i}\right)$$

Add $\log q_m$:

$$\frac{z}{q_m} + \log q_m = \sum_{i=m+1}^n \frac{p_i}{q_m} \log\frac{q_m}{p_i},$$

let this be $H(Y)$. Then,

$$z = H(p_{m+1}, \ldots, p_n) = q_m(H(Y) - \log q_m).$$

But, we have $H(Y) \leq \log(n-m)$ from maximal entropy, so

$$q_m(H(Y) - \log q_m) \leq q_m \log \frac{1}{q_m} + q_m \log(n-m),$$

and thus

$$H(p_1, p_2, \ldots, p_n) \leq H(p_1, p_2, \ldots, p_m, q_m) + q_m \log(n-m).$$

2.4.3 Differential Entropy of Gaussian R.V.

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-x^2/2\sigma^2}.$$

$$\begin{aligned}
h(X) &= -\frac{1}{\ln 2} \int_{-\infty}^{\infty} f(x) \ln f(x) \, \mathrm{d}x \\
&= -\frac{1}{\ln 2} \int_{-\infty}^{\infty} f(x) \left(-\frac{x^2}{2\sigma^2} - \ln \sqrt{2\pi\sigma^2} \right) \\
&= -\frac{1}{\ln 2} \left(\frac{-E[X^2]}{2\sigma^2} - \ln \sqrt{2\pi\sigma^2} \right) \\
&= -\frac{1}{\ln 2} \left(\frac{-E[X^2]}{2\sigma^2} - \frac{1}{2} \ln 2\pi\sigma^2 \right).
\end{aligned}$$

But, $E[X^2] = \sigma^2 + \mu^2 = \sigma^2$, so

$$h(X) = \frac{1}{2\ln 2} \left(1 + \ln 2\pi\sigma^2 \right) = \frac{1}{2} \log_2(2\pi e \sigma^2).$$

2.4.4 Source Coding

Source $\rightarrow$ message $\rightarrow$ Encoder $\rightarrow$ signal

We want a code for English text because america, specifically, we want a scheme to map letters to a sequence of 0's and 1's.

A	00000
B	00001
C	00010
$\vdots$	$\ddots$

This uses 5 bits, but it is bad.

Exercise 1 Let $E = 0$, $T = 1$, $A = 01$, but A=ET????!?!???!? bad

Exercise 3 Now try $E = 1$, $T = 011$, $A = 1110$, $O = 10011$, $I = 01110$,

But now 011101110011 is both IAT and TEIT, so decoding is not unique.

Exercise 3 Let $E = 0$, $T = 10$, $A = 110$, $O = 111$.

Now, 00100110111010 is uniquely EETEAOET.

The difference between the second and third encodings is that no code in the third encoding has a prefix equal to another code.

2.5 Lagrange & Source Coding (9.23.2024)

2.5.1 Review

We want a uniquely decodable code better than 5 bits per letter that is also instantaneous (like a DFA), otherwise known as prefix-free.

We call a code non-singular if every element in X maps to a different string. So

$$x_i \neq x_r \implies C(x_i) \neq C(x_j).$$

We call a code uniquely decodable if any decoded string has only one possible source string that produced it.

We call a code a prefix code or an instantaneous code if no codeword is the prefix of any other codeword.

Instantaneous codes are a subset of uniquely decodable codes, which are a subset of non singular codes, which are a subset of all codes. Example:

X	Singular	Nonsingular but not uniquely decodable	Uniquely Decodable but not Instantaneous	Instantaneous
1	0	0	10	0
2	0	010	00	10
3	0	01	11	110
4	0	10	110	111

Exercise 1

$X =$

Number	Probability	Code
1	1/2	0
2	1/4	10
3	1/8	111
4	1/8	111

We see that $H(X) = 1.75\,\text{bits}$ and $E[\text{length}] = \mathcal{L}(C) = 1.75$.

Exercise 2

$X =$

Number	Probability	Code
1	1/3	0
2	1/3	10
3	1/3	11

We see that $H(X) = \log(3)\,\text{bits}$ and $\mathcal{L}(C) = \frac{5}{3}\,\text{bits}$. Note that

$$H(X) \approx 1.585 < \mathcal{L}(C) \approx 1.667.$$

2.5.2 Huffman Code

This is a way to encode any source with an probability distribution into bits. In short, the more probable a word is, the fewer bits its codeword has. Algorithm:

- Find the two least probable symbols (*e.g.*, letters)

- Combine the two letters to make an imaginary composite symbol. This composite has a probability of the sum of the probabilities of each event.
- Repeat this process until you have one symbol left.
- For every time you combine two symbols, assign the top path with a 1 and the bottom path with a zero
- Make it a tree by concatenating binary numbers the lead to that symbol.

Exercise 3

A	0.87 (1)			1.00 (root)
B	0.04 (1)	0.08 (1)	0.13 (0)	
C	0.04 (0)			
D	0.03 (1)	0.05 (0)		
E	0.02 (0)			

The two least probable symbols are D and E, so make DE with probability 0.05. Then we combine B and C with the lowest probabilities of 0.04 and 0.04 to make BC with probability 0.08. Continue this, and make a tree

2.5.3 Lagrange Multipliers and (Constrained) Optimization

The problem: Maximize scalar function $f(\vec{v})$ under the constraints $c_i(\vec{v}) = 0$.

Solution: Solve for $\nabla f(\vec{v}) = \sum_i \lambda_i \nabla c_i(\vec{v})$. The values of the λ_i do not matter, only the value of $\vec{v}$.

If the gradient of f does not lie purely in the space spanned by the gradients of the constraints, then we can always increase f by moving along the constraint manifold (moving along the constraint manifold = moving in a direction perpendicular to the gradients of the constraint functions, think about contours of constant value). Adding constraints removes constraints on the gradient of f.

Exercise 4 Maximize $f(x, y) = 3x + 4y$ with $g(x, y) = x^2 + y^2 - 1 = 0$. We turn this constrained problem into an unconstrained maximum of a new function $\phi(x, y, \lambda)$. Definition:

$$\phi(x, y, \lambda) = f(x, y) - \lambda g(x, y).$$

The gradient is

$$\nabla \phi(x, y, \lambda) = \begin{bmatrix} \frac{\partial f}{\partial x} + \lambda \frac{\partial g}{\partial x} \\ \frac{\partial f}{\partial y} + \lambda \frac{\partial g}{\partial y} \\ g(x, y) \end{bmatrix} = \vec{0}.$$

So,

$$\begin{bmatrix} 3 + 2\lambda x \\ 4 + 2\lambda y \\ x^2 + y^2 - 1 \end{bmatrix} = \vec{0}.$$

So, $4\lambda^2 = 25$, and thus $\lambda = \pm\frac{5}{2}$. Then,

$$x = \frac{3}{5}, y = \frac{4}{5}.$$

Exercise 5 Optimize $f(x, y, z) = xyz$ given $x^2 + y^2 + z^2 = 12$. We have

$$\begin{bmatrix} yz + 2\lambda x \\ xz + 2\lambda y \\ xy + 2\lambda z \\ x^2 + y^2 + z^2 - 12 \end{bmatrix} = \vec{0}.$$

So,

$$x = y = z = 2.$$

2.6 Lagrange 2 (9.30.2024)

Maximize discrete entropy:

$$H(p_1, p_2, \ldots, p_n) = -\sum_i p_i \log p_i$$

Under the constraint function:

$$C(p_1, p_2, \ldots, p_i) = -1 + \sum_i p_i = 0$$

Lagrange multipliers:

$$\begin{aligned} \Phi(p_1, p_2, \ldots, p_i, \lambda) &= \nabla H - \lambda \nabla C \\ &= \begin{bmatrix} -\log p_1 - 1 \\ -\log p_2 - 1 \\ \cdots \\ -\log p_n - 1 \end{bmatrix} - \lambda \begin{bmatrix} 1 \\ 1 \\ \cdots \\ 1 \end{bmatrix} = 0 \end{aligned}$$

Using additional knowledge that $0 \leq p_i \leq 1$, we know that $p_i = \frac{1}{n}$, for $1 \leq i \leq n$, and thus $H = \log n$.

Continuous Given a continuous distribution that is bounded with

$$a \leq x \leq b,$$

prove that the uniform distribution maximizes entropy.

We want to maximize

$$\int f(x) \log \frac{1}{f(x)} \, \mathrm{d}x,$$

given

$$\int f(x) \, \mathrm{d}x = 1.$$

Let

$$\phi(f) = \int f(x) \log \frac{1}{f(x)} \, \mathrm{d}x - \lambda \left(\int f(x) \, \mathrm{d}x - 1 \right).$$

$$\xrightarrow{f}\boxed{\Phi}\xrightarrow{\mathbb{R}}$$

We want to perturb f slightly and see how $H(f)$ changes.
Perturb the range $f(x)$ to $f(x+\delta)$ by ϵ. How much does ϕ change?

$$\Phi(f_{perturbed})-\Phi(f) = \left((f(x)+\epsilon)\log\frac{1}{f(x)+\epsilon} - f(x)\log\frac{1}{f(x)}\right)\delta - \lambda\left((f(x)+\epsilon)\delta - 1 - (f(x)\delta - 1)\right).$$

After algebra and factoring out a delta,

$$(f(x)+\epsilon)\log\frac{1}{f(x)+\epsilon} - f(x)\log\frac{1}{f(x)} - \lambda(f(x)+\epsilon - f(x)).$$

So,

$$\epsilon\left(\frac{d}{dz}z\log\frac{1}{z}\right)\bigg|_{z=f(x)} - \lambda\epsilon = 0.$$

So,

$$f(x) = e^{-1-\lambda}$$

is constant, so f is uniform.

Problem Find $p_0, p_1, p_2, \ldots$ so that

$$\sum_i p_i \log\frac{1}{p_i} + \log\Delta x$$

is maximized, given

$$\sum_i p_i\Delta x = 1$$

and the mean is contrained:

$$\sum_i i p_i \Delta x^2 = \mu$$

is constant.

We have

$$\Phi(p_1, p_2, \ldots, \Delta x, \lambda_1, \lambda_2) = \sum_i p_i\log\frac{1}{p_i} + \log\Delta x - \lambda_1\left(\sum_i p_i\Delta x - 1\right) - \lambda_2\left(\sum_i i p_i\Delta x^2 - \mu\right).$$

Taking the gradient,

$$\nabla\Phi = \begin{bmatrix} -\log p_0 - 1 - \lambda_1\Delta x^2 \\ -\log p_1 - 1 - \lambda_1\Delta x - \lambda_2\Delta x^2 \\ -\log p_2 - 1 - \lambda_1\Delta x - 2\lambda_2\Delta x^2 \\ \cdots \\ -\log p_i - 1 - \lambda_1\Delta x - i\lambda_2\Delta x^2 \\ \cdots \\ \sum_i p_i\Delta x - 1 \\ \sum_i i p_i\Delta x^2 - \mu \end{bmatrix}$$

You can solve this, and get

$$p_i = \frac{1}{\mu} e^{-x/\mu}.$$

3 Source Coding

3.1 Source Coding Theorem (10.2.2024)

Goal: Construct instantaneous codes of minimum expected length. We know that you cannot assign short codewords to all source symbols if we want the code to still be instantaneous.

3.1.1 Kraft Inequality

$$\sum_{i=1}^{m} \mathcal{D}^{-\ell_i} \leq 1,$$

Where $\mathcal{D}$ is the alphabet size (*e.g.*, binary codes have $\mathcal{D} = 2$), and the codeword lengths are $\ell_1, \ell_2, \ldots, \ell_m$. Conversely, given a $\mathcal{D}$ and ℓ_i that satisfy this inequality, there exists an instantaneous code with lengths ℓ_i.

Proof: Let $\mathcal{D} = 2$ because that's all we care about honestly. Consider a binary tree, with each node having 2 children and each branch is a symbol of some codeword. Each leaf is a codeword (for now).

No prefixes means that no codeword is the descendant of another codeword . Let ℓ_{max} be the length of the longest set of codewords. Now, grow out the tree to length ℓ_{max} for all branches. A codeword at level ℓ_i has

$$2^{\ell_{max} - \ell_i}$$

leaf descendants. The leaf descendants of each codeword form m disjoint sets because no codeword is the descendent of another. The total number of nodes in these leaf descendants must be at most $2^{\ell_{max}}$, so for we have $\sum_{i=1}^{m} 2^{\ell_{max}-\ell_i} \leq 2^{\ell_{max}}$ (remember disjoint sets). Thus:

$$\sum_{i=1}^{m} 2^{-\ell_i} \leq 1.$$

This same idea works for all $\mathcal{D}$.

3.1.2 Minimum Codeword Length

Minimize expected binary codeword length $L = \sum_i p_i \ell_i$ for $\ell_1, \ell_2, \ldots, \ell_m$ satisfying

$$\sum_i 2^{-\ell_i} \leq 1.$$

Pretend it's an equality trust me bro.

$$\Phi = \sum_i p_i \ell_i - \lambda \left(\sum_i 2^{-\ell_i} - 1 \right),$$

$$\implies \frac{\partial \Phi}{\partial \ell_i} = p_i + \lambda 2^{-\ell_i} \ln 2 = 0.$$

$$2^{-\ell_i} = \frac{-p_i}{\lambda \ln 2}.$$

Substitute this into the contsraint that

$$\sum_{i=1}^{m} 2^{-\ell_i} = 1,$$

$$\sum_{i=1}^{m} \frac{-p_i}{\lambda \ln 2} = -\frac{1}{\lambda \ln 2} = 1,$$

$$\lambda = \log_2(e)$$

and the optimal length

$$\ell_i^* = -\log_2 p_i.$$

This means that the average length

$$\mathcal{L}^* = \sum_i p_i \ell_i^* = H(X)!!!!$$

Small problem, ℓ_i should be integral but it's fine...

3.1.3 Source Coding Theorem:

The expected length $\mathfrak{L}$ of a binary codeword is greater than or equal to the entropy:

$$\mathfrak{L} \geq H(X),$$

with equality iff $2^{\ell_i} = p_i$.

Proof:

$$\mathfrak{L} - H(X) = \sum_i p_i(\ell_i + \log p_i) = \sum_i p_i(-\log_2 2^{-\ell_i} + \log_2 p_i) = \sum_i p_i \log \frac{p_i}{r_i} - \log c,$$

where

$$r_i = \frac{2^{-\ell_i}}{\sum_i 2^{-\ell_i}}$$

and

$$c = \sum_i 2^{-\ell_i}.$$

Consider now

$$-\sum p_i \log \frac{p_i}{r_i} = \sum p_i \log \frac{r_i}{p_i} \leq \log \sum p_i \frac{r_i}{p_i} = \log \sum r_i = \log 1 = 0,$$

by Jensen. This means that $\mathfrak{L} \geq H(X)$.

3.2 Huffman is Optimal (2024.10.07)

(? rohan please help) Proof that a huffman code achieves optimal avg codeword length within 1 bit of lower bound:

$$\ell_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil$$

Lengths of codewords satisfy Kraft's inequality (instantaneous code):

$$\sum 2^{-\ell_i} = \sum 2^{-\left\lceil \log \frac{1}{p_i} \right\rceil} \le \sum 2^{-\log \frac{1}{p_i}} = \sum p_i = 1$$

We have

$$\log \frac{1}{p_i} \le \ell_i \le \log \frac{1}{p_i} + 1$$

Taking expected value of everything:

$$H(X) \le L \le H(X) + 1 \quad \blacksquare$$

Lemma: For any probability distribution (without loss of generality, assume that $p_1 \ge p_2 \ge \cdots \ge p_m$), there exists an optimal instantaneous code (of minimum expected codeword length) such that

1. If $p_j > p_k$, then $\ell_j \le \ell_k$.
   - We consider swapping codewords. Say we have code C'_m which is C_m but with the codewords for j and k swapped.

     $$\begin{aligned} L(C'_m) - L(C_m) &= \sum p_i \ell'_i - \sum p_i l_i \\ &= p_j \ell_k + p_k \ell_j - p_j \ell_j - p_k \ell_k \\ &= (p_j - p_k)(\ell_k - \ell_j) \end{aligned}$$

     Because C_m is an optimal code, the left-hand side $L(C'_m) - L(C_m) \ge 0$. By definition, $p_j - p_k > 0$, therefore $\ell_k - \ell_j \ge 0$. $\blacksquare$

2. The two longest codewords have the same length and differ only in the last bit.
   - Proof by Contradiction: assume that the 2 longest codewords have differing lengths. Then, we can delete the last bit of the longest and still be instantaneous (definition of instantaneous codes) so that code cannot have been optimal. By property 1, the longest codewords are the least likely symbols.
   - Alternatively (thanks Aarush), Kraft's inequality will hold with equality if a code is optimal. Therefore,

     $$\sum 2^{-\ell_i} = 1 \implies \sum 2^{\ell_{max} - \ell_i} = 2^{\ell_{max}}$$

     The right-hand side is necessarily even, but if there is only one code with the longest length, the left-hand side will be odd, as that term will be 1 and the rest will be powers of 2.

3. And these longest codewords correspond to the two least likely symbols.
   - Not all optimal codes have this property, but by rearranging the 0 and 1 labels on the tree, I can find a code that does satisfy this.

Actual proof stuff: We want to prove that if a code satisfies those properties, it can be thought of as a Huffman code.

Create a "merged" code C_{m-1} with $m-1$ symbols from C_m as follows: Take the common prefix of the two longest codewords and allot it to a (fictional) symbol with probability $p_{m-1}+p_m$, keeping all other codewords the same. Define p_i and ℓ_i for C_m and p'_i and ℓ'_i for C_{m-1}: for $0 \leq i < m-1$, $p'_i = p_i$ and $\ell'_i = \ell_i$. For the two merged symbols, $p'_{m-1} = p_{m-1}+p_m$ and $\ell'_{m-1} = \ell_m - 1 = \ell_{m-1} - 1$. Thus, removing the 2 longest codewords from an optimal instantaneous code:

$$\begin{aligned} L(C_m) &= \sum p_i \ell_i \\ &= \sum_{i=1}^{m-2} p'_i \ell'_i + (p_{m-1}+p_m)(\ell'_{m-1}+1) \\ &= \sum_{i=1}^{m-1} p'_i \ell'_i + (p_{m-1}+p_m) \\ &= L(C_{m-1}) + p_{m-1} + p_m, \end{aligned}$$

Minimizing $L(C_m)$ is the same as minimizing $L(C_{m-1})$ This process of combining the 2 lowest probability symbols is why Huffman works: Repeat the process down to 1 symbol, and you have created a Huffman tree which is optimal as $L(C_1) = 0$.

4 Guest Speaker: Dr. Subramanian

4.1 Large Language Model: an Information Theory Approach

In 1952, Shannon wrote a paper on the information of the English language, talking about average entropy and average word length. He figured this out by counting words in a library.

- Bag of words distribution (0th order model): Cut out words of a text, place them in a bag, and pick them out randomly. Really stupid but a good start. Surprisingly good, for example sentiment analysis just counts "happy" and "sad" words.

  Zipf's "law:" bag of words plot sorted in decreasing order; looks like $\frac{1}{n}$, this law has been observed in many different languages.

- Unigram model (1st order model): Words depend on the previous word, so we let the probability distribution depend on the previous word, so each word corresponds to the probability distribution of the next word, forming a table. From this, you can generate text, and this is pretty garbage but not as bad as the 0th order method.

- Bigram moddel (2nd order model): Each pair of two words correspond to a probability distribution of the third word. This is better than first model but still pretty garbage.

  The English dictionary has about a million words, of which 100k are used often enough. In the second order case, there are a 100000^2 entries in the table.

- ChatGPT is just an n-gram model for $n = 32776$. Practically, it's just a table with 32777 columns and 100k rows. Obviously 100000^{32776} is too ridiculously large to store. But in reality, the pdfs are pretty sparse, so most words have almost zero probability. This means that this table is compressible. In order to compress this, companies like Google managed to compress this to about 1 billion parameters (which is really good compared to the original size). In short, they made a neural network that takes in 32776 words as an input and outputs a probability distribution: `https://arxiv.org/abs/1706.03762`. Then everyone makes a bunch of LLM's, like Gemini, Llama, GPT, Grok ai, Claude, etc. But, they all are really the same concept fundamentally, regardless of what the hype is.

Occam's Razor: If there are multiple explanations, believe the simplest one, all things equal. For example, Kepler's simple laws validated the pages and pages of measurements that had been made of the positions of the planets, so obviously you should follow Kepler's laws. Then, Newton's law of gravity explains the planets, so you should prefer Newton's laws.

In the AI world, we think of underfitting and overfitting. For example, if we have five equations and three variables, we don't have enough degrees of freedom (underfitting), while if we have five equations and seventeen variables, we have too many degrees of freedom (overfitting). In AI, this is the comparison between the amount of training data and the amount of parameters. Occam's razor would say to find the best fit that's not too simple (i.e., don't overfit and don't underfit). In an LLM, you need to find the right number of parameters to not overfit or underfit. What industry is doing is increasing the number of parameters until the model starts overfitting to figure out the optimal LLM.

Four powerhouses of AI: MIT by Minsky, Stanford by McCarthy, CMU by Simon, IBM by Solomonoff. In the 1980's, when Reagan became president, people said that you shouldn't give funding to "random" people unless for military. Someone that got funding snipped was Geoffrey Hinton, who was working on the unpopular subject of Neural Networks. The first neural network was a perceptron, and people knew that two layers are enough given a large enough hidden layer: `https://en.wikipedia.org/wiki/Universal_approximation_theorem`, but people didn't yet figure out how to train networks with backpropagation, so people thought hidden layers were pointless. However, Minsky showed that XOR isn't possible without a hidden layer, so people thought Neural Networks were useless. So, Hinton moved to Canada because they are nicer people. He then worked on his own without any recognition or care that people said it was stupid to work on NN's. He came up with many innovations in Neural Networks. He went to the ImageNet competition, and in 2012 he got 96% accuracy which was insane.

4.1.1 Vision Foundation Models

People are trying to apply LLM techniques to images (vision foundation models), just how humans can think in both words and images. Instead of looking at each word, think of "filling in the blanks" of the image by removing some small parts of the image and asking the NN to fill it in. How do you train? Well, ChatGPT sampled the entire internet and gave it complete training data to figure out the 32777th word. In reality, supervised and reinforcement learning are the only realistic training strategies. Supervised learning is "monkey see, monkey do," while reinforcement learning is "doggy do good thing, doggy gets biscuit." Unsupervised learning didn't seem like real learning until the "fill in the blank" concept happened: self-supervised learning. The internet creates its own "fill in the blank" puzzles for the Neural Network to solve. Google London is working on this a lot.

4.1.2 "Attention is all you need"

keep spamming the attention block throughout the paper. Attention block: sharpen a probability distribution by increasing the high probability values and attenuating the low probability values. You can also do blunting, which is the opposite. These are calculated based on the Temperature parameter, $T = 0$ is sharpening to the extreme case, $T = \infty$ is blunting to uniform case. We want somewhere in the middle. This is based on the Boltzmann definition of the temperature. "softargmax" formula:

$$q_i = \frac{e^{p_i/T}}{\sum_i e^{p_i/T}} := \frac{e^{p_i/T}}{Z}.$$

if $0 \leq T \leq 1$ it sharpens, if $1 \leq T \leq \infty$ it blunts it.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$$

4.1.3 Hallucinations

Why did it take so long from 2017 to make AI a big deal?

AI is really wasteful for power consumption, every single ChatGPT query uses so much Energy. So people decided that it's way too computationally expensive to publish AI because there aren't enough computers to supply the world. Meanwhile, Sam Altman tried to be like "500 users limited edition," and other companies were pressured to follow suit, the only company who played their cards well was Microsoft funding OpenAI at arm's length, keeping OpenAI responsible in case it doesn't work and making Microsoft profitable if it works. Big challenge of AI: Try to reduce power consumption by 10^6. Back to hallucinations: people expect that ChatGPT is some divine power that knows everything, but in reality it's just a monkey pulling words out of a jar. BUT HALLUCINATIONS ARE INTENDED, IT'S PROBABILITY ONLY!!!

4.1.4 RAG (Retrieval Augmented Generation)

It's a hack that tries to lower hallucination:

- Looks on google for some documents
- "Based on the above documents, what is Boltzmann's role ..."

4.1.5 General Comments

There is a science in Data Science: Statistics in new garb, instead of μ and σ, Data Science uses billions of parameters. While Statisticians were very cautious about making mistakes.

- Sampling Bias
- You have to call it out in advance to say something is lucky, machine learning guys do cross-validation by holding back some of the data, then checking to see if it works

5 More Source Coding

5.1 Shannon-Fano-Elias Code (10.11.24)

Instead of using pdf, we use cumulative distribution. If $p(x) > 0$, then define $F(x) = \sum_{a \le x} p(a)$ for all x. Looking at the graph vertically, you can think of it as a partition of the unit interval, since $0 \le F(x) \le 1$. Define

$$\bar{F}(x) := \frac{1}{2}p(x) + \sum_{a<x} p(a).$$

$\bar{F}$ is surjective, hence we can turn it into a code. But, $\bar{F}$ is real, which requires infinite bits!!! So, if we use an approximate value of $\bar{F}$, how much accuracy do we need? $\bar{F}$ just gives the point in the middle of the ranges for each codeword, which is the optimal point to "aim for" with our binary decimal approximation.

Truncate $\bar{F}$ to $\ell(x)$ bits:

$$\lfloor \bar{F}(x) \rfloor_{\ell(x)}.$$

We use the first $\ell(x)$ bits of $\bar{F}(x)$ as a code for x. Clearly,

$$\bar{F}(x) - \lfloor \bar{F}(x) \rfloor_{\ell(x)} < \frac{1}{2^{\ell(x)}}.$$

If $\ell(x) = \lceil \log \frac{1}{p(x)} \rceil + 1$, then

$$\frac{1}{2^x} < p(x)/2 = \bar{F}(x) - F(x-1).$$

This lets you decode. Thus, $\ell(x)$ bits is enough to describe x. We also want prefix-free codes. Consider a codeword

$$z_1 z_2 \dots z_\ell \in \{0,1\}^\ell$$

and an interval corresponding to it of

$$\left[0.z_1 z_2 \dots z_\ell, 0.z_1 z_2 \dots z_\ell + \frac{1}{2^{-(\ell+1)}} \right].$$

The lower end of the interval is in the lower half of the step and the upper half is below the top of the step. Thus, this interval lies within the entire step.

x	$p(x)$	$F(x)$	$\bar{F}(x)$	binary$[\bar{F}(x)]$	$\ell(x)$	codeword
1	0.25	0.25	0.125	0.001	3	001
2	0.5	0.75	0.5	0.10	2	10
3	0.125	0.875	0.8125	0.1101	4	1101
4	0.125	1	0.9375	0.1111	4	1111

.

This code isn't very good, for example because the last digit is pointless. Average length: 2.75, Entropy=1.75. Huffman code achieves this entropy so it is optimal.

We want to bound the average length of the codeword:

$$L(x) = \sum p(x)\ell(x) = \sum p(x)\left(\left\lceil \log \frac{1}{p(x)} \right\rceil + 1\right) < H(X) + 2$$

5.1.1 Arithmetic Coding

Associate each codeword to a subinterval of $[0,1]$ $[a,b)$. Choose a number in the subinterval and encode the number in binary. Arithmetic codes become better when you have entire words as opposed to separate symbols. Also, if the source pdf changes, we dont have to completely rewrite the code.

Visualizing Arithmetic Codes Ruler Intervals: Focus on one inch of a ruler and put a tickmark at the halfway point, splitting it into two intervals $\left[0,\frac{1}{2}\right)$ and $\left[\frac{1}{2},1\right)$, or, in binary $[0,0.1)$ and $[0.1,1)$ in binary. Then split each interval again in half:

$$[0,0.01) \qquad [0.01,0.10) \qquad [0.10,0.11) \qquad [0.11,1),$$

making the tick marks smaller than the half tick. Repeat forever, so that 1/2 has tallest tickmarks, 1/4 and 3/4 have the next tallest, etc.

Examples Given $[0.52, 0.92)$, the tallest tickmark is at 0.75. In general, where is the tallest tickmark for any interval $[a,b)$. If $b-a \geq \frac{1}{2^m}$, then the tallest tickmark has the same height of $\frac{1}{2^m}$, where

$$m \leq \log \left\lceil \frac{1}{b-a} \right\rceil.$$

you get the idea

Given an interval $[a,b) \in [0,1)$, what is the fattest interval in it. Given $[0.52, 0.92)$, we don't a half or quarter inch interval fully contained in it, we only have intervals of an eighth wide:

$$[0.625, 0.75), [0.75, 0.875).$$

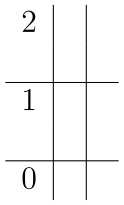
We take the smallest tallest interval:

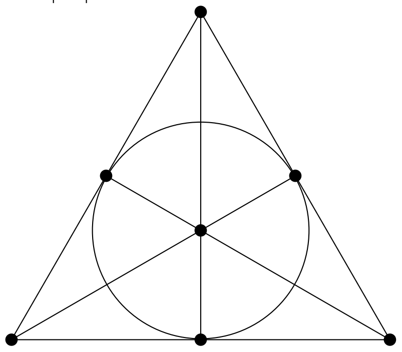
$$m = \left\lceil \frac{2}{b-a} \right\rceil = \left\lceil \log \frac{1}{b-a} \right\rceil + 1$$

The smallest tallest tickmark corresponds to the slimmest fattest interval that is completely contained in $[a, b)$.

x	$p(x)$
0	0.2
1	0.4
2	0.4

Let's try to encode 210. We split the unit interval into 0.2, 0.4, 0.4, and then split each subintervale into 0.2, 0.4, 0.4 of it, and so on.





5.2 Arithmetic Codes + Conditional Probability (10.15.2024)

Review Fano Code

x	$p(x)$	$F(x)$	$\bar{F}(x)$	$\bar{F}(x)$ in binary	$\ell(x)$	cw
1	0.25	0.25	0.125	0.0001	3	000
2	0.25	0.5	0.375	0.011	3	001
3	0.2	0.7	0.6	$0.1\overline{0011}$	4	1001
4	0.15	0.85	0.775	$0.110\overline{0011}$	4	1100
5	0.15	1	0.925	$0.111\overline{0110}$	4	1110

$H(X) = 1.891$.

Back to Arithmetic Codes: Let

$$X \in \{0, 1, 2\}$$

and

$$p = \{p_0, p_1, p_2\} = \{0.2, 0.4, 0.4\}.$$

We want to send 210. Split up the interval: the interval of 2 is $[0.6, 1)$. The interval of 21 is $[0.6 + 0.4 \cdot 0.2, 0.6 + 0.4 \cdot 0.6) = [0.68, 0.84)$. The interval of 210 is $[0.68, 0.68 + 0.2 \cdot 0.4 \cdot 0.4) = [0.68, 0.712)$.

Now we want to encode this interval:

$$\frac{0.712 + 0.68}{2} = 0.696 \approx 0.101100_b,$$

truncating to $\lceil \log(1/0.032) + 1 \rceil$ bits. In other words, we find (the lower bound of) the largest binary interval that is fully contained in $[0.68, 0.712)$.

Example #2

Let

$$X \in \{0, 1, 2, 3\}$$

and

$$p = \{0.05, 0.05, 0.5, 0.4\}.$$

The message we want to send is 2320, which lies in the interval $[0.42, 0.425)$. The length of our code word will be $\lceil -\log(0.5 \cdot 0.4 \cdot 0.5 \cdot 0.05) \rceil + 1 = 9$. The midpoint of the interval, 0.4225, encoded in binary and truncated to nine bits is 011011000.

Optimality

Let $X \in \{0, 1, ..., m\}$ and $p = \{p_0, p_1, ..., p_m\}$. Consider the probability distribution of all words $X_1, X_2, \ldots$. We have

$$\ell(x) \leq \left\lceil \log \frac{1}{p^*(x)} \right\rceil + 1 = \ell_{sh}(x) + 1,$$

where $p^*(x) = p_{x_1} p_{x_2} \ldots p_{EOF}$ is the length of the interval. So the expected length

$$L^* = \sum \ell(x) p^*(x) \leq L_{sh} + 1 < H(X) + 2.$$

So it's pretty good.

6 Conditional Probability

6.1 Conditional Prob (2024.10.17)

Definition: Given two events E and F, the probability that E occurs given that F occurs is

$$P(E|F) = \frac{P(E \wedge F)}{P(F)}$$

where $P(E \wedge F)$ is the probability that both E and F occur.

e.g. probability that 3 tosses are heads given at least one head:

$$P = \frac{P(\text{3 tosses and at least 1 head})}{P(\text{at least one head})} = \frac{1/8}{7/8} = \frac{1}{7}.$$

Bridge Example : North and South vs East and West. We know that North and South have 8 spades between them. What is the probability that East has 3 of the remaining spades.

$$P = \frac{\binom{5}{3}\binom{21}{10}}{\binom{26}{13}}.$$

Extension:

$$P(E_1 E_2 \ldots E_n) = P(E_1) P(E_1|E_2) P(E_3|E_2 E_1) \cdots P(E_n|E_1 E_2 \ldots E_{n-1}).$$

Let us have 52 cards that are randomly split into 4 piles of 13 cards each. What is the probability that each pile has exactly one Ace?

E_1 : ace of spades in one of the piles $\quad$ E_2 : ace of spaces and hearts are in different piles . . . .

$$\begin{aligned} P(E_1) &= 1. \\ P(E_2|E_1) &= \frac{39}{51}. \\ P(E_3|E_1E_2) &= \frac{26}{50}. \\ P(E_4|E_1E_2E_3) &= \frac{13}{49} \end{aligned}$$

Go multiply.

6.1.1 Bayes' Rule

$$\begin{aligned} P(E) &= P(EF) + P(EF^C) \\ &= P(E|F)P(F) + P(E|F^C)P(F^C). \end{aligned}$$

This is the law of the excluded middle lol.

$$P(A|B)P(B) = P(B|A)P(A) = P(\text{A and B}) = P(\text{B and A})$$

Example A lab blood test is 95% effective in detecting disease when actually present, and has a 1% false positive rate for healthy patients. If 0.5% of the population actually have the disease, what is the probability that a person has the disease given a positive test?

Remembering $P(T) = P(T|D)P(D) + P(T|D^C)P(D^C)$:

$$P(D|T) = \frac{P(T|D)P(D)}{P(T)} = \frac{0.95 \cdot 0.005}{0.005 \cdot 0.95 + 0.995 \cdot 0.01} = \boxed{32.3\%}$$

6.1.2 Independent Events

If and only if E and F are independent events, then $P(E|F) = P(E)$ and $P(F|E) = P(F)$. Then, $P(\text{E and F}) = P(E)P(F)$.

Do not confuse "mutually exclusive" with "independent!" They are actually opposites.

6.1.3 Joint Distributions

$$\begin{aligned} p(x,y) &= P(X = x \,\&\&\, Y = y) \\ F(a,b) &= P(X \leq a \,\&\&\, Y \leq b) \end{aligned}$$

This is 2D instead of 1D: we can make a table to represent the joint distribution, and we should double sum/double integrate.

$$p_x(x) = P(X = x) = \sum_y p(x,y)$$

$$p_y(y) = P(Y = y) = \sum_x p(x, y)$$

Consider a set of three red balls, four white balls, and five blueballs Choose 3 balls at random, and let X be the number of Red balls and Y the number of white balls.

$x \downarrow y \rightarrow$	0	1	2	3	margin
0					
1	$\frac{\binom{3}{1}\binom{5}{2}}{\binom{12}{3}}$	$\frac{\binom{3}{1}\binom{4}{1}\binom{5}{1}}{\binom{12}{3}}$	$\frac{\binom{3}{1}\binom{4}{2}}{\binom{12}{3}}$	0	add them up
2					
3					

Looking at $p_x(x)$, we sum the table along the rows y so its the marginal thing.

$$P(x \in A, y \in B) = \int_B \int_A f(x, y)\,\mathrm{d}x\,\mathrm{d}y.$$

$$F(a, b) = \int_{-\infty}^{b} \int_{-\infty}^{a} f(x, y)\,\mathrm{d}x\,\mathrm{d}y.$$

$$f(a, b) = \frac{\partial^2}{\partial a \partial b} F(a, b).$$

$$f_x = \int f(x, y)\,\mathrm{d}y, f_y = \int f(x, y)\,\mathrm{d}x.$$

Example a)

$$P(X > 1, Y > 1) = \int_1^\infty \int_1^\infty 2e^{-x}e^{-2y}\,\mathrm{d}y\,\mathrm{d}x = \frac{1}{e^3}.$$

b)

$$P(X < Y) = \int_0^\infty \int_0^y f(x, y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{3}.$$

c)

$$P(X < a) = \int_a^\infty \int_0^\infty f(x, y)\,\mathrm{d}y\,\mathrm{d}x = 1 - e^{-a}.$$

$X : f_x, Y : f_y$, what is the pdf of $X + Y$?

$$F_{x+y} = P(X + Y \leq a) = \int_{-\infty}^{\infty} \int_{-\infty}^{a-y} f_x(x) f_y(y)\,\mathrm{d}x\,\mathrm{d}y = \int_{-\infty}^{\infty} F_x(a - y) f_y(y)\,\mathrm{d}y.$$

CONVOLUTION!!!

6.2 Conditional Expectation and Distributions (10.23.24)

Exercise Choose a point randomly in a circle with radius R such that all regions within the circle are equally likely.

$$f(x, y) = \begin{cases} c & x^2 + y^2 \leq R^2 \\ 0 & \text{otherwise.} \end{cases}$$

$$c = \frac{1}{\pi R^2}.$$

We have the marginal distributions

$$f_x = \int_{-\infty}^{\infty} f(x,y)\,\mathrm{d}y = 2c\sqrt{R^2-x^2} = \frac{2\sqrt{R^2-x^2}}{\pi R^2}.$$

Similarly,

$$f_y = \frac{2\sqrt{R^2-y^2}}{\pi R^2}.$$

The probability that the distance from the origin is $\leq a$ is

$$\frac{a^2}{R^2}.$$

The expected value of the distance is

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \sqrt{x^2+y^2} f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{\pi R^2}\int_0^R\int_0^{2\pi} r^2\,\mathrm{d}\theta\,\mathrm{d}r = \frac{1}{\pi R^2}\frac{2\pi R^3}{3} = \frac{2}{3}R.$$

If you have two independent Random Variables X and Y, we want to find out f_{x+y}, which is

$$\frac{\mathrm{d}}{\mathrm{d}a}F_{x+y}(a) = \int_{-\infty}^{\infty}\frac{\partial}{\partial a}F_x(a-y)f_y(y)\,\mathrm{d}y = \int_{-\infty}^{\infty} f_x(a-y)f_y(y)\,\mathrm{d}y = f_x * f_y.$$

This is only for independent variables.

The sum of two poisson random variables is another poisson random variable, where the λ rates of the R.V.s just add together. Intuition of Poisson point process.

6.2.1 Conditional Distributions

Discrete:

$$p_{X|Y}(x|y) = P(X = x \,|\, Y = y) = \frac{P(X=x, Y=y)}{P(Y=y)}.$$

The cumulative mass function is

$$P_{X|Y}(x|y) = P(X \leq x \,|\, Y = y) = \sum_{a\leq x} p_{X|Y}(a|y).$$

Example Let

$$P(X=0,Y=0)=0.4, P(X=0,Y=1)=0.2, P(X=1,Y=0)=0.1, P(X=1,Y=1)=0.3.$$

Given $Y=1$, then

$$P(X=0|Y=1)=0.4, P(X=1|Y=1)=0.6.$$

Example 2 Let X and Y be independent poisson R.V. with λ_1 and λ_2. Find the conditional distribution of X given $X+Y=n$. Since the sum of two independent Poissons is a poisson with $\lambda = \lambda_1 + \lambda_2$.

$$P(X=k, Y=n-k) = \frac{(\lambda_1)^k}{k!} e^{-\lambda_1} \frac{(\lambda_2)^{n-k}}{(n-k)!} e^{-\lambda_2}$$

$$\begin{aligned} P(X+Y=n) &= \sum_{k=0}^{n} P(X=k, Y=n-k) \\ &= \sum_{k=0}^{n} \frac{(\lambda_1)^k (\lambda_2)^{n-k}}{k!(n-k)!} e^{-\lambda_1-\lambda_2} \\ &= \sum_{k=0}^{n} \binom{n}{k} \frac{(\lambda_1)^k (\lambda_2)^{n-k}}{n!} e^{-\lambda_1-\lambda_2} \\ &= \frac{(\lambda_1+\lambda_2)^n}{n!} e^{-\lambda_1-\lambda_2} \end{aligned}$$

$$\begin{aligned} P(X=k|X+Y=n) &= \left(\frac{(\lambda_1)^k (\lambda_2)^{n-k}}{k!(n-k)!} e^{-\lambda_1} e^{-\lambda_2} \right) \Big/ \left(\frac{(\lambda_1+\lambda_2)^n}{n!} e^{-\lambda_1-\lambda_2} \right) \\ &= \frac{\binom{n}{k} \lambda_1^k \lambda_2^{n-k}}{(\lambda_1+\lambda_2)^n} \\ &= \binom{n}{k} \left(\frac{\lambda_1}{\lambda_1+\lambda_2} \right)^k \left(\frac{\lambda_2}{\lambda_1+\lambda_2} \right)^{n-k}. \end{aligned}$$

You can verify that this sums to one by binomial.

Continuous:

$$f_{X|Y}(x|y) = \frac{f(x,y)}{f_y(y)}.$$

Exercise

$$f(x,y = \begin{cases} \frac{12}{5}x(2-x-y) & 0<x<1, 0<y<1 \\ 0 & \text{otherwise.} \end{cases}$$

We have

$$f_{X|Y}(x|y) = \frac{f(x,y)}{f_y(y)}.$$

$$f_y(y) = \int_0^1 f(x,y)\,\mathrm{d}x = \frac{8}{5} - \frac{6}{5}y$$

$$f_{X|Y}(x|y) = \frac{\begin{cases} \frac{12}{5}x(2-x-y) & 0<x<1, 0<y<1 \\ 0 & \text{otherwise} \end{cases}}{\frac{8}{5} - \frac{6}{5}y} = \frac{6x^2 - 12x + 6xy}{3y-4} \mathbb{1}_{(0,1)\times(0,1)}.$$

6.3 More Conditional Stuff 10.25.2024

For two random variables X and Y, with join distribution $p(x, y)$, we can easily find $E[X]$ and $E[Y]$.

We want to find the covariance between X and Y. Recall that with $\mu_x = E[X]$, we have

$$Var[X] = E[(X - \mu_x)^2].$$

We define the covariance of X and Y to be the following, with $\mu_x = E[X], \mu_y = E[Y]$.

$$\begin{aligned}\text{Cov}(X, Y) &= E[(X - \mu_x)(Y - \mu_y)] \\ &= E[XY - \mu_x Y - \mu_y X + \mu_x\mu_y] = E[XY] - \mu_x\mu_y.\end{aligned}$$

Properties of Covariance See `https://en.wikipedia.org/wiki/Covariance#Relationship_to_inner_products`. The covariance is bilinear and symmetric.

$$\text{Cov}(X, Y) = \text{Cov}(Y, X),$$

$$\text{Cov}(X, X) = \text{Var}[X].$$

Consider $X_1, X_2, \ldots, X_n$ and $Y_1, Y_2, \ldots, Y_n$. Then

$$\text{Cov}\left(\sum_i X_i,\ \sum_j Y_j\right) = \sum_i\sum_j \text{Cov}(X_i, Y_j).$$

Also,

$$\text{Cov}\left(\sum_i X_i,\ \sum_i X_i\right) = \sum_i \text{Var}X_i + 2\sum_{i<j}\sum \text{Cov}(X_i, X_j).$$

$$\text{Var}\left(\sum_i X_i\right) = \sum_i \text{Var}(X_i)$$

if the X_i are independent.

Conditional Expectation:

$$\begin{aligned}E[X|Y = y] &= \sum_x xP(X = x|Y = y) = \sum_x p_{x|y} \\ &= \int x f_{x|y}\, \mathrm{d}x. \qquad \text{(continuous)}\end{aligned}$$

Example 1 Let X and Y be iid binomial R.V.'s, both with parameters n and p. Find

$$E[X|X + Y = m].$$

$$E[X|X + Y = m] = \sum_{i=0}^{n} k\frac{\binom{n}{i}\binom{n}{m-i}}{\binom{2n}{m}} = \frac{m}{2}.$$

This is a hypergeometric distribution.

Exercise 2

$$f(x,y) = \frac{e^{-x/y}e^{-y}}{y}.$$

With $x, y > 0$.

$$E[X|Y=y] = \int_0^\infty x \frac{f(x,y)}{\int_0^\infty f(x,y)\,\mathrm{d}x}\,\mathrm{d}x = y.$$

6.3.1 Joint & Conditional Entropy

$$H(X,Y) = -\sum_x \sum_y p(x,y)\log p(x,y) = -E_{p(x,y)}[\log p(x,y)].$$

$$\begin{aligned} H(Y|X) &= \sum_x p(x) H(Y|X=x) \\ &= -\sum_x p(x) \sum_y p(y|x)\log p(y|x) \\ &= -\sum_x \sum_y p(x,y)\log \frac{p(x,y)}{p(x)} \end{aligned}$$

Chain Rule

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y).$$

$$\begin{aligned} H(X,Y) &= -\sum_x \sum_y p(x,y)\log p(x,y) \\ &= -\sum_x \sum_y p(x,y)(\log p(x) + \log p(y|x)) \\ &= -\sum_x \sum_y p(x,y)\log p(x) - \sum_x \sum_y p(x,y)\log p(y|x) \\ &= H(X) + H(Y|X). \end{aligned}$$

By symmetry, this is also

$$H(Y) + H(X|Y).$$

Relative Entropy "Distance" between two distributions. See `https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence`: it is the difference between cross-entropy and entropy.

$$D(p||q) = \sum p(x)\log \frac{p(x)}{q(x)}.$$

6.4 Relative Entropy and Mutual Information (10.30.2024)

$y \downarrow x \rightarrow$	1	2	3	4	$\sum$
1	2^{-3}	2^{-4}	2^{-5}	2^{-5}	2^{-2}
2	2^{-4}	2^{-3}	2^{-5}	2^{-5}	2^{-2}
3	2^{-4}	2^{-4}	2^{-4}	2^{-4}	2^{-2}
4	2^{-2}	$2^{-\infty}$	$2^{-\infty}$	$2^{-\infty}$	2^{-2}
$\sum$	2^{-1}	2^{-2}	2^{-3}	2^{-3}	2^{0}

a $H(X) = 7/4$
b $H(Y) = 2$
c $H(X|Y) = 1.375$
d $H(Y|X) = 1.625$
e $H(X,Y) = H(X|Y) + H(Y) + H(Y|X) + H(X) = 3.375$

KL DIVERGENCE IS NOT SYMMETRIC But, it can measure how close these two distribtutions are, but it measures how inefficient q is w.r.t. p, so if you knew p, then q represents how inefficient a coding of q would be.

$$\begin{aligned} D(p||q) &= H_x(p||q) - H(p) \\ &= \sum p(x) \log \frac{1}{q(x)} - \sum p(x) \log \frac{1}{p(x)} \end{aligned}$$

The first term is the average number of bits created from sampling from the distribution of $p(x)$ if you had thought the distribution were actually $q(x)$ (cross-entropy). Thus, you can think of KL divergence as the number of bits you would waste if you used a Huffman coding for $q(x)$ to encode $p(x)$.

6.4.1 Mutual Information

$$I(X;Y) = \sum \sum p(x,y) \log \frac{p(x,u)}{p(x)p(y)} = D(p(x,y)||p(x)p(y)).$$

Note that if x and y are indpendent, then $p(x,y) = p(x)p(y)$, so $I(X;Y)$ would be zero, as expected. I represents how inefficient you are if you assume they are independent but they actually aren't.

Relation between Mutual Information and Entropy:

$$\begin{aligned} I(X;Y) &= \sum p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\ &= \sum p(x,y) \log \frac{p(x|y)p(y)}{p(x)p(y)} \\ &= \sum \log p(x|y) - \sum p(x,y) \log p(x) \\ &= -\sum p(x) \log p(x) - \left[-\sum p(x,y) \log p(x|y) \right] \\ &= H(X) - H(X|Y) \\ &= H(X) + H(Y) - H(X,Y). \end{aligned}$$

6.5 More Mutual Information (11.01.2024)

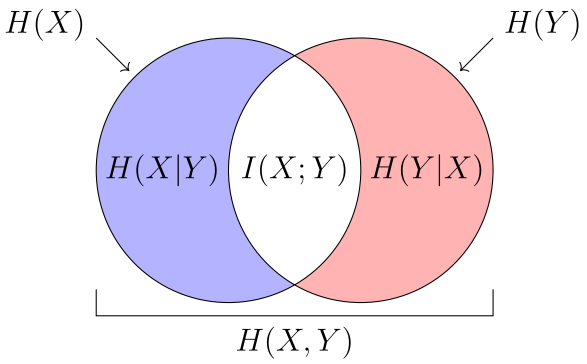
$$D(p||q) = \sum p(x) \log \frac{p(x)}{q(x)}.$$

$$H(X,Y) = \sum p(x,y) \log \frac{1}{p(x,y)}.$$

$$H(Y|X) = -\sum p(x) H(Y|X=x) = \sum p(x,y) \log \frac{1}{p(y|x)}.$$

$$H(X,Y) = H(X) + H(Y|X).$$

$$I(X;Y) = I(Y;X) = \sum p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = D(p(x,y)||p(x)p(y)) = H(X) - H(X|Y).$$



1. If $X_1, X_2, \dots, X_n$ are identical, then (Faraday's law of induction):

$$H(X_1, X_2, \dots, X_n) = \sum_{i=1}^{n} H(X_i \big| [X_{i-1}, \dots, X_1]).$$

2. (Square brackets denote grouping for readability.)

$$I([X;Y] \big| Z) = H(X \big| Z) - H(X \big| [Y,Z]).$$

3. $$D(p(x,y)||q(x,y)) = D(p(x)||q(x)) + D(p(y|x)||q(y|x)).$$

Theorem: Kullback-Leibler divergence is positive ($D(p||q) \geq 0$)

$$D(p||q) = \sum p(x) \log \frac{1}{q(x)} - H(p(x)).$$

This is the same as proving Gibb's inequality because KL divergence is the difference between cross-entropy and entropy. Lagrange multipliers:

$$\Phi = \sum p_i \log \frac{1}{q_i} - \lambda \sum_i q_i,$$

$$\frac{\partial \Phi}{\partial q_i} = -\frac{p_i}{q_i \ln(2)} - \lambda = 0,$$

$$q_i = -\frac{p_i}{\lambda \ln(2)}.$$

$$\sum_i q_i = -\sum_i \frac{p_i}{\lambda \ln(2)} = 1,$$

so $q_i = p_i$, where

$$\sum_i p(i) \log \frac{1}{q_i} = H(p)$$

is a minimum, so

$$D(p||q) \geq 0.$$

$$\begin{aligned} -D(p||q) &= \sum p(x) \log \frac{q(x)}{p(x)} \\ &\leq \log \sum p(x) \frac{q(x)}{p(x)} = \log 1 = 0 \end{aligned}$$

This also means that

$$I(X;Y) = D(p(x,y)||p(x)p(y)) \geq 0,$$

and equality occurs iff X and Y are independent.

Example

$$H(X) \leq \log |\mathscr{X}|,$$

where

$$|\mathscr{X}|$$

denotes the number of elements in X with equality iff X is uniform. If

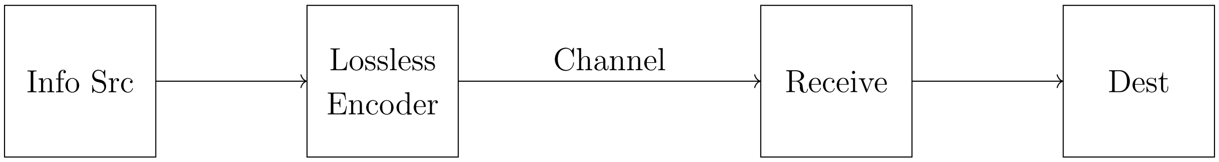
$$u(x) = \frac{1}{|X|},$$

and $p(x)$ is the mass function of X, then

$$D(p||u) = \log |\mathscr{X}| - H(X) \geq 0,$$

so QED.

6.5.1 Channel Capacity



Think of waveforms. Let us have three different kinds of waveforms we can send, corresponding to 0, 1, 2. But, the channel will add noise to these waves. Let the input to the channel be X and the output be Y:

$$P(Y|X = x_1), P(Y|X = x_2), P(Y|X = x_3).$$

1. Send something through the channel. Before it comes out of the pipe, I ask how surprised you would be if $X = x$ was sent, this would be
$$\log \frac{1}{p(x)}.$$
2. When it comes out, we find that $Y = y$. Now, how surprised are you? this would be
$$\log \frac{1}{p(x|y)}.$$
3. The information you gain from seeing y come out is
$$\log \frac{1}{p(X = x)} - \log \frac{1}{P(X = x|Y = y)}.$$
4. In the long run,
$$\begin{aligned} &E_{x,y}\left[\log \frac{1}{P(X = x)} - \log \frac{1}{P(X = x, Y = y)}\right] \\ &= \sum p(x, y) \log \frac{p(x|y)}{p(x)} \\ &= \sum p(x, y) \log \left(\frac{p(x|y)}{p(x)} \frac{p(y)}{p(y)}\right) = I(X; Y). \end{aligned}$$

$$C(Y|X)$$

is the notation for channel capacity, and it is defined by the maximum of $I(X; Y)$ over X and Y.

7 Channels

7.1 Channel Capacity

Recall that
$$C(Y|X) = \sup\{I(X; Y)\}_{X,Y}.$$

Example 1 Noiseless Binary Channel

$$p : 0 \to 0 \qquad (1-p) : 1 \to 1$$

$$I(X;Y) = \sum\sum p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = H(p),$$
$$\therefore C(X|Y) = \sup\{h(p) : p \in [0,1]\} = 1\text{bit},$$

when X is a uniform distribution.

Example 2 Noisy channel with non-overlapping outputs.

X	Y
1	1 ($p = 1/2$)
	2 ($p = 1/2$)
2	3 ($p = 1/3$)
	4 ($p = 2/3$)

$$I(X;Y) = \sum\sum p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = H(X) - H(X|Y) = H(X) = H(p)$$

because $H(X|Y) = 0$ since they are disjoint, and thus

$$C(Y|X) = 1\text{bit}.$$

Example 3 Let $A \to A$ with probability 1/2 and B with probability 1/2, etc. to $Z \to Z_{0.5}, A_{0.5}$

We have

$$\begin{aligned}
I(X;Y) &= \sum\sum p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\
&= \sum\sum_{x=y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} + \\
&\sum \sum_{x+1 \equiv y \pmod{26}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\
&= \sum\sum_{x=y} \frac{1}{2} p_i \log \frac{p_i/2}{p_i \cdot (p_i/2 + p_{i-1}/2)} \\
&+ \sum \sum_{x+1 \equiv y \pmod{26}} \frac{1}{2} p_i \log \frac{p_i/2}{p_i \cdot (p_i/2 + p_{i-1}/2)} \\
&= H(Y) - 1.
\end{aligned}$$

Alternatively,

$$I(X;Y) = H(Y) - H(Y|X) = H(Y) - 1,$$

since it's just a 50/50 given X. But, $H(Y) \leq \log(26)$, and if X is uniform, Y must be as well, so this is attainable.

$$\begin{tikzcd}
0 \arrow[r, "1-p"] \arrow[dr, "p" near start] & 0 \\
1 \arrow[r, "1-p"'] \arrow[ur, "p" near start] & 1
\end{tikzcd}$$

Example 4

$$I(X;Y) = H(Y) - H(Y|X) \leq 1 - H(Y|X) = 1 - H(p),$$

which is achieved when Y and X are both uniform. Alternatively,

$$\begin{aligned}
I(X;Y) &= \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\
&= q(1-p) \log \frac{q(1-p)}{q(q(1-p)+(1-q)p)} \\
&\quad + qp \log \frac{qp}{q(qp+(1-q)(1-p))} \\
&\quad + (1-q)p \log \frac{(1-q)p}{(1-q)(q(1-p)+(1-q)p)} \\
&\quad + (1-q)(1-p) \log \frac{(1-q)(1-p)}{(1-q)(qp+(1-q)(1-p))} \\
&= H(\text{some binary distribution}) - H(p) \\
&\leq \boxed{1 - H(p)}.
\end{aligned}$$

7.2 BEC/Symmetric Channels 11.7.2024

$$\begin{tikzcd}
0 \arrow[r, "1-p"] \arrow[dr, "p"'] & 0 \\
 & ? \\
1 \arrow[r, "1-p"'] \arrow[ur, "p"] & 1
\end{tikzcd}$$

$$\begin{aligned}
I(X;Y) &= \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\
&= q(1-\alpha) \log \frac{q(1-\alpha)}{q^2(1-\alpha)} + q\alpha \log \frac{q\alpha}{q\alpha} + (1-q)\alpha \log \frac{(1-q)\alpha}{(1-q)\alpha} + (1-q)(1-\alpha) \log \frac{(1-q)(1-\alpha)}{(1-q)^2(1-\alpha)} \\
&= (1-\alpha) q \log \frac{1}{q} + (1-\alpha) \log \frac{1}{(1-q)} = (1-\alpha) H(q) \leq 1 - \alpha,
\end{aligned}$$

which occurs when X is uniform.

You might be tempted to do the following:

$$C = \sup\{I(X;Y)\} = \sup\{H(Y) - H(Y|X)\} = \sup\{H(Y) - H(\alpha)\} \neq \log 3 - H(\alpha),$$

since we can't guarantee to create a uniform distribution on the right.

Instead, try the other way:

$$C = \sup\{H(X) - H(X|Y)\} = \sup\{1 - H(X|Y = ?)P(?)\} = 1 - \alpha,$$

which occurs when X is uniform.

7.2.1 Gaussian Channel

This is a continuous channel, so we will be using differential entropy. This channel adds random noise to the input, and this noise is normally distributed. For input X, output $Y = X + Z$ where $Z \sim N(\mu, \sigma^2)$. Without loss of generality, assume $\mu = 0$.

Constraint: "cost" of sending X is $\mathrm{Var}[X] = \sigma_x^2$, otherwise we can send infinite information or something.

Since this must be at best independent,

$$\sigma_y^2 \leq \sigma_x^2 + \sigma_z^2$$

From the entropy of a Gaussian, and the fact that it maximizes differential entropy:

$$\begin{aligned} h(Y) &\leq \frac{1}{2}\log_2 2\pi e \sigma_y^2 \\ &\leq \frac{1}{2}\log_2 2\pi e(\sigma_x^2 + \sigma_z^2). \end{aligned}$$

Because Z and X are independent (i.e. $h(Z|X) = h(Z)$):

$$I(X;Y) = h(Y) - h(Y|X) = h(Y) - h(Z) \leq \frac{1}{2}\log 2\pi \frac{\sigma_x^2 + \sigma_z^2}{\sigma_z^2}.$$

Symmetric Channels We call a channel symmetric if $p(y|x)$ is symmetric as such:

The matrix of $p(y|x)$ is:

$$\begin{bmatrix} 0.3 & 0.2 & 0.5 \\ 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \end{bmatrix}.$$

The entry in the x^{th} row and y^{th} column is the conditional probability $p(y|x)$ that y is received given x is sent. Since all the rows and columns are permutations of each other, we call it symmetric (even though the matrix isn't).

Consider

$$Y = (X + Z) \mod c,$$

where X and Z are some distributions on $\mathbb{Z}/c\mathbb{Z}$ and are independent.

Given

$$p(y|x) : \begin{bmatrix} 0.3 & 0.2 & 0.5 \\ 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \end{bmatrix},$$

find $C(Y|X)$.

$$I(X;Y) = H(Y) - H(Y|X) = H(Y) - H(\{0.3, 0.2, 0.5\}) \le \log 3 - H(\{0.3, 0.2, 0.5\}).$$

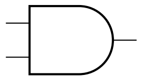
If input is uniform, then Y is uniform as well, so this works.

Computer Architecture Properties of Channel Capacity:

1. Because $I(X;Y) \ge 0$
$$C(Y|X) \ge 0$$
2. Because $I(X;Y) \le H(X), H(Y)$
$$C \le \log |\mathscr{X}| \qquad \text{and} \qquad C \le \log |\mathscr{Y}|$$
3. $I(X;Y)$ is continuous and concave on $p(x)$.

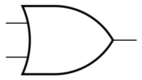
7.3 Comp Arch. 101

AND gate: $z = x \wedge y = x \cdot y = x$ AND y



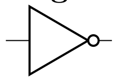
$x(t)$	$y(t)$	$z(t)$
0	0	0
0	1	0
1	0	0
1	1	1

OR gate: $z = x \vee y = x + y = x$ OR y

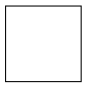


$x(t)$	$y(t)$	$z(t)$
0	0	0
0	1	1
1	0	1
1	1	1

NOT gate: $y = \text{NOT } x = \bar{x} = \sim x$



$x(t)$	$y(t)$
0	1
1	0

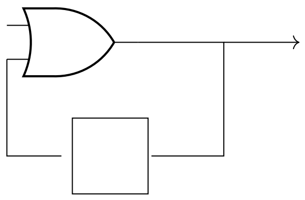
Delay/Flipflop: $y = Dx$  delays output by 1 clock cycle. Only element with a notion of memory. Everything else is memoryless/instantaneous.

Ignore this:

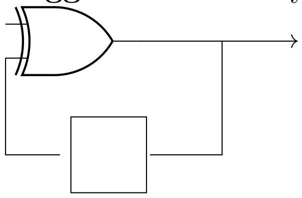
$$D = e^{\frac{\mathrm{d}}{\mathrm{d}x}}$$

$x(t)$	$y(t+1)$
0	0
1	1

Sticky Bit: As long as you see 0, you will output 0. Once you see a 1 you output a 1 and then it remains at 1 forever. $z[t+1] = z[t] \vee x[t]$



Toggle Bit When you see a zero, output the same bit as before, but flip it if you see a 1.



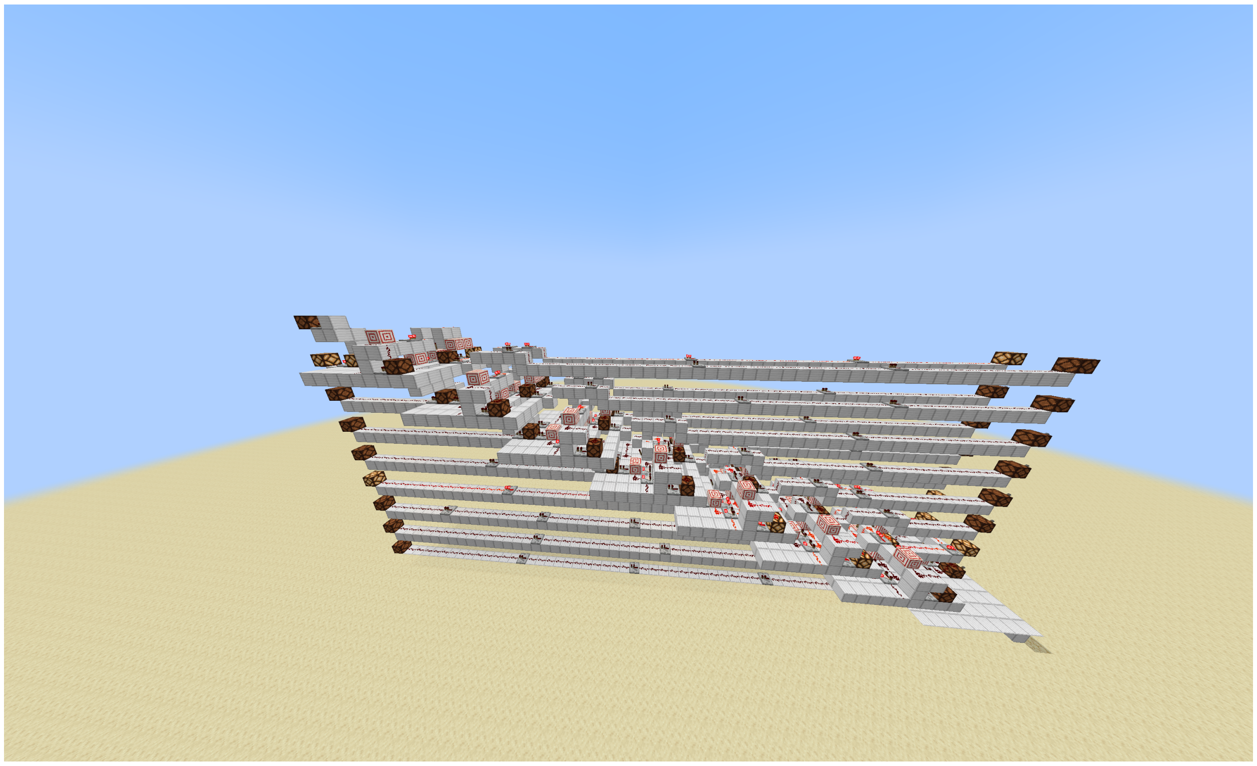
Make sure you understand why these work.

Parallel AND: k bits going to each input, k bits outputted, each correspond bit is ANDed, so 11000 AND 01100 is 01000.

```
uncurry (&&) <$> zip a b
```

Aggregate AND: everything ANDed together.

```
foldr (&&) True $ (a++b)
all id $ (a++b)
```



(Not to be confused with the snake)

k-bit adder: Each layer takes in x_i, y_i, c_i and outputs z_i, c_{i+1} We have

$x_i(t)$	$y_i(t)$	$c_i(t)$	$z_i(t)$	$c_{i+1}(t)$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	1
1	1	1	1	1

So,

$$z_i(t) = \bar{x}_i\bar{y}_i c_i + \bar{x}_i y_i \bar{c}_i + x_i \bar{y}_i \bar{c}_i + x_i y_i c_i.$$

Theorem Given $f : \mathbb{F}_2^k \to \mathbb{F}_2^l$, we can find some memoryless circuit with that does this. This can be done only with AND, OR, and NOT gates.

Define a stream $stream(\mathbb{F}_2^k)$ to be the set of infinite sequences, each element of which is a k-bit block. Then for every function

$$f : stream(\mathbb{F}_2^k) \to stream(\mathbb{F}_2^l),$$

we can build a circuit that implements f using only AND, OR, NOT, FLIPFLOPS, provided that only a finite amount of memory is necessary.

7.4 Hamming Codes and Shift Registers (11.13.2024)

Recall: Source codes try to compress the input to maximize efficiency, but channel codes add redundancy to be able to correct for noise.

There are n basis waveforms, which are combined through the channel. The waveforms are sent at different times to be encodable and decodable (TDM, Time Division Multiplexing). You could also split them in frequency (FDM, Frequency Division Multiplexing).

However, we don't care about this. We will only look at the digital channel coding.

7.4.1 Channel Coding

Example k-bit message: 11001.

Code: 000 111 111 000 000 111 $n = 3k$

k	n	e
arbitrary	$3k$	1 per block of 3

.

Example 2 k bits and a parity bit at the end: counts the number of 1's modulo 2 (even or odd number of 1's correspond to a 0 or 1).

k	n	e
arbitrary	$k + 1$	Only error identification, but not error correction

Example 3 Double Parity:

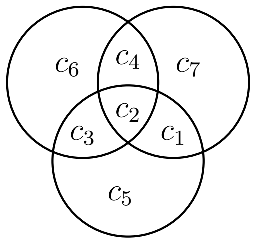
	k_2	parity
k_1	k bits	
parity		x

k	n	e
k_1k_2	$k_1k_2 + k_1 + k_2$	1

7.4.2 Hamming Codes

A good code has a high e and a low n

k	n	e
4	7	1



(Sorry this is upside down from what we did in class but I can't find a venn diagram that is facing the right way online) The message is $c_1c_2c_3c_4$, and the output is $c_1c_2c_3c_4c_5c_6c_7$, where

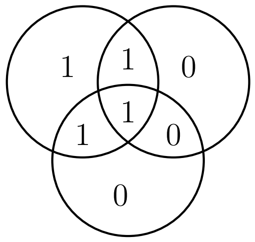
$$c_5 = \text{parity of } c_1c_2c_3 = c_1 \oplus c_2 \oplus c_3$$

$$c_6 = c_2 \oplus c_3 \oplus c_4$$
$$c_7 = c_1 \oplus c_2 \oplus c_4$$

1. 0101 $\rightarrow$ 0101100
2. 1100 $\rightarrow$ 1100010
3. 0000 $\rightarrow$ 0000000
4. 1111 $\rightarrow$ 1111111

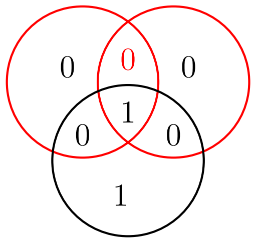
To decode, we put it back on the venn diagram and check if it works.

For 0111010, we have



,

which checks out. However, if we get 0100100, we have

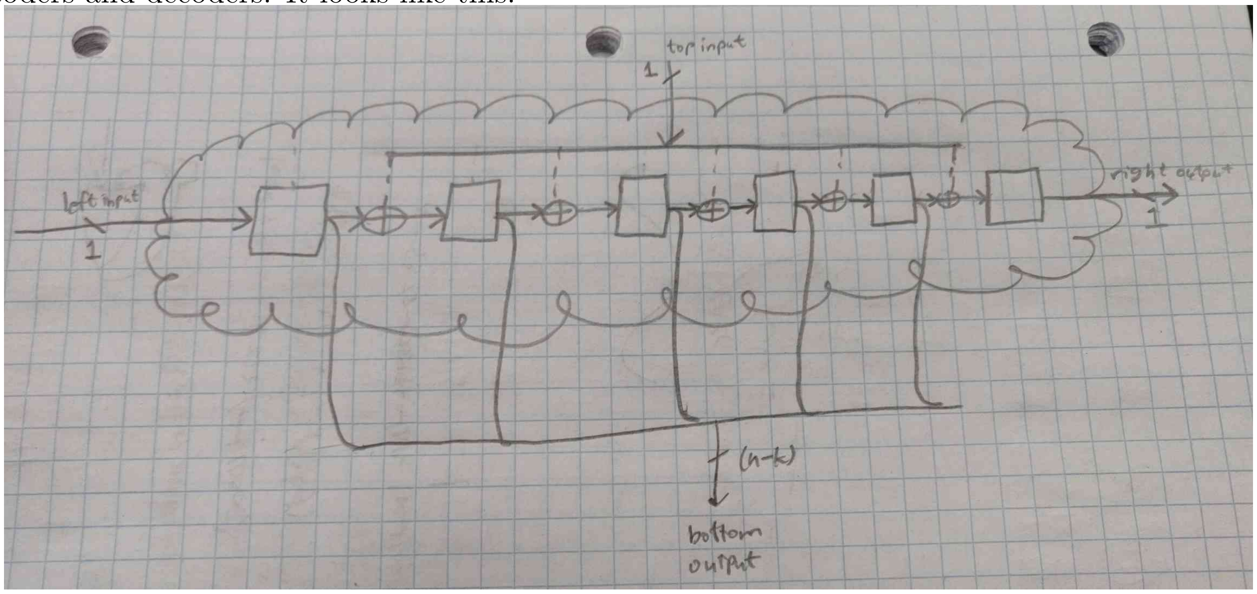


which has only two wrong circles. The only member of the wrong circles that isn't a member of any right circles is c_4, so the correct message must be 0101.

Although we can draw Hamming codes easily with venn diagrams, in general codes are too complicated to do this. So we need shift registers.

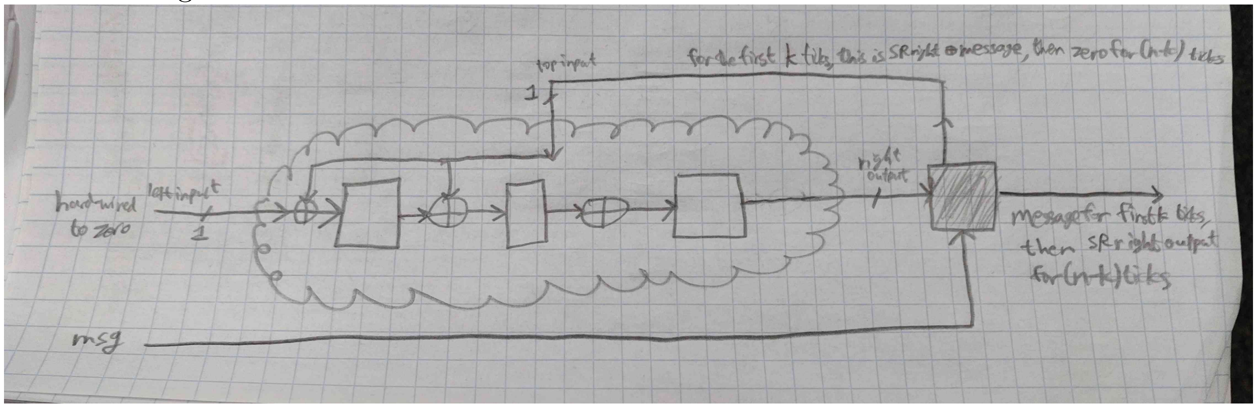
7.4.3 Shift Registers

What is a shift register? A SR is a circuit that forms an important building block in both encoders and decoders. It looks like this:



The length of the SR is arbitrary. The dashed lines do not have to be connected, and using different patterns of the connections produce different codes. We say that the SR has a "tap" in position i if the ith XOR is connected.

For Hamming codes:

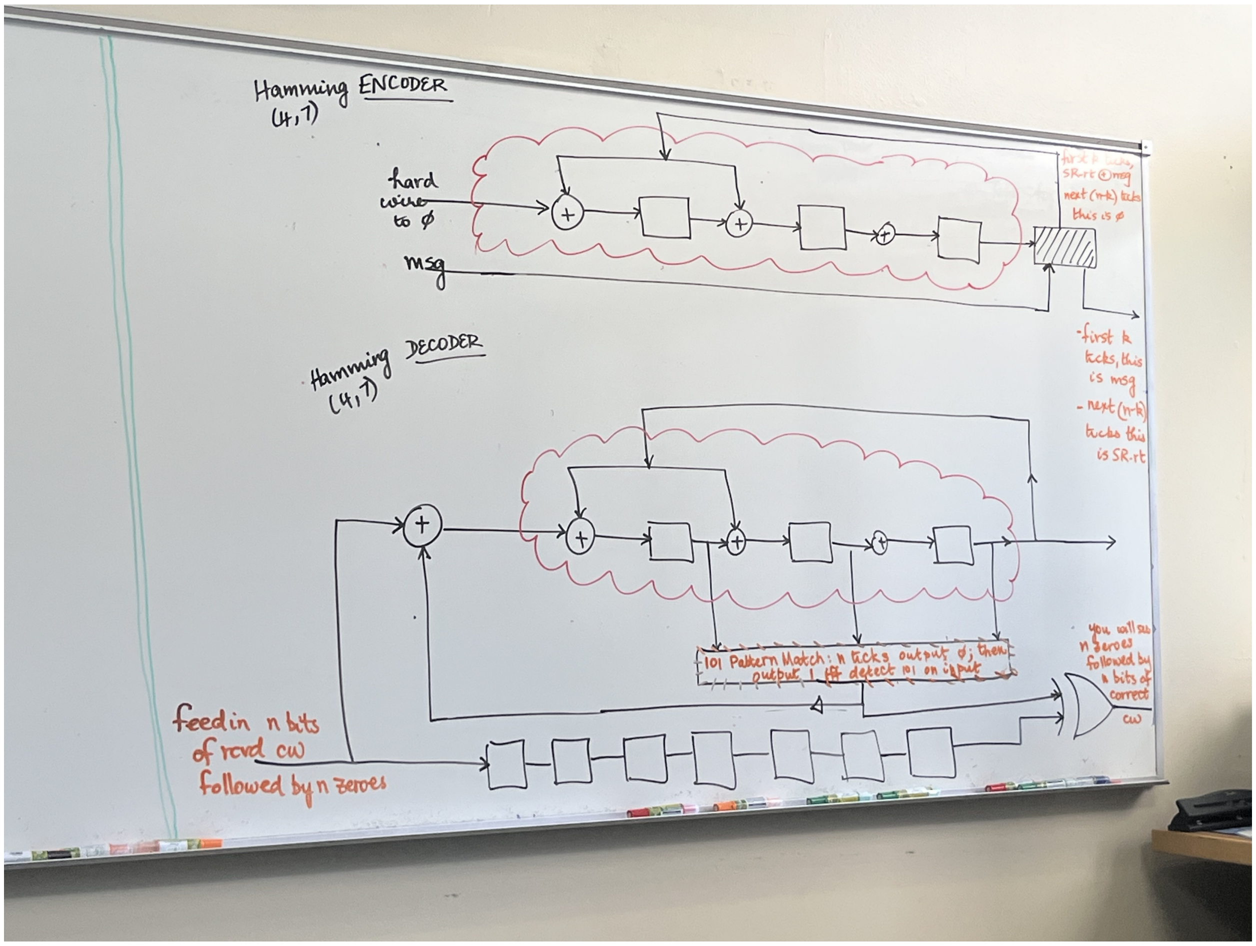

top input
1
for the first k ticks, this is SRright ⊕ message, then zero for (n-k) ticks
hard-wired to zero
left input
1
right output
message for first k ticks, then SR right output for (n-k) ticks
msg

Note that the third XOR isn't tapped (so it doesn't do anything) SR out means the outputs of the internal FlipFlops to keep track of the state throughout the process (SR bottom). For clarity, clock ticks happen between steps, so SROut denotes the state of the system before the step has been propagated.

Our message is 1010, read from right to left, i.e. the order read will be $0, 1, 0, 1$, and then after that it will read zeros only.

Step	SR Out (3 bits)	Message (1 bit)	SR In/Top (1 bit)	Codeword (1 bit)
1	000	0	0	0
2	000	1	1	1
3	110	0	0	0
4	011	1	0	1
5	001	0	0	1
6	000	0	0	0
7	000	0	0	0

7.5 Hamming Encoding and Decoding (11.15.2024)

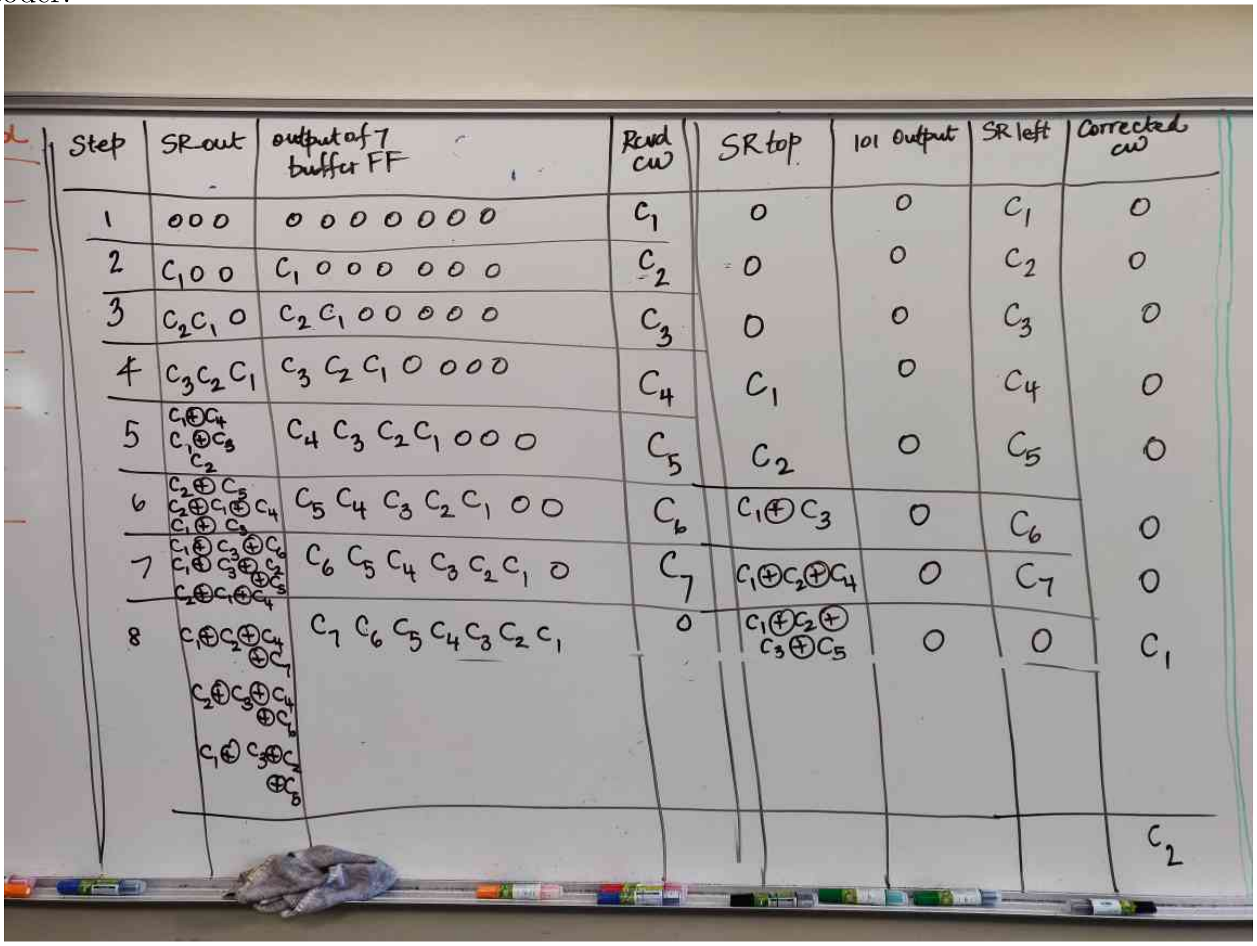

Hamming ENCODER (4,7)
hard wire to ∅
msg
first k ticks, SR-rt ⊕ msg
next (n-k) ticks this is ∅
-first k ticks, this is msg
- next (n-k) ticks this is SR-rt
Hamming DECODER (4,7)
101 Pattern Match: n ticks output ∅; then output 1 iff detect 101 on input
you will see n zeroes followed by n bits of correct cw
feed in n bits of rcvd cw followed by n zeroes

In general, let us feed $c_4c_3c_2c_1$ into the machine from right to left, so that it will read $c_1, c_2, c_3, c_4, 0, 0, 0$; denote $c_{ij} := c_i \oplus c_j$ and $c_{ijk} := c_i \oplus c_j \oplus c_k$.

Step	SR Out (3 bits)	Message (1 bit)	SR In/Top (1 bit)	Codeword (1 bit)
1	000	c_1	c_1	c_1
2	c_1c_10	c_2	c_2	c_2
3	$c_2c_{12}c_1$	c_3	c_{13}	c_3
4	$c_{13}c_{123}c_{12}$	c_4	c_{124}	c_4
5	$c_{124}c_{234}c_{123}$	0	0	c_{123}
6	$0c_{124}c_{234}$	0	0	c_{234}
7	$00c_{124}$	0	0	c_{124}

This is because $c_{11234} = c_{11} \oplus c_{234} = c_{234}$.

Now decoder. Note that there is the same shift register inside the decoder as in the encoder.



Step	SR out	output of 7 buffer FF	Rcvd cw	SR top	101 Output	SR left	Corrected cw
1	000	$0\,0\,0\,0\,0\,0\,0$	c_1	0	0	c_1	0
2	$c_1 0 0$	$c_1\,0\,0\,0\,0\,0\,0$	c_2	0	0	c_2	0
3	$c_2 c_1 0$	$c_2\,c_1\,0\,0\,0\,0\,0$	c_3	0	0	c_3	0
4	$c_3 c_2 c_1$	$c_3\,c_2\,c_1\,0\,0\,0\,0$	c_4	c_1	0	c_4	0
5	$c_1 \oplus c_4$, $c_1 \oplus c_3$, c_2	$c_4\,c_3\,c_2\,c_1\,0\,0\,0$	c_5	c_2	0	c_5	0
6	$c_2 \oplus c_5$, $c_2 \oplus c_1 \oplus c_4$, $c_1 \oplus c_3$	$c_5\,c_4\,c_3\,c_2\,c_1\,0\,0$	c_6	$c_1 \oplus c_3$	0	c_6	0
7	$c_1 \oplus c_3 \oplus c_6$, $c_1 \oplus c_3 \oplus c_2 \oplus c_5$, $c_2 \oplus c_1 \oplus c_4$	$c_6\,c_5\,c_4\,c_3\,c_2\,c_1\,0$	c_7	$c_1 \oplus c_2 \oplus c_4$	0	c_7	0
8	$c_1 \oplus c_2 \oplus c_4 \oplus c_7$, $c_2 \oplus c_3 \oplus c_4 \oplus c_6$, $c_1 \oplus c_3 \oplus c_2 \oplus c_5$	$c_7\,c_6\,c_5\,c_4\,c_3\,c_2\,c_1$	0	$c_1 \oplus c_2 \oplus c_3 \oplus c_5$	0	0	c_1
							c_2

7.6 Hamming Decoder and Generating Functions

Look at the figure in the previous section. Note that on step 8,

$$c_1 \oplus c_2 \oplus c_3 \oplus c_5 = 0$$

since $c_5 = c_1 \oplus c_2 \oplus c_3$ assuming there are no errors, so these combinations of 4 XORS are double checking the bits $5, 6, 7$.

Now assume the error is in c_3.

Step	SR Out (3 bits)	Buffer (7 bits)	SR In/Top (1 bit)	Codeword Out (1 bit)
8	$c_{1247}c_{2346}c_{1325} = 011$	$c_7c_6c_5c_4c_3c_2c_1$	$c_{1325} = 1$	c_1
9	111	$0c_7c_6c_5c_4c_3c_2$	1	c_2
10	101	$00c_7c_6c_5c_4c_3$	1	$c_3 \oplus 1$
11	100	...	0	c_4
12	010		0	c_5
13	001		1	c_6
14	110		0	c_7

Notice how the SR Out column forms a cycle! This is how the pattern 101 corrects errors by locating its position. If there were no errors, SR Out would have started at 000 and never moved. This is a Galois linear feedback shift register.

All Kinds of Codes Hamming codes are a kind of block code, because they act on a block of 4 bits, mapping it to 7 bits. A special kind of block code is a cyclic code, for which a codeword can be shifted around while still working. Some kinds of cyclic codes are Hamming codes, BCH codes, and Reed Solomon codes, which form a family of codes. Golay codes are a also cyclic, but they stick out from the other codes (Steiner system reference). Why? Hamming:

k	n	e
4	7	1
11	15	1
26	31	1
...	...	...

BCH:

k	n	e
7	15	2
5	15	3
21	31	2
17	31	5
...	...	...

Reed-Solomon codes: These are special because they correct bytes, not bits.

k	n	e
223×8	255×8	16×8
...	...	...

But, Golay codes only word for one combination of k, n, e:

k	n	e
12	23	3

Other than block codes, there are also things such as convolutional codes (i.e. Viterbi code). Unlike Block codes, these codes have memory, so a message might not produce the same codeword if run multiple times.

In real life, block codes are normally cyclic because they can be efficiently done with things like shift registers.

Engineers: bits, Mathematician: elements of the field $\mathbb{F}_2 = GF(2) = \mathbb{Z}/2\mathbb{Z}$.

A code is identified by a generator polynomial. Hamming codes have a function

$$g(x) = x^3 + x + 1$$

over $\mathbb{F}_2$.

Why does x^3+x+1 work based on the shift register design? Consider the taps on 1 and 2. This corresponds to $x^0=1$ and $x^1=x$, and because it is a Hamming (4,7), we add on a x^3, because $3=n-k=7-4$.

Design of Encoder and Decoder There are two different views of the world.

Engineer (Shift Register)	Mathematician (Polynomials over $\mathbb{F}_2$)
Bits (0/1)	Elements of $\mathbb{F}_2$
XOR of 2 bits	Addition of two elements
AND	Multiplication of two elements
String of bits (1011001)	$x^6+0x^5+x^4+x^3+0x^2+0x+1$.
Bitwise XOR of two strings	Sum of 2 polynomials
Convolution of two strings	Product of two polynomials

For an example consider 110 (x^2+x) and 11001 (x^4+x^3+1), We have

$$(x^2+x)(x^4+x^3+1)=x^6+x^4+x^2+x.$$

Now convolve these two bits:

$$1100100+110010+00000=1010110\rightarrow x^6+x^4+x^2+x$$

Convolution: We reverse one string and slide them past each other. This was also how we added the PDFs of random variables, adding together the probability of all possible pairs that sum to x. In terms of polynomial multiplication, we look at all possible pairs that get a term of degree x.

$$(f*g)[x]=\sum_t f[t]g[x-t].$$

Going back to Hamming $(7,4,1)$ and

$$g(x)=x^3+x+1,$$

let us try to encode it: If our message polynomial is $m(x)=c_1x^3+c_2x^2+c_3x+c_4$, then our output polynomial is $c(x)=c_1x^6+c_2x^5+\cdots+c_7$. How do we get it?

1. Find $x^3m(x)$
2. $r(x)=x^3m(x) \mod g(x)$,
3. $c(x)=x^3m(x)-r(x)=x^3m(x)+r(x)$.

If our message is 1010, then $m(x)=x^3+x$. $m(x)x^3=x^6+x^4$.
$m(x)x^3 \pmod{g(x)}=x+1$.
$c(x)=x^6+x^4-x-1=x^6+x^4+x+1\rightarrow 1010011$

7.7 Encoding and Decoding Math and Channel Coding Theorem

Recall the encoder:

1. Find $x^3m(x)$
2. $r(x) = x^3m(x) \mod g(x)$,
3. $c(x) = x^3m(x) - r(x) = x^3m(x) + r(x)$.

Decoder: We have some seven bit codeword c' with gf $c'(x) = c'_1x^6 + c'_2x^5 + \cdots + c'_7$. We want $m'(x) = m'_1x^3 + m'_2x^2 + m'_3x + m'_4$.

1. Let $r'(x) = c'(x) \pmod{g(x)}$
2. If $r'(x) = 0$, there are no errors, so $m'(x) = m(x)$, which is the $c'(x)$ truncating the last 3 bits, is sent.
3. If $r'(x) \neq 0$, there must be at least one error. Assuming there is only one (because that is our capability).
4. Compare $r'(x)$ to the different remainders to find the wrong bit (see below).

Consider the remainders of x^n when divided by $g(x) = x^3 + x + 1$.

$$x^6 \to x^2 + 1, x^5 \to x^2 + x + 1, x^4 \to x^2 + x, x^3 \to x + 1, x^2 \to x^2, x \to x, 1 \to 1.$$

In binary:

$$101, 111, 110, 011, 100, 010, 001.$$

Note 101 is the remainder of x^6, just like in our decoder. If we see any of these remainders, we compare them to the gfs' remainders.

If our code is

$$c'(x) = c(x) + e(x),$$

then

$$c'(x) \mod g(x) = [c(x) + e(x)] \mod g(x) = e(x) \mod g(x),$$

since $c(x) \equiv 0 \mod g(x)$.

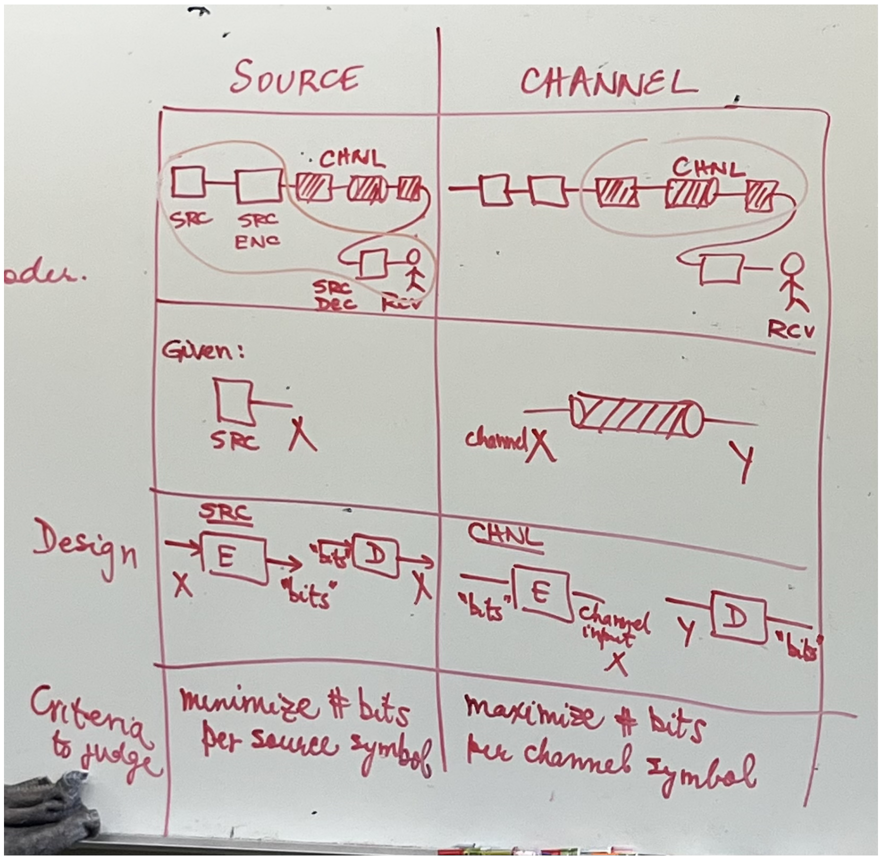
Example Message: 1011, $m(x) = x^3 + x + 1$, so $c(x) = x^6 + x^4 + x^3$.

Codeword: 1100001, $c'(x) = x^6 + x^5 + 1$, so $r'(x) = x + 1 \neq 0$, so there is an error in the x^3 term. So, $c'(x) = x^6 + x^5 + x^3 + 1$, so $m(x) = 1101$.

Just for fun, Hamming (15,11,1): $g(x) = x^4 + x + 1$.

Hamming (15,7,2): $g(x) = x^8 + x^7 + x^6 + x^4 + 1$.

Golay (23,12,3): $g(x) = x^11 + x^9 + x^7 + x^6 + x^5 + x + 1$.



7.7.1 Channel Coding Theorem

$H(X)$ is the lower bound of the performance of any source encoder of decoder (this is the Source Coding theorem). Similarly, we will prove that $C(Y|X)$ is an upper bound on the performance of any channel encoder or decoder.

Lemma 1:

$$I(\langle u_1, u_2, \ldots, u_R\rangle; \langle v_1, v_2, \ldots, v_R\rangle) \leq I(X;Y).$$

Lemma 1.5:

$$R \cdot I(u_i; v_i) \leq I(\langle u\rangle; \langle v\rangle)$$

Lemma 2 (Fano):

$$H(u_i|v_i) \leq H(P_{err})$$

Combining these, we get

$$R \cdot I(u_i; v_i) \leq I(\langle u\rangle; \langle v\rangle) \leq I(X;Y) \leq C(Y|X).$$

Also,

$$I(u_i; v_i) = H(u_i) - H(u_i|v_i) = 1 - H(u_i|v_i) \geq 1 - H(P_{err}).$$

For simplicity, just assume that the u_i are fair and independent coin tosses. This yields

$$C(Y|X) \geq R(1 - H(P_{err}).$$

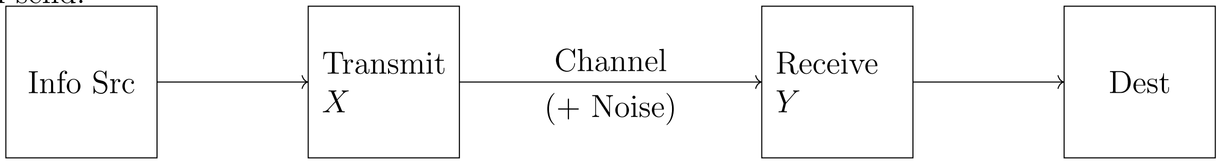
We have

$$R \leq \frac{C(Y|X)}{1 - H(P_{err})},$$

which is a bound on the number of symbols that can be sent. As $P_{err} \to 0$, we find that $R \to C(Y|X)$, so the number of symbols is bound by the channel capacity.

7.8 Channel Coding Theorem (12.02.2024)

We prove that the channel capacity $C(Y|X)$ is the upper bound of the amount of stuff you can send.



$$\langle u_1, u_2, u_3, \ldots, u_k \rangle \to X \to Y \to \langle v_1, v_2, v_3, \ldots, v_k \rangle$$

Lemma 1 Per block:

$$I(\vec{U}; \vec{V}) \leq I(\vec{X}; \vec{Y})$$

or Per Symbol:

$$kI(U; V) \leq nI(X; Y)$$

Consider we have a Markov Chain

$$A \to B \to C.$$

We want to prove that $I(A; C) \leq I(B; C)$, i.e.

$$I(A; C) - I(B; C) \leq 0.$$

Consider the expected value (This comes from $I(A; C) = H(A|C) - H(C)$, with the $H(C)$ terms cancelling out. You also need to Bayes theorem to reverse the conditional probability and use the definition of conditional entropy.):

$$\begin{aligned}
\mathbb{E}_{AC}\left[\log \frac{p(c|a)}{p(c)}\right] - \mathbb{E}_{BC}\left[\log \frac{p(c|b)}{p(c)}\right] &= \mathbb{E}_{ABC}\left[\log \frac{p(c|a)}{p(c)}\right] - \mathbb{E}_{ABC}\left[\log \frac{p(c|b)}{p(c)}\right] \\
&= \mathbb{E}_{ABC}\left[\log \frac{p(c|a)}{p(c|b)}\right] \\
&\leq \log\left(\mathbb{E}_{ABC}\left[\frac{p(c|a)}{p(c|b)}\right]\right) \\
&= \log 1 = 0,
\end{aligned}$$

by Jensen. This is because

$$\begin{aligned}\mathbb{E}_{ABC}\left[\frac{p(c|a)}{p(c|b)}\right] &= \sum_{a,b,c} p(a,b,c)\frac{p(c|a)}{p(c|b)}\\ &= \sum_{a,b,c} p(a)p(b|a)p(c|b)\frac{p(c|a)}{p(c|b)}\\ &= \sum_{a,b,c} p(a)p(b|a)p(c|a)\\ &= \sum_{a,b,c} p(b|a)p(a,c)\\ &= \sum_{a,b} p(b|a)p(a)\\ &= \sum_{a,b} p(a,b) = 1.\end{aligned}$$

By data processing inequality, since C only depends on B and not A, we can expand $p(a,b,c)$ like that. Something something markov property? yeah

By the same logic, $I(A;C) \leq I(A;B)$. Why? Going backwards,

$$I(A;C) - I(A;B) \leq 0$$

$$\mathbb{E}_{AC}\left[\log\frac{p(a|c)}{p(a)}\right] - \mathbb{E}_{AB}\left[\log\frac{p(a|b)}{p(a)}\right].$$

And continue as before. Going back to our channel model, we have the markov chain

$$U \to X \to Y \to V,$$

so we have $I(U;V) \leq I(U;Y) \leq I(X;Y)$, proving Lemma 1.

Lemma 2

$$H(u_i|v_i) \leq H(P_{err}).$$

We are dealing with a **binary channel**. If there is an error, there is only one possibility and can be corrected by flipping the bit. We define P_{err} to be the probability that the bit sent into the encoder is received wrongly out of the decoder, so $v_i \neq u_i$ is received.

If you know the bits to X translation in the encoder and $p(y|x)$ of noise in the channel and the Y to bits translation of the decoder, we can determine P_{err} exactly.

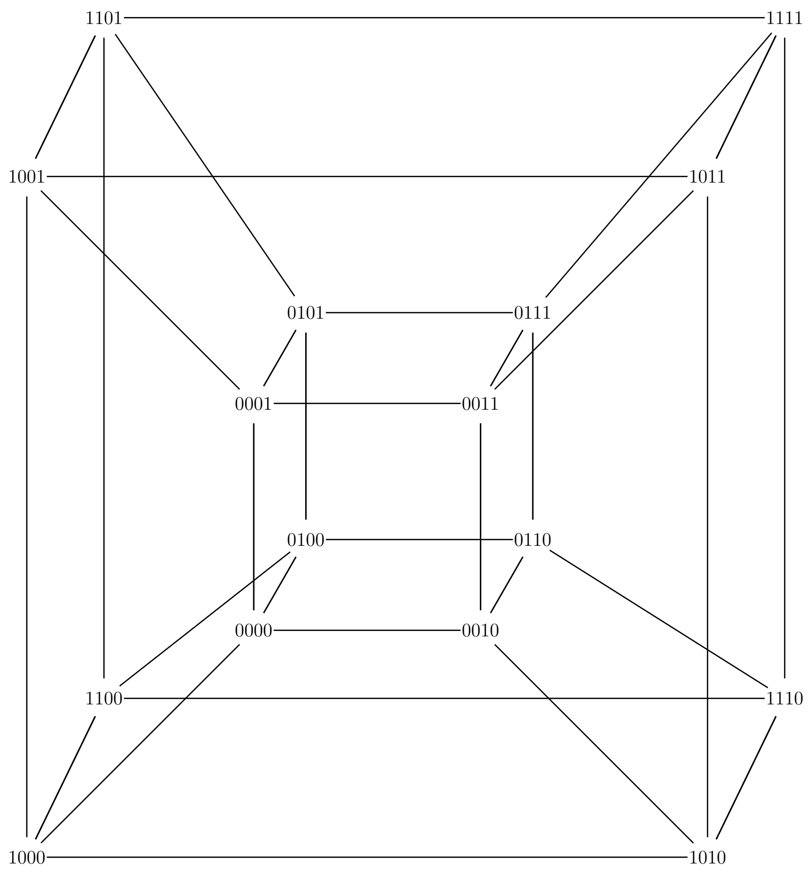
Toss a coin with weight P_{err}, if heads then you know that you have to flip it, but if tails then you know that it is right with no uncertainty, so

$$H(u_i|v_i) \leq H(P_{err}).$$

(it is actually equal but no one cares)

From Lemma 1,

$$\frac{k}{n}I(u_i;v_i) \leq I(X;Y) \leq C(Y|X).$$



From Lemma 2,

$$I(u_i; v_i) = H(u_i) - H(u_i|v_i) \geq 1 - H(P_{err}),$$

assuming that u_i is 0 or 1 uniformly (on average), which is the worst we can send. Again this is also an equality but dw about it. Combining them,

$$\frac{k}{n} \leq \frac{C(Y|X)}{1 - H(P_{err})}.$$

Letting $P_{err} \to 0$, we see that

$$\frac{k}{n} \leq C(Y|X).$$

We can consider k/n to be the rate of data through the channel. Recall that for source coding, we knew of a way to almost get to the lower entropy bound (Huffman is within one), but for channel coding there isn't a way to get to the upper bound.

Geometric Intuition of Codes We need a hypercube graph. If a 1d hypercube graph is a single vertex, we define an nd hypercube graph to be two copies of an $n-1$d hypercube graph with the corresponding edges connected. We can also think of the nd hypercube graph as 2^n vertices labeled in binary from $00\ldots0$ to $11\ldots1$, where two vertices are connected by an edge if they differ by exactly one bit. This will yield $n \cdot 2^{n-1}$ edges. We define the distance $d(x, x')$ between two vertices x and x' to be the minimum number edges between them, or, alternatively, the number of positions that x and x' differ bitwise. This is known as the Hamming distance.

7.9 Hamming "Sphere" or Something (12.04.2024)

We define the Hamming "sphere" at node x with radius r is the set of all nodes on the Hamming hypercube that are $\leq r$ units away from x.

$$\text{“Sphere”}(x, r) := \{x' | d(x, x') \leq r\}.$$

We find that the volume of the "sphere" is

$$\sum_{k=0}^{\lfloor r \rfloor} \binom{n}{k}.$$

This is because there are $\binom{n}{k}$ ways to pick k bits to flip out of an n-bit number. Now, we can determine if a code is good.

Consider a source coder that maps $2^k \to 2^m$.

Recall how a cyclic code works: If there are 2^k red nodes in the 2^n space (valid codewords), such as $\langle b_1, b_2, \ldots, b_n \rangle$, then $\langle b_n, b_1, \ldots, b_{n-1} \rangle$ is also a red node. (idk why we are talking about these)

If a blue dot comes out of the channel, we snap to the closest red dot, which would hopefully fix the error. Naturally, we ask how many errors can be corrected.

1. For every red dot, blow a balloon from radius zero
2. Grow the balloons simultaneously
3. Stop when two touch
4. Take a step back, where the radius is R.

R is the maximum number of errors you can correct. If there are more than R errors, you cannot guarantee an accurate decoding.

Let c_i and c_j be some red codewords. Then

$$d_{min} = min_{i \neq j} d(c_i, c_j),$$

and

$$R = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor = \left\lfloor \frac{d_{min}}{2} \right\rfloor - 1.$$

Imagine a $(4, 10)$ code. Since the volume of a sphere of radius 3 in a 10-d hypercube is

$$\binom{10}{0} + \binom{10}{1} + \binom{10}{2} + \binom{10}{3} = 176,$$

and there are 2^4 red codes, so there would have to be $176 \cdot 2^4$ nodes, which is greater than 2^{10}, so it is impossible.

WHAT IS ON THE TEST

1. Channel capacity (Mutual information, KL Divergence ($D(p||q)$), Binary Symmetric Channel, Binary Erasure Channel)
2. Channel Codes (Shift Registers, Generating Polynomials, Hypercube Geometry intuition)
3. Channel Coding Theorem

7.10 Guest Speaker Professor Beth Malmskog (12.06.2024)

Introduction to Error Correcting Codes She studies locally-recoverable codes using algebraic geometry and number theory, as well as code-based post-quantum cryptography based off error correcting codes.

A code is any system for communicating information, and the alphabet is the set of symbols (actions, words, morse code etc.) used in the system. Communication consists of not only communicating across distance but also across time (storage). For example, books do this, as well as something like floppy disks.

Digital codes: Every symbol is translated into a string (vector) of digits. Every meaningful vector in a system is called a codeword. For example, for an image, we can encode each pixel as a number into a longer vector. The set of all codewords is a code.

We also need to build redundancy into our codes, not only because of human error but also due to electronic and mechanical errors. For example, error detecting codes can be used to determine if an error happened. Consider $\mathbb{Z}/n\mathbb{Z}$, the integers modulo n under addition, where two numbers a and b are considered equivalent if they have the same remainder when divided by n. You can think of this as counting on a clock. Consider the "Mod Detecting Code," other wise known as a Universal Product Code (UPC), which consists of 11 digits of product information (like the label of an item or any other info) and a 12th check digit. If the first 11 digits are $d_1, d_2, \ldots, d_{11}$, then we define the 12th digit d_{12} to be the digit such that

$$3d_1 + d_2 + 3d_3 + d_4 + \cdots + 3d_{11} + d_{12} \equiv 0 \pmod{10}.$$

If the first 11 digits are 09661911248, then the 12th digit would be 7. But why are we multiplying every other digit by 3? This is because we can detect some wrongful digit swaps. Ok, how do we correct errors? Send the message three times and do majority rules for each bit. But this isn't very efficient.

Hamming Codes To be more efficient, we can use Hamming codes. We know how this works, go back up in the notes if you want to review them. Hamming codes have 16 legal codewords.

What makes a good code? First, we want efficiency. A code has length n if each codeword is a vector of n symbols. There are k information symbols if it takes k symbols to send.

We also want effectiveness, that is to correct many errors. We define the Hamming distance of two vectors to be the number of digit places where they are different. The minimum distance of a code is the smallest distance between two distinct codewords of a code. If we find a $\vec{w}$ such that

$$d(\vec{v}, \vec{w}) \leq \frac{d-1}{2},$$

we know that it is probably meant to be $\vec{v}$. if there are up to $d-1$ errors, we will correct wrongly but detect a code regardless.

Theorem: Singleton Bound. If a code has a length n, a minimum distance d and if each codeword has k information symbols, then

$$d \leq n - k + 1,$$

and if equality happens, then the code is called a maximum distance separated (MDS) code. Hamming codes are not an MDS code, since

$$7 - 4 + 1 = 4,$$

but $d = 3$ so it isn't perfect.

Reed-Solomon codes Let q be prime, and let $\vec{a} = (0, 1, 2, \ldots, q-1)$ be the elements of $\mathbb{Z}/q\mathbb{Z}$. (This actually works for the integers mod q^n in general). Let $1 \leq k \leq q$. This is the dimension of the code. Define L_{k-1} to be the set of polynomials $g(x)$ of $\mathbb{Z}/q\mathbb{Z}[x]$ with degree $\leq k-1$. For every $g \in L_{k-1}$, we define $g(\vec{\alpha}) = (g(0), g(1), \ldots, g(q-1))$, evaluating the polynomials in $\mathbb{Z}/q\mathbb{Z}$. The Reed-Solomon code is defined as

$$RS(k, q) = \{g(\vec{\alpha}) : g \in L_{k-1}\}.$$

Every polynomial of degree at most $k-1$ gives a vector of length q in $RS(k, q)$.

Example: Let $q = 5$ and $k = 3$. If $g(x) = x^2 - 4x + 3 = x^2 + x + 3$, then we have $g(\vec{\alpha}) = (3, 0, 4, 0, 3)$. The function has three degrees of freedom (information symbols) and the length of the code is 5, and there are 5^3 total codes. Note that every polynomial in L_2 is a linear combination of $\{1, x, x^2\}$ over $\mathbb{F}_5$, since

$$g(x) = a + bx + cx^2$$

with $a, b, c \in \mathbb{Z}/5\mathbb{Z}$. If $f_1(x) = 1, f_2(x) = x, f_3(x) = x^2$, then

$$f_1(\vec{\alpha}) = (1, 1, 1, 1, 1), \quad f_2(\vec{\alpha}) = (0, 1, 2, 3, 4), \quad f_3(\vec{\alpha}) = (0, 1, 4, 4, 1),$$

and this forms a basis of the $RS(3, 5)$ code. Is it a good code? Every quadratic polynomial is determined by 3 coefficients, so there are three information symbols. If we have two distinct quadratic polynoimals, they can intersect in $0, 1$, or 2 points. Similarly, $\mod q$, they can intersect in 2 points, so the minimum distance at least $5 - 2 = 3$. The Singleton bound yields $d \leq n - k + 1 = 3$, so RS is optimal! We don't want k to be too big, for example if $k = q$ then there is no minimum distance which is bad. Depending on k, we can do the trade-off between k and distance. The way that we can determine the polynomial given the codeword is the polynomial interpolation algorithm. Reed-Solomon codes are still really popular in things like music storage.

Since there are q^k, and we need a lot of codewords, we need a large prime q, which isn't very efficient for calculating on computers. Instead of using the values of $\mathbb{Z}/q\mathbb{Z}$, which can be thought of as points on a line, we can consider points on a curve $\mathcal{X}$, which is the solution to some high dimensional multivariable polynomial. This lets you do coding faster.

Cloud Storage What kind of errors occur in cloud storage? The idea in cloud storage is spreading a codeword across many servers, so that if one server fails, it can be recovered by thinking of the erasure as just an error, which we know how to fix with error correction. But, we don't like storing each symbol in a different symbol. We want a locally recoverable code, which has the property that each symbol can be recovered by accessing only r different symbols; then r is the locality of the code. The set of symbols used to recover that symbol is called the recovery set of that symbol.

`https://en.wikipedia.org/wiki/Berlekamp%E2%80%93Welch_algorithm`. `https://en.wikipedia.org/wiki/Guruswami%E2%80%93Sudan_list_decoding_algorithm`.

The big idea is polynomial interpolation. Remember that if $f(x)$ is a univariable polynomial of degre m, then it is fully determined by $m+1$ points (Lagrange Interpolation). We can use this fact to locally recover an $RS(k, q)$ code. If we erase one symbol, it's like erasing the value of a polynomial in one input. Then, we only have to use k other bits to recover the entire polynomial and the erased bit. But the locality is k which is too big. We can generalize RS codes to a higher dimension. This is the Reed-Muller code.

Let m be a natural number and q prime, then we let $\vec{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{q^m})$, where α is a point in m dimensional space $(\mathbb{Z}/q\mathbb{Z})^m$. If $1 \leq v \leq q-2$, we can let M_v be the set of polynomials $g(x_1, x_2, \ldots, x_m)$ with coefficients in $\mathbb{Z}/q\mathbb{Z}$ and total degree at most v. We let $g(\vec{\alpha}) = g(\vec{\alpha_1}, \ldots, g(\vec{\alpha_{q^m}}))$. We call the Reed-Muller code $RM_q(v, m) = \{g(\vec{\alpha}) : g \in M_v$. Consider $RM_3(2, 2)$, which has code length 9. If $\vec{\alpha} = ((0,0), (0,1), \ldots, (2,1), (2,2))$ and $g(x, y) = x + y$, then $g(\vec{\alpha}) = (0,1,2,1,2,0,2,0,1)$. For $RM_5(3, 2)$ functions are in 2 variables of total degree at most 3 over $\mathbb{Z}/5\mathbb{Z}$. We can restrict the function to a line, which makes it a function of one variable. Since the total degree is 3, the polynomial over the line must also be of degree 3, so we can recover it with only 4 points. But, there are 5 points on every line, so we have achieved locality. We only have to look at 4 other points to figure out one erased symbol. Since each position has 6 lines through it, we have so many different recover sets (called availability). Unfortunately, the rate is at most $\frac{q-1}{q^2} \to 0$, which isn't that good. Decoding is also difficult, because it is NP-hard, but that's good because it can be used as cryptography. (`https://en.wikipedia.org/wiki/McEliece_cryptosystem`)

8 Putting it All Together

8.1 Shannon's Theorem (12.12.24)

	Theoretical Bound	Practical Bound
Source	$\forall$ source encoder/decoders, the # of bits per source symbol is $\geq H(X)$	$\exists$ a source encoder/decoder such that the number of bits per source symbol is around $H(X)$ (e.g. Huffman)
Channel	$\forall$ channel encoders/decoders, the # of bits/channel symbol is $\leq C(Y\|X)$	¯_(ツ)_/¯ Maybe we can get something that comes close to $C(Y\|X)$

Shannon's Theorem Given a source $p(x)$ and a channel $p(y|x)$, we are asked to design an encoder and decoder, such that you achieve reliable communication (i.e. $P_{err} \to 0$). Is this possible? YES but... iff $H(X) \geq C(Y|X)$. This is Shannon's theorem.

Unuseful Equations:

$$\sum_{n\leq x}\mu(n) = O\left(x^{1/2+\epsilon}\right),$$

where

$$\mu * \mathbb{1} = I,$$

assuming the Riemann Hypothesis.

$$\int_{\partial\Omega}\omega = \int_{\Omega}\mathrm{d}\omega$$

$$(u\cdot v)\,\mathbb{1} = u\wedge *v$$

$$\left(\Box + \frac{m^2c^2}{\hbar^2}\right)\psi = 0$$

$$e^{a\frac{\mathrm{d}}{\mathrm{d}x}}f(x) = f(x-a)$$

$$\left(1-\int\right)^{-1}0 = e^x$$

$$e^{-ax}\left(\frac{\mathrm{d}}{\mathrm{d}x}-a\right)^k y = \left(\frac{\mathrm{d}}{\mathrm{d}x}\right)^k\left(e^{-ax}y\right) = 0 \qquad \therefore y = \left(\sum_{n=0}^{k-1}a_nt^n\right)e^{ax}$$