CHAPTER 4

Time-Domain Signal Analysis

This chapter surveys methods for time-domain signal analysis. Signal analysis finds the significant structures, recognizable components, shapes, and features within a signal. By working with signals as functions of a discrete or analog time variable, we perform what is called time-domain signal analysis. We view signals as discrete or analog functions. We distinguish one signal from another by the significant differences in their magnitudes at points in time. Ultimately, then, time-domain methods rely in a central way on the level of the signal at an instant, over an interval, or over the entire domain of the signal. Our analysis tools will be both crude and sophisticated, and our achievements will be both problematic and successful. From the perspective of Hilbert space analysis, we will find that our time-domain signal analysis methods often involve the use of basis functions for signal subspaces that have irregular, blocky shapes.

Signal analysis does encompass many signal processing techniques, but it ultimately goes beyond the signal-in, signal-out framework: Signal analysis breaks an input signal down into a nonsignal form. The output could be, for example, an interpretation of the input, recognition results, or a structural description of the source signal. In time-domain analysis we study signals without first deriving their frequency content. Signals are viewed simply as functions of time (or of another independent spatial variable). This chapter's methods depend upon the foundation in determinate and indeterminate signals, systems, and Hilbert space laid previously, together with some calculus, differential geometry, and differential equations. In contrast, a frequency-domain analysis finds the frequencies within a signal by way of Fourier transformation—or, in more modern vein, using the Gabor or wavelet transforms—and uses the frequency information to interpret the signal.

After some philosophical motivation, this chapter considers some elementary signal segmentation examples. This discussion identifies problems of noise, magnitude, frequency, and scale in detecting special signal regions. Signal edges are the obvious boundaries between segmentable signal regions, but detecting them reliably and optimally proves to be harder that it would seem at first glance. Matched filtering is introduced, but the theoretical justifications are postponed. Scale space decomposition gives a complete and general method for the segmentation and structural description of a signal. Pattern recognition networks offer a hybrid scheme for signal detection.

Signal Analysis: Time, Frequency, Scale, and Structure, by Ronald L. Allen and Duncan W. Mills
ISBN: 0-471-23441-9 Copyright © 2004 by Institute of Electrical and Electronics Engineers, Inc.

They combine ideas from matched filtering and decomposition by scale; provide for design rules, as in conventional pattern recognition systems; and have a network training scheme similar to neural networks. Hidden Markov models (HMMs) are a statistical recognition tool applied widely in speech and text interpretation. Later chapters will expand these techniques further by showing how to use these signal decomposition tools with frequency and mixed-domain signal processing.

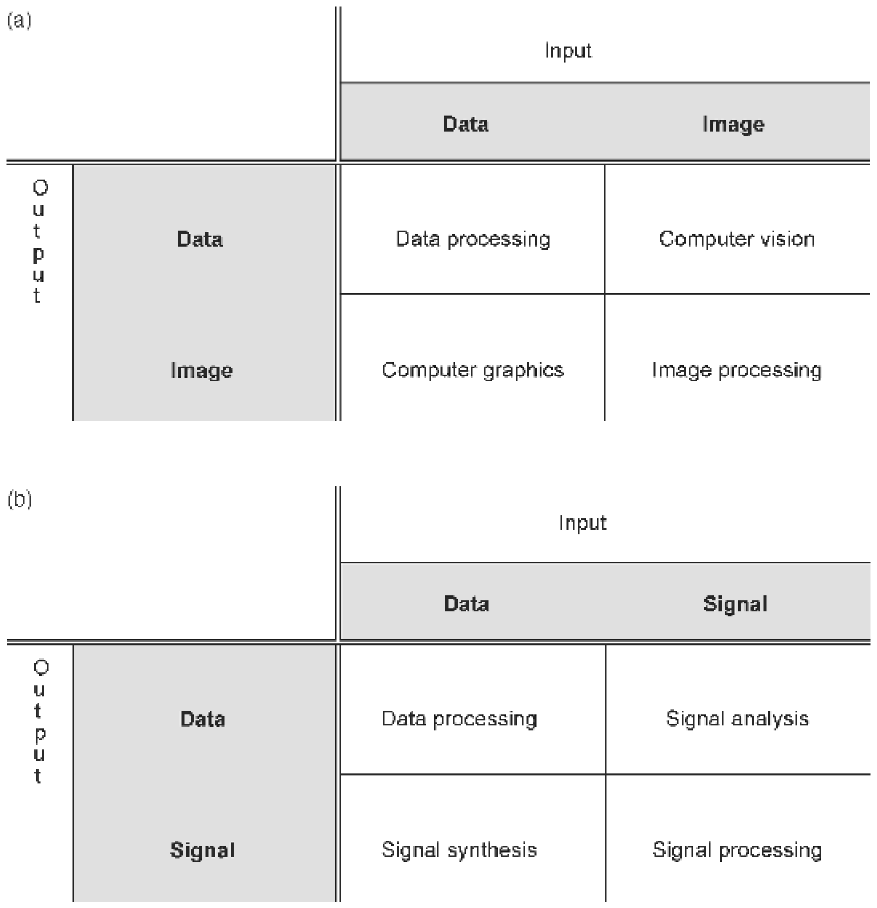
We shall cover in this chapter, as promised, signal analysis techniques that are not commonly found in the standard signal processing texts. But we do more than validate the title of the whole book. By confronting the deep problems involved in finding the content and structure of signals, right after basic presentations on linear systems, we will find both a motivation and a conceptual foundation for the study of frequency and scale in signals. The first three chapters embellished the traditional presentation of signal processing—discrete and analog signals, random signals, periodicity, linear systems—with an introductory development of the theory of Hilbert spaces. The important Hilbert space notion of inner product gives us a theoretical tool for finding the similarity of two signals and therefore a point of departure for finding one signal shape within another. Now the exposition tilts toward techniques for breaking an input signal down into a nonsignal form.

Time-domain signal analysis will prove to have some serious limitations. Some types of signals will prove amenable to the time-domain methods. We can engineer signal analysis solutions for many process monitoring problems using time-domain techniques. Other application areas—especially those where signals contain periodic components—will cause trouble for our methods. Speech recognition and vibration analysis are examples. For such applications, the signal level itself is not so important as the regular assumption of some set of values by the signal. If we retreat again to the abstract vantage point of Hilbert space, we will see that our methods must now rely on basis functions for signals that are regular, textured, periodic, sinusoidal, exponential, and so on. Although this discovery qualifies our successes in preliminary signal analysis applications, it leads naturally to the study of the Fourier transforms of analog and discrete signals, and later it leads to the modern mixed-domain techniques: time-frequency and time-scale transforms.

This decomposition into structural features may be passed on to higher-level interpretation algorithms. Machine recognition of digitized speech is an example. So, by this view, signal analysis is the front-line discipline within artificial intelligence. While it makes extensive use of standard signal processing techniques, the analysis of a signal into some interpretable format is a distinct, challenging, and perhaps neglected area of engineering education and practice.

A small diagram, due to Professor Azriel Rosenfeld,[1] who first formulated it for images, illustrates the relationships between data processing, image synthesis (graphics), image processing, and image analysis (computer vision) (Figure 4.1a). Reducing the dimension of the data, we arrive at a similar diagram for signals (Figure 4.1b)

[1]For many years Professor Rosenfeld has used this diagram in his computer vision courses (personal communication). The authors have often heard—and, in the absence of a contrary reference in the scientific literature, do believe—that the diagram is attributable to Azriel Rosenfeld.



(a)

Output \ Input	Data	Image
Data	Data processing	Computer vision
Image	Computer graphics	Image processing

(b)

Output \ Input	Data	Signal
Data	Data processing	Signal analysis
Signal	Signal synthesis	Signal processing

Fig. 4.1. (a) An input–output diagram of the disciplines that work with two-dimensional signals (i.e., images). Conventional data processing began as soon as programmable machines became available. Image processing (for example, enhancement) has been a successful technology since the 1960s. With the advent of parallel processing and fast reduced instruction set computers, computer graphics has reached such a level of maturity that is both appreciated and expected by the public in general (example: morphing effects in cinema). Computer vision stands apart as the one technology of the four that remains problematic. We seem to be decades away from the development of autonomous, intelligent, vision-based machines. (b) An input–output diagram of the computer technologies that use signals. The rediscovery of the fast Fourier transform in the mid-1960s gave digital signal processing a tremendous boost. Computer music is now commonplace, with an example of signal synthesis possible even on desktop computers since the 1980s. Like computer vision, though, signal analysis is still an elusive technology. Where such systems are deployed, they are greatly constrained, lack generality, and adapt poorly, if at all, to variations in their signal diets.

and reveal the input–output relationships between data processing, signal synthesis, signal analysis, and signal processing. Most computing tasks fall into the data processing categories of Figure 4.1, which covers financial and accounting programs, database applications, scientific applications, numerical methods, control programs, and so on. Signal processing occupies the other diagonal square, where signal data are output from a signal data input. The other squares, converting data to signal information and converting signal information to data, generally do not correspond to classes in the university curriculum, except perhaps in some specialized programs or big departments that offer many courses. An application that accepts abstract data inputs and produces a signal output is an example of a signal synthesis system. Courses such as this have been quite rare. Recently, though, some computer science departments are offering multimedia courses that cover some aspects of producing digital music, speech, and sounds. A computer program that accepts a signal and from it generates a data output is in essence a signal analysis system. It is really the form of the data output that is critical, of course. We envision a description—a breakdown of content, or an *analysis*—of the signal that does not resemble the original at all in form.

The great advances in mixed-domain signal transforms over the last 10 years are the primary reason for reworking the conventional approach to signals. These time-frequency and time-scale transforms cast new light on the Fourier transform, expose its limitations, and point to algorithms and techniques that were either impossible or terribly awkward using the standard Fourier tools. Moreover, advances in computer hardware and software make technologies based on this new theory practically realizable. Applications—commercial products, too—are appearing for speech understanding, automatic translation, fingerprint identification, industrial process control, fault detection, vibration analysis, and so on. An unexpected influence on these emerging signal analysis techniques comes from biology. Research into hearing, vision, the brain, psychophysics, and neurology has been supplemented and stimulated by the investigations into the mathematics of the new signal transforms, neural networks, and artificial intelligence. This book will show that a fruitful interaction of artificial intelligence and signal analysis methods is possible. It will introduce students, engineers, and scientists to the fresh, rapidly advancing disciplines of time-frequency and time-scale signal analysis techniques.

In universities, signal processing is taught in both computer science and electrical engineering departments. (Its methods often arise in other disciplines such as mathematics, mechanical engineering, chemical engineering, and physics.) The courses are often almost identical by their course descriptions. There is controversy, though, about whether signal processing even belongs in the computer science curriculum, so similar in content is it to the electrical engineering course. One point of this book is to show that mainstream computer science methods are applicable to signal analysis, especially as concerns the computer implementation of algorithms and the higher-level interpretation methods necessary for complete signal analysis applications.

Signal analysis includes such technologies as speech recognition [1–4], seismic signal analysis [5, 6], interpretation of medical instrumentation outputs [7], fault detection [8], and online handwriting recognition [9]. Computer user interfaces may nowadays comprise all of the Rosenfeld diagram's technologies for both one-dimensional signals and two-dimensional signals (images): speech synthesis and recognition; image generation, processing, and interpretation; artificial intelligence; and conventional user shells and services [10, 11].

It is possible to slant a signal processing course to computer science course themes. Here, for example, we explore time-domain signal analysis and will see that tree structures can be found for signals using certain methods. One can also emphasize signal transforms in computer science signal processing instead of the design of digital filters which is taught with emphasis in the electrical engineering signal processing course. This too we will explore; however, this task we must postpone until we acquire an understanding of signal frequency through the various Fourier transforms. There are filter design packages nowadays that completely automate the design process, anyway. Understanding transforms is, however, essential for going forward into image processing and computer vision. Finally, one also notices that signal processing algorithms—with complex-valued functions, arrays having negative indices, dynamic tree structures, and so forth—can be elegantly implemented using modern computer languages known almost exclusively by students in the university's computer science department.

In the first sections of this chapter, we consider methods for finding basic features of signals: edges and textures. An edge occurs in a signal when its values change significantly in magnitude. The notion of texture seems to defy formal description; at an intuitive level, it can be understood as a pattern or repetition of edges. We develop several techniques for finding edges and textures, compare them, and discover some problems with their application. Edge and texture detection will not be complete in this chapter. Later, having worked out the details of the Fourier, short-time Fourier, and wavelet transforms, we shall return to edge and texture analysis to check whether these frequency-domain and mixed-domain methods shed light on the detection problems we uncover here.

4.1 SEGMENTATION

Segmentation is the process of breaking down a signal into disjoint regions. The union of the individual regions must be the entire domain of the signal. Each signal segment typically obeys some rule, satisfies some property, or has some numerical parameter associated with it, and so it can be distinguished from neighboring segments. In speech recognition, for example, there may be a segmentation step that finds those intervals which contain an utterance and accurately separates them from those that consist of nothing but noise or background sounds.

This section begins with an outline of the formal concept of segmentation. There are many approaches to the segmentation task, and research continues to add new techniques. We shall confine our discussion to three broad areas: methods based on signal levels, techniques for finding various textures within a signal, and region growing and merging strategies. Later sections of this chapter and later chapters in the book will add further to the segmentation drawer of the signal analysis toolbox.

4.1.1 Basic Concepts

Segmentation is a rudimentary type of signal analysis. Informally, segmentation breaks a signal's domain down into connected regions and assigns a type indication to each region. For signals, the regions are intervals: open, half-open, or closed. Perhaps a region is a single point. Some thinking in the design of a signal segmentation method usually goes into deciding how to handle the transition points between regions. Should the algorithm designer consider these boundaries to be separate regions or deem them part of one of the neighboring segments? This is not too difficult a question, and the next section's edge detection methods will prove useful for advanced segmentation strategies. However, for two-dimensional signals (images) the situation is extremely complex. The connected regions can assume quite complicated shapes[2]; and their discovery, description, and graphical representation by machine too often bring current computer vision systems to their practical limits.

Definition (Segmentation). A segmentation $\Sigma = (\Pi, L)$ of a signal f consists of a partition $\Pi = \{S_1, S_2, ...\}$ of Dom(f) into regions and a logical predicate L that applies to subsets of Dom(f). The predicate L identifies each S_i as a maximal region in which f is homogeneous. Precisely, then, segmentation requires the following [12]:

- $\mathrm{Dom}(f) = S_1 \cup S_2 \cup S_3 \cup ...$, where the S_i are disjoint.
- $L(S_i)$ = True for all i.
- $L(S_i \cup S_j)$ = False when S_i and S_j are adjacent in Dom(f) and $i \neq j$.

It is common to call the regions segments, but they are not necessarily intervals. Commonly, the segments are finite in number, only a specific region of interest within the signal domain is subject to segmentation, and some mathematical operation on the signal defines the logical predicate. It is also very much an application-specific predicate. For example, one elementary technique to segment the meaningful parts of a signal from background noise is to threshold the signal. Let $f(n)$ be a noisy signal, $T > 0$, $M = \{n: |f(n)| \geq T\}$, and $N = \{n: |f(n)| < T\}$. Let $\Pi = \{M, N\}$, and let L be the logical predicate "All signal values in this region exceed T or all signal values in this region do not exceed T." Then $S = (\Pi, L)$ is a segmentation of the signal into meaningful signal and background noise regions. Of course, different threshold values produce different segmentations, and it is possible that neither the M nor N is a connected subset of the integers. Most signal analysts, in point of fact, never formulate the logical predicate for segmentation; the predicate is implicit, and it is usually obvious from the computations that define the partition of the signal's domain.

[2]The famous Four-Color Problem is in fact a question of how many types are necessary to lable a map segmented into countries. For every map ever drawn, it had been shown to be possible to color the bounded regions with only four colors. Therefore, conjecture held that four-colorings of the plane were always possible, but for centuries this simple problem defined solution. Only recently, with a grand effort by both humans and computers, has the question been answered in the affirmative.

Definition (Labeling, Features, Pattern). Let $\Pi = \{S_1, S_2, ...\}$ be a partition of the domain of signal f. Then a labeling of f for Π is a map $\Lambda: \Pi \to \{\Lambda_1, \Lambda_2, ...\}$. Ran($\Lambda$) is the set of labels, which categorize subsets of Dom(f) in Π. If $\Lambda(S_i) = \Lambda_j$, then S_i is called a signal feature with label Λ_j. A pattern is set of features in a signal.

Labeling often follows segmentation in signal analysis systems. The nature of the signal analysis application determines the set of labels which apply to signal regions. The segmentation operation uses some logical predicate—which embodies an algorithm or computation—to decompose the signal into homogeneous regions. The regions are distinct from one another and suitable for labeling. Together, the segmentation and labeling operations assign distinct signal regions to predetermined categories, appropriate to the application. This results in a primitive description of the signal's content known as feature detection. The task of finding an assemblage of features in a signal is called pattern detection or pattern recognition.

By associating the regions with a label, the positions and time-domain extents of signal features are known. One problem is that the regions of a general segmentation may be disconnected, in which case the feature is located in multiple parts of the signal. This is a little awkward, but, in general, we shall not demand that our segmentations produce connected sets. But the partition elements do contain intervals, and a maximal interval within a segment does specify the location of a feature. Thus, finding the maximal intervals that have a given label accomplishes the signal analysis task of registration.

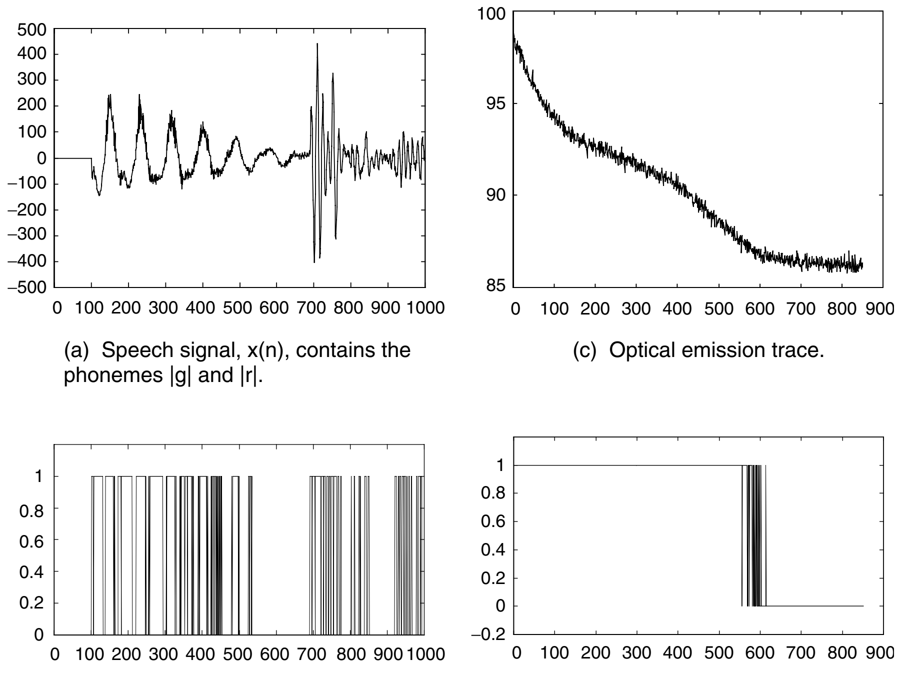
Definition (Registration). Let f be a signal, $\Sigma = (\Pi, L)$ be a segmentation of f, and let $\Lambda: \Pi \to \{\Lambda_1, \Lambda_2, ...\}$ be a labeling of f with respect to Σ. If $\Lambda(S_i) = \Lambda_j$, $I \subseteq S_i$ is an (open, closed, or half-open) interval, and I is contained in no other connected subset of S_i, then I registers the feature with label Λ_j.

Higher-level algorithms may process the labeling and revise it according to rules, prior knowledge of the signal's content, or some model of what form the signal should take. Such high-level algorithms may employ a very small set of rules. But many signal analysis applications use a full rule database or implement a complete expert system for interpreting labeled signals. They often use graph and tree structures. By this point, the design of a signal analysis system is well into the realm of artificial intelligence. Without delving deeply into artificial intelligence issues, Section 4.2.4 discusses some elementary statistical measures and algorithms for refining the segmentation and labeling steps. This is called region splitting and merging. Region merging can be a basis for classification. If the goal is to find a sufficiently large signal region or a region with a particular registration, then a region merging procedure applied to the results of a first-cut segmentation might be adequate. Later, in Section 4.7, we will introduce what are called consistent labeling methods for revising the region label assignments. This strategy uses constraints on both the regions and the labels applied to them. The constraints on signal features can be formulated as a directed graph, a familiar data structure from artificial intelligence. When a signal's features can be labeled so as to obey the constraints, which might be derived from a signal prototype or model, then the signal has been classified.

4.1.2 Examples

Some examples of segmentation, labeling, and registration should help to make these abstract concepts more concrete.

Example (Signal versus Noise Detection). A logical predicate, N, for "either noise or signal," may apply to subsets of the domain of a signal, f. (To be precise, we are using the exclusive "or" here: noise or signal, but not containing both characteristics.) A region $R_n \subseteq \mathrm{Dom}(f)$ for which $N(R_n) = \mathrm{True}$ may be a region of noise, and another region $S_n \subseteq \mathrm{Dom}(f)$ for which $N(S_n) = \mathrm{True}$ may be meaningful information. Many applications involve signals whose noise component are of a much lower magnitude than their information-bearing component. (This is not always the case; signals may follow an active-low policy, and the choice of labels for low- and high-magnitude segments must reflect this.) Reasonably, then, we might apply a threshold to the signal for distinguishing signal from noise. Figure 4.2 shows examples of



(a) Speech signal, x(n), contains the phonemes |g| and |r|.

(c) Optical emission trace.

(b) Threshold result, |x(n)| > 50.

(d)

Fig. 4.2. Signal segmentation in speech recognition and industrial process control. Detecting a speaker's presence by thresholding works when the background noise level is sufficiently low (panels (a) and (b)). But background noises or artifacts of speech (tongue clicks, lip smacks, sniffs, sighs, loud breaths) will often be assigned to a speech region. Panels (c) and (d) show a representative optical trace from a plasma etching reactor. The controller monitors a carbon monoxide optical emission signal, whose high level indicates the presence of etching byproducts in the chamber.

signal segmentation based on thresholding. The goal is to separate the signal into a meaningful signal region, which has a high magnitude, and a background noise region, which has a relatively low magnitude. If the noise magnitude remains modest and the meaninful signal regions do not fade, then one-dimensional tasks like this are quite straightforward. Segmenting images into high-magnitude objects and low-magnitude background is called blob detection, an erstwhile colloquialism that has become standard technical parlance over the years. In two dimensions, because of the difficulties in tracking boundaries, blob detection is often problematic. The equivalent one-dimensional task—bump rather than blob detection—is much easier of course, because the topology of the line is simpler than the topology of the plane.

Application (Touch-Tone Telephone Pulse Detection). This example continues our discussion of digital telephony. Touch dialing telephones have largely supplanted the old rotary dialing units in the public service telephone network. Instead of a series of electromechanically generated pulses, modern telephones generate dual-tone multifrequency (DTMF) pulses for dialing [13]. The telephone company's central office (CO) equipment decodes these pulses; sets up the appropriate communication links; rings the far telephone; and sends a ringback signal, which simulates the sound a distant phone ring, to the near-end telephone. When someone lifts the far-end handset, the CO switching equipment sends a brief test tone through the circuit as a continuity check. The CO listens for facsimile (FAX) equipment tones and then for modem tones; in their absence, it initiates echo cancellation (see Chapter 2), and the call is complete. Table 4.1 shows the tone dialing scheme. Figure 4.3 illustrates a dual-tone sinusoidal pulse in noise and a sample segmentation. This segmentation produces two sets; however, due to the sinusoidal components in the pulse, there are numerous low-magnitude signal values interspersed in the time interval where we expect to detect the pulse. One strategy to improve the segmentation is to integrate the signal, and another strategy is to adjust the labeling to eliminate noise segments that are too brief.

Early telephones had two twin-tee feedback oscillators for producing DTMF signals. Later, and cheaper, designs were digital. Nowadays, digital signal processors

TABLE 4.1. DTMF Telephone Dialing Signal Specifications[a]

	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

[a]DTMF pulses consist of two tones from a high- and a low-frequency group. Telephones in public use do not sport the special keys on the right: A, B, C, D. The telephone company reserves them for test equipment and diagnostic signaling applications. DTMF tones pass at a 10-Hz rate. Pulse width is between 45 ms and 55 ms.

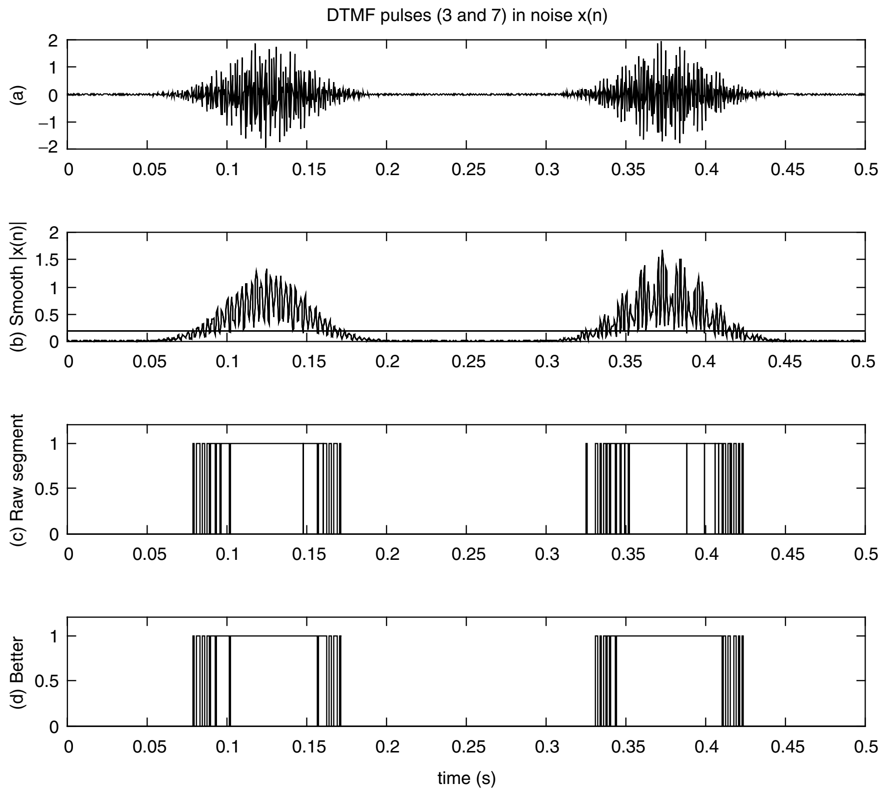


Fig. 4.3. An example of dual-tone multi-frequency (DTMF) pulse detection for telephone company dialing signals. Each number on the telephone dial is coded as a dual frequency pulse (a). Simple threshold (b) and labeling operations (c) mark tones (tone present = 1, background noise = 0). This naive, bottom-up procedure shows that no pulse segment is sufficiently long in duration to be a valid DTMF pulse; hence it finds no pulses present. The improved scheme (d) constrains the labeling operation so that short-duration noise segments are merged with signal regions, thus properly finding two 50-ms-wide pulses. It is possible—and more conventional for that matter—to filter the pulse (in this case its magnitude) for more consistent thresholding rather than rely on consistent labeling.

can be programmed to generate and decode DTMF signals [14, 15]. It is important to note that this example only considers the relatively easy problem of finding a blob-like pulse of the proper width for a DTMF pulse. A complete, practical solution must find the frequencies within the signal blob, verify that only two significant tones exist, and check that the pair of tones corresponds to a cell in Table 4.1. We will consider this more complex task in Chapter 7, after we develop methods for signal frequency analysis in Chapters 5 and 6.

4.1.3 Classification

Following segmentation and labeling, signal analysis may take a further step—classification of the entire signal. Classification is also called recognition. It is much like labeling, except that the term "labeling" in signal and image analysis parlance tends to mean a tentative classification. Labels are applied to small fragments of the signal, meant for later review, and facilitate the application of goal-driven rules. In contrast, the recognition step applies a broader signal region, which comprises several labeled, goal-directed fragments.

Definition (Classification). Let $F = \{f_1, f_2, ...\}$ be a family of signals. Then a classifier for F is a map C: $F \to \{C_1, C_2, ...\}$. The range of C is the set of classes, which categorize signals in F.

In view of our imprecise distinction between labeling and classifying, and with regard to the overlap of usage within the signal analysis community, we need to be casual about this definition. An application may break a long-term signal down into large regions and attempt to identify each chunk. While analyzing each large region, the remainder of the signal can be considered to be zero, and it is quite irrelevant about whether we deem the remaining chunks to be separate signals or just extended pieces of the original. The distinction, therefore, between labeling and classification revolves around which is the preliminary step and which is the ultimate step in the signal analysis.

The concepts of segmentation, labeling, and classification imply a scheme for constructing signal analysis systems (Figure 4.4). Most signal analysis systems work in the order: segmentation, labeling, classification. This strategy proceeds from low-level signal data, through assignment of labels to signal regions, to finally classify the entire signal. Therefore, this is a data-driven or bottom-up methodology. To impart a goal-driven or top-down aspect to the procedure, it is possible to constrain the labeling procedure. In this case, a high-level, rule-based algorithm reviews the initial labels and adjusts their assignments to signal features. The labeled regions may be too small in extent, as the narrow low-magnitude regions in the example of Fig. 4.3, and hence they are merged into the surrounding meaningful signal region.

Example (Voiced versus Unvoiced Speech Segmentation). Linguists typically classify speech events according to whether the vocal cords vibrate during the pronunciation of a speech sound, called a phone. Phones are speech fragments that represent the basic, abstract components of a natural language, called phonemes (Table 4.2). If the vocal cords do vibrate, then there is said to be a voiced speech event. If there is no vocal cord vibration, then the phoneme is unvoiced. It is also possible that a speech signal contains no speech sound; thus, it is simply background, or noise. One approach to segmenting speech classifies its portions as voiced (V), unvoiced (U), or noise (N). For example, a digital recording of the English phrase "linear fit" begins, divides the two words, and ends with noise

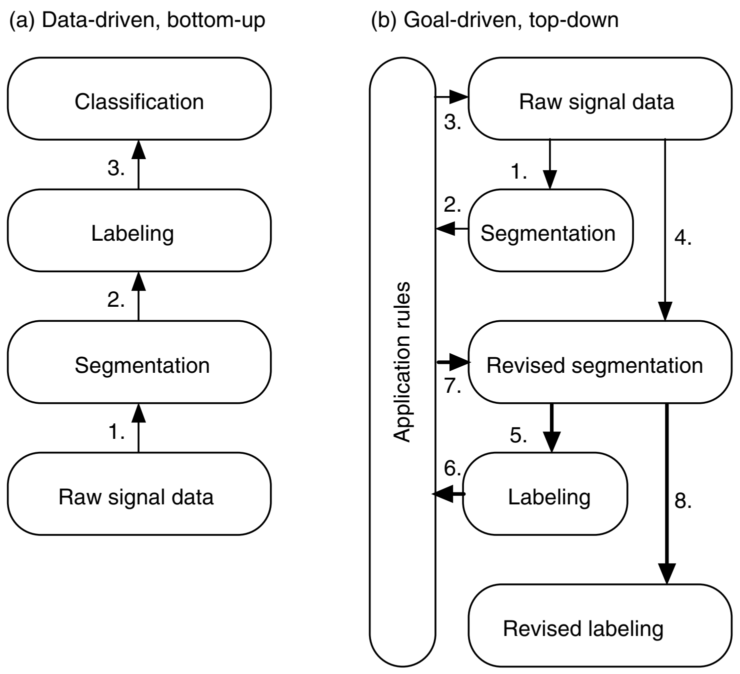


Fig. 4.4. Data-driven versus goal-driven signal analysis systems. In the data-driven, bottom-up scheme of (a), the processing proceeds from segmentation, to labeling, to classification without looking back. The goal-driven, top-down strategy in (b) revises earlier results according to rules specific to the application domain.

regions. The most apparent phonemes, then, are /l I n i э f I t/, of which /f/ and /t/ are unvoiced. A preliminary segmentation, Σ_0, by voiced/unvoiced/noise classification is (N, V, N, U, V, U, N), respecting the stipulation that no adjacent regions have the same type. Actually, there are a number of unvoiced events that only become apparent when the speech signal is digitized and spread out over time. One may find, for example, momentary unvoiced aspirations and even periods of noise, surrounding the voiced segments. A refined segmentation, Σ_1, therefore supplies: (N, U, V, U, V, U, V, U, V, U, V, U, N, U, V, U, N). In practical speech recognition applications, surprisingly, this segmentation is too crude! Subtle periods of background noise, with no voiced or unvoiced sound present, intrude into spoken words. A modern, sophisticated segmentation algorithm finds that several *N* regions split the unvoiced regions U in the refined segmentation above. This means that several of the unvoiced intervals have much shorter time extents than Σ_1 would indicate. The benefit is that a higher-level interpretation algorithm may be better able to recognize the brief U boundaries of the V segments as trailing and leading aspirations instead

TABLE 4.2. Phonemes and Word Examples from American English [16][a]

Phoneme	Example	Class	Phoneme	Example	Class
/i/	even	Front vowel	/I/	signal	Front vowel
/e/	basis	Front vowel	/ɛ/	met	Front vowel
/ae/	at	Front vowel	/a/	father	Mid vowel
/Λ/	but	Mid vowel	/⊃/	all	Mid vowel
/schwa/	signal	Mid vowel	/u/	boot	Back vowel
/o/	boat	Back vowel	/U/	foot	Back vowel
/Ɨ/	roses	Back vowel	/ɜ/	Hilbert	Mid vowel
/a^{w}/	down	Dipthong	/a^{y}/	cry	Dipthong
/$\supset^{y}$/	boy	Dipthong	/y/	yet	Semivowel glide
/w/	wit	Semivowel liquid	/r/	rent	Semivowel glide
/l/	linear	Semivowel liquid	/m/	segment	Nasal consonant
/n/	nose	Nasal consonant	/η/	Nguyen	Nasal consonant
/p/	partition	Unvoiced stop	/t/	fit	Unvoiced stop
/k/	kitten	Unvoiced stop	/b/	bet	Voiced stop
/d/	dog	Voiced stop	/g/	gain	Voiced stop
/h/	help	Aspiration	/f/	fit	Unvoiced fricative
/θ/	thanks	Unvoiced fricative	/s/	sample	Unvoiced fricative
/sh/	shape	Unvoiced fricative	/v/	vector	Voiced fricative
/∂/	that	Voiced fricative	/z/	zoo	Voiced fricative
/zh/	closure	Voiced fricative	/ch/	channel	Affricate
/j/	Jim	Affricate	/ʔ/	no sound	Glottal stop

[a]Vowels are voiced, and they break down further into front, mid, and back classifications, according to the location of the tongue's hump in the mouth. Other essentially vowel-like sounds are the dipthongs and the semivowels. Nasal consonants are voiced, and the affricates are unvoiced. The glottal stop is a special symbol that indicates a momentary suspension of motion by the vocal cords (glottis). For example, without the /ʔ/ symbol, the phoneme sequences for "I scream" and "ice cream" are identical. As big as it is, this table is far from panoramic; it offers but a glimpse of the natural language segmentation problem.

of, for example, unvoiced fricatives. Figure 4.5 illustrates such a speech segmentation example. We shall continue to expose the intricacies of natural language interpretation is this and the remaining chapters; there is much to cover.

Many problems involve a combination of segmentation procedures. For example, in digital telephony, it may be necessary to distinguish voice from DTMF pulses. Many commercial telephony applications that rely on DTMF detection require this capability. A pervasive—some would call it pernicious—application is the office voice mail system. The user presses telephone buttons to control the selection, playback, archival, and deletion of recorded messages. The voice mail application detects the DTMF and performs the appropriate function. Background office noise or talking confuses the DTMF classifier. Thus, a sophisticated DTMF detector sorts out the many possible natural language sounds from the dual-tone signaling pulses. Vowel sounds, such as /i/ and /u/, typically contain two sinusoidal components; and unvoiced fricatives, such as /s/ and /f/, are hard to discern from background

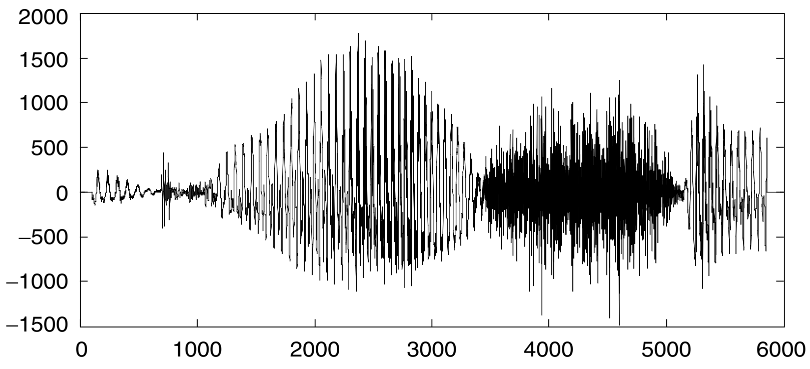


Fig. 4.5. A speech segmentation example. At a sampling rate of 16 kHz, the word "greasy" is shown. The corresponding phonemes are /g/, /r/, /i/, /s/, and /i/; of these /s/ is voiceless.

telephone line noise. Application success revolves around a careful design of segmentation, labeling, and classification algorithms. Sometimes a data-driven computational flow works, even for distinguishing DTMF signals from human voices. Where the bottom-up techniques break down, top-down constraints and goal-directed, artificial intelligence strategies become attractive.

4.1.4 Region Merging and Splitting

Once the domain of a signal f is broken down by a segmentation into a partition $\Pi = \{S_1, S_2, ...\}$, it may be necessary to combine some of the regions together. This reduction in the number of separate regions of the domain is called *region merging* (Figure 4.6). Region merging generally follows labeling. It is invoked because some goal-directed rules have discovered flaws in the labeling. We already considered this situation in the realm of speech analysis. A digitized speech fragment might receive an initial phoneme label, because of its high magnitude. However, subsequent analysis of its time-domain extent, and possibly its position at the beginning or ending of what appears to be a full-word utterance, might relegate the fragment to the category of a tongue click. Merging can also be useful when the signal segmentation must be made as simple as possible—for instance to facilitate signal compression or speed up subsequent analysis steps.

The choice of criteria for region merging depends heavily on the signal analysis application. For example, suppose that two segmentation regions have average signal values that are so close that they seem to be caused by the same physical process. Let μ_R be the mean within region R, and let be μ_S the mean within region S. If $|\mu_R - \mu_S| < \varepsilon$, where ε is an application-specific threshold, then we replace R and S in Π by $R' = R \cup S$. Merging continues by comparing the mean of R' against the mean of the remaining regions. Another statistic useful for merging two adjacent regions, especially when the segmentation is based on measures of signal texture (Section 4.3), is

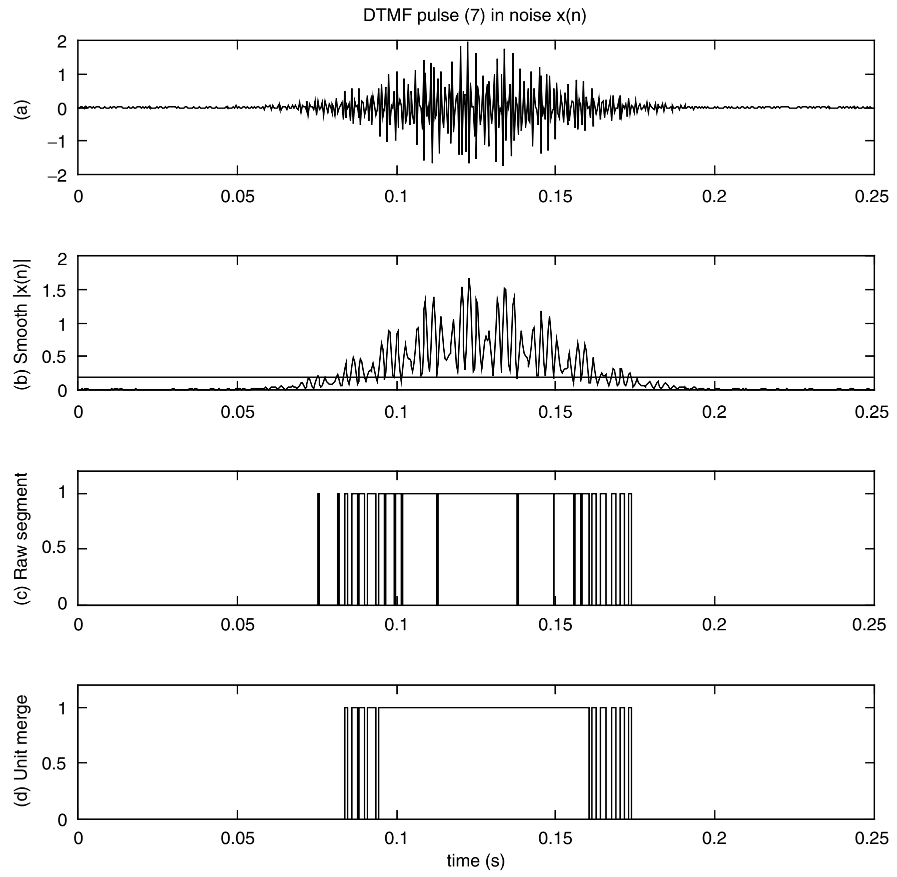


Fig. 4.6. Region merging: DTMF pulse (a), threshold operation (b), and raw labeling (c). The marked pulse segments are too small for a valid tone signal. Simply merging unit width segmented regions into their larger, differently marked neighbors isolates a valid DTMF pulse of 50-ms contiguous length (d).

a similarity in their variances. An edge detector could test the boundary between regions. If the edge detector fails to show a sufficiently distinct edge between the two areas, then they are candidates for merging. These are all low-level considerations for merging regions. If the application is so well characterized as to have a set of goal-directed rules that govern the labeling of signal regions, then the rules may be applied to the preliminary segmentation results. Regions can be joined, therefore, to better satisfy the top-down specifications. Finally, note that the results of the merging operations will supersede the original logical predicate that defined the segmentation.

Splitting a region is harder than merging. When the regions already exist, then the main problem is choosing the right criteria, out of many possibilities, for deciding whether to merge them. When a region is examined as a candidate for splitting, however, what partition of it begins the task? One possibility is to divide a region of interest, $S = [a, b] \subseteq \mathrm{Dom}(f)$, into segments of equal length, $S = S_1 \cup S_2 \cup S_3 \cup \cdots \cup S_N$, where the S_i are disjoint. A good choice for segment length is the smallest time interval for meaningful features of signal f. Then a statistical measure, the mean or variance for example, applied to each segment in succession, can indicate whether the segment should be set apart from the rest. With the original S now subdivided, we can consider whether to unite the various S_i, just as in the above region merging algorithm. Splitting is then a kind of inverse of merging. The end result is to split the original region $S = R_1 \cup R_2 \cup \cdots \cup R_M$, where $M \le N$ and each R_i is a union of a different subset of $\{S_1, S_2, \ldots, S_N\}$. The obvious problem with this approach is that it directly depends upon the arbitrary partition of S into the segments, $S_1, S_2, \ldots, S_N$.

But there is a deeper problem with both the region merging and the "inverse" region splitting algorithms above. They both depend on the order in which the regions are selected for merging or splitting. This is due to the lack of transitivity in the "is similar to" relations we apply. Let's explore this. If we begin with the first segment, we may have $|\mu_1 - \mu_2| < \varepsilon$, and so we unite them, $R_1 = S_1 \cup S_2$. Now, the mean of R_1 is the statistical expectation, $E[\{f(n) \mid n \in R_1\}] = E[f(R_1)]$, and we compare it to μ_3. But it is quite possible that the criterion for merging S_3 into R_1 would be met, while at the same time we have $|\mu_1 - \mu_3| \ge \varepsilon$ and $|\mu_2 - \mu_3| \ge \varepsilon$. The merging of the subregions (and thus the splitting of the original region of interest) depends on the order in which the subregions are considered. To avoid this, we can try to cluster the subregions around some entity that does not depend on an individual region. Suppose we select M values $k_1 < k_2 < k_3 < \cdots < k_M$, where $k_i \in \mathrm{Ran}(f)$. (Perhaps we specify $k_1 = \min\{f(n) \mid n \in [a, b]\}$, and $k_M = \max\{f(n) \mid n \in [a, b]\}$, but this is not essential.) Then we define the subregion S_i to belong to cluster j if $|\mu_i - k_j| \le |\mu_i - k_m|$, for all m, $1 \le m \le M$. That is, the clusters comprise those subregions whose mean falls closest to a target signal value among $\{k_1, k_2, \ldots, k_M\}$. We take $R_j = \cup\{S_i \mid S_i \text{ belongs to cluster } j\}$.

The above approach is called *nearest-neighbor clustering*. The same method can be followed using the variance as a clustering measure. The drawback for this elementary approach is its dependence upon the target values. Another possibility is to discover the target values. We consider all possible assignments of the S_i to clusters indicated by labels $\Lambda_1, \Lambda_2, \ldots, \Lambda_M$. For each permutation, we let $R_j = \cup\{S_i \mid S_i \text{ is assigned to cluster } j\}$ and $X_j = \{\mu_i \mid \mu_i = E[f(S_i)] \text{ and } S_i \text{ is assigned to } \Lambda_j\}$. A cluster is made up of similar subregions if the average values in each X_j are close to one another. Let the target values be $k_j = E[X_j]$. Thus, if we set $\mathrm{Var}[X_j] = E[(X_j - E[X_j])^2] = E[(X_j - k_j)^2]$, then a reasonable measure of the inhomogeneity of our cluster assignment is $\mathrm{Var}[X_1] + \mathrm{Var}[X_2] + \cdots + \mathrm{Var}[X_M]$. An assignment that minimizes this measure is an optimal (i.e., minimal variance) clustering for M labels.

4.2 THRESHOLDING

The threshold system is a nonlinear, but time-invariant operation on signals. In signal analysis applications, thresholding is used to separate the signal domain into regions according to the value that the signal assumes. Thresholding systems are a principal tool for signal segmentation. Thresholding can be applied globally or locally.

4.2.1 Global Methods

For purposes of segmentation, one of the easiest techniques is to select a single threshold value and apply it to the entire signal—global thresholding. For example, for discrete signals, $f(n)$, if $T > 0$, then $M = \{n: |f(n)| \geq T\}$ is the meaningful component, and $N = \{n: |f(n)| < T\}$ is the noise component of $f(n)$, respectively. Of course, with this division of the signal, the meaningful component will have some noise present. These jagged, low-magnitude artifacts are sometimes called "fuzz," "clutter," or "grass," and they are unavoidable for real-world signals. More precisely, the thresholding operation just picks out that part of the signal that contains something more—something relevant to the analysis application—than just the background noise.

The examples in the previous section showed that this technique works for signal features that are sufficiently distinct from the background noise. Difficulties arise, however, when the signal features of interest diminish over time, blend into the noise, or contain oscillatory components. Sometimes supplemental signal filtering helps. Sometimes a more sophisticated labeling procedure, which imposes constraints on the segmentation, corrects any flaws in the preliminary partitioning of the signal's domain. In any case, the main problem is to determine the appropriate threshold.

4.2.2 Histograms

There are several methods for finding global thresholds. Simple inspection of source signals may indeed suffice. Another method is to histogram the signal, producing a density map of the signal values. Histograms count the number of values that fall within selected ranges, or bins, over the signal domain. This is akin to segmentation, but it partitions the signal's range rather than its domain. The histogram is also an approximation of the signal level probability density function. This observation is the key idea in several of the optimal threshold selection techniques in the signal analysis literature.

Definition (Histogram). Let f be a discrete signal and let $\Pi = \{B_1, B_2, ...\}$ be a partition of the range of f. Then the B_k are *bins* for $\mathrm{Ran}(f)$. The histogram of f with respect to Π is defined by $h(k) = \#(f^{-1}(B_k))$. In other words, the value $h(k)$ is the cardinality of $f^{-1}(B_k)$—the number of $n \in \mathrm{Dom}(f)$ with $f(n) \in B_k$.

Even though the definition confines the histogram idea to discrete signals, it is still a very general formulation. It allows for a countably infinite number of bins as well as for infinite histogram values. It is more practical to restrict the domain of discrete signals $f(n)$ to an interval and specify a finite number of bins. That is, suppose $f(n)$ is a digital signal with q-bit values and that f(n) is of finite support (i.e., $f(n) = 0$ outside some finite interval, I). Then $0 \le f(n) \le 2^q - 1$. Let $\Pi = \{\{k\} \mid 0 \le k \le 2^q - 1\}$. Then Π is a set of bins for Ran(f), and $h(k) = \#(\{n \in I \mid f(n) = k\}) = \#(f^{-1}(\{k\}))$ is a histogram for f. Since h counts domain elements only within I, $h(k)$ is a discrete signal with support in $[0, 2^q - 1]$.

Example (Digital Image Histograms). Many one-dimensional signal processing and analysis tasks arise in computer vision. Histograms, in particular, are fundamental to early image analysis tasks. Figure 4.7 shows a gray-scale text image and its histogram. Threshold selection is easy. But this is due to the close cropping of white background from the black text. Including larger areas of background in the image would result in an apparently unimodal histogram. A lesson in text image analysis is that successful segmentation of words into letters depends on a prior successful segmentation of the page into words or lines.

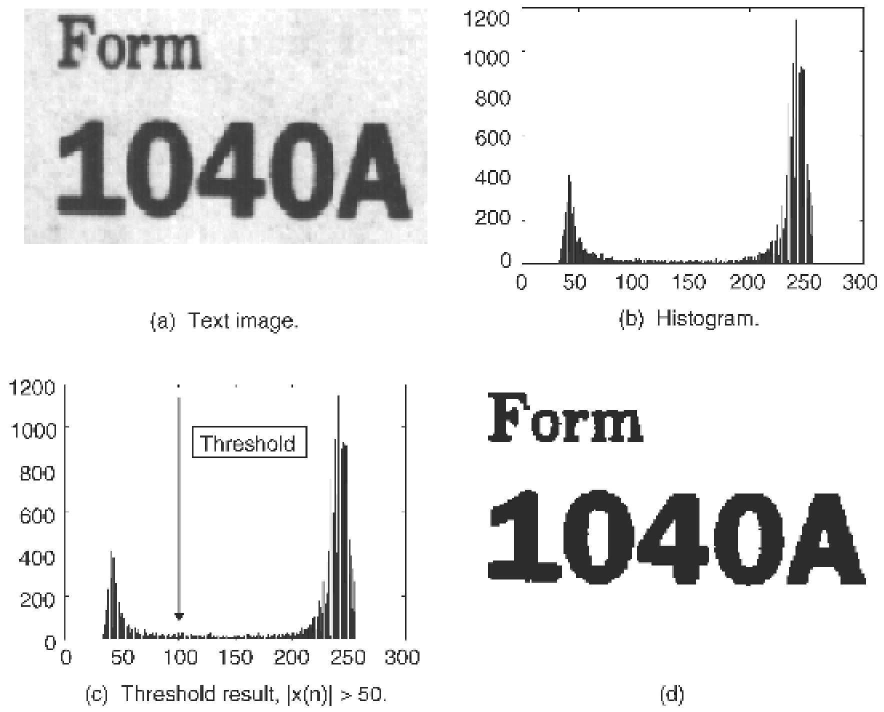


Fig. 4.7. A text image familiar to readers working in the United States. The gray-scale text image (a) has been cropped to exclude surrounding background areas. In the histogram (b), the modes correspond to the largely white and largely black picture elements. Inspection of the histogram reveals a threshold (c), and using this threshold segments the text into purely black letters and purely white background (d).

It is important, however, to automate the histogram threshold selection procedure for autonomous signal analysis systems. A valley-finding algorthm is sometimes effective. If the significant and background signal values are both large in number and different in magnitude, then two large modes appear in the histogram. Beginning from the histogram midpoint, a reverse-gradient search across the histogram should find the valley between the modes. Let $h(k)$ be the signal histogram, $T \in \mathrm{Dom}(h)$, and search to another k as follows:

- If $T-1 \in \mathrm{Dom}(h)$ and $h(T-1) \le h(T)$, then continue the search at $T-1$; otherwise.
- If $T+1 \in \mathrm{Dom}(h)$ and $h(T+1) \le h(T)$, then continue the search at $T+1$; otherwise.
- Accept the value of T last searched as the threshold.

There are several drawbacks to this procedure, and they are common to every type of reverse-gradient technique. The algorithm stops at the first local minimum it finds. The simple valley-finding technique has a direction preference, which may not accord well with the signal analysis application. And there is no guarantee that the final T the method selects does not lie in a local minimum interior to one of the histogram's modes. The next technique offers to improve the valley-finding algorithm by building a global cost function.

If we think of the histogram as a valley, possibly containing foothills, between two mountain ranges, then selecting a threshold amounts to deciding how much effort it takes to get over the mountains to one boundary or the other. At each point on the histogram, we associate a cost of leaving the valley; it is the minimum of the cost of reaching the first and the cost of reaching the last histogram bin. Of course, only uphill walking requires effort, so we increment the separate costs to the far bins only when the histogram is decreasing on the left and increasing on the right. A point with maximal cost lies in the valley and is a feasible threshold for separating the two modes. Less metaphorically, therefore, an algorithm for finding a threshold in a histogram $h(k)$ is as follows:

$$\mathrm{Cost}_L(t) = \sum_{\substack{k \le t \\ h(k-1)>h(k)}} [h(k-1) - h(k)], \tag{4.1}$$

$$\mathrm{Cost}_H(t) = \sum_{\substack{k \ge t \\ h(k)<h(k+1)}} [h(k+1) - h(k)], \tag{4.2}$$

$$\mathrm{Cost}(t) = \min\{\mathrm{Cost}_L(t), \mathrm{Cost}_H(t)\}, \tag{4.3}$$

$$T = \max_t \left\{\mathrm{Cost}(t)\right\} \tag{4.4}$$

We call the above technique a global valley-finding algorithm.

Techniques such as the above algorithm (4.1)–(4.4) furnish useful thresholds for many signal segmentation tasks. However, as the exercises show, it easy to find signals for which the algorithm fails. Furthermore, natural signals or those from engineered systems may have meaningful regions that become lost when the histogram procedure removes their time domain locality. The question therefore arises whether there is some way of inproving the histogram estimate, or, more optimistically, whether there is an optimal method for finding a threshold for segmentation of an image from its histogram.

4.2.3 Optimal Thresholding

In fact, there are several optimal methods for finding thresholds from signal histograms. Each such "optimal" method discovers the best threshold based on some particular assumption about the histogram's statistics. This is a natural thought, since the histogram approximates the probability density function of the signal values. Accordingly, let us consider some techniques for optimal threshold selection. We shall begin with work that is now classic and end with some quite recent improvements in this thread of investigation.

4.2.3.1 Parametric Approaches. Suppose that we know the some of the statistical behavior of signals that arrive as inputs to an analysis application. For example, we might have knowledge of the statistical distribution of signal values, along with knowledge of the likelihoods that signal features have certain labels; and, perhaps the most valuable information of all, we might even know the probabilities for the classification of signals. Such information on statistical parameters associated with a signal generation mechanism is the basis for the parametric approach to signal threshold determination.

Suppose that we know the probabilities that a discrete signal value is high-magnitude (meaningful), P_H, or low-magnitude (background, noise), P_L. These are called the a priori probabilities of meaningful and noise components, respectively. For example, in a digital telephony system, DTMF pulses might occur at 10 Hz with an average pulse width of 48 ms. Hence we assume a priori probabilities of $P_H = .48$ and $P_L = .52$. Some system designs rely on preliminary statistical studies to develop a priori probabilites. Other strategies are adaptive, and the probabilities change slowly while the system operates.

Segmentation errors are due to labeling a noise value as meaningful or a meaningful value as noise. Thus, if a threshold T for a signal histogram produces errors with probability $E(T)$, then

$$E(T) = P_H E_L(T) + P_L E_H(T), \tag{4.5}$$

where $E_L(T)$ and $E_H(T)$ are the probabilities of incorrectly labeling a signal value as noise and as meaningful, respectively. To find the minimum labeling error, we differentiate (4.5) and solve the equation $dE/dT = 0$ for the threshold T:

$$\frac{dE}{dT} = P_H \frac{dE_L}{dT} + P_L \frac{dE_H}{dT} = 0. \tag{4.6}$$

Now, this scheme does not work at all unless we can find estimates for $E_L(T)$ and $E_H(T)$. The idea is to approximate the distributions of signal values in the histograms by standard statistical distributions. In the histogram of Figure 4.7, for instance, the modes resemble normal (Gaussian) density functions. From the tutorial on probability theory in Section 1.8, the Central Limit Theorem shows that whatever the distributions we observe in a histogram, then (given their bounded variance) the average of a great many of these random variables always approaches a Gaussian distribution. Let us assume, therefore, Gaussian distributions of both noise and meaningful signal. Thus,

$$q_L(t) = \frac{1}{\sqrt{2\pi}\sigma_L} e^{-(t-\mu_L)^2/(2\sigma_L^2)} \tag{4.7}$$

is the probability density function for the signal noise values, where μ_L and σ_L are the mean and standard deviation, respectively. Similarly,

$$q_H(t) = \frac{1}{\sqrt{2\pi}\sigma_H} e^{-(t-\mu_H)^2/(2\sigma_H^2)}, \tag{4.8}$$

where μ_H and σ_H are the mean and standard deviation of the signal noise values, respectively. Since noise values are on average less than meaningful signal values, we know that $\mu_L < \mu_H$. From (4.7) and (4.8), it follows that

$$E_L(T) = \int_{-\infty}^{T} q_H(t)\, dt, \tag{4.9}$$

and

$$E_H(T) = \int_{T}^{\infty} q_L(t)\, dt. \tag{4.10}$$

Differentiating (4.9) and (4.10) with respect to T shows that $dE_L/dT = q_H(T)$ and $dE_H/dT = -q_L(T)$. We substitute expressions for these derivatives—(4.7) and (4.8)—into (4.6) to obtain

$$\frac{P_H}{\sigma_H} e^{-(T-\mu_H)^2/(2\sigma_H^2)} = \frac{P_L}{\sigma_L} e^{-(T-\mu_L)^2/(2\sigma_L^2)}. \tag{4.11}$$

We take the natural logarithm on both sides of (4.11), simplify, and reveal a quadratic equation in T:

$$\left(\sigma_L^2 - \sigma_H^2\right)T^2 + 2\left(\sigma_H^2\mu_L - \sigma_L^2\mu_H\right)T + \left(\sigma_L^2\mu_H^2 - \sigma_H^2\mu_L^2\right) - 2\ln\left(\frac{P_H\sigma_L}{P_L\sigma_H}\right) = 0. \tag{4.12}$$

The quadratic equation (4.12) may have two, one, or zero solutions, depending upon the statistics of the true signal and its noise component. It may be necessary to compare the performance of two possible thresholds using (4.5). And any solution T for (4.12) must be in the range of the signal in question. The exercises further explore these ideas.

This procedure is due to Chow and Kaneko [17]. They applied the technique for finding the left ventricle cardioangiograms—x-ray images acquired after injecting a contrast-producing dye into the heart. Some observations on the method are:

- The method requires *a priori* knowledge of the signal, namely the probabilities of the true signal, P_H, and of the background, P_L.
- It is a parametric method, in that it assumes a particular model of the signal histogram and then derives parameters that best describe the model.
- To discover the parameters of the sum of Gaussian distributions in Chow and Kaneko's approach, it is necessary to fit the model to the actual histogram data.
- Thus, parameter determination requires, for example, a least-squares fit of the model to the data, and an accurate or numerically well-behaved convergence is not guaranteed.
- Moreover, as our own examples show, the model (e.g., a sum of two normal distributions) may not be appropriate for the signal histogram.

These are difficulties with any parametric technique. The next section considers a nonparametric strategy. It is an alternative that does not presuppose statistical distributions for the signal values. Then, in Section 4.2.3.3 we will revisit parametric methods. There we will examine a method inspired by information theory that avoids the assumption of *a priori* probabilities.

4.2.3.2 Nonparametric Approaches. A nonparametric approach to threshold determination assumes no knowledge of the statistical parameters that derive from a signal's values. Thus, nonparametric strategies include the valley finding tactics covered earlier. Valley finding methods do not assume any statistical distribution of meaningful signal and noise values, even though these particular algorithms are rather primitive. Methods can combine, too, for better performance. We can use a global valley-finding algorithm to split the histogram with a preliminary threshold, T. We then determine the statistical parameters of the noise and true signal by separate least-squares fits to the histogram for $t < T$ and for $t > T$. And, finally, we apply the Chow and Kaneko technique to improve the estimate of T. If the segmentation that results from the threshold selection is unsatisfactory, goal-directed strategies are worth investigating. It may be possible to alter the preliminary threshold, recompute the mode statistics, or choose another statistical distribution for modeling the histogram.

Let us turn to a nonparametric approach for threshold determination and signal segmentation proposed by Otsu [18]. Otsu hoped to avoid some of the difficulties we noted above. The idea is to select a threshold to segment the signal into labeled regions of minimal variance in signal levels. Let $P_L(t)$ and $P_H(t)$ be the probability of background values and true signal values, respectively. In Chow and Kaneko's approach, these were needed *a priori*; in Otsu's algorithm, on the other hand, these are approximated from the histogram and are functions of the threshold value. We segment the signal values into two groups, according to the threshold. And let us suppose, again without loss of generality, that background values are low and meaningful signal values are high. Then, a measure of within group variance is

$$\sigma_w^2(t) = P_L(t)\sigma_L^2(t) + P_H(t)\sigma_H^2(t), \tag{4.13}$$

where $\sigma_L(t)$ and $\sigma_H(t)$ are the standard deviations of the noise and the meaningful signal, respectively. In order to find t so that (4.13) is minimized, we must also find the statistical distribuitons of the low-level and the high-level regions of the signal. Chow and Kaneko's parametric approach assumes that the histogram is a sum of two normal densities and that *a priori* probabilities are known.

Because it directly approximates the histogram statistics from threshold values and does not make assumptions about *a priori* noise and true signal probabilities, Otsu's method is essentially unsupervised. Let us see how the method works. Suppose that $\mathrm{Ran}(f) \subseteq [0, N-1]$, set $S_k = \{k\}$ for $0 \le k < N$, and suppose that $\mathrm{Dom}(f)$ is finite. Then $\{Sk \mid \}$ is a partition of $\mathrm{Ran}(f)$. Define

$$p_k = \frac{\#(f^{-1}(S_k))}{\#\mathrm{Dom}(f)}, \tag{4.14}$$

Then, p_k is a discrete probability density function for f. Hence,

$$P_L(t) = \sum_{k=0}^{t-1} p_k, \tag{4.15}$$

and

$$P_H(t) = \sum_{k=t}^{N-1} p_k. \tag{4.16}$$

Let $\{\Lambda_L, \Lambda_H\}$ be labels for the noise and meaningful regions of the signal. Then, the conditional probability that $f(n) = k$, given that n has label Λ_L and the threshold is t, is $P(k \mid \Lambda_L) = p_k/P_L(t)$. Similarly, $P(k \mid \Lambda_H) = p_k/P_H(t)$. This observation permits us to write down the following values for the parameters of the distributions comprising the histogram:

$$\mu_L(t) = \sum_{k=0}^{t-1} kP(k \mid \Lambda_L) = \sum_{k=0}^{t-1} k\frac{p_k}{P_L(t)}, \tag{4.17}$$

$$\mu_H(t) = \sum_{k=t}^{N-1} kP(k \mid \Lambda_H) = \sum_{k=t}^{N-1} k\frac{p_k}{P_H(t)}, \tag{4.18}$$

$$\sigma_L^2(t) = \sum_{k=0}^{t-1} (k-\mu_L(t))^2 P(k \mid \Lambda_L) = \sum_{k=0}^{t-1} (k-\mu_L(t))^2 \frac{p_k}{P_L(t)}, \tag{4.19}$$

and

$$\sigma_H^2(t) = \sum_{k=t}^{N-1} (k-\mu_H(t))^2 P(k \mid \Lambda_H) = \sum_{k=t}^{N-1} (k-\mu_H(t))^2 \frac{p_k}{P_H(t)}. \tag{4.20}$$

Having found the histogram statistics that follow from each possible threshold value, we are in a position to search over all threshold values for T which minimizes the within-group variance. Specifically, by a exhaustive search we find the optimal threshold T which satisfies

$$\sigma_w^2(T) = \min_{0 \le t < N} \sigma_w^2(t). \tag{4.21}$$

Otsu's method merits consideration in applications where human intervention in the signal analysis process must be minimized. It does not need *a priori* probability estimates. It does not make any assumptions about the distribution of histogram values. Two problems weigh on the approach, however:

- It requires a search and recomputation of the statistics (4.17)–(4.20) over all possible threshold values.
- There may not be a unique minimum in (4.21), and, unless some goal-directed are imposed, there is no criterion for selecting one variation-reducing threshold over another with the same effect.

The exercises explore some algorithm refinements that reduce the recomputation burden. But the second point is troublesome. If we knew the within-group variance to be unimodal, then the search would always identify an optimal threshold.

Experiments supported—and Otsu conjectured—variance unimodality within segmented regions, but it does not hold true in general. In summary, nonparametric strategies generally involve more computational tasks than their parametric cousins. Thus, a preliminary survey of the statistical parameters of input signals to an analysis application may be warranted. A study of the statistics within histograms, for example, may prove the feasibility of the simpler parametric strategy.

4.2.3.3 An Information-Theoretic Approach. Attempting to avoid the modality problems inherent to Otsu's algorithm, Kittler and Illingworth [19] approached the problem by trying to find the mixture of Gaussian probability distributions that best matches the signal histogram for a given threshold value. For their optimality criterion, they employed relative entropy [20, 21], an information-theoretic tool, which we introduced in Section 1.8.4. They adopted a model of the histogram as a scaled sum of two normal distributions. Thus, theirs is a parametric approach akin to Chow and Kaneko's; however, it avoids the supposition of *a priori* probabilities for noise and meaningful signal segments, and therefore it represents a significant extension to the parametric method.

Following Kittler and Illingworth, let us suppose that p_k is given by (4.14), and q_k is an alternative distribution. Then the relative entropy, $I(p, q)$, of the distribution p_k with respect to q_k is

$$\begin{aligned} I(p,q) &= \sum_{k=0}^{N-1} p_k \log_2 \frac{p_k}{q_k} = \sum_{k=0}^{N-1} p_k \log_2 p_k - \sum_{k=0}^{N-1} p_k \log_2 q_k \\ &= -H(p) - \sum_{k=0}^{N-1} p_k \log_2 q_k, \end{aligned} \tag{4.22}$$

where $H(p)$ is the entropy of p,

$$H(p) = \sum_{k=0}^{N-1} p_k \frac{1}{\log_2 p_k}. \tag{4.23}$$

It can be shown that $I(p, q) \geq 0$ for all distributions p and q. Furthermore, $I(p, q) = 0$ if and only if $p = q$. Finally, let us note that $\log_2(p_k/q_k)$ is the information increment, given that the signal $f(n) = k$, that supports the histogram $h(k)$ having distribution p_k instead of q_k. Thus, the average information in favor of $h(k)$ following distribution p_k instead of q_k is $I(p, q)$, from (4.22). If q_k is a scaled sum of two normal distributions, then

$$q_k = \frac{a_1}{\sqrt{2\pi}\sigma_1} e^{-(k-\mu_1)^2/(2\sigma_1^2)} + \frac{a_2}{\sqrt{2\pi}\sigma_2} e^{-(k-\mu_2)^2/(2\sigma_2^2)}, \tag{4.24}$$

where a_1 and a_2 are constants. Now, q_k represents the histogram $h(k)$ better when signal values $f(n) = k$ discriminate in its favor over p_k; in other words, (4.24) should be minimized for best representing $h(k)$ with a sum of two Gaussians scaled by a_1 and a_2. Since $H(p)$ depends only upon the given histogram, this means that

$$I(p,q)+H(p)=-\sum_{k=0}^{N-1} p_k \log_2 q_k \tag{4.25}$$

should be minimized.

We can minimize (4.25) by approximating the statistical parameters of the two components of the distribution q_k for each candidate threshold, $0 < t < N-1$. If t lies between well-separated means, μ_1 and μ_2, then we should expect reasonable estimates. Thus, for each such t, we take $\mu_{1,t} = \lfloor t/2 \rfloor$ (the floor, or integer part of $t/2$). Let $a_{1,t}$ and $\sigma_{1,t}$ be the mean and standard deviation of $\{p_1, p_2, ... p_{t-1}\}$, respectively. We set $\mu_{2,t} = \lfloor (N-t)/2 \rfloor$, and let $a_{2,t}$ and $\sigma_{2,t}$ be the mean and standard deviation of $\{p_t, p_{t+1}, ..., p_{N-1}\}$, respectively. Lastly, we substitute these values into (4.24) to obtain

$$q_k(t)=\frac{a_{1,t}}{\sqrt{2\pi}\sigma_{1,t}}e^{-(k-\mu_{1,t})^2/(2\sigma_{1,t}^2)}+\frac{a_{2,t}}{\sqrt{2\pi}\sigma_{2,t}}e^{-(k-\mu_{2,t})^2/(2\sigma_{2,t}^2)}. \tag{4.26}$$

Then, the optimal threshold is T where

$$-\sum_{k=0}^{N-1} p_k \log_2 q_k(T) \le -\sum_{k=0}^{N-1} p_k \log_2 q_k(t) \tag{4.27}$$

for all t, $0 < t < N-1$.

An extensive literature on thresholding testifies that no single technique guarantees correct signal segmentation for all applications. General surveys include [22, 23]. Others concentrate on thresholding text [24, 25] or map [26] images. Figure 4.8 shows a signal, its histogram, and the results of thresholding it using the Otsu algorithm. Simple threshold selection methods and bottom-up signal classification strategies very often work just as well as the more sophisticated techniques. When they fail, combining methods is fruitful. Many of the methods we cover can be extended, often in straightforward ways, to multiple threshold values. Such an approach looks for multiple thresholds and partitions the domain into several regions according to signal magnitude. This is also called signal quantization, since it maps signal values that spread over a wide range to a set of signal values that vary significantly less. The exercises explore this important problem. More challenging applications require some top-down, goal-directed mechanism for improving the thresholding, segmentation, and labeling of a signal.

Finally, let us confess that we have neglected an important technique for signal segmentation: local or adaptive thresholding.

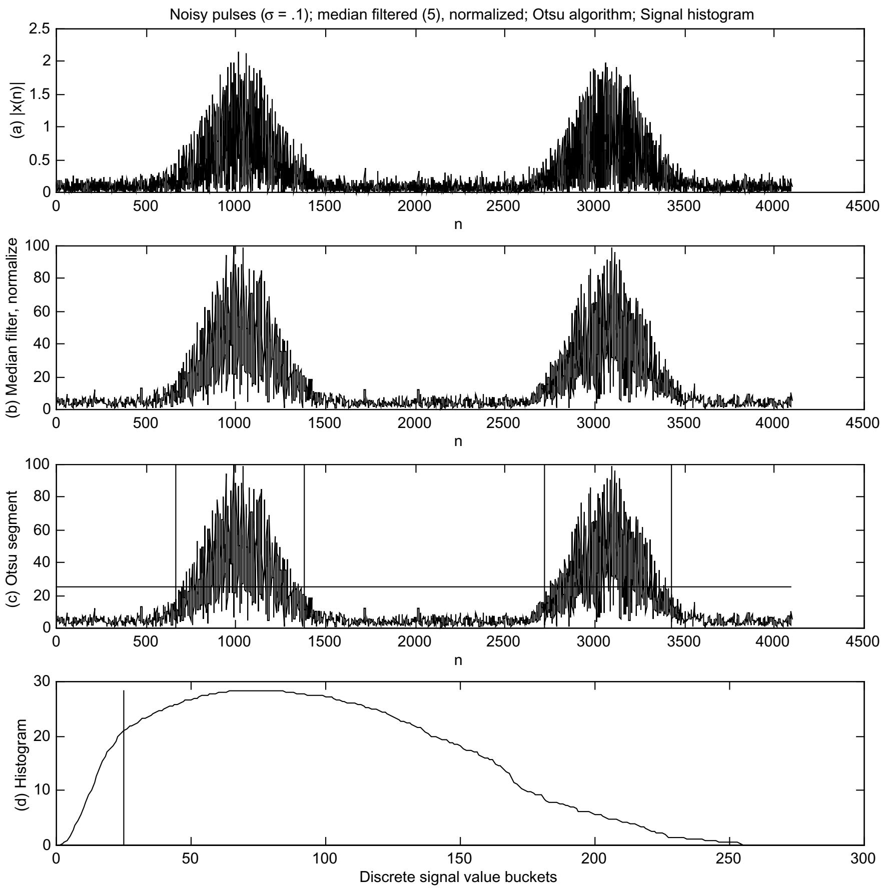


Fig. 4.8. A pulse signal (a), median filtered and normalized (b), threshold from Otsu's algorithm (c), and the signal histogram (d).

4.2.4 Local Thresholding

The methods covered above, by more or less sophisticated means, attempt to arrive at a global threshold value that will break down the signal. In many problems, the threshold that successfully segments a signal in one area fails to perform adequately in another area. But there is nothing in the global thresholding methods we have detailed above—based on histogram analysis—that prevents them from being applied to finite regions of the signal domain. When we find and apply thresholds locally within a signal's domain, the technique is called local thresholding or—since the threshold value adapts to the statistics or entropy of the signal locally—adaptive thresholding.

It may be possible, and indeed it may be essential, to vary the threshold at different places within the signal domain. This could be the case where the gain of the signal processing elements of the system varies. Falling signal strength could cause meaningful components to be misclassified as background noise. Thus, the threshold should adapt to the overall magnitude of local regions of the signal. Varying illumination, shading, and object reflectance make adaptive thresholding vital to many image analysis applications. Optical character recognition is but one example [27]. In digital telephony, echo cancellation should be suspended during episodes of double-talk, when both the near-end and far-end speakers are talking. But the threshold level for deciding that the near-end speaker is talking needs to adapt to the signal level of the far-end speaker; otherwise, cancellation may occur when both speakers are talking softly or might not take place when the far-end speaker is very loud. One approach is to declare double-talk when the the near-end signal level $s(n) > (1/2)\max\{x(n), x(n-1), ..., x(n-N)\}$, for some $N > 0$. By comparing levels over a whole range of recent far-end speaker voice magnitudes, the algorithm accomodates the unknown echo delay in the near-end circuit [28].

4.3 TEXTURE

Signal segmentation often requires that regions with certain regular patterns of signal values be distinguished from one another. It is not the absolute signal level—as with the previous section's principal concern, thresholding—but rather the repetitive transition of the signal through some value range that is of interest in such an application. This section covers methods for segmenting a signal into regions of different texture.

A threshold-based segmentation algorithm does not easily handle signals with periodic components. Examples covered in the previous sections, such as the speech fragments and the dual-tone multifrequency telephone system signaling pulses, are cases in point. It is possible to apply a preliminary smoothing filter to the values prior to thresholding. This blends the oscillation together and reduces the average value of the signal over a region, and then the thresholding operation properly isolates the region of interest. Regions found by one threshold, T_1, but not by another threshold $T_2 \neq T_1$ constitute a distinct texture field. Perhaps several filters and thresholds are necessary. As an alternative, we might consider segmenting the large-magnitude signal regions from the low-magnitude signal regions and then estimating an overall distance between peaks or between troughs. Areas that show different average intervals between crests or troughs represent different textured signal components. We are faced with a number of algorithm design issues, though, whatever approach we select. It is desirable to arrive at some indication of texture that produces a measure directly from the signal values, without preliminary filtering followed by thresholding.

Intuitively, the notion of texture incorporates the presence of smoothness or roughness and the overall character of repetition and continuity as opposed to rupture or discontinuity. One tool in wide use by computer vision researchers is a collection of photographs of textures, originally intended for use by artists, the Brodatz textures [29]. Many pioneering researchers in the field pursued the problem of texture characterization from psychological standpoint [30]. Some, in particular, worked with the principles of Gestalt psychology in mind [31, 32], which stipulates that pictorial features group together in such a way that the overall constellation achieves a form which cannot be expressed by descriptions of the individual elements. These pictorial features include proximity, similarity, continuation, symmetry, closure, and common fate.

Over the years, research efforts in texture analysis have been confined to three broad categories: statistical, spectral, and structural. Statistical approaches apply moment computations to the signal values: variance, skew, and kurtosis, for example [33]. Spectral approaches analyze the signal by comparing it to sinusoidal or Gabor elementary functions (Chapter 1) of varying frequencies [34]. And structural approaches endeavor to find a set of texture elements or primitives that obey a repeating pattern or a relational grammar in the signal [35].

Perhaps one of the most pressing problems is to define the type of repetitive structure, the texture regions in the signal that the application must find. Readers might well suppose that here we would introduce a formal definition of texture; this has, after all, been much the pattern of presentation so far. The U.S. industrial standard for surface metrology [36] defines it as repetitive or random deviations from the nominal surface within the three-dimensional topography of the surface. One might well object to this definition, since the terms "repetitive" and "random" contradict one another! Unfortunately, there is no suitable formal definition for texture, and there is no best approach to segmentation. The methods used are mostly ad hoc, and, if there can be anything close to a definition of texture, it is whatever the chosen method seems to find within the signal! The whole notion of what constitutes texture and what does not is quite uncertain. Let us consider the principal methods for segmenting signals according to their texture content in the next three sections. Though we lack a precise formal definition of texture, each approach constitutes a practical, working concept of what texture is—indeed it is whatever that method finds.

4.3.1 Statistical Measures

The statistical approach to texture segmentation and classification is the most widely used. However problematic the formal definition of texture is, methods to characterize and quantify it are quite important in areas such as materials science and remote sensing. In surface metrology, for example, many standard statistical measures of one- and two-dimensional textures are in use. There are also a number spectral measures of texture, and we will cover them, albeit briefly, in the next section. Some of these texture parameters are admittedly ad hoc, but they are nevertheless widely implemented and widely respected indications of the presence of texture within signals.

4.3.1.1 Basic Measures. One way to measure the amount of variation in a signal is to average the departure of the signal from its mean over a fixed interval. This statistical parameter of signal texture is the roughness average. It is the most widely used measure in the world, especially for manufactured surface characterization, and it has been standardized in the United States and internationally [33, 36–38].

Definition (Roughness Average). Let $x(t)$ be an integrable analog signal; let $[a, b]$ an interval, $a < b$; and let μ be the mean of $x(t)$ over $[a, b]$. Then the roughness average of $x(t)$ over $[a, b]$ is

$$R_a(x(t),[a,b]) = \frac{1}{b-a}\int_a^b |\,x(t)-\mu\,|\,dt. \tag{4.28a}$$

And if $x(n)$ is a discrete signal, $[a, b]$ is an interval, $a < b$, and μ is the mean of $x(n)$ over $[a, b]$, then the roughness average of $x(n)$ over $[a, b]$ is

$$R_a(x(n),[a,b]) = \frac{1}{b-a}\sum_{n=a}^{n=b} |x(n)-\mu\,|. \tag{4.28b}$$

When the interval over which we compute the roughness average is understood, it is common to write this parameter as $R_a(x)$. If the signal values are also unambiguous, it is simply R_a.

The roughness average indicates a change in signal variation, but it fails to explicate the distribution of crests, troughs, and transients in the signal. Somewhat more sophisticated statistical measures can be of help. We can, for example, better understand the distribution of signal values by applying a variance measure to the signal regions of interest. As a texture measure, the variance is also a standard.

Definition (Root Mean Square (RMS) of Roughness). If $x(t)$ is an analog signal, $[a, b]$, is an interval, $a < b$, and μ is the mean of $x(t)$ over $[a, b]$, then the RMS of roughness for $x(t)$ over $[a, b]$ is

$$R_q(x(t),[a,b]) = \left(\frac{1}{b-a}\int_a^b |\,x(t)-\mu\,|^2 dt\right)^{1/2}. \tag{4.29a}$$

For analog signals, the equivalent definition is

$$R_q(x(n),[a,b]) = \left(\frac{1}{b-a}\sum_{n=a}^{b-1} |x(n)-\mu\,|^2\right)^{1/2}. \tag{4.29b}$$

It is common to omit the signal and region from the specification of R_a and R_q, since the problem context often makes it understood.

Example (Surface Profilometers and Atomic Force Microscopes). Let us look at the segmentation of surface profiles by texture characterization. Instruments such as profilometers and atomic force microscopes (AFM) acquire one-dimensional (and in many commercial instruments, two-dimensional) height profiles of a surface. Surface texture measurements from such profiles are critical for the control and diagnosis of the fabrication processes. These measures include such aspects of surface texture as fine-scale roughness, more widely spaced regularities called waviness, and directional features called lay.

The roughness average is by far the most widely computed and reported in manufactured surface characterization. R_a detects general signal profile variations, and a significant change over one part of the signal domain indicates a fundamental change in the process that produces the signal. Nevertheless, it has limitations. To wit, the roughness average parameter fails to detect the presence or absence of widely separated signal transients. R_a completely overlooks subpatterns of texture. (Structural techniques are better suited to this type of problem; see Section 4.2.3.3.) The intervals between texture elements are also invisible to R_a. Some help in characterizing surface texture comes from the R_q parameter. It measures the distribution of deviations from the mean of the signal, so when it is large, it is an indication that the rough features of the signal have wide magnitude variability.

Other texture segmentation methods rely on peak-to-valley measurements within signal regions. Let us consider a few of these next and then proceed to some texture measures that arise from applying statistical moment ideas to a signal's values.

Definition (R_t, R_z). The total indicated reading over $[a, b]$, R_t, is the difference between the signal maximum and signal minimum over the interval. Thus, for an analog signal, $x(t)$, we define

$$R_t(x(t), [a, b]) = \max\{x(t) : t \in [a, b]\} - \min\{x(t) \mid t \in [a, b]\}. \tag{4.30a}$$

The analog for a discrete signal, $x(n)$, is

$$R_t(x(n), [a, b]) = \max\{x(n) : t \in [a, b]\} - \min\{x(n) \mid t \in [a, b]\}. \tag{4.30b}$$

The five-point peak parameter, R_z, is

$$R_z(x,[a,b]) = \frac{1}{5}\left(\sum_{k=1}^{5}\left(p_k - v_k\right)\right), \tag{4.31}$$

where $p_k \geq p_{k+1}$ are the five highest values x takes on $[a, b]$, and $v_{k+1} \geq v_k$ are the five lowest values x takes on $[a, b]$.

These parameters find application in diverse disciplines. Seismologists use the total indicated reading parameter to calculate earthquake magnitude according to the Richter scale.[3] Let $s(t)$ be the seismograph needle's deviation from the centerline of a paper strip chart at time t during an event. If the epicenter is 100 km

[3]Charles F. Richter (1900–1985), a seismologist at the California Institute of Technology, established the popular logarithmic scale for earthquake magnitude in 1935.

away from the instrument, then the seismologist computes the Richter magnitude, $M_L = \log_{10}[R_t(s)]$. The problem with the total indicated reading as a measure of texture is that it is sensitive to impulse noise in the signal. The R_z parameter is one of several that surface metrologists, for example, use to avoid this difficulty. An even larger average is tempting in (4.31); this provides roughness estimates with better immunity to occasional burrs and pits that blemish a generally good surface.

4.3.1.2 Higher-Order Moments. Further texture analysis measures proceed from an analysis of the signal using statistical moments. Following our histogramming discussion, suppose that we can obtain the discrete probability density function for the values of $x(n)$: $p_k = P[x(n) = k]$. For example, let $x(n)$ be a digital signal with $L \le x(n) \le M$ for $a \le n \le b$, where $a < b$. We form the histogram $h(k) = \#(\{n \in [a, b] \,|\, f(n) = k\})$ for each $k \in [L, M]$. If $M > L$, we then set $p_k = h(k)/(M-L)$; otherwise, $p_k = 1$. Then p_k is a discrete probability density function. The mean of x on $[L, M]$ is $\mu = (Lp_L + (L+1)p_{L+1} + \cdots + Mp_M)$. Then the variance, σ^2, skew, μ_3, and kurtosis, μ_4, are, respectively, as follows:

$$\sigma^2 = \sum_{k=L}^{M} (k-\mu)^2 p_k, \tag{4.32}$$

$$\mu_3 = \frac{1}{\sigma^3} \sum_{k=L}^{M} (k-\mu)^3 p_k, \tag{4.33}$$

$$\mu_4 = \frac{1}{4}\left(\sum_{k=L}^{M} (k-\mu)^4 p_k\right). \tag{4.34}$$

The R_q texture parameter is a variance measure. The skew measures the asymmetry of the values about the signal mean. The kurtosis measures the relative heaviness of the outlying signal values within the texture region. Some authors define kurtosis by subtracting three from (4.34); this ensures that a Gaussian distribution has zero kurtosis, but it does not affect the measure as a tool for signal segmentation.

These texture parameters supply some of the information missing from the roughness average figure. To use any of these parameters as a basis for signal segmentation, we compute the parameter over selected regions of the signal domain and label the regions according to the parameter's numerical value. A more informative segmentation of the signal is possible by calculating several of the parameters and assigning labels to the regions that represent combinations of significant texture parameters.

Notice that there is a distinct difference between the application of these texture measures and the previous section's thresholding algorithms to segmentation problems. When thresholding, the segmentation regions were discovered, signal value after signal value, by applying the thresholding criterion. On the other hand, to apply the texture measures, an interval or region of the signal domain must be used to calculate the statistical quantity, R_a or R_q, for example. Thus, a texture segmentation commonly starts with a preliminary partition of the signal domain into intervals, called frames, with texture measures assigned to the intervals. The bounds of these preliminary texture regions may be adjusted later with split and merge criteria, as we considered in Section 4.1.4. For precise registration of texture areas, a data-driven alternative exists. An application may apply texture measures to frames of a minimum size, say $N > 0$. After applying the measure to region $[a, b]$, where $b - a = N$, the statistics are compared to one or more threshold values. If $[a, b]$ contains texture, then the measures are applied to the larger region $[a, b + 1]$; otherwise the frame becomes $[a + 1, b + 1]$ and texture finding continues. Each time a region contains texture, the application attempts to expand it on the right, until at some iteration, say $[a, b + k + 1]$, the texture indication test fails. Then $[a, b + k]$ is declared to be texture-laden, and processing continues with the next minimal frame $[b + k + 1, b + k + N + 1]$. Albeit computationally expensive, this scheme avoids any top-down split and merge procedures.

4.3.1.3 Co-occurrence Matrices. One significant drawback to the above methods for texture segmentation is that they utilize no distance measures between intensity features within the texture. The moment parameters, such as R_q, do incorporate a notion of breadth of variation. In manufactured surface characterization, R_q is often touted as an alternative to the widely quoted R_a value. Nevertheless, signal and image analysts reckon that the most powerful techniques are those that compute statistics for the distribution of signal values separated various time intervals [39]. The next definition [40] incorporates the distance between texture highlights into our statistical indicators of texture.

Definition (Co-occurence Matrix). Let $x(n)$ be a digital signal; let $L \le x(n) \le K$ for some integers L, K and for $a \le n \le b$; and let δ be a time interval. Then $M_\delta = [m_{i,j}]_{N \times N}$ is the $N \times N$ matrix defined by $m_{i,j} = \#\{(x(p), x(q)) \mid p, q \in [a, b], x(p) = i$ and $x(q) = j$, and $\delta = |p - q|\}$. The co-occurrence matrix for $x(n)$ and time interval δ is defined $P_\delta = [m_{i,j}/N_\delta]$, where $N_\delta = \#\{(x(p), x(q)) \mid p, q \in [a, b]$ and $\delta = |p - q|\}$.

Thus, $m_{i,j}$ contains a count of the number of pairs (p, q) for which $x(p) = i$, $x(q) = j$, and p and q are time δ apart. P_δ estimates the joint probability that two signal will take values i and j at a displacement δ apart. Also, it is not necessary to restrict $x(n)$ to be a digital signal; as long as its range is finite, the co-occurrence matrix can be defined.

Example (Co-occurrence Matrix). Let $x(n) = [..., 0, \underline{1}, 2, 1, 1, 2, 0, 0, 1, 0, 2, 2, 0, 1, 1, 0, ...]$ be a digital signal, and suppose we compute the co-occurrence matrices for $\delta = 1$, 2, and 3 within the interval $0 \le n \le 15$. This signal is roughly sawtooth in shape, with the ramps positioned three time instants apart. We compute the co-occurrence matrics, P_1, P_2, and P_3 as follows:

$$15 \times P_1 = M_1 = \begin{bmatrix} 1 & 3 & 1 \\ 2 & 2 & 2 \\ 2 & 1 & 1 \end{bmatrix}, \tag{4.35a}$$

$$14 \times P_2 = M_2 = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 0 \end{bmatrix}, \tag{4.35b}$$

$$13 \times P_3 = M_3 = \begin{bmatrix} 3 & 1 & 1 \\ 2 & 1 & 1 \\ 0 & 3 & 1 \end{bmatrix}, \tag{4.35c}$$

where $N_1 = 15$, $N_2 = 14$, and $N_3 = 13$. Notice that the values are spread out in M_1 and M_2, the main diagonal values are relatively small, and there are few outliers. In contrast, M_3 contains two large values, several small values, and a maximal probability on the diagonal. If the structures within $x(n)$ are square pulses rather than ramps, the matrices M_1, M_2, and M_3 are even more distinct. The exercises explore these ideas further.

To use the co-occurrence matrix as a texture segmentation tool, the time interval δ must be selected according to the size and spacing of the signal regions. Suppose, for instance, that the signal $x(n)$ contains high-magnitude regions approximately Δ time instants apart and that we calculate the matrix P_δ, where δ is smaller than Δ. If $|p - q| < \delta$, then the values $x(p)$ and $x(q)$ are likely to fall into the same region. Hence P_δ entries on the diagonal should be large. Large values do not concentrate on P_δ's diagonal when δ is smaller than the typical region size. If a textured signal contains features of two sizes, δ and ε, then we should expect to find matrices P_δ and P_ε to be largely diagonal. In general, many values of δ are unnecessary for good texture segmentation results.

The idea of co-occurrence measures in texture analysis dates back to the early work of Julesz [41]. The co-occurrence matrix entry on row i and column j, $P_\delta(i, j)$, gives the probability that a sampling of a signal value and its neighbor δ time instants away will have values i and j. It is also possible to sample a point and two others, δ_1 and δ_2 time instants away, to generate third-order co-occurrence statistics. Similarly, fourth- and high-order co-occurrence measures are possible. Julesz conjectured that humans cannot distinguish textures that contain identical first- and second-order co-occurrence statistics; thus, visual texture fields may differ in their third- or higher-order statistics, but this effect is too subtle for the eye–brain system to detect. Julesz's thesis was tremendously attractive to computer vision researchers. It promised to bound the ever-growing number of texture measures by invoking a discriminability criterion based on human pattern detection performance. Possibly, too, researchers could anchor a definition of texture itself in the second-order co-occurrence statistics. But counterexamples were soon found. Julesz and other investigators were able to synthesize texture fields, which humans could distinguish, that had different third- or fourth-order co-occurrence statistics but identical first- and second-order statistics [42]. Under further scrutiny, the synthesized counterexamples themselves were shown to have different local and global co-occurrence statistics. The human visual system excels at perceiving distinct local image characteristics and can group local variations into global patterns over wide areas of the visual field. This causes many visual illusions. Thus, persons viewing the counter-example textures were able to discriminate regions within them, even though the global low-order co-occurrence statistics were identical. Ultimately, it appears that something very close to the Julesz thesis holds true and that humans cannot distinguish textures that locally have the same first- and second-order co-occurrence statistics [43].

However powerful the method of co-occurrence matrices, it evidently turns an analysis problem of a one-dimensional entity—the signal—into the two-dimensional analysis problem of analyzing the co-occurrence matrix. Hence, the key to achieving any analytical power from the method is to keep the co-occurrence matrices and their number small. For developing texture descriptors, researchers have suggested a wide variety of parameters obtained from the co-occurrence matrices [44, 45]. Briefly, let us review some of the most important ones.

Definition (Co-occurrence Matrix Texture Descriptors). Let $x(n)$ be a digital signal; $L \le x(n) \le K$ for some integers L, K and for $a \le n \le b$; let P_δ be the co-occurrence matrix for $x(n)$ with time interval δ; and denote the element at row i and column j of P_δ by $P_\delta(i, j)$. Then, the angular second moment, or energy uniformity, T_a; contrast, T_c; inverse difference moment, T_d; entropy, T_e; and maximum, T_m, descriptors are as follows:

$$T_a(\delta) = \sum_{i=L}^{K} \sum_{j=L}^{K} P_\delta^2(i, j), \tag{4.36}$$

$$T_c(\delta) = \sum_{i=L}^{K} \sum_{j=L}^{K} (i-j)^2 P_\delta(i, j), \tag{4.37}$$

$$T_d(\delta) = \sum_{i=L}^{K} \sum_{j=L}^{K} \frac{P_\delta(i, j)}{1+(i-j)^2}, \tag{4.38}$$

$$T_e(\delta) = -\sum_{i=L}^{K} \sum_{j=L}^{K} P_\delta(i, j) \log_2 P_\delta(i, j), \tag{4.39}$$

$$T_m(\delta) = \max\{P_\delta(i, j) \mid L \le i, j \le K\}. \tag{4.40}$$

These values are easy to compute and their magnitudes shed light on the nature of the co-occurrence matrix. Note that T_a is smaller for uniform P_δ values and larger for widely varying co-occurrence matrix entries. Low-T_c and high-T_d descriptors indicate heavy groupings of values on P_δ's main diagonal. The entropy descriptor is large when P_δ values are relatively uniform. The maximum value can be thresholded, or compared to $(T_a(\delta))^{1/2}$, to detect extreme co-occurrence matrix entries. It may also be useful to study the behavior of the descriptors when the time interval δ varies [46].

Now let us turn to another method for texture segmentation. The descriptors it generates turn out be useful for extracting different sizes of repetitiveness in a signal.

4.3.2 Spectral Methods

The spectral approach to texture segmentation applies to textured signals that are very periodic in nature. The analysis tool used in spectral approaches to texture is consequently the sinusoids, or, more generally, the complex exponential, e^{x+jy}. Rudimentary statistical measures of texture, such as the roughness average, R_a, do not adequately account for the presence of different periodic trends in the signal. For instance, a broad undulation may be modulated by a more frequent periodic phenomenon—a ripple on top of a wave. In machined surface characterization, the term for the broad undulations is waviness. Metrologists distinguish waviness from roughness, which is a variation on a finer scale. But this signal phenomenon is hardly confined to the science of characterizing manufactured surfaces. The next example explores this type of textured signal in the context of biomedicine.

Example (Waviness). Figure 4.9 shows some biomedical signals taken from an anesthetized dog: a blood pressure trace (in millimeters of mercury) and an

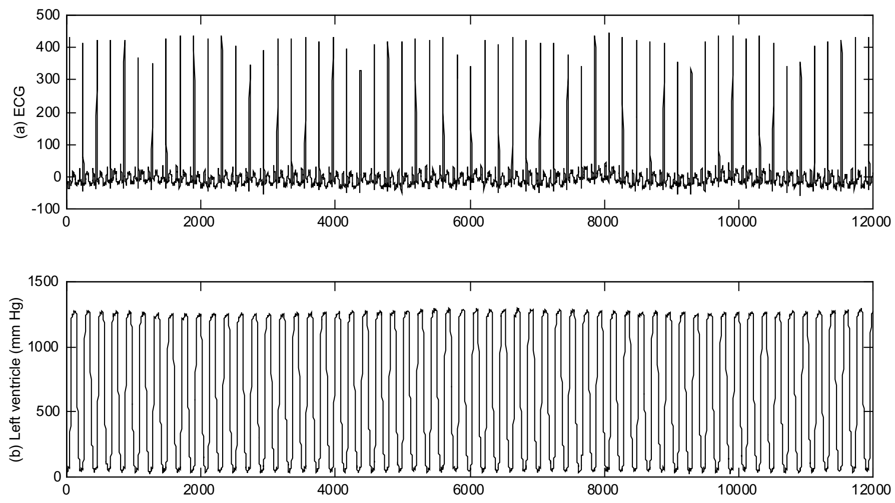


Fig. 4.9. Examples of short-term and long-term periodicities in signals. Panel (a) shows the electrocardiogram from an anesthetized dog. In panel (b) an anesthetized dog's left ventricular blood pressure indicates a short-term variation for the heart contraction and a long-term undulation due to the dog's breathing. This signal has a distinct waviness trend, a long-term undulation, which contrasts with the short-term pressure cycles.

electrocardiogram (dimensionless units). The left ventricle pressure is oscillatory, and—as expected—the histogram has two peaks, or modes, corresponding to the high- and low-pressure intervals within the heart beat. It is easy to identify a threshold that finds the individual pulses, but isolating the gentle waviness that underlies their progression is not as easy. Furthermore, the moment-based statistical methods we considered above do not help to identify the waviness. Such different periodicities in signals occur throughout signal processing and analysis. Problems in biomedicine, digitized speech, industrial control, vibration analysis, remote sensing, shape recognition, and, of course, surface characterization provide many other examples. We need to discover, for such applications, a method distinguish between the short-term periodic features from the long-term undulations in a signal. In surface characterization, the minute variations indicate the roughness of a signal profile, and large-scale, broad repetitions show the presence of waviness.

The sinusoidal functions naturally spring to mind in problems involving short-term and long-term periodicity. Not surprisingly, in order to segment signals containing both a waviness and a roughness character, signal analysts generally employ spectral measures. A comparison of the signal with sinusoids of varying frequency is the natural approach. This is the basis of the spectral method. To compare a signal to a sinusoid, we might proceed as with normalized cross-correlation, by taking the inner product of a portion of the signal with a sinusoidal function. In fact, a family of sinusoids of different frequencies can encompass the range of fine texture, medium texture, and waviness that a signal contains. This is called a spectral strategy, because in its fullest application it can expose an entire range, or spectrum, of periodicities in a signal's values. The foundation of this approach was laid, in fact, by our study of Hilbert spaces in Chapters 2 and 3. Our exploration of spectral approaches to signal texture will be introductory only. In fact, research indicates that statistical methods—and the co-occurrence matrix technique in particular—hold sway over spectral methods [39, 44, 47]. Notwithstanding the superiority of statistical approaches, there is one application for which spectral methods offer an intuitive and powerful approach to texture characterization: the identification of different periodicities—the roughness from the waviness—within a signal.

Let us contemplate the problem of discovering the various periodicities in a discrete signal $x(n)$ over an interval $[a, b] \subset \mathbb{Z}$. Let us assume that $a = 0$ and $b = N - 1$, thus translating our signal to align the beginning of the texture region of interest with the origin. Trigonometric functions, especially the sinusoids, are the natural tool with which to test $x(n)$ on $[0, N - 1]$ for periodic behavior. We know from the introduction to discrete signal frequency in Chapter 1 that the signals $\cos(2\pi nk/N)$, for $k = 0, 1, ..., \lfloor N/2 \rfloor$, range from the lowest ($k = 0$) to the highest (k = largest integer less than $N/2$) possible frequency on $[0, N - 1]$. The inner product $\langle x(n), \cos(2\pi nk/N)\rangle$ of the signals $x(n)$ and $\cos(2\pi nk/N)$ restricted to $[0, N - 1]$ measures of similarity of $x(n)$ to the sinusoid $\cos(2\pi nk/N)$. It is convenient to assume that $x(n) = 0$ outside $[0, N - 1]$. Thus, those values of k over the range $0, 1, ..., \lfloor N/2 \rfloor$, for which $\langle x(n), \cos(2\pi nk/N)\rangle$ has a relatively large magnitude, indicate the presence of a significant periodicity in $x(n)$ of frequency $\omega = 2\pi k/N$ radians per sample. Let's capture this concept by defining the periodicity descriptors $X_c(k)$ for a signal $x(n)$ defined on $[0, N-1]$:

$$X_c(k) = \left\langle x(n), \cos\left(\frac{2\pi nk}{N}\right)\right\rangle = \sum_{n=0}^{N-1} x(n)\cos\left(\frac{2\pi nk}{N}\right). \tag{4.41}$$

Note that for values of $k = \lfloor N/2 \rfloor + 1, \lfloor N/2 \rfloor + 2, \ldots, N-1$, the descriptors repeat in reverse order; specifically, $X_c(N-k) = X_c(k)$ for $k = 1, 2, \ldots, \lfloor N/2 \rfloor$. We can also define periodicity descriptors based on the sine function,

$$X_s(k) = \left\langle x(n), \sin\left(\frac{2\pi nk}{N}\right)\right\rangle = \sum_{n=0}^{N-1} x(n)\sin\left(\frac{2\pi nk}{N}\right). \tag{4.42}$$

In (4.42) we see that $X_s(N-k) = -X_s(k)$ for $k = 1, 2, \ldots, \lfloor N/2 \rfloor$, so that for $k > \lfloor N/2 \rfloor$, the $X_s(k)$ descriptors may be useful for detecting sign-inverted periodic signal components also.

One difficulty with the periodicity descriptors is that we do not in general know whether $x(n)$ has a maximal value at $n = 0$ as does $\cos(2\pi nk/N)$. In other words, the texture field may be shifted so that it does not align with the sinusoids used to compute the inner product periodicity measures. Suppose we shift $x(n)$ values in a circular fashion; thus, we let $y(n) = x((n+p) \bmod N)$. We desire that our texture descriptors respond equally well to $y(n)$ and $x(n)$. That is, the descriptors should support some kind of translation invariance. Equivalently, we can shift the sinusoids by amount p and compute new descriptors, $X_{c,p}(k) = \langle x(n), \cos(2\pi n(k-p)/N)\rangle$. This accounts for possible translation of the texture in the source signal $x(n)$, but now we have quite a large computation task. We must compute $X_{c,p}(k)$ for all possible offsets p and all possible frequencies $2\pi k/N$. Can we relate the shifted descriptor, $X_{c,p}(k)$, to the original $X_c(k)$ descriptor values in such a way that we avoid computation of $X_{c,p}(k)$ for multiple offsets p? We calculate,

$$\begin{aligned} X_{c,p}(k) &= \sum_{n=0}^{N-1} x(n)\cos\left(\frac{2\pi(n-p)k}{N}\right) = \sum_{n=0}^{N-1} x(n)\cos\left(\frac{2\pi nk}{N} - \frac{2\pi pk}{N}\right) \\ &= \sum_{n=0}^{N-1} x(n)\left[\cos\left(\frac{2\pi nk}{N}\right)\cos\left(\frac{2\pi pk}{N}\right) + \sin\left(\frac{2\pi nk}{N}\right)\sin\left(\frac{2\pi pk}{N}\right)\right] \\ &= X_c(k)\cos\left(\frac{2\pi pk}{N}\right) + X_s(k)\sin\left(\frac{2\pi pk}{N}\right), \end{aligned} \tag{4.43}$$

which shows that the descriptor $X_{c,p}(k)$ depends not only on $X_c(k)$ but on the sine-based descriptors as well. In other words, as the repetitive pattern of $x(n)$ shifts, both the cosine-based descriptor and the sine-based descriptor vary.

It turns out that a translation-invariant descriptor arises by combining the cosine- and sine-based descriptors into the exponential $\exp(j2\pi nk/N) = \cos(2\pi nk/N) +$

$j\sin(2\pi nk/N)$. Equation (4.43) already hints of this. We form the inner product of $x(n)$ with $\exp(j2pnk/N)$, as above, to get an exponential-based descriptor, $X(k)$:

$$\begin{aligned} X(k) &= \left\langle x(n), \exp\left(\frac{2\pi jnk}{N}\right)\right\rangle = \sum_{n=0}^{N-1} x(n)\exp\left(\frac{-2\pi jnk}{N}\right) \\ &= \sum_{n=0}^{N-1} x(n)\cos\left(\frac{2\pi nk}{N}\right) - j\sum_{n=0}^{N-1} x(n)\sin\left(\frac{2\pi nk}{N}\right) = X_c(k) - jX_s(k). \end{aligned} \tag{4.44}$$

Now let's consider computing $X(k)$ values for a translated texture $y(n) = x((n+p) \bmod N)$, or, equivalently, by computing the inner product, $X_p(k) = \langle x(n), \exp(2\pi jn(k-p)/N)\rangle$:

$$\begin{aligned} X_p(k) &= \left\langle x(n), \exp\left(\frac{2\pi j(n-p)k}{N}\right)\right\rangle = \sum_{n=0}^{N-1} x(n)\exp\left(\frac{-2\pi j(n-p)k}{N}\right) \\ &= \exp\left(\frac{2\pi jpk}{N}\right)\sum_{n=0}^{N-1} x(n)\exp\left(\frac{-2\pi jnk}{N}\right) = \exp\left(\frac{2\pi jpk}{N}\right)X(k). \end{aligned} \tag{4.45}$$

Now we have $|X_p(k)| = |\exp(2\pi jpk/N)X(k)| = |X(k)|$. In other words, we have shown the following:

Proposition (Translation Invariance of $|X(k)|$). The complex norm of the periodicity descriptor, $|X(k)|$, is invariant with respect to the modulo-N translation of the textured source signal, $x(n)$, defined by (4.44) on $[0, N-1]$.

This, then, is the tool we need to isolate periodic components within textures according to their different frequencies.

The values $X(k)$ in (4.44), for $k = 0, 1, ..., N-1$, represent the discrete Fourier transform of the signal $x(n)$ on $[0, N-1]$. We know the signals $X(k)$ already from Chapter 2. There we considered them, after a suitable normalization, as an orthonormal set on $[0, N-1]$, $\{e_k(n) \mid 0 \le k \le N-1\}$:

$$e_k(n) = \frac{\exp(j2\pi kn/N)}{\sqrt{N}}[u(n) - u(n-N)], \tag{4.46}$$

where $u(n)$ is the unit step signal. It is common to call the values $X(k) = \langle x(n), \exp(2\pi jnk/N)\rangle$ the Fourier coefficients of $x(n)$, even though, precisely speaking, we need to normalize them according to the inner product relation on an orthonormal set (4.46). Readers must pay close attention to the definitions that textbooks provide. When discussing Hilbert space decompositions, textbooks usually normalize the coefficients according to (4.46), whereas ordinarily they omit the $N^{-1/2}$ factor in the DFT's definition.

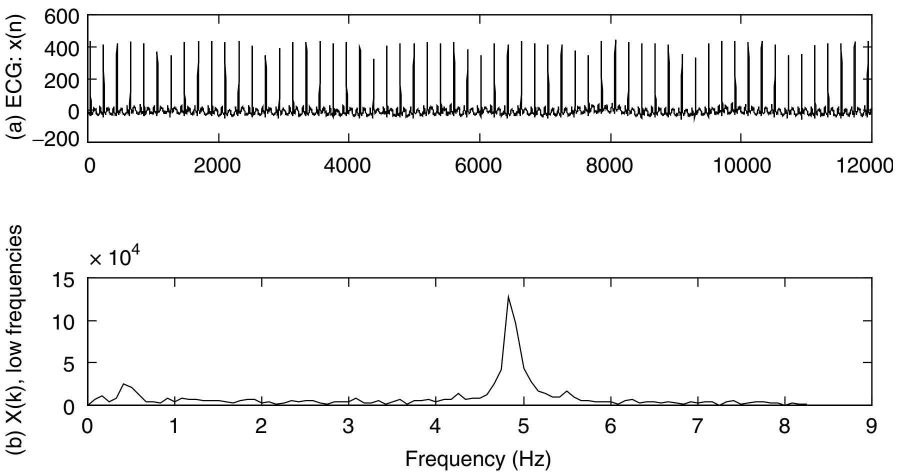


Fig. 4.10. Biomedical signal waviness extraction. The spectral method helps to identify the two sinusoidal components of the ECG signal (a). The main periodic trend is just below 5 Hz, and the waviness is the small peak below 0.5 Hz (b).

The theory and applications of the DFT will comprise much of Chapters 6 and 7. Let us content ourselves here with two applications of spectral texture analysis: first, to the biomedical signals of Fig. 4.9 and, second, to digitized speech segmentation and recognition. Figure 4.10 shows the results of applying the spectral texture descriptors $|X(k)|$ to biomedical signals.

Example (Biomedical Signal Frequency Analysis). Consider the ECG signal for the anesthesized dog.

Application (Voiced versus Unvoiced Speech Segmentation). This example continues the discussion of speech phoneme segmentation that we began in Section 4.2.2 with voiced and unvoiced consonants. Linguists typically classify speech events according to whether the vocal cords vibrate during the pronunciation of a speech sound, or phoneme (Table 4.3). If the vocal cords do vibrate, then there is said to be a voiced speech event. If there is no vocal cord vibration, then the phoneme is unvoiced. It is also possible that a speech signal contains no speech sound; thus, it is simply background, or noise. One approach to segmenting speech classifies its portions as voiced (V), unvoiced (U), or noise (N). For example, a digital recording of the English phrase “linear fit” begins, divides the two words, and ends with noise regions. The most apparent phonemes, then, are /l I n i ɘ f I t/, of which /f/ and /t/ are unvoiced. A preliminary segmentation, Σ_0, by voiced/unvoiced/noise classification is (N, V, N, U, V, U, N), respecting the stipulation that no adjacent regions have the same type. Actually, there are a number of unvoiced events that only become apparent when the speech signal is digitized and spread out over time. One may find, for example, momentary unvoiced aspirations and even periods

TABLE 4.3. More Complete Table of Phonemes and Examples from American English [12].[a]

Phoneme	Example	Class	Phoneme	Example	Class
/i/	even	Front vowel	/I/	signal	Front vowel
/e/	basis	Front vowel	/ε/	met	Front vowel
/ae/	at	Front vowel	/a/	father	Mid vowel
/Λ/	but	Mid vowel	/⊃/	all	Mid vowel
/schwa/	signal	Mid vowel	/u/	boot	Back vowel
/o/	boat	Back vowel	/U/	foot	Back vowel
/ɨ/	roses	Back vowel	/ɜ/	Hilbert	Mid vowel
/a^{W}/	down	Dipthong	/a^{y}/	cry	Dipthong
/$\supset^{y}$/	boy	Dipthong	/y/	yet	Semivowel glide
/w/	wit	Semivowel liquid	/r/	rent	Semivowel glide
/l/	linear	Semivowel liquid	/m/	segment	Nasal consonant
/n/	nose	Nasal consonant	/η/	Nguyen	Nasal consonant
/p/	partition	Unvoiced stop	/t/	fit	Unvoiced stop
/k/	kitten	Unvoiced stop	/b/	bet	Voiced stop
/d/	dog	Voiced stop	/g/	gain	Voiced stop
/h/	help	Aspiration	/f/	fit	Unvoiced fricati
/θ/	thanks	Unvoiced fricative	/s/	sample	Unvoiced fricati
/sh/	shape	Unvoiced fricative	/v/	vector	Voiced fricative
/∂/	that	Voiced fricative	/z/	zoo	Voiced fricative
/zh/	closure	Voiced fricative	/ch/	channel	Affricate
/j/	Jim	Affricate	/?/	no sound	Glottal stop

[a]Vowels are voiced, and they break down further into front, mid, and back classifications, according to t location of the tongue's hump in the mouth. Vowels typically consist of two oscillatory components, call formants. Other essentially vowel-like sounds are the dipthongs and the semivowels. The glottal stop is special symbol that indicates a momentary suspension of motion by the vocal cords (glottis). For examp without the /?/ symbol, the phoneme sequences for "I scream" and "ice cream" are identical. As big as it this table is far from panoramic; it offers but a glimpse of the natural language segmentation problem.

of noise, surrounding the voiced segments. A refined segmentation, Σ_1, therefore supplies: (N, U, V, U, V, U, V, U, V, U, V, U, N, U, V, U, N). In practical speech recognition applications, surprisingly, this segmentation is too crude! Subtle periods of background noise, with no voiced or unvoiced sound present, intrude into spoken words. A modern, sophisticated segmentation algorithm finds that several N regions split the unvoiced regions U in the refined segmentation above. This means that several of the unvoiced intervals have much shorter time extents than Σ_1 would indicate. The benefit is that a higher-level interpretation algorithm may be better able to recognize the brief U boundaries of the V segments as trailing and leading aspirations instead of, for example, unvoiced fricatives. Figure 4.5 illustrates such a speech segmentation example. We shall continue to expose the intricacies of natural language interpretation is this and the remaining chapters. Chapter 9, in particular, introduces several frequency-domain signal analysis tools for speech interpretation.

Vowels contain two frequency components called formants. It is hard to adapt the level-based approach of thresholding to these oscillatory parts of signals. Also, the statistical measures of texture do not account well for the time intervals between transitions, and yet this is the most important aspect of signal frequency. What we need to do is to discover some way to extract the degree of similarity between some raw, noisy signal and a pure sinusoid.

4.3.3 Structural Approaches

When the precise description of local texture elements is important, then the structual approach to texture segmentation comes to the fore. The statistical and spectral methods provide numerical estimates for the amount and distribution of variability within signal regions. Simple statistical measures lose the repeating parts of the signal in the sums and integrations of their mathematical implementation. The co-occurrence matrix method retains this information when co-occurrence matrices are computed for several values of the time offset, δ. The structural approach, on the other hand, relies on texture descriptors that provide a model of a local pattern that is replicated, in one or another degree, at one or another locations across the domain of the signal. Thus, the structural approach has a pattern recognition flavor. We need to develop some basic tools for structural pattern recognition it order to cover this approach adequately, and we will do this later in Section 4.7.

4.4 FILTERING AND ENHANCEMENT

This section considers some supplementary operations that can greatly improve the performance of low-level segmentation processes before the thresholding operation: convolutional smoothing, median filtering, morphological filtering, and histogram enhancement.

4.4.1 Convolutional Smoothing

Let us here expand upon the idea of filtering a signal before it passes to a threshold operation. The purpose of preliminary filtering is to prevent labeling mistakes before they happen. Three types of filters are in common use for this purpose: convolutional filters of the type we covered in Chapters 2 and 3, for discrete and analog signals, respectively; median filters; and morphological filters.

We noted by example in the previous section that oscillatory areas of signals may contain mixtures of low- and high-magnitude signal values. Such a region may be meaningful signal in its totality, and for many applications it is not correct to separately segment the low-magnitude regions and label them as background. A linear smoothing filter proved useful for this purpose. Chapter 2 introduced the *moving average* system given by

$$y(n) = \frac{1}{2N+1} \sum_{k=-N}^{N} x(k). \tag{4.47}$$

It is also called a box filter, from the shape of its impluse response. This linear, translation-invariant (LTI) system averages the input signal values within a window of length $2N+1$ around each $x(n)$. Its effect on a oscillatory signal region, $[a, b]$, is to blur the high and low signal values together. Thus, for some threshold $T > 0$, the system's output, $y(n) = (Hx)(n)$, has magnitude $|y(n)| \geq \mathrm{T}$ within $[a, b]$. It is also possible to smooth using an infinite impulse response filter. For example, $y(n) = ay(n - 1) + x(n)$, with impulse response $h(0) = 1$, and $0 < a < 1$ is a possibility (Section 2.4.2). Smoothing with LTI systems blends oscillations together into more blob-like, easily thresholded regions. It also reduces background noise when the signal of interest is of relatively low magnitude.

The moving average system of (4.47) is not the only filter available for smoothing signals. One problem that readers may already note (see Chapter 2) is that this filter is not causal. So it cannot be applied to a data stream as values arrive in real time. It also has sharp edges where its support ends, which, for some types of input signals, causes its output to change abruptly. We will find in Chapter 7 some problems with how this filter treats frequencies inside the input signal; in fact, this motivates us to search for alternatives. A causal box filter averages only the current and some previous samples: $y(n) = (1/3)[x(n) + x(n-1) + x(n-2)]$, for example.

Noise in signals varies widely. It is usually a detriment to signal quality; but in some interesting cases, it is actually helpful. In digital telephony, for example, comfort noise is generated and automatically inserted into the line during episodes when both talkers are quiet. The reason for this is that silent periods can be used to carry other voices. The telephone company's central office multiplexing equipment routinely disconnects the channel, fills it with other voice traffic, and swiftly provides an alternative link when someone speaks. But it is disconcerting to almost everyone to hear a silent line. Realizing this, telecommunication engineers design the multiplexers to inject a small amount of white noise onto the two ends of the interrupted link. Since the synthetic noise level is matched to the background noise present while the conversants speak, the change is imperceptible. Another example of the value of noise synthesis is in active noise cancellation.

Now let us return to our discussion of noise removal. We consider a filter that has been widely used for noise mitigation—the discrete version of the Gaussian. This filter has the impulse response

$$h(n) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{\tau n}{\sigma}\right)^2}, \tag{4.48}$$

where $\tau > 0$ is some sampling interval. This filter emphasizes the signal values near the value, $x(n)$. Thus, it differs from the moving average filter, which considers all quantities equally valuable in a neighborhood $[x(n - N), \dots, x(n + N)]$ around $x(n)$. We shall investigate this type of filter carefully in the last section of this chapter. For pattern recognition applications based on signal shape over multiple scales and convolutional filtering, there are strong theoretical reasons for using Gaussian filters above all others.

4.4.2 Optimal Filtering

It is also possible to derive filters that are, in some sense at least, optimal for removing the kind of noise that affects input signals to an application. Here, we will outline two approaches for designing such filters: One method uses least-squares principles [48–50], and another involves constrained minimization—a Lagrange[4] multipliers approach [51]. Time-domain methods for noise removal, such as these, generally depend upon statistical information about the signal values.

We can pose the search for an optimal noise removal filter as a least-squares search. Suppose that the noise removal filter H is linear and translation-invariant. Then the Convolution Theorem for LTI Systems (Chapter 2) instructs us that $y = Hx$ is given by convolution of $h(n) = (H\delta)(n)$. We assume a real-valued, noisy input signal $x(n)$. Let us further seek a finite impulse response (FIR) filter, of width $M = 2p + 1$, supported on $[m - p, m + p]$, and ask what filter coefficients $h(n)$ give the best estimate of x at time instant p: $y(p) = (Hx)(p)$. We view the FIR filtering operation as a matrix operation, $Ah = b$, where the rows of matrix A are noisy samples of the input signal $x(n)$ from $n = m - p$ to $n = m + p$; h is a column vector with entries $h(-p), h(-p + 1), ..., h(p)$; and b is a row vector with FIR filtered values of the input. Suppose we filter N sets of noisy input windows of signal $x(n)$ around $n = p$, $(a_{1,1}, a_{1,2}, ..., a_{1,M}), (a_{2,1}, a_{2,2}, ..., a_{2,M}), ..., (a_{N,1}, a_{N,2}, ..., a_{N,M})$, with the filter $h(n)$. Suppose further that we know the true value (i.e., before corrupted by noise) of the input signal for each sample set at $n = p$; call them $b_1, b_2, ..., b_N$. Then we are in fact asking for the vector h for which the matrix product Ah is the best approximation to the ground truth vector b, representing uncorrupted input signal values $x(n)$ at $n = p$. Our quest for a best noise removal filter reduces to a problem of finding the column vector h, given a matrix A, which upon matrix multiplication Ah is closest to known signal values b. The theory of this problem is easiest if we consider the closest approximation to b in vector norm: we need to minimze $\|Ah - b\|^2$. In other words, the best filter problem reduces to a classic least squares minimization problem. The following theorem—mainly from linear algebra and numerical analysis [48, 50, 52], but drawn with a signal processing twist—summarizes this unconstrained search for the best FIR noise removal filter.

Theorem (Least-Squares Optimal Noise Removal Filters). If the vectors x, y, and b are real and the matrices A and M are real, then with the above premises and notation, the following statements are true:

1. If the columns of matrix A are linearly independent, then A^TA is symmetric and positive definite: $(A^TA)^T = A^TA$, and $x^T(A^TA)x \geq 0$ for all vectors $x \neq 0$.
2. If the square matrix M is positive definite, then the quadratic equation in a real vector x,

[4]Joseph Louis Lagrange (1736–1813), professor of mathematics at Turin, Berlin, and Paris, discovered the powerful technique of introducing new variables, subject to additional constraints, into an equation to be solved.

$$Q(x) = (1/2)x^TMx - x^Tc, \tag{4.49}$$

has a minimum of

$$Q(M^{-1}c) = (-1/2)c^TM^{-1}c, \tag{4.50}$$

when $Mx = c$.

3. The vector $h = (A^TA)^{-1}A^Tb$ minimizes $\|Ah - b\|^2$.
4. The FIR filter $h(n) = [h(-p), h(-p+1), ..., \underline{h(0)}, ..., h(p)]$, given by the vector h of (3) is optimal for noise removal from signal $x(n)$, where matrix A in (3) is a set of noisy windows of $x(n)$, and b is the set of noise-free values of $x(n)$ corresponding to A's noisy sample sets.

Proof: To see statement 1, consider some vector x, and note that $x^T(A^TA)x = (Ax)^T(Ax) = \|Ax\|^2$. Since the columns of A are linearly independent, Ax, which is a linear combination of columns of A (the column space of A), cannot be 0 unless $x = 0$. Hence, A^TA is positive definite. Symmetry follows from the basic fact that $(AB)^T = B^TA^T$. For statement 2, observe that if $Mx = c$, then

$$Q(y) - Q(x) = (1/2)(y - x)^TM(y - x) \geq 0 \tag{4.51}$$

by the fact that M is positive definite; so $Q(y) \geq Q(x)$, and the minimum (4.50) follows. Turning to statement 3, now note that $\|Ah - b\|^2 = (Ah - b)^T(Ah - b) = x^TA^TAh - h^TA^Tb - b^TAh + b^Tb$. So minimizing $\|Ah - b\|^2$ is equivalent to minimizing $h^TA^TAh - h^TA^Tb - b^TAh$. Also, thinking of the 1×1 matrices here as real numbers, we find $h^TA^Tb = b^TAh$. So the minimization of $\|Ah - b\|^2$ reduces to a minimization of $Q(h) = (1/2)h^T(A^TA)h - h^TA^Tb$. But this is precisely (4.49) with $M = A^TA$ and $c = A^Tb$. Finally, statement 4 of the theorem simply expresses the equivalence of convolution using an FIR filter, $h(n)$, with vector dot products. Filter $h(n)$ is optimal in the sense that any other noise removal filter $g(n)$ will produce errors with respect to know values for $x(n)$ at $n = p$ that are larger in l^2 norm than those which $h(n)$ produces. ■

Example. Suppose that a signal $x(n)$ is corrupted with Gaussian noise, and we seek an optimal noise removal filter H with $(H\delta)(n) = h(n) = [h(-1), \underline{h(0)}, h(1)]$. We record a series of windows of width 3 of the input signal $x(n)$ around $n = 0$: $[a_{1,1}, a_{1,2}, a_{1,3}]$, $[a_{2,1}, a_{2,2}, a_{2,3}]$, ..., $[a_{m,1}, a_{m,2}, a_{m,3}]$. In this case, the normal equations become $Ah = b$, where, for example,

$$A^T = \begin{bmatrix} 1.12 & 0.94 & 0.77 & 0.96 & 1.83 & 0.52 & 0.91 \\ 0.86 & 1.31 & 0.94 & 1.19 & 2.15 & 0.70 & 1.03 \\ 1.19 & 2.72 & 0.32 & 0.65 & -0.04 & 0.89 & 1.21 \end{bmatrix}. \tag{4.52}$$

Upon solving the normal equations, we find $h(n) = [h(-1), \underline{h(0)}, h(1)] = [1.08, .92, 1.17]$, very close to the box or moving average filter to which our original intuition guided us. Now let us consider the possibilty that the noise on $x(n)$ is correlated; in particular, if we assume a correlation between the noise magnitude at $x(0)$ and that at $x(-1)$ and $x(1)$, then we arrive at normal equations with the following coefficient matrix

$$A^T = \begin{bmatrix} 1.12 & 0.98 & 0.85 & 0.90 & 0.73 & -0.11 & 1.13 \\ 0.86 & 1.24 & 1.32 & 1.23 & 1.07 & 0.89 & 1.20 \\ 1.19 & 1.69 & 0.79 & 0.97 & 1.01 & 0.35 & 1.03 \end{bmatrix}. \tag{4.53}$$

The solution to the normal equations provides an estimate $h(n) = [h(-1), \underline{h(0)}, h(1)] = [0.85, 1.12, 0.93]$, emphasizing the central filter value. Thus it is the correlation between successive noise values within the input signal $x(n)$ that governs the departure of the optimal filter from the moving average filter.

It is possible to adopt a different least-squares approach for optimal noise removal. For example, a popular approach [49] is to fit quadratic (or higher-order) polynomials to the signal values to obtain families of filters of varying support. There result filters that smooth signals, but do not resemble the simple box or Gaussian filters at all. Table 4.4 shows representative finitely supported smoothing filters that derive from this procedure.

Now let's consider a constrained approach to optimal noise removal filter design. Again, we seek an LTI system H with an impulse response $h = H\delta$, of width $M = 2p + 1$, supported on $[m-p, m+p]$. Again we ask what $h(n)$ must then be to optimally remove noise from known signal $x(n)$. Since we are concerned once more with an FIR filter and a noisy source signal $x(n)$, we may view both as row vectors. We view the system output $y = Hx$ as an estimator of the random vector x which has expected value vector $\mu b = E[x]$, where b is the M-dimensional row vector of all ones. In vector terms we seek a row vector h such that $\hat{\mu} = \langle h, x\rangle$ is an estimator of $y(m) = (Hx)(m)$. Here, $\langle\ ,\ \rangle$ is the vector inner (dot) product operation. We desire the random variable $\hat{\mu}$ to be unbiased, so that its mean on $[m-p, m+p]$ is the same as $x(n)$'s value on this interval. In other words, $E[\hat{\mu}] = \mu$. This condition implies the following condition on impulse response h:

TABLE 4.4. Smoothing Filters Derived from Polynomial Fits to Signal Values[a]

Points	Quadratic Polynomial Fit:	Quartic Polynomial Fit:
5	[..., $\underline{.4857}$, .3429, −.0857]	[..., $\underline{1.500}$, −.5000, .2500]
7	[..., $\underline{.3333}$, .2857, .1429, −.0952]	[..., $\underline{.5671}$, .3247, −.1299, .0216]
9	[..., $\underline{.2554}$, .2338, .1688, .0606, −.0909]	[..., $\underline{.4172}$, .3147, .0699, −.1282, .0350]
11	[..., $\underline{.2075}$, .1958, .1608, .1026, .0210, −.0839]	[..., $\underline{.3333}$, .2797, .1399, −.0233, −.1049, .0420]

[a]In the table, all of the impulse responses are even, and only the values for non-negative time instants are shown. Such noise removal filters are popular for smoothing data acquired from laboratory instruments [49].

Proposition (Unbiased Estimator). If the estimator $\hat{\mu}$ is unbiased, then

$$\sum_{n=m-p}^{m+p} h(n) = 1. \tag{4.54}$$

Proof: Since $E[\hat{\mu}] = \mu$, we have $E[\hat{\mu}] = E[\langle h, x\rangle] = \langle h, E[x]\rangle = \mu\langle h, b\rangle$, and therefore $\langle h, b\rangle = 1$. In terms of the impulse response of system H, this is (4.54); the proof is complete. ■

Let us continue this notational framework for the next two results. If we require the further condition that the estimator $\hat{\mu}$ have minimal variance, the following holds:

Proposition (Variance of Unbiased Estimator). If the estimator $\hat{\mu}$ is unbiased, then $\mathrm{Var}[\hat{\mu}] = \langle h, h\Sigma\rangle$, where $\Sigma = \mathrm{E}[(x - \mu b), (x - \mu b)]$ is the covariance matrix of the random vector x.

Proof: We calculate

$$\begin{aligned}\mathrm{Var}[\hat{\mu}] &= E[(\hat{\mu} - E[\hat{\mu}])^2] = E[(\langle h, x\rangle - \mu)^2] = E[(\langle h, x\rangle - \langle h, \mu b\rangle)^2]\\ &= E[\langle h, h\langle(x - \mu b), (x - \mu b)\rangle\rangle] = \langle h, hE[(x - \mu b), (x - \mu b)]\rangle = \langle h, h\Sigma\rangle. \end{aligned} \tag{4.55}$$

■

Now we wish to minimize $\mathrm{Var}[\hat{\mu}]$ subject to the constraint of the first Unbiased Estimator Proposition. The next theorem solves this is typical Lagrange multipliers problem.

Theorem (Unbiased Minimal Variance Estimator). If the estimator $\hat{\mu}$ is unbiased and has minimal variance, then

$$\mathbf{h} = \left(\frac{1}{\left\langle \mathbf{b}\Sigma^{-1}, \mathbf{b}\right\rangle}\right)\mathbf{b}\Sigma^{-1}. \tag{4.56}$$

Proof: To apply Lagrange multipliers to the optimization problem, let us introduce a function, $L(h)$, with an additional parameter, λ:

$$L(h) = \mathrm{Var}[\hat{\mu}] + \lambda(\langle h, b\rangle - 1) = \langle h, h\Sigma\rangle + \lambda(\langle h, b\rangle - 1). \tag{4.57}$$

Recall that the impulse response of the optimal filter we seek, $h(n)$, was supported on $[m-p, m+p]$. So the vector h is a $1 \times M = 1 \times (2p+1)$ row vector: $h = (h_1, h_2, \ldots, h_M)$. We next calculate the partial derivatives:

$$\left(\frac{\partial L}{\partial h_1}, \frac{\partial L}{\partial h_2} \quad \cdots \quad \frac{\partial L}{\partial h_M} \right) = 2\mathbf{h}\Sigma + \lambda \mathbf{b}. \tag{4.58}$$

Since the partial derivatives (4.58) are zero where $L(h)$ has a minimum, we solve $0 = 2h\Sigma + \lambda b$ for the vector h:

$$\mathbf{h} = -\left(\frac{\lambda}{2}\right)\mathbf{b}\Sigma^{-1}. \tag{4.59}$$

Taking the inner product of both sides of (4.59) with the all-ones vector b and applying the constraint $\langle h, b \rangle = 1$ gives

$$1 = \langle \mathbf{h}, \mathbf{b} \rangle = -\left(\frac{\lambda}{2}\right)\langle \mathbf{b}\Sigma^{-1}, \mathbf{b}. \rangle \tag{4.60}$$

and

$$\lambda = \frac{-2}{\langle \mathbf{b}\Sigma^{-1}, \mathbf{b} \rangle}. \tag{4.61}$$

Finally, substituting (4.61) into (4.59) gets us to (4.56), and the proof is complete.

■

The determination of the optimal filter $h(n)$ therefore depends entirely on the covariance matrix Σ.

Example. Suppose we seek a filter supported on $[-1, 1]$, and the random noise embedded in input signal $x(n)$ is uncorrelated. If the variances at $x(-1)$ and $x(1)$ are equal, say they are $\alpha\sigma^2$, where σ^2 is the variance of $x(0)$, then we have the following:

$$\Sigma = \sigma^2 \begin{bmatrix} \alpha & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \alpha \end{bmatrix}, \tag{4.62}$$

$$h = \frac{1}{1+2\alpha^{-1}} \begin{bmatrix} \alpha^{-1} \\ 1 \\ \alpha^{-1} \end{bmatrix}. \tag{4.63}$$

Example. Suppose that a filter with support on [−1, 1] is again necessary, but that the random noise is correlated. Suppose that values one time instant apart have correlation ρ and values two time instants apart have correlation ρ^2. Then we have the following

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & \rho & \rho^2 \\ \rho & 1 & \rho \\ \rho^2 & \rho & 1 \end{bmatrix}, \tag{4.64}$$

$$h = \frac{1}{(1-\rho)(3-\rho)} \begin{bmatrix} (1-\rho) \\ (1-\rho)^2 \\ (1-\rho) \end{bmatrix}. \tag{4.65}$$

The above development of optimal noise removal filters using Lagrange multipliers is usually presented for designing neighborhood smoothing operators in image processing [51]. We have adapted it here for one-dimensional filters. One consequence of this is that the covariance matrices (which are $M \times M$ rather than $M^2 \times M^2$, where M is the filter size) are much easier to invert.

There are some situations where convolutional filtering does not do a good job at enhancement before thresholding and segmentation. The next section considers such cases and develops some simple nonlinear smoothing filters which have proven to be very powerful and very popular.

4.4.3 Nonlinear Filters

Filtering with LTI systems may well result in an incorrect segmentation in those situations where signals contain transient phenomena, such as sharp, high-magnitude spikes. From simple faith in the principal theorems of probability, such as the Central Limit Theorem, we tend to dismiss the possibility of very high magnitude spikes in our data. After all, after repeated trials the sum of any distributions will tend toward a Gaussian distribution. Some 68.26% of the signal values should lie within one standard deviation, σ, of the signal mean, μ. Over 95% of the values should be less than 2σ from μ. And huge values should be extremely rare, using this logic. But this reasoning is flawed. Such transients may occur to severe, short-duration noise in the process that generates the source signal or result from imperfections in the signal acquisition apparatus.

The reason LTI filters become problematic is that for a very large impulse, the output of the smoothing signal $y(n)$ resembles the filter's impulse response. The linearity is the problem. Another problem with linear smoothing filters is their tendency to blur the sharp edges of a signal. Good noise removal requires a filter with a wide support. But this erodes sharp edges, too. Thus, when the thresholding operation is applied, the regions labeled as meaningful signal may be smaller in extent than they would be without the preliminary filtering. Perhaps separate regions blend together, rendering the final form of the signal unrecognizable by the classification step in the application. The problem here is the wide support of the filter impulse response.

Two types of nonlinear filters are widely used for impulse noise removal: median and morphological filters. Properly chosen, these filter a noisy signal without leaving a large-magnitude response that interferes with the subsequent thresholding operation. Like convolutional filters, median and morphological noise removal are handy before thresholding. By dint of their nonlinearity, however, they can preserve the binary nature of a thresholded signal; they can also be used to clean out the speckle noise that often persists in a signal after thresholding.

4.4.3.1 Median Filters. A discrete median filter, H, accepts a signal $x(n)$ as input and produces $y(n)$, which is the median value of $x(n)$ in a neighborhood around $x(n)$.

Definition (Median Filter). If N is a positive integer, then the median filter $y = Hx$ for the neighborhood of width $2N+1$ is defined by $y(n) = \text{median}\{x(m) \mid n-M \le m \le n+N\}$.

To compute a median filter, the values $x(n-N)$, $x(n-N+1)$, ... , $x(n+N)$ must be sorted and the middle element selected. It is possible, and maybe useful, to have asymmetric median filters as well as to take $y(n)$ to be the median of a set that does not contain $x(n)$; nevertheless, the centered filter is the most common.

As an enhancement tool, the median filter is usually an early signal processing stage in an application, and most applications use it to improve segmentation operations that depend on some kind of thresholding. We have noted that convolutional filters tend to smear signal edges and fail to fully eradicate sharp transient noise. The main points to keep in mind about the median filter as an enhancement tool are as follows:

- It removes impulse noise.
- It smoothes the signal.
- The median can be taken over a wider interval so that the filter removes transients longer than a time instant in duration.
- Median filters preserve sharp edges.

In addition, since median filters involve no floating point operations, they are well-suited for implementation on fixed-point digital signal processors (which are much faster than their floating-point capable cousins) and in real-time applications [53]. An alternative to using wide median filters is to apply a small support median filter many times in succession. Eventually this reduces the original signal to the median root, which is no longer affected by further passes through the same filter [54].

4.4.3.2 Morphological Filters. From the very beginning of this book, all of the signal operations have been algebraic—additive, multiplicative, or based upon some extension thereof. To conceptualize analog signals, we freely adopted real- and complex-valued functions of a real variable. To conceptualize discrete signals, we made them into vectors-without-end, as it were. Analog signal operations and analog system theory derived straight from the theory of functions of a real variable. And vector operations inspired discrete signal manipulations, too. Thus, analog and discrete systems added signals, multiplied signals by a scalar value, convolved them, and so on. This is not at all surprising. Signals are time-ordered numerical values, and nothing is more natural for numbers than arithmetic operations. But there is another, distinctly different, and altogether attractive viewpoint for all of signal processing and analysis: mathematical morphology.

Mathematical morphology, or simply morphology, relies on set-theoretic operations rather than algebraic operations on signals. Morphology first casts the concept of a signal into a set theoretic framework. Then, instead of addition and multiplication of signal values, it extends and contracts signals by union and intersection. The concept at the fore in morphology is shape, not magnitude.

It ought to be easier to understand the morphological perspective by thinking for a moment in terms of binary images in the plane, instead of signals. A binary image is a map I: $\mathbb{R} \times \mathbb{R} \to \{0, 1\}$. The binary image, I, and the subset of the plane, $A = I^{-1}(\{1\}) = \{(s, t) \in \mathbb{R} \times \mathbb{R} \mid I(s, t) = 1\}$ mutually define one another. (The image is precisely the characteristic function of the set A.) So we can think of a binary image as either a function or a set. Now suppose we have a small circle $C = \{(s, t) \mid s^2 + t^2 \leq r\}$, with $r > 0$. Suppose we place the center of C on top of a boundary point of $(a, b) \in A$; we find all of the points, (r, s) covered by C, $B = \{(r, s) \mid (r-a)^2 + (s-b)^2 \leq r^2\}$; and we set $A' = A \cup B$. We continue this placement, covering, and union operation for all of the boundary points of A. The result is the set A with a shape change effected for it by the morphological shaping element, B. This scheme works just as well for binary-valued signals: An analog binary signal is the characteristic function of a subset of the real line, and a discrete binary signal is the characteristic function of a subset of the integers.

Let's begin our brief tour of mathematical morphology with the definitions and basic properties for binary morphological operations. This will introduce the terminology and concepts as well as provide a theoretical foundation for our later extension to digital signals. Just as we can identify a binary image with a binary signal, we can also characterize a binary discrete signal with a subset of the integers or a binary analog signal with a subset of the real line. The signal is the characteristic function of the set. Although morphologists work their craft on either $\mathbb{Z}^n$ or $\mathbb{R}^n$ [55–57], we shall consider digital signals and restrict our morphological investigation to discrete n-space, $\mathbb{Z}^2$. Note that the development could proceed as well for n-tuples from an abstract normed linear space.

Definition (Translate of a Set). Let $A \subseteq \mathbb{Z}^2$ and $k \in \mathbb{Z}^2$. Then the translate of A by k is $A_k = A + k = \{a + k \mid a \in A\}$.

Definition (Dilation). Let $A, B \subseteq \mathbb{Z}^2$. Then the dilation of A by B is

$$A \oplus B = \bigcup_{b \in B} (A + \mathbf{b}). \tag{4.66}$$

The set B in (4.66) is called a structuring element; in this particular instance, it is a dilation kernel.

From this meager beginning, we see the development of diverse, interesting, useful properties.

Proposition (Dilation Properties). Let A, B, $C \subseteq \mathbb{Z}^2$ and $k \in \mathbb{Z}$. Then, $A \oplus B = \{a + b \mid a \in A \text{ and } b \in B\}$. Moreover, dilation is commutative, $A \oplus B = B \oplus A$; associative, $(A \oplus B) \oplus C = A \oplus (B \oplus C)$; translation invariant, $(A + k) \oplus B = (A \oplus B) + k$; increasing: if $A \subseteq B$, then $A \oplus D \subseteq B \oplus D$; distributive over unions: $A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$.

Proof: Let's show translation invariance. We need to show that the two sets, $(A + k) \oplus B$ and $(A \oplus B) + k$ are equal; thus, we need to show they contain the same elements, or, equivalently, that they are mutual subsets. If $x \in (A + k) \oplus B$, then there is $a \in A$ and $b \in B$ such that $x = (a + k) + b = (a + b) + k$. Since $a + b \in A \oplus B$, we know $(a + b) + k = x \in (A \oplus B) + k$. Thus, $(A + k) \oplus B \subseteq (A \oplus B) + k$. Now let's consider an element $x \in (A \oplus B) + k$. It must be of the form $y + k$, for some $y \in A \oplus B$, by the definition of the translate of $(A \oplus B)$. Again, there must be $a \in A$ and $b \in B$ such that $y = a + b$. This means $x = (a + b) + k = (a + k) + b \in (A + k) \oplus B$. Thus, $(A \oplus B) + k \subseteq (A + k) \oplus B$. Since these two are mutual subsets, the proof of translation invariance is complete. The other properties are similar, and we leave them as exercises. ■

Notice that the distributivity property implies that dilations by large structuring elements can be broken down into a union of dilations by smaller kernels.

How can dilation serve as a noise removal operation within a digital signal analysis application? To answer this question we adopt a set-theoretic description for discrete signals, $x(n)$, based on the graph of the signal, $\text{Graph}(x) = \{(n, x(n)) \mid n \in \mathbb{Z}\}$. Graph($x$) is a subset of the discrete plane, so it describes x(n) in set theory terms, but we need other tools to adequately define signal dilation. Our exposition draws the concepts of umbra and top surface from the tutorial of Haralick et al. [55].

Definition (Umbra). Let $x(n)$ be a digital signal, $x: \mathbb{Z} \rightarrow [0, N]$, for some natural number $N \geq 0$. Then the umbra of x is $\text{Umbra}(x) = \{(n, m) \in \mathbb{Z} \times [0, N] \mid m \leq x(n)\}$.

Definition (Top Surface). If $A \subseteq \mathbb{Z}^2$ and $B = \{b \in \mathbb{Z} \mid (b, k) \in A, \text{ for some } k \in \mathbb{Z}\}$, then the top surface for A is the function, $T[A]$, that has domain B and is defined $T[A](b) = \max \{k \in \mathbb{Z} \mid (b, k) \in A, \text{ for some } k \in \mathbb{Z}\}$.

Thus, the umbra of a digital signal $x(n)$ is the planar set consisting of Graph(x) and all the points beneath it. The umbra operation turns a signal into a set, and the top surface operation turns a set into a function. Clearly, if $x(n)$ is a digital signal, $x: \mathbb{Z} \rightarrow [0, N]$, for some natural number $N \geq 0$, then $x = T[\text{Umbra}(x)] = T[\text{Graph}(x)]$. The next definition formally captures the idea of digital signal dilation.

Definition (Signal Dilation). Let $x(n)$ and $h(n)$ be digital signals, $x: \mathbb{Z} \rightarrow [0, N]$ and $h: \mathbb{Z} \rightarrow [0, N]$ for some natural number $N \geq 0$. Then the dilation of f by h, is $f \oplus h = T[\text{Umbra}(f) \oplus \text{Umbra}(h)]$.

Dilation is useful for removing transient signal features, such as splintering, when a signal contains a sharp gap within the overall blob-like structure. This can occur due to noise in the signal acquisition equipment. It may even be a genuine aspect of the signal, but its presence disturbs subsequent analysis algorithms. From our earlier examples of electrocardiography, recall that one occasional characteristic of electrocardiograms (ECGs) is such splintering. Its presence should not only be detected, but for purposes of heart rate monitoring, the application should not misread the ECG's splinters as individual ventricular contractions. Linear smoothing spreads the splinter into a large canyon in the signal. Using a dilation kernel on the umbra of the signal erases the splintering, although it produces a signal with an overall higher mean. If the dilation kernel is not as wide as the gap, then it has an insignificant enhancement effect. For digital signals, this decreases the effective dynamic range of the signal. However, rudimentary dilation kernels do a good job of filling gaps in signal structures.

There is a dual operation to dilation—called erosion; the dilation of A by B is equivalent to erosion of the complement, $A^c = \mathbb{Z}^2 \setminus A = \{b \in \mathbb{Z}^2 \mid b \notin A\}$, by the reflection of B across the origin.

Definition (Erosion). Let $A, B \subseteq \mathbb{E}^n$. Then the erosion of A by B is

$$A \ominus B = \bigcap_{b \in B} (A - b). \tag{4.67}$$

The structuring element B is also called an erosion kernel. The next proposition collects some properties of erosion. Note the symmetry between these properties and those in the Dilation Properties Proposition.

Proposition (Erosion Properties). If $A, B, C \subseteq \mathbb{Z}^2$, and $k \in \mathbb{Z}$, then, $A \ominus B = \{d \in \mathbb{Z}^2 \mid d + b \in A \text{ for all } b \in B\} = \{d \in \mathbb{Z}^2 \mid B_d \subseteq A\}$; $(A + k) \ominus B = (A \ominus B) + k$; $A \ominus (B + k) = (A \ominus B) - k$; $A \subseteq B$ implies $A \ominus C \subseteq B \ominus C$; $(A \cap B) \ominus C = (A \ominus B) \cap (B \ominus C)$; and, finally, $(A \ominus B)^c = A^c \oplus (-B)$, where $-B = \{-k \mid k \in B\}$.

Proof: Exercises. ■

The last of the Erosion Properties illustrates the dual nature of dilation and erosion. Note that erosion is neither commutative nor associative. Note also that translation invariance does not hold true for a translated erosion kernel. Let's continue with a definition of signal erosion and its application to enhancement.

Definition (Signal Erosion). Let $x(n)$ and $h(n)$ be digital signals, $x: \mathbb{Z} \rightarrow [0, N]$ and $h: \mathbb{Z} \rightarrow [0, N]$ for some natural number $N \geq 0$. Then the dilation of f by h, is $f \ominus h = T[\text{Umbra}(f) \ominus \text{Umbra}(h)]$.

Erosion can remove spike transients. Such transients present difficulties for linear enhancement filters. The convolution operation tends to blur the spike. Linear smoothing superimposes a replica of its kernel's impulse response on the signal and can obliterate small local features in the process. An erosion kernel removes the spike without spreading it into the rest of the signal's values. If the kernel is narrower than the transient, then it treats the transient as a blob, part of the main signal structure.

The following points summarize the behavior of dilation and erosion enhancement operators:

- (Splinter removal) Dilation blends signal structures separated by gaps narrower than the kernel width.
- (Spike removal) Erosion removes sharp, narrow, upward transients in the signal.
- Dilation has the undesirable effects of adding to the overall signal level and creating new, fine-scale signal features.
- Erosion has the undesirable effects of reducing the overall signal level and destroying existing, fine-scale signal features.

The drawbacks for both dilation and erosion as enhancement tools seem to be counterposed. They modify the signal mean in opposite ways. The have opposite effects on small signal features. These observations have led morphologists to compose dilation and erosion operations while using the same kernel. The operations are not inverses, and it turns out that this composition benefits later analysis steps, such as histogram derivation, thresholding, labeling, and region merging and splitting operations.

Definition (Opening). If $A, B \subseteq \mathbb{Z}^2$, then the opening of A by structuring element B is $A \circ B = (A \ominus B) \oplus B$. If $x(n)$ and $h(n)$ are digital signals, x: $\mathbb{Z} \to [0, N]$ and h: $\mathbb{Z} \to [0, N]$ for some natural number $N \geq 0$, then the opening of f by h, is $f \circ h = (f \ominus h) \oplus h$.

Definition (Closing). If $A, B \subseteq \mathbb{Z}^2$, then the closing of A by structuring element B is $A \bullet B = (A \oplus B) \ominus B$. If $x(n)$ and $h(n)$ are digital signals, x: $\mathbb{Z} \to [0, N]$ and h: $\mathbb{Z} \to [0, N]$ for some natural number $N \geq 0$, then the closing of f by h, is $f \bullet h = (f \oplus h) \ominus h$.

4.5 EDGE DETECTION

The edges of a signal mark the significant changes of its values, and edge detection is the process of determining the presence or absence of such significant changes. This is not a very satisfying definition, of course, since there is much room for disagreement over what makes a change in a signal significant or insignificant. It is most often the nature of the edge detection application that resolves such disputes. As we have already noted in segmentation problems, goal-directed considerations play a considerable role in designing edge detectors. Not all of the operations involved can proceed directly from the signal data to the edge detection result without some overall perspective on what the problem under study requires for a correct result. Thus, in edge detection there is again an interaction between bottom-up (or data-driven) approaches and top-down (or goal-driven) methods. Whatever the tools employed—edge detectors, texture analysis, local frequency components, scale-based procedures—this interplay between data-driven and goal-driven methods will continue to be at work in our signal interpretation efforts.

Despite the apparent simplicity of formulating the problem, edge detection is quite difficult. Among the first attempts to analyze signals and images were edge detection techniques. We will explore these methods in some depth, and this study will bring us up to an understanding of the current research directions, performance issues, and debates surrounding derivative-based edge finding. A second main approach is to fit members of a collection of edge models to a signal. Note that edge models are simply primitive shapes to be found in a signal. Since, as numerical analysts well know, finding an approximation to the derivative of a quantized function is a problematic undertaking, many researchers regard this approach as inherently more robust. Let us look a little further at these two approaches and their advocates.

Remembering that our signals are just one-dimensional functions of a discrete or continuous independent variable, we can take the magnitude of the signal derivative as a starting point for building an edge detector. Furthermore, since the second derivative changes sign over the extent of an edge, it is feasible to base an edge detector on approximations of the second derivative of a signal. Many of the early experiments in edge detection relied upon derivative-based approaches [58–62].

Another approach to edge detection is to fit selected edge-shaped masks or patterns to the signal. The edge detection patterns are signals themselves and (reasonably assuming that we are working with finite-energy signals) are elements of a Hilbert space. For a conceptualization of the edge detection problem, we can resort to our Hilbert space theory from Chapters 2 and 3. The edge detection operators can be orthonormalized. Thus, the edge detection masks become the basis elements $\{e_i: i \in I\}$ of a subspace of linear combinations of perfect edge-containing signals. The whole edge detection problem becomes one of finding the inner products of a signal $x(t)$ with edge detector basis elements $e_i(t)$. Notice, however, that the basis elements are not smoothly undulating functions such as the sinusoids or exponentials as used in the discrete Fourier transform (e.g., Section 4.3.2). Rather, the basis elements contain sharp discontinuities in the first- or higher-order derivatives. These sharp breaks in the basis functions match the shape of the edges to be detected. Thus, large values of $\langle x(t), e_i(t - t_0)\rangle$ indicate the presence of an edge of type i at location t_0 in $x(t)$. This too was a popular path of edge detection pioneers [61, 63–65]. Again, the edge basis elements are simply templates that contain elementary shapes, and this method, therefore, can be considered as a type of template matching. Section 4.6 of this chapter furnishes some basic template matching tools; the inner product operation lies at the heart of such methods.

These references are but a few of those that have appeared in the research literature over the last 30 years. Even at this writing, near the turn of the twenty-first century, investigators twist the interpretation of what is significant in a signal, adjust the mix of top-down and bottom-up goals, optimize in yet another way, and append a new variant to the vast edge detection literature [66–69]. Indeed, in this book we will continue to remark upon and deepen our understanding of edge detection with each new signal processing and analysis tool we develop.

Edge detection is a basic step in signal analysis. The output of an edge detector is a list of the location, quality, and type of edges in a signal. Interpretation algorithms could be designed to work on the list of edges. For an example from speech analysis, between certain pairs of edges, and based on the signal values therein, the interpretation could be that this portion of the signal represents irrelevant background noise. Other parts of the voice signal, bounded by edges, represent human speech. Further analysis may reveal that one significant section is a voiced consonant, another a fricative, and so on. In general, we find the edges in signals as a first step in analysis because the edges mark the transition between important and unimportant parts of a signal or between one and another significant part of a signal. Edge detection draws the boundaries within which later, more intricate algorithms must work. Edge guidelines allow interpretation processes to work in parallel on distinct parts of the signal domain. Finally, preliminary edge detection prevents the waste of processing time by higher-level algorithms on signal fragments that contain no useful information.

4.5.1 Edge Detection on a Simple Step Edge

To introduce the many problems that arise—even in the most elementary detection problem—let us consider a simple step edge in a signal. The unit step signal provides a natural example of a step edge. For discrete signal analysis, $u(n)$, the discrete unit step is a perfect step edge located at $n = 0$. What do we expect that a step edge detector should look like?

This much is obvious: the step edge we consider may not have unit height, and it may not be located at the origin. If the amplitude of the edge diminishes, then it is a less pronounced change in the signal, and our edge detector ought to produce a smaller response. Thus, a first consideration is that our edge detector should be linear: Its output is greater, given a larger amplitude in the edge and therefore a larger amplitude in the underlying signal. The second consideration is that an edge should be detected by the operator wherever it happens to be located within the signal. In other words, whether the step is at the origin or not is irrelevant: We should detect the edge at any time. The implication of our second consideration is that a first-cut edge detector is translation invariant. It is thus linear and translation invariant and, by the results of Chapters 2 and 3, must be convolutional.

Let us suppose that we are working with discrete signals. If the edge detector is $y = Hx$, then the output y is given by convolution with the impulse response, $h(n)$, of the detector system, H:

$$y(n) = (x * h)(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k). \tag{4.68}$$

There may be noise present in the signal $x(n)$ so that the sum in (4.68) does not readily simplify and the response is irregular. Noisy or not, the response of the system H to input $x(n)$ will be a signal $y(n)$ with discrete values:

$$\{..., y(-3), y(-2), y(-1), y(0), y(1), y(2), y(3), ...\}. \tag{4.69}$$

To find the edge, then, we have to threshold the output, supposing that an edge is present in $x(n)$ whenever $|y(n)| \geq T$. If we are very lucky, then there will be only one value that exceeds the threshold, along with a single detection result for a single step edge. If there is a range of high response values, then it may be necessary to select the maximum or a set of maximal responses.

In the case of an analog edge detection problem, the output of the convolution integral (4.70) will be continuous. There will be no hope of thresholding to discover a single detection result unless the threshold coincides with the maximum response value. We must be content with seeking a maximum or a set of maximal responses.

$$y(t) = (x * h)(t) = \int_{-\infty}^{+\infty} x(t-s)h(s)\,ds = \int_{-\infty}^{+\infty} h(t-s)x(s)\,ds. \tag{4.70}$$

It might seem that at this early stage in conceptualizing an edge detector that it is wrong to settle so quickly on a convolutional operator. For example, if the input signal is attenuated to $Ax(t)$, $|A| < 1$, then the response of the detector to the attenuation will be $y(t) = A(x{*}h)(t)$. The same edges—no matter how insignificant they become because of the attenuation—will still be detected as maximal responses due to the linearity of the operator H. So if the goals of our application change, and it becomes necessary to ignore sufficiently small changes in the input signal, then our edge detector will fail by falsely indicating significant transitions in $x(t)$. Sometimes top-down specifications demand non-linearity. Note, however, that we can accommodate such a top-down scheme by adjusting the threshold parameter. If certain jumps should be detected as edges and others should be passed, then the kind of nonlinearity we need can be implemented by thresholding all responses and then selecting the maximal response from those that exceed the threshold. In other words, if Edges[x] is a predicate or list of edges of input signal $x(t)$, then Edges[x] = $\{(t, x(t)) : |y(t)| \geq T$ and t is a local maximum of $y = x{*}h\}$.

There is a further difficulty with uncritical convolutional edge detection. If the input signal contains spike noise, then the output of the edge operator will have a large response to the spike. In fact, around an isolated impulse, the response will tend to look like the impulse response of the edge detector itself.

Example (Difference of Boxes Operator). One of the earliest edge detectors is the Difference of Boxes (DOB) filter [60], defined by $y = h_{\mathrm{DOB}}{*}x$, where, for some $L > 0$,

$$h_{\mathrm{DOB}}(t) = \begin{cases} -1, & -L \leq t < 0, \\ 1, & 0 \leq t \leq L. \end{cases} \tag{4.71}$$

Clearly, the DOB operator is designed to smooth the data on both sides of a possible edge and subtract the earlier values from the later ones. Any sudden transition at time $t = 0$ appears in the output as a large-magnitude value. The DOB operator emphasizes values near $t = 0$ just as much as values near $|t| = L$. It is also typical to zero the center value of the DOB operator, $h_{\mathrm{DOB}}(0) = 0$.

Example (Derivative of Gaussian Operator). Another popular operator for convolutional edge detection is the first derivative of the Gaussian (dG). The impulse response of this system is given by $h_{\mathrm{dG}}(t) = -t\exp[-(t/\sigma)^2/2]$. The dG operator is an alternative to the DOB convolution kernel; it emphasizes values near the edge over values further away. A disadvantage is that it has an infinite extent. Practically, of course, the kernel can be truncated, since the signal diminishes rapidly.

We can add some normally distributed noise to the analog unit step signal to produce noisy step edges. Convolution with DOB and dG operators produces satisfactory results (Figure 4.11).

Application (Plasma Etch Endpoint Detection). Consider an example from a problem of controlling an industrial process, Figure 4.2. A spectrometer monitors the carbon monoxide (CO) optical discharge from a plasma reactor that is etching an oxide layer on a silicon wafer [70]. This is a critical step in the production of integrated circuit chips. The CO is a chemical byproduct of the plasma reaction with the silicon dioxide on the wafer. When the CO spectral line falls abruptly, the layer of silicon dioxide that is exposed to the plasma has been cleared by the plasma reaction.

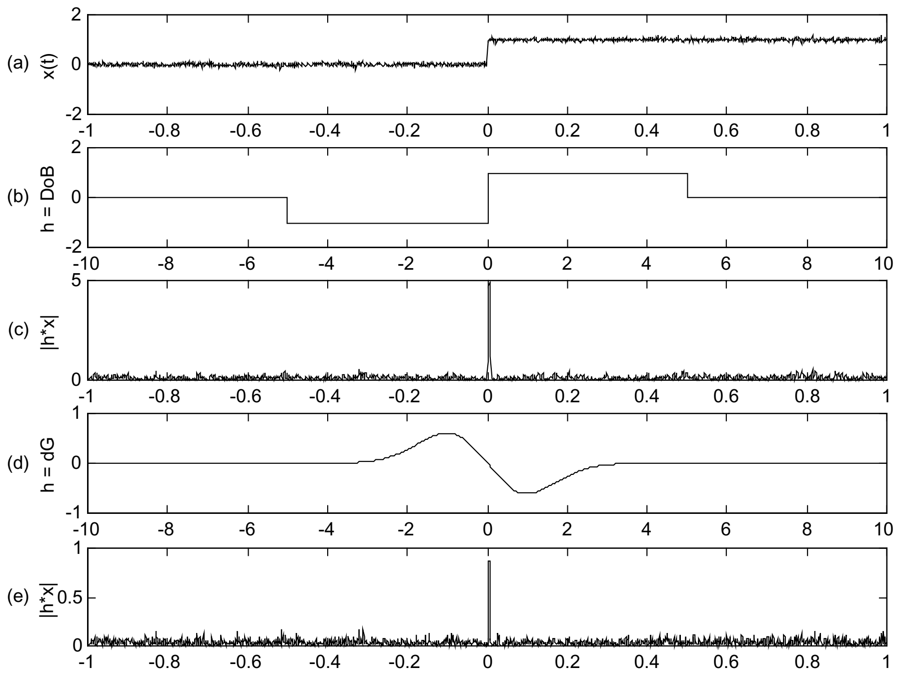


Fig. 4.11. An analog step edge in noise (a). The DOB and dG operators are shown in (b) and (d), respectively. Panels (c) and (e) show the result of convolving the signal with the DOB and dG operators, respectively.

It is essential to stop the plasma reaction as close as possible to the time when the CO intensity falls to avoid plasma damage to circuit layers underneath the target oxide. Thus, plasma etch endpoint detection can be largely viewed as a real-time edge detection problem. Plasma processes are very much subject to transients in the optical emissions. These can be caused by a reactor or acquisition fault or process instability. Transients can result in anomalous edge detection and an incorrect identification of etch process endpoint. As we found in Section 4.3.3, one the solution is to introduce a nonlinear filter that removes impulse noise prior to the linear filtering and application of a convolutional edge detection operator [e.g., the DOB operator (4.71)].

Most plasma etch endpoint detection algorithms rely on simple signal thresholding and first- and second-derivative estimates on the optical emission signal. When variability arises in the endpoint trace, perhaps due to imprecise control of previous thin film deposition steps or to the presence of different circuit configurations on the etched wafers, then strategies that consider the shape of the optical emission signal become attractive [71]. We will consider this aspect later in Section 4.7. In other processes, the signal edge is more difficult to discern, and endpoint detection methods that model the real-time optical emission and detect out-of-trend conditions become important [72]. Lastly, there is the top-down consideration that complete translation invariance is not appropriate for the edge detector. In fact, the process control application of Figure 4.2 is such a case. Typically, it is known in advance, by the process's nature, that the edge cannot occur except within a certain time range. In the case of plasma etching reactor control, it is common to specify an initiation of etch period during which no collapse in the emission spectral line should be detected. The upshot is that the convolutional operator is only applied within in the region known to contain valid edges. An alternative is to run the edge detector over the entire domain of the input signal and, in a further post-processing step, delete edges that occur outside the appropriate interval as spurious.

The next proposition sheds some further light on the nature of convolutional edge detectors [66]. We state and prove this proposition for analog signals and leave the discrete case for an exercise.

Proposition (Maximal Response Edge Detector). If a step edge detection operator locates edges from the maximal responses of a convolutional operator, *H*, and provides a unique maximum for $u(t)$ at $t = 0$, then

(i) $h(t)$ can only have one zero, at $t = 0$.
(ii) The impulse response of system H, $h(t)$, must be odd.

Proof: The first part of the proposition follows from the fundamental theorem of calculus. Since

$$y(t) = \int_{-\infty}^{+\infty} u(t-s)h(s)\,ds = \int_{-\infty}^{t} h(s)\,ds, \tag{4.72}$$

we know that $dy/dt = h(t)$. Since the response $y(t)$ has a maximum at $t = 0$, its derivative must be zero; therefore $h(0) = 0$. If the detector, H, responds correctly to the perfect step edge of $u(t)$ at $t = 0$, then this can be the only maximal point of the response, and it follows that $h(t)$ has precisely one zero.

For (ii) we can decompose $h(t)$ into its even and odd parts and then apply the fundamental theorem of calculus to get the result. Let $h(t) = h_e(t) + h_o(t)$ where $h_e(t)$ is even and $h_o(t)$ is odd. Applying the convolution theorem for analog LTI systems, let $t > 0$, and note that

$$y(t) = \int_{-\infty}^{+\infty} u(t-s)h(s)\,ds = \int_{-\infty}^{t} h(s)\,ds = \int_{-\infty}^{t} h_e(s)\,ds + \int_{-\infty}^{t} h_o(s)\,ds, \tag{4.73}$$

$$y(t) = \int_{-\infty}^{-t} h_o(s)\,ds + \int_{-t}^{0} h_o(s)\,ds + \int_{-\infty}^{-t} h_e(s)\,ds + \int_{-t}^{0} h_e(s)\,ds + \int_{0}^{t} h_o(s)\,ds + \int_{0}^{t} h_e(s)\,ds. \tag{4.74}$$

The finite integrals over $h_o(t)$ cancel, and the finite integrals over $h_e(t)$ are identical; thus,

$$y(t) = \int_{-\infty}^{-t} h_o(s)\,ds + \int_{-\infty}^{-t} h_e(s)\,ds + 2\int_{0}^{t} h_e(s)\,ds = \int_{-\infty}^{-t} h(s)\,ds + 2\int_{0}^{t} h_e(s)\,ds. \tag{4.75}$$

Now from (4.75) we may write

$$y(t) - y(-t) = 2\int_{0}^{t} h_e(s)\,ds. \tag{4.76}$$

Let us now differentiate (4.76), applying the chain rule to the left-hand side and the fundamental theorem to the right-hand side. This gives $2h_e(t) = 0$. Since t was arbitrary, $h_e(t) = 0$ for all $t > 0$. Moreover, since $h_e(t)$ is even, $h_e(t) = 0$ for all t. ■

4.5.2 Signal Derivatives and Edges

At first glance, finding the abrupt changes, or edges, in a signal amounts to finding the points at which the signal has a large derivative. Since we are familiar with analog differentiation from calculus and discrete differentiation from numerical analysis, it would seem that we should already possess the proper tools to begin a formal study of edge detection as a task within signal analysis. But the presence of even a minute amount of noise in a signal can lead to wild variations in its derivative at a time instant. Differentiation is known as a classical ill-posed problem or unstable process; in systems that perform differentiation, small differences in the input signal lead to large differences in the output signal [73]. The standard approach to such ill-posed problems is to convert them to well-posed problems by smoothing the input data. Thus, for signal analysis systems, some preliminary signal conditioning is appropriate.

We recall how to take the derivative of signals from calculus as a limit,

$$x'(t_0) = \left.\frac{dx}{dt}\right|_{t=t_0} = \lim_{t \to t_0} \frac{x(t) - x(t_0)}{t - t_0}. \tag{4.77}$$

This limit leads to standard formulas for differentiation of polynomials, algebraic functions, and trigonometic and exponential functions. Thus, we can obtain estimates for derivatives of discrete signals by finding an interpolating polynomial among a set of discrete vaues. The error estimates that this procedure gives are not as good as when the time-domain signal is approximated; this is a practical manifestation of the ill-posed nature of the differentiation process [74].

Deriving discrete derivatives for the analysis of digital signals by first developing an interpolating function, however, is not necessary for an approximate slope value near a point. We can derive discrete derivative formulas for a signal $x(n)$ by assuming that $x(n)$ arises from sampling an analog signal $x_a(t)$: $x(n) = x_a(nT)$, where T is the sampling interval. To get a formula for the discrete first and second derivatives around time instant $n = 0$, we expand $x_a(t)$ in a Taylor series and neglect the error terms:

$$x(1) = x(0) + x_a'(0) + \frac{1}{2}x_a''(0), \tag{4.78a}$$

$$x(-1) = x(0) - x_a'(0) + \frac{1}{2}x_a''(0). \tag{4.78b}$$

Subtracting (4.78b) from (4.78a) gives a simple formula for the first derivative:

$$x_a''(0) = \frac{1}{2}[x(1) - x(-1)]. \tag{4.79}$$

Adding (4.78a) from (4.78b) gives a simple formula for the second derivative:

$$x_a''(0) = x(1) - 2x(0) + x(-1). \tag{4.80}$$

Thus, the system $y = Hx$ with impulse response $h(n) = [1/2, \underline{0}, -1/2]$ approximates a discrete first derivative. And the system $y = Gx$ with impulse response $g(n) = [1, \underline{-2}, 1]$ approximates the second derivative. Recall that the system $y = Gx$ from Chapter 2 is called the discrete Laplacian operator. There we noted that it detected edges in signals, producing a zero response within flat and linearly sloped signal regions. Now we discover the discrete Laplacian's true nature: It is just a discrete version of the second derivative. The above analysis easily generalizes to approximate discrete derivatives with wider support. We find $x(2)$ and $x(-2)$ in terms of the Taylor series of $x_a(t)$, apply (4.79) and (4.80), and find that

$$x_a'(0) = \frac{1}{12}[x(-2) - 8x(-1) + 8x(1) - x(2)], \tag{4.81}$$

$$x_a''(0) = \frac{1}{12}[-x(-2) + 16x(-1) - 30x(0) + 16x(1) - x(2)]. \tag{4.82}$$

The above rudimentary edge detectors are intuitive, but they turn out to be less than ideal. To see this, we need an appropriate mathematical foundation. Thus, let's formally define signal edges and motivate the conditions on edge detectors that should constitute opimality. This will provide us with some initial results on optimal edge detectors.

Definition (Analog Signal Edge). Let $x(t)$ be an analog signal. Then $x(t)$ has an edge at $t = t_0$ if for some $n > 0$, there is a discontinuity in the nth derivative of $x(t)$. (We consider the 0th derivative of $x(t)$ to be the signal itself.)

It requires a little care to correctly define edges for a discrete signal. By their nature, discrete signals are composed entirely of discontinuities. So we adopt the following definition.

Definition (Discrete Signal Edge). Let $x(n)$ be a discrete signal. Then there is an edge in $x(n)$ at a point n_0 if there is a discontinuity in the derivative of the signal $x_a(t) = x(\lfloor t \rfloor) + (t - \lfloor t \rfloor)x(\lceil t \rceil) - x(\lfloor t \rfloor)$, where $\lfloor t \rfloor$ is the floor of t, and $\lceil t \rceil$ is the ceiling of t.

These definitions do encompass the variety of edge shapes that we find in signals. Of course, the step edge is a discontinuity in the signal itself. When the signal assumes a dihedral or "roof" shape, then the discontinuity lies in the first derivative. Thus, we shall concentrate on step edges in our theoretical development, since other edge shapes can be analyzed by computing the signal derivative and then applying our step edge analysis methods.

4.5.3 Conditions for Optimality

Let us now return to the simple step edge and complete an analysis of the edge detection problem. Specifically, three conditions for an edge detector H are as follows:

(i) The detector should have a high signal-to-noise ratio (SNR). This means that a better edge detector has a higher response to the edge within a signal than to the surrounding noise.

(ii) H should be well-localized about the true signal edge. The output maxima of H should cluster around the true edge, and better edge detectors show a tighter group of maximal responses.

(iii) There should be only one response to a single edge. If H is good, then it should present a number of apparently valid alternatives for the precise location, or registration, of the edge.

Canny chose the overall goals of (i)–(iii) for his detector design [75], which became the overwhelming favorite in research literature and signal analysis applications for many years. His strategy is to develop mathematical formulations for the goals and then to find convolution kernels that maximize the products of individual criteria.

Let's consider just the first two criteria above and see how this analysis unfolds. Let $h(t)$ be the convolution kernel that we seek and let $u(t)$ be the unit step. We will need to compute convolution integrals, $y(t) = (h * u)(t)$, so we must limit the time-domain extent of both $h(t)$ and $s(t)$, or assume that they belong to a signal space (Chapter 3) that supports closure under convolution. To achieve this, we assume that the edge detection filter has support on a finite interval $[-L, L]$. Now we define one form of the signal-to-noise ratio for analog systems and prove a property for the SNR under white Gaussian noise (Chapter 3).

Definition (Signal-to-Noise Ratio). Let $y = Hx$ be an analog system and let $n(t)$ be a noise signal. Then the signal-to-noise ratio (SNR) for H at $t = 0$ is

$$\mathrm{SNR}_{H,t=0} = \frac{|y(0)|}{\left(E\left[\int_{-\infty}^{+\infty} h(t)n(-t)\,dt\right]^2\right)^{\frac{1}{2}}}, \tag{4.83}$$

where $E[y]$ is the expectation operator.

That is, the SNR is a ratio of the system output without noise present to the system's expected output when the input is pure noise.

Proposition (SNR under White Gaussian Noise). Suppose H is an analog system and $n(t)$ is white Gaussian noise. Then $\mathrm{SNR}_{H,t=0} = |y(0)|/(n_0\|h\|_2)$, where n_0 is the standard deviation of $n(t)$.

Proof: From stochastic processes theory [76], we have that

$$E\left[\int_{-\infty}^{+\infty} h(t)n(-t)\,dt\right]^2 = E\left[\int_{-\infty}^{+\infty} h^2(t)n^2(-t)\,dt\right] = n_0^2\int_{-\infty}^{+\infty} h^2(t)\,dt; \tag{4.84}$$

hence,

$$\mathrm{SNR}_{H,t=0} = \frac{|y(0)|}{\left(E\left[\int_{-\infty}^{+\infty} h(t)n(-t)\,dt\right]^2\right)^{\frac{1}{2}}} = \frac{|y(0)|}{\left(n_0^2\int_{-\infty}^{+\infty} h^2(t)\,dt\right)^{\frac{1}{2}}}$$
$$= \frac{|y(0)|}{n_0\left(\int_{-\infty}^{+\infty} h^2(t)\,dt\right)^{\frac{1}{2}}} = \frac{|y(0)|}{n_0\|h\|_2} \tag{4.85}$$

and the proof is complete. ■

This observation allows Canny [75] to specify a signal-to-noise performance measure for the edge detection system H which does not depend on the input signal $x(t)$; in particular, the optimal edge detector must maximize (4.85). It turns out the convolution kernel that provides a maximum SNR, given that the underlying step edge contains noise, is just the reversed step itself. This result is known as the Matched Filter Theorem. We introduce matched filtering in Section 4.6.2, but we need to have the tools of Fourier transform analysis at our disposal in order to complete its theoretical justification. Granting the Matched Filter Theorem, then, the convolutional kernel that is optimal for finding step edges is precisely the DOB operator. Canny attempted to prove that the DOB operator enjoyed an optimal localization property as well. The development proceeds as follows.

Let $y = Hx = H(u + n)$, where $u(t)$ is the (noiseless) unit step signal and $n(t)$ is white Gaussian noise. Then $y = Hu + Hn$ by linearity. Let $w = Hu$ and let $m = Hn$. If a maximum of the detector output appears at time $t = t_0$, then $y'(t_0) = w'(t_0) + m'(t_0) = 0$. Differentiation of the convolution integral for $w = h * u$ gives $w'(t_0) = h(-t_0) = -h(t_0)$, since h must be odd (by the Maximal Response Edge Detector Proposition, Section 4.5.1). If t_0 is close to the ideal edge at $t = 0$, then by a Taylor series expansion for $h(t)$, we know that $h(t_0) \approx h(0) + t_0 h'(0)$. Since $h(0)$ must be zero by the same Proposition, it follows that $t_0 h'(0) = m'(t_0)$. This implies that $h'(0)^2 E[(t_0)^2] = E[m'(t_0)^2]$. But, as in (4.84),

$$E\left[m'(t_0)^2\right] = n_0^2 \int_{-\infty}^{+\infty} h'(t)^2 \, dt. \tag{4.86}$$

Thus,

$$E\left[(t_0)^2\right] = \frac{E\left[m'(t_0)^2\right]}{h'(0)^2} = \frac{n_0^2 \int_{-\infty}^{+\infty} h'(t)^2 \, dt}{h'(0)^2}, \tag{4.87}$$

and the optimal localization criterion that Canny proposes is $(E[(t_0)^2])^{-1/2}$. Canny argued that the DOB operator remains optimal for both this localization criterion and the SNR criterion, for instance by seeking to maximize the product of (4.85) and (4.87). Canny adduced the third criterion, namely that there should be no multiple responses around a noisy edge in order to derive his optimal edge detection kernel. It turns out, after some lengthy calculus of variations, that the operator is very similar to the dG kernel [75].

Some time later, Tagare and deFigueiredo [66] pointed out some flaws in the reasoning above. The principal points are that

- There can be multiple time values $t = t_0$ for which a maximum occurs and $y'(t_0) = w'(t_0) + m'(t_0) = 0$. The performance measure must account for the distribution of all such maxima, and there is no reason to prefer one above the others.
- Equation (4.87) computes the variance of the maximum time $t = t_0$. But, by the previous point, the value of t_0 varies with every realization of the white noise process, and the substitution of (4.86) into $h'(0)^2 E[(t_0)^2] = E[m'(t_0)^2]$ to get (4.87) is invalid.

Tagare and deFigueiredo propose an alternative localization criterion. They propose that an optimal edge detector $y = h * x$ should maximize the quantity

$$J_{H,t=0} = \frac{\int_{-\infty}^{+\infty} t^2 h(t)^2 \, dt}{\int_{-\infty}^{+\infty} \omega^2 \|H(\omega)\|^2 \, d\omega}, \tag{4.88}$$

where $H(\omega)$ is the Fourier transform of the impulse response, $h(t)$, of the edge detector. Since we will develop Fourier transform theory in the next two chapters and outline its applications in Chapter 7, we defer the exposition of the new localization criterion. It turns out, interestingly enough, that the optimal kernel for convolutional edge detection under the criterion (4.88) is the dG function.

4.5.4 Retrospective

Edge detection has played a central role in the theoretical development of signal and image analysis since it was first applied, in the early 1950s, in attempts at image understanding with digital computers [58]. A large number of interdisciplinary researchers have worked on edge detection from the standpoint of building biologically motivated signal and image analysis systems. Edge detection at a variety of resolutions played a central role in Marr's account of animal and machine vision systems [77]. Marr[5] drew careful parallels between biological and computer vision mechanisms and inspired a generation of computer vision researchers to do the same. In particular, Marr conjectured that edge analysis across many scales would be sufficient for a characterization of a signal or image. Since biological signal analysis systems were capable of edge detection and multiscale analysis, it would appear that a major bond would then exist between electronic computers and their software on the one hand and animal brains and their experiences on the other. We will have occasion to revisit this conjecture later in the book; the theory of time-scale transforms, or wavelet theory, would shed light on the question a dozen or so years after Marr proposed it.

From a computational perspective, Canny's work [75]—which comprised his Master's thesis work [78] at the Massachusetts Institute of Technology—was thought to have essentially solved the edge detection problem. There was a period when just about everyone's research paper included a Canny edge detector, if it mentioned edge detection at all. Some practitioners used a derivative of the Gaussian operator as a close approximation to the Canny kernel, which was difficult to compute. Only after the passage of several years did minor flaws in the theory become apparent; fortuitously, those many papers of those many researchers that

[5]David Courtenay Marr (1945–1980). Spurred by Marr's critique ["Artificial intelligence: a personal view," *Artificial Intelligence*, vol. 9, pp. 37–48, 1977], computer vision rose from a collection of ad hoc techniques to a broad discipline unifying ideas from biology, psychology, computer science, and robotics. One of the prominent pioneers of computer vision, Marr died quite young, in the midst of a most productive carreer.

resorted to the dG shortcut were following the optimal road after all! There has begun another round of research papers, controversies, and algorithmic emendations [66, 67, 79, 80]. The most recent research, in addition to emphasizing the dG kernel, has found interconnections between diverse methods reaching back some 20 years, and a unifying framework has been elaborated [68, 69].

4.6 PATTERN DETECTION

Many of the most important signal analysis problems involve detecting more than a single edge in a signal. There may be several edges in a signal region of interest, and ascertaining the signal levels, slopes, and transitional shapes between the edges may be critical to correctly interpreting the signal. The basic signal pattern detection or recognition problem involves finding a region $a \le t \le b$ within a candidate signal $x(t)$ that closely matches a prototype signal $p(t)$ on $[0, b-a]$. For discrete signals, the detection problem reduces to comparing finite sets of values. It seems simple; but in the presence of noise or other distortions of the unknown signals, complications do arise. The difficulties worsen when the detection problem allows the size of the prototype within the candidate to vary. Indeed, the problem can be trivial, challenging, problematic, and frustrating and may even defy robust solution. Signal pattern detection remains a topic of research journals, experimental results, and tentative solutions. This section presents three basic for pattern detection approaches: correlation methods, structural methods, and statistical methods.

4.6.1 Signal Correlation

Correlating two signals seems to be the natural approach to pattern detection. Its ultimate foundation is a dot product relation of similarity between vectors, which generalizes to the inner product operation for Hilbert spaces. Some care in the formulation is essential, however. Thus, we first explore the method of normalized cross-correlation in Section 4.6.1.1. There is a subsequent result, called the Matched Filtering Theorem, which indicates that this approach is in fact optimal. We do not, however, yet possess the theoretical tools with which to prove the matched filtering result; so we simply state it herein and postpone the proof until Chapter 7.

4.6.1.1 Normalized Cross-Correlation. The Cauchy–Schwarz Inequality, covered in Chapters 2 and 3, provides a mathematically sound approach to the signal matching problem. Let's suppose that we are dealing with an unknown, candidate signal x and a prototype signal p, which may come from a collection of model signals. The application may require a comparison of x with each prototype in the collection. Prototype signals represent patterns that we expect to find in the input to the signal analysis application; thus, it is reasonable to stipulate that the patterns are finitely supported on an interval $I = [a, b]$.

For this discussion, let us assume that we are working in the discrete realm, so that the signal space tools from Chapter 2 apply. It is useful to have a measure of match, or distance, between two signals, $d(x, y)$, that is a metric; that is, it statisfies the following:

- The positive definite property: $d(x, y) \geq 0$ for all x, y.
- The identity property: $d(x, y) = 0$ if and only if $x = y$.
- The symmetry property: $d(x, y) = d(y, x)$ for all x, y.
- The triangle inequality: For any z, $d(x, y) \leq d(x, z) + d(z, y)$.

If we take the signals to be square-summable, then the l^2 norm is a comparison measure which is a metric, $d(x, y) = \|x - y\|_2$. Assuming that we work with l^2 signals, a measure of the mismatch between two square-summable signals is the l^2 norm; in other words, if x is an unknown signal, then we need to minimize $\|x - p\|_2$ among all prototype signals p from a family of models our application attempts to detect. If the registration of pattern $p(n)$ within candidate $x(n)$ must be found, then we seek the offset k that provides the smallest $\|x(n+k) - p(n)\|_2$ for all appropriate offsets k of $p(n)$ relative to $x(n)$.

Choosing $d(x, y) = \|x - y\|_2$ as a match measure on real-valued signals is equivalent to using the inner product $\langle x, y\rangle$. To see this, suppose we compare a square-summable candidate signal $x(n)$ to a square-summable prototype $y(n)$. Then,

$$\begin{aligned}\|x-y\|_2^2 &= \sum_{n=-\infty}^{\infty} \left(x(n)-y(n)\right)^2 = \sum_{n=-\infty}^{\infty} \left(x(n)\right)^2 + \sum_{n=-\infty}^{\infty} \left(y(n)\right)^2 - 2\sum_{n=-\infty}^{\infty} x(n)y(n) \\ &= \|x\|_2^2 + \|y\|_2^2 - 2\langle x, y\rangle. \end{aligned} \tag{4.89}$$

Thus, we see that $d(x, y)$ depends on both $\|y\|_2$ and the inner product $\langle x, y\rangle$. If we require that all prototype vectors have unit norm, $\|y\| = 1$, then the equivalence between minimizing $d(x, y)$ and maximizing $\langle x, y\rangle$ is clear.

The inner product relation inspires an important method of matching signals: normalized cross-correlation. We introduced cross-correlation for discrete signals in Chapter 2, where it was noted to be a kind of backwards convolution. Suppose that we have a real-valued pattern signal $p(n)$, supported on the interval $[0, N]$: $p(n) = [p_0, p_1, \ldots, p_N]$. We search for the best match of $p(n)$ at offset location k in real-valued signal $x(n)$. A measure of the match between the two at offset k is the inner product, $y(k) = \langle p(n-k), x(n)\rangle$. When $y(k)$ is at a maximum value, then we have found the offset k at which the prototype pattern $p(n)$ best matches the source signal $x(n)$. This operation, in effect, correlates the shifted pattern $p(n-k)$ with a window of the signal $x(n)$; this windowed function is just $x(n)[u(n-k)-u(n-k-N-1)]$, which is zero outside of the interval $[k, k+N]$. To evaluate $y(k)$, notice that

$$y(k) = \sum_{n=-\infty}^{\infty} p(n-k)x(n) = \sum_{n=k}^{n=N} p(n-k)x(n) \leq \left(\sum_{n=k}^{N} p^2(n-k)\right)^{\frac{1}{2}} \left(\sum_{n=k}^{N} x^2(n)\right)^{\frac{1}{2}}. \tag{4.90}$$

The inequality in (4.90) follows from the Cauchy–Schwarz Inequality for discrete signals applied to $p(n-k)$ and the windowed signal $x(n)[u(n-k)-u(n-k-N-1)]$.

Recall from Chapter 2 that the Cauchy–Schwarz Inequality states that if $x(n)$ and $p(n)$ are in l^2, then the product $x(n)p(n)$ is in l^1, and $\|xp\|_1 \le \|x\|_2 \, \|p\|_2$. Recall that equality holds if and only if $x = cp$ for some constant c. Thus, in (4.90) there is equality if and only if $p(n-k) = cx(n)[u(n-k)-u(n-k-N-1)]$, where c is a constant. The problem with using $y(k)$ as a measure of the match between $p(n-k)$ and $x(n)$ over the region $[k, k+N]$ is that the second term on the right in (4.90) depends on the registration value k. Thus, we take as the measure of match between the pattern $p(n-k)$ and the signal $x(n)$ on $[k, k+N]$ to be the normalized cross-correlation,

$$C_{p(n-k),x(n)} = \frac{y(k)}{\left(\sum\limits_{n=k}^{N} x^2(n)\right)^{\frac{1}{2}}} = \frac{\sum\limits_{n=k}^{N} p(n-k)x(n)}{\left(\sum\limits_{n=k}^{N} x^2(n)\right)^{\frac{1}{2}}}. \tag{4.91}$$

The match measure (4.91) assumes its maximum value $C_{p(n-k),x(n)} = 1$ when the pattern is an exact multiple of the windowed input signal $x(n)$. Thus as a pattern recognition measure, normalized cross-correlation finds patterns that match our prototypes up to an amplification or attenuation factor.

Matching by normalized cross-correlation, dependent as it is on the inner product notion, is quite intuitive. The inner product operation is closed in our Hilbert spaces of square-summable discrete and square-integrable analog signals; we thus have a rich variety of signals with which to work. The inner product generalizes the dot product relation of finite-dimensional vector spaces, which is a geometrically satisfying relation of similarity between two vectors. Without a doubt, these factors recommend normalized cross-correlation as one of the first strategies to be employed in a signal analysis application. It is not the only method, however. Depending upon the application, it may suffer from a couple of problems:

- Computing the sums of squares in to arrive at the l^2 norms is time-consuming.
- There may be a lot of near-misses to the optimal registration of the pattern $p(n)$ in the unknown candidate signal $x(n)$.

To avoid the first problem, it may be feasible to apply the l^∞ norm, taking $d(p(n-k), x(n)) = \|p(n-k) - x(n)[u(n-k)-u(n-k-N-1)]\|_\infty$ as a match measure on real-valued signals instead. Computing this distance measure involves only a comparison of the $N+1$ values of $p(n-k)$ with those of $x(n)$ on the window $[k, k+N]$. In real-time applications, using digital signal processors, for example, this tactic can often help to meet critical time constraints. The second arises when there are a number of structures present in $x(n)$ that resemble the model pattern $p(n)$. An example of this situation arises during electrocardiogram analysis. Suppose a model QRS complex pulse has been extracted. Normalized cross-correlation produces many close responses. Thus, it may be necessary to search for certain critical signal features—other than raw signal values—around which more robust pattern detection methods might be constructed. This leads to the structural techniqes we consider next.

It is also possible to develop normalized cross-correlation for analog signals. Following the derivation of discrete normalized cross-correlation very closely, along with using the Cauchy–Schwarz inequality from Chapter 3, we obtain

$$C_{p(t-s),x(t)} = \frac{y(s)}{\left(\int_I x^2(t)\,dt\right)^{\frac{1}{2}}} = \frac{\left(\int_I p(t-s)x(t)\,dt\right)^{\frac{1}{2}}}{\left(\int_I x^2(t)\,dt\right)^{\frac{1}{2}}}, \tag{4.92}$$

where s is the offset of prototype signal $p(t)$ into input signal $x(t)$, and I is the interval that contains the support of $p(t)$. Deriving this result is left as an exercise.

The next section sheds some light on the importance of cross-correlation for the detection of patterns that are embedded in noise.

4.6.1.2 Matched Filtering: A Glimpse. Understanding that real-life signals contain noise, it is natural to wonder what is the best way to match a signal against a pattern. If the detection process is a linear, translation-invariant system, H, and the input signal, $x(t)$, is corrupted by additive white noise, $s(t)$, then there is an answer to this intriguing question. It turns out that the impulse response of the system H, $h(t)$, is—up to a scaling (amplifying or attenuating) factor—none other than a reflected and translated version of the input signal $x(t)$. The impulse response $h(t)$ is called the matched filter for the signal $x(t)$. Let us state this as a theorem for analog signals, but postpone the proof until we have covered the Fourier transform theory.

Theorem (Matched Filter). Suppose $x(t) = f(t) + n(t)$, where $f(t)$ is a signal of known shape, and $n(t)$ is a zero mean white noise process. Then the optimal LTI system H for detecting $f(t)$ within $x(t)$ has impulse response $h(t) = cf(t_0 - t)$ for some constants c and t_0.

Proof: Chapter 5 prepares the foundation in analog Fourier theory. Chapter 7 proves the Matched Filter Theorem as an application of Fourier methods. ■

In developing the normalized cross-correlation method for signal matching, we found that the best possible match occurs when the pattern is identical to the source signal up to a constant scale factor on the shifted support interval of the prototype. Note that the convolution implied by the Matched Filter Theorem is the same as a cross-correlation with a reflected version of the source. Thus, matched filtering theory formulates an important converse to normalized cross-correlation, namely, that the best filter—for reasonable noise assumptions—is in fact just the target source pattern itself. Many communication theory texts cover matched filtering [81, 82], there are full reviews in the engineering literature [83], and Chapter 7 points out additional resources in print.

If matched filtering is both the obvious and optimal approach, why study anything else? One answer, as with “optimal” thresholding and “optimal” noise removal, is that the conditions that validate the optimality do not always hold. Furthermore, the optimal methods for such problems are often computationally burdensome, even untractable. Convolution, at the heart of matched filtering, can be expensive for signals, and on images may require special-purpose computing hardware. Thus, reasonably fast suboptimal techniques are desirable; a lot of these methods arise in conference presentations, appear in print, and become the basis for practical applications.

4.6.2 Structural Pattern Recognition

A structural pattern recognition application discovers the physical relationships between labeled signal regions and unites the regions into a graph structure. Then the application compares the graph structures that it derives from candidate signals against those from prototype patterns. The structural analysis application estimates the degree of match or mismatch between these graphs. If the match measure exceeds a theoretically or empirically derived threshold, then the candidate signal matches the prototype pattern; recognition is complete.

In signal analysis, structural descriptions derived from signals tend, like their source data, to be one-dimensional entities. Commonly the structure is just a vector, and the components of the vector are numerical weights that indicate the relative presence of some expected characteristic of the signal over a time interval. In computer vision, the structural descriptions are often two-dimensional, like the images they come from. Not unexpectedly, the feature relationships are more intriguing, more varied, and more difficult to analyze. Sometimes it is important to extract features at a range of scales of the signal's domain. Then there is a family of feature vectors, and a feature at a coarse scale may resolve into a set of finer scale characteristics. This forms a two-dimensional structural description of the signal: There are large-scale or low-resolution features at the top; features of intermediate size in the middle; and high-resolution, small-scale features at the bottom. For instance, we might find a texture region when segmenting a signal into wide regions and then resolve it into small curves, edges, and flats. At the finest scale, discrete signal features are just the signal values. The discrete world limits signal resolution. With analog signals, resolution refinment need never stop. This is called a pyramid representation. Pyramidal structural descriptions for signals are an important and widely studied analysis tool. However, with their introduction into the signal analysis application, the problems become as intricate as they are in computer vision.

This section first explains how to reduce the amount of signal data subject to analysis by the pervasive strategy of feature extraction. Section 4.6.2.2 follows with some basic measures for signal matching by comparing feature vectors. The vectors contain fewer numbers than the signals, so comparisons are quick—a happy contrast to correlation-based approaches. Next, we develop some formal theory for structural descriptions of signals. The last subsection explains a comparison measure for structural descriptions. This measure turns out to be a metric, one of our fundamental theoretical results in structural pattern recognition.

4.6.2.1 Feature Extraction. Extraction of feature vectors reduces the dimensionality of the matching problem. The expense of computing the normalized cross-correlation match measure is a drawback in its application. This is all the more the case when a candidate signal must be compared against many prototype signals.

Structural pattern recognition attempts to place input signals $\{f_1, f_2, \ldots, f_N\}$ into categories $\{C_1, C_2, \ldots, C_K\}$ by forming an intermediate, structural description of the signal and then using the structural description as the basis for classification. The simplest form of structural description is the feature vector, $v = (v_1, v_2, \ldots, v_M)$. Components of the feature vector are numerical, usually real but sometimes complex numbers. By thresholding the magnitude of the feature vector components, labels can be assigned to the features to build an abstract or symbolic description of the signal. Selecting the right feature vector is both quite open to design and quite critical to the recognition application's success.

Let's consider how we might extract a feature vector from a signal. First, suppose that we have a candidate signal, $x(n)$, and a set of model or prototype signals $\{e_i(n) \mid 1 \le i \le N\}$ defined on some interval $[a, b]$. The plan is to develop a vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_N)$ such that its components, v_i, are a measure of the similarity between $x(n)$ and prototype e_i. Any of the norms we know from the theory of normed linear spaces in Chapter 2 are suitable; we may choose $v_i = \|x - e_i\|_p$, the l^p norm of the difference between the candidate signal and prototype e_i. This measures the mismatch between x and e_i. The l^p norm is a metric and is suitable for pattern detection applications.

But the results on normalized cross-correlation and matched filtering in Section 4.6 inspire us to use the inner product $v_i = \langle x, e_i \rangle$. Since the inner product $\langle x, e_i \rangle$ depends on the magnitude of e_i, the prototypes should all have the same magnitude; otherwise a large-magnitude prototype will skew all of the matches toward itself. Orthogonality of prototypes is important also. Suppose that $\langle e_1, e_2 \rangle \ne 0$. Without orthogonality, a signal which is a scalar multiple of e_1, say $y = ce_1$ on $[a, b]$, will have a similarity measure to e_2 of $\langle y, e_2 \rangle \ne 0$. Thus, although it is a perfect scaled (amplified or attenuated) replica of prototype e_1, y shows a similarity to e_2. The solution to this identification conundrum is to simply stipulate that the prototypes be orthogonal and of unit length. Hence, orthonomal bases—such as we developed in Chapters 2 and 3 for Hilbert spaces—play an important role in pattern recognition theory. An important problem is how to choose the basis for pattern recognition applications. Of course, we can orthogonalize any linearly independent set of signals on $[a, b]$, using, for instance, the Gram–Schmidt procedure sketched in our reflections on Hilbert space. However, the resulting basis set, $\{e_i\}$, may not resemble the original patterns. One way to ensure the construction of a pattern recognition basis set that preserves the features of input signals is to build the basis set according to candidate signal statistics. A classical technique, the Karhunen–Loève transform, can be applied to select the best possible basis [84, 85]—for instance, when the problem is a simple edge shape [65].

There are many other ways to extract feature vectors from signals. We can study a single signal region with basis functions, as above. The basis function inner products can also be applied to a series of intervals. But other processing steps are possible. The presence of a given signal level can be used as a feature vector component. The presence of a texture parameter—roughness or waviness—is a common technique in feature vector designs. Often the signal features have fixed registrations, assigned as part of the signal analysis application's design. In other situations, an input signal is subject to an initial segmentation, perhaps by signal level (thresholding), texture analysis, or edge detection. This determines the presence and registration of useful signal content. Then the feature vector is extracted from a set of intervals that derive from the preliminary segmentation. It is common, too, to generate a single large feature vector, $\boldsymbol{v} = \{v_{i,j} \mid 1 \le i \le N, 1 \le j \le M\}$, where the components $v_{i,j}$ for $1 \le j \le M$, represent M different feature values, each applied to the same region of interest of the signal, S_i, $1 \le i \le N$. Statisitcal analysis of feature vectors may reveal correlations, and some of them can be deleted from the final application. Optimal selection of features is possible and explored fully in pattern recognition treatises [85]. Feature vector components can also be labeled. Components that exceed certain thresholds that exceed certain thersholds receive an appropriate label, and the vectors of labels are processed by higher-level, goal-directed artificial intelligence algorithms.

Let's consider some examples of feature extraction to see some of the alternatives.

Example (Plasma Etch Reactor Endpoint Detection). The semiconductor manufacturing industry uses plasma reactors to selectively remove materials from the surface of silcon or gallium arsenide wafers [70]. Plasma chemistry is notoriously complex. It is difficult to know how fast etching proceeds and when to halt the process. A popular technique for ending a plasma etch process is to monitor the optical emissions from reaction species for gross changes. The sudden disappearance of an etching byproduct emission indicates the endpoint of the etch cycle. Alternatively, the appearance of optical spectra characteristic of an underlying layer that should be preserved means that the reaction should halt. If a spectrograph monitors the appropriate emissions, its output is digitized, and a computer processes this digital signal, then this at first appears to be a simple real-time edge detection task. However, differences in plasma chemistry across runs, reactor chambers, and semiconductor wafer patterns combine to make this control strategy quite problematic. Moreover, as the semiconductor industry continues to reduce the scale of integrated circuits, etched areas get very small, targeted emission species diminish, and distinct endpoints become increasingly harder to identify. Feature extraction methods in common use for recognizing plasma etch endpoint include the following:

- Estimates of the first or second derivatives of the endpoint trace
- Approxmiations of the signal level within fixed regions of the trace
- Estimates of the departure of the endpoint signal from expected models of the optical emission trace
- Estimates of the endpoint trace curvature in fixed regions of the endpoint trace

Example (Internal Combustion Engine Knock Detection). In automobile engines there is an especially noticeable and harmful abnormal combustion situation known as knock. Knock occurs after the spark plug fires and results from a spontaneous combustion of unburned gasoline vapors in the cylinder. It produces a sharp, metallic, clanking sound, and is very harmful to engine components. By mounting an accelerometer on the engine block and recording the engine vibration signal, Molinaro and Castanié were able to digitally analyze engine knock [86]. From a vibration signal $x_a(t)$, the researchers acquired digital samples $x(n)$, $1 \le n \le N$, and extracted feature vectors $\boldsymbol{v} = (v_1, v_2, ..., v_p)$. The researchers studied a large variety of possible features, v_i, including the signal energy averaged over N samples,

$$v_1 = E_x = \frac{\sum_{n=1}^{N} |x(n)|^2}{N}. \tag{4.93}$$

Another set of features was derived from the histogram values of $|x(n)|$,

$$v_{2+i} = \#\{x(n) \mid k_i \le |x(n)| = k_{i+1}\}, \tag{4.94}$$

where $K = \{k_0, k_1, ..., k_{q-1}\}$, $0 \le i < q$, determines a set of intervals within the range of $|x(n)|$. Another set of parameters, designed to capture the significant periodicities in the $x(n)$ values, are given by $|X(k)|^2$, where, as in our study of texture segmentation in Section 4.3.2, we have

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp\left(\frac{-2\pi jnk}{N}\right). \tag{4.95}$$

Molinaro and Castanié keep $|X(k)|^2$ for $k = 1, 2, ..., \lceil N/2 \rceil$ as feature vector elements. Two other families of feature vector components are more exotic: Prony and cepstral coefficients. The investigators model the trend of $x(n)$ values with a Prony model of order P:

$$x(k) \approx \sum_{k=0}^{p} A_k e^{j\theta_k} e^{n(\alpha_k + j2\pi f_k)}. \tag{4.96}$$

The construction of the model is outside the present scope, but related to the z-transform construction of Chapter 8 [87]. The cepstral coefficients, c_i, are Fourier coefficients of the Fourier transform of |the logarithm of $X(k)$|, given in (4.95):

$$\log|X(k)|^2 \approx \sum_{n=-Q}^{Q} c_i e\left(\frac{-2\pi jnk}{N}\right). \tag{4.97}$$

Both Prony model parameters and cepstral expansion coefficients are useful for compactly encoding the frequency and stability information within a signal. Prony parameters figure in time series analysis studies [88] and cepstral expansions have been widely used in speech detection applications [89]. The Prony model parameters of A_k, θ_k, α_k, and f_k and the cepstral coefficients c_k become feature vector elements. The point of this discussion is not to erode the reader's confidence is selecting feature vectors, but, rather, to show some of the possibilities and variety of feature vector components used in a modern application. In fact, Molinaro and Castanié reduce the number of feature vectors by statistical techniques before invoking their ultimate detection criteria.

4.6.2.2 Distance Measures for Feature Vectors. Let's consider two basic methods for comparing feature vectors. In a typical signal analysis application, feature vectors for a library of prototype signals, or models, may be archived in a database. Feature vectors are similarly extracted from input signals—candidates or unknowns—and compared to the library of prototype vectors.

Once the data reduction from time-series signal to finite feature vector is finished, it is possible to apply the Euclidean norm, from finite-dimensional vector spaces, to match prototype signals to candidates. This is called the *minimum distance classifier.*

Definition (Minimum Distance Classifier). Suppose $\boldsymbol{v}$ is a feature vector extracted from a application input signal, $x(n)$, and $\{\boldsymbol{e}_m \mid 1 \le m \le M\}$ is the set of feature vectors of prototype patterns. Then the minimum distance classifier recognizes $x(n)$ as being of type k if $\|\boldsymbol{v} - \boldsymbol{e}_k\| \le \|\boldsymbol{v} - \boldsymbol{e}_m\|$ for all m, $1 \le m \le M$.

Another popular classifier works on labeled feature vector components. Suppose that signal feature vector components are derived from a partition of the domain of signal $x(n)$; that is, each feature vector component represents a particular subset of Dom(x). Let $\Pi = \{S_1, S_2, ...\}$ be the partition of Dom(x). Suppose also that Λ is labeling of x for Π, Λ: $\Pi \to \{\Lambda_1, \Lambda_2, ... \}$. Typically, labels are applied to feature vector components if the component's magnitude exceeds a threshold associated with the feature. If feature vectors have been extracted from library prototypes as well, then candidate signals can be matched against library prototypes by comparing the labelings. One such method of comparing vectors of labels and labels applied to the vector components is the Hamming distance.

Definition (Hamming Distance Classifier). The Hamming distance between a candidate signal label vector $\boldsymbol{u} = (\alpha_1, \alpha_2, ..., \alpha_N\}$ and a prototype vector $\boldsymbol{w} = (\beta_1, \beta_2, ..., \beta_N\}$ is the number of positions in which $\boldsymbol{u}$ and $\boldsymbol{w}$ differ. We write the Hamming distance between label vectors $\boldsymbol{u}$ and $\boldsymbol{w}$ as $H(\boldsymbol{u}, \boldsymbol{w})$.

It is easy, and left as an exercise, to show that the Hamming distance is a metric.

4.6.3 Statistical Pattern Recognition

The third approach to pattern recognition that we will consider is statistical pattern recognition. It is possible to resort once again to a least-squares approach. The least-squares coefficient matrix derives from a large number of samples and represents the known knowledge of input signals to the analysis application [89, 90]. We considered this very general and very powerful approach to the problem of finding an optimal noise-removal filter in Section 4.3. The extension of the method to pattern recognition is straightforward. Of the many statistical approaches, we will examine one of the most important: the Bayes classifier.

4.6.3.1 Bayes Classifier. The Bayes classifier is a fundamental tool in pattern recognition. It is a parametric approach, in that statistical parameters associated with the source patterns (signals or images) are assumed or approximated by the application. As the first chapter already hinted, this pattern classification method relies on Bayes's formula for conditional probabilities.

Statistical pattern recognition, like correlation-based matching and structural recognition, attempts to associate a class or category, $\{C_1, C_2, ..., C_K\}$, with each input signal $\{f_1, f_2, ..., f_N\}$. In order to develop statistics for each input signal, the signal is decomposed into a feature vector, $\boldsymbol{v} = (v_1, v_2, ..., v_M)$. Each component of the feature vector is numerical, in order to develop statistics for the likelihood of features, classes, and for features within signal classes. Selecting the right feature vector is both quite open to design and quite critical to the recognition application's success.

Standard pattern recognition texts cover the Bayes classifier [91–93]. One of the earliest applications of Bayes classifiers to the design of a character recognition system was by Chow and dates from the 1950s [94].

4.6.3.2 Statistical Decision Rules. This section explains how statistical decision rules for deciding class membership can be made based on the likelihood of each class of signal occurring and on the probability distributions of feature vectors among classes. We examine the problem in several ways. First we frame the problem as a search for a set of discriminant functions that indicate the resemblance of a signal to members of a class. Then we consider the problem of finding risk functions. That is, we seek functions that measure the risk of misclassification, and our problem transposes to one of minimizing the risk of a mistaken classification of a signal. Finally, we pose the problem in terms of the Bayes rule for conditional probabilities. We find that this provides a reasonable statistical tool for building discriminant and risk functions, although there are a number of probability density functions that must be determined.

Discriminant and risk functions are closely related. Suppose that we assigning signals $\{f_1, f_2, ...\}$ to classes $C = \{C_1, C_2, ..., C_K\}$. For each signal, $f \in \{f_1, f_2, ...\}$, a feature vector, $\boldsymbol{v} = (v_1, v_2, ..., v_M)$, is generated. We desire a set of discriminant functions, $D_1, D_2, ..., D_K$, one for each signal class. The idea is that $D_k(\boldsymbol{v})$ tells us how strongly signals with features $\boldsymbol{v}$ resemble signals from class C_k. A discriminant-based classifier assigns signal f with feature vector $\boldsymbol{v}$ to class C_k if $D_k(\boldsymbol{v}) > D_i(\boldsymbol{v})$ for all $i \neq k$. The complementary idea is the risk function. Now we seek functions, R_1, R_2, ..., R_K, such that $R_k(\boldsymbol{v})$ tells us the risk of classifying f with features $\boldsymbol{v}$ as belonging to class C_k. How strongly do signals with features $\boldsymbol{v}$ resemble signals from class C_k? A risk-based classifier places signal f with feature vector $\boldsymbol{v}$ into class C_k if $R_k(\boldsymbol{v}) < R_i(\boldsymbol{v})$ for all $i \neq k$. Taking $R_k(\boldsymbol{v}) = -D_k(\boldsymbol{v})$ makes an easy transition from a discriminant-based classifier to a risk-based classifier.

Now let's consider how to use statistical information about feature vectors and classes to develop statistical discriminant functions. Suppose that we know the a priori probability of occurrence of each of the classes C_k, $P(C_k)$. Suppose further that for each class, C_k, we know the probability density function for the feature vector $\boldsymbol{v}$, $p(\boldsymbol{v}|C_k)$. The conditional probability, $P(C_k|\boldsymbol{v})$, provides the likelihood that class k is present, given that the input signal has feature vector $\boldsymbol{v}$. If we could compute $P(C_k|\boldsymbol{v})$ for each C_k and $\boldsymbol{v}$, then this would constitute a statistical basis for selecting one class over another for categorizing the input signal *f*. But the Bayes formula for conditional probabilities (Chapter 1) provides a tool for calculating this a posteriori probability:

$$P(C_k \mid \boldsymbol{v}) = \frac{p(\boldsymbol{v} \mid C_k)P(C_k)}{p(v)} = \frac{p(\boldsymbol{v} \mid C_k)P(C_k)}{\sum_{i=1}^{K} p(\boldsymbol{v} \mid C_i)P(C_i)}, \tag{4.98}$$

where $p(\boldsymbol{v})$ is the probability density function for feature vector $\boldsymbol{v}$. This inspires the Bayes decision rule:

Definition (Bayes Decision Rule). Given a signal *f* and a feature vector $\boldsymbol{v}$ derived from *f*, the Bayes Decision Rule is to classify *f* as belonging to class $C_k \in C = \{C_1, C_2, ..., C_K\}$ if $P(C_k|\boldsymbol{v}) > P(C_i|\boldsymbol{v})$ for all $i \neq k$; otherwise, classify *f* as belonging to some C_k where $P(C_k|\boldsymbol{v})$ is maximal.

We can also improve upon our concept of risk by incorporating the probabilistic ideas. Suppose that the cost of assigning signal *f* with feature vector $\boldsymbol{v}$ to class C_k, when it really belongs to class C_i is $r(k, i)$, where $r(k, i) = 0$ if $k = i$. Typically, $r(k, i)$ is a 1 or 0, for the cost of an error or the cost of no error, respectively. Then, the total cost simply counts the number of misclassifications. In a real application, it may be necessary to provide more informative risk estimates and cost counters. For example, in a character recognition system, the cost might be the time estimate for human assistance to the application. Then, incorporating the a posteriori probabilities from Bayes's formula, the risk of placing *f* in class *k*, $R(C_k, v)$, is the sum of the individual misclassification risks,

$$R(C_k, \boldsymbol{v}) = \sum_{i=1}^{K} r(k,i)P(C_k \mid \boldsymbol{v}). \tag{4.99}$$

Classifying signal *f* with feature vector $\boldsymbol{v}$ as belonging to class C_k according to whether $R(C_k, \boldsymbol{v}) \leq R(C_i, \boldsymbol{v})$ for all $i \neq k$ implements the Bayes decision rule (4.99).

Now let us consider possible discriminant functions based on the Bayes formula (4.98). We may begin by assigning the discriminants $D_k(\boldsymbol{v}) = P(C_k|\boldsymbol{v})$, but noting that, given feature vector $\boldsymbol{v}$, for each such *k*, $1 \leq k \leq K$, the denominator in (4–6.1) is identical, we can simplify the discriminants to have

$$D_k(\mathbf{v}) = p(\mathbf{v} \mid C_k)P(C_k). \tag{4.100}$$

And it will be convenient in a moment to take the natural logarithm of (4.100) to obtain the alternative discriminant function

$$D_k(\mathbf{v}) = \log(p(\mathbf{v} \mid C_k)) + \log(P(C_k)). \tag{4.101}$$

Indeed, any constant can be added to each of a family of discriminant functions without changing their characterization capability. Each of the family of discriminants may be multiplied by any positive constant. Finally, a monotonic increasing function, such as the logarithm, may be be applied to all of the discriminants without affecting the classification results. These operations are often useful for simplifying the evaluation of discriminants.

4.6.3.3 Estimating Probabilities. Let us now turn to the practical problem of how to estimate the probabilities that are necessary for applying the Bayes rule for pattern recognition. Upon examinination of (4.98), we see that applying the Bayes formula assumes that we have knowledge of:

- The a priori probability of class occurrence C_k, $P(C_k)$
- The probability density function for the feature vector $\boldsymbol{v}$, given the presence of a signal of class C_k, $p(\boldsymbol{v}|C_k)$

How are we to estimate these probabilities?

It may be possible to know the class probabilities, $P(C_k)$, from the nature of the application. For example, in a character recognition application, the class probabilities come from known probabilities of the presence of characters in text. This could come from a broad scientific study of the problem area. To wit, the development of classifiers for the optical character recognition systems used by the United States Postal Service required the collection and analysis of hundreds of thousands of character samples [95–97]. And lacking the luxury of a broad scientific study, the character recognition classifier designer can perform a sampling of representative texts. In the case of speech phoneme recognition, these probabilities are known, but if the application concerns a special application area (i.e., spoken numbers for a telephone dialing application utilizing voice recognition), then more uniform probabilities apply. There are some subtleties, to be sure. In a spoken number recognition system, one must be alert to common alternative pronunciations for numbers. The system designer might assume that $P(C_1) = \cdots = P(C_9) = .1$, with C_n signifying the utterance of a nonzero number. But two common alternatives exist for saying "zero": "zero" and "oh." Thus, the design might posit two classes for this utterance; thus, $P(C_0) = P(C_z) = 0.05$, for saying "oh" or "zero," respectively.

The class conditional feature probabilities are a further problem. If we conduct a thorough statistical study of a large number of samples from each class, then it is possible to arrive at rough ideas of the distribution functions. But when a signal f arrives at the input to the system, and the front-end processing modules derive its feature vector v, how are we to compute the probability density functions $p(\boldsymbol{v}|C_k)$ for each class C_k? If we answer that we should once more adopt a parametric stance—that is, we assume a particular probability density function for the conditional densities required by (4.98)—then there is a elegant and practical resolution to this problem.

It should come as no surprise that the distribution of choice is the Gaussian. Not only is it justified by the Central Limit Theorem, but it provides the most tractable mathematical theory. Since we are dealing with feature vectors, we must consider the multivariate normal distribution. For the probability density function $p(\boldsymbol{v}|C_k)$, this parametric assumption is

$$p(v \mid C_k) = \frac{\exp[-\frac{1}{2}(\boldsymbol{v}-\mu_k)^T \Sigma_k{}^{-1}(\boldsymbol{v}-\mu_k)]}{\sqrt{\det(\Sigma_k)(2\pi)^M}}, \tag{4.102}$$

where $\boldsymbol{v} = (v_1, v_2, ..., v_M)$ is the feature vector of length M extracted from signal f; μ_k is the mean for feature vectors from signals of class C_k, $\mu_k = E[\boldsymbol{v}|C{=}C_k]$; $(\boldsymbol{v} - \mu_k)^T$ is the transpose of $\boldsymbol{v} - \mu_k$; and Σ_k is the $M \times M$ covariance matrix for feature vectors of C_k signals, $\det(\Sigma_k)$ is its determinant, and Σ_k^{-1} is its inverse.

The parametric assumption allows us to estimate the conditional probabilities, $p(\boldsymbol{v}|C_k)$. This requires a preliminary classifier training step in order to establish the statistical parameters that the analysis application uses to classify incoming signals. Data from a large number of signals for each of the different classes $\{C_1, C_2, ..., C_K\}$ is collected. For each such signal, its feature vector $\boldsymbol{v}$ is derived. The average of the feature vectors from class C_k is μ_k. Once we have computed the means for all of the feature vectors from signals in a class C_k, then we can compute the covariance matrices, $\Sigma_k = E[(\boldsymbol{v} - \mu_k)(\boldsymbol{v} - \mu_k)^T$. Once the covariance matrix is computed for a class C_k, its determinant and inverse can be calculated (this is problematic when the number of features is large). The feature vector averages, $\{\mu_1, \mu_2, \ldots, \mu_K\}$, and the covariance matrices, $\{\Sigma_1, \Sigma_2, \ldots, \Sigma_K\}$, are stored for analyzing the signals that the classifier accepts as inputs. This completes the training of the classifier.

The steps in running the classifier are as follows. For every signal f with feature vector $\boldsymbol{v}$, the Bayes classifier

- Computes (4.102) for each class C_k;
- Calculates the a posteriori probability $P(C_k|\boldsymbol{v})$ for each k, $1 \le k \le K$;
- Classifies signal f as belonging to class C_k where $P(C_k|\boldsymbol{v})$ is maximal.

4.6.3.4 Discriminants. Now let us consider different discriminant functions that support a Bayes classifier. Different discriminants arise from the statistics of the feature vectors of input signals to the classifier. In particular, special cases of the covariance matrix Σ_k in (4.102) result in significant simplifications to the classifier's computations.

Let us first look at an alternative discriminant function for (4.98), namely the natural logarithm. Since the natural logarithm is monotone increasing, taking $\log(P(C_k|\boldsymbol{v}))$ provides the same recognition decision as $P(C_k|\boldsymbol{v})$ itself. Thus, if we assume normally distributed feature vectors (4.102), then we have

$$\begin{aligned} D_k(\boldsymbol{v}) &= \log(p(\boldsymbol{v}\,|\,C_k)) + \log(P(C_k)) \\ &= -\frac{1}{2}(\boldsymbol{v} - \mu_k)^T \Sigma_k^{-1}(\boldsymbol{v} - \mu_k) - \frac{m\log(2\pi)}{2} - \frac{\log(\det(\Sigma_k))}{2} + \log(P(C_k)). \quad (4.103) \end{aligned}$$

The term $m \log(2\pi)/2$ does not depend on C_k, so it can be ignored; thus, we may set

$$D_k(\boldsymbol{v}) = -\frac{1}{2}(\boldsymbol{v} - \mu_k)^T \Sigma_k^{-1}(\boldsymbol{v} - \mu_k) - \frac{\log(\det(\Sigma_k))}{2} + \log(P(C_k)). \quad (4.104)$$

In (4.104), $\det(\Sigma_k)$ and $P(C_k)$ can be calculated from the training data, so only the vector product $(\boldsymbol{v} - \mu_k)\Sigma_k^{-1}(\boldsymbol{v} - \mu_k)^T$ needs to be calculated for the feature vector $\boldsymbol{v}$ of every input signal f.

Three simplifying assumptions make the discriminants (4.104) easier to compute:

- The a priori probabilities $P(C_k)$ of the signal classes are all equal.
- The features vectors are statistically independent and have the same variance.
- The convariance matrices are all the same.

In the first case, we may drop the $\log(P(C_k))$ term from (4.104). This helps, but the inversion and determinant of the covariance matrix pose a greater threat to computational tractability.

In the second of these special cases we find $\Sigma_k = \sigma^2 I$, where I is the $M \times M$ identity matrix. Then $\Sigma_k^{-1} = \sigma^{-2} I$, which is independent of class. This allows us to trim the discriminant functions further, so that now, having removed all class-independent terms, we arrive at

$$D_k(\boldsymbol{v}) = -\frac{1}{2\sigma^2}(\boldsymbol{v} - \mu_k)^T(\boldsymbol{v} - \mu_k) + \log(P(C_k)). \tag{4.105}$$

Making the additional assumption that class membership likelihood is the same for all C_k, we find that maximizing $D_k(\boldsymbol{v})$ in (4.105) is the same as minimizing $(\boldsymbol{v} - \mu_k)(\boldsymbol{v} - \mu_k)^T = \|\boldsymbol{v} - \mu_k\|^2$. This classifier we are already familiar with from Section 4.6.2.2. It is the minimum distance match between features drawn from the input signal f and the mean feature vectors. In other words, a Bayes classifier with statistically independent features of equal variance reduces to a minimum distance classifier. The class C_k for which feature vector $\boldsymbol{v}$ is closest to μ_k is the class to which we assign input signal f.

The Bayes classifier is optimal; and when the feature vectors obey a Gaussian distribution, the discriminant functions (4.103) are the proper tool for separating input signals into classes. There are nevertheless some important difficulties with the Bayes classifier, and we must note them. The feature vectors may not, in point of fact, be normally distributed. This makes the computation of the discriminant problematic. Sometimes, alternative features that more closely follow a Gaussian distribution can be selected. Furthermore, if the number of features is large, the computation of the determinant and inversion of the covariance matrix become intractable. There are, finally, some philosophical reasons for objecting to the Bayes classifier [98].

4.7 SCALE SPACE

This section studies a signal analysis technique known as *scale-space decomposition*. From our first studies of representative signal interpretation problems, we noted that the determination of the size of a signal component is a critical step in analyzing the signal. One task in automated electrocardiography, to recall an example from the first chapter, is to distinguish between splintered and normal contractions of the heart's left ventricle. It is the time-domain extent of the signal's jump that determines whether there is a normal contraction or an abnormal, spasmodic contraction. And in edge detection, we noted that understanding scale is a necessary part of simple edge detection. For an edge at a fine scale could just as well be considered an insignificant signal aberration at a coarse scale.

Soon after researchers began examining the magnitude of signal derivatives in study of edges, their interest extended to signal curvature, which is a local measure of signal shape. There are several quite natural reasons for this transition. Psychological experiments revealed that a few curved line fragments suffice to impart object shape information—Attneave's famous example is a sleeping cat—to a human subject [99]. Also, from calculus, the sign of the second derivative determines the curvature of a signal: Where the second derivative is positive defines a concave up region, where it is negative defines a concave down region, and where it is zero an inflection point exists. The problem is that the second derivative information is very noisy in real signals and images, resulting in erratic segmentation in regions of different curvatures. Researchers turned increasingly toward multiple resolution methods that would support the precious information content from curvature. Hierarchical methods for image processing and recognition within natural scenes were disclosed, for example [100, 101].

We have already noted Marr's contribution to multiscale edge detection, which formed the cornerstone of a very important trend in signal and image analysis [77]. Marr oriented the attention of the scientific and engineering communities to the links between the algorithms of engineered systems and the processes within biological systems—for example, animals [102]. Marr's strategy was to study animal sensory systems, especially vision, and from them derive the inspiration for machine vision system designs. Researchers in the new field of psychophysics, which studies the brain's sensory processes at a stage where they are still independent of consciousness, had found evidence of multiple resolution, orientation-sensitive processing channels in animal vision systems [103–105]. Aware of the biological vision system research, recognizing its link to the multiple resolution image analysis efforts, and building upon their earlier work in edge detection, Marr and his coworkers proposed a three-stage architecture for vision systems:

- The raw primal sketch, which segments the signal into edges and concavity regions;
- The extraction of geometric information relative to the observer;
- The determination of geometric information independent of the observer.

While Marr's scheme is a vision system formulation, we wish to focus on signal interpretation and shall reduce his architecture into one dimension. Let's replace Marr's visual terms with auditory terms. Thus, we ought to look not for a raw primal sketch so much, perhaps, as a raw primal listen. And we ought to think of signal or sound content rather than think of geometric information, with its planar and spatial connotations. An example should help. When you hear a whistle at a railroad crossing, the raw primal sketch consists of the auditory edge at the onset of train's whistle and the increasing intensity thereafter. The sudden realization that a train is coming closer to you constitutes the observer–relative signal content. The reflection that a train is about to cross through the intersection makes up the observer-independent information in the experience.

Computationally, the raw primal sketch begins by convolution of the signal with the second derivative of the Gaussian. If $f(t)$ is a signal and $g(t, \sigma)$ is the Gaussian of standard deviation σ and zero mean, then we let $F(t, \sigma)$ be defined by

$$F(t, \sigma) = f(t) * \frac{\partial^2}{\partial x^2} g(t, \sigma). \tag{4.106}$$

We recognize this as the edge detection operation of Marr and Hildreth [62]. The next raw primal step is to link the edges together and segment the signal into regions of concavity (concave down) and convexity (concave up). The remaining steps in Marr's schema, the development of signal information relative to the observer and signal information independent of the observer, then follow.

Note that there does not seem to be any clear direction within Marr's algorithm to decide at what scales σ the edge detection operation (4.106) should be developed. The second two steps in the algorithm are less clear than the first, and they contain a number of thresholds and parameters whose values are difficult determine without extensive experimentation. Finally, there is no clear relation between the primal sketch information in the different channels. How does the content of a large σ channel affect the processing of the conent of a small σ channel? Does processing occur from large to small or from small to large σ channels? These difficulties with the conception brought some researchers to critique Marr's architecture. Prominent among the skeptics, Pentland, deemed it to be too data-driven and lacking in its qualitative aspects [106]. Soon enough, however, in the early 1980s—beginning what would turn out to be a decade of signal analysis research breakthroughs—an important step in demonstrating qualitative computer vision strategies was taken by Witkin [107] and by Koenderink [108] with the concept of scale-space representation.

Like Marr's theory, the scale-space representation smoothes an image with a family of Gaussian filters. There are two substantive differences, however: A full set of smoothed images is maintained, and there is a critical interconnection between the regions of concavity at different scales, σ. We shall see that a complete description of the signal's shape results. The description proceeds from the smallest to the largest scale, and each concavity feature of the signal ranked according to its significance.[6]

Scale space decomposition thus furnishes a useful signal analysis paradigm. It identifies concavity as a critical feature of signals. It highlights the link that this feature shares with biological vision systems, as, for instance, a scattering of curved lines immediately suggests a shape. It shows how the features at one scale affect those at another scale. We shall soon see how to derive a complete graphical or

[6]Interestingly, the first exploration of scale-space decomposition was in the area of theoretical economics. James L. Stansfield, working at the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology in the late 1970s, studied zero crossings under Gaussian smoothing while tracking commodity trends ["Conclusions from the commodity expert project," MIT AI Laboratory Memo, No. 601, November 1980].

structural description of the signal from this multiscale concavity information. Furthermore, this structural description enables us to readily build pattern recognition or object matching applications. We can nevertheless use it to perform a time-domain analysis of a signal, identify signal features of different scale, and derive a nonsignal structure that is useful for interpreting the signal. That is the goal of signal analysis, and scale space decomposition is the methodological exemplar.

We will first examine scale space as originally conceived—in analog form. We will consider the type of nonsignal structures that the scale space decomposition produces for a signal. We will state and prove (for simple, but important classes of signals) the theorems that give the method its power. And we shall highlight some of the drawbacks of the classic continuous-time form of scale-space decomposition. Interestingly enough, it was not until some years after the development of the analog scale-space theory that discrete versions of the theory were discovered. We will look at some approaches to discrete scale-space decomposition and close this section with some applications.

4.7.1 Signal Shape, Concavity, and Scale

Scale-space decomposition of a signal begins by smoothing the signal with a family of Gaussian filters. The smoothing operation is a convolution, so it is linear and translation invariant (Chapter 3). Furthermore, all of the smoothing kernels are the same, except for the standard deviation of the Gaussian, σ, which increases with the amount of smoothing performed. This procedure, quite familiar after our experiments with edge detection, produces a series of representations of the signal at different scales or resolutions. The highest resolution (and hence the smallest scale) occurs with the original signal, and we may derive coarser resolution representations of the signal by increasing σ. This collection of smoothed versions constitutes a scale space decomposition of the signal.

The next idea is to look for regions of curvature in the scale space decomposition's signals. The signal derivative remains the principal tool for recovering signal shape. Recall from calculus that the extrema of the nth derivative of a signal d^nf/dt^n are the zeros of its next higher derivative, $d^{(n+1)}f/dt^{(n+1)}$. Consider in particular the sign of the second derivative of a signal. (Withhold for a moment the objection, drawn from general intuition and practical edge detection endeavors, that this derivative is noisy and misleading.) Where $d^2f/dt^2 < 0$, the signal is concave down; where $d^2f/dt^2 = 0$ and $d^3f/dt^3 \neq 0$ there is a zero crossing of the second derivative, or a point of inflection, using calculus parlance; and regions where $d^2f/dt^2 > 0$ are concave up, or convex. Thus, the sign of the second derivative is the basis for signal segmentation (Section 4.1).

Definition (Curvature). Curvature is a measure of how rapidly the graph of a signal, $G = \{(t, f(t)): t \in \mathrm{Dom}(f)\}$ is turning in on itself. More formally, the osculating circle to the graph G at a point $(t_0, f(t_0)) \in G$ is the circle that is tangent to the curve at $(t_0, f(t_0))$. And the curvature κ equals $1/\rho$, where ρ is the radius of the osculating circle to the graph G.

Note that curvature can be obtained by fitting a polynomial to the signal values and computing the derivative of the polynomial model. Alternatively, a circle can be fitted to the signal data. A least-squares formulation of the problem exists and is widely used in the analysis of machined surfaces [33].

Proposition. The curvature has an analog signal representation as well; letting $y = f(t)$ and denoting differentiation by $y'(t)$, it is given by

$$\kappa(t) = \frac{y''(t)}{\left(1+(y'(t))^2\right)^{3/2}}. \tag{4.107}$$

Proof: Calculus; also Ref. 109. ■

From the proposition, we can classify concave-up and concave-down regions of the signal not only by the sign of the curvature (positive and negative, respectively), but also by the magnitude of the curvature. For a candidate signal analysis method, this is an attractive concept for signal structure. It has considerable descriptive power. There is a link with animal vision systems. There is an important geometric connotation as well, via the idea of the osculating circle. The problem with curvature is that, despite its descriptive power and its link with biological vision systems, in the presence of signal noise it is quite problematic. This objection has likely occurred to many readers, and we need to address the issue now. Gaussian smoothing removes the local bumpiness of the signal. At larger scales, when the smoothing Gaussian kernel has a larger variance, the concave and convex regions of the signal that persist must be more significant.

Suppose $f(t)$ is an analog signal and $G(t, \sigma, \mu) = \sigma^{-1}(2\pi)^{-1/2}\exp(-(x-\mu)^2/(2\sigma^2))$ is the Gaussian with standard deviation σ and mean μ. Let $g(t, \sigma) = G(t, \sigma, 0)$. Convolution of $f(t)$ with $g(t, \sigma)$ gives $F(t, \sigma)$:

$$F(t,\sigma) = f(t) * g(t,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{+\infty} f(u)\exp\left(-\frac{(x-u)^2}{2\sigma^2}\right) du. \tag{4.108}$$

We are concerned with the behavior of concavity regions as we vary σ in (4.108). But the regions where the signal is concave down and concave up are separated by those where the second derivative of $F(t, \sigma)$ is zero:

$$\frac{\partial^2}{\partial t^2} F(t,\sigma) = F_{tt}(t,\sigma) = 0. \tag{4.109}$$

Thus, we can track the concavity regions of a signal by simply keeping track of the zero crossings of the second derivative (4.109). Notice that derivatives of $F(t, \sigma)$ can be computed by the convolution of $f(t)$ with the Gaussian's derivative of the same order:

Proposition. Let *f(t)* be an analog signal and let $g(t, \sigma)$ be the zero mean Gaussian with variance σ^2. Then

$$\frac{\partial^n}{\partial t^n} F(t,\sigma) = f * \frac{\partial^n}{\partial t^n} g(t,\sigma). \tag{4.110}$$

Proof: Write out the convolution integral for (4.110) and interchange the order of the differentiation and integration. ■

Before this gets too abstract, let us consider an example. Consider a fourth-degree polynomial with two concave-up regions surrounding a narrow concave-down region. At each scale σ we determine the zero crossings of the second derivative (4.109), and over a range of scales we can draw a contour plot of the zero crossings. Notice that with sufficient smoothing the concave-down region disappears. As σ increases, the locations of the two zero crossings get closer together, eventually meet, and then there is only a convex region. For this example, once we mark the regions as concave or convex, this marking remains; a concave region does not merge with a convex region.

If this behavior is general, then we have a very great simplification in our task of segmenting the various smoothed versions of *f(t)*. All we have to do is follow the zero crossings. Where a pair meet, we know that smoothing has obliterated a convex (or concave) region and two surrounding concave (or convex, respectively) regions will merge. We can use this simplifying assumption if we know that Gaussian smoothing never creates new zero crossings, but may only destroy them as the scale of the smoothing increases. Let's pursue this line of thinking. Consider how the contour plot of zero crossings might look were the smoothing at some scale σ_0 to create a new zero crossing located at time t_0. In this case, we know that for coarser scales, $\sigma > \sigma_0$, there are regions of opposite concavity on either side of time t_0. On which side of t_0 does the concave-up region lie? We have to reexamine the signal smoothed at scale σ each time such a new zero crossing appears during the smoothing process. Depending on the complexity of the signal, this could be quite a chore! Could a large number of zero crossings abruptly appear at some scale? Could the number of regions we have to type according to concavity increase forever as we continue to convolve with Gaussians of ever-wider support? Indeed, what kind of "smoothing" do we have here that puts new wrinkles in our signal as we proceed?

Fortunately, there is a deep theoretical result for scale space decomposition that relieves us of all of these worries. The theorem is that Gaussian smoothing (4.108) never introduces additional structure as the scale parameter *s* increases. That is, new zero crossings of the second derivative of $F(t, \sigma)$ do not appear with increasing σ. There is a converse too: If a convolution kernel never introduces additional structure, then it must be the Gaussian. Together, these two results are the foundation of scale space theory.

4.7.2 Gaussian Smoothing

A variety of factors motivates the use of the Gaussian signal for smoothing a signal. Of course, tradition among signal analysts is one reason for using it. It was one of the smoothing operators used in some of the earliest edge detection efforts, and we found in Section 4.5.3 that the derivative of the Gaussian is an optimal step edge finder. But the Gaussian and its derivatives have a number of attractive properties, and these properties motivate its use for edge detection as well as for the more general scale space approach to signal analysis.

For scale-space decomposition the Gaussian is well-behaved. In particular, it has the following properties:

- (Symmetry Property) It is symmetric and strictly decreasing about its mean.
- As $\sigma \to 0$, $F(t, \sigma) \to f(t)$; that is, for small scales σ, the smoothed signal resembles the original.
- As $\sigma \to \infty$, $F(t, s) \to E(f(t))$; that is, for large scales σ, the smoothed signal approaches the mean of $f(t)$.
- The Gaussian is an $L^1(\mathbb{R})$ signal (absolutely integrable), and it is C^∞ (infinitely differentiable).
- (Causality Property) As σ increases, zero crossings of $F_{tt}(t, \sigma)$ may disappear, but new ones cannot arise.

While the first four of the above properties are quite nice, the Causality Property is so important that it elevates Gaussian smoothing, in a specific sense, to the status of the only possible choice for a scale-space smoothing kernel. And as indicated in the previous section, this property has an important converse, namely, that the Gaussian is unique in this regard.

The following Scale-Space Kernel Conditions formalize the above properties. We need to state these conditions for a general, candidate smoothing kernel $k(t, \sigma)$. We shall invoke these conditions later, in the course of proving our theorems. These are basic properties that we require for the filtering kernel of any scale-based signal decomposition, and so we state the conditions as a definition.

Definition (Scale-Space Kernel Conditions). A function $k(t, \sigma)$ is a scale-space kernel if it satisfies the following five conditions:

1. $k(t, \sigma)$ is the impulse of a linear, translation-invariant system: If $f(t)$ is an analog signal, then the smoothed version of $f(t)$ at scale σ is given by $F(t, \sigma) = f(t)*k(t, \sigma)$.
2. For different values of σ, $k(t, \sigma)$ should always maintain the same fundamental shape: $k(t, \sigma) = (1/\sigma^2)m(t/\sigma)$ for some one-dimensional signal $m(u)$.
3. As σ decreases, $k(t, \sigma)$ approaches the Dirac delta $\delta(t)$, so that $F(t, \sigma)$ approaches $f(t)$: as $\sigma \to 0$, we have $k(t, \sigma) \to \delta(t)$.

4. $k(t, \sigma)$ is an even signal; as $\sigma \to \infty$, or as $t \to \infty$, we have $k(t, \sigma) \to 0$.
5. The Causality Property holds for $k(t, \sigma)$.

While they might appear much too specific and technical upon first inspection, the Scale-Space Kernel Conditions are quite well-motivated. By requiring that the decomposition be linear and translation-invariant, the Convolution Theorem for Analog LTI Systems (Chapter 3) allows us to write the smoothing operation as a convolution.

4.7.2.1 *Sufficiency of the Gaussian.* Let us prove that the Gaussian is sufficient to produce a scale-space decomposition of a signal which does not create zero crossings in the smoothed second derivatives of $F(t, \sigma)$. Again, suppose $G(t, \sigma, \mu) = \sigma^{-1}(2\pi)^{-1/2}\exp(-(x-\mu)^2/(2\sigma^2))$ is the Gaussian with standard deviation σ and mean μ, and set $g(t, \sigma) = G(t, \sigma, 0)$. Suppose that the smoothed signal $F(t, \sigma)$ is given by the convolution of signal $f(t)$ with $g(t, \sigma)$:

$$F(t, \sigma) = f(t) * g(t, \sigma) = \int_{-\infty}^{+\infty} f(u)g(t-u, \sigma)\,du. \tag{4.111}$$

Our five-step proof relies on concepts from calculus:

- We consider the zero crossings of the second derivative $F_{tt}(t, \sigma)$ in (4.111) as curves in the (t, σ) plane.
- This allows us to invoke the implicit function theorem and second derivative conditions for a local maximum along the (t, σ) plane curves.
- We develop some straightforward characterizations of the Causality Property in a proposition.
- Any signal $k(t, \sigma)$ that is a solution of a particular form of the heat diffusion equation also satisfies one of the proposition's equivalent conditions for the Causality Property.
- Finally, since the Gaussian does solve the diffusion equation, we know that it provides a scale-based decomposition that eliminates structure as the scale of the signal smoothing increases.

To begin with, let us vary σ and observe the behavior of zero crossings of the second derivative of $F(t, \sigma)$. Let $E(t, \sigma) = F_{tt}(t, \sigma) = (\partial^2/\partial t^2)F(t, \sigma)$. Zero crossings are solutions of $E(t, \sigma) = 0$, and such pairs form curves in the (t, σ) plane. The curves may extend over the entire range of scales for which smoothing is performed, say from $0 \le \sigma \le \sigma_{\max}$. Possibly, the curve has a local minimum or maximum at a certain time value, $t = t_0$. This situation allows us to write σ as a function of t along the curve: $\sigma = \sigma(t)$. Our structural description for $f(t)$ at scale σ is its segmentation into regions that are concave-up and concave-down, bounded by the zero crossings where $E(t, \sigma) = 0$. The desired Causality Property tells us that such zero crossings cannot increase with scale. Equivalently, this means that as smoothing proceeds, it continues to remove structure from the signal. This is the crucial idea: We can formulate the condition that zero crossings diminish as the scale of smoothing enlarges by examining the behavior of the curves $\sigma(t)$ where $E(t, \sigma) = 0$ is an extremum in (t, σ) space. The next result provides some elementary facts about zero crossing curves. The next proposition provides some basic results, useful for showing that when the scale of Gaussian filtering increases, the structural detail of a signal diminishes.

Proposition (Zero Crossing Conditions). Consider the curve $E(t, \sigma) = 0$ in the (t, σ) plane. Let the time variable t parameterize the curve $\sigma = \sigma(t)$, so that $E(t, \sigma) = F_{tt}(t, \sigma) = F_{tt}(t, \sigma(t))$ is parameterized by t as well. Then:

- The Causality Property holds if and only if each local extremum of $\sigma(t)$ is a local maximum: $\sigma'(t_0) = 0$ implies $\sigma''(t_0) < 0$.
- Along the curve E, $\sigma' = d\sigma/dt = -E_t/E_\sigma = -(\partial E/\partial t)/(\partial E/\partial \sigma)$.
- The Causality Property holds if and only if whenever $\sigma'(t_0) = 0$, then

$$\left.\frac{\partial^2 \sigma}{\partial t^2}\right|_{t=t_0} = -\left.\frac{E_{tt}}{E_\sigma}\right|_{t=t_0} < 0. \tag{4.112}$$

Proof: Condition (i) is a differential calculus formulation of our observation that an arch-shaped curve $E(t, \sigma) = 0$ is acceptable, while a trough-shaped curve is not.

To see the second condition, first note that we can parameterize the curve $E(t, \sigma) = 0$ with some parameter, say u. By the chain rule for vector-valued functions,

$$\frac{dE}{du} = \nabla E \cdot \left(\frac{dt}{du}, \frac{d\sigma}{du} \right) = \frac{\partial E}{\partial t}\frac{dt}{du} + \frac{\partial E}{\partial \sigma}\frac{d\sigma}{du}. \tag{4.113}$$

Since $E(t, \sigma) = 0$ along the curve, the derivative dE/du in (4.113) is also zero along the curve. Next, we may choose the parameterizing variable, $u = t$, the time variable. This little ruse produces

$$\frac{dE}{dt} = 0 = \frac{\partial E}{\partial t}\frac{dt}{dt} + \frac{\partial E}{\partial \sigma}\frac{d\sigma}{dt} = E_t + E_\sigma \frac{d\sigma}{dt}, \tag{4.114}$$

which gives condition (ii).

Condition (iii) follows from the first two. Indeed, applying the quotient rule for derivatives to condition (ii), we find

$$\frac{d^2\sigma}{dt^2} = -\frac{E_{tt}}{E_\sigma} + \frac{E_t}{(E_\sigma)^2}. \tag{4.115}$$

From (4.114), note that $\sigma'(t_0) = 0$ if and only if $E_t = 0$ at $t = t_0$. Thus, (4.115) ensures yet another equivalent condition: $\sigma''(t_0) = -(E_{tt}/E_\sigma)$ at $t = t_0$. The inequality in condition (iii) follows from condition (i), completing the proof. ■

This next theorem is a basic theoretical result in scale-space decomposition. It shows that a Gaussian filtering kernel can be the foundation for a scale-based signal decomposition method, which removes signal structure as the scale of the smoothing increases.

Theorem (Sufficiency). Suppose that convolution with the Gaussian, $g(t, \sigma) = G(t, \sigma, 0) = 2^{-1/2}\sigma^{-1}\exp(-t^2/(2\sigma^2))$, smoothes the signal $f(t)$ at scale σ to produce $F(t, \sigma)$:

$$F(t,\sigma) = f(t) * g(t,\sigma) = \int_{-\infty}^{+\infty} f(u)g(t-u,\sigma)\,du = \frac{1}{\sigma\sqrt{2}}\int_{-\infty}^{+\infty} f(u)e^{-\frac{(t-u)}{2\sigma^2}}\,du. \tag{4.116}$$

Then the Causality Property holds; that is, $F_{tt}(t, \sigma)$ zero crossings may disappear—but can never appear—as σ increases.

Proof: Consider the partial differential equation

$$\frac{\partial^2 U}{\partial t^2} = \frac{1}{\sigma}\frac{\partial U}{\partial \sigma}. \tag{4.117}$$

This is a form of the heat or diffusion equation from physics, introduced already in Chapter 1. We can easily check that the Gaussian $g(t, \sigma)$ solves (4.117) by calculating:

$$g_{tt}(t,\sigma) = \frac{\partial^2 g}{\partial^2 t} = \left(\frac{t^2}{\sigma^4} - \frac{1}{\sigma^2}\right)g(t,\sigma) \tag{4.118}$$

and

$$g_\sigma(t,\sigma) = \frac{\partial g}{\partial \sigma} = \left(\frac{t^2}{\sigma^3} - \frac{1}{\sigma}\right)g(t,\sigma). \tag{4.119}$$

so that (4.117) holds for $U(t, \sigma) = g(t, \sigma)$. However, we can show that $F(t, \sigma)$ and hence $E(t, \sigma)$ satisfy the diffusion equation as well. In fact, since

$$\begin{aligned} F_{tt}(t,\sigma) &= \frac{\partial^2}{\partial t^2}\int_{-\infty}^{+\infty} f(u)g(t-u,\sigma)\,du = \int_{-\infty}^{+\infty} f(u)g_{tt}(t-u,\sigma)\,du = \int_{-\infty}^{+\infty} f(u)\frac{1}{\sigma}g_{\sigma}(t-u,\sigma)\,du \\ &= \frac{1}{\sigma}\frac{\partial}{\partial\sigma}\int_{-\infty}^{+\infty} f(u)g(t-u,\sigma)\,du = \frac{1}{\sigma}F_{\sigma}(t,\sigma) \end{aligned} \tag{4.120}$$

we have $E_{tt} = F_{tttt} = (1/\sigma)F_{\sigma tt} = (1/\sigma)F_{tt\sigma} = (1/\sigma)E_{\sigma}$. This shows that $E(t, \sigma)$ satisfies the diffusion equation (4.117) and the weaker inequality (4.112). A Gaussian kernel thereby guarantees that increasing the scale of smoothing creates no new signal structure. This completes the sufficiency proof. ■

4.7.2.2 Necessity of the Gaussian. It is quite a bit harder to show that the Gaussian is the only kernel that never allows new signal structure to arise as the scale of smoothing increases. The necessity proof involves several steps:

- We consider a candidate filtering kernel $k(t, \sigma)$ and represent the signal to be analyzed, $f(t)$, as a sum of Dirac delta functions.
- Because of the Sifting Property of the Dirac delta (Chapter 3), this transposes the convolution integral into a discrete sum, and, using the Zero Crossing Conditions Proposition above, there arises a set of simultaneous linear equations, $Ax = b$.
- If these simultaneous equations have a solution, then we can find a signal $f(t)$ for which the proposed kernel $k(t, \sigma)$ creates new structure as σ increases.
- We show that we can always solve the simultaneous equations unless the proposed kernel $k(t, \sigma)$ satisfies a special differential equation, which is a general form of the diffusion equation.
- We prove that the only kernel that satisfies this differential equation is the Gaussian.
- Thus, if the filtering kernel $k(t, \sigma)$ is not Gaussian, it cannot be a solution to the special differential equation; this implies that we can solve the simultaneous linear equations; and this solution at long last reveals a signal, $f(t)$, for which our proposed $k(t, \sigma)$ creates at least one new inflection point during smoothing.

Let's follow the above plan, beginning with some technical lemmas. We show how to derive a set of simultaneous equations by representing $f(t)$ as a sum of Dirac delta functions and using the Zero Crossing Conditions. From the discussion of the Dirac delta's Sifting Property (Chapter 3), $f(t)$ can be represented as the limit of such a sum.

Lemma (Zero Crossing Conditions for Dirac Sum Signals). Suppose that $k(t, \sigma)$ satisfies the Scale-Space Kernel Conditions and that the signal $f(t)$ has the following representation as a sum of Dirac delta functions:

$$f(t) = \sum_{i=1}^{n} c_i \delta(t - t_i). \tag{4.121}$$

As before we define

$$F(t,\sigma) = f(t) * k(t, \sigma) = \int_{-\infty}^{+\infty} f(u) k(t-u,\sigma)\, du, \tag{4.122}$$

and we set $E(t, \sigma) = F_{tt}(t, \sigma)$. To avoid excess subscript clutter, we also define $M(t, \sigma) = k_{tt}(t, \sigma) = (\partial^2/\partial t^2)k(t, \sigma)$. If $E(t_0, \sigma) = 0$ for some $t = t_0$, then

$$\sum_{i=1}^{n} c_i M(t_0 - t_i, \sigma) = 0, \tag{4.123}$$

and at an extremum (t_0, σ) of the curve $E(t, \sigma) = 0$ the following equations hold:

$$\sum_{i=1}^{n} c_i M_t(t_0 - t_i, \sigma) = 0 \tag{4.124}$$

and

$$\frac{\sum_{i=1}^{n} c_i M_{tt}(t_0 - t_i, \sigma) = 0}{\sum_{i=1}^{n} c_i M_{\sigma}(t_0 - t_i, \sigma) = 0} > 0. \tag{4.125}$$

Proof: Since $f(t)$ is a sum of Dirac delta functions (4.121), the Sifting Property of the delta function implies

$$\begin{aligned} E(t,\sigma) &= \frac{\partial}{\partial t^2} \int_{-\infty}^{\infty} f(u) k(t-u,\sigma)\, du = \int_{-\infty}^{\infty} \left\{ \sum_{i=1}^{n} c_i \delta(t - t_i) \right\} \frac{\partial}{\partial t^2} k(t-u,\sigma)\, du \\ &= \sum_{i=1}^{n} c_i k_{tt}(t_i, \sigma) = \sum_{i=1}^{n} c_i M(t_i, \sigma). \end{aligned} \tag{4.126}$$

At a point (t_0, σ) on a zero crossing curve, $E(t_0, \sigma) = 0$, so (4.123) follows from (4.126). Furthermore, the Zero Crossing Conditions Proposition showed that at an extreme point (t_0, σ) on the zero crossing curve we must have $E_t(t_0, \sigma) = 0$, and by

(4.126) this entails (4.124). Finally, note that (4.125) follows from equation (4.112) of the same proposition, and the proof is complete. ■

Corollary (Necessary Linear Equations). Let the lemma's assumptions still hold and let $t_0, t_1, t_2, \ldots, t_n$ be arbitrary. Then, for $P < 0$, the following four simultaneous equations,

$$\begin{bmatrix} M(t_0-t_1,\sigma) & M(t_0-t_2,\sigma) & \cdots & M(t_0-t_n,\sigma) \\ M_t(t_0-t_1,\sigma) & M_t(t_0-t_2,\sigma) & \cdots & M_t(t_0-t_n,\sigma) \\ M_{tt}(t_0-t_1,\sigma) & M_{tt}(t_0-t_2,\sigma) & \cdots & M_{tt}(t_0-t_n,\sigma) \\ M_\sigma(t_0-t_1,\sigma) & M_\sigma(t_0-t_2,\sigma) & \cdots & M_\sigma(t_0-t_n,\sigma) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ P \\ 1 \end{bmatrix}, \qquad (4.127)$$

have no solution $(c_1, c_2, ..., c_n)$.

Proof: The existence of $(c_1, c_2, ..., c_n)$ satisfying (4.127) with $P < 0$ violates (4.125). ■

Once we assume that $f(t)$ is a sum of Dirac delta functions, the lemma and its corollary show that the Causality Property imposes extremely strong conditions on the filtering kernel. In (4.127), finding $(c_1, c_2, ..., c_n)$ is equivalent to finding $f(t)$, and it would appear that solutions for such an underdetermined set of linear equations should be plentiful. If for some $t_0, t_1, t_2, \ldots, t_n, P$, with P < 0, we could discover a solution $(c_1, c_2, ..., c_n)$, then this would give us a signal $f(t)$ and a location in scale space (t_0, σ) at which the kernel $k(t, \sigma)$ fails the Causality Property: New structure unfolds when σ increases at (t_0, σ). The matrix of second-, third-, and fourth-order partial derivatives in (4.127) must be very special indeed if a candidate smoothing kernel is to support the Causality Property. Our goal must be to show that the Causality Property guarantees that for any $t_0, t_1, t_2, \ldots, t_n, P < 0$, (4.127) defies solution. The next proposition recalls a linear algebra result that helps bring the peculiarities of (4.127) to light.

Proposition. Let $\boldsymbol{M}$ be an $m \times n$ matrix, let $\boldsymbol{b}$ be an $m \times 1$ vector, and let $[\boldsymbol{M} \mid \boldsymbol{b}]$ be the $m \times (n+1)$ matrix whose first n columns are the same as $\boldsymbol{M}$ and whose last column is $\boldsymbol{b}$. Then the following are equivalent:

(i) The equation $\boldsymbol{Mx} = \boldsymbol{b}$ has a solution.

(ii) Rank $\boldsymbol{M}$ = rank $[\boldsymbol{M} \mid \boldsymbol{b}]$; that is,

$$\operatorname{rank}\begin{bmatrix} M_{1,1} & M_{1,2} & \cdots & M_{1,n} \\ M_{2,1} & \cdots & \cdots & M_{2,n} \\ \vdots & \cdots & \cdots & \vdots \\ M_{m,1} & M_{m,2} & \cdots & M_{m,n} \end{bmatrix} = \operatorname{rank}\begin{bmatrix} M_{1,1} & M_{1,2} & \cdots & M_{1,n} & b_1 \\ M_{2,1} & \cdots & \cdots & M_{2,n} & b_2 \\ \vdots & \cdots & \cdots & \vdots & \vdots \\ M_{m,1} & M_{m,2} & \cdots & M_{m,n} & b_m \end{bmatrix}. \qquad (4.128)$$

(iii) For all vectors $\boldsymbol{y} = (y_1, y_2, ..., y_m)$: If $\langle \boldsymbol{y}, \boldsymbol{b}\rangle = 0$, then $y_1(M_{1,1}, M_{1,2}, ..., M_{1,n}) + y_2(M_{2,1}, M_{2,2}, ..., M_{2,n}) + \cdots + y_m(M_{m,1}, M_{m,2}, ..., M_{m,n}) = 0$.

Proof: Recall that the column space of the matrix $\boldsymbol{M}$ is the set of vectors spanned by the column vectors of $\boldsymbol{M}$, and the row space of is the set of vectors spanned by rows of $\boldsymbol{M}$. From linear algebra, the dimensions of these two spaces are the same—the rank of $\boldsymbol{M}$. Now, (i) is clearly equivalent to (ii), because (i) is true if and only if $\boldsymbol{b}$ is in $\boldsymbol{M}$'s column space. Also note that the row space of $\boldsymbol{M}$ must have the same dimension as the row space of $[\boldsymbol{M}\,|\,\boldsymbol{b}]$. So if vector $\boldsymbol{y} = (y_1, y_2, ..., y_m)$ is such that the linear combination of rows of $\boldsymbol{M}$, $y_1(M_{1,1}, M_{1,2}, ..., M_{1,n}) + y_2(M_{2,1}, M_{2,2}, ..., M_{2,n}) + \cdots + y_m(M_{m,1}, M_{m,2}, ..., M_{m,n}) = 0$, and also $y_1(M_{1,1}, M_{1,2}, ..., M_{1,n}, b_1) + y_2(M_{2,1}, M_{2,2}, ..., M_{2,n}, b_2) + ... + y_m(M_{m,1}, M_{m,2}, ..., M_{m,n}, b_m)$ is nonzero in the last component (that is, $\langle \boldsymbol{y}, \boldsymbol{b}\rangle \neq 0$), then the dimension of the row space of $[\boldsymbol{M}\,|\,\boldsymbol{b}]$ would exceed the dimension of the row space of $\boldsymbol{M}$, a contradiction. Thus, (ii) entails (iii). Finally, (iii) says that any vector $\boldsymbol{y} \perp \boldsymbol{b}$ must also be orthogonal to every column of $\boldsymbol{M}$. This means that $\boldsymbol{b}$ is in the column space of $\boldsymbol{M}$, $\boldsymbol{Mx} = \boldsymbol{b}$ is solvable, and the proof is complete. ■

The next proposition reveals a differential equation that scale space kernels must satisfy.

Proposition (Necessary Differential Equation). Suppose that $k(t, \sigma)$ satisfies the Scale-Space Kernel Conditions and that the signal $f(t)$ has a representation as a sum of Dirac delta functions (4.121), as in the lemma. Then there are constants, A, B, C, and D, with D/C > 0, such that

$$\frac{Ak(t,\sigma)}{\sigma^2} + \frac{Bk_t(t,\sigma)}{\sigma} + Ck_{tt}(t,\sigma) = \frac{Dk_\sigma(t,\sigma)}{\sigma}. \tag{4.129}$$

Proof: Consider the vector $\boldsymbol{b}$, where $\boldsymbol{b} = (0, 0, P, 1)$ and $P > 0$. Note that it is easy to find a vector $\boldsymbol{y} = (y_1, y_2, y_3, y_4)$ such that $\langle \boldsymbol{y}, \boldsymbol{b}\rangle = 0$. Applying the previous proposition and the Necessary Linear Equations Corollary, it follows that $y_1M_{1,1}(t_0 - t_1) + y_2M_t(t_0 - t_1) + y_3M_{tt}(t_0 - t_1) + y_4M_\sigma(t_0 - t_1) = 0$. Since (4.127) has a solution for any $t_0, t_1, t_2, \ldots, t_n$ we may write this as $y_1M_{1,1}(t) + y_2M_t(t) + y_3M_{tt}(t) + y_4M_\sigma(t) = 0$. Let $A = y_1\sigma^2$, $B = y_2\sigma$, $C = y_3$, and $D = -y_4\sigma$. Then,

$$\frac{AM(t,\sigma)}{\sigma^2} + \frac{BM_t(t,\sigma)}{\sigma} + CM_{tt}(t,\sigma) = \frac{DM_\sigma(t,\sigma)}{\sigma}. \tag{4.130}$$

Observe that $D/C = (-y_4\sigma)/y_3 = P\sigma > 0$. We require, however, that $k(t, \sigma)$, not just its second derivative, $M(t, \sigma)$, satisfy the differential equation. Any two filters with second derivatives satisfying (4.130) must differ by a function with zero second derivative. That is, their difference is a linear term. Since the Scale-Space Kernel Conditions require that as $t \to \infty$, we have $k(t, \sigma) \to 0$, this linear term must be identically zero. Thus, $k(t, \sigma)$ satisfies (4.129). ■

Equation (4.129) is a general form of the heat or diffusion equation. We have come a long way and have taken nearly all the steps toward proving that a filtering kernel which obeys the Scale-Space Kernel Conditions is necessarily Gaussian. The next theorem solves the generalized heat equation. To accomplish this, we must make use of the analog Fourier transform, the formal presentation of which will not be given until the next chapter. We fancy that many readers are already familiar with this technique for solving differential equations. We apologize to the rest for asking them to see into the future and all the more so for suggesting a premonition of the Fourier transform!

Some readers may wish to skip the proofs of the next two theorems, perhaps because the heat equation's solution through Fourier transformation is already familiar, or perhaps to return to them after assimilating Chapter 5's material. The first result solves the generalized heat equation. Because the solution involves an integration, it is less than satisfying, however. The second theorem remedies this. We apply the Scale-Space Kernel Conditions to our general solution and derive a result that clearly shows the Gaussian nature of all structure reducing scale-space filtering kernels.

Theorem (Generalized Heat Equation). If $k(t, \sigma)$ is a solution to the differential equation (4.129), then $k(t, \sigma)$ is the Gaussian

$$k(t,\sigma) = \sigma^{1+\frac{a}{d}}\sqrt{D/2\pi C}\int_{-\infty}^{+\infty} x(t-u)e^{-\frac{D}{2C}\left(\frac{u}{\sigma}+\frac{B}{D}\right)^2}\,du. \tag{4.131}$$

Proof: Let us first simplify (4.129) with the substitution $k(t, \sigma) = \sigma^{a/d}q(t, \sigma)$. This provides a heat equation in more familiar form,

$$\frac{Bq_t(t,\sigma)}{\sigma} + Cq_{tt}(t,\sigma) = \frac{Dq_\sigma(t,\sigma)}{\sigma}, \tag{4.132}$$

which we must solve for $q(t, \sigma)$. Further simplifications are possible, but we need to reach into the next chapter for the representation of a signal by the analog Fourier transform. The normalized radial Fourier transform of a signal $x(t)$ is given by

$$X(\omega) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} x(t)e^{-j\omega t}\,dt, \tag{4.133}$$

where $j^2 = -1$. Here we adopt a widely used convention that lowercase letters stand for time-domain signals and that uppercase letters stand for their Fourier transform counterpart. In the near future, we shall verify that if $x(t)$ is absolutely integrable or has finite energy, then its Fourier transform $X(\omega)$ exists (4.133). Likewise, if $X(\omega) \in l^1(\mathbb{R})$ or $X(\omega) \in l^2(\mathbb{R})$, then an inverse normalized radial Fourier transform exists, which is given by

$$x(t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} X(\omega)e^{j\omega t}\,d\omega. \tag{4.134}$$

The key idea is to insert the representation of $q(t, \sigma)$ by its transform from $Q(\omega, \sigma)$ into the simplified heat equation (4.132). Then, after removing $(2\pi)^{-1}$ from each term, (4.132) becomes

$$\frac{B}{\sigma}\frac{\partial}{\partial t}\int_{-\infty}^{+\infty} Q(\omega,\sigma)e^{j\omega t}\,d\omega + C\frac{\partial^2}{\partial t^2}\int_{-\infty}^{+\infty} Q(\omega,\sigma)e^{j\omega t}\,d\omega = \frac{D}{\sigma}\frac{\partial}{\partial\sigma}\int_{-\infty}^{+\infty} Q(\omega,\sigma)e^{j\omega t}\,d\omega. \tag{4.135}$$

Interchanging the order of integration and differentiation in (4.135) gives

$$\frac{B}{\sigma}\int_{-\infty}^{+\infty} Q(\omega,\sigma)(j\omega)e^{j\omega t}\,d\omega + C\int_{-\infty}^{+\infty} Q(\omega,\sigma)(j\omega)^2 e^{j\omega t}\,d\omega = \frac{D}{\sigma}\int_{-\infty}^{+\infty} Q_\sigma(\omega,\sigma)e^{j\omega t}\,d\omega. \tag{4.136}$$

By the existence of the inverse Fourier transform, the integrands on either side of (4.136) must be equal. After a bit of algebra, the differential equation simplifies considerably:

$$\frac{B\omega j}{D}Q(\omega,\sigma) - \frac{C\sigma\omega^2}{D}Q(\omega,\sigma) = Q_\sigma(\omega,\sigma). \tag{4.137}$$

Let us now separate the variables in (4.137), to obtain

$$\left(\frac{B\omega j}{D} - \frac{C\sigma\omega^2}{D}\right)\partial\sigma = \frac{\partial Q}{Q}. \tag{4.138}$$

We integrate both sides from 0 to r,

$$\int_0^r\left(\frac{B\omega j}{D} - \frac{C\sigma\omega^2}{D}\right)\partial\sigma = \int_0^r\frac{\partial Q}{Q}. \tag{4.139}$$

where $r > 0$ is an integration limit, and remove the logarithms arising from the integration on the right-hand side of (4.139) by exponentiation. Letting $r = \sigma$ then gives

$$Q(\omega,\sigma) = Q(\omega,0)e^{\frac{jB\omega\sigma}{D}}e^{\frac{-C\omega^2\sigma^2}{2D}}. \tag{4.140}$$

Thus we have found the Fourier transform of $q(t, \sigma) = \sigma^{-a/d}k(t, \sigma)$. We can find $q(t, \sigma)$ by once again applying the inverse Fourier transform operation,

$$q(t,\sigma) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} Q(\omega,0)e^{\frac{jB\omega\sigma}{D}}e^{\frac{-C\omega^2\sigma^2}{2D}}e^{j\omega t}\,d\omega, \tag{4.141}$$

which is very close to the form we need. To continue simplifying, we utilize some further properties of the Fourier transform. Notice two things about the integrand in (4.141): It is a product of frequency-domain signals (i.e., their independent variable is ω), and one of the frequency-domain terms is a frequency-domain Gaussian, namely $\exp(-C\omega^2\sigma^2/2D)$. Therefore, one of the properties that we can now apply is the Convolution Theorem for Fourier Transforms. It says that the Fourier transform of a convolution, $s = x*y$, is the attenuated product of the Fourier Transforms; specifically, $S(\omega) = (2\pi)^{-1/2}X(\omega)Y(\omega)$. The second property that comes to mind from inspecting the integrand in (4.141) is the formula for the Fourier transform of the Gaussian signal. This states that if $\lambda > 0$, then the Fourier Transform of the Gaussian $g(t) = \exp(-\lambda t^2)$ is $G(\omega) = (2\lambda)^{-1/2}\exp(-\omega^2/4\lambda)$. We let $X(\omega) = Q(\omega, 0)$ and $Y(\omega, \sigma) = \exp(jB\omega\sigma/D)\exp(-C\omega^2\sigma^2/2D)$. Then $q(t, \sigma)$ is the inverse Fourier transform of $X(\omega)Y(\omega, \sigma)$, so we have $q(t, \sigma) = (2\pi)^{-1/2}x*y$. That is,

$$q(t,\sigma) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} x(t-u)y(u,\sigma)\, du, \tag{4.142}$$

where the Fourier transforms of $x(t)$ and $y(t, \sigma)$ are $X(\omega)$ and $Y(\omega, \sigma)$, respectively. Let us set aside the mysterious signal $x(t)$ for a moment and consider the $y(t, \sigma)$ factor in (4.142). Since the Fourier transform of $s(t, \sigma) = \sigma(D/C)^{1/2}\exp(-Dt^2/(2C\sigma^2))$ is $\exp(-C\sigma^2\omega^2/2D)$, it can easily be shown that the Fourier transform of $y(t) = s(t+B\sigma/D)$ is $Y(\omega, \sigma)$. Thus,

$$q(t,\sigma) = \sigma\sqrt{D/2\pi C} \int_{-\infty}^{+\infty} x(t-u)e^{-\frac{D}{2C}\left(\frac{u}{\sigma}+\frac{B}{D}\right)^2} du, \tag{4.143}$$

and, recalling that $k(t, \sigma) = \sigma^{a/d}q(t, \sigma)$, there follows (4.131), completing the proof of the theorem. ■

Now, the next theorem applies the Scale-Space Kernel Conditions to (4.131).

Theorem (Necessity of the Gaussian). Suppose that $k(t, \sigma)$ satisfies the Scale-Space Kernel Conditions and that the signal $f(t)$ has a representation as a sum of Dirac delta functions (4.121), as in the lemma. Then $k(t, \sigma)$ is a Gaussian of the form

$$k(t,\sigma) = \sigma^{1+\frac{a}{d}}\sqrt{D/2\pi C} \int_{-\infty}^{+\infty} x(t-u)e^{-\frac{D}{2C}\left(\frac{u}{\sigma}+\frac{B}{D}\right)^2} du. \tag{4.144}$$

Proof: Recall the time-delayed Gaussian $y(t) = s(t+B\sigma/D)$ from the previous proof. Since the smoothing filter must not permit zero crossings to shift as the scale of smoothing varies, this Gaussian must be centered at the origin; in other words, $B\sigma/D = 0$, and hence $B = 0$. Thus,

$$k(t,\sigma) = \sigma^{1+\frac{a}{d}} \sqrt{D/2\pi C} \int_{-\infty}^{+\infty} x(t-u) e^{-\frac{Du^2}{2C\sigma^2}}\, du. \tag{4.145}$$

As the scale of smoothing decreases, the smoothed signal must resemble the original, $f(t)$, and so in the limit, as $\sigma \to 0$, the Scale-Space Kernel Conditions demand that $k(t, \sigma) \to \delta(t)$, the Dirac delta. The consequence of this condition is

$$\begin{aligned}\lim_{\sigma\to 0} k(t,\sigma) &= \lim_{\sigma\to 0} \sigma^{1+\frac{a}{d}} \sqrt{\frac{D}{2\pi C}} \int_{-\infty}^{+\infty} e^{-\frac{D(t-u)^2}{2C\sigma^2}} x(u)\, du \\ &= \lim_{\sigma\to 0} \sigma^{\frac{a}{d}} \sqrt{\frac{D}{2\pi C}} \int_{-\infty}^{+\infty} \sigma e^{-\frac{D(t-u)^2}{2C\sigma^2}} x(u)\, du = \delta(t).\end{aligned} \tag{4.146}$$

This completes the proof. ■

This proof contains a valuable lesson: A judicious application of the Fourier integral representation removes the differential equation's partial time derivatives. Higher powers of the frequency variable, ω, replace the derivatives, but this is tolerable because it leads to a simpler differential equation overall. This is a powerful—and quite typical—application of the Fourier transform. Applied mathematicians usually resort to the Fourier transformation for precisely this purpose and no others. For our own purposes, however, the Fourier transform does much more than expedite computations; it is the principal tool for studying the frequency content of analog signals. The next three chapters, no less, explore Fourier theory in detail: analog signal frequency in Chapter 5, discrete signal frequency in Chapter 7, and frequency-domain signal analysis in Chapter 9.

The efforts of many researchers have helped to elucidate the theory of scale-space decompositions. Our treatment here follows most closely the presentation of one-dimensional scale-space decomposition by Yuille and Poggio [110]. Other theoretical studies of scale-space decomposition include Ref. 111. Applications of scale space filtering include the recognition of two-dimensional shapes by extracting object boundaries, formulating the boundary as a curvature signal, and deriving the scale-space representation of the boundary curvature [112]. There results a graph structure, which can be matched against model graphs using artificial intelligence techniques, or matched level-by-level using the structural pattern detection techniques of Section 4.6.2.

Now, when an image is represented in its scale-space decomposition, no arbitrary preferred scale for the objects represented in the decomposed is used. Instead, the set of scales at which distinct regions of different curvature sign arise and disappear is found. Then the changes in the curvature are related to one another in a structure called an interval tree by Witkin. This further abstracts the structural description of the signal from its actual value, $f(x)$, by defining an interval of scales, and an interval of space, x, over which the salient curvature features of the signal are to be found. The signal is segmented in both scale and spatial coordinates.

4.8 SUMMARY

In the introduction to this chapter, we reflected on the distinction, sometimes subtle and sometimes profound, between signal processing and signal analysis. This chapter's methods work primarily with operations on the signal values in the time domain. In some cases, such as the problem of extracting periodicities from the signal values, statistics on signal levels proved to be inadequate. We resorted to comparisons, in the form of inner products, of the given signal to sinusoidal or exponential models. This makes the break with time-domain methods into the realm of frequency-domain methods. Now, we need to look deeper into the theory of analog signal frequency, discrete signal frequency, and the applications that arise from these studies. This task will occupy us for the next five chapters. Then we will consider the combination of the methods of both domains: Time-frequency transforms are the subject of Chapter 10, and time-scale transforms are covered in Chapter 11. This chapter is preparation, with a time-domain perspective, for Chapter 9 on frequency-domain signal analysis and for Chapter 12 on mixed-domain analysis.

REFERENCES

1. F. Jelinek, Continuous speech recognition by statistical methods, *Proceedings of the IEEE*, vol. 64, no. 4, pp. 532–556, April 1976.
2. S. Young, A review of large-vocabulary continuous-speech recognition, *IEEE Signal Processing Magazine*, pp. 45–57, September 1996.
3. K. Lee and R. Reddy, *Automatic Speech Recognition*, Boston: Kluwer Academic Publishers, 1989.
4. L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Englewood Cliffs, NJ: Prentice-Hall, 1993.
5. R. L. Wesson and R. E. Wallace, Predicting the next great earthquake in California, *Scientific American*, pp. 35–43, February 1985.
6. Ö. Yilmaz, *Seismic Data Processing*, Tulsa, OK: Society for Exploratory Geophysics, 1987.
7. M. Akay, *Biomedical Signal Processing*, San Diego, CA: Academic Press, 1994.
8. R. Isermann, Process fault detection based on modeling and estimation methods—a survey, *Automatica*, vol. 20, no. 4, pp. 387–404, 1984.
9. C. C. Tappert, C. Y. Suen, and T. Wakahara, The state of the art in on-line handwriting recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 12, no. 8, pp. 787–808, 1990.

10. A. Marcus and A. Van Dam, User interface developments in the nineties, *IEEE Computer,* pp. 49–57, September 1991.
11. J. Preece, *Human-Computer Interaction*, Reading, MA: Addison-Wesley, 1994.
12. S. L. Horowitz and T. Pavlidis, Picture segmentation in a directed split-and-merge procedure, *Proceedings of the Second International Joint Conference on Pattern Recognition,* Copenhagen, pp. 424–433, August 1974.
13. P. D. van der Puije, *Telecommunication Circuit Design*, New York: Wiley, 1992.
14. P. Mock, Add DTMF generation and decoding to DSP-μP designs, *Electronic Design News*, March 21, 1985; also in K.-S. Lin, ed., *Digital Signal Processing Applications with the TMS320 Family*, vol. 1, Dallas: Texas Instruments, 1989.
15. A. Mar, ed., *Digital Signal Processing Applications Using the ADSP-2100 Family*, Englewood Cliffs, NJ: Prentice-Hall, 1990.
16. J. L. Flanagan, *Speech Analysis, Synthesis, and Perception*, 2nd ed., New York: Springer-Verlag, 1972.
17. C. K. Chow and T. Kaneko, Automatic boundary detection of the left ventricle from cineangiograms, *Computers and Biomedical Research*, vol. 5, pp. 388–410, 1972.
18. N. Otsu, A threshold selection method from gray-level histograms, *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-9, no. 1, pp. 62–66, January 1979.
19. J. Kittler and J. Illingworth, On threshold selection using clustering criteria, *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-15, pp. 652–655, 1985.
20. S. Kullback and R. A. Leibler, On information and sufficiency, *Annals of Mathematical Statistics,* vol. 22, pp. 79–86, 1951.
21. S. Kullback, *Information Theory and Statistics*, Mineola, NY: Dover, 1997.
22. J. S. Weszka, A survey of threshold selection techniques, *Computer Graphics and Image Processing,* vol. 7, pp. 259–265, 1978.
23. P. K. Sahoo, S. Soltani, and A. K. C. Wong, A survey of thresholding techniques, *Computer Vision, Graphics, and Image Processing*, vol. 41, pp. 233–260, 1988.
24. P. W. Palumbo, P. Swaminathan, and S. N. Srihari, Document image binarization: evaluation of algorithms, *Applications of Digital Image Processing IX*, SPIE, vol. 697, pp. 278–285, 1986.
25. M. Kamel and A. Zhao, Extraction of binary character/graphics images from grayscale document images, *CVGIP: Graphical Models and Image Processing*, vol. 55, no. 3, pp. 203–217, 1993.
26. O. D. Trier and T. Taxt, Evaluation of binarization methods for document images, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 17, no. 3, pp. 312–315, March 1995.
27. J. M. White and G. D. Rohrer, Image thresholding for character image extraction and other applications requiring character image extraction, *IBM Journal of Research and Development*, vol. 27, no. 4, pp. 400–411, July 1983.
28. D. L. Duttweiler, A twelve-channel digital voice echo canceller, *IEEE Transactions on Communications*, vol. COM-26, no. 5, pp. 647–653, May 1978.
29. P. Brodatz, *Textures: A Photographic Album for Artists and Designers,* New York: Dover, 1966.
30. B. Julesz, Experiments in the visual perception of texture, *Scientific American*, vol. 232, pp. 34–43, 1975.

31. J. Beck, Similarity grouping and peripheral discrimination under uncertainty, *American Journal of Psychology*, vol. 85, pp. 1–19, 1972.
32. M. Wertheimer, Principles of perceptual organization, in *Readings in Perception*, D. C. Beardslee and M. Wertheimer, eds., Princeton, NJ: Van Nostrand-Reinhold, 1958.
33. D. J. Whitehouse, *Handbook of Surface Metrology*, Bristol, UK: Institute of Physics, 1994.
34. J. Beck, A. Sutter, and R. Ivry, Spatial frequency channels and perceptual grouping in texture segmentation, *Computer Vision, Graphics, and Image Processing*, vol. 37, pp. 299–325, 1987.
35. F. Tomita, Y. Shirai, and S Tsuji, Description of texture by a structural analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-4, no. 2, pp. 183–191, 1982.
36. *Surface Texture: Surface Roughness, Waviness and Lay*, ANSI B46.1, American National Standards Institute, 1978.
37. *Surface Roughness—Terminology—Part 1: Surface and its Parameters*, ISO 4287/1, International Standards Organization, 1984.
38. *Surface Roughness—Terminology—Part 2: Measurement of Surface Roughness Parameters, ISO 4287/2*, International Standards Organization, 1984.
39. R. W. Conners and C. A. Harlow, A theoretical comparison of texture algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-2, no. 3, pp. 204–222, May 1980.
40. A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, vol. 2, Orlando, FL: Academic Press, 1982.
41. B. Julesz, Visual pattern discrimination, *IRE Transactions on Information Theory*, vol. IT-8, no. 2, pp. 84–92, February 1961.
42. B. Julesz, E. N. Gilbert, and J. D. Victor, Visual discrimination of textures with identical third-order statistics, *Biological Cybernetics*, vol. 31, no. 3, pp. 137–140, 1978.
43. A. Gagalowicz, A new method for texture field synthesis: Some applications to the study of human vision, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-3, no. 5, pp. 520–533, September 1981.
44. R. M. Haralick, K. S. Shanmugam, and I. Dinstein, Textural features for image classification, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, November 1973.
45. S. W. Zucker and D. Terzopoulos, Finding structure in co-occurrence matrices for texture analysis, *Computer Graphics and Image Processing*, vol. 12, no. 3, pp. 286–308, March 1980.
46. D. Chetverikov, Measuring the degree of texture regularity, *Proceedings of the International Conference on Pattern Recognition*, Montreal, pp. 80–82, 1984.
47. J. S. Weszka, C. R. Dyer, and A. Rosenfeld, A comparative study of texture measures for terrain classification, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, pp. 269–285, 1976.
48. G. Strang, *Introduction to Applied Mathematics*, Wellesley, MA: Wellesley-Cambridge, 1986.
49. A. Savitsky and M. J. E. Golay, Smoothing and differentiation of data by simplified least squares procedures, *Analytical Chemistry*, vol. 36, pp. 1627–1639, 1964.

50. C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, Englewood Cliffs, NJ: Prentice-Hall, 1974.

51. R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. 1, New York: Addison-Wesley, 1992.

52. A. Jennings and J. J. McKeown, *Matrix Computation*, 2nd ed., Chichester, West Sussex, England: Wiley, 1992.

53. E. R. Dougherty and P. A. Laplante, *Introduction to Real-Time Imaging*, Bellingham, WA: SPIE, 1995.

54. N. C. Gallagher, Jr. and G. L. Wise, A theoretical analysis of the properties of median filters, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, pp. 1136–1141, 1981.

55. R. M. Haralick, S. R. Sternberg, and X. Zhuang, Image analysis using mathematical morphology, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, pp. 532–550, July 1987.

56. M.-H. Chen and P.-F. Yan, A multiscaling approach based on morphological filtering, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 694–700, July 1989.

57. P. Maragos, Pattern spectrum and multiscale shape representation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 701–716, July 1989.

58. L. S. Kovasznay and H. M. Joseph, Processing of two-dimensional patterns by scanning techniques, *Science,* vol. 118, no. 3069, pp. 475–477, 25 October 1953.

59. L. G. Roberts, Machine perception of three-dimensional solids, in *Optical and Electro-optical Information Processing*, J. T. Tippet, ed., Cambridge, MA: MIT Press, 1965.

60. A. Rosenfeld and M. Thurston, Edge and curve detection for visual scene analysis, *IEEE Transactions on Computers,* vol. 20, no. 5, pp. 562–569, May 1971.

61. J. M. S. Prewitt, Object enhancement and extraction, in *Picture Processing and Psychopictorics*, B. S. Lipkin and A. Rosenfeld, eds., New York: Academic Press, 1970.

62. D. Marr and E. Hildreth, Theory of Edge Detection, *Proceedings of the Royal Society of London B.* vol. 207, pp. 187–217, 1979.

63. M. F. Heukel, An operator which locates edges in digitized pictures, *Journal of the Association for Computing Machinery*, vol. 18, pp. 113–125, 1971.

64. F. O'Gorman, Edge detection using Walsh functions, *Artificial Intelligence*, vol. 10, pp. 215–223, 1978.

65. R. A. Hummel, Feature detection using basis functions, *Computer Graphics and Image Processing*, vol. 9, pp. 40–55, 1979.

66. H. D. Tagare and R. J. P. deFigueiredo, On the localization performance measure and optimal edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 12, no. 12, pp. 1186–1190, 1990.

67. K. L. Boyer and S. Sarkar, Comments on "On the localization performance measure and optimal edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 16, no. 1, pp. 106–108, January 1994.

68. R. J. Qian and T. S. Huang, Optimal edge detection in two-dimensional images, *IEEE Transactions on Image Processing,* vol. 5, no. 7, pp. 1215–1220, 1996.

69. M. Gökmen and A. K. Jain, $\lambda\tau$-space representation of images and generalized edge detector, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 19, no. 6, pp. 545–563, June 1997.

70. D. M. Manos and D. L. Flamm, *Plasma Etching: An Introduction*, Boston: Academic Press, 1989.

71. R. L. Allen, R. Moore, and M. Whelan, Multiresolution pattern detector networks for controlling plasma etch reactors, in *Process, Equipment, and Materials Control in Integrated Circuit Manufacturing*, Proceedings SPIE 2637, pp. 19–30, 1995.

72. C. K. Hanish, J. W. Grizzle, H.-H. Chen, L. I. Kamlet, S. Thomas III, F. L. Terry, and S. W. Pang, Modeling and algorithm development for automated optical endpointing of an HBT emitter etch, *Journal of Electronic Materials*, vol. 26, no. 12, pp. 1401–1408, 1997.

73. V. Torre and T. A. Poggio, On edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-8, no. 2, pp. 147–163, March 1986.

74. C. F. Gerald and P. O. Wheatley, *Applied Numerical Analysis*, 4th ed., Reading, MA: Addison-Wesley, 1990.

75. J. Canny, A computational approach to edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-8, no. 6, pp. 679–698, November 1986.

76. A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, New York: McGraw-Hill, 1965.

77. D. Marr, *Vision*, New York: W. H. Freeman, 1982.

78. J. F. Canny, *Finding Edges and Lines in Images*, Technical Report No. 720, MIT Artificial Intelligence Laboratory, June 1983.

79. H. D. Tagare and R. J. P. deFigueiredo, Reply to “On the localization performance measure and optimal edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 16, no. 1, pp. 108–110, January 1994.

80. J. Koplowitz and V. Greco, On the edge location error for local maximum and zero-crossing edge detectors, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 16, no. 12, pp. 1207–1212, December 1994.

81. L. W. Couch II, *Digital and Analog Communication Systems*, 4th ed., Upper Saddle River, NJ: Prentice-Hall, 1993.

82. S. Haykin, *Communication Systems,* 3rd ed., New York: Wiley, 1994.

83. G. L. Turin, An introduction to digital matched filters, *Proceedings of the IEEE*, vol. 64, pp. 1092–1112, 1976.

84. A. Rosenfeld and A. C. Kak, *Digital Picture Processing,* vol. 1, 2nd ed., New York: Academic, 1982.

85. J. R. Tou and R. C. Gonzalez, *Pattern Recognition Principles,* Reading, MA: Addison-Wesley, 1974.

86. F. Molinaro and F. Castanié, Signal processing pattern classification techniques to improve knock detection in spark ignition engines, *Mechanical Systems and Signal Processing*, vol. 9, no. 1, pp. 51–62, January 1995.

87. J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications,* 2nd ed., New York: Macmillan, 1992.

88. A. V. Oppenheim and R. W. Schafer, *Discrete-Timel Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall, 1989.

89. S. S. Yau and J. M. Garnett, least-mean-square approach to pattern classification, in *Frontiers of Pattern Recognition,* S. Wantanabe, ed., New York: Academic Press, pp. 575–588, 1972.

90. N. J. Nilsson, *Learning Machines*, New York: McGraw-Hill, 1965.

91. K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed., Boston: Academic Press, 1990.
92. P. R. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, Englewood Cliffs, NJ: Prentice-Hall, 1982.
93. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.
94. C. K. Chow, An optimum character recognition system using decision functions, *Transactions of the IRE on Electronic Computers,* vol. EC-6, pp. 247–254, 1957.
95. J. Schürmann, Multifont word recognition system with application to postal address reading, *Proceedings of the Third International Joint Conference on Pattern Recognition*, pp. 658–662, 1976.
96. W. Doster, Contextual postprocessing system for cooperation with a multiple-choice character-recognition system, *IEEE Transactions on Computers*, vol. C-26, no. 11, pp. 1090–1101, November 1977.
97. R. Ott, Construction of quadratic polynomial classifiers, *Proceedings of the Third International Joint Conference on Pattern Recognition*, 1976.
98. Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Reading, MA: Addison-Wesley, 1989.
99. F. Attneave, Some informational aspects of visual perception, *Psychological Review*, vol. 61, no. 3, pp. 183–193, 1954.
100. S. Tanimoto and T. Pavlidis, A hierarchical data structure for picture processing, *Computer Graphics and Image Processing,* vol. 4, No. 2, pp. 104–119, June 1975.
101. .R. Y. Wong and E. L. Hall, Sequential hierarchical scene matching, *IEEE Transactions on Computers,* vol. C-27, No. 4, pp. 359–366, April 1978.
102. D. Marr, T. Poggio, and S. Ullman, Bandpass channels, zero-crossings, and early visual information processing, *Journal of the Optical Society of America,* vol. 69, pp. 914–916, 1979.
103. D. H. Hubel and T. N. Wiesel, Receptive fields, binocular interaction and functional architecture in the cat's visual cortex, *Journal of Physiology,* vol. 160, pp. 106–154, 1962.
104. F. W. C. Campbell and J. Robson, Application of Fourier analysis to the visibility of gratings, *Journal of Physiology*, vol. 197, pp. 551–566, 1968.
105. D. H. Hubel, *Eye, Brain, and Vision*, New York: W. H. Freeman, 1988.
106. A. P. Pentland, Models of image formation, in *From Pixels to Predicates*, A. P. Pentland, ed. Norwood, NJ: Ablex, 1986.
107. A. P. Witkin, Scale-space filtering, *Proceedings of the 8th International Joint Conference on Artificial Intelligence,* Karlsruhe, West Germany, 1983. See also A. P. Witkin, Scale-space filtering, in *From Pixels to Predicates*, A. P. Pentland, ed., Norwood, NJ: Ablex, 1986.
108. J. Koenderink, The structure of images, *Biological Cybernetics,* vol. 50, pp. 363–370, 1984.
109. M. P. do Carmo, *Differential Geometry of Curves and Surfaces*, Englewood Cliffs, NJ: Prentice-Hall, 1976.
110. A. L. Yuille and T. A. Poggio, Scaling theorems for zero crossings, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-8, no. 1, pp. 15–25, January 1986.

111. Babaud, A. P. Witkin, M. Baudin, and R. O. Duda, Uniqueness of the Gaussian kernel for scale-space filtering, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-8, no. 1, pp. 26–33, January 1986.

112. F. Mokhtarian and A. Mackworth, Scale-based description and recognition of planar curves and two-dimensional shapes, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-8, no. 1, pp. 34-43, January 1986.

PROBLEMS

1. Consider the analog Gaussian kernel $h(t) = \exp(-t^2)$.

(a) Show that $h(t)$ contains a concave up portion for $t < 0$, a concave-down section symmetric about $t = 0$, a concave up portion for $t > 0$, and exactly two points of inflection where the second derivative of $h(t)$ is zero.

(b) Find a logical predicate, applied to subsets of Dom(h), that segments $h(t)$ into the regions described in (a).

(c) Suppose that a system designer elects to prefilter discrete signals by an approximate, discrete version of $h(t)$ and wishes to maintain the aspects of concavity of $h(t)$ in the filter $h(n)$. What is the minimal support of the filter $h(n)$?

(d) Show that if the analog signal $f(t)$ is segmented according to whether d^2f/dt^2 is negative, zero, or positive and d^3f/dt^3 exists, then segments with $d^2f/dt^2 < 0$ cannot be adjacent to segments with $d^2f/dt^2 > 0$.

2. Consider a system $y = Hx$ that should remove impulse noise from $x(n)$ to produce $y(n)$.

(a) If H is LTI, show that the moving average, or box filter $h(n) = [1/3, \underline{1/3}, 1/3]$ can fail to do an adequate job as the impulse response of H.

(b) Show that any LTI filter can fail to do an adequate job as $h = H\delta$.

3. Suppose we filter out impulse noise by a median filter, $y = Hx$.

(a) Show that H is translation invariant.

(b) Show that H is nonlinear.

(c) Show that the median filter $(Hx)(n) = \text{median}\{x(n-1), x(n), x(n+1)\}$ can remove impulse noise.

(d) Show that when performing preliminary smoothing of a signal before thresholding, a median filter $y(n) = \text{median}\{x(n-1), x(n), x(n+1)\}$ preserves signal edges better than box filters.

4. Consider a morphological filter, $y = Hx$.

(a) Show that if H is a dilation, then H is translation invariant but not linear.

(b) Show that if H is an erosion, then H is translation invariant but not linear.

5. Suppose f(n) is a discrete signal and $T > 0$. Let $M = \{n: |f(n)| > T\}$, $N = \{n: |f(n)| \le T\}$, $\Pi = \{M, N\}$, and L be the logical predicate "(1) For all n, $|f(n)| > T$;

or (2) for all n, $|f(n)| < T$; or for all n, $|f(n)| = T$." Prove that $\Sigma = (\Pi, L)$ is a segmentation of $f(n)$.

6. Let $f(t)$ be an analog signal which is twice differentiable. Show that the sign of the second derivative is the basis for a segmentation of $f(t)$ into concave-down regions, concave-up regions, and zeros of the second derivative.

7. Consider the discrete signal $x(n) = [..., 0, 2, -1, 3, 1, 8, \underline{10}, 9, 11, 5, 2, -1, 1, 0, . . .]$.

(a) Suppose values $|x(n)| \geq T$ are labeled Object and values $|x(n)| < T$ are labeled Background. Consider the effect of thresholding this signal for $T = 5$ with and without preliminary filtering by the moving average filter of width $N = 3$:

$$y(n) = \frac{1}{2N+1} \sum_{k=-N}^{N} x(k).$$

(b) What happens to the segmentation as the filter width increases? Does changing the threshold help?

(c) What happens to the segmentation if a causal box filter performs the preliminary filtering operation? Does this affect the registration of the blob?

(d) Examine the effect of a high-magnitude noise impulse several time samples apart from the blob; suppose, therefore, that $x(-10) = 12$. Consider the effect of preliminary box filtering and threshold changes on segmentation and blob registration.

(e) Can a top-down rule be applied to correct the labeling of impulse noise? Does the width of the smoothing filter affect the success of the rule?

8. Prove that a discrete signal edge detector which locates edges from the maximal responses of a convolutional operator, H, satisfies the following:

(i) The impulse response of H, $h(n)$, must be odd.

(ii) $h(n)$ can only have one zero, at $n = 0$.

9. Consider the estimator $\hat{\mu}$ found in the Unbiased Minimal Variance Estimator Theorem (Section 4.3.2). Suppose that $\boldsymbol{x}$ is a random vector, representing a window $[m - p, m + p]$ of the noisy source signal $x(n)$, and that $\boldsymbol{b}$ is the unit vector of all ones of length $2p + 1$. If the estimator $\hat{\mu}$ is unbiased and has minimal variance, show that

$$\mathrm{Var}[\hat{\mu}] = \left(\frac{1}{\boldsymbol{b} \Sigma^{-1} \boldsymbol{b}^T} \right),$$

where $\Sigma = E[(\boldsymbol{x} - \mu 1), (\boldsymbol{x} - \mu 1)]$ is the covariance matrix of the random vector $\boldsymbol{x}$.

10. Develop the analog version of normalized cross-correlation. Consider an analog signal $x(t)$ input in which a pattern $p(t)$ must be found at an unknown offset s. Suppose that $p(t)$ has finite support contained in the real interval I. Show the following:

(a) The normalized cross-correlation,

$$C_{p(t-s),x(t)} = \frac{y(s)}{\left(\int_I x^2(t)\,dt\right)^{\frac{1}{2}}} = \frac{\left(\int_I p(t-s)x(t)\,dt\right)^{\frac{1}{2}}}{\left(\int_I x^2(t)\,dt\right)^{\frac{1}{2}}},$$

where s is the offset of prototype signal $p(t)$ into input signal $x(t)$ and where I is the interval that contains the support of $p(t)$, is a measure of match.

(b) The cross-correlation, $C_{p(t-s),x(t)}$, assumes a unity maximum when $x(t) = cp(t-s)$ on the interval I for some constant c.

11. Check that all signals in the family of exponentials $B = \{(2\pi)^{-1}\exp(2\pi jnt/T)$: $n \in \mathbb{Z}\}$ have $\|(2\pi)^{-1}\exp(2\pi jnt/T)\|_1 = 1$ in the Hilbert space $L^1[0, T]$.

12. Let $g(t, \sigma, \mu) = \sigma^{-1}(2\pi)^{-1/2}\exp(-(x-\mu)^2/(2\sigma^2))$ be the Gaussian with standard deviation σ and mean μ. Show that $g(t, \sigma, \mu)$ is symmetric and strictly decreasing about its mean, μ.

13. Let $F(t, \sigma)$ be defined as in Equation (4–5c) for analog signal $f(t)$. Show that for small scales σ, the smoothed signal $F(t, \sigma)$ resembles the original as $\sigma \to 0$; that is, we have $F(t, \sigma) \to f(t)$.

14. Show that as $\sigma \to \infty$, $F(t, s) \to E(f(t))$; that is, for large scales σ, the smoothed signal approaches the mean of $f(t)$.

15. Show that the Gaussian is an $L^1(\mathbb{R})$ signal (absolutely integrable), and it is C^∞ (infinitely differentiable).

16. Show that if $f(t)$ is an analog signal and $g(t, \sigma)$ is Gaussian with zero mean and variance σ^2, then

$$\frac{\partial^n}{\partial t^n}F(t,\sigma) = f * \frac{\partial^n}{\partial t^n}g(t,\sigma).$$

17. Consider the following region merging approach to real-time signal segmentation. The application receives a digitized signal, $f(n)$, and segments it into Noise and Signal regions, with labels Λ_N and Λ_S, respectively. Three thresholds are provided: T_N, T_S, and ε. If $x(n) < T_N$, then $x(n)$ is marked as a Λ_N value. If $x(n) > T_S$, then $x(n)$ is labeled Λ_S. If $T_N \le x(n) \le T_S$, then $x(n)$ is labeled the same as $x(n-1)$ if $|x(n) - x(n-1)| < \varepsilon$, and $x(n)$ is deemed the opposite of $x(n-1)$ otherwise.

(a) Show how this real-time segmentation works for $f(n) = [..., \underline{0}, 2, 4, 3, 1, 8, 6, 9, 11, 7, 3, 5, 1, 0, ...]$, with $T_N = 4$, $T_S = 7$, and $\varepsilon = 3$.

(b) What constraints should exist on T_N, T_S, and ε for a useful algorithm?

(c) What is the effect of impulse noise on the algorithm?

(d) Let N_n and S_n be the average of the Λ_N and the Λ_S values, respectively, prior to time instant n. Suppose that the algorithm is changed so that when $T_N \le x(n) \le T_S$, then $x(n)$ is labeled Λ_N when $|x(n) - N_n| < \varepsilon$, and then $x(n)$ is labeled Λ_S when $|x(n) - S_n| < \varepsilon$. How does this twist affect the operation of the algorithm?

(e) Critique this algorithm for real-time speech segmentation. Consider the presence of velar fricatives, noise, and voiced and unvoiced consonants in the digitized speech signal.

(f) Explain how a delay in making the labeling decision for a new signal value $x(n)$ might help improve the segmentation for the front-end of a speech recognition application.

(g) What characteristics of digitized speech signals should be taken into account in order to size the labeling decision delay in *(f)*? Explain.

18. Critique the following region splitting algorithm. We begin with a region of interest for a discrete signal $f(n)$, $S = [a, b]$, and a partition of S into (as closely as possible) equal-length subintervals, $S = S_1 \cup S_2 \cup ... \cup S_N$. We first compute the mean over the entire region S, μ. Then we begin with the region whose mean is closest to μ; by renumbering the regions, if necessary, we may suppose that it is S_1. Then we compute all of the differences, $d(1, i) = |\mu_1 - \mu_i|$, for $i > 1$, and put $R_1 = S_1 \cup \{S_i \mid i > 1 \text{ and } d(1, i) < \varepsilon\}$. We then perform this same operation with the remaining S_i, which are disjoint from R_1, to form R_2, and so on. The process eventually halts after M iterations, and S is split into M regions: $S = R_1 \cup R_2 \cup \cdots \cup R_M$.

19. Show that if $x(n)$ is a digital signal, x: $\mathbb{Z} \to [0, N]$, for some natural number $N \ge 0$, then $x = T[\text{Umbra}(x)] = T[\text{Graph}(x)]$.

20. Consider Chow and Kaneko's optimal threshold finding method, where the quadratic equation below must be solved for T:

$$(\sigma_L^2 - \sigma_H^2)T^2 + 2(\sigma_H^2\mu_L - \sigma_L^2\mu_H)T + (\sigma_L^2\mu_H^2 - \sigma_H^2\mu_L^2) - 2\ln\left(\frac{P_H\sigma_L}{P_L\sigma_H}\right) = 0.$$

(a) Suppose that the variances of the meaningful signal and the background noise are the same. Show that there can be at most one threshold, T.

(b) Show that if the variances of the meaningful signal and the background noise are the same and the *a priori* probabilities of signal and noise are the same, then $T = (mL + mH)/2$.

(c) Can there be two solutions to the quadratic equation? Explain.

(d) If there is only one solution to the quadratic equation, is it necessarily a valid threshold? Explain.

(e) Can there be no solution to the equation? Explain.

21. Suppose a signal $f(n)$ has regions of interest with high values above a relatively low background noise level. Suppose also that a threshold T on the signal histogram produces errors with probability $E(T) = P_H E_L(T) + P_L E_H(T)$, where $E_L(T)$ and $E_H(T)$ are the likelihoods of incorrectly labeling f(n) as noise and as meaningful, respectively.

(a) Assume that the histogram modes obey a log-normal distribution, and, following the Chow–Kaneko method of the text in Section 4.2, use

$$\frac{dE}{dT} = P_H \frac{dE_L}{dT} + P_L \frac{dE_H}{dT} = 0$$

to find T for a minimal labeling error.

(b) As in (a), find a T that minimizes labeling error if the histogram modes obey a Rayleigh distribution.

22. Consider the valley-finding algorithm for finding a threshold using signal histograms.

(a) Show that the search stops when the current threshold T is a local minimum.

(b) Show by example that the algorithm has a direction preference; that is, it tends to select a threshold toward the noise mode or toward the meaningful signal mode.

(c) Find and explicate a technique for resolving the direction preference of the algorithm.

(d) Show by example that the algorithm may halt its search well within the mode of the histogram's noise mode or the true signal mode.

(e) Show that a gradient calculation based on a wider interval may alleviate somewhat the problem exposed in part (d).

(f) For what types of signals does the algorithm find a threshold that is at the limit of the domain of the histogram? Explain.

23. Let f be a discrete signal with a finite domain with Ran(f) $\subseteq [0, N-1]$. For $0 \le k < N$, define

$$p_k = \frac{\#(f^{-1}(\{k\}))}{\#\mathrm{Dom}(f)}.$$

Show that p_k is a discrete probability density function for f.

24. Let f and p_k be as in the previous problem. Let μ_L, μ_H, and μ be the low-level, high-level, and total mean of $h(k)$, the histogram for f. Let σ_L, σ_H, and σ be the low-level, high-level, and total standard deviations of $h(k)$.

(a) Show

$$\sum_{k=0}^{t-1} p_k(k-\mu_L(t))(\mu_L(t)-\mu) + \sum_{k=t}^{N-1} p_k(k-\mu_H(t))(\mu_H(t)-\mu) = 0.$$

(b) Following Otsu [18] define the between-group variance to be

$$\sigma_b^2(t) = P_L(t)(\mu_L(t) - \mu)^2 + P_H(t)(\mu_H(t) - \mu)^2 mp$$

and prove that

$$\sigma^2(t) = \sigma_w^2(t) + \sigma_b^2(t).$$

(Thus, minimizing within-group variance is equivalent to maximizing between-group variance.)

(c) Show the following for $0 \le t < N-1$:

$$p_k = \frac{\#(f^{-1}(\{k\}))}{\#\mathrm{Dom}(f)},$$

(d) Explain how the relationships in (c) reduce the computational burden of finding an optimal threshold (R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. 1, New York: Addison-Wesley, 1992).

25. Consider the likelihood ratio of the distribution q_k with respect to p_k,

$$L(p,q) = \sum_{k=0}^{N-1} p_k \log_2 \frac{p_k}{q_k} = \sum_{k=0}^{N-1} p_k \log_2 p_k - \sum_{k=0}^{N-1} p_k \log_2 q_k .$$

(a) Show that $L(p, q) \ge 0$ for all discrete probability distributions p and q. (*Hint*: Note that $\ln(t) < t - 1$ for all $t > 0$, where $\ln(t)$ is the natural logarithm; $\ln(2)\log_2(t) = \ln(t)$; and $p_0 + p_1 + \cdots + p_{N-1} = 1$.)

(b) Show that $L(p, q) = 0$ if and only if $p = q$.

(c) Show that L is not symmetric; that is, find examples for p_k and q_k such that p_k and p_k are discrete probability density functions, but $L(p, q) \ne L(q, p)$.

(d) Does the triangle inequality hold for the likelihood ratio?

26. Prove the following properties of dilation. Let $A, B, C \subseteq \mathbb{Z}^2$ and $k \in \mathbb{Z}$.

(a) $A \oplus B = \{a + b \mid a \in A \text{ and } b \in B\}$.

(b) $A \oplus B = B \oplus A$.

(c) $(A \oplus B) \oplus C = A \oplus (B \oplus C)$.

(d) If $A \subseteq B$, then $A \oplus C \subseteq B \oplus C$.

(e) $A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$.

(f) $A \oplus (B + k) = (A \oplus B) + k$.

27. Prove the following properties of erosion. Suppose $A, B, C \subseteq \mathbb{Z}^2$ and $k \in \mathbb{Z}$.

(a) $A \ominus B = \{d \in \mathbb{Z}^2 \mid d + b \in A \text{ for all } b \in B\}$.

(b) $A \ominus B = \{d \in \mathbb{Z}^2 \mid B_d \subseteq A\}$.

(c) $(A + k) \ominus B = (A \ominus B) + k$.

(d) $A \ominus (B + k) = (A \ominus B) - k$.

(e) If $A \subseteq B$, then $A \ominus C \subseteq B \ominus C$.

(f) $(A \cap B) \ominus C = (A \ominus B) \cap (B \ominus C)$.

(g) If we define $-B$ to be the reflection of the set B, $-B = \{-k \mid k \in B\}$, then we have $(A \ominus B)^c = A^c \oplus (-B)$.

28. Consider the Bayes classifier discriminant that we developed in (4.104):

$$D_k(\boldsymbol{v}) = -\frac{1}{2}(\boldsymbol{v} - \mu_k)^T \Sigma_k^{-1}(\boldsymbol{v} - \mu_k) - \frac{\log(\det(\Sigma_k))}{2} + \log(P(C_k)).$$

The text considered the simplifications brought from assuming that the feature vectors $\boldsymbol{v}$ are statistically independent and have equal variances. What simplification, if any, results from assuming simple statistical independence? Explain.

29. Let $x(n) = [..., \underline{0}, 1, 1, 2, 1, 2, 1, 0, 1, 2, 2, 1, 0, 1, 1, 0, ...]$ be a digital signal.

(a) Compute the co-occurrence matrices, P_1, P_2, and P_3, within the interval $0 \le n \le 15$.

(b) Compute the energy of values on the main diagonal of P_1, P_2, and P_3, and compare it to the energy of off-diagonal values. How do the two compare? Explain.

(c) What should one expect from co-occurrence matrices P_4, P_5, and P_6?

(d) What differences exist between the co-occurrence matrices computed for a signal with sawtooth features versus a signal with square pulse features?

30. Suppose it is required that an analysis application detect signals containing large-scale, high-magnitude regions of width N separated by low-magnitude regions of width M.

(a) What co-occurrence matrices should be computed? Explain.

(b) Describe an algorithm using these co-occurrence matrices that meets the application's needs.

(c) How do different values of M and N affect the algorithm?

31. Suppose that $f(n) = A\cos(2\pi\omega n) + B\cos(2\pi\Omega n) + Cr(n)$ is a discrete signal with $A > B > C > 0$, $\Omega > \omega$, and $r(n)$ is uniformly distributed noise. Thus, $f(n)$ consists of a sinusoidal roughness component, $A\cos(2\pi\omega n)$, and a waviness component, $B\cos(2\pi\Omega n)$.

(a) Explore the feasibility of using a co-occurrence matrix method to detect waviness versus roughness in $f(n)$. Propose and explain such an algorithm. For what time intervals, δ, does the algorithm calculate P_δ? How are the entries of the P_δ used to discover the waviness and roughness within $f(n)$?

(b) How can the size of the P_δ matrices be kept small?

(c) Explain whatever assumptions you must make on ω, Ω, A, B, and C so that the proposed algorithm works.

32. Consider a discrete signal $x(n) = x_a(nT)$, where $T > 0$ is the sampling interval.

(a) Expand $x(1)$, $x(2)$, $x(-1)$, and $x(-2)$ in terms of the Taylor series of $x_a(t)$, and show that the system with impulse response $h(n) = [-1/12, 2/3, \underline{0}, -2/3, 1/12]$ is a first-derivative operation on signals.

(b) Similarly, show that $g(n) = [-1/12, 4/3, \underline{-5/2}, 4/3, -1/12]$ is a second-derivative operator on signals.

33. Consider an application that extracts and labels feature vectors of input signals, then compares them to labeled feature vectors derived from a library of prototype signals.

(a) Show that the Hamming distance, $H(u, v)$ between two vectors of labels $\boldsymbol{u} = (\alpha_1, \alpha_2, ..., \alpha_N\}$ and $\boldsymbol{w} = (\beta_1, \beta_2, ..., \beta_N\}$ is a metric.

(b) Consider the Levenshtein distance, $L(\boldsymbol{u}, \boldsymbol{v})$, defined as the total number of substitutions into $\boldsymbol{u}$ and transpositions of components of $\boldsymbol{u}$ necessary to convert $\boldsymbol{u}$ into $\boldsymbol{v}$. Prove or disprove: $L(\boldsymbol{u}, \boldsymbol{v})$ is a metric.