STAT 718A Shape and Image Analysis

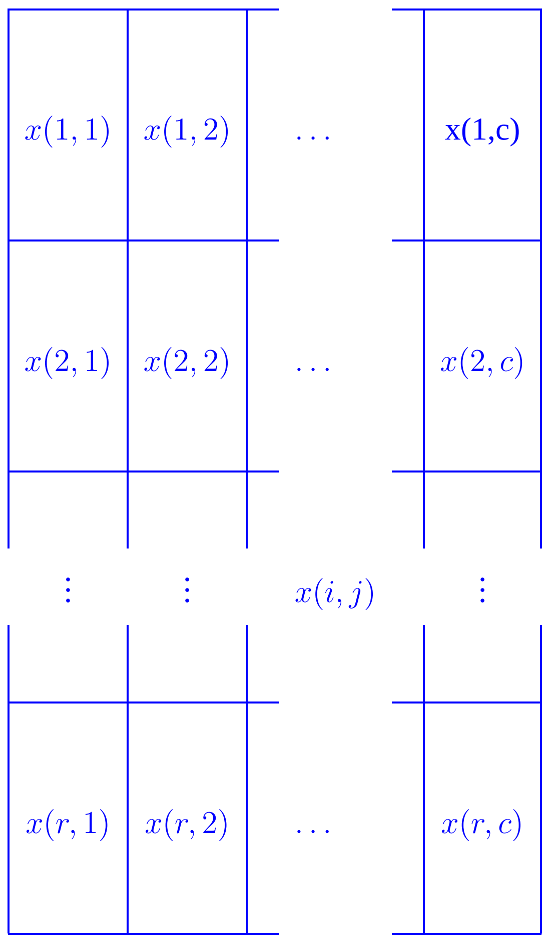
Part II: Statistical Image Analysis

1 INTRODUCTION

1.1 Notation and definitions

An *image* is a two dimensional visual representation of an object (usually 2d or 3d). A *digital image*, $\underline{x}$, is a discrete version of the continuous image, usually achieved by aggregation or sampling. Let $x(i,j)$ be the value of *pixel* (i,j), $i = 1,2,...,r$ and $j = 1,2,...,c$. The term *pixel* is derived from the phrase "picture element".

Clearly, we can think of the image as a large matrix with r rows and c columns, and $x(i,j)$ as the value in row i and column j. Commonly, both r and c are powers of 2, eg $2^4 = 16$.



The pixel values $x(i,j)$ often take an integer value in the range 0 to 255, that is $2^8 = 256$ levels (sometimes this is achieved by scaling), or they might be binary (either 0 or 1). We shall denote the set of possible pixel values by $S = \{0, 1, ..., m\}$. So we can write $0 \leq x(i,j) \leq m$ or $x(i,j) \in S$, and hence $\underline{x} \in S^n$, where $n = r \times c$ is the total number of pixels. When displaying images it is usual, but not universal, to show 0 as "black" and 255 (or 1 for binary) as "white" and other values as various shades of grey.

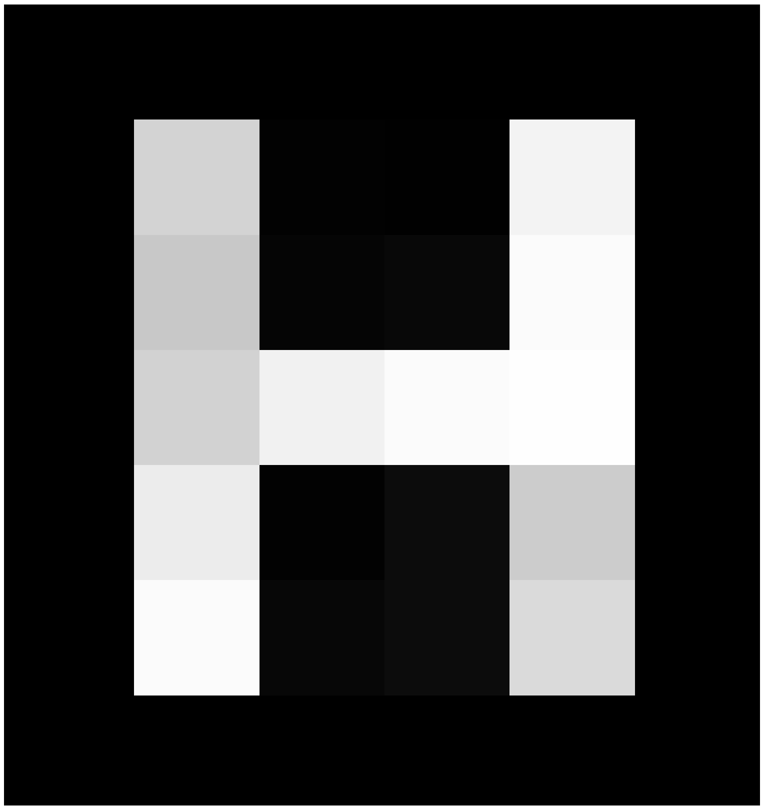


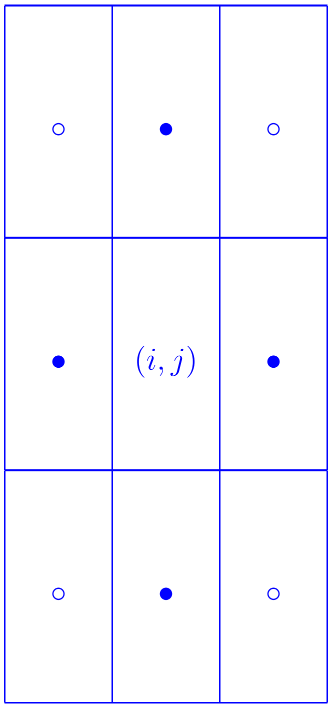
Figure 1: A simple digital image of the letter H.

In Figure 1 we see a simple $(r = 7) \times (c = 6)$ image of the letter 'H' with grey levels in the matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 211 & 2 & 1 & 243 & 0 \\ 0 & 200 & 5 & 8 & 251 & 0 \\ 0 & 210 & 241 & 251 & 254 & 0 \\ 0 & 236 & 2 & 12 & 204 & 0 \\ 0 & 251 & 7 & 12 & 218 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Key ideas used throughout this course are the concepts of *neighbours* and *neighbourhoods*. We shall expect a relationship between pixels which are near to each other, for example the pixel values may be “similar”. More formally, consider the model where pixel (i, j) is a *neighbour* of pixel (i', j') if (i, j) is close to (i', j'). The simplest such case is the *first-order* or *4-connected* neighbourhood system where the neighbours are the two horizontal and two vertical adjacent pixels. The next case is the *second-order* or *8-connected* neighbourhood system, which contains the four diagonal pixels in addition to the first-order neighbours. Higher order neighbourhoods can be defined by obvious extension. Of course, we must specify special conditions for the edges and corners of the image.

Diagrammatically, these two systems can be represented as



So, first-order neighbours are those marked $\bullet$, second-order neighbours are all those marked $\circ$ or $\bullet$. The $\bullet$ are sometimes refered to as edge neighbours, and the $\circ$ as vertex neighbours.

We define

$$\partial_1(i,j) = \{(i-1,j),(i,j-1),(i,j+1),(i+1,j)\}$$

as the first-order *neighbours* of (i, j), and

$$\partial_2(i,j) = \partial_1(i,j) \cup \{(i-1,j-1),(i-1,j+1),(i+1,j-1),(i+1,j+1)\},$$

as the second-order *neighbours* of (i, j).

Further, the *neighbourhood* of pixel (i, j) is denoted $N(i, j)$, the first-order *neighbourhood* of (i, j) is $N_1(i, j) = \partial_1 \cup (i, j)$ and the second-order *neighbourhood* of (i, j) is $N_2(i, j) = \partial_2 \cup (i, j)$.

Note that, “neighbours of” does not include the pixel itself, however, “neighbourhood” does; not surprisingly, these two ideas are often confused.

Often we shall want to refer to the set of pixel values of the neighbours, or of the neighbourhood, so let $\underline{x}_{N(i,j)} = \{x(k,l) \mid (k,l) \in N(i,j)\}$ denote the pixel values of the neighbours of pixel (i, j); other definitions follow similarly.

1.2 Applications

Typically the image represents the level or intensity of some spatially varying quantity. An obvious example is brightness, as in a black and white photograph, but it could be intensity of an X-ray or ultrasound picture in a medical investigation. Also it could be a multivariate image, for example brightness of (Red, Green, Blue) in a colour picture or (ultra-violet, green, infra-red) in satellite images.

Often it is not the pixel values that are of interest, but “objects” in the image. The object is typically made up of points, line segments, arcs, etc. Applications are numerous and new cheap technology is extending possibilities for future exploitation of image analysis techniques.

Current applications include: Classification of land use from remotely sensed data (eg. LANDSAT, METEOSAT); Automatic reading of postcodes; Identification of objects (eg planes, tumours, weeds in crops); Face recognition; Number plate reading; Classification of chromosomes; Image enhancement; Removal of image noise.

2 SUMMARY STATISTICS

2.1 Frequency distributions and histograms

The *pixel-value frequency distribution* gives the frequency of each possible grey value in the image and is defined by

$$f(x_k) = \#\{\,(i,j)\mid x(i,j) = x_k\,\}\,, \qquad k = 0, 1, ..., m$$

where $\#$ is the number in the set and $\sum f(x_k) = rc$. The *image histogram* is a graphical representation of this frequency distribution. Also, the *relative frequency distribution* is defined by

$$p(x_k) = \frac{f(x_k)}{rc}, \qquad k = 0, 1, ..., m$$

with $\sum p(x_k) = 1$. We can think of this as an estimate of the true probability distribution.

As with other applications, we can calculate summary statistics such as mean and variance.

$$\bar{x} = \frac{1}{rc}\sum_{k=0}^{m} f(x_k)\,x_k \qquad s^2 = \frac{1}{rc-1}\sum_{k=0}^{m} f(x_k)\,(x_k-\bar{x})^2 = \frac{1}{rc-1}\left(\sum_{k=0}^{m} f(x_k)x_k^2 - rc\,\bar{x}^2\right)$$

Clearly other measures can be defined, such as skew and kurtosis.

2.2 Histogram normalisation

Histogram normalisation or *contrast enhancement* involves rescaling the grey level values so that the full range of values is present in the image. For example, a very low contrast image might have grey level range 50 to 100. Rescaling them to the full range will result in an image that humans find easier to interpret. Note, however, that there are no more distinct grey levels in the normalised image, but it is easier to interpret.

Contrast enhancement can be expressed as a linear transformation:

$$x' = b\,(x - a)\,,$$

where x' is the transformed pixel value, $a = \min(\underline{x})$ and $b = m/\left(\max(\underline{x}) - \min(\underline{x})\right)$.

2.3 Histogram equalisation

Histogram equalisation involves applying a non-linear transformation to the grey levels. The transformation is chosen so that the new values take the full range and the histogram has an approximately uniform histogram. We replace grey level x by new value x', using:

$$x' = m \sum_{k=0}^{x} p(x_k).$$

The result of applying this transformation can be rather disappointing when the important information in the image is best represented by a few distinct values.

2.4 Co-occurrence

When considering the pixel-value frequency distribution or image histogram all information regarding the spatial arrangement of the pixel values is lost. One way to retain some spatial information is to consider the values of adjacent pixels. The image *co-occurrence* (which is really just a bivariate frequency distribution) gives the frequency of occurrence of each possible pixel value pair, and is defined as

$$f(x_k, x_l) = \#\,\{\,(i,j),(i',j') \mid x(i,j) = x_k, x(i',j') = x_l, (i',j') \in \partial(i,j)\,\}\,, \quad \begin{array}{l} k = 0,1,...,m; \\ l = 0,1,...,m. \end{array}$$

The *relative co-occurrence* is the relative frequency of occurrences,

$$p(x_k, x_l) = \frac{1}{N} f(x_k, x_l), \quad k = 0, 1, ..., m; l = 0, 1, ..., m$$

where N is the total number of pixel pairs, which depends on neighbourhood system and on treatment of edges and corners.

As with the frequency distribution, we can calculate various summary measures based on the co-occurrence, such as the *energy*

$$E = \sum_{k=0}^{m} \sum_{l=0}^{m} p(x_k, x_l)^2$$

or the *entropy* or *information*

$$H = -\sum_{k=0}^{m} \sum_{k=0}^{m} p(x_k, x_l) \log_e p(x_k, x_l).$$

Example

Consider the following image.

2	2	1	1
2	1	1	0
1	1	0	0
1	0	0	0

First calculate the frequency distribution:

x_k	0	1	2
f_k	6	7	3
p_k	6/16	7/16	3/16

Then,

$$\bar{x} = \frac{1}{16}(0 \times 6 + 1 \times 7 + 2 \times 3) = 13/16 \ \ (= 0.8125),$$

and

$$s^2 = \frac{1}{15}\left(0^2 \times 6 + 1^2 \times 7 + 2^2 \times 3 - 16 \times \left(\frac{13}{16}\right)^2\right) = 0.5625$$

therefore $s = 0.75$.

We shall now calculate the co-occurrance matrix using a second-order neighbourhood. For example,

$$\begin{aligned} c(2,2) &= \{(1,1),(1,2);(1,1),(2,1);(1,2),(1,1);(1,2),(2,1);(2,1),(1,1);(2,1),(2,1)\} \\ &= 6 \end{aligned}$$

and hence the relative co-occurrence is $6/84$.

The frequency matrix:

$$\begin{bmatrix} 20 & 11 & 0 \\ 11 & 22 & 7 \\ 0 & 7 & 6 \end{bmatrix}$$

and relative co-occurrence matrix:

$$\begin{bmatrix} 0.24 & 0.13 & 0 \\ 0.13 & 0.27 & 0.08 \\ 0 & 0.08 & 0.07 \end{bmatrix}$$

The resulting energy is $E = 0.182$ and entropy $H = 1.82$.

3 Thresholding

3.1 Introduction

Thresholding is an operation which transforms a grey-level image into a binary image by converting all pixels greater than a particular value, t, to "White" and all those less than or equal to the value to "Black" (or vice versa).

$$x'(i,j) = \begin{cases} 1 & \text{if } x(i,j) \text{ is greater that the threshold},\ x(i,j) > t, \\ 0 & \text{otherwise},\ x(i,j) \leq t \end{cases}$$

where $\underline{x}'$ is the binary output. Of course the threshold t can be chosen manually, perhaps by trial and error, but an automatic procedure is preferable.

A common use of thresholding is to distinguish an object from the background. If the object has generally higher pixel values than the background then it will be shown as "White" on "Black", otherwise as "Black" on "White".

3.2 Notation and definitions

Let D represent the set of pixels in the "Dark" class and B the "Bright" class. Denote the probability function of each class as $\pi_B(x)$ and $\pi_D(x)$. A given image will be composed of bright and dark pixels and so the overall pixel value distribution is a mixture of these

$$\pi(x) = Pr(X(i,j) = x) = Pr(X(i,j) = x, (i,j) \in B) + Pr(X(i,j) = x, (i,j) \in D)$$

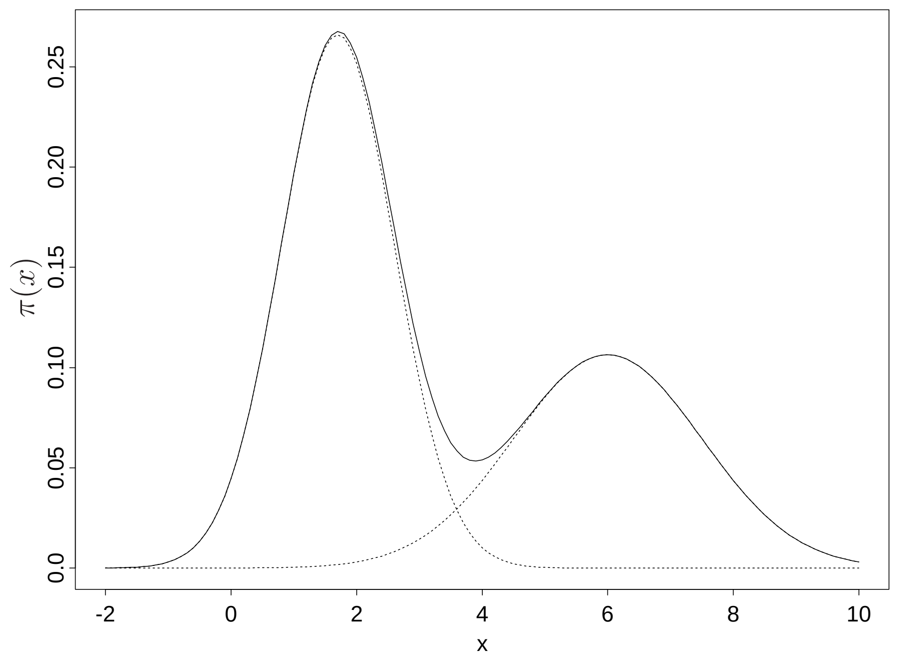
$$= Pr(X(i,j) = x|(i,j) \in B)Pr((i,j) \in B) + Pr(X(i,j) = x|(i,j) \in D)Pr((i,j) \in D)$$

$$= \pi_B(x)\pi_B + \pi_D(x)\pi_D$$

where π_B (π_D) is the (prior) probability of a randomly chosen pixel being from the Bright (Dark) class, and $\pi_B(x)$ ($\pi_D(x)$) the likelihood of a pixel value x given it is from the Bright (Dark) class.

To illustrate this consider the following pixel value distribution. The dotted curve on the left describes the pixel distribution of the Dark pixels ($\pi(x, D) = \pi_D(x)\pi_D$) and on the right, of the Bright pixels ($\pi(x, B) = \pi_B(x)\pi_B$). The solid line gives the combined distribution $\pi(x)$. In this example $\mu_D = 1.7$, $\sigma_D^2 = 0.9$, $\mu_B = 6$, $\sigma_B^2 = 1.5$ and $\pi_D = 0.6$.

Now, with threshold t, the probability of incorrectly labelling a pixel is given



by

$$\begin{aligned}
L &= Pr(X(i,j) \leq t, (i,j) \in B) + Pr(X(i,j) > t, (i,j) \in D) \\
&= Pr(X(i,j) \leq t | (i,j) \in B)\pi_B + Pr(X(i,j) > t | (i,j) \in D)\pi_D \\
&= F_B(t)\pi_B + (1 - F_D(t))\pi_D.
\end{aligned}$$

Unfortunately F_B, F_D, π_B and π_D are not usually known, so minimisation of this is not possible. (See Homework for an example where these are assumed known.)

The image histogram, however, is known. In practice inspection of the histogram is not totally cut-and-dry for making the choice of suitable threshold

t.

We shall now consider methods for automatically choosing the threshold value.

3.3 Minimizing within-group variance

If we suppose that the two classes can be seperated exactly using some choice of threshold, then we can approximate the probability function of the “Dark” pixels by

$$\pi_D(x) \approx Pr(X(i,j) = x | X(i,j) \leq t) = \frac{Pr(X(i,j) = x, X(i,j) \leq t)}{Pr(X(i,j) \leq t)}$$
$$= \begin{cases} \pi(x)/F(t), & x \leq t \\ 0 & x > t \end{cases}$$

and hence estimate it by using sample values:

$$p_D(x) = \frac{p(x)}{F_n(t)}, \qquad \text{for } x \leq t,$$

where $F_n(t) = \sum_{x=0}^{t} p(x)$ is the sample cumulative distribution function. The probability function of the “Bright” pixels is similarly estimated by

$$p_B(x) = \frac{p(x)}{1 - F_n(t)}, \qquad \text{for } x > t.$$

Of course in practice the two distributions will overlap, so some of those designated “Dark” will in fact be “Bright” and vice versa.

Now the sample variance of the “Dark” pixels can be estimated by

$$s_D^2(t) = \sum_{x=0}^{t} (x - \bar{x}_D(t))^2 \, p_D(x)$$

where $\bar{x}_D(t) = \sum_{x=0}^{t} x \; p_D(x)$ is the mean of the “Dark” pixels. Similarly, the variance of the “Bright” pixels is

$$s_B^2(t) = \sum_{x=t+1}^{I} (x - \bar{x}_B(t))^2 \, p_B(x)$$

where $\bar{x}_B(t) = \sum_{x=t+1}^{I} x \; p_B(x)$.

Note that we are effectively dividing the variance sum by n rather than the usual $n-1$; in practice n is so large that the difference does not matter.

Since, in practice, we do not know t one approach is to choose t to minimise

$$s_W^2(t) = F_n(t)s_D^2(t) + (1 - F_n(t))s_B^2(t)$$

which is the within group variance. The minimisation can proceed by evaluating this for all t.

A faster method is to note that the total variance (which does not depend on t) can be written as a sum of the within group variance and the between group variance $s^2 = s_W^2(t) + s_{BG}^2(t)$ (compare this with ANOVA). Since the total variance is fixed, we can therefore equivalently maximise the between group variance, that is

$$s_{BG}^2(t) = F_n(t)(\bar{x}_D(t) - \bar{x})^2 + (1 - F_n(t))(\bar{x}_B(t) - \bar{x})^2.$$

This is equivalent to maximising

$$s_{BG}^2(t) = F_n(t)(1 - F_n(t))(\bar{x}_D(t) - \bar{x}_B(t))^2$$

since $\bar{x} = F_n(t)\bar{x}_D + (1 - F_n(t))\bar{x}_B$.

3.4 Minimising Kullback-Leibler Divergence

Let $f(x)$ be the probability function of a <u>model</u> for the grey level distribution. The Kullback-Leibler divergence, J, is measured by

$$J = \sum_{x=0}^{m} p(x) \log\{p(x)/f(x)\} = \sum_{x=0}^{m} p(x) \log p(x) - \sum_{x=0}^{m} p(x) \log f(x).$$

Note that $J \geq 0$ (Exercise) and $J = 0$ if and only if $p(x) = f(x)$ for all x. We want to choose t to minimize J, however, we need only minimize the *information* measure

$$H = -\sum_{x=0}^{m} p(x) \log f(x).$$

Since m is usually large (typically $m = 255$) the probability distribution of the grey levels can be approximated by a continuous distribution. We assume that the grey levels come from a mixture of normal distributions $N(\mu_D, \sigma_D^2)$, $N(\mu_B, \sigma_B^2)$ with mixing proportions π_D and $\pi_B = 1 - \pi_D$. The probability density function of a mixture of normal distributions is

$$f(x) = \pi_D \left(\frac{1}{\sqrt{2\pi\sigma_D^2}} e^{-\frac{1}{2}\frac{(x-\mu_D)^2}{\sigma_D^2}} \right) + \pi_B \left(\frac{1}{\sqrt{2\pi\sigma_B^2}} e^{-\frac{1}{2}\frac{(x-\mu_B)^2}{\sigma_B^2}} \right) \quad -\infty < x < \infty.$$

If the modes are well separated then, approximately

$$f(x) = \begin{cases} \pi_D \left(\frac{1}{\sqrt{2\pi\sigma_D^2}} e^{-\frac{1}{2}\frac{(x-\mu_D)^2}{\sigma_D^2}} \right), & x \leq t, \\ \\ \pi_B \left(\frac{1}{\sqrt{2\pi\sigma_B^2}} e^{-\frac{1}{2}\frac{(x-\mu_B)^2}{\sigma_B^2}} \right), & x > t \end{cases}$$

and H then simplifies to

$$H \approx \frac{1 + \log 2\pi}{2} - \pi_D \log \pi_D - \pi_B \log \pi_B + \frac{1}{2}(\pi_D \log \sigma_D^2 + \pi_B \log \sigma_B^2).$$

If the parameters μ_D, σ_D, μ_B and σ_B are known this is fairly straightforward, but in practice they must be estimated to give an automatic procedure. Under the assumption that the modes are well separated, we use $p(0), \ldots, p(t)$ to estimate σ_D^2, μ_D and π_D, and $p(t+1), \ldots, p(m)$ to estimate σ_B, μ_B and π_B.

4 NUMERIC FILTERS

4.1 Definitions

We first revisit the ideas of neighbours and neighbourhood systems, in this section we shall require further notation.

Define $\partial_1(i,j)$ as the first-order *neighbours* of (i,j), and $\partial_2(i,j)$, the second-order *neighbours* of (i,j). Further, the *neighbourhood* of pixel (i,j) is denoted $N(i,j)$, the first-order *neighbourhood* of (i,j) is $N_1(i,j) = \partial_1 \cup (i,j)$ and the second-order *neighbourhood* of (i,j) is $N_2(i,j) = \partial_2 \cup (i,j)$.

Note that, “neighbours of” does not include the pixel itself, however, “neighbourhood” does; not surprisingly, these two ideas are often confused.

Often we shall want to refer to the set of pixel values of the neighbours, or the neighbourhood, so let $\underline{x}_{N(i,j)} = \{x(i,j) \mid (i,j) \in N(i,j)\}$ denote the pixel values in the neighbourhood of pixel (i,j); other definitions follow similarly.

A *neighbourhood operator* or *filter* is a, usually simple, operation which is applied to all neighbourhoods of an image. *Numeric filters* are neighbourhood operators which are arithmetic functions of the pixel values, such as addition, averages, maxima, etc.

If $\underline{x}$ denotes the input image and $\underline{x}'$ the output image, then the form of a general neighbourhood operator is:

$$x'(i,j) = \phi\left(\underline{x}_{N(i,j)}\right).$$

The size of the filter is usually odd, say 3, 5 or 7, and hence the neighbourhood has 9, 25 or 49 elements.

Most commonly used numeric filters are examples of the *general linear filter*,

$$x'(i,j) = \sum_{(i',j') \in N(i,j)} w(i-i', j-j')\, x(i',j').$$

The following are a typical set of weights for a second-order neighbourhood, that is a filter of size 3,

1	2	1	
2	4	2	$\times 1/16$
1	2	1	

Note that in this case all weights are positive and $\sum\sum w(k,l) = 1$.

4.2 Mean filters

A *mean filter* is the mean of the pixel values in the neighbourhood. The simplest case is a *simple average* of the pixels, this is also known as the *box filter*. Alternatively, we can use a general weighted average of the neighbourhood pixels

$$x'(i,j) = \sum_{(i',j')\in N(i,j)} w(i-i', j-j')\, x(i',j'), \qquad \sum w = 1.$$

Examples

A 3x3 box filter:

1	1	1	
1	1	1	× 1/9
1	1	1	

A 3x3 weighted mean filter:

0	1	0	
1	4	1	× 1/8
0	1	0	

Note that these definitions apply for “interior” pixels only, special treatment is needed at edges and corners.

Applications: The most common use of the mean filter is to remove noise in an image, however this is achieved at the expense of *blurring* edges.

4.3 Median filter

The *median filter* is the median of the values of the pixels in the neighbourhood.

$$x'(i,j) = median\left(\underline{x}_{N(i,j)}\right).$$

Applications: This filter also smooths out noise, but does not blur the edges to the same extent as the mean filters, see Figure 2.1.

Examples

Consider the following input image

$$\begin{array}{cccc} 4 & 5 & 5 & 7 \\ 5 & 2 & 2 & 1 \\ 2 & 3 & 1 & 1 \\ 5 & 4 & 3 & 1 \end{array}$$

1) A 3x3 box filter with weights:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{9}$$

Consider the output for pixels labelled A, B and C:

B	C		
	A		

4.0	3.2	3.0	2.8
3.5	2.8	2.6	2.2
3.5	2.9	1.9	1.3
3.5	3.0	2.2	1.5

$$A = \begin{pmatrix} 1\times 4 + 1\times 5 + 1\times 5 \\ +1\times 5 + 1\times 2 + 1\times 2 \\ +1\times 2 + 1\times 3 + 1\times 1 \end{pmatrix} \frac{1}{9} = \frac{25}{9} \quad (\approx 2.8)$$

$$B = \begin{pmatrix} 1\times 4 + 1\times 5 \\ +1\times 5 + 1\times 2 \end{pmatrix} \frac{1}{4} = \frac{16}{4} = 4.0$$

$$C = \begin{pmatrix} 1\times 4 + 1\times 5 + 1\times 5 \\ +1\times 5 + 1\times 2 + 1\times 2 \end{pmatrix} \frac{1}{6} = \frac{19}{16} \quad (\approx 3.2)$$

2) A 3x3 weighted average with weights:

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 8 & 2 \\ 1 & 2 & 1 \end{pmatrix} \frac{1}{20}$$

Consider the output for pixels labelled A, B and C:

B	C		
	A		

4.2	3.9	2.8	4.8
4.0	2.6	1.8	1.8
3.0	2.8	1.6	1.1
4.2	3.6	2.5	1.3

$$A = \begin{pmatrix} 1\times 4 + 2\times 5 + 1\times 5 \\ +2\times 5 + 8\times 2 + 2\times 2 \\ +1\times 2 + 2\times 3 + 1\times 1 \end{pmatrix} \frac{1}{20} = \frac{53}{20} \quad (\approx 2.6)$$

$$B = \begin{pmatrix} 8 \times 4 + 2 \times 5 \\ \\ +2 \times 5 + 1 \times 2 \end{pmatrix} \frac{1}{13} = \frac{54}{13} \quad (\approx 4.2)$$

$$C = \begin{pmatrix} 2 \times 4 + 8 \times 5 + 2 \times 5 \\ \\ +1 \times 5 + 2 \times 2 + 1 \times 2 \end{pmatrix} \frac{1}{16} = \frac{62}{16} \quad (\approx 3.9)$$

3) A 3x3 median filter. Consider the output for pixels labelled A, B and C:

B	C		
	A		

4.5	3.0	2.0	1.5
3.5	2.0	2.0	1.0
3.5	3.0	1.0	1.0
3.5	3.0	2.0	1.0

$A = median\{4, 5, 2, 5, 2, 1, 2, 3, 1\} = 2$,

$B = median\{4, 5, 5, 2\} = 4.5$ and

$C = median\{4, 5, 2, 5, 2, 1\} = 3$.

4.4 Quadratic filter

If we assume that over a (small) $m \times n$ neighbourhood the image can be modelled by a quadratic surface

$$f(i, j) = a_1 + a_2 i + a_3 j + a_4 i^2 + a_5 ij + a_6 j^2 + \epsilon(i, j)$$

then we can fit $a_1, ..., a_6$ by least squares. This results in a linear filter with weights

$$w(k,l) = \frac{1}{mn}\left(1 + \frac{5/4(m^2-1) - 15k^2}{m^2-4} + \frac{5/4(n^2-1) - 15l^2}{n^2-4}\right)$$

for $k = -(m-1)/2, ..., (m-1)/2$ and $l = -(n-1)/2, ..., (n-1)/2$. For a 3x3 neighbourhood this gives the weights:

$$\begin{pmatrix} -1 & 2 & -1 \\ 2 & 5 & 2 \\ -1 & 2 & -1 \end{pmatrix}\frac{1}{9}$$

Applications: Again, a smoothing filter.

This filter can be applied, then subtracted from the original image to give an estimate of the noise. We can then apply diagnostic tests to these residuals in a similar manner to usual regression analysis.

4.5 Laplacian filter

The *Laplacian filter* is a weighted mean filter, with special weights ($\sum w = 0$), for a 3x3 neighbourhood these take the following form.

0	1	0
1	-4	1
0	1	0

or

1	1	1	
1	-8	1	$\times$ 1/3
1	1	1	

The Laplacian of a 2d surface, f, is defined by

$$\nabla^2 f = \left(\frac{\partial^2}{\partial i^2} + \frac{\partial^2}{\partial j^2} \right) f = \frac{\partial^2 f}{\partial i^2} + \frac{\partial^2 f}{\partial j^2}.$$

If the values in a neighbourhood of (i, j) can be modelled by a quadratic surface of the form

$$f(i, j) = a_1 + a_2 i + a_3 j + a_4 i^2 + a_5 ij + a_6 j^2$$

then the neighbourhood looks like

$a_1 - a_2 - a_3 + a_4 + a_5 + a_6$	$a_1 - a_2 + a_4$	$a_1 - a_2 + a_3 + a_4 - a_5 + a_6$
$a_1 - a_3 + a_6$	a_1	$a_1 + a_3 + a_6$
$a_1 + a_2 - a_3 + a_4 - a_5 + a_6$	$a_1 + a_2 + a_4$	$a_1 + a_2 + a_3 + a_4 + a_5 + a_6$

and $\nabla^2 f = 2a_4 + 2a_6$, and this is also produced by either of the above filters.

So these filters produce a value which depends on the second derivative. Its effect is to return 0 when the image is flat or linear, and non-zero when a jump is present, hence, the filter is essentially an *edge detector*. There are many other filters for edge detection, including the Roberts' Operator and the Sobel Operator.

4.6 Template matching

This is another special case of the general weighted average numeric filter, with weights chosen to match the application.

For example, if we wish to find the letter "H" in an image (perhaps of a post-code or car registration plate) weights with an "H" pattern are used, this set of weights is called the *template*.

1	0	1
1	1	1
1	0	1

High values in the filtered output image indicate possible locations of the template. Of course the template could be more complicated, and could be part of a grey-level image. A major drawback is that seperate templates may be need for different magnifications and rotations.

5 MORPHOLOGICAL OPERATORS

Symbolic filters are operators which are defined in terms of Boolean or logical operation, such as AND, OR or NOT.

Morphological operators are a special type of symbolic filter which generally extract information on *shape*, that is properties of objects after translation, scale and rotation have been taken into account.

5.1 Region-growing filter

Suppose that the image is composed of labelled regions or classes. Let the set of labels be denoted $L = \{1, 2, ..., k\}$, hence $x \in L^n$ where n is the number of pixels. We can grow region c say ($c \in L$) by using the simple symbolic filter which gives

$$x'(i,j) = \begin{cases} c & \text{if } \# \left(x_{\partial(i,j)} \cap c \right) > 0 \\ x(i,j) & \text{otherwise}. \end{cases}$$

That is we change the label to c if <u>any</u> of the neighbours have label c.

A more sophisticated version can grow a region only if a majority of neighbours take the same label, that is

$$x'(i,j) = \begin{cases} c & \text{if } \#\left(x_{\partial(i,j)} \cap c\right) > \#\left(x_{\partial(i,j)} \cap c'\right) \\ \\ x(i,j) & \text{otherwise} \end{cases}$$

For binary images

For a binary image ($x(i,j) = 0$ or 1), the first of these region-growing filters is just

$$x'(i,j) = \max\{x_{\partial(i,j)}\}$$

and a region-shrinking filter is given by

$$x'(i,j) = \min\{x_{\partial(i,j)}\}.$$

5.2 Binary dilation

This is a transformation, called *Minkowski addition,* which combines two sets using vector addition of set elements. If A and B are subsets of I^d (the d-dimensional regular lattice) with (vector) elements a and b, then the *dilation* of A by B is the set of all vector sums of pairs of elements; one from A and the other from B, that is the dilation of A by B is

$$A \oplus B = \{c \mid c = a + b \text{ for some } a \in A \text{ and } b \in B\}$$

where I^d denotes the d-dimensional lattice with integer labels. Note that since vector addition is symmetric so is dilation, hence $A \oplus B = B \oplus A$.

In practice A is associated with the image and B is refered to as the *structuring element*. For example, with structuring element $B = \{(i, j) \mid i = -1, 0, 1; j = -1, 0, 1\}$ we get the region-growing operator introduced earlier.

Translation operator

Let A be a subset of I^d ($A \subset I^d$) and t a point in I^d ($t \in I^d$). We denote the *translation* of A by the point t by A_t, with

$$A_t = \{c \mid c = a + t \text{ for some } a \in A\}\,.$$

Then dilation is just the union of translations,

$$A \oplus B = \cup_{a \in A} B_a = \cup_{b \in B} A_b.$$

Set identities

Having introduced the new set operator, $\oplus$, we can consider various set results, proof of the following are left as exercises.

- $(A \oplus B) \oplus C = A \oplus (B \oplus C)$,
- $A \oplus B_t = (A \oplus B)_t$,
- $(B \cup C) \oplus A = (B \oplus A) \cup (C \oplus A)$,
- $A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$.

Also, if $A \subset B$, then $A \oplus K \subset B \oplus K$, so dilation is an *increasing* operator.

5.3 Binary erosion

This is the dual or complement of dilation. If A and B are sets, then the erosion of A by B is the set of elements x for which $x + b \in A$ for every $b \in B$, that is the erosion of A by B is defined by

$$A \ominus B = \{x \mid x + b \in A \text{ for every } b \in B\}.$$

An alternative representation is

$$A \ominus B = \{x \mid B_x \subset A\}\,.$$

Thus B may be visualised as a probe that slides across the image A, testing the spatial nature of A at every point, where B translated by x can be contained in A (by placing the origin of B at x), then x belongs to the erosion $A \ominus B$.

A further representation is

$$A \ominus B = \cap_{b \in B} A_{-b}$$

that is an intersection of negative translations.

Erosion-Dilation Relation

Recall DeMorgan's Law, that $(A \cup B)^c = A^c \cap B^c$, the corresponding rule for erosion and dilation is

$$(A \ominus B)^c = A^c \oplus \check{B}$$

where $\check{B}$ is the *reflection* of B defined by

$$\check{B} = \{x \mid \text{ for some } b \in B, x = -b\}\,.$$

Proof

$x \in (A \ominus B)^c$ if and only if $x \notin A \ominus B$.

$x \notin A \ominus B$ if and only if there exists $b \in B$ such that $x + b \notin A$.

There exists $b \in B$ such that $x + b \in A^c$ if and only if there exists $b \in B$ such that $x \in (A^c)_{-b}$.

There exists $b \in B$ such that $x \in (A^c)_{-b}$ if and only if $x \in \cup_{b \in B}(A^c)_{-b}$.

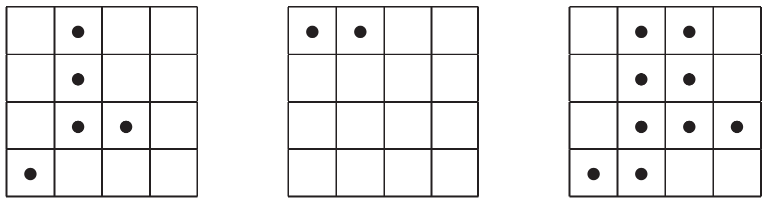
Now $x \in \cup_{b \in B}(A^c)_{-b}$ if and only if $x \in \cup_{b \in \check{B}}(A^c)_b$: and $x \in \cup_{b \in \check{B}}(A^c)_b$ if and only if $x \in A^c \oplus \check{B}$.

Corollary $(A \oplus B)^c = A^c \ominus \check{B}$. (Exercise)

Example 1 - Binary dilation

If $A = \{(0,1),(1,1),(2,1),(2,2),(3,0)\}$ and $B = \{(0,0),(0,1)\}$, then find $A \oplus B$.

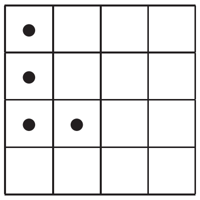
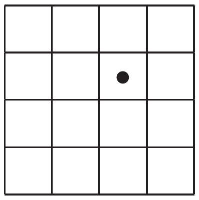
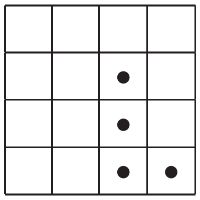
$A \quad \oplus \quad B \quad = \quad A \oplus B$



Example 2 - Translation operator

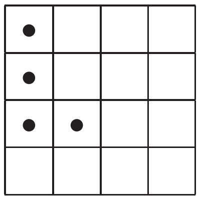
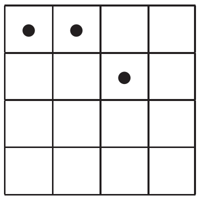
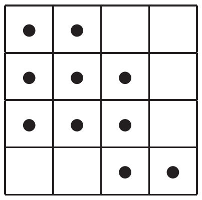
a) If $A = \{(0,0),(1,0),(2,0),(2,1)\}$ and $t = \{(1,2)\}$, then find A_t.

$A \qquad t \qquad A_t$

b) If $A = \{(0,0),(1,0),(2,0),(2,1)\}$ and $B = \{(0,0),(0,1),(1,2)\}$, then find $\cup_{b\in B} A_b$.

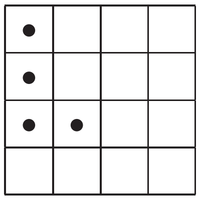
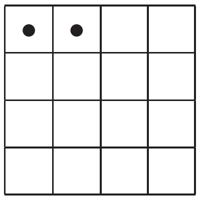
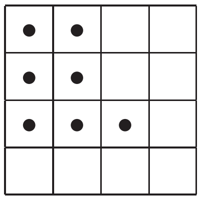
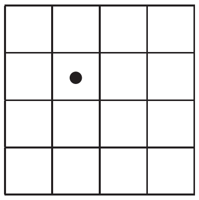
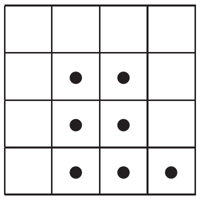
A $\quad$ B $\quad$ $\cup_{b\in B} A_b$

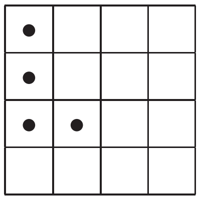
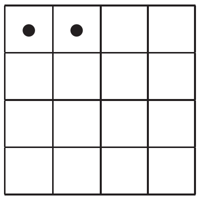
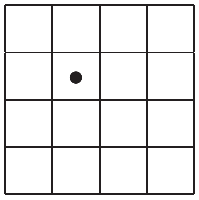
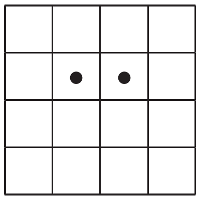
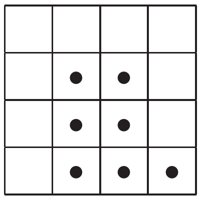
Example 3 - Set identities

If $A = \{(0,0),(1,0),(2,0),(2,1)\}$, $B = \{(0,0),(0,1)\}$ and $C = \{(1,1)\}$, then illustrate the identity $(A \oplus B) \oplus C = A \oplus (B \oplus C)$.

A $\quad$ B $\quad$ $A \oplus B$ $\quad$ C $\quad$ $(A \oplus B) \oplus C$

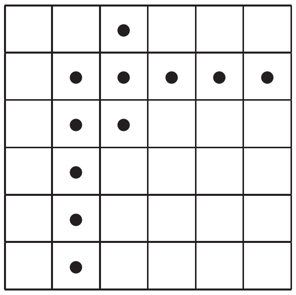
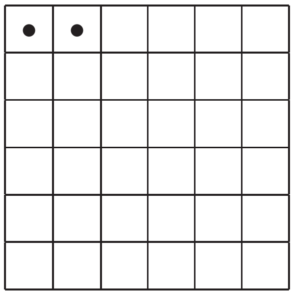
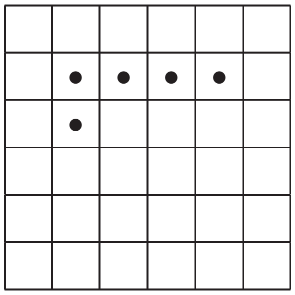
A $\quad$ B $\quad$ C $\quad$ $B \oplus C$ $\quad$ $A \oplus (B \oplus C)$

Example 4 - Binary erosion

If $A = \{(0,2),(1,1),(1,2),(1,3),(1,4),(1,5),(2,1),(2,2),(3,1),(4,1),(5,1)\}$ and $B = \{(0,0),(0,1)\}$, then find $A \ominus B$.

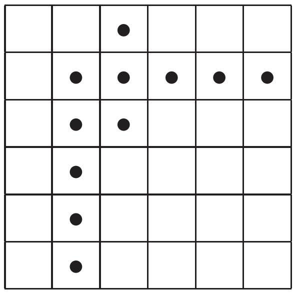
A $\quad$ B $\quad$ $A \ominus B$

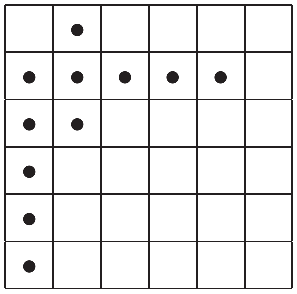
Example 5 - Dilation-Translation

For A and B as above, find $A_{(0,0)}$, $A_{-(0,1)}$ and hence $A_{(0,0)} \cap A_{-(0,1)}$ and compare this with $A \ominus B$.

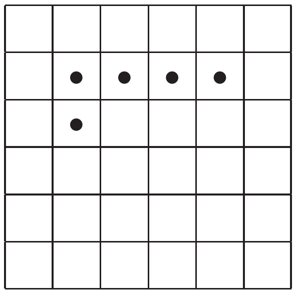
$A_{(0,0)}$



$A_{-(0,1)}$

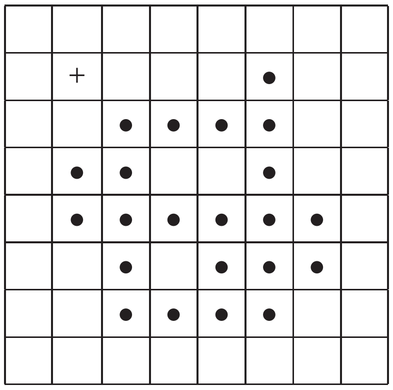


$A_{(0,0)} \cap A_{-(0,1)}$



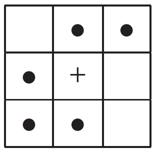
Example 6 - Erosion-Dilation Relation

A



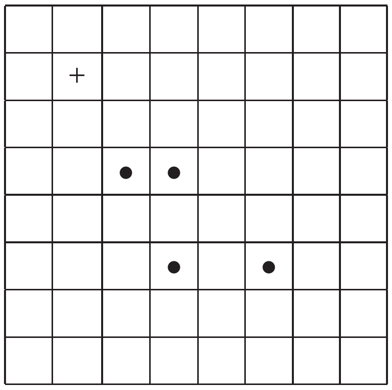
A^c

B

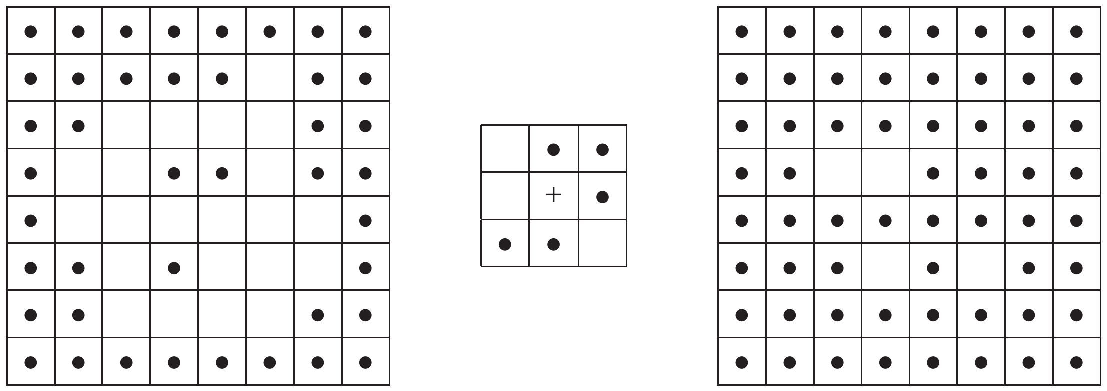


$\check{B}$

$A \ominus B$



$A^c \oplus \check{B}$



5.4 Binary opening and closing

The *opening* of image A by structuring element K is denoted $A \circ K$ and is defined by

$$A \circ K = (A \ominus K) \oplus K$$

that is erosion followed by dilation. The opening characterisation theorem states that

$$A \circ K = \{x \mid \text{ for some } t \in A \ominus K, x \in K_t\}$$

that is

$$A \circ K = \cup_{t \in A \ominus K} K_t.$$

So the points in $A \circ K$ are precisely those obtained by sweeping the structuring element over the inside of A, never permitting any point of the structuring element to be outside A. The set of all points covered by the sweep is the opening of A by K.

Applications: The opening of an image by a small disk structuring element smooths the perimeter of an object and eliminates small “islands”.

The *closing* of A by K is denoted $A \bullet K$ and defined by

$$A \bullet K = (A \oplus K) \ominus K$$

that is dilation followed by erosion.

Applications: If the image contains an ideal object, A, corrupted by noise so that holes are created inside the object, then it can be shown that the ideal object can be recovered without error by the closing of the observed image, B, with a carefully chosen structuring element, K, that is $B \bullet K = A$.

5.5 Grey-scale Morphology

The binary operations considered so far are readily extended to grey-scale images, however, we start with some basic definitions.

Let A be a subset of I^d and $F = \{x | x \in I^{d-1}, y \in I, (x, y) \in A\}$, for example in $n = 2$, $A = \{(0, 2), (1, 2), (1, 3), (2, 1), (2, 2)\}$, then $F = \{0, 1, 2\}$ (or $F' = \{1, 2, 3\}$). In $d = 3$ consider a rectangular box A and F its projection on to a two dimensional plane.

in general, we can define F in n ways, each time projecting A onto a different axis. In image analysis, however, only one such projection will make sense, more later.

Top and Umbra

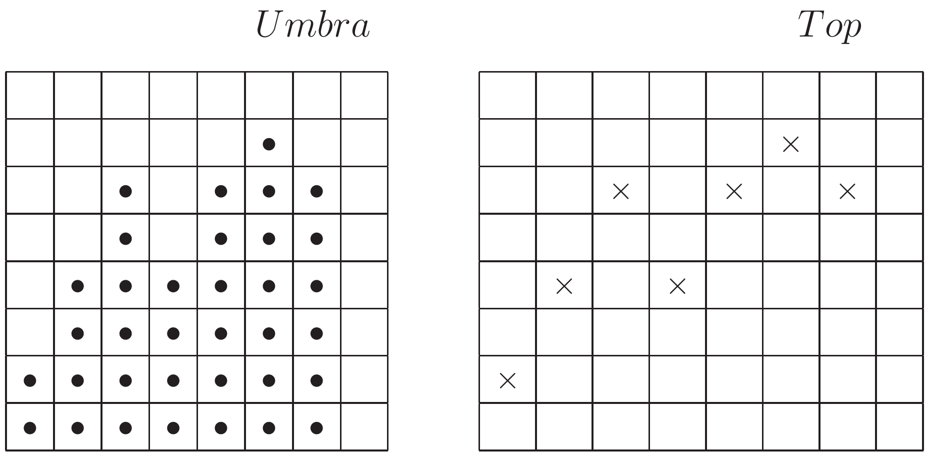
The *top* or *top surface* of A is defined as

$$T(A) = \{z | z = \max\{y \mid (x, y) \in A\}, x \in F\}.$$

The *umbra* of f (any integer-valued function defined on some subset F of I^{n-1} is a set consisting of the surface f and everything below the surface, which is defined by

$$U(f) = \{(x, y) | y \leq f(x), x \in F, y \in I\}.$$

Example



The set f denoted $\times$, and its umbra, $U(f)$, denoted $\bullet$.

The grey-scale dilation of f by structuring element k is defined by

$$f \oplus k = T(U(f) \oplus U(k)\}$$

and the grey-scale erosion of f by k by

$$f \ominus k = T(U(f) \ominus U(k)\}.$$

The grey-scale *opening* of f by structuring element k is defined by

$$B \circ K = (B \ominus K) \oplus K$$

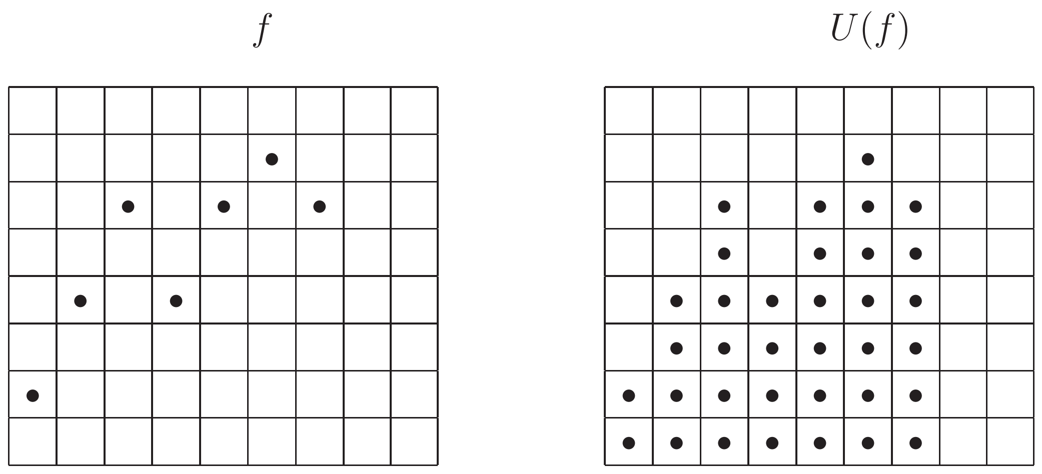
and the grey-scale *closing* of f by K by

$$B \bullet K = (B \oplus K) \ominus K.$$

The grey-scale opening of f by k can be visualised by sliding k under f. The set of all the highest points reached by some part of k is the opening.

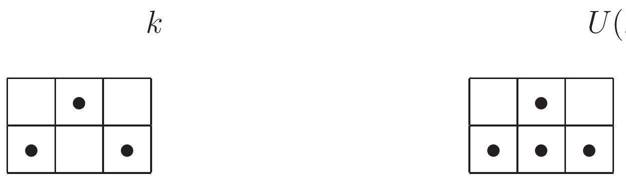
Closing can be thought of by sliding the reflection of k over the top of f. The set of all the lowest points reached by some part of k is the closing of f by k.

A function and its umbra



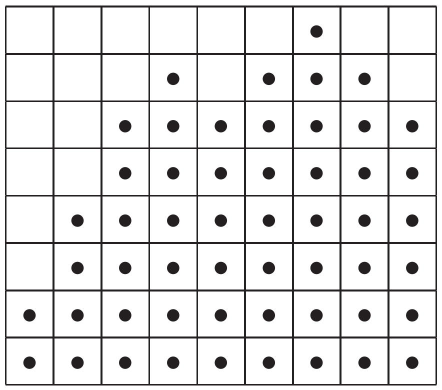
A small structuring element and its umbra

(origin at bottom left of structuring element)

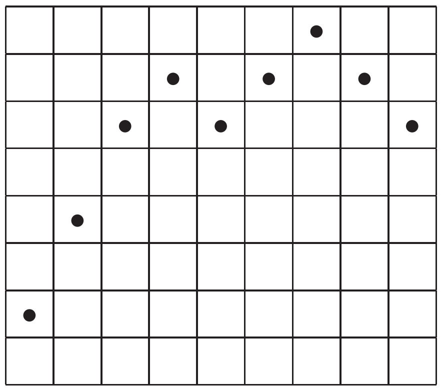


Grey-scale dilation

$U(f) \oplus U(k)$

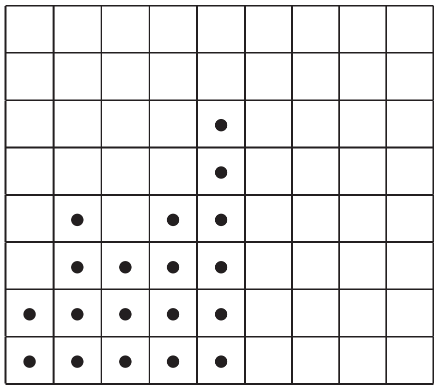


$T(U(f) \oplus U(k)) = f \oplus k$

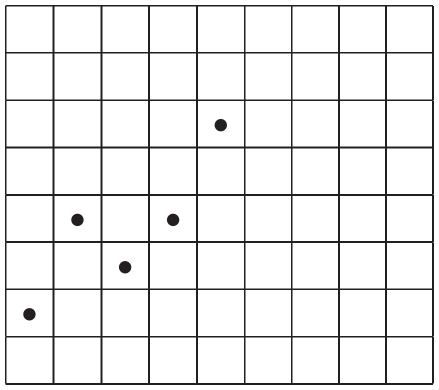


Grey-scale erosion

$U(f) \ominus U(k)$



$T(U(f) \ominus U(k)) = f \ominus k$



6 IMAGE MODELS AND ESTIMATION

6.1 Introduction

In this chapter we shall, for ease of notation, label the pixels $1, 2, \ldots, n$ where $n = r \times c$, and denote a pixel value x_i for pixel i, so $\underline{x} = \{x_1, x_2, \ldots, x_n\}$. Each pixel variable can be discrete or continuous. We refer to the possible values of the pixel variable as intensities. An arbitrary shading will be denoted by $\underline{x}$.

We need a probabilty model or distribution over the possible true signals which reflects our (prior) ideas of the scene. For computational reasons it is highly desirable to have a local description of the probability model, this can be achieved using models defined in terms of Markov random fields which are 2-d models closely related to Markov chains.

6.2 Markov random fields (MRF)

If we want to model images, we need a random process defined not just over a single variable, time, but over 2-dimensional space, this leads to the idea of a *Markov random field*. When considering space, there is no obvious ordering of the axes, unlike time, so we replace the idea of the *recent past* by *neighbours*, and *distant past* by *not neighbours*.

A *Markov random field* is a joint probability distribution with probability function (or p.d.f. if continuous) $\pi(\underline{x})$, on the set of all possible shadings of $\underline{x}$, which satisfies the Markov property,

$$\pi(x_i \mid x_j, j \neq i) = \pi(x_i \mid \underline{x}_{\partial(i)})$$

that is the conditional distribution of x_i depends only on the values of the neighbouring pixels.

Although it is possible to construct Mrfs from first principles, it is tricky to check that the model is valid. A more usual approach is to choose a particular form of the Gibbs distribution. It can be proven that every Mrf corresponds to a unique Gibbs distribution, and vice versa.

Gibbs distributions

A *Gibbs distribution* is defined as

$$\pi(\underline{x}) = \frac{1}{Z(\beta)} \exp\left\{-\beta \sum_i \sum_j \phi(x_i - x_j)\right\} = \frac{1}{Z(\beta)} \exp\left\{-\beta V(\underline{x})\right\}$$

where the sum is over all pairs of neighbouring pixels, $V(\underline{x})$ is known as the energy, and ϕ as the potential. This terminology indicates the origins of this model as coming from statistical physics. In most cases the normalising constant $Z(\beta)$ is intractable.

The parameter β controls the variation between neighbouring pixel values. Note that $\beta \uparrow \infty$ implies constant x, and small β correspondingly small spatially structured variation, only as $\beta \downarrow 0$ does the variation become spatially unstructured.

An alternative specification is in terms of the local conditional distributions

$$\pi(x_i \mid \underline{x}_{\partial(i)}) = \frac{1}{Z^*(\beta)} \exp\left\{-\beta \sum_{j \in \partial(i)} \phi(x_i - x_j)\right\}$$

where $Z^*(\beta)$ is not the same as $Z(\beta)$.

The most widely used Gibbs distribution has a quadratic potential ($\phi(u) = u^2$) and hence is a multivariate Normal distribution, $X_i|\underline{x}_{\partial(i)} \sim N\left(\bar{\underline{x}}_{\partial(i)}, (4\beta)^{-1}\right)$ for a first-order neighbourhood.

Strauss model (or Potts model)

This is a model for unordered categories, such as labels on a map. Here, the energy is the number of pairs of neighbouring pixels with dissimilar labels. Thus the most probably maps are all of one label, and the least probable ones (for 4-neighbours) are patchwork quilts like a chess-board (if $\beta > 0$). This is Markov since

$$P\left(X_i = x \middle| \text{ rest of } X\text{'s}\right) \propto \exp\left[\beta \# \text{ (neighbours of label } k)\right].$$

Ising model

This is a special case of the Strauss distribution with only two categories and $\phi(u) = 1$ if $u = \pm 1$; 0 if $u = 0$, hence the local conditional probability function is

$$\pi\left(x_i = x \mid \underline{x}_{\partial(i)}\right) = \frac{\exp\{-\beta\#\{x_j \mid x_j = 1 - x, j \in \partial(i)\}}{\exp\{-\beta\#\{x_j \mid x_j = 0, j \in \partial(i)\} + \exp\{-\beta\#\{x_j \mid x_j = 1, j \in \partial(i)\}}$$

where $x = 0, 1$. For this simple model we can write down the joint probability function

$$\pi(\underline{x}) = \frac{1}{Z(\beta)} \exp\{-\beta \#\{\text{ discordant pairs }\}\}$$

though the form of $Z(\beta)$ is very complicated.

Example (4 neighbour Ising model). This is the Strauss model with only 2 possible labels and a 4-neighbour graph. In this case

$$P(x_{ij} = x | \text{ neighbours}) = \frac{e^{xT}}{1 + e^T} \qquad x = 0, 1 \tag{1}$$

where $T = 2\beta \sum_{i'j' \text{ neighbours}} X_{i',j'} - 4\beta$.

We can write this as

$$P(X_{ij} = 1 | \text{neighbours}) = \frac{e^{-\beta n_0}}{e^{-\beta n_1} + e^{-\beta n_0}}$$
$$P(X_{ij} = 0 | \text{neighbours}) = \frac{e^{-\beta n_1}}{e^{-\beta n_1} + e^{-\beta n_0}}.$$

where $n_0 = \#$ neighbours of pixel i with value 0

$n_1 = \#$ neighbours of pixel i with value 1.

(A pixel is not considered a neighbour of itself).

If $\beta > 0$ pixels will tend to be more like neighbours

$\beta < 0$ pixels will tend to be less like neighbours (chessboard, spotty).

The joint distribution of X is

$$P\left(X_{11}=x_{11},\ X_{12}=x_{12},\ldots,X_{rc}=x_{rc}\right)=\frac{1}{Z}\exp\left(-\beta\ \textbf{(\# discordant pairs of neighbours)}\right)$$

where Z is a very complicated constant to ensure $\sum P\left(X\right)=1$.

Proof that the Ising Model satisfies the Markov property

Denote $\{X_{lm}=x_{lm},(l,m)\neq(i,j)\}$ as "rest" for all pixels $\neq(i,j)$.

$$P\left(X_{ij}=1|\text{ rest }\right)=\frac{P\left(X_{ij}=1\cap\text{ rest }\right)}{P\left(\text{ rest }\right)}$$

and

$$\begin{aligned}P(\text{rest}) &= P\left(X_{ij}=1\cap\text{ rest }\right)+P\left(X_{ij}=0\cap\text{ rest }\right)\\ &= \frac{1}{Z}\exp\left(-\beta\left(n_0+d\right)\right)+\frac{1}{Z}\exp\left(-\beta\left(n_1+d\right)\right)\end{aligned}$$

where $d=\#$ of discordant pairs of neighbours not involving (i,j).Hence,

$$P(X_{ij}=1|\text{ rest })=\frac{\frac{1}{Z}e^{-\beta(n_0+d)}}{\frac{1}{Z}e^{-\beta(n_0+d)}+\frac{1}{Z}e^{-\beta(n_1+d)}}=\frac{e^{-\beta n_0}}{e^{-\beta n_0}+e^{-\beta n_1}}.$$

Also, $P(X_{ij}=0|\text{ rest })=1-P(X_{ij}=1|\text{ rest })=\frac{e^{-\beta n_1}}{e^{-\beta n_0}+e^{-\beta n_1}}$. So the conditional distribution of X_{ij} given the rest only depends on the neighbours of X_{ij}, i.e.

$$P\left(X_{ij}=x_{ij}|\text{ rest }\right)=P\left(X_{ij}=x_{ij}|\text{ neighbours }\right)$$

So the Markov Property is satisfied.

The Ising model has a critical value of $\beta_{crit} = \sinh^{-1} 1 \approx .88$ such that, as $rc \to \infty$, for $\beta < \beta_{crit}$ there are no infinite patches of one colour, whereas, for $\beta > \beta_{crit}$ there will always be such infinite patches.

Auto-models

Another class of model can be defined by specifying the the form of the conditional distribution. For example suppose that the conditional distribution of X_i has a binomial distribution with index parameter c_i and probability π_i, which is dependent on the values of the neighbours; the conditional probability is then

$$p(x_i|\underline{x}_{\partial(i)}) = \binom{c_i}{x_i} \pi_i^{x_i}(1-\pi_i)^{c_i - x_i} \quad x_i = 0, 1, \ldots, c_i \;\text{ where }\; \pi_i = \frac{\exp\{a + b\sum x_j\}}{1 + \exp\{a + b\sum x_j\}}.$$

The resultant joint probability distribution, $\pi(\underline{x})$, is *auto-binomial*. The Ising model is a special case of the auto-binomial with index parameter $c_i = 1$ and constraint $a = -2b$.

Clearly, other auto models can be defined simply by specifying the form of the conditional distribution: auto-Poisson, $\lambda_i = \exp\{a + b\sum x_j\}$, auto-normal, $\mu_i = a + b \sum x_j$. The relationships between model parameter and function of neighbours ($a + b \sum x_j$) is clear when these models are written in regular exponential family form. It can be shown that $a + b \sum x_j$ must be equal to the natural parameter of the distribution.

Isotropy versus Anisotropy

We can further generalise by allowing different parameters for different directions, that is replace b (or β) by $b(1,1)$ and $b(1,2)$ for the horizontal and vertical neighbours, and $b(2,1)$ and $b(2,2)$ for the diagonal neighbours. If there is a common b parameter, the model is *isotropic* (same in all directions) otherwise it is *anisotropic*.

6.3 Simulation of MRFs

From the definitions alone it is rather difficult to get a feel for the properties of these models. One way is to simulate samples from the distribution and study properties of the samples. Standard methods of simulation, however, are not appropriate due to the very high dimension of the sample space, for a binary image with $r = c = 256$ there are more than $10^{140\times140}$ elements in the sample space (i.e. huge!). When the model is a Gibbs distribution, we can use the Gibbs Sampler (introduced in 1984 by Geman and Geman).

The Gibbs Sampler is a simple iterative algorithm which produces a Markov chain with the required distribution as its limiting or equilibrium distribution. The algorithm is started from an arbitrary starting position, then after an initial transient period subsequent values are collected. These realisations of the Markov chain form a *pseudo*-sample from the required distribution and can be used to estimate various statistical measures of the images.

At each iteration, only one pixel changes in value: the sequence in which the pixels are visitied is arbitrary, but it is usual to visit them in the labelled order. The sampling algorithm is defined as follows:

- Pick an arbitrary starting vector, $\underline{x}^0 = (x_1^0, x_2^0, ..., x_n^0)$.
- Simulate a new value for each pixel from the following conditional distributions:

  x_1^1 from $\pi(x_1|x_2^0, x_3^0, ..., x_n^0)$

  x_2^1 from $\pi(x_2|x_1^1, x_3^0, ..., x_n^0)$

  $\vdots$

  x_i^1 from $\pi(x_i|x_1^1, x_2^1, ..., x_{i-1}^1, x_{i+1}^0, ..., x_n^0)$

  $\vdots$

  x_n^1 from $\pi(x_n|x_1^1, x_2^1, ..., x_{n-1}^1)$

  This completes the transition from $\underline{x}^0$ to $\underline{x}^1$.
- For the t^{th} iterative sweep obtain a new intensity for pixel i by simulating x_i^{t+1} from the conditional distribution, $\pi(x_i|x_1^{t+1}, x_2^{t+1}, ..., x_{i-1}^{t+1}, x_{i+1}^t, ..., x_n^t)$.

After an initial transient period all subsequent sweeps give realisations from the required probability distribution. It is important to monitor convergence to the equilibrium distribution and discard the realisations obtained before equilibrium is reached. This is often achieved using various summary statistics of the evolving chain. Also, we must determine a suitable sample size; this can be achieved by making modifications to the usual approach to sample size calculations to make allowances for the correlation between sample members.

6.4 Parameter Estimation

Suppose that we have a single observation from the auto-binomial model, then how can we make inferences about parameters a and b. The obvious suggestion is to use maximum likelihood, that is maximise the joint probability $\pi(\underline{x}|a,b)$ of the data given the parameters. We have noted, however, that the normalising constant in this distribution can not be easily evalued, but we do know the local conditional distributions, $\pi(x_i|\underline{x}_{\partial_i},a,b)$. If the x_i were independent then the joint distribution would be given by the product of the local distribution, but in image applications they are not independent. Instead we can use conditional likelihood procedures based on the coding method due to Besag (1972, 1974).

Suppose we wish to consider a Markov random field with a first-order neighbourhood system. We start by labelling the sites with $\bullet$ and $\times$ alternatively.

$$\begin{array}{cccccccc}
\bullet & \times & \bullet & \times & \bullet & \times & \bullet & \times \\
\times & \bullet & \times & \bullet & \times & \bullet & \times & \bullet \\
\bullet & \times & \bullet & \times & \bullet & \times & \bullet & \times \\
\times & \bullet & \times & \bullet & \times & \bullet & \times & \bullet
\end{array}$$

Coding pattern for a first-order neighbourhood.

Now the variables associated with the sites $\times$ given the observed values of the other sites are mutually independent. Hence, the conditional likelihood, l of the pixels in one coding set given the values of the pixels in the second coding set can be obtained as the product of the conditional distributions of the corresponding pixels,

$$l = \prod \pi(x_i|\underline{x}_{\partial_i})$$

the product is over the pixels in a coding set and the log-likelihood,

$$L = \sum \log\{\pi(x(i,j)|x_{\partial_i})\}$$

where the sum is over pixels in the coding set. For the auto-binomial this be-comes

$$L = \sum \Bigg(\log \binom{c}{x_i} + x_i \log \pi_i - (c - x_i) \log(1 - \pi_i) \Bigg)$$

where, for a first-order anisotropic scheme, π_i is given by

$$\pi_i = \frac{\exp\{a + b_1(u + u') + b_2(v + v')\}}{1 + exp\{a + b_1(u + u') + b_2(v + v')\}}$$

where u and u' are the horizontal neighbours, and v and v' the vertical neighbours. The resulting normal equations are,

$$\begin{aligned} \sum x_i &= \sum c\, \hat{\pi}_i \\ \sum x_i(u + u') &= \sum (u + u') c\, \hat{\pi}_i \\ \sum x_i(v + v') &= \sum (v + v') c\, \hat{\pi}_i \end{aligned}$$

where

$$\hat{\pi}_i = \frac{\exp\left\{\hat{a} + \hat{b}_1(u + u') + \hat{b}_2(v + v')\right\}}{1 + \exp\left\{\hat{a} + \hat{b}_1(u + u') + \hat{b}_2(v + v')\right\}}.$$

Clearly there are similar equation for other neighbourhood schemes, such as first-order isotropic.

6.5 Bayesian Image Analysis: segmentation and posterior estimation

The simplest type of segmentation is thresholding which allocates a pixel (i, j) to class A_k if $x_{ij} \in [t_k, t_{k+1})$ where $t_1, t_2, \dots t_{q+1}$ are threshold levels. This approach has been discussed in §2. It is based on a pixel-by-pixel consideration, whereby observations $x_{i,j}$ are considered independently. We can achieve better results by using the fact that neighbouring pixel will often have similar grey levels, and will generally belong to the same class. In order to proceed we need to formulate statistical models for images.

Suppose we do not observe the true image, instead we observe image $\underline{y}$ which is a *degraded* version of $\underline{x}$. This degradation typically includes *blur* and *noise* which may be due to the equipment used to detect and record the image, or physical properties of the process being imaged such as radioactive decay in medical imaging.

The most commonly used model is

$$y_i = \sum_{j \in N(i)} h_{i,j}\, x_j + \epsilon_i$$

where h is a numeric filter (as in chapter 3) and ϵ are independent additive

Normal noise, $\epsilon_j \sim N(0, \sigma^2)$. A more compact formulation is in matrix terms

$\underline{y} = H\underline{x} + \underline{\epsilon}$.

Possible estimators are the direct inversion $H^{-1}\underline{y}$ and the least-squares solution $(H^T H)^{-1} H^T \underline{y}$. These solutions, however, are rarely acceptable because of numerical difficulties caused by ill-conditioning. Briefly, high-frequency components in the truth are smoothed away by the blur function, but in practice almost certainly exist in the observed image because of noise. These components are then grossly magnified by the inversion process, leading to a estimate which may consist largely of random noise.

The following approach to image analysis has been proposed, by some for totally pragmatic reasons and by others for philosophical reasons. This is often called *Bayesian image analysis* and was first proposed in 1984 by Geman and Geman, and by Besag. Instead of basing estimation on the likelihood function, we consider the *posterior distribution* which incorporates additional (prior) information.

Suppose that we have some idea of the nature of the true scene, and can quantify this knowledge as a probability distribution, $\pi(\underline{x})$ say. Assuming that we also know the form of the noise, which defines the *likelihood*, $l(\underline{y}|\underline{x})$, we can use Bayes' rule to give

$$\pi(\underline{x}|\underline{y}) = l(\underline{y}|\underline{x})\pi(\underline{x})/\pi(\underline{y}).$$

We call $\pi(\underline{x})$ the *prior distribution* and $\pi(\underline{x}|\underline{y})$ the *posterior distribution*.

The direct analogy of the maximum likelihood principle is maximum a posteriori (MAP), that is the value of $\underline{x}$ which maximises the posterior probability (or density); we could also use the posterior mean, that is the mean of the posterior distribution.

To define the likelihood we need to consider two components, blur and noise. In some applications the blur function is very complex, for example in the medical imaging technique of emission tomography. However, in many cases instead of using the exact function an approximation will be used, such as a normal distribution, which will capture the gross properties of the blur. For example using a (bivariate) normal blur

$$h_{i,j} = \frac{1}{2\pi\sigma_h^2} \exp\left\{-\frac{1}{2\sigma_h^2}(i^2 + j^2)\right\}.$$

One way to visualise blurring, is to consider a single point of light. Without blurring we would see this as a single point, however the blurring means that we see it as a "disk" which is brightest in the middle. For a point source of unit intensity, the blur or point-spread function gives the observed brightness around the point. Also, as the variance increases, the peak brightness decreases and the size of the disk increases.

The most commonly used noise models are Normal, e.g. to describe measurement errors, and Poisson, e.g. for radioactive decay in medical imaging:

• Normal: $\underline{Y}|\underline{x} \sim N(H\underline{x}, \sigma^2 I)$ with likelihood

$$l(\underline{y}|\underline{x}) = \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(y_j - \mu_j)^2\right\} \qquad \text{where } \mu_j = \sum_i h_{i,j}\, x_i$$

• Poisson: $\underline{Y}|\underline{x} \sim Po(H\underline{x})$ with likelihood

$$l(\underline{y}|\underline{x}) = \prod_{j=1}^{m} \frac{\mu_j^{y_j} e^{-\mu_j}}{y_j!} \qquad \text{where } \mu_j = \sum_i h_{i,j}\, x_i$$

As discussed earlier we shall base estimation on the posterior distribution, that is by considering both prior information and evidence from the data. An obvious approach is to look for the value of $\underline{x}$ which maximises the posterior probability, the image which is most likely given the data and our prior belief.

Recall, however, that a typical image might be 1024 pixels by 1024 pixels, so we shall be attempting to estimate over 1 million parameters, hence it is unlikely that standard optimisation algorithms (such as quasi-Newton or conjugate gradient procedures) will work well. Instead we could adopt a procedure similar to the simulation-based Gibbs sampler discussed in the previous section, called the *Metropolis-Hastings algorithm.* (In fact for some examples where the posterior conditional distribution comes from a standard family the Gibbs sampler can be used.) Although standard maximisation approaches are unlikely to work, other procedures for finding the MAP estimate are available, one such method is called *simulated annealing*.

6.6 Segmentation via Simulated Annealing

The challenge in the MAP problem is that there are q^{rc} possible image segmentations (q classes possible for each pixel and rc pixels). So for a 256×256 image, and only 2 classes, there are over $10^{140\times140}$ possible solutions. A proposed optimization technique is simulated annealing, which is a simple idea.

Let P denote a probability distribution over a finite set $\mathcal{X}$. We want to choose

$x^* \in \mathcal{X}$ to maximize $P(x)$. Suppose we consider

$$P_\lambda(x) = P(x)^\lambda \Big/ \sum_{y \in \mathcal{X}} P(y)^\lambda .$$

Then, as $\lambda \to \infty$ the distribution P_λ increasingly concentrates on x^*, i.e.

$$\lim_{\lambda\to\infty} P_\lambda(x) = \begin{cases} 1 & \text{if } x = x^* \\ 0 & \text{otherwise.} \end{cases}$$

So, if we simulate from $P_\lambda(x)$ with λ large then we hope that x_λ is close to x^*.

Example.

$$\begin{aligned} P(1) &= p \\ P(0) &= 1-p \end{aligned}$$

$$P_\lambda(0) = \frac{(1-p)^\lambda}{p^\lambda + (1-p)^\lambda} \qquad P_\lambda(1) = \frac{p^\lambda}{p^\lambda + (1-p)^\lambda}$$

$$\lim_{\lambda\to\infty} P_\lambda(0) = \lim_{\lambda\to\infty} \frac{(1-p)^\lambda}{p^\lambda + (1-p)^\lambda} = \lim_{\lambda\to\infty} \frac{\left(\frac{1-p}{p}\right)^\lambda}{1 + \left(\frac{1-p}{p}\right)^\lambda} = 0 \text{ if } 1-p < p$$

$$= \lim_{\lambda\to\infty} \frac{1}{\left(\frac{p}{1-p}\right)^\lambda + 1} = 1 \text{ if } 1-p > p$$

$$= \frac{2^{-\lambda}}{2^{1-\lambda}} = \frac{1}{2} \text{ if } p = 1-p = 1/2 \qquad \text{etc.}$$

Now suppose $P \propto \exp(-\beta u)$ is a Gibbs probability distribution. Thus P_λ corresponds to replacing β by $\beta\lambda$. Since in statistical physics β is inversely proportional to temperature, it is usual now to write $\lambda = 1/T$ and speak of a decreasing temperature to 0.

We now want to sample from P_λ, but there is no way to enumerate the entire sample space $\mathcal{X}$, so we cannot use conventional methods. A commonly used method is an iterative one, known as the Metropolis algorithm. The following sections are for the Ising Model, but can be generalised.

6.7 Metropolis Algorithm

Let $\mathcal{X}$ be $\{X_j, j = 1, \dots q^{rc}\}$, the set of all possible image segmentations, and let $\pi = (\pi_j)$ be a non-constant distribution over $\mathcal{X}$. If we can find a transition matrix $P = [P_{ij}]$ (where P_{ij} denotes the one step transition probability from x_i to x_j) such that

$$\pi_i p_{ij} = \pi_j p_{ji} \qquad \forall\, i \neq j$$

then the $\pi = (\pi_j)$ is an equilibrium distribution of P, i.e. the probability that the system is in state X_j at a particular time point a long way from the time origin, this being unaffected by the initial state.

An *irreducible* Markov chain is one in which all states inter-communicate.

Choose *any* symmetric irreducible transition matrix $Q = [q_{ij}]$. When in state X_i select a new state j from the i^{th} row of Q. Make the transition with probability $\min(1, \pi_j/\pi_i)$, otherwise stay at X_i.

This defines P as

$$
\begin{aligned}
p_{ij} &= q_{ij} \min(1, \pi_j/\pi_i) \qquad i \neq j \\
p_{ij} &= q_{ii} + \sum_k q_{ik} \max(0, 1 = \pi_k/\pi_i) \qquad i = j.
\end{aligned}
$$

In practice we can choose $q_{ij} = \frac{1}{rc}$ (binary case) for all states X_j which differ from X_i at just one site/pixel, $q_{ij} = 0$ for all other X_j.

Implementation. Let X and Y be two realizations of a binary MRF on a pixel. Then if X and Y differ only at one pixel, say pixel j, then

$$
\frac{\pi_x}{\pi_y} = \frac{P(x_j|\,\text{rest})\,P(\text{rest})}{P(y_j|\,\text{rest})\,P(\text{rest})} = \frac{P(x_j|\,\text{rest})}{P(y_j|\,\text{rest})}
$$

and since these are binary $(0,1)$ pixels we have

$$\frac{\pi_x}{\pi_y} = \frac{P\left(x_j \middle| X \text{ at neighbours of } j\right)}{P\left(1 - x_j \middle| X \text{ at neighbours of } j\right)}$$

with $P\left(x_j = x | X \text{ at neighbours}\right) = \frac{e^{xT}}{1+e^{T}}$ $\quad x = 0,1$ as before, where T is defined after equation (1).

Metropolis algorithm for the Ising model with first-order neighbourhood

1. Choose site k at random
2. Set $R = e^{2\beta(1-2x_k)\left(\sum_{j\in N(k)} x_j - 2\right)}$
3. Generate $U \sim U(0,1)$
4. If $R \geq U$ then $x_k = 1 - x_k$
5. Go to 1.

This can be modified to scan the pixels in a deterministic manner.

6.8 Gibbs Sampler

This is a variant of Metropolis' algorithm which replace $\min(1, \pi_j/\pi_i)$ by $\frac{\pi_j}{\pi_i+\pi_j}$.

If we note that $\frac{\pi_x}{\pi_x+\pi_y} = \frac{1}{1+\pi_y/\pi_x}$, then we have

$$\frac{\pi_x}{\pi_x+\pi_y} = P\left(x_j | X \text{ at neighbours of } j\right)$$

since

$$P\left(x_j | X \text{ at neighbours of } j\right) + P\left(1 - x_j | X \text{ at neighbours of } j\right) = 1.$$

This algorithm changes the colour of a pixel less often in practice.

For the Ising model with first-order neighbourhood the algorithm is then:

1. Choose a site k at random
2. Set $R = e^{+2\beta(1-x_k)\left(\sum_{j\in N(k)} x_j - 2\right)} \Big/ \left\{1 + e^{+2\beta\left(\sum_{j\in N(k)} x_j - 2\right)}\right\}$
3. Generate $U \sim U(0,1)$
4. If $R \geq U$ then $x_k = 1 - x_k$
5. Go to 1.

Again there is also a systematic version.

6.9 Implementation of Simulated Annealing

We give an example for the Strauss model. The distribution of the labels is

$$P\left(\text{labels}\right) \propto \exp\left(\beta\#\left(\text{neighbour pairs of the same label}\right)\right)$$

and for additive Gaussian noise

$$P\left(X|\text{ labels}\right) \propto \exp\left[-\frac{1}{2\kappa}\sum_{\text{pixels}}\left(X_{ij}-\mu_{l_{ij}}\right)^2\right]$$

where l_{ij} is the class label of the (i,j)th pixel, μ_k is the true grey-level of class k pixels, and κ is the variance of the noise. We then have

$$\log P\left(\text{labels}|X\right) = \text{ const } - \frac{1}{2\kappa}\sum\left(x_{ij}-\mu_{l_{ij}}\right)^2 + \beta\sum_{\text{pairs}} I\left(\text{same label}\right)$$

$$\text{where } I\left(\text{same label}\right) = \begin{array}{l} 1 \text{ if pairs have same label} \\ 0 \text{ otherwise.} \end{array}$$

So we have to choose the labels to minimize:

$$E = \sum_{\text{pixels}}\left(X_{ij}-\mu_{l_{ij}}\right)^2 - 2\beta\kappa\sum_{\text{pixels}} I\left(\text{same label}\right).$$

The algorithm to find the labels is also known as a Gibbs sampler, and is similar to that described above.

The pixels are chosen in sequence, we replace the label for pixel(i, j) by a sample from

$$P\left(l_{ij}|X, \text{ other labels}\right) \text{ i.e.}$$

$$P\left(l_{ij}=k|X, \text{ other labels}\right) \propto \exp\left[\frac{-1}{2\kappa}\left(X_{ij}-\mu_k\right)^2+\beta\sum_{\text{neighbours}} I\left(\text{label } k\right)\right].$$

In the simulated annealing procedure we simulate from

$$P_{ijk} \propto \exp\left(-\frac{\lambda}{2\kappa}\left(X_{ij}-\mu_k\right)^2+\beta\lambda\sum_{N(i,j)} I\left(\text{label } k\right)\right).$$

As the simulation continues we gradually let $\lambda \to \infty$. Theory shows that the rate of convergence that we should take to guarantee a global optimum is $\lambda \sim$ const $\log(t)$ which is much too slow for practical usage. However, a faster rate will lead to a reasonable approximate solution.

An alternative approach, known as *iterative conditional modes* (ICM) is to set $\lambda = \infty$. This corresponds to updating each pixel to maximize $P\left(l_{ij}=k|X\right.$, other labels$\left.\right)$.

6.10 Choice of β

The value of β will determine the smoothness of the final labelling some simple geometric arguments can give some guidance. Suppose there are c labels and we use 8 neighbours.

If we have

$$\begin{array}{ccc} A & A & A \\ A & ? & A \\ A & A & A \end{array}$$

Then $P\left(? \neq A \middle| \text{ other labels}\right) = 1 - \frac{e^{8\beta}}{e^{8\beta}+c-1} = \frac{c-1}{c-1+e^{8\beta}}$.

If we have

$$\begin{array}{ccc} A & A & A \\ A & ? & B \\ A & B & B \end{array}$$

Then

$$P\left(? = B \middle| \text{ other labels}\right) = \frac{e^{3\beta}}{e^{3\beta} + e^{5\beta} + c - 2}.$$

For the case $c = 2$, if we specify the first probability to be less than 0.1% and the second to be at least 10% this gives a plausible range of β as $0.86 < \beta < 1.10$.

In the case of 4 neighbours we have

$$\begin{array}{ccc} & A & \\ A & ? & A \\ & A & \end{array}$$

so

$$P\,(? \neq A|\text{ other labels}) = \frac{c-1}{c-1+e^{4\beta}}$$

which gives a lower band for β as twice that above. Whereas for

$$\begin{array}{ccc} & A & \\ A & ? & B \\ & B & \end{array}$$

$$P\,(? = A) = P\,(? = B) \quad (= .5 \text{ if } c = 2)$$

so there must be a greater acceptability of sharp corners.