9

Speech and handwriting

Conceptually, the techniques of linguistic pattern recognition are largely independent of the medium, but overall performance is influenced by the preprocessing to such an extent that until a few years ago the pattern recognition step was generally viewed as a small appendix to the main body of signal processing knowledge. To this day, it remains impossible to build a serious system without paying close attention to preprocessing, and deep algorithmic work on the recognizer will often yield smaller gains than seemingly more superficial changes to the front end. In Section 9.1, we introduce a speech coding method, linear prediction, that has played an important role in practical application since the 1970s. We extend the discussion of quantization started in Section 8.1 from scalars to vectors and discuss the Fourier transform-based (homomorphic) techniques that currently dominate the field.

These techniques, in spite of their analytic sophistication, are still low-level inasmuch as the signal can still be reconstructed, often without perceptually noticeable loss from the encoding, yet they suffice to decrease the bitrate by several orders of magnitude. As we shall see, the bitrate provides a surprisingly good measure of our understanding of the nature of speech: the more we know, the better we can compress the signal. This observation extends well beyond low-level signal processing in that incorporating deeper knowledge about the speech signal leads to further gains in compression. In Section 9.2, we discuss how a central component of the linguistic theory of speech, the phonemic principle introduced in Section 3.1, can be leveraged to yield further compression gains in HMMs.

The recognition of handwritten or printed text by computer is referred to as *optical character recognition* (OCR). When the input device is a digitizer tablet that transmits the signal in real time (as in pen-based computers and personal digital assistants) or includes timing information together with pen position (as in signature capture), we speak of *dynamic* recognition. When the input device is a still camera or a scanner, which captures the position of digital ink on the page but not the order in which the ink was laid down, we speak of *static* or *image-based* OCR. The difficulties encountered in dynamic OCR are largely similar to those found in speech recognition: the stream of position/pen pressure values output by the digitizer tablet is analogous to the stream of speech signal vectors output by the audio processing front end, and the same kinds of low-level signal processing and pattern recognition techniques are widely employed for both. In Section 9.3, we will deal primarily with static OCR, emphasizing those aspects of the problem that have no counterpart in the recognition of spoken or signed language.

9.1 Low-level speech processing

A two-way infinite sequence $s = \ldots s_{-2}, s_{-1}, s_0, s_1, s_2, \ldots$ will be called a (discrete) **signal**, and its generating function $\sum_{n=-\infty}^{\infty} s_n z^{-n}$ will be called its **z transform** $\mathbf{Z}(s)$. Signals will also be denoted by $\{s_n\}$. If the signal is bounded (a condition we can always enforce by clipping it; see Section 8.1) and satisfies further conditions that are generally met, the z transform will be absolutely convergent on a disk of positive radius and the signal can be uniquely reconstructed from it. A **filter** (sometimes called a *system*) is a mapping from signals to signals: we are particularly interested in the case where the mapping is both linear and time-invariant (invariant under the shift operator S). The signal u defined by $u_0 = 1, u_n = 0$ $(n \neq 0)$ is called the unit **impulse**, and the image $\{h_n\}$ of this signal under some filter F is called the **impulse response** $h = F(u)$ of the filter F. As long as h_n is absolute convergent, h completely characterizes any linear and time-invariant filter. To see this, consider any arbitrary input x and write it as $\sum_{m=-\infty}^{\infty} x_m u_0^m$, where u_0^m is u_0 shifted by m. Since F is linear and time-invariant, we obtain

$$F(x)_n = \sum_{m=-\infty}^{\infty} x_m h_{n-m} = \sum_{m=-\infty}^{\infty} x_{n-m} h_m \tag{9.1}$$

This sum, which will always converge for x bounded, determines the output uniquely just from the values of h. A linear and time-invariant filter is called **causal** if $h_n = 0$ for $n < 0$ and **stable** if bounded input always produces bounded output – it is trivial to see that absolute convergence of h_n is both necessary and sufficient for stability. In what follows, we will be chiefly concerned with filters that are linear, time-invariant, causal, and stable and will omit these qualifications.

In speech processing applications, we are particularly concerned with the output of a filter F with impulse response h when the input is a sampled complex sine wave $x = \{e^{i\omega n}\}$. Using (9.1), we obtain

$$y_n = F(x)_n = \sum_{m=-\infty}^{\infty} e^{i\omega(n-m)} h_m = e^{i\omega n} \sum_{m=-\infty}^{\infty} h_m e^{-i\omega m} \tag{9.2}$$

The term outside the sum is x_n. The function $\sum_{m=-\infty}^{\infty} h_m e^{-i\omega m}$ will be denoted $H(e^{i\omega})$ and called the **frequency response** or **transfer function** of F. With this notation, (9.2) becomes the more perspicuous

$$y = H(e^{i\omega})x \tag{9.3}$$

where equality among signals means termwise equality. In general, we define the **frequency spectrum** $\mathbf{F}(x)$ of a signal x as

$$X(e^{i\omega}) = \sum_{m=-\infty}^{\infty} x_m e^{-i\omega m} \tag{9.4}$$

so that the frequency response is just the frequency spectrum of the impulse response. Since (9.4) is a Fourier series with coefficients x, we can make good use of the orthogonality of $e^{i\omega n}$ and $e^{i\omega m}$ on the $-\pi \leq \omega \leq \pi$ interval and find x from its frequency spectrum by the well-known Fourier inversion formula

$$x_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{i\omega}) e^{i\omega n} d\omega \tag{9.5}$$

Now if $y = F(x)$, and we denote the frequency spectrum of x, y, and h (the impulse response of F) by X, Y, and H, respectively, we have, by applying (9.5) to y,

$$y_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} Y(e^{i\omega}) e^{i\omega n} d\omega \tag{9.6}$$

Since (9.5) holds for all n, we can apply F to the whole series, obtaining

$$y_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{i\omega}) F(e^{i\omega n}) d\omega \tag{9.7}$$

By applying (9.3) to the series $e^{i\omega n}$, the last term in this integral, $F(e^{i\omega n})$, can be expressed as $H(e^{i\omega})e^{i\omega n}$. Comparing this to (9.6) yields

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{i\omega}) H(e^{i\omega}) e^{i\omega n} d\omega = y_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} Y(e^{i\omega}) e^{i\omega n} d\omega \tag{9.8}$$

which, by the uniqueness of Fourier coefficients, leads to

$$X(e^{i\omega}) H(e^{i\omega}) = Y(e^{i\omega}) \tag{9.9}$$

Thus, the frequency spectrum of the output is obtained by multiplying the frequency spectrum of the input with the frequency response of the filter.

The frequency spectrum and the z transform are both obtained from a signal by using the terms as coefficients in a series of complex functions: the frequency spectrum is the z transform with $z = e^{i\omega}$ (i.e. investigated over the unit circle). Taking the z transform of both sides of (9.1), it is trivially seen that (9.9) is valid for X, Y, H z transforms (rather than frequency spectra) of x, y, h. In general, we make little distinction between $\mathbf{Z}$ and $\mathbf{F}$ and for any signal $s, t, \ldots, x, y, z$ will follow the engineering convention of using uppercase $S, T, \ldots, X, Y, Z$ to denote its z transform and frequency spectrum alike. The use of lowercase letters indicates, in engineering parlance, signals and operations in the *time domain*, where the independent variable is time, and the use of uppercase letters refers to the *frequency domain*, where the independent variable is frequency. The two domains are connected by the transforms and their inverses.

While we are chiefly interested in discrete (digital) signals, it is clear that the speech waveform is inherently continuous (analog), and one could replace (9.4) by a continuous version (distinguished by a subscript C):

$$X_C(i\Omega) = \int_{-\infty}^{\infty} x(t)e^{-i\Omega t}dt \tag{9.10}$$

If we measure the continuous signal $x(t)$ at time intervals T apart (this is called using a **sampling frequency** $\omega_s = 2\pi/T$), by the inverse Fourier transform (which of course remains applicable in the continuous case), we obtain

$$x(nT) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_C(i\Omega)e^{i\Omega nT}d\Omega \tag{9.11}$$

We shall now subdivide the integration domain to intervals of length $2\pi/T$ starting at $(2m-1)\pi/T$ and obtain the Fourier expansion

$$x(nT) = \frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} \frac{1}{T}\sum_{m=-\infty}^{\infty} X_C(i(\omega + 2\pi m/T))e^{i\omega nT}d\omega \tag{9.12}$$

where we substituted $\omega + 2\pi m/T$ for Ω in the mth interval. By (9.5) the discrete signal $\{x(nT)\}$ satisfies

$$x(nT) = \frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} X(e^{i\omega T})e^{i\omega nT}d\omega \tag{9.13}$$

Comparing (9.12) with (9.13) yields

$$\frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} \frac{1}{T}\sum_{m=-\infty}^{\infty} X_C(i(\omega+2\pi m/T))e^{i\omega nT}d\omega = \frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} X(e^{i\omega T})e^{i\omega nT}d\omega \tag{9.14}$$

and by the uniqueness of Fourier coefficients we have

$$\frac{1}{T}\sum_{m=-\infty}^{\infty} X_C(i(\omega + 2\pi m/T)) = X(e^{i\omega T}) \tag{9.15}$$

If $x(t)$ is **bandlimited** in the sense that $X_C(i\omega) = 0$ for any $|\omega| \geq \omega_c$ and we restrict ourselves to sampling frequencies $\omega_s = 2\pi/T > 2\omega_c$, within the interval $|\omega_c| < \omega_s/2$ all but the central term of (9.15) will be zero and there we have

$$\frac{1}{T}X_C(i\omega) = X(e^{i\omega T}) \tag{9.16}$$

In other words, the continuous frequency spectrum (Fourier transform) X_C is fully determined by the discrete frequency spectrum X, even though though X will have *sidebands* ($k\omega_s$ translates of the central band) while X_C will be, as we assumed, identically zero outside the critical band. Since the Fourier transform uniquely determines the function, and the discrete spectrum was computed entirely on the basis of the sampled values, these values uniquely determine the original continuous function, and we have the following theorem.

Theorem 9.1.1 Sampling theorem. If a continuous signal $x(t)$ is bandlimited with cutoff frequency ω_c, sampling it at any frequency $\omega_s = 2\pi/T > 2\omega_c$ or higher provides a discrete signal $\{x(nT)\}$ from which $x(t)$ can be fully reconstructed.

Discussion As is well-known, the upper frequency threshold of human hearing is about 20 kHz, and human speech actually contains only a negligible amount of energy above 12 kHz. The 44.1 kHz sampling rate of audio CDs was set with the sampling theorem in mind since analog HiFi equipment was designed to operate in the 20 Hz – 20 kHz range and the intention was to preserve all high fidelity audio in digital format. Telephone speech, sampled at 8 kHz, is perfectly understandable, in spite of the fact that much of the frequency range below 350 Hz is also severely attenuated. In speech research, a sampling rate of 20 kHz, and in speech communications a 10 kHz sampling rate, is common. In what follows, we concentrate on the digital case, since Theorem 9.1.1 guarantees that nothing of importance is lost that way.

In Section 8.1, we discussed log PCM speech coding, which, by means of fitting the quantization steps to the long-term amplitude distribution of speech, achieves toll quality transmission at 64 kbps. To go beyond this limit, we need to exploit short-term regularities. Perhaps the simplest idea is to use **delta coding**, transmitting the difference between two adjacent samples rather than the samples themselves. Since the differences are generally smaller than the values, it turns out we can save about 1 bit per sample and still provide speech quality equivalent to the original (see Jayant and Noll 1984). *Linear predictive coding* (LPC) extends the basic idea of delta coding by considering not only the previous sample but the previous p samples to be available for predicting the next sample. Thus we shall consider the general problem of obtaining the signal s from some unknown input signal u such that

$$s_n = -\sum_{k=1}^{p} a_k s_{n-k} + G \sum_{l=0}^{q} b_l u_{n-l} \tag{9.17}$$

i.e. as a linear combination of the past $q + 1$ inputs and the past p outputs. (We assume $b_0 = 1$ and separate out a *gain factor* G for putting the equations in a more intuitive form.) With these conventions, the transfer function can be written in terms of z transforms as

$$H(z) = \frac{S(z)}{U(z)} = G \frac{1 + \sum_{l=1}^{q} b_l z^{-l}}{1 + \sum_{k=1}^{p} a_k z^{-k}} \tag{9.18}$$

i.e. as a Padé approximation. Since the zeros of the denominator appear as poles in the whole expression, in signal processing (9.18) is known as the *pole-zero* model, with the key case where $b_l = 0$ for $l \geq 1$ called the *all-pole* model – the number of terms p is called the *order* of the model.

Engineers call a signal *stationary* if its statistical characteristics, as summarized in the frequency spectrum, stay constant over long periods of time. The vibrations produced by rotating machinery or the spattering of rain are good examples. We say a signal $X(t)$ is **stationary in the wide sense** if its expectation $E(X(t))$ is constant and the correlation function $E(X(t)X(s))$ depends only on the difference $t - s$. This is less strict than Definition 8.2.1 because we do not demand full invariance under a single sample shift (and thus by transitivity over the entire timeline) but use a weaker notion of 'approximate' invariance instead. The situation is further complicated by the fact that speech is rarely stationary, even in this wide sense, over more than a few pitch periods – this is referred to as the signal being *quasistationary*. At the high end (infants, sopranos), the glottal pulses can follow each other as fast as every 2 ms, for adult male speech 6–12 ms is typical, while at the low end (basso profondo) 22 ms or even longer pitch periods are found.

To take the quasistationary nature of the signal into account, speech processing generally relies on the use of *windowing* techniques, computing spectra on the basis of samples within a 20 ms stretch. Typically such a window will contain more than a full pitch period and thus allow for very good reconstruction of the signal. Because of edge effects (produced when the pitch period is close to the window size), rectangular windows are rarely used. Rather, the signal within the window is multiplied with a windowing function such as the *Hamming window*, $w_n = 0.54 - 0.46\cos(2\pi n/(N-1))$. In speech processing, given a 20 kHz sampling rate and 20 ms windows, N is about 400, depending on how the edges are treated. Windows are generally overlapped 50% so that each sample appears in two successive windows. In other words, analysis proceeds at a *frame rate* of 100 Hz.

Aside from computing windowed functions, pointwise multiplication of two signals is rarely called for. A more important operation is the **convolution** $s \circ t$ of two signals s and t given by $(s \circ t)_k = \sum_{n=0}^{N-1} s_n t_{k-n}$ in the discrete case and by $\int s(\tau)t(x-\tau)d\tau$ in the continuous case. When the signal has only finite support (e.g. because of windowing), we can consider the indexes mod N in performing the summation, a method known as *circular convolution*, or we can take values of s and t outside the range $0, \dots, N$ to be zeros, a method known as *linear convolution*. Unless explicitly stated otherwise, in what follows we assume circular convolution for finite signals. For the infinite case, (9.1) asserts that the output of a filter F is simply the convolution of the input and the impulse response of F, and (9.9) asserts that convolution in the time domain amounts to multiplication in the frequency domain.

The practical design of filters, being concerned mostly with the frequency response, generally proceeds in the frequency domain. Of particular interest are *highpass* filters, whose response is a step function (0 below the frequency cutoff and 1 above), *lowpass* filters (1 below the cutoff and 0 above), and *bandpass* filters (0 below the bottom and above the top of the band, 1 inside the band). Since these ideal pass characteristics can be approximated by both analog and digital circuitry at reasonable cost, it is common for signal processing algorithms to be implemented using filters as building blocks, and much of the literature on speech production, processing, and perception is actually presented in these terms. One tool used particularly often is a *filter bank* composed of several filters, each sensitive only in a specified *subband* of the region of interest. In *mel* filtering, triangular transfer function filters are used in an overlapping sequence that follows the classical mel subjective pitch scale (Stevens and Volkman 1940, Beranek 1949).

Exercise 9.1 Given capacitors, resistors, and coils of arbitrary precision, design high- low- and bandpass filters that pass pure sinusoidal voltage signals within a prescribed error factor relative to the ideal filter characteristics.

After windowing, we are faced with a very different data reduction problem: instead of a fast (20 kHz) sequence of (16 bit) *scalars*, we need to compress a much slower (0.1 kHz) sequence of *vectors*, where a single vector accounts for the contents of a whole frame. We begin by replacing the signal within a window by some appropriate function of it such as its **discrete Fourier transform** (DFT), given by

$$S_k = \sum_{n=0}^{N-1} s_n e^{-\frac{2\pi i}{N}kn} \tag{9.19}$$

Here S_k is the amplitude of the signal at frequency $2\pi k/N$, and it is common to use notation like $S(\omega)$ in analogy with the continuous case even though, strictly speaking, we are now concerned only with frequencies that are a multiple of $2\pi/N$. Of special note is the discrete version of Parseval's theorem:

$$\sum_{n=0}^{N-1} |s_n|^2 = \frac{1}{N}\sum_{n=0}^{N-1} |S_n|^2 \tag{9.20}$$

As in the continuous case, Parseval's theorem is interpreted as saying that the total energy of the signal (defined there by $\int_{-\infty}^{\infty} s^2(t)dt$) can also be obtained by integrating the energy in the frequency domain. The contribution of a frequency range (also known as a frequency *band* in signal processing) $a \leq \omega \leq b$ is given by $\int_a^b S(\omega)d\omega + \int_{-b}^{-a} S(\omega)d\omega$ – the first term is known as the contribution of the *positive frequency* and the second as that of the *negative frequency*. In analogy with the continuous case, the squared absolute values of the DFT coefficients are called the **energy spectrum** and are denoted $P(\omega)$ (with implicit time normalization, the term *power spectrum* is also often used).

From a given energy spectrum, we can recover the moduli of the DFT coefficients by taking square roots, but we lose their argument (known in this context as the *phase*), so we cannot fully reconstruct the original signal. However, hearers are relatively insensitive to the distinction between waveforms inverted from full spectra and from modulus information alone (at least for sound presented by loudspeakers in a room with normal acoustics – for sounds presented directly through earphones, discarding the phase information is more problematic; see Klatt 1987) and for many, if not all, purposes in speech processing, energy spectra are sufficient. This is not to say that the phase is irrelevant (to the contrary, providing phase continuity is important to the perceived naturalness of synthesized speech) or that it contains no information (see Paliwal and Atal (2003) for recognition based on phase information alone), but the situation is somewhat analogous to speech, which contains enough information in the 150 Hz – 1.5 kHz band to be nearly perfectly intelligible but also contains enough information in the 1.5–15 kHz band to be equally intelligible.

The DFT has several important properties that make it especially well-suited for the analysis of speech. First, note that (9.19) is actually a *linear* transform of the inputs (with complex coefficients, to be sure, but still linear). The inverse DFT (IDFT) is therefore also a linear transform and one that is practically identical to the DFT:

$$s_k = \frac{1}{N} \sum_{n=0}^{N-1} S_n e^{\frac{2\pi i}{N} kn} \tag{9.21}$$

i.e. only the sign in the exponents and the normalization factor $1/N$ are different. There are, significantly, *fast Fourier transform* (FFT) algorithms that compute the DFT or the IDFT in $O(N \log(N))$ steps instead of the expected N^2 steps.

Since in (linear) acoustics an echo is computed as a convolution of the original signal with a function representing the objects on which the sound is reflected, it is natural to model speech as an acoustic source (either the glottis, or, in the case of unvoiced sounds, frication noise generated at some constriction in the vocal tract) getting filtered by the shape of the vocal tract downstream from the source. This is known as the *source-filter model of speech production* (Fant 1960). As convolution corresponds to multiplication of generating functions or DFTs, the next natural step is to take logarithms and investigate the additive version of the transform, especially as signal processing offers very effective filtering techniques for separating signals whose energy lies in separate frequency ranges.

Experience shows that before taking the logarithm it is advantageous to rescale the energy spectrum by using $\Omega = 6 \log(\omega/1200\pi + \sqrt{1 + (\omega/1200\pi)^2})$, which converts the frequency variable ω given in radians/sec into *bark* units Ω (Schroeder 1977), as the bark scale (Zwicker et al. 1957) models important features of human hearing better than the linear (physical) or logarithmic (musical) frequency scale would. Essentially the same step, nonlinear warping of frequencies, can be accomplished by more complex signal processing in the time domain using filter banks arranged on the mel scale (Davis and Mermelstein 1980). Further perceptually motivated signal processing steps, such as preemphasis of the data to model the different sensitivities of human hearing at different frequencies, or amplitude compression to model the Weber-Fechner law, are used in various schemes, but we do not follow this line of development here as it contributes little to speech recognition beyond making it more noise-robust (Hermansky 1990, Hermansky et al. 1991, Shannon and Paliwal 2003).

If we now take the log of the (mel- or bark-scaled) energy spectral coefficients, what was a multiplicative relationship (9.9) between a *source* signal (a glottal pulse for voiced sounds or fricative noise for unvoiced sounds) and a *filter* given by the shape of the vocal tract becomes an additive relation. The log energy spectrum itself, being a finite sequence of (real) values, can be considered a finite signal, amenable to analysis by DFT or by IDFT (the two differ only in circular order and a constant multiplier). We call the IDFT of the log of the energy spectrum the **cepstrum** of the original signal and call its independent variable the *quefrency* in order to avoid collision with the standard notion of frequency. For example, if we find a cepstral peak at quefrency 166, this means recurrence at every 166 samples, which at a sampling rate of 20 kHz amounts to a real frequency of 120 Hz (Noll 1964,1967).

As we are particularly interested in the case where the original signal or the energy spectrum is mel- or bark-wrapped, we note here that in mel scaling, a great deal of economy can be achieved by keeping only a few (generally 12–16) filter bank outputs, so that the dimension of the cepstral parameter space is kept at 12–16. In the bark case, we achieve the same effect by *downsampling*; i.e. keeping only a few (generally 16–20) points in the energy spectrum. To fully assess the savings entailed by these steps would require an analysis of the quantization losses. Clearly, 32 bit resolution on the cepstral parameters is far more than what we need. We postpone this step but note that 18 32-bit parameters for 100 frames per second would mean 57.6 kbps coding, not a particularly significant improvement over the 64 kbps log PCM scheme discussed earlier.

The savings will come from the source-filter approach: the glottal pulse can be modeled by an impulse function (in many cases, more realistic models are desirable – source modeling is a major research topic on its own) and frication can be modeled by white noise. Obviously, neither the impulse nor the white noise needs to be transmitted. All that is required is some encoding of the filter shape plus one bit per frame to encode whether the frame is voiced or unvoiced – in voiced frames, another 8–10 bits are used to convey the fundamental frequency F0. As we shall see, the relevant information about the transfer function of the filter can be transmitted in 40–80 bits, and frame rates as low as 40–50 Hz are sufficient, yielding toll or communications quality speech coding at 16 kbps or lower. The main advantage of the cepstral representation is that spectral characteristics of the source and the filter are largely separated in the quefrency domain and can be extracted (or suppressed) by bandpass (resp. notch) filtering (called *liftering* in this domain).

Let us now turn to the issue of modeling a signal, be it an actual time domain signal or a series of spectral or cepstral coefficients, in the form (9.17), using the all-pole model. We will for the moment ignore the input signal u_n and look for a least squares error solution to $e_n = s_n + \sum_{k=1}^{p} a_k s_{n-k}$. By setting the partial derivatives of $\sum_n e_n^2$ to zero, we obtain the set of normal equations

$$\sum_{k=1}^{p} a_k \sum_n s_{n-k} s_{n-i} = -\sum_n s_n s_{n-i} \tag{9.22}$$

For N finite, the expressions $\sum_{n=0}^{N-1} s_{n-k} s_{n-i}$ are known as the *covariances* and will be collected in the covariance matrix ϕ_{ik}, which is symmetric. If N is infinite, only the differences $i - k$ matter, and the same expression is called the *autocorrelation* and is denoted by $R(i-k)$. Either way, we have p linear equations in p unknowns, the LPC coefficients, and may take advantage of the special form of the covariance or autocorrelation matrix to solve the problem relatively quickly.

Exercise 9.2 Assuming that the autocorrelation coefficients $R(j)$ have already been computed (e.g. from estimating them in a fixed size window), find an algorithm to solve equations (9.22) in $O(p^2)$ steps.

If we assume that the signal values s_i are samples of random variables X_i, it is the expected value $E(e_n^2)$ of the squared error that we wish to minimize, and again by setting the partial derivatives to zero we obtain

$$\sum_{k=1}^{p} a_k E(s_{n-k} s_{n-i}) = -E(s_n s_{n-i}) \tag{9.23}$$

For a stationary process, $E(s_{n-k}s_{n-i}) = R(k-i)$ holds, and speech is generally considered quasistationary to the extent that this remains a reasonable approximation if R is estimated on a window comprising at most a few pitch periods.

So far, we have ignored the input signal u in (9.17), but it is clear that the equation can hold with error $e_n = 0$ in an all-pole model only if $Gu_n = e_n$. We cannot take advantage of this observation point by point (to introduce a term that corrects for the prediction error would require knowing the error in advance), but we can use it in the average in the following sense: if we want the energy of the actual signal to be the same as the energy of the predicted signal, the total energy of the input signal must equal the total energy of the error signal, $\sum_n e_n^2 = \sum_n (s_n + \sum_{k=1}^{p} a_k s_{n-k}^2)$. Using (9.22), we can see this to be

$$E_p = \sum_n s_n^2 + \sum_{k=1}^{p} a_k \sum_n s_n s_{n-k} \tag{9.24}$$

or, using the autocorrelations $R(i)$, just $E_p = R(0) + \sum_{k=1}^{p} a_k R(k)$. If the input signal u is the unit impulse, its energy will be G^2, which must be set equal to the energy E_p of the error signal that we just computed. Since the $R(i)$ must be computed anyway if the autocorrelation method is used to determine the a_i, the gain G now can also be computed from

$$G^2 = R(0) + \sum_{k=1}^{p} a_k R(k) \tag{9.25}$$

to provide a complete characterization of the transfer function $H(z)$ in (9.18) as long as no zeros, just poles (zeros in the denominator), are used.

Exercise 9.3 If the samples u_n are uncorrelated (white noise), show that the autocorrelations $\hat{R}(i)$ of the output signal $\hat{s}_n = -\sum_{k=1}^{p} a_k \hat{s}_{n-k} + Gu_n$ are the same as the autocorrelations $R(i)$ of the original signal as long as G is set so as to preserve the total energy $\hat{R}(0) = R(0)$. Does (9.25) remain true in this case?

The considerations above provide justification only for the LPC coding of voiced (glottal pulse source) and unvoiced (white noise source) signals, but it turns out that all-pole models are applied with noticeable success to signals such as cepstra that have characteristics very different from those of the raw speech signal. So far, we have decomposed the original signal into voiced and unvoiced frames and devoted 8–10 bits per frame to encoding the pitch (F0) of voiced frames. What remains is transmitting the LPC coefficients a_i and the gain G. Since these are sensitive to quantization noise, the predictors a_i are often replaced by *reflection coefficients* k_i, which can be computed essentially by the same recursive procedure that is used in solving (9.22). Taking $p = 10$ and using 10 bits per coefficient, 50 frames/sec, we obtain cell grade speech coding at about one-tenth the bitrate of log PCM.

To make further progress, it makes sense to model the properties of the LPC (reflection or predictor) coefficients jointly. A **vector quantizer** (VQ) is an algorithm that maps a sequence $\mathbf{x}_i$ of (real or complex) vectors on binary sequences $\gamma(\mathbf{x}_i)$ suitable for data transmission. It is generally assumed that the range of γ is finite: elements of the range are called **channel symbols**. For each channel symbol c, the decoder recovers a fixed vector $\hat{\mathbf{x}}_c$ from a table M called the **codebook** or the **reproduction alphabet**. Since the coding is generally lossy (the exception would be the rare case when all inputs exactly match some codebook vector), we do not insist that the dimension of the output match the dimension of the input. When they do, the error (quantization loss) can be measured by the average (expected) distortion introduced by the VQ scheme, again measured in decibels:

$$\mathrm{SQNR} = 10 \log_{10} \frac{E(\|\mathbf{x}\|)}{E(d(\mathbf{x}, \hat{\mathbf{x}}))} \tag{9.26}$$

In general, there is no reason to believe that Euclidean distance is ideally suited to measuring the actual distortion caused by a VQ system. For example, if the vectors to be transmitted are vowel formant frequencies, as in the Peterson-Barney data discussed in Section 8.1, it is well-known that the perceptual effects of perturbing F1, F2, and F3 are markedly different (see Flanagan 1955), and a distortion measure that is invariant under permutation of the vector components is structurally incapable of describing this situation well.

If we know the probability distribution P over some space X of the vectors to be transmitted and know the ideal distance measure d, the problem reduces to a task of *unsupervised clustering*: find codebook vectors $\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_n$ such that SQNR is maximized. Since the numerator of (9.26) is given, the task is to minimize the denominator, which, assuming the $\mathbf{x}$ inputs are drawn randomly according to P, is given by

$$\int_X d(\mathbf{x}, \hat{\mathbf{x}}) dP(\mathbf{x}) \tag{9.27}$$

a quantity known as the **reconstruction error** of the VG system defined by γ and the vectors $\hat{\mathbf{x}}_i$. Clearly, the larger the codebook, the more we can decrease the reconstruction error, but this is offset by an increased number of bits required to transmit the VQ codes. According to the MDL principle (see Section 7.3.1), we need to optimize the cost (in bits) of transmitting the channel symbols plus the cost of encoding $\mathbf{x}$ given $\hat{\mathbf{x}}_i$. In practical systems where the inputs are N-dimensional vectors of 32-bit reals, the cost of transmitting the residuals $\mathbf{x} - \mathbf{x}_i$ would be overwhelming compared with the $\log_2(|M|)$ bits required to transmit the codes. Without knowing much about the probability distribution P, we would have to dedicate $32N$ bits to transmitting a residual, and in practice codebooks over $|M| > 2^{32}$ make little sense as they would take up too much memory (in fact, typical codebook sizes are in the 2^{12}–2^{20} range).

If the size m of the codebook (also known as the *number of levels* in analogy to the scalar case) is set in advance, many clustering algorithms are applicable. The most commonly used one is Lloyd's algorithm, also known as the LBG algorithm, which uses a random sample of signals $\mathbf{s}_k$ $(1 \leq k \leq r \gg m)$. First we pick, either based on some knowledge about P or randomly, cluster seeds $\mathbf{x}_i^{(0)}$ $(1 \leq i \leq m)$ and consider for each $\mathbf{s}_i$ the $\mathbf{x}_j^{(0)}$ that is closest to it as measured by distance d: the index j (given in binary) is defined as $\gamma^{(0)}(\mathbf{s}_i)$. In step $k+1$, we take the (Euclidean) centroid of all samples in the inverse image of $\gamma^{(k)}(i)$, and use these as the reconstruction vector $\mathbf{x}_i^{(k+1)}$ transmitted by the label $\gamma^{(k+1)}(i)$. We stop when the recognition error no longer improves (which can happen without having reached a global optimum; see Gray and Karnin 1982). With carefully controlled VQ techniques, taking full advantage of the HMM structure, communications quality speech coding at 100–400 bps, approaching the phonemic bitrate is possible (Picone and Doddington 1989). Concatenation techniques, which use the same Viterbi search as HMMs, can improve this to toll quality without increasing the bitrate (Lee and Cox 2001).

If the final goal is not just compression but recognition of the compressed signal, the situation changes quite markedly in that *supervised clustering* techniques now become available. Here we assume that we have *labeled* (also known as *truthed*) vectors $\mathbf{s}_k$ whose recognized (truth) value is $t(\mathbf{s}_k)$ taken from a finite set $t_1, \ldots, t_l$. Ideally, we would want to set $m = l$, so that the quantization provides exactly as many levels as we would ultimately need to distinguish in the classification stage, but in practice this is not always feasible. For example, if the labels correspond to phonemes, there may be very distinct signals (corresponding e.g. to trilled vs. tapped r sounds) that occupy very distinct regions of the signal space, so that clustering them together would result in a centroid that is part of neither the trilled nor the tapped region of signals. (English makes no phonological distinction between these two clusters, but in languages like Spanish the distinction is phonological: consider *perro* 'dog' vs. *pero* 'but'.) Assuming $m \geq l$ still makes sense because transmitting differently labeled $\mathbf{s}$ values by the same code would doom to failure whatever recognition algorithm we may want to use for the reconstructed signal.

In the supervised case, any distortion that leaves the truth value intact can be neglected, so the distance $d(\mathbf{s}, \hat{\mathbf{s}})$ between the original and the reconstructed signal should be set as 0 if they have the same label and as 1 if they do not. As long as there is no absolute neutralization (i.e. no indistinguishable signals can carry different labels, see Section 8.1), the inverse images of the labels $t_1, \ldots, t_l$ partition the signal space X into disjoint sets $X_1, \ldots, X_l$ (plus possibly a remainder set X_0 containing nonlabelable 'junk' data), and it is natural to take the centroids of X_i as the codebook vectors. When the X_i are very far from perfect spheres in Euclidean space, it may make a great deal of sense to approximate them by the union of many spheres and simply take the labeled samples as the center of each sphere. Such systems are known as **nearest neighbor classifiers** since for any incoming signal $\mathbf{x}$ they find the nearest (in Euclidean distance) labeled sample $\mathbf{s}_i$ and assign it the same label $t(\mathbf{s}_i)$.

The ideal metric d that would yield 0 (resp. 1) between two samples exactly when they have the same (resp. different) labels is only known to us in artificially created examples. On real data we must work with some approximation. In the case of speech signals, it is reasonable to assume that if two signals have very little difference in their energy spectra, they are more likely to be tokens of the same type than when their energy spectra indicate audible differences at some frequency. We therefore investigate the case where d measures the total energy of the quantization error – for this to be meaningful, we assume the original signals are all normalized (amplitude scaled) to have the same energy $\sum_i s_i^2 = 1$ and that the gain of the all-pole filters used to encode the data is set in accordance with (9.25).

Suppose the task is to recognize utterances of roughly equal duration (N samples) from a closed set (e.g. isolated digits), with each of the utterance types $t_1, \ldots, t_l$ having equal probability $1/l$. We have a number of labeled sample utterances, but our goal is not to identify them (since no two utterances are perfectly identical, the sample could just be memorized) but rather to identify hitherto unseen signals taken from the same subset X of $\mathbb{R}^N$, $X = \bigcup_{i=0}^{l} X_l$, where $P(X_0) = 0, P(X_i) = 1/l \ (1 \leq i \leq l)$. Our goal is to create codebook vectors $\mathbf{c}_i$ in p-dimensional space, $p \ll N$ for each of the l clusters X_i so that we get a nearest neighbor classifier. For any incoming signal s, we compute $d(s, \mathbf{c}_i)$ for $1 \leq i \leq l$ and select the i for which this value is minimal. We want the quantization error to be relatively small when the LPC model of s is close to one of the $\mathbf{c}_i$, and we are not much worried about the fringe cases when it is far from each $\mathbf{c}_i$ since their probability is low.

Let us pick a single cluster center $\mathbf{c}$, given by an all-pole filter with predictor parameters $a_1, \ldots, a_p$ and gain G. If we use white noise for input to this filter, we should obtain unvoiced versions (akin to whispering but without decrease in signal energy) of the original signals. The log probability of obtaining any particular signal s from $\mathbf{c}$ is given approximately by

$$\log P(s|\mathbf{c}) = \frac{N}{2}\left(\log 2\pi G^2 + \frac{1}{G}\mathbf{c}R_s\mathbf{c}^T\right) \tag{9.28}$$

Here R_s is the autocorrelation matrix of s, which is symmetric and positive semidefinite. Instead of a true distance (symmetrical and satisfying the triangle inequality) we only have a weaker type of d called a *divergence*, which will be small if the signal s is close to the model $\mathbf{c}$ and large if it is not. Equation (9.28), known as the **Itakura-Saito divergence**, is a special case of the Kullback-Leibler divergence (see Section 7.2). If we apply Lloyd's algorithm with the Itakura-Saito divergence, what we do in effect is average (in the Euclidean sense) the R_x matrices for each class X_i and compute the model $\mathbf{c}_i$ on the basis of these average matrices using the autocorrelation version of (9.22).

9.2 Phonemes as hidden units

Conceptually we can distinguish two main clusters of phenomena in the study of speech: *phonological* and *phonetic*. On the phonological side, we find discrete mental units defined in terms of contrast, where change from one unit to the other results in change of meaning. A good example is tone, where typically only two levels, high and low, are available – some languages have three, but languages such as Mandarin Chinese that distinguish many 'tones' are really distinguishing many tonal configurations (sequences of high and low tones, see Section 3.3). On the phonetic side we have continuously variable physical parameters like pitch (the frequency F0 with which the vocal folds open and close) that can be changed by any small amount without affecting the meaning. Almost all phenomena we discussed in Chapters 3 and 4 fit squarely in the phonological cluster, while almost everything about the signals discussed so far indicates continuity, and discretization by sampling or VQ does not alter this picture since the number of discrete levels used in these steps is vastly larger than the number of discrete units. For example, in discretizing pitch, 256–1024 levels (8–10 bits) are common, while for tone we would generally need only 2–4 levels (1 or 2 bits).

The conceptual distinction is matched by reliance on different sorts of evidence: phonology views the human apparatus for speech production and perception as a legitimate instrument of data collection and relies almost exclusively on data (judgments concerning the grammaticality and well-formedness of certain forms) that phoneticians regard as subjective, while phonetics prefers to consider objective data such as speech waveforms. Yet the distinction between phonological and phonetic is by no means clear-cut, and the theory of lexical phonology and morphology (LPM, see Kiparsky 1982) distinguishes between two classes of phonological rules, *lexical* and *postlexical*, of which only the lexical class has clearly and unambiguously phonological character – the postlexical class shares many key features with purely phonetic rules. Here the distinctions, as summarized in Kaisse and Hargus (1993), are drawn as follows:

	Lexical	**Postlexical**
(a)	word-bounded	not word-bounded
(b)	access to word-internal structure assigned at same level	access to phrase structure only
(c)	precede all postlexical rules	follow all lexical rules
(d)	cyclic	apply once
(e)	disjunctively ordered with respect to other lexical rules	conjunctively ordered with respect to lexical rules
(f)	apply in derived environments	apply across the board
(g)	structure-preserving	not structure-preserving
(h)	apply to lexical categories only	apply to all categories
(i)	may have exceptions	automatic
(j)	not transferred to a second language	transferable to second language
(k)	outputs subject to lexical diffusion	subject to neogrammarian sound change
(l)	apply categorically	may have gradient outputs

While the criteria *(a–l)* are only near-truths, they are sufficient for classifying almost any rule as clearly lexical or postlexical. This is particularly striking when processes that historically start out as phonetic get *phonologized.* Not only will such rules change character from gradual (continuous) to discrete *(l)*, they will also begin to affect different elements of the lexicon differently *(k)*, acquire exceptions *(i)* and morphological conditions *(h)*, and begin to participate in the phonological rule system in ways phonetic processes never do *(c–f)*.

A good example is provided by phonetic *coarticulation*, a process that refers both to the local smoothing of articulatory trajectories and to longer-range interactions that can be observed e.g. between two vowels separated by a consonant (Öhman 1966) whenever some speech organs move into position before the phoneme that will require this comes up or stay in position afterwards. When the effect gets phonologized, it can operate over very long, in principle unbounded, ranges – a famous example is the *ruki* rule in Sanskrit, which triggers retroflexation of *s* after phonemes in the ruki class no matter how many nonruki consonants and vowels intervene. In reference to their phonetic origin, such rules are known in phonology as *anticipatory* and *perseveratory* rules of assimilation, irrespective of their range.

In Chapters 3 and 4 we presented all the discrete units generally agreed upon in phonology: features (autosegments), phonemes, syllables, and words. (We also presented some that are less widely used, such as moras, feet, and cola – here we will simplify the discussion by concentrating on the better-known units.) Of these, only words have a clear relationship to meanings. All others are motivated by the mechanics of speech production and are meaningless in themselves. Even for words, the appropriate phonological notion, the *prosodic word*, does not entirely coincide with the grammatical (syntactic) notion of wordhood (see Section 4.1), but the two are close enough to say that we can separate words from one another on the basis of meaning most of the time.

How can the discrete and meaningless units used in phonology be realized in, and recovered from, the undifferentiated continuous data provided by acoustic signals? What we call the *naturalistic* approach is to trace the causal chain, to the extent feasible, from the brain through the movement of articulators and the resulting air pressure changes. Within the brain, we assume some kind of combinatorical mechanism capable of computing the phonological representation of an utterance from pieces stored in the mental lexicon. This representation in turn serves as a source of complex nerve impulse patterns driving the articulators (Halle 1983), with the final output determined by the acoustics of the vocal tract.

Note that the combinatorical mechanism does not necessarily operate left to right. In particular, intermediate representations, procedures, and structures are generally viewed as having no theoretical status whatsoever, comparable to the scratch paper that holds the intermediate results in long division. This view is shared by context-sensitive rule-based phonology (Chomsky and Halle 1968), finite-state approaches (Koskenniemi 1983), and optimality theory (Prince and Smolensky 1993). Only multistratal theories, such as LPM, treat the output of the individual levels as real, and even then there is no promise of left to right computation. In particular, there is no *Greibach normal form* (see Ex. 2.4) that would force outputting of a segment for each unit of computation effort.

To the extent that speech production and speech perception rely on the same discrete phonological representation, tracing the causal chain on the decoder side implies that the acoustic signal is perceived in terms of the same articulator movement patterns as were used in producing the signal. This is known as the *motor theory of speech perception* (Liberman 1957). To quote Liberman and Mattingly (1989:491), speech perception

> processes the acoustic signal so as to recover the coarticulated gestures that produced it. These gestures are the primitives that the mechanisms of speech production translate into actual articulator movements, and they are also the primitives that the specialized mechanisms of speech perception recover from the signal.

Tracing the causal chain this way goes a long way toward explaining what the phonological representations, so painstakingly built by the linguist, are good for (besides accounting for the linguistic data of course). If the representations can be recast in terms of articulatory gestures, and moreover if indeed these gestures provide the key to speech perception, a wealth of extralinguistic evidence, ranging from X-ray microbeam tracing of the articulators (Fujimura et al. 1973) to perception studies of formant location (Klein et al. 1970), can be brought to bear on the description of these representations.

There are two problems left unanswered by following the causal chain. First, the gap between the discrete and the continuous is left unbridged: even if we identify phonological representations with gestural scores, these are continuous at best for timing parameters and the main gestural subcomponents (such as opening or closing the lips, raising or lowering the velum, etc.) remain discrete. Whatever we may do, we still need to recover the discrete articulatory configurations from a continuum of signals. Second, there is an important range of phenomena from sign language to handwriting that raises the same technical issues, but this time without the benefit of a complex (and arguably genetically set) mechanism between the representation and the perceived signal.

It is at this point that the modern theory of speech recognition parts with the naturalistic theory: if there is a need to create a mapping from discrete elements to continuous realizations in any case, there does not seem to be a significant advantage in creating an intermediate representation that is tightly coupled to a complex mechanism that is specific to the physiology of the vocal tract. As a case in point, let us consider the theory of distinctive features (see Section 3.2). A rather detailed qualitative description of the articulatory and acoustic correlates of distinctive features was available as early as in Jakobson et al. (1952). Nearly three decades later, Stevens and Blumstein (1981) still had not found a way of turning this into a quantitative description that could be used to automatically detect features (see e.g. Remez 1979), and to this day research in this area has failed to reveal a set of reliable acoustic cues for phonological features of the sort envisioned in Cherry, Halle, and Jakobson (1953) and Cherry (1956).

Thus the naturalistic model that interposes a gestural layer between the mental representations and the acoustic signal has been replaced by a simpler and more direct view of the mental lexicon that is assumed to store highly specific acoustic engrams recorded during the language acquisition process: these engrams can be directly used as lookup keys into a mental database that will contain syntactic, semantic, morphological, and other nonacoustic information about the form in question (Klatt 1980). Under this view, surface forms are just acoustic signals, while underlying forms could contain detailed articulatory plans for the production of the form, together with links to semantic, syntactic, and morphological information stored in various formats.

The relationship between psychological units of linguistic processing and their physical realizations is many to many, both with different phonological representations corresponding to the same utterance (neutralization; see Section 8.1) and with the same representation having many realizations, and many conditioning factors, ranging from the physical differences among speakers sharing the same competence to the amount of distortion tolerated in the realization process. While phonology generally works with idealized data that preserve dialectally and grammatically conditioned variation but suppress variation within the speech of an individual and across individuals sharing the same dialect/sociolect, for the moment we lump all sources of variation together, and defer the issue of how to separate these out.

The units that we shall take as basic are the phonemes, which are instrumental in describing such a broad range of phenomena that their psychological reality can hardly be disputed. A subjective, but nevertheless important factor is that most researchers are convinced that they are in fact communicating using sentences, words, syllables, and phonemes. A great deal of the reluctance of speech engineers to accept distinctive features can no doubt be attributed to the fact that for features this subjective aspect is missing: no amount of introspection reveals the featural composition of vowels, and to the extent introspection works (e.g. with place of articulation) it is yielding results that are not easily expressible in terms of distinctive features unless a more complex structure (feature geometry, see Section 3.2) is invoked.

A less subjective argument in favor of certain linguistic units can be made on the basis of particular systems of writing. To the extent that a morpheme-, mora-, syllable-, or phoneme-based writing system can be easily acquired and consistently used by any speaker of the language, the psychological reality of the units forming the basis of the system becomes hard to deny. Distinctive features fare slightly better under this argument, given sound-writing systems such as Bell's (1867) Visible Speech or Sweet's (1881) Sound Notation, but to make the point more forcefully, the ease of use and portability of such writing systems to other languages needs to be demonstrated. For now, the most portable system we have, the *International Phonetic Alphabet* (IPA), is alphabetic, though the idea that its organization should reflect the featural composition of sounds is no longer in doubt (see Halle and Ladefoged 1988).

The simplest direct model with phonemic units would be one where the mapping is one to one, storing a single signal template with each phoneme. To account for variation, we need to use a probability model of templates instead, leaving open the possibility, corresponding to neutralization, that the same template may have nonzero probability in the distribution associated to more than one phonemic unit. We thus obtain a hidden Markov model, where the hidden units are phonemic, and the emission probabilities model the acoustic realization of phonemes. Transition probabilities can be set in accordance with the phonotactics of the language being modeled or, if a description of the lexically stored phoneme strings (words) is available, in accordance with this description.

Since the phonemic units are concatenative, the output signals should also be, meaning either that we *smooth* the abrupt change between the end of one signal (coming from one hidden state) and the beginning of the next signal (coming from another hidden state) or we adjust the emissions so that only smoothly fitting signals can follow each other. While concatenation and smoothing continues to play an important role in speech synthesis systems (see Klatt 1987, van Santen et al. 1997), in speech recognition the second option has proven more fruitful: instead of directly modeling phonemes, we model phonemes *in context*. For example, instead of a single model for the i in *mint*, *hint*, *lint*, and *tint* we build as many separate models as there can be phonemes preceding the i. If we do the same with the l phoneme, building as many models as there can be phonemes following it (so that different ls are used in *lee*, *lie*, *low*, *lieu*, etc.), we can be reasonably certain that the appropriate l model (one that is based on the context $\_i$), when followed by the appropriate i model (one that is based on the context $l\_$), will contain only signals that can be concatenated without much need for smoothing.

HMMs in which the hidden units are phonemes in two-sided phoneme contexts are called *triphone* models (the name is somewhat misleading in that the units are single phones, restricted to particular contexts, rather than sequences of three phones) and will contain, if there were n phonemes, no more than n^3 hidden units, and possibly significantly fewer if phonotactic regularities rule out many cases of phoneme b appearing in context a_c. State of the art systems extend this method to *quinphones* (phonemes in the context of two phonemes on each side) and beyond, using cross-word contexts where necessary.

Starting with Bloomfield, a great deal of effort in mathematical linguistics has been devoted to defining models that explicate the relation between the low-level (continuous, phonetic, meaning-preserving) and the high-level (discrete, phonological, meaning-changing) items and processes involved in speech. But there are some persistent difficulties that could not be solved without a full appreciation of the variability of the system. A triphone or quinphone model will account for a great deal of this variability, but even a cursory look at Fig. 8.1 makes it evident that other sources, in particular the identity of the speaker, will still contribute significant variability once contextual effects are factored out. Also, it is clear that steady-state vowels are more of an idealization than typical speech samples: major spectral shifts occur every 5–10 milliseconds, and windows that contain no such shifts are a rarity. This problem is to some extent mitigated by adding *delta* (first derivative) and *delta delta* (second derivative) features to the basic feature set (Furui 1986), since this will re-emphasize the spectral shifts that the original features may dampen.

The most natural probabilistic model of emission is a normal distribution, where a single sample acts as the mean μ, and all samples are assigned probabilities in accordance with the density function

$$\frac{1}{(2\pi)^{N/2}|R|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)R^{-1}(x-\mu)^T\right] \tag{9.29}$$

where R is the N-dimensional covariance matrix that determines the distribution. In the simplest case, $N = 1$; i.e. there is a single measurable parameter that characterizes every sample. Ideally, this is what we would like to see e.g. for tone, which is determined by a single parameter F0. But when we measure F0 for H (phonologically high tone) and L (phonologically low tone) syllables, we obtain two distributions that are nearly identical, with neither the means μ_H and μ_L nor the variances σ_H and σ_L showing marked differences. The reason for this is that tonal languages show a steady, cumulative lowering of F0 called *downdrift*, which obscures the differences between H and L tones on average. At any given position, in particular sentence-initially, the differences between H and L produced by the same speaker are perceptible, but averaging speakers and positions together blurs the distinction to such an extent that separation of H and L by unsupervised clustering becomes impossible. This is not to say that the distinction cannot be captured statistically, e.g. by focusing on the phrase-initial portion of the data, but rather to emphasize that our current techniques are often insufficient for the automatic discovery of structure from the raw data: unless we *know* about downdrift, there is no reason to inspect the phrase-initial portion of the data separately.

This is not to say that unsupervised, or minimally supervised, clustering techniques have no chance of obtaining the correct classes, at least if the data are presented clearly. For example, if we restrict attention to steady-state vowels with unambiguous pronunciation and a homogeneous set of speakers (adult males), the approximate (F1, F2, F3) centroids for the ten vowels measured by Peterson and Barney (1952) can be found using just the fact that there are exactly ten clusters to be built (Kornai 1998a). But to accommodate cases of major allophonic variation, such as trilled vs. tapped r, distinct Gaussians must be assigned to a single phoneme model. In this case, we talk about *mixture models* because the density function describing the distribution of data points belonging to a single phoneme is the mixture (weighted sum) of ordinary Gaussians. By using a large number of mixture components, we can achieve any desired fit with the training data. In the limiting case, we can fit a very narrow Gaussian to each data point and thereby achieve a perfect fit.

The number of Gaussians that can be justified is limited both by the MDL principle (see Section 7.3) and by the availability of training data. If we are to model n^3 hidden units (triphones), each with N parameters for the mean and $N(N+1)/2$ for variances and covariances, using m mixture components will require a total of $O(n^3N^2m/2)$ parameters. With typical values like $n = 50,\ N = 40,\ m = 10$, this would mean 10^9 parameters. Of the many strategies used for reducing this number, we single out the use of *diagonal* models where only the variances are kept and the covariances are all set to zero. This will reduce the parameter space by a factor of $N/2$ at minimal cost since the covariances refer to different dimensions of a highly compressed (mel cepstral) representation whose individual components should already be almost entirely uncorrelated. Another method is to use only a limited number of Gaussians and share these (but not the mixture weights) across different hidden units. This is called a *tied mixture* model (Bellegarda and Nahamoo 1990). Tying the parameters and reducing the number of Gaussians is particularly important in those cases where not all phonotactically permissible triphones occur in the training data.

Another important approach to reducing variation is based on *speaker adaptation*. The Peterson-Barney data already show clearly the effects of having men, women, and children in the sample, and indeed training separate models for men and women (Nishimura and Sugawara 1988) is now common. If there are separate mixture components for men and women, it makes eminent sense to deploy *selection* strategies that use some short initial segment of speech to determine which component the data fits best and afterwards suppress the mixtures belonging in the other component or components.

Speaker adaptation can also work by noting the characteristic differences between the speech of the current speaker and those whose speech was used to train the model, and employ some transformation T^{-1} to the incoming speech or, alternately, applying T to the model data, as a means of achieving a better fit between the two. The former method, e.g. by the normalization of cepstral means, is used more when the variation is due to the environment (background noises, echoes, channel distortion), while the latter is used chiefly to control variability across speakers and to some extent across dialects (especially for nonnative speakers).

Adapting the variances (and covariances, if full covariance models are used) is less important than adapting the means, for which generally T is chosen as an affine transform $\mathbf{x} \mapsto \mathbf{x}A+\mathbf{b}$, and the new means are computed by maximum likelihood linear regression (MMLR; see Leggetter and Woodland 1995). A more naturalistic, but not particularly more successful, method is vocal tract length normalization (VTLN, see Wakita 1977), where the objective is to compensate for the normal biological variation in the length of the vocal tract. Altogether, the naturalistic model remains a source of inspiration for the more abstract direct approach, but the success of the latter is no longer tied to progress in articulatory or perceptual research.

9.3 Handwriting and machine print

In dynamic OCR, we can be reasonably certain that the input the system receives is writing, but in image-based OCR the first task is *page decomposition*, the separation of linguistic material from photos, line drawings, and other nonlinguistic information. A further challenge is that we often find different scripts, such as Kanji and Kana, or Cyrillic and Latin, in the same running text.

The input is generally a scanned image, increasingly available in very high resolution (600 dpi or more) and in full (4 bytes per pixel) color. Uncompressed, such an image of a regular letter-size page would take up over 130 MB. The naturalistic program would suggest applying algorithms of *early vision* such as edge detection, computation of optical flow, lightness, albedo, etc., to derive representations more suitable for page decomposition and character recognition. However, practical systems take the opposite tack and generally begin with a variety of data reduction steps, such as *downsampling*, typically to fax resolution (200 dpi horizontal, 100 dpi vertical) and *binarization*; i.e. replacing color or grayscale pixel values by binary (black and white) values. After these steps, a typical uncompressed image will be over 1.8 MB, still too large for fax transmission of multipage documents.

Here we describe the **G3** (CCITT Group 3) standard in some detail because it illustrates not only the basic ideas of Huffman coding (see Section 7.1) but also the distance between theoretical constructs and practical standards. Each line is defined as having 1728 binary pixels (thus somewhat exceeding the 8.5 inch page width used in the United States at 200 dpi) that are *run-length encoded* (RLE). In RLE, instead of transmitting the 0s and 1s, the length of the alternating runs of 0s and 1s get transmitted. In the G3 standard, length is viewed as a base 64 number. For shorter runs, only the last digit (called *terminating length* in this system) gets transmitted, but for longer runs, the preceding digit (called the *make-up*) is also used. Since the length distribution of white and black runs differs considerably, two separate codebooks are used. Terminating length and make-up codes jointly have the prefix-free property both for white and black, but not for the union of the two codebooks. The following terminating length and make-up codes are used:

term. length	white code	black code	term. length	white code	black code	make up	white	black
0	00110101	0000110111	32	00011011	000001101010			
1	000111	010	33	00010010	000001101011	64	11011	0000001111
2	0111	11	34	00010011	000011010010	128	10010	000011001000
3	1000	10	35	00010100	000011010011	192	010111	000011001001
4	1011	011	36	00010101	000011010100	256	0110111	000001011011
5	1100	0011	37	00010110	000011010101	320	00110110	000000110011
6	1110	0010	38	00010111	000011010110	384	00110111	000000110100
7	1111	00011	39	00101000	000011010111	448	01100100	000000110101
8	10011	000101	40	00101001	000001101100	512	01100101	0000001101100
9	10100	000100	41	00101010	000001101101	576	01101000	0000001101101
10	00111	0000100	42	00101011	000011011010	640	01100111	0000001001010
11	01000	0000101	43	00101100	000011011011	704	011001100	0000001001011
12	001000	0000111	44	00101101	000001010100	768	011001101	0000001001100
13	000011	00000100	45	00000100	000001010101	832	011010010	0000001001101
14	110100	00000111	46	00000101	000001010110	896	011010011	0000001110010
15	110101	000011000	47	00001010	000001010111	960	011010100	0000001110011
16	101010	0000010111	48	00001011	000001100100	1024	011010101	0000001110100
17	101011	0000011000	49	01010010	000001100101	1088	011010110	0000001110101
18	0100111	0000001000	50	01010011	000001010010	1152	011010111	0000001110110
19	0001100	00001100111	51	01010100	000001010011	1216	011011000	0000001110111
20	0001000	00001101000	52	01010101	000000100100	1280	011011001	0000001010010
21	0010111	00001101100	53	00100100	000000110111	1344	011011010	0000001010011
22	0000011	00000110111	54	00100101	000000111000	1408	011011011	0000001010100
23	0000100	00000101000	55	01011000	000000100111	1472	010011000	0000001010101
24	0101000	00000010111	56	01011001	000000101000	1536	010011001	0000001011010
25	0101011	00000011000	57	01011010	000001011000	1600	010011010	0000001011011
26	0010011	000011001010	58	01011011	000001011001	1664	011000	0000001100100
27	0100100	000011001011	59	01001010	000000101011	1728	010011011	0000001100101
28	0011000	000011001100	60	01001011	000000101100			
29	00000010	000011001101	61	00110010	000001011010			
30	00000011	000001101000	62	00110011	000001100110			
31	00011010	000001101001	63	00110100	000001100111			

In order to ensure that the receiver maintains color synchronization, all lines must begin with a white run length code word (if the actual scanning line begins with a black run, a white run length of zero will be sent). For each page, first end of line (EOL, 000000000001) is sent, followed by variable-length line codes, each terminated by EOL, with six EOLs at the end of the page. On the average, G3 compression reduces the size of the image by a factor of 20.

Exercise 9.4 Research the CCITT Group 4 (G4) standard, which also exploits some of the redundancy between successive lines of the scanned image and thereby improves compression by a factor of 2. Research JBIG and JBIG2, which generally improve G4 compression by another factor of 4.

Since the black and white runs give a good indication of the rough position of content elements, the first step of page decomposition is often performed on RLE data, which is generally sufficient for establishing the local horizontal and vertical directions and for the appropriate grouping of titles, headers, footers, and other material set in a font different from the main body of the text. Adaptation to the directions inherent in the page is called *deskewing*, and again it can take the form of transforming (rotating or shearing) the image or transforming the models. The tasks of deskewing and page decomposition are somewhat intertwined because the simplest page decomposition methods work best when the image is not skewed. Unlike in speech recognition, where models that incorporate an explicit segmentation step have long been replaced by models that integrate the segmentation and the recognition step in a single HMM search, in OCR there is still very often a series of segmentation steps, first for *text zones* (i.e. rectangular windows that contain text only), then for *lines*, and finally for *characters.*

The search for text zones can proceed top-down or bottom-up. In the top-down approach, we first count the black pixels in each row (column), obtaining a column (row) of blackness counts known as the *projection profiles* (see Wang and Srihari 1989). These are generally sufficient for finding the headers and footers. Once these are separated out, the vertical profile can be used to separate text columns, and on each column horizontal profiles can be used to separate the text into lines. Besides sensitivity to skew, a big drawback of the method is that it presumes a regular, rectangular arrangement of the page, and more fancy typography, such as text flowing around a circular drawing, will cause problems. In the bottom-up approach, we begin with the smallest elements, *connected components*, and gradually organize them or their *bounding boxes* (the smallest rectangle entirely containing them) into larger structures. By searching for the dominant peak in a histogram of vectors connecting each component to its nearest neighbor, the skew of the document can be reliably detected (Hashizume et al. 1986, O'Gorman 1993). Both the identification of connected components and the finding of nearest neighbors are computationally very expensive steps, and once we are willing to spend the resources, other skew-insensitive methods of segmenting text from images, such as the use of Gabor filters (Jain and Bhattacharjee 1992) are also available.

Besides deskewing, there are several other normalization steps performed on the entire image as needed; for example, xeroxing or scanning thick, bound documents introduces *perspective distortion* at the edge of the page (see Kanungo et al. 1993), and the vibrations of the scanner can cause blurring. Detection and removal of *speckle noise* (also known as *salt-and-pepper noise* because noise can take the form of unwanted white pixels, not just black) is also best performed on the basis of estimating the noise parameters globally. Other normalization steps, in particular removing distortions in the horizontal baseline caused by a loose paper forwarding mechanism (common in mechanical typewriters and low-quality scanners/faxes), are better performed line by line.

To the extent that small skew (generally within one degree) and simple rectangular page layout are valid assumptions for the vast majority of holdings in digital libraries, the less expensive top-down algorithms remain viable and have the advantage that finding text lines and characters can generally proceed by the same steps. For machine-printed input and for *handprint* (block letters, as opposed to cursive writing), these steps reduce the problem to that of *isolated character recognition.* Here the dominant technology is template matching, typically by neural nets or other trainable algorithms. Since direct matching of templates at the bitmap level, the method used in the first commercially available OCR system in 1955, works well only for fixed fonts known in advance, attention turned early on to deriving a suitable set of features so that variants of the same character will map to close points in feature space.

For isolated characters, the first step is generally *size* normalization; i.e. rescaling the image to a standardized window. Since this window is typically much smaller than the original (it can be as small as 5 by 7 for Latin, Cyrillic, and similar alphabets, 16 by 20 for Oriental characters), the new pixels correspond to larger zones of the original bounding box. The new pixel values are set to the average blackness of each zone, so that the rescaled image will be grayscale (4–8 bits) even if the original was binary (Bokser 1992). Because characters can vary from the extremely narrow to the extremely broad, the aspect ratio of the original bounding box is generally kept as a separate feature, together with the position of the bounding box relative to the baseline so as to indicate the presence or absence of ascenders and descenders.

Besides absolute size, we also wish to normalize *stroke width* in handwritten characters and *font weight* in machine print. An elegant, and widely used, technology for this is *mathematical morphology* (MM), which is based on the dual operations of *erosion* and *dilation*. We begin with a fixed set $B \subset \mathbb{R}^2$ called the *structuring element*, typically a disk or square about the origin. The **reflection** of B, denoted $\hat{B}$, is defined as $\{-\mathbf{x}|\mathbf{x} \in B\}$ – for the typical structuring elements, symmetrical about the origin, we have $\hat{B} = B$. The **translation** $B_\mathbf{x}$ of B by vector $\mathbf{x}$ is defined as $\{\mathbf{b}+\mathbf{x}|\mathbf{b} \in B\}$. The **dilation** $A \oplus B$ of A by B is defined as $\{x|\hat{B}_\mathbf{x} \cap A \neq \emptyset\}$, and the **erosion** $A \ominus B$ of A by B is defined as $\{x|B_\mathbf{x} \subset A\}$. Two MM operations defined on top of these are the **opening** $A \circ B$ of A by B given by $(A \ominus B) \oplus B$ and the **closing** $A \bullet B$ of A by B given by $(A \oplus B) \ominus B$. Finally, the **boundary** ∂A of A according to B is defined as $A \setminus (A \ominus B)$.

What makes MM particularly useful for image processing is that all the operations above remain meaningful if the sets A and B are composed of pixels (elements of a tiling of $\mathbb{R}^2$). For example, if the structuring element B is chosen to be slightly smaller than the regular printed dot (both periods at sentence end and dots over is), the operation $(A \circ B) \bullet B$ will filter out all salt-and-pepper noise below this size while leaving the information-carrying symbols largely intact. By iterated erosion, we can also obtain the **skeleton** of an image, defined as the set of those points in the image that are equidistant from at least two points of the boundary. While in principle the skeleton should be an ideal replacement of characters with greater stroke width, in practice skeletonization and all forms of thinning are quite sensitive to noise, even after despeckling. Noise is also a persistent problem for approaches based on *chain codes* that express a simply connected two-dimensional shape in terms of its one-dimensional boundary.

In **vertex chain coding**, we apply the same principle to triangular, rectangular, and hexagonal tilings: we pick any vertex of the polygon that bounds the object and count the number of pixels that are connected to the boundary at that vertex. The total chain code is obtained by reading off these numbers sequentially following the polygon counterclockwise. Several chain codes, all cyclic permutations of the same string, could be obtained, depending on the vertex at which we start. For uniqueness, we pick the one that is minimal when interpreted as a number. This will make the code rotation invariant.

Chain codes offer a relatively compact description of simply connected binary images, and efficient algorithms exist to compute many important properties of an image (such as its skew; see Kapogiannopoulos and Kalouptsidis 2002) based on chain codes for all connected components. However, for isolated character recognition, chain codes are very brittle: many characters have holes, and features such as 'being n times connected' (and in general all Betti numbers and other topological invariants) are greatly affected by noise. In the analysis of handwriting, it is a particularly attractive idea (already present in Eden 1961) to build a naturalistic model along the same lines we were all taught in first grade: t is a straight line down, curve to the right, cross near the top. Noise stands in the way of such *structural decomposition* approaches to a remarkable degree, and simple, robust features such as the *height contour*, which is obtained from a line by computing the highest and the lowest black pixel in each column of pixels, turn out to be more helpful in OCR even though they clearly ignore structural features of the original such as loops.

Therefore it is particularly important to look for noise-robust features that preserve as many of the desirable invariance properties of chain codes as feasible. After size normalization, the character image fits in a fixed domain and can be thought of as a function $f(x, y)$ with value zero outside e.g. the unit circle (which is mathematically more convenient than a rectangular bounding box). A typical normalization/feature extraction step is computing the **central moments** μ_{pq} defined in the usual manner by computing the x and y directional means $\overline{x} = \iint xf(x, y)dxdy / \iint f(x, y)dxdy$ and $\overline{y} = \iint yf(x, y)dxdy / \iint f(x, y)dxdy$ and defining

$$\mu_{pq} = \iint (x - \overline{x})^p (y - \overline{y})^q f(x, y) d(x - \overline{x}) d(y - \overline{y}) \qquad (9.30)$$

Clearly, the central moments are invariant under arbitrary translation. Since the moment-generating function and the two-dimensional Fourier transform of f are essentially the same complex function viewed on the real and the imaginary axes, it comes as little surprise that Fourier techniques, both discrete and continuous, play the same pivotal role in the two-dimensional (handwriting signal) case that they have played in the one-dimensional (speech signal) case.

To obtain features that are also invariant under rotation, we can express f in terms of any orthogonal basis where rotation invariance is easily captured. One standard method is to define the n, m radial polynomials $R_{nm}(\rho)$, where n is the *order* of the polynomial, m is the *winding number*, $n - |m|$ is assumed even, $|m| < n$ by

$$R_{nm}(\rho) = \sum_{s=0}^{n-|m|/2} (-1)^s \frac{(n-s)!\rho^{n-2s}}{s!\left(\frac{n+|m|}{2} - s\right)!\left(\frac{n-|m|}{2} - s\right)!} \tag{9.31}$$

and define the n, m **Zernike basis function** $V_{nm}(\rho, \theta)$ as $V_{nm}(\rho, \theta) = R_{nm}(\rho)e^{im\theta}$. These functions are orthogonal on the unit disk, with

$$\iint_{x^2+y^2\leq 1} \overline{V_{nm}(x, y)} V_{pq}(x, y)dxdy = \frac{\pi}{n+1}\delta_{np}\delta_{mq} \tag{9.32}$$

The **Zernike moment** A_{nm} of a function f (already assumed size and translation normalized) is given by

$$A_{nm} = \iint_{x^2+y^2\leq 1} f(x, y)\overline{V_{nm}(\rho, \theta)}dxdy \tag{9.33}$$

If $g(\rho, \theta)$ is obtained from $f(\rho, \theta)$ by rotation with angle α (i.e. $g(\rho, \theta) = f(\rho, \theta - \alpha)$), the n, m-th Zernike moment of g is $A_{nm}e^{-im\alpha}$, where A_{nm} was the n, m-th Zernike moment of f. Thus, the absolute values $|A_{nm}|$ are rotation invariant. Notice that fully rotation-invariant features are not ideally suited for OCR – for example, 6 and 9 would get confused. A more important goal is the normalization of *slant*, both for the recognition of handwriting and for machine print that contains italicized portions. Projection profiles, taken in multiple directions, give a good indication of writing slant and are often used.

Altogether, the computation of features for isolated characters often involves a mixture of normalization and feature extraction steps that may end up producing more data, as measured in bits, than were present in the original image. This is particularly true in cases such as Zernike moments or other series expansion techniques (Fourier, wavelet, and similar techniques are often used), which could lead to a large number of terms limited only by two-dimensional versions of Theorem 9.1.1. In general, the goal is not simply data reduction but rather finding the features that provide good clustering of the data for VQ and other techniques the way Itakura-Saito divergence does for speech signals.

Typically, data reduction is accomplished by **Karhunen-Loève transformation**, also known as **principal component analysis** (PCA). We begin with a set of (real or complex) feature vectors $\mathbf{x}_i$ $(1 \leq i \leq n)$ with some high dimension N and seek to find the projection P to d-dimensional space that retains as much of the variance in the data as possible. To this end, we first compute the (empirical) mean of the data set $\overline{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ and subtract it from all feature vectors (in practical applications, the variances are also often normalized). The covariance matrix of the (now zero mean) vectors is symmetric (or Hermitian, if complex features are used) and thus has orthogonal eigenvectors. These are ordered according to decreasing size of the eigenvalues, and only the first d are kept. P projects the (zero mean) data on the subspace spanned by these eigenvectors.

If the task is recognition, it is at least in principle possible for important information to be lost in PCA since the variability according to some critical feature may be low. To account for this possibility, sometimes linear discriminant analysis (LDA; see Fisher 1936, 1937) is used, but the improvement over PCA is generally slight when LDA is just a preprocessing stage for some more complex recognition strategy such as nearest neighbor classification. The advantage of LDA is in the fact that it will derive a robust classifier in its own right and the training process, which uses only the first and second moments of the distributions of $\mathbf{x}_i$, is very fast.

Exercise 9.5†† Develop a handprint classifier using the NIST isolated character database available at `http://www.nist.gov/srd/nistsd19.htm`. Compare your results with the state of the art.

Most classification methods that work well for machine print and handprint run into serious problems when applied to cursive writing because in the character identification stage it is very hard to recover from errors made in the segmentation stage. The filter- and projection-based page decomposition methods discussed here generalize reasonably well to cursive writing as far as segmentation into text blocks and lines is concerned, but segmenting a line into separate words and a word into separate characters based on these and similar methods is prone to very significant errors. For cursive writing, the segmentation problem must be confronted the same way as in speech recognition. To quote Halle and Stevens (1962:156),

> The analysis procedure that has enjoyed the widest acceptance postulates that the listener first segments the utterance and then identifies the individual segments with particular phonemes. *No analysis scheme based on this principle has ever been successfully implemented.* This failure is understandable in the light of the preceding account of speech production, where it was observed that segments of an utterance do not in general stand in a one-to-one relation with the phonemes. The problem, therefore, is to devise a procedure which will transform the continuously-changing speech signal into a discrete output without depending crucially on segmentation.

To this day, we do not have successful early segmentation, and not for lack of trying. Until the advent of HMMs, there were many systems based on the segmentation-classification-identification pipeline, but none of them achieved performance at the desired level. Today, many of the design features deemed necessary by the prescient Halle-Stevens work, such as the use of generative language models for the lexicon and larger utterances or the pruning of alternatives by multiple passes, are built into HMMs, but the main strength of these systems comes from two principal sources: first, that the recognition algorithm explores segmentation and classification alternatives in parallel, and second, that the system is *trainable*. Parallelism means that

HMMs are capable of considering all segmentation alternatives (hypothesizing every new frame as potentially beginning a new phoneme) without unduly burdening a separate phoneme-level recognizer. Perhaps ironically, the best segmentation results are the ones obtained in the course of HMM recognition: rather than furnishing an essential first step of analysis, segment boundaries arise as a by-product of the full analysis.

9.4 Further reading

Viewed from a contemporary perspective, the great bulk of (digital) signal processing knowledge rests on a clean, elegant foundation of 19th century complex function theory and should be very accessible to mathematicians once the terminological and notational gap between mathematics and electrical engineering is crossed.[1] Yet the rational reconstruction of the fundamentals presented here has little in common with the actual historical development, which is better understood through major textbooks such as Flanagan (1972) (for analog) and Rabiner and Schafer (1978) (for early digital). In particular, Theorem 9.1.1 goes back to Whittaker (1915) but has been rediscovered many times, most notably by Shannon and Weaver (1949).

In the theory of time series analysis, Padé approximation is known as the *autoregressive moving average* (ARMA) model, with the all-pole case referred to as *autoregressive* (AR) and the all-zero case as *moving average* (MA); see e.g. Box and Jenkins (1970). In digital signal processing, there are a wide variety of windowing functions in use (see e.g. Oppenheim and Schafer 1999), but for speech little improvement, if any, results from replacing the standard Hamming window by other windowing functions. The importance of the fast Fourier transform can hardly be overstated – see Brigham (1988) for a textbook devoted entirely to this subject. Even though harmonic analysis is a natural framework for dealing with speech, algorithms that relied on actually computing Fourier coefficients were considered impractical before the modern rediscovery of the FFT (Cooley and Tukey 1965). We mention here that the FFT was already known to Gauss (see Heideman et al. 1984).

The standard introduction to homomorphic speech processing is Schafer and Rabiner (1975), but the presentation here follows more closely the logic of Makhoul (1975). Cepstra, and the attendant syllable-reversal terminology, were introduced in Bogert et al. (1963); see also Childers et al. (1977). Mel-cepstral features have effectively replaced direct (time domain) LPC features, but linear prediction, either applied in the quefrency domain or directly (as in the GSM 6.10 standard), remains a standard data compression method in modern audio transmission.

The savings effected by the fast algorithm of Exercise 9.2 are trivial by today's standards since p is generally on the order of 10^1, so p^3 gives less than 10^5 instructions per second, while the chips embedded in contemporary cell phones are increasingly capable of hundreds of MIPS. Note, however, that in the image domain a variety of operations, such as adaptive binarization (setting the binarization

[1] In particular, in engineering works we often find the imaginary unit i denoted by j.

threshold according to local conditions; see Kamel and Zhao 1993), are still quite expensive.

For an overview of unsupervised clustering, see Everitt (1980) and Duda et al. (2000 Ch. 10), and for the supervised case, see Anderberg (1973). For Itakura-Saito divergence, see Itakura (1975), and for its relation to Bregman divergences, see McAulay (1984), Wei and Gibson (2000), and Banerjee at al (2005). For optimal decorrelation of cepstral features, see Demuynck et al. (1998). Tying techniques, which are critical for training high-quality HMMs, are discussed further in Jelinek (1997 Ch. 10). For speaker adaptation, see Richard Stern's survey article on robust speech recognition in Cole (1997, with a new edition planned for 2007).

Although somewhat dated, both Bunke and Wang (1997) and O'Gorman and Kasturi (1995) offer excellent introductions to the many specialized topics related to OCR. For an overview of the early history, see Mori et al. (1992). For adaptive thresholding, see Sezgin and Sankur (2004). For page segmentation, see Antonacopoulos et al. (2005). Mathematical morphology was invented by Matheron and Serra (see Serra 1982); for various generalizations, see Maragos et al. (1996). For a comparison of skeletalization and thinning algorithms see Lee et al. (1991) and Lam et al. (1992). Chain codes were introduced by Freeman (1961). The vertex chain code presented here is from Bribiesca (1999).

Moment normalization originated with Hu (1962). Zernike polynomials arise in the study of wavefronts for optical systems with a central axis (Zernike 1934) and are widely used in opthalmology to this day. Their use in character recognition was first proposed in Khotanzad and Hong (1990). For an overview of feature extraction methods for isolated character recognition, see Trier et al. (1996). PCA and LDA are basic tools in pattern recognition; see e.g. Duda et al. (2001 Sec. 3.8). For the use of MMLR in dynamic handwriting recognition, see Senior and Nathan (1997), and for image-based handwriting recognition, see Vinciarelli and Bengio (2002). The state of the art in handwriting recognition is closely tracked by the International Workshop on Frontiers of Handwriting Recognition (IWFHR).

Early systems incorporating explicit rule-based segmentation steps are discussed in Makhoul (2006) from a historical perspective. In linguistic pattern recognition, trainability (called *adaptive learning* at the time) was first explored in early OCR work (see Highleyman 1962, Munson 1968), but the practical importance and far-reaching theoretical impact of trainable models remained something of a trade secret to speech and OCR until the early 1990s, when Brown et al. (1990) demonstrated the use of trainable models in machine translation. This is not to say that theoretical linguistics ignored the matter entirely, and certainly the single most influential work of the period, Chomsky (1965), was very explicit about the need for grammatical models to be learnable. Until the 1990s, theoretical linguistics focused on the relationship of learnability and child language development (see e.g. Pinker 1984, 2nd revised ed. 1996), mostly from the perspective of stimulus poverty, and it took the clear victory of trainable models over handcrafted rule systems in what was viewed as a core semantic competence, translation, to bring the pure symbol-manipulation and the statistical approaches together again (Pereira 2000).