About Chapter 14

In this chapter we will draw together several ideas that we've encountered so far in one nice short proof. We will simultaneously prove both Shannon's noisy-channel coding theorem (for symmetric binary channels) and his source coding theorem (for binary sources). While this proof has connections to many preceding chapters in the book, it's not essential to have read them all.

On the noisy-channel coding side, our proof will be more constructive than the proof given in Chapter 10; there, we proved that almost any random code is 'very good'. Here we will show that almost any *linear* code is very good. We will make use of the idea of typical sets (Chapters 4 and 10), and we'll borrow from the previous chapter's calculation of the weight enumerator function of random linear codes (section 13.5).

On the source coding side, our proof will show that *random linear hash functions* can be used for compression of compressible binary sources, thus giving a link to Chapter 12.

14

Very Good Linear Codes Exist

In this chapter we'll use a single calculation to prove simultaneously the source coding theorem and the noisy-channel coding theorem for the binary symmetric channel.

Incidentally, this proof works for much more general channel models, not only the binary symmetric channel. For example, the proof can be reworked for channels with non-binary outputs, for time-varying channels and for channels with memory, as long as they have binary inputs satisfying a symmetry property, c.f. section 10.6.

▶ 14.1 A simultaneous proof of the source coding and noisy-channel coding theorems

We consider a linear error-correcting code with binary parity-check matrix $\mathbf{H}$. The matrix has M rows and N columns. Later in the proof we will increase N and M, keeping $M \propto N$. The rate of the code satisfies

$$R \geq 1 - \frac{M}{N}. \tag{14.1}$$

If all the rows of $\mathbf{H}$ are independent then this is an equality, $R = 1 - M/N$. In what follows, we'll assume the equality holds. Eager readers may work out the expected rank of a random binary matrix $\mathbf{H}$ (it's very close to M) and pursue the effect that the difference ($M -$ rank) has on the rest of this proof (it's negligible).

A codeword $\mathbf{t}$ is selected, satisfying

$$\mathbf{Ht} = \mathbf{0} \bmod 2, \tag{14.2}$$

and a binary symmetric channel adds noise $\mathbf{x}$, giving the received signal

In this chapter $\mathbf{x}$ denotes the noise added by the channel, not the input to the channel.

$$\mathbf{r} = \mathbf{t} + \mathbf{x} \bmod 2. \tag{14.3}$$

The receiver aims to infer both $\mathbf{t}$ and $\mathbf{x}$ from $\mathbf{r}$ using a syndrome decoding approach. Syndrome decoding was first introduced in section 1.2 (p.10 and 11). The receiver computes the syndrome

$$\mathbf{z} = \mathbf{Hr} \bmod 2 = \mathbf{Ht} + \mathbf{Hx} \bmod 2 = \mathbf{Hx} \bmod 2. \tag{14.4}$$

The syndrome only depends on the noise $\mathbf{x}$, and the decoding problem is to find the most probable $\mathbf{x}$ that satisfies

$$\mathbf{Hx} = \mathbf{z} \bmod 2. \tag{14.5}$$

This best estimate for the noise vector, $\hat{\mathbf{x}}$, is then subtracted from $\mathbf{r}$ to give the best guess for $\mathbf{t}$. Our aim is to show that, as long as $R < 1-H(X) = 1-H_2(f)$, where f is the flip probability of the binary symmetric channel, the optimal decoder for this syndrome decoding problem has vanishing probability of error, as N increases, for random $\mathbf{H}$.

We prove this result by studying a sub-optimal strategy for solving the decoding problem. Neither the optimal decoder nor this *typical set decoder* would be easy to implement, but the typical set decoder is easier to analyze. The typical set decoder examines the typical set T of noise vectors, the set of noise vectors $\mathbf{x}'$ that satisfy $\log 1/P(\mathbf{x}') \simeq NH(X)$, checking to see if any of those typical vectors $\mathbf{x}'$ satisfies the observed syndrome,

We'll leave out the ϵs and βs that make a typical set definition rigorous. Enthusiasts are encouraged to revisit section 4.4 and put these details into this proof.

$$\mathbf{H}\mathbf{x}' = \mathbf{z}. \tag{14.6}$$

If exactly one typical vector $\mathbf{x}'$ does so, the typical set decoder reports that vector as the hypothesized noise vector. If no typical vector matches the observed syndrome, or more than one does, then the typical set decoder reports an error.

The probability of error of the typical set decoder, for a given matrix $\mathbf{H}$, can be written as a sum of two terms,

$$P_{\mathrm{TS}|\mathbf{H}} = P^{(I)} + P^{(II)}_{\mathrm{TS}|\mathbf{H}}, \tag{14.7}$$

where $P^{(I)}$ is the probability that the true noise vector $\mathbf{x}$ is itself not typical, and $P^{(II)}_{\mathrm{TS}|\mathbf{H}}$ is the probability that the true $\mathbf{x}$ is typical and at least one other typical vector clashes with it. The first probability vanishes as N increases, as we proved when we first studied typical sets (Chapter 4). We concentrate on the second probability. To recap, we're imagining a true noise vector, $\mathbf{x}$; and if *any* of the typical noise vectors $\mathbf{x}'$, different from $\mathbf{x}$, satisfies $\mathbf{H}(\mathbf{x}'-\mathbf{x}) = 0$, then we have an error. We use the truth function

$$\mathbb{1}\left[\mathbf{H}(\mathbf{x}'-\mathbf{x}) = 0\right], \tag{14.8}$$

whose value is one if the statement $\mathbf{H}(\mathbf{x}'-\mathbf{x}) = 0$ is true and zero otherwise. We can bound the number of type II errors made when the noise is $\mathbf{x}$ thus:

$$[\text{Number of errors given } \mathbf{x} \text{ and } \mathbf{H}] \leq \sum_{\mathbf{x}':\ \substack{\mathbf{x}' \in T \\ \mathbf{x}' \neq \mathbf{x}}} \mathbb{1}\left[\mathbf{H}(\mathbf{x}'-\mathbf{x}) = 0\right]. \tag{14.9}$$

The number of errors is either zero or one; the sum on the right-hand side may exceed one, in cases where several typical noise vectors have the same syndrome.

We can now write down the probability of a type-II error by averaging over $\mathbf{x}$:

$$P^{(II)}_{\mathrm{TS}|\mathbf{H}} \leq \sum_{\mathbf{x}\in T} P(\mathbf{x}) \sum_{\mathbf{x}':\ \substack{\mathbf{x}' \in T \\ \mathbf{x}' \neq \mathbf{x}}} \mathbb{1}\left[\mathbf{H}(\mathbf{x}'-\mathbf{x}) = 0\right]. \tag{14.10}$$

Now, we will find the average of this probability of type-II error over all linear codes by averaging over $\mathbf{H}$. By showing that the *average* probability of type-II error vanishes, we will thus show that there exist linear codes with vanishing error probability, indeed, that almost all linear codes are very good.

We denote averaging over all binary matrices $\mathbf{H}$ by $\langle \ldots \rangle_{\mathbf{H}}$. The average probability of type-II error is

$$\bar{P}^{(II)}_{\mathrm{TS}} = \sum_{\mathbf{H}} P(\mathbf{H}) P^{(II)}_{\mathrm{TS}|\mathbf{H}} = \left\langle P^{(II)}_{\mathrm{TS}|\mathbf{H}} \right\rangle_{\mathbf{H}} \tag{14.11}$$

$$= \left\langle \sum_{\mathbf{x}\in T} P(\mathbf{x}) \sum_{\mathbf{x}':\ \substack{\mathbf{x}'\in T \\ \mathbf{x}'\neq\mathbf{x}}} \mathbb{1}\big[\mathbf{H}(\mathbf{x}'-\mathbf{x})=0\big] \right\rangle_{\mathbf{H}} \tag{14.12}$$

$$= \sum_{\mathbf{x}\in T} P(\mathbf{x}) \sum_{\mathbf{x}':\ \substack{\mathbf{x}'\in T \\ \mathbf{x}'\neq\mathbf{x}}} \big\langle \mathbb{1}\big[\mathbf{H}(\mathbf{x}'-\mathbf{x})=0\big] \big\rangle_{\mathbf{H}} . \tag{14.13}$$

Now, the quantity $\langle \mathbb{1}[\mathbf{H}(\mathbf{x}'-\mathbf{x})=0]\rangle_{\mathbf{H}}$ already cropped up when we were calculating the expected weight enumerator function of random linear codes (section 13.5): for any non-zero binary vector $\mathbf{v}$, the probability that $\mathbf{Hv}=0$, averaging over all matrices $\mathbf{H}$, is 2^{-M}. So

$$\bar{P}_{\rm TS}^{(II)} = \left(\sum_{\mathbf{x}\in T} P(\mathbf{x}) \right) (|T|-1)\, 2^{-M} \tag{14.14}$$

$$\leq |T|\, 2^{-M}, \tag{14.15}$$

where $|T|$ denotes the size of the typical set. As you will recall from Chapter 4, there are roughly $2^{NH(X)}$ noise vectors in the typical set. So

$$\bar{P}_{\rm TS}^{(II)} \leq 2^{NH(X)} 2^{-M}. \tag{14.16}$$

This bound on the probability of error either vanishes or grows exponentially as N increases (remembering that we are keeping M proportional to N as N increases). It vanishes if

$$H(X) < M/N. \tag{14.17}$$

Substituting $R = 1 - M/N$, we have thus established the noisy-channel coding theorem for the binary symmetric channel: very good linear codes exist for any rate R satisfying

$$R < 1 - H(X), \tag{14.18}$$

where $H(X)$ is the entropy of the channel noise, per bit. □

Exercise 14.1.[3] Redo the proof for a more general channel.

▶ 14.2 Data compression by linear hash codes

The decoding game we have just played can also be viewed as an *uncompression* game. The world produces a noise vector $\mathbf{x}$ from a source $P(\mathbf{x})$. The noise has redundancy (if the flip probability is not 0.5). We compress it with a linear compressor that maps the N-bit input $\mathbf{x}$ (the noise) to the M-bit output $\mathbf{z}$ (the syndrome). Our uncompression task is to recover the input $\mathbf{x}$ from the output $\mathbf{z}$. The rate of the compressor is

$$R_{\rm compressor} \equiv M/N. \tag{14.19}$$

[We don't care about the possibility of linear redundancies in our definition of the rate, here.] The result that we just found, that the decoding problem can be solved, for almost any $\mathbf{H}$, with vanishing error probability, as long as $H(X) < M/N$, thus instantly proves a source coding theorem:

> Given a binary source X of entropy $H(X)$, and a required compressed rate $R > H(X)$, there exists a linear compressor $\mathbf{x} \to \mathbf{z} = \mathbf{Hx} \bmod 2$ having rate M/N equal to that required rate R, and an associated uncompressor, that is virtually lossless.

This theorem is true not only for a source of independent identically distributed symbols but also for any source for which a typical set can be defined: sources with memory, and time-varying sources, for example; all that's required is that the source be ergodic.

Notes

This method for proving that codes are good can be applied to other linear codes, such as low-density parity-check codes (MacKay, 1999b; Aji *et al.*, 2000). For each code we need an approximation of its expected weight enumerator function.