6
Image restoration

Building upon our consideration of the Fourier (frequency) domain in the previous chapter, we now progress to explore a practical use of these methods in the field of image restoration.

6.1 Imaging models

Image restoration is based on the attempt to improve the quality of an image through knowledge of the physical processes which led to its formation. As we discussed in Chapter 5, we may consider image formation as a process which transforms an *input distribution* into an *output distribution*.[1] The input distribution represents the ideal, i.e. it is the 'perfect' image to which we do not have direct access but which we wish to recover or at least approximate by appropriate treatment of the imperfect or corrupted output distribution. Recall that, in 2-D linear imaging systems, the relationship between the input distribution $f(x', y')$ and the measured output distribution $g(x, y)$ is represented as a linear superposition integral. For linear, shift invariant (LSI) systems, this reduces to the form of a convolution:

$$\begin{aligned} g(x,y) &= \iint f(x',y')h(x-x',y-y')\mathrm{d}x'\mathrm{d}y' + n(x,y) \\ g(x,y) &= f(x,y) ** h(x,y) + n(x,y) \end{aligned} \tag{6.1}$$

where $**$ is used to denote 2-D convolution. In Equation (6.1), the quantity $h(x-x', y-y')$ is the Point Spread Function (PSF) or impulse response and $n(x, y)$ is an additive noise term. These two factors are responsible for the imperfect output distribution which is obtained.

The image restoration task is (in principle at least) simple:

> Estimate the input distribution $f(x', y')$ using the measured output $g(x, y)$ and any knowledge we may possess about the PSF $h(x-x', y-y')$ and the noise $n(x, y)$.

Recovery of the input distribution $f(x', y')$ from Equation (6.1) is known as *deconvolution*. Image restoration has now evolved into a fascinating but quite complex field of research. The simple observation that *any* approach to image restoration will be *ultimately*

[1] These are also often referred to as the *object* and the *image*.

Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab Chris Solomon and Toby Breckon
 ISBN: 978-0-470-84472-4

limited by our explicit or implicit knowledge of the PSF and the noise process is helpful in establishing firm ground at the beginning.

Equation (6.1) is not the only model which describes image formation – some situations require a more complex model – but it is certainly by far the most important in practice. Some imaging situations need to be described by more complex models, such as *spatially variant* and *nonlinear* models. However, these are relatively rare and, in any case, can be best understood by first developing a clear understanding of approaches to restoration for the LSI model. Accordingly, Equation (6.1) will form the basis for our discussion of image restoration.

6.2 Nature of the point-spread function and noise

We have said that the quality of our image restoration will be limited by our knowledge of the PSF and the noise. What, then, are the natures of these two entities? Although important exceptions do exist in practice,[2] the system PSF is typically *a fixed or deterministic quantity* that is determined by the physical hardware which constitutes the overall imaging system and which may, therefore, be considered to be unchanging with time. For example, in a simple optical imaging system, the overall PSF will be determined by the physical nature and shape of the lenses; in a medical diagnostic imaging system, such as the Anger camera (which detects gamma-rays emitted from the body of a patient), the PSF is determined by a combination of a mechanical collimator and a scintillator–photomultiplier system which allows the detection of the origin of an emitted gamma photon with a certain limited accuracy.

Thus, given certain physical limitations relating to the nature of the detected flux in the imaging system (i.e. optical, infrared, sonar, etc.), the PSF is typically something over which we initially have some control in principle but which becomes fixed once the system has been designed and engineered. Precisely because the PSF is the result of a design and engineering process, it is something of which we can usually expect to have knowledge and which will assist in approaching the restoration problem.

By contrast, the noise term in Equation (6.1) is typically *stochastic* in nature and produces a random and unwanted fluctuation on the detected signal. The key characteristic of noise processes is that we usually have no control over them and we cannot predict the noise which will be present in a given instant. Noise originates from the physical nature of detection processes and has many specific forms and causes. Whatever the specific physical process that gives rise to the formation of an image, the common characteristic is the unpredictable nature of the signal fluctuations. However, although we cannot know the values of a specific realization of the noise, we can often understand and model its statistical properties. The variety of noise models and behaviour are often a major factor in the subtly different approaches to image restoration. Figure 6.1 summarizes the basic image formation/restoration model described by Equation (6.1).

Image *formation* results from convolution of the input with the system PSF and the addition of random noise. Linear *restoration* employs a linear filter whose specific form

[2] The PSF of a telescope system viewing light waves which propagate through atmospheric turbulence is one important example in astronomical imaging where the PSF may be considered as *random* in nature and changes significantly over a short time-scale.

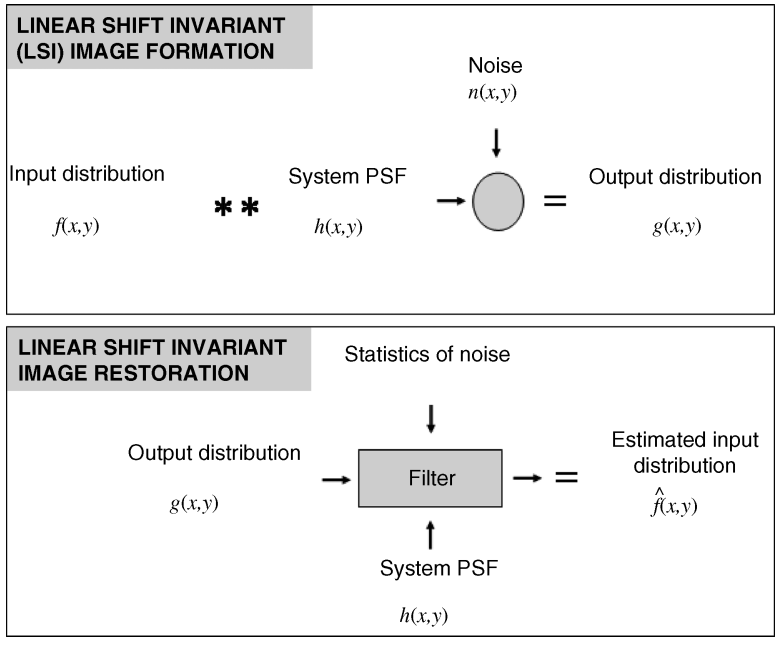

Figure 6.1 The main elements of linear (shift invariant) imaging

depends on the system PSF, our knowledge of the statistics of the noise and, in certain cases, known or assumed statistical properties of the input distribution.

6.3 Restoration by the inverse Fourier filter

Consider again the LSI imaging equation presented in Equation (6.1):

$$g(x,y) = \int\int f(x',y')h(x-x',y-y')\mathrm{d}x'\mathrm{d}y' + n(x,y)$$

in which the quantity $h(x-x',y-y')$ is the PSF, $n(x,y)$ is an additive noise term and $f(x',y')$ is the quantity we wish to restore.

If we Fourier transform both sides of this equation, then we may apply the *convolution theorem* of Chapter 5 to obtain

$$\begin{aligned} \mathbf{F}_{\mathrm{T}}\{g(x,y)\} &= \mathbf{F}_{\mathrm{T}}\{f(x,y) * *h(x,y) + n(x,y)\} \\ G(k_x,k_y) &= F(k_x,k_y)H(k_x,k_y) + N(k_x,k_y) \end{aligned} \tag{6.2}$$

Convolution in the spatial domain has now become a simple multiplication in the frequency domain. Let us consider the ideal situation in which the additive noise term in our imaging equation $n(\mathbf{x})$ is negligible, i.e. $n(\mathbf{x}) = 0$.

In this case a trivial solution results. We simply divide both sides of Equation (6.2) by $H(K_x, K_y)$ and then take the inverse Fourier transform of both sides. Thus:

$$F(k_x, k_y) = \frac{G(k_x, k_y)}{H(k_x, k_y)} = Y(k_x, k_y)G(k_x, k_y) \tag{6.3}$$

and so

$$f(x, y) = F^{-1}\{Y(k_x, k_y)G(k_x, k_y)\} \tag{6.4}$$

where F^{-1} denotes the operation of inverse Fourier transformation. The frequency-domain filter

$$Y(k_x, k_y) = \frac{1}{H(k_x, k_y)} \tag{6.5}$$

where H(k_x, k_y) is the system optical transfer function (OTF) is called the *inverse filter*.

In practice, the straight inverse filter rarely works satisfactorily and should only ever be used with extreme caution. The reason for this can be understood by examining Equation (6.5) and considering the simple fact that *the OTF of any imaging system will generally not extend to the diffraction limit*. This means that any attempt to recover spatial frequency pairs (k_x, k_y) which may exist in the input but at which the OTF of the system has dropped to an effective value of zero will be disastrous. The magnitude of the multiplicative filter $Y(k_x, k_y)$ at these frequencies will then tend to $\sim 1/0 \rightarrow \infty$.

One way round this is to use a truncated inverse filter in which one effectively monitors the value of the OTF $H(k_x, k_y)$ and sets the value of the filter function $Y(k_x, k_y)$ to zero whenever $H(k_x, k_y)$ falls below a predetermined 'dangerously low' value. Technically, this is an example of a *band-pass filter* because it allows certain spatial frequency pairs to be passed into the reconstruction and suppresses others. This is, in fact, how the restoration was achieved in Figure 6.2 (the Matlab® code used to generate this figure is given in Example 6.1).

However, whenever *noise is present*, the use of an inverse filter will have unpredictable and often disastrous effects. Consider applying the inverse filter in Equation (6.5) to a situation in which noise is present. We obtain

$$\begin{aligned} \hat{F}(k_x, k_y) = Y(k_x, k_y)G(k_x, k_y) &= \frac{G(k_x, k_y)}{H(k_x, k_y)} + \frac{N(k_x, k_y)}{H(k_x, k_y)} \\ &= F(k_x, k_y) + \frac{N(k_x, k_y)}{H(k_x, k_y)} \end{aligned} \tag{6.6}$$

where we use the common 'hat' notation (i.e. $\hat{F}(k_x, k_y)$) to denote an estimated quantity. Equation (6.6) shows that our recovered frequency spectrum has an additional term: the noise spectrum $N(k_x, k_y)$ divided by the system OTF $H(k_x, k_y)$. Clearly, we would like this additional term to be as small as possible, since the estimated spectrum will then approach the true input spectrum. The noise spectrum, however, is an unknown and random addition

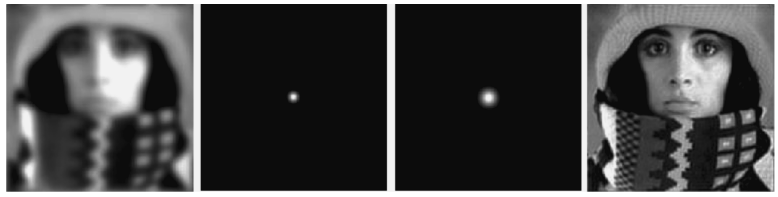

Figure 6.2 Restoration of a blurred but noiseless image through *inverse* filtering. From left to right (a) blurred original; (b) Gaussian PSF; (c) corresponding MTF; (d) recovered original

Example 6.1

Matlab code	**What is happening?**
A=imread('trui.png'); B=fft2(A); B=fftshift(B);	%Read in image and take FT
[x y]=size(A); [X Y]=meshgrid(1:x,1:y);	%Construct Gaussian PSF
h=exp(-(X-x/2).^2./48).*exp(-(Y-y/2).^2./48);	%extending over entire array
H=psf2otf(h,size(h)); H=fftshift(H);	%Get OTF corresponding to PSF
g=ifft2(B.∗H); g=abs(g);	%Generate blurred image via %Fourier domain
G=fft2(g); G=fftshift(G);	%Take FT of image
indices=find(H>1e-6);	%Do inverse filtering AVOIDING
F=zeros(size(G)); F(indices)=G(indices)./ H(indices);	%small values in OTF !!
f=ifft2(F); f=abs(f);	%Inverse FT to get filtered image
subplot(1,4,1), imshow(g,[min(min(g)) max(max(g))]);	%Display "original" blurred image
subplot(1,4,2), imagesc(h); axis square; axis off;	%Display PSF
subplot(1,4,3), imagesc(abs(H)); axis square; axis off;	%Display MTF
subplot(1,4,4), imagesc(f); axis square; axis tight; axis off;	%Display filtered image

Comments

New Matlab functions: ***find***, ***psf2otf***.

The first section of code generates a Gaussian blurred image. The second section estimates the original image using an inverse filter.

to the data to which we 'do not have access' and which is actually quite inseparable from the output image spectrum. Moreover, it is characteristic of many noise processes that they have *significant high-frequency content*; in other words, there is a spatial frequency regime for which $|N(k_x, k_y)| \gg |G(k_x, k_y)|$. In this case, it is clear that the first term on the right-hand side in Equation (6.6) (the true spatial frequency content) will be completely dominated by the second term (the noise). This is the case shown in the Figure 6.3 (the Matlab code used to generate this figure is given in Example 6.2).

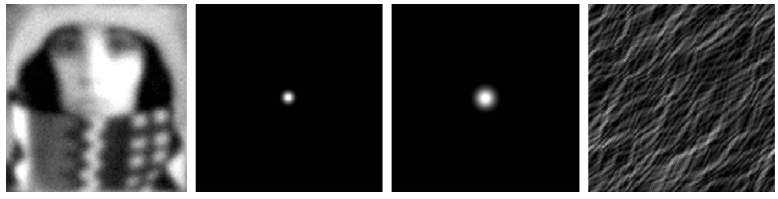

Figure 6.3 Restoration of a blurred and noisy image through *inverse* filtering. From left to right: (a) blurred original with Gaussian white noise (zero mean, 0.2 % variance); (b) system PSF; (c) corresponding MTF; (d) recovered image

Example 6.2

Matlab code	**What is happening?**
A=imread('trui.png'); B=fft2(A); B=fftshift(B);	
[x y]=size(A); [X Y]=meshgrid(1:x,1:y);	
h=exp(-(X-x/2).^2./48).*exp(-(Y-y/2).^2./48);	%CODE SAME AS EXAMPLE 6.1
H=psf2otf(h,size(h)); H=fftshift(H);	%Get OTF
g=ifft2(B.*H); g=mat2gray(abs(g));	%Blur image in Fourier domain
g=imnoise(g,'gaussian',0,0.002);	%Add noise to image
	%CODE HEREAFTER SAME AS 6.1

Comments
Matlab functions: ***imnoise***, ***mat2gray***.
Similar to Example 6.1 except that white noise is added to the image.

6.4 The Wiener–Helstrom filter

As Example 6.2 and the corresponding Figure 6.3 show, whenever noise is present in an image we require a more sophisticated approach than a simple inverse filter. Inspection of the LSI imaging equation in the frequency domain

$$\hat{F}(k_x, k_y) = Y(k_x, k_y)G(k_x, k_y) = Y(k_x, k_y)[H(k_x, k_y)F(k_x, k_y) + N(k_x, k_y)] \tag{6.7}$$

suggests that we would ideally like our frequency-domain filter $Y(k_x, k_y)$ to have the following qualitative properties:

- At those spatial frequency pairs for which the noise component $|N(k_x, k_y)|$ is much smaller than the image component $|G(k_x, k_y)|$, our filter should approach the inverse filter. Thus, we want

$$Y(k_x, k_y) \approx \frac{1}{H(k_x, k_y)} \text{ when } |N(k_x, k_y)| \ll |G(k_x, k_y)|$$

  This ensures accurate recovery of these frequency components in the restored image.

- At those spatial frequency pairs for which the noise component $|N(k_x, k_y)|$ is much larger than the image component $|G(k_x, k_y)|$, our filter should approach *zero*. Thus:

$$Y(k_x, k_y) \approx 0 \text{ when } |N(k_x, k_y)| \gg |G(k_x, k_y)|$$

  This ensures that we do not attempt to restore spatial frequency pairs which are dominated by the noise component.

- At those spatial frequency pairs for which the noise component $|N(k_x, k_y)|$ and the image component $|G(k_x, k_y)|$ are comparable, our filter should 'damp' these frequencies, effecting an appropriate compromise between complete acceptance (inverse filter) and total suppression.

These three properties are broadly achieved by the *Wiener–Helstrom* (often abbreviated simply to *Wiener*) *filter*, defined as

$$Y(k_x, k_y) = \frac{H^*(k_x, k_y) W_F(k_x, k_y)}{|H(k_x, k_y)|^2 W_F(k_x, k_y) + W_N(k_x, k_y)} \tag{6.8}$$

In Equation (6.8) H^* denotes the complex conjugate of the OTF and the quantities $W_F(k_x, k_y)$ and $W_N(k_x, k_y)$ are respectively the *input* and *noise power spectra*:

$$W_F(k_x, k_y) = \langle |F(k_x, k_y)|^2 \rangle \quad \text{and} \quad W_N(k_x, k_y) = \langle |N(k_x, k_y)|^2 \rangle \tag{6.9}$$

Division of the right-hand side of Equation (6.8) by the input power spectrum enables us to express the *Wiener–Helstrom filter* in an alternative and more transparent form:

$$Y(k_x, k_y) = \frac{H^*(k_x, k_y)}{|H(k_x, k_y)|^2 + \text{NSR}(k_x, k_y)} \tag{6.10}$$

where the quantity $\text{NSR}(k_x, k_y) = W_N(k_x, k_y)/W_F(k_x, k_y)$ gives the noise/signal power ratio. Equation (6.10) thus shows clearly that the Wiener–Helstrom filter approximates an inverse filter for those frequencies at which the signal/noise power ratio is large, but becomes increasingly small for spatial frequencies at which the signal-noise power ratio is small.

6.5 Origin of the Wiener–Helstrom filter

The Wiener–Helstrom filter defined by Equation (6.8) (or equivalently by Equation (6.10)) is not simply an ad hoc empirical filter but has a firm theoretical basis. It is important because it is, in a clearly defined sense, an *optimal linear restoration filter*. Let us restate the LSI imaging equation presented in Equation (6.1):

$$g(x, y) = \iint f(x', y') h(x-x', y-y')\, \mathrm{d}x'\, \mathrm{d}y' + n(x, y)$$

which, via the convolution theorem, has the corresponding frequency domain equivalent presented in Equation (6.2):

$$G(k_x, k_y) = F(k_x, k_y)H(k_x, k_y) + N(k_x, k_y)$$

A well-defined optimal criterion is to seek an estimate of the input distribution $\hat{f}(x, y)$, which varies minimally from the true input $f(x, y)$ in the *mean-square* sense. Namely, we will seek an estimate $\hat{f}(x, y)$ such that the following quantity is minimized:

$$Q = \left\langle \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} [\hat{F}(x, y) - F(x, y)]^2 \, dx \, dy \right\rangle \tag{6.11}$$

This optimization criterion is called the minimum mean-square error (MMSE) or minimum variance criterion. It is important to be aware that the angle brackets in Equation (6.11) denote ensemble averaging; that is, we seek an estimator $\hat{f}(x, y)$ that is optimal with respect to statistical averaging over many realizations of the random noise process $n(x, y)$and, indeed, to the PSF in those special situations where it may be treated as a stochastic quantity.[3]

Application of Parseval's theorem to Equation (6.11) allows us to express an entirely equivalent criterion in the Fourier domain:

$$Q = \left\langle \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} [\hat{F}(x, y) - F(x, y)]^2 \, dx \, dy \right\rangle \tag{6.12}$$

The basic approach to deriving the Wiener–Helstrom filter is to postulate a linear filter $Y(k_x, k_y)$ which acts on the image data $G(k_x, k_y) = F(k_x, k_y)H(k_x, k_y) + N(k_x, k_y)$ and then to take variations in the operator $Y(k_x, k_y)$ such that the quantity Q is *stationary*. The formal derivation is quite lengthy and requires methods from statistical estimation theory and variational calculus, and so is not repeated here.[4]

It is further possible to show that the resulting mean-square error after application of the Wiener–Helstrom filter is given by

$$Q = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{W_F(k_x, k_y) W_N(k_x, k_y)}{|H(k_x, k_y)|^2 W_F(k_x, k_y) + W_N(k_x, k_y)} \, dx \, dy \tag{6.13}$$

In summary, the Wiener–Helstrom filter works by accepting low-noise frequency components and rejecting high-noise frequency components. Further, it does so in a

[3] A good example of a randomly varying and thus stochastic PSF is the PSF of a ground-based telescope observing through atmospheric turbulence.

[4] Interested readers can study a detailed derivation on the book's website at http://www.fundipbook.com/materials/.

well-defined 'optimal' way. This is very reasonable behaviour and it is tempting to think that image restoration in the frequency domain should begin and end here.

However, despite its practical and historical importance the Wiener–Helstrom filter is no panacea, and two points are essential to make:

(1) The exact implementation of Equation (6.10) requires knowledge of the *input* power spectrum. Since the input distribution is precisely the quantity we are trying to estimate, we cannot generally expect to know its power spectrum![5] Thus, in the vast majority of cases, the Wiener–Helstrom filter cannot be precisely implemented. However, there are several ways in which we can attempt to get round this problem to produce a practical filter:

   (a) When the input distribution belongs to a well-defined class in which the input is constrained, reasonably accurate estimates of the power spectrum may be available. A typical instance might be a standard medical imaging procedure, such as taking a chest radiograph, in which the input structure can essentially be considered as a modest variation on a basic prototype (i.e. the skeletal structure is very similar across the population). In this case, the prototype may be used to provide the approximation to the input power spectrum.

   (b) When the input is not constrained in this way, the Wiener–Helstrom filter can be approximated by substituting the *output* power spectrum $W_G(k_x, k_y)$ (which we can easily calculate) or preferably some filtered version of it *in place* of the input power spectrum.

   (c) Another simple, but less accurate, approach is to approximate the noise/signal ratio (the NSR function in Equation (6.10)) by an appropriate scalar constant. By removing all spatial frequency dependence from this term, one typically trades off some high-frequency content in the restoration for increased stability.

(2) Restorations produced by the Wiener–Helstrom filter often produce restorations which are rather too blurred from the perceptual viewpoint – suggesting that the MMSE criterion, although a perfectly valid mathematical optimisation criterion, is almost certainly not the perceptually optimal from a human standpoint. For this reason, many other filters have been developed which can produce better results for specific applications.

Figure 6.4 (for which the Matlab code is given in Example 6.3) shows the results of two different approximations to the ideal Wiener filter on an image exhibiting a significant level of noise. The restored image is far from perfect but represents a significant improvement on the original.

[5] This point is strangely neglected in many discussions of the Wiener filter.

Example 6.3

Matlab code	What is happening?
I = imread('trui.png');I=double(I);	%Read in image
noise =15.*randn(size(I));	%Generate noise
PSF = fspecial('motion',21,11);	%Generate motion PSF
Blurred = imfilter(I,PSF,'circular');	%Blur image
BlurredNoisy = Blurred + noise;	%Add noise to blurred image
NSR = sum(noise(:).^2)/sum(I(:).^2);	% Calculate SCALAR noise-to-power ratio
NP = abs(fftn(noise)).^2;	%Calculate noise power spectrum
NPOW = sum(NP(:))/prod(size(noise));	%Calculate average power in noise spectrum
NCORR = fftshift(real(ifftn(NP)));	%Get autocorrelation function of the noise, %centred using fftshift
IP = abs(fftn(I)).^2;	%Calculate image power spectrum
IPOW = sum(IP(:))/prod(size(I));	%Calculate average power in image spectrum
ICORR = fftshift(real(ifftn(IP)));	%Get autocorrelation function of the image, %centred using fftshift
NSR = NPOW./IPOW;	%SCALAR noise-to-signal power ratio
subplot(131);imshow(BlurredNoisy,[min(min(BlurredNoisy)) max(max(BlurredNoisy))]);	%Display blurred and noisy image';
subplot(132);imshow(deconvwnr(BlurredNoisy,PSF,NSR),[]);	%Wiener filtered – PSF and scalar noise/ %signal power ratio
subplot(133);imshow(deconvwnr(BlurredNoisy,PSF,NCORR,ICORR),[]);	%Wiener filtered – PSF and noise and signal %autocorrelations

Comments

Matlab functions: ***fftn***, ***ifftn***, ***prod***, ***deconvwnr***.

This example generates a noisy, blurred image and then demonstrates restoration using two variations of the Wiener filter. The first approximates the noise/signal power ratio by a constant and the second assumes knowledge of the noise and signal autocorrelation functions.

The Image Processing Toolbox has a function specifically for carrying out Wiener filtering, ***deconvwnr***. Essentially, it can be used in three different ways:

- The user supplies the PSF responsible for the blurring only. In this case, it is assumed that there is no noise and the filter reduces to the inverse filter Equation (6.5).
- The user supplies the PSF responsible for the blurring and a *scalar* estimate of the noise/signal power ratio NSR. In other words, only the total amounts of power in the noise and in the input are provided and their frequency dependence is not supplied.
- The user supplies the PSF responsible for the blurring and a *frequency-dependent* estimate of the noise/signal power ratio, via their respective autocorrelation functions. (Recall that the autocorrelation theorem allows the calculation of the autocorrelation from the power spectrum.)

Type *doc deconvwnr* for full details.

(a) (b) (c)

Figure 6.4 Restoration of a blurred and noisy image through *Wiener* filtering. From left to right: (a) blurred original with Gaussian white noise (zero mean, standard deviation ~10 % of mean signal); (b) Wiener filtered with scalar estimate of total power in noise and signal; (c) Wiener filtered with autocorrelation of noise and signal provided

6.6 Acceptable solutions to the imaging equation

A solution to the imaging equation, Equation (6.1), is deemed *acceptable* if it is consistent with all the information that we have about the system PSF and the noise process. Explicitly, this means that if we take a solution for the input distribution $\hat{f}(x, y)$, pass it through our system by convolution with the system PSF and subtract the result from the output, the residual (i.e. $\hat{n} = g - h * *\hat{f}$) should be a noise distribution possessing the same statistical properties as that of the noise model. Any spatial structure in the residual which is not consistent with the noise model indicates that the solution is not consistent with the imaging equation. A completely equivalent view can be taken in the frequency domain, where subtraction of the estimated output spectrum $\hat{G}(k_x, k_y)$ from the measured output spectrum $G(k_x, k_y)$ should be consistent with our statistical knowledge of the noise spectrum $N(k_x, k_y)$.

6.7 Constrained deconvolution

The Wiener filter is designed to minimize the sum of the mean-square errors between the actual input distribution and our estimate of it. This is a perfectly sensible criterion for a restoration filter, but we must recognize that it is not the only criterion that we may be interested in. Speaking loosely, the Wiener filter criterion considers *all parts of the image to have equal importance*, as its only goal is to minimize the sum of the squared errors over the entire image. Important visual criteria, such as the preservation of smoothness (typically for noise suppression) or enhancing sharpness (to preserve edges), are not explicitly encapsulated by the Wiener filter. Given that edge structure or overall smoothness, for example, are so important in the perception of images, it stands to reason that we may wish to ensure that the restoration attempts to recover certain characteristics of the image particularly accurately.

In constrained least-squares deconvolution, the task is to restrict ourselves to solutions which minimize some desired quantity in the restored image *but which impose constraints on the solution space*. In many instances, we may know (or at least be able to make an educated estimate) the overall noise power $\|n(x, y)\|^2 = \int n^2(x, y)\, dx\, dy$ in the output

distribution. It stands to reason that convolution of any restored image $\hat{f}(x,y)$ with the PSF of our imaging system (this is our predicted output distribution $\hat{g}(x,y) = h(x,y) * *\hat{f}(x,y)$ if you will) should not differ from the *actual* distribution $g(x,y)$ by an amount that would exceed the known noise power. For this reason, one common and sensible constraint is to demand that the noise power in our restored image be similar to its known value. Thus, this constraint requires that:

$$\|g(x,y) - h(x,y) * *\hat{f}(x,y)\|^2 = \|n(x,y)\|^2. \tag{6.14}$$

We stress that there are *many* possible solutions $\hat{f}(x,y)$ which satisfy the particular constraint expressed by Equation (6.14). This follows from the random nature of the noise and ill-conditioned nature of the system. The specific mathematical approach taken to constrained restoration is to specify a *cost function* which comprises two basic parts. The first part consists of some linear operator **L** (often referred to as the *regularization* operator) acting on the output distribution $\mathbf{L}f(x,y)$. The second part consists of one or more Lagrange multiplier terms which specify the constraints. The aim is to minimize the size of $\mathbf{L}f(x,y)$ subject to the chosen constraints. The precise form of the linear operator **L** will naturally depend on what property of the restored image we wish to minimize and will be discussed shortly.

Formally then, we seek a solution $\hat{f}(x,y)$ which will minimize the cost function:

$$Q = \|\mathbf{L}f(x,y)\|^2 + \lambda\{\|g(x,y) - h(x,y) * *f(x,y)\|^2 - \|n(x,y)\|^2\} \tag{6.15}$$

where **L** is some general linear operator and λ is an unknown Lagrange multiplier.

This problem can just as easily be formulated in the frequency domain by considering Equation (6.2), the Fourier equivalent of the imaging equation. Eq. 6.15 then becomes

$$Q = \|\mathbf{L}F(k_x,k_y)\|^2 + \lambda\{\|G(k_x,k_y) - H(k_x,k_y)F(k_x,k_y)\|^2 - \|N(k_x,k_y)\|^2\} \tag{6.16}$$

By taking vector derivatives of Equation (6.16) with respect to $F(k_x,k_y)$ and setting to zero, we can derive the constrained restoration filter[6] $Y(k_x,k_y)$ such that the restored spectrum is given by

$$\hat{F}(k_x,k_y) = Y(k_x,k_y)G(k_x,k_y)$$

where

$$Y(k_x,k_y) = \frac{\mathrm{H}^*(k_x,k_y)}{|\mathrm{H}(k_x,k_y)|^2 + \alpha|L(k_x,k_y)|^2} \tag{6.17}$$

[6] A derivation of constrained least-squares restoration is available on the book website at http://www.fundipbook.com/materials/.

Thus, $Y(k_x, k_y)$ simply multiplies the output frequency spectrum to produce our estimate of the input spectrum. In Equation (6.17), $H(k_x, k_y)$ denotes the system OTF, $L(k_x, k_y)$ is the frequency-domain representation of the selected linear operator and the parameter $\alpha = 1/\lambda$ (the inverse Lagrange multiplier) is treated as a regularization parameter whose value is chosen to ensure that the least-squares constraint is satisfied. Although analytical methods have been developed to estimate α, it is often treated as a 'user tuneable' parameter.

We stress that the generality of the linear operator $\mathbf{L}$ in Equations (6.15) and (6.16) allows a variety of restoration criteria to be specified. For example, choosing $\mathbf{L} = \mathbf{I}$, the identity operator effectively results in the so-called parametric Wiener filter which produces a minimum norm solution – this is the minimum energy solution which satisfies our constraint. Another criterion is to seek overall maximum 'smoothness' to the image. A good measure of the sharpness in an image is provided by the Laplacian function $\nabla^2 f(x, y)$. By choosing $\mathbf{L} = \nabla^2$ and minimizing $\nabla^2 f(x, y)$, we effectively ensure a smooth solution.

Figure 6.5 was produced by Matlab Example 6.4 and shows examples of constrained least-squares solutions using a Laplacian operator on a blurred and noisy image. Note the increased smoothness which results from fixing the value of λ to be high.

Figure 6.5 Example of constrained deconvolution. Left: original image; centre: restored by minimizing Laplacian of image subject to least-squares noise constraint; right: similar to centre, except Lagrange multiplier fixed at 10 times the previous value

Example 6.4

Matlab code	**What is happening?**
I = imread('trui.png');	%Read in image
PSF = fspecial('gaussian',7,10);	%Define PSF
V = .01;	%Specify noise level
BlurredNoisy = imnoise(imfilter(I,PSF), 'gaussian',0,V);	%Produce noisy blurred image
NP = V.*prod(size(I));	%Calculate noise power
[J LAGRA_J] = deconvreg(BlurredNoisy, PSF,NP);	%Constrained deconvolution
	%default Laplacian operator, Lagrange % multiplier optimised

```
[K LAGRA_K] = deconvreg(BlurredNoisy,PSF,
    [],LAGRA_J*10);
                                          %Lagrange multiplier fixed
                                          %(10 times larger)
subplot(131);imshow(BlurredNoisy);        %Display original
subplot(132);imshow(J);                   %Display 1st deconvolution result
subplot(133);imshow(K);                   %Display 2nd deconvolution result
```

Comments

Matlab functions: ***imfilter***, ***imnoise***, ***fspecial***, ***deconvreg***.

This example generates a degraded image and then shows the results of constrained deconvolution using two different regularization parameters.

The Image Processing Toolbox has a function specifically for carrying out constrained or regularized filtering, ***deconvreg***. Several points are worthy of mention:

- Unless the user *explicitly specifies otherwise*, the default linear operator is taken to be the Laplacian. The reconstruction will then aim to optimize smoothness.
- The user must supply the PSF responsible for the blurring.
- The overall noise power should be specified. The default value is 0.
- The user may specify a given range within which to search for an optimum value of the Lagrange multiplier or may specify the value of the multiplier directly.

Type *doc deconvreg* for full details.

6.8 Estimating an unknown point-spread function or optical transfer function

As we have seen, in the frequency domain the 'standard' deconvolution problem may be expressed by Equation (6.2):

$$G(k_x, k_y) = F(k_x, k_y)H(k_x, k_y) + N(k_x, k_y)$$

and our discussion of frequency-domain restoration so far has covered the simple inverse, Wiener and constrained least-squares approaches to recovering the input spectrum $F(k_x, k_y)$. All of these methods have implicitly assumed that we have knowledge of the system OTF $H(k_x, k_y)$. In many practical situations, this information is not readily available. For example, image analysis and restoration undertaken for forensic purposes very often falls into this category: corrupted images taken by cameras long since lost or otherwise inaccessible are presented in the hope of verifying personal identity or for some other matter of legal importance. In such cases, attempts at restoration can only proceed if some means is found

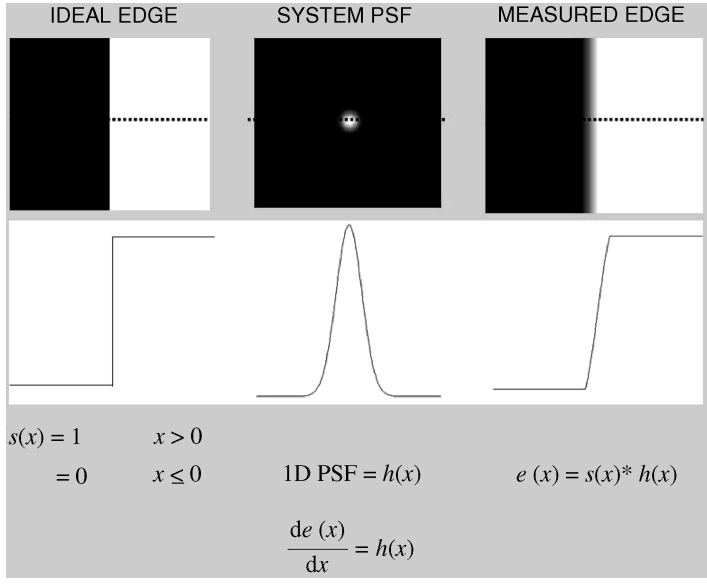

Figure 6.6 The measured 1-D profile through an edge is the result of convolution of the idealized edge with a 1-D PSF $h(x)$. $h(x)$ is calculated as the derivative of the edge $\mathrm{d}e/\mathrm{d}x = h(x)$ and this 1-D PSF is the marginal distribution of the 2-D PSF along the direction of the edge

to sensibly estimate the system OTF (or, equivalently, its PSF) from the image itself. This can be attempted in two basic ways.

- The first is conceptually simple. We can inspect the image for objects which we know *a priori* are point-like in nature (stars are a good example). Their appearance in the image will by definition give an estimate of the PSF. Clearly, such point-like objects are not guaranteed in any image.

- A closely related approach is to look for straight lines and, by implication, *edges*, since the PSF can also be estimated if we can identify these in the image.

To understand this second approach, consider the idealized sharp edge and a blurred version of that edge in Figure 6.6. A profile taken through the edge in a direction perpendicular to the direction of the line is a 1-D signal which is termed the edge-spread function (ESF), which we will denote $e(x)$. Clearly, $e(x)$ is the result of a convolution of the idealized edge with a 1-D system PSF $h(x)$; thus, $e(x) = s(x) * h(x)$. Now, it can be shown formally[7] that the derivative of the ideal edge function is given by a delta function, i.e. $\delta(x-x_0) = \mathrm{d}s/\mathrm{d}x$ (entirely equivalently, the integral of the delta function yields the ideal edge function). Using this relation, we find that simple differentiation of the ESF yields the system PSF, i.e. $\mathrm{d}e/\mathrm{d}x = h(x)$.

Of course, this expression only yields a *1-D version* of the PSF and actually represents an integration (the marginal distribution) of the true 2-D PSF $h(x, y)$ along the direction of

[7] See http://www.fundipbook.com/materials/.

the edge. Now, if there is good reason to believe that $h(x, y)$ is circularly symmetric (and this is reasonable in many cases), then the job is done and our 1-D estimate is simply rotated in the plane through 360 degrees to produce the corresponding 2-D version. However, in the most general case, the situation is rather more involved. Fusing a large number of such marginal distributions of a quite general function $h(x, y)$ can only be done practically by using Fourier techniques (typically employed in computed tomography) which exploit the central slice theorem. The problem becomes much simpler if we can approximate the 2-D PSF $h(x, y)$ by a 2-D Gaussian function. This is very often a good approximation to reality, and the function takes the specific form

$$p(\mathbf{x}) = \frac{1}{2\pi|\mathbf{C}_x|}\exp\left(-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{C}_x^{-1}\mathbf{x}\right) \tag{6.18}$$

where the vector $\mathbf{x} = [\mathrm{x\ y}]$ and the matrix $\mathbf{C}_x$ can be expressed as

$$\mathbf{C}_x = \mathbf{R}_\theta \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \mathbf{R}_\theta^T \tag{6.19}$$

Here, the parameters σ_1^2 and σ_2^2 represent the width of the PSF along two orthogonal directions. Note that the matrix $\mathbf{R}_\theta$ is a 2-D rotation matrix (which posesses one degree of freedom, the angle θ) included to allow for the possibility that the principal axes of the PSF may not be aligned with the image axes. When the principal axes of the PSF are aligned with the image axes, the matrix $\mathbf{R}_\theta = \mathbf{I}$, the identity matrix and the 2-D Gaussian is separable in x and y:-

$$\begin{aligned} p(\mathbf{x}) &= \frac{1}{2\pi|\mathbf{C}_x|}\exp\left(-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{C}_x^{-1}\mathbf{x}\right) \\ &\rightarrow p(x, y) = \frac{1}{\sqrt{2\pi}\sigma_1}\exp\left(-\frac{x^2}{\sigma_1^2}\right)\frac{1}{\sqrt{2\pi}\sigma_2}\exp\left(-\frac{y^2}{\sigma_2^2}\right) \end{aligned} \tag{6.20}$$

Under this assumption of a 2-D Gaussian PSF, the response of the system to an arbitrary edge will be an integrated Gaussian function (the so-called error function familiar from statistics). The error function has a single free parameter and by least-squares fitting the edge response to the error function at a number of edges a robust estimate of the PSF can be formed.

6.9 Blind deconvolution

Consider once more the 'standard' deconvolution problem in the frequency domain given by Equation (6.2):

$$G(k_x, k_y) = F(k_x, k_y)H(k_x, k_y) + N(k_x, k_y)$$

in which we seek to estimate the input spectrum $F(k_x, k_y)$. We have already stressed that we do not know the noise spectrum $N(k_x, k_y)$. **What happens if we also don't know the system OTF $H(k_x, k_y)$?** At first sight, this problem, known as *blind deconvolution*, looks an impossible one to solve. After all, the measured output spectrum $G(k_x, k_y)$ is given by the product of *two unknown quantities* plus a random noise process.

Remarkably, feasible solutions can be found to this problem by iterative procedures (usually carried out in the frequency domain) which enforce only basic constraints on the feasible solutions, namely:

(1) that they have *finite support*, i.e. the sought input distribution is known to be confined to a certain spatial region and is zero outside this region;

(2) any proposed solution must be strictly positive (the input which corresponds to a flux of some nature cannot have negative values).

The blind deconvolution procedure attempts to estimate/restore not only the original input distribution, but also the PSF responsible for the degradation. A detailed account of approaches to blind deconvolution lies outside the scope of this text, but some related references are available on the book website.[8]

Examples of restoration using maximum-likelihood blind deconvolution were generated with Matlab Example 6.5 as shown in Figure 6.7. The PSF of the blurred holiday snap on the far left was unknown and we see that, although the deconvolution procedure has produced some ringing effects, the restored images are indeed sharper.

Example 6.5

Matlab code	**What is happening?**
A=imread('test_blur.jpg');	%Read in image
A=edgetaper(A,ones(25));	%Smooth edges of image
[J,PSF] = deconvblind(A,ones(10));	%Deconvolve – initial estimate PSF 'flat'
subplot(1,4,1), imshow(A,[]);	%Display original
subplot(1,4,2), imshow(J,[]);	%Display deconvolved
h=fspecial('gaussian',[10 10],3);	
[J,PSF] = deconvblind(A,h);	%Deconvolve – initial estimate PSF normal
subplot(1,4,3), imshow(J,[]);	%Display
J = deconvwnr(A,PSF,0.01);	%Wiener filter with 'blind' recovered PSF
subplot(1,4,4), imshow(J,[]);	%Display Wiener deconvolution

Comments

Matlab functions: ***edgetaper***, ***deconvblind***, ***deconvwnr***.

The example above shows the results of blind deconvolution starting from two different initial estimates of the PSF. In general, the results of blind deconvolution are quite sensitive to the initial PSF estimate. The third restoration shows the results of Wiener filtering but using the PSF recovered from blind deconvolution.

The Image Processing Toolbox has a function specifically for carrying out blind deconvolution, ***deconvblind***. Type *doc deconvblind* for full details on how to use the blind deconvolution algorithm in Matlab.

[8] See http://www.fundipbook.com/materials/.

Figure 6.7 Maximum likelihood blind deconvolution. (a) Original image obtained with unknown PSF. (b) Restoration by maximum-likelihood blind deconvolution (initial guess for PSF a 10×10 averaging filter). (c) Restoration by maximum-likelihood blind deconvolution (initial guess for PSF a 10×10 Gaussian filter with $\sigma = 3$). (d) Wiener filtered using the final estimated PSF from (c) and an assumed scalar NSR $= 0.01$

6.10 Iterative deconvolution and the Lucy–Richardson algorithm

Image restoration remains a vast field of research. It would be tempting to offer some discussion on the array of Bayesian-related reconstruction techniques and nonlinear image restoration problems. However, this lies beyond our scope, and this chapter aims only to give the reader a flavour of the basic restoration problem and of some common and basic approaches to solutions. Readers interested in pursuing this subject at a deeper level will find a vast amount of literature at their disposal.

However, partly because it has found such popularity amongst the astronomy community and partly because it is an algorithm that is explicitly provided by the Matlab Image Processing Toolbox, we briefly describe one further technique, namely that of the Lucy–Richardson (LR) deconvolution algorithm. The LR algorithm is best understood if we first consider a simple iterative algorithm and then see how the LR method is an extension of this approach.

The linear imaging equation states that $g(x, y) = f(x, y) * * h(x, y) + n(x, y)$, so that the noise $n(x, y)$ is the difference between the output distribution $g(x, y)$ (i.e. the image we actually measured) and the unknown input distribution $f(x, y)$ convolved with the PSF $h(x, y)$:

$$n(x, y) = g(x, y) - f(x, y) * * h(x, y) \tag{6.21}$$

Substitution of a *good* estimate of the input distribution $f(x, y)$ in Equation (6.21) would tend to make $n(x, y)$ small, whereas a poor estimate would make it large and in the limit of negligible noise our ideal estimate would satisfy:

$$g(x, y) - f(x, y) * * h(x, y) = 0 \tag{6.22}$$

If we add the input distribution $f(x, y)$ to both sides of Equation (6.22), then we have:

$$f(x, y) = f(x, y) + [g(x, y) - f(x, y) * * h(x, y)] \tag{6.23}$$

This equation can be viewed as an iterative procedure in which a new estimate of the input (the left-hand side) is given as the sum of the previous estimate (first term of right-hand side) and a *correction term* (in brackets). The correction term is actually the difference between our measured image and our prediction of it using the current estimate of the input. This certainly seems a reasonable correction to add and is certainly an easy one to calculate. Writing this explicitly as an iterative procedure, the $(k + 1)$th estimate of the input is thus given by:

$$f_{k+1}(x, y) = f_k(x, y) + [g(x, y) - f_k(x, y) * * h(x, y)] \tag{6.24}$$

The procedure described by Equation (6.24) is started by setting $f_0(x, y) = g(x, y)$. In other words, we seed the algorithm by taking our first estimate of the input distribution to be the measured output. Unless the image is very severely degraded, the output is not hugely different from the true input distribution and this simple procedure converges nicely. Note that Equation (6.24) can never be satisfied exactly, unless there is no noise at all, but will reach a point at which the size of the correction term[9] in Equation (6.24) reaches a minimum value.

A variation on Equation (6.24) is the Van Cittert algorithm, which uses a pixel-dependent weighting factor or *relaxation parameter* $w(x, y)$ to control the speed of the convergence:

$$f_{k+1}(x, y) = f_k(x, y) + w(x, y)[g(x, y) - f_k(x, y) * * h(x, y)] \tag{6.25}$$

The basic assumptions of the LR method are twofold:

- we assume that the PSF is known;
- we assume that the noise in the output is governed by the Poisson density function.

The LR method is an iterative algorithm which attempts to find the *maximum-likelihood* solution given knowledge of the PSF and the assumption of Poisson noise. As is customary in most other discussions of the algorithm, we will consider a discrete form of Equation (6.1) in which the input and output distributions are represented by vectors and the PSF by a 2-D matrix. In other words, let the ith pixel in the input distribution have value f_i. This is related to the observed value of the ith pixel in the output g_i by

$$g_i = \sum_j h_{ij} f_j \tag{6.26}$$

where the summation over index j provides the contribution of each input pixel, as expressed by the PSF h_{ij}, to the observed output pixel. It is customary to normalize the discrete PSF so that $\sum_i \sum_j h_{ij} = 1$.

[9] By *size* we mean here the total sum of the squared pixel values.

The iterative LR formula is given by

$$f_j = f_j \sum_i \left(\frac{h_{ij} g_i}{\sum_k h_{jk} f_k} \right) \tag{6.27}$$

where the kernel in Equation (6.27) approaches unity as the iterations progress. The theoretical basis for Equation (6.27) can be established from Bayes' theorem, and the interested reader is referred to the literature for details.[10] Examples of restorations achieved using the LR deconvolution algorithm are given by the code in Matlab Example 6.6 and shown in Figure 6.8.

Example 6.6

Matlab code	**What is happening?**
A = imread('trui.png'); A=mat2gray(double(A));	%Read in image and convert to intensity
PSF = fspecial('gaussian',7,10);	%Specify PSF
V = .0001;	%Define variance of noise
J0 = imnoise(imfilter(A,PSF),'gaussian',0,V);	%Create blurred and noisy image
WT = zeros(size(A));WT(5:end-4,5:end-4) = 1;	%Define weighting function
J1 = deconvlucy(J0,PSF,10);	%LR deconvolution 10 iterations
J2 = deconvlucy(J0,PSF,20,sqrt(V));	%20 iterations, deviation of noise provided
J3 = deconvlucy(J0,PSF,20,sqrt(V),WT);	%weight function to suppress ringing
subplot(141);imshow(J0);	%Display various results
subplot(142);imshow(J1);	
subplot(143);imshow(J2);	
subplot(144);imshow(J3);	

Comments

Matlab functions: ***imnoise***, ***mat2gray***, ***deconvlucy***.

The example above shows the results of LR deconvolution under three different conditions. In the first, only the PSF is supplied and 10 iterations are applied. In the second, an estimate of the overall standard deviation of the noise is supplied. In the third example, a weighting function (which assigns zero weight to pixels near the border of the image) is applied to suppress the ringing effects evident in the previous reconstructions.

The Image Processing Toolbox has a function specifically for carrying out LR deconvolution, ***deconvlucy***. Type *doc deconvlucy* for full details on how to use the LR deconvolution algorithm in Matlab.

[10] Richardson WH. Bayesian-based iterative method of image restoration, *J. Opt. Soc. Am.* **62** (1972) 55. Lucy, LB, An iterative technique for the rectification of observed distributions, *Astron. J.* **79** (1974) 745.

Figure 6.8 LR deconvolution. From left to right: (a) original image; (b) 10 iterations, PSF only supplied; (c) 20 iterations, variance of noise also supplied; (d) 20 iterations, band of five pixels around edge zero-weighted to suppress ringing effects

6.11 Matrix formulation of image restoration

We will end our introductory discussion on image restoration by briefly examining restoration from an alternative perspective. Specifically, we will discuss the *linear algebraic or matrix formulation* of the image restoration problem. This presentation will be fairly mathematical and might fairly be glossed over on a first reading. However, the formulation is a powerful one because it effectively places many seemingly different techniques within an identical mathematical framework. For this reason, we choose to devote the remaining sections to it.

Our starting point is the linear superposition integral introduced in Chapter 5, but with an additional noise term added:

$$g(x,y) = \iint f(x',y')h(x,y;x',y')\,\mathrm{d}x'\,\mathrm{d}y' + n(x,y) \tag{6.28}$$

Note that this equation expresses a linear mapping between a distribution in some input domain specified by coordinates (x',y') to a distribution in some output domain (x,y). With the noise term added, this constitutes our most general expression for a continuous, 2-D, linear imaging model. If the PSF $h(x,y;x',y')$ (which is, in general, four-dimensional) is shift invariant, then this reduces to the standard convolution imaging given by Equation (6.2). However, we will stick with the more general case here, since it is equally simple to treat.

Digital imaging systems produce a *discretized* approximation to the linear superposition integral in which the input domain (x',y') and the output domain (x,y) are sampled into a finite number of pixels. It is straightforward to show[11] that the discretized version of Equation (6.1) results in the matrix equation

$$\mathbf{g} = \mathbf{H}\mathbf{f} + \mathbf{n} \tag{6.29}$$

[11] See http://www.fundipbook.com/materials/.

where **g** is the output or image vector, which contains all the pixels in the image suitably arranged as a single column vector;[12] **f** is the input vector, similarly containing all the pixel values in the input distribution arranged as a single column vector; **n** is the noise vector, where the noise contributions to each pixel are arranged as a single column vector; **H** is the discrete form of the system PSF, describing the transfer of each pixel in the input to each and every pixel in the output, and is sometimes called the system or transfer matrix.

This formulation does not, of course, change anything fundamental about the image restoration problem, which remains essentially the same – namely, how to recover **f** given **g** and various degrees of knowledge of **H** and **n**. However, we do have a new mathematical perspective. Equation (6.29) actually *comprises a large system of linear equations*. This linear model has been the subject of much study within the field of statistical estimation theory and is very well understood. The key advantage of expressing the restoration problem in these terms is that it brings the problem under a simple and unifying framework which is particularly attractive from a theoretical perspective.

The solution of the linear system Equation (6.29) can take a variety of forms. Our discussion must necessarily be modest in scope and will concentrate on just two aspects:

(1) to explain the essence of the estimation theoretic approach to solving this linear problem;

(2) to outline some of the more important and commonly used solutions.

Solutions to the linear model of Equation (6.29) are called *estimators* and the goal is to use our knowledge of the specific situation to derive an estimator $\hat{\mathbf{f}}$ which is as close as possible to the actual input vector **f**.

Generally, different estimators are appropriate, depending on the following three criteria:

- which of the quantities in Equation (6.29) we can treat as stochastic (i.e. random) and which as fixed or deterministic;
- the specific minimization criterion used;
- whether we wish to impose any constraints on the permissible solutions.

We look first at the simplest solution: the standard least-squares estimator.

6.12 The standard least-squares solution

Consider our system of linear equations described by Equation (6.29). The simplest solution from a conceptual point of view is to choose that value of **f** that minimizes the squared

[12] The term stacking operation is sometimes used to refer to this representation of a 2-D image, as each of the original columns of pixels is stacked one on top of another in an ordered fashion.

length of the error vector $\mathbf{n}$. This least-squares criterion effectively says 'find an estimate $\hat{\mathbf{f}}$ which minimizes the sum of the squared errors between the *actual measurement vector* $\mathbf{g}$ and the predicted value $\hat{\mathbf{g}} = \mathbf{H}\hat{\mathbf{f}}$'. Note that such a criterion does not make any explicit assumptions about the statistics of the noise or the *a priori* probability of specific solution vectors.

Accordingly, we define the scalar cost function

$$Q = \mathbf{n}^{\mathrm{T}}\mathbf{n} = \|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 \tag{6.30}$$

and seek a solution vector which minimises Q. Q is a scalar quantity that depends on many free parameters (all the elements of the solution vector $\hat{\mathbf{f}}$). Taking the vector derivative of Q with respect to $\mathbf{f}$ in Equation (6.30) and demanding that this be zero at a minimum imposes the requirement that

$$\begin{aligned} &\frac{\partial Q}{\partial \mathbf{f}} = \frac{\partial}{\partial \mathbf{f}}\left\{[\mathbf{H}\mathbf{f} - \mathbf{g}]^{\mathrm{T}}[\mathbf{H}\mathbf{f} - \mathbf{g}]\right\} = 0 \\ &\Rightarrow \quad \{2\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{f} - \mathbf{H}^{\mathrm{T}}\mathbf{g} - \mathbf{g}^{\mathrm{T}}\mathbf{H}\} = 2\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{f} - 2H^{\mathrm{T}}\mathbf{g} = 0 \\ &\hat{\mathbf{f}} = (\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{g} \end{aligned} \tag{6.31}$$

Note that we have here assumed a zero mean noise process ($\langle \mathbf{n} \rangle = 0$) for simplicity, but a similar result can easily be obtained for a nonzero mean noise process.

The least-squares solution given by Equation (6.31) is conceptually simple and we obtain a neat closed form for the optimal vector $\hat{\mathbf{f}}$.[13] It does, however, have several weaknesses.

(1) The least-squares solution is 'image blind'. By this we mean that its goal is simply to minimize the squared length of the error vector and, as such, it 'treats everything the same', i.e. it does not take into account any image properties or feature specifics which may be of perceptual importance and which we may wish to preserve in our restored image.

(2) There is no guarantee that a true inverse to the term $(\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}$ exists. However, this is typically accommodated by calculating a pseudo-inverse using a singular value decomposition of $\mathbf{H}^{\mathrm{T}}\mathbf{H}$.

(3) It makes no assumptions about the *relative likelihood* of particular solutions. In many cases, it is reasonable to treat the input vector as stochastic in which certain input distributions *are a priori more probable*. This particular issue is explained shortly.

6.13 Constrained least-squares restoration

We offered a discussion of constrained deconvolution in Section 6.7. Here, we re-emphasize that several useful estimators can be derived by minimizing some specific linear operator $\mathbf{L}$

[13] See http://www.fundipbook.com/materials/ for details of the solution.

on the input distribution $\mathbf{f}$ *subject* to the squared noise constraint $\|\mathbf{Hf}-\mathbf{g}\|^2-\|\mathbf{n}\|^2 = 0$. Note that this constraint was mentioned in Section 6.7 and is entirely equivalent, albeit in matrix formulation, to Equation (6.14). Using the method of Lagrange multipliers, we thus form the scalar cost function

$$Q = \|\mathbf{Lf}\|^2-\lambda\{\|\mathbf{Hf}\|^2-\|\mathbf{n}\|^2\} \tag{6.32}$$

and take vector derivatives of Q with respect to $\mathbf{f}$. The general solution is

$$\hat{\mathbf{f}} = [\mathbf{H}^{\mathrm{T}}\mathbf{H}-\alpha\mathbf{L}^{\mathrm{T}}\mathbf{L}]^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{g} \tag{6.33}$$

where $\alpha = 1/\lambda$, the inverse of the Lagrange multiplier. In principle, Equation (6.33) may be used to replace $\mathbf{f}$ in the noise constraint term $\|\mathbf{Hf}-\mathbf{g}\|^2-\|\mathbf{n}\|^2 = 0$ and the value of α adjusted to ensure that the noise constraint is indeed satisfied. In practice, the value of α is often treated as an adjustable parameter.

We note that some interesting specific solutions can be obtained through a specific choice for the linear operator $\mathbf{L}$:

$\mathbf{L} = \mathbf{I}$: parametric Wiener filter

$$\hat{\mathbf{f}} = [\mathbf{H}^{\mathrm{T}}\mathbf{H}-\alpha\mathbf{I}]^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{g} \tag{6.34}$$

The similarity to Equation (6.10) is apparent. α acts like a noise/signal ratio and clearly, in the limit that $\alpha \to 0$, we tend towards the direct inverse or least-squares filter.

$\mathbf{L} = \nabla^2$: maximum smoothness filter

$$\hat{\mathbf{f}} = [\mathbf{H}^{\mathrm{T}}\mathbf{H}-\Gamma^{\mathrm{T}}\Gamma\mathbf{I}]^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{g} \tag{6.35}$$

where the matrix Γ is the discrete representation of the Laplacian operator.

The two forms given by Equations (6.34) and (6.35) are very common, but we stress that, in principle, *any linear operator* which can be implemented in discrete form can be substituted into Equation (6.33).

To end this brief discussion, we note that the computational implementation of the various solutions such as Equation (6.31) and Equation (6.33) cannot be sensibly attempted naively. This is because of the sheer size of the system matrix $\mathbf{H}$. For example, even a digital image of size 512^2 (very modest by today's standards) would result in a system matrix comprising ~69 000 million elements, the direct inversion of which presents a huge calculation. The implementation of Equation (6.33) and similar forms discussed in this section is made possible in practice by the *sparse* nature of $\mathbf{H}$ (most of its elements are zero) and the fact that it exhibits cyclic patterns which are a consequence of the shift invariance of the PSF of the imaging system. $\mathbf{H}$ is said to possess a block circulant structure, and algorithms for efficiently computing quantities, such as $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ and its inverse, have been extensively developed.[14]

[14] H. C. Andrews and B. R. Hunt (1977) *Digital Image Restoration*, Prentice-Hall (ISBN 0-13-214213-9). This is now an old book, but it contains an excellent and comprehensive discussion of such techniques.

6.14 Stochastic input distributions and Bayesian estimators

A key assumption in deriving the least-squares estimators given by Equations (6.22) and (6.24) is that both the system matrix **H** and the input distribution **f** are treated as *deterministic* quantities. Superficially, it would seem that they could not really be otherwise. After all, an imaging system whose properties are not changing with time is clearly deterministic. Further, treating the input distribution as a deterministic (i.e. non-random) quantity seems very reasonable.

Taking the input distribution to be deterministic is certainly acceptable in many instances. However, sometimes we know a priori that *certain input distributions are more likely to occur than others are.* In this case, we may treat the input as stochastic. To illustrate this point, consider a simple but topical example. If we know a priori that we are imaging a human face (but do not know who the actual individual is) it stands to reason that only certain configurations of the vector **f** are at all feasible as solutions. In such a case we treat the input distribution as belonging to a specific pattern class – a concept we explore further in Chapter 11 on classification. Briefly, a pattern class is a conceptual or actual group of patterns which shares certain features in common because (typically) they are sampled from the same underlying probability density function. The specification of an individual pattern in the given class is usually made through an N-dimensional vector, each element of which corresponds to a variable used to describe the pattern. Accordingly, if we have knowledge of the input pattern probability density function (or even just some of its moments), we can try to include this knowledge in our estimation procedure. This is the fundamental motivation behind the whole class of Bayesian estimators. Speaking in loose and rather general terms, Bayesian methods effectively attempt to strike the right balance between an estimate based on the data alone (the recorded image, the PSF and the noise characteristics) and *our prior knowledge of how likely certain solutions are in the first place*. We will finish this chapter with a powerful but simple derivation from 'classical' estimation theory which incorporates the idea of prior knowledge about likely solutions without explicitly invoking Bayes' theorem.[15] This is the famous generalized Gauss–Markov estimator.

6.15 The generalized Gauss–Markov estimator

Our starting point is the general linear system written in matrix form in Equation (6.29):

$$\mathbf{Hf} + \mathbf{n} = \mathbf{g}$$

We make the following important assumptions on our linear model:

- The system matrix **H** is a deterministic quantity.

[15] It is worth noting that many statisticians are adherents of either classical estimation theory (and do not use Bayes' theorem) or 'die-hard Bayesians' who enthusiastically apply it at more or less every opportunity. Both approaches yield ultimately similar results and the difference really lies in the philosophical perspective.

- The error (noise) vector is a stochastic quantity. We will assume that it has zero mean (this is not restrictive, but it simplifies the algebra somewhat) and a known error covariance function $\mathbf{V} = \langle \mathbf{n}\mathbf{n}^{\mathrm{T}} \rangle$. The input distribution $\mathbf{f}$ is a stochastic quantity. We will also assume without restricting our analysis that it has zero mean and a known covariance function given by $\mathbf{C}_{\mathrm{f}} = \langle \mathbf{f}\mathbf{f}^{\mathrm{T}} \rangle$.

- We will further assume that the noise and signal are *statistically uncorrelated* (a condition that is usually satisfied in most instances). Thus, we have $\langle \mathbf{n}\mathbf{f}^{\mathrm{T}} \rangle = \langle \mathbf{n}^{\mathrm{T}}\mathbf{f} \rangle = 0$.

The angle brackets denote expectation averaging over an ensemble of many observations of the random process. In principle, the covariance matrices $\mathbf{C}_{\mathrm{f}}$ and $\mathbf{V}$ are derived from an infinite number of their respective random realizations. In practice, they must usually be estimated from a finite sample of observations.

We seek an estimator for the input distribution which is given by some linear operator on the observed data -

$$\hat{\mathbf{f}} = \mathbf{L}\mathbf{g} = \mathbf{L}(\mathbf{H}\mathbf{f} + \mathbf{n}) \tag{6.36}$$

and which will minimize the error covariance matrix of the input distribution given by

$$\mathbf{E} = \langle (\mathbf{f} - \hat{\mathbf{f}})(\mathbf{f} - \hat{\mathbf{f}})^{\mathrm{T}} \rangle \tag{6.37}$$

where the matrix operator $\mathbf{L}$ is to be determined. Inserting Equation (6.36) into Equation (6.37) and using the definition of $\mathbf{C}_{\mathrm{f}} = \langle \mathbf{f}\mathbf{f}^{\mathrm{T}} \rangle$, the quantity to be minimized is

$$\mathbf{E} = [\mathbf{I} - \mathbf{L}\mathbf{A}]\mathbf{C}_{\mathbf{f}}[\mathbf{I} - \mathbf{L}\mathbf{A}]^{\mathrm{T}} + \mathbf{L}\mathbf{V}\mathbf{L}^{\mathrm{T}} \tag{6.38}$$

As a momentary aside, we note that whilst the reader will almost certainly be familiar with the notion of minimizing a scalar function of many variables, the notion of 'minimizing' a multivalued matrix quantity as required by Equation (6.38) may, however, be novel.[16] The required approach is to take first variations in $\mathbf{E}$ with respect to the linear operator $\mathbf{L}$ and set these to zero. This yields

$$\delta\mathbf{E} = \delta\mathbf{L}(\mathbf{H}\mathbf{C}_{\mathbf{f}}[\mathbf{I} - \mathbf{L}\mathbf{H}]^{\mathrm{T}} + \mathbf{V}\mathbf{L}^{\mathrm{T}}) + ([\mathbf{I} - \mathbf{L}\mathbf{H}]\mathbf{C}_{\mathbf{f}}\mathbf{H}^{\mathrm{T}} + \mathbf{L}\mathbf{V})\delta\mathbf{L}^{\mathrm{T}} = \mathbf{0} \tag{6.39}$$

The optimal matrix is thus given by

$$\mathbf{L}_{\mathrm{OPT}} = \mathbf{C}_{\mathbf{f}}\mathbf{H}^{\mathrm{T}}(\mathbf{H}\mathbf{C}_{\mathbf{f}}\mathbf{H}^{\mathrm{T}} + \mathbf{V})^{-\mathbf{1}} \tag{6.40}$$

Matrix algebraic manipulation shows that this may also be written as

$$\mathbf{L}_{\mathrm{OPT}} = (\mathbf{H}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{H} + \mathbf{C}_{\mathbf{f}}^{-\mathbf{1}})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{V}^{-1} \tag{6.41}$$

[16] This has a close relation to the calculus of variations. See the book's website http://www.fundipbook.com/materials/ for a brief discussion of this point.

Substitution of the optimal estimator given by Equation (6.41) into the definition of the error covariance matrix on the input distribution given by Equation (6.37) yields

$$\mathbf{E}_{\mathrm{OPT}} = (\mathbf{H}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{H} + \mathbf{C}_{\mathbf{f}}^{-1})^{-1} \tag{6.42}$$

Equation (6.42) is a useful expression because it provides an actual measure of the closeness of the reconstructed or estimated input to the actual input.

We note two limiting forms of interest for Equation (6.41):

- If we assume that we have no prior knowledge of the input distribution, this is equivalent to treating it as deterministic quantity and we effectively allow $\mathbf{C}_{\mathbf{f}} \to \infty$ (i.e. infinite variance/covariance is possible), the Gauss–Markov estimator then reduces to

$$\mathbf{L}_{\mathrm{OPT}} = (\mathbf{H}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{V}^{-1} \tag{6.43}$$

  This is known as the BLUE (best linear unbiased estimator).

- If further we may assume that the noise on the pixels has equal variance σ^2 and is uncorrelated (a reasonable assumption in many cases) then we may write $\mathbf{V} = \sigma^2\mathbf{I}$ and we obtain a standard least-squares solution:

$$\mathbf{L}_{\mathrm{OPT}} = [\mathbf{H}^{\mathrm{T}}\mathbf{H}]^{-1}\mathbf{H}^{\mathrm{T}} \tag{6.44}$$

We stress that the solution presented here is a quite general solution to the linear problem. However, computational implementation of the solutions when the underlying signals are images can generally only be attempted by exploiting the sparse and cyclic structure of the system transfer matrix.

For further examples and exercises see http://www.fundipbook.com