Information and Coding Theory **Winter 2025**

Lecture 13: February 18, 2025

Lecturer: Madhur Tulsiani

1 Achieving capacity for the binary symmetric channel

We will show next that a random collection of codewords (called *codebook* or simply *code*) can achieve capacity for the binary symmetric channel $\mathrm{BSC}(p)$. Recall that the capacity for the channel is $1 - H_2(p)$. We will show that for every $\varepsilon > 0$, there is a sequence of codes with rate at least $1 - H_2(p) - \varepsilon$ and vanishing probability of error. We can assume that $p < 1/2$ (why?)

The code construction. For parameters R to be chosen later, let $M = 2^{nR}$. We define the codewords, the maps Enc and Dec as below. We will use $\Delta(\overline{\mathbf{x}}, \overline{\mathbf{y}})$ for $\overline{\mathbf{x}}, \overline{\mathbf{y}} \in \{0,1\}$ to denote the Hamming distance, i.e., the number of positions in which the two strings differ.

- **Codewords:** Select M independent random codewords $\overline{\mathbf{x}}_1, \ldots, \overline{\mathbf{x}}_M \in \{0,1\}^n$ with each bit of each codeword chosen independently and uniformly at random in $\{0,1\}$. Here we are using the fact that for $\mathrm{BSC}(p)$, the distribution $P(X)$ maximizing the mutual information is uniform on $\{0,1\}$. For the case of general channels and alphabet $\mathcal{X}$, each symbol is chosen independently from the distribution $P(X)$ maximizing the mutual information $I(X;Y)$.
- **Encoding:** For each $w \in [M]$, define $\mathsf{Enc}(w) = \overline{\mathbf{x}}_w$ (the w-th codeword).
- **Decoding:** Given $\overline{\mathbf{y}} \in \{0,1\}^n$, define $\mathsf{Dec}(\overline{\mathbf{y}})$ as

$$\mathsf{Dec}(\overline{\mathbf{y}}) = \begin{cases} w & \text{if } \exists \text{ unique } w \in [M] \text{ s.t. } \Delta(\overline{\mathbf{x}}_w, \overline{\mathbf{y}}) \leq (p+\delta) \cdot n \\ \text{arbitrary} & \text{otherwise} \end{cases}.$$

  Note that we will always count the second case towards the error probability, so we don't care how the decoding is defined there.

Before analyzing the error probability, we note that the noise in $\mathrm{BSC}(p)$ can be written a nice form. For input and output sequences $\overline{\mathbf{x}}$ and $\overline{\mathbf{y}}$, we can write $\overline{\mathbf{y}} = \overline{\mathbf{x}} + \overline{\mathbf{z}} \mod 2$, where $\overline{\mathbf{z}} \in \{0,1\}^n$ is a sequence each bit independently 1 with probability p and 0 with probability

$1-p$. We will refer to this distribution for each bit of $\overline{\mathbf{z}}$ as the Bernoulli distribution with parameter p, denoted $\mathsf{Bern}(p)$. We thus have $\overline{\mathbf{z}} \sim (\mathsf{Bern}(p))^n$.

We now analyze the *expected* probability of error for a random collection of codewords C, chosen as above. Obtaining a bound on the error probability (for each n) will show that there *exists* a good collection of codewords for each n, although we don't explicitly know what this code is. We will discuss explicit constructions in the next lecture. We now prove the following.

Claim 1.1. *Let C be random code constructed as above. Then*

$$\mathop{\mathbb{E}}_{C}[p_e] \;\leq\; n \cdot 2^{-n \cdot D(p+\delta \| p)} \;+\; 2^{nR} \cdot n \cdot 2^{-n \cdot D(p+\delta \| \frac{1}{2})},$$

where $D(p\|q)$ denotes $D(\mathsf{Bern}(p)\|\mathsf{Bern}(q))$ as usual.

Proof: We get

$$\mathop{\mathbb{E}}_{C}[p_e] \;=\; \mathop{\mathbb{E}}_{C}\left[\mathbb{P}\left[\widehat{W} \neq W\right]\right] \;=\; \mathop{\mathbb{E}}_{C}\left[\sum_{w \in [M]} \frac{1}{M} \cdot \mathbb{P}\left[\widehat{W} \neq w | W = w\right]\right].$$

By symmetry in the code construction, we can say that $\mathbb{E}_C\left[\mathbb{P}\left[\widehat{W} \neq w | W = w\right]\right]$ is the same for all $w \in [M]$. Replacing all these by the case for $w = 1$, we get

$$\mathop{\mathbb{E}}_{C}[p_e] \;=\; \mathop{\mathbb{E}}_{C}\left[\mathbb{P}\left[\widehat{W} \neq 1 | W = 1\right]\right].$$

We consider two cases in which we can have an error: either the output $\overline{\mathbf{y}}$ of the channel was too far from the input $\overline{\mathbf{x}}_1$, or $\Delta(\overline{\mathbf{x}}_w, \overline{\mathbf{y}}) \leq (p+\delta) \cdot n$ for some other $w > 1$. Thus, we have

$$\mathop{\mathbb{E}}_{C}[p_e] \;\leq\; \mathop{\mathbb{E}}_{C}\left[\mathbb{P}\left[\Delta(\overline{\mathbf{x}}_1, \overline{\mathbf{y}}) > (p+\delta) \cdot n\right]\right] + \sum_{w>1} \mathop{\mathbb{E}}_{C}\left[\mathbb{P}\left[\Delta(\overline{\mathbf{x}}_w, \overline{\mathbf{y}}) \leq (p+\delta) \cdot n\right]\right]$$

For a fixed $\overline{\mathbf{x}}_1$, let $\overline{\mathbf{y}} = \overline{\mathbf{x}}_1 + \overline{\mathbf{z}} \mod 2$, where $\overline{\mathbf{z}} \sim \mathsf{Bern}(p)$ is independent of $\overline{\mathbf{x}}_1$. The event $\Delta(\overline{\mathbf{x}}_1, \overline{\mathbf{y}}) > (p+\delta) \cdot n$ can be written in terms of the "type" $P_{\overline{\mathbf{z}}}$ of $\overline{\mathbf{z}}$ as $P_{\overline{\mathbf{z}}} \in \Pi$, where $\Pi = \{\mathsf{Bern}(p') \mid p' > p + \delta\}$. By Sanov's theorem, we then have that for each fixed $\overline{\mathbf{x}}_1$

$$\mathop{\mathbb{P}}_{\overline{\mathbf{y}}}\left[\Delta(\overline{\mathbf{x}}_1, \overline{\mathbf{y}}) > (p+\delta) \cdot n\right] \;\leq\; n \cdot 2^{-n \cdot D(p+\delta \| p)}.$$

For the second term, we use the fact that for each $\overline{\mathbf{y}}$ (which may depend on $\overline{\mathbf{x}}_1$), $\overline{\mathbf{x}}_w$ is independent of $\overline{\mathbf{y}}$ for all $w > 1$ (since codewords are chosen independently). Now defining $\overline{\mathbf{z}}$ so that $\overline{\mathbf{y}} + \overline{\mathbf{x}}_w = \overline{\mathbf{z}} \mod 2$, we get that $\overline{\mathbf{z}} \sim (\mathsf{Bern}(1/2))^n$ (why?) For this $\overline{\mathbf{z}}$, we can now

write the even $\Delta(\overline{\mathbf{x}}_w, \overline{\mathbf{y}}) \leq (p+\delta)\cdot n$ as $P_{\overline{\mathbf{z}}} \in \Pi'$, where $\Pi' = \{\mathsf{Bern}(p') | p' \leq p+\delta\}$. Applying Sanov's theorem again, we get that

$$\mathop{\mathbb{P}}_{\overline{\mathbf{x}}_w}\left[\Delta(\overline{\mathbf{x}}_w, \overline{\mathbf{y}}) \leq (p+\delta)\cdot n\right] \;\leq\; n\cdot 2^{-n\cdot D(p+\delta\|\frac{1}{2})}\,.$$

Combining the above bounds, we get

$$\mathop{\mathbb{E}}_{C}\left[p_e\right] \;\leq\; n\cdot 2^{-n\cdot D(p+\delta\|p)} + \sum_{w>1} n\cdot 2^{-n\cdot D(p+\delta\|\frac{1}{2})} \;\leq\; n\cdot 2^{-n\cdot D(p+\delta\|p)} \;+\; 2^{nR}\cdot n\cdot 2^{-n\cdot D(p+\delta\|\frac{1}{2})}\,,$$

as claimed. ∎

To analyze the bound, and compare it to the channel capacity $1-H_2(p)$, we note that $D(p+\delta\|\frac{1}{2}) = 1-H_2(p+\delta)$. Check that $\forall \varepsilon > 0$, there exists $\delta > 0$ such that $H_2(p+\delta) \leq H_2(p)+\varepsilon$. Using a δ such that $H_2(p+\delta) \leq H_2(p)+\varepsilon/2$, we get that

$$\mathop{\mathbb{E}}_{C}\left[p_e\right] \;\leq\; n\cdot 2^{-n\cdot D(p+\delta\|p)} \;+\; 2^{nR}\cdot n\cdot 2^{-n\cdot(1-H_2(p)-\varepsilon/2)}\,,$$

which tends to zero for $R = (1-H_2(p)-\varepsilon)$. Thus, for every $\varepsilon > 0$, we have a sequence of codes (as $n\to\infty$) with rate at least $(1-H_2(p)-\varepsilon)$, and $p_e^{(n)} \to 0$.

Exercise 1.2. *For $R = 1-H_2(p)-\varepsilon$ in the above proof, let $n_0(\varepsilon)$ be the smallest n (block-length) such that the probability of error $p_e^{(n)} \to 0$ for $n \geq n_0(\varepsilon)$. Check that $n_0(\varepsilon) = O(1/\varepsilon^2)$ suffices in the above proof.*

2 Linear Codes

A linear code $C \subseteq \mathbb{F}_q^n$ is a subspace of $\mathbb{F}_q^n$, viewed as a vector space over the finite field $\mathbb{F}_q$. We will always take q to be a prime number, with addition and multiplication in $\mathbb{F}_q$ defined modulo q (although the discussion can also be extended to the case when q is a prime power). If $\dim(C) = k$, we can think of C as encoding a message in $\mathbb{F}_q^k$ by *linearly* mapping it to an element $x \in C$. Overloading notation to denote $\mathsf{Enc}(w) \in C$ by $C(w)$, the encoding map $C : \mathbb{F}_q^k \to \mathbb{F}_q^n$ satisfies

$$C(\alpha\cdot u + \beta\cdot v) \;=\; \alpha\cdot C(u) + \beta\cdot C(v) \qquad \forall\, u,v \in \mathbb{F}_q^k,\ \alpha,\beta \in \mathbb{F}_q\,.$$

Since a linear encoding is a linear map from a finite dimensional vector space to another, we can write it as a matrix of finite size. That is, there is a corresponding $G \in \mathbb{F}_q^{n\times k}$ s.t. $C(w) = Gw$ for all $w \in \mathbb{F}_q^k$. This matrix is referred to as a generator matrix for the code C.

If the encoding map is injective (which is the bare minimum for a good code), then the rank of G must be k (otherwise there exist $u, v \in \mathbb{F}_q^k$ such that $Gu = Gv$). Hence, the null space of G^T has dimension $n-k$. This defines another useful matrix, known as the parity check matrix of the code.

Definition 2.1 (Parity Check Matrix). *Let $b_1, \ldots, b_{n-k} \in \mathbb{F}_q^n$ be a basis for the null space of G^T corresponding to a linear code C. Then $H \in \mathbb{F}_q^{(n-k)\times n}$, defined by*

$$H^T = \left[\; b_1 \;\middle|\; b_2 \;\middle|\; \ldots \;\middle|\; b_{n-k} \; \right]$$

is called a parity check matrix for C.

Remark 2.2. *As defined above, the generator and parity-check matrices for a code are not unique. However, the column span of G is unique (is equal to C), and so is the row-span of H. In many cases however, there is a canonical definition of the generator or parity-check matrix based on the construction of the code, which may be referred to as* the *generator or parity-check matrix.*

Since $G^T H^T = 0 \Leftrightarrow HG = 0$, we have $(HG)w = 0$ for all $x \in \mathbb{F}_q^k$, i.e., $Hx = 0$ for all $x \in C$. Moreover, since the columns of H^T are a basis for the null-space of G^T, we have that

$$x \in C \;\Leftrightarrow\; Hx = 0\,.$$

So the parity check matrix gives us a way to quickly check a codeword, by checking the parities of some bits of x (each row of H gives a parity constraint on x). Also, one can equivalently define a linear code by either giving G or the parity check matrix H.

Note that for linear codes, encoding $w \in F_2^k$ to the codeword $Gw \in C$ can always be done in polynomial time, by simply multiplying with the matrix G. Also, given $x \in C$, one can always find $w \in F_2^k$ such that $Gw = x$, either by Gaussian elimination, or (equivalently) by multiplying $x = Gw$ by an appropriate matrix G^* such that $G^*Gw = w$ for all $w \in \mathbb{F}_2^k$. Since we will only be concerned with polynomial time decoding in our discussion of codes, we can view the decoding problem as: given y which is a corruption of x, find x. The problem of going from $x \in \mathbb{F}_2^n$ to $w \in \mathbb{F}_2^k$ can always be solved for linear codes, as outlined above. Of course, if one is interested in a more fine-grained analysis of the decoding complexity, one needs to look carefully at the structure of the matrix G^*, but we will restrict our notion of efficiency to polynomial time.

2.1 Hamming Code

Consider the following code from $\mathbb{F}_2^4$ to $\mathbb{F}_2^7$, known as the Hamming Code.

Example 2.3. *Let $C : \mathbb{F}_2^4 \to \mathbb{F}_2^7$, where*

$$C(x_1, x_2, x_3, x_4) = (x_1, x_2, x_3, x_4, x_2 + x_3 + x_4, x_1 + x_3 + x_4, x_1 + x_2 + x_4).$$

Note that each element of the image is a linear function of the x_i's, i.e., one can express C with

matrix multiplication as follows:

$$C(x_1, x_2, x_3, x_4) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Example 2.4. *The parity check matrix of our example Hamming Code is:*

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

Note that the i^{th} column is the integer i in binary. One can easily check that $HG = 0$.

Now suppose $x = (x_1, \ldots, x_7)^T$ is our codeword and we make a single error in the i^{th} entry. Then the output codeword with the error is

$$x + e_i = \begin{bmatrix} x_1 \\ \vdots \\ x_i \\ \vdots \\ x_7 \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

and $H(x + e_i) = Hx + He_i = He_i = H_i$, the i^{th} column of H, which reads i in binary. So this is a very efficient decoding algorithm just based on parity checking. Thus, the Hamming code can correct one *arbitrary* error in any position. One can generalize the Hamming code to larger message and block lengths, we can create a parity matrix $H \in \mathbb{F}_2^{(n-k)\times n}$, where the i^{th} column reads i in binary.