Chapter 4

SEGMENTATION

Image segmentation is the division of an image into **regions** or **categories**, which correspond to different objects or parts of objects. *Every* pixel in an image is allocated to one of a number of these categories. A good segmentation is typically one in which:

- pixels in the same category have similar greyscale of multivariate values and form a connected region,
- neighbouring pixels which are in different categories have dissimilar values.

For example, in the muscle fibres image (Fig 1.6(b)), each cross-sectional fibre could be viewed as a distinct object, and a successful segmentation would form a separate group of pixels corresponding to each fibre. Similarly in the SAR image (Fig 1.8(g)), each field could be regarded as a separate category.

Segmentation is often the critical step in image analysis: the point at which we move from considering each pixel as a unit of observation to working with *objects* (or parts of objects) in the image, composed of many pixels. If segmentation is done well then all other stages in image analysis are made simpler. But, as we shall see, success is often only partial when automatic segmentation algorithms are used. However, manual intervention can usually overcome these problems, and by this stage the computer should already have done most of the work.

This chapter fits into the structure of the book as follows. Segmentation algorithms may either be applied to the images as originally recorded (chapter 1), or after the application of transformations and filters considered in chapters 2 and 3. After segmentation, methods of mathematical morphology can be used to improve the results. These will be considered in chapter 5. Finally, in chapter 6, the segmentation results will be used to extract quantitative information from the images.

There are three general approaches to segmentation, termed thresholding, edge-based methods and region-based methods.

- In **thresholding**, pixels are allocated to categories according to the range of values in which a pixel lies. Fig 4.1(a) shows boundaries which were obtained by thresholding the muscle fibres image. Pixels with values less than 128 have been placed in one category, and the rest have been placed in the other category. The boundaries between adjacent pixels in different categories has been superimposed in white on the original image. It can be seen that the threshold has successfully segmented the image into the two predominant fibre types.

- In **edge-based** segmentation, an edge filter is applied to the image, pixels are classified as *edge* or *non-edge* depending on the filter output, and pixels which are not separated by an edge are allocated to the same category. Fig 4.1(b) shows the boundaries of connected regions after applying Prewitt's filter (§3.4.2) and eliminating all non-border segments containing fewer than 500 pixels. (More details will be given in §4.2.)

- Finally, **region-based** segmentation algorithms operate iteratively by grouping together pixels which are neighbours and have similar values and splitting groups of pixels which are dissimilar in value. Fig 4.1(c) shows the boundaries produced by one such algorithm, based on the concept of *watersheds*, about which we will give more details in §4.3.

Note that none of the three methods illustrated in Fig 4.1 has been completely successful in segmenting the muscle fibres image by placing a boundary between every adjacent pair of fibres. Each method has distinctive faults. For example, in Fig 4.1(a) boundaries are well placed, but others are missing. In Fig 4.1(c), however, more boundaries are present, and they are smooth, but they are not always in exactly the right positions.

The following three sections will consider these three approaches in more detail. Algorithms will be considered which can either be *fully automatic* or require some *manual intervention*. The key points of the chapter will be summarized in §4.4.

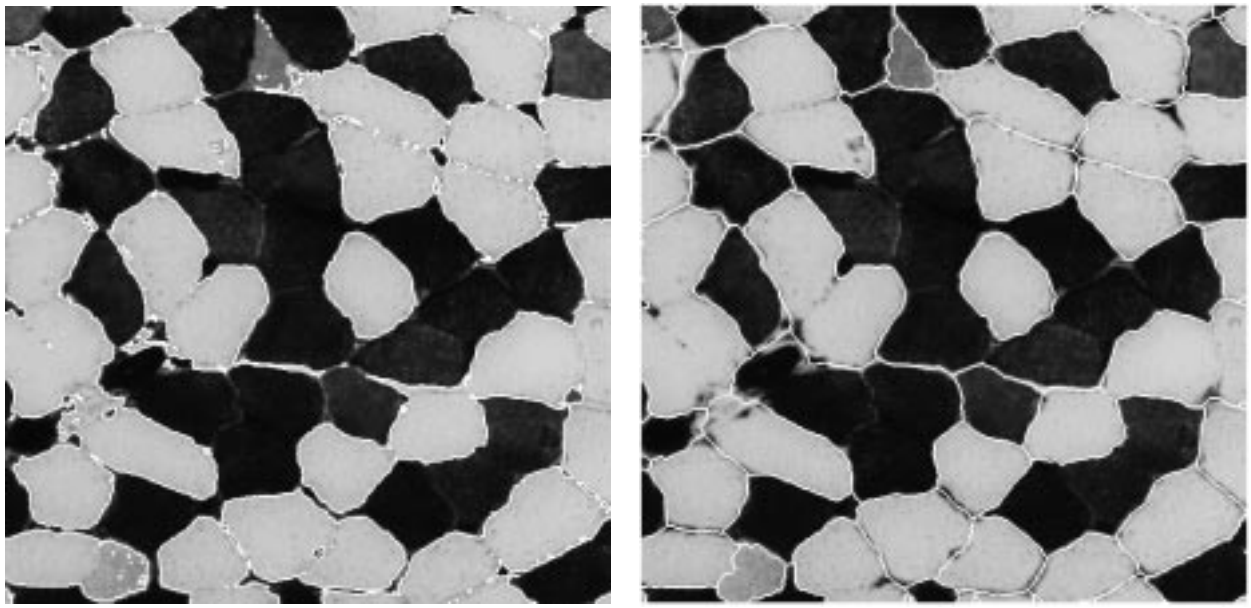
4.1 Thresholding

Thresholding is the simplest and most commonly used method of segmentation. Given a single **threshold**, t, the pixel located at lattice position (i, j), with greyscale value f_{ij}, is allocated to category 1 if

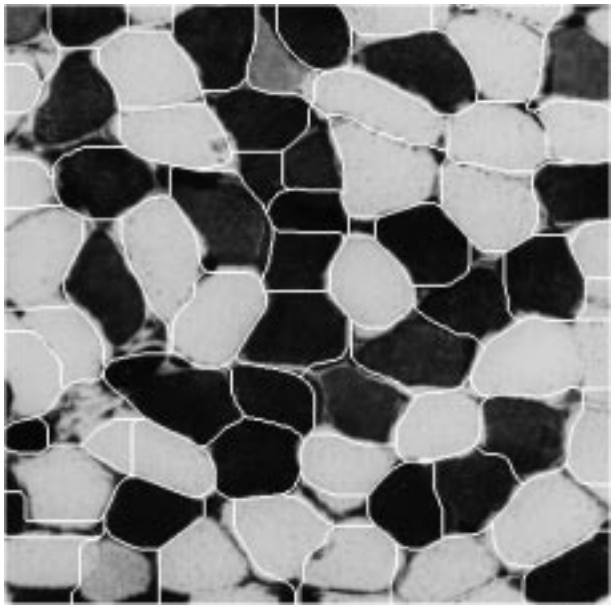
$$f_{ij} \leq t.$$

Otherwise, the pixel is allocated to category 2.

In many cases t is chosen *manually* by the scientist, by trying a range of values of t and seeing which one works best at identifying the objects of interest. Fig 4.2 shows some segmentations of the soil image. In this application, the aim was to isolate soil material from the air-filled pores which appear as the darker pixels in Fig 1.6(c). Thresholds of 7, 10, 13, 20, 29 and 38 were chosen in Figs 4.2(a) to (f) respectively, to identify approximately 10, 20, 30, 40, 50 and 60% of the pixels as being pores. Fig 4.2(d), with a threshold of 20, looks best because most

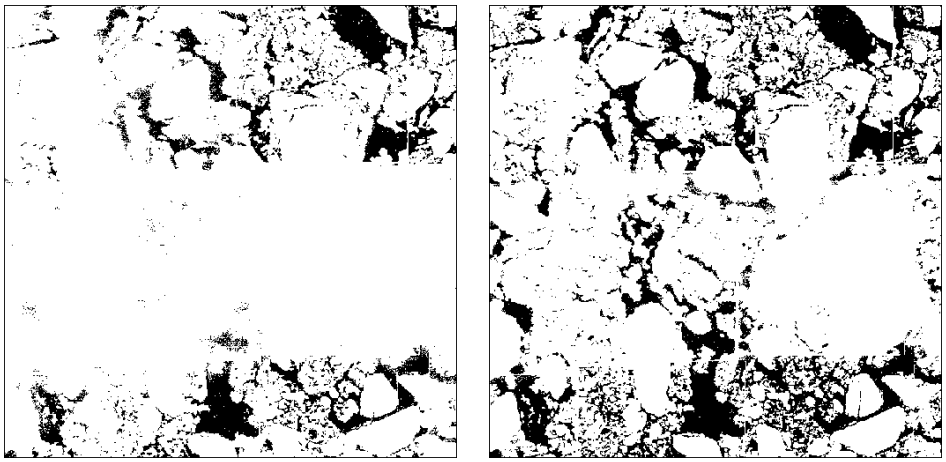


(a) (b)



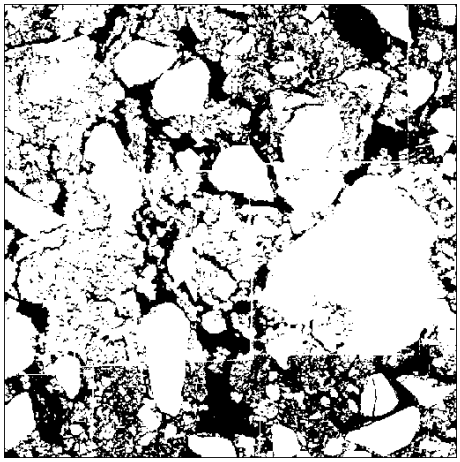
(c)

Figure 4.1: Boundaries produced by three segmentations of the muscle fibres image: **(a)** by thresholding, **(b)** connected regions after thresholding the output of Prewitt's edge filter and removing small regions, **(c)** result produced by watershed algorithm on output from a variance filter with Gaussian weights ($\sigma^2 = 96$).

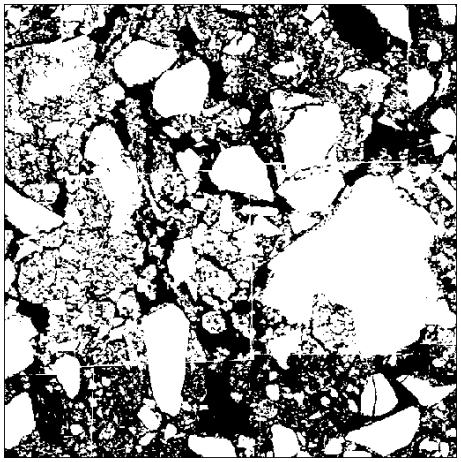


(a)

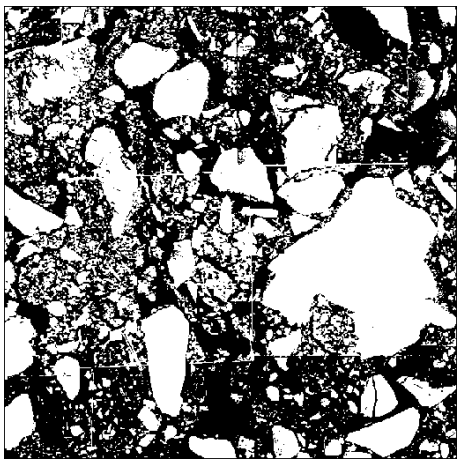
(b)



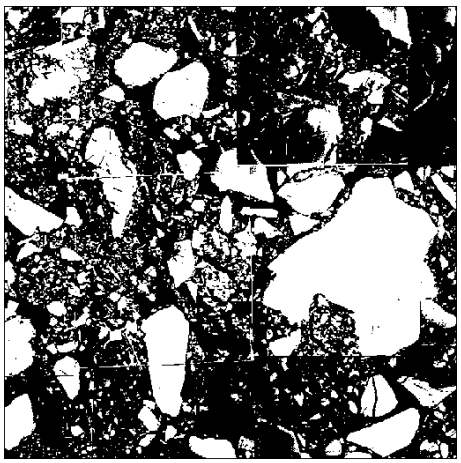
(c)



(d)



(e)



(f)

Figure 4.2: Six segmentations of the soil image, obtained using manually-selected thresholds of **(a)** 7, **(b)** 10, **(c)** 13, **(d)** 20, **(e)** 29 and **(f)** 38. These correspond to approximately $10\%, 20\%, \ldots, 60\%$, respectively, of the image being displayed as black.

of the connected pore network evident in Fig 1.6(c) has been correctly identified, as has most of the soil matrix.

Note that:

- Although pixels in a single thresholded category will have similar values (either in the range 0 to t, or in the range $(t+1)$ to 255), they will not usually constitute a single connected component. This is not a problem in the soil image because the object (air) is not necessarily connected, either in the imaging plane or in three-dimensions. In other cases, thresholding would be followed by dividing the initial categories into sub-categories of connected regions.
- More than one threshold can be used, in which case more than two categories are produced.
- Thresholds can be chosen *automatically*.

In §4.1.1 we will consider algorithms for choosing the threshold on the basis of the histogram of greyscale pixel values. In §4.1.2, manually- and automatically-selected classifiers for multivariate images will be considered. Finally, in §4.1.3, thresholding algorithms which make use of context (that is, values of neighbouring pixels as well as the histogram of pixel values) will be presented.

4.1.1 Histogram-based thresholding

We will denote the **histogram** of pixel values by $h_0, h_1, \ldots, h_N$, where h_k specifies the number of pixels in an image with greyscale value k and N is the maximum pixel value (typically 255). Ridler and Calvard (1978) and Trussell (1979) proposed a simple algorithm for choosing a single threshold. We shall refer to it as the **intermeans algorithm**. First we will describe the algorithm in words, and then mathematically.

Initially, a guess has to be made at a possible value for the threshold. From this, the mean values of pixels in the two categories produced using this threshold are calculated. The threshold is repositioned to lie exactly half way between the two means. Mean values are calculated again and a new threshold is obtained, and so on until the threshold stops changing value. Mathematically, the algorithm can be specified as follows.

1. Make an initial guess at t: for example, set it equal to the median pixel value, that is, the value for which
$$\sum_{k=0}^{t} h_k \geq \frac{n^2}{2} > \sum_{k=0}^{t-1} h_k,$$
where n^2 is the number of pixels in the $n \times n$ image.

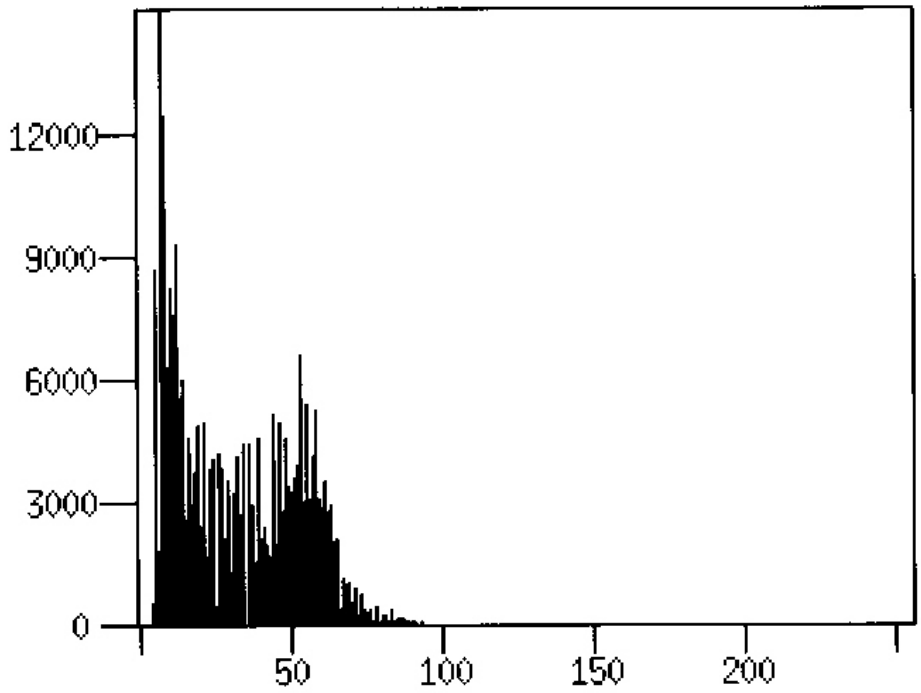


Figure 4.3: Histogram of soil image.

2. Calculate the mean pixel value in each category. For values less than or equal to t, this is given by:

$$\mu_1 = \sum_{k=0}^{t} kh_k \Big/ \sum_{k=0}^{t} h_k \ .$$

Whereas, for values greater than t, it is given by:

$$\mu_2 = \sum_{k=t+1}^{N} kh_k \Big/ \sum_{k=t+1}^{N} h_k \ .$$

3. Re-estimate t as half-way between the two means, i.e.

$$t = \left[\frac{\mu_1 + \mu_2}{2} \right],$$

where [ ] denotes 'the integer part of' the expression between the brackets.

4. Repeat steps (2) and (3) until t stops changing value between consecutive evaluations.

Fig 4.3 shows the histogram of the soil image. From an initial value of $t = 28$ (the median pixel value), the algorithm changed t to 31, 32, and 33 on the first three iterations, and then t remained unchanged. The pixel means in the two categories are 15.4 and 52.3. Fig 4.4(a) shows the result of using this threshold. Note that this value of t is considerably higher than the threshold value of 20 which we favoured in the manual approach.

The intermeans algorithm has a tendency to find a threshold which divides the histogram in two, so that there are approximately equal numbers of pixels in the two categories. In many applications, such as the soil image, this is not appropriate. One way to overcome this drawback is to modify the algorithm as follows.

Consider a distribution which is a mixture of two Gaussian distributions. Therefore, in the absence of sampling variability, the histogram is given by:

$$h_k = n^2\{p_1\phi_1(k) + p_2\phi_2(k)\}, \qquad \text{for } k = 0, 1, \ldots, N.$$

Here, p_1 and p_2 are proportions (such that $p_1 + p_2 = 1$) and $\phi_l(k)$ denotes the probability density of a Gaussian distribution, that is

$$\phi_l(k) = \frac{1}{\sqrt{2\pi\sigma_l^2}} \exp\left\{\frac{-(k-\mu_l)^2}{2\sigma_l^2}\right\} \qquad \text{for } l = 1, 2,$$

where μ_l and σ_l^2 are the mean and variance of pixel values in category l. The best classification criterion, i.e. the one which misclassifies the least number of pixels, allocates pixels with value k to category 1 if

$$p_1\phi_1(k) \geq p_2\phi_2(k),$$

and otherwise classifies them as 2. After substituting for ϕ and taking logs, the inequality becomes

$$k^2\left\{\frac{1}{\sigma_1^2} - \frac{1}{\sigma_2^2}\right\} - 2k\left\{\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2}\right\} + \left\{\frac{\mu_1^2}{\sigma_1^2} - \frac{\mu_2^2}{\sigma_2^2} + \log\frac{\sigma_1^2 p_2^2}{\sigma_2^2 p_1^2}\right\} \leq 0.$$

The left side of the inequality is a quadratic function in k. Let A, B and C denote the three terms in curly brackets, respectively. Then the criterion for allocating pixels with value k to category 1 is:

$$k^2 A - 2kB + C \leq 0.$$

There are three cases to consider:

(a) If $A = 0$ (i.e. $\sigma_1^2 = \sigma_2^2$), the criterion simplifies to one of allocating pixels with value k to category 1 if

$$2kB \geq C.$$

(If, in addition, $p_1 = p_2$ and $\mu_1 < \mu_2$, the criterion becomes $k \leq \frac{1}{2}\{\mu_1 + \mu_2\}$. Note that this is the intermeans criterion, which implicitly assumes that the two categories are of equal size.)

(b) If $B^2 < AC$, then the quadratic function has no real roots, and all pixels are classified as 1 if $A < 0$ (i.e. $\sigma_1^2 > \sigma_2^2$), or as 2 if $A > 0$.

(c) Otherwise, denote the roots t_1 and t_2 , where $t_1 \leq t_2$ and

$$t_1, t_2 = \frac{B \pm \sqrt{B^2 - AC}}{A}.$$

The criteria for category 1 are

$$t_1 < k \leq t_2 \qquad \text{if } A > 0,$$

$$k \leq t_1 \text{ or } k > t_2 \qquad \text{if } A < 0.$$

In practice, cases (a) and (b) occur infrequently, and if $\mu_1 < \mu_2$ the rule simplifies to the threshold:

$$\text{category 1 if a pixel value} \quad k \leq \frac{B + \sqrt{B^2 - AC}}{A}.$$

Kittler and Illingworth (1986) proposed an iterative **minimum-error algorithm**, which is based on this threshold and can be regarded as a generalization of the intermeans algorithm. Again, we will describe the algorithm in words, and then mathematically.

From an initial guess at the threshold, the proportions, means and variances of pixel values in the two categories are calculated. The threshold is repositioned according to the above criterion, and proportions, means and variances are recalculated. These steps are repeated until there are no changes in values between iterations. Mathematically:

1. Make an initial guess at a value for t.

2. Estimate p_1, μ_1 and σ_1^2 for pixels with values less than or equal to t, by

$$p_1 = \frac{1}{n^2} \sum_{k=0}^{t} h_k,$$

$$\mu_1 = \frac{1}{n^2 p_1} \sum_{k=0}^{t} k h_k,$$

$$\text{and } \sigma_1^2 = \frac{1}{n^2 p_1} \sum_{k=0}^{t} k^2 h_k \quad - \quad \mu_1^2.$$

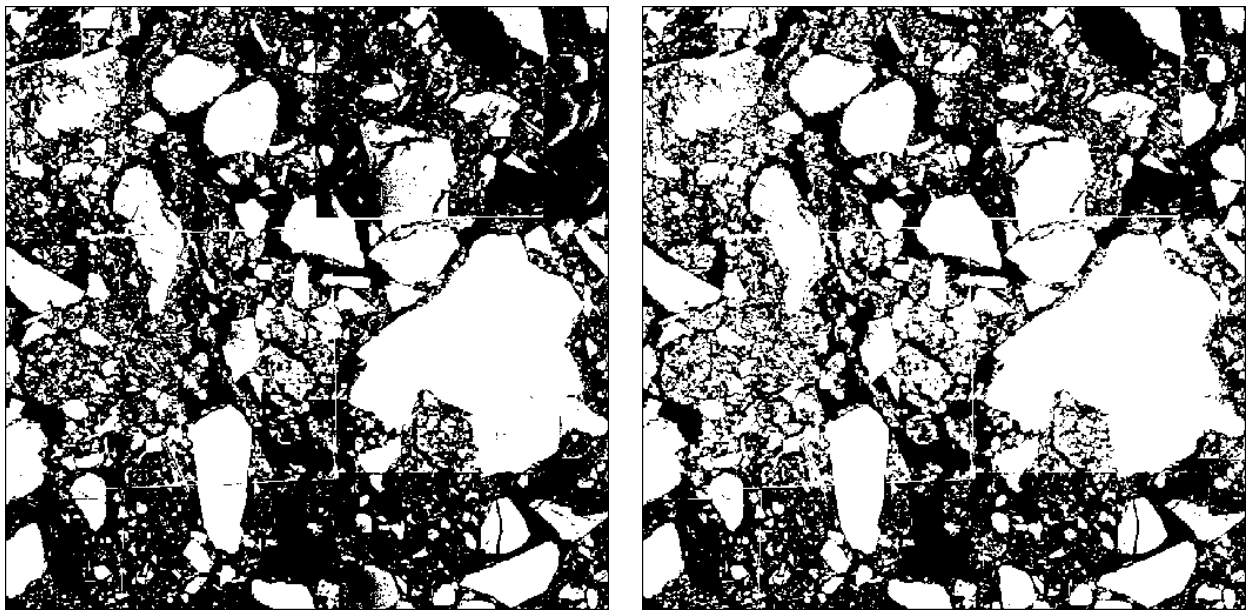
   Similarly, estimate p_2, μ_2 and σ_2^2 for pixels in the range $t+1$ to N.

3. Re-estimate t by

$$t = \left[\frac{B + \sqrt{B^2 - AC}}{A} \right],$$

   where A, B, C and [ ] have already been defined.

4. Repeat steps (2) and (3) until t converges to a stable value.



(a) (b)

Figure 4.4: Segmentations of the soil image, obtained by thresholding at: **(a)** 33, selected by the intermeans algorithm, **(b)** 24, selected by the minimum-error algorithm.

When applied to the soil image, the algorithm converged in 4 iterations to $t = 24$. Fig 4.4(b) shows the result, which is more satisfactory than that produced by the intermeans algorithm because it has allowed for a smaller proportion of air pixels ($p_1 = 0.45$, as compared with $p_2 = 0.55$). The algorithm has also taken account of the air pixels being less variable in value than those for the soil matrix ($\sigma_1^2 = 30$, whereas $\sigma_2^2 = 186$). This is in accord with the left-most peak in the histogram plot (Fig 4.3) being quite narrow.

Note that:

- The estimators in step (2) of the minimum error algorithm are biased because they do not allow for overlaps between the distributions in the mixture, although Cho, Haralick and Yi (1989) have suggested one way of doing so. However, distributions of pixel values seldom conform exactly to the Gaussian, so such refinements seem unnecessary.
- Many other automatic, histogram-based thresholding algorithms have been proposed (see Glasbey (1993) for a review). It was a popular area for research in the 1980s. Most algorithms have less theoretical justification than the two considered here.
- The intermeans and minimum-error algorithms extend simply to multiple thresholds, and to multivariate thresholds (which are the topic of the next subsection). It is also possible to specify a different threshold in each part of the image (see for example Chow and Kaneko, 1972). Such an approach might have been of assistance in segmenting the soil image, because some of the electron micrographs in the composite image montage had

a greater overall brightness than others. Alternatively, trend can sometimes be removed before thresholding an image, for example by using a top-hat transform (§5.5).

4.1.2 Multivariate classifiers

There are many ways to extend the concept of a threshold in order to segment a multivariate image, such as:

- Allocate pixel (i, j) to category 1 if

$$f_{ij,m} \leq t_m \quad \text{for } m = 1, \ldots, M,$$

  where subscript m denotes the variate number, and there are M variates.

- Or, more generally, use a **box classifier** where the conditions for category 1 are:

$$t_{1,m} < f_{ij,m} \leq t_{2,m} \quad \text{for } m = 1, \ldots, M.$$

- The condition could be a linear threshold:

$$\boldsymbol{c}^T \boldsymbol{f_{ij}} \leq t,$$

  (using *vector notation*, with $\boldsymbol{f_{ij}} = (f_{ij,1}, f_{ij,2}, \ldots, f_{ij,M})^T$, bold type used to denote vectors and the superscript T used to represent a vector transpose).

- Or, the range of pixel values for category 1 could be a general set S in M-dimensional space:

$$\boldsymbol{f_{ij}} \in S \subset \mathbb{R}^M.$$

  For example, if $M = 2$, S could be a circular disc, the set of all integer pairs (k, l) such that $k^2 + l^2 < 100$.

For the rest of this subsection we will be making use of *matrix notation*. Less-mathematical readers may prefer to skip to §4.1.3, which can be done without losing the sense of the rest of the chapter.

Multivariate thresholding criteria are more difficult to specify than univariate ones. Therefore the approach often adopted in a segmentation which is manual (i.e. under the user's control) is to choose pixels which are known to belong to target categories (known as the **training set**) and then use them to classify the rest of the image. This is called **supervised classification**. Suppose that regions $B_1, \ldots, B_R$ of the image have been identified as belonging to categories 1 to R respectively, then the mean pixel values in the regions can bee estimated as

$$\boldsymbol{\mu}_r = \frac{1}{N_r} \sum_{(i,j) \in B_r} \sum \boldsymbol{f_{ij}} \qquad \text{for } r = 1, \ldots, R,$$

where summation is over all the N_r pixels in region B_r. The simplest case to consider is where the variance-covariance matrices in the different categories are assumed to be the same. In this case, the common variance-covariance matrix, $\boldsymbol{V}$, can be estimated as

$$\boldsymbol{V} = \sum_{r=1}^{R} \left(\sum\sum_{(i,j)\in B_r} \boldsymbol{f_{ij}} \boldsymbol{f_{ij}}^T \quad - \quad N_r \boldsymbol{\mu}_r \boldsymbol{\mu}_r^T \right) \Big/ \sum_{r=1}^{R} N_r \ .$$

Distances in multidimensional space are measured taking into account the variance-covariance matrix $\boldsymbol{V}$. The **squared Mahalanobis distance** between pixel value $\boldsymbol{f_{ij}}$ and the rth category mean is

$$(\boldsymbol{f_{ij}} - \boldsymbol{\mu}_r)^T \boldsymbol{V}^{-1} (\boldsymbol{f_{ij}} - \boldsymbol{\mu}_r),$$

where the superscript '-1' denotes a matrix inverse.

The segmentation rule is to allocate each pixel in the image to the nearest category mean, as measured by the above distance. The rule simplifies to a set of linear thresholds. For example, in the two category case, a pixel is allocated to category 1 if

$$\boldsymbol{\mu}_1^T \boldsymbol{V}^{-1} \boldsymbol{\mu}_1 - 2\boldsymbol{\mu}_1^T \boldsymbol{V}^{-1} \boldsymbol{f_{ij}} \quad \leq \quad \boldsymbol{\mu}_2^T \boldsymbol{V}^{-1} \boldsymbol{\mu}_2 - 2\boldsymbol{\mu}_2^T \boldsymbol{V}^{-1} \boldsymbol{f_{ij}} \quad .$$

This is **linear discrimination**. If the variances are not assumed to be equal, then this results in **quadratic discrimination**. For more details, see for example Krzanowski (1988, §12.2).

Fig 4.5 shows this method applied to the bivariate image obtained by combining the MRI proton density and inversion recovery images of a woman's chest (Figs 1.7(a) and (b)). Four training areas have been selected manually as representatives of

1. lung tissue,
2. blood in the heart,
3. muscle, and other lean tissues,
4. adipose tissues, such as subcutaneous fat.

These areas are shown superimposed on the inversion recovery and proton density images in Figs 4.5(a) and (b).

Fig 4.5(c) is a scatter plot of $f_{ij,2}$ against $f_{ij,1}$ for the training pixels, together with the linear separators of all pixel values into the four classes. For example, the muscle pixels are placed in the top-left of Fig 4.5(c) because they have high values of proton density but low values from the inversion recovery signal, as can be seen in Figs 4.5(a) and (b). The points have been coded in progressively darker shades of grey, to denote lung, muscle, blood and fat pixels respectively. There is substantial overlap between blood and fat pixels in Fig 4.5(c), so that even the training set has not been classified precisely.

Fig 4.5(d) shows the pixel allocation of the whole image. Air pixels, for which $\boldsymbol{f_{ij}} = \boldsymbol{0}$, have been excluded from all classes. The lungs, shown as the lightest shade of grey, have been

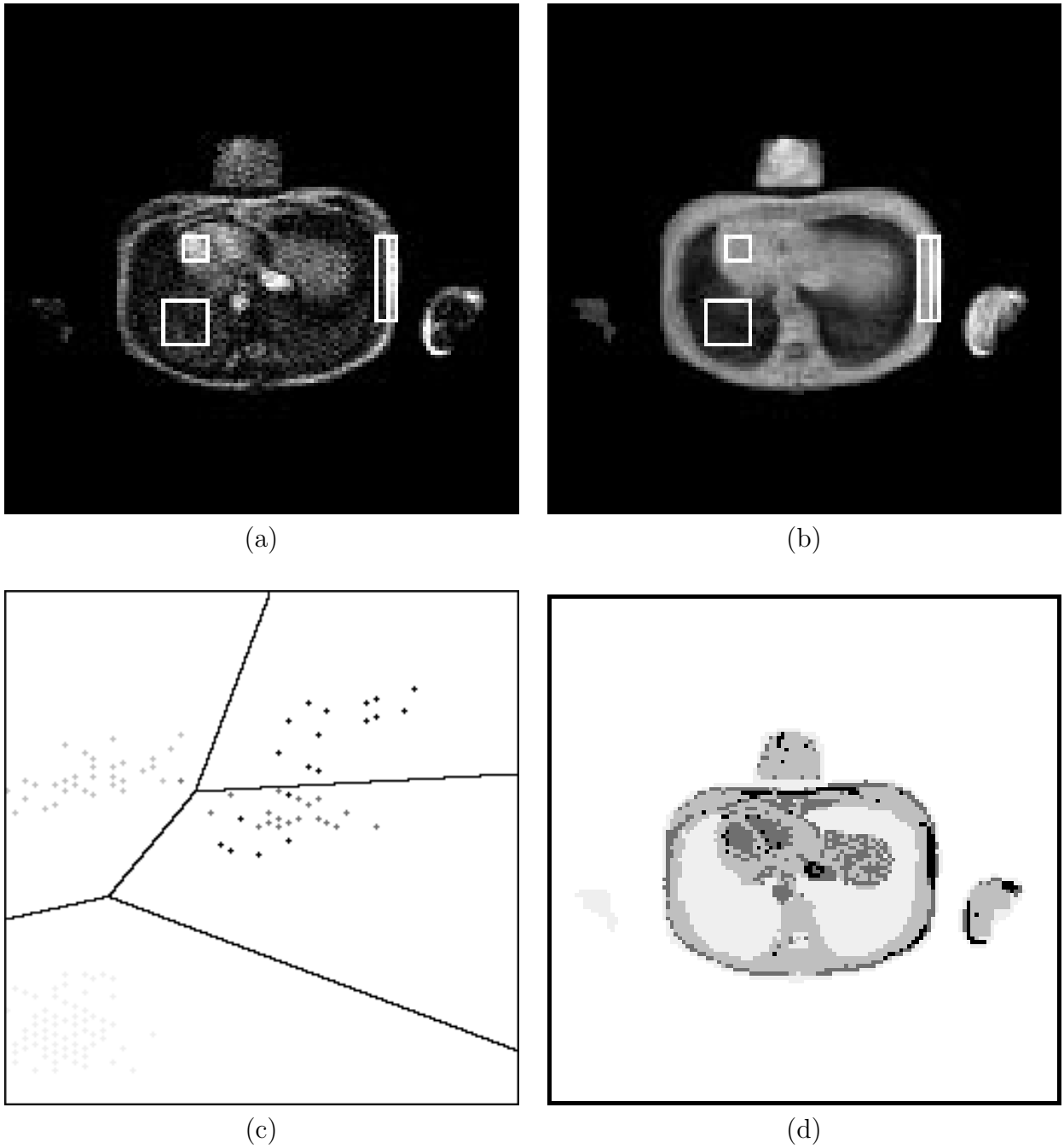


Figure 4.5: Manual segmentation of the bivariate MRI image: **(a)** inversion recovery image with boundaries of 4 training areas superimposed — these are, from left to right, lung tissue, moving blood, muscle and subcutaneous fat; **(b)** proton density image with the same training areas shown; **(c)** scatter plot of values of training pixels, and linear discrimination classification, with points displayed in increasing levels of darkness for lung pixels, muscle, blood and fat; **(d)** segmentation of image by linear discrimination, using the same greyscale as in (c).

correctly identified, although note that boundary pixels around the whole body have also been misspecified in this category. Most other pixels have been classified as muscle, the second lightest shade of grey. The remaining pixels, displayed as dark grey for the blood category and black for fat, are mixed up, but taken together they serve to identify the major blood vessels and subcutaneous fat round the body.

Automatic methods can be devised which do not require a training set. Such methods are termed **unsupervised classification**. One approach, **k-means clustering**, is an automatic version of linear discrimination. We will describe the algorithm in words, and then more formally.

The number of categories, R, must be known. More commonly, several different values are tried. From an initial guess of the parameter values, that is the category means and the variance-covariance matrix, all pixels in an image are classified by the linear discrimination criterion. Then the parameters are estimated using all pixels classified in each category, and pixels are reclassified. These steps are repeated until there are no changes in values between iterations. More formally:

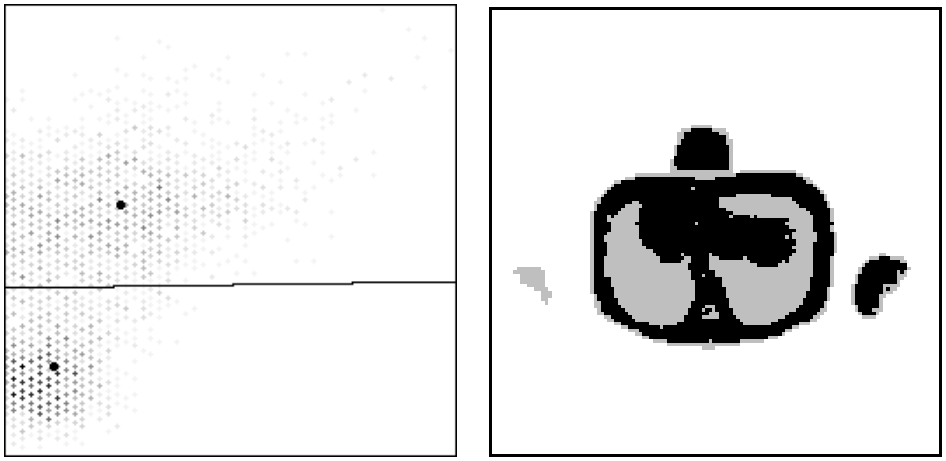
1. Make initial guesses at $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_R$, such as modes in the multivariate histogram or random values, and initially set the matrix $\boldsymbol{V}$ to the identity matrix.
2. Segment the image according to the linear discrimination criterion.
3. Estimate $\boldsymbol{\mu}_r$ as the mean of pixel values classified as r, for $r = 1, \ldots, R$, and estimate $\boldsymbol{V}$ as the variance-covariance matrix averaged over categories, using the equations already given.
4. Repeat steps (2) and (3) until convergence.

Fig 4.6 shows the results produced by applying the k-means clustering algorithm to the bivariate MRI image. For values of R of 2, 3 and 4, the algorithm was run 50 times from randomly chosen starting points, and the segmentation which had the smallest within-group sum of squares,

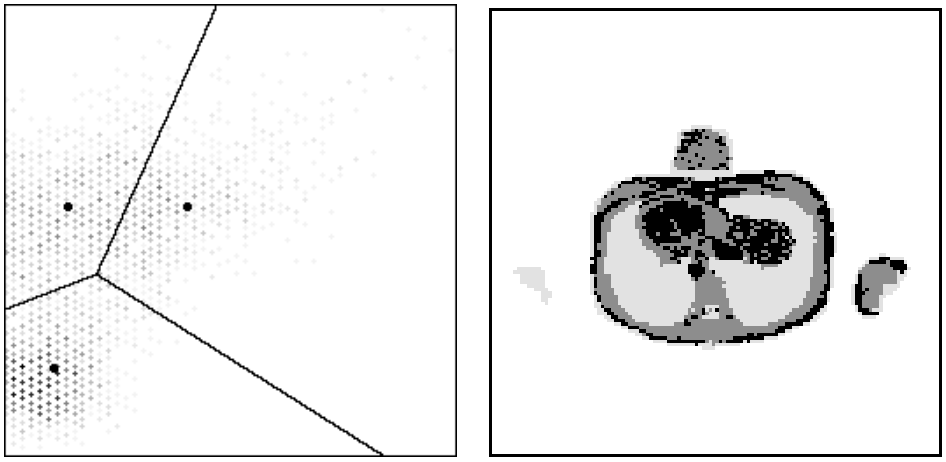
$$\sum_{i=1}^{n}\sum_{j=1}^{n}\left\{\min_{r=1,\ldots,R}\ (\boldsymbol{f_{ij}}-\boldsymbol{\mu}_r)^T\boldsymbol{V}^{-1}(\boldsymbol{f_{ij}}-\boldsymbol{\mu}_r)\right\},$$

was chosen. Fig 4.6(a) shows the bivariate histogram of all pixel values, the means of the two categories and the division between them, when two categories were sought. Fig 4.6(b) shows the segmented image using this criterion. Lung tissue appears to have been distinguished from other tissues. Figs 4.6(c) and (d) give analogous results for three clusters, and Figs 4.6(e) and (f) for four clusters. The segmentation seems to be as good as that achieved in Fig 4.5(d). The second-lightest shade of grey in Fig 4.6(f) appears to identify muscle tissue, and blood and subcutaneous fat constitute a combined third and fourth class.

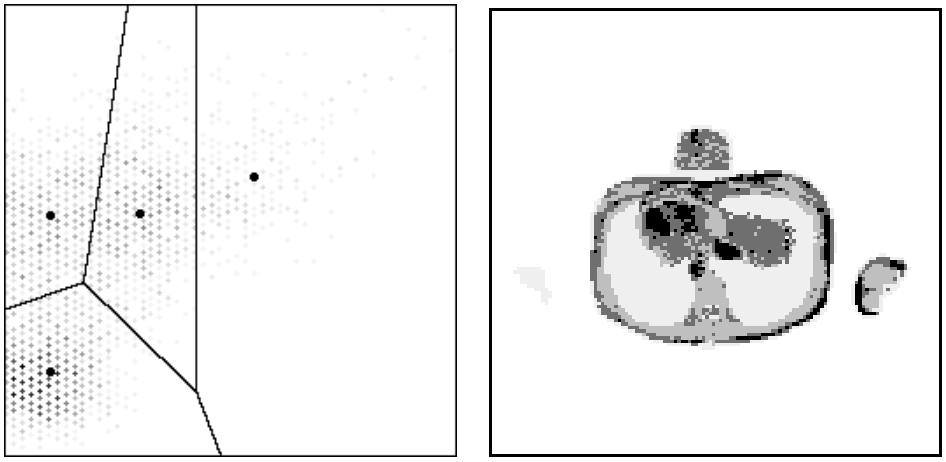
The algorithm can be extended to handle unequal variances and proportions, as considered by Maronna and Jacovkis (1974). This is a multivariate generalization of the minimum error



(a) (b)



(c) (d)



(e) (f)

Figure 4.6: Automatic segmentations of the MRI image into 2, 3 and 4 classes: **(a)** bivariate histogram of pixel values, together with centres of two clusters and the division between them obtained by k-means clustering, **(b)** result displayed as an image, **(c)** bivariate histogram and optimal division into 3 clusters, and **(d)** image display, **(e)** bivariate histogram and optimal division into 4 clusters, and **(f)** image display.

algorithm considered in the previous subsection. However, note that it is not always necessary to segment an image in order to extract summary information from it. For example, areas of different tissue types could have been estimated directly from the bivariate histogram of pixel values in the MRI image. Similarly, areas of different land use can be estimated from the distribution of pixel values in the multivariate Landsat image. We will return to this example in §6.1.

4.1.3 Contextual classifiers

Thresholding is most successful when there is little overlap in distributions of pixel values from the different categories in an image. Noise is one cause of overlap, which can be reduced by using a smoothing filter. Unfortunately, filters, linear ones in particular, also blur edges to some extent and therefore produce pixels with values intermediate between category means. To illustrate the effects of filtering, Fig 4.7(a) shows the histogram of the muscle fibres image (Fig 1.6(b)), and Fig 4.7(b) shows the histogram after the application of a Gaussian filter with $\sigma^2 = 6\frac{2}{3}$ (§3.1.1) to the image. Both histograms are displayed on square-root scales in order to reveal the details in small values. Note that Fig 4.7(b) shows a greater number of pixels with values intermediate between the two peaks. Usually, smoothing would also produce narrower peaks in the histogram, but this is not the case here because noise levels are low.

Kirby and Rosenfeld (1979) proposed forming a bivariate histogram by plotting pixel values in the original image against those from its filtered form. For the muscle fibres image this is shown in Fig 4.7(c). (The histograms in Figs 4.7(a) and (b) are the marginal histograms obtainable by summing over column or rows in Fig 4.7(c).) Some denser regions of pixel values are visible on the diagonal joining the two main clusters of points.

If a histogram is formed only from pixels within 5 of the diagonal of this plot, then Fig 4.7(d) is the result. The histogram has been sharpened in comparison with Figs 4.7(a) and (b), to the extent that additional, small peaks have become visible. Fig 4.8 shows the effect of multiple thresholding of the muscle fibres image, at values of 50, 90, 130 and 165 which are indicated by arrows in Fig 4.7(d). Pixels outside the range 50 to 165 are displayed as white. Within this range, pixels are displayed as three shades of grey of increasing lightness in the three intervals 51–90, 91–130 and 131–165. The sub-peaks can be seen to be from pixels on the boundaries between fibres and from four muscle fibres with pixel values intermediate between the two main fibre types. These are the third type of fibre, the fast-twitch glycolytic ones mentioned in §1.1.3.

Rosenfeld and co-workers published several variants on the basic idea of a bivariate histogram. The work culminated in the paper by Weszka and Rosenfeld (1979), in which it was advocated that the bivariate histogram should be formed from image values and output from an edge filter. This can be viewed as a variant of Fig 4.7(c) because the difference between an image and its smoothed form is a type of edge filter. The general principle is that, by constructing a

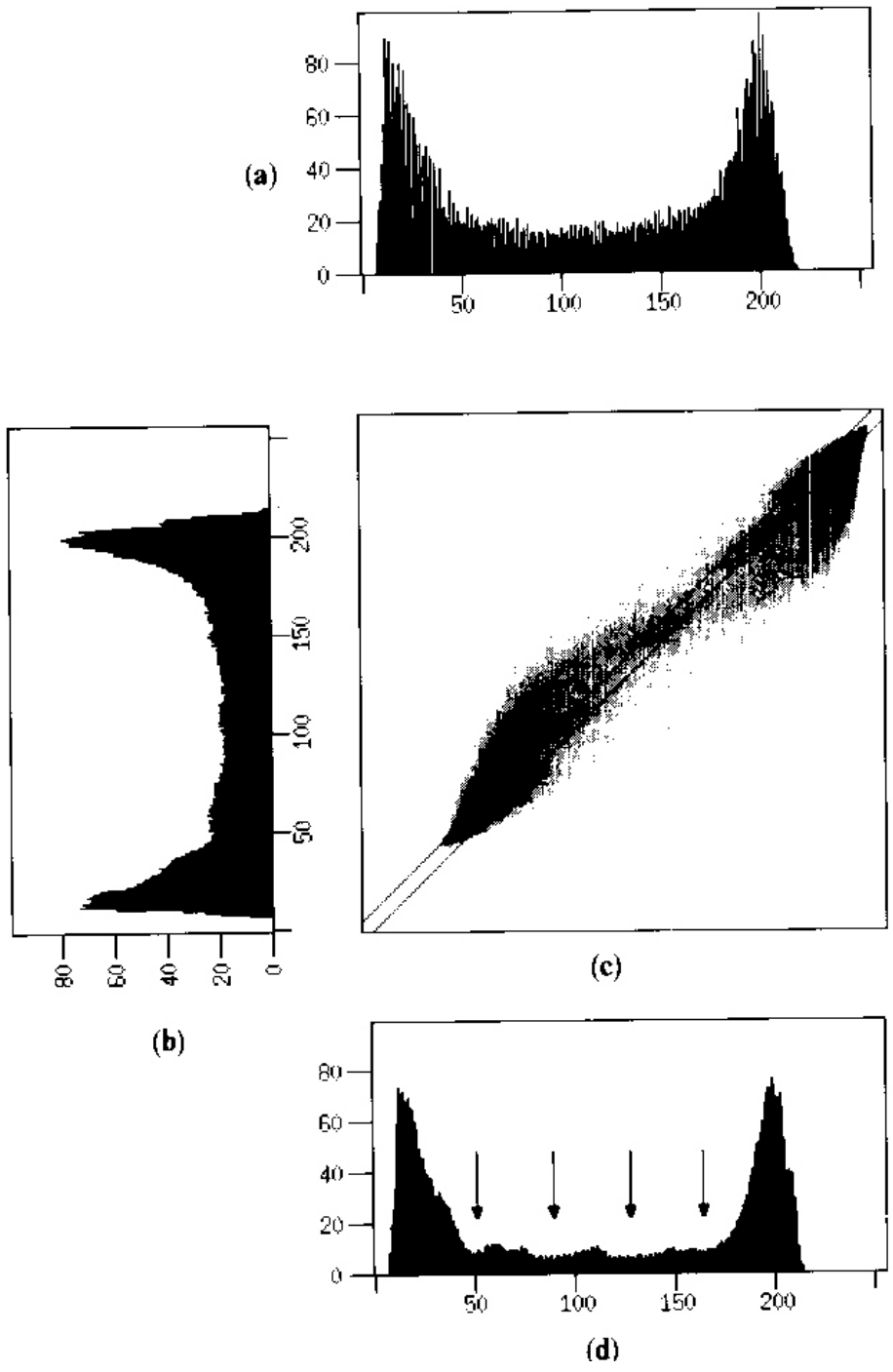


Figure 4.7: Histograms of muscle fibres image: **(a)** original pixel values, displayed on a square-root scale; **(b)** image after smoothing by Gaussian filter ($\sigma^2 = 6\frac{2}{3}$); **(c)** bivariate histogram of original pixel values and those after smoothing, with all counts exceeding 25 displayed as black and the parallel diagonal lines indicating the points extracted for (d); **(d)** histogram of pixel values after smoothing those values within 5 of those before smoothing, the arrows indicate the thresholds used in Fig 4.8.

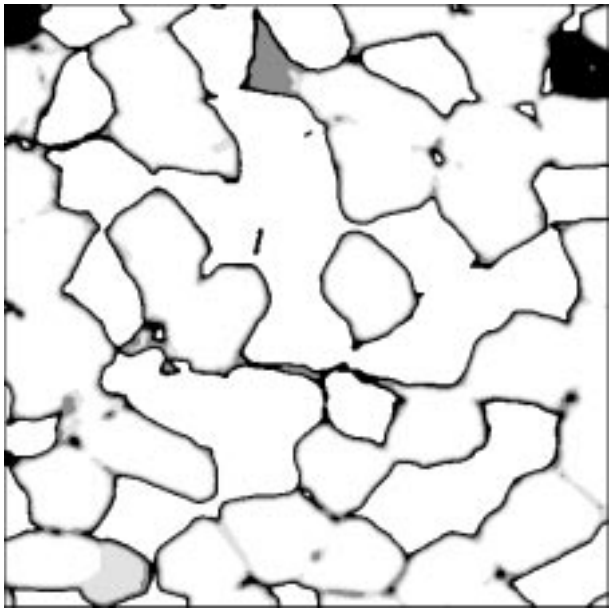


Figure 4.8: Thresholded muscle fibres image. Pixels in the range 0 to 50 and 166 to 255 are displayed as white, those between 51 and 90 are black, between 91 and 130 they are dark grey and between 131 and 165 they are shown as light grey.

histogram from non-edge pixels only, its peaks should be clearer.

Another way of using contextual information is by applying a **majority filter** to a segmented image, termed **post-classification smoothing**. For example, if pixel (i, j) has been labelled as category g_{ij} by a segmentation algorithm, then a majority filter relabels (i, j) as the most common category in a $(2m+1) \times (2m+1)$ window centred on pixel (i, j), i.e. the set $\{g_{i+k,j+l}$ for $k, l = -m, \ldots, m\}$.

Contextual information can also be used in automatic segmentation algorithms. Mardia and Hainsworth (1988) investigated incorporating post-classification smoothing into a thresholding algorithm. (Their paper also provides a succinct review of algorithms for thresholding images whose pixel values are mixtures of Gaussian distributions.) To explain the method, we will consider the simplest case. First we will discuss it in words, and then more formally.

From an initial guess at a threshold, an image is segmented. Then the classified image is smoothed using a majority filter, so that neighbouring pixels are more often classified the same. Mean pixel values are evaluated in each category and the threshold is re-estimated using the intermeans criterion (§4.1.1). The image is segmented and smoothed again, and so on until the threshold does not change between successive iterations. Formally:

1. Make an initial guess at a threshold t.

2. Segment the image, by thresholding, and store the result in an array g:

$$g_{ij} = 1 \quad \text{if } f_{ij} \leq t, \quad \text{otherwise } g_{ij} = 2.$$

3. Apply a majority filter to g, and record the new labels in array g'. Therefore, g'_{ij} is set to the most common category in the set $\{g_{i+k,j+l} \text{ for } k, l = -m, \ldots, m\}$. This is repeated for every value of i and j from $(m+1)$ to $(n-m)$. (Ranges having been chosen to avoid image borders, as in chapter 3.)

4. Calculate the mean pixel value in category 1,

$$\mu_1 = \frac{1}{N_1} \sum\sum_{\{(i,j) \text{ such that } g'_{ij}=1\}} f_{ij},$$

where N_1 is the number of pixels in category 1. Similarly, calculate μ_2.

5. Re-estimate t as $[\frac{1}{2}(\mu_1 + \mu_2)]$.

6. Repeat steps (2) to (5) until t converges to a stable value, and then stop after step (3).

Fig 4.9(b) shows the segmentation of the log-transformed SAR image (as shown in Fig 2.6) into two classes using the above algorithm and a 5×5 smoothing window. The algorithm converged in four iterations, starting with a threshold set at the median pixel value. The window size was chosen by trial-and-error: a size less than 5×5 was inadequate for smoothing the inconsistencies in classification caused by the noise, whereas a larger size produced an over-smooth segmentation — corners of fields became rounded. Fig 4.9(a) shows the result of the same algorithm without smoothing — that is, the intermeans algorithm. The noise is so great in this image that the histogram of pixel values is unimodal and no threshold can achieve a better segmentation.

Mardia and Hainsworth compared their algorithm with a simple form of **Bayesian image restoration**, which uses Besag's (1986) **iterated conditional modes (ICM)** algorithm. We will consider the simplest case of ICM, first in words and then formally.

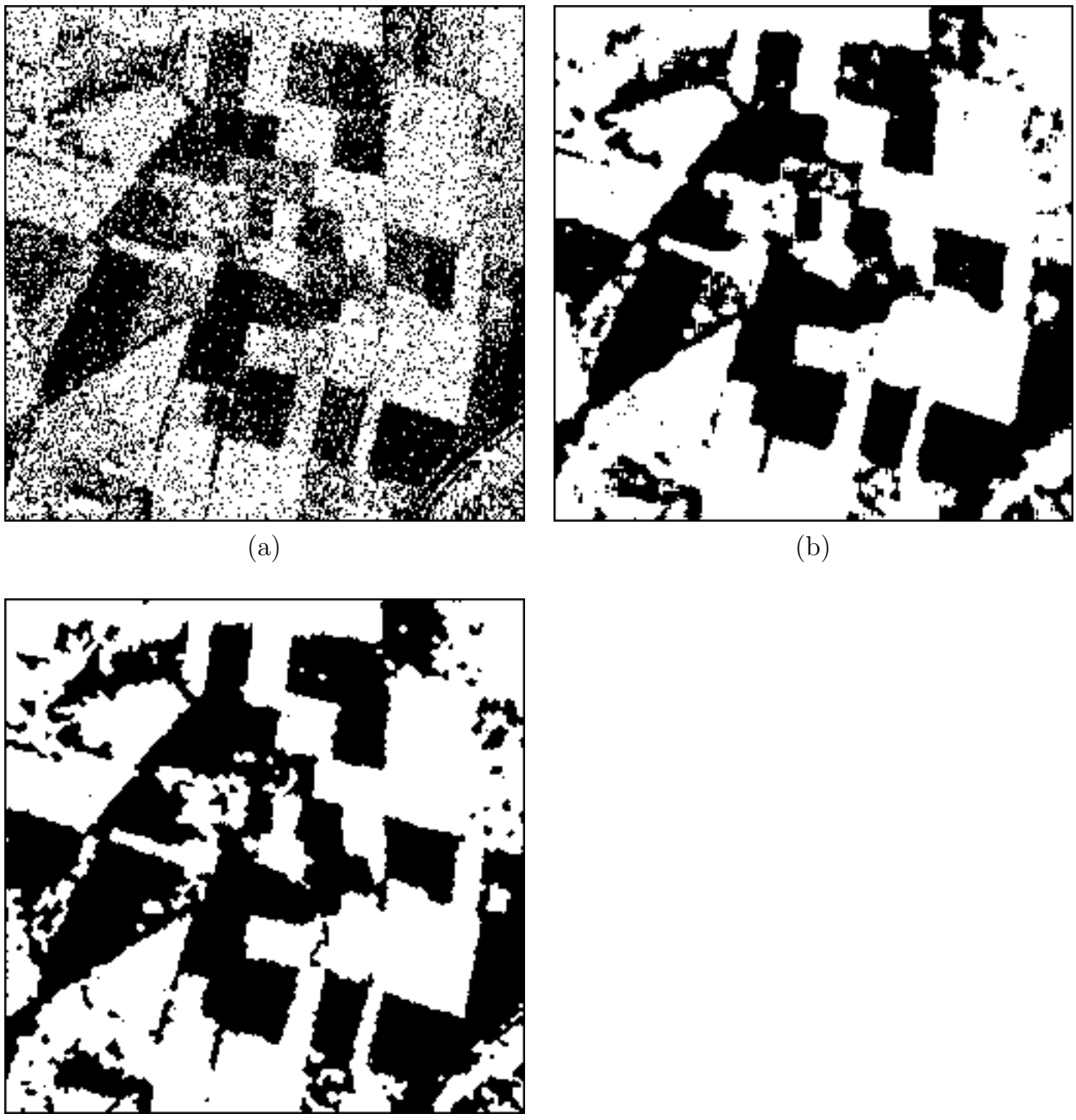
From an initial guess at a threshold, or equivalently of the two category means, the image is segmented. Mean pixel values are evaluated in each category and the image is re-segmented, pixel by pixel, using a variable threshold which takes account both of the category means and of the current classification of neighbouring pixels. Category means are recalculated and the recursive segmentation procedure is re-applied. These steps are repeated until the segmentation remains (almost) unchanged on successive iterations. More formally:

1. Make an initial guess at μ_1 and μ_2, and set

$$g_{ij} = 1 \quad \text{if } (f_{ij} - \mu_1)^2 \leq (f_{ij} - \mu_2)^2, \quad \text{otherwise } g_{ij} = 2.$$

Or equivalently,

$$g_{ij} = 1 \quad \text{if } f_{ij} \leq t, \quad \text{where } t = \left[\frac{\mu_1 + \mu_2}{2}\right].$$



(a) (b)

(c)

Figure 4.9: Three segmentations of the log-transformed SAR image: **(a)** obtained by thresholding using the intermeans algorithm, **(b)** thresholding combined with a post-classification smoothing by majority filter in a 5×5 window, **(c)** iterated conditioned modes (ICM) classification, with $\beta = 1.5$.

2. Calculate the mean pixel values in the two categories, μ_1 and μ_2. Also calculate the within-category variance
$$\sigma^2 = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(f_{ij} - \mu_{g_{ij}})^2.$$

3. Re-segment, recursively, as follows. For each value of i between 1 and n apply the following criterion in turn for j between 1 and n:
$$g_{ij} = 1 \quad \text{if} \quad (f_{ij} - \mu_1)^2 - \beta\sigma^2 N_{ij} \quad \leq \quad (f_{ij} - \mu_2)^2 - \beta\sigma^2(8 - N_{ij}),$$
where N_{ij} is the number of the eight neighbours of pixel (i, j), currently classified as 1 and β is a constant whose value must be chosen. A value of 1.5 has often been used in the literature.

   Equivalently, this reclassification be viewed as a variable threshold:
$$g_{ij} = 1 \quad \text{if } f_{ij} \leq t_{ij}, \qquad \text{where } t_{ij} = \left[\frac{(\mu_1 + \mu_2)}{2} \quad + \quad \frac{\beta\sigma^2(2N_{ij} - 8)}{2(\mu_2 - \mu_1)}\right].$$
   Note that $(i-1, j-1), (i-1, j), (i-1, j+1)$ and $(i, j-1)$ will have already been updated, although alternatively, all pixels can be reclassified simultaneously. The two approaches are termed **asynchronous** and **synchronous**, respectively.

4. Repeat steps (2) and (3) until no (or only a few) pixels change category.

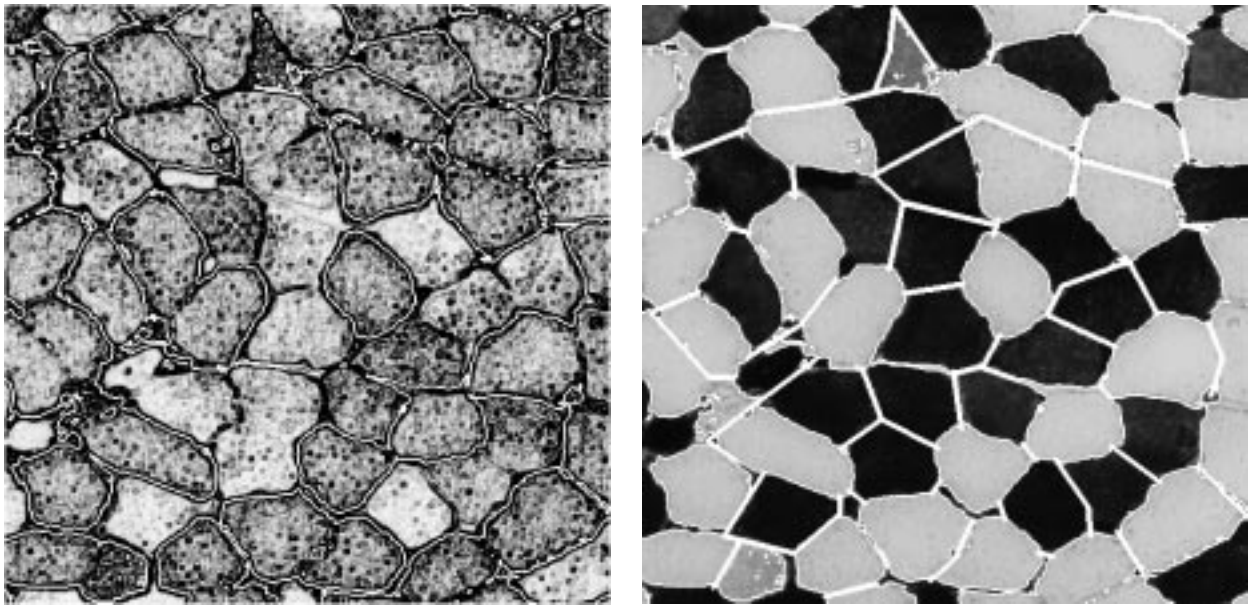
Fig 4.9(c) shows the results of applying the ICM algorithm to the SAR image. In this case, 20 iterations were performed, by which stage fewer than 5 pixels changed classes between iterations. The final segmentation is very similar to that produced by applying the majority filter, shown in Fig 4.9(b). Mardia and Hainsworth (1988) also found that the two algorithms often gave very similar results.

The ICM algorithm is a very simple example of Bayesian image restoration — a potentially powerful technique which has received much attention from statisticians in the last decade, following seminal papers by Geman and Geman (1984) and Besag (1986). The underlying principle is that of combining

prior information on what the segmented image should be like, and

data, the observed pixel values in an image,

in order to produce a segmentation. In our example, the term βN_{ij} is motivated by a **Markov Random Field model**, and favours a segmentation in which neighbouring pixels are classified the same. The main attraction of Bayesian image restoration is that it is open to many refinements, such as incorporating more specific prior information. However, computational feasibility of Bayesian restoration algorithms can be a limiting factor. See, for example, Aykroyd and Green (1991), Ripley and Sutherland (1990) and Qian and Titterington (1991) for a range of applications.



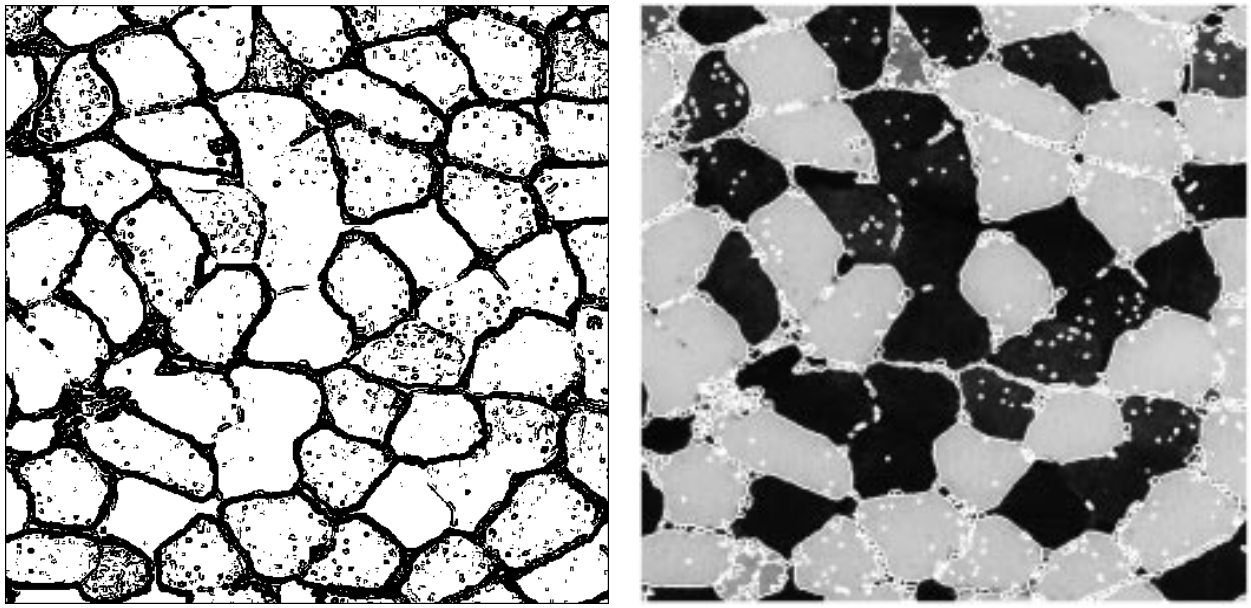
(a) (b)

Figure 4.10: **(a)** Boundaries obtained by thresholding the muscle fibres image, superimposed on output for Prewitt's filter, with values between 0 and 5 displayed in progressively darker shades of grey and values in excess of 5 displayed as black. **(b)** Manual segmentation of the image by addition of extra lines to boundaries obtained by thresholding, superimposed in white on the original image.

4.2 Edge-based segmentation

As we have seen, the results of threshold-based segmentation are usually less than perfect. Often, a scientist will have to make changes to the results of automatic segmentation. One simple way of doing this is by using a computer mouse to control a screen cursor and draw boundary lines between regions. Fig 4.10(a) shows the boundaries obtained by thresholding the muscle fibres image (as already displayed in Fig 4.1(a)), superimposed on the output from Prewitt's edge filter (§3.4.2), with the contrast stretched so that values between 0 and 5 are displayed as shades of grey ranging from white to black and values exceeding 5 are all displayed as black. This display can be used as an aid to determine where extra boundaries need to be inserted to fully segment all muscle fibres. Fig 4.10(b) shows the result after manually adding 71 straight lines.

Algorithms are available for semi-automatically drawing edges, whereby the scientist's rough lines are smoothed and perturbed to maximise some criterion of match with the image (see, for example, Samadani and Han, 1993). Alternatively, edge finding can be made fully automatic, although not necessarily fully successful. Fig 4.11(a) shows the result of applying Prewitt's edge filter to the muscle fibre image. In this display, the filter output has been thresholded at a value of 5: all pixels exceeding 5 are labelled as edge pixels and displayed as black. Connected chains of edge pixels divide the image into regions. Segmentation can be achieved by allocating to a single category all non-edge pixels which are not separated by an edge. Rosenfeld and



(a) (b)

Figure 4.11: **(a)** Thresholded output from Prewitt's edge filter applied to muscle fibres image: values greater than 5 are displayed as black, those less than or equal to 5 as white. **(b)** Boundaries produced from connected regions in (a), superimposed in white on the original image.

Pfaltz (1966) gave an efficient algorithm for doing this for 4- and 8-connected regions, termed a **connected components algorithm**. We will describe this algorithm in words, and then mathematically.

The algorithm operates on a **raster scan**, in which each pixel is visited in turn, starting at the top-left corner of the image and scanning along each row, finishing at the bottom-right corner. For each non-edge pixel, (i, j), the following conditions are checked. If its already-visited neighbours — $(i-1, j)$ and $(i, j-1)$ in the 4-connected case, also $(i-1, j-1)$ and $(i-1, j+1)$ in the 8-connected case — are all edge pixels, then a new category is created and (i, j) is allocated to it. Alternatively, if all its non-edge neighbours are in a single category, then (i, j) is also placed in that category. The final possibility is that neighbours belong to two or more categories, in which case (i, j) is allocated to one of them and a note is kept that these categories are connected and therefore should from then on be considered as a single category. More formally, for the simpler case of 4-connected regions:

- Initialize the count of the number of categories by setting $K = 0$.
- Consider each pixel (i, j) in turn in a raster scan, proceeding row by row ($i = 1, \ldots, n$), and for each value of i taking $j = 1, \ldots, n$.
- One of four possibilities apply to pixel (i, j):
    1. If (i, j) is an edge pixel then nothing needs to be done.

2. If both previously-visited neighbours, $(i-1, j)$ and $(i, j-1)$, are edge pixels, then a new category has to be created for (i, j):

$$K \to K+1, \quad h_K = K, \quad g_{ij} = K,$$

where the entries in $h_1, \ldots, h_K$ are used to keep track of which categories are equivalent, and g_{ij} records the category label for pixel (i, j).

3. If just one of the two neighbours is an edge pixel, then (i, j) is assigned the same label as the other one:

$$g_{ij} = \begin{cases} g_{i-1,j} & \text{if } (i, j-1) \text{ is the edge pixel,} \\ g_{i,j-1} & \text{otherwise.} \end{cases}$$

4. The final possibility is that neither neighbour is an edge pixel, in which case (i, j) is given the same label as one of them:

$$g_{ij} = g_{i-1,j},$$

and if the neighbours have labels which have not been marked as equivalent, i.e. $h_{g_{i-1,j}} \neq h_{g_{i,j-1}}$, then this needs to be done (because they are connected at pixel (i, j)). The equivalence is recorded by changing the entries in $h_1, \ldots, h_K$, as follows:
   - Set $l_1 = \min(h_{g_{i-1,j}}, h_{g_{i,j-1}})$ and $l_2 = \max(h_{g_{i-1,j}}, h_{g_{i,j-1}})$.
   - For each value of k from 1 to K, if $h_k = l_2$ then $h_k \to l_1$.

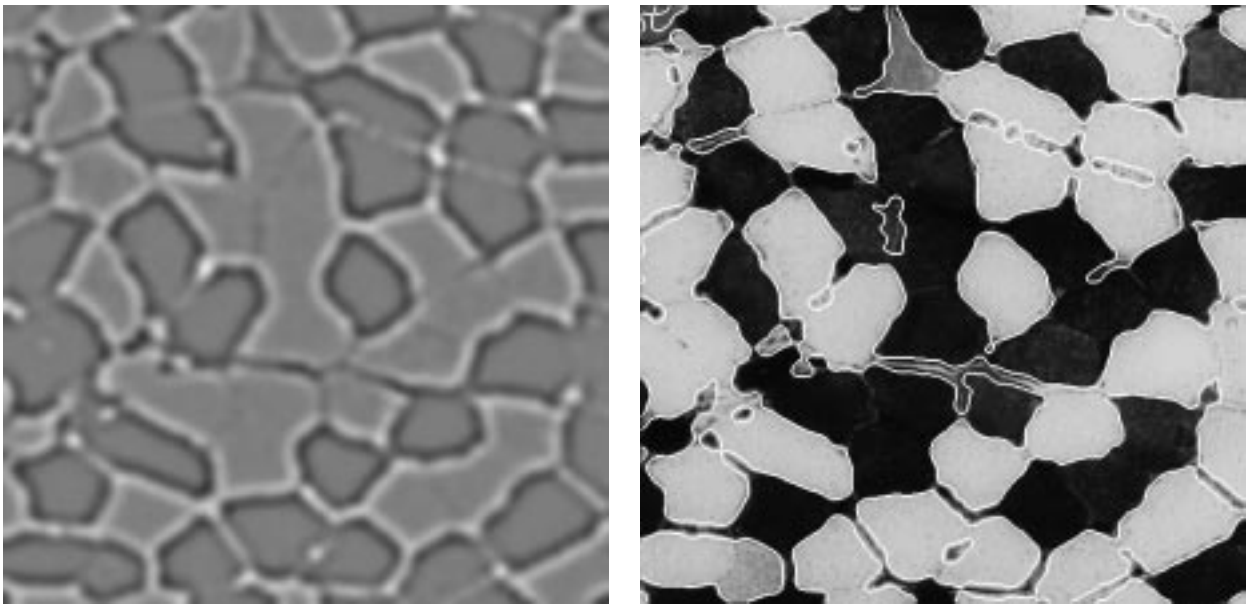
- Finally, after all the pixels have been considered, the array of labels is revised, taking into account which categories have been marked for amalgamation:

$$g_{ij} \to h_{g_{ij}} \quad \text{for } i, j = 1, \ldots, n.$$

After application of the labelling algorithm, superfluous edge pixels — that is, those which do not separate classes — can be removed: any edge-pixel which has neighbours only of one category is assigned to that category.

Fig 4.11(b) shows the result of applying the labelling algorithm with edges as shown in Fig 4.11(a), and removing superfluous edge pixels. The white boundaries have been superimposed on the original image. Similarly, small segments (say less than 500 pixels in size) which do not touch the borders of the image can be removed, leading to the previously displayed Fig 4.1(b). The segmentation has done better than simple thresholding, but has failed to separate all fibres because of gaps in output from Prewitt's edge filter. Martelli (1976), among others, has proposed algorithms for bridging these gaps.

Another edge-detection filter considered in chapter 3 was the Laplacian-of-Gaussian (§3.1.2). This filter can also be used to segment images, using the zero-crossings of the output to specify positions of boundaries. One advantage over Prewitt's filter is that the zero-crossings always form closed boundaries. Fig 4.12(a) shows output from the Laplacian-of-Gaussian filter



(a) (b)

Figure 4.12: **(a)** Output from Laplacian-of-Gaussian filter ($\sigma^2 = 27$) applied to muscle fibres image. **(b)** Zero-crossings from (a) with average image gradients in excess of 1.0, in white superimposed on the image.

($\sigma^2 = 27$), applied to the muscle fibres image. Boundaries corresponding to weak edges can be suppressed by applying a threshold to the average gradient strength around a boundary. Fig 4.12(b) shows zero-crossings of the Laplacian-of-Gaussian filter with average gradients exceeding unity, superimposed on the original image. In this application the result can be seen to be little better than simple thresholding.

4.3 Region-based segmentation

Segmentation may be regarded as **spatial clustering**:

- *clustering* in the sense that pixels with similar values are grouped together, and
- *spatial* in that pixels in the same category also form a single connected component.

Clustering algorithms may be agglomerative, divisive or iterative (see, for example, Gordon, 1981). Region-based methods can be similarly categorized into:

- those which **merge** pixels,
- those which **split** the image into regions, and

- those which both **split-and-merge** in an iterative search scheme.

The distinction between edge-based and region-based methods is a little arbitrary. For example, in §4.2 one of the algorithms we considered involved placing all neighbouring non-edge pixels in the same category. In essence, this is a merging algorithm.

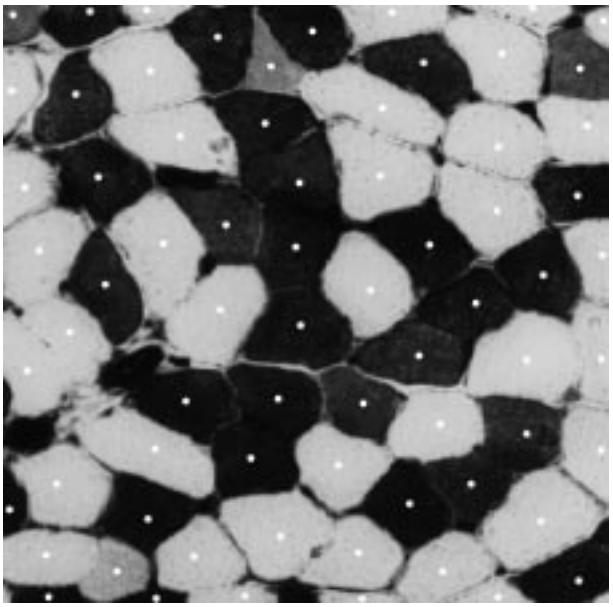
Seeded region growing is a semi-automatic method of the merge type. We will explain it by way of an example. Fig 4.13(a) shows a set of *seeds*, white discs of radius 3, which have been placed inside all the muscle fibres, using an on-screen cursor controlled by a computer mouse. Fig 4.13(b) shows again the output from Prewitt's edge filter. Superimposed on it in white are the seeds and the boundaries of a segmentation produced by a form of **watershed algorithm**. The boundaries are also shown superimposed on the original muscle fibres image in Fig 4.13(c). The watershed algorithm operates as follows (we will explain the name later).

For each of a sequence of increasing values of a threshold, all pixels with edge strength less than this threshold which form a connected region with one of the seeds are allocated to the corresponding fibre. When a threshold is reached for which two seeds become connected, the pixels are used to label the boundary. A mathematical representation of the algorithm is too complicated to be given here. Instead, we refer the reader to Vincent and Soille (1991) for more details and an efficient algorithm. Meyer and Beucher (1990) also consider the watershed algorithm, and added some refinements to the method.

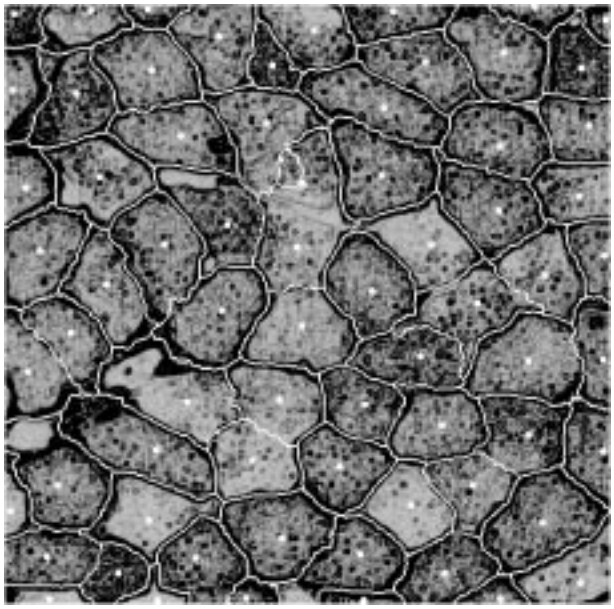
Note that:

- The use of discs of radius 3 pixels, rather than single points, as seeds make the watershed results less sensitive to fluctuations in Prewitt's filter output in the middle of fibres.
- The results produced by this semi-automatic segmentation algorithm are almost as good as those shown in Fig 4.10(b), but the effort required in positioning seeds inside muscle fibres is far less than that required to draw boundaries.
- Adams and Bischof (1994) present a similar seeded region growing algorithm, but based directly on the image greyscale, not on the output from an edge filter.

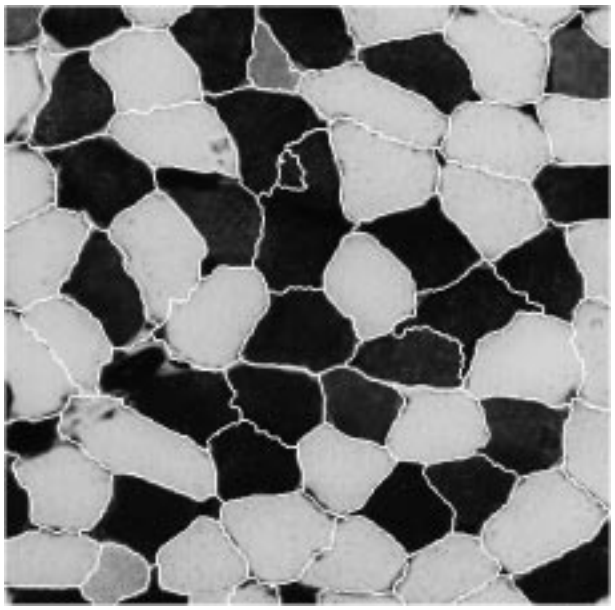
Th watershed algorithm, in its standard use, is fully automatic. Again, we will demonstrate this by illustration. Fig 4.14 shows the output produced by a variance filter (§3.4.1) with Gaussian weights ($\sigma^2 = 96$) applied to the muscle fibres image after histogram-equalization (as shown in Fig 2.7(d)). The white seeds overlie all the local minima of the filter output, that is, pixels whose neighbours all have larger values and so are shaded lighter. Note that it is necessary to use a large value of σ^2 to ensure that the filter output does not have many more local minima. The boundaries produced by the watershed algorithm have been added to Fig 4.14. An intuitive way of viewing the watershed algorithm is by considering the output from the variance filter as an *elevation map*: light areas are high ridges and dark areas are valleys (as in Fig 1.5). Each local minimum can be thought of as the point to which any water falling on the region drains, and the segments are the catchments for them. Hence, the boundaries, or watersheds, lie along tops of ridges. The previously mentioned Fig 4.1(c) shows this segmentation superimposed on the original image.



(a)



(b)



(c)

Figure 4.13: Manual segmentation of muscle fibres image by use of watersheds algorithm: **(a)** manually positioned 'seeds' in centres of all fibres, **(b)** output from Prewitt's edge filter, together with watershed boundaries, **(c)** watershed boundaries superimposed on the image.

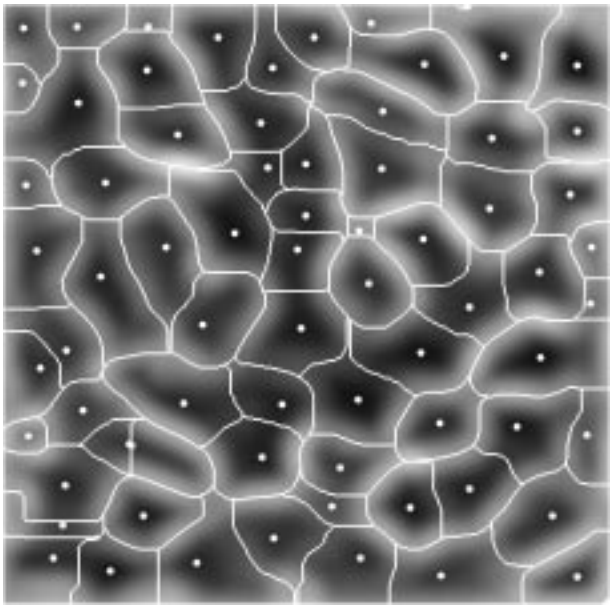


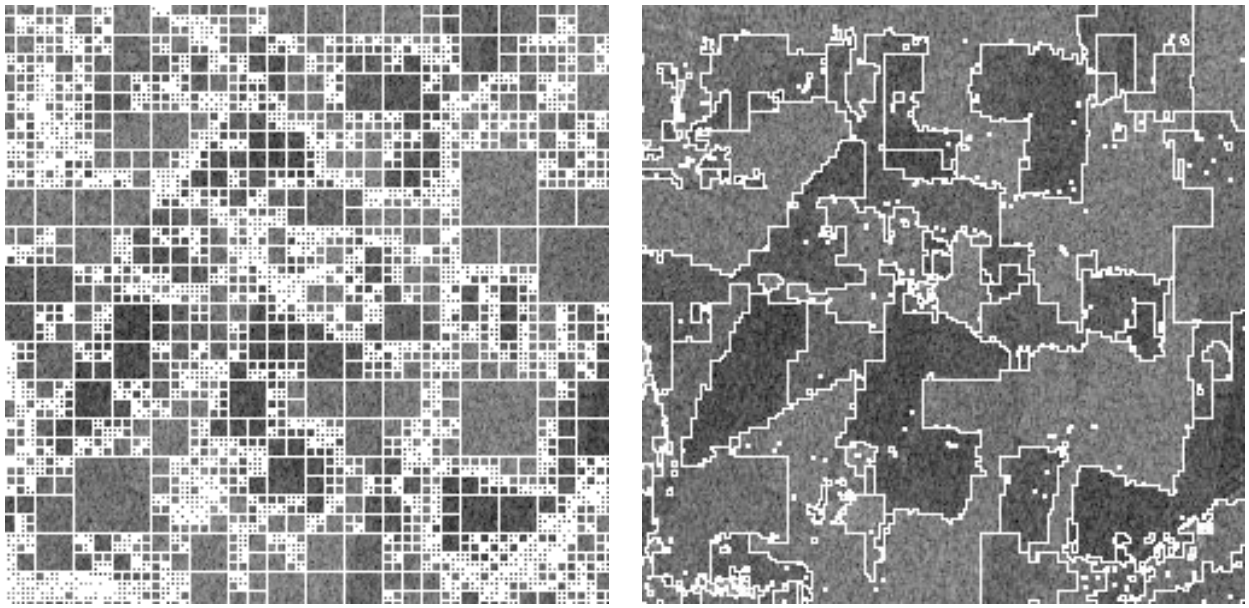
Figure 4.14: Output of variance filter with Gaussian weights ($\sigma^2 = 96$) applied to muscle fibres image, together with seeds indicating all local minima and boundaries produced by watershed algorithm.

There are very many other region-based algorithms, but most of them are quite complicated. In this section we will consider just one more, namely an elegant split-and-merge algorithm proposed by Horowitz and Pavlidis (1976). We will present it in a slightly modified form to segment the log-transformed SAR image (Fig 2.6), basing our segmentation decisions on variances, whereas Horowitz and Pavlidis based theirs on the range of pixel values. The algorithm operates in two stages, and requires a limit to be specified for the maximum variance in pixel values in a region.

The first stage is the *splitting* one. Initially, the variance of the whole image is calculated. If this variance exceeds the specified limit, then the image is subdivided into four quadrants. Similarly, if the variance in any of these four quadrants exceeds the limit it is further subdivided into four. This continues until the whole image consists of a set of squares of varying sizes, all of which have variances below the limit. (Note that the algorithm must be capable of achieving this because at the finest resolution of each square consisting of a single pixel the variances are taken to be zero.)

Fig 4.15(a) shows the resulting boundaries in white, superimposed on the log-transformed SAR image, with the variance limit set at 0.60. Note that:

- Squares are smaller in non-uniform parts of the image.



(a) (b)

Figure 4.15: Region-growing segmentation of log-transformed SAR image: **(a)** division of image into squares with variance less than 0.60, obtained as first step in algorithm, **(b)** final segmentation, after amalgamation of squares, subject to variance limit of 0.60.

- The variance limit was set to 0.60, rather than to the speckle variance value of 0.41 (Horgan, 1994), because in the latter case the resulting regions were very small.

- The algorithm requires the image dimension, n, to be a power of 2. Therefore, the 250×250 SAR image was filled out to 256×256 by adding borders of width 3.

The second stage of the algorithm, the *merging* one, involves amalgamating squares which have a common edge, provided that by so doing the variance of the new region does not exceed the limit. Once all amalgamations have been completed, the result is a segmentation in which every region has a variance less than the set limit. However, although the result of the first stage in the algorithm is unique, that from the second is not — it depends on the order of which squares are considered.

Fig 4.15(b) shows the boundaries produced by the algorithm, superimposed on the SAR image. Dark and light fields appear to have been successfully distinguished between, although the boundaries are rough and retain some of the artefacts of the squares in Fig 4.15(a).

Pavlidis and Liow (1990) proposed overcoming the deficiencies in the boundaries produced by the Horowitz and Pavlidis algorthm by combining the results with those from an edge-based segmentation. Many other ideas for region-based segmentation have been proposed (see, for example, the review of Haralick and Shapiro, 1985), and it is still an active area of research.

One possibility for improving segmentation results is to use an algorithm which over-segments an image, and then apply a rule for amalgamating these regions. This requires '*high-level*' knowledge, which falls into the domain of artificial intelligence. (All that we have considered in this chapter may be termed '*low-level*'.) For applications of these ideas in the area of remote sensing, see Tailor, Cross, Hogg and Mason (1986) and Ton, Sticklen and Jain (1991). It is possible that such domain-specific knowledge could be used to improve the automatic segmentations of the SAR and muscle fibres images, for example by constraining boundaries to be straight in the SAR image and by looking only for convex regions of specified size for the muscle fibres.

We briefly mention some other, more-complex techniques which can be used to segment images.

- The **Hough transform** (see, for example, Leavers, 1992) is a powerful technique for finding straight lines, and other parametrized shapes, in images.
- Boundaries can be constrained to be smooth by employing roughness penalties such as bending energies. The approach of varying a boundary until some such criterion is optimized is known as the fitting of **snakes** (Kass, Witkin and Terzopoulos 1988).
- Models of expected shapes can be represented as **templates** and matched to images. Either the templates can be rigid and the mapping can be flexible (for example, the thin-plate spline of Bookstein, 1989), or the template itself can be flexible, as in the approach of Amit, Grenander and Piccioni (1991).
- Images can be broken down into fundamental shapes, in a way analogous to the decomposition of a sentence into individual words, using **syntactic** methods (Fu, 1974).

4.4 Summary

The key points about image segmentation are:

- Segmentation is the allocation of every pixel in an image to one of a number of categories, which correspond to objects or parts of objects. Commonly, pixels in a single category should:
  - have similar pixel values,
  - form a connected region in the image,
  - be dissimilar to neighbouring pixels in other categories.
- Segmentation algorithms may either be applied to the original images, or after the application of transformations and filters (considered in chapters 2 and 3).
- Three general approaches to segmentation are:

  - thresholding,
  - edge-based methods,
  - region-based methods.

- Methods within each approach may be further divided into those which:

  - require manual intervention, or
  - are fully automatic.

A single threshold, t, operates by allocating pixel (i, j) to category 1 if $f_{ij} \leq t$, and otherwise putting it in category 2. Thresholds may be obtained by:

- manual choice, or
- applying an algorithm such as intermeans or minimum-error to the histogram of pixel values.
  - Intermeans positions t half-way between the means in the two categories.
  - Minimum-error chooses t to minimize the total number of misclassifications on the assumption that pixel values in each category are Normally distributed.
- Thresholding methods may also be applied to multivariate images. In this case, two possibilities are:
  - manually selecting a training set of pixels which are representative of the different categories, and then using linear discrimination,
  - k-means clustering, in which the categories are selected automatically from the data.
- The context of a pixel, that is the values of neighbouring pixels, may also be used to modify the threshold value in the classification process. We considered three methods:
  - restricting the histogram to those pixels which have similarly valued neighbours,
  - post-classification smoothing,
  - using Bayesian image restoration methods, such as the iterated conditional modes (ICM) algorithm.

In edge-based segmentation, all pixels are initially labelled as either being on an edge or not, then non-edge pixels which form connected regions are allocated to the same category. Edge labelling may be:

- manual, by using a computer mouse to control a screen cursor and draw boundary lines between regions,

- automatic, by using an edge-detection filter. Edges can be located either:
  - where output from a filter such as Prewitt's exceeds a threshold, or
  - at zero crossings from a Laplacian-of-Gaussian filter.

Region-based algorithms act by grouping neighbouring pixels which have similar values and splitting groups of pixels which are heterogeneous in value. Three methods were considered:

- Regions may be grown from manually-positioned 'seed' points, for example, by applying a watershed algorithm to output from Prewitt's filter.
- The watershed algorithm may also be run fully automatically, for example, by using local minima from a variance filter as seed points.
- One split-and-merge algorithm finds a partition of an image such that the variance in pixel values within every segment is below a specified threshold, but no two adjacent segments can be amalgamated without violating the threshold.

Finally, the results from automatic segmentation can be improved by:

- using methods of mathematical morphology (chapter 5).
- using domain-specific knowledge, which is beyond the scope of this book

The segmentation results will be used in chapter 6 to extract quantitative information from images.