Deblurring with Framelets in the Sparse Analysis Setting

by

Travis Danniels
B.Eng., University of Victoria, 2010

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Deblurring with Framelets in the Sparse Analysis Setting

by

Travis Danniels
B.Eng., University of Victoria, 2010

Supervisory Committee

Dr. T. Aaron Gulliver, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Michael McGuire, Departmental Member
(Department of Electrical and Computer Engineering)

Supervisory Committee

Dr. T. Aaron Gulliver, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Michael McGuire, Departmental Member
(Department of Electrical and Computer Engineering)

ABSTRACT

In this thesis, algorithms for blind and non-blind motion deblurring of digital images are proposed. The non-blind algorithm is based on a convex program consisting of a data fitting term and a sparsity-promoting regularization term. The data fitting term is the squared ℓ_2 norm of the residual between the blurred image and the latent image convolved with a known blur kernel. The regularization term is the ℓ_1 norm of the latent image under a wavelet frame (framelet) decomposition. This convex program is solved with the first-order primal-dual algorithm proposed by Chambolle and Pock in [19]. The proposed blind deblurring algorithm is based on the work of Cai, Ji, Liu, and Shen [13]. It works by embedding the proposed non-blind algorithm in an alternating minimization scheme and imposing additional constraints in order to deal with the challenging non-convex nature of the blind deblurring problem. Numerical experiments are performed on artificially and naturally blurred images, and both proposed algorithms are found to be competitive with recent deblurring methods.

Contents

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

Aaron Gulliver, for our many enjoyable chats about research and not research. I always left your office more inspired than when I had entered.

My family, for a lifetime of encouragement and support.

My friends, for tolerating me and keeping me sane.

DEDICATION

To my parents

Chapter 1

Introduction

Image restoration is the class of signal processing problems in which, given a degraded image, one attempts to undo the degradation and recover the original image. This is in contrast to *image enhancement*, where one wishes to modify an image in order to improve it according to subjective criteria. The key difference between the two problem classes is that image restoration problems are *inverse problems*, and thus require models of how the original image was transformed into the degraded image, while image enhancement problems in general do not.

Image restoration techniques have been studied since at least the 1950s. They were born out of necessity when, during the United States and Soviet space programs, astronomical images were being captured and transmitted at great cost [5]. These images were often degraded by quite severe noise and blurring, and, given their immense scientific value, needed to be restored to a clearer state.

Since then, image restoration has grown into a deep and well-studied field, with thousands of publications. Commonly studied image restoration problems include denoising, deblurring, inpainting, and super-resolution. An additional distinction can be made between *non-blind* and *blind* image restoration problems. In non-blind problems, the specific instance of the applied degradation operator (e.g. a blur kernel in the case of deblurring) is assumed to be known, while in blind problems, only statistical assumptions can be made about the degradation operator. In this thesis, we restrict our attention to the problems of non-blind and blind deblurring.

The process by which images are blurred is frequently assumed in the literature to be spatially invariant, implying a uniformly blurred image. This yields a model in which a blurry image y is obtained by a two-dimensional convolution between a sharp image x and a blur kernel h (also called a point spread function). In the case of non-blind deblurring, y and h are given as inputs to an algorithm which then attempts to deconvolve its inputs to obtain x. In the case of blind deblurring,

only y is known, making the task of recovering x significantly more difficult. The problem is further complicated by the presence of noise, rendering naïve methods (e.g. direct inversion of the convolution matrix) ineffective due to noise amplification. For this reason, *regularization* is frequently employed in order to obtain a better-posed problem. In this thesis, we propose algorithms utilizing sparsity-promoting regularization via wavelet frame image decomposition.

1.1 Related Work

Classical approaches to non-blind deblurring include inverse, least-squares, Wiener, and Kalman filters [8]. The regularization-based approach to the solution of inverse problems was largely popularized by the work of Tikhonov [51]. Early approaches to regularization-based deblurring tended to focus on ℓ_2 norm regularization functionals, which had difficulty recovering the sharp edges frequently found in natural images. (Throughout this thesis the term *functional* is used to denote a function that maps from a vector space to its underlying scalar field, usually from $\mathbb{R}^n \to \mathbb{R}$.)

A significant breakthough in regularization-based image restoration came with the popularization by Rudin, Osher, and Fatemi [47] of the total variation (TV) as a regularization functional for the image denoising problem. This result is highly relevant to our proposed deblurring methods, as not only can TV regularization be viewed as a form of ℓ_1 regularization, but also as an analysis-type regularization – two techniques which feature prominently in our proposed algorithms (see Sections 3.1 and 3.2).

Following the popularization of the TV norm for denoising, approaches applying it to deblurring followed (e.g. [21]). TV-regularized deblurring is still an area of active research, and many recent efforts have focused on developing faster algorithms for solving this challenging, non-smooth convex optimization problem. Recent examples include FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) [7, 6], split Bregman iterations [33, 11] and primal-dual Arrow-Hurwicz-type algorithms [55, 19]. Additionally, these algorithms allow for more general regularizers than the total variation; in particular, they allow for framelet-ℓ_1 regularization as used in this thesis.

Study of the blind image deblurring problem dates back to at least 1975 [49]. Early approaches to blind deblurring include iterative refinement of the image and blur kernel estimates in the frequency and spatial domains [4], and alternating applications of the Richardson-Lucy non-blind deconvolution algorithm [31]. Following TV regularization for non-blind deblurring, blind TV-regularized algorithms were proposed [23, 54]. The previous two algorithms employ alternating minimization between image and kernel estimates with TV-regularized inner iterations. More

recent approaches have employed different objective functions and/or introduced faster inner solvers, such as in [36, 13, 38]. The blind deblurring portion of this thesis builds on the work of [13] in particular.

1.2 Our Contributions

The intent of this thesis is to present algorithms for the efficient and accurate solution of blind and non-blind motion deblurring problems.

The proposed algorithms are primarily based on work by Chambolle and Pock [19], and Cai, Ji, Liu, and Shen [13]. For the proposed non-blind deblurring algorithm, we apply the algorithm of [19] to a least squares problem regularized by the ℓ_1 norm of the latent image's framelet decomposition. The proposed blind deblurring algorithm is based on the alternating minimization problem presented in [13], and replaces its inner solvers with the algorithm of [19].

The thesis is structured as follows:

In Chapter 2 we describe the deblurring problem, explain why naïve approaches do not tend to work well, and discuss the effects of boundary conditions.

In Chapter 3 we briefly review the mathematical background required to understand the proposed algorithms. This includes sections on ill-posed linear inverse problems, sparsity, convex optimization, and framelets.

In the first half of Chapter 4 we derive our proposed non-blind deblurring algorithm. The second half of the chapter is devoted to the comparison of experimental results obtained from the proposed algorithm and other related algorithms.

Chapter 5 takes the same form as Chapter 4, but with its focus shifted to blind deblurring.

Chapter 6 summarizes the results of the thesis, presents conclusions as to the relative performance of and appropriate use cases for the proposed algorithms, and discusses potential directions for future research.

Chapter 2

The Deblurring Problem

Deblurring is the problem in which, given a noisy, blurred image, one wishes to recover the original (sharp) image. The physical blurring process may be modelled [17] as a two-dimensional convolution

$$\begin{aligned} y(s,t) &= h(s,t) * x(s,t) + w(s,t) \\ &= \int_{\mathbb{R}^2} h(s-\bar{s}, t-\bar{t}) x(\bar{s},\bar{t}) \, \mathrm{d}\bar{s} \, \mathrm{d}\bar{t} + w(s,t) \end{aligned} \tag{2.1}$$

where x represents a grayscale image, h represents a blur kernel (also called a "point-spread function" or PSF), w is a sample from some noise process, and y represents the blurred noisy image. All of these functions map from $\mathbb{R}^2$ to $\mathbb{R}$.

To facilitate numerical solution, the continuous model above is discretized as [35]

$$Y = X * h + W \tag{2.2}$$

where $W, X, Y \in \mathbb{R}^{m \times n}$, $h \in \mathbb{R}^{k \times k}$, and $*$ is a discrete two-dimensional convolution operator. Note that h is assumed to be square – this does not result in any loss of generality, as it is always possible to zero-pad a non-square h in order to make it square.

For notational and implementational convenience we prefer to express (2.2) in matrix form

$$y = Hx + w \tag{2.3}$$

where $w, x, y \in \mathbb{R}^N$, $N = mn$, and $H \in \mathbb{R}^{N \times N}$ is a two-dimensional convolution matrix structured according to some boundary condition. Boundary conditions and the construction of H are discussed in Section 2.2. Throughout this thesis we refer

to h as the blur *kernel* and H as the blur *operator*.

Note that in (2.3), w, x and y are vectors. The vector forms of these images are obtained by lexicographically stacking the columns of their respective matrix representations. For example, a 3×3 matrix

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

would be vectorized as

$$\begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{12} & a_{22} & a_{32} & a_{13} & a_{23} & a_{33} \end{bmatrix}^T$$

Two short remarks on the assumptions built into model (2.3). First, note that due to its linearity, (2.3) assumes a spatially-invariant blur. In practice this may not always be the case due to, e.g., camera rotation during image acquisition or focal blur. Second, w is assumed to be zero mean, additive white Gaussian noise (AWGN) with variance σ^2. We restrict our attention to the zero mean AWGN case for simplicity. Other noise models could of course be considered.

One may in fact obtain two different flavours of deblurring problems from (2.3) based on whether or not h is assumed to be a known quantity. When h is known *a priori*, the resulting inverse problem is called *non-blind deblurring*. When h is unknown, the problem is called *blind deblurring*. We consider approaches to both types of deblurring in this work.

The continuous model in (2.1) is known to yield an ill-posed inverse problem [35]. That is to say, direct solutions of (2.3) for x are extremely sensitive to slight changes in y. The ill-posedness of (2.1) translates to H in (2.3) being ill-conditioned, i.e., having a high condition number. Informally, the condition number of a matrix A represents an upper bound on the achievable accuracy of a solution to $Ax = b$, given that computations are performed with finite precision. We further describe the effects of an ill-conditioned H below.

Consider the model in (2.3). Suppose the w term was identically 0, i.e., there was no noise present. Denote the noiseless observation y by y_0, i.e., $y_0 = Hx$. We could then obtain x by inverting H, i.e. $H^{-1}y_0 = x$. However, if we attempted this inversion in the case that $w \neq 0$, we would instead obtain $x' = H^{-1}(y_0 - w) = x - H^{-1}w$. That is, we obtain the original image x corrupted by the inverted noise $H^{-1}w$. Next, we describe why $H^{-1}w$ is frequently large in magnitude relative to x.

If we take the singular value decomposition (SVD) of a square matrix H, we obtain $H = U\Sigma V^T$, where U and $V \in \mathbb{R}^{N \times N}$ are orthogonal matrices, and $\Sigma = \text{diag}(\sigma_i)$ is

an $N \times N$ diagonal matrix of H's singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_N \geq 0$. The condition number of H is defined as σ_1/σ_N, the ratio of H's largest to H's smallest singular value. Additionally, it is a well-known property of the SVD that H^{-1} may be expressed as $H^{-1} = V\Sigma^{-1}U^T$. Since Σ is diagonal, Σ^{-1} is as well, and is given by $\Sigma^{-1} = \text{diag}(1/\sigma_i)$.

It is not hard to verify that the SVD of H can also be represented as $H = \sum_{i=1}^{N} \sigma_i U_i V_i^T$ where U_i and V_i denote the i'th columns of U and V, respectively. In a similar fashion

$$H^{-1} = \sum_{i=1}^{N} \frac{1}{\sigma_i} V_i U_i^T \tag{2.4}$$

With this in mind, we may write

$$\begin{aligned} x' &= x - H^{-1}w \\ &= x - \sum_{i=1}^{N} \frac{U_i^T w}{\sigma_i} V_i \end{aligned} \tag{2.5}$$

From (2.5) it is clear that the noise term w is amplified by the reciprocal of the singular values σ_i. Given that in general the singular values of blur operators H tend rapidly towards zero [35], the amplified noise term tends to dominate the actual solution x.

Rather than attempting a direct solution of (2.3), a popular approach is to introduce additional information on the structure of the required solution x (and h in the blind case) through *regularization*, a topic that we discuss in depth in Chapter 3.

2.1 Another Perspective

The problem of solving the non-blind form of (2.3) can be considered from a probabilistic perspective. In the absence of any *a priori* knowledge of x, we may consider a maximum-likelihood approach – that is, we wish to find the x which leads to the most probable observation y. First, the probability of a single sample w_i of the noise vector w is given by

$$P(w_i) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} w_i^2\right) \tag{2.6}$$

Then, the probability of a single sample y_i of the observation y, given the original image x, is

$$P(y_i|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} (y_i - (Hx)_i)^2\right) \tag{2.7}$$

By independence, the probability of the entire observation y is

$$P(y|x) = \prod_{i=1}^{M} P(y_i|x) \tag{2.8}$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{M}(y_i - (Hx)_i)^2\right) \tag{2.9}$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\|y - Hx\|_2^2\right) \tag{2.10}$$

Finally, we obtain the maximum-likelihood estimate x_{ML} of x as

$$x_{ML} = \arg\max_x P(y|x) \tag{2.11}$$

$$= \arg\min_x \|y - Hx\|_2^2 \tag{2.12}$$

There are some problems with this approach. For example, consider the case when there is in fact no blur at all, i.e., $H = I$. The minimizer x_{ML} of (2.12) in this case is $x_{ML} = y$. This would not be a problem if y was a noiseless observation of x; however, in practical cases some noise is always present. Thus, a practical deblurring algorithm not only needs to deblur, but it should denoise as well. This failing of the ML approach further motivates the development of the techniques described in Chapter 3.

2.2 Boundary Conditions

An important decision in the design of a deblurring algorithm relates to the handling of boundary conditions. In particular, when an image is naturally blurred due to, e.g., camera shake, the scene outside the image boundary will "spill over" and affect the values within the boundary. In order to avoid boundary artifacts, deblurring algorithms must be designed with this in mind and have their blurring models adjusted accordingly. As we will see, the choice of boundary conditions turns out to be a trade-off between restoration accuracy and computational convenience.

Throughout this section we will refer to an example with a 3×3 image x and a 3×3 blur kernel h. A blur operator H will be constructed from h and applied to x to obtain a blurred image y.

(a) Original "Lena" image. The white box indicates the image boundary

(b) Zero boundary condition

(c) Periodic boundary condition

(d) Reflective boundary condition

Figure 2.1: An illustration of different boundary conditions.

2.2.1 Zero Boundary Conditions

The simplest boundary condition is the "zero" boundary condition, in which one simply assumes that the scene beyond the image boundary is entirely zero-valued. An image X's zero boundary-extended image X' may be represented as

$$X' = \begin{bmatrix} 0 & 0 & 0 \\ 0 & X & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.13}$$

where the 0s represent 0-valued block matrices.

Blur operators H constructed from a blur kernel h according to a zero boundary condition have a block-Toeplitz structure with Toeplitz blocks (BTTB) [35]. That is, the left-to-right descending diagonals of H are blockwise constant, and each block of H has constant left-to-right descending diagonals. The blur operator for the 3×3 example is

$$\begin{bmatrix} y_{11} \\ y_{21} \\ y_{31} \\ y_{12} \\ y_{22} \\ y_{32} \\ y_{13} \\ y_{23} \\ y_{33} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} h_{22} & h_{12} & & h_{21} & h_{11} & & & & \\ h_{32} & h_{22} & h_{12} & h_{31} & h_{21} & h_{11} & & & \\ & h_{32} & h_{22} & & h_{31} & h_{21} & & & \\ \hline h_{23} & h_{13} & & h_{22} & h_{12} & & h_{21} & h_{11} & \\ h_{33} & h_{23} & h_{13} & h_{32} & h_{22} & h_{12} & h_{31} & h_{21} & h_{11} \\ & h_{33} & h_{23} & & h_{32} & h_{22} & & h_{31} & h_{21} \\ \hline & & & h_{23} & h_{13} & & h_{22} & h_{12} & \\ & & & h_{33} & h_{23} & h_{13} & h_{32} & h_{22} & h_{12} \\ & & & & h_{33} & h_{23} & & h_{32} & h_{22} \end{array}\right] \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix} \tag{2.14}$$

Unfortunately, the zero boundary condition is not a realistic model of natural image boundaries except in the case of, e.g., astronomical images with entirely black backgrounds. This motivates the discussion of the next two boundary conditions.

2.2.2 Periodic Boundary Conditions

The "periodic" (or circular) boundary condition is a boundary condition in which the image X is assumed to repeat in all directions as below.

$$X' = \begin{bmatrix} X & X & X \\ X & X & X \\ X & X & X \end{bmatrix} \tag{2.15}$$

Blur operators constructed according to a periodic boundary condition have a block-circulant structure with circulant blocks (BCCB) [35]. A circulant matrix is a Toeplitz matrix (see Section 2.2.1) with the additional condition that each row is a right circular shift of the previous row (with the first row being a shift of the last). In this case the blur operator for the 3×3 example is

$$\begin{bmatrix} y_{11} \\ y_{21} \\ y_{31} \\ y_{12} \\ y_{22} \\ y_{32} \\ y_{13} \\ y_{23} \\ y_{33} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} h_{22} & h_{12} & h_{32} & h_{21} & h_{11} & h_{31} & h_{23} & h_{13} & h_{33} \\ h_{32} & h_{22} & h_{12} & h_{31} & h_{21} & h_{11} & h_{33} & h_{23} & h_{13} \\ h_{12} & h_{32} & h_{22} & h_{11} & h_{31} & h_{21} & h_{13} & h_{33} & h_{23} \\ \hline h_{23} & h_{13} & h_{33} & h_{22} & h_{12} & h_{32} & h_{21} & h_{11} & h_{31} \\ h_{33} & h_{23} & h_{13} & h_{32} & h_{22} & h_{12} & h_{31} & h_{21} & h_{11} \\ h_{13} & h_{33} & h_{23} & h_{12} & h_{32} & h_{22} & h_{11} & h_{31} & h_{21} \\ \hline h_{21} & h_{11} & h_{31} & h_{23} & h_{13} & h_{33} & h_{22} & h_{12} & h_{32} \\ h_{31} & h_{21} & h_{11} & h_{33} & h_{23} & h_{13} & h_{32} & h_{22} & h_{12} \\ h_{11} & h_{31} & h_{21} & h_{13} & h_{33} & h_{23} & h_{12} & h_{32} & h_{22} \end{array}\right] \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix} \tag{2.16}$$

A major benefit of assuming a blur operator with a periodic boundary condition is that it permits efficient operations – notably convolution – via the FFT. This is due to the fact that BCCB matrices are diagonalized by the two-dimensional discrete Fourier transform (DFT). To see why this is so, we begin with the fact that all BCCB matrices H are normal [35] (i.e., $H^*H = HH^*$). This implies that H has a unitary spectral decomposition – in this case, it is obtained via the two-dimensional DFT. We thus may write $H = F^*\Lambda F$, where F is a two-dimensional DFT matrix and Λ is a diagonal matrix of H's eigenvalues. The eigenvalues of H are easily computed as the first column of F is a column of all ones scaled by $\frac{1}{\sqrt{N}}$. Denoting the first columns of F and H by F_1 and H_1, we have, by unitarity of the 2D DFT

$$H = F^*\Lambda F \tag{2.17}$$
$$\iff FH = \Lambda F \tag{2.18}$$
$$\iff FH_1 = \Lambda F_1 \tag{2.19}$$
$$= \frac{\lambda}{\sqrt{N}} \tag{2.20}$$

where λ is a vector of the eigenvalues of H.

Next we describe why it is not actually necessary to form the matrix H in order to perform operations such as convolution. Consider a practical case involving a MATLAB implementation. MATLAB's fast 2D DFT function, `fft2`, takes its argument in the form of a matrix. If we place the first column of H in example

(2.16) column-wise into matrix form, we obtain

$$\text{matrix}\,(H_1) = \begin{bmatrix} h_{22} & h_{32} & h_{12} \\ h_{23} & h_{33} & h_{13} \\ h_{21} & h_{31} & h_{11} \end{bmatrix} \tag{2.21}$$

Note that matrix (H_1) can be obtained from h by two circular shifts of length 1: one leftwards, and one upwards. This holds true for a general $N \times N$ BCCB H, in which case each shift is of length $\lfloor N/2 \rfloor$. This may be achieved in MATLAB through use of the `circshift` function. Thus Λ, the spectrum of a BCCB matrix H, may be efficiently computed with the following MATLAB code:

```
Lambda = fft2(circshift(h, -floor(size(h)/2)))
```
(2.22)

Having obtained Λ without constructing H, the (circular) convolution $y = h * x$ (i.e., Hx) may be obtained according to the equalities

$$y = Hx \tag{2.23}$$
$$= F^* \Lambda F x \tag{2.24}$$

or, in MATLAB code

```
Lambda = fft2(circshift(h, -floor(size(h)/2)));
```
(2.25)
```
y = ifft2(Lambda .* fft2(x));
```
(2.26)

The ability to efficiently compute the convolution $h * x$ is highly desirable when deblurring large images. For this reason we assume a periodic boundary condition throughout this thesis. The main drawback to this choice is the fact that assuming a periodic boundary condition is not necessarily realistic for all images, particularly those with significant high-frequency content around the edges. If care isn't taken, artifacts can be introduced when deblurring images under a periodic boundary condition. This drawback motivates the discussion of a third boundary condition in the next section.

2.2.3 Symmetric Boundary Conditions

The symmetric (also called reflexive or Neumann) boundary condition is the most complicated of the three types of boundary conditions we present. It assumes the scene beyond an image's boundaries is reflected along the corresponding edges, as in Figure 2.1. An image X's symmetric boundary-extended image X' may be

represented in matrix form as

$$X' = \begin{bmatrix} X_\times & X_\updownarrow & X_\times \\ X_\leftrightarrow & X & X_\leftrightarrow \\ X_\times & X_\updownarrow & X_\times \end{bmatrix} \tag{2.27}$$

where $X_\leftrightarrow$ denotes a horizontal reflection of X, $X_\updownarrow$ represents a vertical reflection of X, and $X_\times$ represents a combined horizontal and vertical reflection of X (a 180° rotation).

Blur operators constructed according to a symmetric boundary condition can be represented as a sum of four structured matrices. In particular, they have a BTTB + BTHB + BHTB + BHHB structure [35], where "T" represents a Toeplitz matrix as in Section 2.2.1, and "H" here indicates a Hankel matrix – a matrix with constant skew-diagonals. (A Hankel matrix is a vertically reflected Toeplitz matrix).

Blur operators H constructed according to a symmetric boundary condition can be diagonalized by the two-dimensional discrete cosine transform (DCT) provided the underlying blur kernel h satisfies certain symmetry conditions [44] – namely, that h has both horizontal and vertical symmetry, i.e. $h = h_\leftrightarrow$ and $h = h_\updownarrow$ (with the affordance that h need not be perfectly centered on its background of zero values). This in turn permits efficient convolution via the fast discrete cosine transform (FDCT), much in the same way that efficient matrix operations are made possible by the FFT in the case of symmetric boundary conditions.

Unfortunately, in the case of motion blurring, blur kernels cannot be assumed to possess the two-fold symmetry required for efficient FDCT-based operations. However, symmetric boundary conditions can still be a good choice due to their tendency to introduce fewer artifacts when deblurring general images as compared to periodic boundary conditions. The performance of algorithms employing both types of boundary conditions is assessed in Chapter 5 in the context of blind deblurring.

Chapter 3

Background

In this chapter, we briefly review the various mathematical and conceptual ingredients that form the basis of our proposed deblurring algorithms.

3.1 Solving Ill-Posed Linear Inverse Problems via Sparsity Regularization

In this section, we describe a fairly general approach to solving ill-posed linear inverse problems, a category into which deblurring fits. This approach will form the basis of our proposed deblurring method.

Hadamard defined a well-posed problem as possessing a unique solution which is not highly sensitive to slight changes in initial data [34]. As shown in Chapter 2, the deblurring problem possesses neither of these properties.

Consider the signal acquisition model of (2.3), $y = Hx + w$. Because H is assumed to be ill-conditioned, this problem is difficult to solve directly, as shown in Section 2. Additionally, the maximum likelihood approach to deblurring described in Section 2.1 fails to effectively handle noisy observations. For these reasons we employ regularization as an effective method of solving ill-posed linear inverse problems.

One popular approach to solving (2.3) is to introduce the regularized least-squares optimization problem

$$\min_x \frac{\lambda}{2}\|y - Hx\|_2^2 + R(x) \tag{3.1}$$

where $\lambda \in [0, +\infty)$ is a *regularization parameter* and $R : \mathbb{R}^N \to [0, +\infty)$ is a *regularization functional* that represents a prior on x (i.e., it assumes larger values for less probable values of x). Intuitively, we may think of this optimization problem

as trading off between two important qualities of a solution: fidelity and regularity. The *fidelity term* $\|y - Hx\|_2^2$ promotes a solution which makes sense in the context of the observation y, the operator H, and the distribution on the noise w. Note that the squared-ℓ_2 norm functional of this fidelity term reflects a Gaussian distribution on w, as discussed in Section 2.1. The regularization functional $R(x)$ promotes a solution that agrees with its associated prior, and serves to reduce the sensitivity of (3.1) to small changes in y. Though a slight abuse of terminology, throughout this thesis we shall use the terms "regularization functional" and "prior" interchangably when there is no room for confusion. The regularization parameter λ controls the degree of trade-off between these two properties, and should be set according to the amount of noise and expected regularity.

Many regularization functionals for (3.1) have been proposed over the years, including (Tikhonov regularization or ridge regression) $\|x\|_2^2$, (Basis Pursuit Denoising or LASSO) $\|x\|_1$, and (Rudin-Osher-Fatemi or Total Variation (TV)) $\|\nabla x\|_1$. The latter two regularization functionals are *sparsity promoting priors*, which are discussed in Section 3.2. In this thesis we focus on a generalization of TV regularization in which we replace the ∇ operator by a different analysis operator A – in particular, a framelet transform (discussed in Section 3.4).

3.2 Analysis and Synthesis Sparsity Priors

Central to our approach are the notions of *sparsity* and *compressibility*. A signal is called "sparse" if it is comprised mostly of zeros. A signal is said to be *compressible* if, while not necessarily sparse, the sorted magnitudes of its coefficients decay rapidly, e.g. according to a power law [28]. This implies that a compressible signal is well-approximated by its k largest-magnitude coefficients, and can thus be "sparsified" via thresholding with little loss in fidelity.

Recall the definition of the finite-dimensional ℓ_p norm, $\|\cdot\|_p : \mathbb{R}^N \to \mathbb{R}$ for $1 \leq p \leq \infty$

$$\|x\|_p \triangleq \left(\sum_{i=1}^{N} |x_i|^p \right)^{\frac{1}{p}} \tag{3.2}$$

The ℓ_0 "pseudo-norm" is then defined as $\|x\|_0 \triangleq |\{i \,|\, x_i \neq 0\}|$, i.e., $\|\cdot\|_0$ counts the number of non-zero entries in its argument. A signal $x \in \mathbb{R}^N$ is said to admit a sparse decomposition under an analysis operator $A \in \mathbb{R}^{M \times N}$ if $\|Ax\|_0 \ll M$. Similarly, x admits a sparse representation under the *synthesis operator* A^T if there exists $\alpha \in \mathbb{R}^M$ such that $A^T\alpha = x$ and $\|\alpha\|_0 \ll N$.

Note that A may be (and frequently is) *redundant*, i.e., $M > N$. For the remainder

of this section it is assumed that our analysis (synthesis) operators satisfy $A^TA = I$. Note that we do not also require $AA^T = I$, as this would imply that A is an orthogonal matrix.

It is well-known that natural (structured) images tend to admit compressible representations in bases such as those of the discrete Fourier, cosine, or wavelet transforms. This suggests an approach to solving the deblurring problem (2.3): incorporate the prior knowledge that the original image x is likely to be sparse (under some well-chosen analysis operator), by seeking the sparsest possible solution $\hat{x}$. Following this line of thought, consider the minimization problem

$$\begin{aligned} \hat{x} = \arg\min_x \|y - Hx\|_2^2 \\ \text{subject to} \quad \|Ax\|_0 \leq k \end{aligned} \tag{3.3}$$

where A is a sparsity-inducing analysis operator and k is some desired level of sparsity. Assuming a solution exists for the chosen k, and with a suitable choice of regularization parameter λ, (3.3) is equivalent to

$$\hat{x} = \arg\min_x \frac{\lambda}{2}\|y - Hx\|_2^2 + \|Ax\|_0 \tag{3.4}$$

As desired, (3.3) and (3.4) find solutions of (2.3) that admit sparse decompositions under A. There is however a serious drawback to the use of the ℓ_0 pseudo-norm as a sparsity-promoting functional: it renders (3.3) and (3.4) not only non-convex, but NP-hard [42], and thus (3.3) and (3.4) are likely to be computationally intractable. For this reason the ℓ_0 pseudo-norm is rarely used in this way.

A convexity-preserving alternative to using the ℓ_0 pseudo-norm as a sparsity-promoting functional is to use the ℓ_1 norm. Though it has been known since at least the 1970s that the ℓ_1 norm promotes sparsity [50], in recent years it has seen massively renewed interest due to the work of Donoho, Candès, et. al. in the field of compressed sensing (see e.g. [26, 16]). A heuristic explanation for why ℓ_1 regularization promotes sparsity over, e.g., ℓ_2 regularization is that the ℓ_1 norm places much more weight on small coefficients than the ℓ_2 norm. For this reason ℓ_1 regularization is likely to yield more zero coefficients than other ℓ_p norms for $p > 1$. The ℓ_1 norm is the most sparsity-promoting among the ℓ_p norms; ℓ_p pseudo-norms for $0 \leq p < 1$ yield sparser solutions, but are not convex, and thus create additional computational difficulties.

Two popular approaches to incorporating sparse priors into (3.1) are known as the *analysis* and *synthesis* sparsity models. Although in this thesis we focus exclusively on the analysis model, there are useful insights to be found in comparing the two.

In the synthesis sparsity setting, (3.1) becomes

$$\hat{\alpha} = \arg\min_{\alpha} \frac{\lambda}{2}\|y - HA^T\alpha\|_2^2 + \|\alpha\|_1 \tag{3.5}$$

where A^T is a synthesis operator and α is related to x by $x = A^T\alpha$. From this it can be seen where the synthesis sparsity model gets its name: in this model we seek a sparse coefficient vector α which synthesizes the signal x through $x = A^T\alpha$.

In the analysis sparsity setting, (3.1) becomes

$$\hat{x} = \arg\min_{x} \frac{\lambda}{2}\|y - Hx\|_2^2 + \|Ax\|_1 \tag{3.6}$$

where A is an analysis operator. In contrast to the synthesis model, in the analysis model it is assumed that the signal x admits a sparse representation under the analysis operator A.

Note that when A is an orthogonal matrix (i.e. $A^TA = AA^T = I$), (3.5) and (3.6) are equivalent [30]. However, when A is redundant, the results of the two regularizations are in general different.

Owing to the fact that $A^TA = I$, (3.6) may also be expressed as

$$\hat{u} = \arg\min_{u \in \text{range}(A)} \frac{\lambda}{2}\|y - HA^Tu\|_2^2 + \|u\|_1 \tag{3.7}$$

where $\hat{x} = A^T\hat{u}$. Comparing (3.5) and (3.7), we see that while the synthesis model minimizes over all $\alpha \in \mathbb{R}^M$, the analysis model only minimizes over A's column space. Assuming that $M > N$, the practical consequence of this difference is that while the synthesis model generally obtains sparser solutions than the analysis model owing to its less restricted search space, the analysis model can obtain solutions that better agree with the signal prior implied by A.

Historically, the synthesis model has received more attention than the analysis model (e.g. [1, 27]), though that is beginning to change as evidenced by e.g. [41, 52]. As stated by Elad in [29], "In recent years there is a growing interest in [the analysis model], and a growing belief that it encompasses a potential for better signal modeling [than the synthesis model]." For further information on the analysis and synthesis models, see e.g. [30, 48].

3.3 Convex Optimization

We now turn to the question of how to actually solve (3.6). First, it is highly relevant that (3.6) is *convex*. This allows the minimization to be solved efficiently even for large problem sizes. Though convex, due to the presence of the ℓ_1 term, the objective function of (3.6) is not smooth (i.e. not continuously differentiable), which precludes the use of more general algorithms due to their poor performance on nonsmooth problems. See, e.g., [10] for an introduction to the theory of convex optimization.

In this thesis we solve (3.6) with the first-order primal-dual algorithm of Chambolle and Pock [19]. In the remainder of this section we provide a brief overview of the algorithm of [19] and the theory required to use it.

We say that a set X in a vector space is *convex* if, for all x and y in X, and all $t \in [0, 1]$, the vector $(1 - t)x + ty$ is also in X. Intuitively, a set is convex if a line segment may be drawn between any two points in the set without having any point on the line segment lie outside the set.

Given a convex set X in a vector space, a function $f : X \to \mathbb{R}$ is said to be convex if, for any two points x_1, x_2 in X, and all $t \in [0, 1]$, $f((1 - t)x_1 + tx_2) \leq (1 - t)f(x_1) + tf(x_2)$. That is, a function is convex if its epigraph (the set of points on or above a function's graph) is convex. A convex function is called *closed* if its epigraph is a closed set.

The *convex* (or *Fenchel*) *conjugate* f^* of a function f is defined as

$$f^*(y) = \sup_{x \in \mathrm{dom}(f)} \langle y, x \rangle - f(x) \tag{3.8}$$

where $\mathrm{dom}(f)$ is the domain of f. The Fenchel-Moreau theorem states [9] that $f^{**} = f$ if and only if f is a closed convex function.

We define the *indicator function* δ_S of a set S as

$$\delta_S(s) = \begin{cases} 0 & \text{if } s \in S \\ +\infty & \text{if } s \notin S \end{cases} \tag{3.9}$$

The *proximal mapping* of a closed convex function f is defined as

$$\mathrm{prox}_f(y) = \arg\min_x f(x) + \frac{1}{2}\|x - y\|_2^2 \tag{3.10}$$

The proximal mapping may be thought of as a generalization of the problem of

projecting a point y onto a convex set C

$$\min_x \delta_C(x) + \frac{1}{2}\|x - y\|_2^2 \tag{3.11}$$

For closed convex f, $\text{prox}_f(y)$ exists and is unique for all y [10].

More information on the theory of convex functions may be found in, e.g., [45, 9].

We now proceed to describe the algorithm of [19]. Consider the optimization problem

$$\min_{x \in \mathcal{X}} G(x) + F(Kx) \tag{3.12}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are real finite-dimensional inner product spaces, $K : \mathcal{X} \to \mathcal{Y}$ is a continuous linear operator, and $G : \mathcal{X} \to [0, +\infty)$ and $F : \mathcal{Y} \to [0, +\infty)$ are closed convex functionals. Denote by $\|K\|$ the induced operator norm

$$\max\{\|Kx\| \, : \, x \in \mathcal{X}, \|x\| \leq 1\} \tag{3.13}$$

Applying convex conjugation to F and G in (3.12), we obtain the dual formulation

$$\max_{y \in \mathcal{Y}} -(G^*(-K^*y) + F^*(y)) \tag{3.14}$$

Because F and G are closed and convex, the optimal values of (3.12) and (3.14) are the same – a condition called strong duality [9, Theorem 3.3.5].

If we only dualize (3.12) with respect to F, we instead obtain the primal-dual formulation

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} G(x) + \langle Kx, y \rangle - F^*(y) \tag{3.15}$$

The minimizer $(\hat{x}, \hat{y})$ of (3.15), if it exists, is called a *saddle-point*. The algorithm of Chambolle and Pock is capable of efficiently solving problems of this type. Its general form is given in Algorithm 1.

Algorithm 1 [19] Algorithm 1

Choose step sizes τ, $\sigma > 0$, $\theta \in [0, 1]$, arbitrary initial values $(x^0, y^0) \in \mathcal{X} \times \mathcal{Y}$, and set $\bar{x}^0 := x^0$.
for $n \geq 0$:

$$\begin{aligned} y^{n+1} &:= \operatorname{prox}_{\sigma F^*}(y^n + \sigma K \bar{x}^n) \\ x^{n+1} &:= \operatorname{prox}_{\tau G}(x^n - \tau K^* y^{n+1}) \\ \bar{x}^{n+1} &:= x^{n+1} + \theta(x^{n+1} - x^n) \end{aligned}$$

end for

Upon reaching the stopping criterion, the most recent iterate x^{n+1} is the output. Stopping criteria may include a maximum number of iterations and/or a measure of convergence.

Informally, the arguments $y^n + \sigma K \bar{x}^n$ and $x^n - \tau K^* y^{n+1}$ of the prox operators in Algorithm 1 may be understood as, respectively, performing gradient ascent in the dual variable y and gradient descent in the primal variable x. The prox operators serve to resolve the ambiguity brought about by the fact that F^* and G are not necessarily smooth and thus may not possess unique gradients at all points. The third update step of Algorithm 1 is required with $\theta = 1$ in [19] to prove fast convergence of the algorithm. When $\theta = 0$, Algorithm 1 corresponds to the classical Arrow-Hurwicz algorithm [3].

A notable advantage of solving the saddle-point problem (3.15) instead of the primal problem (3.12) is that at each iteration of the primal-dual solver, it is possible to obtain an estimate of how close the current iterate is to a solution. This is done by computing the *primal-dual gap*. Suppose Algorithm 1 has a current solution estimate (x^n, y^n), and the optimal solution is $(\hat{x}, \hat{y})$. Then the primal dual gap $\mathcal{G}$ is given by the difference between the primal and dual objectives

$$\mathcal{G}(x^n, y^n) \triangleq F(Kx^n) + G(x^n) + F^*(y^n) + G^*(-K^* y^n) \tag{3.16}$$

and due to strong duality, $\mathcal{G}(\hat{x}, \hat{y}) = 0$.

It is proven in [19] that for $\theta = 1$ and $\tau\sigma < \|K\|^{-2}$, the primal-dual gap of Algorithm 1 converges to zero at rate $O(1/n)$ (n being the iteration number), which is optimal for non-smooth convex programs whose objective functions have a known structure [43].

3.4 Frames, Wavelets, and Framelets

It is necessary to address one final ingredient in the solution to the non-blind deblurring problem (3.6), namely the form of the analysis operator A. Recall that the purpose of A is to (reversibly) map natural images from the spatial domain to a domain in which they are sparse. Additionally, less distorted images should map to sparser representations than more distorted images. There is a wide variety of candidates for this task: the classical discrete Fourier and cosine transforms, wavelets, or more recently introduced transforms such as curvelets or ridgelets ([15, 14]). In this thesis, wavelet frames are used to obtain sparse representations of images for use during the deblurring process. In the following sections we briefly discuss the theory and applications of frames, wavelets, and wavelet frames.

A *frame* may be thought of as a generalized basis in which there is permitted to be some redundancy among its elements. Formally, a collection of vectors $\Phi = \{\phi_i\}_{i \in I}$ indexed over a set I in a Hilbert space $\mathbb{H}$ is called a frame if there exist constants $0 < A \leq B < +\infty$ such that for all x in $\mathbb{H}$

$$A\|x\|_2^2 \leq \sum_{i \in I} |\langle x, \phi_i \rangle|^2 \leq B\|x\|_2^2 \tag{3.17}$$

where A and B are referred to as *frame bounds*. If $A = B$, the frame is *tight*. If $A = B = 1$, the frame is a *Parseval frame*, since in this case (3.17) reduces to

$$\|x\|_2^2 = \sum_{i \in I} |\langle x, \phi_i \rangle|^2 \tag{3.18}$$

which has the same form as the well-known Parseval identity. It is easy to show that any tight frame can be made into a Parseval frame by an appropriate rescaling of its elements. Note that in the case of a Parseval frame Φ, a vector $x \in \mathbb{H}$ has the expansion

$$x = \sum_{i \in I} \langle x, \phi_i \rangle \phi_i \tag{3.19}$$

which resembles the equation for the expansion of x in an orthonormal basis. This is called the *perfect reconstruction property*. There is, however, a key difference between frames and bases: the vectors that constitute a frame are not necessarily linearly independent, and thus in general do not form a basis. This leads to the fact that frame expansions are not necessarily unique. For example, consider a linearly dependent subset S of a frame Φ. Then there exist coefficients $\{c_i\}$, not all equal to zero, such that $\sum_{\phi_i \in S} c_i \phi_i = 0$. From this we can obtain an infinite number of expansions of x by adding multiples of c_i to $\langle x, \phi_i \rangle$ in (3.19). The expansion given in (3.19) is called the *canonical* expansion, as it has the minimum norm among all expansions of x. Interested readers are referred to [37] for a detailed introduction to

frame theory.

We will be particularly interested in Parseval frames due to their fast convergence properties when used in our implementation of Algorithm 1.

In the context of frame theory we define the analysis operator Φ^* as the matrix whose rows are the conjugate-transposed frame vectors ϕ_i^*. Its adjoint Φ is the synthesis operator. For a general frame, (3.17) implies $AI \leq \Phi\Phi^* \leq BI$. For a Parseval frame, it implies

$$\Phi\Phi^* = I \tag{3.20}$$

Note that as in Section 3.2, $\Phi^*\Phi \neq I$ in general. If it is the case that $\Phi\Phi^* = I$ and $\Phi^*\Phi = I$, then Φ^* is an orthogonal matrix.

We now briefly define wavelets. Let Ψ be a finite subset $\{\psi^1, \ldots, \psi^r\}$ of $L_2(\mathbb{R})$, the set of square-integrable functions. The *wavelet system* $X(\Psi)$ is then defined as a family of shifts and dilations of the elements of Ψ

$$X(\Psi) \triangleq \{\psi_{j,k} : j \in \mathbb{Z}, k \in \mathbb{Z}, \psi \in \Psi\} \tag{3.21}$$

where $\psi_{j,k} : x \mapsto 2^{j/2}\psi(2^j x - k)$. In this thesis we are interested in wavelet systems $X(\Psi)$ that are also tight frames. In this case we call $X(\Psi)$ a *wavelet tight frame*, and each $\psi \in \Psi$ is a *framelet*. A short summary of the theory and construction of wavelet tight frames is provided in the remainder of this section. For a rigorous exposition of these topics, see [25] and the references contained therein.

In order to construct a compactly supported wavelet frame, one may begin with a compactly supported function $\phi \in L_2(\mathbb{R})$ and a 2π-periodic trigonometric polynomial $\tau_0(\omega) = \sum_k h_0(k)e^{ik\omega}$ with $\tau_0(0) = 1$. τ_0 and ϕ are related by $\hat{\phi}(\omega) = \tau_0\hat{\phi}(\omega/2)$, where $\hat{\phi}$ denotes the Fourier transform of ϕ, which we define as

$$\hat{f}(\omega) = \int_{\mathbb{R}} f(x)e^{-i\omega x}\mathrm{d}x \tag{3.22}$$

The functions ϕ and τ_0 are respectively called a *refinable* or *scaling function* and a *refinement mask*.

One may construct a set of framelets according to the *unitary extension principle* (UEP) [25, 46]: given ϕ and τ_0 as previously described, the framelet system $X(\Psi)$ generated by $\Psi = \{\psi_1, \ldots, \psi_r\}$ forms a tight frame in $L_2(\mathbb{R})$ if there exists a set of 2π-periodic trigonometric polynomials $\{\tau_i\}_{1\leq i\leq r}$, called *wavelet masks*, which are defined for $1 \leq i \leq r$ by $\hat{\psi}_i(\omega) = \tau_i\hat{\psi}_i(\omega/2)$ and satisfy

$$\sum_{i=0}^{r} \tau_i(\omega)\tau_i^*(\omega + \nu) = \begin{cases} 1 & \text{if } \nu = 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.23}$$

for all $\nu \in \{0, \pi\}$ and almost all $\omega \in \mathbb{R}$.

The UEP implies that the construction of a set of framelets for a given refinable function ϕ and refinement mask τ_0 can be accomplished by finding a set of wavelet masks $\{\tau_i\}_{1 \leq i \leq r}$ that satisfy (3.23).

Note that as a consequence of the framelets discussed in this section being derived from a multiresolution analysis [39], the coefficients $h_0(k)$ of τ_0 represent a low-pass filter, and the coefficients $h_i(k)$ of $\{\tau_i\}_{1 \leq i \leq r}$ represent high-pass filters.

In order to apply wavelet frames to deblurring problems, we must first shift our focus from the infinite-dimensional function space $L_2(\mathbb{R})$ to the finite-dimensional space $\mathbb{R}^N$ in which digital images exist. Recall that a Parseval frame in matrix form has the perfect reconstruction property (3.20). Given the refinement and wavelet masks $\{\tau_i\}_{0 \leq i \leq r}$ (together called a combined MRA mask) for a framelet system $X(\Psi)$, one may construct the analysis operator $A \in R^{M \times N}$ $(M \geq N)$ which satisfies (3.20). This is discussed next.

Throughout this thesis we use framelet analysis operators A constructed from piecewise linear uniform B-splines according to the UEP. This particular construction was given as an example in [46, 18] and later used in, e.g., [12, 13]. In our experiments this construction was found to outperform more complex constructions such as those based on piecewise cubic B-splines not only in computational efficiency but also in visual fidelity. In the following paragraphs we derive these framelets via the UEP.

Begin by letting the refinable function $\phi(x)$ be the piecewise linear B-spline (triangle function)

$$\phi(x) = \max(1 - |x|, 0) \tag{3.24}$$

Its refinement mask τ_0 is given by

$$\tau_0(\omega) = \cos^2\left(\frac{\omega}{2}\right) \tag{3.25}$$

The Fourier transform of ϕ is the squared sinc function

$$\hat{\phi}(\omega) = \left(\frac{\sin(\frac{\omega}{2})}{\frac{\omega}{2}}\right)^2 \tag{3.26}$$

which indeed satisfies the requirement $\hat{\phi}(\omega) = \tau_0\hat{\phi}(\omega/2)$:

$$\tau_0\hat{\phi}\left(\frac{\omega}{2}\right) = \frac{\cos^2\left(\frac{\omega}{4}\right)\sin^2\left(\frac{\omega}{4}\right)}{\left(\frac{\omega}{4}\right)^2} \tag{3.27}$$

$$= \frac{\left(\frac{1+\cos\left(\frac{\omega}{2}\right)}{2}\right)\left(\frac{1-\cos\left(\frac{\omega}{2}\right)}{2}\right)}{\left(\frac{\omega}{4}\right)^2} \tag{3.28}$$

$$= \frac{\left(\frac{1-\cos^2\left(\frac{\omega}{2}\right)}{4}\right)}{\left(\frac{\omega}{4}\right)^2} \tag{3.29}$$

$$= \left(\frac{\sin\left(\frac{\omega}{2}\right)}{\frac{\omega}{2}}\right)^2 \tag{3.30}$$

$$= \hat{\phi}(\omega) \tag{3.31}$$

Next, consider τ_0's Fourier series

$$\tau_0(\omega) = \cos^2\left(\frac{\omega}{2}\right) \tag{3.32}$$

$$= \frac{1}{4}\left(e^{-i\omega} + 2 + e^{i\omega}\right) \tag{3.33}$$

from which we may directly obtain τ_0's low-pass filter coefficients

$$h_0 = \frac{1}{4}[1, 2, 1] \tag{3.34}$$

Similarly, consider two framelet masks τ_1 and τ_2 defined by

$$\tau_1 = -\frac{i\sqrt{2}}{2}\sin(\omega) \tag{3.35}$$

$$= \frac{\sqrt{2}}{4}\left(e^{-i\omega} - e^{i\omega}\right) \tag{3.36}$$

$$\tau_2 = -\sin^2\left(\frac{\omega}{2}\right) \tag{3.37}$$

$$= \frac{1}{4}\left(-e^{-i\omega} + 2 - e^{i\omega}\right) \tag{3.38}$$

with high-pass filter coefficients

$$h_1 = \frac{\sqrt{2}}{4}[1, 0, -1] \tag{3.39}$$

$$h_2 = \frac{1}{4}[-1, 2, -1] \tag{3.40}$$

It is easily verified that $\{\tau_i\}_{0\leq i\leq 2}$ satisfies the UEP. Having obtained the filter coefficients $\{h_i\}_{0\leq i\leq 2}$, convolution matrices $\mathcal{C}(h_i)_{0\leq i\leq 2}$ may be constructed according to some boundary condition (usually periodic or symmetric), as described in Section 2.2. An analysis operator A corresponding to a single-level univariate framelet decomposition without downsampling [12] is then given by

$$A = \begin{bmatrix} \mathcal{C}(h_0) \\ \mathcal{C}(h_1) \\ \mathcal{C}(h_2) \end{bmatrix} \tag{3.41}$$

Note that due to the fact that we intend to deblur images, we actually desire a bivariate (2D) analysis operator. This may be obtained from A via tensor products as described in [18]. Finally, note that we have restricted our attention to a single-level decomposition, as that is what is used throughout this thesis. Information on multi-level constructions may be found in the references.

Chapter 4

Non-Blind Deblurring

4.1 Algorithm

In this chapter we propose and derive a method of solving the non-blind deblurring problem of recovering a clear image x from a blurred, noisy image $y = x * h + w$, where the blur kernel h is a known parameter. Briefly, the proposed deblurring method is to solve a regularized least squares problem in the form of (3.6) using Algorithm 1, where the regularization functional is the ℓ_1 norm of the latent image's coefficients under a framelet decomposition. The remainder of this chapter is spent deriving this approach in detail, and providing experimental results and comparisons of the proposed method with other deblurring algorithms.

Recall the regularized least-squares optimization problem (3.1). This is the starting point for the following derivations. As previously mentioned, the choice was made to use the ℓ_1 norm of the image's framelet coefficients as the regularization functional of the optimization problem. Substituting this for $R(x)$ in (3.1), we obtain

$$\min_x \frac{\lambda}{2}\|y - Hx\|_2^2 + \|Ax\|_1 \tag{4.1}$$

where A is the framelet analysis operator constructed in Section 3.4.

In order to apply Algorithm 1 to (4.1), it is necessary to identify in (4.1) functions F and G as in (3.12). One such identification is as follows:

$$\min_x \underbrace{\frac{\lambda}{2}\|y - Hx\|_2^2}_{\text{G}} + \underbrace{\|Ax\|_1}_{\text{F}} \tag{4.2}$$

The next step towards being able to apply Algorithm 1 to (4.1) is to obtain its primal-dual formulation as in (3.15). To do this, it is necessary to know the convex conjugate $\|\cdot\|_1^*$ of $\|\cdot\|_1$. This is provided by Theorem 1.

Theorem 1 ([10])**.**

$$\|y\|_1^* = \sup_x \langle y, x\rangle - \|x\|_1 \tag{4.3}$$

$$= \begin{cases} 0 & \text{if } y \in P \\ +\infty & \text{if } y \notin P \end{cases} \tag{4.4}$$

$$= \delta_P(y) \tag{4.5}$$

where $P = \{p : \|p\|_\infty \leq 1\}$ and $\|p\|_\infty = \max_i |p_i|$.

Proof. ([10]) First, consider the case when $y \in P$, i.e. $\|y\|_\infty \leq 1$. Then, for all x, $\langle y, x\rangle \leq \|x\|_1$ since

$$\begin{aligned} \|y\|_\infty \leq 1 &\Rightarrow \forall i\ |y_i| \leq 1 \\ &\Rightarrow \forall i\ x_i y_i \leq |x_i| \\ &\Rightarrow \langle y, x\rangle = \sum_i x_i y_i \leq \sum_i |x_i| = \|x\|_1 \end{aligned}$$

Since equality is achieved when $x = 0$, $\sup_x \langle y, x\rangle - \|x\|_1 = 0$ for all $y \in P$.

Next, if $y \notin P$ ($\|y\|_\infty > 1$), we can find $\bar{x}$ such that $\|\bar{x}\|_1 \leq 1$ and $\langle y, \bar{x}\rangle > 1$ (e.g., $\bar{x}$ is the zero vector but with $\bar{x}_i = \operatorname{sign} y_i$ where i is the location of y's largest-magnitude element). Then, $\|y\|_1^* \geq \langle y, t\bar{x}\rangle - \|t\bar{x}\|_1 = t(\langle y, \bar{x}\rangle - \|\bar{x}\|_1) \to \infty$ as $t \to \infty$. □

We thus obtain the saddle-point formulation of (4.2) as

$$\min_x \max_u \frac{\lambda}{2}\|Hx - y\|_2^2 + \langle Ax, u\rangle - \delta_P(u) \tag{4.6}$$

In order to apply Algorithm 1 to (4.6), it is necessary to determine the proximal mappings for $\tau G(x) = \frac{\tau\lambda}{2}\|Hx - y\|_2^2$ and $\sigma F^*(u) = \sigma\delta_P(u)$. The proximal mapping for $\sigma F^*(u)$ is given by Theorem 2.

Theorem 2 ([19])**.**

$$\text{prox}_{\sigma F^*}(\tilde{u}) = \arg\min_u \frac{\|u - \tilde{u}\|_2^2}{2\sigma} + \delta_P(u) \tag{4.7}$$

$$\Rightarrow \text{prox}_{\sigma F^*}(\tilde{u})_i = \frac{\tilde{u}_i}{\max(1, |\tilde{u}_i|)} \tag{4.8}$$

Proof. Computing the proximal mapping of σF^* requires finding the vector u that (a) is in the set P, and (b) has the minimum Euclidean distance from $\tilde{u}$. That is, we seek the Euclidean projection Proj_P of $\tilde{u}$ onto P, the L_∞ unit ball.

First, note that the optimization problem (4.7) may be simplified and separated into scalar sub-problems of the form $\min_{|u_i|\leq 1}(u_i - \tilde{u}_i)^2$. Now consider the case when $|\tilde{u}_i| \leq 1$. Here we can achieve an objective function value of 0 by setting $u_i = \tilde{u}_i$. Next, if $|\tilde{u}_i| > 1$, the minimum value of the objective function is achieved by $u_i = \tilde{u}_i/|\tilde{u}_i|$, since any additional component of u_i not colinear with $\tilde{u}_i$ would only increase the value of $(u_i - \tilde{u}_i)^2$. Thus we obtain the solution (4.8) and the proof is complete. □

Next we derive the proximal mapping for $\tau G(x)$. Note that due to the fact that H is a convolution matrix constructed according to a circular boundary condition, we may write the matrix-vector multiplication Hx as a circular convolution $h * u$ where h is the kernel of H. This allows us to efficiently compute the proximal mapping of $\tau G(x)$ via an FFT-based method. The proximal mapping for $\tau G(x)$ is given by Theorem 3.

Theorem 3 ([19, 55])**.**

$$\text{prox}_{\tau G}(\tilde{x}) = \arg\min_x \frac{\|x - \tilde{x}\|_2^2}{2\tau} + \frac{\lambda}{2}\|Hx - y\|_2^2 \tag{4.9}$$

$$= \mathcal{F}^{-1}\left(\frac{\tau\lambda\mathcal{F}(y)\mathcal{F}(h)^* + \mathcal{F}(\tilde{x})}{\tau\lambda\mathcal{F}(h)^2 + 1}\right) \tag{4.10}$$

where * denotes complex conjugation, $\mathcal{F}$ and $\mathcal{F}^{-1}$ respectively denote the discrete Fourier and inverse discrete Fourier transforms, and the multiplications and divisions in (4.10) are performed element-wise.

Proof. Consider the gradient of the objective function in (4.9)

$$\nabla\left(\frac{\|x - \tilde{x}\|_2^2}{2\tau} + \frac{\lambda}{2}\|Hx - y\|_2^2\right) = \frac{x - \tilde{x}}{\tau} + \lambda H^T(Hx - y) \tag{4.11}$$

Setting the gradient to 0, solving for x, and taking the DFT and IDFT as necessary, we obtain

$$x = \tilde{x} - \tau\lambda H^T(Hx - y) \tag{4.12}$$
$$\Rightarrow \mathcal{F}(x) = \mathcal{F}(\tilde{x}) - \tau\lambda\,\mathcal{F}(h)^*(\mathcal{F}(h)\,\mathcal{F}(x) - \mathcal{F}(y)) \tag{4.13}$$
$$= \mathcal{F}(\tilde{x}) - \tau\lambda\,\mathcal{F}(h)^2\,\mathcal{F}(x) + \tau\lambda\,\mathcal{F}(h)^*\,\mathcal{F}(y) \tag{4.14}$$
$$\Rightarrow \mathcal{F}(x)(1 + \tau\lambda\,\mathcal{F}(h)^2) = \mathcal{F}(\tilde{x}) + \tau\lambda\,\mathcal{F}(h)^*\,\mathcal{F}(y) \tag{4.15}$$
$$\Rightarrow \mathcal{F}(x) = \frac{\mathcal{F}(\tilde{x}) + \tau\lambda\,\mathcal{F}(h)^*\,\mathcal{F}(y)}{1 + \tau\lambda\,\mathcal{F}(h)^2} \tag{4.16}$$
$$\Rightarrow x = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\tilde{x}) + \tau\lambda\,\mathcal{F}(h)^*\,\mathcal{F}(y)}{1 + \tau\lambda\,\mathcal{F}(h)^2}\right) \tag{4.17}$$

where the solution x is a minimizer of (4.9) due to the objective function being convex. □

As mentioned in [19], computing the proximal mapping above requires only one forward FFT and one inverse FFT per iteration, as all other quantities can be pre-computed.

Having obtained the necessary proximal mappings, Algorithm 1 may be modified to solve the non-blind deblurring problem (3.6) as in Algorithm 2.

Algorithm 2 Non-Blind Deblurring

Choose τ, $\sigma > 0$, $\theta \in [0, 1]$, $(x^0, u^0) \in \mathcal{X} \times \mathcal{Y}$, set $\bar{x}^0 := x^0$, and $n := 0$.
while $n \leq$ N_MAX:

$$\begin{aligned}
(u^{n+1})_i &:= \frac{(u^n + \sigma A\bar{x}^n)_i}{\max(1, |(u^n + \sigma A\bar{x}^n)_i|)} \\
x^{n+1} &:= \mathcal{F}^{-1}\left(\frac{\tau\lambda\mathcal{F}(y)\mathcal{F}(h)^* + \mathcal{F}(x^n - \tau A^T u^{n+1})}{\tau\lambda\mathcal{F}(h)^2 + 1}\right) \\
\bar{x}^{n+1} &:= x^{n+1} + \sigma(x^{n+1} - x^n) \\
n &:= n + 1
\end{aligned}$$

end while

As recommended in [19], we set $\theta = 1$. Note that due to A being a Parseval frame, $\|A\| = 1$, and setting $\tau = \sigma = 1$ guarantees convergence.

4.2 Numerical Experiments

4.2.1 Methodology

Numerical experiments were performed to assess the speed and deblurring performance of Algorithm 2. Two other deblurring algorithms were chosen for comparison: an implementation of Algorithm 2 with the only difference being that it uses the discrete gradient as its analysis operator A (i.e total variation regularization), and a solver for (4.1) based on a Split Bregman iteration [33]. The algorithm implementations were respectively drawn from the software release associated with [19] and The Bregman Cookbook [32]. Both of these algorithms are designed to efficiently solve non-smooth, ℓ_1-regularized convex optimization problems. The TV-regularized algorithm, being based on the same solver as that of Algorithm 2, was chosen to illustrate the differences in speed and deblurring performance brought about by choosing framelet regularization over TV regularization. The Split Bregman algorithm was chosen to highlight differences in speed between solvers.

All experiments were performed on a 2.66 GHz Intel Core 2 Duo processor with 2 GB of RAM, running 32-bit Linux. All experimental code was written in MATLAB. Each experiment was run three times, and the fastest execution time of each algorithm was recorded in order to account for the experiments being run on a pre-emptive multitasking operating system. In practice, little variation was observed between runs. Furthermore, the regularization parameter λ was hand-tuned for each algorithm in order to maximize, in the author's subjective opinion, each output's visual fidelity to the original image.

Images were blurred by random motion blurs generated by MATLAB's `fspecial` function according to a periodic boundary condition, and corrupted by zero mean AWGN with noise power $\sigma^2 = 10^{-4}$.

4.2.2 Results

(a) Original “cameraman” image

(b) Blurred “cameraman” image

Figure 4.1: Original and blurred “cameraman” images.

(a) Algorithm 2 Framelet; 1 iteration

(b) Algorithm 2 TV; 1 iteration

(c) Split Bregman Framelet; 1 iteration

Figure 4.2: Deblurring progress – 1 iteration.

(a) Algorithm 2 Framelet; 8 iterations

(b) Algorithm 2 TV; 8 iterations

(c) Split Bregman Framelet; 8 iterations

Figure 4.3: Deblurring progress – 8 iterations.

(a) Algorithm 2 Framelet; 64 iterations

(b) Algorithm 2 TV; 64 iterations

(c) Split Bregman Framelet; 64 iterations

Figure 4.4: Deblurring progress – 64 iterations.

(a) Algorithm 2 Framelet; 128 iterations

(b) Algorithm 2 TV; 128 iterations

(c) Split Bregman Framelet; 128 iterations

Figure 4.5: Deblurring progress – 128 iterations.

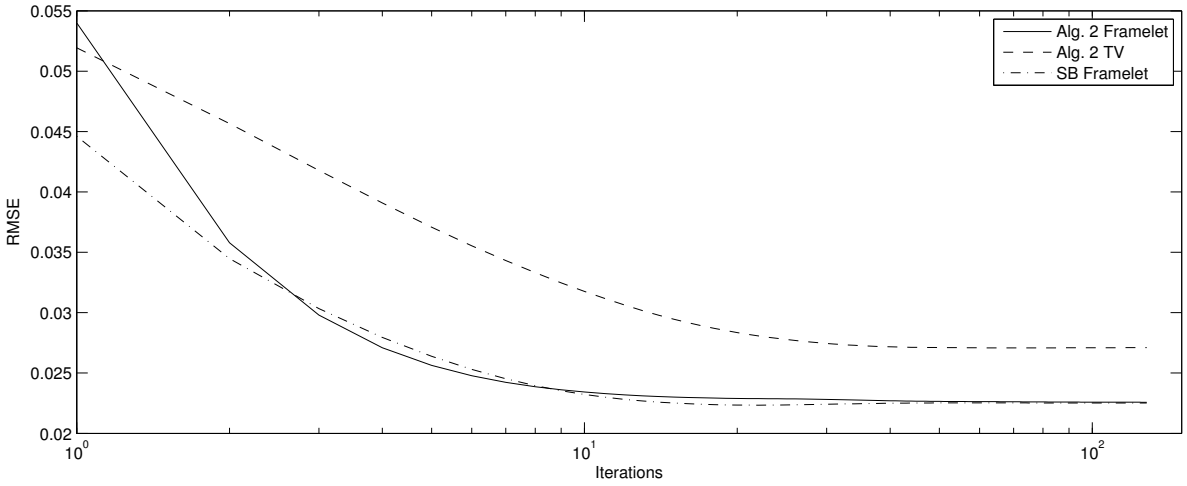


Figure 4.6: RMSE versus number of iterations for the "cameraman" image.

(a) Original “house” image

(b) Blurred “house” image

(c) Algorithm 2 Framelet; 64 iterations; 26.3 seconds

(d) Algorithm 2 TV; 256 iterations; 38.2 seconds

(e) Split Bregman Framelet; 64 iterations; 32.7 seconds

Figure 4.7: Deblurring results for the “house” test image.

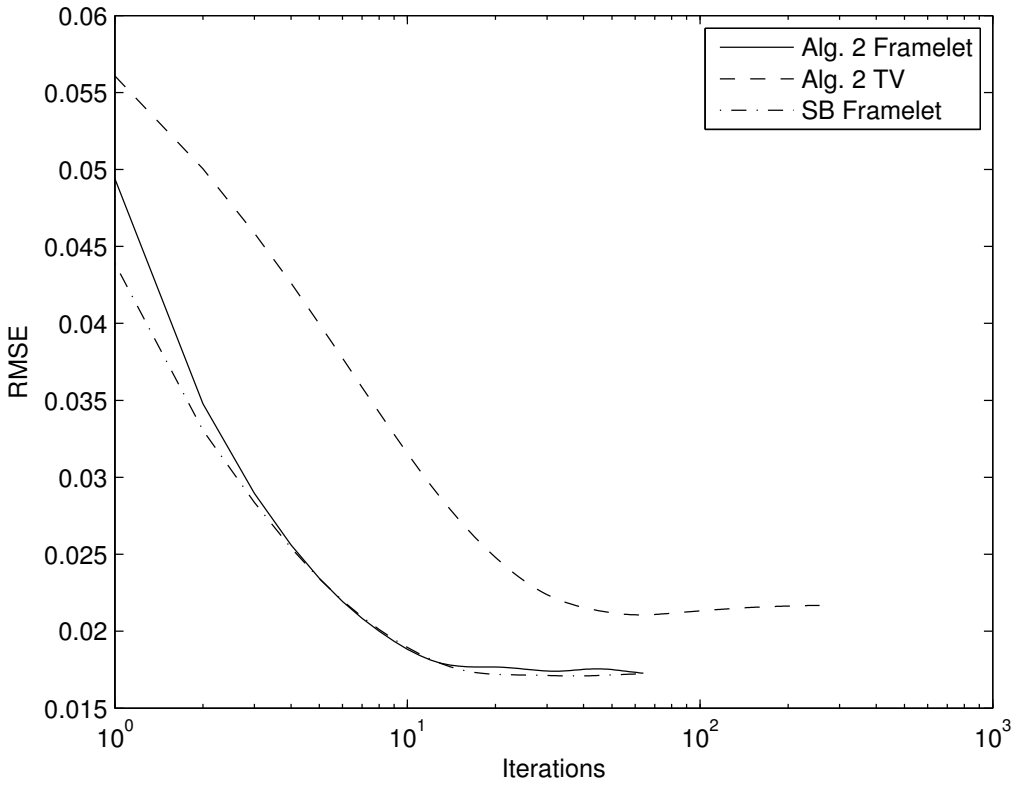
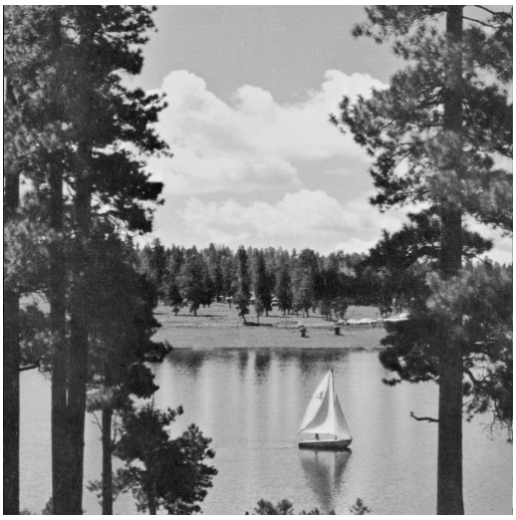


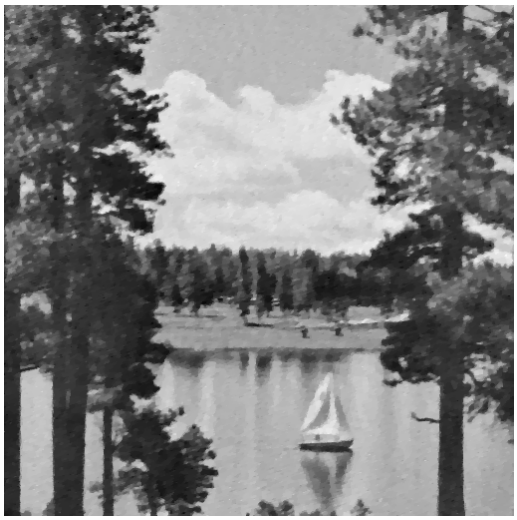
Figure 4.8: RMSE versus number of iterations for the "house" image.



(a) Original “lake” image

(b) Blurred “lake” image

(c) Algorithm 2 Framelet; 64 iterations; 27.5 seconds



(d) Algorithm 2 TV; 256 iterations; 36.5 seconds

(e) Split Bregman Framelet; 64 iterations; 33.8 seconds

Figure 4.9: Deblurring results for the “lake” test image.

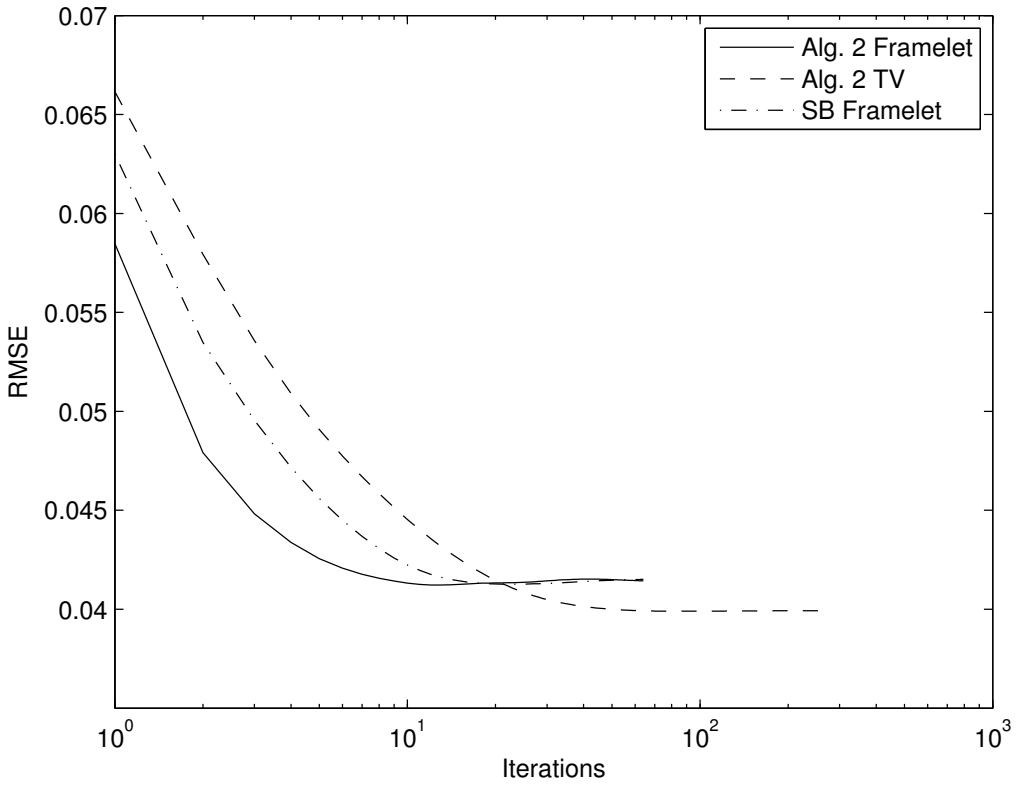


Figure 4.10: RMSE versus number of iterations for the "lake" image.

4.2.3 Comments on the Results

First, it is clear from the run times of the experiments that framelet regularization incurs a significant processing overhead compared to TV regularization. The time-per-iteration of the framelet-regularized algorithms was roughly 4× the time-per-iteration of the TV-regularized algorithm. This overhead can be explained by the additional operations and handling of boundary conditions required by the framelet analysis operator over the single matrix multiplication of the TV analysis operator. To compensate for this and bring all of the execution times in line with one another, more iterations of the TV-regularized algorithm were run.

It is also worth noting that in some cases the framelet-regularized algorithms produced results of higher visual fidelity than the TV-regularized algorithm. TV regularization tends to favor piecewise constant images, resulting in blocky artifacts due to the "staircasing effect" [20]. The two framelet-regularized algorithms appear to suffer less from these artifacts, most noticeably in the "house" experiment.

The differences between the results of Algorithm 2 and the results of the Split Bregman Framelet algorithm are less pronounced. As expected, there are almost no visual or empirical differences between the results of the two algorithms. The iterations of Algorithm 2 were roughly 1.2× faster than those of the Split Bregman algorithm, though the Split Bregman algorithm appears to converge slightly faster than Algorithm 2 (see, e.g., Figure 4.3).

Also worth noting is the fact that the TV-regularized algorithm achieved a lower RMSE than the other two algorithms on the "lake" image. However, upon close inspection, the output of the TV-regularized algorithm visibly suffers from staircasing relative to the other outputs, and arguably bears less similarity (as perceived by a human) to the original than the other outputs. This illustrates the tendency of the RMSE to favour latent images with smaller residuals over smoother images, and highlights the inadequacy of the RMSE (and related metrics such as the PSNR) as an image similarity metric. The challenges of obtaining algorithmic, empirical measurements of image similarity consistent with human perception have been well studied (see, e.g., [53] and references 1-9 within). It should also be mentioned that the RMSEs of the deblurred images with respect to the originals were not the only image similarity metrics computed during the experiments. The structural similarities (SSIMs) [53] of the deblurred images with respect to the originals were also computed. However, in the case of our experiments, the SSIM was not found to differ meaningfully from the RMSE, and so was omitted from the writeup.

Chapter 5

Blind Deblurring

5.1 Algorithm

In this chapter we propose a method of solving the blind deblurring problem of recovering a clear image x from a blurry image $y = x * h + w$, where the blur kernel h is unknown. Not surprisingly, blind deblurring is a significantly harder problem than its non-blind counterpart. Regularization-based approaches to non-blind deblurring frequently result in non-convex optimization problems. We adopt an approach based on the alternating minimization scheme of [13], which has roots in, e.g., [22].

In the case of non-blind deblurring, the regularized least squares problem of (3.1) may be extended to involve the unknown blur kernel as an optimization variable

$$(\hat{x}, \hat{h}) = \arg\min_{x,h} \|y - h * x\|_2^2 + 2\lambda_1^{-1} R_1(x) + 2\lambda_2^{-1} R_2(h) \tag{5.1}$$

where λ_1, R_1, λ_2 and R_2 are regularization parameters and functionals for the image and blur kernel, respectively. The blur kernel's regularization functional R_2 might be chosen to promote properties of the estimated kernel such as smoothness, connectedness, sparsity in some domain, etc. This formulation, though attractive in its simplicity, is regrettably non-convex. In practice, alternating minimization methods have been found to be effective on problems of this type (see e.g. [13, 22]).

The basic idea behind an alternating minimization method is to break a non-convex optimization of two variables such as (5.1) into two related convex optimization problems of one variable each as in Algorithm 3. One can then alternate between minimizing over the first variable (i.e., x) while holding the second variable (i.e., h) fixed, and minimizing over the second variable while holding the first fixed. Applying this approach to (5.1), we obtain the alternating minimization of Algorithm 3.

Algorithm 3 Alternating Minimization for (5.1)

Set h^0 to an initial guess for the blur kernel.
for $n := 0, 1, 2, \ldots$ **do**:

$$x^{n+1} := \arg\min_x \frac{\lambda_1}{2}\|y - h^n * x\|_2^2 + R_1(x) \tag{5.2}$$

$$h^{n+1} := \arg\min_h \frac{\lambda_2}{2}\|y - h * x^{n+1}\|_2^2 + R_2(h) \tag{5.3}$$

end for

In the remainder of this chapter, we derive an approach to blind deblurring based on Algorithm 3 using Chambolle and Pock's primal-dual algorithm as the solver for its alternating minimizations.

In order to obtain a fully specified alternating minimization, it is necessary to fix the regularization functionals R_1 and R_2. Given that the image estimation step of Algorithm 3 strongly resembles the non-blind deblurring problem investigated in Chapter 4, it seems reasonable to use the same regularization functional as was used then: the ℓ_1 norm of x under a framelet decomposition. Thus we set $R_1(x) = \|Ax\|_1$, where A is a framelet analysis operator. However, there is an important distinction between the image estimation step of Algorithm 3 and the non-blind deblurring problem: in the blind case, h^n is not guaranteed to be an accurate estimate of the real blur kernel.

The last step in obtaining a fully specified alternating minimization problem is to choose R_2. Through numerical experiments it was found in [13] that a combination of framelet sparsity regularization via the ℓ_1 norm and ℓ_2 regularization of the latent image can yield accurate estimates of the blur kernel. Mathematically, $R_2(h) = \frac{\gamma}{2}\|h\|_2^2 + \|Ah\|_1$, where γ is a weighting parameter. Their reasoning was as follows: natural motion blur kernels tend to be connected and have limited support. The $\frac{\gamma}{2}\|h\|_2^2$ term favors connected kernels, while the $\|Ah\|_1$ term penalizes sharp discontinuities between pixels, thus favoring kernel estimates with small supports. Without the ℓ_2 term, it was found that kernel estimates tended to consist of sparse, isolated points, which do not correspond to physically sound blur kernels. Our experiments confirmed these findings, and so we opted to include the ℓ_2 term as well. We thus seek a solution to the completely specified alternating minimization of Algorithm 4.

Algorithm 4 Alternating Minimization for Blind Deblurring [13]

Set h^0 to an initial guess for the blur kernel.
for $n := 0, 1, 2, \ldots$ **do**:

$$x^{n+1} := \arg\min_x \frac{\lambda_1}{2}\|y - h^n * x\|_2^2 + \|Ax\|_1 \tag{5.4}$$

$$h^{n+1} := \arg\min_h \frac{\lambda_2}{2}\|y - h * x^{n+1}\|_2^2 + \frac{\gamma}{2}\|h\|_2^2 + \|Ah\|_1 \tag{5.5}$$

end for

Having specified an alternating minimization, the next step was to derive a method of solving it. In [13] an algorithm based on a Split Bregman iteration [33] was proposed for solving the sub-problems of Algorithm 4. As previously mentioned, we instead chose to base our solution on the primal-dual algorithm of Chambolle and Pock [19] in hopes of obtaining a faster overall algorithm and/or more accurate deblurring results.

First, the minimization (5.4) is nearly identical to the non-blind deblurring problem detailed in Chapter 4. The only difference is that the current estimate of the blur kernel, h^n, is not guaranteed to be an accurate estimate of the real blur kernel. We are, however, still able to solve (5.4) using Algorithm 2.

Next we must obtain a solver for (5.5). First we identify in the objective function of (5.5) functions F and G as in (3.12)

$$\underbrace{\frac{\lambda_2}{2}\|y - h * x^{n+1}\|_2^2 + \frac{\gamma}{2}\|h\|_2^2}_{G} + \underbrace{\|Ah\|_1}_{F} \tag{5.6}$$

Note that F is the same as was identified in (4.2) and thus has the same proximal mapping as described in Chapter 4.

The more interesting function is G. Observe that G is the same as the one identified in (4.2), but with an extra term: $\frac{\gamma}{2}\|h\|_2^2$. As it turns out, knowledge of the proximal mapping for G in (4.2) allows us to easily find the proximal mapping of its quadratically perturbed relative.

From [24], let $\phi : \mathcal{X} \to [-\infty, +\infty]$ be a closed convex function, where $\mathcal{X}$ is a real Hilbert space. Suppose

$$\psi = \phi + \frac{\alpha}{2}\|\cdot\|^2 + \langle \cdot, u \rangle + \beta$$

where $u \in \mathcal{X}$, $\alpha \in [0, +\infty)$, and $\beta \in \mathbb{R}$. Then

$$\operatorname{prox}_{\psi} x = \operatorname{prox}_{\phi/(\alpha+1)}((x-u)/(\alpha+1))$$

for all $x \in \mathcal{X}$.

We may apply the above result to (4.10) to obtain the following relationships:

$$\begin{aligned} \phi &= \frac{\lambda}{2}\|y - h * x\|_2^2 \\ \operatorname{prox}_{\tau\phi}(x) &= \mathcal{F}^{-1}\left(\frac{\tau\lambda\mathcal{F}(y)\mathcal{F}(x)^* + \mathcal{F}(h)}{\tau\lambda\mathcal{F}(x)^2 + 1}\right) \\ \alpha &= \gamma \\ u &= 0 \\ \beta &= 0 \end{aligned}$$

to obtain

$$\operatorname{prox}_{\tau G}(h) = \mathcal{F}^{-1}\left(\frac{\tau\lambda\mathcal{F}(y)\mathcal{F}(x)^* + \mathcal{F}(h)}{\tau\left(\lambda\mathcal{F}(x)^2 + \gamma\right) + 1}\right) \tag{5.7}$$

Note that this derivation can also be done in a manner similar to the proof of Theorem 3.

Having obtained the necessary proximal mappings, we may apply Algorithm 1 to the alternating minimization problem in Algorithm 3 in order to obtain Algorithm 5.

Algorithm 5 Blind Deblurring

Choose τ, σ, λ_1, λ_2, $\gamma > 0$, $\theta \in [0,1]$, $(x^0, u^0) \in \mathcal{X} \times \mathcal{Y}$, $(h^0, v^0) \in \mathcal{H} \times \mathcal{V}$, set $\bar{x}^0 := x^0$, $\bar{h}^0 := h^0$, and $n := 0$.
while $n \leq$ N_MAX:

$$
\begin{aligned}
(u^{n+1})(j) &:= \frac{(u^n + \sigma A\bar{x}^n)_i}{\max(1, |(u^n + \sigma A\bar{x}^n)_i|)} \\
x^{n+1/2} &:= \mathcal{F}^{-1}\left(\frac{\tau\lambda_1 \mathcal{F}(y)\mathcal{F}(h^n)^* + \mathcal{F}(x^n - \tau A^T u^{n+1})}{\tau\lambda_1 \mathcal{F}(h^n)^2 + 1}\right) \\
x^{n+1}(j) &:= \begin{cases} 0 & \text{if } x^{n+1}(j) < 0 \\ 1 & \text{if } x^{n+1}(j) > 1 \\ x^{n+1/2}(j) & \text{otherwise} \end{cases} \\
\bar{x}^{n+1} &:= x^{n+1} + \sigma(x^{n+1} - x^n) \\
(v^{n+1})_i &:= \frac{(v^n + \sigma A\bar{h}^n)_i}{\max(1, |(v^n + \sigma A\bar{h}^n)_i|)} \\
h^{n+1/2} &:= \mathcal{F}^{-1}\left(\frac{\tau\lambda_2 \mathcal{F}(y)\mathcal{F}(x^{n+1})^* + \mathcal{F}(h^n - \tau A^T v^{n+1})}{\tau(\lambda_2 \mathcal{F}(x^{n+1})^2 + \gamma) + 1}\right) \\
\tilde{h}^{n+1}(j) &:= \max(h^{n+1/2}(j), 0) \\
h^{n+1} &:= \frac{h^{n+1/2}}{\|\tilde{h}^{n+1}\|_1} \\
\bar{h}^{n+1} &:= h^{n+1} + \sigma(h^{n+1} - h^n) \\
n &:= n + 1
\end{aligned}
$$

end while
return (x^n, h^n)

Some comments on Algorithm 5 are given below.

First, note that the solvers for the inner minimizations are only run for one iteration each per outer iteration. This decision was influenced by the approach to solving Algorithm 4 detailed in [13]. The idea is that since the kernel estimate h^n (image estimate x^{n+1}) used during the estimation of the image (kernel) is likely to be inaccurate, performing multiple inner iterations is unlikely to yield much benefit. The time would be better spent on performing additional outer iterations. This hypothesis was tested during our experiments and found to be true.

Next, due to the challenging non-convex nature of the problem, some additional measures were required to obtain realistic solutions (e.g., connected kernel estimates). First, each pixel of the image estimate x^{n+1} was projected onto $[0,1]$ during each

iteration. Also, each iteration the blur kernel estimate h^{n+1} was projected onto $[0, +\infty)$ and then normalized. This was done because blur kernels are usually assumed to be normalized so as to not affect the total image intensity. These measures were taken in [13] as well. Also note that due to the problem being non-convex, the solutions ontained by Algorithm 5 are sensitive to initial conditions – in particular, h^0. It is recommended in [13] that h^0 be set to a delta, i.e., a kernel with a single pixel of unit intensity in the centre.

Finally, we borrowed an idea from the implementation of the blind deblurring algorithm of [38] and centered the kernel estimate h^{n+1} based on its center of mass after each iteration. This step is not mentioned in the formal description of Algorithm 5 as it is a purely heuristic addition. In practice, it helped alleviate the problem of the kernel estimate "drifting" between iterations.

5.2 Numerical Experiments

5.2.1 Methodology

Two types of experiments were performed: one with an artificially blurry image blurred according to a periodic boundary condition by a known kernel, and the other on four naturally blurred images with unknown blur kernels. Two recent blind motion deblurring algorithms were selected for comparison: the framelet-regularized algorithm of [13] which uses an alternating minimization approach with a Split Bregman [33] inner solver, and the ℓ_1/ℓ_2-regularized algorithm of [38] which uses ISTA [7] and iteratively reweighted least-squares inner solvers. The algorithm in [13] was selected for comparison as it is what Algorithm 5 is primarily based on, while the algorithm of [38] was selected simply because it is a recent, well-performing algorithm using a regularization-based approach, and has freely available source code.

It is important to note that the algorithms of [13] and [38] both assume symmetric boundary conditions, whereas Algorithm 5 assumes a periodic boundary condition. As described in Section 2.2, it is expected that algorithms assuming a symmetric boundary condition will provide better quality results on images with significant high frequency content near the edges at the cost of a longer execution time.

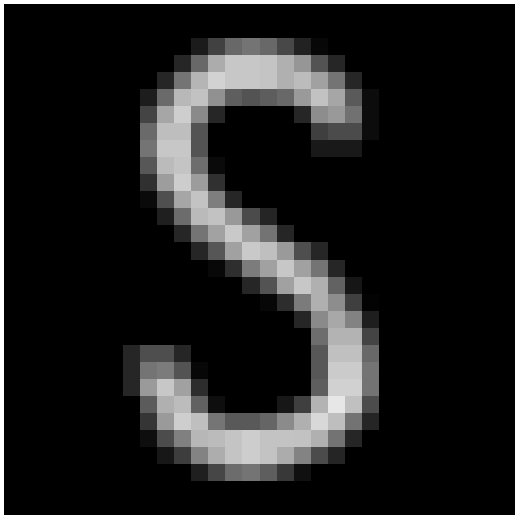
Experiments were run in the same environment and according to the same criteria as described in Section 4.2.1, except the artifically blurred "cameraman" image was instead corrupted by noise with power $\sigma^2 = 10^{-6}$.

Three additional notes: the estimated blur kernels from Algorithm 5 and the algorithm of [38] have been uniformly brightened for visibility, and the iteration counts

for the algorithm of [38] have been omitted as they are not readily comparable to the iteration counts of the other two algorithms. All iteration count-related settings for the algorithm of [38] were left at their defaults. Lastly, the "elephant" test image is in fact non-uniformly blurred, and was included as an indicator of the algorithms' robustness to non-uniform blurs.

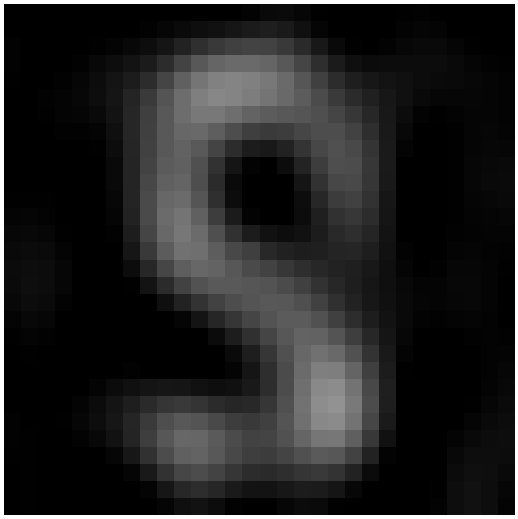
5.2.2 Results

(a) 512×512 blurred noisy "cameraman" image



(b) 30×30 blur kernel

(c) Algorithm 5 image estimate; 200 iterations; 196 seconds



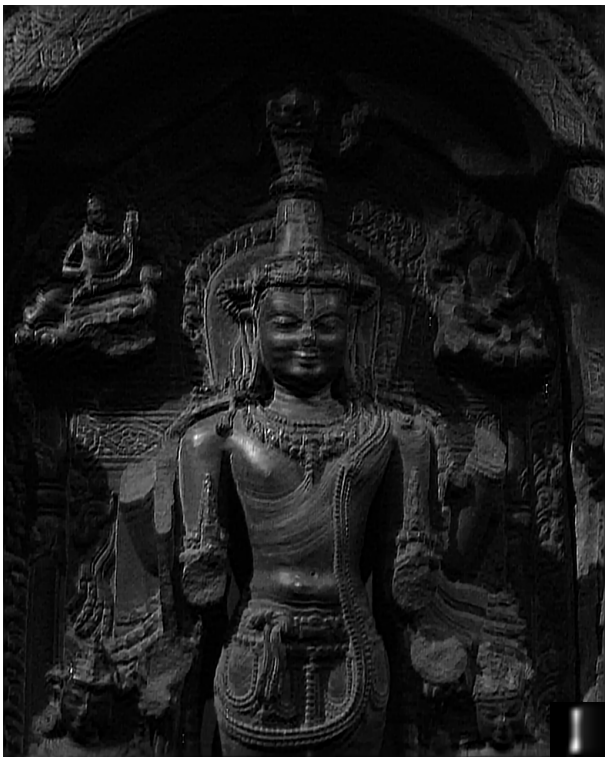
(d) Algorithm 5 kernel estimate; 200 iterations; 196 seconds

Figure 5.1: Blind deblurring results for the "cameraman" test image.

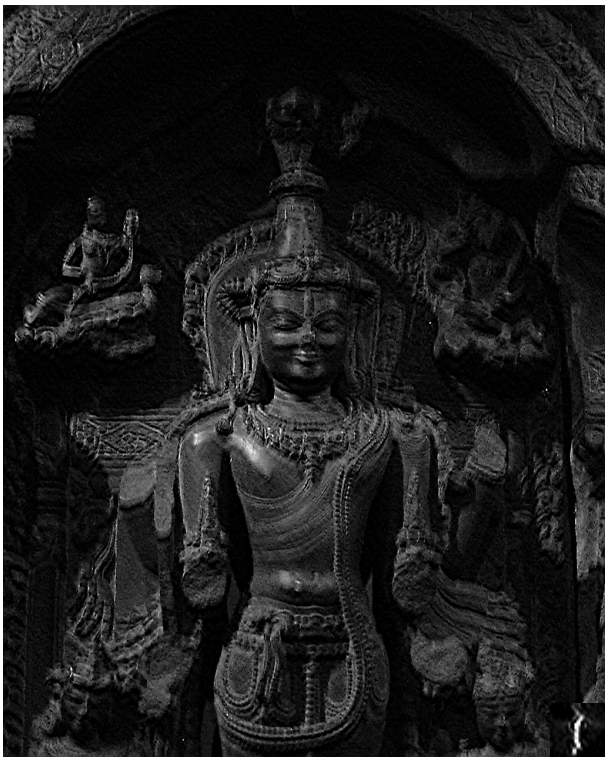
(a) 1024×1280 blurry "Lyndsey" image

(b) Algorithm 5's image and kernel estimate; 200 iterations; 442 seconds

Figure 5.2: Blind deblurring results for the "Lyndsey" test image.



(a) [13]'s image and kernel estimate; 200 iterations; 4027 seconds



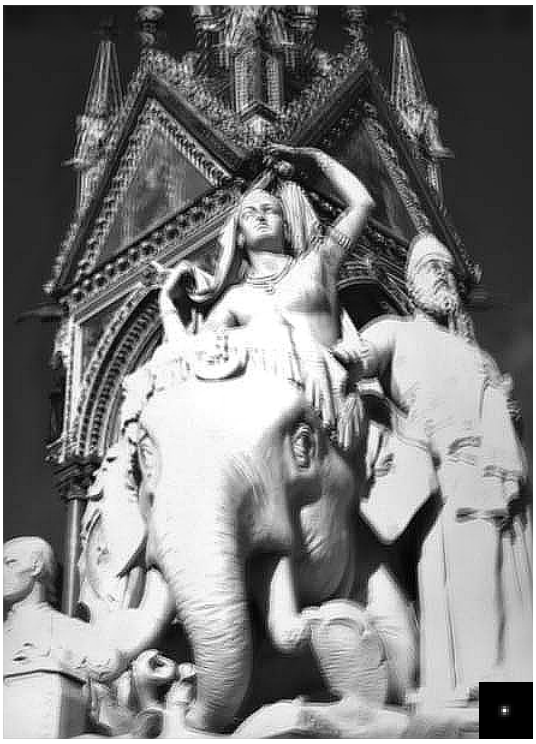
(b) [38]'s image and kernel estimate; 462 seconds

Figure 5.3: Blind deblurring results for the "Lyndsey" test image, continued.

(a) 441 × 611 blurry “elephant” image

(b) Algorithm 5’s image and kernel estimate; 200 iterations; 180 seconds

Figure 5.4: Blind deblurring results for the “elephant” test image.



(a) [13]'s image and kernel estimate; 200 iterations; 940 seconds

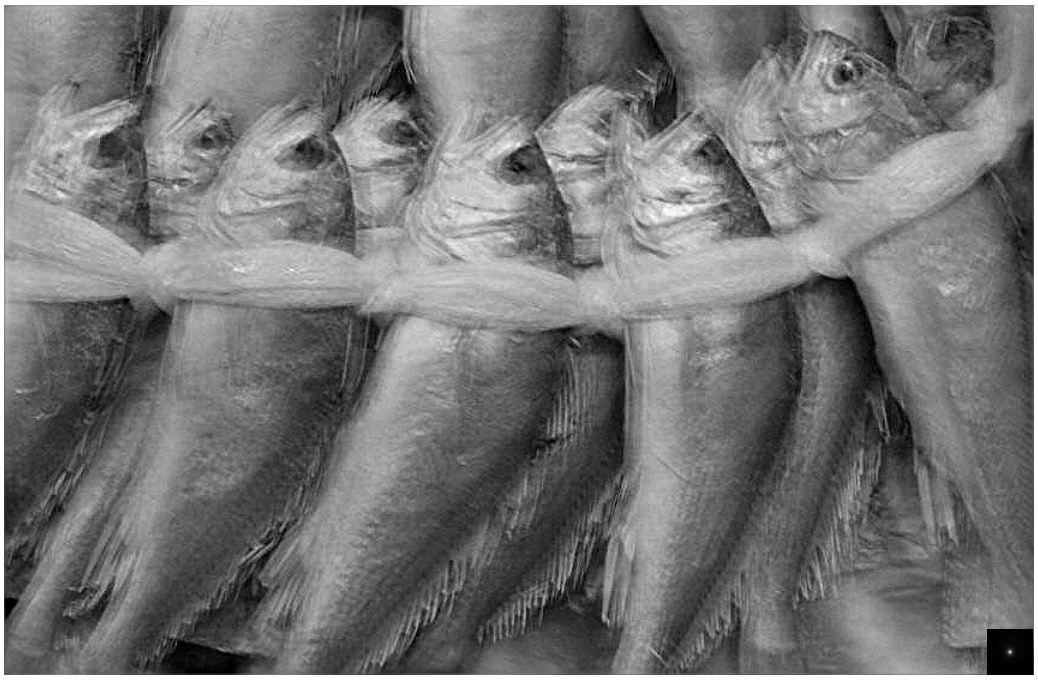
(b) [38]'s image and kernel estimate; 142 seconds

Figure 5.5: Blind deblurring results for the "elephant" test image, continued.

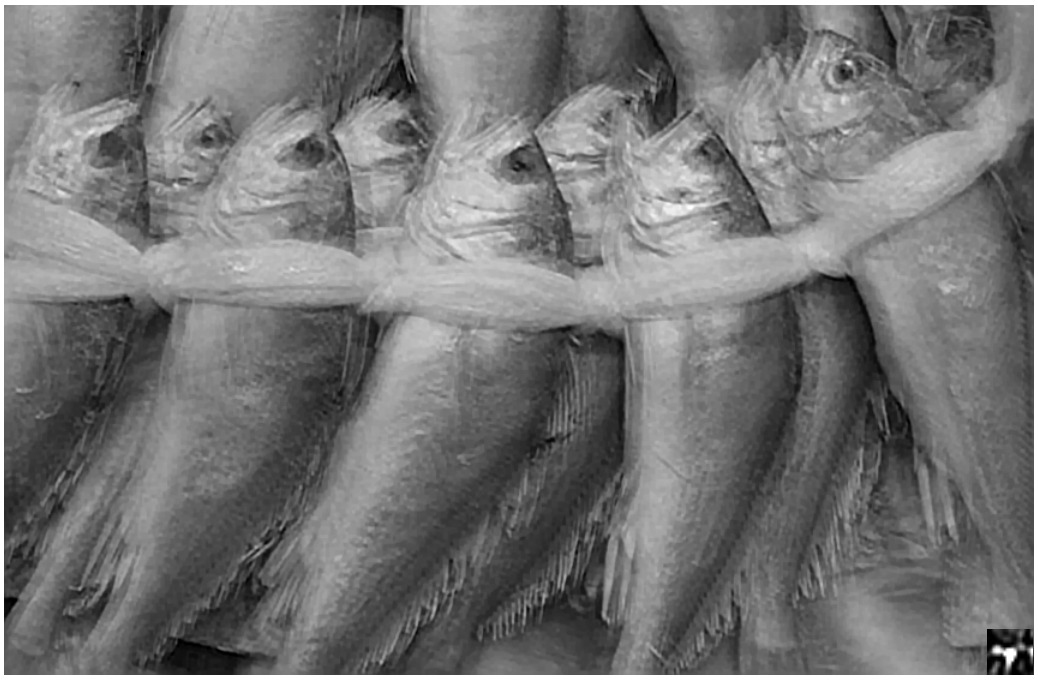
(a) 858 × 558 blurry “fish” image

(b) Algorithm 5’s image and kernel estimate; 200 iterations; 309 seconds

Figure 5.6: Blind deblurring results for the “fish” test image.



(a) [13]'s image and kernel estimate; 200 iterations; 1659 seconds



(b) [38]'s image and kernel estimate; 175 seconds

Figure 5.7: Blind deblurring results for the "fish" test image, continued.

(a) 255 × 255 blurry “painting” image

(b) Algorithm 5’s image and kernel estimate; 200 iterations; 52 seconds

Figure 5.8: Blind deblurring results for the “painting” test image.

(a) [13]'s image and kernel estimate; 200 iterations; 248 seconds

(b) [38]'s image and kernel estimate; 26 seconds

Figure 5.9: Blind deblurring results for the "painting" test image, continued.

5.2.3 Comments on the Results

The algorithm of [38] is the fastest of the three algorithms on all inputs except the large 1024×1280 "Lyndsey" test image, in which case Algorithm 5 is the fastest. This illustrates the computational advantage of FFT-based deblurring methods in the case of large inputs. Based on the results of the previous section, the input size threshold at which Algorithm 5 overtakes the algorithm of [38] is estimated to be around one megapixel.

The algorithm of [13], while perhaps providing the sharpest and most accurate results of the three algorithms, required over an hour to complete 200 iterations on the "Lyndsey" test image. This can be be attributed to the absence of fast solvers for its inner minimizations, due to the assumption of symmetric boundary conditions. For large inputs, this issue could be partially remedied by an appropriate prescaling of the input image, with the full-size image and kernel used only during the final iteration.

A practical concern encountered when adjusting the tuning parameters of the three algorithms was that the algorithms with more parameters required significantly more tuning effort in order to obtain satisfactory results. This is the curse of dimensionality at work: the volume of the space that must be explored when selecting tuning parameters is exponential in the number of parameters. Algorithms with fewer parameters are preferable for this reason. For reference, Algorithm 5 has three tunable parameters, the algorithm of [13] has three (of which only two were adjustable in the provided implementation), and the algorithm of [38] has several, of which only one was varied during the experiments.

The three algorithms appear to cope reasonably well with the non-uniformly blurred "elephant" image. Given that most naturally blurred images are likely to be non-uniformly blurred to some degree, this is an important property for a general-purpose motion deblurring algorithm to have.

Finally, Algorithm 5 was often found to perform worse than the other two algorithms on heavily blurred images with significant high-frequency content around the edges, such as the "fish" test image. In particular, note the artifacts near the edges of Figure 5.6b caused by the periodic boundary condition inaccurately modeling the scene near the boundaries of the image. This is the price of the fast execution speed made possible by the assumption of periodic boundary conditions. It is worth noting, however, that recent progress [40, 2] has been made on reducing the effect of boundary artifacts on deblurring algorithms while still allowing for efficient implementation. Integrating the ideas of [40, 2] into the deblurring frameworks discussed in this thesis is an interesting prospect for future work.

Chapter 6

Conclusion

In this thesis, we have derived algorithms for non-blind and blind deblurring using framelet-sparsity regularization and the algorithm of [19]. Throughout the thesis we have assumed periodic boundary conditions in order to use fast convolution operations without placing restrictions on the form of the blur kernel.

Chapter 1 introduced the non-blind and blind deblurring problems, described previous work related to ours, and summarized the contributions of this thesis.

Chapter 2 described deblurring problems in more depth, gave explanations as to why naïve deblurring methods fail, and provided a review of the effects of different boundary assumptions.

Chapter 3 reviewed the relevant mathematical background on linear inverse problems, sparsity, convex optimization, frames, wavelets, and framelets.

Chapter 4 presented a derivation of our proposed non-blind deblurring algorithm. In Section 4.2.2 we saw how framelet-sparsity regularization could offer improved results over TV regularization at the cost of execution speed. We also saw how the algorithm of [19] can offer small improvements in speed over Split Bregman algorithms. For these reasons, we recommend Algorithm 2 for general non-blind motion deblurring, especially when TV-regularized deblurring algorithms yield results that suffer noticeably from staircasing.

Chapter 5 presented a derivation of our proposed blind deblurring algorithm. In Section 5.2.2 we saw how the assumption of periodic versus symmetric boundary conditions could lead to signifcantly faster execution times at the cost of worse results on images with significant high-frequency content near their boundaries. Additionally, we saw how the primal-dual algorithm of [19] is competitive with ISTA- and Split Bregman-based methods in the context of blind deblurring. Based on these results, we recommend the use of Algorithm 5 on large (> 1 MP) motion-blurred images

without large amounts of high frequency content near the edges.

One interesting direction for future work is to adapt the recent results of [40, 2] on improved handling of boundary conditions to Algorithm 5. This has the potential benefit of improving the performance of Algorithm 5 on images with high edge activity without incurring large costs in execution speed.

Bibliography

[1] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54(11):4311–4322, 2006.

[2] M.S.C. Almeida and M.A.T. Figueiredo. Deconvolving images with unknown boundaries using the alternating direction method of multipliers. *IEEE Trans. Image Processing*, 22(8):3074–3086, 2013.

[3] Kenneth J Arrow, Leonid Hurwicz, and Hirofumi Uzawa. *Studies in Linear and Non-linear Programming*. Stanford University Press Stanford, California, 1964.

[4] G. R. Ayers and J. C. Dainty. Iterative blind deconvolution method and its applications. *Optics Letters*, 13(7):547–549, 1988.

[5] M. R. Banham and Aggelos K. Katsaggelos. Digital image restoration. *IEEE Signal Processing Magazine*, 14:24–41, 1997.

[6] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE Trans. Image Processing*, 18(11):2419–2434, 2009.

[7] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[8] J. Biemond, R.L. Lagendijk, and R.M. Mersereau. Iterative methods for image deblurring. *Proceedings of the IEEE*, 78(5):856–883, 1990.

[9] J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples (CMS Books in Mathematics)*. Springer, 2nd edition, 2005.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[11] J. Cai, S. Osher, and Z. Shen. Split bregman methods and frame based image restoration. *Multiscale Modeling & Simulation*, 8(2):337–369, 2010.

[12] J.-F. Cai, R. H. Chan, and Z. Shen. A framelet-based image inpainting algorithm. *Applied and Computational Harmonic Analysis*, 24:131–149, 2008.

[13] J.-F. Cai, H. Ji, C. Liu, and Z. Shen. Framelet-based blind motion deblurring from a single image. *IEEE Trans. Image Processing*, 21(2):562–572, 2012.

[14] E. Candès and D. L. Donoho. Ridgelets: a key to higher-dimensional intermittency? *Phil. Trans. of the Royal Society of London A*, 357(1760):2495+, 1999.

[15] E. J. Candès and D. L. Donoho. Curvelets – a surprisingly effective nonadaptive representation for objects with edges. In A. Cohen, C. Rabut, and L. L. Schumaker, editors, *Curves and Surfaces*, Nashville, TN, 1999. Vanderbilt University Press.

[16] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Information Theory*, 52(2):489–509, 2006.

[17] M. Cannon. Blind deconvolution of spatially invariant image blurs with phase. *IEEE Trans. Acoustics, Speech and Signal Processing,*, 24(1):58–63, 1976.

[18] A. Chai and Z. Shen. Deconvolution: a wavelet frame approach. *Numerische Mathematik*, 106(4):529–587, 2007.

[19] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[20] T. Chan, S. Esedoglu, F. Park, and A. Yip. Recent developments in total variation image restoration. In *Mathematical Models of Computer Vision*. Springer Verlag, 2005.

[21] T. F. Chan, G. H. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM Journal on Scientific Computing*, 20(6):1964–1977, 1999.

[22] T. F. Chan and C.-K. Wong. Total variation blind deconvolution. *IEEE Trans. Image Processing*, 7(3):370–375, 1998.

[23] T.F. Chan and C.-K. Wong. Total variation blind deconvolution. *IEEE Trans. Image Processing*, 7(3):370–375, 1998.

[24] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[25] I. Daubechies, B. Han, A. Ron, and Z. Shen. Framelets: Mra-based constructions of wavelet frames. *Applied and Computational Harmonic Analysis*, 14:1–46, 2003.

[26] D. L. Donoho. Compressed sensing. *IEEE Trans. Information Theory*, 52:1289–1306, 2006.

[27] D. L. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

[28] M. Duarte and M. Davenport. Connexions web site - compressible signals. `http://cnx.org/content/m37166/1.5/`, 2011. Accessed: 28/03/2013.

[29] M. Elad. Sparse and redundant representation modeling – what next? *IEEE Signal Processing Letters*, 19(12):922–928, 2012.

[30] M. Elad, P. Milanfar, and R. Rubinstein. Analysis versus synthesis in signal priors. *Inverse Problems*, 23(3):947, 2007.

[31] D. A. Fish, A. M. Brinicombe, E. R. Pike, and J. G. Walker. Blind deconvolution by means of the richardson–lucy algorithm. *Journal of the Optics Society of America A*, 12(1):58–65, 1995.

[32] J. Gilles. The bregman cookbook v3.0. `http://www.math.ucla.edu/~jegilles/BregmanCookbook.html`, 2012. Accessed: 15/03/2013.

[33] T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

[34] J. Hadamard. Sur les problèmes aux dérivés partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

[35] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. SIAM, 1st edition, 2006.

[36] L. He, A. Marquina, and S. J. Osher. Blind deconvolution using tv regularization and bregman iteration. *International Journal of Imaging Systems and Technology*, 15(1):74–83, 2005.

[37] J. Kovacevic and A. Chebira. An introduction to frames. *Foundations and Trends in Signal Processing*, 2(1):1–94, 2008.

[38] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 233–240, 2011.

[39] S. Mallat. Multiresolution approximation and wavelet orthonormal bases of $l^2(\mathbb{R})$. *Transactions of the American Mathematical Society*, 315:69–87, 1989.

[40] A. Matakos, S. Ramani, and J. A. Fessler. Accelerated edge-preserving image restoration without boundary artifacts. *IEEE Trans. Image Processing*, 22(5):2019–2029, 2013.

[41] S. Nam, M.E. Davies, M. Elad, and R. Gribonval. The cosparse analysis model and algorithms. *Applied and Computational Harmonic Analysis*, 34(1):30 – 56, 2013.

[42] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal of Computing*, 24(2):227–234, 1995.

[43] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[44] M. K. Ng, R. H. Chan, and W. Tang. A fast algorithm for deblurring models with neumann boundary conditions. *SIAM Journal on Scientific Computing*, 21(3):851–866, 1999.

[45] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, USA, 1970.

[46] A. Ron and Z. Shen. Affine systems in $l_2(\mathbb{R}^d)$: the analysis of the analysis operator. *J. Functional Analysis*, 148:408–447, 1996.

[47] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992.

[48] I. W. Selesnick and M. A. T. Figueiredo. Signal restoration with overcomplete wavelet transforms: Comparison of analysis and synthesis priors. *In Proceedings of SPIE*, 7446 (Wavelets XIII), 2009.

[49] Jr. Stockham, T.G., T.M. Cannon, and R.B. Ingebretsen. Blind deconvolution through digital signal processing. *Proceedings of the IEEE*, 63(4):678–692, 1975.

[50] H. L. Taylor, S. C. Banks, and J. F. McCoy. Deconvolution with the l1 norm. *Geophysics*, 44(1):39–52, 1979.

[51] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York, 1977.

[52] S. Vaiter, G. Peyré, C. Dossal, and J. Fadili. Robust sparse analysis regularization. *IEEE Trans. Information Theory*, 59(4):2001–2016, 2013.

[53] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, 2004.

[54] Y.-L. You and M. Kaveh. Blind image restoration by anisotropic regularization. *IEEE Trans. Image Processing*, 8(3):396–407, 1999.

[55] M. Zhu and T. Chan. An efficient primal-dual hybrid gradient algorithm for total variation image restoration. *UCLA CAM Report*, pages 08–34, 2008.