Chapter 1
The Image Deblurring Problem

You cannot depend on your eyes when your imagination is out of focus.
– Mark Twain

When we use a camera, we want the recorded image to be a faithful representation of the scene that we see—but every image is more or less blurry. Thus, image deblurring is fundamental in making pictures sharp and useful.

A digital image is composed of picture elements called pixels. Each pixel is assigned an intensity, meant to characterize the color of a small rectangular segment of the scene. A small image typically has around $256^2 = 65536$ pixels while a high-resolution image often has 5 to 10 million pixels. Some blurring always arises in the recording of a digital image, because it is unavoidable that scene information "spills over" to neighboring pixels. For example, the optical system in a camera lens may be out of focus, so that the incoming light is smeared out. The same problem arises, for example, in astronomical imaging where the incoming light in the telescope has been slightly bent by turbulence in the atmosphere. In these and similar situations, the inevitable result is that we record a blurred image.

In image deblurring, we seek to recover the original, sharp image by using a mathematical model of the blurring process. The key issue is that some information on the lost details is indeed present in the blurred image—but this information is "hidden" and can only be recovered if we know the details of the blurring process.

Unfortunately there is no hope that we can recover the original image exactly! This is due to various unavoidable errors in the recorded image. The most important errors are fluctuations in the recording process and approximation errors when representing the image with a limited number of digits. The influence of this noise puts a limit on the size of the details that we can hope to recover in the reconstructed image, and the limit depends on both the noise and the blurring process.

POINTER. Image enhancement is used in the restoration of older movies. For example, the original Star Wars trilogy was enhanced for release on DVD. These methods are not model based and therefore not covered in this book. See [33] for more information.

POINTER. Throughout the book, we provide example images and MATLAB code. This material can be found on the book's website:

`www.siam.org/books/fa03`

For readers needing an introduction to MATLAB programming, we suggest the excellent book by Higham and Higham [27].

One of the challenges of image deblurring is to devise efficient and reliable algorithms for recovering as much information as possible from the given data. This chapter provides a brief introduction to the basic image deblurring problem and explains why it is difficult. In the following chapters we give more details about techniques and algorithms for image deblurring.

MATLAB is an excellent environment in which to develop and experiment with filtering methods for image deblurring. The basic MATLAB package contains many functions and tools for this purpose, but in some cases it is more convenient to use routines that are only available from the *Signal Processing Toolbox* (SPT) and the *Image Processing Toolbox* (IPT). We will therefore use these toolboxes when convenient. When possible, we provide alternative approaches that require only core MATLAB commands in case the reader does not have access to the toolboxes.

1.1 How Images Become Arrays of Numbers

Having a way to represent images as arrays of numbers is crucial to processing images using mathematical techniques. Consider the following 9×16 array:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 8 & 0 & 0 & 0 & 0 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 8 & 0 & 0 & 0 & 0 & 4 & 4 & 0 & 3 & 3 & 3 & 3 & 3 & 0 \\ 0 & 8 & 8 & 0 & 0 & 0 & 0 & 4 & 4 & 0 & 3 & 3 & 3 & 3 & 3 & 0 \\ 0 & 8 & 8 & 0 & 0 & 0 & 0 & 4 & 4 & 0 & 3 & 3 & 3 & 3 & 3 & 0 \\ 0 & 8 & 8 & 0 & 0 & 0 & 0 & 4 & 4 & 0 & 3 & 3 & 3 & 3 & 3 & 0 \\ 0 & 8 & 8 & 8 & 8 & 8 & 0 & 4 & 4 & 0 & 3 & 3 & 3 & 3 & 3 & 0 \\ 0 & 8 & 8 & 8 & 8 & 8 & 0 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

If we enter this into a MATLAB variable `X` and display the array with the commands `imagesc(X), axis image, colormap(gray)`, then we obtain the picture shown at the left of Figure 1.1. The entries with value 8 are displayed as white, entries equal to zero are black, and values in between are shades of gray.

Color images can be represented using various formats; the RGB format stores images as three components, which represent their intensities on the red, green, and blue scales. A pure red color is represented by the intensity values $(1, 0, 0)$ while, for example, the values $(1, 1, 0)$ represent yellow and $(0, 0, 1)$ represent blue; other colors can be obtained with different choices of intensities. Hence, to represent a color image, we need three values per pixel. For example, if `X` is a multidimensional MATLAB array of dimensions $9 \times 16 \times 3$

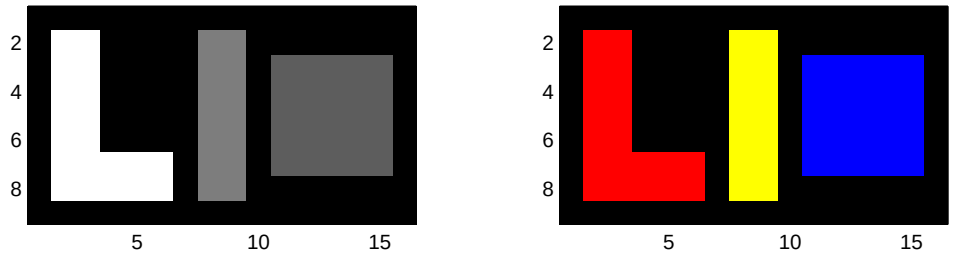


Figure 1.1. *Images created by displaying arrays of numbers.*

defined as

$$\texttt{X(:,:,1)} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&1&1&1&1&1&0&1&1&0&0&0&0&0&0&0\\ 0&1&1&1&1&1&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix},$$

$$\texttt{X(:,:,2)} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix},$$

$$\texttt{X(:,:,3)} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0\\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix},$$

then we can display this image, in color, with the command `imagesc(X)`, obtaining the second picture shown in Figure 1.1. This brings us to our first Very Important Point (VIP).

VIP 1. A digital image is a two- or three-dimensional array of numbers representing intensities on a grayscale or color scale.

Most of this book is concerned with grayscale images. However, the techniques carry over to color images, and in Chapter 7 we extend our notation and models to color images.

1.2 A Blurred Picture and a Simple Linear Model

Before we can deblur an image, we must have a mathematical model that relates the given blurred image to the unknown true image. Consider the example shown in Figure 1.2. The left is the "true" scene, and the right is a blurred version of the same image. The blurred image is precisely what would be recorded in the camera if the photographer forgot to focus the lens.

Figure 1.2. *A sharp image (left) and the corresponding blurred image (right).*

Grayscale images, such as the ones in Figure 1.2, are typically recorded by means of a CCD (charge-coupled device), which is an array of tiny detectors, arranged in a rectangular grid, able to record the amount, or intensity, of the light that hits each detector. Thus, as explained above, we can think of a grayscale digital image as a rectangular $m \times n$ array, whose entries represent light intensities captured by the detectors. To fix notation,

$\mathbf{X} \in \mathbb{R}^{m\times n}$ represents the desired sharp image, while

$\mathbf{B} \in \mathbb{R}^{m\times n}$ denotes the recorded blurred image.

Let us first consider a simple case where the blurring of the columns in the image is independent of the blurring of the rows. When this is the case, then there exist two matrices $\mathbf{A}_c \in \mathbb{R}^{m\times m}$ and $\mathbf{A}_r \in \mathbb{R}^{n\times n}$, such that we can express the relation between the sharp and blurred images as

$$\mathbf{A}_c \, \mathbf{X} \, \mathbf{A}_r^T = \mathbf{B}. \tag{1.1}$$

The left multiplication with the matrix $\mathbf{A}_c$ applies the same *vertical* blurring operation to all n columns $\mathbf{x}_j$ of $\mathbf{X}$, because

$$\mathbf{A}_c \, \mathbf{X} = \mathbf{A}_c \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{bmatrix} = \begin{bmatrix} \mathbf{A}_c\mathbf{x}_1 & \mathbf{A}_c\mathbf{x}_2 & \cdots & \mathbf{A}_c\mathbf{x}_n \end{bmatrix}.$$

Similarly, the right multiplication with $\mathbf{A}_r^T$ applies the same *horizontal* blurring to all m rows of $\mathbf{X}$. Since matrix multiplication is associative, i.e., $(\mathbf{A}_c \, \mathbf{X}) \, \mathbf{A}_r^T = \mathbf{A}_c \, (\mathbf{X} \, \mathbf{A}_r^T)$, it does

POINTER. Image deblurring is much more than just a useful tool for our vacation pictures. For example, analysis of astronomical images gives clues to the behavior of the universe. At a more mundane level, barcode readers used in supermarkets and by shipping companies must be able to compensate for imperfections in the scanner optics; see Wittman [63] for more information.

not matter in which order we perform the two blurring operations. The reason for our use of the transpose of the matrix $\mathbf{A}_r$ will be clear later, when we return to this blurring model and matrix formulations.

1.3 A First Attempt at Deblurring

If the image blurring model is of the simple form $\mathbf{A}_c\,\mathbf{X}\,\mathbf{A}_r^T = \mathbf{B}$, then one might think that the naïve solution

$$\mathbf{X}_{\text{naïve}} = \mathbf{A}_c^{-1}\mathbf{B}\,\mathbf{A}_r^{-T}$$

will yield the desired reconstruction, where $\mathbf{A}_r^{-T} = (\mathbf{A}_r^T)^{-1} = (\mathbf{A}_r^{-1})^T$. Figure 1.3 illustrates that this is probably not such a good idea; the reconstructed image does not appear to have any features of the true image!

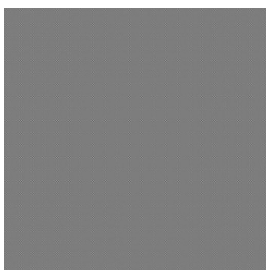


Figure 1.3. *The naïve reconstruction of the pumpkin image in Figure* 1.2*, obtained by computing* $\mathbf{X}_{naïve} = \mathbf{A}_c^{-1}\mathbf{B}\,\mathbf{A}_r^{-T}$ *via Gaussian elimination on both* $\mathbf{A}_c$ *and* $\mathbf{A}_r$*. Both matrices are ill-conditioned, and the image* $\mathbf{X}_{naïve}$ *is dominated by the influence from rounding errors as well as errors in the blurred image* $\mathbf{B}$*.*

To understand why this naïve approach fails, we must realize that the blurring model in (1.1) is not quite correct, because we have ignored several types of errors.

Let us take a closer look at what is represented by the image $\mathbf{B}$. First, let $\mathbf{B}_{\text{exact}} = \mathbf{A}_c\,\mathbf{X}\,\mathbf{A}_r^T$ represent the ideal blurred image, ignoring all kinds of errors. Because the blurred image is collected by a mechanical device, it is inevitable that small random errors (noise) will be present in the recorded data. Moreover, when the image is digitized, it is represented by a finite (and typically small) number of digits. Thus the recorded blurred image $\mathbf{B}$ is really given by

$$\mathbf{B} = \mathbf{B}_{\text{exact}} + \mathbf{E} = \mathbf{A}_c\,\mathbf{X}\,\mathbf{A}_r^T + \mathbf{E}, \tag{1.2}$$

where the noise image $\mathbf{E}$ (of the same dimensions as $\mathbf{B}$) represents the noise and the quantization errors in the recorded image. Consequently the naïve reconstruction is given by

$$\mathbf{X}_{\text{naïve}} = \mathbf{A}_c^{-1}\mathbf{B}\,\mathbf{A}_r^{-T} = \mathbf{A}_c^{-1}\mathbf{B}_{\text{exact}}\,\mathbf{A}_r^{-T} + \mathbf{A}_c^{-1}\mathbf{E}\,\mathbf{A}_r^{-T}$$

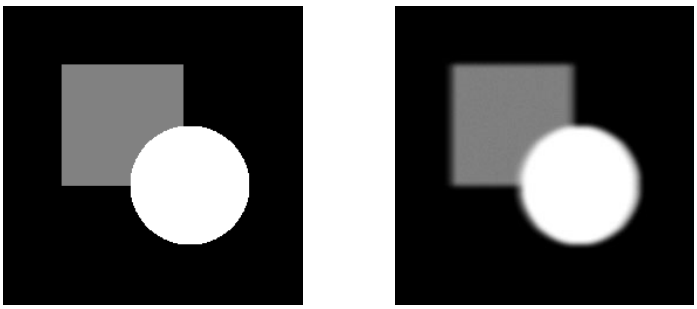
and therefore

$$\mathbf{X}_{\text{naïve}} = \mathbf{X} + \mathbf{A}_c^{-1}\mathbf{E}\,\mathbf{A}_r^{-T}, \tag{1.3}$$

where the term $\mathbf{A}_c^{-1}\mathbf{E}\,\mathbf{A}_r^{-T}$, which we can informally call inverted noise, represents the contribution to the reconstruction from the additive noise. This inverted noise will dominate the solution if the second term $\mathbf{A}_c^{-1}\mathbf{E}\,\mathbf{A}_r^{-T}$ in (1.3) has larger elements than the first term $\mathbf{X}$. Unfortunately, in many situations, as in Figure 1.3, the inverted noise indeed dominates.

Apparently, image deblurring is not as simple as it first appears. We can now state the purpose of our book more precisely, namely, to describe effective deblurring methods that are able to handle correctly the inverted noise.

CHALLENGE 1.



The exact and blurred images `X` and `B` in the above figure can be constructed in MATLAB by calling

```
[B, Ac, Ar, X] = challenge1(m, n, noise);
```

with `m = n = 256` and `noise = 0.01`. Try to deblur the image `B` using

```
Xnaive = Ac\B / Ar';
```

To display a grayscale image, say, `X`, use the commands

```
imagesc(X), axis image, colormap gray
```

How large can you choose the parameter `noise` before the inverted noise dominates the deblurred image? Does this value of `noise` depend on the image size?

CHALLENGE 2.

The above image `B` as well as the blurring matrices `Ac` and `Ar` are given in the file `challenge2.mat`. Can you deblur this image with the naïve approach, so that you can read the text in it?

As you learn more throughout the book, use Challenges 1 and 2 as examples to test your skills and learn more about the presented methods.

CHALLENGE 3. For the simple model $\mathbf{B} = \mathbf{A}_c \mathbf{X} \mathbf{A}_r^T + \mathbf{E}$ it is easy to show that the relative error in the naïve reconstruction $\mathbf{X}_{\text{naïve}} = \mathbf{A}_c^{-1}\mathbf{B}\mathbf{A}_r^{-T}$ satisfies

$$\frac{\|\mathbf{X}_{\text{naïve}} - \mathbf{X}\|_F}{\|\mathbf{X}\|_F} \leq \text{cond}(\mathbf{A}_c)\,\text{cond}(\mathbf{A}_r)\,\frac{\|\mathbf{E}\|_F}{\|\mathbf{B}\|_F},$$

where

$$\|\mathbf{X}\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}^2\right)^{1/2}$$

denotes the Frobenius norm of the matrix $\mathbf{X}$. The quantity $\text{cond}(\mathbf{A})$ is computed by the MATLAB function `cond(A)`. It is the condition number of $\mathbf{A}$, formally defined by (1.8), measuring the possible magnification of the relative error in $\mathbf{E}$ in producing the solution $\mathbf{X}_{\text{naïve}}$.
For the test problem in Challenge 1 and different values of the image size, use this relation to determine the maximum allowed value of $\|\mathbf{E}\|_F$ such that the relative error in the naïve reconstruction is guaranteed to be less than 5%.

1.4 Deblurring Using a General Linear Model

Underlying all material in this book is the assumption that the blurring, i.e., the operation of going from the sharp image to the blurred image, is *linear*. As usual in the physical sciences, this assumption is made because in many situations the blur is indeed linear, or at least well approximated by a linear model. An important consequence of the assumption

POINTER. Our basic assumption is that we have a linear blurring process. This means that if $\mathbf{B}_1$ and $\mathbf{B}_2$ are the blurred images of the exact images $\mathbf{X}_1$ and $\mathbf{X}_2$, then $\mathbf{B} = \alpha\,\mathbf{B}_1 + \beta\,\mathbf{B}_2$ is the image of $\mathbf{X} = \alpha\,\mathbf{X}_1 + \beta\,\mathbf{X}_2$. When this is the case, then there exists a large matrix $\mathbf{A}$ such that $\mathbf{b} = \mathrm{vec}(\mathbf{B})$ and $\mathbf{x} = \mathrm{vec}(\mathbf{X})$ are related by the equation

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}.$$

The matrix $\mathbf{A}$ represents the blurring that is taking place in the process of going from the exact to the blurred image. The equation $\mathbf{A}\,\mathbf{x} = \mathbf{b}$ can often be considered as a discretization of an underlying integral equation; the details can be found in [23].

is that we have a large number of tools from linear algebra and matrix computations at our disposal. The use of linear algebra in image reconstruction has a long history, and goes back to classical works such as the book by Andrews and Hunt [1].

In order to handle a variety of applications, we need a blurring model somewhat more general than that in (1.1). The key to obtaining this general linear model is to rearrange the elements of the images $\mathbf{X}$ and $\mathbf{B}$ into column vectors by stacking the columns of these images into two long vectors $\mathbf{x}$ and $\mathbf{b}$, both of length $N = m\,n$. The mathematical notation for this operator is vec, i.e.,

$$\mathbf{x} = \mathrm{vec}(\mathbf{X}) = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^N, \qquad \mathbf{b} = \mathrm{vec}(\mathbf{B}) = \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_n \end{bmatrix} \in \mathbb{R}^N.$$

Since the blurring is assumed to be a linear operation, there must exist a large blurring matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ such that $\mathbf{x}$ and $\mathbf{b}$ are related by the linear model

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}, \tag{1.4}$$

and this is our fundamental image blurring model. For now, assume that $\mathbf{A}$ is known; we will explain how it can be constructed from the imaging system in Chapter 3, and also discuss the precise structure of the matrix in Chapter 4.

For our linear model, the naïve approach to image deblurring is simply to solve the linear algebraic system in (1.4), but from the previous section, we expect failure. Let us now explain why.

We repeat the computation from the previous section, this time using the general formulation in (1.4). Again let $\mathbf{B}_{\text{exact}}$ and $\mathbf{E}$ be, respectively, the noise-free blurred image and the noise image, and define the corresponding vectors

$$\mathbf{b}_{\text{exact}} = \mathrm{vec}(\mathbf{B}_{\text{exact}}) = \mathbf{A}\,\mathbf{x}, \qquad \mathbf{e} = \mathrm{vec}(\mathbf{E}).$$

Then the noisy recorded image $\mathbf{B}$ is represented by the vector

$$\mathbf{b} = \mathbf{b}_{\text{exact}} + \mathbf{e},$$

and consequently the naïve solution is given by

$$\mathbf{x}_{\text{naïve}} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{A}^{-1}\mathbf{b}_{\text{exact}} + \mathbf{A}^{-1}\mathbf{e} = \mathbf{x} + \mathbf{A}^{-1}\mathbf{e}, \tag{1.5}$$

POINTER. Good presentations of the SVD can be found in the books by Björck [4], Golub and Van Loan [18], and Stewart [57].

where the term $\mathbf{A}^{-1}\mathbf{e}$ is the inverted noise. Equation (1.3) in the previous section is a special case of this equation. The important observation here is that the deblurred image consists of two components: the first component is the exact image, and the second component is the inverted noise. If the deblurred image looks unacceptable, it is because the inverted noise term contaminates the reconstructed image.

Important insight about the inverted noise term can be gained using the singular value decomposition (SVD), which is the tool-of-the-trade in matrix computations for analyzing linear systems of equations. The SVD of a square matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ is essentially unique and is defined as the decomposition

$$\mathbf{A} = \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^T,$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, satisfying $\mathbf{U}^T\mathbf{U} = \mathbf{I}_N$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}_N$, and $\mathbf{\Sigma} = \text{diag}(\sigma_i)$ is a diagonal matrix whose elements σ_i are nonnegative and appear in nonincreasing order,

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_N \geq 0.$$

The quantities σ_i are called the singular values, and the rank of $\mathbf{A}$ is equal to the number of positive singular values. The columns $\mathbf{u}_i$ of $\mathbf{U}$ are called the left singular vectors, while the columns $\mathbf{v}_i$ of $\mathbf{V}$ are the right singular vectors. Since $\mathbf{U}^T\mathbf{U} = \mathbf{I}_N$, we see that $\mathbf{u}_i^T\mathbf{u}_j = 0$ if $i \neq j$, and, similarly, $\mathbf{v}_i^T\mathbf{v}_j = 0$ if $i \neq j$.

Assuming for the moment that all singular values are strictly positive, it is straightforward to show that the inverse of $\mathbf{A}$ is given by

$$\mathbf{A}^{-1} = \mathbf{V}\,\mathbf{\Sigma}^{-1}\mathbf{U}^T$$

(we simply verify that $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_n$). Since $\mathbf{\Sigma}$ is a diagonal matrix, its inverse $\mathbf{\Sigma}^{-1}$ is also diagonal, with entries $1/\sigma_i$ for $i = 1, \ldots, N$.

Another representation of $\mathbf{A}$ and $\mathbf{A}^{-1}$ is also useful to us:

$$\begin{aligned}
\mathbf{A} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \\
&= \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_N \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_N \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_N^T \end{bmatrix} \\
&= \mathbf{u}_1\sigma_1\mathbf{v}_1^T + \cdots + \mathbf{u}_N\sigma_N\mathbf{v}_N^T \\
&= \sum_{i=1}^{N} \sigma_i \mathbf{u}_i\,\mathbf{v}_i^T.
\end{aligned}$$

Similarly,

$$\mathbf{A}^{-1} = \sum_{i=1}^{N} \frac{1}{\sigma_i}\mathbf{v}_i\,\mathbf{u}_i^T.$$

Using this relation, it follows immediately that the naïve reconstruction given in (1.5) can be written as

$$\mathbf{x}_{\text{naïve}} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{V}\,\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{b} = \sum_{i=1}^{N} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\,\mathbf{v}_i \tag{1.6}$$

and the inverted noise contribution to the solution is given by

$$\mathbf{A}^{-1}\mathbf{e} = \mathbf{V}\,\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{e} = \sum_{i=1}^{N} \frac{\mathbf{u}_i^T\mathbf{e}}{\sigma_i}\,\mathbf{v}_i\,. \tag{1.7}$$

In order to understand when this error term dominates the solution, we need to know that the following properties generally hold for image deblurring problems:

- The error components $|\mathbf{u}_i^T\mathbf{e}|$ are small and typically of roughly the same order of magnitude for all i.
- The singular values decay to a value very close to zero. As a consequence the condition number
$$\text{cond}(\mathbf{A}) = \sigma_1/\sigma_N \tag{1.8}$$
is very large, indicating that the solution is very sensitive to perturbation and rounding errors.
- The singular vectors corresponding to the smaller singular values typically represent higher-frequency information. That is, as i increases, the vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ tend to have more sign changes.

The consequence of the last property is that the SVD provides us with basis vectors $\mathbf{v}_i$ for an expansion where each basis vector represents a certain "frequency," approximated by the number of times the entries in the vector change signs.

Figure 1.4 shows images of some of the singular vectors $\mathbf{v}_i$ for the blur of Figure 1.2. Note that each vector $\mathbf{v}_i$ is reshaped into an $m \times n$ array $\mathbf{V}_i$ in such a way that we can write the naïve solution as

$$\mathbf{X}_{\text{naïve}} = \sum_{i=1}^{N} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\,\mathbf{V}_i\,.$$

All the $\mathbf{V}_i$ arrays (except the first) have negative elements and therefore, strictly speaking, they are not images. We see that the spatial frequencies in $\mathbf{V}_i$ increase with the index i.

When we encounter an expansion of the form $\sum_{i=1}^{N} \xi_i\,\mathbf{v}_i$, such as in (1.6) and (1.7), then the ith expansion coefficient ξ_i measures the contribution of $\mathbf{v}_i$ to the result. And since each vector $\mathbf{v}_i$ can be associated with some "frequency," the ith coefficient measures the amount of information of that frequency in our image.

Looking at the expression (1.7), for $\mathbf{A}^{-1}\mathbf{e}$ we see that the quantities $\mathbf{u}_i^T\mathbf{e}/\sigma_i$ are the expansion coefficients for the basis vectors $\mathbf{v}_i$. When these quantities are small in magnitude, the solution has very little contribution from $\mathbf{v}_i$, but when we divide by a small singular value such as σ_N, we greatly magnify the corresponding error component, $\mathbf{u}_N^T\mathbf{e}$, which in turn contributes a large multiple of the high-frequency information contained in $\mathbf{v}_N$ to

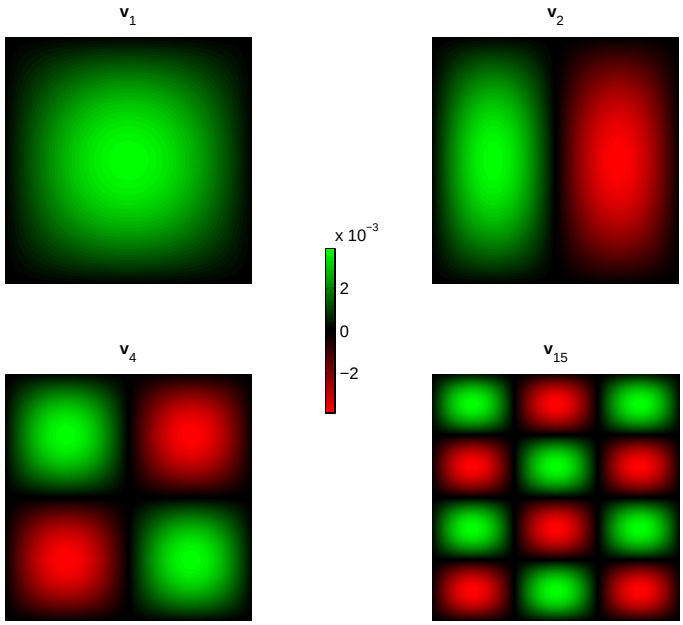


Figure 1.4. *A few of the singular vectors for the blur of the pumpkin image in Figure* 1.2. *The "images" shown in this figure were obtained by reshaping the $mn \times 1$ singular vectors $\mathbf{v}_i$ into $m \times n$ arrays.*

the computed solution. This is precisely why a naïve reconstruction, such as the one in Figure 1.3, appears as a random image dominated by high frequencies.

Because of this, we might be better off leaving the high-frequency components out altogether, since they are dominated by error. For example, for some choice of $k < N$ we can compute the truncated expansion

$$\mathbf{x}_k = \sum_{i=1}^{k} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i \equiv \mathbf{A}_k^\dagger \mathbf{b}$$

in which we have introduced the rank-k matrix

$$\mathbf{A}_k^\dagger = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{u}_1^T \\ \vdots \\ \mathbf{u}_k^T \end{bmatrix} = \sum_{i=1}^{k} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^T.$$

Figure 1.5 shows what happens when we replace $\mathbf{A}^{-1}\mathbf{b}$ by $\mathbf{x}_k$ with $k = 800$; this reconstruction is much better than the naïve solution shown in Figure 1.3. We may wonder if a different value for k will produce a better reconstruction!

The truncated SVD expansion for $\mathbf{x}_k$ involves the computation of the SVD of the large $N \times N$ matrix $\mathbf{A}$, and is therefore computationally feasible only if we can find fast algorithms

Figure 1.5. *The reconstruction* $\mathbf{x}_k$ *obtained for the blur of the pumpkins of Figure* 1.2 *by using* $k = 800$ *(instead of the full* $k = N = 169744$*).*

to compute the decomposition. Before analyzing such ways to solve our problem, though, it may be helpful to have a brief tutorial on manipulating images in MATLAB, and we present that in the next chapter.

VIP 2. We model the blurring of images as a linear process characterized by a blurring matrix $\mathbf{A}$ and an observed image $\mathbf{B}$, which, in vector form, is $\mathbf{b}$. The reason $\mathbf{A}^{-1}\mathbf{b}$ cannot be used to deblur images is the amplification of high-frequency components of the noise in the data, caused by the inversion of very small singular values of $\mathbf{A}$. Practical methods for image deblurring need to avoid this pitfall.

CHALLENGE 4. For the simple model $\mathbf{B} = \mathbf{A}_c\,\mathbf{X}\,\mathbf{A}_r^T + \mathbf{E}$ in Sections 1.2 and 1.3, let us introduce the two rank-k matrices $(\mathbf{A}_c)_k^\dagger$ and $(\mathbf{A}_r)_k^\dagger$, defined similarly to $\mathbf{A}_k^\dagger$. Then for $k < \min(m, n)$ we can define the reconstruction

$$\mathbf{X}_k = (\mathbf{A}_r)_k^\dagger\,\mathbf{B}\left((\mathbf{A}_c)_k^\dagger\right)^T.$$

Use this approach to deblur the image from Challenge 2. Can you find a value of k such that you can read the text?