About Chapter 11

Before reading Chapter 11, you should have read Chapters 9 and 10.

You will also need to be familiar with the *Gaussian distribution*.

One-dimensional Gaussian distribution. If a random variable y is Gaussian and has mean μ and variance σ^2, which we write:

$$y \sim \text{Normal}(\mu, \sigma^2), \text{ or } P(y) = \text{Normal}(y; \mu, \sigma^2), \tag{11.1}$$

then the distribution of y is:

$$P(y \,|\, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-(y-\mu)^2/2\sigma^2\right]. \tag{11.2}$$

[I use the symbol P for both probability densities and probabilities.]

The inverse-variance $\tau \equiv {}^1\!/\!_{\sigma^2}$ is sometimes called the *precision* of the Gaussian distribution.

Multi-dimensional Gaussian distribution. If $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ has a multivariate Gaussian distribution, then

$$P(\mathbf{y} \,|\, \mathbf{x}, \mathbf{A}) = \frac{1}{Z(\mathbf{A})} \exp\left(-\frac{1}{2}(\mathbf{y}-\mathbf{x})^{\mathsf{T}}\mathbf{A}(\mathbf{y}-\mathbf{x})\right), \tag{11.3}$$

where $\mathbf{x}$ is the mean of the distribution, $\mathbf{A}$ is the inverse of the variance–covariance matrix, and the normalizing constant is $Z(\mathbf{A}) = (\det(\mathbf{A}/2\pi))^{-1/2}$.

This distribution has the property that the variance Σ_{ii} of y_i, and the covariance Σ_{ij} of y_i and y_j are given by

$$\Sigma_{ij} \equiv \mathcal{E}\left[(y_i - \bar{y}_i)(y_j - \bar{y}_j)\right] = A^{-1}_{ij}, \tag{11.4}$$

where $\mathbf{A}^{-1}$ is the inverse of the matrix $\mathbf{A}$.

The marginal distribution $P(y_i)$ of one component y_i is Gaussian; the joint marginal distribution of any subset of the components is multivariate-Gaussian; and the conditional density of any subset, given the values of another subset, for example, $P(y_i \,|\, y_j)$, is also Gaussian.

11

Error-Correcting Codes & Real Channels

The noisy-channel coding theorem that we have proved shows that there exist reliable error-correcting codes for any noisy channel. In this chapter we address two questions.

First, many practical channels have real, rather than discrete, inputs and outputs. What can Shannon tell us about these continuous channels? And how should digital signals be mapped into analogue waveforms, and *vice versa*?

Second, how are practical error-correcting codes made, and what is achieved in practice, relative to the possibilities proved by Shannon?

▶ 11.1 The Gaussian channel

The most popular model of a real-input, real-output channel is the Gaussian channel.

The Gaussian channel has a real input x and a real output y. The conditional distribution of y given x is a Gaussian distribution:

$$P(y \,|\, x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-(y-x)^2/2\sigma^2\right]. \tag{11.5}$$

This channel has a continuous input and output but is discrete in time. We will show below that certain continuous-time channels are equivalent to the discrete-time Gaussian channel.

This channel is sometimes called the additive white Gaussian noise (AWGN) channel.

As with discrete channels, we will discuss what rate of error-free information communication can be achieved over this channel.

Motivation in terms of a continuous-time channel

Consider a physical (electrical, say) channel with inputs and outputs that are continuous in time. We put in $x(t)$, and out comes $y(t) = x(t) + n(t)$.

Our transmission has a power cost. The average power of a transmission of length T may be constrained thus:

$$\int_0^T \mathrm{d}t\, [x(t)]^2/T \leq P. \tag{11.6}$$

The received signal is assumed to differ from $x(t)$ by additive noise $n(t)$ (for example Johnson noise), which we will model as white Gaussian noise. The magnitude of this noise is quantified by the *noise spectral density*, N_0.

How could such a channel be used to communicate information? Consider transmitting a set of N real numbers $\{x_n\}_{n=1}^{N}$ in a signal of duration T made up of a weighted combination of orthonormal basis functions $\phi_n(t)$,

$$x(t) = \sum_{n=1}^{N} x_n \phi_n(t), \tag{11.7}$$

where $\int_0^T \mathrm{d}t\, \phi_n(t)\phi_m(t) = \delta_{nm}$. The receiver can then compute the scalars:

$$\begin{aligned} y_n \;\equiv\; \int_0^T \mathrm{d}t\, \phi_n(t) y(t) \;&=\; x_n + \int_0^T \mathrm{d}t\, \phi_n(t) n(t) && (11.8)\\ &\equiv\; x_n + n_n && (11.9) \end{aligned}$$

for $n = 1 \ldots N$. If there were no noise, then y_n would equal x_n. The white Gaussian noise $n(t)$ adds scalar noise n_n to the estimate y_n. This noise is Gaussian:

$$n_n \sim \text{Normal}(0, N_0/2), \tag{11.10}$$

where N_0 is the spectral density introduced above. Thus a continuous channel used in this way is equivalent to the Gaussian channel defined at equation (11.5). The power constraint $\int_0^T \mathrm{d}t\, [x(t)]^2 \leq PT$ defines a constraint on the signal amplitudes x_n,

$$\sum_n x_n^2 \leq PT \qquad \Rightarrow \qquad \overline{x_n^2} \leq \frac{PT}{N}. \tag{11.11}$$

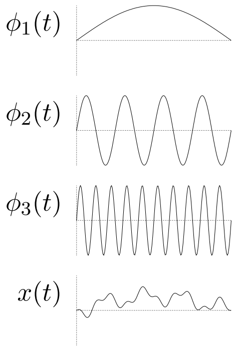

Figure 11.1. Three basis functions, and a weighted combination of them, $x(t) = \sum_{n=1}^{N} x_n \phi_n(t)$, with $x_1 = 0.4$, $x_2 = -0.2$, and $x_3 = 0.1$.

Before returning to the Gaussian channel, we define the *bandwidth* (measured in Hertz) of the continuous channel to be:

$$W = \frac{N^{\max}}{2T}, \tag{11.12}$$

where $N^{\max}$ is the maximum number of orthonormal functions that can be produced in an interval of length T. This definition can be motivated by imagining creating a band-limited signal of duration T from orthonormal cosine and sine curves of maximum frequency W. The number of orthonormal functions is $N^{\max} = 2WT$. This definition relates to the Nyquist sampling theorem: if the highest frequency present in a signal is W, then the signal can be fully determined from its values at a series of discrete sample points separated by the Nyquist interval $\Delta t = {}^1\!/_{2W}$ seconds.

So the use of a real continuous channel with bandwidth W, noise spectral density N_0, and power P is equivalent to $N/T = 2W$ uses per second of a Gaussian channel with noise level $\sigma^2 = N_0/2$ and subject to the signal power constraint $\overline{x_n^2} \leq {}^P\!/_{2W}$.

Definition of E_b/N_0

Imagine that the Gaussian channel $y_n = x_n + n_n$ is used with an encoding system to transmit *binary* source bits at a rate of R bits per channel use. How can we compare two encoding systems that have different rates of communication R and that use different powers $\overline{x_n^2}$? Transmitting at a large rate R is good; using small power is good too.

It is conventional to measure the rate-compensated signal-to-noise ratio by the ratio of the power per source bit $E_\mathrm{b} = \overline{x_n^2}/R$ to the noise spectral density N_0:

$$E_\mathrm{b}/N_0 = \frac{\overline{x_n^2}}{2\sigma^2 R}. \tag{11.13}$$

E_b/N_0 is dimensionless, but it is usually reported in the units of decibels; the value given is $10 \log_{10} E_\mathrm{b}/N_0$.

E_b/N_0 is one of the measures used to compare coding schemes for Gaussian channels.

▶ 11.2 Inferring the input to a real channel

'The best detection of pulses'

In 1944 Shannon wrote a memorandum (Shannon, 1993) on the problem of best differentiating between two types of pulses of known shape, represented by vectors $\mathbf{x}_0$ and $\mathbf{x}_1$, given that one of them has been transmitted over a noisy channel. This is a pattern recognition problem. It is assumed that the noise is Gaussian with probability density

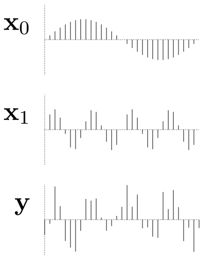

Figure 11.2. Two pulses $\mathbf{x}_0$ and $\mathbf{x}_1$, represented as 31-dimensional vectors, and a noisy version of one of them, $\mathbf{y}$.

$$P(\mathbf{n}) = \left[\det\left(\frac{\mathbf{A}}{2\pi}\right)\right]^{1/2} \exp\left(-\frac{1}{2}\mathbf{n}^{\mathsf{T}}\mathbf{A}\mathbf{n}\right), \qquad (11.14)$$

where $\mathbf{A}$ is the inverse of the variance–covariance matrix of the noise, a symmetric and positive-definite matrix. (If $\mathbf{A}$ is a multiple of the identity matrix, $\mathbf{I}/\sigma^2$, then the noise is 'white'. For more general $\mathbf{A}$, the noise is 'coloured'.) The probability of the received vector $\mathbf{y}$ given that the source signal was s (either zero or one) is then

$$P(\mathbf{y}\,|\,s) = \left[\det\left(\frac{\mathbf{A}}{2\pi}\right)\right]^{1/2} \exp\left(-\frac{1}{2}(\mathbf{y}-\mathbf{x}_s)^{\mathsf{T}}\mathbf{A}(\mathbf{y}-\mathbf{x}_s)\right). \qquad (11.15)$$

The optimal detector is based on the posterior probability ratio:

$$\begin{aligned}\frac{P(s=1\,|\,\mathbf{y})}{P(s=0\,|\,\mathbf{y})} &= \frac{P(\mathbf{y}\,|\,s=1)}{P(\mathbf{y}\,|\,s=0)}\frac{P(s=1)}{P(s=0)} & (11.16)\\ &= \exp\left(-\frac{1}{2}(\mathbf{y}-\mathbf{x}_1)^{\mathsf{T}}\mathbf{A}(\mathbf{y}-\mathbf{x}_1)+\frac{1}{2}(\mathbf{y}-\mathbf{x}_0)^{\mathsf{T}}\mathbf{A}(\mathbf{y}-\mathbf{x}_0)+\ln\frac{P(s=1)}{P(s=0)}\right)\\ &= \exp\left(\mathbf{y}^{\mathsf{T}}\mathbf{A}(\mathbf{x}_1-\mathbf{x}_0)+\theta\right), & (11.17)\end{aligned}$$

where θ is a constant independent of the received vector $\mathbf{y}$,

$$\theta = -\frac{1}{2}\mathbf{x}_1^{\mathsf{T}}\mathbf{A}\mathbf{x}_1 + \frac{1}{2}\mathbf{x}_0^{\mathsf{T}}\mathbf{A}\mathbf{x}_0 + \ln\frac{P(s=1)}{P(s=0)}. \qquad (11.18)$$

If the detector is forced to make a decision (i.e., guess either $s=1$ or $s=0$) then the decision that minimizes the probability of error is to guess the most probable hypothesis. We can write the optimal decision in terms of a *discriminant function*:

$$a(\mathbf{y}) \equiv \mathbf{y}^{\mathsf{T}}\mathbf{A}(\mathbf{x}_1-\mathbf{x}_0)+\theta \qquad (11.19)$$

with the decisions

$$\begin{array}{lcl} a(\mathbf{y})>0 & \rightarrow & \text{guess } s=1\\ a(\mathbf{y})<0 & \rightarrow & \text{guess } s=0\\ a(\mathbf{y})=0 & \rightarrow & \text{guess either.}\end{array} \qquad (11.20)$$

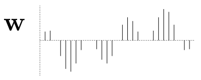

Figure 11.3. The weight vector $\mathbf{w} \propto \mathbf{x}_1 - \mathbf{x}_0$ that is used to discriminate between $\mathbf{x}_0$ and $\mathbf{x}_1$.

Notice that $a(\mathbf{y})$ is a linear function of the received vector,

$$a(\mathbf{y}) = \mathbf{w}^{\mathsf{T}}\mathbf{y}+\theta, \qquad (11.21)$$

where $\mathbf{w} \equiv \mathbf{A}(\mathbf{x}_1-\mathbf{x}_0)$.

▶ 11.3 Capacity of Gaussian channel

Until now we have measured the joint, marginal, and conditional entropy of discrete variables only. In order to define the information conveyed by continuous variables, there are two issues we must address – the infinite length of the real line, and the infinite precision of real numbers.

Infinite inputs

How much information can we convey in one use of a Gaussian channel? If we are allowed to put *any* real number x into the Gaussian channel, we could communicate an enormous string of N digits $d_1 d_2 d_3 \ldots d_N$ by setting $x = d_1 d_2 d_3 \ldots d_N 000 \ldots 000$. The amount of error-free information conveyed in just a single transmission could be made arbitrarily large by increasing N, and the communication could be made arbitrarily reliable by increasing the number of zeroes at the end of x. There is usually some power cost associated with large inputs, however, not to mention practical limits in the dynamic range acceptable to a receiver. It is therefore conventional to introduce a cost function $v(x)$ for every input x, and constrain codes to have an average cost $\bar{v}$ less than or equal to some maximum value. A generalized channel coding theorem, including a cost function for the inputs, can be proved – see McEliece (1977). The result is a channel capacity $C(\bar{v})$ that is a function of the permitted cost. For the Gaussian channel we will assume a cost

$$v(x) = x^2 \tag{11.22}$$

such that the 'average power' $\overline{x^2}$ of the input is constrained. We motivated this cost function above in the case of real electrical channels in which the physical power consumption is indeed quadratic in x. The constraint $\overline{x^2} = \bar{v}$ makes it impossible to communicate infinite information in one use of the Gaussian channel.

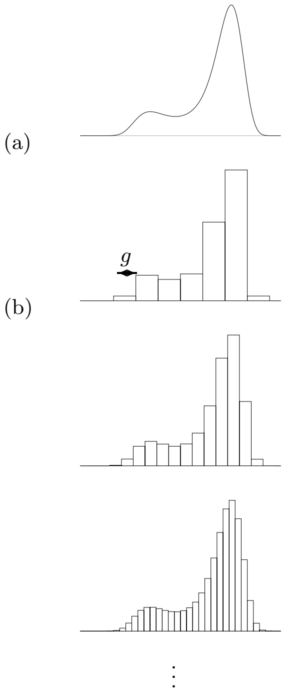

Figure 11.4. (a) A probability density $P(x)$. Question: can we define the 'entropy' of this density? (b) We could evaluate the entropies of a sequence of probability distributions with decreasing grain-size g, but these entropies tend to $\int P(x) \log \frac{1}{P(x)g} \, dx$, which is not independent of g: the entropy goes up by one bit for every halving of g.
$\int P(x) \log \frac{1}{P(x)} \, dx$ is an illegal integral.

Infinite precision

It is tempting to define joint, marginal, and conditional entropies for real variables simply by replacing summations by integrals, but this is not a well defined operation. As we discretize an interval into smaller and smaller divisions, the entropy of the discrete distribution diverges (as the logarithm of the granularity) (figure 11.4). Also, it is not permissible to take the logarithm of a dimensional quantity such as a probability density $P(x)$ (whose dimensions are $[x]^{-1}$).

There is one information measure, however, that has a well-behaved limit, namely the mutual information – and this is the one that really matters, since it measures how much information one variable conveys about another. In the discrete case,

$$I(X;Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}. \tag{11.23}$$

Now because the argument of the log is a ratio of two probabilities over the same space, it is OK to have $P(x,y)$, $P(x)$ and $P(y)$ be probability densities and replace the sum by an integral:

$$\begin{aligned} I(X;Y) &= \int dx\, dy\, P(x,y) \log \frac{P(x,y)}{P(x)P(y)} && (11.24) \\ &= \int dx\, dy\, P(x)P(y\,|\,x) \log \frac{P(y\,|\,x)}{P(y)}. && (11.25) \end{aligned}$$

We can now ask these questions for the Gaussian channel: (a) what probability distribution $P(x)$ maximizes the mutual information (subject to the constraint $\overline{x^2} = v$)? and (b) does the maximal mutual information still measure the maximum error-free communication rate of this real channel, as it did for the discrete channel?

Exercise 11.1.[3, p.189] Prove that the probability distribution $P(x)$ that maximizes the mutual information (subject to the constraint $\overline{x^2} = v$) is a Gaussian distribution of mean zero and variance v.

▷ Exercise 11.2.[2, p.189] Show that the mutual information $I(X;Y)$, in the case of this optimized distribution, is

$$C = \frac{1}{2}\log\left(1+\frac{v}{\sigma^2}\right). \tag{11.26}$$

This is an important result. We see that the capacity of the Gaussian channel is a function of the *signal-to-noise ratio* v/σ^2.

Inferences given a Gaussian input distribution

If $P(x) = \text{Normal}(x;0,v)$ and $P(y \mid x) = \text{Normal}(y;x,\sigma^2)$ then the marginal distribution of y is $P(y) = \text{Normal}(y;0,v{+}\sigma^2)$ and the posterior distribution of the input, given that the output is y, is:

$$\begin{aligned} P(x \mid y) &\propto P(y \mid x)P(x) && (11.27)\\ &\propto \exp(-(y-x)^2/2\sigma^2)\exp(-x^2/2v) && (11.28)\\ &= \text{Normal}\left(x; \frac{v}{v+\sigma^2}\,y\,,\,\left(\frac{1}{v}+\frac{1}{\sigma^2}\right)^{-1}\right). && (11.29)\end{aligned}$$

[The step from (11.28) to (11.29) is made by completing the square in the exponent.] This formula deserves careful study. The mean of the posterior distribution, $\frac{v}{v+\sigma^2}\,y$, can be viewed as a weighted combination of the value that best fits the output, $x = y$, and the value that best fits the prior, $x = 0$:

$$\frac{v}{v+\sigma^2}\,y = \frac{1/\sigma^2}{1/v+1/\sigma^2}\,y + \frac{1/v}{1/v+1/\sigma^2}\,0. \tag{11.30}$$

The weights $1/\sigma^2$ and $1/v$ are the *precisions* of the two Gaussians that we multiplied together in equation (11.28): the prior and the likelihood.

The precision of the posterior distribution is the sum of these two precisions. This is a general property: whenever two independent sources contribute information, via Gaussian distributions, about an unknown variable, the precisions add. [This is the dual to the better-known relationship 'when independent variables are added, their variances add'.]

Noisy-channel coding theorem for the Gaussian channel

We have evaluated a maximal mutual information. Does it correspond to a maximum possible rate of error-free information transmission? One way of proving that this is so is to define a sequence of discrete channels, all derived from the Gaussian channel, with increasing numbers of inputs and outputs, and prove that the maximum mutual information of these channels tends to the asserted C. The noisy-channel coding theorem for discrete channels applies to each of these derived channels, thus we obtain a coding theorem for the continuous channel. Alternatively, we can make an intuitive argument for the coding theorem specific for the Gaussian channel.

Geometrical view of the noisy-channel coding theorem: sphere packing

Consider a sequence $\mathbf{x} = (x_1, \ldots, x_N)$ of inputs, and the corresponding output $\mathbf{y}$, as defining two points in an N dimensional space. For large N, the noise power is very likely to be close (fractionally) to $N\sigma^2$. The output $\mathbf{y}$ is therefore very likely to be close to the surface of a sphere of radius $\sqrt{N\sigma^2}$ centred on $\mathbf{x}$. Similarly, if the original signal $\mathbf{x}$ is generated at random subject to an average power constraint $\overline{x^2} = v$, then $\mathbf{x}$ is likely to lie close to a sphere, centred on the origin, of radius $\sqrt{Nv}$; and because the total average power of $\mathbf{y}$ is $v + \sigma^2$, the received signal $\mathbf{y}$ is likely to lie on the surface of a sphere of radius $\sqrt{N(v + \sigma^2)}$, centred on the origin.

The volume of an N-dimensional sphere of radius r is

$$V(r, N) = \tfrac{\pi^{N/2}}{\Gamma(N/2+1)} r^N. \tag{11.31}$$

Now consider making a communication system based on non-confusable inputs $\mathbf{x}$, that is, inputs whose spheres do not overlap significantly. The maximum number S of non-confusable inputs is given by dividing the volume of the sphere of probable $\mathbf{y}$s by the volume of the sphere for $\mathbf{y}$ given $\mathbf{x}$:

$$S \leq \left(\frac{\sqrt{N(v + \sigma^2)}}{\sqrt{N\sigma^2}} \right)^N \tag{11.32}$$

Thus the capacity is bounded by:

$$C = \frac{1}{N} \log M \leq \frac{1}{2} \log \left(1 + \frac{v}{\sigma^2} \right). \tag{11.33}$$

A more detailed argument like the one used in the previous chapter can establish equality.

Back to the continuous channel

Recall that the use of a real continuous channel with bandwidth W, noise spectral density N_0 and power P is equivalent to $N/T = 2W$ uses per second of a Gaussian channel with $\sigma^2 = N_0/2$ and subject to the constraint $\overline{x_n^2} \leq P/2W$. Substituting the result for the capacity of the Gaussian channel, we find the capacity of the continuous channel to be:

$$C = W \log \left(1 + \frac{P}{N_0 W} \right) \quad \text{bits per second.} \tag{11.34}$$

This formula gives insight into the tradeoffs of practical communication. Imagine that we have a fixed power constraint. What is the best bandwidth to make use of that power? Introducing $W_0 = P/N_0$, i.e., the bandwidth for which the signal-to-noise ratio is 1, figure 11.5 shows $C/W_0 = W/W_0 \log(1 + W_0/W)$ as a function of W/W_0. The capacity increases to an asymptote of $W_0 \log e$. It is dramatically better (in terms of capacity for fixed power) to transmit at a low signal-to-noise ratio over a large bandwidth, than with high signal-to-noise in a narrow bandwidth; this is one motivation for wideband communication methods such as the 'direct sequence spread-spectrum' approach used in 3G mobile phones. Of course, you are not alone, and your electromagnetic neighbours may not be pleased if you use a large bandwidth, so for social reasons, engineers often have to make do with higher-power, narrow-bandwidth transmitters.

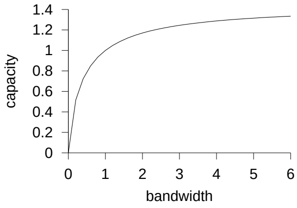

Figure 11.5. Capacity versus bandwidth for a real channel: $C/W_0 = W/W_0 \log(1 + W_0/W)$ as a function of W/W_0.

▶ 11.4 What are the capabilities of practical error-correcting codes?

Nearly all codes are good, but nearly all codes require exponential look-up tables for practical implementation of the encoder and decoder – exponential in the blocklength N. And the coding theorem required N to be large.

By a *practical* error-correcting code, we mean one that can be encoded and decoded in a reasonable amount of time, for example, a time that scales as a polynomial function of the blocklength N – preferably linearly.

The Shannon limit is not achieved in practice

The non-constructive proof of the noisy-channel coding theorem showed that good block codes exist for any noisy channel, and indeed that nearly all block codes are good. But writing down an explicit and practical encoder and decoder that are as good as promised by Shannon is still an unsolved problem.

Very good codes. Given a channel, a family of block codes that achieve arbitrarily small probability of error at any communication rate up to the capacity of the channel are called 'very good' codes for that channel.

Good codes are code families that achieve arbitrarily small probability of error at non-zero communication rates up to some maximum rate that may be *less than* the capacity of the given channel.

Bad codes are code families that cannot achieve arbitrarily small probability of error, or that can achieve arbitrarily small probability of error only by decreasing the information rate to zero. Repetition codes are an example of a bad code family. (Bad codes are not necessarily useless for practical purposes.)

Practical codes are code families that can be encoded and decoded in time and space polynomial in the blocklength.

Most established codes are linear codes

Let us review the definition of a block code, and then add the definition of a linear block code.

An (N, K) **block code** for a channel Q is a list of $S = 2^K$ codewords $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(2^K)}\}$, each of length N: $\mathbf{x}^{(s)} \in \mathcal{A}_X^N$. The signal to be encoded, s, which comes from an alphabet of size 2^K, is encoded as $\mathbf{x}^{(s)}$.

A linear (N, K) **block code** is a block code in which the codewords $\{\mathbf{x}^{(s)}\}$ make up a K-dimensional subspace of $\mathcal{A}_X^N$. The encoding operation can be represented by an $N \times K$ binary matrix $\mathbf{G}^{\mathsf{T}}$ such that if the signal to be encoded, in binary notation, is $\mathbf{s}$ (a vector of length K bits), then the encoded signal is $\mathbf{t} = \mathbf{G}^{\mathsf{T}}\mathbf{s}$ modulo 2.

The codewords $\{\mathbf{t}\}$ can be defined as the set of vectors satisfying $\mathbf{Ht} = \mathbf{0} \bmod 2$, where $\mathbf{H}$ is the *parity-check matrix* of the code.

$$\mathbf{G}^{\mathsf{T}} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & 1 \\ 1 & 1 & 1 & \cdot \\ \cdot & 1 & 1 & 1 \\ 1 & \cdot & 1 & 1 \end{bmatrix}$$

For example the $(7, 4)$ Hamming code of section 1.2 takes $K = 4$ signal bits, $\mathbf{s}$, and transmits them followed by three parity-check bits. The $N = 7$ transmitted symbols are given by $\mathbf{G}^{\mathsf{T}}\mathbf{s} \bmod 2$.

Coding theory was born with the work of Hamming, who invented a family of practical error-correcting codes, each able to correct one error in a block of length N, of which the repetition code R_3 and the $(7, 4)$ code are the simplest. Since then most established codes have been generalizations of Hamming's codes: Bose–Chaudhury–Hocquenhem codes, Reed–Müller codes, Reed–Solomon codes, and Goppa codes, to name a few.

Convolutional codes

Another family of linear codes are *convolutional codes*, which do not divide the source stream into blocks, but instead read and transmit bits continuously. The transmitted bits are a linear function of the past source bits. Usually the rule for generating the transmitted bits involves feeding the present source bit into a linear-feedback shift-register of length k, and transmitting one or more linear functions of the state of the shift register at each iteration. The resulting transmitted bit stream is the convolution of the source stream with a linear filter. The impulse-response function of this filter may have finite or infinite duration, depending on the choice of feedback shift-register.

We will discuss convolutional codes in Chapter 48.

Are linear codes 'good'?

One might ask, is the reason that the Shannon limit is not achieved in practice because linear codes are inherently not as good as random codes? The answer is no, the noisy-channel coding theorem can still be proved for linear codes, at least for some channels (see Chapter 14), though the proofs, like Shannon's proof for random codes, are non-constructive.

Linear codes are easy to implement at the encoding end. Is decoding a linear code also easy? Not necessarily. The general decoding problem (find the maximum likelihood $\mathbf{s}$ in the equation $\mathbf{G}^{\mathsf{T}}\mathbf{s}+\mathbf{n} = \mathbf{r}$) is in fact NP-complete (Berlekamp *et al.*, 1978). [NP-complete problems are computational problems that are all equally difficult and which are widely believed to require exponential computer time to solve in general.] So attention focuses on families of codes for which there is a fast decoding algorithm.

Concatenation

One trick for building codes with practical decoders is the idea of concatenation.

An encoder–channel–decoder system $\mathcal{C} \rightarrow Q \rightarrow \mathcal{D}$ can be viewed as defining a super-channel Q' with a smaller probability of error, and with complex correlations among its errors. We can create an encoder $\mathcal{C}'$ and decoder $\mathcal{D}'$ for this super-channel Q'. The code consisting of the outer code $\mathcal{C}'$ followed by the inner code $\mathcal{C}$ is known as a *concatenated code*.

$$\mathcal{C}' \rightarrow \underbrace{\mathcal{C} \rightarrow Q \rightarrow \mathcal{D}}_{Q'} \rightarrow \mathcal{D}'$$

Some concatenated codes make use of the idea of *interleaving*. We read the data in blocks, the size of each block being larger than the blocklengths of the constituent codes $\mathcal{C}$ and $\mathcal{C}'$. After encoding the data of one block using code $\mathcal{C}'$, the bits are reordered within the block in such a way that nearby bits are separated from each other once the block is fed to the second code $\mathcal{C}$. A simple example of an interleaver is a *rectangular code* or *product code* in which the data are arranged in a $K_2 \times K_1$ block, and encoded horizontally using an (N_1, K_1) linear code, then vertically using a (N_2, K_2) linear code.

▷ Exercise 11.3.[3] Show that either of the two codes can be viewed as the inner code or the outer code.

As an example, figure 11.6 shows a product code in which we encode first with the repetition code R_3 (also known as the Hamming code $H(3, 1)$)

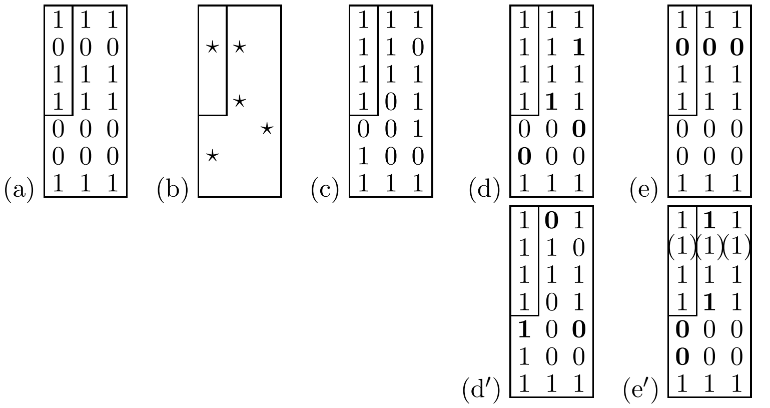

Figure 11.6. A product code. (a) A string `1011` encoded using a concatenated code consisting of two Hamming codes, $H(3,1)$ and $H(7,4)$. (b) a noise pattern that flips 5 bits. (c) The received vector. (d) After decoding using the horizontal $(3,1)$ decoder, and (e) after subsequently using the vertical $(7,4)$ decoder. The decoded vector matches the original.
(d′, e′) After decoding in the other order, three errors still remain.

horizontally then with $H(7,4)$ vertically. The blocklength of the concatenated code is 27. The number of source bits per codeword is four, shown by the small rectangle.

We can decode conveniently (though not optimally) by using the individual decoders for each of the subcodes in some sequence. It makes most sense to first decode the code which has the lowest rate and hence the greatest error-correcting ability.

Figure 11.6(c–e) shows what happens if we receive the codeword of figure 11.6a with some errors (five bits flipped, as shown) and apply the decoder for $H(3,1)$ first, and then the decoder for $H(7,4)$. The first decoder corrects three of the errors, but erroneously modifies the third bit in the second row where there are two bit errors. The $(7,4)$ decoder can then correct all three of these errors.

Figure 11.6(d′−e′) shows what happens if we decode the two codes in the other order. In columns one and two there are two errors, so the $(7,4)$ decoder introduces two extra errors. It corrects the one error in column 3. The $(3,1)$ decoder then cleans up four of the errors, but erroneously infers the second bit.

Interleaving

The motivation for interleaving is that by spreading out bits that are nearby in one code, we make it possible to ignore the complex correlations among the errors that are produced by the inner code. Maybe the inner code will mess up an entire codeword; but that codeword is spread out one bit at a time over several codewords of the outer code. So we can treat the errors introduced by the inner code as if they are independent.

Other channel models

In addition to the binary symmetric channel and the Gaussian channel, coding theorists keep more complex channels in mind also.

Burst-error channels are important models in practice. Reed–Solomon codes use Galois fields (see Appendix C.1) with large numbers of elements (e.g. 2^{16}) as their input alphabets, and thereby automatically achieve a degree of burst-error tolerance in that even if 17 successive bits are corrupted, only 2 successive symbols in the Galois field representation are corrupted. Concatenation and interleaving can give further protection against burst errors. The concatenated Reed–Solomon codes used on digital compact discs are able to correct bursts of errors of length 4000 bits.

▷ Exercise 11.4.[2, p.189] The technique of interleaving, which allows bursts of errors to be treated as independent, is widely used, but is theoretically a poor way to protect data against burst errors, in terms of the amount of redundancy required. Explain why interleaving is a poor method, using the following burst-error channel as an example. Time is divided into chunks of length $N = 100$ clock cycles; during each chunk, there is a burst with probability $b = 0.2$; during a burst, the channel is a binary symmetric channel with $f = 0.5$. If there is no burst, the channel is an error-free binary channel. Compute the capacity of this channel and compare it with the maximum communication rate that could conceivably be achieved if one used interleaving and treated the errors as independent.

Fading channels are real channels like Gaussian channels except that the received power is assumed to vary with time. A moving mobile phone is an important example. The incoming radio signal is reflected off nearby objects so that there are interference patterns and the intensity of the signal received by the phone varies with its location. The received power can easily vary by 10 decibels (a factor of ten) as the phone's antenna moves through a distance similar to the wavelength of the radio signal (a few centimetres).

▶ 11.5 The state of the art

What are the best known codes for communicating over Gaussian channels? All the practical codes are linear codes, and are either based on convolutional codes or block codes.

Convolutional codes, and codes based on them

Textbook convolutional codes. The 'de facto standard' error-correcting code for satellite communications is a convolutional code with constraint length 7. Convolutional codes are discussed in Chapter 48.

Concatenated convolutional codes. The above convolutional code can be used as the inner code of a concatenated code whose outer code is a Reed–Solomon code with eight-bit symbols. This code was used in deep space communication systems such as the Voyager spacecraft. For further reading about Reed–Solomon codes, see Lin and Costello (1983).

The code for Galileo. A code using the same format but using a longer constraint length – 15 – for its convolutional code and a larger Reed–Solomon code was developed by the Jet Propulsion Laboratory (Swanson, 1988). The details of this code are unpublished outside JPL, and the decoding is only possible using a room full of special-purpose hardware. In 1992, this was the best code known of rate 1/4.

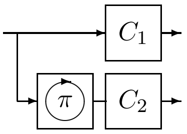

Figure 11.7. The encoder of a turbo code. Each box C_1, C_2, contains a convolutional code. The source bits are reordered using a permutation π before they are fed to C_2. The transmitted codeword is obtained by concatenating or interleaving the outputs of the two convolutional codes. The random permutation is chosen when the code is designed, and fixed thereafter.

Turbo codes. In 1993, Berrou, Glavieux and Thitimajshima reported work on *turbo codes*. The encoder of a turbo code is based on the encoders of two convolutional codes. The source bits are fed into each encoder, the order of the source bits being permuted in a random way, and the resulting parity bits from each constituent code are transmitted.

The decoding algorithm involves iteratively decoding each constituent code using its standard decoding algorithm, then using the output of the decoder as the input to the other decoder. This decoding algorithm is an instance of a *message-passing* algorithm called the *sum–product algorithm*.

Turbo codes are discussed in Chapter 48, and message passing in Chapters 16, 17, 25, and 26.

Block codes

Gallager's low-density parity-check codes. The best block codes known for Gaussian channels were invented by Gallager in 1962 but were promptly forgotten by most of the coding theory community. They were rediscovered in 1995 and shown to have outstanding theoretical and practical properties. Like turbo codes, they are decoded by message-passing algorithms.

We will discuss these beautifully simple codes in Chapter 47.

The performances of the above codes are compared for Gaussian channels in figure 47.17, p.568.

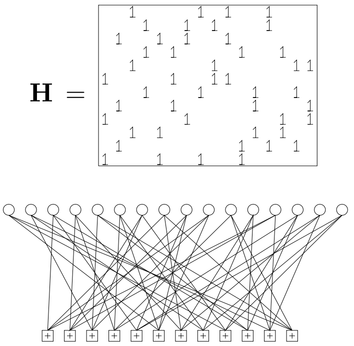

Figure 11.8. A low-density parity-check matrix and the corresponding graph of a rate-1/4 low-density parity-check code with blocklength $N = 16$, and $M = 12$ constraints. Each white circle represents a transmitted bit. Each bit participates in $j = 3$ constraints, represented by ⊞ squares. Each constraint forces the sum of the $k = 4$ bits to which it is connected to be even. This code is a $(16, 4)$ code. Outstanding performance is obtained when the blocklength is increased to $N \simeq 10\,000$.

▶ 11.6 Summary

Random codes are good, but they require exponential resources to encode and decode them.

Non-random codes tend for the most part not to be as good as random codes. For a non-random code, encoding may be easy, but even for simply-defined linear codes, the decoding problem remains very difficult.

The best practical codes (a) employ very large block sizes; (b) are based on semi-random code constructions; and (c) make use of probability-based decoding algorithms.

▶ 11.7 Nonlinear codes

Most practically used codes are linear, but not all. Digital soundtracks are encoded onto cinema film as a binary pattern. The likely errors affecting the film involve dirt and scratches, which produce large numbers of 1s and 0s respectively. We want none of the codewords to look like all-1s or all-0s, so that it will be easy to detect errors caused by dirt and scratches. One of the codes used in digital cinema sound systems is a nonlinear $(8, 6)$ code consisting of 64 of the $\binom{8}{4}$ binary patterns of weight 4.

▶ 11.8 Errors other than noise

Another source of uncertainty for the receiver is uncertainty about the *timing* of the transmitted signal $x(t)$. In ordinary coding theory and information theory, the transmitter's time t and the receiver's time u are assumed to be perfectly synchronized. But if the receiver receives a signal $y(u)$, where the receiver's time, u, is an imperfectly known function $u(t)$ of the transmitter's time t, then the capacity of this channel for communication is reduced. The theory of such channels is incomplete, compared with the synchronized channels we have discussed thus far. Not even the *capacity* of channels with synchronization errors is known (Levenshtein, 1966; Ferreira *et al.*, 1997); codes for reliable communication over channels with synchronization errors remain an active research area (Davey and MacKay, 2001).

Further reading

For a review of the history of spread-spectrum methods, see Scholtz (1982).

▶ 11.9 Exercises

The Gaussian channel

▷ Exercise 11.5.[*2*, p.190] Consider a Gaussian channel with a real input x, and signal to noise ratio v/σ^2.

(a) What is its capacity C?

(b) If the input is constrained to be binary, $x \in \{\pm\sqrt{v}\}$, what is the capacity C' of this constrained channel?

(c) If in addition the output of the channel is thresholded using the mapping

$$y \rightarrow y' = \begin{cases} 1 & y > 0 \\ 0 & y \leq 0, \end{cases} \tag{11.35}$$

what is the capacity C'' of the resulting channel?

(d) Plot the three capacities above as a function of v/σ^2 from 0.1 to 2. [You'll need to do a numerical integral to evaluate C'.]

▷ Exercise 11.6.[*3*] For large integers K and N, what fraction of all binary error-correcting codes of length N and rate $R = K/N$ are *linear* codes? [The answer will depend on whether you choose to define the code to be an *ordered* list of 2^K codewords, that is, a mapping from $s \in \{1, 2, \ldots, 2^K\}$ to $\mathbf{x}^{(s)}$, or to define the code to be an unordered list, so that two codes consisting of the same codewords are identical. Use the latter definition: a code is a set of codewords; how the encoder operates is not part of the definition of the code.]

Erasure channels

▷ Exercise 11.7.[*4*] Design a code for the binary erasure channel, and a decoding algorithm, and evaluate their probability of error. [The design of good codes for erasure channels is an active research area (Spielman, 1996; Byers *et al.*, 1998); see also Chapter 50.]

▷ Exercise 11.8.[*5*] Design a code for the q-ary erasure channel, whose input x is drawn from $0, 1, 2, 3, \ldots, (q-1)$, and whose output y is equal to x with probability $(1-f)$ and equal to `?` otherwise. [This erasure channel is a good model for packets transmitted over the internet, which are either received reliably or are lost.]

Exercise 11.9.[*3*, p.190] How do redundant arrays of independent disks (RAID) work? These are information storage systems consisting of about ten disk drives, of which any two or three can be disabled and the others are able to still able to reconstruct any requested file. What codes are used, and how far are these systems from the Shannon limit for the problem they are solving? How would *you* design a better RAID system? Some information is provided in the solution section. See `http://www.acnc.com/raid2.html`; see also Chapter 50.

[Some people say RAID stands for 'redundant array of inexpensive disks', but I think that's silly – RAID would still be a good idea even if the disks were expensive!]

▶ 11.10 Solutions

Solution to exercise 11.1 (p.181). Introduce a Lagrange multiplier λ for the power constraint and another, μ, for the constraint of normalization of $P(x)$.

$$\begin{aligned} F &= I(X;Y) - \lambda \textstyle\int \mathrm{d}x\, P(x)x^2 - \mu \textstyle\int \mathrm{d}x\, P(x) && (11.36) \\ &= \int \mathrm{d}x\, P(x) \left[\int \mathrm{d}y\, P(y\,|\,x) \ln \frac{P(y\,|\,x)}{P(y)} - \lambda x^2 - \mu \right]. && (11.37) \end{aligned}$$

Make the functional derivative with respect to $P(x^*)$.

$$\begin{aligned} \frac{\delta F}{\delta P(x^*)} &= \int \mathrm{d}y\, P(y\,|\,x^*) \ln \frac{P(y\,|\,x^*)}{P(y)} - \lambda x^{*2} - \mu \\ &\quad - \int \mathrm{d}x\, P(x) \int \mathrm{d}y\, P(y\,|\,x) \frac{1}{P(y)} \frac{\delta P(y)}{\delta P(x^*)}. && (11.38) \end{aligned}$$

The final factor $\delta P(y)/\delta P(x^*)$ is found, using $P(y) = \int \mathrm{d}x\, P(x)P(y\,|\,x)$, to be $P(y\,|\,x^*)$, and the whole of the last term collapses in a puff of smoke to 1, which can be absorbed into the μ term.

Substitute $P(y\,|\,x) = \exp(-(y-x)^2/2\sigma^2)/\sqrt{2\pi\sigma^2}$ and set the derivative to zero:

$$\int \mathrm{d}y\, P(y\,|\,x) \ln \frac{P(y\,|\,x)}{P(y)} - \lambda x^2 - \mu' = 0 \qquad (11.39)$$

$$\Rightarrow \int \mathrm{d}y\, \frac{\exp(-(y-x)^2/2\sigma^2)}{\sqrt{2\pi\sigma^2}} \ln\left[P(y)\sigma\right] = -\lambda x^2 - \mu' - \frac{1}{2}. \qquad (11.40)$$

This condition must be satisfied by $\ln[P(y)\sigma]$ for all x.

Writing a Taylor expansion of $\ln[P(y)\sigma] = a+by+cy^2+\cdots$, only a quadratic function $\ln[P(y)\sigma] = a + cy^2$ would satisfy the constraint (11.40). (Any higher order terms y^p, $p > 2$, would produce terms in x^p that are not present on the right-hand side.) Therefore $P(y)$ is Gaussian. We can obtain this optimal output distribution by using a Gaussian input distribution $P(x)$.

Solution to exercise 11.2 (p.181). Given a Gaussian input distribution of variance v, the output distribution is $\mathrm{Normal}(0, v+\sigma^2)$, since x and the noise are independent random variables, and variances add for independent random variables. The mutual information is:

$$\begin{aligned} I(X;Y) &= \int \mathrm{d}x\, \mathrm{d}y\, P(x)P(y\,|\,x) \log P(y\,|\,x) - \int \mathrm{d}y\, P(y) \log P(y) && (11.41) \\ &= \frac{1}{2} \log \frac{1}{\sigma^2} - \frac{1}{2} \log \frac{1}{v+\sigma^2} && (11.42) \\ &= \frac{1}{2} \log \left(1 + \frac{v}{\sigma^2}\right). && (11.43) \end{aligned}$$

Solution to exercise 11.4 (p.186). The capacity of the channel is one minus the information content of the noise that it adds. That information content is, per chunk, the entropy of the selection of whether the chunk is bursty, $H_2(b)$, plus, with probability b, the entropy of the flipped bits, N, which adds up to $H_2(b) + Nb$ per chunk (roughly; accurate if N is large). So, per bit, the capacity is, for $N = 100$,

$$C = 1 - \left(\frac{1}{N} H_2(b) + b\right) = 1 - 0.207 = 0.793. \qquad (11.44)$$

In contrast, interleaving, which treats bursts of errors as independent, causes the channel to be treated as a binary symmetric channel with $f = 0.2 \times 0.5 = 0.1$, whose capacity is about 0.53.

Interleaving throws away the useful information about the correlatedness of the errors. Theoretically, we should be able to communicate about $(0.79/0.53) \simeq 1.6$ times faster using a code and decoder that explicitly treat bursts as bursts.

Solution to exercise 11.5 (p.188).

(a) Putting together the results of exercises 11.1 and 11.2, we deduce that a Gaussian channel with real input x, and signal to noise ratio v/σ^2 has capacity

$$C = \frac{1}{2} \log \left(1 + \frac{v}{\sigma^2} \right). \tag{11.45}$$

(b) If the input is constrained to be binary, $x \in \{\pm\sqrt{v}\}$, the capacity is achieved by using these two inputs with equal probability. The capacity is reduced to a somewhat messy integral,

$$C'' = \int_{-\infty}^{\infty} \mathrm{d}y \, N(y;0) \log N(y;0) - \int_{-\infty}^{\infty} \mathrm{d}y \, P(y) \log P(y), \tag{11.46}$$

where $N(y;x) \equiv (1/\sqrt{2\pi}) \exp[(y-x)^2/2]$, $x \equiv \sqrt{v}/\sigma$, and $P(y) \equiv [N(y;x) + N(y;-x)]/2$. This capacity is smaller than the unconstrained capacity (11.45), but for small signal-to-noise ratio, the two capacities are close in value.

(c) If the output is thresholded, then the Gaussian channel is turned into a binary symmetric channel whose transition probability is given by the error function Φ defined on page 156. The capacity is

$$C'' = 1 - H_2(f), \text{ where } f = \Phi(\sqrt{v}/\sigma). \tag{11.47}$$

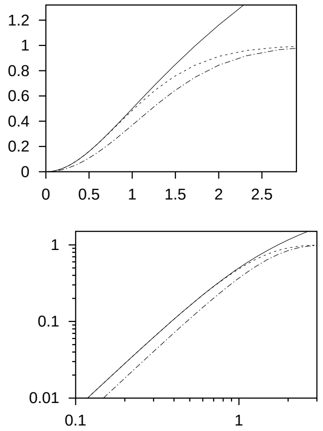

Figure 11.9. Capacities (from top to bottom in each graph) C, C', and C''', versus the signal-to-noise ratio $(\sqrt{v}/\sigma)$. The lower graph is a log–log plot.

Solution to exercise 11.9 (p.188). There are several RAID systems. One of the easiest to understand consists of 7 disk drives which store data at rate $4/7$ using a $(7,4)$ Hamming code: each successive four bits are encoded with the code and the seven codeword bits are written one to each disk. Two or perhaps three disk drives can go down and the others can recover the data. The effective channel model here is a binary erasure channel, because it is assumed that we can tell when a disk is dead.

It is not possible to recover the data for *some* choices of the three dead disk drives; can you see why?

▷ Exercise 11.10.[*2*, p.190] Give an example of three disk drives that, if lost, lead to failure of the above RAID system, and three that can be lost without failure.

Solution to exercise 11.10 (p.190). The $(7,4)$ Hamming code has codewords of weight 3. If any set of three disk drives corresponding to one of those codewords is lost, then the other four disks can recover only 3 bits of information about the four source bits; a fourth bit is lost. [cf. exercise 13.13 (p.220) with $q = 2$: there are no binary MDS codes. This deficit is discussed further in section 13.11.]

Any other set of three disk drives can be lost without problems because the corresponding four by four submatrix of the generator matrix is invertible. A better code would be a digital fountain – see Chapter 50.