1 Probability and Entropy

1.1 Entropy

The *entropy* of a discrete random variable X with pmf P is:

$$H(X) = \sum_x P(x) \log \frac{1}{P(x)}$$

$H(X)$ can be written as $\mathbb{E}\left[\log \frac{1}{P(X)}\right]$ and has unit *bits*. $H(X)$ is the uncertainty associated with the rv X.

X is called a Bernoulli(p) random variable if takes value 1 with probability p and 0 with probability $1-p$.

Binary Entropy Function:

$$H_2(p) = p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}$$

Let X be discrete random variable taking values in $\mathcal{X}$. Denote the alphabet size $|\mathcal{X}|$ by M. Then we have the following properties of entropy:

1. $H(X) \geq 0$
2. $H(X) \leq \log M$
3. Among all random variables taking values in $\mathcal{X}$, the equiprobable distribution $\left(\frac{1}{M}, \dots, \frac{1}{M}\right)$ has the maximum entropy, equal to $\log M$.

1.2 Joint and Conditional Entropy

The *joint entropy* of discrete rvs X, Y with joint pmf P_{XY} is:

$$H(X,Y) = \sum_{x,y} P_{XY}(x,y) \log \frac{1}{P_{XY}(x,y)}$$

The *conditional entropy* of Y given X is:

$$\begin{aligned} H(Y|X) &= \sum_x P_X(x) H(Y|X=x) \\ &= \sum_{x,y} P_{XY}(x,y) \log \frac{1}{P_{Y|X}(y|x)} \end{aligned}$$

Using product and sum rule of probability:

$$\begin{aligned} H(X,Y) &= H(X) + H(Y|X) \\ &= H(Y) + H(X|Y) \end{aligned}$$

Chain Rule of Joint Entropy: The joint entropy can be decomposed as:

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

If $X_1, \dots, X_n$ are independent random variables, then:

$$H(X_1, \dots, X_n) = \sum_{i=1}^{n} H(X_i)$$

1.3 Relative Entropy

The *relative entropy* or the Kullback-Leibler (KL) divergence between two pmfs P and Q is:

$$D(P\|Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}$$

Relative entropy is a measure of distance between distributions P and Q. However, it is not a true distance:

$$D(P\|Q) \neq D(Q\|P)$$

Relative Entropy is always non-negative: $D(P\|Q) \geq 0$ with equality if and only if $P = Q$.

1.4 Hypothesis Testing

Suppose we have data $X_1, \dots, X_n$, and the knowledge that one of the following is true.

$$\begin{aligned} H_0 &: X_1, \dots, X_n \sim \text{ i.i.d. } P \\ H_1 &: X_1, \dots, X_n \sim \text{ i.i.d. } Q \end{aligned}$$

H_0 is often called the null hypothesis.

Type I error: This occurs when H_0 is true, but the decision rule chooses H_1.

Type II error: This occurs when H_1 is true, but the decision rule chooses H_0.

The *likelihood ratio* (LR) is defined as:

$$LR(X_1, \dots, X_n) = \frac{Q(X_1, \dots, X_n)}{P(X_1, \dots, X_n)}$$

The normalized *log-likelihood ratio* (LLR) is:

$$LLR(X_1, \dots, X_n) = \frac{1}{n} \log \frac{Q(X_1, \dots, X_n)}{P(X_1, \dots, X_n)}$$

For the problem of testing between distributions P and Q, the optimal decision rule is a likelihood-ratio thresholding rule. For some threshold T:

Choose H_1 if $\frac{Q(X_1,\dots,X_n)}{P(X_1,\dots,X_n)} \geq T$; otherwise choose H_0.

Equivalently, the optimal rule can be expressed using LLR:

Choose H_1 if $\frac{1}{n} \log \frac{Q(X_1,\dots,X_n)}{P(X_1,\dots,X_n)} \geq t$; otherwise choose H_0.

Under H_1, where $X_i \sim$ i.i.d. Q, therefore $LLR(X_1, \dots, X_n) \to D(Q\|P)$. Under H_0, where $X_i \sim$ i.i.d. P, $LLR(X_1, \dots, X_n) \to -D(P\|Q)$.

1.5 Mutual Information

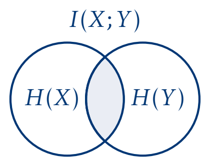
Consider two random variables X and Y with joint pmf P_{XY} . The *mutual information* between X and Y is defined as:

$$I(X;Y) = H(X) - H(X \mid Y)$$

Mutual information is the reduction in the uncertainty of X when you observe Y.

$$\begin{aligned} I(X;Y) &= H(X) + H(Y) - H(X,Y) \\ &= H(Y) - H(Y \mid X) \end{aligned}$$

Venn Diagram



The two circles together represent $H(X,Y)$.

Mutual information is the relative entropy between the joint pmf and the product of the marginals:

$$I(X;Y) = D(P_{XY} \| P_X P_Y)$$

$I(X;Y) \geq 0$ because $D(P\|Q) \geq 0$ for any pair of pmfs P, Q. Hence, $H(X \mid Y) \leq H(X)$ and $H(Y \mid X) \leq H(Y)$. Given X, Y, Z jointly distributed according to P_{XYZ}, the *conditional* mutual information $I(X;Y|Z)$ is defined as:

$$I(X;Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z)$$

Chain Rule of Mutual Information:

$$I(X_1, X_2, \dots, X_n; Y) = \sum_{i=1}^{n} I(X_i; Y \mid X_{i-1}, X_{i-2}, \dots, X_1)$$

2 Data Compression

2.1 Estimating Tail Probabilities

Markov and Chebyshev inequalities are ways to bound tail probabilities with limited information about the random variable.

Markov's Inequality: For a non-negative rx X and any $a > 0$,

$$P(X \geq a) \leq \frac{\mathbb{E}[X]}{a}$$

Often, we bound the tail probabilities of deviations around the mean of an rv.

Chebyshev's inequality: For any rv X and $a > 0$,

$$P(|X - \mathbb{E}X| \geq a) \leq \frac{\text{Var}(X)}{a^2}$$

2.2 Weak Law of Large Numbers

Weak Law of Large Numbers (WLLN) states that empirical average converges to the mean. Let $X_1, X_2, \cdots$ be a sequence of i.i.d. random variables with finite mean μ. Let $S_n = \frac{1}{n} \sum_{i=1}^{n} X_i$.

Formal statement of WLLN: For any $\epsilon > 0$,

$$\lim_{n \to \infty} P(|S_n - \mu| \geq \epsilon) = 0$$

.

2.3 Typicality

If $X_1, \cdots, X_n$ are chosen $\sim$ i.i.d. Bernoulli(p), then for large n, the fraction of ones in the observed sequence will be close to p with high probability (due to WLLN). Equivalently, the observed sequence will have probability close to $p^{np}(1-p)^{n(1-p)} = 2^{-nH_2(p)}$.

Asymptotic Equipartition Property (AEP): If $X^n = (X_1, \cdots, X_n)$ are i.i.d. $\sim P_X$, then for any $\epsilon > 0$,

$$\lim_{n \to \infty} Pr\left(\left|-\frac{1}{n} \log P(X^n) - H(X)\right| < \epsilon\right) = 1$$

The *typical set* $A_{\epsilon,n}$ with respect to P is the set of sequences $X^n \in \mathcal{X}^n$ with the property:

$$2^{-n(H(X)+\epsilon)} \leq P(X^n) \leq 2^{-n(H(X)-\epsilon)}$$

If $X^n = (X_1, \cdots, X_n)$ are i.i.d. $\sim P_X$, then:

$$\lim_{n \to \infty} Pr\left(X^n \in A_{\epsilon,n}\right) = 1$$

Let $|A_{\epsilon,n}|$ denote the number of elements in the typical set $A_{\epsilon,n}$.

$$\left|A_{\epsilon,n}\right| \leq 2^{n(H(X)+\epsilon)}$$

For sufficiently large n,

$$\left|A_{\epsilon,n}\right| \geq (1-\epsilon) 2^{n(H(X)-\epsilon)}$$

2.4 Compression

For any n, a compression code is defined as follows: To each source sequence $X^n = (X_1, \cdots, X_n)$, the code assigns a *unique* binary sequence $c(X^n)$ called the *codeword* for the source sequence X^n. Let $l(X^n)$ be the length of the codeword assigned to X^n, i.e., the number of bits in $c(X^n)$, the expected code length is defined as:

$$\mathbb{E}[l(X^n)] = \sum_{x^n} P(x^n) l(x^n)$$

Compression via the Typical Set:

1. Index each sequence in $A_{\epsilon,n}$ using $\lceil n(H(X)+\epsilon) \rceil$ bits. Prefix each of these by a flag bit 0.
2. Index each sequence not in $A_{\epsilon,n}$ using $\lceil \log |\mathcal{X}|^n \rceil$ bits. Prefix each of these by a flag bit 1.

$$\mathbb{E}[l(X^n)] \leq n(H(X) + \epsilon')$$

$\epsilon' = \epsilon + \epsilon \log |\mathcal{X}| + \frac{2}{n}$ can be made arbitrarily small by picking ϵ small enough and then n sufficiently large.

Let X^n be i.i.d. $\sim P$. Fix any $\epsilon > 0$. For n sufficiently large, there exists a code that maps sequences X^n of length n into binary strings such that the mapping is one-to-one and

$$\mathbb{E}\left[\frac{1}{n} l(X^n)\right] \leq H(X) + \epsilon$$

The expected length of any uniquely decodable code satisfies

$$\mathbb{E}\left[\frac{1}{n} l(X^n)\right] \geq H(X)$$

Hence entropy is the fundamental limit of lossless compression.

A code is called *prefix-free* or *instantaneously decodable* if no codeword is the prefix of another.

Kraft Inequality: A binary prefix-free code with codeword lengths $l_1, l_2, \cdots, l_N$ exists if and only if

$$\sum_{i=1}^{N} 2^{-l_i} \leq 1$$

Suppose any prefix-free code that assigns binary codewords to blocks of N source symbols $X^N = (X_1, \cdots, X_N)$. If X is an iid source, then

$$\frac{\mathbb{E}\left[l(X^N)\right]}{N} \geq \frac{H(X^N)}{N} = H(X)$$

2.5 Practical Source Coding

For a source which can take m values with probabilities $p_1, \cdots, p_m$, expected code length L is:

$$L = \sum_{i=1}^{m} p_i l_i \geq \sum_{i=1}^{m} p_i \log_2 \frac{1}{p_i} = H(X)$$

Shannon-Fano Coding:

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

To construct the code with these code lengths, we simply grow a tree from its root placing codewords on leaves as we reach the required depths.

$$L < \sum_i p_i \left(\log_2 \frac{1}{p_i} + 1 \right) = H(X) + 1$$

Huffman Coding:

1. Take the two least probable symbols in the alphabet. These two symbols will be given the longest codewords, which will have equal length, and differ only in the last digit.
2. Combine these two symbols into a single symbol, and repeat.

Optimality: For a given set of probabilities, there is no prefix-free code that has smaller expected length than the Huffman code.

Interval Coding: The binary codeword for a symbol with probability p represented by the interval $[a, a+p)$ can be obtained as follows:

1. Find the largest dyadic interval of the form $[\frac{j}{2^l}, \frac{j+1}{2^l})$ that lies within $[a, a+p)$. (Here j, l are integers)
2. Take the binary representation of the lower end-point of the dyadic interval as the codeword. (This will be the integer j converted to binary and represented using l bits.)

$$L \leq \sum_i p_i \left(\left\lceil \log_2 \frac{1}{p_i} \right\rceil + 1 \right) < H(X) + 2$$

Arithmetic Coding: From interval coding for the first symbol, we divide the chosen interval for X_1 in the proportions of the symbol probabilities for X_2, and repeat. The expected code length per symbol is:

$$\frac{L_n}{n} < \frac{H(X^n)}{n} + \frac{2}{n} = H(X) + \frac{2}{n}$$

With growing n, arithmetic coding can achieve expected code length/symbol that is arbitrarily close to the source entropy.

Often the true distribution of the source is unknown. Suppose that the true pmf of a rv is $P = \{p_1, \ldots, p_m\}$ and the estimated pmf is $\hat{P} = \{\hat{p}_1, \ldots, \hat{p}_m\}$. The average code-length is:

$$L = \sum_i p_i \log \frac{1}{\hat{p}_i} = H(P) + D(P \| \hat{P})$$

Design a code using a distribution $\hat{P}$ that minimizes the worst-case redundancy over this class of distributions $\mathcal{P}$. With this choice, the *minimax redundancy* is:

$$R^* = \min_{\hat{P}} \max_{P \in \mathcal{P}} D(P \| \hat{P})$$

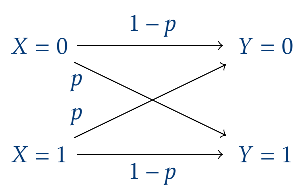
3 Data Transmission

3.1 Discrete Channels

Transmitter does two things:

1. *Coding:* Adding redundancy to the data bits to protect against noise.
2. *Modulation:* Transforming the coded bits into waveforms.

Binary Symmetric Channel (BSC):



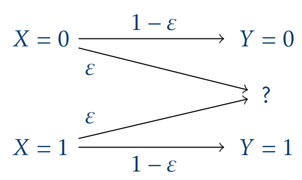
The channel is called BSC(p) and p is the crossover probability.

$(1, n)$ Repetition Code: Data rate $= \frac{1}{n}$ bits/transmission.

A discrete memoryless channel (DMC) is a system consisting of an input alphabet $\mathcal{X}$, output alphabet $\mathcal{Y}$, and a set of transition probabilities:

$$P_{Y|X}(b \mid a) = \Pr(Y = b \mid X = a)$$

Binary erasure channel (BEC):



When the demodulator thinks the (real-valued) output symbol is too noisy, it can declare an erasure.

For a general DMC, we can construct a set of input sequences which have *non-intersecting* sets of output sequences with high probability.

3.2 Channel Capacity

The *channel capacity* of a discrete memoryless channel is defined as:

$$\mathcal{C} = \max_{P_X} I(X;Y)$$

For Noiseless Binary Channel, $I(X;Y) = H(X)$.

$$\mathcal{C} = \max_{P_X} H(X) = 1$$

For BSC, $I(X;Y) = H(Y) - H_2(p)$.

$$\mathcal{C} = \max_{P_X} H(Y) - H_2(p) = 1 - H_2(p)$$

For BEC, $I(X;Y) = H(Y) - H_2(\varepsilon)$. Let the input distribution be $P_X = (\alpha, 1-\alpha)$.

$$H(Y) = H_2(\varepsilon) + (1-\varepsilon) H_2(\alpha)$$

$$\mathcal{C} = \max_{P_X} (1-\varepsilon) H_2(\alpha) = 1 - \varepsilon$$

Maximum value is attained when $P_X = \left(\frac{1}{2}, \frac{1}{2}\right)$.

3.3 Channel Code

An (n, k) *channel code* of rate R for the channel $(\mathcal{X}, \mathcal{Y}, P_{Y|X})$ consists of:

1. A set of messages $\left\{1, \ldots, 2^k = 2^{nR}\right\}$
2. An encoding function $X^n : \left\{1, \ldots, 2^{nR}\right\} \to \mathcal{X}^n$ that assigns a codeword to each message. The set of codewords $\left\{X^n(1), \ldots, X^n\left(2^{nR}\right)\right\}$ is called the *codebook*
3. A decoding function $g : \mathcal{Y}^n \to \left\{1, \ldots, 2^{nR}\right\}$ which produces a guess of the transmitted message for each received vector

The *rate* R of the code is $R = \frac{k}{n}$ bits/transmission.

The maximal probability of error of the code is defined as:

$$\max_{j \in \{1, \ldots, 2^{nR}\}} \Pr(\hat{W} \neq j \mid W = j)$$

The average probability of error of the code is

$$\frac{1}{2^{nR}} \sum_{j=1}^{2^{nR}} \Pr(\hat{W} \neq j \mid W = j)$$

W and $\hat{W}$ denote the transmitted, and decoded messages respectively.

The Channel Coding Theorem:

1. Fix $R < \mathcal{C}$ and pick any $\epsilon > 0$. Then, for all sufficiently large n there exists a length-n code of rate R with maximal probability of error less than ϵ.
2. Conversely, any sequence of length-n codes of rate R with average/maximal probability of error $P_e^{(n)} \to 0$ as $n \to \infty$ must have $R \leq C$.

Assuming a uniform prior on the messages, the optimal decoding rule is maximum-likelihood decoding:

$$\hat{W} = \underset{\hat{W}}{\operatorname{argmax}} \prod_{i=1}^{n} P_{Y|X}(Y_i \mid X_i(W))$$

3.4 Joint Typicality

The set $A_{\epsilon,n}$ of *jointly typical* sequences $\{(x^n, y^n)\}$ with respect to a joint pmf P_{XY} is defined as $A_{\epsilon,n} = \{(x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n\}$ such that:

$$\left| -\frac{1}{n} \log P_X(x^n) - H(X) \right| < \epsilon$$

$$\left| -\frac{1}{n} \log P_Y(y^n) - H(Y) \right| < \epsilon$$

$$\left| -\frac{1}{n} \log P_{XY}(x^n, y^n) - H(X,Y) \right| < \epsilon$$

The Joint AEP: Let (X^n, Y^n) be a pair of sequences drawn i.i.d. according to P_{XY}, then for any $\epsilon > 0$:

1. $\Pr\left((X^n, Y^n) \in A_{\epsilon,n}\right) \to 1$ as $n \to \infty$
2. $\left|A_{\epsilon,n}\right| \leq 2^{n(H(X,Y)+\epsilon)}$
3. If $\left(\tilde{X}^n, \tilde{Y}^n\right)$ are a pair of sequences drawn i.i.d. according to $P_X P_Y$:

$$\Pr\left(\left(\tilde{X}^n, \tilde{Y}^n\right) \in A_{\epsilon,n}\right) \leq 2^{-n(I(X;Y)-3\epsilon)}$$

Joint Typicality Decoder: The decoder declares that the message $\hat{W}$ was sent if both the following conditions are satisfied:

1. $\left(X^n(\hat{W}), Y^n\right)$ is jointly typical with respect to $P_X P_{Y|X}$.
2. There exists no other message $W' \neq \hat{W}$ such that $(X^n(W'), Y^n)$ is jointly typical.

If no such $\hat{W}$ is found or there is more than one such, an error is declared.

The average probability of error for a given codebook $\mathcal{B}$ is:

$$P_e(\mathcal{B}) = \frac{1}{2^{nR}} \sum_{w=1}^{2^{nR}} \Pr(\hat{W} \neq w \mid \mathcal{B}, W = w)$$

For any $\epsilon > 0$, when $R < I(X;Y) - 3\epsilon$, the probability of error averaged over all messages and all codebooks is:

$$\overline{P}_e = \sum_{\mathcal{B}} P_e(\mathcal{B}) \Pr(\mathcal{B}) \leq 2\epsilon$$

There exists at least one codebook $\mathcal{B}^*$ with $P_e(\mathcal{B}^*) \leq 2\epsilon$.

3.5 Data Processing

Random variables X, Y, Z are said to form a *Markov chain* if their joint pmf can be written as:

$$P_{XYZ} = P_X P_{Y|X} P_{Z|Y}$$

If $X - Y - Z$ form a Markov chain, then $I(X;Y) \geq I(X;Z)$.

Fano's Inequality: For any estimator $\hat{X}$ such that $X - Y - \hat{X}$, the probability of error $P_e = \Pr(\hat{X} \neq X)$ satisfies:

$$1 + P_e \log|\mathcal{X}| \geq H(X \mid \hat{X}) \geq H(X \mid Y)$$

$$P_e \geq \frac{H(X \mid Y) - 1}{\log|\mathcal{X}|}$$

The data-processing inequality tells us that $H(X \mid \hat{X}) \geq H(X \mid Y)$.

Let Y^n be the result of passing a sequence X^n through a DMC of channel capacity $\mathcal{C}$. Then $I(X^n;Y^n) \le n\mathcal{C}$ regardless of the distribution of X^n.

Channel Coding Converse: Consider any $(2^{nR}, n)$ channel code with average probability of error P_e:

$$P_e \ge 1 - \frac{\mathcal{C}}{R} - \frac{1}{nR}$$

Thus, unless $R \le \mathcal{C}$, P_e is bounded away from 0 as $n \to \infty$.

4 Channel Coding

4.1 The Additive White Gaussian Noise (AWGN) Channel

The continuous-time AWGN channel:

$$Y(t) = X(t) + N(t)$$

Usual assumptions on the channel:

1. Input $X(t)$ is *power-limited* to P.
2. $X(t)$ is *band-limited* to W.
3. Noise $N(t)$ is a random process assumed to be *white* Gaussian.

The discrete-time AWGN channel:

$$Y_k = X_k + Z_k, \quad k = 1, 2, \dots$$

Average power constraint P on input:

$$\frac{1}{n}\sum_{k=1}^{n} X_k^2 \le P$$

Z_k are i.i.d. Gaussian with mean 0, variance σ^2. $\mathcal{N}\left(0, \sigma^2\right)$.

4.2 Differential Entropy

The *differential entropy* of a continuous random variable X with pdf f_X is:

$$h(X) = \int_{-\infty}^{\infty} f_X(u) \log \frac{1}{f_X(u)} du$$

Gaussian random variable: Let $X \sim \mathcal{N}\left(\mu, \sigma^2\right)$. The pdf ϕ is given by:

$$\phi(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

The *joint* differential entropy of X, Y is:

$$h(X,Y) = \int f_{XY}(u,v) \log \frac{1}{f_{XY}(u,v)} du dv$$

The *conditional* differential entropy of X given Y is:

$$h(X \mid Y) = \int f_{XY}(u,v) \log \frac{1}{f_{X|Y}(u \mid v)} du dv$$

4.3 The AWGN Channel

The capacity of the AWGN channel with power constraint P is $\mathcal{C} = \max I(X;Y)$

$$I(X;Y) = h(Y) - h(Y \mid X) = h(Y) - h(Z)$$

Among all random variables Y with $\mathbb{E}Y^2 \le \left(P + \sigma^2\right)$, the maximum differential entropy is achieved when Y is Gaussian $\mathcal{N}\left(0, P + \sigma^2\right)$.

$$h(Y) \le \frac{1}{2} \log 2\pi e \left(P + \sigma^2\right)$$

The capacity of the discrete-time AWGN channel with input power constraint P and noise variance σ_2 is:

$$\mathcal{C} = \frac{1}{2} \log\left(1 + \frac{P}{\sigma^2}\right)$$

4.4 Channel Coding

A *channel coding* system consists of two parts:

1. *Channel Encoder:* Adds redundancy to the source bits in a controlled manner.
2. *Channel Decoder:* Recovers the source bits from the received bits by exploiting the redundancy.

An (n,k) binary block code maps every block of k data bits into a length n binary codeword. The rate R of the code is $R = \frac{k}{n}$. Assuming the codewords are distinct, the number of codewords is $M = 2^k$.

The *Hamming distance* $d(\underline{x}, \underline{y})$ between two binary sequences $\underline{x}, \underline{y}$ of length n is the number of positions in which x and y differ.

Let $\mathcal{B}$ be a code with codewords $\left\{\underline{c}_1, \dots, \underline{c}_M\right\}$. Then the minimum distance $d_{\min}$ is the smallest Hamming distance between any pair of codewords:

$$d_{\min} = \min_{i \ne j} d\left(\underline{c}_i, \underline{c}_j\right)$$

4.5 Optimal Decoding of a Block Code

The optimal decoder is the one that minimises the probability of decoding error.

Optimal decoding on the BSC(p):

$$\text{Decode } \hat{\underline{c}} = \underset{\underline{c} \in \{\underline{c}_1, \dots, \underline{c}_M\}}{\operatorname{argmin}} \; d(\underline{y}, \underline{c})$$

For $p < \frac{1}{2}$, the optimal decoder for BSC picks the codeword closest in Hamming distance to $\underline{y}$. We can successfully correct any pattern of t errors if $t \le \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$.

4.6 Linear Block Codes

A (n,k) linear block code (LBC) is defined in terms of k length-n binary vectors $\underline{g}_1, \dots, \underline{g}_k$. A sequence of k data bits $\underline{x} = (x_1, \dots, x_k)$ is mapped to a length-n codeword $\underline{c}$ as follows.

$$\underline{c} = x_1 \underline{g}_1 + x_2 \underline{g}_2 + \dots + \dots + x_k \underline{g}_k$$

$$\underline{c} = \underline{x} G = \underline{x} \begin{bmatrix} \underline{g}_1 \\ \vdots \\ \underline{g}_k \end{bmatrix}$$

The $k \times n$ matrix G is called a *generator matrix* of the code. k is called the *code dimension*, n is the *block length*. The generator matrix for a code (i.e., a set of codewords) is not unique.

The *systematic generator matrix* is of the form:

$$G = [I_k \mid P]$$

In a systematic code, the length-n codeword consists of the k data bits $\underline{x}$, followed by $(n-k)$ parity bits $\underline{x}P$.

$$\underline{c} = \underline{x}[I_k \mid P] = [\underline{x} \mid \underline{x}P]$$

Let $\mathcal{C}$ be an (n,k) LBC with codewords $\{c_0, \dots, c_{M-1}\}$. $\mathcal{C}$ is a *subspace* of $\{0,1\}^n$, i.e., it is closed under vector addition and scalar multiplication. Hence, the sum of any two codewords is also a codeword; the all-zero vector $\underline{0}$ is always a codeword.

4.7 The Parity Check Matrix

The orthogonal complement of $\mathcal{C}$, denoted $\mathcal{C}^{\perp}$ is defined as the set of all vectors in $\{0,1\}^n$ that are orthogonal to each vector in $\mathcal{C}$. We can find a basis $\left\{\underline{h}_1, \dots, \underline{h}_{n-k}\right\}$ for $\mathcal{C}^{\perp}$, expressed as:

$$H = \begin{bmatrix} \underline{h}_1 \\ \vdots \\ \underline{h}_{n-k} \end{bmatrix}$$

The $(n-k) \times n$ matrix H is called the *parity check matrix*. If we have systematic $G = [I_k \mid P]$ then:

$$H = \left[P^T \mid I_{n-k}\right]$$

The codewords satisfy $\underline{c}H^T = \underline{0}$.

The minimum distance of an LBC equals the minimum Hamming weight among the non-zero codewords.

$$d_{\min} = \min_{i \ne j} wt\left(\underline{c}_i + \underline{c}_j\right) = \min_{\underline{c}_k \ne 0} wt\left(\underline{c}_k\right)$$

Let $\mathcal{C}$ be an linear block code with parity check matrix H. The minimum distance of $\mathcal{C}$ is the smallest number of columns of H that sum to $\underline{0}$.

4.8 Low Density Parity Check Codes

In a regular (n,k) LDPC code:

1. Each of the n codeword bits (variable) is involved in d_v parity check equations, where d_v is the column weight.
2. Each of the $(n-k)$ parity check equations (check) involves d_c code bits, where d_c is the row weight.

The *design rate* of a regular LDPC code is:

$$\frac{k}{n} = 1 - \frac{d_v}{d_c}$$

The *design rate* is the true code rate if the rows of the parity check matrix are linearly independent.

For irregular codes, we need to specify the *weight distributions* on the columns and rows.

4.9 Iterative Decoding as Message Passing

Message passing is a class of iterative algorithms, where in each step:

1. Each variable node v sends a message m_{vc} to each check node c that it is connected to.

   $m_{vc} = x \in \{0,1\}$ if $v = x$ or at least one incoming $m_{c'v} = x$.

   $m_{vc} =?$ if $v =?$ and all incoming $m_{c'v} =?$.
2. Each check node c sends a message m_{cv} to each variable node v that it is connected to.

   $m_{cv} = \sum_{v'} m_{v'c} \bmod 2$ if no $m_{v'c} =?$.

   $m_{cv} =?$ if at least one incoming $m_{v'c} =?$.

4.10 Degree Distributions

We can define degree distributions from the *node perspective*:

- L_i : Fraction of left (variable) nodes of degree i, i.e., the fraction of columns in H with weight i.
- R_i : Fraction of right (check) nodes of degree i, i.e., the fraction of rows in H with weight i.

The node-perspective polynomials are:

$$L(x) = \sum_{i=1}^{d_{v,\max}} L_i x^i, \quad R(x) = \sum_{i=1}^{d_{c,\max}} R_i x^i$$

The average degree of a variable node is $\bar{d}_v = \sum_{i=1}^{d_{v,\max}} i L_i = L'(1)$.

The average degree of a check node is $\bar{d}_c = \sum_{i=1}^{d_{c,max}} i R_i = R'(1)$.

The total number of edges in the graph (number of ones in H) is $\bar{d}_v n = \bar{d}_c (n-k)$.

We can also define degree distributions from the *edge perspective*:

- λ_i : Fraction of edges connected to variable nodes of degree i, i.e., the fraction of ones in H in columns of weight i.
- ρ_i : Fraction of edges connected to check nodes of degree i, i.e., the fraction of ones in H in rows of weight i.

The edge-perspective polynomials are:

$$\lambda(x) = \sum_{i=1}^{d_{v,max}} \lambda_i x^{i-1}, \quad \rho(x) = \sum_{i=1}^{d_{c,\max}} \rho_i x^{i-1}$$

The average variable node degreee and average check node degree satisfy:

$$\bar{d}_v = \left(\int_0^1 \lambda(x) dx\right)^{-1}$$

$$\bar{d}_c = \left(\int_0^1 \rho(x) dx\right)^{-1}$$

The design rate can be expressed as:

$$\frac{k}{n} = 1 - \frac{\bar{d}_v}{\bar{d}_c} = 1 - \frac{\int_0^1 \rho(x) dx}{\int_0^1 \lambda(x) dx}$$

4.11 Density Evolution

For *regular* LDPC codes:

Let p_t denote the probability that an outgoing $v \to c$ message (along an edge picked uniformly at random) is an erasure (?) in step t.

$$p_t = \varepsilon (q_{t-1})^{d_v - 1}$$

Let q_t denote the probability that an outgoing $c \to v$ message is a ? in step t.

$$q_t = 1 - (1 - p_t)^{d_c - 1}$$

The *density evolution* recursion predicts the fraction of erased bits at the end of each step t. We initialise the recursion with $p_0 = \varepsilon$ and $q_0 = 1$.

$$p_t = \varepsilon \left(1 - (1 - p_{t-1})^{d_c - 1}\right)^{d_v - 1}$$

The Shannon limit: The maximum possible ε for reliable decoding with any rate R code, is $\varepsilon^* = 1 - R = 0.5$.

For *irregular* LDPC ensembles with $\lambda(x) = \sum_i \lambda_i x^{i-1}$ and $\rho(x) = \sum_i \rho_i x^{i-1}$.

$$p_t = \varepsilon \sum_i \lambda_i q_{t-1}^{i-1} = \varepsilon \lambda (q_{t-1})$$

$$q_t = 1 - \sum_j \rho_j (1 - p_t)^{j-1} = 1 - \rho (1 - p_t)$$

The density evolution equation for a $(\lambda(x), \rho(x))$ ensemble:

$$p_t = \varepsilon \lambda (1 - \rho (1 - p_{t-1}))$$

For a given rate R, we want to get the maximum possible threshold ε^{MP} for which $p_t \to 0$.

4.12 Message Passing Decoding

In each iteration, the message passing decodes computes:

1. Variable-to-check messages:

$$m_{ji}(0) \propto P\left(c_j = 0 \mid y_j\right) \prod_{i' \backslash i} m_{i'j}(0)$$

$$m_{ji}(1) \propto P\left(c_j = 1 \mid y_j\right) \prod_{i' \backslash i} m_{i'j}(1)$$

$$m_{ji}(0) + m_{ji}(1) = 1$$

   $m_{ji}(0)$ is an updated estimate of the posterior probability (or belief) that the code bit $c_j = 0$.

2. Check-to-variable message:

$$m_{ij}(0) = \frac{1}{2} + \frac{1}{2} \prod_{j' \backslash j} \left(1 - 2m_{j'i}(1)\right)$$

$$m_{ij}(1) = 1 - m_{ij}(0)$$

   $m_{ij}(0)$ is an updated estimate of the probability that the parity check equation i is satisfied when $c_j = 0$.

At $t = 1$, set $m_{ji}(0) = P\left(c_j = 0 \mid y_j\right)$ for all edges $j \to i$. Also set $m_{ij}(0) = \frac{1}{2}$ for all edges $i \to j$. The message passing decoding algorithm is often called the *sum-product algorithm* or *belief propagation*.

4.13 Computing a Posteriori Probabilities (APPs)

The *a posteriori probabilities* (APPs) can be calculated using Bayes rule and the channel transition probabilities:

$$P\left(c_j \mid y_j\right) = \frac{P\left(c_j\right) P\left(y_j \mid c_j\right)}{P\left(y_j\right)}$$

For $\text{BEC}(\varepsilon) : y_j \in \{0, 1, ?\}$:

$$P\left(c_j = 0 \mid y_j\right) = \begin{cases} 1, & \text{if } y_j = 0 \\ 0, & \text{if } y_j = 1 \\ 1/2, & \text{if } y_j = ? \end{cases}$$

For $\text{BSC}(p) : y_j \in \{0, 1\}$:

$$P\left(c_j = 0 \mid y_j\right) = \begin{cases} 1 - p, & \text{if } y_j = 0 \\ p, & \text{if } y_j = 1 \end{cases}$$

For B-AWGN channel: $y_j \in \mathbb{R}$:

$$P\left(c_j = 0 \mid y_j\right) = \frac{1}{1 + e^{\frac{-2y_j}{\sigma^2}}}$$

4.14 Log-Domain Message Passing

The belief propagation decoding is usually implemented with log-likelihood ratios (LLRs).

$$L_{ji} = \ln \frac{m_{ji}(0)}{m_{ji}(1)}, \quad L_{ij} = \ln \frac{m_{ij}(0)}{m_{ij}(1)}$$

The LLR-based belief propagation updates are given by:

1. Variable-to-check message:

$$L_{ji} = L\left(y_j\right) + \sum_{i' \backslash i} L_{i'j}$$

2. Check-to-variable message:

$$L_{ij} = 2 \tanh^{-1} \left[\prod_{j' \backslash j} \tanh\left(\frac{1}{2} L_{j'i}\right) \right]$$

At $t = 1$, set $L_{ji} = L\left(y_j\right)$ for all edges $j \to i$, and set $L_{ij} = 0$ for all edges $i \to j$.

(The End)