Efficient Image Restoration Algorithms for Near-Circulant Systems

Except where reference is made to the work of others, the work described in this dissertation is my own or was done in collaboration with my advisory committee. This dissertation does not include proprietary or classified information.

Ruimin Pan

Certificate of Approval:

Thomas S. Denney
Professor
Electrical and Computer Engineering

Stanley Reeves, Chair
Professor
Electrical and Computer Engineering

Jitendra K. Tugnait
Professor
Electrical and Computer Engineering

George T. Flowers
Interim Dean
Graduate School

Efficient Image Restoration Algorithms for Near-Circulant Systems

Ruimin Pan

A Dissertation

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Doctor of Philosophy

Auburn, Alabama
May 10, 2007

Efficient Image Restoration Algorithms for Near-Circulant Systems

Ruimin Pan

Signature of Author

Date of Graduation

Vita

Ruimin Pan, daughter of Weizhong Pan and Xuejun Tan, was born in Chongqing, Sichuan Province, P.R.China on June 2nd, 1976. She received her Bachelor degree and Master degree in Electronic Engineering from Southeast University in July, 1998, and March, 2001, respectively. She entered the Ph.D program in the Department of Electrical and Computer Engineering at Auburn University in August 2001. Her research interests are in the areas of digital signal/image processing, image restoration, compression and inverse problems.

Dissertation Abstract

Efficient Image Restoration Algorithms for Near-Circulant Systems

Ruimin Pan

Doctor of Philosophy, May 10, 2007
(M.E., Southeast University, P.R. China, March 2001)
(B.E., Southeast University, P.R. China, July 1998)

130 Typed Pages

Directed by Stanley J. Reeves

The problem of image restoration has been extensively studied for its practical importance as well as its theoretical interest. Restoration problem arises in almost every branch of engineering and applied physics. The goal of image restoration is to recover the original scene from the degraded observations. It seeks to model the degradations, blur and noise, and apply an inverse procedure to obtain an approximation of the original scene.

Due to the fact that image restoration requires a huge amount of computation and storage, techniques and algorithms that can improve the speed or quality are desirable. A great variety of fast restoration methods have been proposed. The most well known fast restoration algorithms involve the use of fast Fourier transforms (FFT's) to implement shift-invariant deblurring. However, in the face of unknown boundaries, shift-variant smoothing, and other conditions, FFT's cannot be used in a straightforward manner in the inversion process.

A group of efficient image restoration algorithms for circulant or near-circulant systems are proposed in this dissertation that overcome some of these limitations in the direct

application of fast transforms. We assume that the system is a circulant or near-circulant system so that the convolution property of the FFT can be used.

The regularization of the least-squares criterion is an effective approach in image restoration to reduce noise amplification. To avoid the smoothing of edges, edge-preserving regularization using a Gaussian Markov random field (GMRF) model is often used to allow realistic edge modeling and provide stable maximum a posteriori (MAP) solutions. However, this approach is computationally demanding because the introduction of a non-Gaussian image prior makes the restoration problem shift-variant. In this case, a direct solution using FFT's is not possible even when the blurring is shift-invariant.

We consider a class of edge-preserving GMRF functions that are convex and have non-quadratic regions that impose less smoothing on edges. We propose a decomposition-enabled edge-preserving image restoration (DEEPIR) algorithm for maximizing the likelihood function. By decomposing the problem into two sub-problems with one shift-invariant and the other shift-variant, our algorithm exploits the sparsity of edges to define an FFT-based iteration that requires few iterations and is guaranteed to converge to the MAP estimate.

The assumption of a circulant system does not always hold under certain circumstances. Many problems in signal and image processing require the solution of systems with a Toeplitz-block-Toeplitz (TBT) structure in which both the Toeplitz blocks and the block structure are banded. Some fast algorithms make use of the persymmetry (symmetry about the main antidiagonal) of the Toeplitz blocks, while the "bandedness" remains unexplored. Other algorithms exploit block bands but not banded blocks (or vice versa).

We present a fast algorithm that exploits both the bandedness of the blocks and the block-bandedness of the block Toeplitz structure by extending the system to a circulant-block-circulant (CBC) system and solve for the original image by solving a larger system. Since a Toeplitz-block-Toeplitz system has a near-circulant structure, the computation involved in the extending and solving is comparatively small. Other published algorithms for TBT matrices typically involves $O(M^5)$ operations or $O(6M^3)$ operations with 'bandlimited assumption'. This method requires $O(k^2M^3)$ operations for an $M^2 \times M^2$ Toeplitz-block-Toeplitz matrix with bandwidth k without any assumptions about the system.

The edge-preserving regularization method proposed for edge preserving also has applications in deblocking JPEG images where the block discrete cosine transform and quantization cause contouring and blocky artifacts in the compressed image. By integrating regularization into decompression of a compressed JPEG image, this algorithm significantly reduces blocky effects in the image.

The proposed edge-preserving regularization scheme can be applied to reduce ringing artifacts on edges and smooth block boundaries in JPEG images. When the image restoration system is a TBT system instead of a CBC system, we can still make use of the FFT by extending and displacing the system to achieve a fast solution. In general, the proposed techniques in this dissertation improve the quality of restored images and improve the computational performance.

Acknowledgments

This dissertation would not have been possible without the generous help of my research advisor Stanley J. Reeves. I would like to sincerely thank him for his invaluable guidance, encouragement and inspiration during the course of my research. I am particularly indebted to him for having faith in me and never losing patience in answering my questions and concerns on both technical and nontechnical topics. Our stimulating discussions not only helped to shape the contents of this work, but also gave me a unique insight into the challenging process of conducting independent scientific research.

I would like to thank Dr. Thomas S. Denney for being my committee member and setting up the computer lab where I conducted most of my experiments and simulations. I would like to thank Dr. Jitendra K. Tugnait and Dr. Amnon J. Meir for their advice on the thesis and for being on my committee. I would also like to express my deep gratitude to the Department of Electrical and Computer Engineering, my research advisor and the Air Force Office of Scientific Research for providing indispensable financial support for my graduate studies.

Finally, I would like to especially thank my beloved parents, Weizhong Pan and Xuejun Tan, my husband, Tao Zhou for their love and support that have made this work possible. This dissertation is dedicated to them.

Style manual or journal used Discrete Mathematics (together with the style known as "auphd"). Bibliograpy follows van Leunen's *A Handbook for Scholars*.

Computer software used The document preparation package TEX (specifically LATEX) together with the departmental style-file `auphd.sty`.

Table of Contents

List of Figures

Chapter 1

Introduction and Background

1.1 Digital Image Processing

1.1.1 Digital Versus Analog

A digital image is an array of real or complex numbers represented by a finite number of bits [1]. Normally, the term "digital image processing" refers to processing of a two-dimensional picture by a digital computer. Although an image by nature is usually an analog quantity, there are some compelling reasons why digital image processing is preferable to optical methods [2].

- Increasingly, the natural form in which an image is formed and acquired is not analog but discrete. Digital imaging devices like digital cameras and camcorders have a CCD image sensor array that converts light into electrons and transport the charges to an analog-to-digital converter where the volume of charge is measured and converted to binary form. With the development of such devices, higher-resolution images are available so that the image quality is close enough to analog ones.

- There has been and will be a continuing increase in the price/performance capabilities of digital hardware. Computers with great computing power can perform image processing operations in seconds. Increasing sophistication in digital image input/output systems has resulted in eliminating what used to be a major source of errors in digital image processing [2]. Modern scanners, digital cameras and displays are not only accurate and reliable but also inexpensive.

- Image processing algorithms have also developed vastly during the last three decades. Fast transform and fast convolution algorithms have led to several orders of magnitude improvement in the computation time.

Digital image processing has a broad spectrum of applications, such as remote sensing via satellites, medical imaging, radar, sonar, and acoustic image processing. Image transmission and storage is also made easy with digital methods. In fact, there is almost no area of technical endeavor that is not impacted in some way by digital image processing. In nuclear medicine, the patient is injected with a radioactive isotope that emits gamma rays as it decays. Images are recorded from the emissions and used to locate sites of bone pathology, such as infections or tumors. Multi-spectral imaging is widely used in monitoring environmental conditions such as vegetation, maximum water penetration and soil moisture [3].

1.1.2 Digital Image Processing System

An image processing system consists of a sequence of steps. First, the object is observed and recorded on an imaging system. A typical example of an imaging system is the human eye. An image forms upon the human retina by the iris-lens portion of the eye, then the image is sent to the brain through the neural system. In a digital system, the image is sampled and quantized so that it can be stored in a digital medium. Then the digital image, normally a two-dimensional array is processed by computer with certain algorithms and sent to display or recording devices such as printers (Fig. 1.1).

In general, any two-dimensional function that bears information can be considered an image [1]. An image can represent luminance of an object, temperature profile of a region,

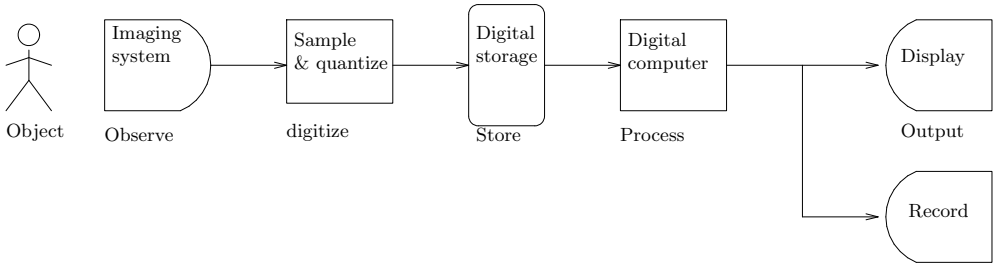


Figure 1.1: Digital image processing system

the absorption characteristics of the body tissue or the radar cross section of a target. An image can also be a color image or a monochrome image. Monochrome images are often described using gray levels. Color images have more than one representation; the most commonly used include the RGB model, CMYK model and YCbCr model. Another classical method of image representation is by an orthogonal series expansion, such as the Fourier series.

According to Nyquist's sampling theorem, when sampling a band-limited signal, the sampling frequency must be greater than twice the input signal bandwidth in order to be able to reconstruct the original perfectly from the sampled version. In reality, an image usually is not strictly band-limited. However, by ignoring (or filtering out) the high-frequency components and sampling at a comparatively large sampling rate, the sample image can be close enough to the original version. After the image is sampled, it is quantized so that it has a finite number of gray levels that can be stored in a computer. Quantization and sampling may introduce non-invertible errors and noise to the image and should be part of the image model in the processing algorithm.

After an image is digitized, it is then processed according to certain image processing requirements. For example, image enhancement techniques could be applied to improve

the contrast or brightness of the image or highlight certain features of interest in it. If the image is blurry because of some optical distortion, it should be restored. With the dramatic booming of the internet, more and more information is transmitted online. To save bandwidth and improve the speed, most pictures and graphs are compressed images such as JPEG and GIF images. These images are converted from basic RGB data by all kinds of compression toolkits using compression algorithms such as the Discrete Cosine Transform, Huffman coding or LZW compression.

The output of an image processing system can be sent to a display device such as a computer monitor or some recording devices like printers or plotters.

1.1.3 Image Processing Methods

Digital image processing is a general concept referring to a group of different operations on an image. Mathematically, any operation on a two-dimensional matrix is meaningful in some way. But for an image, there are a few special operations besides image restoration that help extract useful information from the array.

Image enhancement is among the simplest and most appealing areas of digital image processing [3]. Typical image enhancement includes contrast manipulation, histogram modification, noise cleaning, edge sharpening and color enhancement. Most acquired images are not in ideal condition; for example, an image might be too dark to tell any details. To accentuate certain features in the image for future analysis, brightness or contrast can be improved by amplitude rescaling of each pixel, or the image histogram can be stretched to separate different gray levels (Fig 1.2). In many image processing systems, image enhancement is used as the pre-processing step.

Figure 1.2: Contrast stretching

Image reconstruction is closely related to image restoration. It differs from image restoration in that image reconstruction operates on a set of image projections and not on a full image. Image reconstruction shares the same objective with restoration—recovering the original image—and ends up solving the same mathematical problem, which is finding a solution to a set of linear or nonlinear equations [4].

Geometric transformation is another common image processing operation, in which an image is spatially translated, scaled, rotated or nonlinearly warped. Image translation, scaling and rotation are all linear operations on the coordinates while image warping is a nonlinear mapping of coordinates. It can also be viewed as a spatial distortion or rubber-sheet stretching. Geometric transformations have found uses in medical imaging, computer vision, and computer graphics.

Image analysis refers to the extraction of measurements, data or information from an image by automatic or semi-automatic methods. It is distinguished from other types of image processing, such as enhancement, restoration, in that the ultimate product of an image

analysis system is usually numerical output rather than a picture [5]. Edge detection is a commonly used image analysis operation. Discontinuities in an image amplitude attribute such as luminance are fundamentally important primitive characteristics of an image because they often provide an indication of the physical extent of objects in the image. In edge detection, an edge model is established first, then a certain algorithm is applied to match the edge model with the original image, thus finding the edges. First or second derivatives are often used in edge detection.

Image compression is concerned with minimizing the number of bits required to represent an image. In broadcast television, remote sensing, teleconferencing and computer communications, fast transmission and small storage are of significant practical and commercial interest. Data compression exploits data redundancy to represent the information with less data. In an image, certain correlations exist among pixels. Similarly, inter-frame redundancy exists among frames of a video. Additionally, a lower image fidelity criterion requires less data and can still present enough information since human eyes are relatively tolerant when measuring image quality. With the development of fast algorithms, more and more applications of image compression are available. For example, almost all images on the internet are compressed images such as JPEG or GIF images.

Other image analysis such as texture definition, image segmentation and registration are also widely used in separating text or different patterns, evaluating image feature and pattern recognition.

1.2 Image Restoration

1.2.1 Image Formation

Any image acquired by optical or electronic means is likely to be degraded by the sensing environment. Sensor noise, blur due to camera misfocus, relative object-camera motion or random atmospheric turbulence contribute to the degradation. A degraded image is often a blurry image. The output of an image restoration system is called a restored image. The effectiveness of image restoration depends on the extent and the accuracy of the knowledge of the degradation process as well as on the filter design criterion [1].

As in image enhancement, the ultimate goal of image restoration is to improve an image in some sense. The major difference between enhancement and restoration lies in the judgement of the result. Image enhancement is largely a subjective process in that it manipulates the luminance or colors of the image to make certain features available to human eyes. On the other hand, image restoration recovers an image using a priori knowledge about the degradation and the result is evaluated using objective criteria like mean square error. Image restoration problems can be quantified precisely, whereas enhancement criteria are difficult to represent mathematically [1].

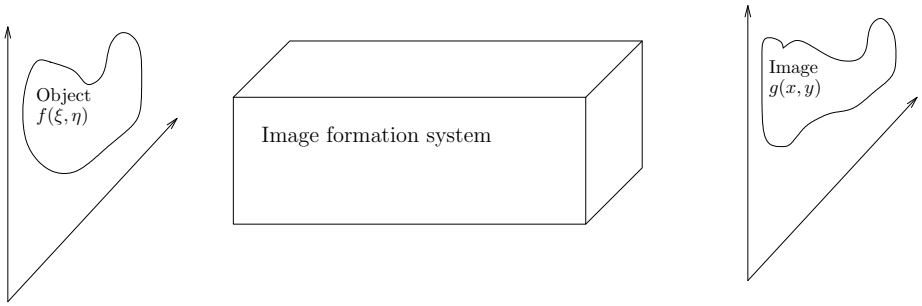


Figure 1.3: Image formation system

Since image restoration is the inverse procedure of image formation, it is necessary to understand the process of forming an image to better understand the process of restoring one.

An image formation system is where the object is mapped onto the image plane and an image is formed. Either the object is illuminated by a source of radiant energy or itself is a source of radiant energy. No matter what kind of imaging system is used, digital or optical, the image $g(x, y)$ (Fig. 1.3) on the image plane is a projection of object $f(\xi, \eta)$. It should be noted that images are representations of objects that are indirectly sensed and that various forms of radiant energy transport are the mechanisms by which the sensing is carried out [2]. So there are a few general principles for image formation:

- Neighborhood Processes. The point (x_0, y_0) on the image plane comes not only from (ξ_0, η_0), but also all other points in the object plane. That is, all points on the object plane may have an effect on any point on the image plane. It is also reasonable to expect that as the distance from the object (ξ_0, η_0) to other points in the object plane increases, the effect on point (x_0, y_0) decreases. In other words, the image formation process is a neighborhood process [2].

- Nonnegativity. An image is formed by the transport of radiant energy. Radiant energy is nonnegative. The smallest possible amount of radiant energy is zero. i.e.,

$$f(\xi, \eta) \geq 0$$

$$g(x, y) \geq 0$$

for any $\xi,\ \eta,\ x$ and y.

- Superposition. radiant energy is superposable. Consider two points in the object plane (ξ_0, η_0) and (ξ_1, η_1). They both emit radiant energy and form image (x_0, y_0) and (x_1, y_1) respectively on the image plane. The result on the image plane is the superposition of energy distribution corresponding to all x and y's in the object plane.

Accordingly, if we consider a single point source in the object $f(\xi, \eta)$, the function that describes the transformation of energy from object plane to image plane should be as follows:

$$g(x, y) = h(x, y, \xi, \eta, f(\xi, \eta))$$

If the above image formation system is linear,

$$h(x, y, \xi, \eta, f(\xi, \eta)) = h(x, y, \xi, \eta) f(\xi, \eta)$$

By superposition, the image $g(x, y)$ on the image plane is formed from contributions of all object points in the object plane.

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x, y, \xi, \eta, f(\xi, \eta) d\xi d\eta$$

For the linear case,

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x, y, \xi, \eta) f(\xi, \eta) d\xi d\eta$$

where function $h(x, y, \xi, \eta)$ is the point-spread function (PSF). Generally, a point-spread function varies with the position in both image and object plane since it is a function of (x, y) and (ξ, η). Nevertheless, some imaging systems act uniformly across image and object planes. Such an image formation system is said to have a shift-invariant point-spread

function. The function $h(\cdot)$ is a function only of the difference in variables in the coordinate systems [2].

$$g(x,y) = \int_{-\infty}^{\infty} h(x-\xi, y-\eta, f(\xi,\eta))d\xi d\eta$$

For the linear case,

$$\begin{aligned} g(x,y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x-\xi, y-\eta) f(\xi,\eta) d\xi d\eta \\ &= h(x,y) * f(x,y) \end{aligned} \tag{1.1}$$

where $*$ represents linear convolution.

For a digital imaging system, the image is sampled and quantized during the process of energy transformation. Now the acquired image is a 2D matrix with positive values. A shift-invariant and linear digital imaging process can be described in (1.2)

$$g(m,n) = \sum_{k,l} h(m-k, n-l) f(k,l) \tag{1.2}$$

1.2.2 Ill-Conditioned Nature of Restoration

In (1.2), the output of a linear shift-invariant system $g(x,y)$ is the convolution of the point-spread function and the original image $f(\xi,\eta)$. In the presence of additive noise, the expression of the linear degradation model becomes

$$g(x,y) = h(x,y) * f(x,y) + n(x,y) \tag{1.3}$$

The values of the noise term $n(x, y)$ are random, and the statistics are assumed to be independent of position. We can express (1.3) in the frequency domain

$$G(u, v) = H(u, v)F(u, v) + N(u, v) \tag{1.4}$$

The convolution in (1.3) now becomes multiplication according to the property of the Fourier transform. Equations (1.3) and (1.4) are the spatial-domain and frequency-domain representation of a linear shift-invariant noisy imaging system. The advantage of a frequency-domain representation is the use of an FFT algorithm.

Although real imaging systems may not be linear and shift-invariant, most of them can be approximated by such processes in order to make use of the extensive tools of linear system theory. In fact, nonlinear and shift-variant models introduce difficulties that often have no known solution or are very difficult to solve computationally, even though they are more general and accurate.

In most image restoration systems, noise is assumed to be independent of spatial coordinates and uncorrelated with respect to the image itself. There are exceptions like X-ray and nuclear-medicine imaging. In this dissertation, we restrict our discussion to the case of independent and uncorrelated noise.

A great variety of methods have been proposed for image restoration problems during the last 30 years. But without common benchmarks or points of reference, it is impossible to pick the optimal method. Even if a few benchmarks have been established, it is still difficult to provide a unified view of image restoration. This is partly due to the ill-conditioned nature of image restoration problems.

In the imaging model above, the image g is a transformation from the original function f. Suppose there is no noise in this system, then

$$\begin{aligned} g(x,y) &= h(x,y) * f(x,y) \\ &= H[f(x,y)] \end{aligned} \tag{1.5}$$

Given the blurred image g and knowledge of the transformation H, the problem seems to be to find the inverse transformation H^{-1} such that

$$H^{-1}[g(x,y)] = f(x,y) \tag{1.6}$$

Then the existence and uniqueness of the inverse transform H^{-1} are important in finding $f(x,y)$. If H^{-1} does not exist, there is no mathematical basis for asserting that $f(x,y)$ can be exactly recovered from $g(x,y)$; it is then called a singular problem. Finally, if there is a unique H^{-1}, it might be ill-conditioned. That is, a trivial perturbation in $g(x,y)$ can produce nontrivial perturbations in $f(x,y)$.

$$H^{-1}[g(x,y) + \epsilon] = f(x,y) + \delta$$

where $\delta \gg \epsilon$; δ is not arbitrarily small and is not negligible. In other words, an ill-conditioned problem is one in which inherent data perturbations can result in undesirable effects in the solution by inverse transformation [2].

In actual calculation, small perturbations like computer round-off errors can present undesirable effects in the solution by inverse transformation. Besides the ill-conditioned

nature of image restoration problem, the existence of noise makes it impossible to find the exact solution f. In conclusion, image restoration is an ill-conditioned problem at best and a singular problem at worst [2].

1.2.3 Image Restoration Filters

Because of the ill-conditioned nature of image restoration problems, the seemingly feasible inverse filter will give an unstable solution. Let $G(u,v)$ be the frequency domain representation of the blurred image, $H(u,v)$ be the point-spread function and $F(u,v)$ be the original image. By (1.4),

$$G(u,v) = H(u,v)F(u,v) + N(u,v).$$

The inverse filter will produce an estimate

$$\begin{aligned} \hat{F}(u,v) &= \frac{G(u,v)}{H(u,v)} \\ &= F(u,v) + \frac{N(u,v)}{H(u,v)} \end{aligned} \tag{1.7}$$

Notice the second term $\frac{N(u,v)}{H(u,v)}$ is actually unknown since the noise is a random function. Furthermore, if the degradation $H(u,v)$ has zero or very small values, the ratio $\frac{N(u,v)}{H(u,v)}$ could easily dominate the estimate $F(u,v)$. In fact, this is frequently the case. Figure 1.4 shows the restoration using inverse filter.

To handle noise and avoid the zero or small-value problem, Hellstrom [6] proposed the application of the Wiener filter in image processing in 1967. The Wiener filter is founded on considering images and noise as mutually uncorrelated random processes, and the objective

is to find an estimate $\hat{f}$ of the uncorrupted image f such that the mean square error between them is minimized [3].

$$\begin{aligned} \hat{f} &= \arg\min_{\hat{f}} E\{e^t e\} \\ &= \arg\min_{\hat{f}} E\{(f-\hat{f})^t (f-\hat{f})\} \end{aligned} \tag{1.8}$$

where $e = f - \hat{f}$. Let f and g be column-ordered original and degraded images and H be the corresponding matrix for the blur h, then the blurred image

$$g = Hf + n$$

and the estimate

$$\hat{f} = Lg,$$

where L is the matrix solution that restores f from g. The mean square error is as follows:

$$\begin{aligned} E\{e^t e\} &= E\{\mathrm{tr}[ee^t]\} \\ &= E\{\mathrm{tr}[(f-Lg)(f-Lg)^t]\} \\ &= E\{\mathrm{tr}[(f-L(Hf+n))(f-L(Hf+n))^t]\} \\ &= E\{\mathrm{tr}[ff^t - L(Hff^t + nf^t) - (ff^t H^t + fn^t)L^t \\ &\quad + L(Hff^t H^t + nf^t H^t + nn^t)L^t]\} \end{aligned} \tag{1.9}$$

Since the trace is a linear operation, it interchanges with the expectation operator E and $E\{fn^t\} = E\{nf^t\} = 0$,

$$E\{e^t e\} = Tr[\mathcal{R}_f - 2LH\mathcal{R}_f + L(H\mathcal{R}_f H^t + \mathcal{R}_n)L^t], \tag{1.10}$$

where $\mathcal{R}_f = E\{ff^t\}$ and $\mathcal{R}_n = E\{nn^t\}$. Taking derivative of (1.10) with respect to L and setting it to 0, we have

$$L = \mathcal{R}_f H^t (H\mathcal{R}_f H^t + \mathcal{R}_n)^{-1}. \tag{1.11}$$

Thus the estimate

$$\begin{aligned} \hat{f} &= Lg \qquad (1.12) \\ &= \mathcal{R}_f H^t (H\mathcal{R}_f H^t + \mathcal{R}_n)^{-1} g \end{aligned}$$

In frequency domain, the solution can be expressed as

$$F(\omega_1, \omega_2) = \frac{H^*(\omega_1, \omega_2) S_f G(\omega_1, \omega_2)}{|H(\omega_1, \omega_2)|^2 S_f + S_n}, \tag{1.13}$$

where S_f and S_n are the power spectral densities of the original image f and the noise n.

Figure 1.5 shows the Lena image restored with a Wiener filter. The Wiener filter minimizes the mean square error between the estimate and the original, so it is also called the minimum mean square error filter. Notice in (1.13), when there is no noise, i.e., $S_n = 0$, it becomes the inverse filter.

The Wiener estimate has desirable properties, since it controls ill-conditioning in a fashion that is explicitly determined by the signal-to-noise ratio as a function of spatial

frequency in the term S_n/S_f. Although there has been much success in applying the Wiener filter to images from a variety of real-world problems, the results of Wiener estimation do not appear as good visually as those produced by other criteria in low signal-to-noise ratio (SNR) cases [2]. The following factors contribute to this shortcoming,

- The Wiener filter is based on linear assumptions, and there are nonlinearities in image recording and in the human visual system.
- Mean squared error is not the criterion that the human visual system naturally employs. Wiener restoration in low SNR cases appears too smooth; the human eye is not often willing to accept more visual noise in exchange for the additional image structure lost in the process [2].
- The Wiener estimate assumes stationary random process models, which is probably insufficient to describe the meaningful structure adequately in images of interest, and uses only the covariance information of the stationary model in the estimate [2].

In general, the Wiener filter is only optimal in an average sense since it minimizes a statistical criterion. It might not be the optimal solution for a specific image. In the mean time, it presents another difficulty in addition to the problem of having to know something about the degradation H: the power spectra of the original image and noise must be known.

On the other hand, the inverse filter can be derived from another approach. The objective function $W(f) = \|g - Hf\|^2$ aims at minimizing the difference between the noisy blurred image and the degraded image without noise. $W(f)$ reaches its minimum value of 0 when $g = Hf$, i.e., there is no noise. Take derivative of $W(f)$ with respect to f and set

it to 0. The minimizer of $W(f)$ is

$$\hat{f} = H^{-1}g,$$

which is exactly the same as inverse filter. As mentioned earlier, the inverse filter does not deal with the noise because it assumes there is zero noise. In order for the filter to have more effect than simple inversion, a constraint is developed so that there is more control over the restoration process.

First, the cost function $W(f) = \|g - H\hat{f}\|^2 \leq \epsilon^2$ subject to $\|L\hat{f}\|^2 \leq E^2$ where ϵ and E are some fixed values and the control variables in the filter. L is a high-pass filter so that it imposes a smoothness condition on the restored image. After the Lagrangian multiplier α is applied, the overall objective function becomes

$$\phi(\hat{f}) = \|g - H\hat{f}\|^2 + \alpha\|L\hat{f}\|^2$$

where $\alpha = (\frac{\epsilon}{E})^2$. By minimizing the objective function, we minimize the data error but bias solution away from "rough" restored image. This is called the constrained least-squares filter. It is in the class of Wiener filters but has much more applications than general Wiener filter because of the control variable α.

1.2.4 Regularized Restoration

As described in previous discussion, noisy, blurred images are often represented in the following algebraic model:

$$g = Hf + n,$$

where the linear degradation H is known and n is random noise uncorrelated with f. f and g are column-ordered versions of the original and the blurred images. Image restoration is an ill-conditioned problem and sometimes even a singular problem. Therefore, the exact original image f cannot be computed from the blurred image g. Instead, restoration algorithms focus on achieving an estimate $\hat{f}$ that is as close as possible to the original image [4].

Among all sorts of restoration algorithms, the constrained least-squares filter not only avoids rough results but also gives us more control over the restoration procedure. The objective function for the constrained least-squares filter is as follows:

$$\phi(\hat{f}) = \|g - H\hat{f}\|^2 + \alpha\|L\hat{f}\|^2 \tag{1.14}$$

To find the minimizer $\hat{f}$ to (1.14), we take a derivative with respect to f and set it to 0.

$$\frac{\partial\phi}{\partial f} = -2H^t g + 2H^t H f + 2\alpha L^t L f = 0,$$

thus

$$\hat{f} = (H^t H + \alpha L^t L)^{-1} H^t g \tag{1.15}$$

or in the frequency domain

$$\hat{F} = \frac{H^* G}{\|H\|^2 + \alpha\|L\|^2}. \tag{1.16}$$

There are a few interesting facts about regularized restoration.

- When $\alpha L^{tL} = R_n R_f^{-1}$, where R_n and R_f are the autocorrelation matrices of the noise and the original image, the regularization filter is actually a Wiener filter (see equation 1.13).

- To gain more control over the restoration and avoid ringing effects, "norms in a weighted Hilbert space" can be used to adapt the restoration process to local properties of the image [7]. In other words,

$$\hat{F} = (H^t R H + \alpha L^t S L)^{-1} H^t R g \tag{1.17}$$

where R and S are diagonal matrices containing weight coefficients with prior information about the original image.

(a) Original image

(b) Noisy blurred image

(c) Restored image using inverse filter

Figure 1.4: Inverse filter

Figure 1.5: Wiener filter

Chapter 2

Huber-Markov Edge-Preserving Regularization

2.1 Edge-Preserving Regularization

As discussed in previous chapters, by using regularization, a numerically good estimation to the original image can be obtained. However, many restored images are liable to incur the phenomenon of ringing (over- and undershoots, super-blacks and whites) [7]. The ringing effects occur on boundaries of the image and in the vicinity of steep intensity transitions such as edges. In linear shift-invariant restoration filters, the ringing effects are caused generally by the poor match between the stationary assumed image model and the actual image data, because shift-invariant regularization penalizes high frequencies at the same level across the image. Fig 2.1 shows the ringing effect in the cameraman image.

The ringing effects near the boundaries of the image result from the intensity jumps at the boundaries, which introduce leakage frequencies. When an FFT is used for fast computation, it assumes that signals are periodic. Because of the sudden change on boundaries, the image spectrum is broadened, the leakage frequencies get amplified, thus introducing the ringing phenomenon. Several methods have been proposed to reduce this kind of ringing artifacts [8, 9].

On the other hand, ringing effects near edges are dependent on the local structures within the image. To reduce this kind of ringing artifact, different penalties should be applied. In (1.14), the minimization process penalizes the square of local pixel differences, which will generate an estimate $\hat{f}$ either excessively noisy or generally blurred. This is because the squared difference of pixel values is so high on edges that the penalties are too

(a) (b)

Figure 2.1: Ringing effect
(a) blurred cameraman image (b) restored image with ringing effects

high. One way to avoid it is to reduce the penalties at edge locations. In a shift-invariant regularization, the cost function (1.14) is quadratic everywhere in the image. Normally the squared pixel difference, or the value of $\|Lf\|^2$ in (1.14) is relatively high in the vicinity of edges but much smaller in smooth areas. In order to put less penalties on edges but a normal amount of penalty elsewhere, a non-quadratic cost function is often proposed in edge-preserving regularization algorithms.

In (1.14), the first term $\|g - H\hat{f}\|^2$ is the data-matching term, where the difference between the noisy image and the noiseless blurred image is minimized. The second term $\|Lf\|^2$ is the constraint that minimizes the roughness of the solution. It is largely dependent on the local properties of the image. To apply different constraints or penalties, we can use

a more general cost function instead of (1.14)

$$\hat{\phi}(\hat{f}) = \|g - H\hat{f}\|^2 + \rho(L\hat{f}) \tag{2.1}$$

In (1.14),

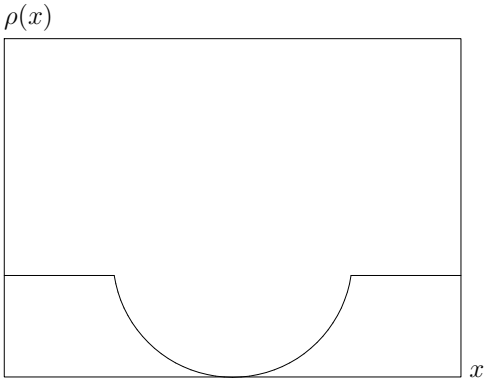
$$\rho(L\hat{f}) = \|L\hat{f}\|^2,$$

which is too high a penalty near edges. A smaller penalty can be applied to edge locations if $\rho(x)$ has smaller values for large x. That is, the function is no longer quadratic outside of a certain region but may be, for example, a small constant or a linear function. In figure 2.2(a) and (b), $\rho(x)$ is flat or straight lines for large x values. There can be a lot of different shapes for the penalty as long as they have smaller values than the quadratic function [10]. Notice that when x is small enough, which is the case for smooth regions in an image, the quadratic penalty is ok.

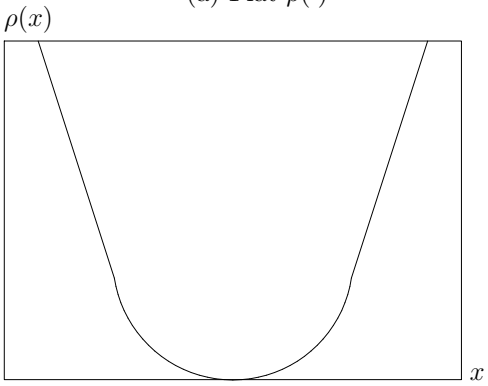
Most edge-preserving regularization algorithms exploit the shift-variant property of penalty functions although different kinds of non-quadratic or half-quadratic cost functions may be used.

2.2 Restoration Models

Shift-invariant image restoration algorithms often introduce ringing effects near sharp intensity transitions. To reduce the ringing effects, one can make use of deterministic *a priori* knowledge about the original image or locally regulate the severity of the noise magnification and the ringing phenomenon depending on the edge information in the image [7]. Many papers also have described edge-preserving methods for image reconstruction and



(a) Flat $\rho(\cdot)$



(b) Linear $\rho(\cdot)$

Figure 2.2: Non-quadratic $\rho(\cdot)$

image restoration based on the Bayesian formalism with non-Gaussian priors [11, 12, 13]. Most of these methods incorporate the edge information as a constraint in the optimization problem.

Typical regularization approaches, including the constrained least-squares (CLS) and the Tikhonov-Miller formulations, utilize quadratic functionals and compensate for the ill-posedness of the restoration problem by applying prior information such as smoothness into the restoration process [14, 15]. Quadratic functions are attractive because they enable the

analytic derivation of the corresponding estimators and provide cost-efficient implementation. Robust estimation, however, induces nonquadratic objective functions that can be utilized in the representation of the prior signal statistics [16]. The existence of sharp edges manifests uncertainty regarding the distribution of the signal [17, 18]. To capture the detailed structure around sharp edges, most robust functions employed in these formulations induce limits and/or structural parameters that must be selected appropriately. Such functions have been motivated in stochastic maximum *a posteriori* (MAP) formulations under Markov random fields with nonquadratic Gibbs distributions [19, 20].

A MAP technique maximizes the conditional probability of a restored image given a certain blurred image. For MAP estimation, the image estimate $\hat{x}$ is given by

$$\hat{x} = \arg\max_x L(x|y)$$

where $L(x|y)$ is the log-likelihood function. This function can be computed by using Bayes' formula, with [21]

$$\begin{aligned} L(x|y) &= \log P_r(x|y) \\ &= \log P_r(y|x) + \log P_r(x) - \log P_r(y). \end{aligned} \tag{2.2}$$

Since the term $\log P_r(y)$ is independent of x, it can be eliminated from the optimization in (2.3). Then the MAP estimate of the image becomes

$$\begin{aligned} \hat{x} &= \arg\max_x \{\log P_r(y|x) + \log P_r(x)\} \\ &= \arg\min_x \{-\log P_r(y|x) - \log P_r(x)\} \end{aligned} \tag{2.3}$$

For a noisy system ($y = Ax + n$, where $n \neq 0$), the conditional density $P_r(y|x)$ is related to the noise density by

$$P(y|x) = P_r(n)\|_{n=y-Ax}$$

Thus, for cases with i.i.d. Gaussian noise, the conditional density has the Gaussian form

$$P_r(y|x) = \frac{1}{(2\pi\sigma^2)^{\frac{MN}{2}}} exp\left(\frac{-\|y - Ax\|^2}{2\sigma^2}\right)$$

where σ^2 is the variance of the density, M and N denote the dimensions of the image, and $\|.\|$ is the Euclidean norm.

2.3 Huber-Markov Random Field Model

For the second term $P_r(x)$ in (2.3), a Gaussian Markov random field (GMRF) model is commonly chosen as the prior image model. This density has the form

$$P_r(x) = \frac{1}{(2\pi)^{\frac{MN}{2}}|C|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}x^t C^{-1} x\right\}$$

where C is the covariance matrix. By decomposing C^{-1} into a sum of products, the above can be written as:

$$P_r(x) = \frac{1}{(2\pi)^{\frac{MN}{2}}|C|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\sum_{c\in\mathcal{C}} x^t d_c d_c^t x\right\}$$

where d_c is a coefficient vector for clique c and is determined by a priori assumptions about the image.

While the GMRF prior has many analytical advantages, it generally results in estimates $\hat{x}$ which are either excessively noisy or generally blurred. This is because the squared

difference of pixel values applies too high a penalty to edges that often occur in images [17]. Besides Gaussian Markov Random Field models, non-Gaussian MRF's are also interesting because they can potentially model both the edges and smooth regions of images. Early approaches often used an additional unobserved random field called a line process that determines the location of edges [22, 23]. Many non-Gaussian MRF methods have focused on MRF's with a simpler Gibbs distribution of the general form

$$\log g(x) = -\sum_{\{s,r\}\in C} b_{sr}\rho(\lambda|x_s - x_r|) + constant$$

where λ is a scaling parameter, and ρ is a monotone increasing but not convex function [24, 25, 20, 26]. However, analysis on non-Gaussian MRFs shows unstable behavior of the MAP estimate under the nonconvex prior [24]. Not only does the solution often depend substantially on the method used to perform the minimization, it often puts the same penalties on any edges. (see Fig 2.2 (a)) [17]. Consequently, the MAP estimate may abruptly change as the magnitude of an edge increases, which may lead to an unnatural quality in the reconstruction.

Recently, convex functions have also been considered for $\rho(\cdot)$ [19, 20, 27]. Green [20] employed a function of the form

$$\rho(\Delta) = 2T^2 \log cosh(\frac{\Delta}{T})$$

which produces useful Bayesian estimates of emission tomograms. This potential function is approximately quadratic for small Δ, and linear for large values. Lange [26] derived several other strictly convex potential functions in a study of convergence of the expectation

maximization algorithm. Stevenson and Delp studied the use of an alternative convex energy function for the problem of surface reconstruction [19]. They chose the Huber function first introduced in robust statistics [28].

An MRF image model with Gibbs density function has the form [21]

$$P_r(x) = \frac{1}{Z} \exp\left(-\frac{1}{\lambda} \sum_{c \in C} V_c(x)\right) \tag{2.4}$$

where Z is a normalizing constant, λ is the "temperature" parameter of the density, V_c is some function of a local group of points c called cliques, and $\mathcal{C}$ denotes the set of all cliques throughout the image [21, 22].

Let us denote the criterion to be minimized (the negative log-likelihood) as $M_\lambda[x, T]$. If the original image is corrupted with i.i.d. Gaussian noise, the functional for the noisy image will be:

$$M_\lambda[x, T] = \frac{\|y - Ax\|^2}{2\sigma^2} + \frac{1}{\lambda} \sum_{c \in \mathcal{C}} V_c(x)$$

The selection of $\sum_{c \in \mathcal{C}} V_c(x)$ is the key factor to the quality of the estimate. To make the problem well-posed, we only consider convex functions. A convex surface also has an obtainable global minimum, which makes the minimization job easier. Otherwise, local minima will be present, and the function must be minimized through a computationally expensive technique such as simulated annealing [21, 22].

The image prior in (2.4) can be rewritten as

$$P_r(x) = \frac{1}{Z} \exp\left\{-\frac{1}{\lambda} \sum_{c \in \mathcal{C}} \rho(d_c^t x)\right\}$$

where $\rho(\cdot)$ is some function satisfying the following properties:

- Convexity:

$$\forall \alpha, x, y \in R,$$

$$\rho[\alpha x + (1-\alpha)y] \leq \alpha\rho(x) + (1-\alpha)\rho(y),$$

- Symmetry:

$$\forall x \in R,$$

$$\rho(x) = \rho(-x),$$

- Allows regions of discontinuities: for $|x|$ large,

$$\rho(x) < x^2$$

The convexity property of $\rho(\cdot)$ insures that the sum of clique functions in the exponential argument remains convex. The smoothness measure $d_c^t x$ has a small magnitude in smooth regions and a large magnitude in discontinuous regions. Edges in an image are thus penalized to a lesser extent by a cost function $\rho(d_c^t x)$ that increases less rapidly than the quadratic cost $(d_c^t x)^2$ [21, 29].

Following the discontinuity-preserving stabilizing functionals in Tikhonov regularization, Stevenson and Delp used the Huber-Markov Random Field, where $\rho(\cdot)$ is defined as

$$\rho_T(x) = \begin{cases} x^2, & |x| \leq T \\ T^2 + 2T(|x| - T), & |x| > T \end{cases} \tag{2.5}$$

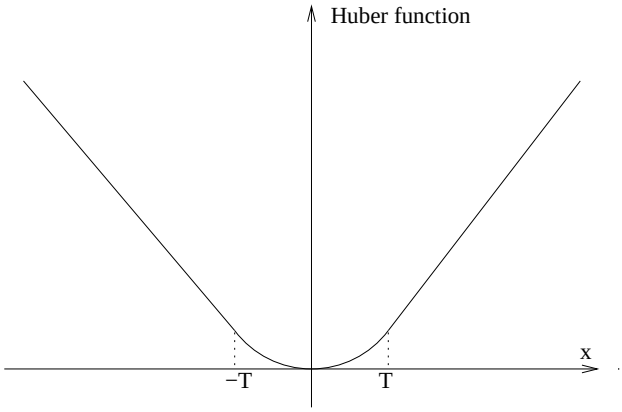


Figure 2.3: Huber function

The Huber function is convex and nonquadratic [28] (see Figure 2.3). The methodology here can be applied to any penalty function that is quadratic in a non-edge region around the origin. However, we restrict the development to the Huber function to keep the discussion specific.

By observing the definition above, we can see that below the threshold in $\rho_T(d_c^t x)$ the quadratic penalty applies. If the value is greater than the threshold, a linearly varying cost is used, and the discontinuities of the original image are not penalized as severely as with a quadratic. This characteristic provides an edge-preserving regularization effect.

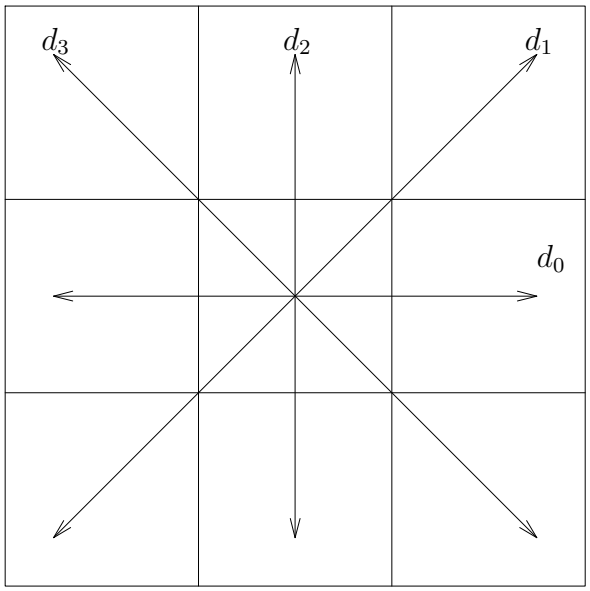
We consider an approximately rotationally symmetric operator within a 3×3 grid. Now the exponential kernel of the Huber-Markov random field (HMRF) image model is

$$\Omega[x, T] = \sum_{c \in \mathcal{C}} V_c(x) = \sum_k \sum_l \sum_{m=0}^{3} \rho_T(d_{k,l,m}^t x)$$

Finite-difference approximations to second-order derivatives are used to define the image roughness measure at pixel $x_{k,l}$, which are given as

$$
\begin{aligned}
d^t_{k,l,0}x &= x_{k,l+1} - 2x_{k,l} + x_{k,l-1} \\
d^t_{k,l,1}x &= \tfrac{1}{2}(x_{k-1,l+1} - 2x_{k,l} + x_{k+1,l-1}) \\
d^t_{k,l,2}x &= x_{k-1,l} - 2x_{k,l} + x_{k+1,l} \\
d^t_{k,l,3}x &= \tfrac{1}{2}(x_{k-1,l-1} - 2x_{k,l} + x_{k+1,l+1})
\end{aligned}
$$

These four discrete directional derivatives approximate a rotationally invariant operator such that approximately the same image is produced by the restoration regardless of image orientation [21].



2.4 Majorization

By using the MAP estimation and the Huber-Markov random field image model, we have the functional to minimize:

$$\begin{aligned} M_\lambda[x,T] &= \Omega[x,T] + \frac{\lambda}{2\sigma^2}\|y-Ax\|^2 \\ &= \sum_{c\in\mathcal{C}} V_c(x) + \frac{\lambda}{2\sigma^2}\|y-Ax\|^2 \\ &= \sum_k \sum_l \sum_{m=0}^{3} \rho_T(d^t_{k,l,m}x) + \frac{\lambda}{2\sigma^2}\|y-Ax\|^2 \end{aligned} \tag{2.6}$$

where $\rho(d^t_{k,l,m}x) = (d^t_{k,l,m}x)^t d^t_{k,l,m}x$ for non-edge pixels ($|d^t_{k,l,m}x| \leq T$) and $\rho(d^t_{k,l,m}x) = T^2 + 2T(|d^t_{k,l,m}x| - T)$ for edges ($|d^t_{k,l,m}x| > T$).

As previously mentioned, the HMRF is a convex but non-Gaussian MRF; i.e., it is nonquadratic and convex. The HMRF will pose a nonquadratic cost function that is not easy to minimize compared to the one based on the GMRF. To simplify the minimization process and guarantee convergence, one can use a quadratic surrogate function with the same minimizer $\hat{x}$ that is easy to minimize. Thus we can minimize (2.7) by minimizing the surrogate function.

The use of surrogate functions is common in solving minimization problems. Many variations have been proposed, among them is "majorization" proposed by Stoica and Selén [30].

A function $g(\theta)$ is said to majorize the function $f(\theta)$ at θ^i if

$$\begin{aligned} f(\theta^i) &= g(\theta^i) \\ f(\theta) &\leq g(\theta) \text{ for all } \theta \end{aligned}$$

and θ^i is called the majorizing point.

To minimize (2.7), we define a series of functionals $N_\lambda^i[x,T]$ such that

$$\begin{aligned} N_\lambda^i[x^i,T] &= M_\lambda[x^i,T] \\ N_\lambda^i[x,T] &\geq M_\lambda[x,T] \text{ for all } x \end{aligned} \tag{2.7}$$

Assuming that the minimization of $N_\lambda^i[x,T]$ at each i with respect to θ is easier than the minimization of $M_\lambda[x,T]$, we can minimize $N_\lambda^i[x,T]$ directly and quickly converge to the solution of $M_\lambda[x,T]$. This iterative algorithm is called majorization [30].

Let x^i denote the minimum of $N_\lambda^i(x,T)$. This point is then used as the majorizing point of the next function $N_\lambda^{i+1}(x,T)$, of which x^{i+1} is the minimum. This procedure is repeated until the minimum of $N_\lambda^j(x,T)$ at a certain j is close enough to

$$\arg\min_x M_\lambda[x^i,T].$$

This technique satisfies the descent property

$$M_\lambda(x^{i+1},T) \leq M_\lambda(x^i,T),$$

since

$$N_\lambda^{i+1}(x^{i+1},T) \leq N_\lambda^{i+1}(x^i,T).$$

Thus

$$M_\lambda(x^{i+1},T) \leq N_\lambda^{i+1}(x^{i+1},T)$$

$$\begin{aligned} &\leq N_\lambda^{i+1}(x^i, T) \\ &= M_\lambda(x^i, T). \end{aligned}$$

$M_\lambda(x^{i+1}, T) = M_\lambda(x^i, T)$ only when

$$N_\lambda^{i+1}(x^{i+1}, T) = N_\lambda^{i+1}(x^i, T) \tag{2.8}$$

and

$$M_\lambda(x^{i+1}, T) = N_\lambda^{i+1}(x^{i+1}, T) \tag{2.9}$$

i.e., when x^i is already the minimum of $M_\lambda[x^i, T]$ (2.8) and the majorizing function is $M_\lambda[x^i, T]$ itself (2.9). In other words, as long as we have not reached the global minimum

$$\arg\min_x M_\lambda[x^i, T],$$

then

$$M_\lambda(x^{i+1}, T) < M_\lambda(x^i, T).$$

Convergence of this algorithm is guaranteed.

To form the majorizing functional $N_\lambda^i[x, T]$ in this application, we need a quadratic function that has the same minimum. One practical choice $N_\lambda^i(\cdot)$ for $\rho(\cdot)$ is defined as follows:

$$N_\lambda^i(x) = \begin{cases} x^2 & |x^i| \leq T \\ \frac{T}{|x^i|}x^2 + T|x^i| - T^2 & |x^i| > T \end{cases} \tag{2.10}$$

where x^i is the majorizing point of function $N^i_\lambda(\cdot)$ at the ith iteration. The majorizing function is shown in Figure 2.4.

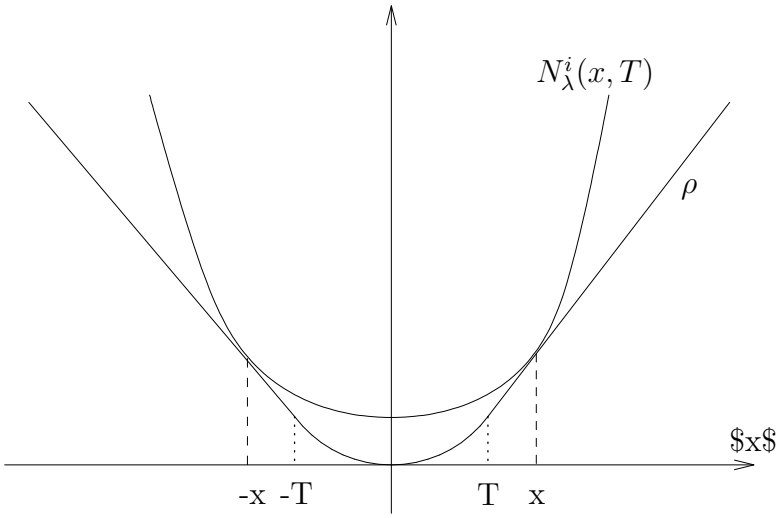


Figure 2.4: Majorizing function

Note that $N^i_\lambda(x)$ is quadratic everywhere, with a constant smoothing penalty where $|x| \leq T$ (non-edge regions) and a penalty that varies with x_i elsewhere (edge regions).

Let D_m denote the convolution operator corresponding to $\sum_{k,l} d^t_{k,l,m} x$. Consider the definition of $N^i(.)$. Since a constant term in the objective functional has no effect on the solution, $T|x^i| - T^2$ can be ignored, which leaves the modified $N^i_\lambda[x,T]$ (denoted as $\tilde{N}$) as:

$$\begin{aligned} & \tilde{N}^i_\lambda[x,T] \qquad (2.11) \\ &= \sum_k \sum_l \sum_{m=0}^3 \rho(d^t_{k,l,m} x) + \tfrac{\lambda}{2\sigma^2} \|y - Ax\|^2 \\ &= \sum_{m=0}^3 (D_m x)^T \Gamma^0_m D_m x + \tfrac{\lambda}{2\sigma^2} \|y - Ax\|^2 \end{aligned}$$

where Γ_m^0 are diagonal matrices whose elements are the quadratic scale factors in (2.10) (either 1 or $\frac{T}{|x^i|}$).

2.5 HMRF Edge-Preserving Regularization

Since the cost function (2.11) has the exact same structure as the constrained minimization method known as Tikhonov regularization, the minimum of $\tilde{N}_\lambda^i[x,T]$ is given below [31]:

$$\hat{x} = (A^T A + \alpha \sum_{i=0}^{3} D_i^T \Gamma_i D_i)^{-1} A^T y. \tag{2.12}$$

This system cannot be solved efficiently even though A and D_i represent shift-invariant operations because of the presence of the Γi. Suppose instead that we have a matrix

$$B = (A^T A + \alpha \sum_{i=0}^{3} D_i^T D_i)^{-1}. \tag{2.13}$$

B can be inverted via FFTs because it is shift-invariant. If we can rewrite the inverse in (2.12) in terms of B, we can invert it efficiently.

The Sherman-Morrison matrix inversion is given by:

$$(X + WYZ)^{-1} = X^{-1} - X^{-1}W(Y^{-1} + ZX^{-1}W)ZX^{-1}, \tag{2.14}$$

Applying this formula to (2.12),

$$\begin{aligned} & (A^T A + \alpha \textstyle\sum_{i=0}^{3} D_i^T \Gamma_i D_i)^{-1} \qquad (2.15) \\ = \quad & (B^{-1} - \alpha \textstyle\sum_{i=0}^{3} D_i^T (I - \Gamma_i) D_i)^{-1} \\ = \quad & (B^{-1} - \alpha \textstyle\sum_{i=0}^{3} d_i^T d_i)^{-1} \end{aligned}$$

$$= \quad (B^{-1} - \alpha d^T d)^{-1}$$

$$= \quad B + \alpha B d^T [I - dBd^T]^{-1} dB$$

In the above expressions,

$$d = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix}$$

where $d_0, d_1, d_2,$ and d_3 are matrices formed from the rows of $D_0, D_1, D_2,$ and D_3 corresponding to the diagonal elements of $\Gamma_0, \Gamma_1, \Gamma_2,$ and Γ_3 that are not equal to 1. From this, it follows that:

$$\begin{aligned} \hat{x} &= (B + \alpha B d^T [I - dBd]^{-1} dB) A^T y \\ &= \hat{x}_\alpha + B d^T \omega \end{aligned} \tag{2.16}$$

where

$$\hat{x}_\alpha = B A^T y \tag{2.17}$$

and

$$\omega = \alpha [I - \alpha \; dBd^T]^{-1} d\hat{x}_\alpha \tag{2.18}$$

Thus the estimate $\hat{x}$ is computed in two steps. The first step, expressed by (2.17) is a shift-invariant restoration. It generates a basic solution with all the artifacts associated with the ringing effect. Then the basic solution is corrected by a second term, which is obtained through a shift-variant method involving values corresponding to edges only in

the image. Generally, a shift-variant method is computationally expensive. But in most cases, if the two-dimensional image has N^2 pixels, the number of edge pixels in an image is $O(N)$. This makes the shift-variant component a much lower-dimensional problem. Thus, the computation is reduced significantly [32].

By formulating the solution for the estimate $\hat{x}$ as (2.16), the fixed-point iteration is assumed. Notice the choice of rows in the D_i's depends on $\hat{x}$. In other words, the matrix d is a function of $\hat{x}$. The iteration process can be described as follows:

1. Given an initial $\hat{x}$, use HMRF as the prior image model and find out Γ_i and then d using Huber function.

2. Compute the next $\hat{x}$ using (2.16).

3. Plug in the new $\hat{x}$ to find out the new d and repeat the second step.

By majorizing the original HMRF model, the minimization has become a quadratic problem for each iteration. But it is nonquadratic overall, since the majorizing function varies with the result of each iteration. Similar ideas such as the half-quadratic method have been proposed [33, 34]. Half-quadratic regularization formulates the cost function dependent on an auxiliary variable. Like our method, when the auxiliary variable is fixed, it becomes a constrained least squares problem with quadratic regularization even though it is not quadratic overall [33]. In our method, the auxiliary variables are related to a map of edges in the image. Different smoothing penalties are put in the cost function according to the map. For each fixed edge map, the problem becomes a quadratic one. With varying edge maps, the overall criterion is not quadratic.

With majorization, we set up a new minimization criterion which converges to the solution of the original criterion. The new minimizing technique using the Sherman-Morrison matrix inversion lemma is applied to improve the computational performance of the algorithm. Then we can apply the two-step algorithm including both shift-invariant and shift-variant methods to minimize the Huber criterion. In the next section, we compare our method to conjugate gradients, which is well established as an efficient minimization method. Since this algorithm is based on decomposing the blurred image into two components and applying shift-invariant and shift-variant restorations, we refer to it as decomposition-based edge-preserving image restoration (DEEPIR).

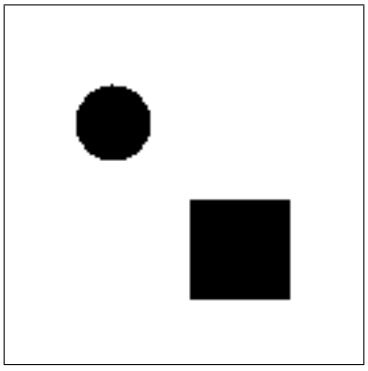
2.6 Experiment

In this section, the performance of DEEPIR is examined by simulation. Experiments were conducted using 128-by-128 images with different shapes and letters as shown in Figure 2.5. The original image was blurred with a 5×5 averaging blur, a 9×1 motion blur and a 7×7 Gaussian blur with standard deviation 1 and 30 dB Gaussian noise. All computation was done on a workstation with an AMD Opteron 280 processor.

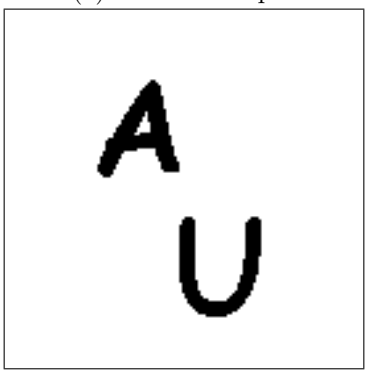
The blurred images, corresponding shift-invariant restoration, and restoration with DEEPIR are shown in Figure 2.6.

We observe that the new algorithm considerably reduces the ringing effects on edges. The mean square error (MSE) between the original and the restored image is now less than 40% of that of shift-invariant restoration, as shown in Tables 2.4 and 2.2.

This new algorithm not only has better edges but is also efficient. Experiments on the above images show a reduced computation time compared to conjugate gradients (CG), as



(a) different shapes



(b) letters

Figure 2.5: Original images

shown in Fig 2.7. Fig 2.7 shows how the criterion decreases vs. time in CG and DEEPIR. Each circle on the DEEPIR curve denotes one iteration. Each diamond on the CG curve denotes 40 iterations. We can see that each iteration in the DEEPIR takes longer than conjugate gradients, but the criterion goes down faster. This indicates that DEEPIR requires more computation at each iteration, but it needs fewer iterations than CG to reach the same level of precision. For conjugate gradients, the curve is smooth since each iteration takes much less time than DEEPIR. Table 2.3 gives specific computing times for each image and PSF.

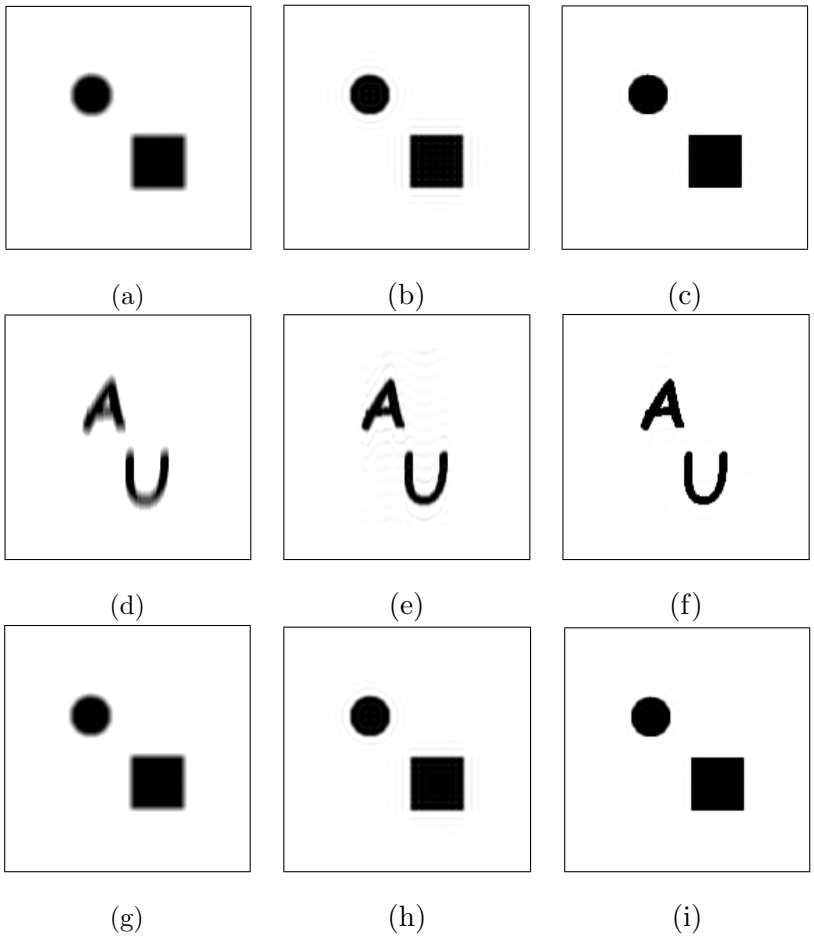


Figure 2.6: Restored images
(a) original image blurred with 5×5 averaging blur and 40dB Gaussian noise (b) shift-invariant restoration for averaging blur (c) restoration by DEEPIR for averaging blur (d) original image blurred with 9×1 motion blur and 40dB Gaussian noise (e) shift-invariant restoration for motion blur (f) restoration by DEEPIR for motion blur (g) original image blurred with 7×7 Gaussian blur with standard deviation 1 and 40dB Gaussian noise (h) shift-invariant restoration for Gaussian blur (i) restoration by DEEPIR for Gaussian blur.

Table 2.1: MSE of shape Image

Blur	MSE of shift-invariant restoration	MSE of DEEPIR
Gaussian	55	6.26
Motion	86	9.33
Average	165.56	58.87

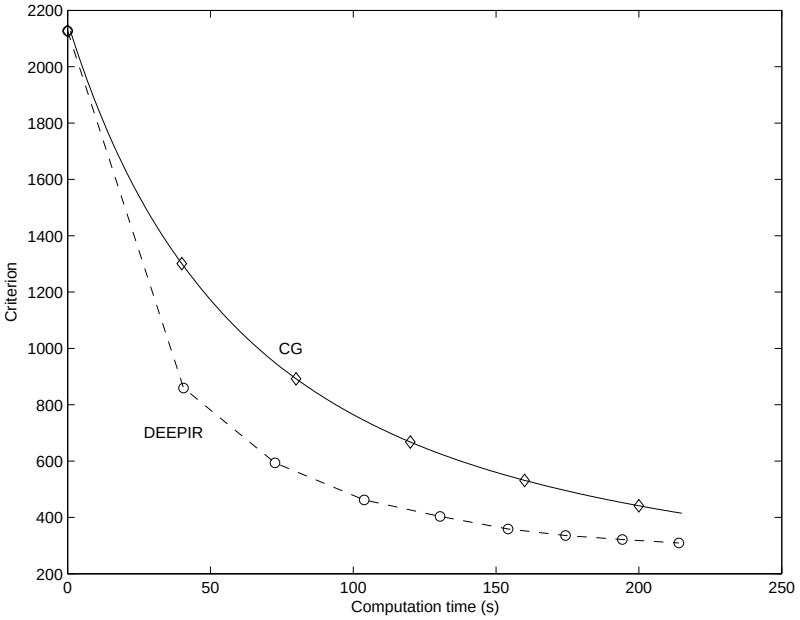
Table 2.2: MSE of letter Image

Blur	MSE of shift-invariant restoration	MSE of DEEPIR
Gaussian	83.66	11.69
Motion	103.77	18.64
Average	140.93	55.89

Figure 2.8 is another example showing the restoration of a picture of Samford Hall at Auburn University. The original 256×256 image was blurred with a 5×5 averaging blur and 40 dB Gaussian noise. The Huber threshold T is 25 and $\alpha = 0.05$. The blurred image, the shift-invariant restoration, and restoration with DEEPIR are shown in Figure 2.8 (b)(c)(d). Table 2.4 compares MSE and computing time among shift-invariant restoration, CG and DEEPIR.

2.7 Block-Based Implementation

The computational complexity of DEEPIR grows with the cube of the number of edges, and the memory usage grows with the square of the number of edges. Consequently, the computation or memory requirements may be prohibitive for large images. In DEEPIR, only a system whose size corresponds to the number of edge pixels must be solved. The

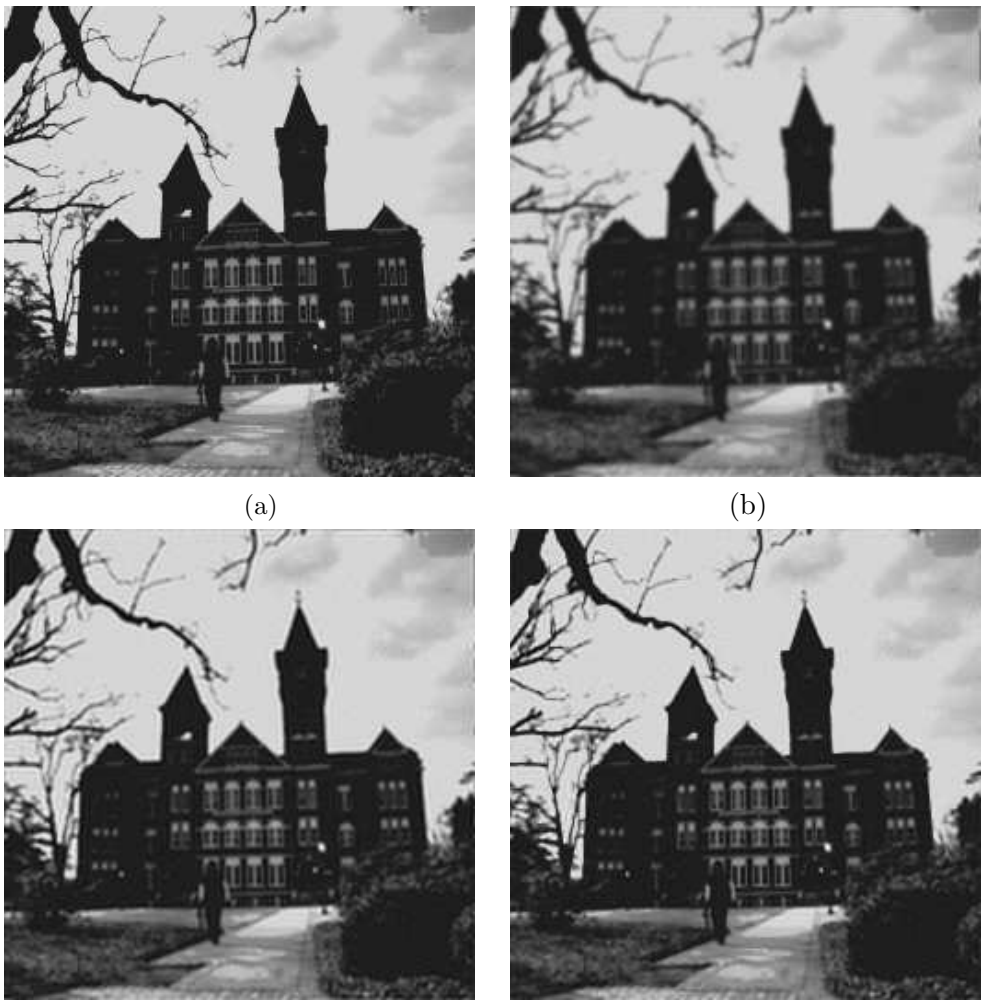


criterion vs. time

Figure 2.7: Experiment on 'shape'

fewer the edge pixels, the faster the system solution. Therefore, we introduce a modification to DEEPIR by partitioning the system into sub-systems. Instead of solving one big system, now we solve a group of smaller systems. To do this, we divide the image into k^2 equal-size blocks where k denotes the number of blocks we have in each coordinate, i.e., $k = 1$ represents the non-block method; $k = 2$ means we have 4 blocks, 2 on each side of the image; $k = 4$ represents 16 blocks and so on [35]. Then we group the unknown edge values according to their block membership.

We apply a modified block Gauss-Seidel (BGS) approach to limit the growth in computational resources and speed up the algorithm. Further improvement in computing time is possible with a block approximation method proposed later in this section. Generally, block iterative methods require more computation per iteration, but this is offset by a fast



(a) (b)

(c) (d)

Figure 2.8: Samford hall
(a) original Samford hall image (b) blurred image (c) shift-invariant restoration (d) DEEPIR restoration

Table 2.3: Computation Time

Blur	Letters (s)		Different shapes (s)	
	DEEPIR	CG	DEEPIR	CG
Gaussian	12.17	24.5	4.81	8.89
Motion	8.23	22.7	4.15	8.33
Average	6.44	21.3	3.71	8.16

Table 2.4: Mean Square Error and Time

	MSE	Time (s)
Shift-invariant	791.7	8.9
CG	308.1	301.3
DEEPIR	308.1	196.2

rate of convergence [36]. The most well-established block iterative methods include block Gauss-Seidel and block Jacobi.

Let $Q = I - \alpha d B d^T$, and $b = \alpha d \hat{x}_\alpha$. To solve $Q\omega = b$ for ω, the system Q is partitioned into $k \times k$ sub-systems such that each group of edge variables in ω corresponds to one partition of Q.

$$\begin{pmatrix} Q_{11} & Q_{12} & \dots & Q_{1k} \\ Q_{21} & Q_{22} & \dots & Q_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{k1} & Q_{k2} & \dots & Q_{kk} \end{pmatrix}.$$

Then it is split as follows:

$$Q = D - (L + U), \tag{2.19}$$

where

$$D = \begin{pmatrix} D_{11} & 0 & \dots & 0 \\ 0 & D_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & D_{kk} \end{pmatrix},$$

and

$$L = \begin{pmatrix} 0 & 0 & \dots & 0 \\ L_{21} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{k1} & L_{k2} & \dots & 0 \end{pmatrix},$$

$$U = \begin{pmatrix} 0 & U_{12} & \dots & U_{1k} \\ 0 & 0 & \dots & U_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}.$$

The Gauss-Seidel iterative procedure is given by

$$D\omega^{r+1} = L\omega^{r+1} + U\omega^{r} + b. \tag{2.20}$$

In block Gauss-Seidel, only the D_{ii} matrices need to be inverted at each iteration, since L is strictly block-lower triangular. With this partitioning, the D_{ii} matrices are equivalent to a system that would arise if only the ith image block contained edges. To make the algorithm efficient, a few relaxations of (2.20) are applied to get ω for a fixed Q. Then the

matrix Q is updated according to the edge locations in the current estimate of the image. This constitutes a single iteration of DEEPIR in this context.

The cameraman image was blurred with a 7×7 circular averaging blur with radius 3 pixels. Gaussian noise was added to a level of 40 dB SNR. We applied three relaxations of (2.20) for each iteration. The regularization parameter was chosen to be $\alpha = 0.05$. Since the goal of this work is primarily to demonstrate the computational efficiency of the algorithm, no attempt was made to optimize the regularization parameter α.

Table 2.5 shows significant improvement on computation time. Theoretically, the larger the block size, the fewer the iterations needed to achieve convergence [36]. On the other hand, the larger the block size, the more computation needed for each block and therefore the slower the convergence. As shown in Table 2.5, it is the most efficient when $k = 2$ for the cameraman image.

Table 2.5: Time and MSE for BGS

k	Time (s)	MSE
1	316.6	897
2	138.0	897
4	208.5	897
8	287.3	897

2.7.1 Block-based Approximation

The modified block Gauss-Seidel has proved to improve the performance of DEEPIR. However, there is still more room for speed enhancement when a small sacrifice is made in image quality. We divide the edge image into several blocks and apply the edge-preserving regularization to each block independently. That is, we treat the whole image as having

edges only in a particular block and apply DEEPIR to get a correction image that is edge-preserving only in the particular block; we repeat this procedure for every part of the image. Then we extract the solution in each block to put together a full correction image. The correction image is added to the shift-invariant restoration to get an edge-preserving result. Due to the smaller number of edge pixels in each block, all the block systems can be solved independently much more efficiently than the full system at once. Thus we can reduce the computation to a larger extent in general and also deal with large images.

The computation in inverting an $M \times M$ matrix is $O(M^3)$. Suppose the edge image is decomposed into k^2 blocks, where k is the number of blocks in each coordinate. If we assume edges are evenly distributed in each block, the computation needed to invert matrices associated with these blocks is $k^2 O((\frac{M}{k^2})^3) = \frac{1}{k^4} O(M^3)$. Theoretically, the bigger k is, the faster the inversion is; therefore, the more efficient DEEPIR is. Certain limitations on k are necessary to ensure acceptable quality, as discussed in the next section.

To avoid errors on the boundary of each small block, we use overlapping blocks. That is, we apply the edge-preserving regularization to a larger part of the image, then crop the central part and use it as the result. In Figure 2.9, edges inside the dashed square are used while the shaded part is saved as the result.

It needs to be noted that the approximation method is related to block Gauss-Seidel in that in the non-overlapping case, it leaves out L and U in (2.19) and treats Q as its block diagonal D. By ignoring L and U, the blocks are considered uncoupled. This is motivated by the fact that edge values in different blocks are very loosely correlated. It does not give us the exact solution but an efficient one as long as the tradeoff in image quality is still tolerable.

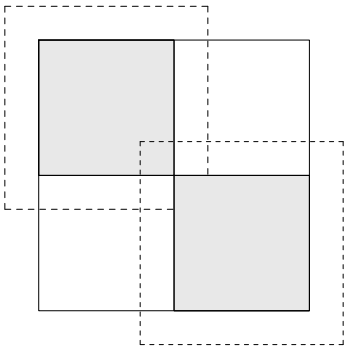


Figure 2.9: Overlapping blocks

2.7.2 Experiment on Block-based Approximation

To test the approximation method, we applied the same blur and noise to the cameraman image as in block Gauss-Seidel. The image shown in Figure 2.10(c) was restored using a standard circular deconvolution algorithm with 2-D FFT's. Figure 2.10(d)(e)(f) show images restored using the block-based algorithm with four 128×128, sixteen 64×64 and sixty-four 32×32 blocks, respectively.

In Table 2.6, it is shown that with more blocks in the image, the mean square error increases while the computation time decreases. But when we continue to reduce the size of each block, the gains in speed decrease. This happens because there is a fixed overhead for restoring each block regardless of size. Using bigger blocks produces better image quality but with more computation since a bigger edge matrix must be inverted.

2.7.3 Discussion about Block-based Approximation

In the above experiments, we used overlapping blocks with an overlapping margin of 4. In image restoration, each restored pixel value in the image affects the restored values of pixels within a certain neighborhood. Similarly, the existence of an edge pixel has a

Table 2.6: Time and MSE for Block-Based Approx.

k	Time (s)	MSE
Conjugate Gradients	315.8	896
2	102.4	937
4	81.6	982
8	70.9	990

larger impact on the restored image in a neighborhood near the edge pixel. When the size of the affected neighborhood is small enough, the error caused by decomposing the image into blocks is also small. In the block-based method, we only consider one block of edges at a time; i.e., we are ignoring the effect of other edge pixels on this particular edge block. When there is more overlapping among the blocks, more edge pixels that could possibly affect the restored image nearby are included, thereby reducing error.

Under certain circumstances, preconditioning can make the CG algorithm more competitive [37]. However, the application of a preconditioner requires some further design choices and computational complexity. Furthermore, our method itself can be used as a preconditioner in a CG algorithm by using the block-based method as an approximate solver within an iterative framework. Define the matrix M by the block-based approximation such that the exact solution

$$x = (A^T A + \alpha \sum_{i=0}^{3} D_i^T \Gamma_i D_i)^{-1} A^T y.$$

is replaced by an approximate solution

$$x = M A^T y, \tag{2.21}$$

where M is the block-based approximation of $(A^TA + \alpha \sum_{i=0}^{3} D_i^T \Gamma_i D_i)^{-1}$. Since M is easy to calculate and includes information about edges, it could help reduce computational time in CG.

Let l be the number of blocks in the block-based restoration, r be the number of iterations. Assuming the number of edges is proportional to n, the computation for an $n \times n$ image is proportional to $21rn^2 \log_2 n + \frac{2r}{l} n^3$. Increasing the number of blocks will reduce the computation significantly since it is mainly proportional to $\frac{n^3 r}{l}$. For $l = 1$, i.e., the non-block restoration, the computation is proportional to $21rn^2 \log_2 n + 2rn^3$. When the number of blocks changes from 4 to 16 for a 256×256 image, the computation drops by almost one third from 19.3×10^6 to 13.1×10^6 multiplications.

2.7.4 Analysis of Overlapping

Our experiments show that in the block-based approximation the restored image is not as faithful to the original image as the non-block method. The MSE gets worse as the size of the blocks is decreased. In other words, we see the effect of different levels of approximation with different block sizes. Meanwhile, the overlapping margin also affects the error because it determines the actual block size used in calculation. Using a higher overlapping margin reduces error while increasing computation. In order to study the effect of the overlapping margin, we consider a simplified case to examine the effect that one edge pixel has on its neighbors. A simple model function with only one edge pixel is used. The edge-preserving restoration now becomes

$$\hat{x}_{\alpha,S} = \hat{x}_\alpha + \alpha B l_i^T [I - \alpha l_i B l_i^T]^{-1} l_i \hat{x}_\alpha \tag{2.22}$$

where l_i is a row vector from D corresponding to a particular edge location i. The difference between the image without any edges and the image with one edge pixel is

$$E = \hat{x}_{\alpha,S} - \hat{x}_\alpha = \alpha B l_i^T [I - \alpha l_i B l_i^T]^{-1} l_i \hat{x}_\alpha. \tag{2.23}$$

Since $[I - \alpha l_i B l_i^T]$ is a scalar,

$$E = \frac{\alpha}{1 - \alpha l_i B l_i^T} B l_i^T l_i \hat{x}_\alpha \tag{2.24}$$

Since l_i is a row vector and $\hat{x}_\alpha$ is the column-ordered image, $l_i \hat{x}_\alpha$ is a scale factor. Now the difference becomes:

$$E = \mathcal{K} B l_i^T \tag{2.25}$$

where $\mathcal{K} = \frac{\alpha}{1-\alpha l_i B l_i^T} l_i \hat{x}_\alpha$ is a scale factor. Notice E is an image showing the difference between the shift-invariant restoration and the edge-preserving restoration. In this particular case, we only care about the distribution of the image instead of the actual pixel values because the purpose of this analysis is to find out how one single edge pixel will affect its neighborhood as a function of distance from the edge pixel. The scalar factor $\mathcal{K}$ will vary in different parts of the image and will tend to be higher where i is any index corresponding to pixels near an edge. However, the values die out only a few pixels away from the edge pixel, so the effect of this edge is still negligible. In this case, a small overlapping margin will not introduce much error.

The distribution of E only depends on distance from the edge pixel. Since l_i is a row vector from a BCCB (block circulant with circulant blocks) matrix, a different edge location

gives an l_i with the same elements circularly shifted. Thus, the distribution around the edge pixel will be the same around every location.

To illustrate the E image, we calculated a normalized version of E. Figure 2.11 shows the absolute value of the normalized image E. The spreading pattern in Figure 2.11 seems to be confined to a very small neighborhood around the edge location and dies out very fast as it goes away from the center. According to Figure 2.11, the effect between two pixels that are 6 pixels apart is close to zero. Thus a very small overlap can be used to avoid unnecessary computation without neglecting the most important neighbor pixels.

(a) original image

(b) blurred image

(c) shift-invariant restoration

(d) DEEPIR with $k = 2$

(e) DEEPIR with $k = 4$

(f) DEEPIR with $k = 8$

Figure 2.10: Block-based restoration of 'cameraman' image

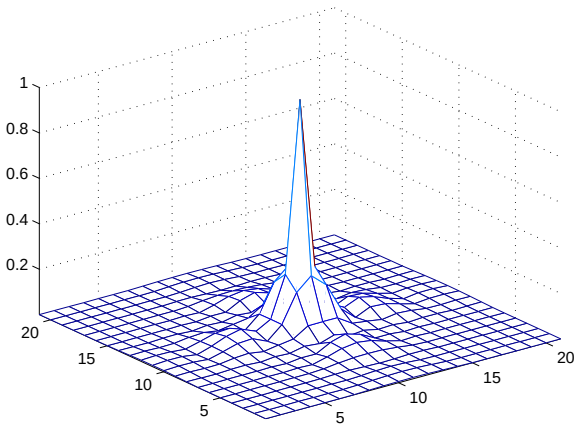


Figure 2.11: Normalized E image

Chapter 3

Fast Algorithm for Solving Block Banded Toeplitz Systems with Banded Toeplitz Blocks

3.1 Introduction

In many image restoration problems, linear and shift-invariant systems are assumed such that the blurring procedure can be expressed as

$$g(x,y) = h(x,y) * f(x,y) + n(x,y) \tag{3.1}$$

where $*$ stands for linear convolution and $n(x,y)$ is the additive noise that is independent of the signal. Rewrite the above system in matrix form, it becomes

$$g = Hf + n \tag{3.2}$$

where g, f and n are the column-ordered version of the blurred image, the original image and noise of size $n^2 \times 1$ for an $n \times n$ image. To represent the convolution operation, matrix H has block-Toeplitz-with-Toeplitz-block (BTTB) structure.

Many problems arising in signal and image processing are in fact characterized by matrices with a block-Toeplitz-with-Toeplitz-block (BTTB) structure in which both the Toeplitz blocks and the block structure are banded. This is due to the fact that most point spread functions (PSF) h have finite extent and are much smaller than the image. When the image formation equations are written with the original and blurred images ordered as single

vectors, either row- or column-wise, the matrix that describes the blurring operation has a banded block-Toeplitz structure with banded Toeplitz blocks [2]. Also, in some non-banded cases, the matrix may be approximated by such a banded structure.

On the other hand, most image restoration algorithms assume circulant systems to simplify the computation by making use of FFT's while a Toeplitz system should be used instead according to the definition of the point spread function. For example, the discussion about edge-preserving restoration methods in the previous chapter assumes a block-circulant-with-circulant-block (BCCB) system. Thus artifacts on image boundaries (not edges) are ignored. This is often the case because FFTs significantly improves computational performance with a small price in image quality. For most images, information on boundaries is not as important as that in the center of the image.

Although the BCCB assumption is used in many algorithms, it is not the exact representation of the original system and can pose problems when there is important information along the image boundaries. Fast algorithms that help solve Toeplitz systems have been proposed since the late 1960's and are still being studied by many not only for their applications in image processing but also general purposes in solving linear systems.

It is well known that the solution of a general Toeplitz system can be found efficiently using the Levinson algorithm or the Schur algorithm [38, 39]. These algorithms require $O(N^2)$ operations for an $N \times N$ matrix or $O(N \log^2 N)$ if a superfast algorithm is used [40, 41]. (Superfast algorithms are generally only competitive for very large N due to a large scale factor.) Efficient algorithms that exploit the band structure of a banded Toeplitz matrix using the Schur algorithm require $O(kN)$ operations for a bandwidth of k. More

efficient methods have also been developed that require $O(N \log k + k^2 \log \frac{N}{k})$ operations [42].

The case of block-Toeplitz matrices with Toeplitz blocks is much more problematic. Algorithms that exploit the block-Toeplitz structure usually destroy the Toeplitz structure of the blocks in the process, and vice versa. Thus, general solvers for block-Toeplitz-with-Toeplitz-block (BTTB) systems require $O(M^5)$ operations for $M^2 \times M^2$ matrices with $M \times M$ blocks [38, 43, 44]. The banded Toeplitz outer (inner) structure can be exploited in a Schur algorithm to yield an operation count of $O(kM^4)$. However, this does not fully exploit the inner (outer) banded Toeplitz structure. Since multiplication of a BTTB matrix by a vector can be performed in $O(M^2 \log M)$ operations [45], iterative algorithms can be used. However, iterative algorithms can take thousands of iterations to converge and are not always parallelizable [46]. Furthermore, iterative algorithms must repeat the bulk of the computation for each new input even if the system is the same. A fast algorithm exploiting the doubly Toeplitz structure with $O(6M^3)$ flops has been proposed in [46], but it requires the assumption that the PSF is bandlimited.

In this chapter, we discuss a new algorithm that not only makes use of the BTTB structure, but also exploits the bandedness of the system. By exploiting a form of the generalized Schur algorithm in [38, 47], this algorithm only requires $O(k^2M^3)$ operations without any further assumption about the BTTB system.

3.2 Solution Formulation

We consider a matrix T to be a square block-Toeplitz matrix of order M and block bandwidth k if

$$T_{\{M^2 \times M^2\}} = \begin{bmatrix} t_0 & t_{-1} & \cdots & t_{1-k} & 0 & \cdots \\ t_1 & t_0 & t_{-1} & \cdots & t_{1-k} & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ t_{k-1} & t_{k-2} & \cdots & t_0 & t_{-1} & \cdots \\ 0 & t_{k-1} & t_{k-2} & \cdots & \ddots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \cdots & t_0 \end{bmatrix} \tag{3.3}$$

where each block entry $t_i, i = 1-k, 2-k, \ldots, 0, \ldots, k-1$ is an $M \times M$ Toeplitz matrix with bandwidth k. No symmetry is assumed in (3.3), i.e., $t_i = t_{-i}, i = 1, \ldots, k-1$ is not necessary. The algorithm is easily generalized to the case where the block size is not equal to the number of blocks and where the bandwidth in each block is not equal to the block bandwidth. We consider the simpler case for simplicity of presentation. We desire to find x such that

$$y = Tx \tag{3.4}$$

Here the noise factor is also ignored for the purpose of simplifying the description. By focusing on the structure of the blurring operation T, we are able to show the mechanism of the algorithm in an effective fashion. The impact of noise can always be added later on.

Define T_c as the matrix T extended so that the structure of T_c is block-circulant with circulant blocks (BCCB) having dimensions of (at least) $(M+k-1)^2 \times (M+k-1)^2$.

$$T_c = \left[\begin{array}{cccccc|ccc}
c_0 & \cdots & c_{1-k} & 0 & \cdots & 0 & c_{k-1} & \cdots & c_1 \\
c_1 & c_0 & \cdots & c_{1-k} & 0 & \cdots & 0 & c_{k-1} & \cdots \\
\vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \cdots & \ddots & \vdots \\
c_{k-1} & c_{k-2} & \cdots & c_0 & \cdots & c_{1-k} & 0 & \cdots & 0 \\
0 & c_{k-1} & c_{k-2} & \cdots & c_0 & \cdots & c_{1-k} & 0 & \cdots \\
\vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \cdots & \ddots & \cdots \\
0 & \cdots & 0 & c_{k-1} & c_{k-2} & \cdots & c_0 & \cdots & c_{1-k} \\
\hline
c_{1-k} & 0 & \cdots & 0 & c_{k-1} & c_{k-2} & \cdots & c_0 & \cdots \\
\vdots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \cdots & \cdots \\
c_{-1} & \cdots & c_{1-k} & 0 & \cdots & 0 & c_{k-1} & c_{k-2} & \cdots
\end{array}\right] \tag{3.5}$$

Each block entry $c_i, i = 1-k, 2-k, \ldots, 0, \ldots, k-1$ is an $(M+k-1) \times (M+k-1)$ circulant matrix expanded from corresponding Toeplitz matrix t_i in the same fashion.

Then define unitary permutation P, i.e., $PP^T = P^TP = I$, such that in $\tilde{T}_c = PT_cP^T$ the blocks augmenting the inner Toeplitz structure and the circulant blocks augmenting the outer structure are in the upper left corner [48]. Details of this permutation are shown in the experiment section. Following the procedure in [46], we can rewrite (3.4) as a 2-D circular convolution operation

$$\begin{aligned}
\begin{bmatrix} z \\ y \end{bmatrix} &= \tilde{T}_c \begin{bmatrix} \mathbf{0} \\ x \end{bmatrix} \\
&= \begin{bmatrix} C & D \\ B & T \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ x \end{bmatrix}
\end{aligned} \tag{3.6}$$

The vector z is an unknown extension of y. If z were known, the right-hand side could be calculated directly using a 2-D FFT-based deconvolution operation.

From (3.6) we can write

$$\begin{aligned} \begin{bmatrix} \mathbf{0} \\ x \end{bmatrix} &= \tilde{T}_c^{-1} \begin{bmatrix} z \\ y \end{bmatrix} \\ &= \begin{bmatrix} Y & W \\ V & U \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} \end{aligned} \tag{3.7}$$

Then we must solve

$$-Wy = Yz \tag{3.8}$$

for z to find the solution to the original problem.

3.3 Generalized Displacement Rank

A BCCB system will remain block-circulant with circulant blocks after inversion [49], so

$$\begin{aligned} \tilde{T}_c^{-1} &= (PT_cP^T)^{-1} \\ &= PT_c^{-1}P^T \end{aligned} \tag{3.9}$$

where T_c^{-1} is a BCCB system with the same block structure as T_c and $\tilde{T}_c^{-1}$ is the permuted version of T_c^{-1} with the same block structure as $\tilde{T}_c$.

The matrix Y in (3.8) is a $(k-1)(2M+k-1) \times (k-1)(2M+k-1)$ matrix with a special structure; that is, Y is permuted so that the blocks augmenting the inner Toeplitz

structure to a circulant structure are permuted to the lower right corner of Y, while the circulant blocks augmenting the outer block structure are left in the upper left corner. This matrix has a near BTTB structure. It is not block Toeplitz but has Toeplitz blocks where the blocks are not equal in size. Furthermore, the matrix contains some structure that is not immediately obvious from inspection. This structure is best revealed using the concept of generalized displacement rank [38].

3.3.1 Definition and properties

Let Z be a strictly lower triangular matrix and A be an $N \times N$ matrix. Then the matrix

$$\Delta_Z(A) = A - ZAZ^T \tag{3.10}$$

is the positive displacement of A with respect to the displacement operator Z and $\Delta_{Z^T}(A)$ is the negative displacement. Then

$$\alpha_+(A) = \operatorname{rank} \Delta_Z(A) \tag{3.11}$$

is the positive displacement rank of A and

$$\alpha_-(A) = \operatorname{rank} \Delta_{Z^T}(A) \tag{3.12}$$

is the negative displacement rank of A.

One can show [50] that

$$\alpha_+(A) = \alpha_-(A^{-1}) \tag{3.13}$$

and

$$\alpha_-(A) = \alpha_+(A^{-1}) \tag{3.14}$$

Furthermore, we can represent A as

$$A = \sum_{i=1}^{\alpha_+(A)} L(x_i)U(y_i^T) \tag{3.15}$$

where $L(x_i)$ ($U(y_i^T)$) denotes a lower (upper) triangular Toeplitz matrix whose first column (row) is x_i (y_i^T). Likewise,

$$A = \sum_{i=1}^{\alpha_-(A)} U(\tilde{x}_i^T)L(\tilde{y}_i) \tag{3.16}$$

where $\tilde{x} = [x_N \cdots x_1]^T$. Thus, a matrix with displacement rank α can be fully represented by vectors $x_i, y_i, i = 1, \ldots, \alpha$. Furthermore, procedures exist for computing the LU (UL) decomposition of a matrix with positive (negative) displacement rank α in $O(\alpha N^2)$ operations [38]. Procedures also exist for converting from representation (3.15) for A to representation (3.16) for A^{-1} in $O(\alpha N^2)$ operations.

3.3.2 Choice of Z

The key to solving (3.8) as efficiently as possible is to represent Y in terms of (3.15) for an appropriate choice of displacement operator Z. Define Z_c as an $(M+k-1)^2 \times (M+k-1)^2$ matrix with $(M+k-1) \times (M+k-1)$ identity matrices on the first block subdiagonal and

zeros elsewhere.

$$Z_c = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots \\ \vdots & \ddots & \ddots & \ddots & \cdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \tag{3.17}$$

where $\mathbf{I}$ is an $(M+k-1) \times (M+k-1)$ identity matrix. It should be noted that Z_c is a block shift matrix [38]; i.e., for any matrix X, $Z_c X$ shifts rows of X down by the size of the block, which is $M+k-1$ in this case. XZ_c^T shifts columns of X to the left by the size of the block.

Then define Z as

$$Z = PZ_cP^T \tag{3.18}$$

Theorem 1 *For Z defined above,*

$$\alpha_+(Y) \leq 2(M+k-1)$$

Proof: Using Z defined above and knowledge of $\tilde{T}_c$,

$$\begin{aligned} \Delta_Z(\tilde{T}_c) &= \tilde{T}_c - Z\tilde{T}_cZ^T \\ &= PT_cP^T - PZ_cP^TPT_cP^TPZ_c^TP^T \\ &= PT_cP^T - PZ_cT_cZ_c^TP^T \\ &= PT_cP^T - P\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & T_c(1,1) \end{bmatrix} P^T \end{aligned} \tag{3.19}$$

$$
\begin{aligned}
&= P \begin{bmatrix} T_{c_0} & T_{c_2} \\ T_{c_1} & \mathbf{0} \end{bmatrix} P^T \\
&= P\Delta_Z(T_c)P^T.
\end{aligned}
$$

Let $N_1 = (M+k-1)(M+k-2)$, $N_2 = M+k-1$. $Z_c T_c Z_c^T$ shifts T_c to the left and down by N_2. Since T_c is a block-circulant matrix, $T_c(1,1)$ is an $N_1 \times N_1$ principle submatrix of T_c obtained by deleting the first block row and the first block column of T_c. Then $T_{c_0\{N_2\times N_2\}}$, $T_{c_2\{N_2\times N_1\}}$ and $T_{c_1\{N_1\times N_2\}}$ are the upper left, upper and left part of T_c, respectively. Since $\tilde{T}_c^{-1}$ has the same block structure as $\tilde{T}_c$, $\Delta_Z(\tilde{T}_c^{-1})$ has the same structure as $\Delta_Z(\tilde{T}_c)$ in (3.19). Following (3.9), $\Delta_Z(T_c^{-1})$ can also be expressed as

$$
\begin{bmatrix} & | & \\ T_{col\{N\times N_2\}} & | & T_{row\{N_2\times N_1\}} \\ & | & \mathbf{0} \end{bmatrix} \tag{3.20}
$$

where $N = N_1 + N_2$. Let $l_i, i = 1, \ldots, N_2$ be the columns in $T_{col\{N\times N_2\}}$ and $r_i, i = 1, \ldots, N_2$ be $1 \times N$ row vectors such that $r_i = [\mathbf{0} \quad r_i']$ where r_i' is the ith row in $T_{row\{N_2\times N_1\}}$. It is easy to verify that

$$
\Delta_Z(T_c^{-1}) = \sum_{i=1}^{N_2} l_i e_i^T + e_i r_i \tag{3.21}
$$

where $e_i, i = 1, \ldots, N_2$ is the ith column of an $N \times N$ identity matrix. Properties of the rank of a matrix show [51]

$$
rank(A+B) \le rank(A) + rank(B),
$$

so

$$\begin{aligned} rank(\Delta_Z(T_c^{-1})) &\leq \sum_{i=1}^{N_2} rank(l_i e_i^T) + rank(e_i r_i) \\ &\leq 2N_2 \\ &= 2(M+k-1) \end{aligned} \tag{3.22}$$

Then we have $\alpha_+(\tilde{T}_c^{-1}) \leq 2(M+k-1)$ since elementary operations do not change the rank of a matrix. If we let

$$Z = \begin{bmatrix} Z_a & Z_b \\ Z_c & Z_d \end{bmatrix} \tag{3.23}$$

and

$$\Delta_Z(\tilde{T}_c^{-1}) = \begin{bmatrix} \Delta_y & \Delta_w \\ \Delta_v & \Delta_u \end{bmatrix} \tag{3.24}$$

then because Z is lower triangular,

$$\begin{aligned} \Delta_Z(\tilde{T}_c^{-1}) &= \begin{bmatrix} \Delta_y & \Delta_w \\ \Delta_v & \Delta_u \end{bmatrix} \\ &= \begin{bmatrix} Y & W \\ V & U \end{bmatrix} - \begin{bmatrix} Z_a & Z_b \\ Z_c & Z_d \end{bmatrix} \begin{bmatrix} Y & W \\ V & U \end{bmatrix} \begin{bmatrix} Z_a^T & Z_c^T \\ Z_b^T & Z_d^T \end{bmatrix}. \end{aligned} \tag{3.25}$$

If we define $N_3 = (k-1)(2M+k-1)$,

$$\begin{aligned} \Delta_{y\{N_3 \times N_3\}} &= Y - Z_a Y Z_a^T - Z_b V Z_a^T - Z_a W Z_b^T - Z_b U Z_b^T \\ &= Y - Z_a Y Z_a^T = \Delta_{Z_a}(Y) \end{aligned} \tag{3.26}$$

since $Z_b = \mathbf{0}$.

Following (3.20) and (3.19), $\Delta_{Z_a}(Y)$ has a structure similar to $\Delta_Z(\tilde{T}_c^{-1})$:

$$\Delta_{Z_a}(Y) = \begin{bmatrix} \Delta_{y0\{N_2 \times N_2\}} & \Delta_{y2\{N_2 \times N_4\}} \\ \Delta_{y1\{N_4 \times N_2\}} & \mathbf{0} \end{bmatrix} \tag{3.27}$$

where $N_4 = N_3 - N_2$. Δ_{y0}, $\Delta_{y1}$and Δ_{y2} are the left and upper part of Y. Similar to (3.22), we can show that $rank(\Delta_{Z_a}(Y)) \leq 2(M + k - 1)$ as long as $N_3 \geq 2(M + k - 1)$, i.e., the dimension of $Y \geq 2(M + k - 1)$.

3.4 Modified System Solution

For a square block-Toeplitz matrix with square blocks, a few fast triangular factorization algorithms exist such as the Bareiss algorithm [52], the Levinson algorithm [39] and the Schur algorithm [53]. These approaches require matrix permutation and inversion, which makes them restricted to certain block structures. The generalized Schur algorithm described in [38] avoids those problems by treating block-Toeplitz matrices in the same way as scalar Toeplitz matrices. Since matrix Y is not a square block-Toeplitz matrix with square blocks, we use a modified version of the generalized Schur algorithm with the defined Z to obtain $Y = LU$ [38]. Then (3.8) can be solved by forward- and back-substitution.

Let (R, S) be a generator of Y. That is,

$$\Delta_{Z_a}(Y) = RS^T \tag{3.28}$$

Then an LU factorization of Y can be obtained using the following procedure [38]:

for $j = 1 : (k-1)(2M+k-1)$**,**

1. $[\text{pvt}, \text{R}, \text{S}] = \text{MakeProper}(\text{R}, \text{S})$, make (R, S) a proper generator to obtain the pvtth column of R r_{pvt} and the pvtth column of S s_{pvt}.

2. Set the jth column of L to r_{pvt} and the jth row of U to s_{pvt}^T.

3. Replace r_{pvt} with $Z_a r_{\text{pvt}}$ and s_{pvt} with $Z_a^T s_{\text{pvt}}$.

A generator is proper with respect to Step j above if all the elements in R are zero on and above row j except for a particular element r_{ji} and all the elements in S are zero on and above row j except for the element s_{ji} in the corresponding location in S. The MakeProper procedure is described below:

1. $r^T =$ first non-zero row of R; $s^T =$ first non-zero row of S.

2. $[\text{pvt}, \text{FIRST}, \text{NEXT}] = \text{FindOrdering}(\text{r}, \text{s})$.

3. For each $k \in FIRST$ then for each $k \in NEXT$ except $k =$ pvt: determine matrix $C_{(k|pvt)}$ and $R = RC_{(k|pvt)}$, $S = SC_{(k|pvt)}^{-T}$.

4. return pvt R S.

The procedure FindOrdering divides the indices $i, i = 1, \ldots, 2(M+k-1)$ for each row into two groups: FIRST and NEXT so that real spinors exist [38]. Since $rank(\Delta_Z(Y) \leq 2(M+k-1)$, the procedure is valid for indices $i = 1, \ldots, 2(M+k-1)$. A complete set of i can also be used; i.e., $i = 1, \ldots, (k-1)(2M+k-1)$. But the smallest possible number should be used for the efficiency of the algorithm. The reader is referred to [38] for details on FindOrdering. The generalized Schur algorithm requires $2(M+k-1)(k-1)^2(2M+k-1)^2$ operations while the forward- and back-substitution requires $O(k^2M^2)$ operations.

Assuming $M \gg k$, the overall computation is $O(k^2M^3)$. Storage of $O(k^2M^2)$ is needed to store the LU decomposition result.

To use this algorithm, an initial generator (R, S) must be available. For the Z defined above, $\Delta_{Z_a}(Y)$ has the form in (3.27).

By inspection, the following (R, S) is an appropriate initial generator.

$$R = \begin{bmatrix} \Delta_{y0\{N_2 \times N_2\}} & I_{N_2 \times N_4} \\ \Delta_{y1\{N_4 \times N_2\}} & \mathbf{0}_{N_4 \times N_4} \end{bmatrix} \tag{3.29}$$

and

$$S^T = \left[\begin{array}{cc} I_{N_2 \times N_2} & \mathbf{0}_{N_2 \times N_4} \\ & - - - - - - \\ & \vert \quad \Delta_{y2\{N_2 \times N_4\}} \\ \mathbf{0}_{\{N_4 \times N_2\}} & \\ & \vert \quad \mathbf{0}_{(N_4 - N_2) \times N_4} \end{array}\right] \tag{3.30}$$

It is straightforward to show that $\Delta_{Z_a}(Y) = RS^T$. To obtain the system solution, we solve $-Wy = LUz$ using forward- and back-substitution and then use a circular deconvolution operation with $[z^T y^T]^T$ as input to obtain $[\mathbf{0}^T x^T]^T$. Notice that $-Wy$ can be obtained with circular deconvolution from

$$\tilde{T}_c^{-1} \begin{bmatrix} \mathbf{0} \\ y \end{bmatrix} = \begin{bmatrix} Y & W \\ V & U \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ y \end{bmatrix} = \begin{bmatrix} Wy \\ Uy \end{bmatrix} \tag{3.31}$$

3.5 Experiments

This fast algorithm applies to any square BTTB system. In this section, we show a small numerical example solving an arbitrary BTTB system and a realistic example in regularized image restoration.

3.5.1 Numerical Example

Once a BTTB system is expanded to a BCCB system, it can be solved with FFT's. However, to acquire the BCCB system, we have to solve for a smaller system using the generalized Schur algorithm described in the previous section. This fast algorithm exploits the bandedness of the BTTB system and turns the problem of solving an $M^2 \times M^2$ system into one that only involves a $(k-1)(2M+k-1) \times (k-1)(2M+k-1)$ matrix.

The mechanism of expanding and representing a BTTB matrix in FFT's is described in the following example. Consider a 16×16 BTTB system

$$T = \begin{bmatrix} t_0 & t_1 & t_2 & \mathbf{0} \\ t_3 & t_0 & t_1 & t_2 \\ t_4 & t_3 & t_0 & t_1 \\ \mathbf{0} & t_4 & t_3 & t_0 \end{bmatrix} \tag{3.32}$$

where

$$t_0 = \begin{bmatrix} 1 & 2 & 3 & 0 \\ 4 & 1 & 2 & 3 \\ 5 & 4 & 1 & 2 \\ 0 & 5 & 4 & 1 \end{bmatrix}, \tag{3.33}$$

$$t_1 = \begin{bmatrix} 6 & 7 & 8 & 0 \\ 9 & 6 & 7 & 8 \\ 10 & 9 & 6 & 7 \\ 0 & 10 & 9 & 6 \end{bmatrix}, \tag{3.34}$$

$$\vdots$$

and

$$t_4 = \begin{bmatrix} 21 & 22 & 23 & 0 \\ 24 & 21 & 22 & 23 \\ 25 & 24 & 21 & 22 \\ 0 & 25 & 24 & 21 \end{bmatrix}. \tag{3.35}$$

In this case, $M = 4$, $k = 3$. T can be expanded to a BCCB system

$$T_c = \begin{bmatrix} c_0 & c_1 & c_2 & \mathbf{0} & c_4 & c_3 \\ c_3 & c_0 & c_1 & c_2 & \mathbf{0} & c_4 \\ c_4 & c_3 & c_0 & c_1 & c_2 & \mathbf{0} \\ \mathbf{0} & c_4 & c_3 & c_0 & c_1 & c_2 \\ c_2 & \mathbf{0} & c_4 & c_3 & c_0 & c_1 \\ c_1 & c_2 & \mathbf{0} & c_4 & c_3 & c_0 \end{bmatrix} \tag{3.36}$$

where each block c_i, $i = 0, 1, \ldots, 4$ is expanded from t_i in the same fashion. For example,

$$c_0 = \begin{bmatrix} 1 & 2 & 3 & 0 & 5 & 4 \\ 4 & 1 & 2 & 3 & 0 & 5 \\ 5 & 4 & 1 & 2 & 3 & 0 \\ 0 & 5 & 4 & 1 & 2 & 3 \\ 3 & 0 & 5 & 4 & 1 & 2 \\ 2 & 3 & 0 & 5 & 4 & 1 \end{bmatrix}. \tag{3.37}$$

Now the problem becomes

$$\begin{bmatrix} z \\ y \end{bmatrix} = PT_cP^T \begin{bmatrix} \mathbf{0} \\ x \end{bmatrix}$$

According to the properties of FFT's,

$$T_c\tilde{x} = \text{column ordered version of } (t_c \otimes \tilde{x}_I) \tag{3.38}$$

where t_c is the convolution kernel corresponding to T_c, in other words, the PSF that T_c stands for. $\tilde{x}$ and $\tilde{x}_I$ are the column-ordered version and the image form of the expanded original, respectively; $\otimes$ stands for circular convolution. By inspection, we have

$$t_c = \begin{bmatrix} 1 & 16 & 21 & 0 & 11 & 6 \\ 4 & 19 & 24 & 0 & 14 & 9 \\ 5 & 20 & 25 & 0 & 15 & 10 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 18 & 23 & 0 & 13 & 8 \\ 2 & 17 & 22 & 0 & 12 & 7 \end{bmatrix} \tag{3.39}$$

Since a BCCB matrix can be diagonalized by 2-D FFT, we obtain the convolution kernel of T_c^{-1} without directly inverting T_c:

$$t_c^{-1} = 10^{-3} \times \begin{bmatrix} -.991 & -6.02 & 4.78 & -3.70 & 10.80 & .002 \\ -.222 & -3.39 & -1.77 & -3.09 & 5.66 & 4.03 \\ .503 & -2.68 & -3.59 & -3.31 & 4.28 & 5.19 \\ -.827 & 17.24 & -4.85 & 15.84 & -30.83 & -8.74 \\ 1.26 & -4.41 & 2.40 & -4.97 & 8.25 & 1.44 \\ .535 & -5.12 & 4.22 & -4.75 & 9.63 & .284 \end{bmatrix} \tag{3.40}$$

Notice t_c^{-1} is not the inverse of t_c considered as a matrix. Following (3.27), we have

$$\Delta_{Z_a}(Y) = \begin{bmatrix} \Delta_{y0} & \Delta_{y2} \\ \Delta_{y1} & \mathbf{0} \end{bmatrix}$$

where

$$\Delta_{y0\{6\times 6\}} = 10^{-3} \times \begin{bmatrix} -.991 & .535 & 1.26 & -.827 & .503 & -.222 \\ -.222 & -.991 & .535 & 1.26 & -.827 & .503 \\ .503 & -.222 & -.991 & .535 & 1.26 & -.827 \\ -.827 & .503 & -.222 & -.991 & .535 & 1.26 \\ 1.26 & -.827 & .503 & -.222 & -.991 & .535 \\ .535 & 1.26 & -.827 & .503 & -.222 & -.991 \end{bmatrix},$$

$$\Delta_{y1\{14\times 6\}} = 10^{-3} \times \begin{bmatrix} -6.02 & -5.12 & -4.41 & 17.24 & -2.68 & -3.39 \\ -3.39 & -6.02 & -5.12 & -4.41 & 17.24 & -2.68 \\ -2.68 & -3.39 & -6.02 & -5.12 & -4.41 & 17.24 \\ 17.24 & -2.68 & -3.39 & -6.02 & -5.12 & -4.41 \\ -4.41 & 17.24 & -2.68 & -3.39 & -6.02 & -5.12 \\ -5.12 & -4.41 & 17.24 & -2.68 & -3.39 & -6.02 \\ 4.78 & 4.22 & 2.40 & -4.85 & -3.59 & -1.77 \\ -1.77 & 4.78 & 4.22 & 2.40 & -4.85 & -3.59 \\ -3.70 & -4.75 & -4.97 & 15.84 & -3.31 & -3.09 \\ -3.09 & -3.70 & -4.75 & -4.97 & 15.84 & -3.31 \\ 10.80 & 5.66 & 4.28 & -30.83 & 8.25 & 9.63 \\ 9.63 & 10.80 & 5.66 & 4.28 & -30.83 & 8.25 \\ .002 & 4.03 & 5.19 & -8.74 & 1.44 & .284 \\ .284 & .002 & 4.03 & 5.19 & -8.74 & 1.44 \end{bmatrix}.$$

and

$$\Delta^T_{y2\{6\times14\}} = 10^{-3} \times \begin{bmatrix} .002 & 4.03 & 5.19 & -8.74 & 1.44 & .284 \\ .284 & .002 & 4.03 & 5.19 & -8.74 & 1.44 \\ 1.44 & .284 & .002 & 4.03 & 5.19 & -8.74 \\ -8.74 & 1.44 & .284 & .002 & 4.03 & 5.19 \\ 5.19 & -8.74 & 1.44 & .284 & .002 & 4.03 \\ 4.03 & 5.19 & -8.74 & 1.44 & .284 & .002 \\ 10.80 & 5.66 & 4.28 & -30.83 & 8.25 & 9.63 \\ 9.63 & 10.80 & 5.66 & 4.28 & -30.83 & 8.25 \\ -3.70 & -3.09 & -3.31 & 15.84 & -4.97 & -4.75 \\ -4.75 & -3.70 & -3.09 & -3.31 & 15.84 & -4.97 \\ 4.78 & -1.77 & -3.59 & -4.85 & 2.40 & 4.22 \\ 4.22 & 4.78 & -1.77 & -3.59 & -4.85 & 2.40 \\ -6.02 & -3.39 & -2.68 & 17.24 & -4.41 & -5.12 \\ -5.12 & -6.02 & -3.39 & -2.68 & 17.24 & -4.41 \end{bmatrix}$$

The size of $\Delta_{Z_a}(Y)$ is $(k-1)(2M+k-1) \times (k-1)(2M+k-1) = 20 \times 20$.

Then the initial generator (R, S) is

$$R_{20\times20} = \begin{bmatrix} \Delta_{y0\{6\times6\}} & I_{6\times14} \\ \Delta_{y1\{14\times6\}} & \mathbf{0}_{14\times14} \end{bmatrix} \tag{3.41}$$

and

$$S^T_{20\times20} = \begin{bmatrix} I_{6\times6} & \mathbf{0}_{6\times14} \\ \mathbf{0}_{14\times6} & \Delta_{y2\{6\times14\}} \\ & \mathbf{0}_{8\times14} \end{bmatrix} \tag{3.42}$$

where

$$I_{6\times14} = \begin{bmatrix} I_{6\times6} & \mathbf{0}_{6\times8} \end{bmatrix}$$

It is straightforward to verify that $\Delta_{Z_a}(Y) = RS^T$. Applying the generalized Schur algorithm, we have the LU decomposition of Y.

$$LUz = -Wy$$

With forward- and back substitution, we can solve for z. Then we plug z in (3.6), and x can be solved with FFT's.

3.5.2 Regularized Image Restoration

A blurred image is often modeled as:

$$y = Hx + n, \tag{3.43}$$

where H is a BTTB system representing the blurring when linear convolution is assumed, i.e., H is taller than wide and the pixels outside x are considered all zeros. x and y are original and blurred images; n is noise. Regularization is applied to reduce noise amplification. Using a simple shift-invariant penalty for the regularization, the solution is expressed as:

$$\hat{x} = (H^T H + \alpha L^T L)^{-1} H^T y, \tag{3.44}$$

where L is a BTTB matrix representing 3×3 Laplacian smoothing, $H^T H$ and $L^T L$ are Hermitian BTTB systems, so $T = H^T H + \alpha L^T L$ is also a Hermitian BTTB system. The regularization parameter α is chosen to be 0.001.

We considered the case of a 256×256 image with finite support blurred by a 5×5 uniform PSF. 40 dB Gaussian noise is added. Thus $M = 256$ and the one-sided bandwidth

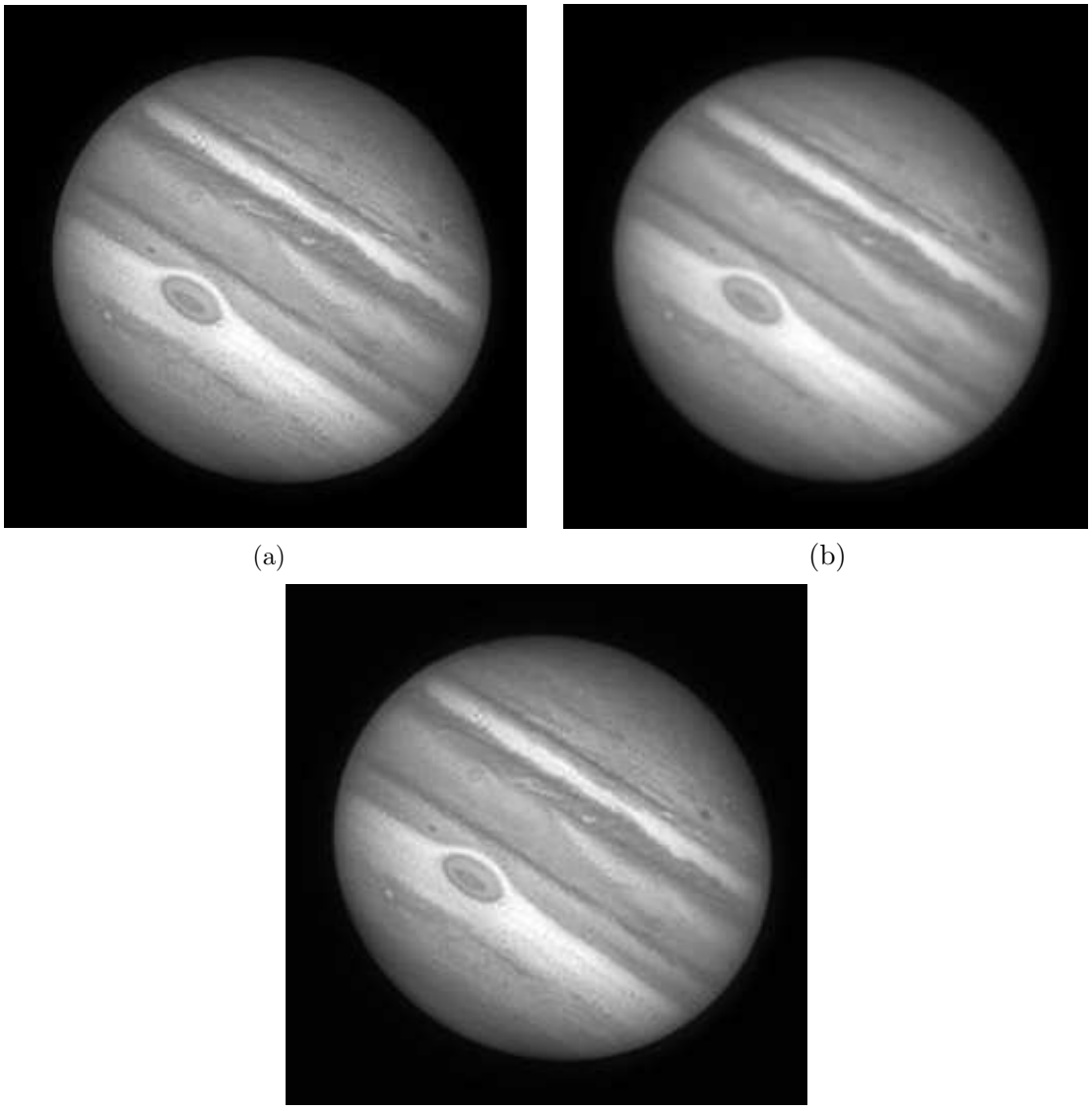
of both the inner and outer structures is $k = 5$. Using the procedure described in the previous section and forward- and back-substitution, we obtain the restored image shown in Fig 3.1. Because we assumed a shift-invariant penalty in regularization, there will be ringing effects along edges in the image. The restored $\hat{x}$ is the exact solution to (3.44).

The computation required to compute the solution with the algorithm described is 151 seconds on a workstation with an AMD Opteron 280 processor. By comparison, a circular deconvolution with approximate solutions requires 0.19 seconds. The solution using a simple banded generalized Schur decomposition is not available due to memory limitation. We applied the algorithm to images with different sizes, and the computation time is shown in Table 3.1. The new fast algorithm requires $O(k^2M^3)$ flops while the general solver needs

Table 3.1: Computation Time

Image size	General solver(s)	Fast algorithm(s)
32×32	1.3	0.35
64×64	36.4	2.8
128×128	1210	22.9
256×256	N/A	151.1

$O(M^5)$. The bigger the image is, the faster the new algorithm is compared to the general solver. Although this example shows great improvement in solving a symmetric BTTB system, it should be noted that our method applies to non-symmetric BTTB systems as well. Applications of BTTB matrices in signal and image processing problems are not limited to regularized image restoration. Other problems such as correlation matrices in autoregressive models also have BTTB structure [54, 55].



(a) (b)

(c)

Figure 3.1: Restoration of Jupiter image
(a) original jupiter image (b) blurred image (c) restored image

3.6 Summary and Discussion

This new algorithm extends a banded $M^2 \times M^2$ BTTB system with bandwidth k to a $(M+k-1)^2 \times (M+k-1)^2$ BCCB system and solves it with circular deconvolution. In order to obtain the BCCB system, a smaller system $-Wy = Yz$ needs to be solved. The generalized Schur algorithm is used to get an LU decomposition for Y, and then forward- and back-substitution are used to solve for z.

The overall algorithm is dominated by the cost to compute the LU decomposition of Y for an overall operation requirement of $O(k^2M^3)$. Thus, both the block-band structure and the banded block structure are exploited in the algorithm count compared to the full structure operation count of $O(M^5)$.

The matrix extension is also proposed in [46], although the augmented part z is estimated with an assumption that the original PSF is bandlimited. Superfast algorithms that requires $O(8M^2 \log^2 M^2)$ flops exploit the block Toeplitz structures but are only efficient for huge systems with $M \geq 256$ [40, 41]. Compared to [46], [40] and [41], the new algorithm is applicable to arbitrary square BTTB systems without any assumption about the original PSF or the size of the system. Notice in the BTTB system, the block size does not need to equal the number of blocks. For a matrix with $M \times M$ blocks of size $p \times p$, the LU decomposition will need $2(p+k-1)(k-1)^2(M+p+k-1)^2$ operations. Compared to $2(M+k-1)(k-1)^2(2M+k-1)^2$ in Section 4, it can be faster if $p \leq M$.

This algorithm works best when k is small, i.e., the Toeplitz structure has a small bandwidth. As $k \to M$, the efficiency approaches the non-banded algorithm efficiency of $O(M^5)$.

Chapter 4

Deblocking of JPEG-compressed Images

4.1 JPEG Compression

4.1.1 Introduction

JPEG is a commonly used standard method of compression for photographic images [56]. The name JPEG stands for Joint Photographic Experts Group, the name of the committee which created the standard. The group was organized in 1986, issuing a standard in 1992 which was approved in 1994 as ISO 10918-1. JPEG provides for lossy compression of images (although there are variations on the standard baseline JPEG that are lossless). The file format that employs this compression is commonly also called JPEG; the most common file extension for this format is .jpg, though .jpeg, .jpe, .jfif and .jif are also used [57].

JPEG compression has certain characteristics that distinguish it from other compression mechanisms:

- JPEG is designed for compressing either full-color or gray-scale images of natural, real-world scenes. It works well on photographs, naturalistic artwork, and similar material but not so well on lettering, simple cartoons, or line drawings, unlike Graphics Interchange Format (GIF).

- JPEG is "lossy", meaning that the decompressed image is not quite the same as the original. (There are lossless image compression algorithms such as GIF, but JPEG achieves much greater compression than is possible with lossless methods.) JPEG

is designed to exploit known limitations of the human eye, notably the fact that small color changes are perceived less accurately than small changes in brightness. Thus, JPEG is intended for compressing images that will be looked at by humans. If analyzed by machine, the small errors introduced by JPEG may seem a big problem, even if they are invisible to the eye.

- A useful property of JPEG is that the degree of lossiness can be varied by adjusting a compression parameter called the quality factor. This means that the image maker can trade off file size against output image quality. You can make extremely small files if you don't mind poor quality; this is useful for applications such as indexing image archives. Conversely, if you are not happy with the output quality at the default compression setting, you can improve the quality until you are satisfied, and accept lesser compression.

- Another important aspect of JPEG is that decoders can trade off decoding speed against image quality, by using fast but inaccurate approximations to the required calculations. Some viewers obtain remarkable speedups in this way. (Encoders can also trade accuracy for speed, but there is usually less reason to make such a sacrifice when writing a file.)

Although JPEG is a lossy compression, it actually loses far less information than other compression methods such as GIF in the case of a real-world scene. As long as the image is not repeatedly compressed and decompressed, there is very little difference visible to human eyes, given a reasonable quality factor [58]. As mentioned above, a low quality factor can result in poor image quality (Fig 4.1).

Now there is an improved JPEG standard, JPEG2000, that uses state-of-the-art compression techniques based on wavelet technologies. Its architecture should lend itself to a wide range of uses from portable digital cameras through to advanced pre-press, medical imaging and other key sectors. JPEG2000 was created by the Joint Photographic Experts Group committee with the intention of superseding their original discrete cosine transform-based JPEG standard. Common filename extensions include .jp2 and .j2c [57, 59].

JPEG 2000 can operate at higher compression ratios without generating the characteristic blocky and blurry artifacts of the original DCT-based JPEG standard. It also allows more sophisticated progressive downloads. Part of JPEG2000 has been published as an ISO standard, ISO/IEC 15444-1:2000. As of 2006, JPEG2000 is not widely supported in web browsers, and hence is not generally used on the World Wide Web [57].

4.1.2 JPEG Algorithm

The JPEG algorithm performs its compression in four steps (Fig 4.2):

1. The JPEG algorithms first cuts up an image in separate blocks of 8x8 pixels. Since the format is based on luminance/chrominance perception, it does not analyze RGB or CMYK color values but instead converts image data to a luminance/chrominance color space, such as YUV. This allows for separate compression of these two factors. Since luminance is more important than chrominance for our visual system, the algorithm retains more of the luminance in the compressed file.

2. The next step in the compression process is to apply a Discrete Cosine Transform (DCT) for the entire block. DCT replaces actual color data for each pixel for values that are relative to the average of the entire matrix that is being analyzed. This

operation does not compress the file, it simply replaces 8x8 pixel values by an 8x8 matrix of DCT coefficients.

3. Once the data is in DCT domain, the actual compression can start. First, the compression software looks at the JPEG image quality the user requested and calculates two tables of quantization constants, one for luminance and one for chrominance. Once these tables have been constructed, the constants from the two tables are used to quantize the DCT coefficients. Each DCT coefficient is then divided by its corresponding constant in the quantization table and rounded off to the nearest integer. The result of quantizing the DCT coefficients is that smaller, unimportant coefficients will be replaced by zeros and larger coefficients will lose precision. It is this rounding-off that causes loss in image quality.

4. The resulting data are a list of streamlined DCT coefficients. The last step in the process is to compress these coefficients using either a Huffman or arithmetic encoding scheme. Usually Huffman encoding is used. This is a second (lossless) compression that is applied. Now the compressed data includes the Huffman code for the DCT coefficients, the quantization table and the Huffman table.

The decompression works as the inverse procedure of the compression. First, the Huffman code is decoded according to the Huffman table. Then the same quantization table is used to dequantize the DCT coefficients. Notice at this stage, some information is already lost in the rounding-off during compression. The so-called "dequantization" is simply a rough restoration of the DCT coefficients to the original values. An inverse block DCT is then applied to the coefficients to obtain the image data (Fig 4.3).

Among the steps of generating a JPEG-compressed image, the DCT is where the energy is packed into low-frequency coefficients; quantization is where part of the information is lost and the "compression" happens; Huffman coding works efficiently in this case only if quantization gets rid of some "unimportant" coefficients. In the sections that follow, we will explain how the DCT and quantization work. Readers interested in entropy coding are referred to [59, 60] for details.

4.2 Discrete Cosine Transform

4.2.1 1-D DCT

The rapid growth of digital imaging applications, including desktop publishing, multimedia, teleconferencing, and high-definition television (HDTV) has increased the need for effective and standardized image compression techniques. Among the emerging standards are JPEG, for compression of still images [61]; MPEG, for compression of motion video; and ITU H.26x for compression of video telephony and teleconferencing. All three of these standards employ a basic technique known as the discrete cosine transform (DCT) [62].

The discrete cosine transform of a list of real numbers $c(n)$, $n = 0, ..., N-1$, is the list of length N given by:

$$C(k) = \alpha(k) \sum_{n=0}^{N-1} c(n) \cos \frac{\pi(2n+1)k}{2N}, \quad 0 \leq k \leq N-1 \tag{4.1}$$

where

$$\alpha(0) = \sqrt{\frac{1}{N}}, \quad \alpha(k) = \sqrt{\frac{2}{N}} \text{ for } \ 1 \leq k \leq N-1.$$

Each element of the transformed list $C(k)$ is the inner product of the input list $c(n)$ and a basis vector. The factors $\alpha(k)$ are chosen so that the basis vectors are orthogonal and normalized. The eight basis vectors for $n = 8$ are shown in Fig 4.4.

The inverse transformation is given by

$$c(n) = \sum_{k=0}^{N-1} \alpha(k)C(k) \cos \frac{\pi(2n+1)k}{2N}, \quad 0 \leq n \leq N-1 \tag{4.2}$$

where

$$\alpha(0) = \sqrt{\frac{1}{N}}, \quad \alpha(k) = \sqrt{\frac{2}{N}} \text{ for } \ 1 \leq k \leq N-1.$$

The DCT is real and orthogonal, that is, for DCT matrix $\mathcal{C}$ in $C(k) = \mathcal{C}c(n)$, $\mathcal{C}^{-1} = \mathcal{C}^T$. Notice that the DCT is closely related to the discrete Fourier transform (DFT) although not the real part of the unitary DFT. In fact, it is possible to compute the DCT via the DFT by symmetrically extending the original sequence [1]. The DCT is often used in signal and image processing, especially for lossy data compression, because it has a strong energy compaction property: most of the signal information tends to be concentrated in a few low-frequency components of the DCT, approaching the Karhunen-Loeve transform (which is optimal in the decorrelation sense) for signals based on certain limits of Markov processes [63].

There are eight standard DCT variants, of which four are common. The definition given above is also called the type-II DCT, which is the one used in image compression. The 1-D DCT is useful in processing one-dimensional signals such as speech waveforms. For analysis of 2-D signals such as images, we need a 2-D version of the DCT, which is explained in detail in the next subsection.

4.2.2 2-D DCT

In JPEG compression, the 2-D block DCT is used for the transformation. The original image is divided into 8×8 blocks and a DCT is applied to each block. The 2-D DCT is defined as:

$$C(k,l) = \alpha(k)\alpha(l) \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} c(m,n) \cos\left[\frac{(2m+1)k\pi}{2N}\right] \cos\left[\frac{(2n+1)l\pi}{2N}\right] \tag{4.3}$$

where

$$\alpha(0) = \sqrt{\frac{1}{N}}, \quad \alpha(k) = \sqrt{\frac{2}{N}} \text{ for } \ 1 \leq k \leq N-1.$$

The discrete cosine basis function for $N = 8$ is shown in Fig 4.5.

For an $m \times n$ matrix, the 2-D DCT is computed in a simple way: The 1-D DCT is applied to each row and then to each column of the result. Since the 2-D DCT can be computed by applying 1-D transforms separately to the rows and columns, the 2-D DCT is separable in the two dimensions, like the 2-D DFT [62].

To compute a block DCT (BDCT), we do not actually have to divide the image into blocks. Since the 2-D DCT is separable, we can partition each row into sequences of length 8, apply the DCT to them, rejoin the resulting lists, and then transpose the whole image and repeat the process. Separability is in fact one of the numerous advantages of the DCT over the Karhunen-Loeve transformation (KLT), since this makes the implementation of the DCT highly efficient.

4.3 Quantization

Quantization is the procedure where the DCT coefficients are divided by a specially designed table of numbers and then rounded so that some of the coefficients turn out to be zeros. The entropy coding following quantization can then make use of the redundancy of information [60].

There are various ways to design a quantization table. Correspondingly, different sorts of quantization algorithms exist. Among these quantization algorithms, scalar quantization quantizes DCT samples individually while vector quantization joins all samples. A simple example of scalar quantization is a function that maps each element in a subset of the real line to a particular value in that subset (Fig 4.6). Other quantization algorithms include Lloyd-Max scalar quantizer [64, 65], generalized Lloyd algorithm [66], Trellis coded quantization [67], etc.

In our discussion about quantization in this work, we assume that the simple scalar quantization (Fig 4.6) is used to simplify the presentation. For the quantization table, the standard table for luminance (4.4) is used in experiments.

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 51 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \tag{4.4}$$

Notice in (4.4), the quantization coefficient increases roughly from the upper left corner to the lower right corner. It can be shown [68] that the DCT approximately diagonalizes the covariance matrix of a first-order Gauss-Markov random process. In fact, one of the properties of any wide-sense stationary (WSS) random process is that the Fourier coefficients are uncorrelated. That is, as N in (4.3) becomes very large, the DFT, the DCT and other related frequency transforms all diagonalize the source covariance matrix [69]. Thus, these transforms are asymptotically equivalent to the KLT for WSS sources, up to a reordering of the coefficients [59].

One suitable data-independent ordering for the DCT coefficients is the "zig-zag" scan shown in Fig 4.7. This order is based on the observation that the power density spectra of most images tends to decrease rapidly with increasing spatial frequency; it is employed by the JPEG image compression standard and most video compression standards. Although most images are not well modeled as WSS random processes, the DCT has been found to be a robust approximation to the KLT for natural image sources [59].

4.4 Deblocking JPEG Images

4.4.1 Blocking Effect

It has been shown that the DCT is an asymptotic approximation to the optimal Karhunen-Loeve transform when the statistical properties of the image can be described by a first-order Markov model. Furthermore, it has been demonstrated that most images encountered in visual communication applications are modeled extremely well by first-order Markov models [1]. Because the DCT can be computed very efficiently using fast algorithms similar in nature to the well-known fast Fourier transform (FFT), it has been recommended by both the Joint Photography Experts Group (JPEG) and the Motion Pictures Experts Group (MPEG) for compression of still and sequences of motion images, respectively [70].

Although JPEG-compressed images are good approximations to the uncompressed images, most JPEG images have the problem of visible block boundaries due to coarse quantization of the coefficients [71]. This problem becomes especially serious when the DC or near-DC coefficients are coarsely quantized. Therefore, more bits are usually required for the DC or near DC coefficients than for other coefficients. An example of the blocking effect in the Lena image is shown in Fig 4.8.

To cope with the blockiness problem without increasing the bit rate, a variety of post-processing approaches have been proposed:

1. Low-pass filtering on boundary pixels [72, 73, 74, 75].

2. Postprocessing algorithms based on estimation theory, for example Stevenson [76] proposed a postprocessing algorithm based on the maximum a posteriori probability

approach. Yang et al. [77] proposed a postprocessing algorithm based on noise estimation. The algorithm estimates the quantization noise causing blocking artifacts, and subtracts the estimated noise from the quantized images. There are other approaches based on the wavelet representation. Xiong et al. [78] proposed a wavelet-based postprocessing algorithm, which employs an edge classifier using the overcomplete wavelet representations. Choi and Kim [79] proposed a postprocessing algorithm in the subband domain. They employed a minimum mean square error filter to remove the blocking artifacts located at the block boundaries of subbands. Nosratinia [80] proposed a postprocessing algorithm based on the reapplication of JPEG compression. The postprocessed image is obtained by averaging the images, which are obtained from the reapplication of the JPEG compression with various shifts. This algorithm is known to provide an excellent result among the current postprocessing algorithms. However, the computational complexity is high due to multiple reapplication of the JPEG compression.

3. Iterative algorithms, which are based on projection onto convex sets (POCS) or the constrained least squares, have also been proposed. Youla and Webb [81] introduced the theory of POCS to restore images. To postprocess the DCT-based encoded images, Zakhor [82] employed the quantization constraint set (QCS) as a convex constraint set and the lowpass filter that is used by Reeve and Lim [72]. The algorithm consists of the iteration of lowpass filtering and projecting onto QCS. As pointed out by Reeves and Eddins [83], the iterative algorithm is based on the theory of constrained least squares. Based on the POCS theory, Yang et al. [84] introduced a projection operator onto the smoothness constraint set, which smooths the blocking artifacts instead of

using the linear lowpass filter. Park and Kim [85], on the other hand, proposed a narrow QCS (NQCS) to improve the performance of the QCS-based postprocessing algorithms. They also extended the notion of NQCS to the vector-quantized signals by introducing hypercubes. At high bit rates, employing NQCS in a POCS-based post-processing algorithm can preserve details and edges from undesirable blurring without any sophisticated, edge-oriented classifiers and empirically determined thresholds [86].

4. Non-iterative algorithms have several advantages over iterative algorithms. In order to secure the convergence of an iterative algorithm, the operators should meet several requirements. For example, the operators should be projectors if the iteration is based on the POCS theory. In non-iterative algorithms, on the other hand, the operators can be more flexible. Liew and Yan [87] proposed a simple, non-iterative, wavelet-based deblocking algorithm. The algorithm exploits the fact that block discontinuities are constrained by the DC quantization interval of the quantization table, as well as the behavior of the wavelet modulus maxima evolution across scales for singular image structures, to derive appropriate threshold maps at different wavelet scales.

Postprocessing algorithms do not modify the compression system itself and can thus be directly applicable to commercial products, such as the JPEG and MPEG compression systems [86]. Since the blocking effect is primarily due to the inability of the DCT to exploit inter-block correlations, most of the above approaches exploit correlations of intensity values in neighboring blocks.

4.4.2 Deblocking with Regularization

Mathematical Model

In this section, we propose a new algorithm that is involved in the decompression process. This method makes use of the regularized restoration describe in Chapter 2 and is closely related to the lowpass filtering in [72, 73, 74, 75] but differs in that it is not a postprocessing algorithm.

In a regularized restoration, the objective is to minimize the cost function

$$\phi(\hat{f}) = \|g - H\hat{f}\|^2 + \alpha\|L\hat{f}\|^2 \tag{4.5}$$

where L is a high-pass filter that imposes a smoothness condition on the restored image, α is the Lagrangian multiplier. If we model the JPEG compression as a degradation of the image, we should be able to describe the degradation process as

$$y = HQDx \tag{4.6}$$

where y is the compressed data, H, Q and D are the Huffman coding, quantization and BDCT, respectively. To simplify the problem, we can remove the Huffman coding since it is a lossless procedure.

$$y = QDx \tag{4.7}$$

where y is the data before entropy coding, Q and D are the quantization and BDCT. Furthermore, the distortion introduced by coarse quantization is not linear in reality and can be modeled as noise. Normally, if uniform quantization is assumed, the noise is correlated

with the data. However, if the bit-rate allocated is high enough we may assume that the noise and input are uncorrelated with each other [88]. This assumption does not hold for low bit-rates but it is commonly used for the analysis of the quantization error as the problem becomes extremely complicated otherwise and we often find that the results carry on to the low bit-rate case as well [89]. Using this assumption we simplify the degradation process into

$$y = Dx + n \tag{4.8}$$

Now we have the mathematical equivalent of a standard deblurring problem where we are given the blurred image y and the blurring matrix D. The objective is to find the original image x. We assume that the noise n is uncorrelated with the signal thus can be added to the model. As described in (4.5), the estimate $\hat{x}$ should be the minimizer of the function

$$\phi(\hat{x}) = \|y - D\hat{x}\|^2 + \alpha\|L\hat{x}\|^2. \tag{4.9}$$

For this specific case of decompressing a JPEG image, L would be a high-pass filter not only affects natural edges in the image, but also block boundaries for each 8×8 block. Since the block boundaries and the natural edges are not correlated, we can separate the penalty term into two lowpass filters L_D for natural edges and L_e for block boundaries. Now the cost function becomes:

$$\phi(\hat{x}) = \|y - D\hat{x}\|^2 + \alpha\|L_D\hat{x}\|^2 + \beta\|L_e\hat{x}\|^2 \tag{4.10}$$

where β is another Lagrangian multiplier as the smoothing control over the block boundaries. Taking derivative of $\phi(\hat{x})$ with respect to x and setting it to 0, we have

$$\hat{x} = (D^t D + \alpha L_D^t L_D + \beta L_e^t L_e)^{-1} D^t y \tag{4.11}$$

Following the technique in Chapter 2, we can apply the Sherman-Morrison matrix inversion lemma and then divide the solution into two parts: one with the diagonalizable matrix so that we can easily invert it in the transform domain; the other that has to be inverted directly but easily since it is small. Notice that L_e works on all the block boundaries inside the image (dash lines in Fig 4.9). To avoid the overcomplicated L_e matrix, we divide it up into two matrix that deal with vertical and horizontal block boundaries separately (dash lines in Fig 4.10 and Fig 4.11).

Now L_e becomes L_{ve} and L_{he}, where L_{ve} is the smoothing matrix on vertical boundaries and L_{he} is the smoothing matrix on horizontal boundaries. Considering the fact that L_{ve} has little effect on horizontal boundaries and L_{he} has little effect on vertical boundaries, the correlation between them can be ignored. The cost function becomes

$$\phi(\hat{x}) = \|y - D\hat{x}\|_S^2 + \alpha\|L_D\hat{x}\|^2 + \beta\|L_{ve}\hat{x}\|^2 + \beta\|L_{he}\hat{x}\|^2 \tag{4.12}$$

where S is a diagonal weighting matrix. Taking the derivative with respect to x and setting it to 0, we have

$$\hat{x} = (D^t S D + \alpha D^t \mathcal{L}_D D + \beta(L_{ve}^t L_{ve} + L_{he}^t L_{he}))^{-1} D^t S y \tag{4.13}$$

where $\mathcal{L}_D$ is the diagonal matrix representing smoothing operation in the cosine domain.

$$L_D = D^t \mathcal{L}_D D, \tag{4.14}$$

so

$$L_D^t L_D = D^t \mathcal{L}_D D. \tag{4.15}$$

To apply the Sherman-Morrison matrix inversion lemma, we define

$$A = (S + \alpha \mathcal{L}_D). \tag{4.16}$$

Plugging (4.16) in (4.13), we have

$$\hat{x} = (D^t A D + \beta (L_{ve}^t L_{ve} + L_{he}^t L_{he}))^{-1} D^t S y \tag{4.17}$$

So far we have been trying to separate the vertical and horizontal smoothing L_{ve} and L_{he}, but in (4.17) they are still together. To simplify the computation, we consider the following equation: since DCT is orthogonal, i.e.,

$$D^t D = D D^t = I, \tag{4.18}$$

$$\begin{aligned} & (D^t A^{1/2} D + \beta L_{ve}^t L_{ve} D^t A^{-1/2} D)(D^t A^{1/2} D + \beta D^t A^{-1/2} D L_{he}^t L_{he}) \qquad (4.19) \\ = \quad & D^t A D + \beta (L_{ve}^t L_{ve} + L_{he}^t L_{he}) + \beta^2 L_{ve}^t L_{ve} D^t A^{-1} D L_{he}^t L_{he} \end{aligned}$$

Considering the fact that β^2 is very small and L_{ve} and L_{he} are almost uncorrelated, the last term $\beta^2 L_{ve}^t L_{ve} D^t A^{-1} D L_{he}^t L_{he}$ is relatively insignificant. Then (4.19) can be rewritten as

$$\begin{aligned} & (D^t A^{1/2} D + \beta L_{ve}^t L_{ve} D^t A^{-1/2} D)(D^t A^{1/2} D + \beta D^t A^{-1/2} D L_{he}^t L_{he}) \qquad (4.20) \\ \approx \quad & D^t A D + \beta (L_{ve}^t L_{ve} + L_{he}^t L_{he}). \end{aligned}$$

Then (4.17) can be rewrite as

$$\begin{aligned} & \hat{x} \qquad (4.21) \\ = \quad & ((D^t A^{1/2} D + \beta L_{ve}^t L_{ve} D^t A^{-1/2} D)(D^t A^{1/2} D + \beta D^t A^{-1/2} D L_{he}^t L_{he}))^{-1} D^t S y \\ = \quad & (D^t A^{1/2} D + \beta D^t A^{-1/2} D L_{he}^t L_{he})^{-1} (D^t A^{1/2} D + \beta L_{ve}^t L_{ve} D^t A^{-1/2} D)^{-1} D^t S y \end{aligned}$$

In (4.21), L_{ve} and L_{he} can be implemented separately. Applying the Sherman-Morrison matrix inversion lemma to the first inversion $(D^t A^{1/2} D + \beta D^t A^{-1/2} D L_{he}^t L_{he})^{-1}$ in (4.21),

$$\begin{aligned} & (D^t A^{1/2} D + \beta D^t A^{-1/2} D L_{he}^t L_{he})^{-1} \qquad (4.22) \\ = D^t A^{-1/2} D - \beta D^t A^{-1/2} D D^t A^{-1/2} D L_{he}^t & (I + \beta L_{he} D^t A^{-1/2} D D^t A^{-1/2} D L_{he}^t)^{-1} L_{he} D^t A^{-1/2} D \\ = D^t A^{-1/2} D - \beta D^t A^{-1} D L_{he}^t & (I + \beta L_{he} D^t A^{-1} D L_{he}^t)^{-1} L_{he} D^t A^{-1/2} D. \end{aligned}$$

Since $I + \beta L_{he} D^t A^{-1} D L_{he}^t$ is a smaller matrix than A, the inversion is relatively fast. Similarly, we have

$$\begin{aligned} & (D^t A^{1/2} D + \beta L_{ve}^t L_{ve} D^t A^{-1/2} D)^{-1} \qquad (4.23) \\ = D^t A^{-1/2} D - \beta D^t A^{-1/2} D L_{ve}^t & (I + \beta L_{ve} D^t A^{-1/2} D D^t A^{-1/2} D L_{ve}^t)^{-1} L_{ve} D^t A^{-1/2} D D^t A^{-1/2} D \end{aligned}$$

$$= D^t A^{-1/2} D - \beta D^t A^{-1/2} D L_{ve}^t (I + \beta L_{ve} D^t A^{-1} D L_{ve}^t)^{-1} L_{ve} D^t A^{-1} D.$$

The techniques of computing the above expressions are described in Chapter 2.

Experiment

In this section, experiments were conducted using 64-by-64 and 128-by-128 sub-images from the Lena image. The 64-by-64 example is shown in Figure 4.12. All computation was done on a workstation with an AMD Opteron 280 processor. The computation time for a 64-by-64 image is 26 seconds. By applying the block Gauss-Seidel to the matrices, it can be implemented in 18 seconds. The mean square errors of the compressed JPEG images and the deblocked images are shown in Table 4.1.

Table 4.1: Deblocking JPEG image

size	Time (s)	MSE of original	MSE of deblocked
64	18	62	58
128	119	231	201

Notice that in Table 4.1, the difference in MSE between the original JPEG image and the deblocked image is not as considerable as that in a regular deblurring problem. One of the reasons is that deblocking blurs along block boundaries, thus degrading the image to some extent. By paying the price of a little blurrier image, the visual quality has been improved since the blocking effect is reduced. The trade-off between blurry image and blocky image can be controlled by choosing different Lagrangian multiplier β.

4.5 Summary

In this chapter, we developed a new iterative deblocking algorithm for JPEG-compressed images that is designed as part of the decompression process. It makes use of the shift-variant regularization method to blur between boundaries thus get rid of blocking effect. Compared to other deblocking algorithms based on blurring [72, 73, 74, 75], it is not a postprocessing method.

By applying the Sherman-Morrison matrix inversion lemma and block Gauss-Seidel, this algorithm is promising in computation time. Although it is an iterative method, its convergence can be proved in the same fashion as in Chapter 2. The result also shows a smoother image with less blocking effects.

(a)Original Lena

(b)JPEG-compressed Lena with low quality factor (Q=25)

Figure 4.1: JPEG-compressed image

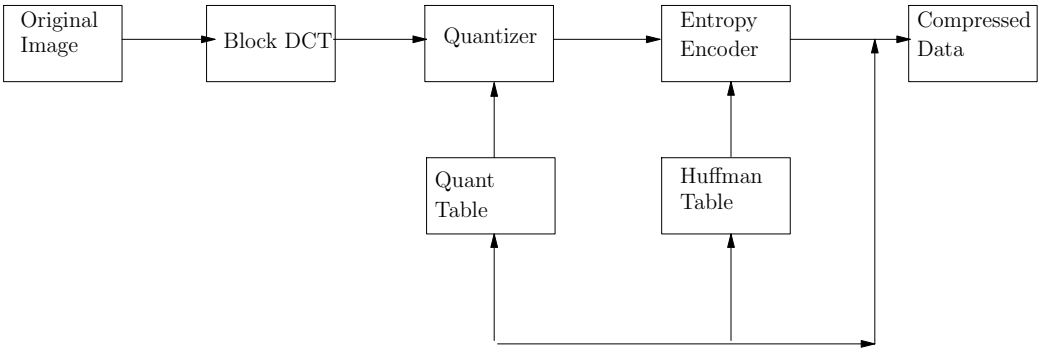


Figure 4.2: JPEG encoder

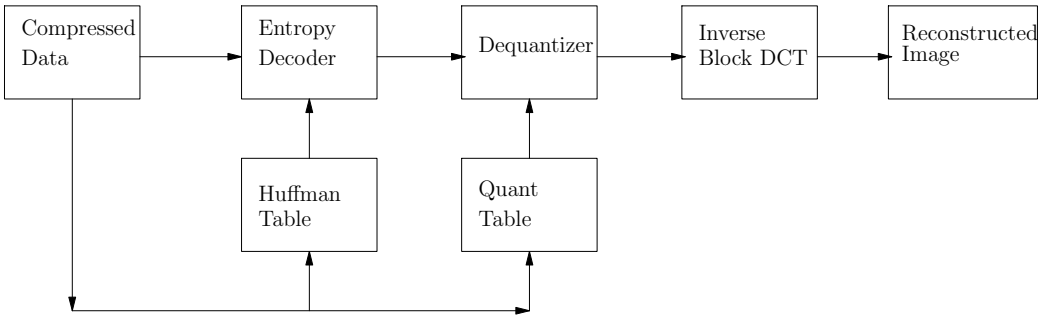


Figure 4.3: JPEG decoder

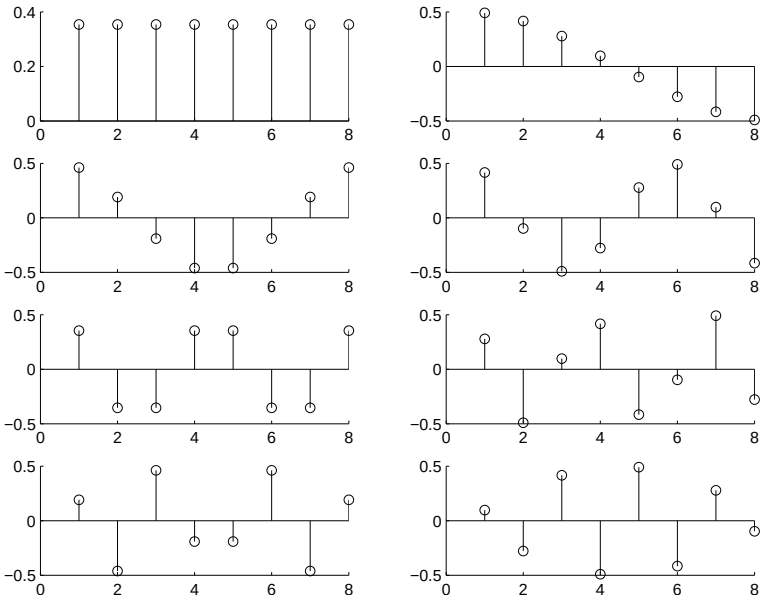


Figure 4.4: The eight basis vectors for the discrete cosine transform of length eight

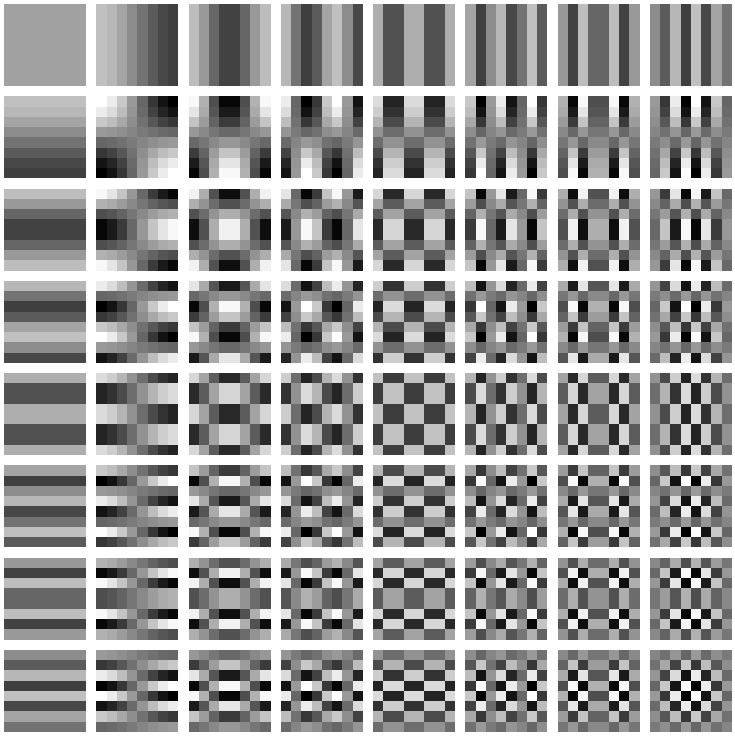


Figure 4.5: Discrete cosine basis functions for $N = 8$

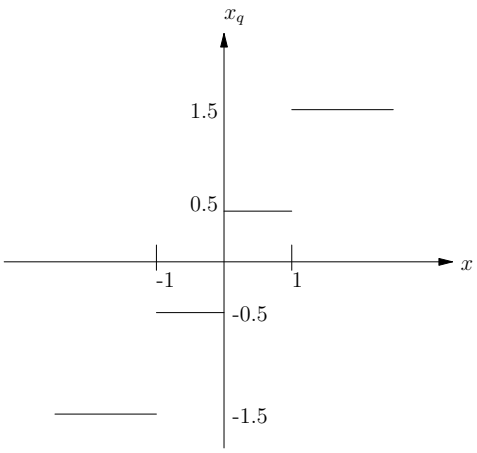


Figure 4.6: Scalar quantization

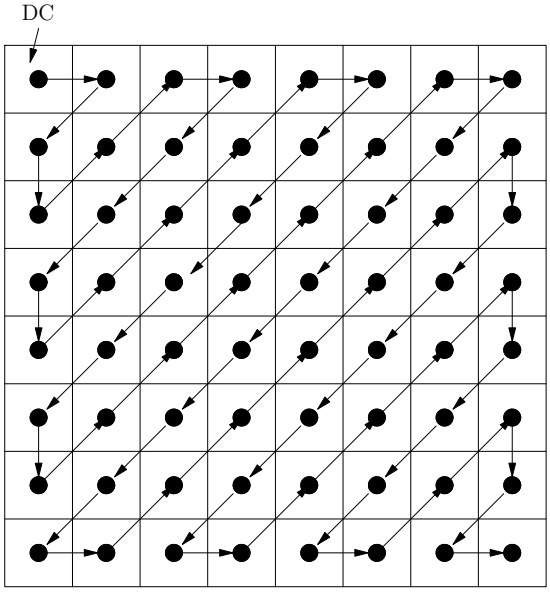


Figure 4.7: Zigzag scan

(a) Original Lena image

(b) Blocking effect in Lena image

Figure 4.8: Blocking effect

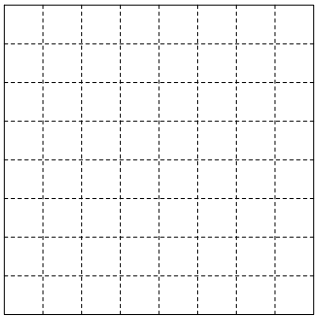


Figure 4.9: Block boundaries

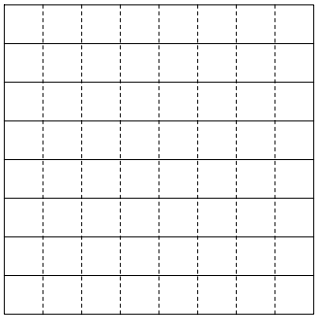


Figure 4.10: Vertical boundaries

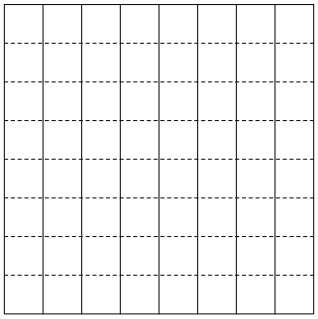


Figure 4.11: Horizontal boundaries

(a) JPEG-compressed Lena image

(b) Lena's hat with blocking effect

(c) Lena's hat with reduced blocking effect

Figure 4.12: Deblocking Lena image

Chapter 5

Conclusion and Discussion

Image restoration is a computationally intensive image processing task. Many image processing situations involve solving inverse problems for either block-Toeplitz-with-Toeplitz-block (BTTB) or block-circulant-with-circulant-block (BCCB) systems. In this work, efficient schemes for solving BCCB system using FFTs are proposed. Since a BTTB system can be extended to BCCB, it is a near-circulant system and can be solved using a similar technique. Efficient algorithms for solving BCCB and BTTB systems are proposed in this work to reduce ringing artifacts in the restored image and reduce computation time. The same techniques are also applied to JPEG images to reduce the blocking effect.

5.1 Edge-Preserving Regularization

In Chapter 2, we proposed a method DEEPIR. MAP estimation, a Gauss-Markov random field, and the Huber function are used as the optimization criterion in this context. Then the optimization criterion is modified with a majorization technique to achieve a more efficient algorithm. Finally, the majorized criterion is minimized by a fast implementation where the restoration is decomposed into a sum of two independent restorations: one yields an image that comes directly from an FFT restoration; the other involves a set of unknowns whose number equals the number of weights in the matrix that deviate from 1 (weights associated with edges). By summing the two, the ringing artifacts from FFTs are canceled. Because the second restoration has a significantly reduced set of unknowns, it can be calculated very efficiently even though no circular convolution structure exists.

Since we do not have the original image to derive edge information, an iterative method is used. Although fixed point iterations often do not converge, this algorithm with the Huber function converges monotonically to the minimizer. DEEPIR compares favorably to conjugate gradients in computational time. It can be improved considerably by dividing the system into blocks using a modified block Gauss-Seidel procedure. A block-based approximation technique is also proposed to yield an approximation of the direct solution at a much reduced computational cost.

5.2 Fast Algorithm for Solving Block Banded Toeplitz Systems with Banded Toeplitz Blocks

In Chapter 3, we developed a new algorithm that extends a banded $M^2 \times M^2$ BTTB system with bandwidth k to a $(M + k - 1)^2 \times (M + k - 1)^2$ BCCB system and solves it with circular deconvolution. In order to obtain the BCCB system, a smaller system $-Wy = Yz$ needs to be solved. The generalized Schur algorithm is used to get an LU decomposition for Y , and then forward- and back-substitution are used to solve for z. The overall algorithm is dominated by the cost to compute the LU decomposition of Y for an overall operation requirement of $O(k^2M^3)$. Thus, both the block-band structure and the banded block structure are exploited in the algorithm count compared to the full structure operation count of $O(M^5)$. The new algorithm is applicable to arbitrary square BTTB systems without any assumption about the original PSF or the size of the system. Notice in the BTTB system, the block size does not need to equal the number of blocks. This algorithm works best when k is small, i.e., the Toeplitz structure has a small bandwidth. As k approaches M, the efficiency approaches the non-banded algorithm efficiency of $O(M^5)$.

5.3 Deblocking of JPEG-compressed Images

In Chapter 4, we applied a scheme similar to that in Chapter 2 for deblocking JPEG-compressed images. The method applies the Sherman-Morrison matrix inversion lemma and block the Gauss-Seidel method to reduce computation. It is different from postprocessing algorithms since it is involved in the decompression process but similar to some postprocessing methods that are based on blurring.

The deblocked image looks smoother and the computation time is promising.

5.4 Future Work

1. The efficient algorithm DEEPIR in Chapter 2 leads to several issues and opportunities. First, we have flexibility in defining the threshold for the regularization weights by adjusting the value of T. The fewer edge weights we allow, the faster the restoration will be. A small T will give a sharp restoration while it can be noisy if any variation in gray level is mistaken as an edge; a large T reduces the number of edges and thus reduces computation but results in blurry edges. Second, the block-based method can be used as a preconditioner in CG since it is considered an approximation to the original edge-preserving regularization. Finally, we have shown that a similar approach can address the fact that real-world blurring does not conform to circular convolution but rather linear convolution followed by windowing. This approach can easily be incorporated into the algorithm described above. Thus, this approach provides an efficient method for shift-variant regularization.

2. The deblocking method in Chapter 4 reduces the blocking effect by blurring between blocks. Since we made an approximation in (4.20), the deblocked image might not

be the exact reconstruction from the compressed data, which introduces some error. The control over the smoothness (or blockiness) is in the Lagrangian multiplier β. The computation time can be further reduced with a block implementation describe in Chapter 2.

Bibliography

[1] A. K. Jain, Fundamentals of Digital Image Processing. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[2] H. C. Andrews and B. R. Hunt, Digital Image Restoration. New Jersey: Prentice Hall, 1977.

[3] R. C. Gonzalez and R. E. Woods, Digital Image Processing. New Jersey: Prentice Hall, 2002.

[4] A. K. Katsaggelos, Digital Image Restoration. Tiergartenstrasse 17, W-6900 Heidelberg, Germany: Springer-Verlag, 1991.

[5] W. K. Pratt, Digital Image Processing. New York, NY: John Wiley & Sons, 2001.

[6] C. W. Hellstrom, "Image restoration by the method of lease square," J. Opt. Soc. Amer., vol. 57, pp. 297–303, 1967.

[7] R. L. Lagendijk, J. Biemond, and D. E. Boekee, "Regularized iterative image restoration with ringing reduction," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 36, pp. 1874–1888, Dec. 1988.

[8] J. W. Woods, J. Biemond, and A. M. Tekalp, "Boundary value problem in image restoration," in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 692–695, 1985.

[9] S. J. Reeves, "Fast image restoration without boundary artifacts," IEEE Transactions on Image Processing, to appear.

[10] J. A. Fessler, H. Erdŏgan, and W. Wu, "Exact distribution of edge-preserving map estimators for linear signal models with gaussian measurement noise," IEEE Transactions on Image Processing, vol. 9, pp. 1049–1055, June 2000.

[11] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud, "Deterministic edge-preserving regularization in computed imaging," IEEE Transactions on Image Processing, vol. 6, pp. 298–311, February 1997.

[12] M. Nilkolova, J. Idier, and A. Mohammad-Djafari, "Inversion of large-support ill-posed linear operators using a piecewise gaussian mrf," IEEE Transactions on Image Processing, vol. 7, pp. 571–585, April 1998.

[13] A. Delaney and Y. Bresler, "Globally convergent edge-preserving regularized reconstruction: An application to limited-angle tomography," IEEE Transactions on Image Processing, vol. 7, pp. 204–221, February 1998.

[14] A. K. Katsaggelos, "Iterative image restoration algorithms," Optical Engineering, vol. 28, pp. 735–748, July 1989.

[15] A. K. Katsaggelos, J. Biemond, R. M. Mersereau, and R. W. Schafer, "An iterative method for restoring noisy blurred images," Circuits, Systems, and Signal Processing, vol. 3, no. 2, pp. 139–160, 1984.

[16] M. E. Zervakis, A. K. Katsaggelos, and T. M. Kwon, "Robust estimation techniques in regularized image restoration," Optical Engineering, vol. 4, pp. 752–773, June 1995.

[17] C. Bouman and K. Sauer, "A Generalized gaussian image model for edge-preserving map estimation," IEEE Transactions on Image Processing, vol. 2, July 1993.

[18] M. E. Zervakis and T. M. Kwon, "Robust estimation techniques in regularized image restoration," Optical Engineering, vol. 31, pp. 2174–2190, October 1992.

[19] R. Stevenson and E. Delp, "Fitting curves with discontinuities," in Proceeding of First International Workshop on Robust Computer Vision, (Seattle, WA), 1990.

[20] P. J. Green, "Bayesian reconstructions from emission tomography data using a modified EM algorithm," IEEE Transactions on Medical Imaging, vol. 9, pp. 84–93, March 1990.

[21] R. R. Schultz and R. L. Stevenson, "A Bayesian approach to image expansion for improved definition," IEEE Transactions on Image Processing, vol. 3, pp. 233–242, May 1994.

[22] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 6, pp. 721–741, November 1984.

[23] J. Hutchinson, C. Koch, J. Luo, and C. Mead, "Computing motion using analog and binary resistive networks," Computer, vol. 21, pp. 53–63, March 1988.

[24] A. Blake and A. Zisserman, Visual Reconstruction. Cambridge, MA: MIT Press, 1987.

[25] C. Bouman and B. Liu, "A multiple resolution approach to regularization," in Proceeding of SPIE Conference on Visual Communications and Image Processing, (Cambridge, MA), pp. 512–520, 1988.

[26] K. Lange, "Convergence of EM image reconstruction algorithms with gibbs priors," IEEE Transactions on Medical Imaging, vol. 9, pp. 439–446, December 1990.

[27] C. Bouman and K. Sauer, "An edge preserving method for image reconstruction from integral projections," in Proceedings of Conference on Information Science and System, (Johns Hopkins University, Baltimore, MD), pp. 382–387, 1991.

[28] P. J. Huber, Robust Statistics. New York: Wiley, 1981.

[29] R. L. Stevenson, B. E. Schmitz, and E. J. Delp, "Discontinuity preserving regularization of inverse visual problems," IEEE Transactions on Systems, Man and Cybernetics, vol. 24, March 1994.

[30] P. Stoica and Y. Selén, "Cyclic minimizers, majorization techniques, and the expectation-maximization algorithm: a refresher," IEEE Signal Processing Magazine, pp. 112–114, January 2004.

[31] R. Raich and A. O. H. III, "Sparse image reconstruction for partially known blur functions," in Proceedings of the International Conference on Image Processing, (Atlanta, GA), October 2006.

[32] R. Pan and S. J. Reeves, "Huber-Markov edge-preserving image restoration," in Proceedings of IS&T/SPIE Symposium on Electronic Imaging, (San Jose, CA), January 2005.

[33] D. Geman and C. Yang, "Nonlinear image recovery with half-quadratic regularization," IEEE Transactions on Image Processing, vol. 4, July 1995.

[34] J. Idier, "Convex half-quadratic criteria and interacting auxiliary variables for image restoration," IEEE Transactions on Image Processing, vol. 10, pp. 1001–1009, July 2001.

[35] R. Pan and S. J. Reeves, "Efficient Huber-Markov edge-preserving image restoration," IEEE Transactions on Image Processing, vol. 15, pp. 3728–3735, December 2006.

[36] W. Stewart, Introduction to the Numerical Solution of Markov Chains. Princeton University Press, 1994.

[37] M. K. Ng, R. H. Chan, and W. Tang, "A fast algorithm for deblurring models with Neumann boundary conditions," SIAM J. Sci. Comput., vol. 21, pp. 851–866, 1999.

[38] J. Chun and T. Kailath, Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms, ch. Generalized displacement structure for block-Toeplitz, Toeplitz-block, and Toeplitz-derived matrices, pp. 215–236. Springer-Verlag, 1991.

[39] N. Levinson, "The Wiener RMS error criterion in filter design and prediction," J. of Math. Phys., vol. 25, pp. 261–278, 1947.

[40] G. S. Ammar and W. B. Gragg, "Superfast solution of real positive definite toeplitz systems," SIAM Journal of Matrix Analysis and Applications, vol. 9, pp. 61–76, January 1988.

[41] T. Huckle, "Implementation of a superfast algorithm for symmetric positive definite linear equations of displacement rank 2," in Proceedings of SPIE, Advanced Signal Processing: Algorithms, Architectures, and Implementations V, October 1994.

[42] D. A. Bini and B. Meini, "Toeplitz systems," SIAM Journal of Matrix Analysis and Applications, vol. 20, no. 3, pp. 700–719, 1999.

[43] N. Kalouptsidis, G. Carayannis, and D. Manolakis, "Fast algorithms for block Toeplitz matrices with Toeplitz entries," Signal Processing, vol. 6, pp. 77–81, 1984.

[44] M. Wax and T. Kailath, "Efficient inversion of Toeplitz-block Toeplitz matrix," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 31, no. 5, pp. 1218–1221, 1983.

[45] G. H. Golub and C. V. Loan, Matrix Computations. Baltimore, MD: Johns Hopkins University Press, 1989.

[46] A. E. Yagle, "A fast algorithm for Toeplitz-block-Toeplitz linear systems," in Proceedings of the 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 3, pp. 1929–1932, 2001.

[47] J. Chun and T. Kailath, Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms. Springer-Verlag, 1991.

[48] S. J. Reeves, "Fast algorithm for solving block banded Toeplitz systems with banded Toeplitz blocks," in Proceedings of the 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 3, pp. 3325–3328, May 2002.

[49] P. Davis, Circulant Matrices. John Wiley & Sons, 1979.

[50] T. Kailath, S.-Y. Kung, and M. Morf, "Displacement ranks of matrices and linear equations," Journal of Mathematical Analysis and Applications, vol. 68, pp. 395–407, 1979.

[51] R. A. Horn and C. R. Johnson, Matrix Analysis. Cambridge: Cambridge University Press, 1985.

[52] E. Bareiss, "Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices," Numer. Math., vol. 13, pp. 404–424, 1969.

[53] T. Kailath, "Signal processing applications of some moment problems," in Proceedings of Symposia in Applied Mathematics, vol. 37, pp. 71–109, 1987.

[54] D. E. Dudgeon and R. M. Mersereau, Multidimensional Digital Signal Processing. New Jersey: Prentice-Hall, 1984.

[55] S. Haykin, Adaptive Filter Theory. New Jersey: Prentice Hall, 1996.

[56] ISO/IEC JTCl 10918-1, Information technologydigital compression and coding of continous-tone still image: Requirements and guidelines, ITU- T Rec. T.81, 1994.

[57] Official Joint Photographic Experts Group site, http://www.jpeg.org.

[58] Independent JPEG Group site, http://www.ijg.org.

[59] D. S. Taubman and M. W. Marcellin, JPEG2000 Image Compression Fundamentals, Standards and Practice. Kluwer Academic Publishers, 2002.

[60] D. A. Huffman, "A method for the reconstruction of minimum-redundancy codes," in Proceedings of I.R.E., pp. 1098–1101, September 1952.

[61] G. K. Wallace, "The JPEG still picture compression standard," Commun. ACM, vol. 34, no. 4, pp. 30–44, 1991.

[62] A. Watson, "Image compression using the discrete cosine transform," Mathematica Journal, vol. 4, pp. 81–88, January 1994.

[63] K. R. Rao and P. Yip, Discrete Cosine Transform: Algorithms, Advantages, Applications. Academic Press, 1990.

[64] S. P. Lloyd, "Least squares quantization in pcm," IEEE Transactions on Information Theory, vol. 28, pp. 129–137, 1982.

[65] J. Max, "Quantizing for minimum distortion," IRE Transactions on Information Theory, vol. 6, pp. 7–12, March 1960.

[66] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," IEEE Transactions on Communications, vol. 28, pp. 84–95, 1980.

[67] M. W. Marcellin and T. R. Fischer, "Trellis coded quantization of memoryless and gauss-markov sources," IEEE Transactions on Communications, vol. 38, pp. 82–93, 1990.

[68] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," IEEE Transactions on Computers, vol. 23, pp. 88–93, 1974.

[69] M. Unser, "On the approximation of the discrete Karhunen-Loeve transform for stationary processes," Signal Proc., vol. 5, no. 3, pp. 229–240, 1983.

[70] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos, "Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images," IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, vol. 3, pp. 421–432, December 1993.

[71] S. Minami and A. Zakhor, "An optimization approach for removing blocking effects in transform coding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 5, January 1995.

[72] H. C. Reeve and J. S. Lim, "Reduction of blocking effects in image coding," Opt. Eng., vol. 23, pp. 34–37, February 1984.

[73] C. J. Kuo and R. J. Hsieh, "Adaptive postprocessing for block encoded images," IEEE Trans. Circuits Syst. Video Technol., vol. 5, pp. 298–304, April 1995.

[74] B. Ramamurthi and A. Gersho, "Nonlinear space-variant postprocessing of block coded images," IEEE Trans. Acoust. Speech, Signal Process., vol. 34, pp. 1258–1268, October 1986.

[75] T. Meier, K. N. Ngan, and G. Cheng, "Reduction of blocking artifacts in image and video coding," IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 490–500, April 1999.

[76] R. L. Stevenson, "Reduction of coding artifacts in transform image coding," in Proceedings of the 1993 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. V, (Minneapolis, MN), pp. 401–404, March 1993.

[77] J. Yang, H. Choi, and T. Kim, "Noise estimation for blocking artifacts reduction in DCT coded images," IEEE Trans. Circuits Syst. Video Technol., vol. 10, pp. 1116–1120, October 2000.

[78] Z. Xiong, M. T. Orchard, and Y.-Q. Zhang, "A deblocking algorithm for JPEG compressed images using overcomplete wavelet representations," IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, vol. 7, pp. 433–437, April 1993.

[79] H. Choi and T. Kim, "Blocking artifact reduction in block-coded images using wavelet-based subband decomposition," IEEE Trans. Circuits Syst. Video Technol., vol. 10, pp. 801–805, August 2000.

[80] A. Nosratinia, "Denoising of JPEG images by re-application of JPEG," J. VLSI Signal Process., vol. 27, pp. 69–79, 2001.

[81] D. C. Youla and H. Webb, "Image restoration by the method of convex projections: Part 1 — theory," IEEE Transactions on Medical Imaging, vol. MI-1, pp. 81–94, October 1982.

[82] R. Rosenholtz and A. Zakhor, "Iterative procedures for reduction of blocking effects in transform image coding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 2, pp. 91–95, March 1992.

[83] S. J. Reeves and S. L. Eddins, "Comments on 'Iterative procedures for reduction of blocking effects in transform image coding'," IEEE Transactions on Circuits and Systems for Video Technology, vol. 3, pp. 439–440, December 1993.

[84] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos, "Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images," IEEE Trans. Circuits Syst. Video Technol., vol. 3, pp. 421–432, December 1993.

[85] S. H. Park and D. S. Kim, "Theory of projection onto narrow quantization constraint set and its application," IEEE Transactions on Image Processing, vol. 8, pp. 1361–1373, October 1999.

[86] K. Lee, D. S. Kim, and T. Kim, "Regression-based prediction for blocking artifact reduction in JPEG-compressed images," IEEE Transactions on Image Processing, vol. 14, January 2005.

[87] A. W.-C. Liew and H. Yan, "Blocking artifacts reduction in JPEG compressed images using overcomplete wavelet representation," in Proceedings of 2001 International Symposium on Intelligent Multimedia, video and Speech Processing, pp. 129–132, May 2001.

[88] N. S. Jayant and P. Noll, Digital Coding of Waveforms. Englewood Cliffs, New Jersey: Prentice-Hall, 1984.

[89] P. P. Vaidyanathan, Multirate Systems and Filter Banks. Englewood Cliffs, New Jersey: Prentice-Hall, 1992.